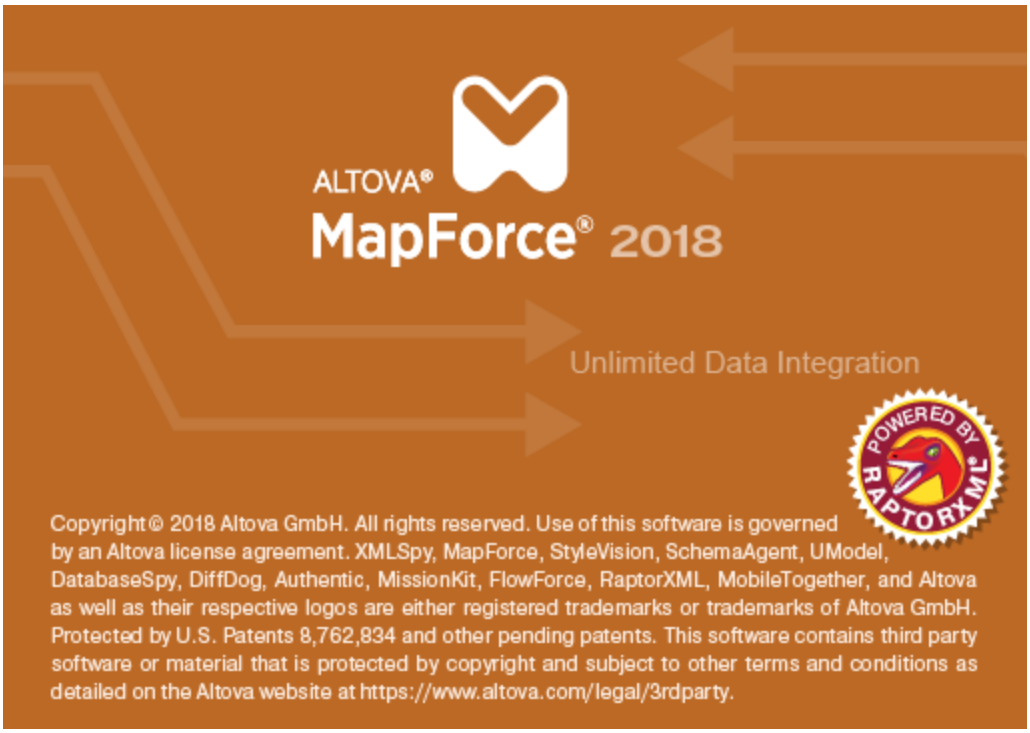
Manual del usuario y referencia



Copyright© 2018 Altova GmbH. All rights reserved. Use of this software is governed by an Altova license agreement. XMLSpy, MapForce, StyleVision, SchemaAgent, UModel, DatabaseSpy, DiffDog, Authentic, MissionKit, FlowForce, RaptorXML, MobileTogether, and Altova as well as their respective logos are either registered trademarks or trademarks of Altova GmbH. Protected by U.S. Patents 8,762,834 and other pending patents. This software contains third party software or material that is protected by copyright and subject to other terms and conditions as detailed on the Altova website at https://www.altova.com/legal/3rdparty.

Manual del usuario y referencia de Altova MapForce 2018 Enterprise Edition

Todos los derechos reservados. Ningún fragmento de esta publicación podrá ser reproducido de manera alguna (ya sea de forma gráfica, electrónica o mecánica, fotocopiado, grabado o reproducido en sistemas de almacenamiento y recuperación de información) sin el consentimiento expreso por escrito de su autor/editor.

Los productos a los que se hace referencia en este documento pueden ser marcas registradas de sus respectivos propietarios. El autor y editor no afirman ser propietarios de dichas marcas registradas.

Durante la elaboración de este documento se tomaron todas las precauciones necesarias para prevenir errores. Sin embargo, el autor y editor no se responsabilizan de los errores u omisiones que pudiese contener el documento ni de los posibles daños o perjuicios derivados del uso del contenido de este documento o de los programas y código fuente que vengan con el documento. Bajo ninguna circunstancia se podrá considerar al autor y editor responsables de la pérdida de beneficios ni de cualquier otro daño y perjuicio derivado directa o indirectamente del uso de este documento.

Fecha de publicación: 2018

© 2018 Altova GmbH

Contenido

1 Altova MapForce 2018 Enterprise Edition 3

1.1 Novedades 4

2 Introducción 12

2.1 Notas sobre compatibilidad 13
2.2 ¿Qué es MapForce? 14
2.3 Conceptos básicos 21
2.4 Interfaz del usuario 23
2.5 Notas generales 32

3 Tutoriales 34

3.1 Pasar datos XML a un esquema nuevo 35
3.2 Asignar varios orígenes de datos a un destino 46
3.3 Trabajar con varios esquemas de destino 53
3.4 Procesar y generar archivos de forma dinámica 62

4 Tareas generales 74

4.1 Trabajar con asignaciones de datos 75
4.1.1 Agregar componentes a la asignación 75
4.1.2 Agregar componentes desde una URL 76
4.1.3 Notas sobre transmisión por secuencias de datos 79
4.1.4 Seleccionar el lenguaje de transformación 80
4.1.5 Validar la asignación de datos 81
4.1.6 Validar resultados de la asignación 83
4.1.7 Vista previa de resultados 84
4.1.8 Características de la vista Texto 85
4.1.9 Búsquedas en la vista Texto 90
4.1.10 Vista previa del código XSLT 94
4.1.11 Generar código XSLT 95
4.1.12 Vista previa del código XQuery 95
4.1.13 Trabajar con varias ventanas de asignación 95

4.1.14 Cambiar configuración de la asignación ... 97
4.2 Trabajar con componentes ... 101
4.2.1 Búsquedas dentro de los componentes ... 102
4.2.2 Alinear componentes ... 103
4.2.3 Cambiar configuración de los componentes ... 104
4.2.4 Duplicar entradas ... 105
4.3 Trabajar con conexiones ... 107
4.3.1 Nota sobre entradas obligatorias ... 109
4.3.2 Opciones de presentación de las conexiones ... 111
4.3.3 Crear anotaciones en las conexiones ... 111
4.3.4 Configuración de las conexiones ... 112
4.3.5 Menú contextual de las conexiones ... 114
4.3.6 Conectar los secundarios equivalentes ... 115
4.3.7 Notificaciones sobre conexiones antecesoras ausentes ... 118
4.3.8 Mover conexiones y conexiones secundarias ... 119
4.3.9 Conservar conexiones tras eliminación de componentes ... 122
4.3.10 Nota sobre elementos ausentes ... 124
4.4 Trabajar con proyectos de asignación de datos ... 129
4.4.1 Abrir, cerrar proyectos y realizar búsquedas ... 129
4.4.2 Crear proyectos nuevos ... 130
4.4.3 Configurar la generación de código ... 131
4.4.4 Gestionar las carpetas de proyecto ... 133

5 Diseño de asignaciones 136

5.1 Usar rutas de acceso absolutas y relativas ... 138
5.1.1 Usar rutas de acceso relativas en un componente ... 138
5.1.2 Configurar la ruta de acceso de BD basadas en archivos ... 140
5.1.3 Corregir referencias a rutas de acceso rotas ... 142
5.1.4 Rutas de acceso según el entorno de ejecución ... 143
5.1.5 Operaciones cortar y pegar con rutas de acceso relativas ... 144
5.2 Tipos de conexión ... 145
5.2.1 Conexiones basadas en el destino ... 145
5.2.2 Conexiones basadas en el origen ... 145
Crear asignaciones entre contenido mixto ... 146
Ejemplo de contenido mixto ... 151
Usar conexiones estándar en nodos de contenido mixto ... 152
5.2.3 Conexiones de copia total ... 154
5.3 Asignaciones en cadena ... 157
5.3.1 Ejemplo: paso a través activo ... 159

5.3.2 Ejemplo: paso a través inactivo ... 163
5.4 Procesar varios archivos de entrada o salida simultáneamente ... 167
5.4.1 Asignar varios archivos de entrada a un solo archivo de salida ... 170
5.4.2 Asignar varios archivos de entrada a varios archivos de salida ... 172
5.4.3 Especificar nombres de archivo como parámetros de asignación ... 172
5.4.4 Vista previa de varios archivos de salida ... 173
5.4.5 Ejemplo: dividir un archivo XML en varios archivos ... 174
5.4.6 Ejemplo: dividir una tabla de BD en varios archivos XML ... 175
5.4.7 Varios archivos XML a partir de filas Excel ... 177
5.5 Pasar parámetros a la asignación ... 183
5.5.1 Agregar componentes de entrada simples ... 184
5.5.2 Configurar componentes de entrada simples ... 184
5.5.3 Crear un valor de entrada predeterminado ... 186
5.5.4 Ejemplo: usar nombres de archivo como parámetros de asignación ... 187
5.6 Obtener valores de cadena de una asignación ... 191
5.6.1 Agregar componentes de salida simples ... 193
5.6.2 Ejemplo: vista previa de resultados de una función ... 193
5.7 Usar variables ... 195
5.7.1 Agregar variables ... 197
5.7.2 Cambiar el contexto y ámbito de las variables ... 200
5.7.3 Ejemplo: contar filas de tabla de BD ... 202
5.7.4 Ejemplo: filtrar y numerar nodos ... 203
5.7.5 Ejemplo: crear grupos y subgrupos de registros ... 205
5.8 Ordenar datos ... 207
5.8.1 Ordenar según varias claves ... 209
5.8.2 Ordenar con variables ... 211
5.9 Filtros y condiciones ... 213
5.9.1 Ejemplo: filtrar nodos ... 215
5.9.2 Ejemplo: devolver un valor de forma condicional ... 217
5.10 Combinar datos ... 219
5.10.1 Agregar condiciones de combinación ... 221
5.10.2 Combinar tres o más estructuras ... 224
5.10.3 Ejemplo: combinar estructuras XML ... 225
5.11 Usar asignaciones de valores ... 230
5.11.1 Pasar datos sin modificarlos ... 234
5.11.2 Propiedades de las asignaciones de valores ... 237
5.12 Añadir excepciones ... 239
5.12.1 Ejemplo: excepción en la condición Greater than ... 240
5.12.2 Ejemplo: excepción cuando un nodo no existe ... 241
5.13 Redistribuir y serializar cadenas de texto ... 243

5.13.1 Información sobre el componente de análisis/serialización ... 243
5.13.2 Ejemplo: redistribuir cadena (de texto de longitud fija a Excel) ... 246
5.13.3 Ejemplo: serializar datos a una cadena (de XML a BD) ... 251
5.14 Asignar nombres de nodos ... 256
5.14.1 Obtener acceso al nombre de los nodos ... 257
5.14.2 Obtener acceso a determinado tipo de nodos ... 265
5.14.3 Ejemplo: asignar nombres de elemento a valores de atributo ... 269
5.14.4 Ejemplo: agrupar y filtrar nodos según su nombre ... 273
5.15 Reglas y estrategias de asignación de datos ... 278
5.15.1 Cambiar el orden de procesamiento de los componentes ... 282
5.15.2 Nodo de contexto prioritario ... 285
5.15.3 Reemplazar el contexto de la asignación ... 287

6 Depuración de asignaciones 294

6.1 Preparar la depuración ... 298
6.2 Comandos del depurador ... 299
6.3 Información sobre el modo depurador ... 301
6.4 Agregar y quitar puntos de interrupción ... 304
6.5 Ventana Valores ... 307
6.6 Ventana Contexto ... 309
6.7 Ventana Puntos de interrupción ... 311
6.8 Vista previa parcial de resultados generados ... 313
6.9 Ver el valor actual de un conector ... 314
6.10 Retroceder en el pasado reciente ... 315
6.11 Ver historial de valores procesados por un conector ... 316
6.12 Establecer el contexto en un valor ... 318
6.13 Configurar el depurador ... 319

7 Orígenes y destinos de datos 322

7.1 XML y esquemas XML ... 323
7.1.1 Generar un esquema XML ... 323
7.1.2 Configuración de componentes XML ... 324
7.1.3 Usar documentos DTD como componentes de esquema ... 329
7.1.4 Tipos XML Schema derivados ... 329
7.1.5 Nombres QName ... 332
7.1.6 Valores Nil y Nillable ... 332
7.1.7 Comentarios e instrucciones de procesamiento ... 336
7.1.8 Secciones CDATA ... 337

7.1.9 Comodines: xs:any / xs:anyAttribute ... 339
7.1.10 Combinar datos de varios esquemas distintos ... 343
7.1.11 Declarar espacios de nombres personalizados ... 345
7.1.12 Firmas digitales ... 348
Configurar la firma digital XML ... 352
7.2 Bases de datos en MapForce ... 356
7.2.1 Conectarse a una base de datos ... 357
Iniciar el asistente para la conexión de base de datos ... 359
Resumen de controladores de base de datos ... 361
Configurar una conexión ADO ... 364
Conectarse a una base de datos Microsoft Access ... 366
Configurar las propiedades de vínculo de datos de SQL Server ... 366
Configurar las propiedades de vínculo de datos de Microsoft Access ... 367
Configurar una conexión ADO.NET ... 369
Crear una cadena de conexión en Visual Studio ... 371
Ejemplo: cadenas de conexión ADO.NET ... 374
Notas sobre compatibilidad con ADO.NET ... 375
Configurar una conexión ODBC ... 376
Ver los controladores ODBC disponibles ... 378
Configurar una conexión JDBC ... 379
Configurar la variable CLASSPATH ... 381
Configurar una conexión PostgreSQL ... 383
Configurar una conexión SQLite ... 384
Conectarse a una base de datos SQLite ... 385
Usar una conexión a través de recursos globales ... 385
Ejemplos de conexión a bases de datos ... 386
Conectarse a Firebird (ODBC) ... 386
Conectarse a Firebird (JDBC) ... 389
Conectarse a IBM DB2 (ODBC) ... 390
Conectarse a IBM DB2 for i (ODBC) ... 395
Conectarse a IBM Informix (JDBC) ... 398
Conectarse a MariaDB (ODBC) ... 399
Conectarse a Microsoft Access (ADO) ... 401
Conectarse a Microsoft SQL Server (ADO) ... 404
Conectarse a Microsoft SQL Server (ODBC) ... 406
Conectarse a MySQL (ODBC) ... 409
Conectarse a Oracle (ODBC) ... 410
Conectarse a Oracle (JDBC) ... 415
Conectarse a PostgreSQL (ODBC) ... 417
Conectarse a Progress OpenEdge (ODBC) ... 419
Conectarse a Progress OpenEdge (JDBC) ... 421
Conectarse a Sybase (JDBC) ... 422
Conectarse a Teradata (ODBC) ... 423
Conectarse a Teradata (JDBC) ... 428
Conexiones de BD en Linux y macOS ... 430
Conexiones SQLite en Linux y macOS ... 432
Conexiones JDBC en Linux y macOS ... 432

Conexiones Oracle en OS X Yosemite ... 432
7.2.2 Introducción a las asignaciones de datos de BD ... 433
Agregar bases de datos a la asignación ... 433
Ejemplo: agregar la base de datos "altova.mdb" a la asignación ... 435
Agregar, editar y quitar objetos de BD ... 438
Gestionar relaciones de BD ... 441
Definir relaciones locales ... 451
Ejecutar asignaciones que modifican BD ... 457
Reemplazar caracteres especiales ... 458
Gestionar valores null ... 459
Generar valores secuenciales y únicos ... 460
Sugerencias de finalización automática SQL ... 460
Configuración de componentes de BD ... 461
Asignar datos a bases de datos ... 465
Insertar datos en una tabla ... 466
Insertar datos en tablas vinculadas ... 470
Actualizar una tabla ... 475
Acción "Actualizar si... Insertar restantes" ... 478
Instrucciones de combinación MERGE ... 485
Opciones para tablas secundarias cuando se actualiza la tabla principal ... 486
Acción "Eliminar si..." ... 494
Acción "Omitir si..." ... 498
Revertir transacciones ... 503
Inserciones masivas ... 504
Gestionar valores null en acciones de tabla de BD ... 507
Configurar acciones de tabla de BD ... 511
Ejemplo: asignar datos XML a datos SQLite ... 514
7.2.3 Combinar datos de BD ... 518
Combinaciones en modo SQL ... 520
Ejemplo: combinar tablas en modo SQL ... 522
Ejemplo: crear un informe CSV a partir de varias tablas ... 532
7.2.4 Filtrar y ordenar datos de BD (SQL WHERE/ORDER) ... 537
Crear cláusulas WHERE y ORDER BY ... 540
7.2.5 Instrucciones SQL SELECT como tablas virtuales ... 544
Crear instrucciones SELECT ... 544
Ejemplo: SELECT con parámetros ... 547
7.2.6 Asignaciones entre datos XML y campos de BD ... 553
Asignar un esquema XML a un campo de BD ... 553
Ejemplo: escribir datos XML a un campo SQLite ... 556
Ejemplo: extraer datos de columnas tipo XML de IBM DB2 ... 560
7.2.7 Examinar y consultar bases de datos ... 564
Seleccionar o conectarse a una base de datos ... 565

Crear y editar instrucciones SQL ... 566
Generar instrucciones SQL ... 567
Ejecutar instrucciones SQL ... 568
Importar y exportar scripts SQL ... 568
Agregar y quitar comentarios SQL ... 568
Uso de marcadores ... 570
Insertar regiones ... 570
Examinar objetos de la base de datos ... 571
Filtrar y buscar objetos de la base de datos ... 573
Opciones contextuales del explorador ... 574
Copiar, ordenar y buscar resultados de la consulta ... 576
Ver el estado de las consultas ejecutadas ... 578
Configurar la consulta de la base de datos ... 579
Codificación del archivo SQL ... 579
Configuración general del editor SQL ... 580
Generación de instrucciones SQL ... 581
Vista de resultados de la consulta ... 582
Fuentes del editor SQL ... 582
7.2.8 Procedimientos almacenados ... 583
Insertar procedimientos almacenados en componentes de BD ... 586
Casos de uso ... 588
Procedimientos almacenados y relaciones locales ... 588
Procedimientos almacenados como origen de datos ... 589
Procedimientos almacenados sin parámetros de entrada ... 590
Llamada con parámetros: entrada/salida ... 593
Componentes de origen y relaciones locales ... 596
Procedimientos almacenados en componentes de destino ... 600
Usar procedimientos almacenados para generar claves principales ... 602
7.3 Archivos CSV y archivos de texto ... 607
7.3.1 Ejemplo: asignar archivos CSV a XML ... 607
7.3.2 Ejemplo: recorrer elementos ... 610
7.3.3 Ejemplo: crear jerarquías a partir de archivos CSV y de texto de longitud fija 12
7.3.4 Opciones de configuración de componentes CSV ... 616
7.3.5 Ejemplo: asignar archivos de texto de longitud fija a bases de datos ... 621
7.3.6 Opciones de configuración de componentes FLF ... 627
7.4 MapForce FlexText ... 634
7.4.1 Información general ... 634
7.4.2 Tutorial ... 637
Paso nº1: crear la plantilla FlexText ... 637
Paso nº2: definir condiciones de división ... 639
Paso nº3: definir varias condiciones por contenedor ... 642
Paso nº4: crear el componente de destino de MapForce ... 646
Paso nº5: usar plantillas FlexText en MapForce ... 648

7.4.3 Configuración de componentes FlexText ... 649
7.4.4 Usar FlexText como componente de destino ... 651
7.4.5 Referencia del usuario ... 652
División repetida ... 652
Modo: longitud fija ... 654
Modo: delimitado (flotante) ... 654
Modo: delimitado (basado en línea) ... 656
Modo: delimitado (la línea empieza con) ... 658
Dividir una vez ... 661
Modo: longitud fija ... 662
Modo: delimitado (flotante) ... 663
Modo: delimitado (basado en línea) ... 664
Modo: delimitado (la línea empieza con) ... 665
Conmutador ... 667
Nodo ... 671
Omitir ... 673
Guardar como CSV (delimitado) ... 674
Guardar como FLF (longitud fija) ... 681
Guardar como valor ... 684
7.4.6 Expresiones regulares en FlexText ... 685
Dividir texto con expresiones regulares ... 685
Usar expresiones regulares en condiciones del conmutador ... 687
7.5 EDI ... 692
7.5.1 Terminología EDI ... 694
7.5.2 Agregar componentes EDI a la asignación ... 695
7.5.3 Ejecutar asignaciones EDI ... 697
7.5.4 Configuración de componentes EDI ... 698
7.5.5 Validación de componentes EDI ... 702
7.5.6 ASC X12 ... 706
Reglas de validación ASC X12 ... 706
Reglas de finalización automática de datos ASC X12 ... 707
Generar una confirmación funcional X12 997 ... 708
Generar una confirmación de implementación X12 999 ... 710
7.5.7 HIPAA X12 ... 711
Ejemplo: asignar datos HIPAA X12 a XML ... 712
Nota sobre asignación de transacciones HIPAA ... 714
7.5.8 HL7 Versión 2 ... 716
Ejemplo: asignar datos HL7 a XML ... 717
7.5.9 IATA PADIS ... 719
Ejemplo: asignar datos IATA a XML ... 720
7.5.10 NCPDP SCRIPT ... 722
Regla de validación NCPDP SCRIPT ... 724

Reglas de finalización automática NCPDP SCRIPT ... 724
7.5.11 SAP IDoc ... 725
7.5.12 TRADACOMS ... 728
Agregar archivos TRADACOMS como componentes ... 732
Componentes TRADACOMS en MapForce ... 734
Validación y finalización automática de datos ... 735
Conservar ceros iniciales durante la conversión ... 738
Archivos de configuración ... 739
Ejemplo: convertir una factura TRADACOMS en XML ... 740
7.5.13 UN/EDIFACT ... 744
Mensajes EDIFACT por lotes e interactivos ... 745
Reglas de validación UN/EDIFACT (por lotes) ... 746
Reglas de finalización automática de datos UN/EDIFACT (por lotes) ... 747
Reglas de validación UN/EDIFACT (interactivo) ... 748
Reglas de finalización automática UN/EDIFACT (interactivo) ... 748
Ejemplo: asignar datos UN/EDIFACT a XML ... 749
7.5.14 Personalizar estructuras EDI ... 757
Crear una colección EDI personalizada ... 758
Archivos de configuración EDI ... 759
Esquemas de configuración EDI ... 761
Actualizar archivos de configuración antiguos ... 762
Agregar o eliminar tipos de mensaje ... 764
Cambiar estructura del mensaje ... 766
Varios elementos de datos consecutivos ... 768
Dividir entradas combinadas en distintos nodos ... 771
Cambiar posibles valores de código de elementos de datos ... 773
Ejemplo: agregar un segmento TA1 ... 775
Ejemplo: personalizar un mensaje EDIFACT ... 776
Ejemplo: personalizar una transacción ANSI X12 ... 783
7.5.15 Personalizar la validación de datos EDI ... 790
Relajar la validación de campos ... 791
Adaptar valores de código de fuentes externas ... 793
Definir valores de código a nivel de mensaje (inline) ... 797
Configurar validación a nivel de caracteres ... 798
7.6 JSON ... 802
7.6.1 Agregar archivos JSON como componentes de asignación ... 804
7.6.2 Configuración de componentes JSON ... 804
7.6.3 Compatibilidad con JSON5 ... 807

7.6.4 Ejemplo: asignación de datos entre JSON y CSV ... 808
7.7 Microsoft OOXML Excel 2007+ ... 814
7.7.1 Agregar archivos Excel 2007+ como componentes de la asignación ... 816
7.7.2 Información sobre componentes Excel 2007+ ... 817
7.7.3 Agregar y eliminar hojas de cálculo ... 819
7.7.4 Agregar y eliminar rangos de filas ... 821
7.7.5 Seleccionar rangos de celdas ... 822
7.7.6 Insertar columnas entre columnas actuales ... 826
7.7.7 Configuración de componentes Excel 2007+ ... 828
7.7.8 Ejemplo: asignación de datos Excel 2007+ a XML ... 831
7.7.9 Ejemplo: asignación de datos de BD a Excel 2007+ ... 833
7.7.10 Ejemplo: suministrar datos a hojas Excel con formato previo ... 835
7.8 XBRL ... 839
7.8.1 Agregar archivos XBRL como componentes de la asignación ... 840
7.8.2 Leer datos de Inline XBRL ... 841
7.8.3 Paquetes de taxonomías XBRL ... 843
7.8.4 Información sobre componentes XBRL ... 846
7.8.5 Seleccionar vistas de estructuras ... 849
7.8.6 Configuración de componentes XBRL ... 850
7.8.7 Establecer preferencias para XBRL ... 853
Habilitar información rápida y anotaciones ... 853
Opciones XBRL generales ... 854
7.8.8 Trabajar con valores XBRL predeterminados ... 856
7.8.9 Trabajar con hipercubos XBRL ... 860
Ver las dimensiones de un componente ... 861
Cambiar el orden de las dimensiones ... 864
Generar asignaciones de valores para dimensiones explícitas de hipercubo ... 866
7.8.10 Trabajar con tablas XBRL ... 868
Mostrar y ocultar desgloses ... 869
Cambiar el orden de los desgloses ... 869
Trabajar con parámetros ... 871
7.8.11 Ejemplos de asignaciones de datos XBRL ... 872
Asignar datos de Microsoft Access a XBRL ... 872
Asignar datos de Microsoft Excel a XBRL ... 874
7.9 HL7 versión 3 ... 878

8 Funciones 882

8.1 Trabajar con funciones ... 883
8.2 Funciones definidas por el usuario ... 888

8.2.1 Parámetros de las funciones ... 894
8.2.2 Funciones definidas por el usuario estándar e inline ... 897
8.2.3 Crear una función de búsqueda simple ... 899
8.2.4 Ejemplo de función definida por el usuario ... 903
8.2.5 Función definida por el usuario compleja: nodo XML como entrada ... 908
Definir componentes de entrada complejos ... 909
8.2.6 Función definida por el usuario compleja: nodo XML como salida ... 915
Definir componentes de salida complejos ... 916
8.2.7 Asignación recursiva definida por el usuario ... 920
Crear una función recursiva definida por el usuario ... 922
8.3 Importar funciones XSLT 1.0/2.0 personales ... 930
8.3.1 Ejemplo: agregar funciones XSLT personales ... 931
8.3.2 Ejemplo: sumar valores de nodos ... 934
8.4 Importar funciones XQuery 1.0 personales ... 937
8.5 Importar bibliotecas Java y .NET personales ... 938
8.5.1 Ejemplo: importar clase Java personales ... 940
8.5.2 Ejemplo: importar ensamblado .NET DLL personal ... 941
8.6 Referencias manuales a bibliotecas Java, C# y C++ personales ... 943
8.6.1 Configurar el archivo .mff ... 943
8.6.2 Importar el archivo .mff a MapForce ... 949
8.6.3 Correspondencias entre tipos de datos ... 949
8.6.4 Ejemplo: crear una biblioteca C# personal ... 952
8.6.5 Ejemplo: crear una biblioteca C++ personal ... 953
8.6.6 Ejemplo: crear una biblioteca Java personal ... 956
8.7 Expresiones regulares ... 958
8.8 Referencia de la biblioteca de funciones ... 961
8.8.1 core | aggregate functions (agregado) ... 961
avg ... 962
count ... 963
max ... 963
max-string ... 964
min ... 964
min-string ... 965
string-join ... 965
sum ... 966
8.8.2 core | conversion functions (conversión) ... 966
boolean ... 967
format-date ... 967
format-dateTime ... 968
format-number ... 971

format-time ... 973
number ... 973
parse-date ... 974
parse-dateTime ... 974
parse-number ... 976
parse-time ... 977
string ... 977
8.8.3 core | file path functions (ruta de archivos) ... 977
get-fileext ... 978
get-folder ... 978
main-mfd-filepath ... 978
mfd-filepath ... 978
remove-fileext ... 979
remove-folder ... 979
replace-fileext ... 979
resolve-filepath ... 979
8.8.4 core | generator functions (generador) ... 980
auto-number ... 980
8.8.5 core | logical functions (lógica) ... 982
equal ... 983
equal-or-greater ... 983
equal-or-less ... 983
greater ... 983
less ... 984
logical-and ... 984
logical-not ... 984
logical-or ... 985
not-equal ... 985
8.8.6 core | math functions (matemáticas) ... 985
add ... 986
ceiling ... 986
divide ... 986
floor ... 987
modulus ... 987
multiply ... 987
round ... 988
round-precision ... 988
subtract ... 988
8.8.7 core | node functions (nodo) ... 988
is-xsi-nil ... 989
node-name ... 989

set-xsi-nil ... 990
static-node-annotation ... 990
static-node-name ... 990
substitute-missing-with-xsi-nil ... 991
8.8.8 core | QName functions (QName) ... 991
QName ... 991
local-name-from-QName ... 991
namespace-uri-from-QName ... 993
8.8.9 core | sequence functions (secuencia) ... 993
distinct-values ... 993
exists ... 994
first-items ... 995
generate-sequence ... 995
group-adjacent ... 995
group-by ... 997
group-ending-with ... 999
group-into-blocks ... 1000
group-starting-with ... 1000
item-at ... 1001
items-from-till ... 1001
last-items ... 1002
not-exists ... 1002
position ... 1003
replicate-item ... 1006
replicate-sequence ... 1008
set-empty ... 1008
skip-first-items ... 1008
substitute-missing ... 1008
8.8.10 core | string functions (cadena) ... 1009
char-from-code ... 1009
code-from-char ... 1009
concat ... 1009
contains ... 1010
normalize-space ... 1010
starts-with ... 1010
string-length ... 1010
substring ... 1010
substring-after ... 1011
substring-before ... 1011
tokenize ... 1011
tokenize-by-length ... 1013

tokenize-regexp ... 1015
translate ... 1016
8.8.11 db (base de datos) ... 1017
is-not-null ... 1017
is-null ... 1017
set-null ... 1017
substitute-null ... 1018
8.8.12 edifact ... 1019
auto-format ... 1019
to-date ... 1019
to-datetime ... 1019
to-duration ... 1020
to-time ... 1020
8.8.13 lang | datetime functions (fechaHora) ... 1021
age ... 1021
convert-to-utc ... 1021
date-from-datetime ... 1022
datetime-add ... 1022
datetime-diff ... 1023
datetime-from-date-and-time ... 1024
datetime-from-parts ... 1025
day-from-datetime ... 1026
day-from-duration ... 1027
duration-add ... 1027
duration-from-parts ... 1028
duration-subtract ... 1028
hour-from-datetime ... 1029
hour-from-duration ... 1030
leapyear ... 1030
millisecond-from-datetime ... 1030
millisecond-from-duration ... 1031
minute-from-datetime ... 1031
minute-from-duration ... 1031
month-from-datetime ... 1031
month-from-duration ... 1032
now ... 1032
remove-timezone ... 1032
second-from-datetime ... 1032
second-from-duration ... 1033
time-from-datetime ... 1033
timezone ... 1033

weekday ... 1033
weeknumber ... 1034
year-from-datetime ... 1034
year-from-duration ... 1034
8.8.14 lang | generator functions (generador) ... 1034
create-guid ... 1035
8.8.15 lang | logical functions (lógica) ... 1035
logical-xor ... 1035
negative ... 1035
numeric ... 1035
positive ... 1035
8.8.16 lang | math functions (matemáticas) ... 1036
abs ... 1036
acos ... 1036
asin ... 1036
atan ... 1036
cos ... 1037
degrees ... 1037
divide-integer ... 1037
exp ... 1037
log ... 1037
log10 ... 1038
max ... 1038
min ... 1038
pi ... 1038
pow ... 1038
radians ... 1039
random ... 1039
sin ... 1039
sqrt ... 1039
tan ... 1039
unary-minus ... 1039
8.8.17 lang | QName functions (QName) ... 1040
QName-as-string ... 1040
string-as-QName ... 1040
8.8.18 lang | string functions (cadena) ... 1040
capitalize ... 1040
count-substring ... 1040
empty ... 1041
find-substring ... 1041
format-guid-string ... 1041

left ... 1041
left-trim ... 1042
lowercase ... 1042
match-pattern ... 1042
pad-string-left ... 1042
pad-string-right ... 1043
repeat-string ... 1043
replace ... 1043
reversefind-substring ... 1043
right ... 1044
right-trim ... 1044
string-compare ... 1044
string-compare-ignore-case ... 1044
uppercase ... 1045
8.8.19 xbrl ... 1045
xbrl-measure-currency ... 1045
xbrl-measure-pure ... 1046
xbrl-measure-shares ... 1046
8.8.20 xlsx ... 1046
columnname-to-index ... 1046
date-to-xlsx ... 1046
datetime-to-xslx ... 1047
index-to-columnname ... 1047
time-to-xlsx ... 1047
xlsx-to-date ... 1047
xlsx-to-datetime ... 1047
xlsx-to-time ... 1048
8.8.21 xpath2 | accessors (descriptores de acceso) ... 1048
base-uri ... 1048
node-name ... 1048
string ... 1048
8.8.22 xpath2 | anyURI functions ... 1048
resolve-uri ... 1048
8.8.23 xpath2 | boolean functions (booleanas) ... 1049
false ... 1049
true ... 1049
8.8.24 xpath2 | constructors (constructores) ... 1049
8.8.25 xpath2 | context functions (contexto) ... 1050
current-date ... 1050
current-dateTime ... 1050
current-time ... 1050

default-collation ... 1051
implicit-timezone ... 1051
last ... 1051
8.8.26 xpath2 | durations, date, time functions (duración, fecha y hora) ... 1052
8.8.27 xpath2 | node functions (nodo) ... 1054
8.8.28 xpath2 | numeric functions (numéricas) ... 1055
8.8.29 xpath2 | string functions (cadena) ... 1055
8.8.30 xslt | xpath functions ... 1058
8.8.31 xslt | xslt functions ... 1060
currrent ... 1060
document ... 1060
element-available ... 1060
function-available ... 1061
generate-id ... 1061
system-property ... 1061
unparsed-enity-uri ... 1062

9 Implementar servicios web SOAP 1064

9.1 Compatibilidad con SOAP y WSDL ... 1067
9.2 Crear proyectos de servicio web a partir de archivos WSDL ... 1070
9.3 Definir errores de servicio web SOAP ... 1078
9.4 Información sobre servicios web SOAP Java ... 1080
9.5 Ejemplo: generar servicios web SOAP (C#) ... 1083
9.6 Ejemplo: generar servicios web SOAP (Java) ... 1097

10 Llamar a servicios web 1106

10.1 Agregar llamadas a servicios web (de tipo REST) ... 1108
10.2 Agregar llamadas a servicios web (de tipo WSDL) ... 1121
10.3 Configurar llamadas a servicios web ... 1122
10.4 Configurar seguridad HTTP ... 1126
10.5 Configurar seguridad WS-Security ... 1128
10.6 Ejemplo: llamar a un servicio web de tipo REST ... 1131
10.7 Ejemplo: asignar datos de un canal RSS ... 1135
10.8 Ejemplo: llamar a un servicio web SOAP ... 1140
10.9 Gestión de certificados digitales ... 1145
10.9.1 Confiar en certificados servidor en Linux ... 1147
10.9.2 Confiar en certificados servidor en OS X / macOS ... 1150
10.9.3 Confiar en certificados servidor en Windows ... 1151

10.9.4 Acceso a los almacenes de certificados en Windows 1155
10.9.5 Exportar certificados de Windows 1155
10.9.6 Certificados de cliente en Linux 1161
10.9.7 Certificados de cliente en OS X / macOS 1162
10.9.8 Certificados de cliente en Windows 1164

11 Automatizar asignaciones de datos 1168

11.1 Automatización con RaptorXML Server 1169
11.2 Automatización con MapForce Server 1170
11.3 Preparar asignaciones para ejecutarlas en servidores 1171
11.4 Compilar asignaciones en archivos de ejecución de MapForce Server 1176
11.5 Implementar asignaciones en FlowForce Server 1179
11.6 Integración con AS2 1184
11.7 Interfaz de la línea de comandos de MapForce 1190

12 Personalizar MapForce 1198

12.1 Cambiar opciones de MapForce 1199
12.2 Recursos globales de Altova 1201
12.2.1 Crear recursos globales 1202
12.2.2 Bases de datos como recursos globales 1203
12.2.3 Resultados de transformaciones de MapForce y StyleVision como recursos globales
12.2.4 Archivo XML de recursos globales 1206
12.2.5 Recursos globales en distintos entornos de ejecución 1207
12.2.6 Ejemplo: ejecutar asignación con archivos de entrada variables 1209
12.2.7 Ejemplo: generar resultados en carpetas variables 1210
12.2.8 Ejemplo: cambiar de base de datos 1212
12.2.9 Ejemplo: crear un flujo de trabajo de aplicación 1216
12.3 Aplicar estilos a resultados de la asignación con StyleVision 1221
12.3.1 Ejemplos de asignaciones con hojas de estilos StyleVision 1223
12.4 Generar y personalizar la documentación de la asignación 1226
12.4.1 Hojas de estilos predefinidas de StyleVision 1231
12.4.2 Diseño personalizado 1234
12.5 Personalizar teclas de acceso rápido 1236
12.6 Archivos de catálogo 1239

13 Complemento de MapForce para Visual Studio 1246

13.1 Habilitar el complemento ... 1247
13.2 Trabajar con asignaciones y proyectos ... 1249
13.3 Acceso a menús y funciones frecuentes ... 1251

14 Complemento de MapForce para Eclipse 1256

14.1 Instalación ... 1257
14.2 Perspectiva MapForce ... 1262
14.3 Acceso a menús y funciones frecuentes ... 1265
14.4 Trabajar con asignaciones y proyectos ... 1268
14.4.1 Crear un proyecto MapForce/Eclipse ... 1268
14.4.2 Crear asignaciones nuevas ... 1270
14.4.3 Importar asignaciones a un proyecto de Eclipse ... 1272
14.4.4 Configurar la generación automática de código MapForce ... 1275
14.5 Ampliar el complemento de MapForce para Eclipse ... 1278

15 Referencia de menús 1284

15.1 Archivo ... 1285
15.2 Edición ... 1288
15.3 Insertar ... 1289
15.4 Proyecto ... 1291
15.5 Componente ... 1293
15.6 Conexión ... 1295
15.7 Función ... 1296
15.8 Resultados ... 1297
15.9 Depuración ... 1298
15.10 Vista ... 1299
15.11 Herramientas ... 1301
15.12 Ventanas ... 1303
15.13 Ayuda ... 1304

16 Generador de código 1312

16.1 Introducción al generador de código ... 1313
16.2 Novedades ... 1315
16.3 Generar código C++ ... 1317
16.3.1 Generar código a partir de una asignación de datos ... 1318
16.3.2 Generar código a partir de un proyecto de asignación ... 1319

16.3.3 Compilar el proyecto ... 1320
16.3.4 Ejecutar la aplicación ... 1320
16.4 Generar código C# ... 1322
16.4.1 Generar código a partir de una asignación de datos ... 1323
16.4.2 Generar código a partir de un proyecto de asignación ... 1323
16.4.3 Compilar el proyecto ... 1324
16.4.4 Ejecutar la aplicación ... 1324
16.5 Generar código Java ... 1326
16.5.1 Generar código a partir de una asignación de datos ... 1327
16.5.2 Generar código a partir de un proyecto de asignación ... 1328
16.5.3 Control de referencias JDBC ... 1329
16.5.4 Compilar el proyecto con Ant ... 1329
16.5.5 Ejemplo: crear una aplicación Java con Eclipse y Ant ... 1330
Paso nº1: generar código Java ... 1331
Paso nº2: importar el proyecto en Eclipse ... 1331
Paso nº3: ejecutar el proyecto como aplicación de interfaz gráfica ... 1334
Paso nº4: ejecutar el proyecto como aplicación de consola ... 1336
Paso nº5: compilar el archivo JAR con Ant ... 1337
16.6 Integrar código generado con MapForce ... 1340
16.6.1 Ejemplo: Java ... 1341
16.6.2 Ejemplo: C# ... 1343
16.6.3 Ejemplo: C++ ... 1345
16.6.4 Cambiar el tipo de datos de la entrada/salida de la asignación (... 1346
16.7 Generar código fuente a partir de esquemas o DTD ... 1351
16.7.1 Bibliotecas contenedoras de esquemas (C++) ... 1354
16.7.2 Bibliotecas contenedoras de esquemas (C#) ... 1357
16.7.3 Bibliotecas contenedoras de esquemas (Java) ... 1359
16.7.4 Integrar las bibliotecas contenedoras de esquemas ... 1361
16.7.5 Ejemplo: usar las bibliotecas contenedoras de esquemas ... 1364
Ejemplo de esquema ... 1365
Leer y escribir documentos XML (C++) ... 1367
Leer y escribir documentos XML (C#) ... 1372
Leer y escribir documentos XML (Java) ... 1380
16.8 Referencia de las clases generadas (C++) ... 1391
16.8.1 altova::DateTime ... 1391
16.8.2 altova::Duration ... 1394
16.8.3 altova::DayTimeDuration ... 1396
16.8.4 altova::YearMonthDuration ... 1397
16.8.5 altova::meta::Attribute ... 1398

16.8.6 altova::meta::ComplexType ... 1399
16.8.7 altova::meta::Element ... 1400
16.8.8 altova::meta::SimpleType ... 1401
16.8.9 [SuEsquema]::[CDoc] ... 1402
16.8.10 [SuEsquema]::MemberAttribute ... 1405
16.8.11 [SuEsquema]::MemberElement ... 1405
16.9 Referencia de las clases generadas (C#) ... 1407
16.9.1 Altova.Types.DateTime ... 1407
16.9.2 Altova.Types.DateTimeFormat ... 1411
16.9.3 Altova.Types.Duration ... 1412
16.9.4 Altova.Xml.Meta.Attribute ... 1415
16.9.5 Altova.Xml.Meta.ComplexType ... 1415
16.9.6 Altova.Xml.Meta.Element ... 1416
16.9.7 Altova.Xml.Meta.SimpleType ... 1417
16.9.8 [SuEsquema].[Doc] ... 1418
16.9.9 [SuTipoEsquema].MemberAttribute ... 1420
16.9.10 [SuTipoEsquema].MemberElement ... 1421
16.10 Referencia de las clases generadas (Java) ... 1423
16.10.1 com.altova.types.DateTime ... 1423
16.10.2 com.altova.types.Duration ... 1428
16.10.3 com.altova.xml.meta.Attribute ... 1432
16.10.4 com.altova.xml.meta.ComplexType ... 1432
16.10.5 com.altova.xml.meta.Element ... 1433
16.10.6 com.altova.xml.meta.SimpleType ... 1433
16.10.7 com.[SuEsquema].[Doc] ... 1435
16.10.8 com.[SuEsquema].[SuTipoEsquema].MemberAttribute ... 1436
16.10.9 com.[SuEsquema].[SuTipoEsquema].MemberElement ... 1437
16.11 Consejos para la generación de código ... 1439
16.12 Opciones del generador de código ... 1440
16.13 SPL: el lenguaje de programación Spy ... 1442
16.13.1 Estructura básica de SPL ... 1442
16.13.2 Declaraciones ... 1443
16.13.3 Variables ... 1444
16.13.4 Variables predefinidas ... 1445
16.13.5 Crear archivos de salida ... 1447
16.13.6 Operadores ... 1448
16.13.7 Condiciones ... 1448
16.13.8 Colecciones y foreach ... 1449
16.13.9 Subrutinas ... 1451
Declaración de subrutinas ... 1451

Invocación de subrutinas ... 1452
Ejemplo ... 1453
16.13.10 Tipos integrados ... 1453
Library ... 1454
Namespace ... 1454
Type ... 1454
Member ... 1455
NativeBinding ... 1456
Facets ... 1456

17 The MapForce API 1460

17.1 Overview ... 1461
17.1.1 Accessing the API ... 1461
17.1.2 The Object Model ... 1463
17.1.3 Error Handling ... 1464
17.1.4 Examples ... 1465
Example C# Project ... 1466
Example Java Project ... 1470
JScript Examples ... 1473
Start Application ... 1473
Simple Document Access ... 1474
Generate Code ... 1476
Example: Code Generation ... 1477
Example: Mapping Execution ... 1479
Example: Project Support ... 1483
17.2 Object Reference ... 1488
17.2.1 Application ... 1488
Events ... 1489
OnDocumentOpened ... 1489
OnProjectOpened ... 1490
OnShutdown ... 1490
ActiveDocument ... 1490
ActiveProject ... 1490
Application ... 1490
Documents ... 1491
Edition ... 1491
GlobalResourceConfig ... 1491
GlobalResourceFile ... 1492
HighlightSerializedMarker ... 1492
IsAPISupported ... 1492
MajorVersion ... 1492
MinorVersion ... 1493

Name ... 1493
NewDocument ... 1493
NewProject ... 1493
OpenDocument ... 1494
OpenProject ... 1494
OpenURL ... 1494
Options ... 1495
Parent ... 1495
Quit ... 1495
ServicePackVersion ... 1495
Status ... 1496
Visible ... 1496
WindowHandle ... 1496
17.2.2 AppOutputLine ... 1497
Application ... 1497
ChildLines ... 1498
GetCellCountInLine ... 1498
GetCellIcon ... 1498
GetCellSymbol ... 1498
GetCellText ... 1499
GetCellTextDecoration ... 1499
GetIsCellText ... 1499
GetLineCount ... 1499
GetLineSeverity ... 1500
GetLineSymbol ... 1500
GetLineText ... 1500
GetLineTextEx ... 1500
GetLineTextWithChildren ... 1501
GetLineTextWithChildrenEx ... 1501
Parent ... 1501
17.2.3 AppOutputLines ... 1502
Application ... 1502
Count ... 1502
Item ... 1502
Parent ... 1503
17.2.4 AppOutputLineSymbol ... 1503
Application ... 1503
GetSymbolHREF ... 1503
GetSymbolID ... 1504
IsSymbolHREF ... 1504
Parent ... 1504

17.2.5 Component ... 1504
Application ... 1505
CanChangeInputInstanceFile ... 1505
CanChangeOutputInstanceFile ... 1506
ComponentName ... 1506
GenerateOutput ... 1506
GetRootDatapoint ... 1507
HasIncomingConnections ... 1508
HasOutgoingConnections ... 1508
ID ... 1508
InputInstanceFile ... 1508
IsParameterInputRequired ... 1509
IsParameterSequence ... 1509
Name ... 1509
OutputInstanceFile ... 1510
Parent ... 1510
Preview ... 1510
Schema ... 1511
SubType ... 1511
Type ... 1511
UsageKind ... 1511
17.2.6 Components ... 1512
Application ... 1512
Count ... 1512
Item ... 1512
Parent ... 1513
17.2.7 Connection ... 1513
Application ... 1513
ConnectionType ... 1513
Parent ... 1514
17.2.8 Datapoint ... 1514
Application ... 1514
GetChild ... 1515
Parent ... 1515
17.2.9 Document ... 1515
Events ... 1516
OnDocumentClosed ... 1516
OnModifiedFlagChanged ... 1517
Activate ... 1517
Application ... 1517
Close ... 1517

CreateUserDefinedFunction ... 1518
FindComponentByID ... 1518
FullName ... 1518
GenerateCHashCode ... 1518
GenerateCodeEx ... 1519
GenerateCppCode ... 1519
GenerateJavaCode ... 1519
GenerateOutput ... 1520
GenerateOutputEx ... 1520
GenerateXQuery ... 1520
GenerateXSLT ... 1521
GenerateXSLT2 ... 1521
HighlightSerializedMarker ... 1521
JavaSettings_BasePackageName ... 1522
MainMapping ... 1522
MapForceView ... 1522
Mappings ... 1522
Name ... 1523
OutputSettings_ApplicationName ... 1523
OutputSettings_Encoding (obsolete) ... 1523
Parent ... 1523
Path ... 1524
Save ... 1524
SaveAs ... 1524
Saved ... 1524
17.2.10 Documents ... 1525
ActiveDocument ... 1525
Application ... 1525
Count ... 1526
Item ... 1526
NewDocument ... 1526
OpenDocument ... 1526
Parent ... 1527
17.2.11 ErrorMarker ... 1527
Application ... 1527
DocumentFileName ... 1528
ErrorLevel ... 1528
Highlight ... 1528
Serialization ... 1528
Text ... 1529
Parent ... 1529

17.2.12 ErrorMarkers 1529
Application 1529
Count 1530
Item 1530
Parent 1530
17.2.13 MapForceView 1530
Active 1531
ActiveMapping 1531
ActiveMappingName 1531
Application 1532
HighlightMyConnections 1532
HighlightMyConnectionsRecursivey 1532
InsertWSDLCall 1533
InsertXMLFile (obsolete) 1533
InsertXMLSchema (obsolete) 1533
InsertXMLSchemaWithSample (obsolete) 1534
Parent 1534
ShowItemTypes 1535
ShowLibraryInFunctionHeader 1535
17.2.14 Mapping 1535
Application 1536
Components 1536
CreateConnection 1536
InsertFunctionCall 1536
InsertXMLFile 1537
InsertXMLSchema 1537
InsertXMLSchemaInputParameter 1538
InsertXMLSchemaOutputParameter 1538
IsMainMapping 1539
Name 1539
Parent 1539
17.2.15 Mappings 1540
Application 1540
Count 1540
Item 1540
Parent 1541
17.2.16 Options 1541
Application 1541
CodeDefaultOutputDirectory 1542
CPPSettings_DOMType 1542
CPPSettings_GenerateVC6ProjectFile 1542

CppSettings_GenerateVSProjectFile 1542
CPPSettings_LibraryType 1543
CPPSettings_UseMFC 1543
CSharpSettings_ProjectType 1543
DefaultOutputByteOrder 1544
DefaultOutputByteOrderMark 1544
DefaultOutputEncoding 1544
GenerateWrapperClasses 1544
JavaSettings_ApacheAxisVersion (obsolete) 1545
Parent 1545
ShowLogoOnPrint 1545
ShowLogoOnStartup 1545
UseGradientBackground 1546
XSLTDefaultOutputDirectory 1546
17.2.17 Project 1546
Events 1547
OnProjectClosed 1547
_NewEnum 1547
AddActiveFile 1548
AddFile 1549
Application 1549
Close 1549
Count 1549
CreateFolder 1550
FullName 1550
GenerateCode 1550
GenerateCodeEx 1551
GenerateCodeIn 1551
GenerateCodeInEx 1551
InsertWebService 1552
Item 1552
Java_BasePackageName 1552
Name 1553
Output_Folder 1553
Output_Language 1553
Output_TextEncoding 1554
Parent 1554
Path 1554
Save 1554
Saved 1555
17.2.18 ProjectItem 1555

_NewEnum ... 1556
AddActiveFile ... 1556
AddFile ... 1556
Application ... 1557
CodeGenSettings_Language ... 1557
CodeGenSettings_OutputFolder ... 1557
CodeGenSettings_UseDefault ... 1558
Count ... 1558
CreateFolder ... 1558
CreateMappingForProject ... 1558
GenerateCode ... 1559
GenerateCodeEx ... 1559
GenerateCodeIn ... 1560
GenerateCodeInEx ... 1560
Item ... 1560
Kind ... 1561
Name ... 1561
Open ... 1561
Parent ... 1562
QualifiedName ... 1562
Remove ... 1562
WSDLFile ... 1562
17.3 Enumerations ... 1564
17.3.1 ENUMApacheAxisVersion (obsolete) ... 1564
17.3.2 ENUMApplicationStatus ... 1564
17.3.3 ENUMAppOutputLine_Severity ... 1564
17.3.4 ENUMAppOutputLine_TextDecoration ... 1565
17.3.5 ENUMCodeGenErrorLevel ... 1565
17.3.6 ENUMComponentDatapointSide ... 1565
17.3.7 ENUMComponentSubType ... 1566
17.3.8 ENUMComponentType ... 1566
17.3.9 ENUMComponentUsageKind ... 1566
17.3.10 ENUMConnectionType ... 1566
17.3.11 ENUMDOMType ... 1567
17.3.12 ENUMLibType ... 1567
17.3.13 ENUMProgrammingLanguage ... 1567
17.3.14 ENUMProjectItemType ... 1568
17.3.15 ENUMProjectType ... 1568
17.3.16 ENUMSearchDatapointFlags ... 1568
17.3.17 ENUMViewMode ... 1569

18 ActiveX Integration 1572

18.1 Prerequisites ... 1573
18.2 Adding the ActiveX Controls to the Toolbox ... 1575
18.3 Integration at Application Level ... 1577
18.4 Integration at Document Level ... 1580
18.5 ActiveX Integration Examples ... 1584
18.5.1 C# ... 1584
Running the Sample C# Solution ... 1584
Retrieving Command Information ... 1587
Handling Events ... 1589
18.5.2 HTML ... 1591
HTML Integration at Application Level ... 1591
Instantiate the Control ... 1591
Add Button to Open Default Document ... 1591
Add Buttons for Code Generation ... 1592
Connect to Custom Events ... 1593
HTML Integration at Document Level ... 1593
Instantiate the MapForceControl ... 1594
Create Editor Window ... 1594
Create Project Window ... 1594
Create Placeholder for Helper Windows ... 1595
Create a Custom Toolbar ... 1595
Create More Buttons ... 1596
Create Event Handler to Update Button Status ... 1597
18.5.3 Java ... 1598
Example Java Project ... 1600
Creating the ActiveX Controls ... 1602
Loading Data in the Controls ... 1603
Basic Event Handling ... 1603
Menus ... 1604
UI Update Event Handling ... 1605
Listing the Properties of a MapForce Mapping ... 1606
18.5.4 VB.NET ... 1608
18.6 Command Reference ... 1611
18.6.1 "File" Menu ... 1611
18.6.2 "Edit" Menu ... 1612
18.6.3 "Insert" Menu ... 1613
18.6.4 "Project" Menu ... 1613
18.6.5 "Component" Menu ... 1614
18.6.6 "Connection" Menu ... 1616
18.6.7 "Function" Menu ... 1616

18.6.8 "Output" Menu ... 1616
18.6.9 "Debug" Menu ... 1617
18.6.10 "View" Menu ... 1618
18.6.11 "Tools" Menu ... 1618
18.6.12 "Window" Menu ... 1619
18.6.13 "Help" Menu ... 1619
18.7 Object Reference ... 1621
18.7.1 MapForceCommand ... 1621
Accelerator ... 1622
ID ... 1622
IsSeparator ... 1622
Label ... 1622
Name ... 1622
StatusText ... 1623
SubCommands ... 1623
ToolTip ... 1623
18.7.2 MapForceCommands ... 1623
Count ... 1623
Item ... 1624
18.7.3 MapForceControl ... 1624
Properties ... 1624
Appearance ... 1625
Application ... 1625
BorderStyle ... 1625
CommandsList ... 1625
EnableUserPrompts ... 1626
IntegrationLevel ... 1626
MainMenu ... 1627
Toolbars ... 1627
Methods ... 1628
Exec ... 1628
Open ... 1629
QueryStatus ... 1629
Events ... 1629
OnCloseEditingWindow ... 1630
OnContextChanged ... 1630
OnDocumentOpened ... 1630
OnFileChangedAlert ... 1630
OnLicenseProblem ... 1631
OnOpenedOrFocused ... 1631
OnToolWindowUpdated ... 1631
OnUpdateCmdUI ... 1631
OnValidationWindowUpdated ... 1632
18.7.4 MapForceControlDocument ... 1632
Properties ... 1633
Appearance ... 1633

BorderStyle 1633
Document 1633
IsModified 1634
Path 1634
ReadOnly 1634
Methods 1634
Exec 1635
New 1635
Open 1635
QueryStatus 1635
Reload 1636
Save 1636
SaveAs 1636
Events 1636
OnActivate 1637
OnContextChanged 1637
OnDocumentClosed 1637
OnDocumentOpened 1637
OnDocumentSaveAs 1638
OnFileChangedAlert 1638
OnModifiedFlagChanged 1638
OnSetEditorTitle 1638
18.7.5 MapForceControlPlaceHolder 1638
Properties 1639
Label 1639
PlaceholderWindowID 1639
Project 1640
Methods 1640
OpenProject 1640
CloseProject 1640
Events 1641
OnModifiedFlagChanged 1641
OnSetLabel 1641
18.7.6 Enumerations 1641
ICActiveXIntegrationLevel 1641
MapForceControlPlaceholderWindow 1642

19 Anexos 1644

19.1 Información sobre motores 1645
19.1.1 Información sobre motores XSLT y XQuery 1645
XSLT 1.0 1645
XSLT 2.0 1646
XQuery 1.0 1648
19.1.2 Funciones XSTL y XPath/XQuery 1651
Funciones de extensión de Altova 1652
Funciones XSLT 1654
Funciones XPath/XQuery: de fecha y hora 1657

Funciones XPath/XQuery: de geoubicación ... 1671
Funciones XPath/XQuery: relacionadas con imágenes ... 1680
Funciones XPath/XQuery: numéricas ... 1685
Funciones XPath/XQuery: de secuencia ... 1687
Funciones XPath/XQuery: de cadena ... 1695
Funciones XPath/XQuery varias ... 1701
Funciones de extensión para gráficos ... 1702
Estructura XML de los datos de gráficos ... 1706
Ejemplo: funciones para gráficos ... 1712
Funciones de extensión varias ... 1715
Funciones de extensión Java ... 1716
Archivos de clases definidos por el usuario ... 1717
Archivos JAR definidos por el usuario ... 1720
Constructores ... 1721
Métodos estáticos y campos estáticos ... 1722
Métodos de instancia y campos de instancia ... 1723
Tipos de datos: conversión de XPath/XQuery en Java ... 1724
Tipos de datos: conversión de Java en XPath/XQuery ... 1725
Funciones de extensión .NET ... 1725
Constructores ... 1728
Metodos estáticos y campos estáticos ... 1729
Métodos de instancia y campos de instancia ... 1730
Tipos de datos: conversión de XPath/XQuery en .NET ... 1731
Tipos de datos: conversión de .NET en XPath/XQuery ... 1732
Scripts MSXSL para XSLT ... 1732
19.2 Datos técnicos ... 1735
19.2.1 Requisitos de SO y memoria ... 1735
19.2.2 Validador XML de Altova ... 1735
19.2.3 Motores XSLT y XQuery de Altova ... 1736
19.2.4 Compatibilidad con Unicode ... 1736
19.2.5 Uso de Internet ... 1736
19.3 Información sobre licencias ... 1738
19.3.1 Distribución electrónica de software ... 1738
19.3.2 Activación del software y medición de licencias ... 1739
19.3.3 Derechos de propiedad intelectual ... 1740
19.3.4 Contrato de licencia para el usuario final ... 1740

20 Glosario 1742

20.1 A ... 1743
20.2 C ... 1744
20.3 F ... 1746
20.4 M ... 1747
20.5 R ... 1748

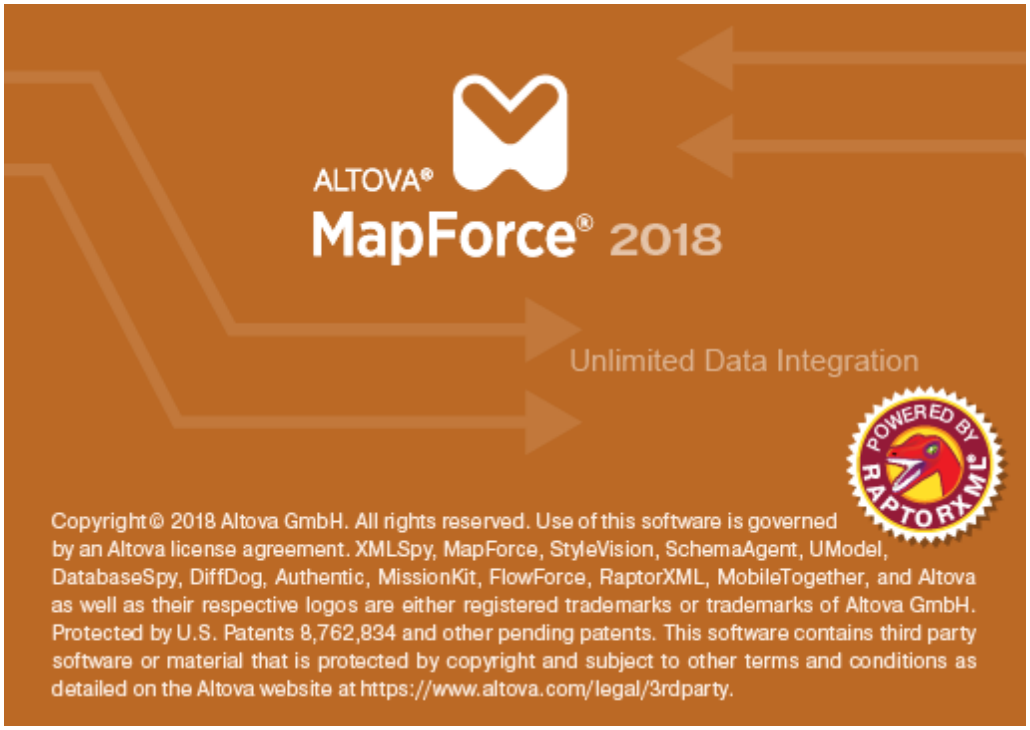
Índice

Altova MapForce 2018 Enterprise Edition

Altova MapForce 2018 Enterprise Edition

1 Altova MapForce 2018 Enterprise Edition

MapForce® 2018 Enterprise Edition es una herramienta de asignación de datos visual para crear proyectos avanzados de integración de datos. MapForce® es una aplicación para Windows de 32/64 bits compatible con Windows Server 2008 R2 SP1 con actualización de la plataforma o superior y Windows 7 SP1 con actualización de la plataforma, Windows 8, Windows 10. Las ediciones Enterprise y Professional están disponibles en 32 y 64 bits. MapForce también puede integrarse con Visual Studio y Eclipse, así como con los productos de Microsoft Office (*véase* *Notas sobre compatibilidad*).



Última actualización: 6/22/2018

1.1 Novedades

Novedades de **MapForce 2018**:

- Si tiene una licencia para trabajar con FlowForce Server Advanced Edition junto con MapForce y MapForce Server, podrá crear flujos de trabajo de datos que automaticen procesos AS2 (*véase Integración con AS2*).
- Compatibilidad con el estándar NCPDP (National Council for Prescription Drug Programs) SCRIPT (*véase NCPDP SCRIPT*).
- Opción para generar código de programa para Visual Studio 2013, 2015 y 2017 (*véase Generador de código*).
- Compatibilidad con estas versiones de bases de datos: Sybase ASE 16, PostgreSQL 9.6, MySQL 5.7
- En las asignaciones de datos Excel, desde la asignación se pueden añadir nuevas columnas Excel entre las columnas que ya existen sin necesidad de volver a crear las conexiones existentes (*véase Insertar columnas entre columnas actuales*).
- Actualizaciones internas y mejoras.

Novedades de **MapForce 2017 Release 3**:

- Nuevo tipo de componente (Combinación) para unir datos de dos o más estructuras distintas según las condiciones definidas por el usuario (*véase Combinar datos*). Cuando la asignación lee datos de una base de datos, se pueden combinar tablas o vistas de la base de datos en modo SQL JOIN (*véase Combinar datos de BD*).
- Ya puede asignar datos de archivos JSON que usan sintaxis JSON5. También puede generar archivos JSON5 con claves de objeto sin comillas (*véase Compatibilidad con JSON5*).
- Mejoras en las opciones de búsqueda de texto en los paneles *Resultados*, *XQuery* y *XSLT* (*véase Búsquedas en la vista Texto*). Además, ahora la aplicación resalta texto en todos estos paneles (*véase Resaltado de texto*).
- Las asignaciones que actualizan bases de datos se pueden configurar para que comparen datos teniendo en cuenta los valores NULL. Las comparaciones que tienen en cuenta los valores NULL (tras configurarse según el tipo de BD) permiten trabajar mejor con datos que contienen valores NULL (*véase Gestionar valores null en acciones de tabla de BD*).
- En el control ActiveX de MapForce la estructura del objeto `MapForceCommand` se ha mejorado para incluir una propiedad nueva llamada `Name`, que puede utilizarse para obtener el nombre único del comando. Esto simplifica la recuperación de información sobre comandos de MapForce mediante programación (*véase Recuperar información sobre comandos*).
- Actualizaciones internas y mejoras.

Novedades de **MapForce 2017**:

- Ya se pueden leer nombres de nodo desde un XML de origen (o nombres de campo desde archivos CSV/FLF) y asignar esta información al componente de destino. También se puede crear de forma dinámica nuevos atributos o elementos XML en el componente de destino en función de los valores que vienen del origen (*véase Asignar*

© 2018 Altova GmbH

nombres de nodos).

- Se pueden crear archivos de instancia XML con espacios de nombres personalizados, a nivel de elemento (*véase Declarar espacios de nombres personalizados*).
- Compatibilidad con paquetes de taxonomías XBRL
- Lectura de datos Inline XBRL
- Compatibilidad con directorios EDIFACT D.15B y D.16A (véase la sección *EDI*). El directorio D.16A es compatible con los mensajes Verification of Mass (VERMAS).
- Puede personalizar la validación de mensajes EDI con ayuda de archivos de configuración. Esto permite relajar las reglas de validación cuando sea conveniente o adaptar los valores de código externos de MapForce a los que remite el estándar EDI (*véase Personalizar la validación de datos EDI*).
- Ya puede depurar asignaciones en el complemento de MapForce para Visual Studio y en el complemento de MapForce para Eclipse
- Puede compilar archivos de ejecución de MapForce Server (.mfx) para versiones concretas de MapForce Server (véase *Compilar asignaciones en archivos de ejecución de MapForce Server*).
- Las asignaciones se pueden conectar a bases de datos PostgreSQL a través de conexiones nativas (*véase Configurar una conexión PostgreSQL*).
- Las asignaciones se pueden conectar a bases de datos SQL Server y otros tipos de base de datos a través de proveedores ADO.NET (*véase Configurar una conexión ADO.NET*).
- Compatibilidad con un nuevo tipo de base de datos: Progress OpenEdge. Consulte los apartados Conectarse a Progress OpenEdge (ODBC) y Conectarse a Progress OpenEdge (JDBC).
- A la hora de conectarse a una base de datos a través de JDBC, la ruta de búsqueda de las bibliotecas .jar se puede especificar en el cuadro de diálogo de conexión con la base de datos directamente (*véase Configurar una conexión JDBC*).
- Cuando se actualiza una base de datos con la asignación a través de acciones "Actualizar si... Insertar restantes", se crean instrucciones de combinación MERGE para algunas bases de datos (*véase Instrucciones de combinación MERGE*).
- Actualizaciones internas y mejoras.

Novedades destacadas de **MapForce 2016 R2**:

- Mejoras en el plegamiento de código en el panel *XSLT*: el texto contraído aparece con un símbolo de puntos suspensivos y tiene una vista previa en un cuadro emergente. Estas mejoras también se incluyen en el panel *XQuery* y en el Editor SQL.
- Ahora puede buscar todas las instancias de una función dentro de la asignación activa (en la ventana Bibliotecas, haciendo clic con el botón derecho en la función y seleccionando el comando **Buscar todas las llamadas**).
- Llamadas a servicios web de tipo REST desde una asignación de datos (*véase Agregar una llamada a servicio web (de tipo REST)*).
- Compatibilidad con el directorio EDIFACT 2015A (véase EDI).
- Compatibilidad con Eclipse 4.5 (*véase Complemento de MapForce para Eclipse*).
- Actualizaciones internas y mejoras.

Novedades destacadas de **MapForce 2016**:

- Mejoras en la generación de código XSLT 1.0 (las hojas de estilos generadas ahora se pueden leer más fácilmente y se pueden ejecutar más rápido)

- Nuevas funciones de agregado en la biblioteca de MapForce: **`min-string`** y **`max-string`**. Con estas funciones puede obtener el valor mínimo y máximo de una secuencia de cadenas.
- Ahora puede depurar asignaciones de datos escritas para el motor de ejecución integrado (véase Depuración de asignaciones).
- El complemento de MapForce para Visual Studio ya es compatible con Visual Studio 2015.
- Mayor compatibilidad con bases de datos: SQL Server 2014, Oracle 12c, IBM DB2 10.5, PostgreSQL 9.4, MySQL 5.6.
- Compatibilidad con bases de datos Firebird (véase Conectarse a Firebird (ODBC) y Conectarse a Firebird (JDBC))

Novedades destacadas de **MapForce 2015 Release 4**:

- Compatibilidad con el formato TRADACOMS.
- El complemento de MapForce para Eclipse ya incluye comandos específicos de archivos MapForce bajo un nuevo menú llamado **MapForce** (véase Acceso a menús y funciones frecuentes).
- Actualizaciones internas y mejoras.

Novedades destacadas de **MapForce 2015 Release 3**:

- Opción para suprimir la declaración `<?xml ... ?>` en documentos XML de salida.
- En llamadas a servicios web se puede utilizar autenticación preventiva HTTP, HTTPS y seguridad de servicio web (véase Configuración de seguridad HTTP y Configuración de seguridad de servicio web).
- Lectura de archivos Excel 2013 Strict Open XML.
- Opción de división y conmutación de FlexText compatible con expresiones regulares (véase FlexText y expresiones regulares).
- Componentes basados en texto (incluidos componentes EDI, CSV, campos de longitud fija, JSON y XML) pueden analizar y serializar cadenas además de archivos planos.
- Compatibilidad con bases de datos SQLite (véase Configurar una conexión SQLite).
- Nuevas funciones para completar cadenas de texto: pad-string-left y pad-string-right.
- Nuevo componente: Resultado simple.
- Actualizaciones internas y mejoras.

Novedades destacadas de **MapForce 2015**:

- Nuevo argumento `language` para las funciones format-date y format-dateTime.
- Nueva función de secuencia: replicate-item
- Compatibilidad con bases de enlaces de tablas XBRL
- Nuevo modo para la opción "Dividir una vez" de FlexText: **delimitado** (la línea empieza con).
- Compatibilidad con archivos JSON.

Novedades destacadas de **MapForce 2014 Release 2**:

- Nuevas funciones de secuencia: generate sequence, item-at, etc.
- Opción para definir secciones CDATA en los componentes de salida.

© 2018 Altova GmbH

- Opción para definir valores de tiempo de espera para ejecuciones de BDy llamadas a servicios web.
- Opción para conservar conexiones tras la eliminación de componentes.
- Transferencia masiva de datos de BD (inserción masiva).
- Opción para resaltar automáticamente los elementos obligatorios en los componentes de destino.

Novedades destacadas de **MapForce 2014**:

- Integración del validador RaptorXML y compatibilidad básica con XML Schema 1.1
- Integración de los nuevos motores XSLT y XQuery de RaptorXML
- Compatibilidad con comodines de XML Schema: xs:any y xs:anyAttribute
- Opción para usar comentarios e instrucciones de procesamiento en componentes XML de destino
- Función **Age**
- Opción para siempre insertar comillas en archivos CSV

Novedades destacadas de **MapForce 2013 Release 2 SP1**:

- Altova publica el rapidísimo motor de transformaciónRaptorXML Server

Novedades destacadas de **MapForce 2013 Release 2**:

- Compatibilidad con el módulo MapForce Server.
- Capacidad para generar archivos de ejecución de MapForce Server desde la línea de comandos y ejecutarlo en MapForce Server.
- Capacidad para implementar asignaciones de MapForce en FlowForce Server.
- Compatibilidad con bases de datos Informix 11.7 y mayor compatibilidad con otros tipos de bases de datos.
- Configuración de final de línea definida por el usuario para los archivos de resultados.
- Actualizaciones y mejoras internas

Novedades destacadas de **MapForce 2013**:

- Opción para llamar a procedimientos almacenados en las asignaciones
- Opción para usar funciones de base de datos (similar a la función para procedimientos almacenados)
- Opción para usar instrucciones SELECT con parámetros
- Actualizaciones internas y mejoras

Novedades destacadas de **MapForce 2012 Release 2**:

- Nuevo componente de Ordenación para XSLT 2.0, XQuery y el motor de ejecución integrado
- Nombres de componente definidos por el usuario
- Más funciones para SQL-Where: ORDER BY
- MapForce ya es compatible con archivos lógicos de la base de datos IBM iSeries y muestra los archivos lógicos como vistas
- Compatibilidad con archivos lógicos de IBM DB2. En la edición IBM iSeries de la base de datos DB2 un archivo lógico representa a uno o varios archivos físicos. Un archivo lógico permite al usuario acceder a los datos en una secuencia o en un formato diferentes al del

archivo físico. Los usuarios que se conecten a equipos IBM iSeries pueden encontrar bases de datos ya existentes construidas con archivos lógicos. Antes de la versión 2012 R2 no era posible acceder a estas BD con MapForce.

Novedades destacadas de **MapForce 2012**:

- Transmisión por secuencias de entrada/lectura para archivos XML, CSV y de longitud fija (con el motor de ejecución integrado)
- Nuevo motor de BD compatible con conexiones ODBC y conexiones JDBC
- Alineación automática de componentes en la ventana de asignación
- Nuevas funciones: **`parse-date`** y **`parse-time`**
- Búsqueda de elementos en la ventana Proyecto
- Avisos para conectarse al nodo primario de destino
- Reglas específicas que determinan la secuencia en la que se procesan los componentes en una asignación
- Nueva sección de ejemplos de lenguajes de programación para la API de MapForce

Novedades destacadas de **MapForce 2011 Release 3**:

- Variables intermedias
- Posibilidad de usar varios rangos de filas en componentes Excel
- Compatibilidad con EDI X12 - HIPAA Implementation Guides
- Generación del componente X12 EDI 999 Implementation Acknowledgement
- Firmas XML digitales
- Compatibilidad con US-GAAP 2011
- Compatibilidad con archivos de ensamblado .NET Framework 4.0
- Posibilidad de generar documentos diseñados con StyleVision desde la línea de comandos

Novedades destacadas de **MapForce 2011 Release 2**:

- El motor de ejecución integrado ahora admite la transmisión por secuencias de resultados
- Función de búsqueda en la ventana Bibliotecas
- Asignación invertida
- Función **`IF-ELSE`** ampliable
- Funciones **`node-name`** y funciones de análisis en la biblioteca de funciones **core**
- Nuevo formato EDI IATA PADIS
- Posibilidad de procesar varios mensajes EDI por componente
- Mejoras en el cuadro de diálogo Acciones de tabla de la base de datos con opciones integradas de generación de claves
- Nueva opción para usar StyleVision Power Stylesheets para documentar las asignaciones

Novedades destacadas de **MapForce 2011**:

- Vista previa de componentes intermedios en una asignación en cadena de dos o más componentes conectados a un componente de destino (vista previa de paso a través).
- Funciones de formato para tipos de datos dateTime y números para todos los lenguajes

© 2018 Altova GmbH

compatibles
- Mejoras en la función **auto-number**
- Nuevas funciones de uso horario: **remove-timezone** y **convert-to-utc**
- Vista previa de componentes de destino con ayuda de hojas de estilos Power Stylesheets de StyleVision con gráficos de StyleVision

Novedades destacadas de **MapForce 2010 Release 3**:

- Generación de archivos de proyecto de Visual Studio 2010 para C# y C++
- Posibilidad de definir una fila de hoja de cálculo como nombres de columna en un componente Excel
- Compatibilidad con MSXML 6.0 en el código C++ generado
- Compatibilidad con valores valores nillable y con el atributo xsi:nil en archivos XML de instancia
- Posibilidad para deshabilitar la conversión automática de tipos en los documentos XML
- Compatibilidad con SAP IDocs

Novedades destacadas de **MapForce 2010 Release 2**:

- MapForce Enterprise y Professional Edition para sistemas operativos de 64 bits
- Compatibilidad con Excel 2010
- Conexión automática de conectores secundarios idénticos al mover un conector primario
- Opción para usar campos en el componente WHERE de SQL
- Posibilidad de añadir archivos de clases Java compilados y archivos de ensamblados .NET
- Posibilidad de acortar cadenas de entrada antes de procesarlas
- Validación de archivos de origen EDI UN/EDIFACT y ANSI X12
- Generación del componente X12 EDI 997 Functional Acknowledgement

Novedades destacadas de **MapForce 2010**:

- Varios archivos de entrada/salida por componente
- Mejoras en el uso de rutas de acceso relativas
- Compatibilidad con xsi:type, permitiendo el uso de tipos derivados
- Nuevo sistema interno de tipos de datos
- Mejoras en la navegación de funciones definidas por el usuario
- Validación de datos de salida EDI en el código generado (*véase Validación de componentes EDI*)
- Compatibilidad con los mensajes de servicio EDIFACT CONTRL, AUTACK y KEYMAN
- Compatibilidad con servicios web definidos usando WSDL 2.0
- Mejoras en el tratamiento de contenido mixto en elementos XML

Novedades destacadas de **MapForce 2009 SP1**:

- El usuario puede definir el orden de los parámetros en las funciones definidas por el usuario
- Posibilidad de procesar archivos XML que no son válidos con respecto al esquema XML
- Las funciones normales (estándar) definidas por el usuario ahora admiten parámetros jerárquicos complejos

- Compatibilidad con Apache Xerces 3.x al generar código C++

Novedades destacadas de **MapForce 2009**:

- Posibilidad de usar archivos de instancia XBRL y Dimension XBRL y con taxonomías XBRL como componentes de origen y destino
- Posibilidad de usar EDI HL7 2.2 - 2.6 como componentes de origen y destino
- Posibilidad de usar EDI HL7 versión 3.x XML como componentes de origen y destino
- Documentación de proyectos de asignación
- Soporte nativo para campos XML en SQL Server
- Agrupación de nodos o del contenido del nodo
- Posibilidad de filtrar los datos según la posición del nodo en una secuencia
- Compatibilidad con QName
- Búsqueda de nodos y elementos en componentes

Novedades destacadas de **MapForce 2008 Release 2**:

- Función para generar esquemas XML automáticamente a partir de archivos XML
- Posibilidad de usar archivos Office Open XML Excel 2007 y superior (*.xslx) como componentes de origen y destino
- Compatibilidad con el uso de secuencias como entrada/salida en el código Java y C# generado
- Generación de archivos de proyecto de Visual Studio 2008 para C++ y C#
- Compatibilidad con SOAP 1.2 en servicios web
- Modo nuevo para la opción **División repetida** de FlexText: **delimitado (la línea empieza con)**
- Función para borrar nombres de esquema de BD del código generado
- Instrucciones SQL SELECT como tablas virtuales en componentes de BD
- Relaciones locales: creación instantánea de relaciones de clave foránea/primaria
- Función Recursos globales de Altova
- Mejoras del rendimiento

Novedades destacadas de **MapForce 2008**:

- Funciones de agregado
- Componente de búsqueda Asignación de valores
- Mejoras en las opciones de presentación de XML: pretty-print, omisión de referencias de esquema XML y configuración de codificación para cada componente
- Actualizaciones internas

Novedades destacadas de **MapForce 2007 Release 3**:

- Asignaciones entre datos XML y bases de datos
- Consulta directa de bases de datos
- Filtro SQL-WHERE y asistente para instrucciones SQL
- Mejoras en el generador de código y en la documentación
- Compatibilidad total con EDI X12 versiones 3040 - 5030
- Compatibilidad total con mensajes UN/EDIFACT de los directorios 93A - 06B

© 2018 Altova GmbH

Altova MapForce 2018 Enterprise Edition

Introducción

2 Introducción

En esta sección ofrecemos información general sobre las características de MapForce y su interfaz de usuario y definimos algunos conceptos básicos. Al final de la sección encontrará algunas notas sobre esta documentación.

© 2018 Altova GmbH

2.1 Notas sobre compatibilidad

MapForce® es una aplicación Windows de 32/64 bits compatible con estos sistemas operativos:

- Windows 7 SP1 con actualización de la plataforma, Windows 8, Windows 10
- Windows Server 2008 R2 SP1 con actualización de la plataforma o superior

Solamente las ediciones Enterprise y Professional Edition están disponibles como aplicación de 64 bits.

MapForce también puede utilizarse como complemento e integrarse con estos entornos de desarrollo:

- Visual Studio 2008/2010/2012/2013/2015/2017 (*véase Complemento de MapForce para Visual Studio*).
- Eclipse 4.5 / 4.6 / 4.7 (*véase Complemento de MapForce para Eclipse*).

MapForce también se integra con los productos de Microsoft Office:

- para crear asignaciones de datos de bases de datos Access (*véase Bases de datos en MapForce*).
- para generar la documentación de la asignación en Word 2000 o superior (*véase Generar y personalizar la documentación de la asignación*).
- para crear asignaciones de datos de hojas de cálculo Excel 2007 o superior (*véase Microsoft OOXML Excel 2007+*).

Para más información sobre compatibilidad en la función de generación de código de programa consulte el apartado Introducción al generador de código.

Consulte también el apartado Datos técnicos, que incluye información general práctica sobre aspectos técnicos del software.

2.2 ¿Qué es MapForce?

Sitio web de Altova: HErramienta de asignación de datos

MapForce es un entorno IDE multiuso basado en Windows que sirve para transformar datos de un formato en otro o pasar datos de un esquema a otro mediante operaciones visuales de arrastrar y colocar. Para ello ofrece una interfaz gráfica donde el usuario no necesita crear código de programación. De hecho, MapForce genera automáticamente el código de programa necesario para realizar la transformación de datos propiamente dicha (o la asignación de datos). Si prefiere no generar código de programa, puede simplemente ejecutar la transformación con el lenguaje de transformación integrado de MapForce (disponible en las ediciones Professional y Enterprise).

Con MapForce podrá convertir y transformar datos con total comodidad, independientemente del formato. MapForce infiere automáticamente la estructura de los datos y también ofrece la posibilidad de usar esquemas para lo datos o generar el esquema automáticamente a partir de un archivo de instancia. Por ejemplo, si tiene un archivo de instancia XML pero no tiene una definición de esquema, MapForce puede generarlo automáticamente. Así podrá asignar los datos del archivo XML a otros archivos o formatos.

Estas son las tecnologías que puede usar como origen o destino de las asignaciones de datos en MapForce:

© 2018 Altova GmbH

MapForce Basic Edition	MapForce Professional Edition	MapForce Enterprise Edition
• XML y XML schema • HL7 versión 3.x (basado en esquemas)	• XML y XML schema • Archivos planos, incluido el formato CSV (valores separados por comas) y FLF (campo de longitud fija) • Bases de datos (las principales bases de datos relacionales, incluidas las de Microsoft Access y SQLite)	• XML y XML schema • Archivos planos, incluido el formato CSV (valores separados por comas) y FLF (campo de longitud fija) • Datos de archivos de texto heredados pueden asignarse y convertirse a otros formatos con la aplicación MapForce FlexText • Bases de datos (las principales bases de datos relacionales, incluidas las de Microsoft Access y SQLite) • Familia de formatos EDI (incluidos UN/EDIFACT, ANSI X12, HL7, IATA PADIS, SAP IDoc, TRADACOMS) • Archivos JSON • Archivos Microsoft Excel 2007 y superior • Archivos de instancia y taxonomías XBRL

Estos son los lenguajes de transformación de datos disponibles en cada edición de MapForce:

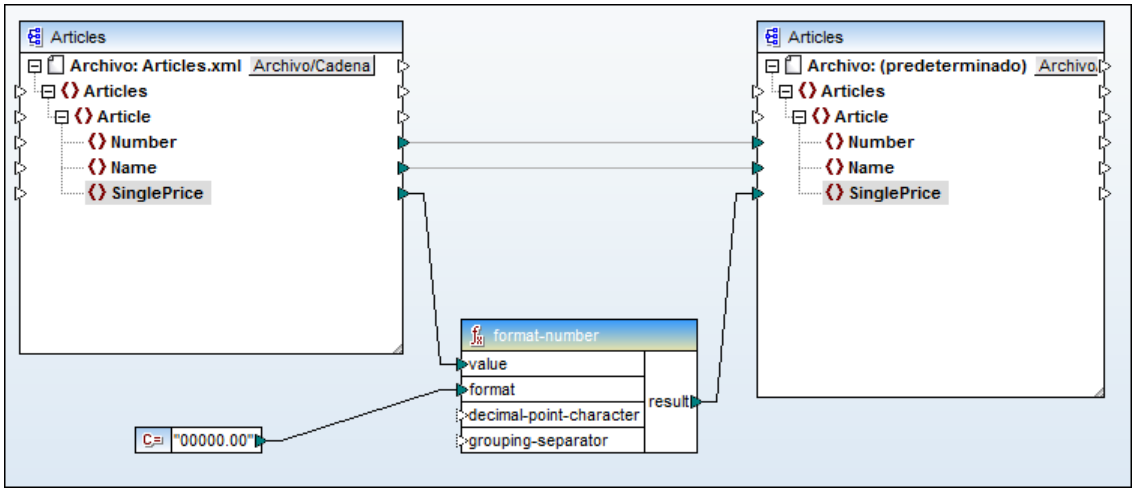
MapForce Basic Edition	MapForce Professional Edition	MapForce Enterprise Edition
• XSLT 1.0 • XSLT 2.0	• Lenguaje de transformación integrado de MapForce • XSLT 1.0 • XSLT 2.0 • XQuery • Java • C# • C++	• Lenguaje de transformación integrado de MapForce • XSLT 1.0 • XSLT 2.0 • XQuery • Java • C# • C++

En la interfaz gráfica puede consultar una vista previa de los resultados de la transformación y el código XSLT o XQuery generado. Recuerde que durante el proceso de diseño MapForce valida constantemente la integridad de los esquemas y de las transformaciones y muestra errores de validación en una ventana de mensajes. Así podrá corregir errores inmediatamente.

Cuando elija los lenguajes de transformación Java, C# o C++, MapForce generará los proyectos y las soluciones necesarios para que pueda abrirlos en Visual Studio o Eclipse directamente y

pueda ejecutar el programa de asignación de datos generado. Cuando se trate de un proyecto de integración de datos más complejo, podrá añadir código al programa generado con ayuda de las bibliotecas de Altova y de la API de MapForce.

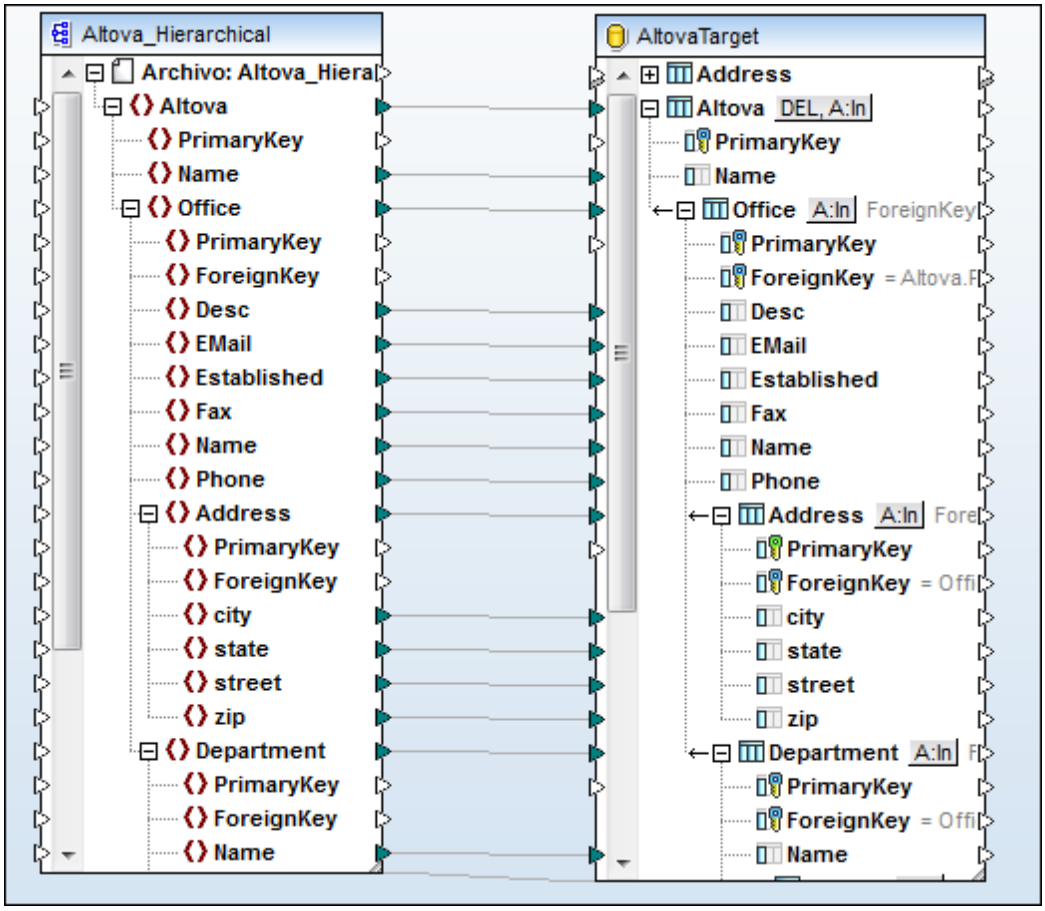
En MapForce las transformaciones de datos se diseñan de forma visual. Por ejemplo, si trabaja con datos XML podrá conectar los elementos, atributos o comentarios de un archivo XML a los elementos, atributos o comentarios de otro archivo XML. Estas conexiones ordenan que se lean los datos del elemento o atributo de origen y se escriban en el elemento o atributo de destino.



Ejemplo de transformación de datos con dos archivos XML

Asimismo, cuando trabaje con bases de datos en la edición Professional o Enterprise, podrá ver las columnas de la BD en MapForce y crear asignaciones gráficas entre el origen y el destino. Recuerde que cuando configure la conexión a la BD, podrá elegir el controlador de BD y el tipo de conexión (ADO, ODBC o JDBC). Por último, podrá generar consultas SQL de forma gráfica, utilizar procedimientos almacenados o consultar la BD directamente (aunque las opciones disponibles dependen del tipo de BD, la edición y el controlador).

© 2018 Altova GmbH



Ejemplo de transformación de datos con un archivo XML y una base de datos

La manera más sencilla de describir un diseño de asignación creado con MapForce es diciendo que el diseño lee lo datos del origen X y los escribe en el destino Y. Sin embargo, podrá crear diseños más complejos que lean datos del origen X y los escriba en Y y después lea datos del origen Y y los escriba en el destino Z. En este caso hablaríamos de asignaciones *encadenadas* o *de paso a través*. Este tipo de asignaciones permiten el acceso a datos del paso intermedio del proceso de asignación (para guardarlos en un archivo, por ejemplo).

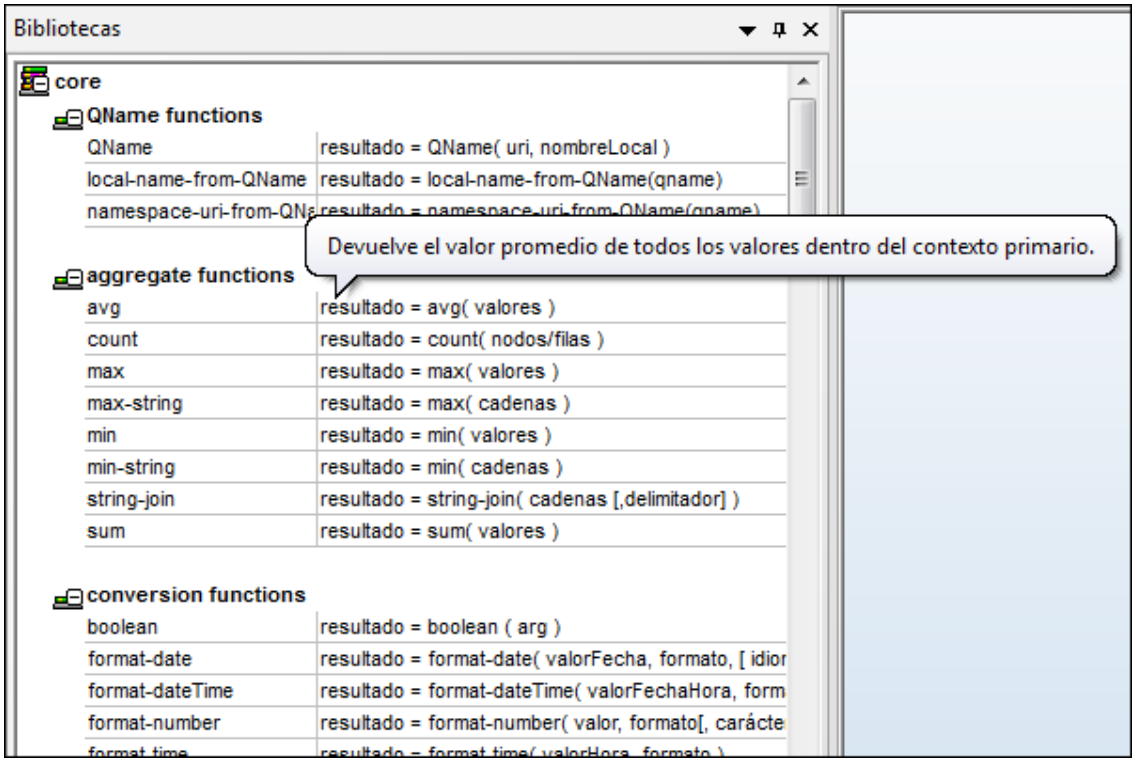
Además, en una transformación de datos de MapForce pueden procesarse varios archivos de entrada de forma dinámica y pueden generarse varios archivos de salida. Por tanto, podrá crear diseños que lean datos de varios archivos X y los escriban en un solo archivo Y o diseños que lean datos del archivo X y los escriba en varios archivos Y, etc.

Es importante tener en cuenta que se pueden mezclar varios orígenes y varios destinos en la misma transformación y que pueden ser de cualquier tipo de datos compatible. Por ejemplo, si trabaja con MapForce Professional o Enterprise, podrá combinar datos de dos bases de datos distintas en un solo archivo XML. O podrá combinar datos de varios archivos XML y escribir algunos datos en una base de datos y otros en otra. Antes de confirmar los cambios en las bases de datos podrá consultar una vista previa de las instrucciones SQL.

En ocasiones el proyecto de integración de datos exige procesar los datos (mediante ordenación, agrupación o filtrado, por ejemplo) antes de escribirlos en el destino. Por este motivo, MapForce

ofrece varios componentes funcionales muy prácticos que no son más que construcciones de lenguaje de programación simplificadas (como constantes, variables, condiciones WHERE de SQL, componentes de filtrado y ordenación, etc.). Por otra parte, MapForce ofrece extensas bibliotecas de funciones que le ayudarán a manipular todo tipo de datos.

La biblioteca integrada de funciones puede ampliarse con funciones diseñadas en MapForce (las denominadas *funciones definidas por el usuario*) o con bibliotecas externas creadas en XSLT, XQuery, Java o C#.



Ventana Bibliotecas

Si tiene demasiados archivos de asignación, puede organizarlos en un proyecto de asignación (en la edición Enterprise y Professional). Además podrá generar código de programa para el proyecto entero, en lugar de generar código para cada asignación.

Si tiene requisitos más avanzados (p. ej. cuando tenga que ejecutar transformaciones con la API de MapForce Server), puede diseñar una asignación de modo que se le puedan pasar valores en tiempo de ejecución o pueda obtener un valor de cadena simple en tiempo de ejecución. Esta característica también permite probar el resultado de funciones o asignaciones enteras que producen un valor de cadena simple. Las ediciones Enterprise y Professional también incluyen componentes que permiten analizar y serializar cadenas en tiempo de ejecución.

En la edición Enterprise puede diseñar servicios web SOAP 1.0 y SOAP 2.0 Web de forma gráfica a partir de archivos WSDL. También podrá realizar llamadas a servicios web WSDL 1.0/2.0 desde la asignación y obtener datos desde ellos. Esto incluye servicios web a los que se accede por protocolos HTTP y HTTPS y servicios web que exigen que el usuario utilice el mecanismo de seguridad de servicio web o de autenticación HTTP.

 © 2018 Altova GmbH

En las ediciones Enterprise y Professional puede generar documentación detallada sobre los archivos de diseño en formato HTML, Word 2007+ o RTF. El diseño de la documentación se puede personalizar (p. ej. puede incluir o excluir determinados componentes).

MapForce se integra perfectamente con otros productos de Altova MissionKit y otros productos servidor de Altova.

<table>
<tr><th>MapForce
Basic Edition</th><th>MapForce
Professional Edition</th><th>MapForce
Enterprise Edition</th></tr>
<tr><td colspan="3">Puede ejecutar el código XSLT generado directamente en MapForce y obtener una vista previa del resultado de la transformación de datos. Cuando necesite un mayor rendimiento, puede procesar la asignación con RaptorXML Server, un rapidísimo motor de transformación XML.</td></tr>
<tr><td colspan="3">Si tiene XMLSpy instalado en el mismo equipo que MapForce, podrá abrir y editar cualquier tipo de archivo compatible en XMLSpy desde MapForce directamente. Por ejemplo, si hace clic con el botón derecho en un componente XML de MapForce, el menú contextual incluye el comando Componente | Editar la definición del esquema en XMLSpy.</td></tr>
<tr><td></td><td colspan="2">Puede ejecutar las transformaciones de datos en MapForce directamente o implementarlas en otro equipo e incluso en otro sistema operativo para ejecutarlas de forma automatizada. Concretamente puede diseñar asignaciones en Windows y ejecutarlas en equipos servidor Windows, Linux o Mac que tengan instalado MapForce Server (como producto independiente o bajo control de FlowForce Server).</td></tr>
<tr><td></td><td colspan="2">Si tiene StyleVision instalado en el mismo equipo que MapForce, podrá usar hojas de estilos de StyleVision o diseñar hojas nuevas para obtener vistas previas del resultado de la transformación de datos en formato HTML, RTF, PDF o Word 2007+.</td></tr>
</table>

La edición Enterprise y Professional de MapForce se puede instalar como complemento de Visual Studio y Eclipse. Así podrá diseñar asignaciones y acceder a las funciones de MapForce sin salir de su entorno de desarrollo favorito.

En MapForce no solo puede personalizar el aspecto del entorno de desarrollo (la interfaz gráfica del usuario), sino que además puede configurar todos los tipos de componente y de formato:

- Si se trata de una asignación de datos XML, puede elegir si se incluye una referencia de esquema o si la declaración XML se suprime en los archivos XML de salida. También puede elegir la codificación de los archivos generados (p. ej. UTF-8).
- Si se trata de una asignación de datos de BD, puede definir el tiempo de espera para la ejecución de instrucciones, puede elegir si MapForce utiliza las transacciones de la BD o si a la hora de generar código se elimina el nombre del esquema de la BD de los nombres de tabla.
- Si se trata de una asignación de datos XBRL, puede seleccionar las vistas que MapForce debe mostrar (p. ej. la vista *Bases de enlaces de presentación y definición*, la vista *Base de enlaces de tabla* o la vista *Todos los conceptos*).

Todas las ediciones de MapForce tienen una versión de 32 bits. Además, las ediciones Enterprise y Professional están disponibles en 64 bits.

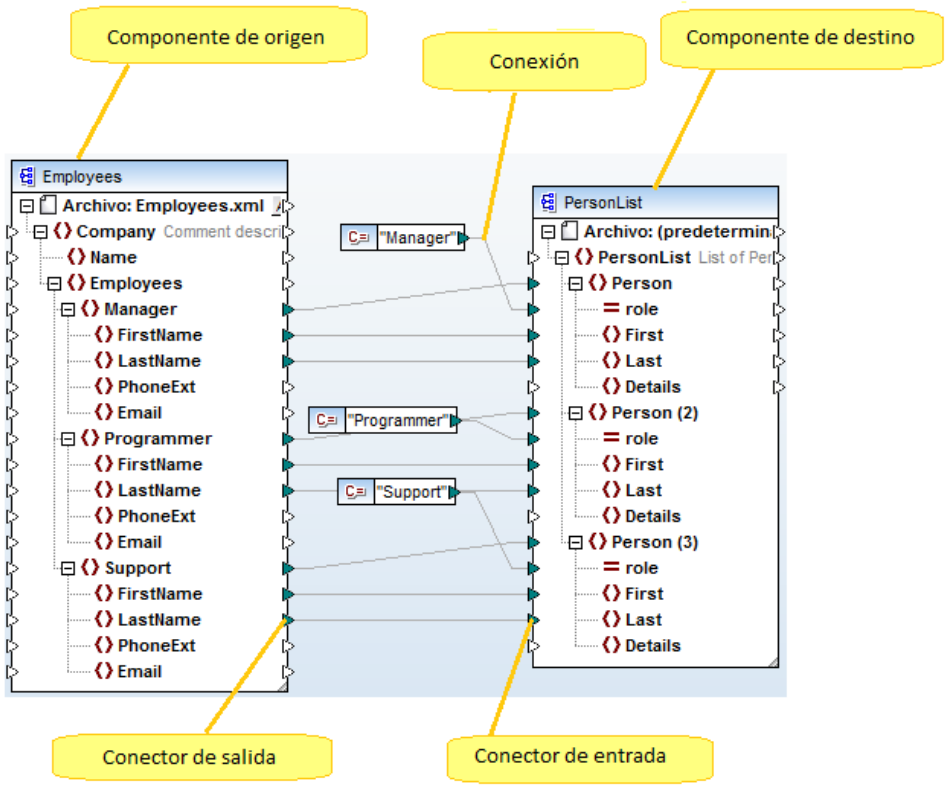
© 2018 Altova GmbH

2.3 Conceptos básicos

En este apartado definimos conceptos básicos de las asignaciones de datos.

Asignación

Un diseño de asignación de datos (o simplemente *asignación de datos*) es la representación visual de cómo se deben transformar datos de un formato a otro. Una asignación está formada por componentes, que se van añadiendo al área de asignación de MapForce para crear las transformaciones de datos (p. ej. para convertir documentos XML que siguen un esquema en documentos que siguen otro esquema). Una asignación válida está formada por uno o varios componentes de origen que están conectados a uno o varios componentes de destino. La asignación se puede ejecutar y MapForce ofrece una vista previa del resultado. También puede generar código desde MapForce y ejecutar la asignación en una aplicación externa. Por último, puede compilar la asignación en un archivo de ejecución de MapForce y automatizar la ejecución con MapForce Server o FlowForce Server. Los diseños de asignación de MapForce tienen la extensión de archivo **.mfd**.



Estructura básica de una asignación de datos de MapForce

Componente

En MapForce un componente es el elemento gráfico que representa visualmente la estructura (esquema) de los datos o el elemento que determina cómo se deben transformar los datos (funciones). Los componentes son piezas fundamentales necesarias para construir una asignación. En el área de asignación de MapForce los componentes aparecen en forma de rectángulo. Estos son algunos ejemplos de componente:

- Constantes
- Bases de datos
- Filtros
- Condiciones
- Archivos planos (archivos de texto, CSV o de longitud fija)
- Funciones
- Documentos EDI (UN/EDIFACT, ANSI X12, HL7)
- Archivos Excel 2007+
- Componentes de entrada simples
- Componentes de salida simples
- Variables
- Documentos XBRL
- Esquemas XML y documentos DTD

Conector

Un conector es un pequeño triángulo situado a la izquierda o derecha de un componente. Los conectores situados a la izquierda del componente ofrecen puntos de entrada para el componente. Los situados a la derecha ofrecen puntos de salida desde el componente.

Conexión

Una conexión es la línea que se puede dibujar para unir dos conectores. Al dibujar la línea estamos dando a MapForce instrucciones de transformar lo datos de una forma determinada (p. ej. leer datos de un documento XML y escribirlos en otro documento XML).

Componente de origen

Un componente de salida es un componente donde MapForce lee datos. Al ejecutar la asignación, MapForce lee lo datos que aporta el conector del componente de origen, los convierte al tipo elegido y los envía al conector del componente de destino.

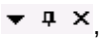
Componente de destino

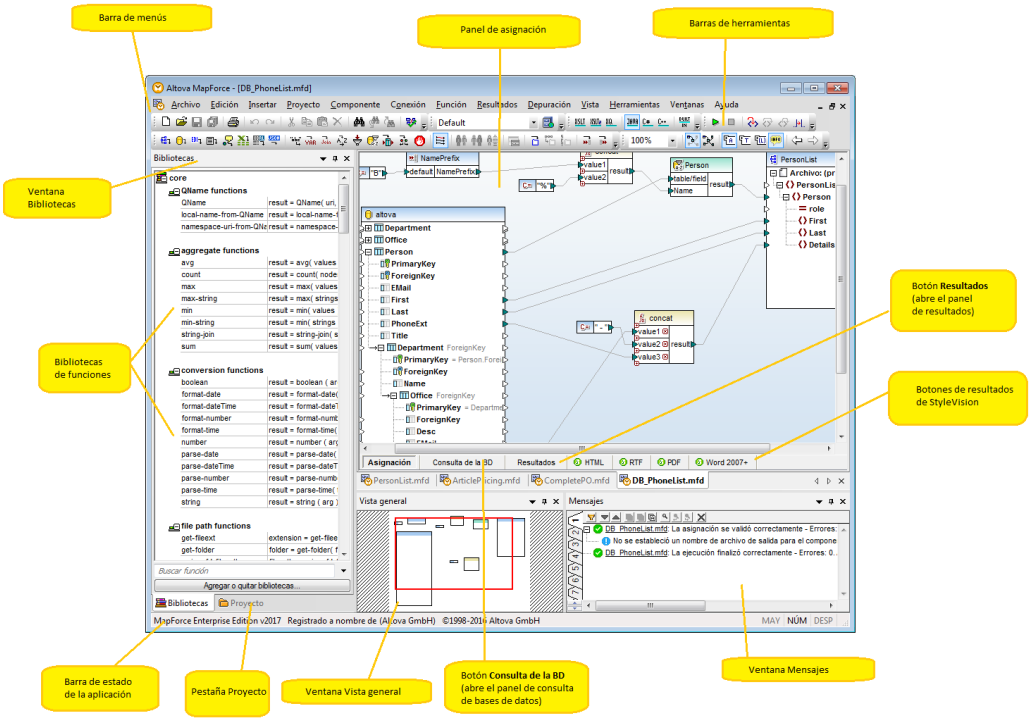
Un componente de destino es un componente donde MapForce escribe datos. Al ejecutarse la asignación, el componente de destino da a MapForce instrucciones para generar archivos de salida o generar el resultado como valor de cadena para poder procesarlo en una aplicación externa. El componente de destino es lo opuesto de un componente de origen.

© 2018 Altova GmbH

2.4 Interfaz del usuario

La interfaz gráfica del usuario de MapForce está organizada como un entorno de desarrollo integrado. A continuación puede ver los componentes principales de la interfaz. Además, puede cambiar la configuración de la interfaz con el comando de menú **Herramientas | Personalizar**.

Todas las ventanas de la interfaz incluyen los botones  , que sirven para mostrar, ocultar, anclar o acoplar las ventanas. Si necesita restaurar las barras de herramientas y ventanas a su estado predeterminado, haga clic en el comando de menú **Herramientas | Restaurar barras de herramientas y ventanas**.



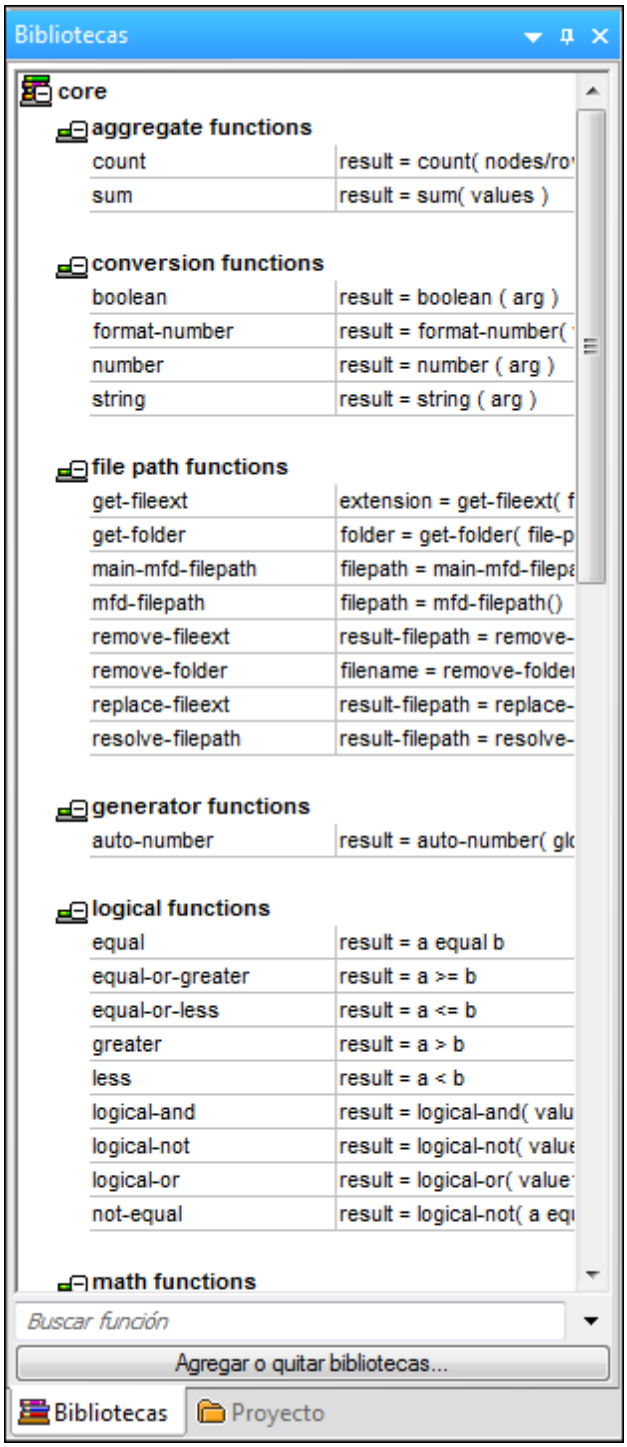
Interfaz gráfica del usuario de MapForce (Professional Edition)

Barra de menú y barras de herramientas

La barra de menú muestra los diferentes menús de la aplicación. Por su parte, las barras de herramientas muestran un grupo de botones que corresponden a comandos de MapForce. Las barras de herramientas se pueden arrastrar a una posición diferente si lo desea.

Ventana Bibliotecas

En la ventana Bibliotecas aparecen todas las funciones integradas de MapForce, organizadas por biblioteca. La lista de funciones que aparecen en esta ventana depende del lenguaje de transformación elegido. Además, en esta ventana aparecen también las funciones definidas por el usuario que se hayan creado y las bibliotecas externas que se hayan importado.

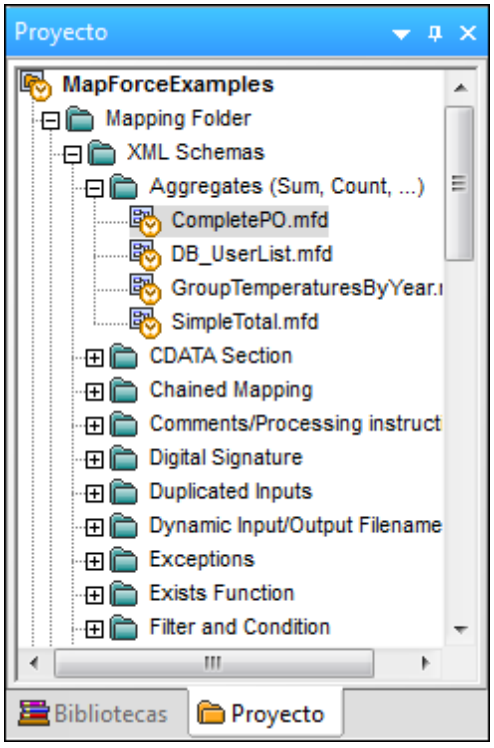


© 2018 Altova GmbH

Para buscar funciones por nombre o por su descripción, escriba el valor de búsqueda en el cuadro de texto situado en la parte inferior de la ventana Bibliotecas. Para buscar todas las instancias de una función dentro de la asignación activa basta con hacer clic con el botón derecho en la función y elegir el comando **Buscar todas las llamadas** en el menú contextual. En la ventana Bibliotecas también puede ver el tipo de datos de la función y su descripción. Para más información consulte la sección Trabajar con funciones.

Ventana Proyecto

MapForce ofrece una interfaz compatible con múltiples documentos y permite agrupar las asignaciones en proyectos. La ventana Proyecto muestra todos los archivos y todas las carpetas del proyecto. La extensión de los archivos de proyecto de MapForce es **`*.mfp`**. Cuando quiera buscar una asignación dentro de un proyecto, haga clic dentro de la ventana Proyecto y pulse **Ctrl+F**. Para obtener más información consulte la sección Trabajar con proyectos de asignación de datos.



Panel *Asignación*

El panel *Asignación* es el área de trabajo donde se diseñan las asignaciones. En este panel puede añadir componentes de asignación (archivos, esquemas, constantes, variables, etc.) desde el menú **Insertar** (*véase Agregar componentes a la asignación*). Además puede arrastrar funciones desde la ventana Bibliotecas hasta el panel *Asignación* (*véase Trabajar con funciones*).

Panel *XSLT* (*XSLT2*)

El panel *XSLT* (o *XSLT2*) muestra el código de transformación XSLT 1.0 (o 2.0) que se genera a partir de la asignación. Para cambiar a este panel seleccione el lenguaje de transformación XSLT (o XSLT 2) y después haga clic en **XSLT** (o **XSLT2**).

Este panel incluye funciones de numeración de líneas y plegamiento de código. Para expandir y contraer porciones de código basta con hacer clic en lo iconos **+** y **-** situados en el lateral izquierdo del panel. Las porciones de código contraído se señalan con puntos suspensivos. Para ver el código contraído sin expandirlo, pase el cursor por encima de los puntos suspensivos y aparecerá un cuadro emergente con una vista previa del código (*imagen siguiente*). No olvide que si la vista previa no cabe en el cuadro emergente, al final del código aparecen otra vez puntos suspensivos.



Puede configurar las opciones de presentación del código (guías de sangría, marcadores de fin de línea, etc.) haciendo clic con el botón derecho dentro del panel y seleccionando el comando configurar la **Configurar la vista Texto** en el menú contextual. También puede hacer clic en el botón **Configurar la vista Texto** de la barra de herramientas.

Panel *XQuery*

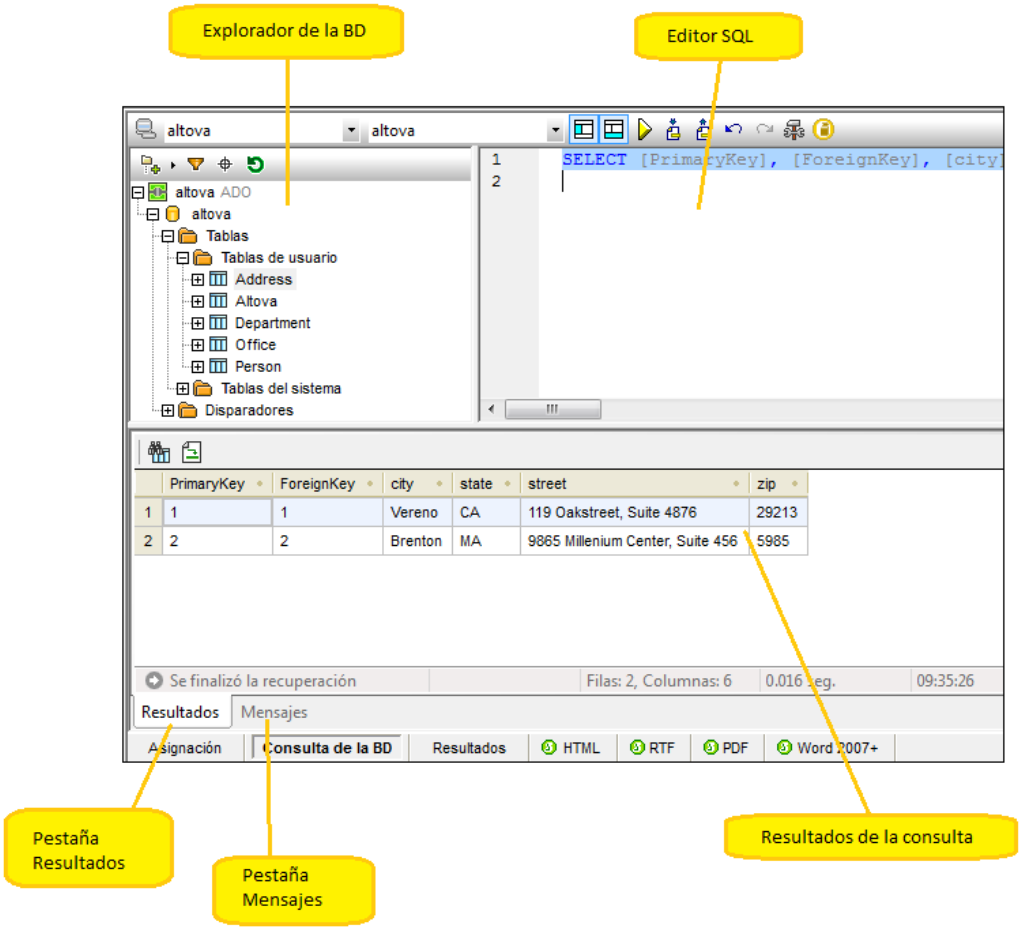
El panel *XQuery* muestra el código de transformación XQuery que se genera a partir de la asignación. Para cambiar a este panel seleccione el lenguaje de transformación XQuery y después haga clic en **XQuery**.

Este panel ofrece funciones de numeración de líneas y plegamiento de código idénticas a las del panel *XSLT*.

Panel *Consulta de BD*

El panel *Consulta de BD* sirve para consultar las principales bases de datos. En este panel puede trabajar con varias conexiones activas a diferentes bases de datos.

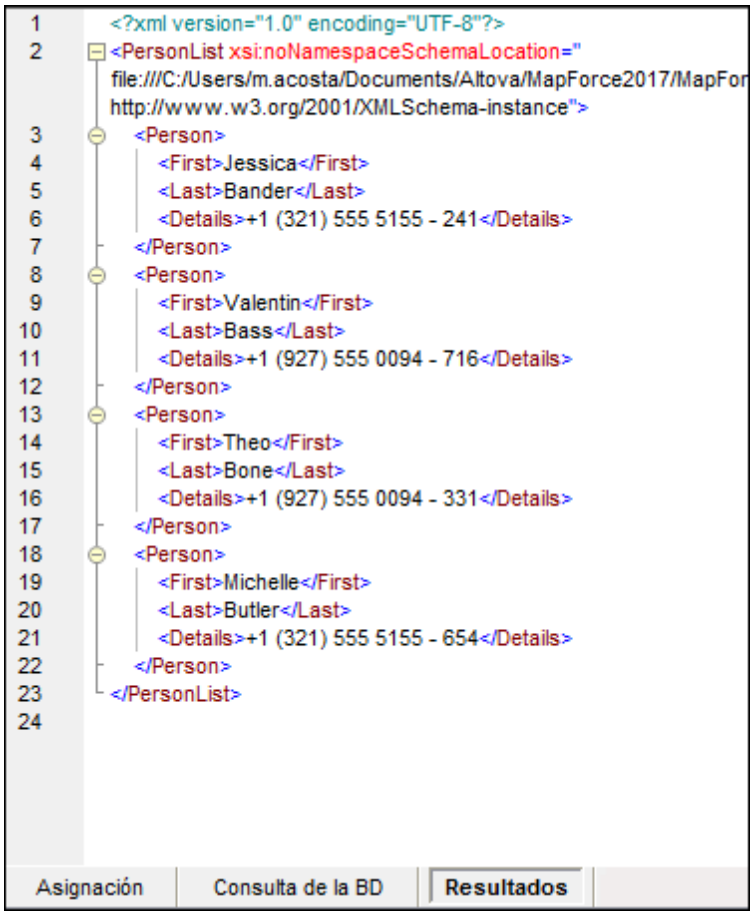
© 2018 Altova GmbH



Para más información consulte el apartado Examinar y consultar bases de datos.

Panel *Resultados*

El panel *Resultados* muestra el resultado de la transformación de datos (p. ej. un archivo XML) y se abre haciendo clic en **Resultados**. Si la asignación genera varios archivos podrá navegar por echos de forma secuencial.



Este panel también incluye funciones de numeración de líneas y plegamiento de código idénticas a las del panel *XSLT*.

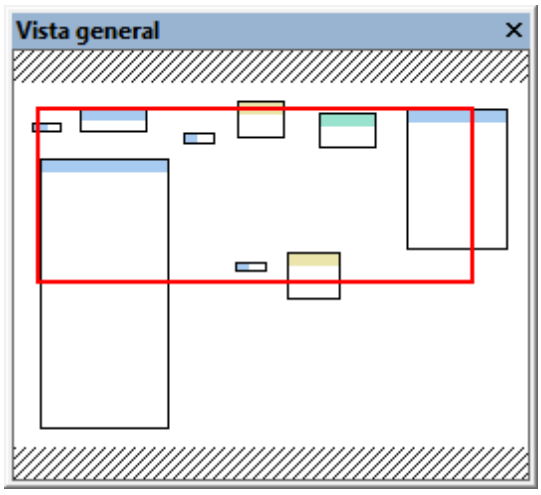
Resultado de StyleVision (HTML, RTF, PDF y Word)

Si tiene instalado Altova StyleVision (https://www.altova.com/es/stylevision.html), la interfaz de MapForce incluye también botones para obtener una vista previa de los resultados de StyleVision y guardarlos en formato HTML, RTF, PDF y Word 2007+. Esto es posible gracias a los archivos SPS (StyleVision Power Stylesheet) diseñados en StyleVision y asignados a un componente de asignación en MapForce.

Ventana Vista general

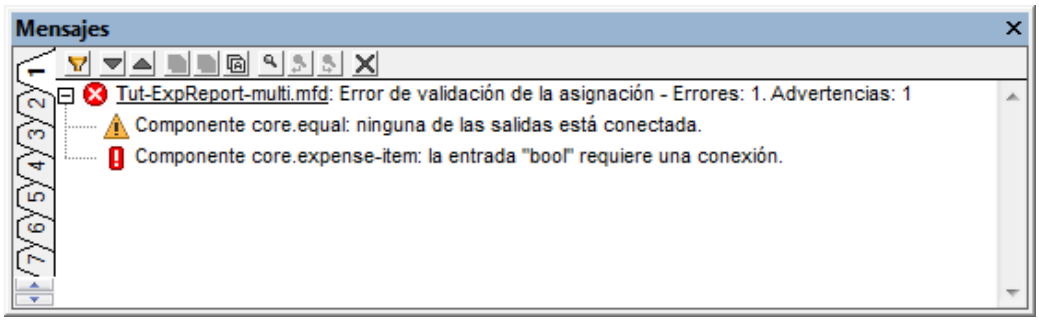
La ventana Vista general ofrece una perspectiva general del panel *Asignación*. Desde esta ventana podrá navegar más fácilmente por el área de asignación cuando la asignación sea muy grande. Para navegar por la asignación basta con arrastrar el rectángulo rojo por la ventana.

© 2018 Altova GmbH



Ventana Mensajes

La ventana Mensajes muestra mensajes, errores y advertencias cuando se ejecuta la asignación (*véase* *Vista previa de resultados*) o cuando se valida la asignación (*véase* *Validar asignaciones*).



Para ir hasta el componente o estructura que provocó el mensaje, el error o la advertencia haga clic en el texto subrayado en la ventana Mensajes.

El resultado de la operación de ejecución o validación aparece en la ventana Mensajes acompañado de uno de estos iconos de estado:

Icono	Descripción
	La operación finalizó correctamente.
	La operación finalizó con advertencias.
	Error en la operación.

La ventana Mensajes también puede mostrar mensajes de información, advertencia y error:

Icono	Descripción
	Mensaje de información. La ejecución de la asignación no se interrumpe.
	Mensaje de advertencia. La ejecución de la asignación no se interrumpe. Pueden generarse cuando, por ejemplo, no se crean conexiones obligatorias con conectores de entrada. Cuando esto ocurra, se generarán resultados para los componentes que tengan conexiones válidas.
	Mensaje de error. Cuando ocurre un error falla la ejecución de la asignación y no se genera ningún resultado. Tampoco es posible acceder a la vista previa del código XSLT o XQuery.

Además de los iconos anteriores la ventana Mensajes ofrece botones para completar estas acciones:

Icono	Descripción
	Filtra los mensajes según el nivel de gravedad (información, errores, advertencias). Seleccione **Activar todos** para incluir todos los niveles de gravedad (opción predeterminada). Seleccione **Desactivar todos** para quitar todos los niveles de gravedad del filtro. Cuando seleccione esta opción, solamente se mostrarán mensajes relacionados con la ejecución o con el estado de validación.
	Ir a la siguiente línea.
	Ir a la línea anterior.
	Copiar la línea seleccionada en el portapapeles.
	Copiar la línea seleccionada en el portapapeles, incluidas todas sus líneas anidadas.
	Copiar todo el contenido de la ventana Mensajes en el portapapeles.
	Buscar un texto en la ventana Mensajes. Ofrece una opción para buscar **Solo palabras completas** y la opción **Coinc. mayús/min** para refinar la búsqueda.
	Buscar un texto desde la línea seleccionada hasta el final de los mensajes.
	Buscar un texto desde la línea seleccionada hasta el principio de los mensajes.
	Borra el contenido de la ventana Mensajes.

Cuando trabaje con varios archivos de asignación a la vez, puede consultar los mensajes de información, advertencia y error de cada archivo en pestañas diferentes. Para ello, antes de ejecutar o validar la asignación, seleccione la pestaña numerada correspondiente en el lateral izquierdo de la ventana Mensajes.

© 2018 Altova GmbH

Barra de estado de la aplicación

La barra de estado está situada en la parte inferior de la ventana de la interfaz y muestra información general sobre la aplicación. Por ejemplo, muestra información contextual sobre el botón o comando señalado con el puntero del ratón. Si usa la versión de 64 bits de MapForce, el nombre de la aplicación que aparece en la barra de estado incluye el sufijo (x64). La versión de 32 bits no incluye ningún sufijo.

2.5 Notas generales

Archivos de ejemplo

La mayoría de los archivos de diseño de asignación de datos que aparecen en esta documentación pueden encontrarse en estas carpetas:

- `C:\Usuarios\<usuario>\Documentos\Altova\MapForce2018\MapForce Examples`
- `C:\Usuarios\<usuario>\Documentos\Altova\MapForce2018\MapForce Examples\Tutorials`

Las asignaciones de datos y archivos de instancia que vienen con MapForce sirven para aprender a usar la mayoría de las características del software. Por este motivo le animamos a crear copias de estos archivos y experimentar con ellas.

Interfaz gráfica del usuario

Algunas imágenes (capturas de pantalla) que acompañan a esta documentación muestran elementos de la interfaz gráfica del usuario que quizás no correspondan a la edición de MapForce que tiene instalada. Por lo general, las capturas de pantalla incluyen el nombre del diseño de asignación (archivo **`*.mfd`**), así como la edición de MapForce a la que corresponde la imagen.

© 2018 Altova GmbH

Altova MapForce 2018 Enterprise Edition

Tutoriales

3 Tutoriales

Los tutoriales de MapForce están diseñados para ayudarle a comprender y a utilizar las funciones básicas de transformación de datos de MapForce lo más rápido posible. El objetivo de este *curso rápido sobre MapForce* no es explicar en detalle todas las funciones de MapForce, sino describir sus funciones básicas paso a paso por lo que recomendamos seguir los tutoriales en el orden correcto. También recomendamos comprender bien todos los conceptos de un tutorial antes de pasar al siguiente porque los tutoriales aumentan en complejidad. También es recomendable tener algo de experiencia previa con XML y XML Schema.

Pasar datos XML a un esquema nuevo

En este tutorial aprenderá a pasar datos de una estructura XML a otra usando el lenguaje XSLT 2.0 y sin necesidad de escribir ni una línea de código. El tutorial también define los conceptos de *secuencias* y *elementos* y explica cómo crear conexiones de asignación, usar funciones, validar una asignación y obtener una vista previa y guardar los resultados en el disco.

Asignar varios orígenes de datos a un destino

En este tutorial aprenderá a leer datos de dos archivos XML con esquemas diferentes y combinarlos en un solo archivo XML de destino. El tutorial también explica cómo se cambia el nombre y los archivos de instancia de cada componente de la asignación y define el concepto de *duplicados de entrada*.

Trabajar con varios esquemas de destino

En este tutorial aprenderá a trabajar con asignaciones más complejas que producen dos o más resultados. Concretamente aprenderá a generar en la misma asignación un archivo XML que almacena una lista de libros y otro archivo XML que solo contiene un subconjunto de los libros del primer archivo, filtrados por año de publicación. Para ello deberá utilizar el componente **Filtro**, una función y una constante numérica.

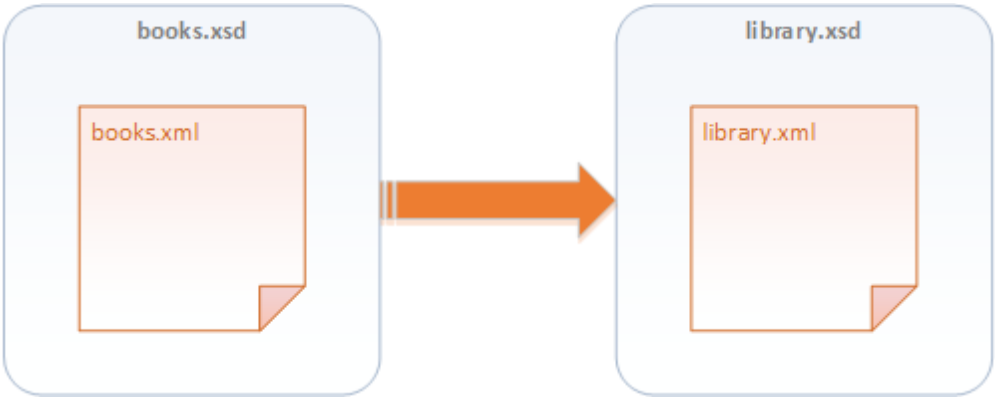
Procesar y generar archivos de forma dinámica

En este tutorial aprenderá a leer datos de varios archivos XML de instancia ubicados en la misma carpeta y escribir los datos en varios archivos XML generados instantáneamente. El tutorial también explica cómo quitar las declaraciones XML y de esquema y cómo usar funciones para concatenar cadenas y extraer extensiones de archivo.

© 2018 Altova GmbH

3.1 Pasar datos XML a un esquema nuevo

Este tutorial explica cómo pasar datos de un archivo XML a otro y define los conceptos básicos del entorno de desarrollo de MapForce. Ambos archivos XML sirven para almacenar una lista de libros, pero sus elementos tienen nombres diferentes y siguen estructuras distintas (es decir, los archivos tienen diferentes esquemas).



Modelo abstracto de la transformación de datos

El fragmento de código que aparece a continuación muestra algunos datos del archivo que usaremos como origen de datos (para mayor simplicidad se omitieron las declaraciones XML y de espacio de nombres).

```
<books>
   <book id="1">
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <category>Fiction</category>
      <year>1876</year>
   </book>
   <book id="2">
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <category>Fiction</category>
      <year>1912</year>
   </book>
</books>
```

books.xml

Y a continuación puede ver los mismos datos en el archivo de destino:

```
<library>
   <last_updated>2015-06-02T16:26:55+02:00</last_updated>
   <publication>
      <id>1</id>
```

```
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <genre>Fiction</genre>
      <publish_year>1876</publish_year>
   </publication>
   <publication>
      <id>2</id>
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <genre>Fiction</genre>
      <publish_year>1912</publish_year>
   </publication>
</library>
```

library.xml

Como puede observar, algunos nombres de elemento no coinciden en el XML de origen y de destino. Nuestro objetivo es rellenar los elementos `<author>`, `<title>`, `<genre>` y `<publish_year>` del archivo de destino con los datos de los elementos equivalentes del archivo de origen (`<author>`, `<title>`, `<category>`, `<year>`). El atributo `id` del archivo XML de origen debe asignarse al elemento `<id>` del archivo XML de destino. Por último, debemos rellenar el elemento `<last_updated>` del archivo XML de destino con la fecha y la hora en que se actualizó el archivo por última vez.

Para conseguir nuestro objetivo debemos seguir estos pasos:

Paso nº1: seleccionar XSLT2 como lenguaje de transformación

Esto puede hacerse de dos maneras distintas:

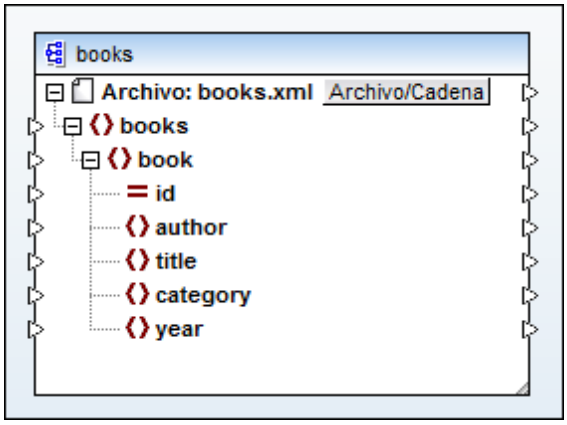
- Haciendo clic en el botón **XSLT2**  de la barra de herramientas.
- Haciendo clic en el comando **XSLT 2.0** del menú **Resultados**.

Paso nº2: agregar el archivo XML de origen a la asignación

El archivo XML de origen para la asignación está en esta ruta de acceso `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\books.xml`. Hay varias maneras de añadir un archivo XML de origen a la asignación:

- Haciendo clic en el botón **Insertar archivo o esquema XML**  de la barra de herramientas.
- Haciendo clic en el comando **Archivo o esquema XML** del menú **Insertar**.
- Arrastrando el archivo XML desde el explorador de Windows hasta el panel de asignación.

Una vez añadido el archivo al panel de asignación, podrá ver su estructura, definida claramente de forma gráfica. En MapForce esta estructura se denomina componente de asignación o *componente* sencillamente. Puede expandir y contraer los elementos del componente con los iconos ⊟ y ⊞ o pulsando las teclas **+** y **-** del teclado numérico.



Componente de asignación

Para mover el componente a otra posición del panel de asignación basta con arrastrarlo por la barra de título. Para ajustar su tamaño haga clic en la esquina del componente y arrastre el puntero del ratón. También puede hacer doble clic en la esquema y MapForce ajustará el tamaño del componente automáticamente.

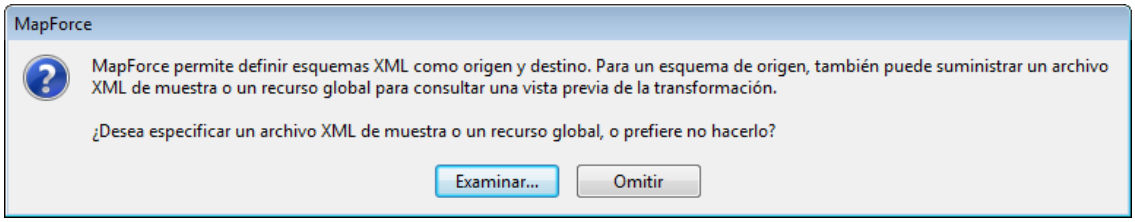
El nodo de nivel superior representa el nombre del archivo (en este caso concreto se trata del nombre del archivo XML). Los elementos XML de la estructura se representan con el icono , mientras que los atributos XML se representan con el icono .

Los pequeños triángulos que aparecen en los laterales del componente representan entradas (si están en el lado izquierdo) o salidas (si están en el lado derecho) de la asignación. En MapForce reciben el nombre de *conectores de entrada* y *conectores de salida* respectivamente.

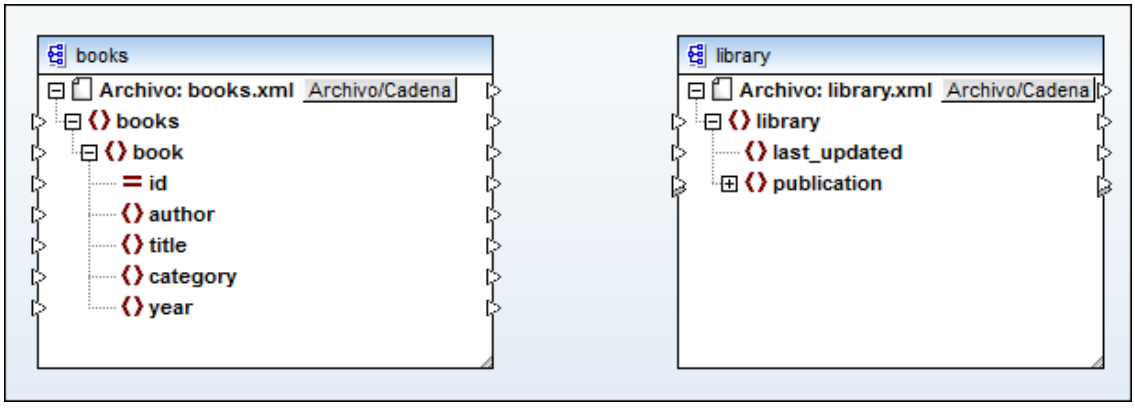
Paso nº3: agregar el esquema XML de destino a la asignación

Para generar el archivo XML de destino usaremos un esquema XML ya existente aunque también podría ser un archivo dado por un colaborador o un esquema XML nuevo creado con XMLSpy, por ejemplo. Si no tiene un esquema para los datos XML, MapForce solicita que genere uno cada vez que se añade a la asignación un archivo XML que no cuenta con la referencia de esquema correspondiente.

Para nuestro ejemplo usamos el esquema XML `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\library.xsd`. Para agregarlo a la asignación hacemos lo mismo que hicimos para añadir el archivo XML de origen (es decir, hacemos clic en el botón **Insertar archivo o esquema XML** de la barra de herramientas). Cuando MapForce solicite un archivo de instancia haga clic en **Omitir**.



En este momento el diseño de asignación tiene este aspecto:

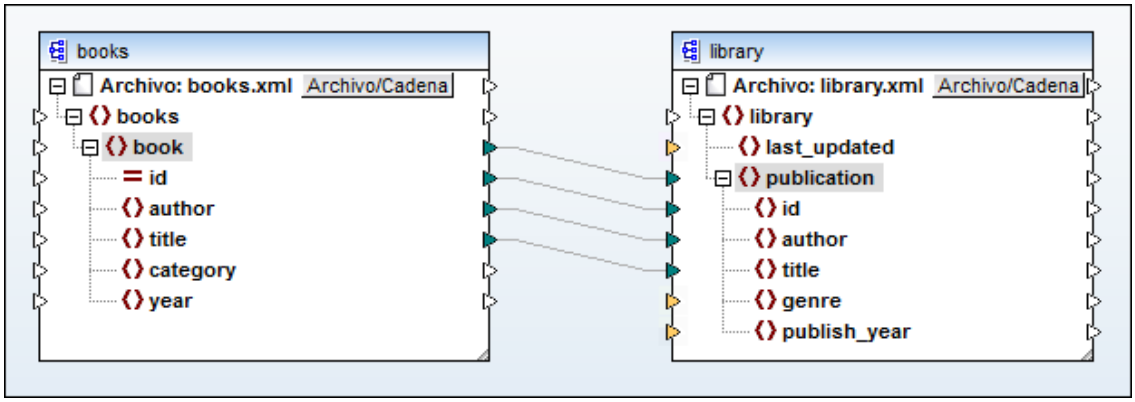


Paso nº4: crear las conexiones

Por cada elemento <book> del archivo XML de origen, queremos crear un elemento <publication> nuevo en el archivo XML de destino. Por tanto, crearemos una conexión de asignación entre el elemento <book> del componente de origen y el elemento <publication> del componente de destino. Para crear la conexión de asignación basta con hacer clic en el conector de salida (el triángulo situado a la derecha del elemento <book>) y arrastrarlo hasta el conector de entrada del elemento <publication> del componente de destino.

A continuación MapForce crea automáticamente una conexión entre todos los elementos secundarios de <book> y los elementos del mismo nombre del componente de destino. Es decir, MapForce crea 4 conexiones simultáneamente. Esta característica recibe el nombre de *conexión automática de secundarios equivalentes* y puede deshabilitarse o configurarse, según corresponda.

© 2018 Altova GmbH



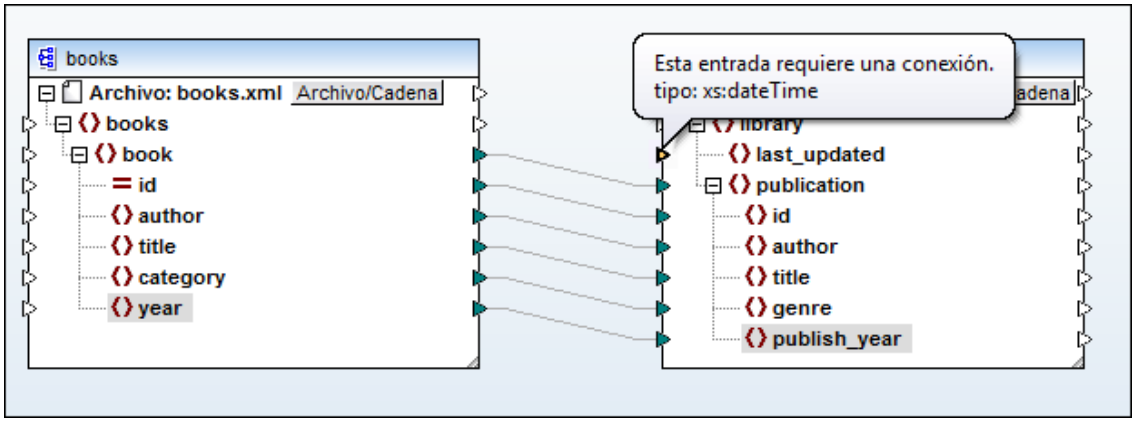
La función de conexión automática de secundarios equivalentes se puede habilitar y deshabilitar de dos formas distintas:

- Haciendo clic en el botón **Alternar la conexión automática de secundarios** de la barra de herramientas.
- Haciendo clic en el comando **Conectar automáticamente los secundarios equivalentes** del menú **Conexión**.

Observe que MapForce resaltó en color naranja algunos conectores de entrada del componente de destino. Esto significa que estos elementos son obligatorios. Para garantizar la validez del archivo XML de destino es necesario indicar valores para estos elementos obligatorios:

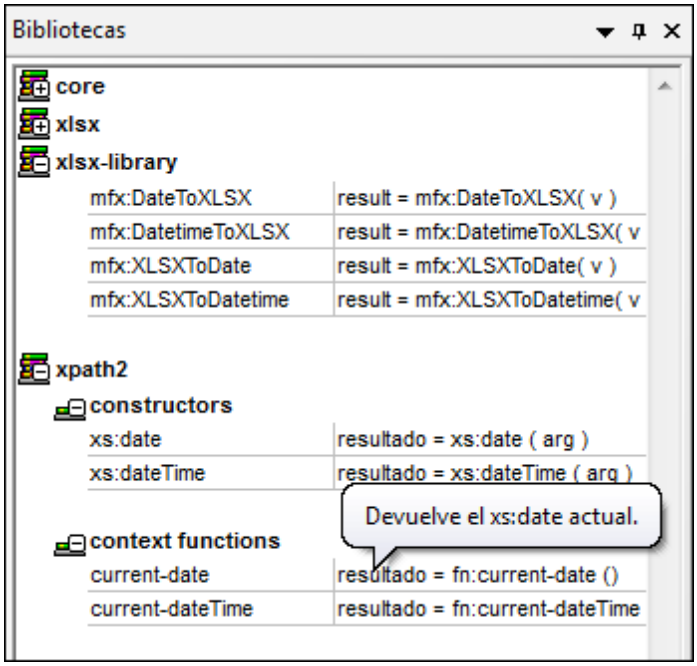
- Conectamos el elemento `<category>` del componente de origen con el elemento `<genre>` del componente de destino.
- Conectamos el elemento `<year>` del componente de origen con el elemento `<publish_year>` del componente de destino.

Por último debemos indicar un valor para el elemento `<last_updated>`. Si pasamos el puntero por encima de su conector de entrada podemos ver que el elemento es de tipo `xs:dateTime`. Recuerde que para poder ver esta información rápida debe habilitar el botón **Mostrar información rápida** de la barra de herramientas.



También puede cambiar la configuración para que el tipo de datos siempre esté visible (haciendo clic en el botón **Mostrar tipos de datos** de la barra de herramientas).

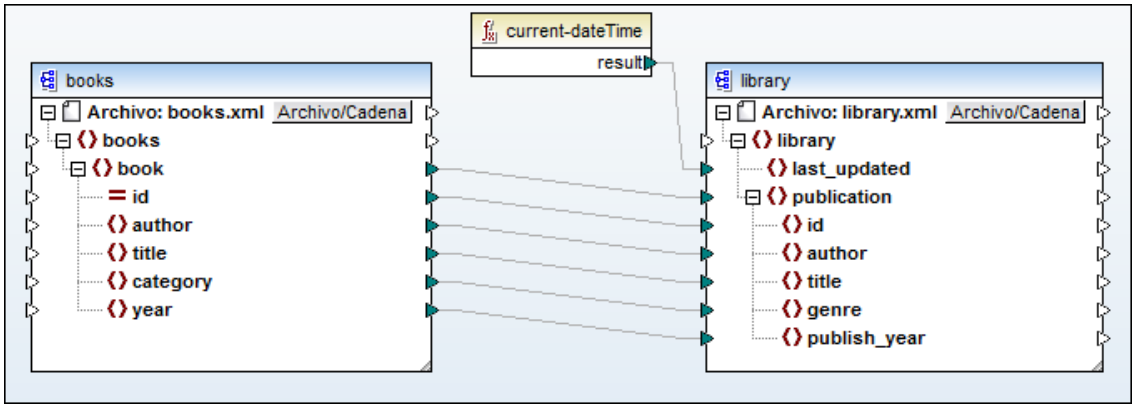
Para obtener la fecha y hora actuales (es decir, el valor `xs:dateTime`) podemos usar una función XSLT de fecha y hora. Para buscar esta función XSLT tecleamos `date` en el cuadro de texto situado en la parte inferior de la ventana Bibliotecas.



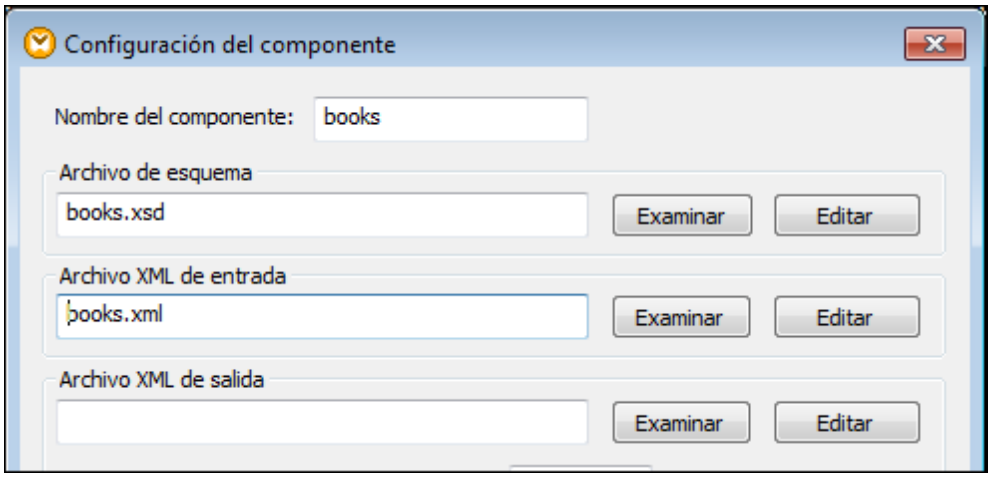
Tal y como puede ver en la imagen anterior, al pasar el puntero por la segunda columna de la función, aparece su descripción. Para que esta descripción aparezca debe estar habilitado el botón **Mostrar información rápida** de la barra de herramientas.

 © 2018 Altova GmbH

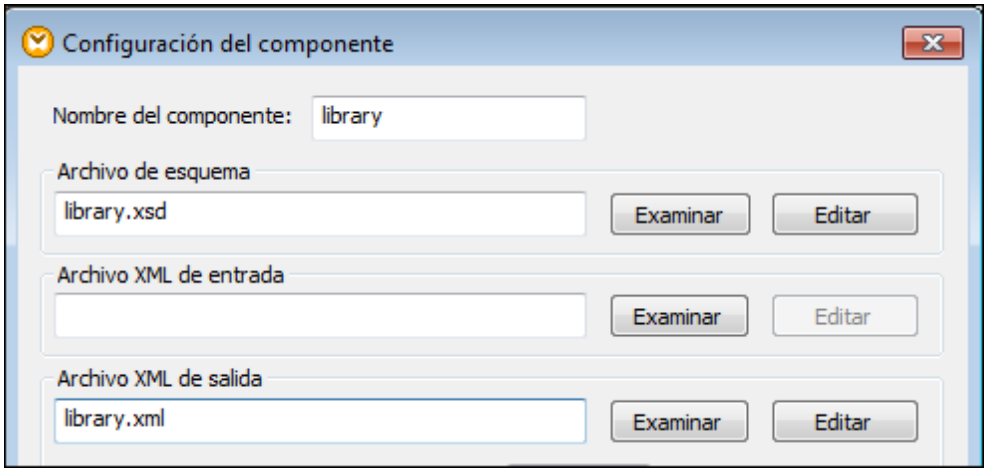
Para agregar la función a la asignación arrástrela hasta el panel de asignación y conecte su salida a la entrada del elemento <last_updated>.



El diseño de asignación ya está terminado y pasa los datos del archivo de instancia **books.xml** (que tiene el esquema **books.xsd**) a un archivo nuevo llamado **library.xml** (que tiene el esquema **library.xsd**). Si hacemos doble clic en el título de un componente, aparece el cuadro de diálogo "Configuración del componente":



Configuración del componente de origen



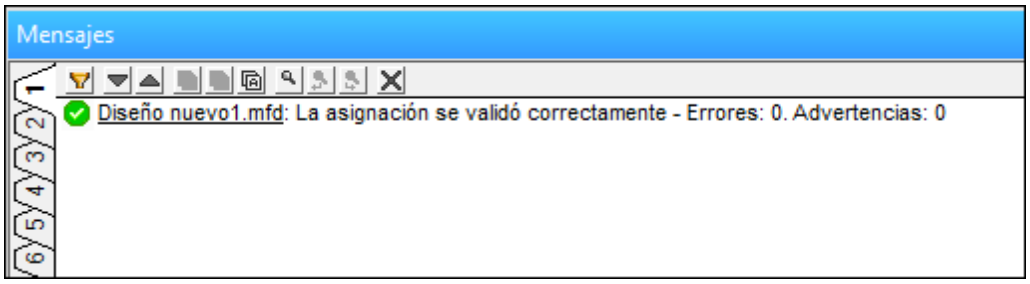
Configuración del componente de destino

Paso nº5: validar y guardar la asignación de datos

El paso de validar la asignación de datos es opcional pero importante porque permite corregir errores y ver advertencias antes de ejecutar la asignación. Hay dos maneras de comprobar si la asignación es válida:

- Haciendo clic en el comando **Validar asignación** del menú **Archivo**.
- Haciendo clic en el botón **Validar** de la barra de herramientas.

La ventana Mensajes muestra los resultados de la validación:



Ventana Mensajes

Llegados a este también puede guardar la asignación en un archivo:

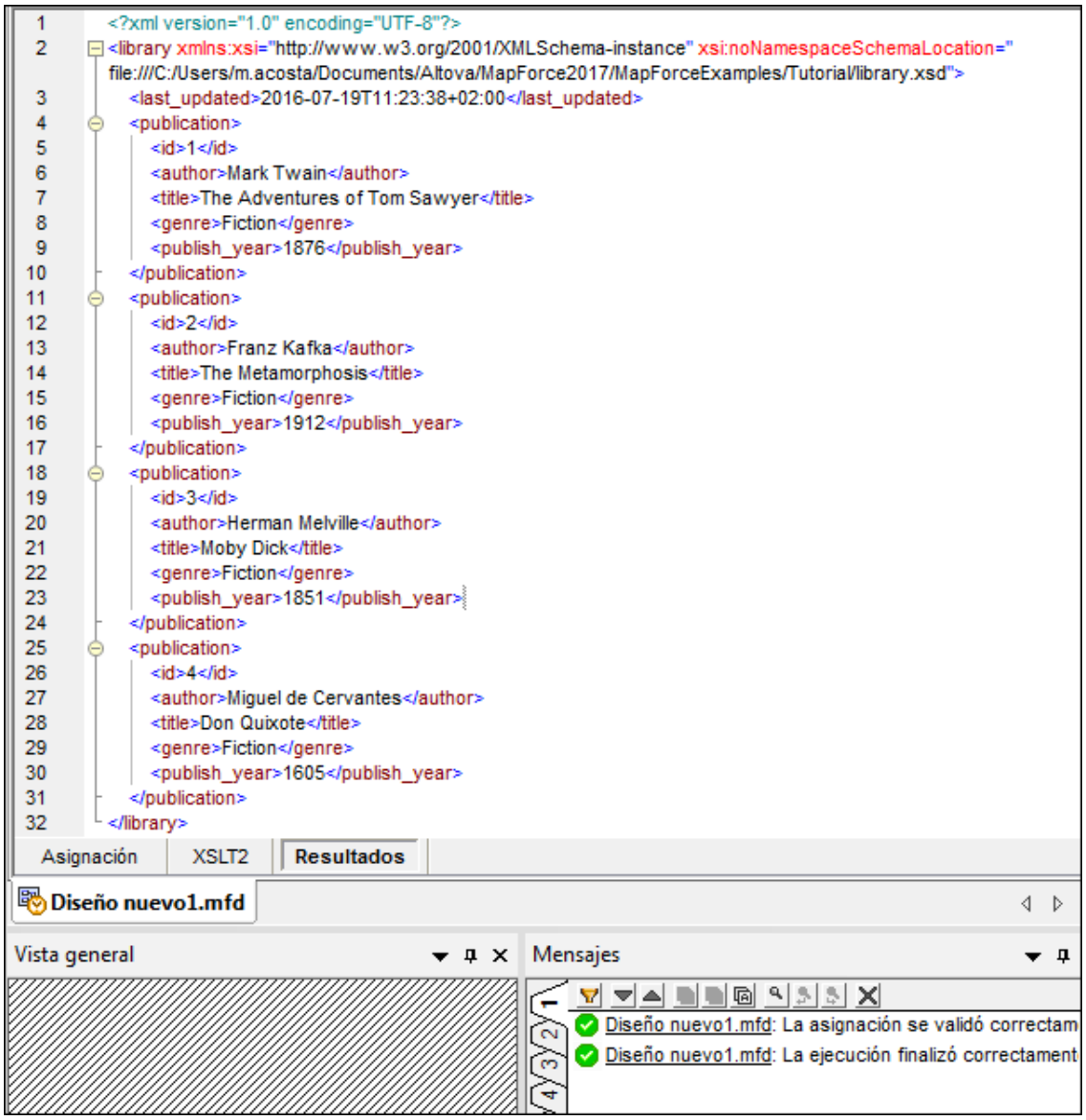
- Haciendo clic en el comando Guardar del menú Archivo.
- Haciendo clic en el botón **Guardar** de la barra de herramientas.

Este diseño de asignación está en esta ruta de acceso: `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\\BooksToLibrary.mfd`. Es decir, a partir de ahora puede continuar trabajando con el archivo de asignación que creó siguiendo las instrucciones del tutorial o continuar con el archivo **BooksToLibrary.mfd**.

 © 2018 Altova GmbH

Paso nº6: vista previa del resultado de la asignación

MapForce puede ofrecer una vista previa de lo resultados de la asignación de datos. Para obtener esta vista previa basta con hacer clic en el botón **Resultados** situado en la parte inferior del panel de asignación. MapForce ejecuta la transformación de datos y muestra los resultados en el panel *Resultados*.



Panel Resultados

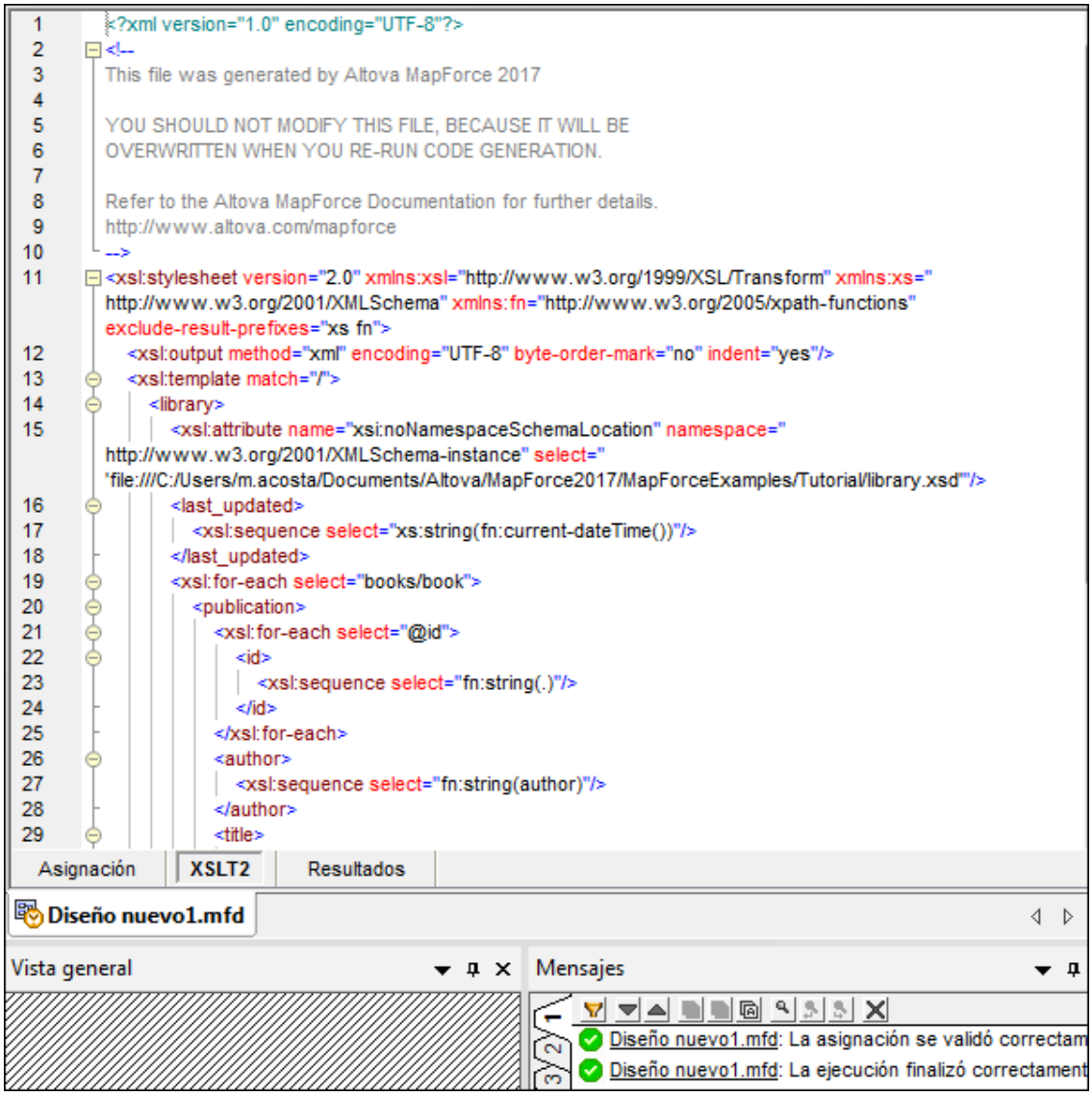
Ahora puede ver en MapForce los resultados de la transformación.

La vista previa del panel *Resultados* no se guarda en disco, sino que MapForce crea archivos de resultados temporales. Para guardar en disco el archivo que muestra el panel *Resultados*, seleccione el comando **Resultados Guardar el archivo de salida** o haga clic en el botón **Guardar resultado generado**  de la barra de herramientas.

Si quiere que MapForce escriba los resultados en archivos finales directamente, haga clic en **Herramientas | Opciones | Generales** y marque la casilla *Escribir directamente en archivos de salida finales*. Sin embargo, no recomendamos marcar esta casilla mientras realiza el tutorial para evitar sobrescribir datos en los archivos del tutorial originales.

También puede crear una vista previa del código XSLT generado que realiza la transformación. Para ello basta con hacer clic en el botón **XSLT2** situado en la parte inferior del panel de asignación.

© 2018 Altova GmbH



Panel XSLT2

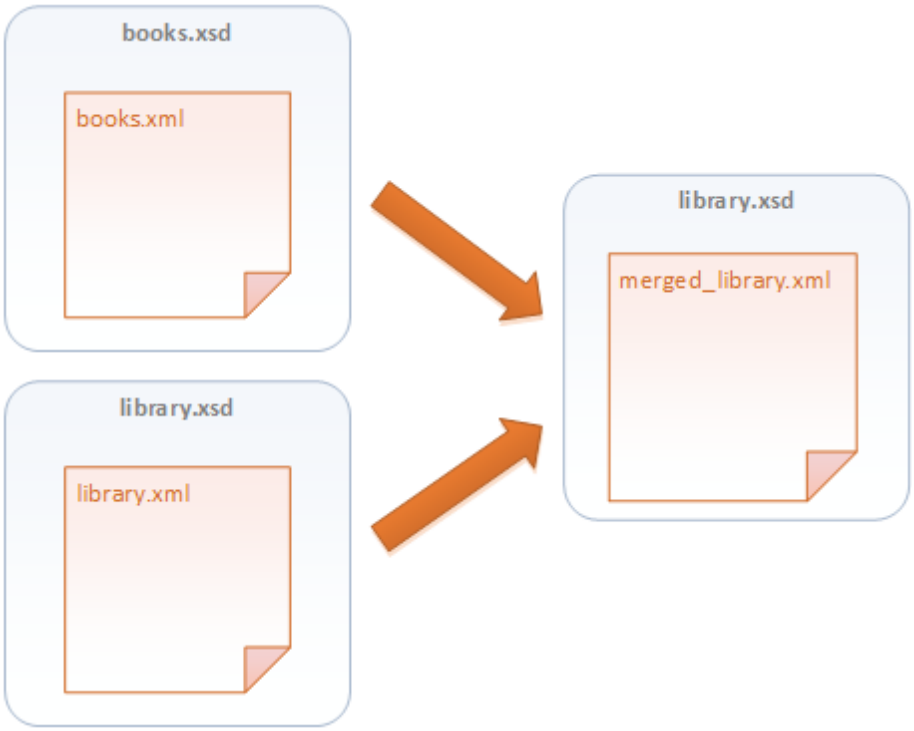
Para generar y guardar el código XSLT2 en un archivo seleccione el comando **Archivo | Generar código en | XSLT 2.0**. Cuando se le solicite, seleccione la carpeta donde se debe guardar el código generado. Una vez generado el código, la carpeta de destino incluirá estos dos archivos:

1. Un archivo de transformación XSLT que se llama como el esquema de destino (p. ej. **MappingMaptolibrary.xslt**).
2. Un archivo **DoTransform.bat**, que sirve para ejecutar la transformación XSLT con RaptorXML Server (consulte https://www.altova.com/es/raptorxml.html para obtener más información).

3.2 Asignar varios orígenes de datos a un destino

En el tutorial anterior pasamos datos de un archivo de origen (**books.xml**) a un archivo de destino (**library.xml**). El archivo de destino no existía antes de ejecutar la asignación, es decir, se generó al transformar la asignación. Imaginemos ahora que ya tuviéramos un archivo llamado **library.xml** y que deseáramos combinar sus datos con los datos convertidos de **books.xml**.

En este tutorial el objetivo es diseñar una asignación de datos que genere un archivo llamado **merged_library.xml**. El archivo de resultados generado incluirá los datos de ambos archivos **books.xml** y **library.xml**. Recuerde que los archivos utilizados como fuente (**books.xml** y **library.xml**) tienen esquemas diferentes. Si los archivos fuente tuvieran el mismo esquema, también podría combinar sus datos usando una técnica diferente (consulte el tutorial *Procesar y generar archivos de forma dinámica*).



Modelo abstracto de la transformación de datos

Para conseguir nuestro objetivo debemos seguir estos pasos:

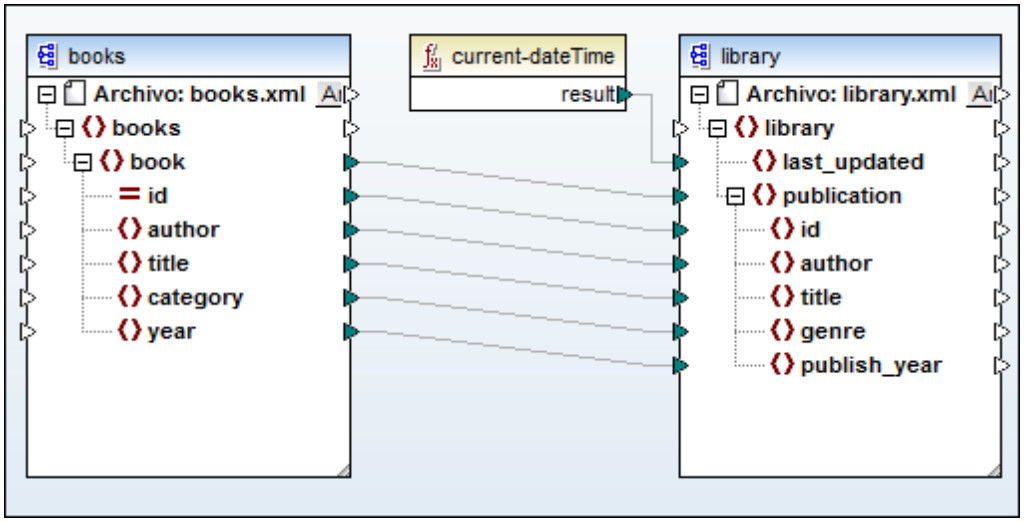
Paso nº1: preparar el archivo de diseño de asignación

Para este tutorial usamos como punto de partida la asignación **BooksToLibrary.mfd** de la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`. Esta asignación es la que diseñamos en el primer tutorial (*Pasar datos XML a un esquema nuevo*). Para empezar abrimos el archivo **BooksToLibrary.mfd** en MapForce y lo guardamos con otro nombre.

Asegúrese de guardar la asignación nueva en la carpeta `<Documentos>\Altova`

© 2018 Altova GmbH

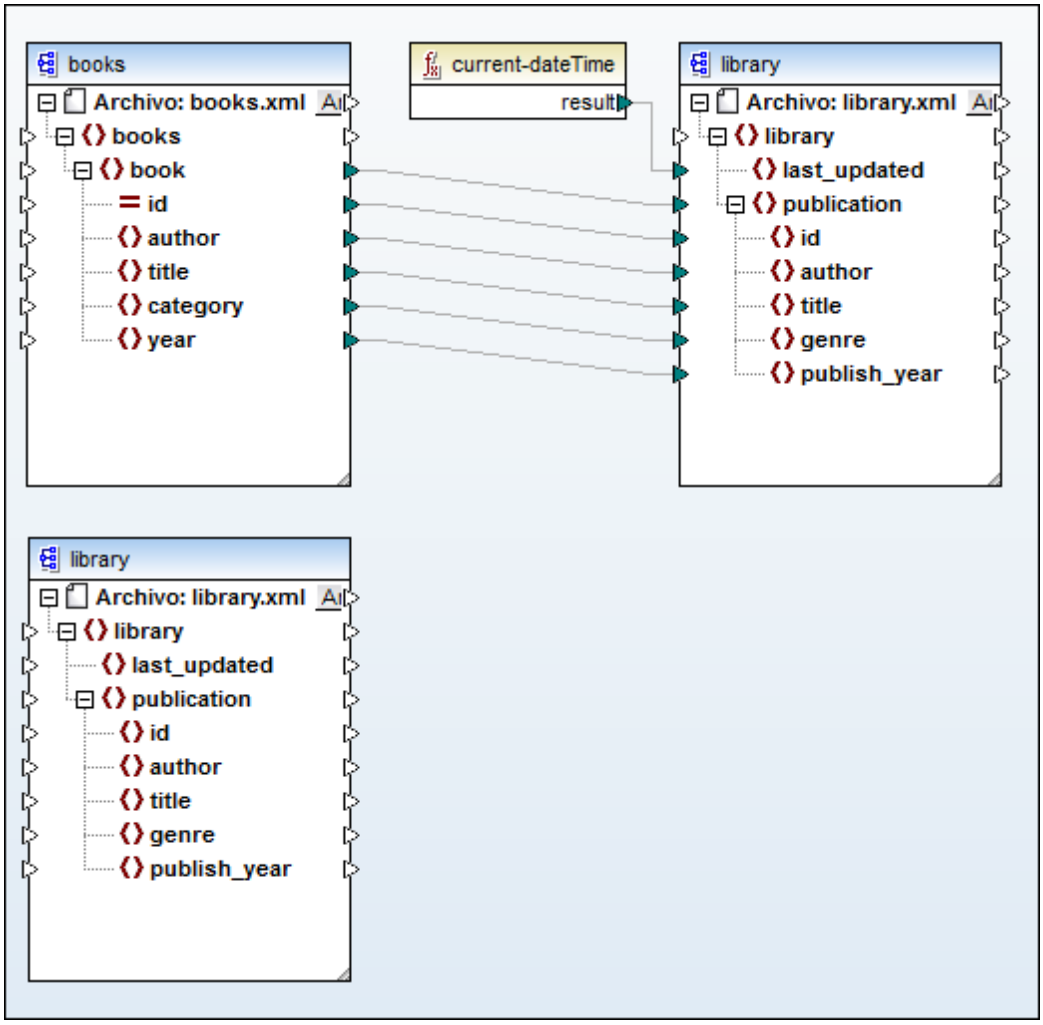
`\MapForce2018\MapForceExamples\Tutorial\` porque hace referencia a varios archivos que están en esa carpeta.



BooksToLibrary.mfd

Paso nº2: crear otro componente de origen

Primero seleccionamos el componente de destino, lo copiamos (**Ctrl+C**) y lo pegamos (**Ctrl+V**) en el panel de asignación. Haga clic en el título de este nuevo componente y colóquelo debajo del componente **books**.



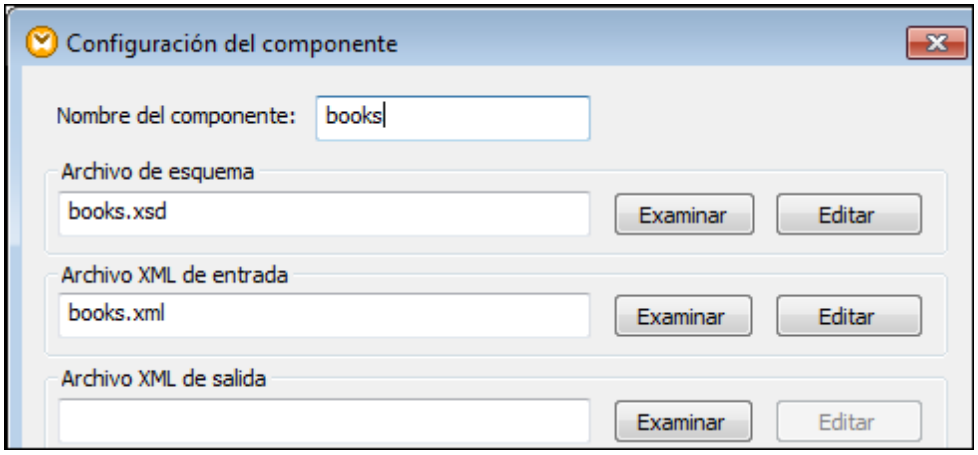
Ahora la asignación tiene dos componentes de origen (**books** y **library**) y un componente de destino (**library**).

Los componentes de asignación se pueden mover en cualquier dirección (izquierda, derecha, abajo, arriba). Sin embargo, las tareas de asignación serán más fáciles si coloca los componentes de origen a la izquierda de los componentes de destino. Esta es la convención utilizada en esta documentación y en los archivos de ejemplo que vienen con MapForce.

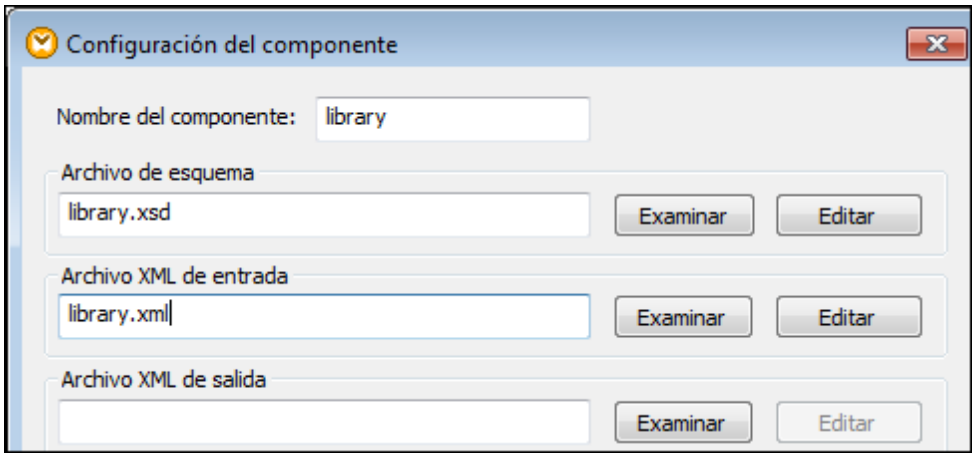
Paso nº3: verificar y configurar los archivos de entrada y salida

En el paso anterior tomamos el componente de destino y lo pegamos como componente de origen. Esto significa que el nuevo componente de origen hereda la configuración del componente de destino. Ahora debemos modificar el nombre de los archivos de instancia de entrada y salida para que sean correctos. Haga doble clic en el título de cada componente y, en el cuadro de diálogo "Configuración del componente", verifique y corrija el nombre de los archivos de entrada y salida de cada uno de ellos.

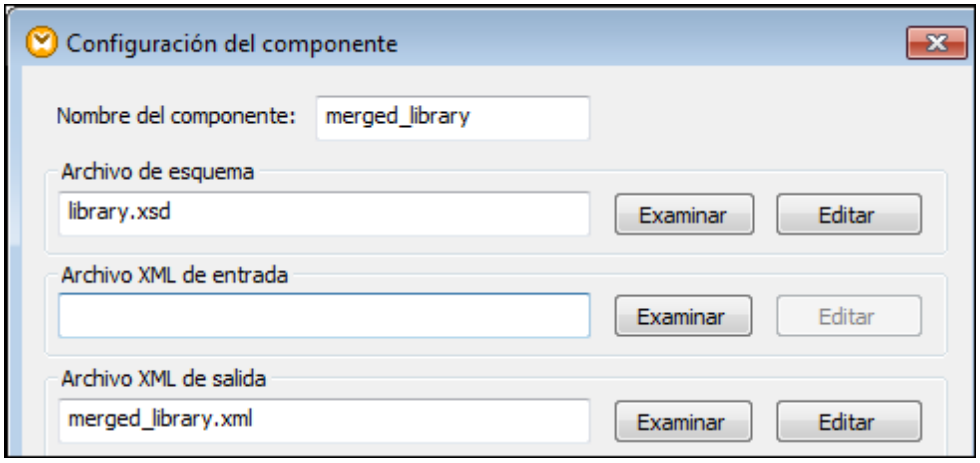
© 2018 Altova GmbH



*Configuración del primer componente de origen (**books**)*



*Configuración del segundo componente de origen (**library**)*



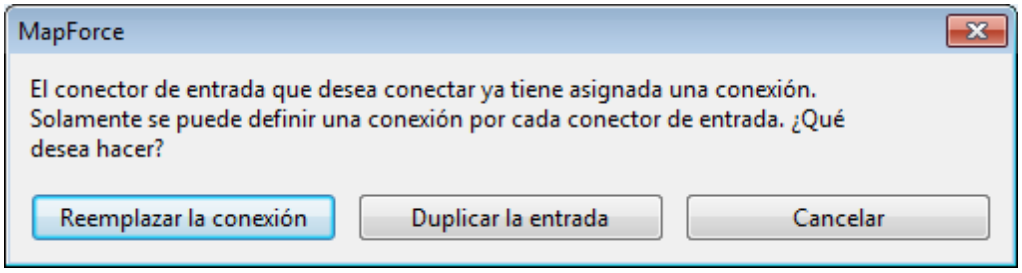
*Configuración del componente de destino (**merged_library**)*

Como puede ver en las imágenes el primer componente de origen lee datos del archivo **books.xml**, mientras que el segundo lee datos de **library.xml**. Por su parte, el componente de destino genera datos en un archivo llamado **merged_library.xml**.

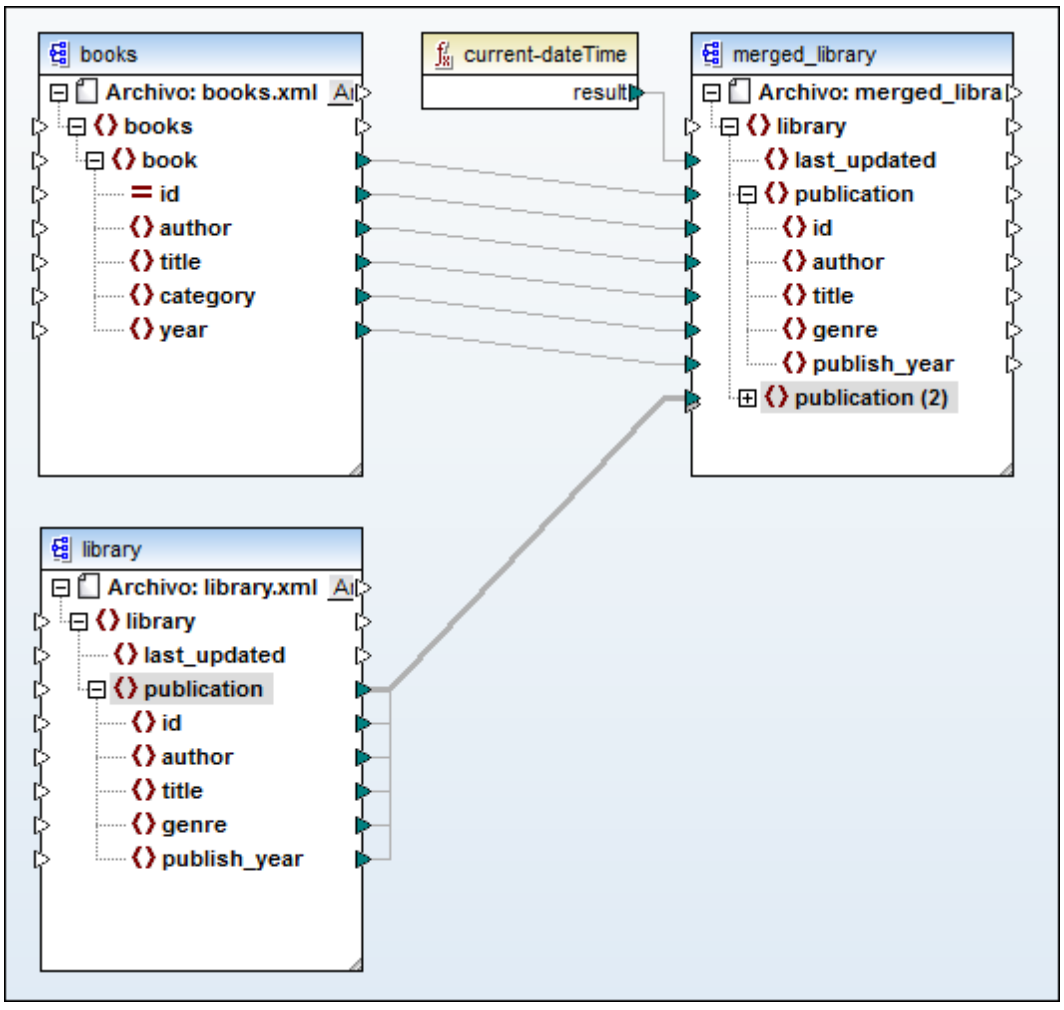
Paso nº4: crear las conexiones

El siguiente paso es dar la orden de escribir datos del segundo componente de origen en el componente de destino. Haga clic en el conector de salida del elemento `publication` del componente de origen **library** y arrástrelo hasta el conector de entrada del elemento `publication` del componente de destino **library**. Como el conector de entrada del destino ya está conectado, aparece esta notificación:

© 2018 Altova GmbH



En este ejemplo concreto no queremos reemplazar la conexión porque nuestro objetivo es asignar datos procedentes de dos fuentes distintas. Por tanto, en este ejemplo elegimos la opción **Duplicar la entrada**. El resultado es que el componente de destino también aceptará datos del nuevo componente de origen. Este es el aspecto que tiene ahora el diseño de asignación:



Observe que el elemento `publication` del componente de destino está duplicado. El nodo `publication(2)` nuevo aceptará datos del componente de origen **library**. Y, lo que es más importante, aunque el nombre de este nodo aparezca como `publication(2)` en la asignación,

en el archivo XML resultante se llamará `publication`.

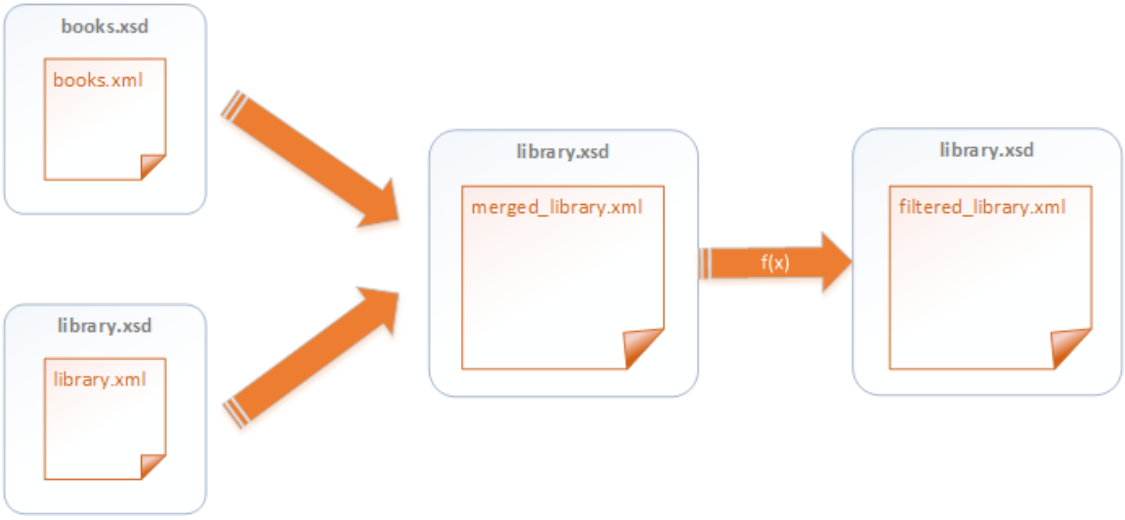
Ahora puede hacer clic en el botón **Resultados** situado en la parte inferior del panel de asignación para ver el resultado. Observe que tanto los datos de **library.xml** como los de **books.xml** se incluyeron en el archivo de salida **merged_library.xml**.

© 2018 Altova GmbH

3.3 Trabajar con varios esquemas de destino

En el tutorial anterior aprendimos a asignar datos de varios esquemas de origen a un solo esquema de destino y creamos un archivo llamado **merged_library.xml** donde se almacenan los registros de libros procedentes de dos fuentes. Ahora imaginemos que alguien de otro departamento solicita un subconjunto de datos de este archivo XML. Concretamente solicita un archivo XML que contenga solamente los libros publicados después de 1900.

Si quiere puede modificar la asignación **MultipleSourcesToOneTarget.mfd** que viene con MapForce para poder generar tanto la biblioteca XML completa como la biblioteca con los datos filtrados.



Modelo abstracto de la transformación de datos

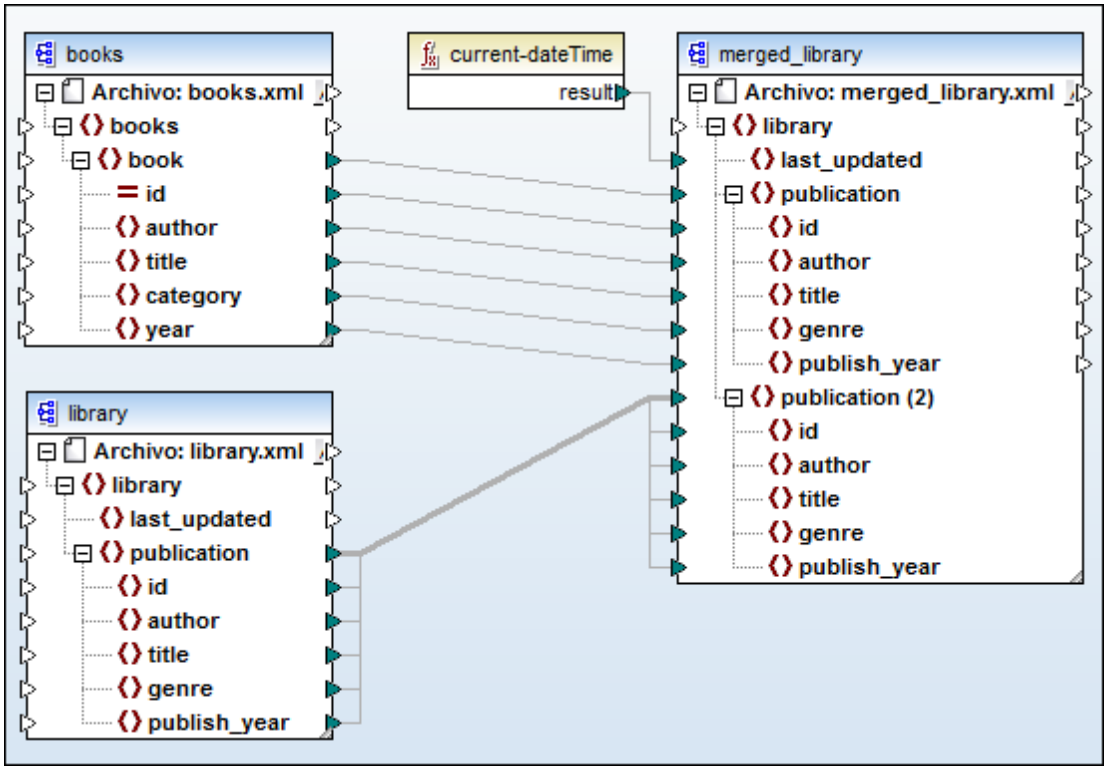
En este diagrama puede ver que primero se combinan lo datos de dos esquemas diferentes (**books.xsd** y **library.xsd**) y se guardan en un solo archivo XML llamado **merged_library.xml**. Después se transforman los datos con ayuda de una función de filtrado y se envían al siguiente componente, que se encarga de generar un archivo XML llamado **filtered_library.xml**. El componente intermedio hace tanto de destino como de origen. En MapForce esta técnica recibe el nombre de *encadenar asignaciones* y también se utiliza en este tutorial.

En este tutorial nuestro objetivo es poder generar tanto el archivo **merged_library.xml** como el archivo **filtered_library.xml** siempre que se quiera. Para conseguir nuestro objetivo es necesario seguir estos pasos:

Paso nº1: preparar el archivo de diseño de asignación

Como partida utilizamos la asignación **MultipleSourcesToOneTarget.mfd** de la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`. Esta asignación es la que diseñamos en el tutorial *Asignar varios orígenes de datos a un destino*. Para empezar, abra el archivo **MultipleSourcesToOneTarget.mfd** en MapForce y guárdelo con otro nombre.

Asegúrese de guardar la asignación nueva en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\` porque hace referencia a varios archivos que están en esa carpeta.

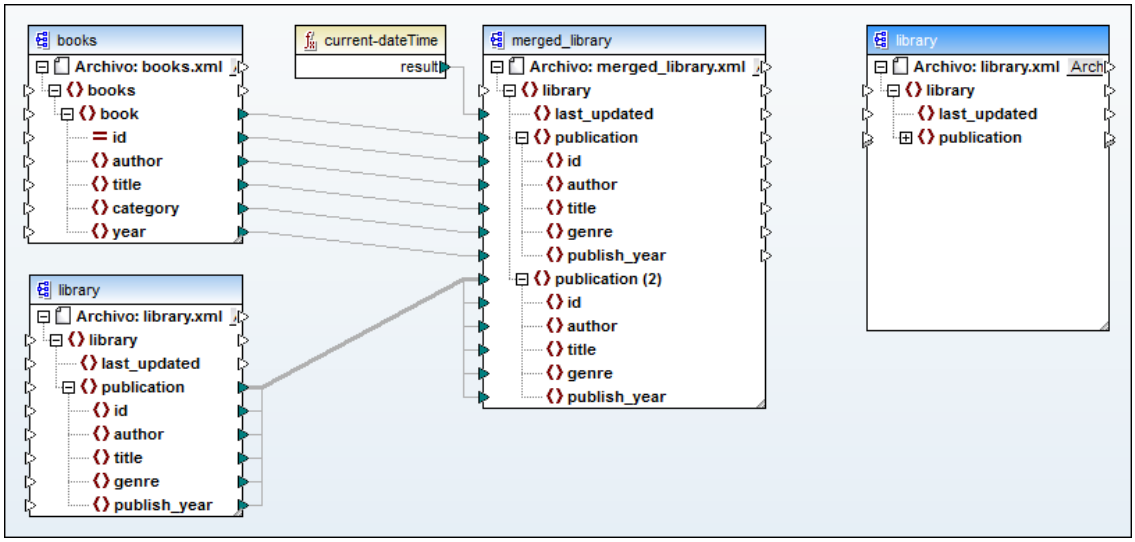


MultipleSourcesToOneTarget.mfd

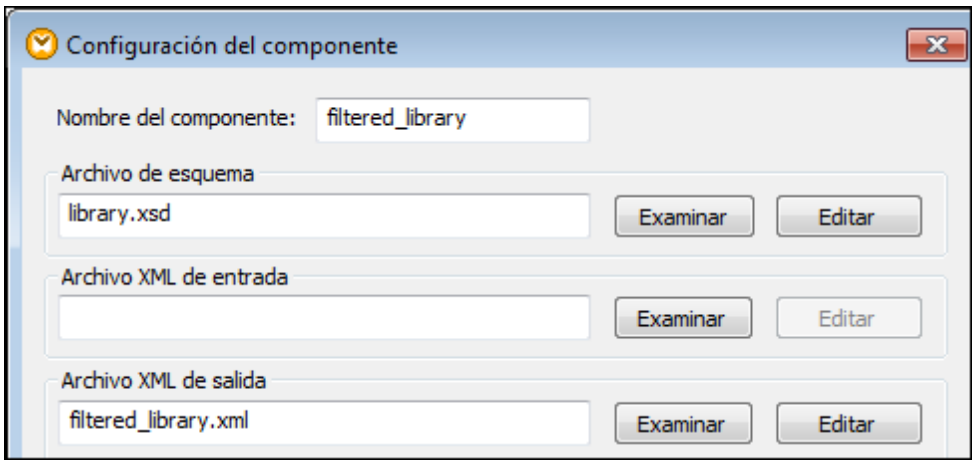
Paso nº2: agregar y configurar el segundo componente de destino

Para agregar otro componente de destino haga clic en el botón **Insertar archivo o esquema XML** de la barra de herramientas y abra el archivo **library.xsd** situado en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`. Haga clic en Omitir cuando la aplicación solicite un archivo de instancia de muestra. La asignación tendrá este aspecto:

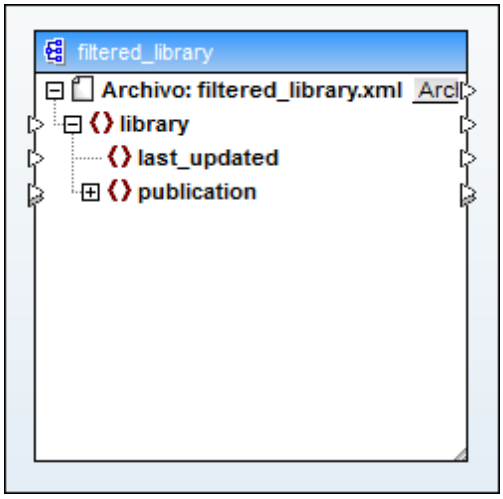
© 2018 Altova GmbH



Como se aprecia en la imagen, ahora la asignación tiene dos componentes de origen (**books** y **library**) y dos componentes de destino. Para distinguirlos bien pondremos el nombre **filtered_library** al segundo componente de destino y definiremos el nombre del archivo XML que debe generar. Esto se consigue haciendo doble clic en el título del segundo componente de destino y editando su configuración:

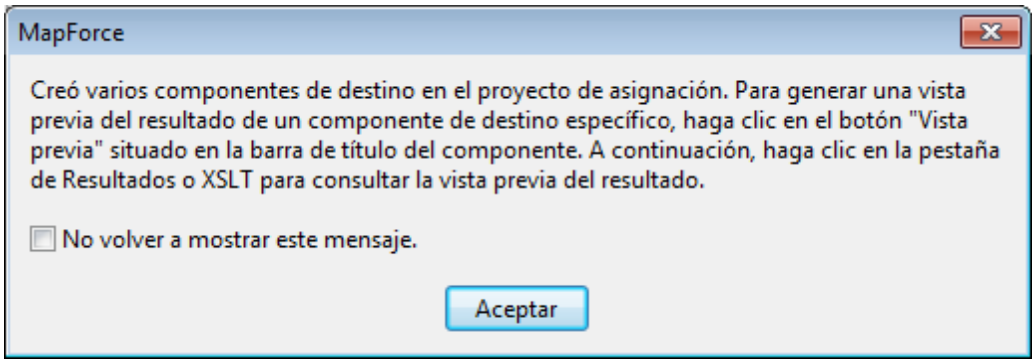


Observe que ahora el componente nuevo se llama **filtered_library** y que su archivo XML de salida se llama **filtered_library.xml**.



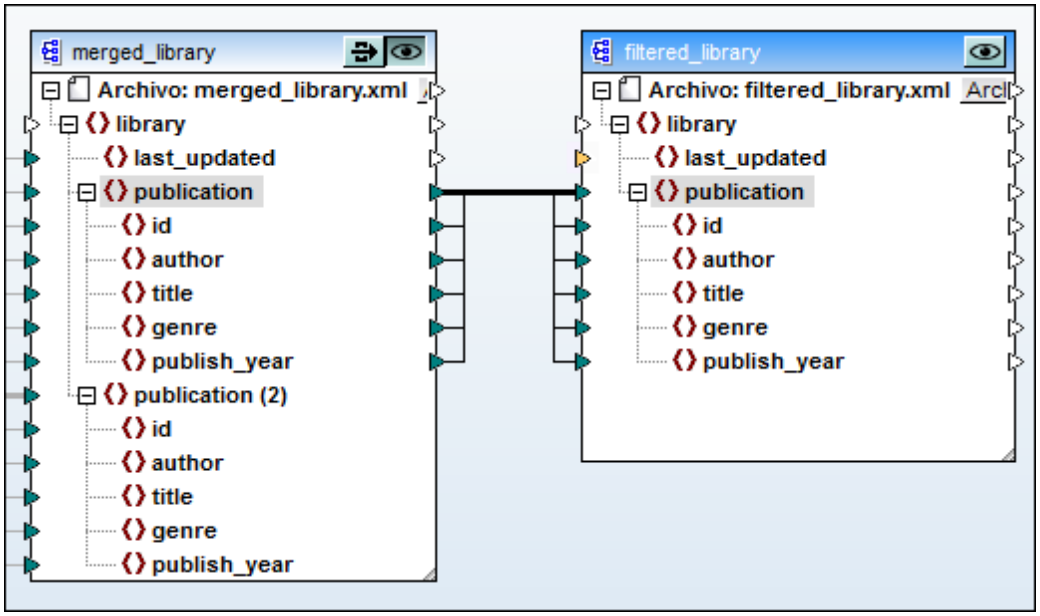
Paso nº3: crear las conexiones

Cree una conexión entre el elemento **publication** del componente **merged_library** y el elemento **publication** del componente **filtered_library**. Aparece una notificación:



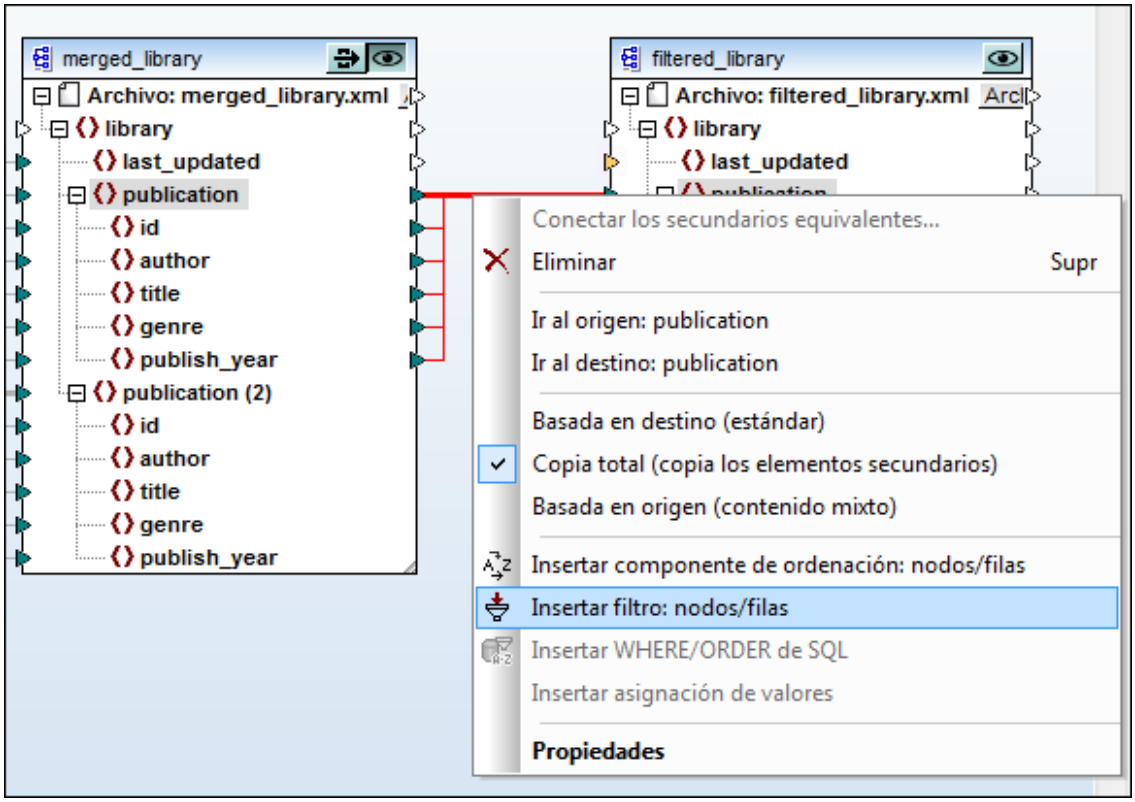
Haga clic en **Aceptar**. Observe que en los componentes de destino ahora aparecen nuevos botones: **Vista previa** ( ) y **Paso a través** ( ). En los siguientes pasos del tutorial aprenderemos a usar estos botones.

© 2018 Altova GmbH

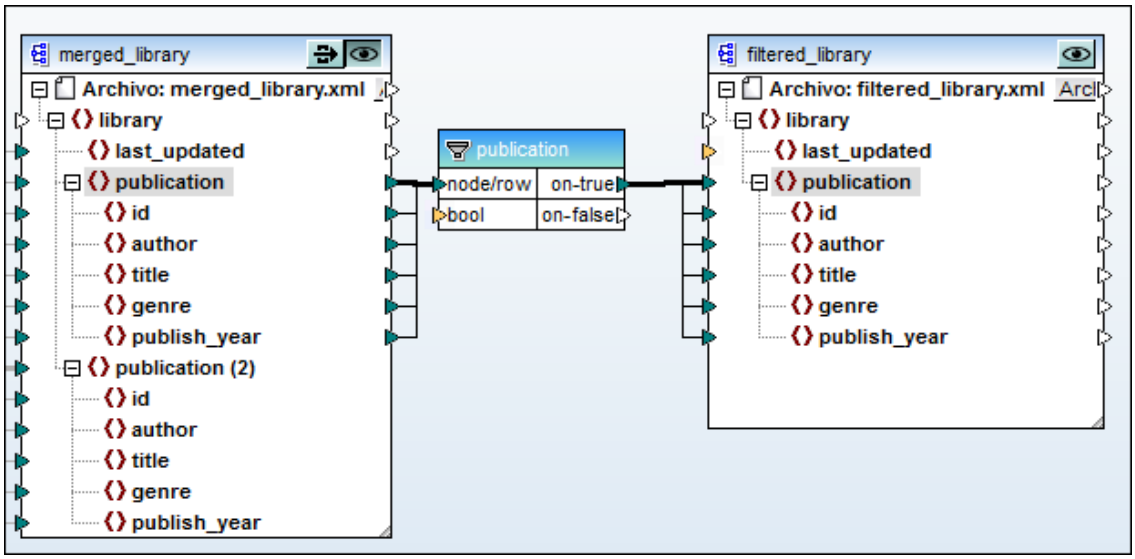


Paso nº4: filtrar los datos

Para filtrar los datos antes de pasarlos a **filtered_library** usaremos un componente de filtrado. Para agregar un componente de filtrado haga clic con el botón derecho en la conexión que existe entre **merged_library** y **filtered_library** y seleccione el comando **Insertar filtro: nodos/filas** del menú contextual.

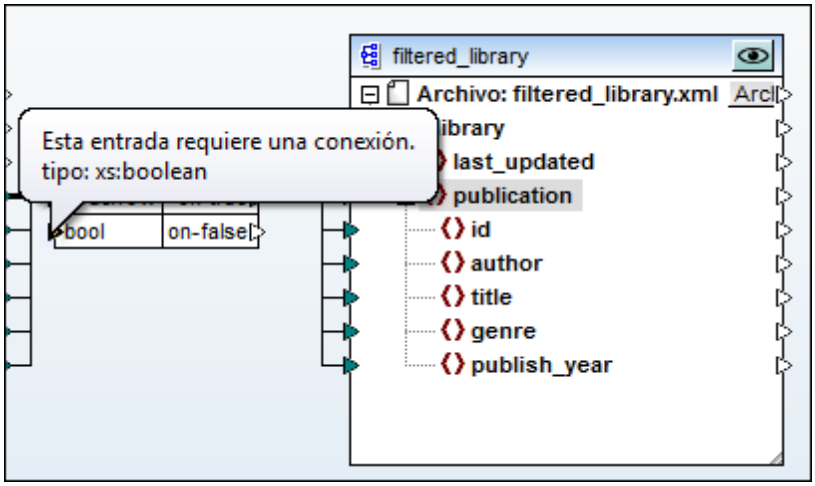


El componente de filtrado aparece ahora en la asignación.



Como se aprecia en la imagen, el conector de entrada bool es de color naranja, lo cual indica que exige una entrada. Si pasa el puntero por el conector podrá ver que exige una entrada de tipo `xs:boolean`. Recuerde que para poder ver la información rápida es necesario tener habilitado el botón **Mostrar información rápida** de la barra de herramientas.

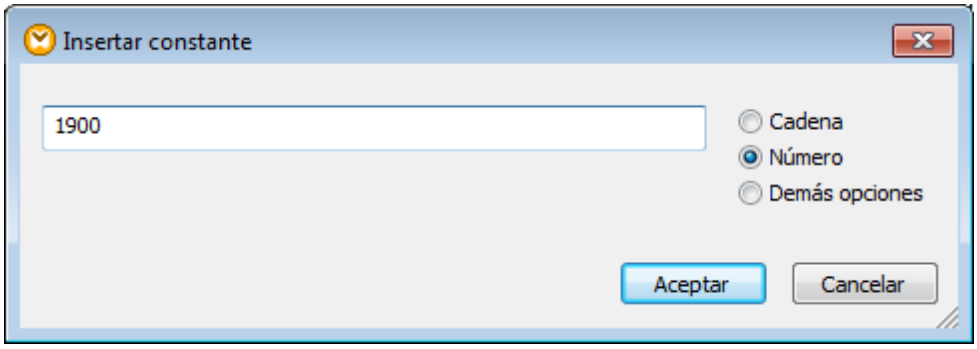
 © 2018 Altova GmbH



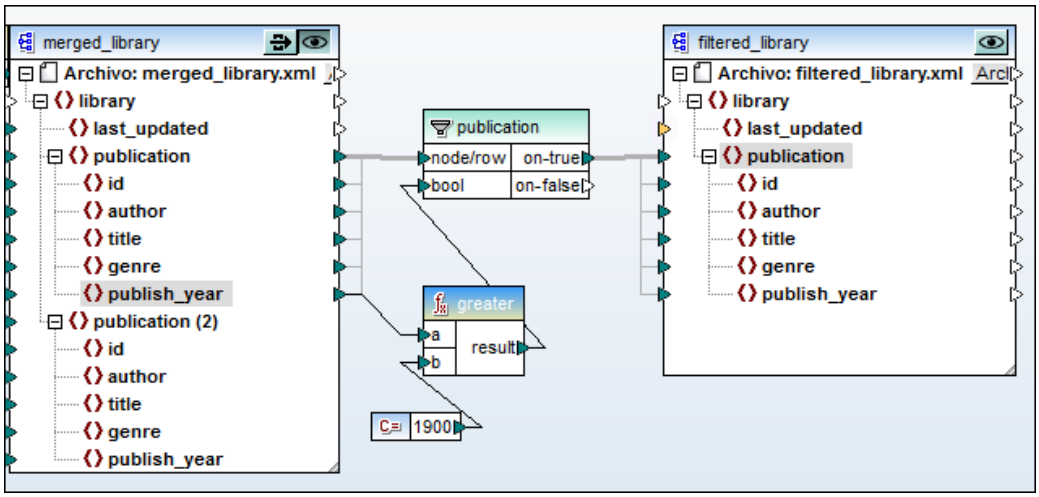
El componente de filtrado exige una condición que devuelva el valor `true` o `false`. Cuando la condición booleana devuelva `true`, los datos de la secuencia **publication** actual se copiarán en el componente de destino. Sin embargo, cuando devuelva `false`, no se copiarán datos.

En este tutorial nuestro objetivo es filtrar todos los libros que se publicaron después de 1900. Para crear la condición:

1. Añada una constante de tipo numérico que tenga el valor "1900" (haciendo clic en **Insertar | Constante**). Elija el tipo *Número*.



2. En la ventana Bibliotecas busque la función **`greater`** y arrástrela hasta el área de asignación.
3. Cree conexiones con la función **`greater`** tal y como muestra la siguiente imagen. Estas conexiones dan a MapForce esta instrucción: "cuando `publish_year` sea mayor que 1900, copiar el elemento de origen `publication` actual en el elemento de destino `publication`".



Paso nº5: consultar la vista previa de resultados y guardar los resultados de cada componente de destino

Ahora ya puede consultar la vista previa de resultados y guardar los resultados de ambos componentes de destino. Cuando en la misma asignación hay varios componentes de destino, puede elegir cuál de los dos se usa para generar la vista previa de resultados. Esto se hace con el botón **Vista previa** de los componentes. Si el botón **Vista previa** de un componente está pulsado, significa que ese componente concreto se usará para generar la vista previa de resultados. Este botón no puede estar pulsado en más de un componente a la vez.

Por tanto, cuando quiera consultar y guardar los resultados de **merged_library** (es decir, del componente intermedio), siga estos pasos:

1. Haga clic en el botón **Vista previa** del componente **merged_library**.
2. Haga clic en el botón **Resultados** situado en la parte inferior del panel de asignación.
3. En el menú **Resultados** haga clic en el comando **Guardar el archivo de salida** si quiere guardar los resultados en un archivo.

Cuando quiera cuando quiera consultar y guardar los resultados del componente **filtered_library**:

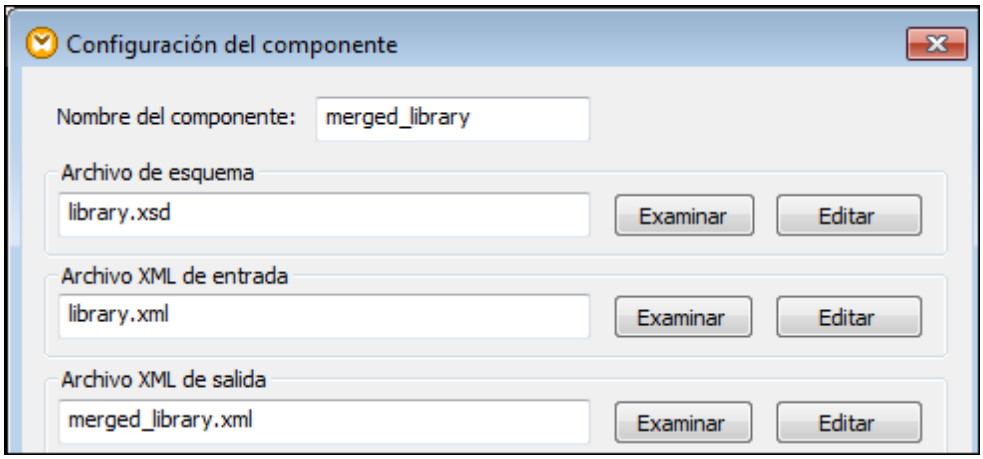
1. Haga clic en el botón **Paso a través** del componente **merged_lilbrary**.
2. Haga clic en el botón **Vista previa** del componente **filtered_library**.
3. Haga clic en el botón **Resultados** situado en la parte inferior del panel de asignación.
4. En el menú **Resultados** haga clic en el comando **Guardar el archivo de salida** si quiere guardar los resultados en un archivo.

El estado del botón **Paso a través** desempeña un papel importante en las asignaciones que tienen varios componentes de destino. Si este botón está pulsado, MapForce deja pasar los datos por el componente intermedio para que pueda consultar la vista previa de los resultados de toda la asignación.

Si el botón **Paso a través** no está pulsado, la vista previa incluirá solamente los resultados de la asignación de datos entre **merged_library** y **filtered_library**. En el ejemplo que nos

© 2018 Altova GmbH

ocupa esto daría lugar a un error porque el componente intermedio no tiene un archivo XML de entrada válido para leer datos. Para resolver este problema haga doble clic en el título del componente y especifique un archivo XML de entrada válido:

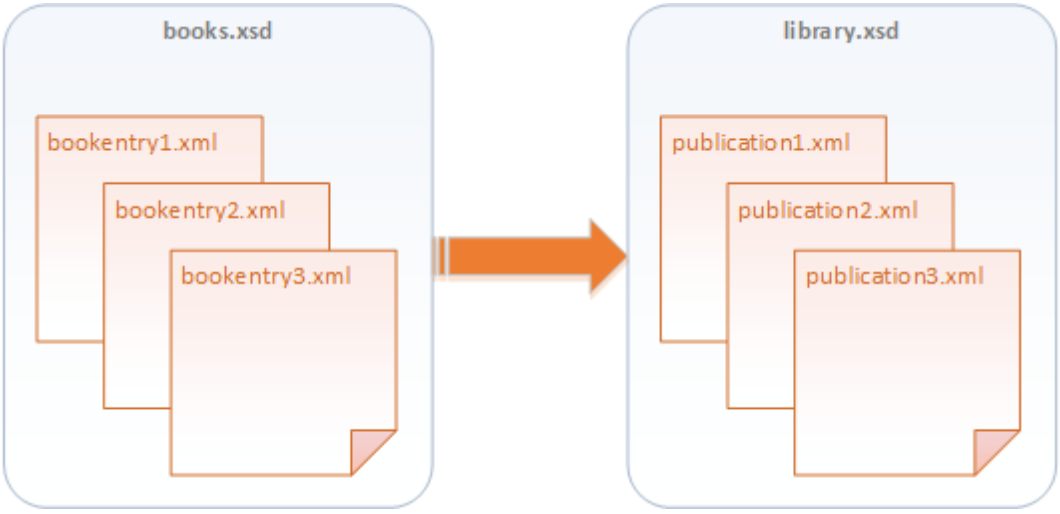


La asignación creada en este tutorial tiene varios componentes de salida y podemos consultar y guardar los resultados de cada uno de ellos. Para más información sobre el funcionamiento de los componentes de paso a través consulte el apartado *Asignaciones encadenadas*.

3.4 Procesar y generar archivos de forma dinámica

Este tutorial explica cómo leer datos de varios archivos XML de origen y escribirlos en varios archivos de destino en la misma transformación. Para comprender esta característica crearemos una asignación cuyos objetivos serán:

1. Leer datos de varios archivos XML ubicados en el mismo directorio.
2. Pasar cada archivo XML a un esquema XML nuevo.
3. Por cada archivo XML de origen generar un archivo XML de destino nuevo bajo un esquema nuevo.
4. Eliminar la declaración XML y la declaración de espacio de nombres de los archivos generados.



Modelo abstracto de la transformación de datos

Como ejemplo usaremos tres archivos XML de origen, ubicados en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`, que se llaman **bookentry1.xml**, **bookentry2.xml** y **bookentry3.xml**. Cada uno de ellos contiene los datos de un solo libro.

```
<?xml version="1.0" encoding="UTF-8"?>
<books xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="books.xsd">
   <book id="1">
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <category>Fiction</category>
      <year>1876</year>
   </book>
</books>
```

bookentry1.xml

© 2018 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<books xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="books.xsd">
   <book id="2">
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <category>Fiction</category>
      <year>1912</year>
   </book>
</books>
```

bookentry2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<books xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="books.xsd">
   <book id="3">
      <author>Herman Melville</author>
      <title>Moby Dick</title>
      <category>Fiction</category>
      <year>1851</year>
   </book>
</books>
```

bookentry3.xml

Los tres archivos XML de origen usan el esquema **books.xsd**, ubicado en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`. Para pasar los archivos de origen a otro esquema XML usaremos el esquema **library.xsd** (ubicado en la misma carpeta). Una vez completada la transformación, la asignación generará tres archivos basados en este nuevo esquema (consulte los fragmentos de código que aparecen a continuación). También configuraremos la asignación para que los nombres de los archivos de salida sean **publication1.xml**, **publication2.xml** y **publication3.xml**. Recuerde que debemos eliminar la declaración XML y la declaración de espacio de nombres.

```
<library>
   <publication>
      <id>1</id>
      <author>Mark Twain</author>
      <title>The Adventures of Tom Sawyer</title>
      <genre>Fiction</genre>
      <publish_year>1876</publish_year>
   </publication>
</library>
```

publication1.xml

```
<library>
   <publication>
      <id>2</id>
      <author>Franz Kafka</author>
      <title>The Metamorphosis</title>
      <genre>Fiction</genre>
      <publish_year>1912</publish_year>
   </publication>
</library>
```

publication2.xml

```
<library>
   <publication>
      <id>3</id>
      <author>Herman Melville</author>
      <title>Moby Dick</title>
      <genre>Fiction</genre>
      <publish_year>1851</publish_year>
   </publication>
</library>
```

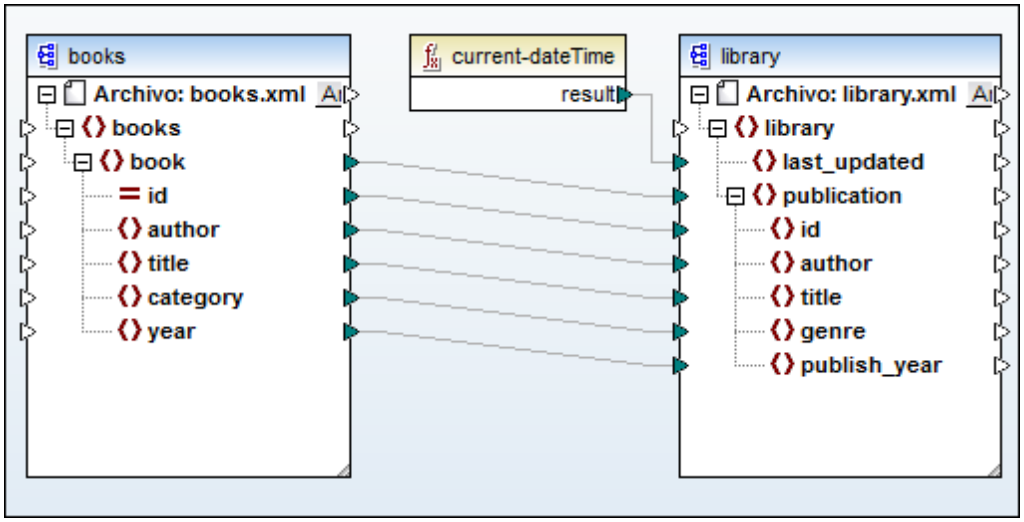
publication3.xml

Paso nº1: preparar el archivo del diseño de asignación

Como punto de partida usamos la asignación **BooksToLibrary.mfd** de la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`. Se trata de la asignación que diseñamos en el tutorial *Pasar datos XML a un esquema nuevo*. Para empezar abra el archivo **BooksToLibrary.mfd** en MapForce y guárdelo en la misma carpeta con otro nombre.

> Asegúrese de guardar la asignación nueva en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\` porque hace referencia a varios archivos que están en esa carpeta.

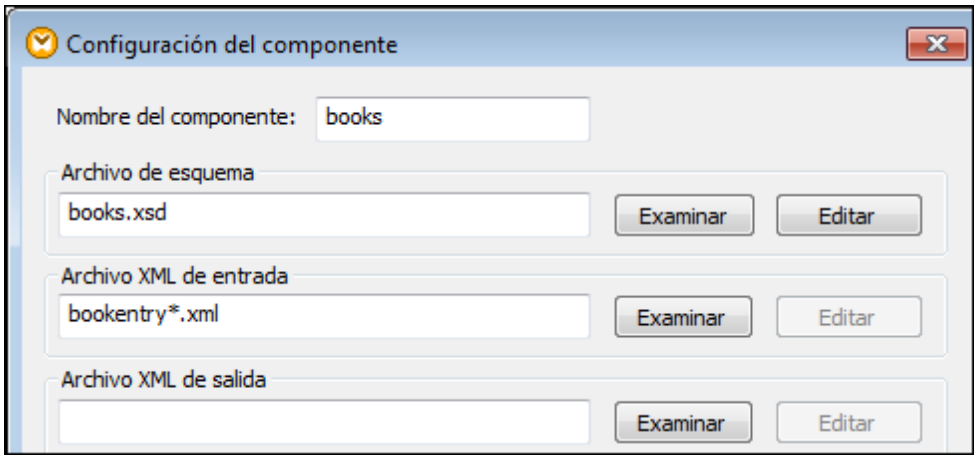
© 2018 Altova GmbH



BooksToLibrary.mfd

Paso nº2: configurar la entrada

Para que MapForce procese varios archivos de instancia XML haga doble clic en el título del componente de origen y escriba **bookentry*.xml** en el campo *Archivo XML de entrada* del cuadro de diálogo "Configuración del componente".



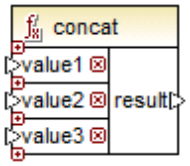
Cuadro de diálogo "Configuración del componente"

El carácter comodín asterisco (*) del nombre de archivo indica que MapForce debe usar todos los archivos que tengan el prefijo **bookentry-** como entrada para la asignación. Como la ruta de acceso es relativa, MapForce buscará todos los archivos **bookentry-** que estén en el directorio del archivo de asignación. De todas maneras, también puede usar el carácter comodín * con rutas de acceso absolutas.

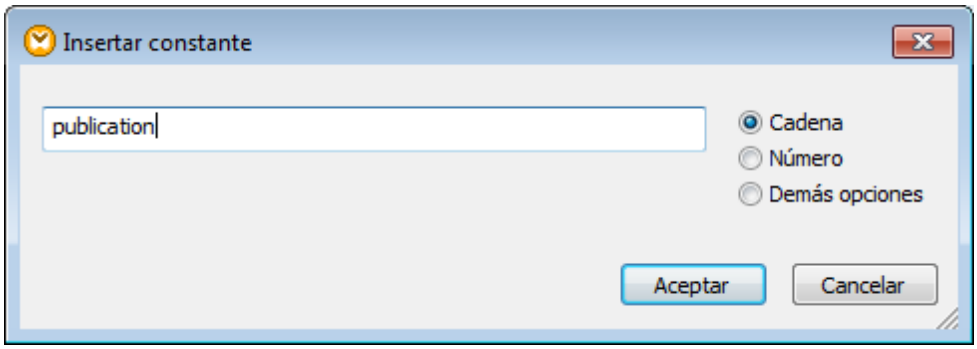
Paso nº3: configurar la salida

Para crear el nombre de archivo de cada archivo de salida usaremos la función **concat**. Esta función concatena (une) todos los valores que recibe como argumento.

1. Busque la función **concat** en la ventana Bibliotecas y arrástrela hasta el área de asignación. Esta función se añade a la asignación con dos parámetros predeterminados pero puede añadirle más parámetros si lo necesita. Haga clic en el símbolo **Agregar parámetro** (⊡) situado dentro de la función y añádale otro parámetro más. El símbolo **Eliminar parámetro** (⊠) sirve para eliminar parámetros.

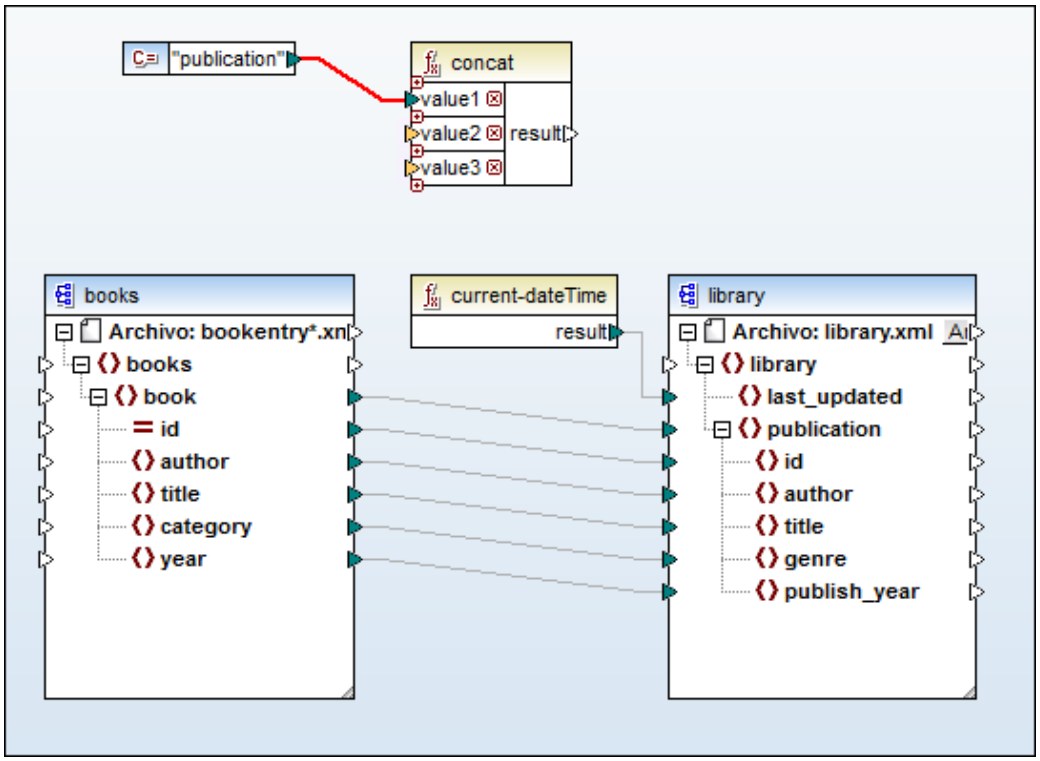


2. Inserte una constante (con el comando de menú **Insertar | Constante**) que tenga el valor **publication** y con el tipo *Cadena*.

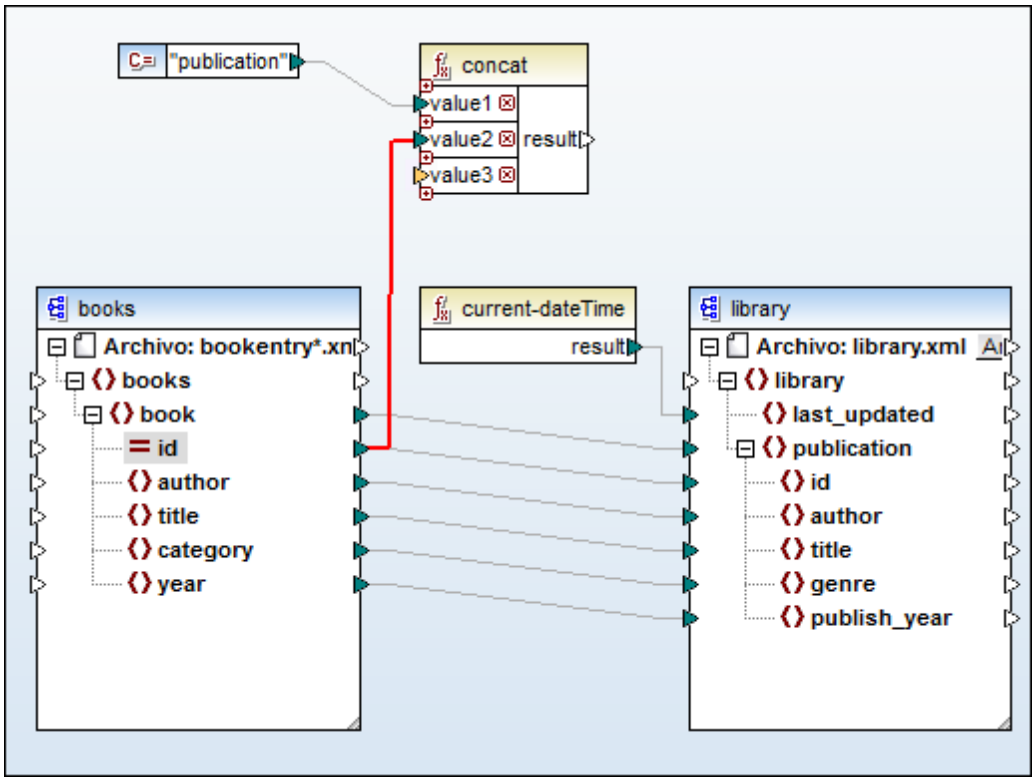


3. Conecte la constante con el valor **value1** de la función **concat**.

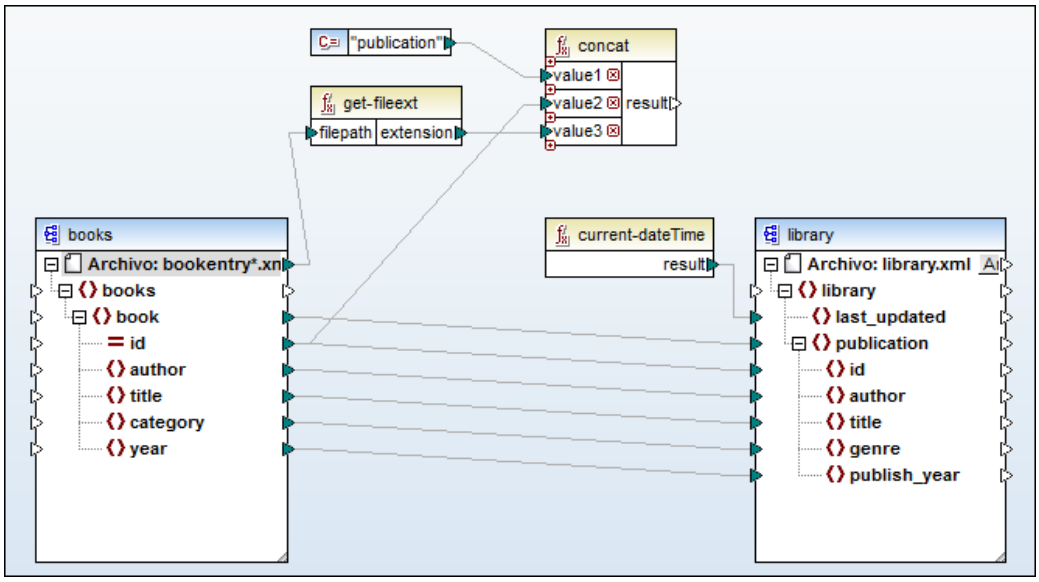
© 2018 Altova GmbH



4. Conecte el atributo `id` del componente de origen con el valor **value2** de la función **`concat`**.



5. Ahora busque la función **get-fileext** en la ventana Bibliotecas y arrástrela hasta el área de asignación. Cree una conexión entre el nodo de nivel superior del componente de origen (**Archivo: books.xml**) y el parámetro **filepath** de esta función. Después cree una conexión entre el resultado **result** de la función **get-fileext** y el valor **value3** de la función **concat**. De este modo se extrae solamente la extensión (en este caso `.xml`) del nombre del archivo de origen.

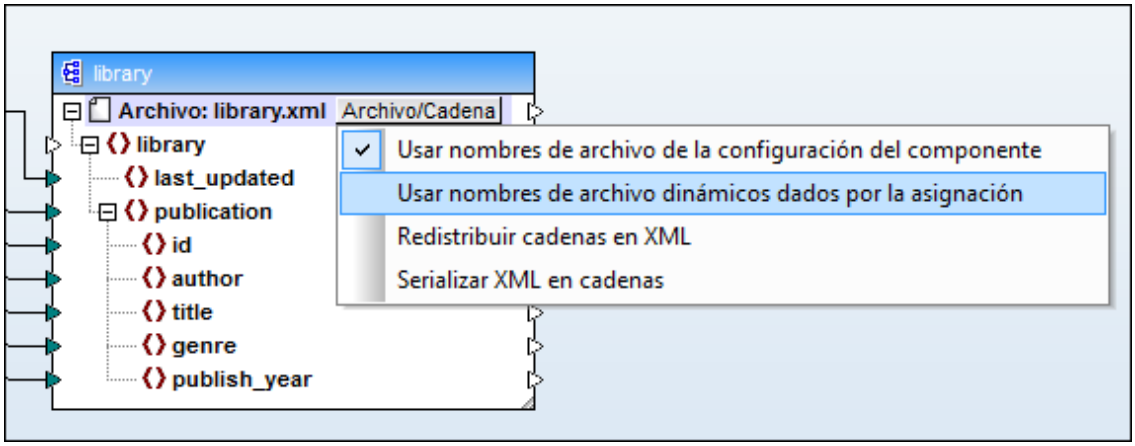


Llegados a este punto la función **concat** recibe como parámetros tres valores, que tras concatenarlos, crearán en nombre de los archivos de salida (p. ej. **publication1.xml**):

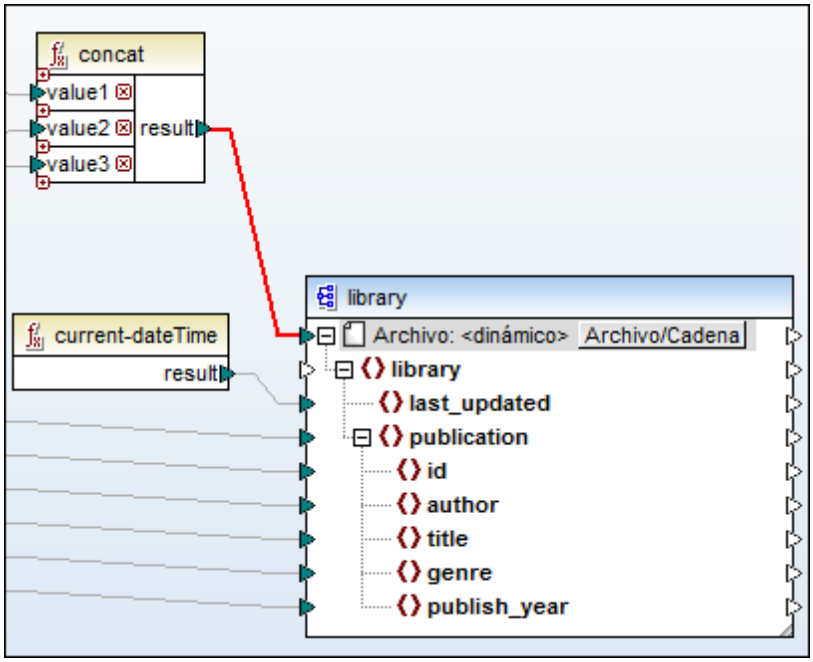
Partes del nombre de archivo de salida	Ejemplo
La constante **publication** aporta el valor de cadena constante `publication`.	publication
El atributo `id` del archivo XML de origen aporta un valor de identificador único para cada archivo. Esto evita que todos los archivos de salida tengan el mismo nombre.	1
La función **get-fileext** devuelve la extensión del nombre de archivo que se debe generar.	.xml

Ahora debemos dar a MapForce la orden de generar el nombre de archivo cuando se ejecute la asignación. Para ello haga clic en el botón Archivo o Archivo/Cadena del componente de destino y seleccione el comando **Usar nombres de archivo dinámicos dados por la asignación**.

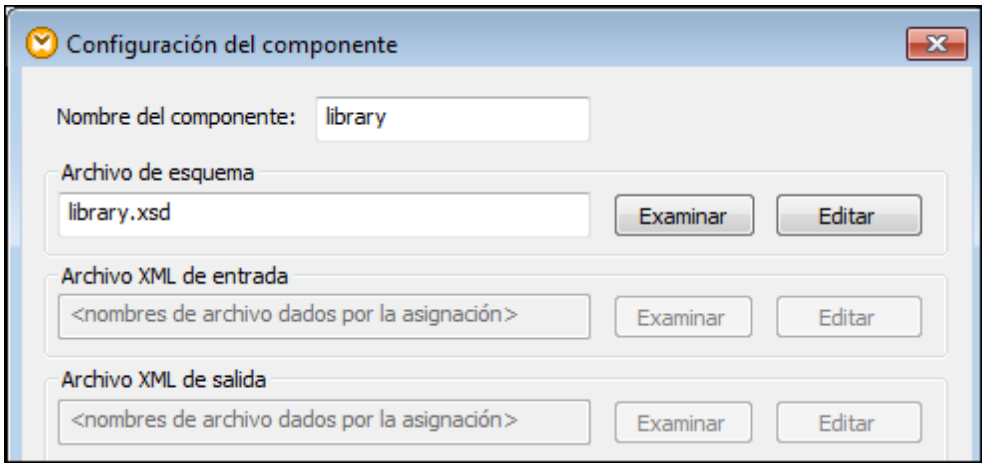
© 2018 Altova GmbH



MapForce ya tiene la orden de generar los archivos de instancia de forma dinámica, con el nombre dado por la asignación. En nuestro ejemplo el nombre se crea con la función **concat**. Por tanto, conectaremos el resultado de la función **concat** con el nodo **Archivo: <dinámico>** del componente de destino.



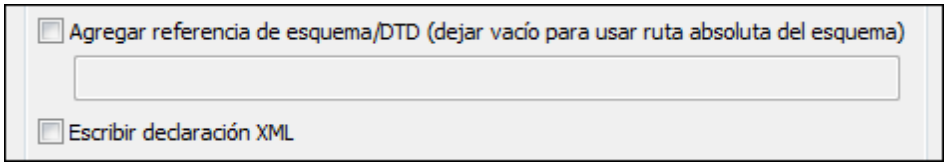
Si hace doble clic en el título del componente de destino, observará que los campos *Archivo XML de entrada* y *Archivo XML de salida* están deshabilitados y que en el cuadro de texto aparece *<nombres de archivos dados por la asignación>*.



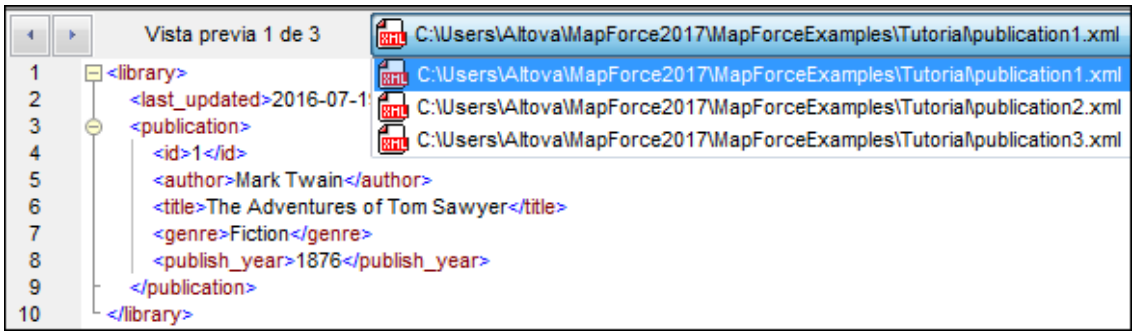
Esto nos indica que los nombres de los archivos de instancia se obtienen de forma dinámica de la asignación y por ello ya no es necesario definirlos en la configuración del componente.

Por último, eliminaremos la declaración de esquema y de espacio de nombres del componente de destino. Para ello desactivamos las casillas *Agregar referencia de esquema/DTD...* y *Escribir declaración XML* en el cuadro de diálogo "Configuración del componente".

 © 2018 Altova GmbH



Ahora puede ejecutar la asignación y consultar una vista previa de lo resultados y el nombre de los archivos que se generarán. Esta asignación genera varios archivos de salida y puede navegar por ellos con los botones de flecha situados en la esquina superior izquierda del panel *Resultados* (o seleccionando un archivo en la lista desplegable).



Altova MapForce 2018 Enterprise Edition

Tareas generales

4 Tareas generales

Esta sección del tutorial describe tareas y conceptos generales de MapForce, como trabajar con asignaciones, componentes, conexiones y proyectos de asignación de datos.

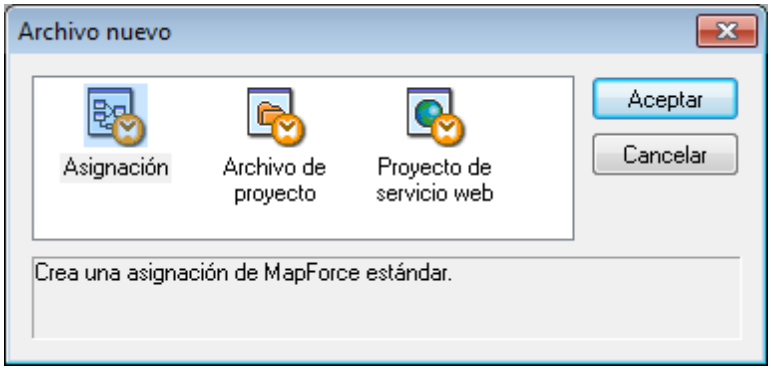
© 2018 Altova GmbH

4.1 Trabajar con asignaciones de datos

Un diseño de asignación de datos (o simplemente *asignación de datos*) es la representación visual de cómo se deben transformar datos de un formato a otro. Una asignación está formada por componentes, que se van añadiendo al área de asignación de MapForce para crear las transformaciones de datos (p. ej. para convertir documentos XML que siguen un esquema en documentos que siguen otro esquema). Una asignación válida está formada por uno o varios componentes de origen que están conectados a uno o varios componentes de destino. La asignación se puede ejecutar y MapForce ofrece una vista previa del resultado. También puede generar código desde MapForce y ejecutar la asignación en una aplicación externa. Por último, puede compilar la asignación en un archivo de ejecución de MapForce y automatizar la ejecución con MapForce Server o FlowForce Server. Los diseños de asignación de MapForce tienen la extensión de archivo **.mfd**.

Para crear una asignación nueva:

1. Tiene dos opciones:
   - o hacer clic en el comando de menú **Archivo | Nuevo** o
   - o hacer clic en el botón **Nuevo** de la barra de herramientas.
2. Ahora seleccione **Asignación** y después haga clic en el botón **Aceptar**.



La asignación se crea pero el área de asignación está vacía. Para ser válida la asignación necesita tener como mínimo dos componentes conectados, de modo que el siguiente paso consiste en agregar componentes a la asignación (*véase Agregar componentes a la asignación*) y dibujar conexiones entre ellos (*véase Trabajar con conexiones*).

4.1.1 Agregar componentes a la asignación

En MapForce un componente es el elemento gráfico que representa visualmente la estructura (esquema) de los datos o el elemento que determina cómo se deben transformar los datos (funciones). Los componentes son piezas fundamentales necesarias para construir una asignación. En el área de asignación de MapForce los componentes aparecen en forma de rectángulo. Estos son algunos ejemplos de componente:

- Constantes
- Bases de datos

- Filtros
- Condiciones
- Archivos planos (archivos de texto, CSV o de longitud fija)
- Funciones
- Documentos EDI (UN/EDIFACT, ANSI X12, HL7)
- Archivos Excel 2007+
- Componentes de entrada simples
- Componentes de salida simples
- Variables
- Documentos XBRL
- Esquemas XML y documentos DTD

Para agregar un componente a la asignación:

Hay varias maneras de agregar componentes a la asignación:

- En el menú **Insertar** haga clic en el comando que corresponda al tipo de componente que desea agregar (p. ej. **Archivo o esquema XML**).
- Arrastre un archivo desde el explorador de Windows hasta el área de asignación. Tenga en cuenta que esta operación solo es válida para componentes basados en archivos compatibles.
- Haga clic en el botón correspondiente de la barra de herramientas **Insertar componente**.



*Barra de herramientas **Insertar componente** (MapForce Enterprise Edition)*

Cada tipo de componente tiene un objetivo y comportamiento concretos. En algunos casos MapForce ofrece un asistente que le guiará en el proceso de creación del componente. Por ejemplo, si añade un esquema XML, aparece un cuadro de diálogo donde tiene la opción de seleccionar también un archivo de instancia.

Para más información consulte el apartado Trabajar con componentes. Además, la sección Orígenes y destinos de datos explica en detalle todas las tecnologías compatibles que pueden utilizarse como origen o destino de las asignaciones. Y en la sección Diseño de asignaciones encontrará información sobre los componentes integrados de MapForce que sirven para almacenar datos temporalmente o transformarlos (p. ej. filtros y componentes de ordenación).

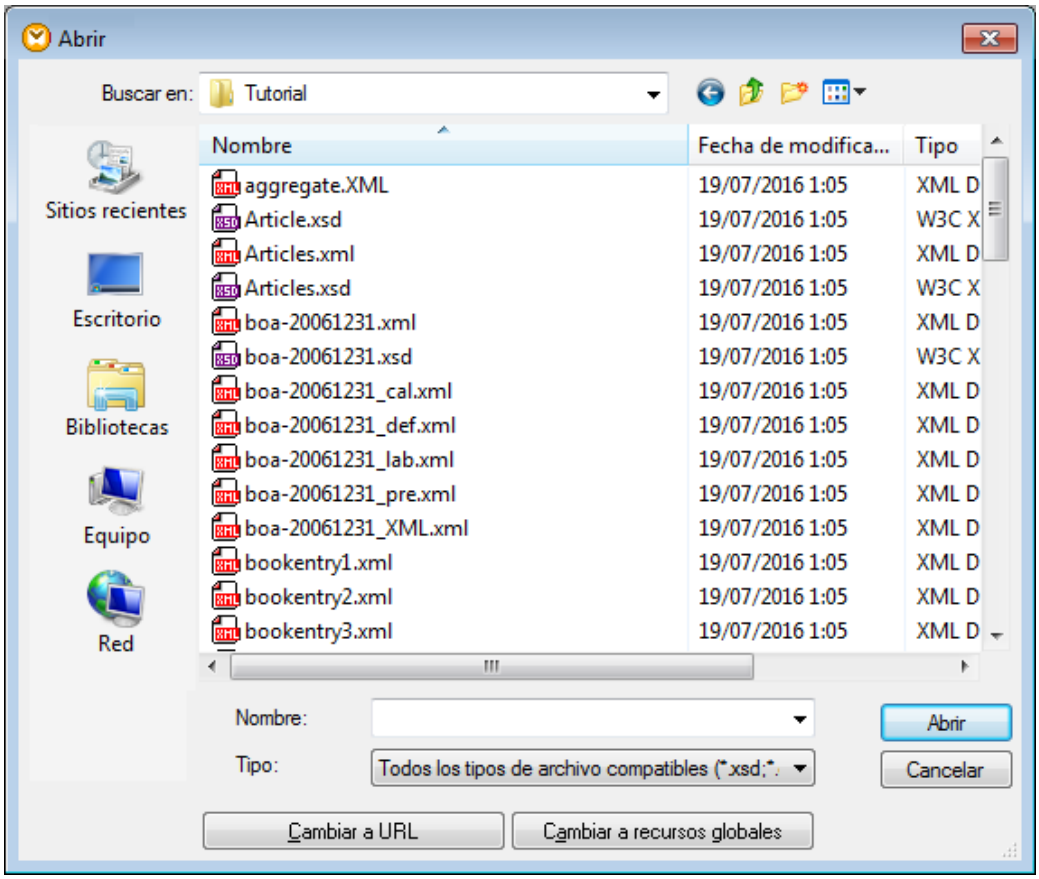
4.1.2 Agregar componentes desde una URL

Los componentes de la asignación también pueden estar basados en archivos ubicados en una URL, no solo en archivos locales. Sin embargo, esta operación solamente es compatible con los componentes de origen (es decir, la asignación puede leer datos de archivos remotos). Los protocolos compatibles son HTTP, HTTPS y FTP.

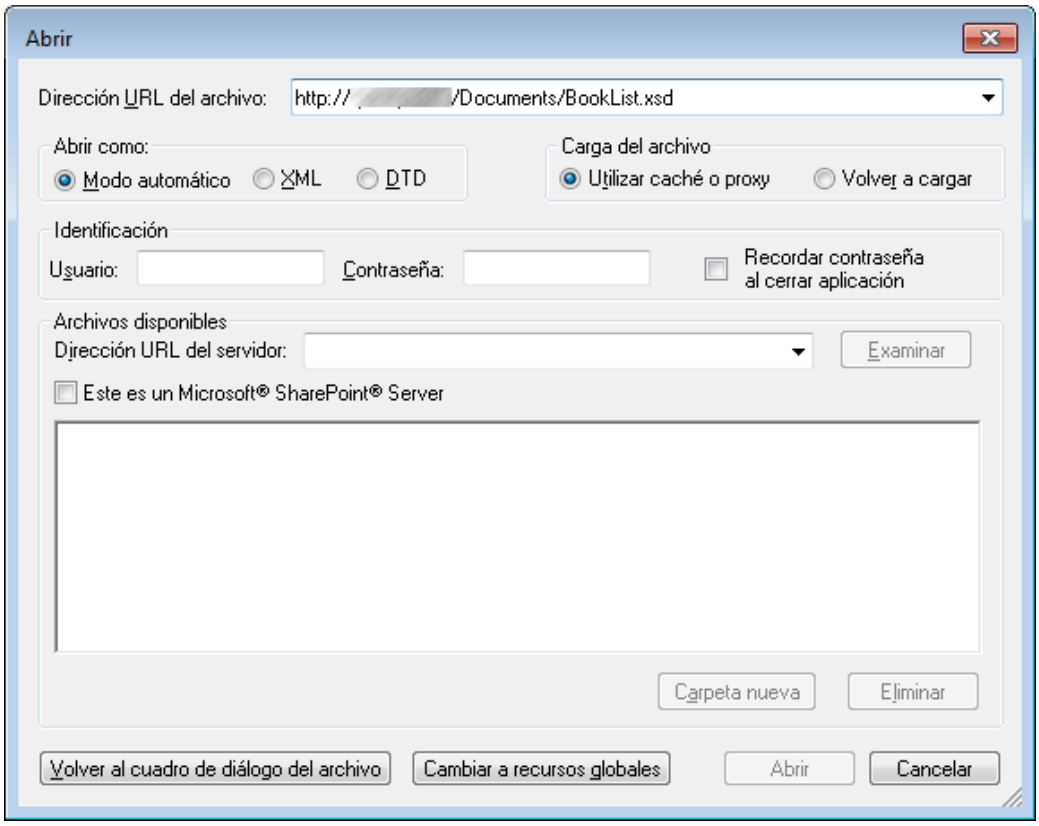
© 2018 Altova GmbH

Para agregar un componente desde una URL:

1. En el menú b seleccione el comando que corresponda al tipo de componente que desea agregar (p. ej. **Archivo o esquema XML**).
2. En el cuadro de diálogo "Abrir" haga clic en el botón **Cambiar a URL**.



3. Introduzca la URL del archivo en el cuadro de texto *Dirección URL del archivo:* y haga clic en **Abrir**.



Asegúrese que el tipo de archivo del campo *Dirección URL del archivo* coincide con el tipo de archivo especificado en el paso nº1.

Si el servidor requiere autenticación con contraseña, deberá introducir el nombre de usuario y la contraseña. Si desea que MapForce recuerde el nombre y la contraseña la próxima vez que se inicie, introduzca los datos de autenticación en el cuadro de diálogo "Abrir" y marque la casilla *Recordar contraseña al cerrar aplicación.*

El grupo de opciones *Abrir como* sirven para definir la gramática utilizada por el analizador para abrir el archivo. La opción predeterminada y recomendada es *Modo automático*.

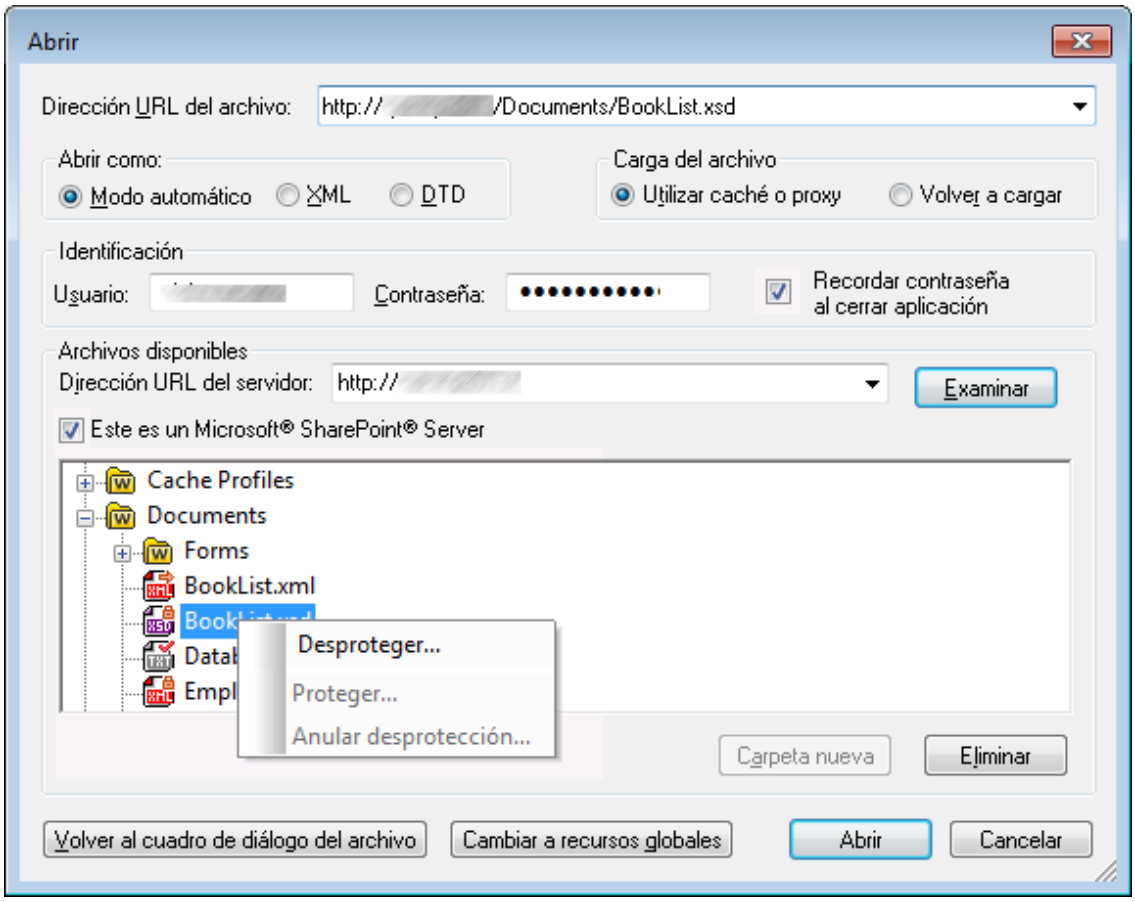
Si lo más probable es que el archivo que desea cargar no sufra modificaciones, seleccione la opción *Utilizar caché o proxy* para almacenar en caché los datos y así acelerar el proceso de carga del archivo. Sin embargo, si prefiere que el archivo se vuelva a cargar cada vez que abra la asignación, elija la opción *Volver a cargar*.

Si trabaja con un servidor WebDAV, introduzca la URL del servidor en el cuadro de texto *Dirección URL del servidor:* y haga en **Examinar** para explorar los archivos. Aunque la vista previa muestra todos los tipos de archivo, asegúrese de elegir el mismo tipo de archivo que especificó en el paso nº1 para evitar errores.

Si trabaja con un servidor Microsoft SharePoint Server, marque la casilla *Este es un Microsoft® SharePoint® Server*. Esto permite ver el estado de protección o desprotección de los archivos en

© 2018 Altova GmbH

el panel de vista previa. Si quiere impedir que otros usuarios editen el archivo en el servidor mientras lo utiliza en MapForce, haga clic con el botón derecho en el archivo y seleccione el comando **Desproteger**. Para proteger un archivo desprotegido con anterioridad, haga clic con el botón derecho y seleccione **Proteger**.



Cuadro de diálogo "Abrir" (en el modo URL)

4.1.3 Notas sobre transmisión por secuencias de datos

La transmisión por secuencias de datos es un mecanismo integrado de MapForce que permite usar orígenes de datos de gran tamaño como entrada o salida de las asignaciones de datos. No se debe confundir *transmisión por secuencias de datos* con los *objectos de secuencia* del código que MapForce genera. Los objetos de secuencia sirven para controlar datos si se integra el código generado por MapForce con una aplicación C# o Java personal).

La transmisión por secuencias de datos es relevante cuando se utilizan estos orígenes de datos:

- Archivos XML
- Archivos CSV
- Archivos de campo de longitud fija
- Bases de datos

Cuando use uno de estos orígenes de datos como entrada o salida de una asignación, MapForce tratará el origen de datos como una secuencia abierta de datos y procesará su contenido de forma secuencial, en lugar de cargar todos los datos en memoria.

Nota: la transmisión por secuencias de datos solamente es posible si tiene seleccionado el lenguaje de transformación integrado de MapForce (*véase* *Seleccionar el lenguaje de transformación*).

Nota sobre uso de memoria

Cuando la asignación de datos tenga entradas y salidas compatibles con la transmisión por secuencias de datos, pueden producirse errores por falta de memoria cuando la asignación exija un acceso aleatorio al origen de datos de entrada.

Por ejemplo, imagine que la asignación contiene un componente que aplica una función **`group-by`** a los datos de origen. Si se aplica la función **`group-by`** a toda la estructura del archivo de entrada, será necesario cargar en memoria todo el archivo de entrada y, por tanto, no será posible realizar la transmisión por secuencias de datos. Lo mismo ocurre cuando la operación exija cargar en memoria todo el contenido del origen de la asignación (p. ej. las operaciones de ordenación).

Cuando esto ocurra, la transformación se realizará correctamente si hay suficiente memoria virtual y suficiente espacio en disco disponible en el sistema.

4.1.4 Seleccionar el lenguaje de transformación

MapForce ofrece varios lenguajes de transformación de datos.

El lenguaje predeterminado de MapForce utiliza un potente motor integrado capaz de realizar las mismas transformaciones que los demás lenguajes. Cuando implemente asignaciones de MapForce en MapForce Server, el motor integrado las ejecuta sin necesidad de utilizar procesadores externos. Además, si la asignación no exige intervención manual en el proceso de transformación de datos, podrá usar FlowForce Server para automatizar los procesos de asignación por medio de trabajos programados.

Pruebe los diferentes lenguajes de transformación y decida cuál es el más adecuado para sus datos:

- BUILT-IN (este es el motor de transformación nativo que utiliza MapForce)
- C++
- C#
- Java
- XQuery
- XSLT 1.0
- XSLT 2.0

Para elegir el lenguaje de transformación tiene dos opciones:

- En el menú **Resultados** haga clic en el nombre del lenguaje que desea usar para la transformación.
- En la barra de herramientas **Selección del lenguaje** 

 © 2018 Altova GmbH

haga clic en el nombre del lenguaje que desea usar.

Nota: no todos los lenguajes son compatibles con todos los tipos de entrada y salida. Por ejemplo, si usa una base de datos como entrada o salida de la asignación, no podrá generar código XSLT. Por tanto, MapForce emite un mensaje de error cuando intente generar código o una vista previa de los resultados de una asignación con entradas o salidas incompatibles con el lenguaje seleccionado.

Opción BUILT-IN

Cuando se selecciona el lenguaje de transformación integrado BUILT IN para la asignación, MapForce utiliza su motor de transformación interno para ejecutar la asignación de datos. MapForce también usa implícitamente esta opción cada vez que se genera la vista previa de los resultados cuando el lenguaje de transformación elegido es Java, C# o C++.

Recomendamos elegir el lenguaje de transformación integrado BUILT IN en estos casos:

- Como opción predeterminada cuando no necesite usar un lenguaje concreto para transformar datos.
- Cuando necesite procesar archivos de gran tamaño y le preocupe el uso de memoria.
- Cuando necesite generar firmas digitales en el archivo XML de salida.

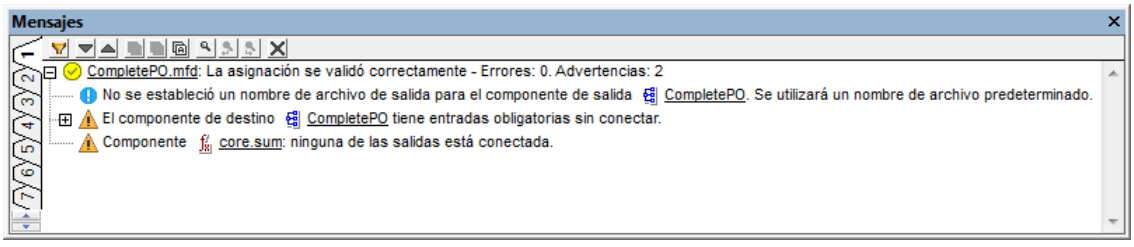
4.1.5 Validar la asignación de datos

MapForce valida las asignaciones automática cuando se hace clic en el panel *Resultados* para obtener una vista previa de los resultados de la transformación. También puede validar la asignación de forma explícita para identificar y corregir errores antes de ejecutar la asignación. Tenga en cuenta que mientras se ejecuta una asignación pueden darse errores de tiempo de ejecución o advertencias, dependiendo del tipo de datos procesado (p. ej. cuando se sobrescriben valores asignado a atributos).

Hay dos maneras de validar una asignación de forma explícita:

- Haciendo clic en el comando de menú **Archivo | Validar asignación**.
- Haciendo clic en el botón **Validar** de la barra de herramientas.

El resultado de la validación aparece en la ventana Mensajes:



Para validar la asignación MapForce comprueba, entre otras muchas cosas, si faltan conexiones

o si hay conexiones incorrectas, si hay tipos de componente incompatibles, etc. El resultado de la validación aparece en la ventana Mensajes con estos iconos de estado:

Icono	Significado
	La validación finalizó correctamente.
	La validación finalizó con advertencias.
	Error de validación.

La ventana Mensajes también puede mostrar mensajes de información, advertencia y error:

Icono	Significado
	Mensaje de información. La ejecución de la asignación no se interrumpe.
	Mensaje de advertencia. La ejecución de la asignación no se interrumpe. Pueden generarse cuando, por ejemplo, no se crean conexiones obligatorias con conectores de entrada. Cuando esto ocurra, se generarán resultados para los componentes que tengan conexiones válidas.
	Mensaje de error. La ejecución de la asignación se interrumpe y no se generan resultados. Tampoco se puede generar una vista previa del código XSLT o XQuery.

Para ver resaltar en el área de asignación el componente o la estructura que dio lugar al mensaje de información, advertencia o error basta con hacer clic en el texto subrayado del mensaje.

Cuando la asignación contiene componentes que transforman datos (p. ej. funciones o variables), la validación funciona de la siguiente manera:

- Si algún **conector de entrada** obligatorio está desconectado, se genera un mensaje de error y se interrumpe la transformación.
- Si algún **conector de salida** está desconectado, se genera una advertencia y continúa el proceso de transformación. El componente infractor y sus datos se pasan por alto y los datos no se asignan al documento de salida.

Para ver los resultados de cada validación por separado, basta con hacer clic en las diferentes pestañas numeradas situadas en el lateral izquierdo de la ventana Mensajes. Esta característica es muy práctica cuando se trabaja con varios archivos de asignación simultáneamente.

La ventana Mensajes también ofrece botones para:

- Filtrar los mensajes por tipo (p. ej. para ver solamente errores o advertencias).
- Navegar hacia arriba o hacia abajo por los mensajes.
- Copiar el texto del mensaje en el portapapeles.
- Buscar texto concreto en la ventana.
- Borrar el contenido de la ventana Mensajes.

Para más información consulte el apartado Interfaz del usuario.

© 2018 Altova GmbH

4.1.6 Validar resultados de la asignación

Al hacer clic en el panel *Resultados*, aparece una vista previa del resultado de la asignación. Este resultado se puede validar usando el esquema que tenga asociado. Por ejemplo, si la transformación genera un archivo XML, puede validarse con el esquema XML asociado. Si el componente de salida es un archivo EDI, el resultado se puede validar con la especificación EDI (*véase Validación de componentes EDI*). Asimismo, si el destino es un objeto JSON, el resultado se valida con el esquema JSON.

Cuando trabaje con archivos XML puede especificar el esquema asociado al archivo de instancia en el campo *Agregar referencia de esquema/DTD* del cuadro de diálogo "Configuración del componente" (*véase Configuración de componentes XML*). La ruta de acceso especifica la ubicación del archivo de esquema al que hace referencia el archivo XML de salida. Esto permite poder validar el archivo de instancia de salida cuando se ejecute la asignación. En el campo *Agregar referencia de esquema/DTD* puede introducir una dirección HTTP o una ruta de acceso absoluta o relativa. Si no marca la casilla *Agregar referencia de esquema/DTD,* no se podrá validar el archivo de salida. Por el contrario, si marca la casilla pero deja el cuadro de texto vacío, el nombre de archivo del esquema se genera en el resultado la validación se lleva a cabo con este nombre de esquema.

Para validar el resultado de la asignación tiene dos opciones:

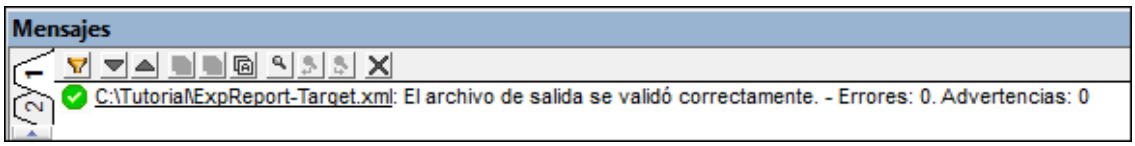
- Hacer clic en el botón **Validar archivo de salida**  de la barra de herramientas.



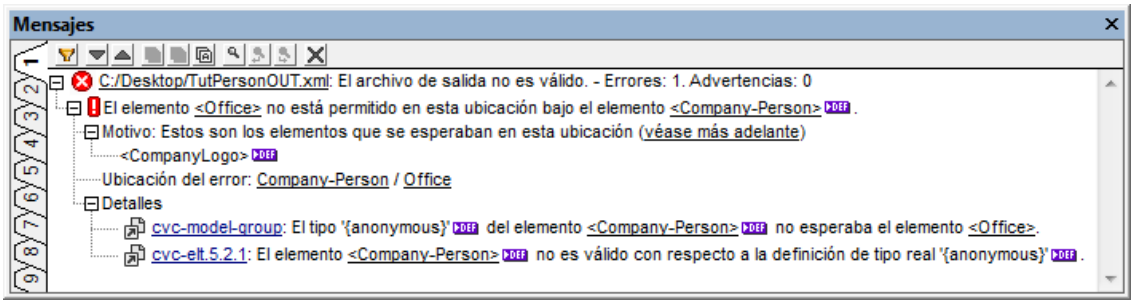
- Hacer clic en el comando de menú **Resultados | Validar el archivo de salida**.

Note: el botón **Validar archivo de salida** y su correspondiente comando de menú (**Resultados | Validar el archivo de salida**) solamente se habilitan si el archivo de salida admite la validación con un esquema.

El resultado de la validación aparece en la ventana Mensajes. Por ejemplo:



Si se produce un error de validación, la ventana Mensajes ofrece información al respecto:



Los mensajes de validación contienen hipervínculos que dirigen a información más detallada:

- Haga clic en la ruta de acceso del archivo para abrir el resultado de la transformación en el panel *Resultados* de MapForce.
- Haga clic en los enlaces de tipo `<nombreElemento>` para resaltar el elemento en el panel *Resultados*.
- Haga clic en el icono para abrir la definición del elemento en Altova XMLSpy (si está instalado).
- Haga clic en los hipervínculos de la sección *Detalles* (p. ej. `cvc-model-group`) para abrir la descripción de la regla de validación correspondiente en el sitio web http://www.w3.org/ .

4.1.7 Vista previa de resultados

En MapForce puede consultar una vista previa de los resultados de la asignación sin necesidad de ejecutar ni compilar el código generado con un procesador o compilador externo. Por lo general, se recomienda consultar la vista previa de resultados de la transformación en MapForce antes de intentar procesar el código generado en una aplicación externa.

Para generar la vista previa de los resultados MapForce ejecuta la asignación y rellena el panel *Resultados*.

Una vez disponibles en el panel *Resultados*, los datos se pueden validar y guardar (*véase Validar resultados de la asignación*). También puede buscar texto dentro del archivo de salida con el comando **Buscar** (**Ctrl+F**). Consulte también el apartado *Búsquedas en la vista Texto*.

Los mensajes de error, advertencia e información relacionados con la ejecución de la asignación aparecen en la ventana Mensajes (*véase Interfaz del usuario*).

Para generar la vista previa de resultados:

- Haga clic en el panel *Resultados* situado en la parte inferior de la ventana de asignación. MapForce usa el lenguaje de transformación seleccionado en la barra de herramientas **Lenguaje** para ejecutar la asignación y rellena el panel *Resultados* con el resultado.

Nota: si selecciona el lenguaje de transformación C++, C# o Java, MapForce ejecuta la asignación con su motor de transformación integrado. El resultado que aparece en el panel *Resultados* es idéntico al código Java, C++ o C# que MapForce genera, compila y ejecuta.

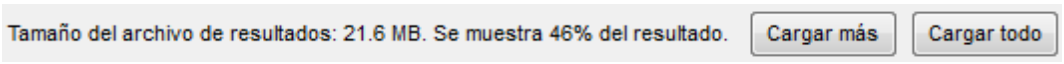
 © 2018 Altova GmbH

Puede guardar el resultado de la transformación de dos maneras diferentes:

- Haciendo clic en el comando de menú **Resultados | Guardar el archivo de salida**.
- Haciendo clic en el botón **Guardar resultado generado** de la barra de herramientas.

Vista previa parcial de los resultados

A la hora de generar una vista previa de archivos de gran tamaño, MapForce limita la cantidad de datos que se presentan en el panel *Resultados*. En concreto se presenta solamente parte del archivo y aparece el botón **Cargar más** en la parte inferior del panel. Al hacer clic en este botón se anexa la siguiente parte del archivo a la vista previa actual y así sucesivamente.



Nota: el botón **Pretty-print** se habilita cuando está cargado todo el archivo en el panel *Resultados*.

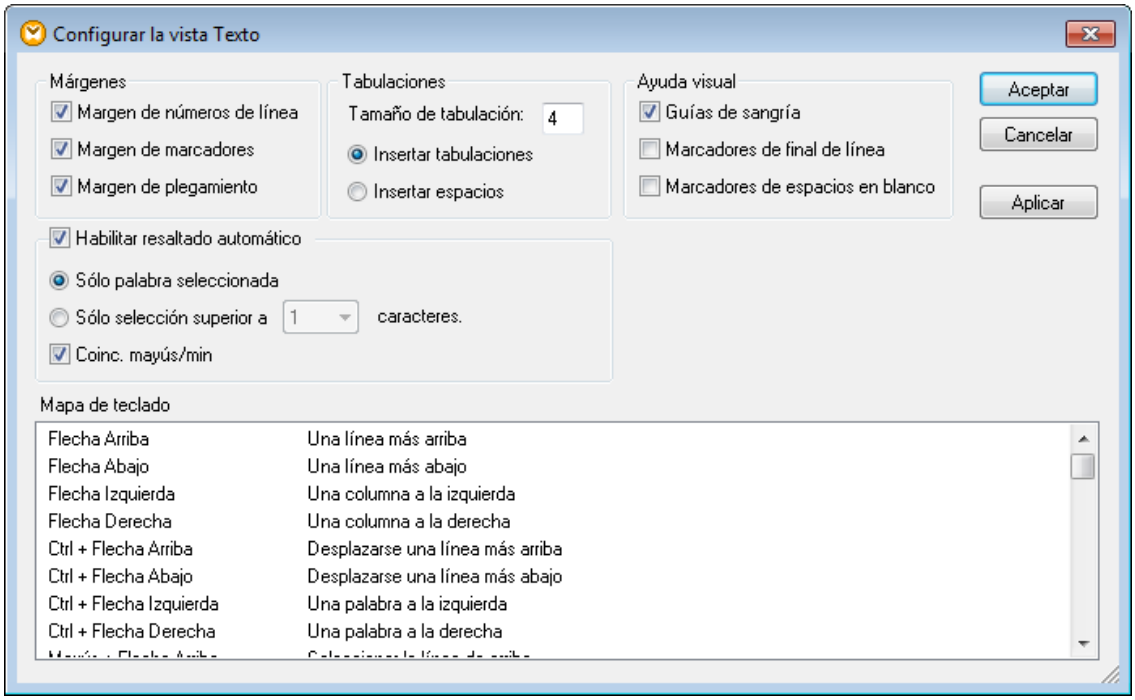
La vista previa también se puede configurar en la pestaña *Generales* del cuadro de diálogo "Opciones" (*véase Cambiar opciones de MapForce*).

4.1.8 Características de la vista Texto

Los paneles *Resultados*, *XSLT* y *XQuery* ofrecen varias funciones visuales para facilitar la consulta y edición del texto. Estas son las características de la vista Texto:

- Números de línea
- Color de sintaxis
- Marcadores
- Plegamiento de código
- Guías de sangría
- Marcadores de final de línea y marcadores de espacios en blanco
- Zoom
- Formato pretty-print
- Ajuste automático de línea
- Resaltado de texto

Estas características se pueden habilitar, deshabilitar y personalizar en el cuadro de diálogo "Configurar la vista Texto". La configuración elegida en este cuadro de diálogo se aplica a toda la aplicación y no solo al documento activo.

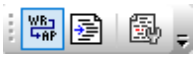


Cuadro de diálogo "Configurar la vista Texto"

Hay varias maneras de abrir el cuadro de diálogo "Configurar la vista Texto":

- Con el comando de menú **Resultados | Configurar la vista Texto**.
- Con el botón **Configurar la vista Texto** de la barra de herramientas.
- Haciendo clic con el botón derecho en el panel *Resultados* y seleccionando **Configurar la vista Texto** en el menú contextual.

Algunas características visuales de la vista Texto también se pueden activar/desactivar desde la barra de herramientas Vista Texto, desde el menú de la aplicación o con teclas de acceso rápido.



Barra de herramientas Vista Texto

Para ver la lista de teclas de acceso rápido consulte el panel *Mapa de teclado* situado en la parte inferior del cuadro de diálogo "Configurar la vista Texto" (*imagen anterior*).

Números de líneas

Los números de línea aparecen en el margen de números de línea (*imagen anterior*), que se puede activar o desactivar en el cuadro de diálogo "Configurar la vista Texto". Cuando se contrae una sección de texto, los números de línea del texto contraído también se ocultan.

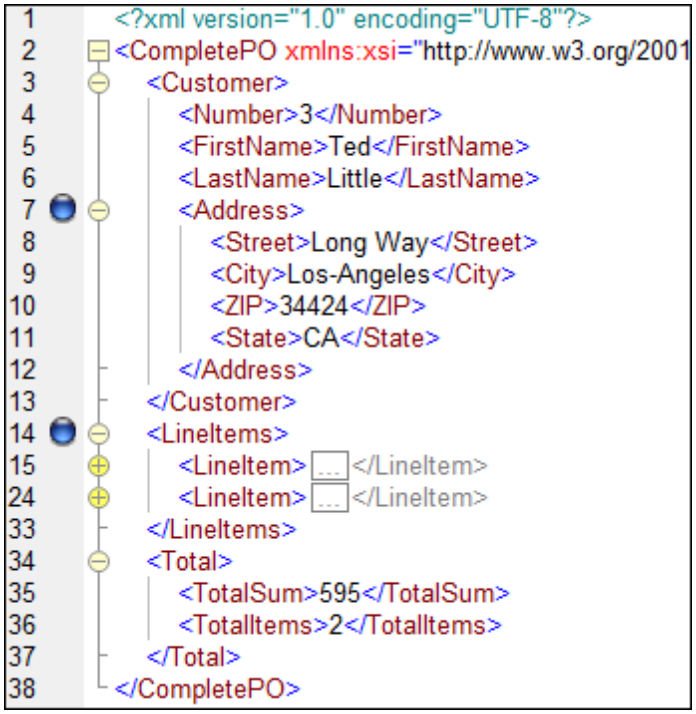
Color de sintaxis

El color de sintaxis se aplica de acuerdo al valor semántico del texto. Por ejemplo, en los

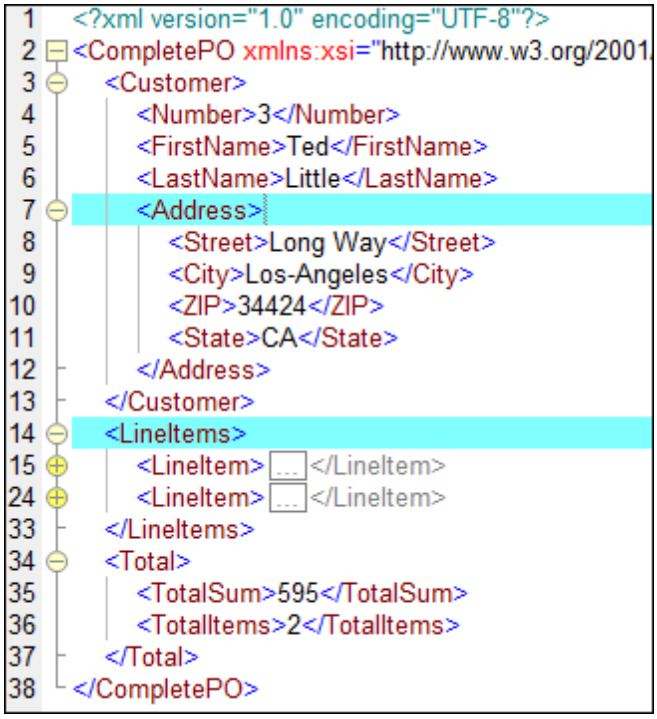
documentos XML, dependiendo de si el nodo XML es un elemento, un atributo, contenido, una sección CDATA, un comentario o una instrucción de procesamiento, el nombre del nodo (y en ocasiones el contenido del nodo) tendrá un color diferente.

Marcadores

Las líneas del documento se pueden marcar para realizar consultas rápidas más adelante. Si el margen de marcadores está activado, los marcadores aparecen en el margen de marcadores.



De lo contrario, las líneas marcadas se resaltan en color cian.



El margen de marcadores se puede activar o desactivar en el cuadro de diálogo "Configurar la vista Texto".

Puede navegar por los marcadores y editarlos con estos comandos:

Insertar o quitar marcador (Ctrl + F2)

Ir al siguiente marcador (F2)

Ir al marcador anterior (Mayús + F2)

Eliminar todos los marcadores (Ctrl + Mayús + F2)

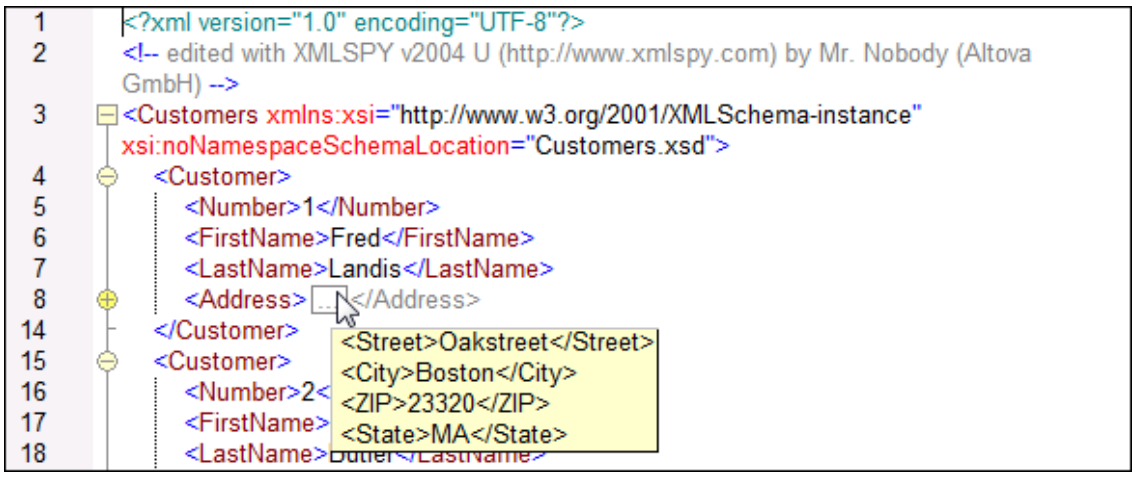
Todos estos comandos también están disponibles en el menú **Resultados** y en el menú contextual que aparece cuando se hace clic con el botón derecho en los paneles *Resultados*, *XSLT* y *XQuery*.

Plegamiento de código

La característica plegamiento de código permite expandir y contraer nodos en documentos XML, XQuery, JSON y CSS. Los nodos que se pueden expandir/contraer se marcan en el margen de plegamiento de código con el signo **+/-** (*imagen siguiente*). El margen puede activarse o desactivarse en el cuadro de diálogo "Configurar la vista Texto". Cuando está contraído, el nodo se marca con puntos suspensivos. Si pasamos el puntero por encima de los puntos suspensivos, aparece en pantalla un cuadro emergente con el contenido del nodo contraído (ver marca azul en la imagen). Si el contenido del nodo es demasiado largo para el cuadro emergente, esto se indica

 © 2018 Altova GmbH

con puntos suspensivos también.



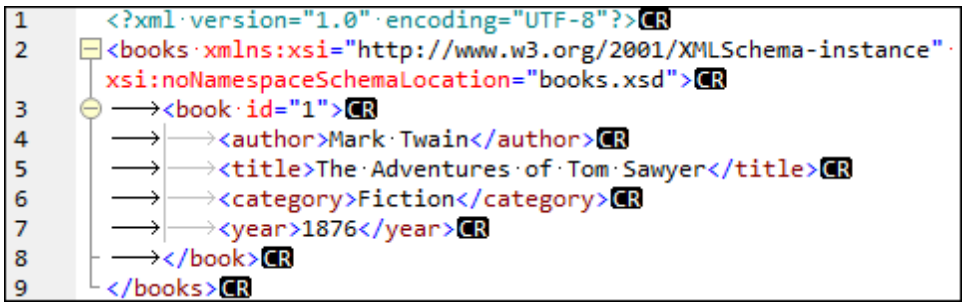
Guías de sangría

Las guías de sangría son líneas de puntos verticales que indican la longitud de la sangría de una línea. Las guías de sangría se pueden activar o desactivar en el cuadro de diálogo "Configurar la vista Texto".

Nota: las opciones *Insertar tabulaciones* e *Insertar espacios* del cuadro de diálogo "Configurar la vista Texto" tienen efecto cuando se ejecuta el comando **Resultados | Texto XML pretty-print**.

Marcadores de final de línea y marcadores de espacios en blanco

Los marcadores de final de línea (**EOL**) y los marcadores de espacios en blanco se pueden activar o desactivar en el cuadro de diálogo "Configurar la vista Texto". La captura de pantalla que aparece a continuación incluye estos marcadores. Cada punto representa un espacio en blanco, cada flecha representa un carácter de tabulación y CR representa un retorno de carro.



Alejarse y acercarse con el zoom

Puede acercarse y alejarse en la vista Texto si mueve la rueda del ratón mientras pulsa la tecla **Ctrl**. También puede usar las teclas - o + mientras pulsa la tecla **Ctrl**.

Formato pretty-print

El comando Texto XML pretty-print ajusta el formato del documento XML que está activo en la vista Texto para representarlo de forma estructurada. Por defecto se desplaza cada nodo secundario de su nodo principal insertando cuatro caracteres de espaciado. Esta configuración se puede personalizar en el cuadro de diálogo "Configuración de la vista Texto".

Para aplicar formato pretty-print a un documento XML seleccione el comando de menú **Resultados | Texto XML pretty-print** o haga clic en el botón **Pretty-print**  de la barra de herramientas.

Ajuste automático de línea

Para activar el ajuste automático de línea en el documento activo haga clic en el botón **Ajuste automático de línea**  de la barra de herramientas.

Resaltado de texto

Si se habilita la función de resaltado de texto en el cuadro de diálogo "Configurar la vista Texto", la aplicación resalta en el documento todas las coincidencias del texto seleccionado por el usuario. La selección se resalta en azul pálido y las coincidencias se resaltan en naranja pálido. La selección y sus coincidencias aparecen marcadas en la barra de desplazamiento por medio de marcadores grises cuadrados. Observe que la barra de desplazamiento también indica la posición actual del cursor por medio de un marcador azul.

Para habilitar la función de resaltado basta con marcar la casilla *Habilitar resaltado automático* en el cuadro de diálogo "Configurar la vista Texto". Puede definir como selección una palabra entera o un número fijo de caracteres. También puede especificar si se deben tener en cuenta o no las mayúsculas y minúsculas.

En el caso de la selección de caracteres, podrá especificar el número mínimo de caracteres que deben coincidir con la selección, empezando por el primer carácter de la selección. Por ejemplo, puede elegir que coincidan dos caracteres o más. En este caso, si selecciona un solo carácter del documento, no se encontrarán coincidencias. Pero si selecciona dos o más caracteres, sí se encontrarán coincidencias. Por ejemplo, si selecciona `t`, la aplicación no encontrará coincidencias. Pero si selecciona `ty`, la aplicación mostrará todas las coincidencias que contengan `ty`; si selecciona `typ`, la aplicación mostrará todas las coincidencias que contengan `typ` y así sucesivamente.

Cuando se trate de búsquedas de palabras, la aplicación considera que estos elementos son palabras independientes: nombres de elementos (sin paréntesis angulares), paréntesis angulares de etiquetas de elementos, nombres de atributos y valores de atributos sin comillas.

4.1.9 Búsquedas en la vista Texto

Puede realizar búsquedas en el texto de los paneles *Resultados*, *XQuery* y *XSLT* con ayuda de numerosas características gráficas.

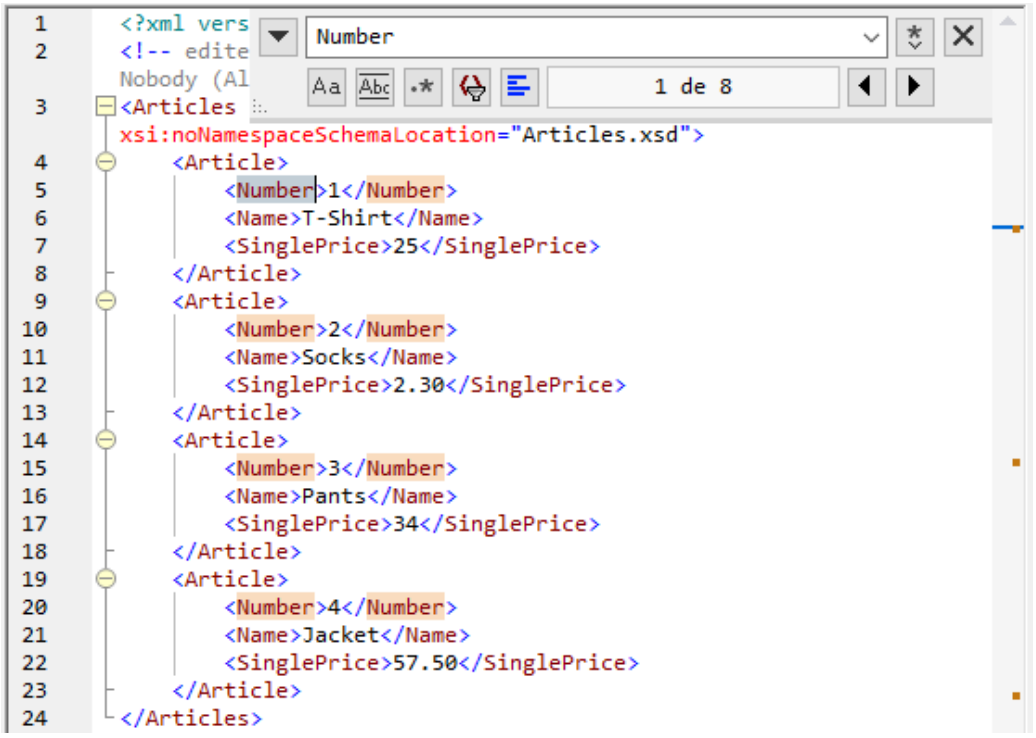
Para iniciar una búsqueda pulse **Ctrl+F** (o seleccione el comando de menú **Edición | Buscar**). La búsqueda puede realizarse en todo el documento o dentro de una selección de texto.

- Introduzca la cadena de texto que se debe buscar o seleccione una cadena de texto

© 2018 Altova GmbH

buscada en las últimas diez búsquedas en el cuadro combinado.

- Cuando introduzca o seleccione la cadena de búsqueda, todas las coincidencias se resaltarán y las posiciones de los resultados se indican en la barra de desplazamiento por medio de marcadores color beige.
- El resultado que esté seleccionado se resalta en un color distinto al de los demás resultados y su posición se indica en la barra de desplazamiento por medio de un marcador de cursor azul oscuro.
- El número total de resultados aparece debajo del término de búsqueda, además de la posición de índice del resultado que esté seleccionado. Por ejemplo, **2 de 4** indica que el segundo resultado (de un total de cuatro) está seleccionado.
- Puede recorrer los resultados en ambos sentidos con los botones **Anterior** **(Mayús +F3)** y **Siguiente** **(F3)** situados en la esquina inferior derecha.



Para cerrar el cuadro de diálogo "Buscar" haga clic en el botón **Cerrar** situado en la esquina superior derecha o pulse **Esc**.

Debe tener en cuenta estas características de la función de búsqueda y reemplazo:

- El cuadro de diálogo "Buscar" no es modal. Esto significa que puede dejarlo abierto mientras utiliza la vista Texto.
- Si tiene texto seleccionado antes de abrir el cuadro de diálogo, el texto seleccionado se inserta automáticamente en el campo del término de búsqueda.
- Para realizar búsquedas dentro de una selección: (i) marque la selección, (ii) active la opción **Buscar en la selección** para bloquear la selección e (iii) introduzca el término de búsqueda. Para buscar dentro de otra selección, desbloquee la selección actual desactivando la opción **Buscar en la selección**, marque una nueva selección

y active otra vez la opción **Buscar en la selección**  .

- Después de cerrar el cuadro de diálogo "Buscar" puede repetir la búsqueda actual con solo pulsar **F3** (búsqueda hacia adelante) o **Mayús+F3** (búsqueda hacia atrás). El cuadro de diálogo aparece al pulsar estas teclas.

Opciones de búsqueda

Los criterios de búsqueda pueden configurarse con los botones situados debajo del campo del término de búsqueda. Si una opción está activada, su botón aparece en color azul. Estas son las opciones de búsqueda disponibles:

Opción	Icono	Descripción
Coinc. mayús/ min		La búsqueda tiene en cuenta las mayúsculas y minúsculas a la hora de realizar la búsqueda (`Address` no es lo mismo que `address`).
Sólo palabras completas		Solo se consideran coincidencias las palabras completas. Por ejemplo, si activa esta opción y busca el término `fit`, la aplicación devuelve como resultado la palabra `fit`, pero no `fitness`.
Expresión regular		Si activa esta opción, el término de búsqueda se lee como expresión regular (*véase más abajo*).
Buscar delimitador		Cuando se introduce un término de búsqueda, los resultados de la búsqueda se resaltan y uno de ellos se marcará como selección actual. Con la opción **Buscar delimitador** puede definir si esta primera selección se hace en relación a la posición actual del cursor o no. Es decir, si la opción **Buscar delimitador** está activada, entonces el primer resultado seleccionado será el siguiente resultado a partir de la posición actual del cursor. Por el contrario, si la opción **Buscar delimitador** está desactivada, el primer resultado seleccionado será el primer resultado del documento, empezando desde el principio.
Buscar en selección		Si activa esta opción, la selección actual se bloquea y la búsqueda se ejecuta en la selección solamente. De lo contrario, la búsqueda se ejecuta en todo el documento. Antes de realizar una selección nueva deberá desbloquear la selección actual desactivando el botón de la opción **Buscar en selección**.

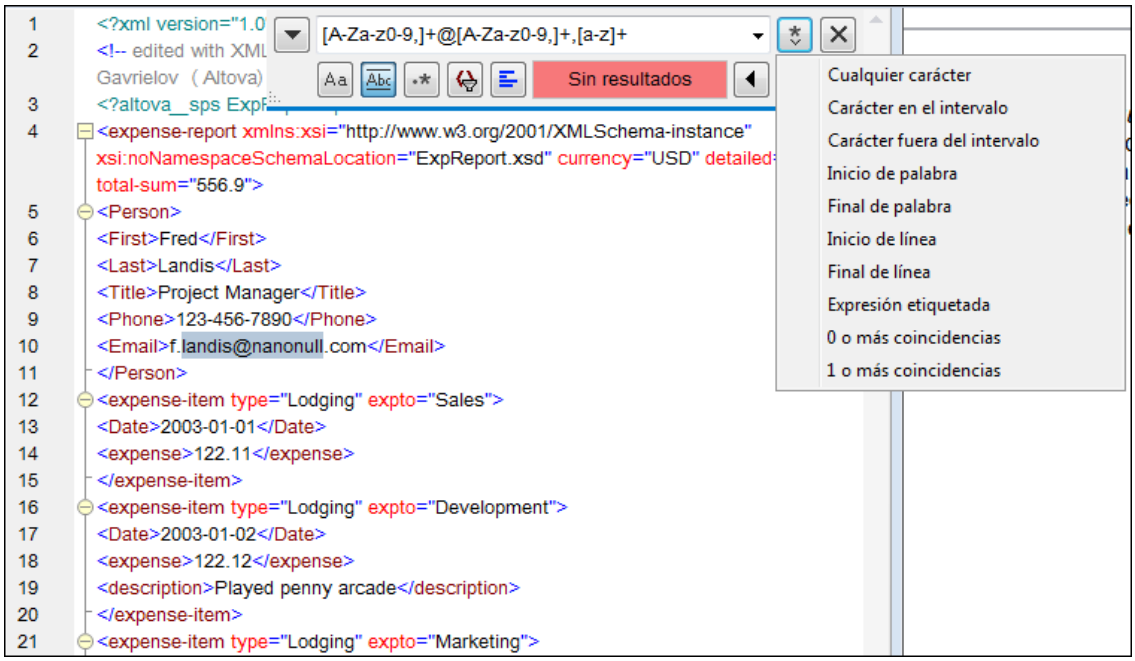
Uso de expresiones regulares

Puede usar expresiones regulares (regex) para buscar cadenas de texto en el documento. Para ello lo primero es activar la opción **Expresión regular**  . Al activar esta opción estamos especificando que el texto del campo del término de búsqueda debe evaluarse como expresión regular. El segundo paso consiste en introducir la expresión regular en el campo de búsqueda. Si

© 2018 Altova GmbH

necesita ayuda para construir su expresión regular, haga clic en el botón **Generador de expresiones regulares** (situado a la derecha del campo de búsqueda). Seleccione un elemento de la lista desplegable para introducir los caracteres correspondientes en el campo de búsqueda.

A continuación puede ver un ejemplo de expresión regular que se utiliza para buscar direcciones de correo electrónico.



A continuación puede ver una lista de metacaracteres de expresión regular compatibles con la función de búsqueda y reemplazo.

`.`	Cualquier carácter. Es un comodín para un solo carácter.
`(abc)`	Los metacaracteres `(` y `)` marcan el inicio y el final de una expresión regular. Las expresiones regulares pueden serle de utilidad a la hora de etiquetar (es decir, recordar) una región concreta del resultado de la búsqueda y poder hacerle referencia más adelante (referencia inversa). Las expresiones regulares son similares a las subexpresiones (grupos indizados) de la versión .NET de las expresiones regulares. Puede etiquetar (y hacer referencia inversa a) un máximo de 9 subexpresiones. Por ejemplo, **`(the) \1`** encuentra la cadena `the the`. Esta expresión significa literalmente: buscar la cadena "the" (y recordarla como región etiquetada), seguida de un espacio, seguida de una referencia inversa a la región etiquetada encontrada previamente.
`\n`	Siendo `n` un número del 1 al 9, `n` hace referencia a la correspondiente región etiquetada (*ver fila anterior*).
`\<`	Inicio de palabra.

\>	Final de palabra.
\	Escapes the character following the backslash. In other words, the expression \x allows you to use the character x literally. For example, \[would be interpreted as [and not as the start of a character set. Inserta un carácter de escape al carácter que aparece después de la barra diagonal inversa. En otras palabras, la expresión \x permite usar el carácter \ de forma literal. Por ejemplo, \[se interpretaría como [y no como el principio de un conjunto de caracteres.
[...]	Encuentra cualquiera de los caracteres del conjunto. Por ejemplo, **[abc]** encuentra los caracteres a, b o c. También puede usar intervalos como **[a-z]** para buscar cualquier carácter en minúsculas.
[^...]	Encuentra cualquier carácter que no esté en este conjunto. Por ejemplo, **[^A-Za-z]** encuentra cualquier carácter excepto caracteres alfabéticos en mayúsculas o minúsculas.
^	Encuentra el inicio de línea (a no ser que se use dentro de un conjunto de caracteres, ver fila anterior).
$	Encuentra el final de línea. Por ejemplo, **A+$** encuentra una A o más de una A que estén al final de una línea.
*	Encuentra cero o más instancias de la expresión precedente. Por ejemplo, **Sa*m** encuentra Sm, Sam, Saam, Saaam, etc.
+	Encuentra una o más instancias de la expresión precedente. Por ejemplo **Sa+m** encuentra Sam, Saam, Saaam, etc

4.1.10 Vista previa del código XSLT

Puede consultar una vista previa del código XSLT generado con MapForce seleccionando **XSLT 1.0** o **XSLT 2.0** como lenguaje de transformación (*véase* *Seleccionar el lenguaje de transformación*).

Para obtener la vista previa del código XSLT generado:

- Si se trata de código XSLT 1.0, haga clic en el botón **XSLT** situado en la parte inferior de la ventana de asignación.
- Si se trata de código XSLT 2.0, haga clic en el botón **XSLT2** situado en la parte inferior de la ventana de asignación.

Nota: los paneles *XSLT* y *XSLT2* de la ventana de asignación solamente se habilitan cuando el lenguaje de transformación seleccionado es **XSLT** o **XSLT2** respectivamente.

© 2018 Altova GmbH

4.1.11 Generar código XSLT

Para generar código XSLT:

1. Seleccione el comando de menú **Archivo | Generar código en | XSLT 1.0 (XSLT 2.0)**.
2. Seleccione la carpeta donde desea guardar el archivo XSLT generado y haga clic en **Aceptar**. MapForce genera el código y muestra el resultado de la operación en la ventana Mensajes.

El nombre del archivo .xslt que se genera tiene el formato **<A>MapTo<B>.xslt**:

- "<A>" es el valor del campo *Nombre de la aplicación* del cuadro de diálogo "Configurar asignación" (*véase Cambiar configuración de la asignación*).
- "<B>" es el nombre del componente de destino. Para cambiar este valor abra la configuración del componente de destino y edite el valor del campo *Nombre del componente* (*véase Cambiar configuración de los componentes*).

La carpeta donde se guarda el archivo .xslt también contiene un archivo por lotes llamado **DoTransform.bat** que se puede ejecutar con RaptorXML Server para transformar los datos (*véase Automatización con RaptorXML Server*).

Para ejecutar la transformación con RaptorXML Server:

1. Descargue e instale el software servidor RaptorXML Server desde la página de descargas de Altova (https://www.altova.com/es/download-trial-server.html).
2. Inicie el archivo de procesamiento por lotes **DoTransform.bat** situado en la carpeta de salida designada con anterioridad.

Recuerde que a veces es necesario agregar la carpeta de instalación de RaptorXML a la variable **path** de las variables de entorno. Para más información consulte la documentación de RaptorXML en el sitio web de Altova (https://www.altova.com/es/documentation.html).

4.1.12 Vista previa del código XQuery

Puede consultar una vista previa del código XQuery generado con MapForce seleccionando **XQuery** como lenguaje de transformación (*véase Seleccionar el lenguaje de transformación*).

Para obtener la vista previa del código XQuery generado:

- Haga clic en el botón **XQuery** situado en la parte inferior de la ventana de asignación.

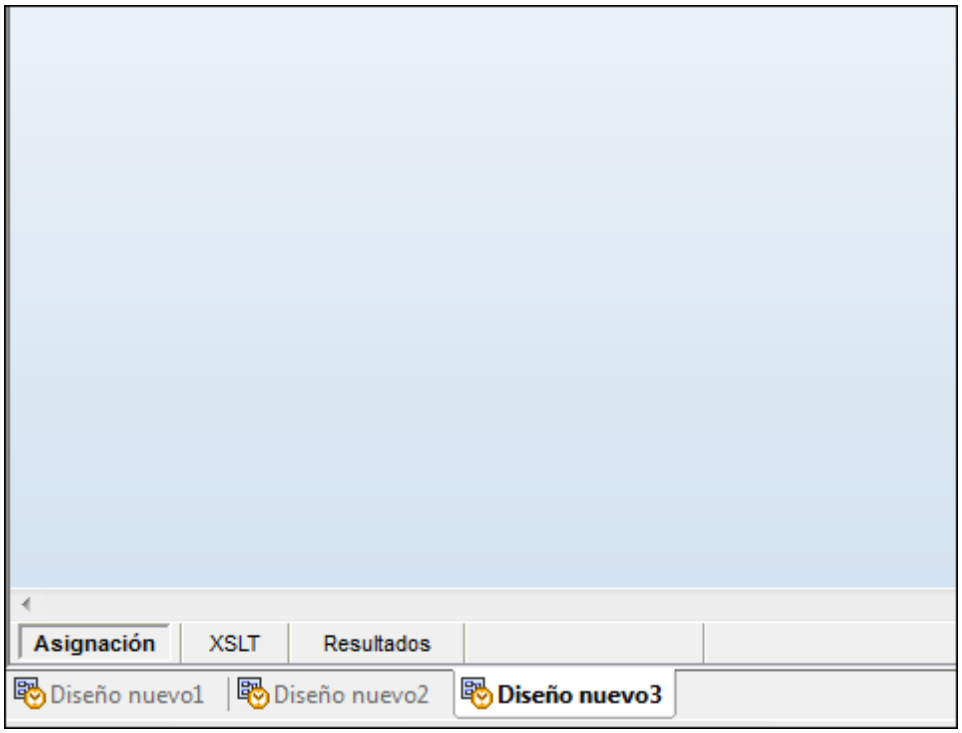
Nota: el panel *XQuery* de la ventana de asignación solamente se habilita cuando el lenguaje de transformación seleccionado es **XQuery**.

4.1.13 Trabajar con varias ventanas de asignación

MapForce utiliza la especificación de interfaz de múltiples documentos MDI. Cada archivo de asignación que se abre en MapForce tiene una ventana distinta. Esto permite trabajar con varias

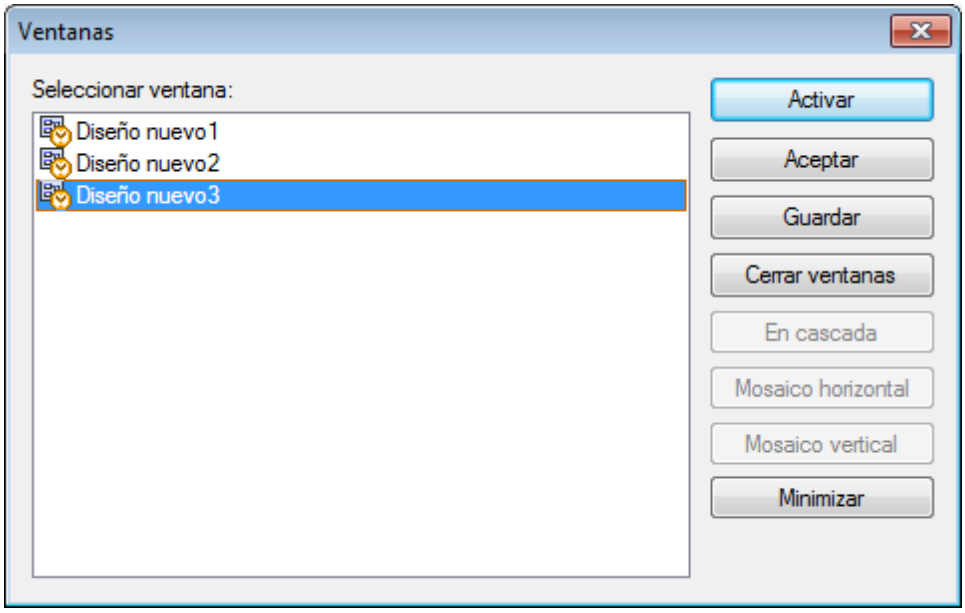
ventanas de asignación y organizarlas o ajustar su tamaño dentro de la ventana de la aplicación. También puede organizar las ventanas abiertas en mosaico horizontal o vertical y en cascada.

Cuando hay varias asignaciones abiertas en MapForce puede navegar por ellas haciendo clic en las pestañas que aparecen en la parte inferior del panel de asignación.



Además, el menú **Ventanas** ofrece comandos para gestionar las ventanas de las asignaciones y el comando **Ventanas | Ventanas...** abre el cuadro de diálogo del mismo hombre donde puede realizar operaciones como guardar, cerrar o minimizar ventanas.

© 2018 Altova GmbH



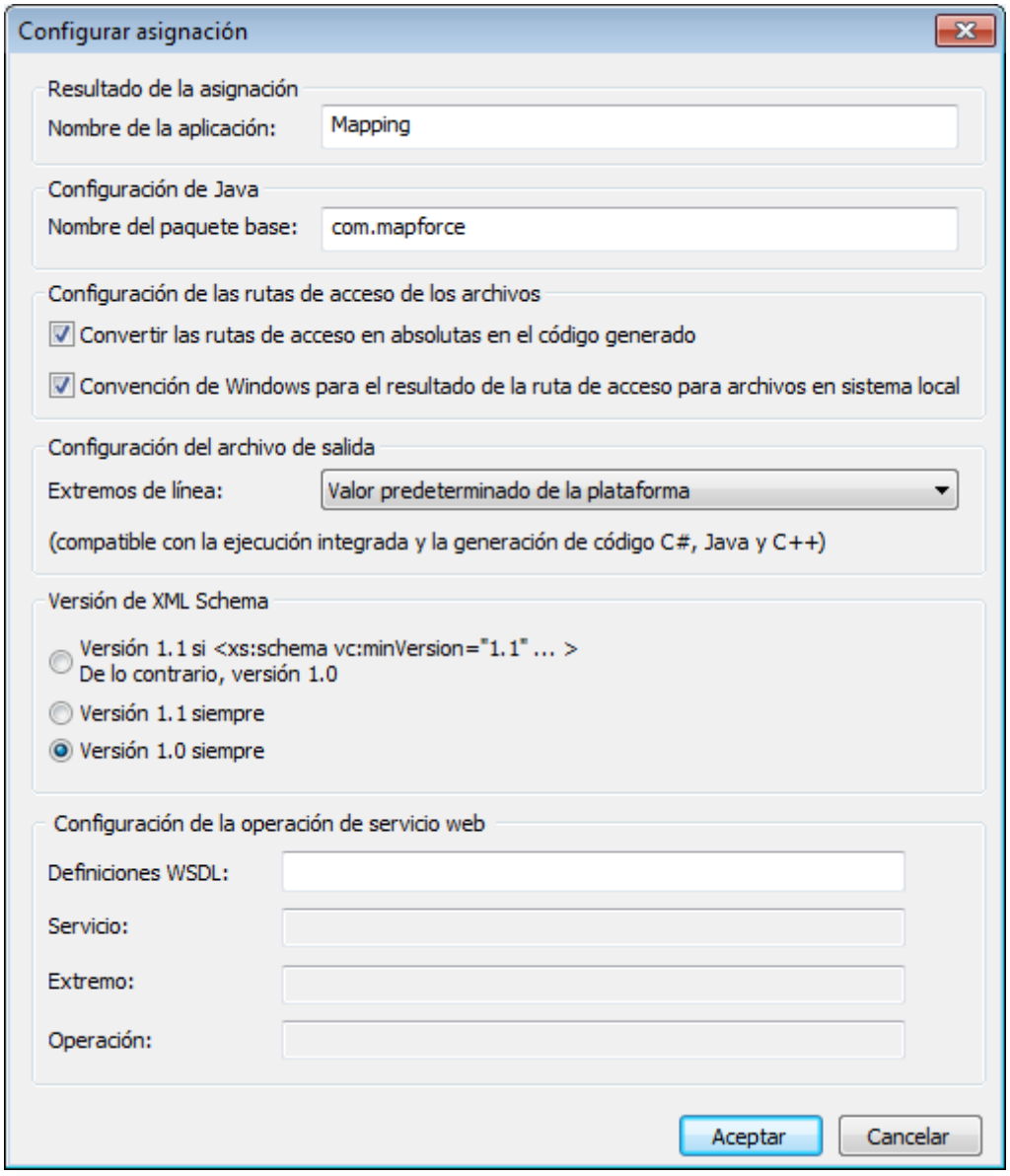
Para seleccionar varias ventanas en el cuadro de diálogo "Ventanas" haga clic en las entradas pertinentes mientras pulsa la tecla **Ctrl**.

4.1.14 Cambiar configuración de la asignación

Puede cambiar la configuración del diseño de asignación activo desde el cuadro de diálogo "Configurar asignación". Las opciones de configuración definidas en este cuadro de diálogo se almacenan en el archivo `*.mfd`.

Para abrir el cuadro de diálogo "Configurar asignación":

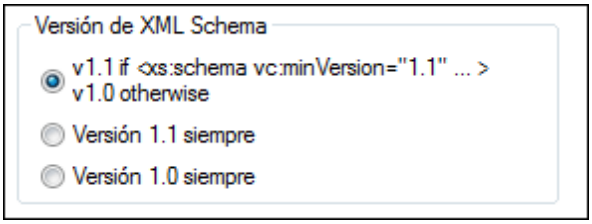
- Haga clic en el comando de menú **Archivo | Configurar asignación**.



Estas son las opciones que se pueden configurar en este cuadro de diálogo:

Nombre de la aplicación	Define el prefijo de nombre del archivo XSLT1.0/2.0 o el nombre de aplicación Java, C# o C++ para los archivos de transformación generados.
Nombre del paquete base	Define el nombre del paquete base para los resultados Java.
Convertir las rutas de acceso en absolutas en el código generado	Define si las rutas de acceso de los archivos deben ser relativas o absolutas en el código de programa generado y en los archivos de ejecución de MapForce Server (mfx) y funciones de asignación implementadas en FlowForce

© 2018 Altova GmbH

	Server. Consulte Rutas de acceso en el código generado para obtener más información.
Convención de Windows para el resultado de la ruta de acceso para archivos en sistema local	Marque esta casilla para que MapForce siga la convención de Windows para rutas de acceso. Cuando genere código XSLT2 o XQuery, el nombre del archivo procesado se recupera internamente usando la función `document-uri` que devuelve una ruta con el formato archivo:// URI para lo archivos locales. Si marca esta casilla, la especificación archivo:// ruta URI se convierte automáticamente a una ruta de archivo Windows completa (p. ej. "C:\...") y así poder seguir con el precesamiento
Extremos de línea	En este cuadro combinado puede especificar el extremo de las líneas de los archivos de salida. La opción **Valor predeterminado de la plataforma** es la opción predeterminada para el sistema operativo de destino (p. ej. CR+LF para Windows, LF para Mac OS X o LF para Linux). También puede seleccionar el extremo de línea que prefiera. Las opciones seleccionadas aquí son de una importancia fundamental para la implementación de asignaciones en servidores FlowForce Server que se ejecuten en sistemas operativos distintos.
Versión de XML Schema	En este grupo de opciones puede indicar qué versión de XML Schema se usa en el archivo de asignación. Aquí puede indicar si prefiere **cargar** siempre los esquemas conforme a la versión 1.0 o conforme a la versión 1.1. Sin embargo, tenga en cuenta que no todas las características propias de la versión 1.1 son compatibles con MapForce. Si la declaración `xs:schema vc:minVersion="1.1"` está presente en el esquema, se usa la versión 1.1. De lo contrario se usa la versión 1.0.  Si el documento XSD no tiene el atributo `vc:minVersion` o el valor de `vc:minVersion` no es `1.0` ni `1.1`, entonces el modo predeterminado será XSD 1.0. **Nota:** no se debe confundir el atributo `vc:minVersion` con el atributo `xsd:version`. El primero almacena el número de versión XSD, mientras que el segundo almacena el número de versión del documento.

	Cuando se cambia esta opción de configuración en una asignación, todos los esquemas que tengan la versión de esquema XML seleccionada se volverán a cargar y puede que la validez de la asignación se vea afectada.
Configuración de la operación de servicio web	Los campos *Definiciones WSDL*, *Servicio*, *Extremo* y *Operación* se rellenan automáticamente si el documento de asignación forma parte de una implementación de servicio web.

© 2018 Altova GmbH

4.2 Trabajar con componentes

Los componentes son los elementos centrales de los diseños de asignación de datos en MapForce. Por lo general, el término *componente* se utiliza para denominar cualquier objeto que actúe como origen o destino de datos o que represente los datos en la asignación en una fase de procesamiento intermedio.

Hay dos categorías de componentes: componentes de estructura y componentes de transformación.

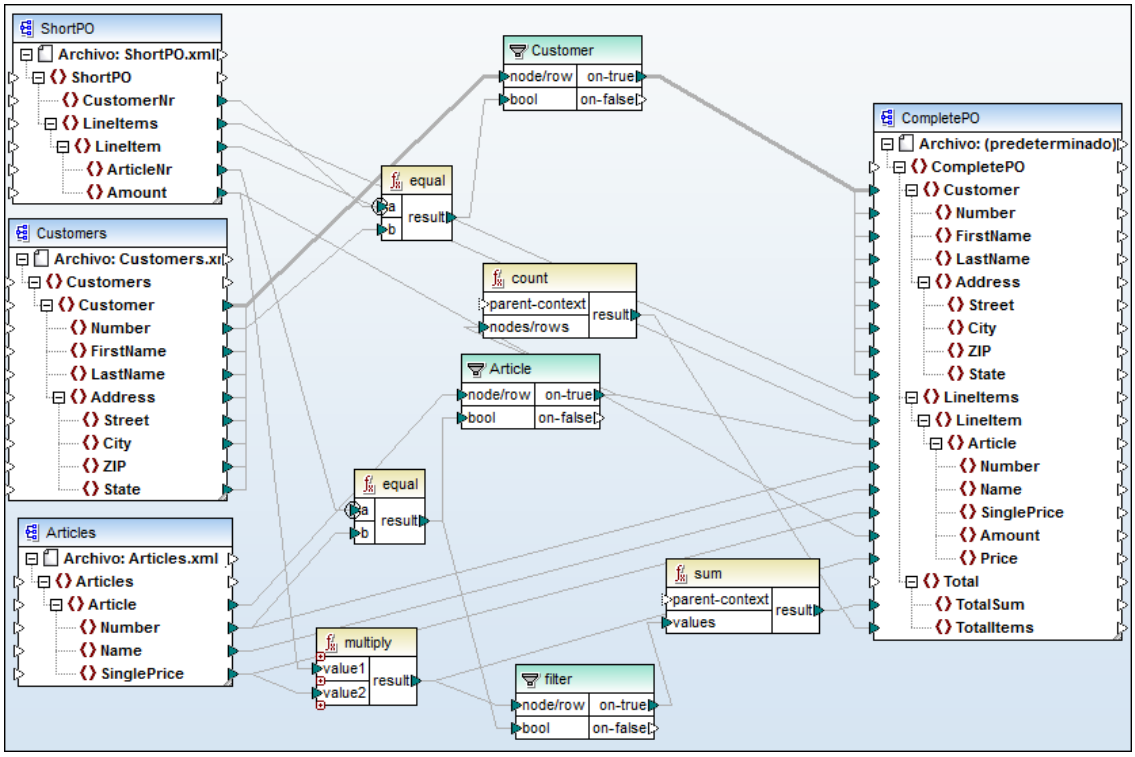
Los **componentes de estructura** representan la estructura abstracta o el esquema de los datos. Por ejemplo, cuando se añade un archivo XML al área de asignación (con el comando de menú **Insertar | Archivo o esquema XML**), el archivo se convierte en un componente de la asignación. Un **elemento** es la unidad de asignación de menor nivel (por ejemplo, un atributo de un archivo XML o un elemento de tipo simple). Y una **secuencia** es una colección de elementos.

Los **componentes de transformación** transforman datos (p. ej. las funciones) o prestan ayuda durante la transformación (p. ej. las constantes y variables). Para más información consulte la sección Diseño de asignaciones.

Con la ayuda de los componentes de estructura podrá leer datos de archivos y otros orígenes de datos, escribir datos en archivos y otros orígenes o almacenar datos en una fase intermedia del proceso de asignación (p. ej. para obtener una vista previa de los datos). Por tanto, podemos hablar de tres tipos de componentes de estructura:

- **Origen:** un componente se declara como componente de origen al colocarlo en el lado izquierdo del área de asignación (dando a MapForce la orden de leer datos de este componente).
- **Destino:** un componente se declara como componente de destino al colocarlo en el lado derecho del área de asignación (dando a MapForce la orden de escribir datos en este componente).
- **Paso a través:** este tipo especial de componente hace las veces de origen y destino a la vez (para más información consulte el apartado Asignaciones en cadena / componentes de paso a través).

En el área de asignación los componentes toman la forma de rectángulos. Por ejemplo, en esta imagen la asignación está compuesta por tres componentes de origen, un componente XML de destino y varios componentes de transformación (funciones y filtros) por los que se pasan los datos antes de escribirse en el destino.



CompletePO.mfd

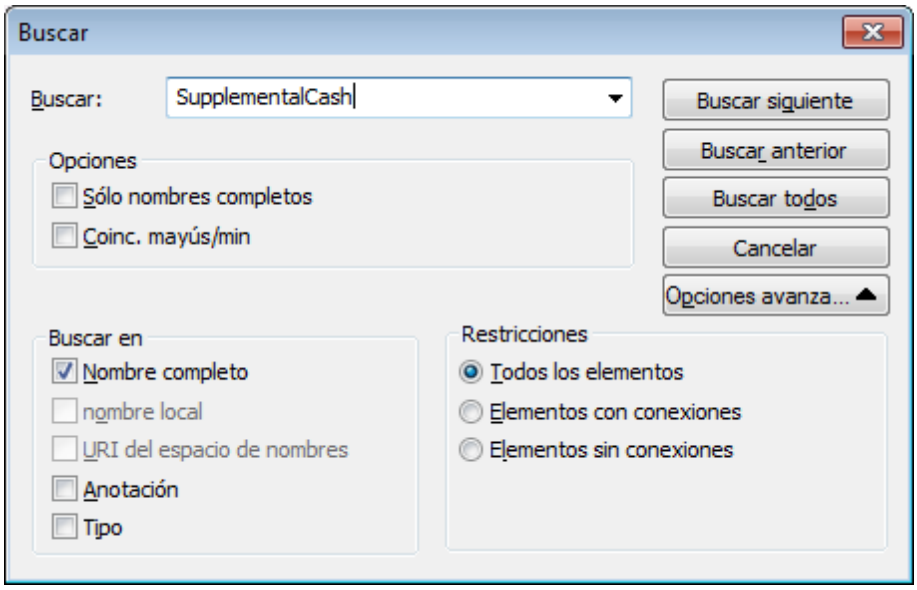
Este ejemplo de asignación está disponible en esta ruta de acceso: `<Documentos>\Altova \MapForce2018\MapForceExamples\CompletePO.mfd`.

4.2.1 Búsquedas dentro de los componentes

Para buscar un elemento o nodo determinado de un componente:

1. Haga clic en el componente donde se debe buscar y pulse **Ctrl+F**.
2. Introduzca el término de búsqueda y haga clic en **Buscar siguiente**.

© 2018 Altova GmbH

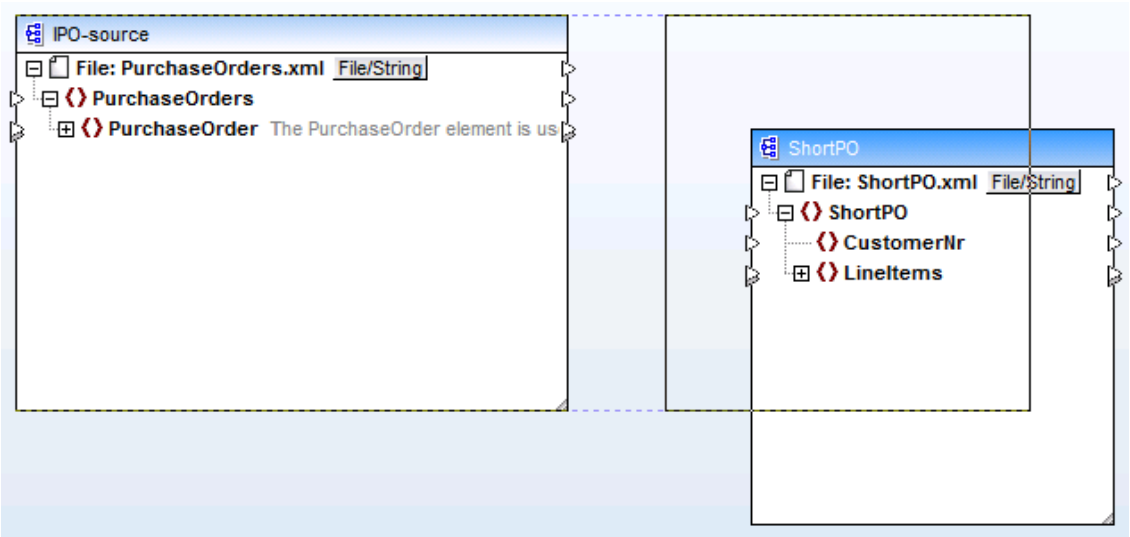


Haga clic en el botón **Opciones avanzadas** para definir en qué tipo de elementos (nodos) se debe realizar la búsqueda y restringir las opciones de búsqueda a determinadas conexiones solamente.

4.2.2 Alinear componentes

Mientras el usuario mueve componentes por el área de asignación MapForce muestra líneas de guía para ayudarle a alinear los componentes del diseño correctamente.

Por ejemplo, en la imagen siguiente el usuario sube el componente situado a la derecha un poco más arriba en el área de asignación. Las líneas de guía indican que el componente que se está moviendo puede alinearse con el componente situado a la izquierda.



Líneas de guía para la alineación de componentes.

Para habilitar o deshabilitar las líneas de guía:

1. En el menú **Herramientas** haga clic en **Opciones**.
2. En la pestaña de opciones **Edición** active/desactive la casilla *Alinear componentes al arrastrarlos con el ratón*.

4.2.3 Cambiar configuración de los componentes

Tras añadir un componente al área de asignación podrá configurarlo desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de varias maneras:

- Seleccionando el componente y haciendo clic en el comando de menú **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y seleccionando **Propiedades** en el menú contextual.

Recuerde que las opciones de configuración de este cuadro de diálogo dependen del tipo de componente. Consulte estos apartados para ver una descripción de las opciones de cada tipo de componente:

- Configuración de componentes XML
- Configuración de componentes de base de datos
- Configuración de componentes CSV
- Configuración de componentes FLF (de longitud fija)
- Configuración de componentes FlexText
- Configuración de componentes JSON
- Configuración de componentes Excel 2007+
- Configuración de componentes XBRL
- Configuración de componentes EDI

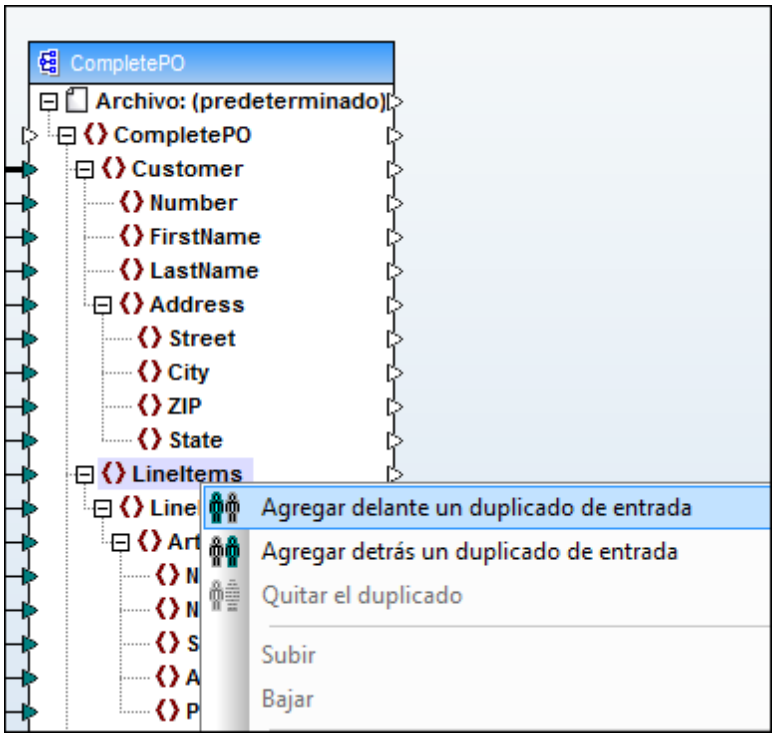
© 2018 Altova GmbH

En los componentes basados en archivos, como los componentes XML por ejemplo, junto al nodo raíz aparece el botón Archivo/Cadena . Este botón sirve para especificar opciones avanzadas cuando se quiera procesar o generar varios archivos de salida en la misma asignación (*véase* *Procesar varios archivos de entrada o salida simultáneamente*). Además, con este botón podrá habilitar opciones avanzadas para el análisis de cadenas y la serialización de datos en cadenas (*véase* *Analizar y serializar cadenas*).

4.2.4 Duplicar entradas

A veces es necesario configurar un componente para que acepte datos de varios orígenes de datos a la vez. Por ejemplo, imagine que necesita convertir datos de dos esquemas XML diferentes y pasarlos a un solo esquema. Para que el esquema de destino acepte los datos de ambos esquemas de origen, lo más fácil es crear duplicados de uno de los elementos de entrada dentro del componente. Los duplicados de entrada están diseñados solamente para los componentes de destino y pueden crearse tantos duplicados de entrada como sean necesarios.

Para crear un duplicado de un elemento de entrada, haga clic con el botón derecho en el elemento y seleccione **Agregar delante/detrás un duplicado de entrada** en el menú contextual.



En la imagen anterior puede ver que el elemento `LineItem` se va a duplicar para poder asignar datos de otro origen de datos más.

Tras crear el duplicado de entrada se pueden realizar conexiones tanto con la entrada original como con el duplicado de entrada. Por ejemplo, puede copiar datos del origen A a la entrada original y datos del origen B al duplicado de entrada.

Nota: no está permitido crear duplicados de atributos XML porque daría lugar a una instancia XML no válida. La duplicación de elementos XML de entrada está permitida independientemente del valor que tenga el atributo `maxOccurs` en el esquema. Esto se permite porque el esquema puede cambiar y porque los datos de origen pueden ser opcionales. Por ejemplo, una asignación podría generar un solo elemento XML incluso si la entrada está duplicada en el diseño de asignación.

Para ver un ejemplo paso a paso consulte el apartado Asignar varios orígenes de datos a un destino.

© 2018 Altova GmbH

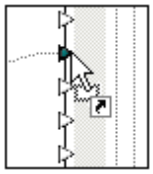
4.3 Trabajar con conexiones

Una asignación de datos tiene como último objetivo transformar datos de un formato a otro o transformar su estructura a otra distinta. Tomemos una asignación muy básica como ejemplo: el usuario añade al área de asignación los componentes que representan sus datos de origen y de destino (p. ej. un esquema XML de origen y uno de destino) y después dibuja las conexiones de asignación entre las dos estructuras. Por tanto, una conexión no es más que la representación visual de la asignación de datos entre el componente de origen y el de destino.

Los componentes tienen entradas y salidas que aparecen representados en la asignación en forma de triángulos y reciben el nombre de *conectores*. Los conectores de entrada están situados a la izquierda de los elementos para los que se puede dibujar una conexión. Los conectores de salida, por otro lado, están situados a la derecha de los elementos para los que se puede dibujar una conexión.

Para dibujar una conexión entre dos elementos:

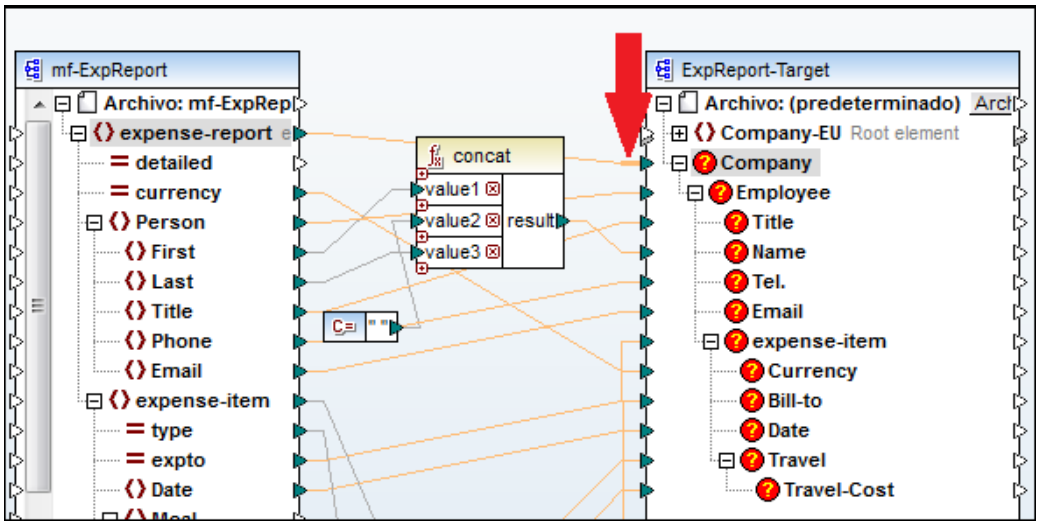
- Haga clic en el conector de salida de un elemento de origen y arrastre el puntero hasta el elemento de destino. Cuando la acción de colocar esté permitida, aparece junto al cursor aparece un icono de enlace.



Cada conector de entrada acepta un máximo de una conexión de entrada. Si intenta agregar otra conexión a la misma entrada, aparece un cuadro de mensaje preguntando si desea reemplazar la conexión con una nueva o crear un duplicado del elemento de entrada. Cada conector de salida, por el contrario, puede tener varias conexiones con diferentes entradas.

Para mover una conexión a otro elemento:

- Haga clic en el extremo de la conexión (es decir, la sección recta situada junto al destino) y arrástrelo hasta el nuevo destino.

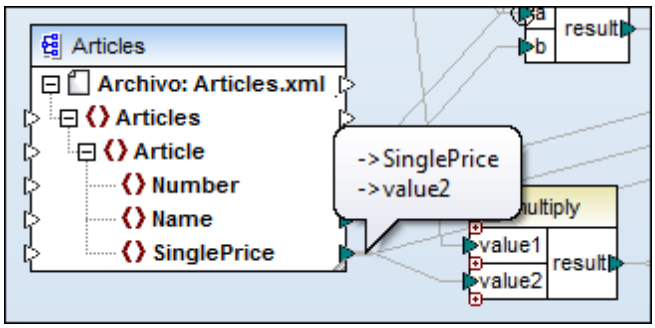


Para copiar una conexión con otro elemento:

- Haga clic en el extremo de la conexión (es decir, la sección recta situada junto al destino) y arrástrelo hasta el nuevo destino mientras pulsa la tecla **Ctrl**.

Para ver el elemento situado en el otro extremo de una conexión:

- Coloque el puntero sobre la sección recta situada junto a un extremo de la conexión. Aparece un mensaje de información rápida con el nombre de los elementos situados en el otro extremo de la conexión. Si se definieron varias conexiones desde la misma salida, la información rápida ofrece un máximo de diez nombres de elemento. En la imagen siguiente, por ejemplo, los dos elementos de destino de la función de multiplicación son `SinglePrice` y `value2`.



Para cambiar la configuración de la conexión tiene tres opciones:

- En el menú **Conexión** haga clic en el comando **Propiedades** (este comando solamente se habilita si tiene seleccionada una conexión).
- Haga doble clic en la conexión.
- Haga clic con el botón derecho en la conexión y después elija **Propiedades** en el menú contextual.

 © 2018 Altova GmbH

Consulte el apartado Configuración de las conexiones para obtener más información.

Para eliminar una conexión tiene dos opciones:

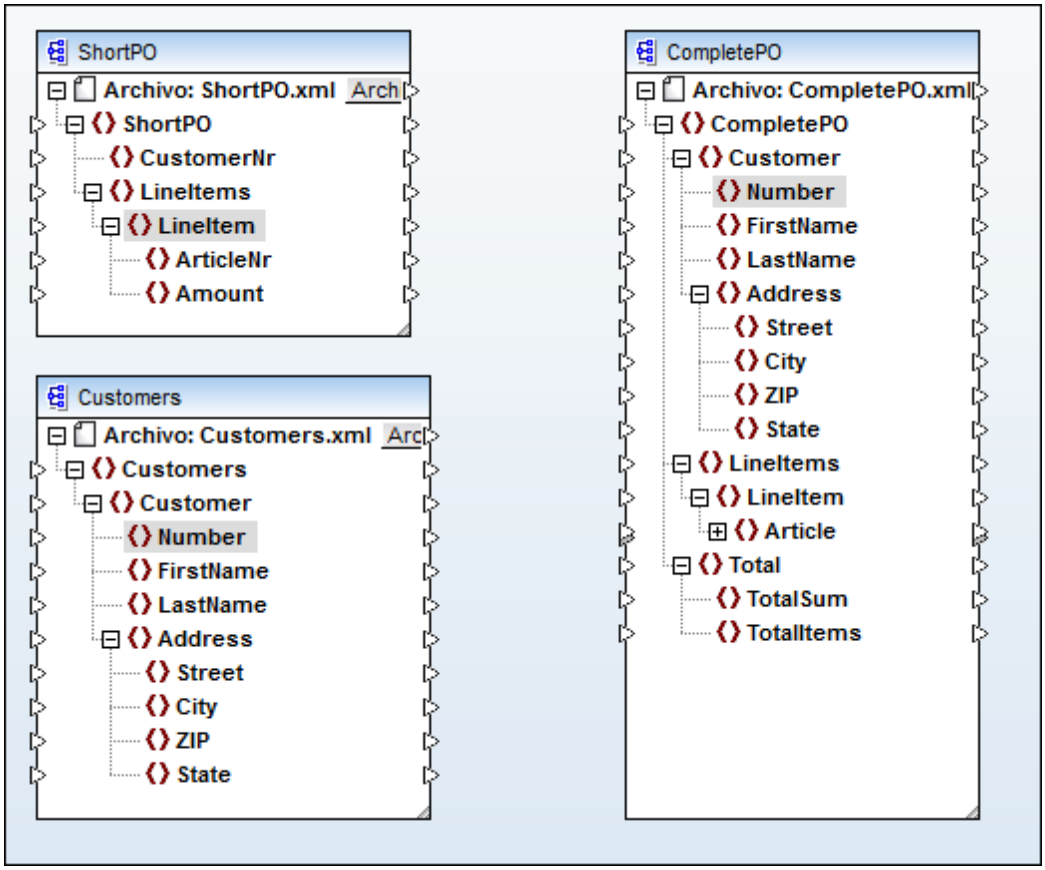
- Haga clic en la conexión y pulse la tecla **Supr**.
- Haga clic con el botón derecho en la conexión y elija **Eliminar** en el menú contextual.

4.3.1 Nota sobre entradas obligatorias

MapForce resalta en color naranja las entradas obligatorias de los componentes de destino:

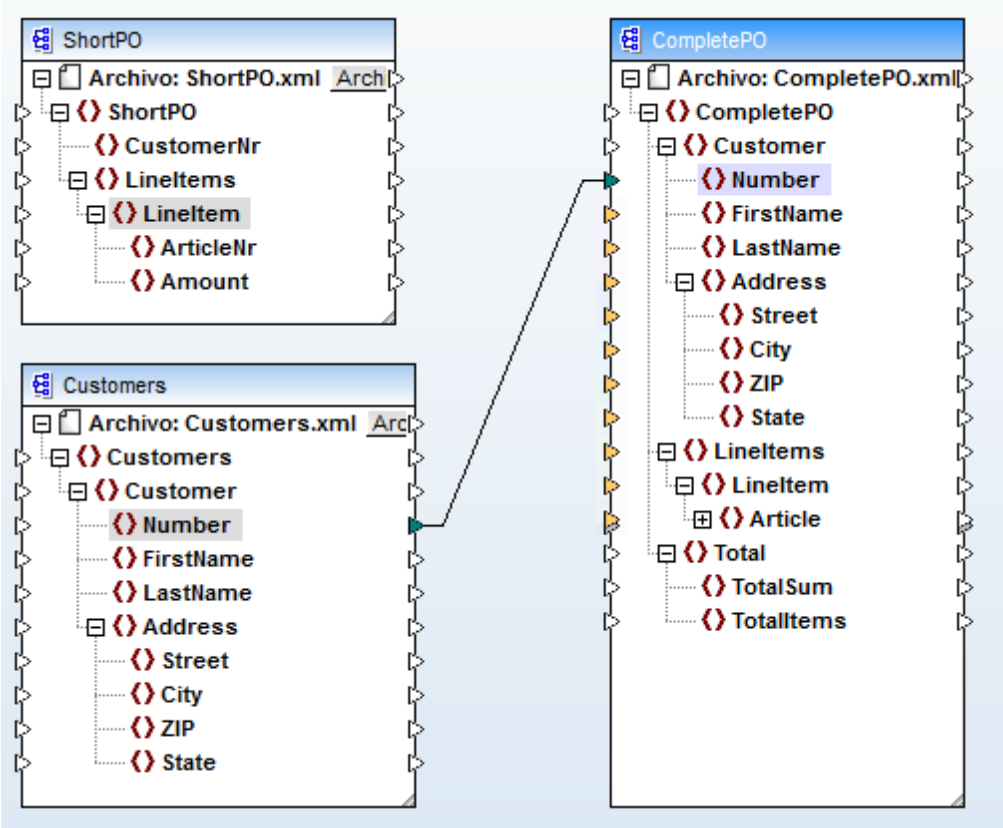
- En los componentes XML y EDI son entradas obligatorias los elementos cuyo parámetro `minOccurs` tiene un valor igual o mayor que 1.
- En las bases de datos son entradas obligatorias los campos definidos como "not null".
- Todos los nodos de las llamadas y respuestas WSDL.
- Todos los nodos XBRL definidos como obligatorios.
- En las funciones, una vez creada una asignación con un parámetro, MapForce resalta los parámetros obligatorios para señalar que es necesario realizar una conexión. Por ejemplo: una vez creada la asignación con los parámetros de entrada de un filtro, el otro parámetro se resalta automáticamente.
- En las hojas de cálculo MS Excel son entradas obligatorias todos los nombres de hoja de cálculo.

Ejemplo (asignación de datos CompletePO.mfd, disponible en la carpeta . . . `\MapForceExamples`)
Cuando creamos una asignación como la de este archivo de ejemplo, los archivos de esquema XML insertados tienen este aspecto:



Después conectamos el elemento `Number` del componente **`Customers`** con el elemento `Number` del componente **`CompletePO`**. Nada más crearse la conexión, MapForce resalta los elementos/nodos obligatorios del componente **`CompletePO`**. Observe que también se resalta el nodo/icono contraído `Article`.

© 2018 Altova GmbH



4.3.2 Opciones de presentación de las conexiones

Puede elegir qué tipo de conexiones aparecen en la ventana de asignación.



El botón **Mostrar los conectores del componente seleccionado** permite alternar entre:

- todos los conectores de la asignación, que aparecen en negro y
- todos los conectores relacionados con el componente seleccionado, que aparecen en negro. Los demás conectores aparecen atenuados.



El botón **Mostrar los conectores desde su origen a su destino** permite alternar entre:

- los conectores que están conectados con el componente seleccionado **directamente** y
- los conectores que están unidos al componente seleccionado, que parten del origen y terminan en los componentes de destino.

4.3.3 Crear anotaciones en las conexiones

MapForce ofrece la posibilidad de etiquetar conexiones para poder comentar la asignación en detalle. Esta característica está disponible para **todos los tipos de conexiones**.

Para anotar una conexión:

1. Haga clic con el botón derecho en la conexión y elija **Propiedades** en el menú contextual.
2. En el campo *Descripción* introduzca un nombre para la conexión seleccionada. Al introducir el texto se habilitan las opciones del grupo *Configuración de la anotación*.
3. Defina la *ubicación*, la *alineación* y la *posición* de la etiqueta.
4. Para ver el texto de la anotación active el icono **Mostrar anotaciones** en la barra de herramientas Opciones de vista.

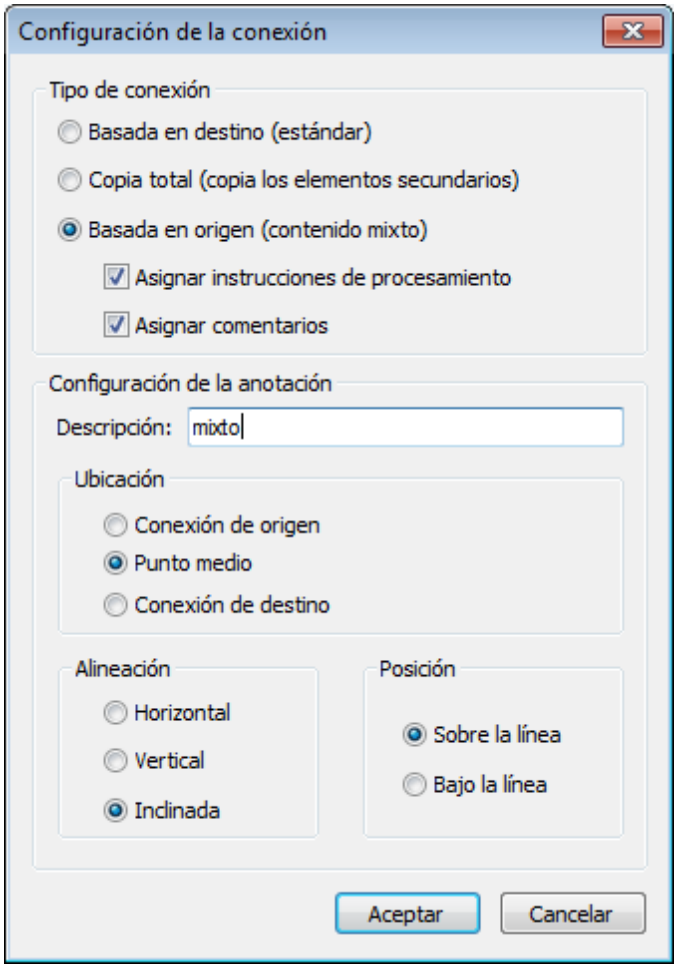


Nota: puede ver el texto de la notación aunque el icono **Mostrar anotaciones** no esté activo: pase el cursor por encima de la conexión y el texto de la anotación aparecerá en un mensaje emergente, pero solamente si está activo el icono **Mostrar información rápida** de la barra de herramientas Opciones de vista.

4.3.4 Configuración de las conexiones

El cuadro de diálogo "Configuración de la conexión" se abre al hacer doble clic en una conexión o al hacer clic con el botón derecho en la conexión y elegir el comando **Propiedades** en el menú contextual. En este cuadro de diálogo puede definir opciones de configuración de la conexión (excepto aquellas que no estén disponibles, que aparecerán deshabilitadas).

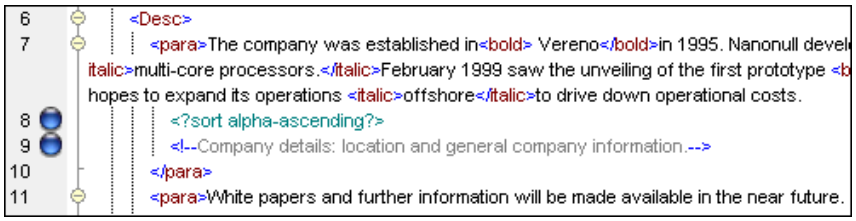
© 2018 Altova GmbH



Cuadro de diálogo "Configuración de la conexión"

En el caso de elementos de tipo `complexType` podrá elegir entre los tipos de conexión que aparecen en esta tabla (es el caso también de los elementos de tipo `complexType` que no tienen nodos de texto):

Basada en destino (estándar)	Cambia la conexión a un tipo de conexión basada en destino (*véase* *Conexiones basadas en el destino*).
Copia total (copia los elementos secundarios)	Cambia la conexión a un tipo de conexión de copia total y conecta automáticamente todos los elementos que sean idénticos en el componente de origen y destino (*véase* *Conexiones de copia total*).
Basada en origen (contenido mixto)	Cambia la conexión a un tipo de conexión basada en origen y habilita la selección de más elementos. Estos elementos adicionales deben ser elementos secundarios del elemento del archivo XML de origen que se está conectando o de lo contrario no se podrán conectar. Marque las casillas *Asignar instrucciones de procesamiento* y *Asignar comentarios* para incluir estos grupos de datos en el archivo de salida.

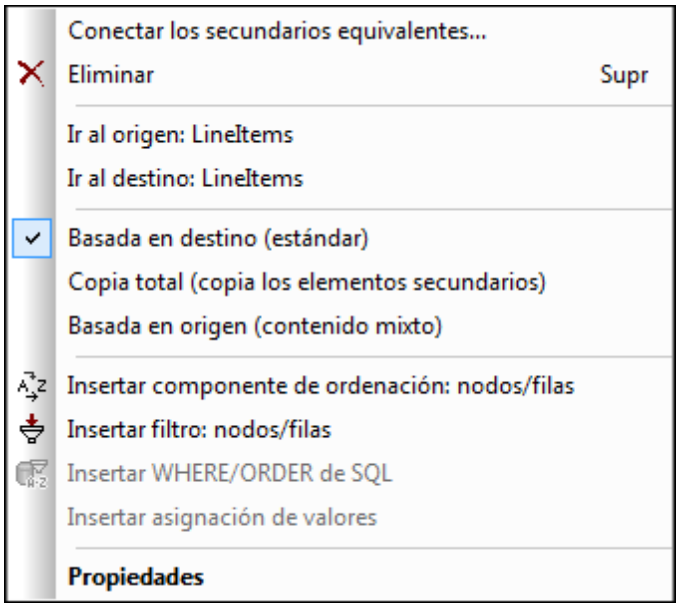


Nota: las secciones CDATA se tratan como si fueran texto.

En el grupo de opciones *Configuración de la anotación* puede especificar cómo se debe anotar la conexión (*véase* *Crear anotaciones en las conexiones*).

4.3.5 Menú contextual de las conexiones

Cuando se hace clic con el botón derecho en una conexión aparece este menú contextual:



Conectar lo secundarios equivalentes...	Abre el cuadro de diálogo "Conectar los secundarios equivalentes" (*véase* *Conectar los secundarios equivalentes*). Este comando se habilita si la conexión puede tener secundarios equivalentes.
Eliminar	Elimina la conexión seleccionada.
Ir al origen <nombre del elemento>	Selecciona el conector de origen de la conexión seleccionada.

© 2018 Altova GmbH

Ir al destino <nombre del elemento>	Selecciona el conector de destino de la conexión seleccionada.
Basada en destino (estándar)	Cambia la conexión a un tipo de conexión basada en destino (*véase Conexiones basadas en el destino*).
Copia total (copia lo elementos secundarios)	Cambia la conexión a un tipo de conexión de copia total y conecta automáticamente todos los elementos que sean idénticos en el componente de origen y destino (*véase Conexiones de copia total*). Este comando solamente se habilita si el elemento de origen y el elemento de destino tienen secundarios.
Basada en origen (contenido mixto)	Cambia la conexión a un tipo de conexión basada en origen (*véase Conexiones basadas en el origen*). Este comando solamente se habilita si el elemento de origen y el elemento de destino tienen secundarios.
Insertar componente de ordenación: nodos/filas	Añade un componente de ordenación entre el elemento de origen y el elemento de destino (*véase Ordenar datos*).
Insertar filtro: nodos/filas	Añade un componente de filtrado entre el elemento de origen y el elemento de destino (*véase Filtros y condiciones*).
Insertar WHERE/ORDER de SQL	Añade un componente WHERE de SQL entre el elemento de origen y el elemento de destino (*véase Componente WHERE/ ORDER de SQL*)..
Insertar asignación de valores	Añade un componente de asignación de valores entre el elemento de origen y el elemento de destino (*véase Usar asignaciones de valores*).
Propiedades	Abre el cuadro de diálogo "Configuración de la conexión" (*Configuración de las conexiones*).

4.3.6 Conectar los secundarios equivalentes

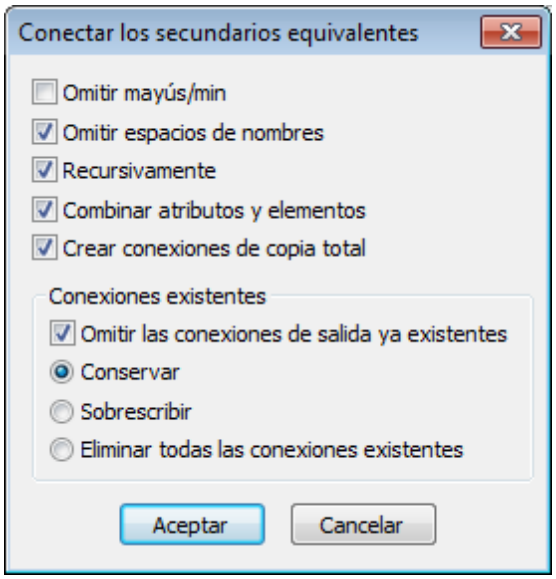
Puede crear varias conexiones entre elementos que tengan el mismo nombre en el componente de origen y de destino. No obstante, debe tener en cuenta que la aplicación crea por defecto una conexión de copia total (*véase Conexiones de copia total*).

Para activar/desactivar la opción Conectar automáticamente los secundarios equivalentes:

- Haga clic en el botón **Conectar automáticamente los secundarios equivalentes**  de la barra de herramientas.
- En el menú **Conexión** haga clic en el comando **Conectar automáticamente los secundarios equivalentes**.

Para configurar la opción Conectar los secundarios equivalentes:

1. Conecte dos elementos (de nivel superior) que tengan **elementos secundarios** con nombres idénticos en el componente de origen y de destino.
2. Haga clic con el botón derecho en la línea de conexión y elija el comando **Conectar los secundarios equivalentes** del menú contextual.



3. Elija la configuración correspondiente (*ver tabla más abajo*) y haga clic en **Aceptar**. La aplicación crea conexiones para todos los elementos secundarios que tengan nombres idénticos utilizando la configuración definida en el cuadro de diálogo anterior.

Nota: la configuración que defina en este cuadro de diálogo se utiliza al crear una conexión entre dos elementos si está activado el botón **Conectar automáticamente los secundarios equivalentes** de la barra de herramientas.

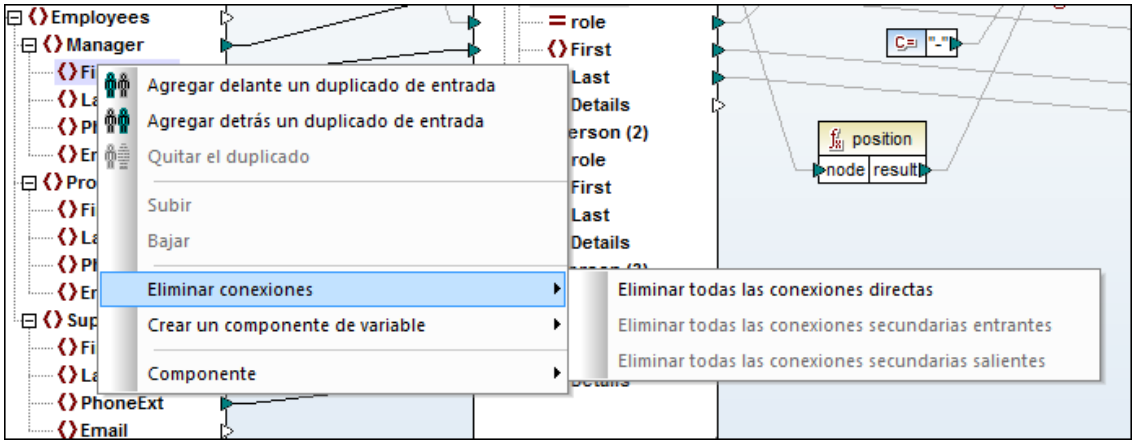
Omitir mayús/min	No tiene en cuenta el uso de mayúsculas y minúsculas a la hora de conectar secundarios equivalentes.
Omitir espacios de nombres	No tiene en cuenta los espacios de nombres a la hora de conectar secundarios equivalentes.
Recursivamente	Crea conexiones nuevas entre los secundarios equivalentes de forma recursiva. Es decir, se crea una conexión independientemente del nivel en el que los elementos estén anidados en la jerarquía, siempre y cuando tengan nombres idénticos.
Combinar atributos y elementos	Si marca esta casilla, se crearán conexiones entre atributos y elementos que tengan nombres idénticos. Por ejemplo, se puede crear una conexión si existen dos ítems llamados `Name`, aunque uno sea un elemento y el otro un atributo.
Crear conexiones de copia	Esta casilla está activada por defecto. La aplicación crea

© 2018 Altova GmbH

total	una conexión de copia total entre los elementos de origen y destino, siempre que sea posible.
Omitir las conexiones de salida ya existentes	Crea más conexiones para todos los secundarios equivalentes, aunque ya tengan conexiones de salida.
Conservar	Conserva las conexiones ya existentes.
Sobrescribir	Vuelve a crear conexiones basadas en la configuración definida. Las conexiones ya existentes se descartan.
Eliminar todas las conexiones existentes	Elimina todas las conexiones existentes antes de crear conexiones nuevas.

Eliminar conexiones

Las conexiones creadas con ayuda del cuadro de diálogo "Conectar los secundarios equivalentes" o durante el proceso de asignación pueden eliminarse en grupo.



Para eliminar conexiones:

1. Haga clic con el botón derecho en el nombre de un elemento (no en la conexión) dentro del componente.
2. Seleccione el comando **Eliminar conexiones | Eliminar todas las conexiones...** que corresponda.

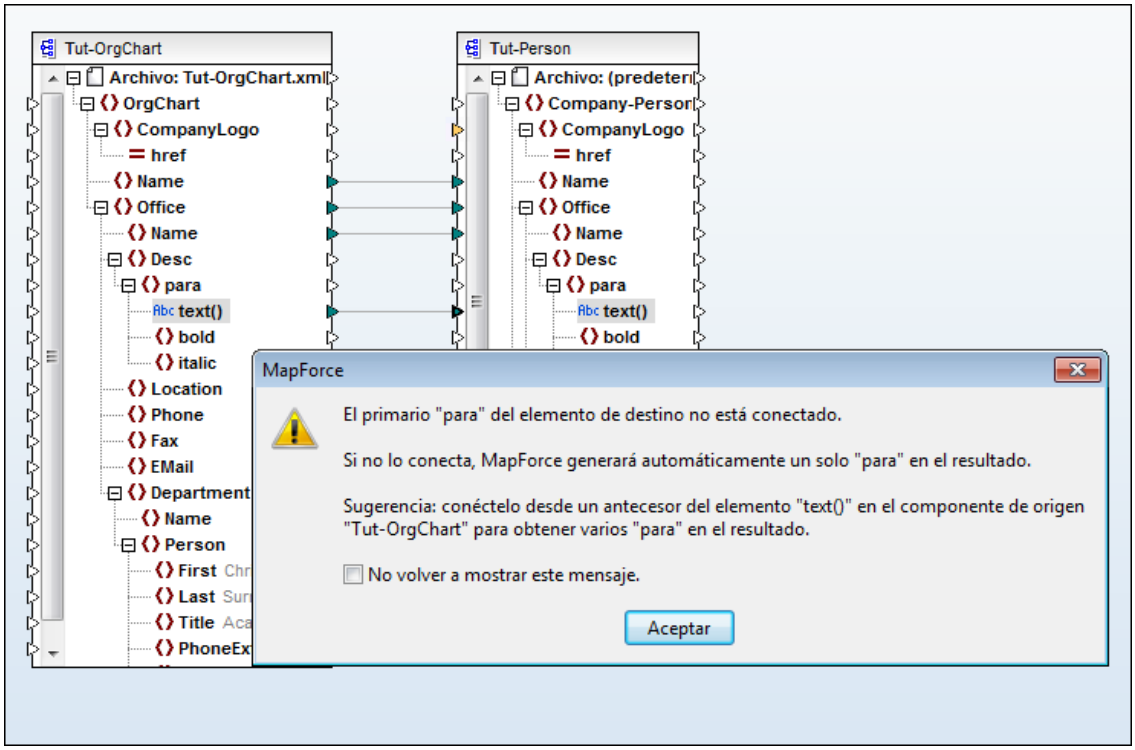
Eliminar todas las conexiones directas	Elimina todas las conexiones directas salientes o entrantes del componente actual con los demás componentes de origen o destino.
Eliminar todas las conexiones secundarias entrantes	Este comando solamente está disponible si se ejecuta sobre un elemento de un componente de destino. Elimina todas las conexiones secundarias entrantes.
Eliminar todas las conexiones secundarias salientes	Este comando solamente está disponible si se ejecuta sobre un elemento de un componente de origen. Elimina todas las conexiones secundarias salientes.

4.3.7 Notificaciones sobre conexiones antecesoras ausentes

Cuando se crean conexiones entre elementos de origen y destino de forma manual, MapForce analiza automáticamente los posibles resultados de la asignación. Si se crea una asignación entre dos elementos secundarios, MapForce puede emitir una notificación para sugerir una conexión entre el elemento primario del elemento de origen y el elemento primario del elemento de destino.

Esta notificación le ayudará a evitar que en la ventana de vista previa de resultados aparezca un solo elemento secundario. Esto es lo que suele ocurrir si el nodo de origen ofrece una secuencia en lugar de un valor.

Veamos un ejemplo en la asignación de muestra llamada **Tut-OrgChart.mfd** (disponible en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`). Si conecta el elemento de origen `text()` con el elemento de destino `text()`, aparece una notificación advirtiendo que el elemento primario `para` no está conectado y que solamente se generará una vez en el resultado de la asignación.



Tut-OrgChart.mfd (MapForce Basic Edition)

Para generar varios elementos `para` en el resultado debemos conectar los elementos `para` de origen y destino.

Si prefiere deshabilitar este tipo de notificaciones:

1. En el menú **Herramientas** seleccione el comando **Opciones**.
2. Haga clic en el grupo **Mensajes**.

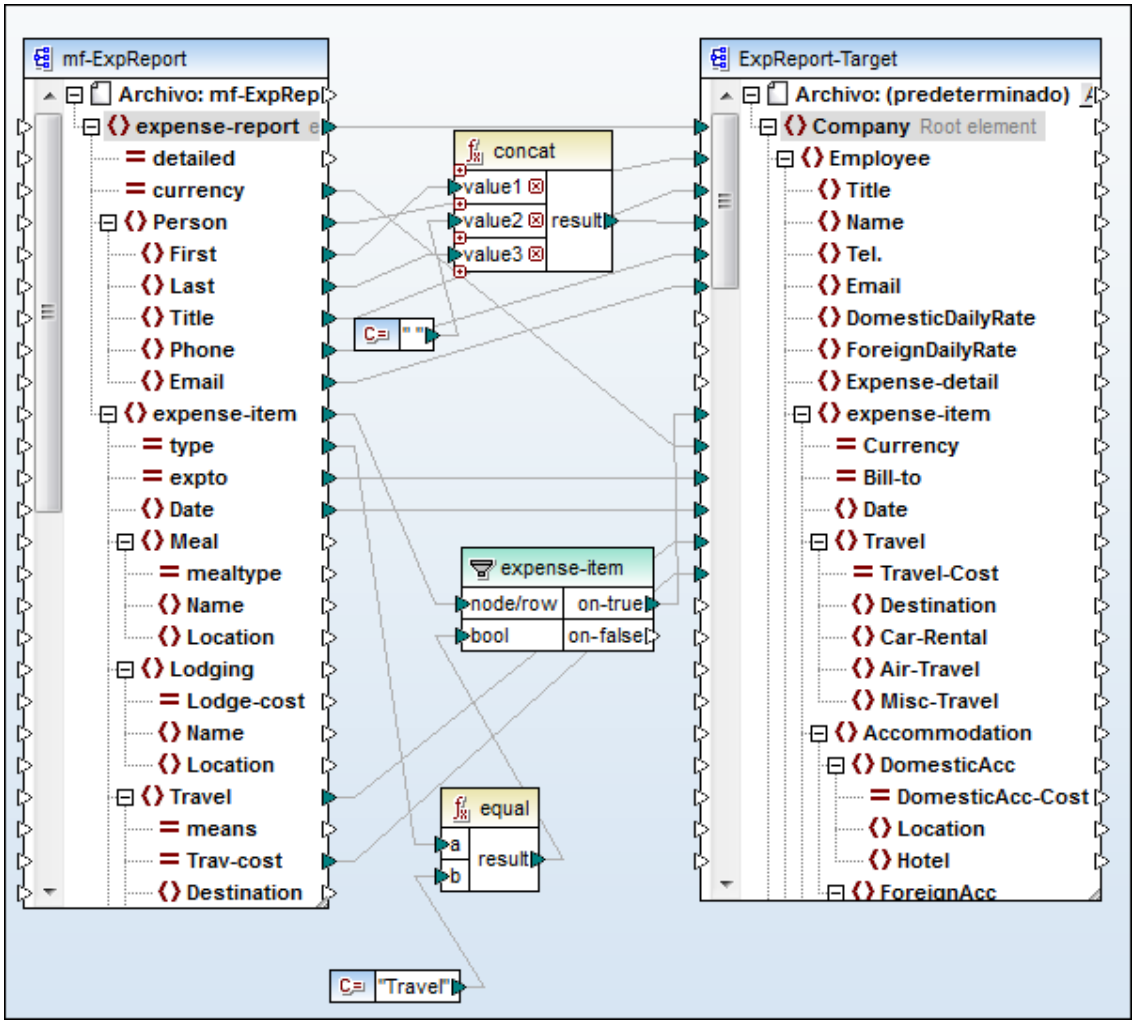
© 2018 Altova GmbH

3. Desactive la casilla *Al crear una conexión nueva, sugerir conexión de elementos antecesores*.

4.3.8 Mover conexiones y conexiones secundarias

Cuando el usuario mueve una conexión a otro componente, MapForce encuentra conexiones secundarias idénticas y pregunta si también deben moverse a la nueva posición. Esta característica puede ser útil si ya tiene un diseño de asignación y cambia el elemento raíz del esquema de destino. Cuando esto ocurre, lo normal es que sea necesario reasignar todas las conexiones descendientes a mano. Gracias a esta característica podrá evitar este tipo de situaciones.

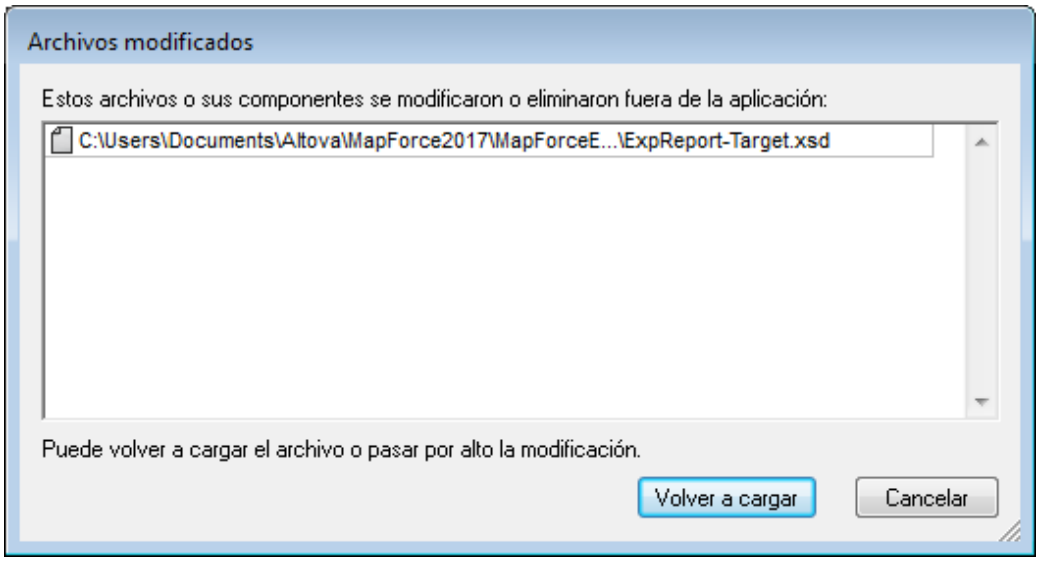
Veamos un ejemplo en la asignación de muestra **Tut-ExpReport.mfd** (disponible en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`).



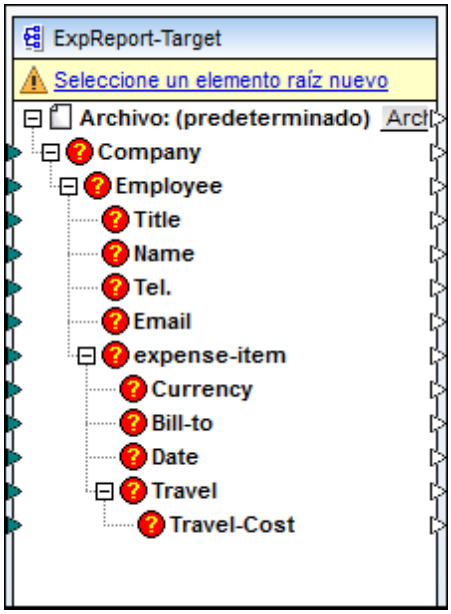
Tut-ExpReport.mfd (MapForce Basic Edition)

Hagamos una prueba para comprender el funcionamiento de esta característica:

1. Abra la asignación de muestra **Tut-ExpReport.mfd**.
2. Edite el esquema **ExpReport-Target.xsd** fuera de MapForce, cambiando el elemento raíz `Company` del esquema de destino por `Company-EU`. No es necesario cerrar MapForce para realizar este cambio.
3. Tras cambiar el elemento raíz `Company` del esquema de destino por `Company-EU`, MapForce emite esta notificación:



4. Haga clic en el botón **Volver a cargar** para volver a cargar el esquema que modificó. Como el elemento raíz se eliminó, el componente muestra varios nodos ausentes.

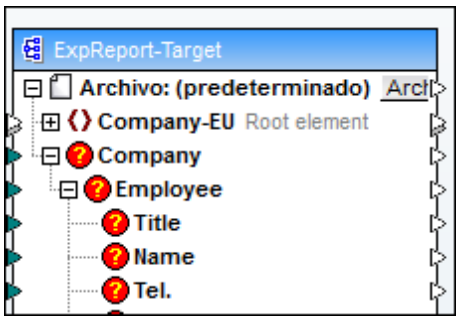


5. Haga clic en **`Seleccione un elemento raíz nuevo`** en la parte superior del componente. (También puede cambiar de elemento raíz haciendo clic con el botón derecho en el título del componente y seleccionando **Cambiar de elemento raíz** en el menú contextual.)

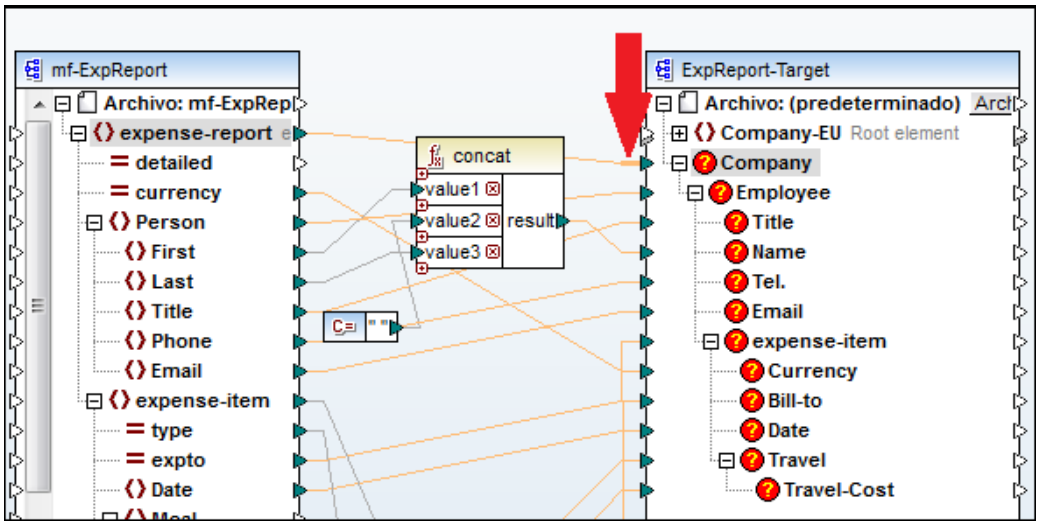
© 2018 Altova GmbH



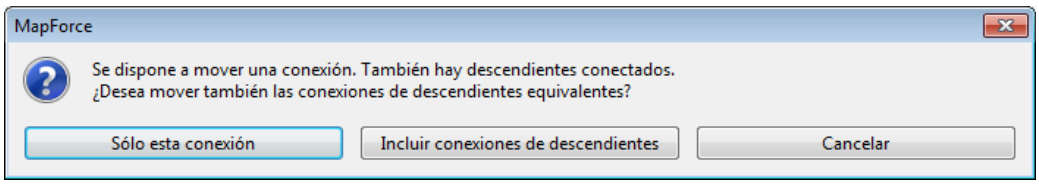
6. Seleccione `Company-EU` como nuevo elemento raíz y haga clic en **Aceptar** para confirmar. Ahora el elemento raíz `Company-EU` aparece al principio del componente.



7. Haga clic en el extremo de destino de la conexión que une el elemento `expense-report` del componente de origen y el elemento `Company` del elemento de destino. Después arrástrelo hasta el elemento raíz `Company-EU` del componente de destino.



Ahora aparece una notificación.



8. Haga clic en **Incluir conexiones de descendientes**. Esto da la orden para que

MapForce reasigne los elementos secundarios correctos bajo el nuevo elemento raíz y la asignación vuelva a ser válida.

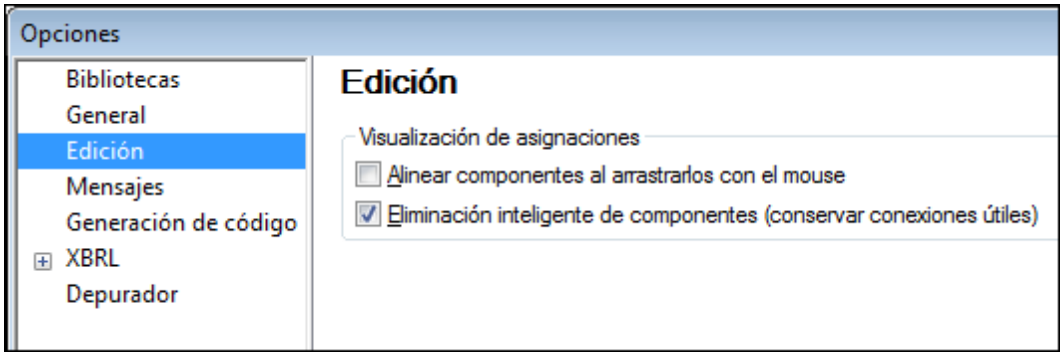
Nota: si el nodo con el que crea la asignación tiene el mismo nombre que el nodo de origen pero un espacio de nombres distinto, la notificación incluirá un botón más llamado **Incluir descendientes y asignar espacio de nombres**. Haga clic en este botón para mover las conexiones secundarias que estén en el mismo espacio de nombres que el nodo primario de origen a los mismos nodos secundarios situados bajo el nodo que tiene diferente espacio de nombres.

4.3.9 Conservar conexiones tras eliminación de componentes

Tiene la opción de elegir qué ocurre cuando se elimina un componente que tiene varias conexiones (secundarias) con otro componente, como cuando elimina un filtro o un componente de ordenación, por ejemplo. Esto puede ser de gran utilidad si quiere conservar todas las conexiones secundarias para no tener que restaurarlas una a una.

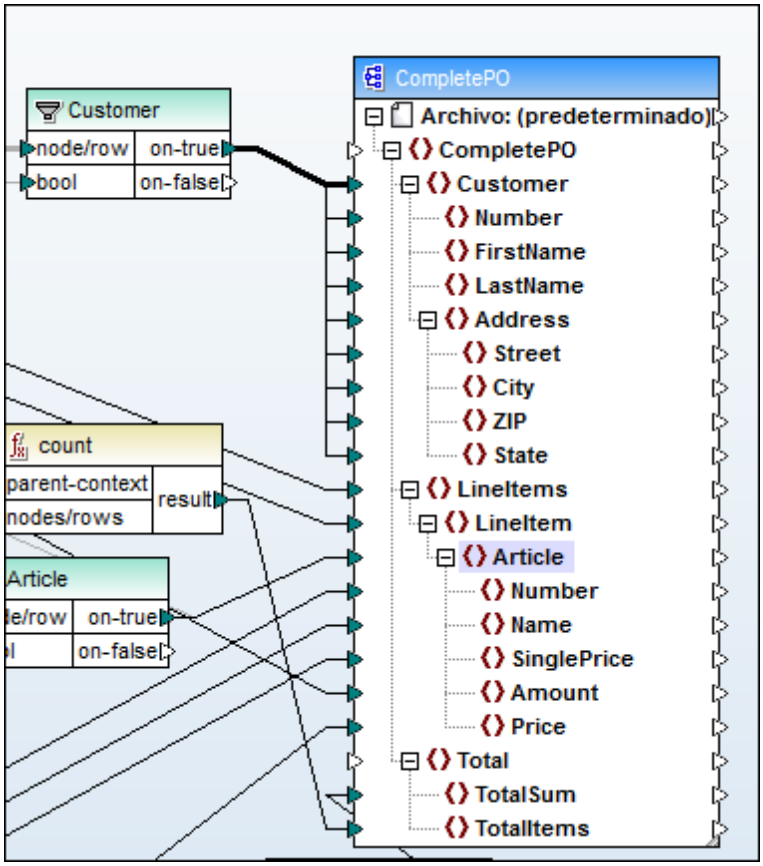
Tras eliminar el componente tendrá la opción de conservar o restaurar las conexiones secundarias o de eliminar todas inmediatamente.

Seleccione la pestaña **Edición** del cuadro de diálogo "Opciones" para ver la configuración actual. En la configuración predeterminada la casilla *Eliminación inteligente de componentes (conservar conexiones útiles)* está desactivada.

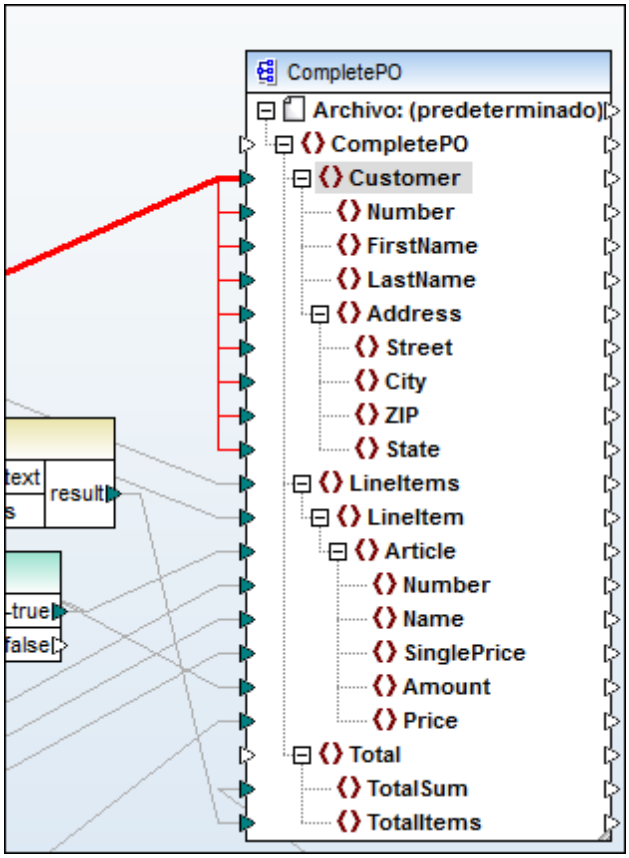


Veamos un ejemplo en la asignación de muestra **CompletePO.mfd** (disponible en la carpeta `...\MapForceExamples`) con la casilla *Eliminación inteligente de componentes (conservar conexiones útiles)* **activada**. Como puede ver en la imagen siguiente, el filtro **`Customer`** es una conexión de **copia total** que tiene varios elementos secundarios conectados.

© 2018 Altova GmbH



Si eliminamos el filtro **`Customer`** aparece una notificación preguntando si desea eliminar este componente. Si hace clic en **Sí**, el filtro se elimina pero las conexiones secundarias se conservan.



Observe que los conectores que se conservan todavía están seleccionados (es decir, están marcados en color rojo). Si desea eliminarlos, basta con pulsar la tecla **Supr**.

Haga clic en cualquier parte de la asignación para dejar de seleccionar los conectores.

Si la casilla *Eliminación inteligente de componentes (conservar conexiones útiles)* estuviera **desactivada**, al eliminar el filtro se eliminarían también todas sus conexiones secundarias.

Nota: si un componente de filtrado tiene conectadas las dos salidas (**`on-true`** y **`on-false`**), entonces se conservan las conexiones secundarias de ambas salidas.

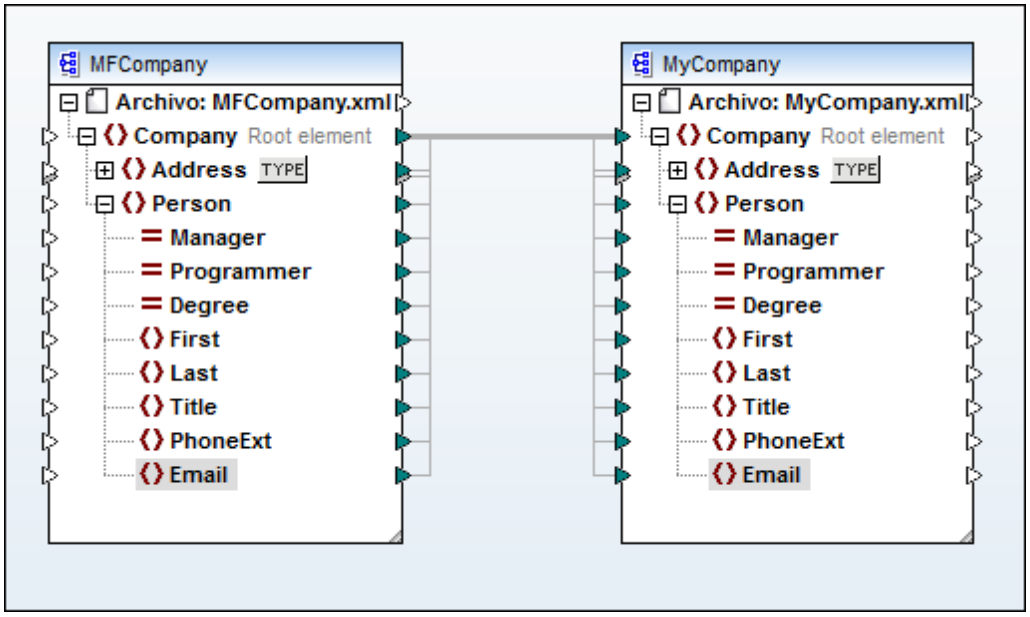
4.3.10 Nota sobre elementos ausentes

A medida que pasa el tiempo lo más probable es que la estructura de uno de los componentes de la asignación de datos sufra cambios (p. ej. es posible que se añadan o eliminen elementos o atributos en un esquema XML). MapForce utiliza elementos marcadores de posición para conservar todos los conectores y todos los datos de conexión entre componentes.

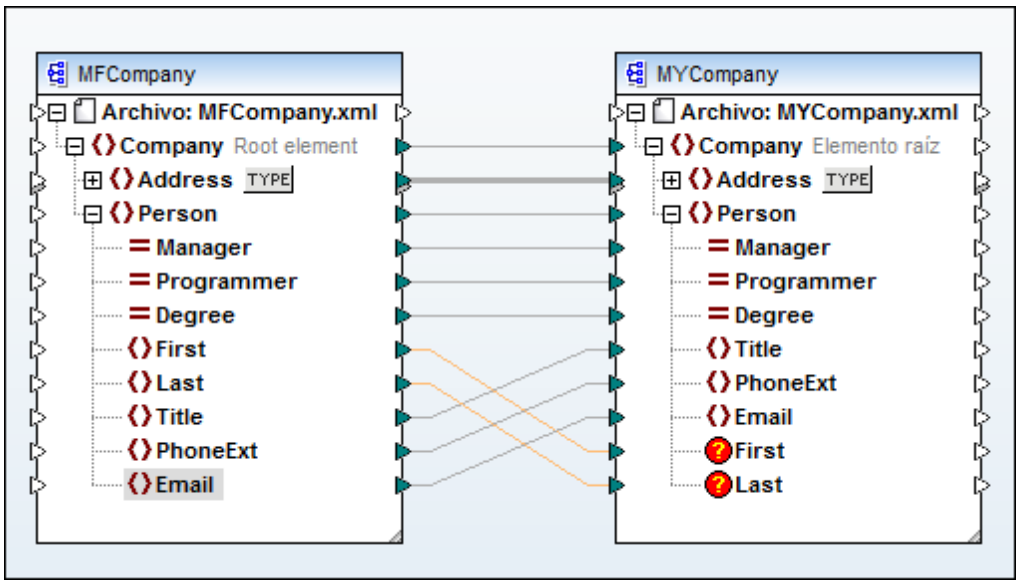
Ejemplo

Para este ejemplo usaremos el archivo de esquema **MFCompany.xsd**. Cambiemos el nombre del esquema por `MyCompany.xsd` y creemos un conector entre el elemento `Company` disponible en ambos esquemas. Esta operación crea conectores para todos los elementos secundarios de los componentes (si está activada la opción **Conectar los secundarios equivalentes**).

© 2018 Altova GmbH



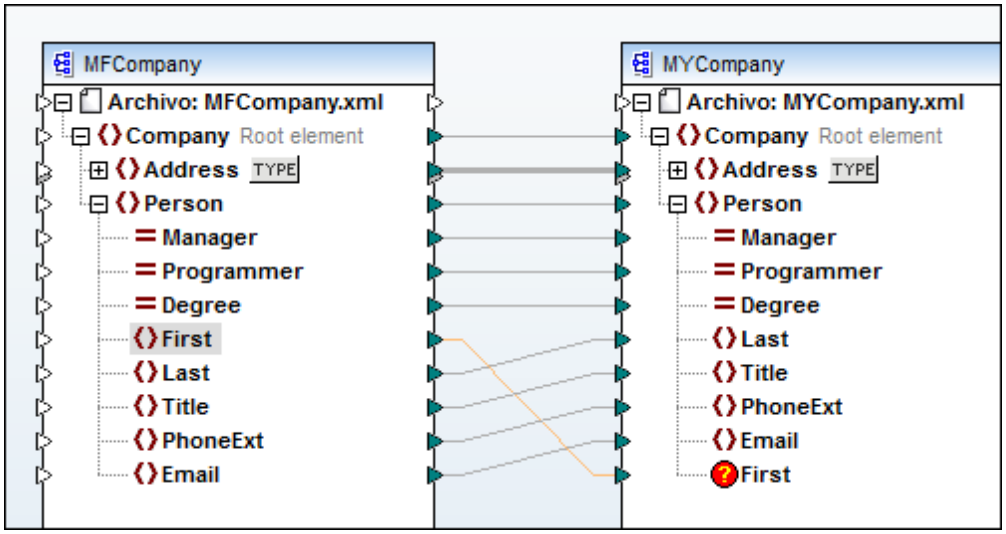
Mientras editamos `MyCompany.xsd` en XMLSpy, eliminamos los elementos `First` y `Last` del esquema. Al volver a MapForce aparece una notificación solicitando que volvamos a cargar el esquema. Al hacer clic en **Volver a cargar** se actualizan los componentes en MapForce.



Ahora los elementos eliminados y sus conectores aparecen marcados en el componente **MyCompany**. Ahora tenemos la opción de volver a conectar los conectores con otros elementos o eliminarlos.

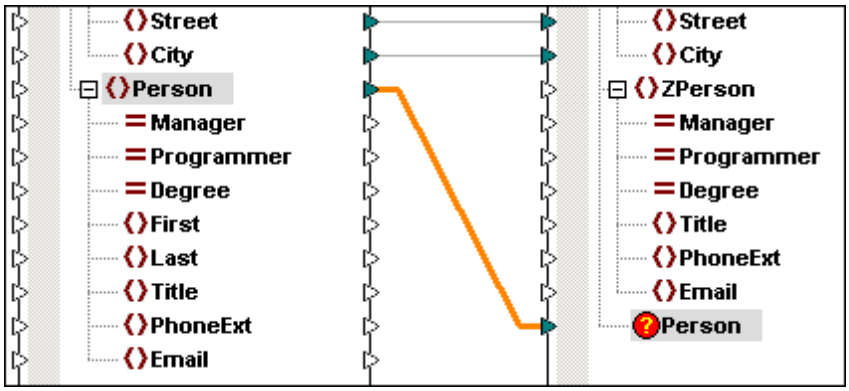
Recuerde que podrá obtener una vista previa de la asignación (y generar código) pero la ventana Mensajes emitirá advertencias si lo hace en este momento. Todas las conexiones entrantes y salientes de elementos ausentes se omiten durante la generación de código y de vista previa.

si hace clic en un conector resaltado y lo elimina, el elemento ausente se elimina del componente (p. ej. `Last` en el componente **MyCompany**).



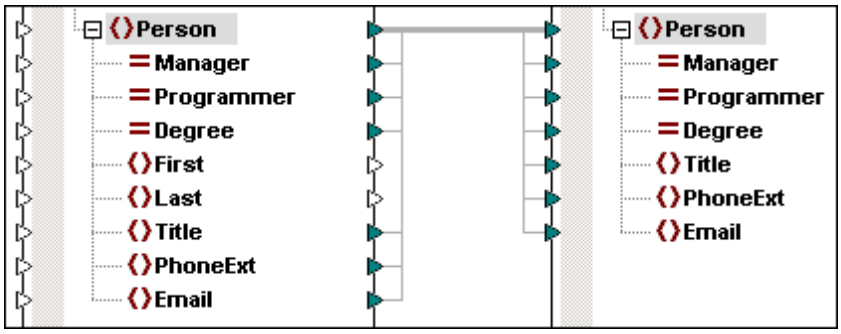
Cambiar el nombre de los elementos

Si cambia el nombre de un elemento primario (p. ej. cambiamos `Person` por `ZPerson`), entonces se conserva el conector del antecesor original pero sus elementos secundarios y conectores se eliminan.



Conectores de copia total y elementos ausentes

Las conexiones de copia total se tratan como si fueran conexiones normales. La única diferencia es que lo conectores con elementos secundarios ausentes no se conservan ni señalan.

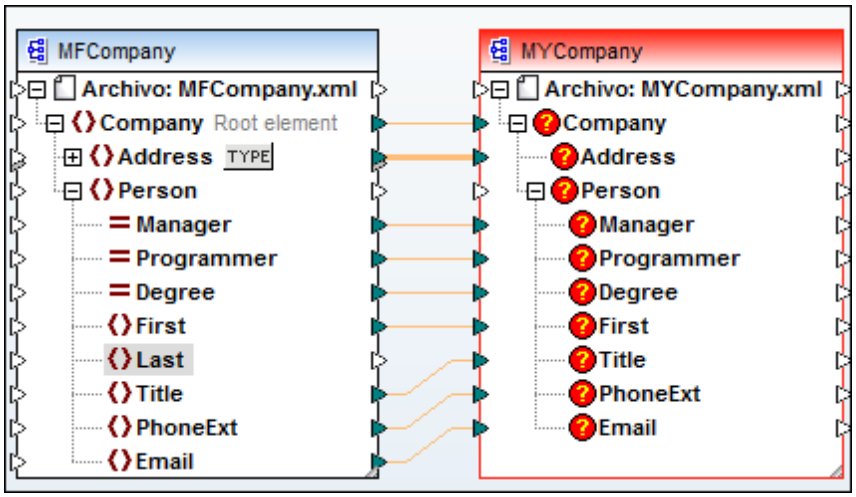


Orígenes de datos renombrados o eliminados

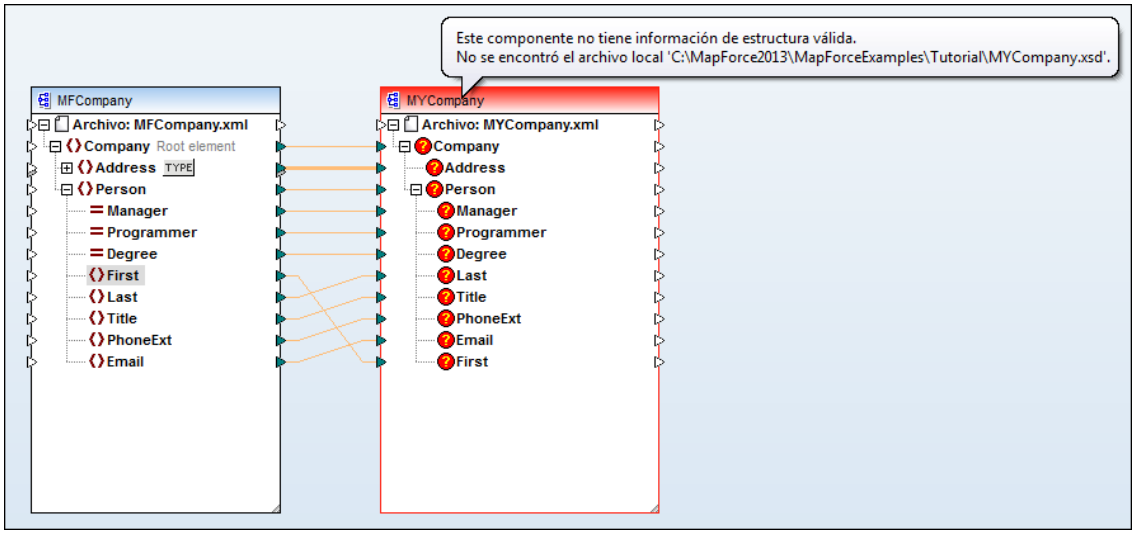
Si el **origen de datos** de un componente (es decir, el esquema, la base de datos, etc.) cambió de nombre o se eliminó, entonces la aplicación resalta todos los elementos que contenía.

© 2018 Altova GmbH

Cuando la aplicación marque el recuadro de un componente en color rojo, significa que falta una conexión válida a un archivo de esquema o de base de datos, lo cual impide la generación de la vista previa y de código.

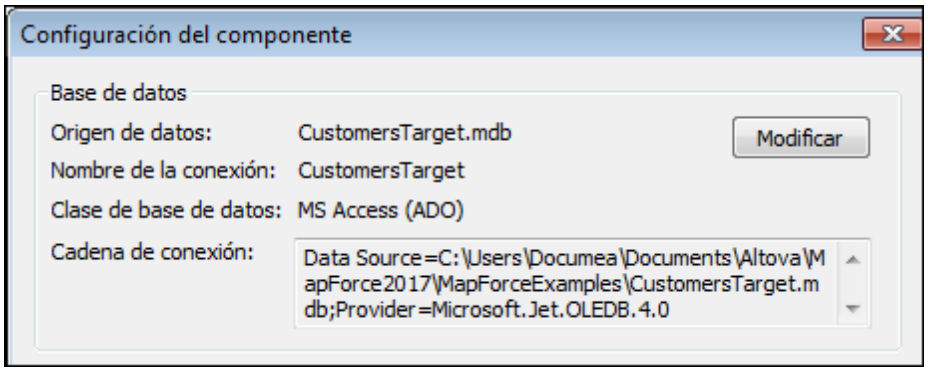


Al pasar el cursor por encima del componente resaltado aparece un mensaje de información rápida.



Al hacer doble clic en la barra de título del componente resaltado aparece el cuadro de diálogo "Configuración del componente". Haga clic en el botón **Examinar** del grupo de opciones *Archivo de esquema* para seleccionar un esquema distinto o una copia de seguridad del esquema original. Para más información consulte la sección dedicada al menú de comandos **Componente**.

Si el componente es una base de datos, el cuadro de diálogo incluye el botón **Modificar**. Este botón sirve para seleccionar una base de datos distinta o cambiar las tablas que aparecen en el componente. La aplicación conservará los conectores de las tablas que se llamen igual.

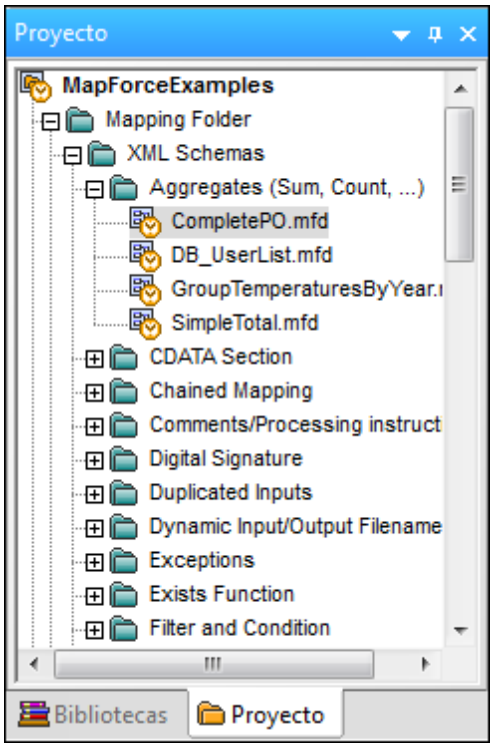


La aplicación conservará todas las conexiones válidas/correctas (y, si se trata de un componente de BD, todos los datos relevantes de la BD) si selecciona un esquema o base de datos con la misma estructura.

© 2018 Altova GmbH

4.4 Trabajar con proyectos de asignación de datos

En MapForce también puede crear proyectos de asignación de datos formados por varios diseños de asignación. Todas las asignaciones de datos que añada a un proyecto estarán disponibles en la ventana Proyecto de la aplicación.



Ventana Proyecto (MapForce Enterprise Edition)

La principal ventaja de trabajar con un proyecto es que podrá definir una configuración de generación de código común (como el lenguaje de destino y el directorio de salida) para todos los archivos de asignación que formen parte del proyecto. También se pueden crear carpetas de proyecto y definir una configuración de código especial para cada carpeta del proyecto. Para más información sobre la función de generación de código de programa de MapForce (en C++, C# y Java), consulte la sección Generador de código.

En MapForce Enterprise Edition también puede crear proyectos de servicio web. Con este tipo de proyectos podrá generar código de programa Java o C# que implemente servicios web SOAP basados en archivos WSDL. Consulte la sección Implementar servicios web para obtener más información.

4.4.1 Abrir, cerrar proyectos y realizar búsquedas

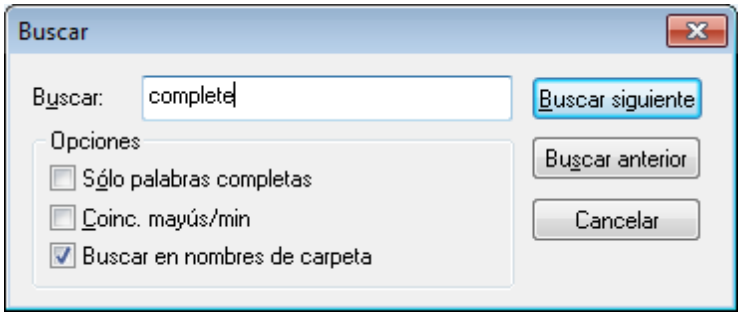
Los archivos de proyecto de MapForce tienen la extensión de archivo ***.mfp** y se abren desde el comando de menú **Archivo | Abrir**.

Cuando se abre un proyecto de asignación en MapForce la ventana Proyecto muestra todos los

archivos y carpetas que se añadieron al proyecto. Cuando se inicia MapForce por primera vez, el proyecto que se carga en esta ventana es el proyecto de ejemplo **MapForceExamples.mfp**.

Para buscar archivos en un proyecto:

1. En la ventana Proyecto haga clic en el nombre del proyecto o de la carpeta donde desea buscar archivos.
2. Pulse **Ctrl + F**.
3. También puede configurar la búsqueda. Por ejemplo, si quiere incluir nombres de carpeta en la búsqueda, marque la opción *Buscar en nombres de carpeta.*



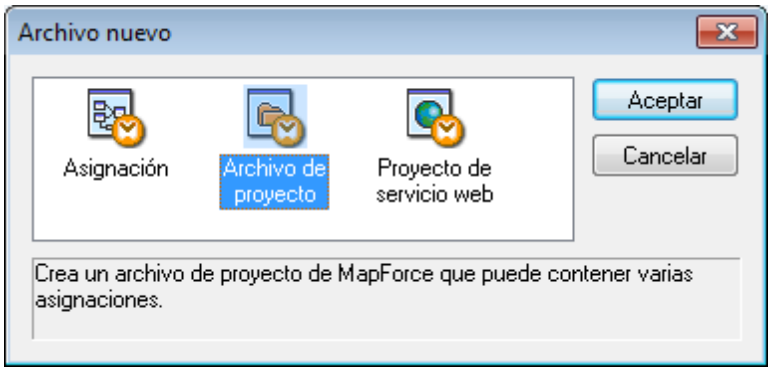
Para cerrar un proyecto:

- En el menú **Proyecto** haga clic en el comando **Cerrar el proyecto**.

4.4.2 Crear proyectos nuevos

Para crear un proyecto nuevo:

1. En el menú **Archivo** haga clic en el comando **Nuevo**.
2. Seleccione la opción **Archivo de proyecto** y haga clic en el botón **Aceptar**.



3. En el cuadro de diálogo "Guardar proyecto como" introduzca el nombre del proyecto y haga clic en el botón **Guardar**. El proyecto nuevo aparece ahora en la ventana Proyecto.
4. Enter the project name in the **Save Project As** dialog box, and click **Save**. The new project is now displayed in the Project window.

 © 2018 Altova GmbH



Ahora puede añadir archivos de asignación de datos al proyecto.

Para añadir la asignación activa al proyecto tiene dos opciones:

- En el menú **Proyecto** haga clic en el comando **Agregar archivo activo al proyecto**.
- Haga clic con el botón derecho en el proyecto y elija **Agregar archivo activo al proyecto** en el menú contextual.

Para agregar archivos de asignación al proyecto tiene dos opciones:

- En el menú **Proyecto** haga clic en el comando **Agregar archivos al proyecto**.
- Haga clic con el botón derecho en el proyecto y elija **Agregar archivos al proyecto** en el menú contextual.

Consejo: para abrir varios archivos a la vez pulse la tecla **Ctrl** mientras selecciona archivos en el cuadro de diálogo "Abrir".

Para quitar un archivo o una carpeta de un proyecto tiene dos opciones:

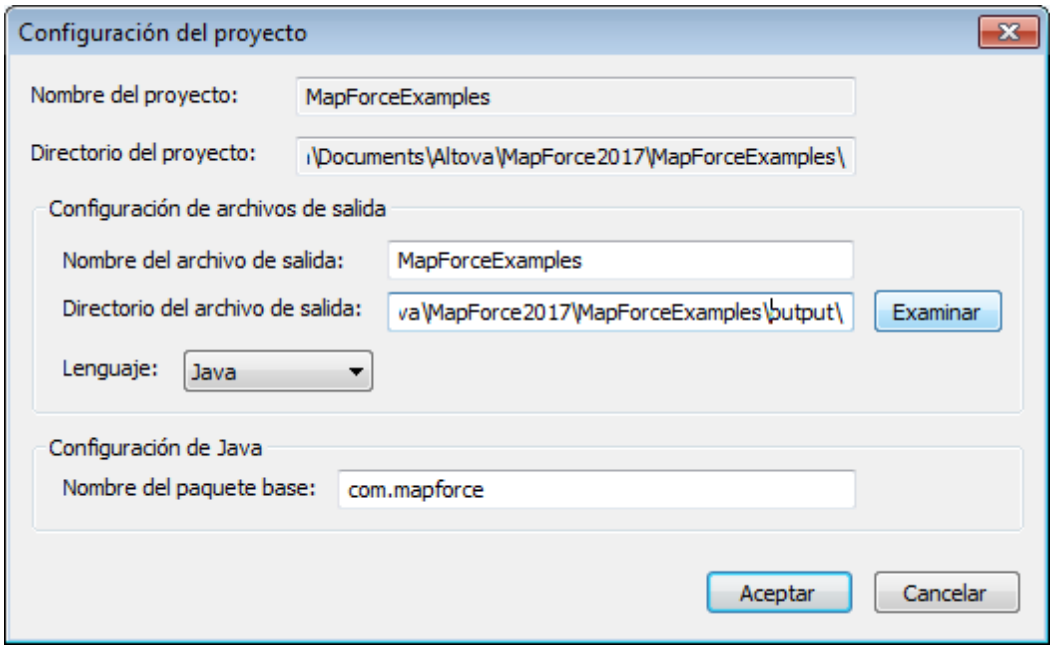
- En la ventana Proyecto haga clic con el botón derecho en el componente que desea eliminar y elija la opción **Eliminar** en el menú contextual.
- En la ventana Proyecto seleccione el componente que desea eliminar y pulse la tecla **Supr**.

4.4.3 Configurar la generación de código

Puede elegir una configuración de generación de código para cada proyecto y dicha configuración se aplicará a todas las asignaciones de datos del proyecto.

Hay dos maneras de abrir el cuadro de diálogo "Configuración del proyecto":

- Haciendo clic con el botón derecho en el nombre del proyecto en la ventana Proyecto y eligiendo el comando de menú **Propiedades** del menú contextual.
- Haciendo clic en el comando **Propiedades** del menú **Proyecto**.



Cuadro de diálogo "Configuración del proyecto"

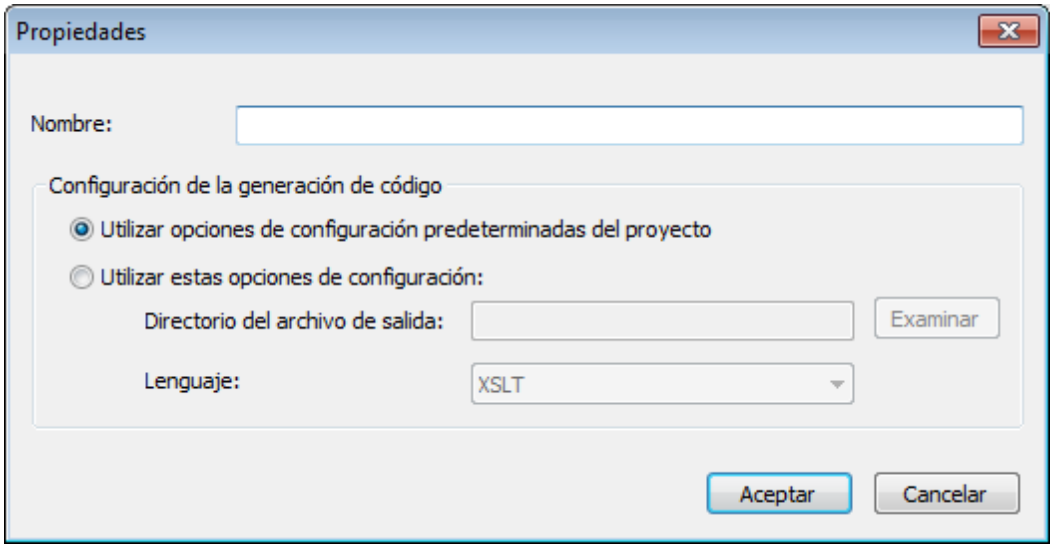
Estas son las opciones que ofrece este cuadro de diálogo (recuerde que el nombre y el directorio del proyecto no se pueden cambiar después de crear el proyecto).

Nombres de salida	Nombre del proyecto o solución que se genera. También será el nombre de los objetos que se creen en el código generado.
Directorio de salida	Carpeta de Windows donde se guardará el código generado (a partir de todas las asignaciones del proyecto). El directorio de salida predeterminado es el directorio **output/** situado en el directorio del proyecto.
Lenguaje	Lenguaje de generación de código que se debe usar para todos los archivos de asignación del proyecto.
Nombre del paquete base	Esta opción solo es relevante si seleccionó el lenguaje Java. Define el nombre del paquete base en el proyecto de Java generado.

© 2018 Altova GmbH

4.4.4 Gestionar las carpetas de proyecto

Si quiere organizar las asignaciones que forman parte del proyecto en carpetas diferentes, basta con agregar carpetas al proyecto y después arrastrar o agregar asignaciones a las carpetas pertinentes. Estas carpetas de proyecto son en realidad carpetas virtuales, relevantes solamente dentro del proyecto de MapForce. Es decir, no se corresponden con ninguna carpeta del sistema operativo. Una de las ventajas de crear carpetas de proyecto es que podrá definir una configuración de generación de código común para todos los archivos de asignación de cada carpeta.



Cuadro de diálogo "Propiedades" de una carpeta

Para crear una carpeta dentro de un proyecto de MapForce:

1. Para empezar tiene dos opciones:
   - Hacer clic en el comando **Crear carpeta** del menú **Proyecto**.
   - Hacer clic con el botón derecho en el proyecto y elegir el comando **Crear carpeta** del menú contextual.
2. Aparece el cuadro de diálogo "Propiedades". Elija la configuración de generación de código pertinente y haga clic en **Aceptar**.

Estas son las opciones de configuración que ofrece el cuadro de diálogo "Propiedades" de una carpeta de proyecto:

Nombre	Nombre de la carpeta de proyecto.
Utilizar opciones de configuración predeterminadas del proyecto	Esta es la opción predeterminada y significa que la configuración de generación de código de la carpeta actual es idéntica a la del proyecto entero. Por tanto, cuando genere código a partir del proyecto, MapForce usará la

	configuración de generación de código definida a nivel de proyecto y no la de la carpeta. Si necesita una configuración de generación de código especial para la carpeta, seleccione la opción *Utilizar estas opciones de configuración* y especifique el directorio y el lenguaje de salida.
Directorio de salida:	Define la carpeta de Windows donde se guardará el código generado (a partir de todas las asignaciones que están en la carpeta de proyecto).
Language	Define el lenguaje de generación de código que se debe utilizar para todos los archivos de asignación que están en la carpeta de proyecto.

© 2018 Altova GmbH

Altova MapForce 2018 Enterprise Edition

Diseño de asignaciones

5 Diseño de asignaciones

Sitio web de Altova: Herramienta de integración de datos

En esta sección encontrará información para trabajar con diseños de asignación de datos e instrucciones para transformar datos en el área de asignación.

Si su objetivo es....	Consulte este apartado
Crear o editar referencias de rutas de acceso a archivos de esquema, de instancia, etc. utilizados por una asignación de datos.	Usar rutas de acceso absolutas y relativas
Adaptar la asignación de datos a requisitos específicos (p. ej. para ajustar la secuencia de los nodos en un componente de destino).	Tipos de conexión
Usar el resultado de un componente como entrada de otro componente.	Asignaciones en cadena y componentes de paso a través
Procesar varios archivos (p. ej. todos los archivos de un directorio) en una misma asignación, como origen o como destino.	Procesar varios archivos de entrada o salida simultáneamente
Pasar un valor externo (p. ej. un parámetro de cadena) a la asignación.	Pasar parámetros a la asignación
Obtener como resultado de la asignación un valor de cadena en lugar de un archivo.	Obtener valores de cadena de una asignación
Almacenar ciertos datos de asignación temporalmente para procesarlos más adelante (similar al concepto de variable en los lenguajes de programación).	Usar variables
Ordenar datos de forma ascendente o descendente.	Ordenar datos
Filtrar nodos/filas en función de determinados criterios o procesar valores de forma condicional.	Filtros y condiciones
Combinar o unir datos de varios componentes de origen con esquemas distintos.	Combinar datos Combinar datos de varios esquemas distintos
Procesar pares de clave/valor. Por ejemplo, para convertir los meses de una representación numérica (01, 02, etc.) en una representación textual (enero, febrero, etc.).	Usar asignaciones de valores
Configurar una asignación para que devuelva un error si se produce una condición determinada.	Añadir excepciones

© 2018 Altova GmbH

Si su objetivo es....	Consulte este apartado
Convertir estructuras de asignación complejas en tipos de datos de cadena y viceversa.	Analizar y serializar cadenas de texto
Aprender a evitar resultados no deseados en asignaciones de datos complejas.	Reglas y estrategias de asignación de datos

Nota importante: MapForce viene con una amplia biblioteca de funciones integradas (*véase Referencia de la biblioteca de funciones*) diseñada para ayudar al usuario con una multitud de tareas de procesamiento. Si esta biblioteca de funciones integradas no es suficiente, puede crear sus propias funciones en MapForce o reciclar archivos XSLT externos, así como archivos .dll o bibliotecas .class de Java. Para más información consulte la sección Funciones.

5.1 Usar rutas de acceso absolutas y relativas

Un archivo de diseño de asignación de datos (*.mfd) puede tener referencias a varios archivos de instancia o de esquema. Los archivos de esquema le sirven a MapForce para determinar la estructura de los datos que se deben asignar y para validarlos. Por su parte, los archivos de instancia sirven para leer los datos de origen, validarlos con el esquema y generar una vista previa de la asignación de datos.

Las asignaciones de datos también pueden incluir referencias a archivos *.sps de Altova StyleVision (StyleVision Power Stylesheets), que sirven para aplicar formato a los datos y generar documentos de salida en PDF, HTML y Word. Además, las asignaciones pueden incluir referencias a bases de datos basadas en archivos, como Microsoft Access o SQLite.

Cuando añadimos un componente a la asignación, MapForce crea las referencias a los archivos. No obstante, estas referencias se pueden cambiar y configurar a mano en cualquier momento.

En esta sección explicamos cómo configurar y modificar las rutas de acceso de los diferentes tipos de archivo a los que se hace referencia en las asignaciones de datos. También describimos las diferencias que existen entre usar rutas de acceso relativas y absolutas.

5.1.1 Usar rutas de acceso relativas en un componente

En el cuadro de diálogo "Configuración del componente" podemos indicar rutas de acceso relativas o absolutas para los diferentes archivos a los que puede hacer referencia el componente:

- Archivos de entrada (es decir, archivos en los que MapForce lee datos).
- Archivos de salida (es decir, archivos en los que MapForce escribe datos).
- Archivos de esquema (si el componente tiene un esquema).
- Archivos de estructura (si el componente tiene estructuras complejas, como parámetros de entrada o salida de funciones definidas por el usuario o variables).
- Archivos StyleVision Power Stylesheet (*.sps) utilizados para aplicar formato a los datos y generar documentos de salida en PDF, HTML y Word.

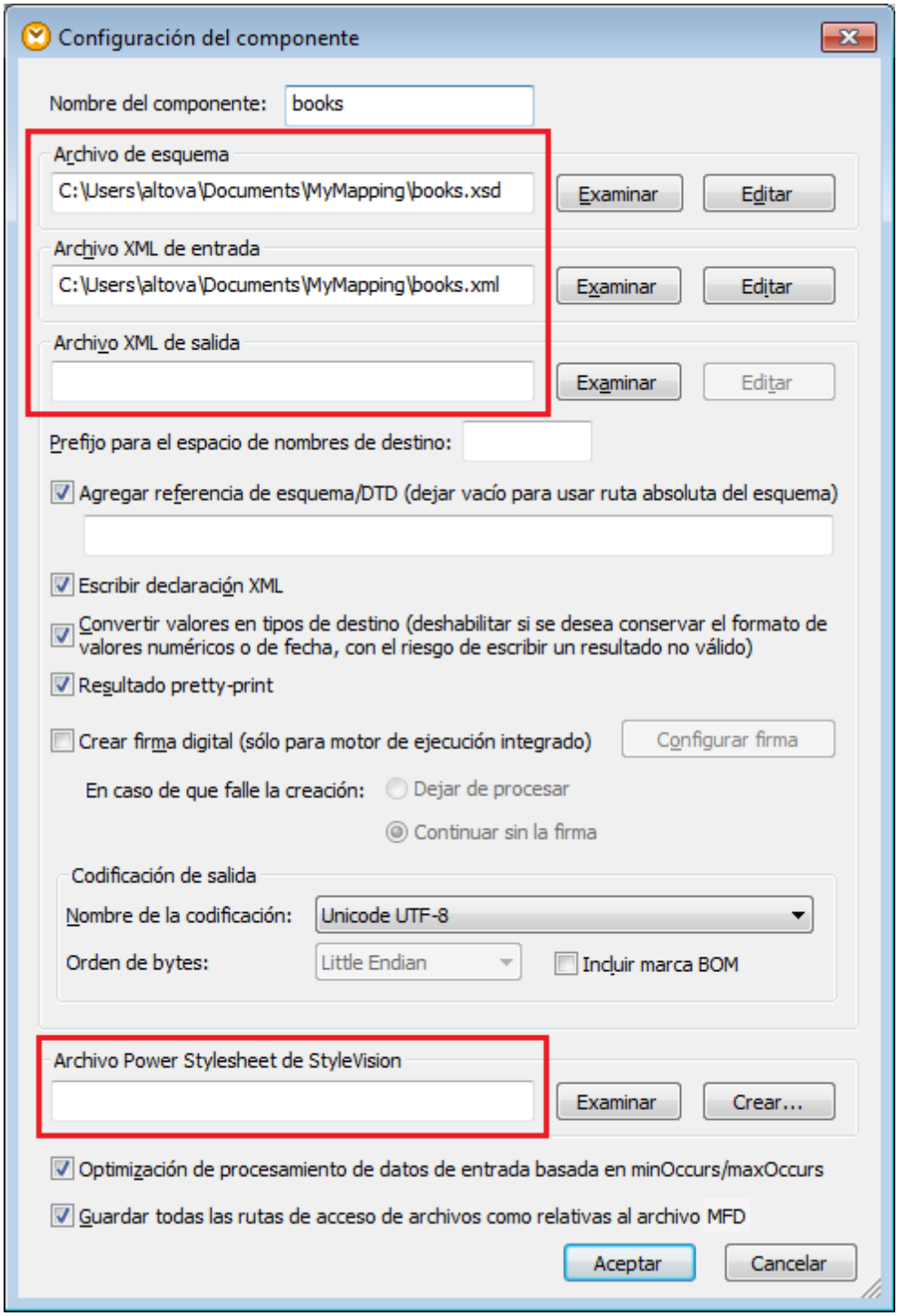
Puede introducir rutas de acceso relativas en los diferentes cuadros de texto directamente (marcados en rojo en la imagen siguiente).

> Antes de introducir las rutas de acceso relativas, asegúrese de guardar el archivo de asignación de datos (.mfd). Si no lo hace, todas las rutas de acceso relativas se resolverán a partir de la carpeta de aplicación personal de Windows (`Documentos\Altova\MapForce2018`).

También puede hacer que MapForce guarde todas las rutas de acceso mencionadas anteriormente como relativas al archivo de asignación (.mfd). Por ejemplo, en la imagen siguiente puede ver que al final del cuadro de diálogo "Configuración del componente" hay una casilla llamada *Guardar todas las rutas de acceso de archivos como relativas al archivo MFD.* Si esta casilla está marcada (opción predeterminada y recomendada), las rutas de acceso de los archivos a los que hace referencia el componente se guardarán como relativas a la ruta de acceso del archivo de diseño de asignación (.mfd). Esto afecta a todos los archivos a los que

© 2018 Altova GmbH

hace referencia el componente (marcados con un recuadro rojo en la imagen siguiente).



Cuadro de diálogo "Configuración del componente"

En el ejemplo anterior se trataba de un componente XML, pero la casilla *Guardar todas las rutas de acceso de archivos como relativas al archivo MFD* tiene el mismo funcionamiento con estos archivos:

- Archivos de estructuras utilizados por parámetros de entrada o salida complejos de funciones definidas por el usuario o por variables de tipo complejo.
- Archivos planos de entrada o salida*.
- Archivos de esquema a los que hacen referencia componentes de BD compatibles con campos XML*.
- Archivos de entrada o salida XBRL, FlexText, EDI, Excel 2007+ y JSON files.

* MapForce Professional y Enterprise Edition
** MapForce Enterprise Edition solamente

Volvamos al componente XML del ejemplo. Si el archivo .mfd está en la misma carpeta que los archivos **books.xsd** y **books.xml**, MapForce realizará estos cambios eb las rutas de acceso:

C:\Usuarios\altova\Documentos\MyMapping\books.xsd se cambiará por **books.xsd**
C:\Usuarios\altova\Documentos\MyMapping\books.xml se cambiará por **books.xml**

Las rutas de acceso que hagan referencia a unidades de disco no locales o que utilicen una URL no se convertirán en rutas relativas.

Cuando la casilla *Guardar todas las rutas de acceso de archivos como relativas al archivo MFD* esté marcada, MapForce también vigilará los archivos a los que hace referencia el componente cuando la asignación se guarde en una carpeta nueva con el comando de menú **Guardar como**. Además, si todos los archivos están en la misma carpeta que la asignación, las referencias de rutas de acceso no se romperán cuando mueva el directorio entero a una ubicación nueva en el disco.

El uso de rutas de acceso relativas (y, por tanto, la activación de la casilla *Guardar todas las rutas de acceso de archivos como relativas al archivo MFD*) puede se muy importante en algunos casos. Por ejemplo:

- Si es probable que cambie la ubicación de la asignación en el sistema operativo.
- Si la asignación está en un directorio que está bajo control de código fuente (controlado por un sistema de control de código fuente como TortoiseSVN, por ejemplo).
- Si tiene pensado implementar la asignación en un equipo distinto (e incluso en un sistema operativo distinto) para ejecutarla.

Si no está marcada la casilla *Guardar todas las rutas de acceso de archivos como relativas al archivo MFD*, cuando guarde la asignación no se modificarán las rutas de acceso de los archivos (es decir, quedarán tal y como aparecen en el cuadro de diálogo "Configuración del componente").

5.1.2 Configurar la ruta de acceso de BD basadas en archivos

Cuando añada un archivo de BD (como Microsoft Access o SQLite) a la asignación (*véase Iniciar el asistente para la conexión de base de datos*), podrá usar una ruta de acceso relativa en lugar de una ruta de acceso absoluta. Para ello basta con introducir la ruta de acceso relativa correspondiente en lugar de hacer clic en el botón **Examinar** del asistente para la conexión de base de datos.

© 2018 Altova GmbH

Antes de introducir rutas de acceso relativas deberá guardar el archivo de asignación (.mfd). De lo contrario, todas las rutas de acceso se resolverán con la carpeta de aplicación personal de Windows (`...Documentos\Altova\MapForce2018`).



Asistente para la conexión de base de datos

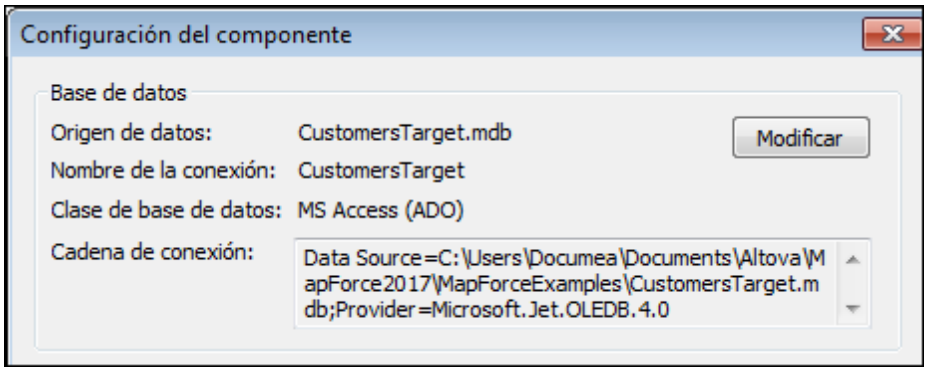
Si se trata de una base de datos SQLite, el botón Conectarse se habilitará solamente si se cumplen estos dos requisitos:

- La ruta de acceso debe apuntar a un archivo que se pueda resolver en relación con el archivo de asignación (.mfd).

- El archivo al que se hace referencia es una base de datos SQLite.

Para cambiar la ruta de acceso de un componente de base de datos que ya está en la asignación:

1. Haga clic con el botón derecho en el título del componente de base de datos y elija **Propiedades** (*véase* Cambiar configuración de los componentes). Otra opción es hacer doble clic en la barra de título del componente.
2. En el cuadro de diálogo "Configuración del componente" haga clic en el botón **Modificar**.



Esto vuelve a abrir el asistente para la conexión de base de datos, donde puede cambiar las propiedades de la conexión, incluida la ruta de acceso.

Debe tener en cuenta que el campo *Cadena de conexión* siempre contiene una ruta de acceso absoluta. Se trata de la base de datos que se utiliza para obtener información sobre la estructura del componente. La ruta de acceso relativa del campo *Origen de datos* indica que el componente se creó con una ruta de acceso relativa.

Nota: cuando genere código de programa, cuando compile archivos de ejecución para MapForce Server o cuando implemente la asignación en FlowForce Server, las rutas de acceso relativas se convertirán en absolutas si se marcó la casilla *Convertir las rutas de acceso en absolutas en el código generado* del cuadro de diálogo "Configurar asignación" (*véase* Rutas de acceso en el código generado).

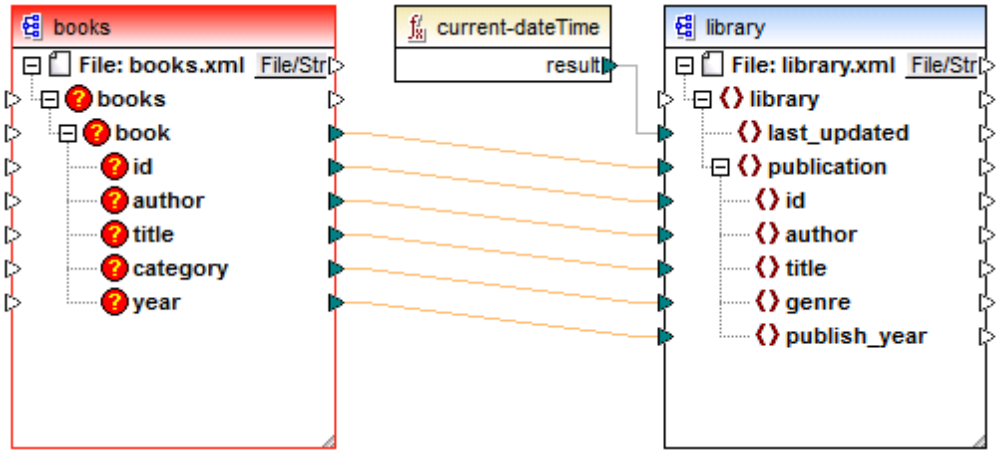
5.1.3 Corregir referencias a rutas de acceso rotas

Cuando añada o modifique una referencia de archivo en una asignación y no se pueda resolver la ruta de acceso, MapForce emite un mensaje de advertencia. De este modo MapForce reduce el riesgo de que haya referencias de ruta de acceso rotas. No obstante, pueden darse referencias de ruta de acceso rotas en estos casos:

- Cuando use rutas de acceso relativas y después mueva el archivo de asignación a un directorio nuevo sin mover también los archivos de esquema y de instancia.
- Cuando use rutas de acceso absolutas de archivos que están en el mismo directorio que el archivo de asignación y después mueva el directorio a otra ubicación.

Cuando esto ocurra, MapForce resaltará el componente en color rojo. Por ejemplo:

© 2018 Altova GmbH



Referencia de ruta de acceso rota

En casos así, la solución consiste en hacer doble clic en el título del componente y actualizar las referencias de ruta de acceso rotas en el cuadro de diálogo "Configuración del componente" (*véase* *Cambiar configuración de los componentes*).

5.1.4 Rutas de acceso según el entorno de ejecución

Si genera código a partir de la asignación de datos, si compila asignaciones en archivos de ejecución de MapForce Server (.mfx) o implementa asignaciones en FlowForce Server, los archivos generados ya no se ejecutan en MapForce, sino en el entorno de destino que usted elija (p. ej. RaptorXML Server, MapForce Server o una aplicación C#). Sin embargo, para que la asignación de datos pueda ejecutarse correctamente es imprescindible que las rutas de acceso relativas sean significativas en el entorno donde se ejecuta la asignación.

Por tanto, cuando la asignación utilice rutas de acceso relativas de archivos de instancia o esquema, tenga en cuenta estas rutas de acceso base para cada lenguaje de destino:

Lenguaje de destino	Ruta de acceso base
XSLT/XSLT2	Ruta de acceso del archivo XSLT.
XQuery*	Ruta de acceso del archivo XQuery.
C++, C#, Java*	Directorio de trabajo de la aplicación generada.
BUILT-IN* (cuando genere vista previa de la asignación en MapForce)	Ruta de acceso del archivo de asignación (.mfd).
BUILT-IN* (cuando ejecute la asignación con MapForce Server)	El directorio de trabajo actual.
BUILT-IN* (cuando ejecute la asignación con	El directorio de trabajo del trabajo o de

Lenguaje de destino	Ruta de acceso base
MapForce Server bajo control de FlowForce Server)	FlowForce Server.

** Lenguajes disponibles en las ediciones Professional y Enterprise de MapForce*

Si lo necesita, también puede hacer que MapForce convierta todas las rutas de acceso relativas en rutas de acceso absolutas a la hora de generar código para una asignación. Esta opción puede ser muy útil si el código de asignación (o el archivo de ejecución de MapForce Server) se ejecuta en el mismo sistema operativo o incluso en otro sistema operativo donde se puedan resolver las rutas de acceso absolutas utilizadas por la asignación.

Para convertir todas las rutas de acceso en absolutas en el código generado basta con marcar la casilla *Convertir las rutas de acceso en absolutas en el código generado* en el cuadro de diálogo "Configurar asignación" (*véase* *Cambiar configuración de la asignación*).

Cuando genere código con esta casilla activada, MapForce resolverá las rutas de acceso relativas en base al directorio del archivo de asignación (.mfd) y las convertirá en absolutas en el código generado. Esta configuración afectará a la ruta de acceso de estos archivos:

- Archivos de instancia de entrada y salida para todas las clases de componente basados en archivos.
- Archivos de base de datos Access y SQLite utilizados como componentes de asignación.

Cuando esté desactivada la casilla *Convertir las rutas de acceso en absolutas en el código generado,* las rutas de acceso se conservarán tal y como se definieran en la configuración del componente.

5.1.5 Operaciones cortar y pegar con rutas de acceso relativas

Cuando copie un componente de una asignación y lo pegue en otra, MapForce comprueba que las rutas de acceso relativas de los archivos de esquema se puedan resolver con la carpeta de la asignación de destino. Si las rutas de acceso no se pueden resolver, la aplicación solicitará que convierta las rutas de acceso relativas en absolutas a través de la carpeta de la asignación de origen. Se recomienda guardar en primer lugar la asignación de destino porque, de lo contrario, las rutas de acceso relativas se resolverán con la carpeta de la aplicación personal.

© 2018 Altova GmbH

5.2 Tipos de conexión

Cuando cree una conexión de asignación (y tanto el nodo de origen como el de destino tengan secundarios) tendrá la opción de elegir entre tres tipos de conexiones:

- Conexión basada en el destino (estándar).
- Conexión basada en el origen (contenido mixto).
- Conexión de copia total (copia de secundarios).

El tipo de conexión determina la secuencia de los secundarios en el resultado que se genera a partir de la asignación. En esta sección encontrará información sobre cada tipo de conexión y cómo utilizar cada uno de ellos.

5.2.1 Conexiones basadas en el destino

Cuando una conexión está basada en el destino (conexión estándar), la secuencia de los nodos secundarios en el resultado de la asignación viene dado por la secuencia que siguen los nodos en el esquema de destino. Este tipo de conexión es adecuada para la mayoría de las asignaciones y es el tipo de conexión predeterminada de MapForce.

En las asignaciones las conexiones basadas en el destino se representan por medio de una línea sólida.
On a mapping, target-driven connections are shown with a solid line.



Es posible que las conexiones basadas en el destino no sean adecuadas cuando su intención sea asignar nodos XML que tienen contenido mixto (datos de caracteres y elementos secundarios). Por ejemplo:

```
<p>Este es nuestro producto <i>más vendido</i>.</p>
```

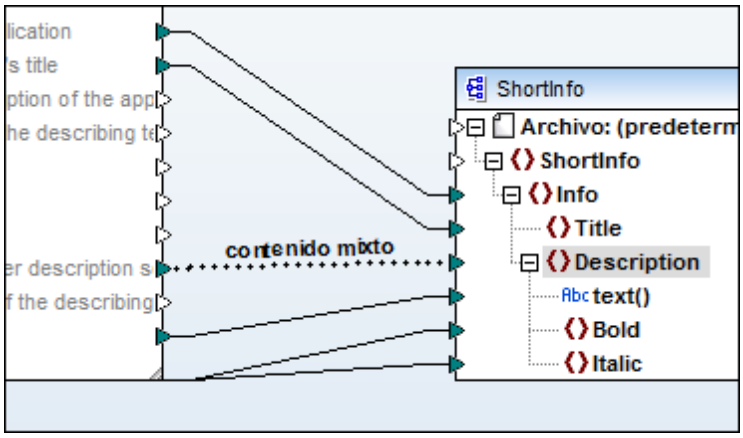
Cuando los nodos tienen contenido mixto, es probable que el objetivo sea conservar la secuencia de los nodos tal y como aparecen en el archivo de origen. Para ello se recomienda el uso de conexiones basadas en el origen (*véase* *el apartado siguiente*).

5.2.2 Conexiones basadas en el origen

Las conexiones basadas en el origen (contenido mixto) permiten asignar automáticamente nodos de texto y nodos secundarios en la misma secuencia en que aparecen en el archivo XML de **origen**.

- El contenido de nodos de texto se puede asignar.

- La secuencia de los nodos secundarios depende del archivo XML de instancia de origen.



En las asignaciones las conexiones de contenido mixto se presentan por medio de una línea de puntos.

Las conexiones basadas en el origen o de contenido mixto también se pueden aplicar a elementos **complextType**. Los nodos secundarios se asignarán en base a su secuencia en el archivo XML de origen.

Las conexiones basadas en el origen o de contenido mixto son compatibles con asignaciones:

- De componentes de **origen**:
  - elementos complexType del esquema XML (incluidos elementos de contenido mixto, es decir mixed=true)
  - elementos complexType del esquema XML (incluidos elementos de contenido mixto) en esquemas incrustados de un campo de base de datos
- De componentes de **destino**:
  - elementos complexType del esquema XML (incluidos elementos de contenido mixto)
  - elementos complexType del esquema XML (incluidos elementos de contenido mixto) en esquemas incrustados de un campo de base de datos

Nota: las secciones CDATA se tratan como si fueran texto.

5.2.2.1 Crear asignaciones entre contenido mixto

Los archivos utilizados para ilustrar este ejemplo son **Tut-OrgChart.mfd**, **Tut-OrgChart.mfd.xml**, **Tut-OrgChart.mfd.xsd** y **Tut-Person.xsd** y están disponibles en la carpeta `...\MapForceExamples\Tutorial\`.

Instancia XML de origen

En la imagen que aparece a continuación puede ver parte del archivo **Tut-OrgChart.xml** utilizado para ilustrar este ejemplo. Ahora mismo nos centraremos en el elemento de contenido mixto `para`, así como en sus nodos secundarios `bold` e `italic`.

El elemento `para` también contiene una instrucción de procesamiento (`<?sort alpha-ascending?>`) y un comentario (`<!--Company details... -->`) que también se pueden asignar, tal y como se puede ver más abajo.

 © 2018 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2005 sp2 U (http://www.altova.com) by Mr. Nobody (Altova GmbH) -->
<OrgChart xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Tut-OrgChart.xsd">
  <CompanyLogo href="nanonull.gif"/>
  <Name>Organization Chart</Name>
  <Office>
    <Name>Nanonull, Inc.</Name>
    <Desc>
      <para>The company was established in<bold> Vereno</bold>in 1995. Nanonull
develops nanoelectronic technologies for<italic>multi-core processors.</italic>February 1999
saw the unveiling of the first prototype <bold>Nano-grid.</bold>The company hopes to expand
its operations <italic>offshore</italic>to drive down operational costs.
        <?sort alpha-ascending?>
        <!--Company details: location and general company information.-->
      </para>
      <para>White papers and further information will be made available in the near future.
    </Desc>
```

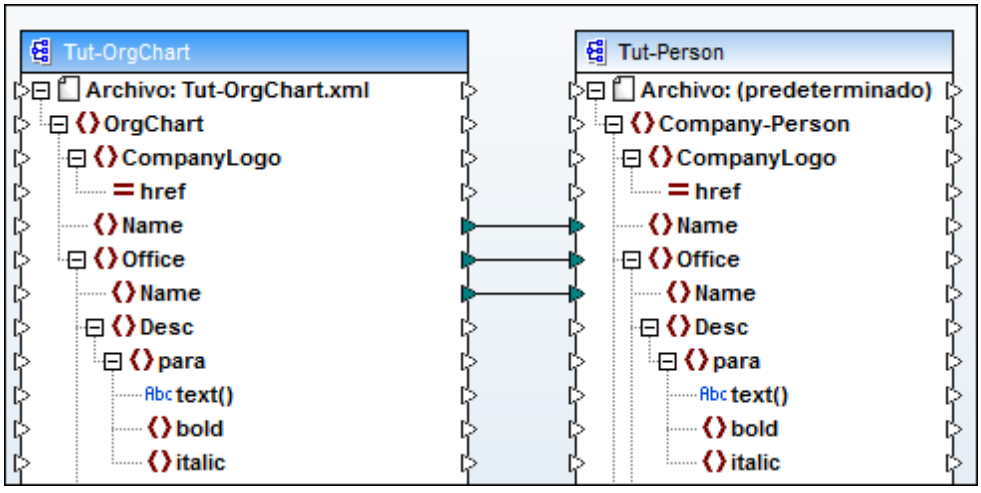
Veamos ahora qué secuencia tienen los nodos de texto y los nodos `bold` e `italic` de Nanonull., Inc en el archivo XML de instancia:

<para> The company...
<**bold**>Vereno<**/bold**>in 1995 ...
<**italic**>multi-core...<**/italic**>February 1999

<**bold**>Nano-grid.<**/bold**>The company ...
<**italic**>offshore...<**/italic**>to drive...
</para>

Asignación inicial

A continuación puede ver el estado inicial que tiene la asignación al abrir el archivo **Tut-Orgchart.mfd**.



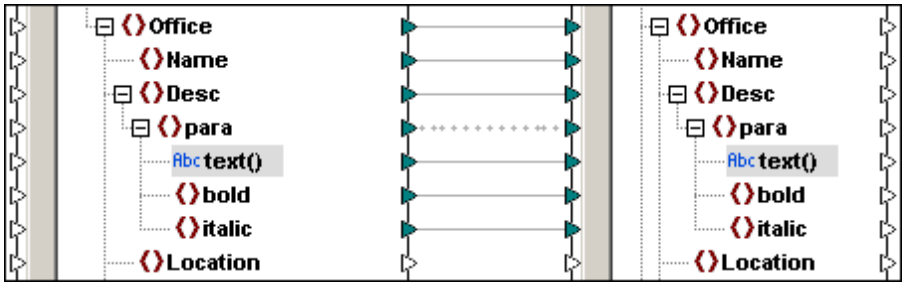
Resultado de la asignación anterior

El resultado de la asignación inicial puede verse en la imagen siguiente. En el resultado se generó `Organization Chart` junto con el nombre de las diferentes oficinas.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company-Person xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNames
3    <Name>Organization Chart</Name>
4    <Office>
5      <Name>Nanonull, Inc.</Name>
6    </Office>
7    <Office>
8      <Name>Nanonull Europe, AG</Name>
9    </Office>
10 </Company-Person>
11
```

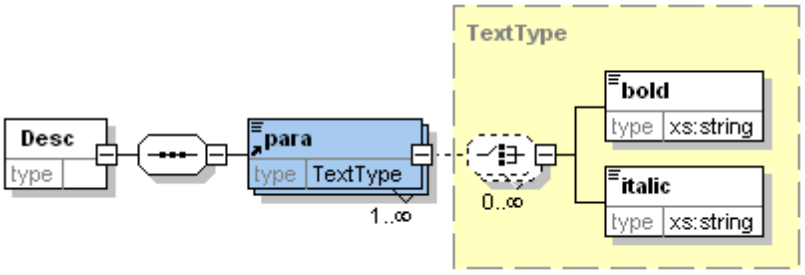
Asignación del elemento para

En la imagen siguiente puede ver un ejemplo de asignación de contenido mixto. El elemento `para` es de contenido mixto y su conexión se representa por medio de una línea de puntos. El nodo `text()` contiene los datos de texto y debe asignarse para que el texto aparezca en el componente de destino.



Para anotar una conexión (es decir, para añadirle una etiqueta) haga clic con el botón derecho en la conexión y elija el comando **Propiedades** (*véase* *Crear anotaciones en las conexiones*).

En la imagen siguiente podemos ver el modelo de contenido del elemento `Description` (`Desc`) del archivo de esquema **Tut-OrgChart.xsd**. Esta definición es idéntica en los esquemas de origen y destino utilizados en este ejemplo.



En el modelo de contenido podemos observar estas propiedades del elemento `para`:

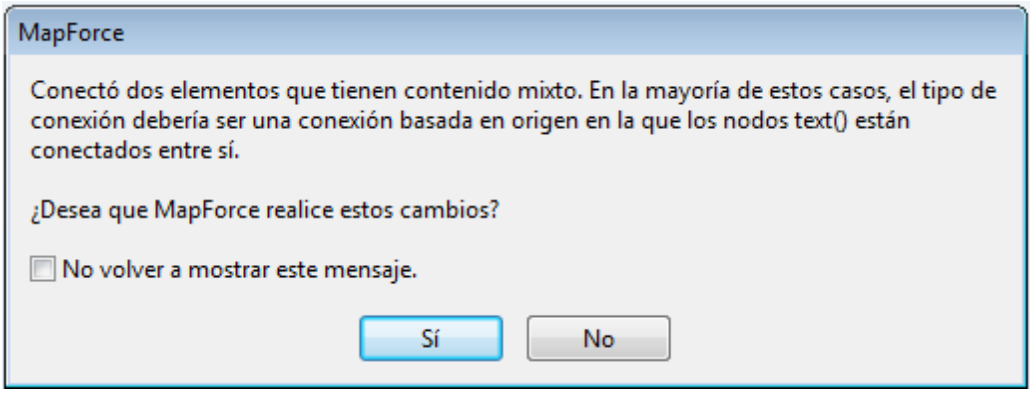
- **para** es un elemento complexType con mixed="true", de tipo "TextType".
- los elementos **bold** e **italic** son de tipo "xs:string" y no se definieron como recursivos en este ejemplo (es decir, ni **bold** ni **italic** son de tipo "TextType").
- los elementos **bold** e **italic** pueden aparecer tantas veces como sea necesario en cualquier secuencia dentro del elemento **para**.
- dentro del elemento **para** pueden aparecer tantos nodos de texto como sean necesarios, intercalados con tantos elementos **bold** e **italic** como sean necesarios.

© 2018 Altova GmbH

Para crear conexiones de contenido mixto entre elementos:

1. Seleccione el comando de menú **Conexión | Conectar automáticamente los secundarios equivalentes** para activar esta opción si aún no lo está.
2. Conecte el nodo **para** del esquema de origen con el nodo **para** del esquema de destino.

   Ahora aparece un mensaje preguntando si desea que MapForce defina los conectores como conectores basados en el origen.



3. Haga clic en **Sí** para crear una conexión de contenido mixto

   Nota: para es de contenido mixto y desencadena el mensaje en este momento dado. El mensaje de contenido mixto también aparece si solo se asignan los nodos **para** directamente, sin tener activada la opción de conexión automática de secundarios equivalentes.

   Ahora todos los elementos secundarios de **para** están conectados. El conector que une los elementos **para** es una línea de puntos, lo cual indica que se trata de una conexión de contenido mixto.

4. Haga clic en el panel *Resultados* para ver el resultado de la asignación.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company-Person xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNames
3    <Name>Organization Chart</Name>
4    <Office>
5      <Name>Nanonull, Inc.</Name>
6      <Desc>
7        <para>The company was established in<bold> Vereno</bold>in 1995. Nanonull devel
8          </para>
9        <para>White papers and further information will be made available in the near future.
10       </para>
11     </Desc>
12   </Office>
13   <Office>
14     <Name>Nanonull Europe, AG</Name>
15     <Desc>
16       <para>In May 2000, Nanonull<italic>Europe</italic> was set up in Vienna. The team co
17     </Desc>
18   </Office>
19 </Company-Person>
20
```

5. Ahora haga clic en el icono **Ajuste automático de línea**  en la barra de iconos del panel *Resultados* para ver todo el texto.

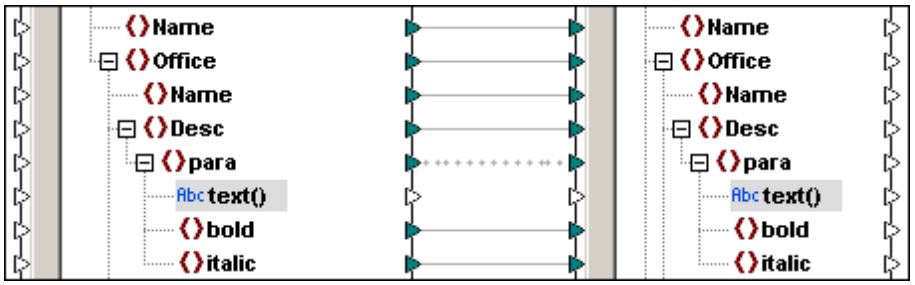
```
3     <Name>Organization Chart</Name>
4     <Office>
5       <Name>Nanonull, Inc.</Name>
6       <Desc>
7         <para>The company was established in<bold> Vereno</bold>in 1995. Nanonull develops
      nanoelectronic technologies for<italic>multi-core processors.</italic>February 1999 saw the
      unveiling of the first prototype <bold>Nano-grid.</bold>The company hopes to expand its
      operations <italic>offshore</italic>to drive down operational costs.
8           </para>
9           <para>White papers and further information will be made available in the near future.
10          </para>
11        </Desc>
12    </Office>
13    <Office>
14      <Name>Nanonull Europe, AG</Name>
15      <Desc>
16        <para>In May 2000, Nanonull<italic>Europe</italic> was set up in Vienna. The team
      consists of<bold> five research scientists </bold>and one administrative staff.</para>
17      </Desc>
18    </Office>
19  </Company-Person>
20
```

El texto de contenido mixto de cada descripción de las oficinas se asignó correctamente. El texto, además del contenido de las etiquetas `bold` e `italic` se asignó en el orden en el que aparecen en el archivo **XML** de origen.

6. Vuelva al panel *Resultados*.

Para quitar nodos de texto de elementos de contenido mixto:

1. Haga clic en el conector del nodo **text()** y pulse **Supr** para eliminarlo.



2. Ahora haga clic en el panel *Resultados* para ver el resultado de la asignación.

© 2018 Altova GmbH

```
<Name>Nanonull, Inc.</Name>
<Desc>
<para>
<bold> Vereno</bold>
<italic>multi-core processors.</italic>
<bold>Nano-grid.</bold>
<italic>offshore</italic>
</para>
<para/>
</Desc>
</Office>
<Office>
<Name>Nanonull Europe, AG</Name>
<Desc>
<para>
<italic>Europe</italic>
<bold> five research scientists </bold>
</para>
</Desc>
```

Resultado:

- se eliminaron todos los nodos de **texto** del elemento `para`.
- se conservó el contenido de texto de los elementos `bold` e `italic` asignados.
- la **secuencia** de los elementos `bold` e `italic` sigue la secuencia del archivo XML de origen.

Para asignar instrucciones de procesamiento y comentarios:

1. Haga clic con el botón derecho en la conexión de contenido mixto y seleccione el comando **Propiedades**.
2. Bajo el grupo de opciones *Basada en origen (contenido mixto)* marque las casillas *Asignar instrucciones de procesamiento* y *Asignar comentarios*.

5.2.2.2 Ejemplo de contenido mixto

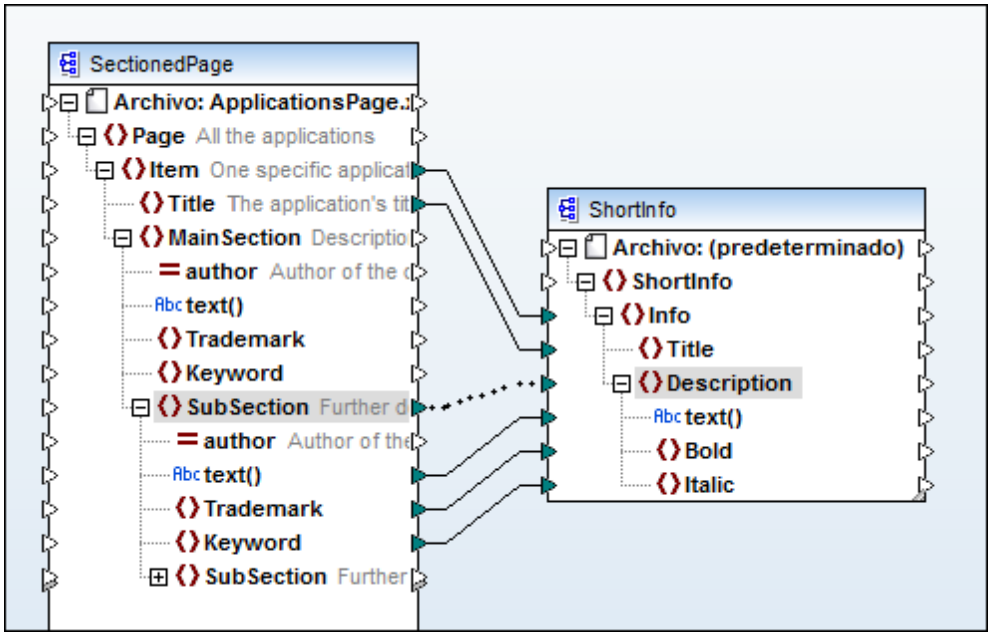
En este apartado usamos como ejemplo el archivo de asignación **ShortApplicationInfo.mfd**, disponible en la carpeta `...\MapForceExamples`.

Aquí puede ver un fragmento del archivo XML de origen de esta asignación:

```
<Page xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="SectionedPage.xsd">
    <Item>
        <Title>XMLSpy</Title>
        <MainSection author="altova">
            Altova <Trademark>XMLSpy</Trademark>
            <SubSection>Altova <Trademark>XMLSpy</Trademark> 2005 Enter
is the industry standard <Keyword>XML</Keyword> development environment
editing, debugging and transforming all <Keyword>XML</Keyword> technolo
automatically generating runtime code in multiple programming languages
        </MainSection>
    </Item>
```

Más abajo puede ver también una imagen de la asignación, en la que podrá observar que:

- El conector del nodo `SubSection` es de contenido mixto y está asignado al nodo `Description` del esquema/XML de destino.
- Los nodos `text()` están conectados.
- El texto de `Trademark` está asignado al nodo `Bold` del componente de destino.
- El texto de `Keyword` está asignado al nodo `Italic` del componente de destino.



Resultado de la asignación

El texto de contenido mixto de cada descripción se asignó correctamente. El texto, así como el contenido de las etiquetas `bold` e `italic`, se asignó tal y como aparecen en el archivo XML de origen.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ShortInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="
   C:/PROGRA~1/Altova/MapForce2005/MapForceExamples/ShortInfo.xsd">
3    <Info>
4      <Title>XMLSpy</Title>
5      <Description>Altova <Bold>XMLSpy</Bold> 2005 Enterprise Edition is the industry standard
   <Italic>XML</Italic> development environment for modeling, editing, debugging and transforming
   all <Italic>XML</Italic> technologies, then automatically generating runtime code in multiple
   programming languages.</Description>
6    </Info>
```

5.2.2.3 Usar conexiones estándar en nodos de contenido mixto

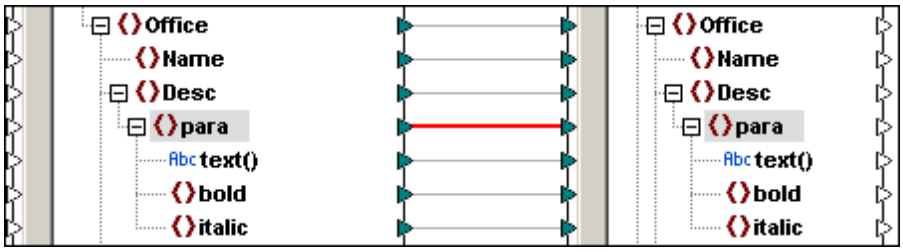
Como se dijo previamente, las conexiones basadas en el origen (o conexiones no estándar) suelen utilizarse para asignar datos de nodos de contenido mixto. De lo contrario, podrían obtenerse resultados no deseados. A continuación mostramos un ejemplo de lo que ocurre cuando se usa una conexión estándar (basada en el destino) para asignar datos de un nodo de contenido mixto:

1. Abra la asignación **Tut-OrgChart.mfd** de la carpeta **<Documentos>\Altova**

© 2018 Altova GmbH

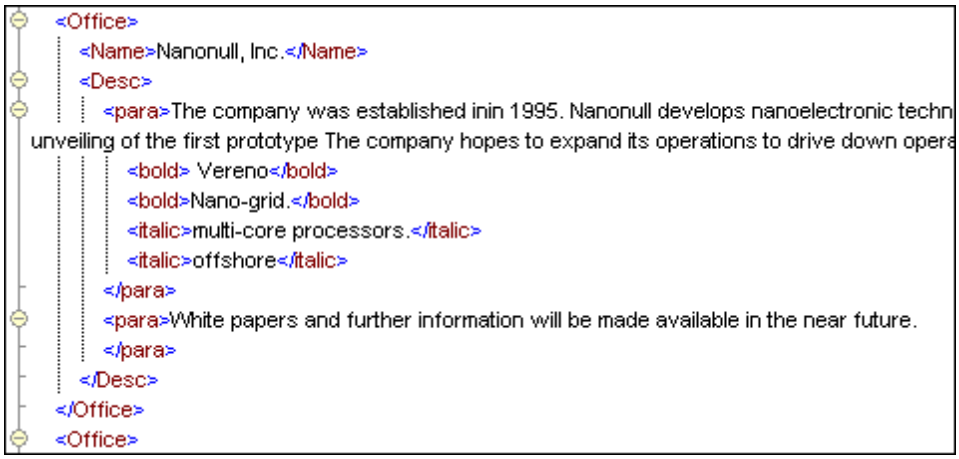
\MapForce2018\MapForceExamples\Tutorial\.

2. Cree una conexión entre el nodo `para` del componente de origen y el nodo `para` del componente de destino. Aparece un mensaje preguntando si desea que MapForce defina las conexiones como basadas en el origen. Haga clic en **No** para ignorar la sugerencia de MapForce y crear una conexión estándar.



Nota: compruebe que la conexión es estándar (basada en el destino). Si se creó una conexión de copia total automáticamente, haga clic con el botón derecho en la conexión y seleccione **Basada en destino (estándar)** en el menú contextual.

3. Abra el panel *Resultados* para ver el resultado de la asignación.



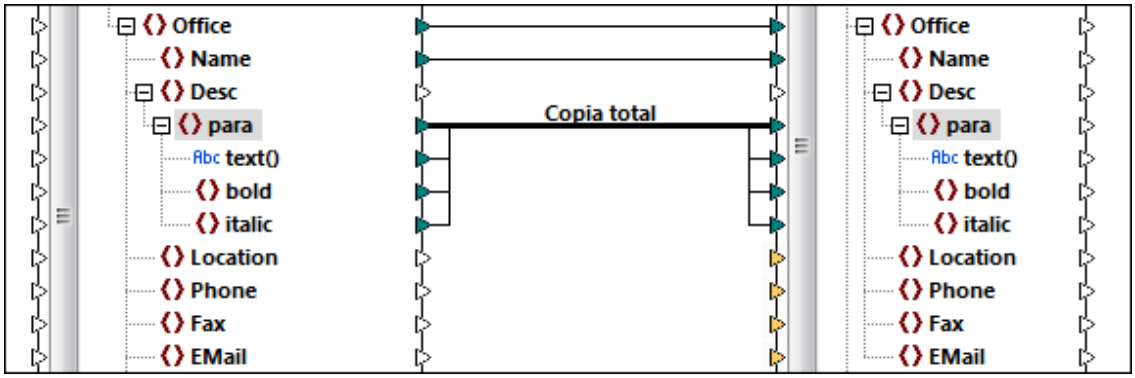
Como puede verse en la imagen, la asignación de nodos de contenido mixto por medio de conexiones estándar produce este resultado:

- El contenido del elemento de origen `text()` se copia en el destino, pero la secuencia de nodos secundarios (`bold` e `italic` en este caso) en el documento de salida corresponde a la secuencia de los nodos en el esquema XML de destino. En otras palabras, los nodos secundarios (`bold` e `italic` en este caso) aparecen después del texto del nodo de contenido mixto.
- Por cada elemento `para` MapForce asigna primero el nodo `text()`, después todos los elementos `bold` y, por último, todos los elementos `italic`. Como resultado aparecen varios elementos `bold` y `italic` apilados. Observe que el contenido de cada elemento se asigna solo si está conectado con el origen.

5.2.3 Conexiones de copia total

Las conexiones de copia total crean asignaciones de datos entre estructuras complejas (nodos con elementos secundarios) que son muy parecidas o idénticas. La ventaja principal de las conexiones de copia total es que simplifican el área de trabajo (se crea una conexión gruesa en lugar de muchas).

En la asignación las conexiones de copia total se representan por medio de una línea gruesa (con horquillas de entrada y salida para cada uno de los elementos secundarios) que conecta dos estructuras idénticas o similares.



Conexión de copia total

Cuando se dibuja una conexión entre dos estructuras en la asignación y MapForce detecta que la estructura de origen y la de destino son compatibles (es decir, que ambas son del mismo tipo que la de destino es un subtipo de la de origen), entonces se crea automáticamente una conexión de copia total. Cuando se ejecute la asignación, todos los datos de instancia se copiarán del origen al destino recursivamente, incluidos los datos secundarios.

Para crear conexiones de copia total a mano haga clic con el botón derecho en una conexión que conecte dos nodos similares y que tengan elementos secundarios y seleccione **Copia total (copia los elementos secundarios)** en el menú contextual.

Debe tener en cuenta que:

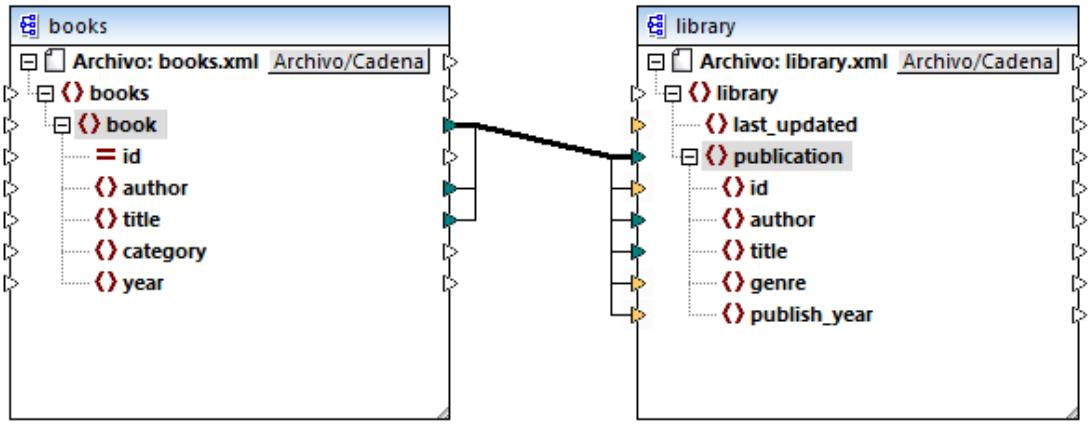
- En contextos donde la conexión de copia total no es relevante o no es compatible, no se puede crear este tipo de conexión de forma manual.
- No se puede crear una conexión de copia total con el elemento raíz `root` de un componente XML o de esquema.
- Cuando se creen conexiones de copia total entre un esquema y un parámetro de una función definida por el usuario, los dos componentes deben estar basados en el mismo esquema (aunque no es necesario que tengan el mismo elemento raíz).

Para crear una conexión de copia total a mano:

1. Cree una asignación nueva.
2. Haga clic en el comando **Insertar | Archivo o esquema XML** y navegue hasta el archivo **books.xml** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**.

© 2018 Altova GmbH

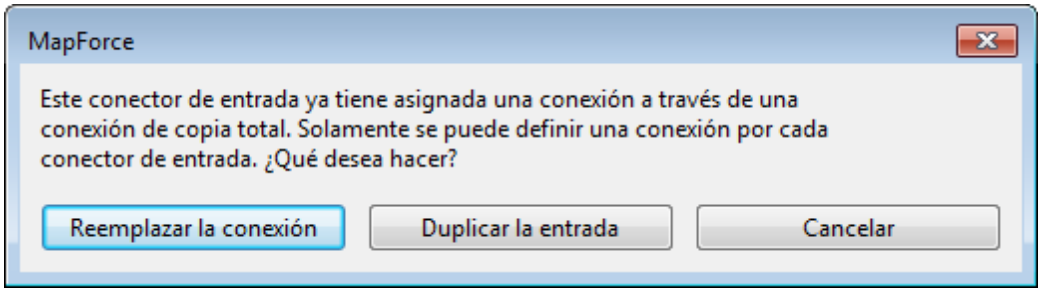
3. Haga clic en el comando **Insertar | Archivo o esquema XML** y navegue hasta el archivo **library.xsd** de la carpeta **<Documentos>\Altova\MapForce2018 \MapForceExamples\Tutorial**.
4. Dibuje una conexión de asignación entre el nodo `book` del componente "books" y el nodo `publication` del componente "library".
5. Haga clic con el botón derecho en la nueva conexión y seleccione **Copia total (copia los elementos secundarios)** en el menú contextual.



Si existen diferencias entre la estructura de origen y de destino, en tiempo de ejecución la conexión de copia total enumerará los nodos de origen (elementos y atributos) y solo copiará los que existan en el tipo de destino (esto se repite recursivamente).

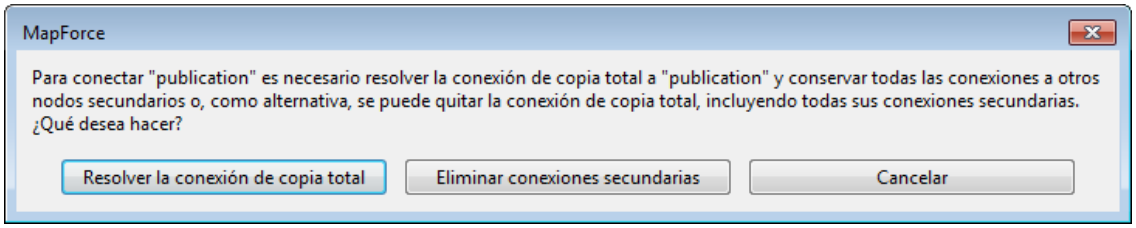
Por ejemplo, en la asignación de la imagen anterior, solamente hay dos elementos secundarios que son idénticos en ambas estructuras (`author` y `title`) y, por tanto, están conectados. El nodo `id` no se incluye automáticamente porque en el origen es un atributo y en el destino es un elemento. Si necesita asignar `category` a `genre`, por ejemplo, la conexión de copia total ya no es posible porque se trata de nodos distintos.

Cuando un conector de entrada (el icono en forma de triángulo situado en el lateral del componente) recibe una conexión de copia total, no podrá aceptar ninguna otra conexión. Por ejemplo, en la imagen anterior, si intentamos crear una conexión entre `category` y `genre`, MapForce solicitará que reemplace la conexión o que cree un duplicado de entrada.



Crear un duplicado de la entrada (*véase* *Duplicar entradas*) solo tiene sentido si el objetivo es que el destino acepte datos de más de una entrada (este no es el caso en este ejemplo). Si elige reemplazar la conexión de copia total, MapForce emite otro mensaje solicitando que se resuelva

o elimine la conexión de copia total.



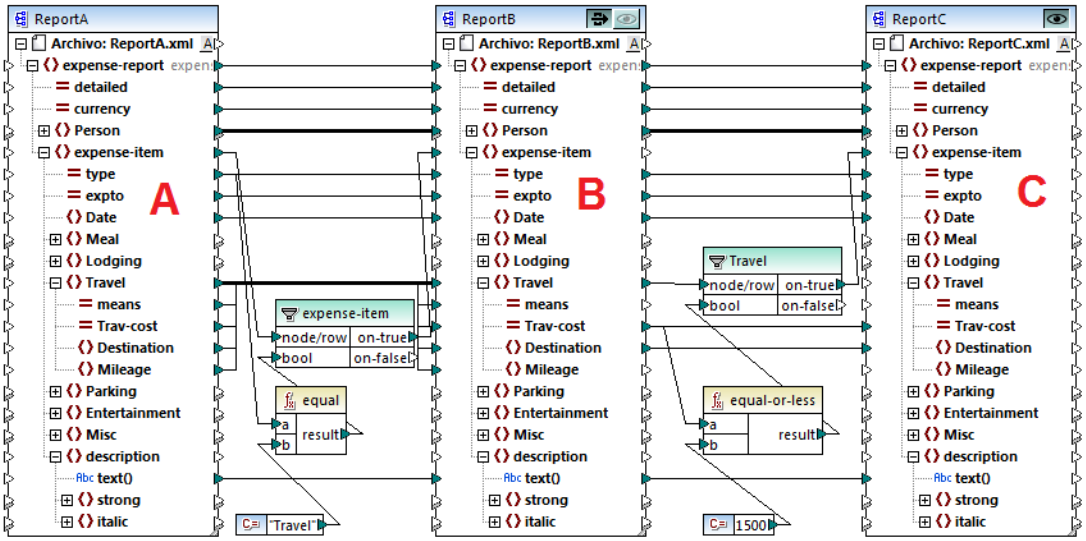
Haga clic en **Resolver la conexión de copia total** si desea reemplazar la conexión de copia total con varias conexiones estándar basadas en el destino. Si prefiere eliminar la conexión de copia total completamente, haga clic en **Eliminar conexiones secundarias**.

© 2018 Altova GmbH

5.3 Asignaciones en cadena

En MapForce puede crear asignaciones de datos formadas por varios componentes en cadena. Las asignaciones en cadena son asignaciones donde como mínimo un componente hace de origen y de destino a la vez. Este tipo de componentes crea un resultado que se usa después como entrada del siguiente paso de la asignación en cadena. A estos componentes se les denomina *componente intermedio*.

Por ejemplo, la asignación que aparece a continuación muestra un informe de gastos (en formato XML) que se procesa en dos etapas. La parte de la asignación que va de A a B filtra los gastos marcados como "Travel" (viaje). La parte que va de B a C filtra los gastos de viaje por un importe inferior a 1500. El componente B es el componente intermedio porque tiene conexiones de entrada y salida. Puede encontrar esta asignación en esta ruta de acceso: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\ChainedReports.mfd**.



ChainedReports.mfd

Las asignaciones en cadena presentan una característica llamada *paso a través.* Se trata de una función de vista previa que permite consultar los resultados que produce cada etapa de la asignación en cadena (en el panel *Resultados*). Por ejemplo, en la asignación del ejemplo puede consultar y guardar la vista previa del resultado XML que produce la etapa A - B y del resultado XML que produce la etapa B - C.

Nota: la característica *paso a través* solamente está disponible para componentes basados en archivos (como componentes XML, CSV y de texto). Los componentes de base de datos pueden ser componentes intermedios pero no disponen del botón **Paso a través**. El componente intermedio siempre se regenera desde cero cuando se crea la vista previa o se genera código. Esto no sería posible con bases de datos porque antes de regenerar la base de datos sería necesario eliminarla.

Si la asignación se ejecuta con MapForce Server o con código generado, entonces se ejecuta

toda la cadena de asignación. La asignación genera los archivos de salida necesarios en cada etapa de la cadena y el resultado de una etapa se reenvía como entrada de la siguiente etapa.

También se pueden generar nombres de archivo dinámicos a partir de los componentes intermedios. Es decir, se pueden crear conexiones con el elemento `Archivo:` de los componentes intermedios, siempre y cuando éstos estén configurados correctamente (*véase* *Procesar varios archivos de entrada o salida simultáneamente*).

Botón Vista previa

Tanto el componente B como el componente C tienen un botón de vista previa. Este botón genera en MapForce una vista previa del resultado intermedio de B, así como el resultado final de la asignación en cadena. Haga clic en el botón **Vista previa** del componente correspondiente y después haga clic en el panel *Resultados* para ver el resultado de la asignación.

Los componentes intermedios que tengan activo el botón **Paso a través** no pueden verse en la vista previa porque su botón **Vista previa** se deshabilita automáticamente (ya que no tiene sentido obtener una vista previa de los datos y pasarlos al mismo tiempo). Para ver el resultado de los componentes intermedios, primero es necesario desactivar el botón **Paso a través** y después hacer clic en el botón **Vista previa**.

Botón Paso a través

El componente intermedio B tiene un botón más en la barra de título que se llama **Paso a través**.

Si el botón **Paso a través** está **activo** , MapForce asigna dirige todos los datos a la ventana de vista previa de una vez, del componente A al B y después al C. Se crearán dos archivos de resultados:

- el resultado de asignar el componente A al componente intermedio B.
- el resultado de asignar el componente intermedio B al componente de destino C.

Si el botón **Paso a través** está **inactivo** , MapForce solamente ejecutará partes de la cadena de asignación. Los datos que se generan dependen de qué botones **Vista previa** están activos:

- Si está activo el botón **Vista previa** del componente B, se genera el resultado de asignar el componente A al componente B. La cadena de asignación se detiene en el componente B. El componente C no se utiliza para nada en la vista previa.
- Si está activo el botón **Vista previa** del componente C, se genera el resultado de asignar el componente intermedio B al componente C. Como el paso a través está inactivo, se interrumpe el encadenamiento automático para el componente B. Solamente se ejecuta la parte derecha de la cadena de asignación. Es decir, el componente A no se utiliza.

> Cuando el botón Paso a través está inactivo, es importante que en los campos *Archivo XML de entrada* y *Archivo XML de salida* del componente intermedio se especifiquen nombres de archivo idénticos. Esto garantiza que el archivo de salida que se genera durante la vista previa de la etapa A - B sirva de entrada para la vista previa de la etapa B - C. Además, en el código generado o en la ejecución con MapForce Server, esto garantiza que la cadena de asignación no se rompa.

Como ya dijimos anteriormente, si la asignación se ejecuta con MapForce Server o con el código

generado, se da salida a todos los componentes. Cuando así sea, el estado del botón **Paso a través** del componente B no se tiene en cuenta (tampoco se tiene en cuenta qué componente de visa previa está seleccionado). Por ejemplo, a partir de la asignación de la imagen se generarían dos archivos de resultados:

1. el archivo de salida que se obtiene de la asignación de A a B y
2. el archivo de salida que se obtiene de la asignación de B a C.

Los apartados Ejemplo: paso a través activo y Ejemplo: paso a través inactivo explican con más detalle cómo se transfieren los datos de origen cuando el botón **Paso a través** está activo e inactivo, respectivamente.

5.3.1 Ejemplo: paso a través activo

La asignación utilizada en este ejemplo se llama **ChainedReports.mfd** y está en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**. Esta asignación procesa un archivo XML llamado **ReportA.xml** que contiene gastos de viaje (ver más abajo). Para mayor claridad se omitieron la declaración de espacio de nombres y algunos elementos `expense-item`:

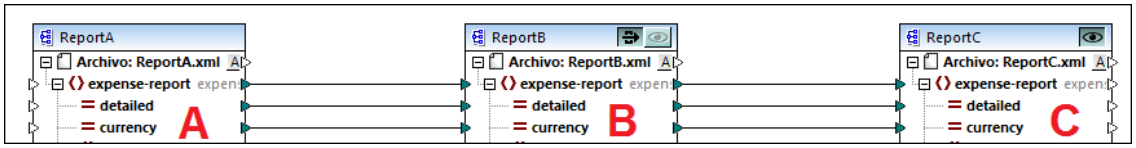
```
<?xml version="1.0" encoding="UTF-8"?>
<expense-report currency="USD" detailed="true">
   <Person>
      <First>Fred</First>
      <Last>Landis</Last>
      <Title>Project Manager</Title>
      <Phone>123-456-78</Phone>
      <Email>f.landis@nanonull.com</Email>
   </Person>
   <expense-item type="Travel" expto="Development">
      <Date>2003-01-02</Date>
      <Travel Trav-cost="337.88">
         <Destination/>
      </Travel>
      <description>Biz jet</description>
   </expense-item>
   <expense-item type="Lodging" expto="Sales">
      <Date>2003-01-01</Date>
      <Lodging Lodge-cost="121.2">
         <Location/>
      </Lodging>
      <description>Motel mania</description>
   </expense-item>
   <expense-item type="Travel" expto="Marketing">
      <Date>2003-02-02</Date>
      <Travel Trav-cost="2000">
         <Destination/>
      </Travel>
      <description>Hong Kong</description>
   </expense-item>
</expense-report>
```

Archivo ReportA.xml

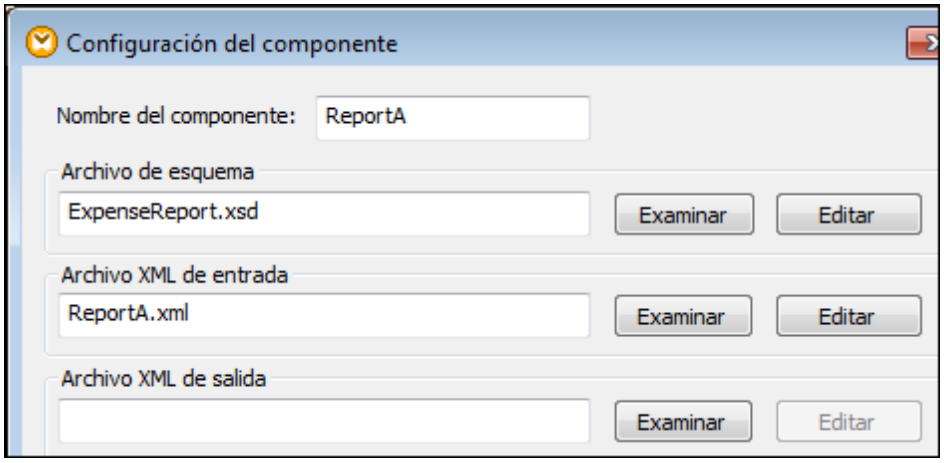
El objetivo de la asignación es producir dos informes a partir del archivo XML:

- El informe **ReportB.xml**, que debe contener los gastos de viaje que correspondan al tipo "Travel".
- El informe **ReportC.xml**, que debe contener los gastos de viaje que correspondan al tipo "Travel" y que no superen un valor de 1500.

Para conseguirlo el componente intermedio de la asignación (el componente B) tiene activado el botón **Paso a través** (*ver más abajo*). Esto hace que la asignación se ejecute en dos etapas: de A a B y de B a C. El resultado que genera el componente intermedio se usará como entrada para la asignación entre B y C.



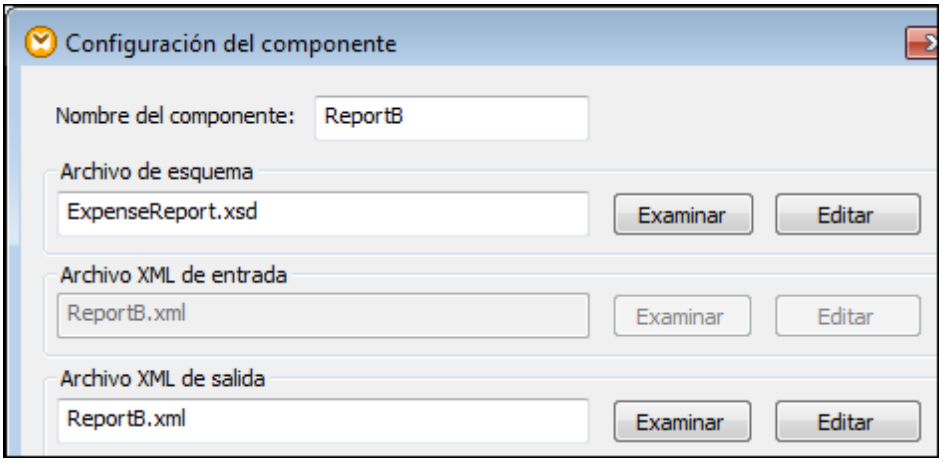
El nombre de los archivos de salida que se generan en cada etapa de la cadena de asignación viene dado por la configuración de cada componente. Para abrir la configuración de un componente haga clic con el botón derecho en el componente y seleccione **Propiedades** en el menú contextual. En nuestro ejemplo el primer componente está configurado para leer datos de un archivo XML llamado **ReportA.xml**. Como este es el componente de origen, el archivo de salida XML no es relevante y este campo se dejó vacío en la configuración del componente.



Configuración del componente de origen

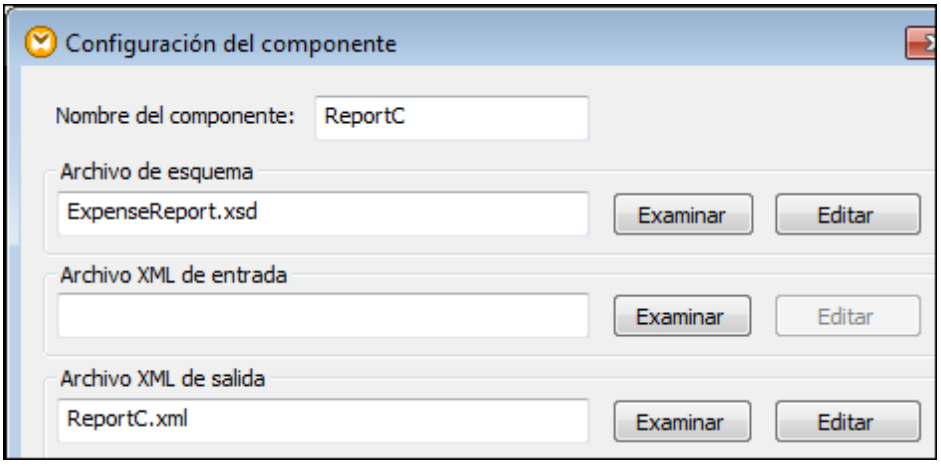
Como puede verse en la siguiente imagen, el segundo componente (**ReportB**) está configurado para generar un archivo de salida llamado **ReportB.xml**. Observe que el campo *Archivo XML de entrada* está atenuado. Esto se debe a que cuando el paso a través está activo (como en nuestro ejemplo) el campo *Archivo XML de entrada* del componente intermedio se desactiva automáticamente. Para poder ejecutar la asignación no se necesita un nombre de archivo de entrada porque el resultado que se crea en esta etapa de la cadena se almacena en un archivo temporal y se vuelve a utilizar en las etapas siguientes. Además, si se define un archivo XML de salida (*imagen siguiente*), éste se usará para el nombre de archivo del archivo de salida

intermedio. Si no se define el valor del campo *Archivo XML de salida*, se usará automáticamente un nombre de archivo predeterminado.



Configuración del componente intermedio

Por último, el tercer componente está configurado para generar un archivo de salida llamado **ReportC.xml**. El valor del campo *Archivo XML de entrada* no es relevante porque se trata del componente de destino.

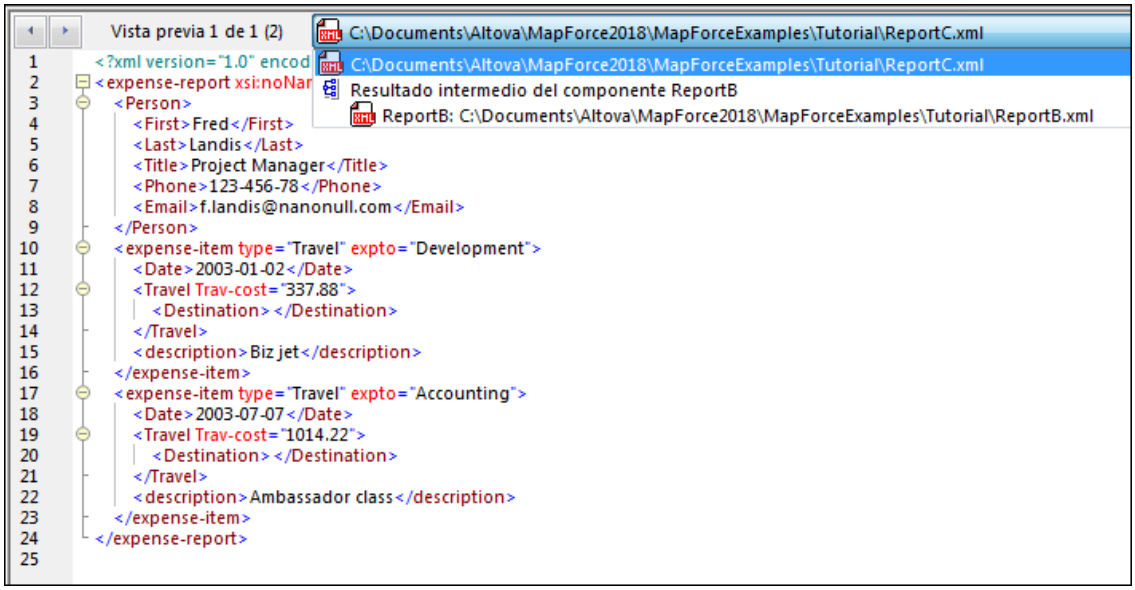


Configuración del componente de destino

Si consultamos la vista previa abriendo el panel *Resultados*, observaremos que se generaron dos archivos de salida:

1. **ReportB.xml**, que representa el resultado de la asignación de A a B.
2. **ReportC.xml**, que representa el resultado de la asignación de B a C.

Para seleccionar cuál de los dos archivos de salida se presentan en el panel *Resultados* haga clic en los botones de flecha o seleccione una entrada en la lista desplegable.

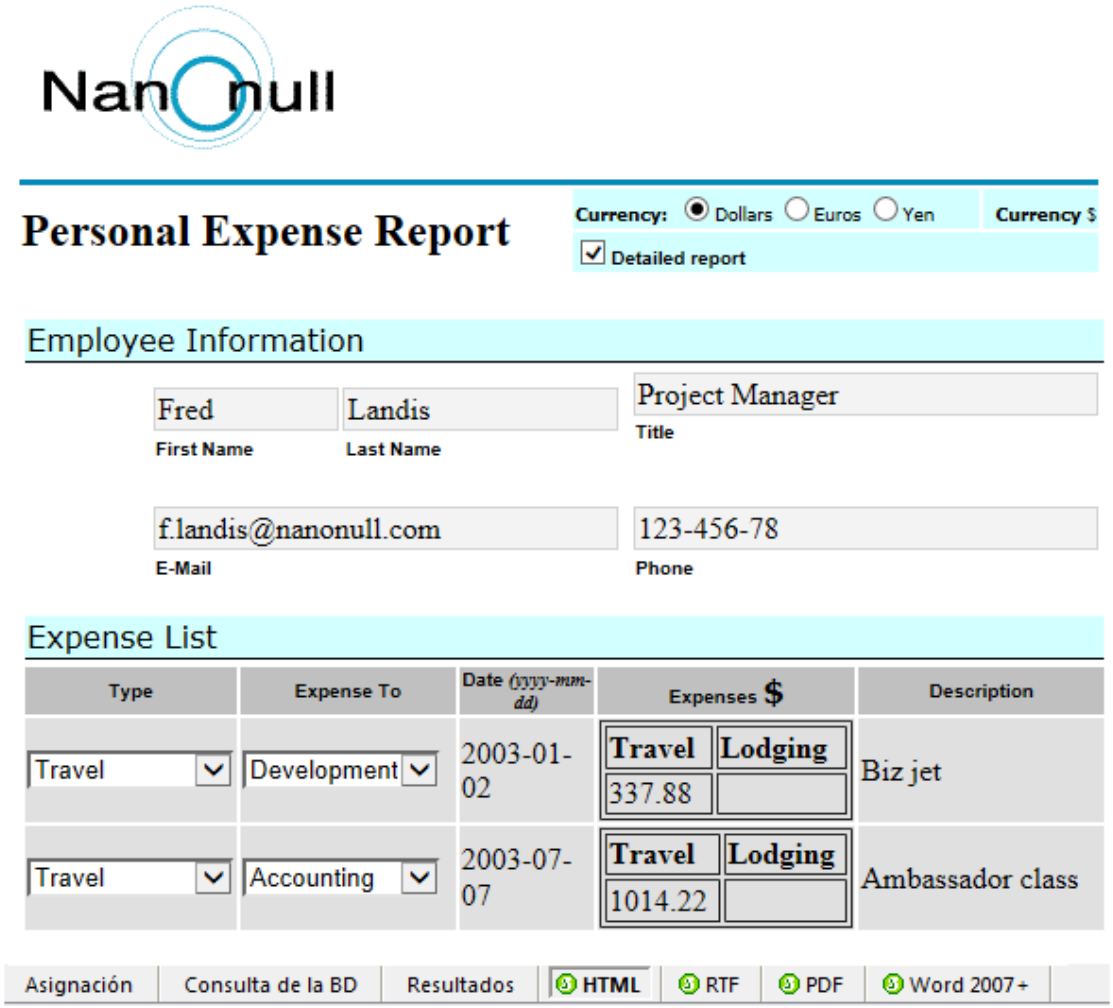


Archivos de salida generados

Cuando MapForce ejecuta la asignación, la configuración de la opción *Escribir directamente en archivos de salida finales* (que se configura desde **Herramientas | Opciones | Generales**) determina si los archivos intermedios se deben guardar como archivos temporales o físicos. Recuerde que esto solo es válido cuando se consulta una vista previa en MapForce. Si la asignación se ejecutara con MapForce o por medio de código, se producirían archivos en cada etapa de la cadena de asignación.

Si tiene StyleVision instalado y asignó un archivo SPS (StyleVision Power Stylesheet) al componente de destino (como en nuestro ejemplo), entonces podrá ver (y guardar) el resultado final de la asignación como HTML, RTF, PDF o Word. Para generar y consultar este resultado en MapForce basta con abrir el panel correspondiente.

© 2018 Altova GmbH

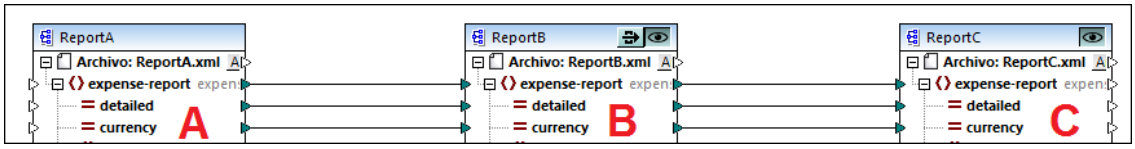


Resultado HTML generado

Recuerde que solamente se presenta el resultado del componente de destino final de la cadena de asignación. Para ver el resultado de StyleVision de componentes intermedios deberá desactivar el botón de paso a través y generar una vista previa del componente intermedio (*véase* *Ejemplo: paso a través inactivo*).

5.3.2 Ejemplo: paso a través inactivo

La asignación utilizada en este ejemplo se llama **ChainedReports.mfd** y está en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**. Este ejemplo muestra el resultado que genera la asignación cuando el botón **Paso a través** del componente intermedio está desactivado.

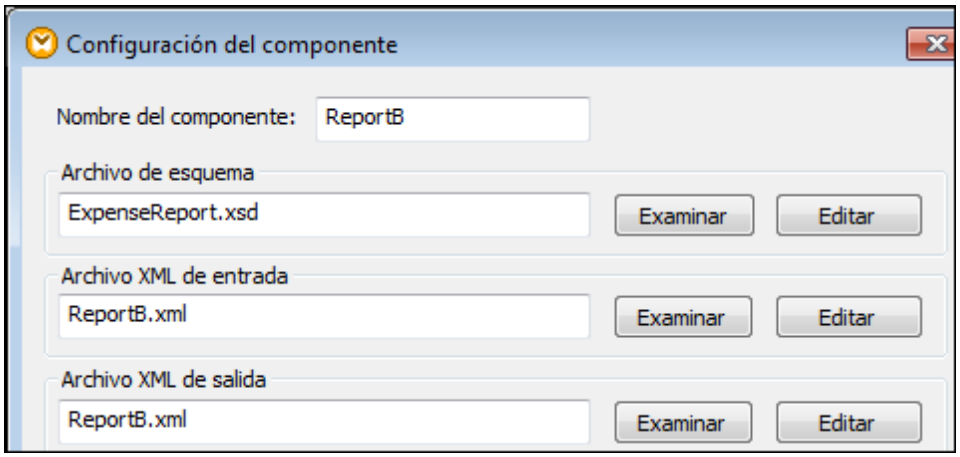


Como explicamos anteriormente en el apartado Ejemplo: paso a través activo, el objetivo de la asignación es producir dos informes distintos. En el ejemplo anterior, el botón **Paso a través** estaba activo y los informes generados podían verse en el panel *Resultados*. Sin embargo, si quiere consultar la vista previa de un solo informe (de **ReportB.xml** o de **ReportC.xml**), entonces debe desactivar el botón **Paso a través**. Desactivar el botón de paso a través puede ser de utilidad si su objetivo es:

- Consultar la vista previa del resultado que genera la asignación A - B solamente, ignorando la etapa B - C de la asignación.
- Consultar la vista previa del resultado que genera la asignación B - C solamente, ignorando la etapa A - B de la asignación.

Una vez desactivado el botón de paso a través, podrá elegir si consulta la vista previa de **ReportB** o de **ReportC** (observe que ambos tienen activo el botón **Vista previa**).

Además, desactivar el botón de paso a través permite elegir qué archivo de entrada debe leer el componente intermedio. En la mayoría de los casos, el archivo de entrada debería ser el mismo archivo que se definió en el campo *Archivo XML de salida* (como en este ejemplo).



Configuración del componente intermedio

El que el componente intermedio tenga el mismo archivo de entrada que de salida es importante si tiene pensado generar código a partir de la asignación o ejecutar la asignación con MapForce Server. Como se dijo anteriormente, en estos entornos se generan todos los resultados que crean los componentes de la cadena de asignación. Por tanto, lo más razonable es que el componente intermedio reciba un archivo (en este caso **ReportB.xml**), lo procese y lo reenvíe a la siguiente asignación, en lugar de buscar un nombre de archivo distinto. Tenga en cuenta que si en el componente intermedio el archivo de entrada y el de salida tienen nombres distintos (con el botón de paso a través inactivo), pueden producirse errores tipo *El sistema no encontró el archivo especificado* en el código generado o en la ejecución con MapForce Server.

 © 2018 Altova GmbH

Si hace clic en el botón **Vista previa**  del tercer componente (**ReportC**) e intenta generar la vista previa de resultados en MapForce, se producirá un error de ejecución. Se trata de un error esperado porque, según la configuración, se espera como entrada un archivo llamado **ReportB.xml** pero la asignación todavía no ha producido dicho archivo (porque el botón de paso a través no está activo y solo se ejecuta la etapa B - C de la asignación). Para solucionar este problema:

1. Haga clic en el botón de vista previa del componente intermedio.
2. Haga clic en el panel *Resultados* para generar una vista previa de los resultados.
3. Guarde el archivo de salida resultante como **ReportB.xml** en la carpeta donde está la asignación (**<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**).

Si ahora vuelve a hacer clic en el botón de vista previa del tercer componente (**ReportC**), el error ya no se produce.

Desactivar el botón de paso a través también permite consultar la vista previa de resultados generados con StyleVision creados a partir de todos los componentes que tengan un archivo SPS (StyleVision Power StyleSheet) asociado. También podrá ver la versión HTML del informe intermedio (además del informe final):



Resultado HTML del componente intermedio

© 2018 Altova GmbH

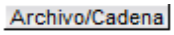
5.4 Procesar varios archivos de entrada o salida simultáneamente

MapForce puede configurarse para procesar varios archivos simultáneamente (p. ej. todos los archivos de un directorio) cuando se ejecute la asignación. Gracias a esta característica podrá:

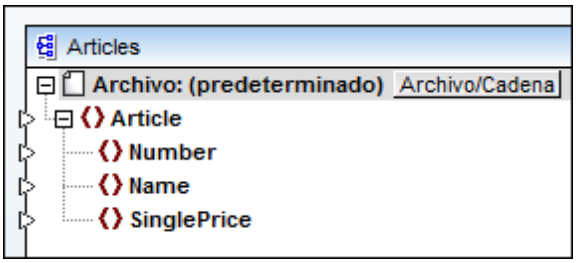
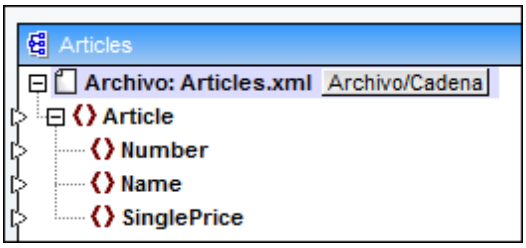
- Suministrar a la asignación una lista de archivos de entrada que se deben procesar.
- Generar como resultado de la asignación una lista de archivos en lugar de un solo archivo de salida.
- Generar una aplicación de asignación de datos donde el nombre de los archivos de entrada y salida se definen en tiempo de ejecución.
- Pasar un grupo de archivos a otro formato.
- Dividir un archivo (o base de datos) de gran tamaño en varios archivos más pequeños.
- Agrupar varios archivos diferentes en un solo archivo (o cargarlos en una base de datos).

Hay dos maneras de configurar componentes de MapForce para que procese varios archivos a la vez:

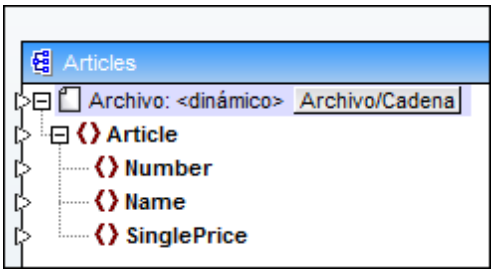
- Suministrando la ruta de acceso de los archivos de entrada/salida pertinentes por medio de caracteres de comodín (en lugar de dar un nombre de archivo fijo) en la configuración del componente (*véase* *Cambiar configuración de los componentes*). Es decir, puede introducir los comodines * y ? en el cuadro de diálogo "Configuración del componente" para que MapForce resuelva la ruta de acceso correspondiente cuando se ejecute la asignación.
- Conectando el nodo raíz de un componente con una secuencia que suministra la ruta de acceso de forma dinámica (p. ej. el resultado de la función **`replace-fileext`**). Cuando se ejecute la asignación, MapForce leerá de forma dinámica todos los archivos de entrada o generará todos los archivos de salida de forma dinámica.

Dependiendo de cual sea el objetivo del proyecto, podrá usar una de estas técnicas o ambas en la misma asignación. Sin embargo, no tiene sentido utilizar ambas técnicas al mismo tiempo en el mismo componente. Para indicar qué técnica desea usar para cada componente haga clic en el botón  o  disponible junto al nodo raíz de cada componente. Estos son los comportamientos que puede especificar con estos botones:

Usar nombres de archivo de la configuración del componente	Si el componente debe procesar un archivo de instancia o varios, esta opción ordena el procesamiento de los nombres de archivo definidos en el cuadro de diálogo "Configuración del componente". Si selecciona esta opción, el nodo raíz no tiene un conector de entrada porque no es relevante.

	 Si todavía no ha especificado archivos de entrada/ salida en el cuadro de diálogo "Configuración del componente", el nombre del nodo raíz será **`Archivo: (predeterminado)`**. De lo contrario el nodo raíz muestra el nombre del archivo de entrada seguido de un punto y coma más el nombre del archivo de salida. Si el nombre de la entrada es idéntico al archivo de salida, entonces aparece como nombre del nodo raíz.  Esta opción y la opción **Usar nombres de archivo dinámicos dados por la asignación** se excluyen mutuamente.
Usar nombres de archivo dinámicos dados por la asignación	Esta opción ordena el procesamiento de los nombres de archivo definidos en el área de asignación al conectar valores con el nodo raíz del componente. Si selecciona esta opción, el nodo raíz recibe un conector de entrada con el que se pueden conectar valores que suministren de forma dinámica los nombres de archivo que se deben procesar durante la ejecución de la asignación. Si también definió nombres de archivo en el cuadro de diálogo "Configuración del componente", dichos valores se omitirán. Cuando se selecciona esta opción, el nombre del nodo raíz aparece como **`Archivo: <dinámico>`**.

© 2018 Altova GmbH

	 Esta opción y la opción **Usar nombres de archivo de la configuración del componente** se excluyen mutuamente.
Redistribuir cadenas en XML, **Redistribuir cadenas en JSON**, **Redistribuir cadenas en CSV**, **Redistribuir cadenas en FLF**, **Redistribuir cadenas en EDI**	Si se activa, esta opción permite al componente aceptar un valor de cadena como entrada del nodo raíz y convertirlo en una estructura XML, JSON, CSV, FLF o EDI respectivamente. Consulte el apartado Analizar y serializar cadenas de texto para obtener más información.
Serializar XML en cadenas, **Serializar JSON en cadenas**, **Serializar CSV en cadenas**, **Serializar FLF en cadenas**, **Serializar EDI en cadenas**	Si se activa, esta opción permite al componente aceptar una estructura como entrada y convertirla en cadenas. La estructura de entrada puede ser XML, JSON, CSV, FLF o EDI respectivamente. Consulte el apartado Analizar y serializar cadenas de texto para obtener más información.

Estos son los componentes donde se pueden definir varios archivos de entrada/salida:

- Archivos XML
- Archivos de texto (archivos CSV*, FLF* y FlexText**)
- Documentos EDI**
- Hojas de cálculo Excel**
- Documentos XBRL**

** Con MapForce Professional Edition*
*** Con MapForce Enterprise Edition*

En esta tabla puede ver la compatibilidad de cada lenguaje de MapForce con archivos de entrada/salida dinámicos y con comodines:

Lenguaje de destino	Nombre de archivo de entrada dinámico	Nombre de archivo de entrada admite comodines	Nombre de archivo de salida dinámico
XSLT 1.0	*	Incompatible con XSLT 1.0	Incompatible con XSLT 1.0
XSLT 2.0	*	*(1)	*

XQuery	*	*(1)	Incompatible con XQuery
C++	*	*	*
C#	*	*	*
Java	*	*	*
BUILT-IN (motor integrado)	*	*	*

Leyenda:

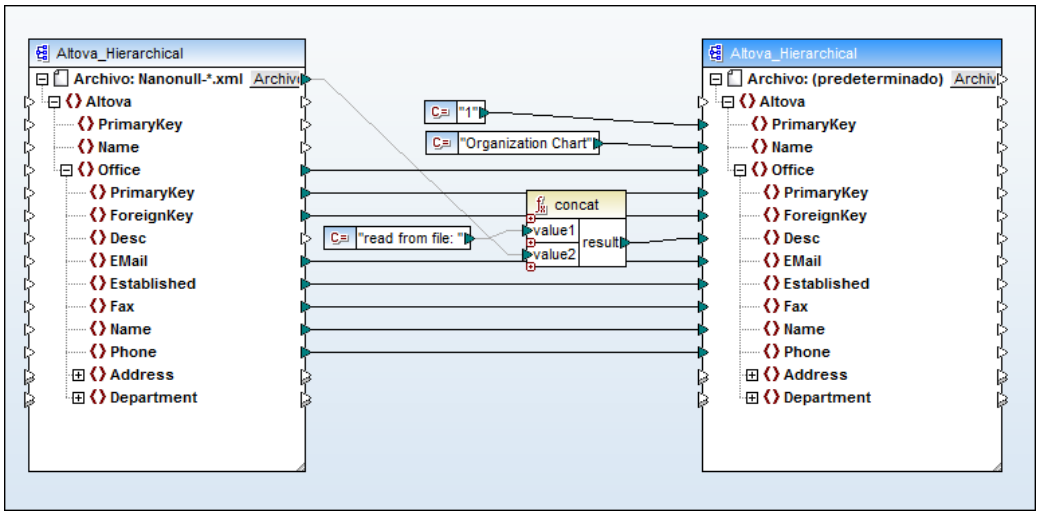
*	Compatible
(1)	Utiliza la función **`fn:collection`**. La implementación en los motores XSLT 2.0 y XQuery de Altova resuelve comodines. Los demás motores pueden comportarse de modo distinto. Para más información sobre transformación de código XSLT 1.0/2.0 y XQuery con el motor RaptorXML Server, consulte los apartados Generar código XSLT y Generar código XQuery 1.0.

5.4.1 Asignar varios archivos de entrada a un solo archivo de salida

Hay dos maneras de procesar varios archivos de entrada a la vez:

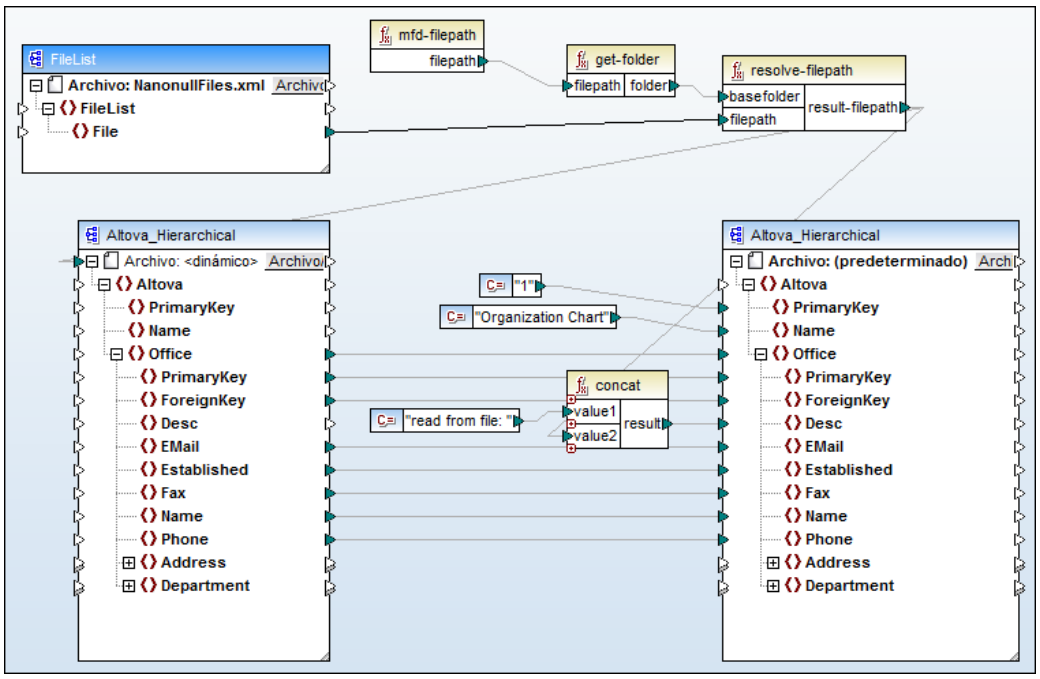
- Introduciendo una ruta de acceso con comodines (* o ?) como archivo de entrada en el cuadro de diálogo "Configuración del componente". Se procesarán todos los archivos que coincidan con la ruta de acceso indicada. En la imagen siguiente, por ejemplo, en el campo *Archivo XML de entrada* se usa el comodín * para indicar que todos los archivos cuyo nombre empiece por `Nanonull-` serán los archivos de entrada de la asignación. En este ejemplo se combinan varios archivos de entrada para crear **un solo archivo de salida**. Como se puede apreciar en la imagen, no hay ningún conector dinámico con el componente de destino y el componente de origen tiene acceso a varios archivos gracias al comodín *. Observe también que el nombre del nodo raíz del componente de destino es **`Archivo: <predeterminado>`**. Es decir, no se definió ninguna ruta de acceso de archivo de salida en el cuadro de diálogo "Configuración del componente". Por tanto, al documento de destino se anexan varios archivos de origen.

© 2018 Altova GmbH



MergeMultipleFiles.mfd (MapForce Basic Edition)

- Asignando una **secuencia** de cadenas al nodo **`Archivo`** del componente de origen. Cada cadena de la secuencia representa un nombre de archivo. Las cadenas también pueden contener comodines (que se resuelven automáticamente). Los componentes que pueden suministrar secuencias de nombres de archivos son los archivos XML, los campos de texto de BD, los archivos de texto (CSV o FLF) y las hojas Excel.

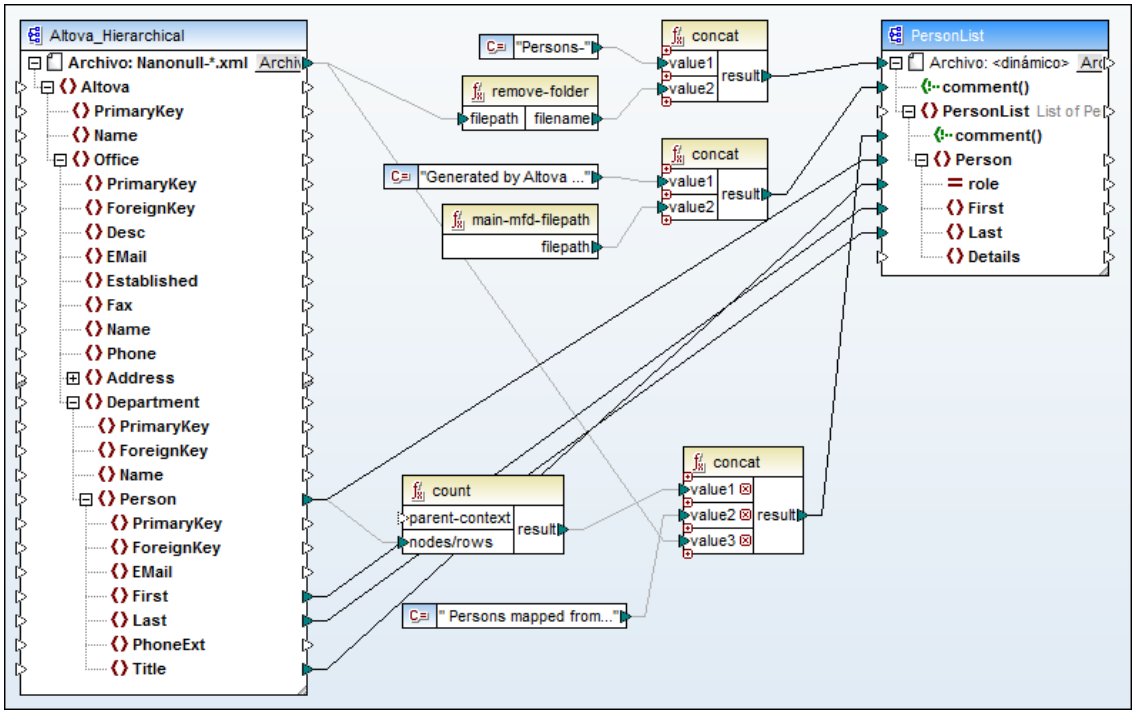


MergeMultipleFiles_List.mfd (MapForce Basic Edition)

5.4.2 Asignar varios archivos de entrada a varios archivos de salida

Para asignar varios archivos a varios archivos de destino es necesario generar nombres de archivo de salida únicos. En algunos casos, el nombre de los archivos de salida se puede derivar de cadenas existentes en los datos de entrada. En otros casos, lo más práctico es derivar el nombre de los archivos de salida del nombre del archivo de entrada (p. ej. cambiando la extensión de archivo).

En la asignación que aparece a continuación el nombre del archivo de salida se deriva del nombre del archivo de entrada: se añadió el prefijo **`Persons-`** con ayuda de la función **`concat`**.



MultipleInputToMultipleOutputFiles.mfd (MapForce Basic Edition)

Nota: recomendamos conectar simplemente los nodos raíz de entrada y salida directamente, sin usar funciones de procesamiento. Esto sobrescribiría los archivos de entrada cuando se ejecute la asignación. Puede cambiar los nombres de los archivos de salida con ayuda de funciones como la función **`concat`**.

Con el comando de menú **Archivo | Configurar asignación** puede definir la configuración de rutas de acceso global que debe utilizar la asignación (véase Cambiar configuración de la asignación).

5.4.3 Especificar nombres de archivo como parámetros de asignación

Para indicar nombres de archivo personalizados como parámetros de entrada de la asignación:

1. Añada un componente de entrada a la asignación (**Función | Insertar componente de**

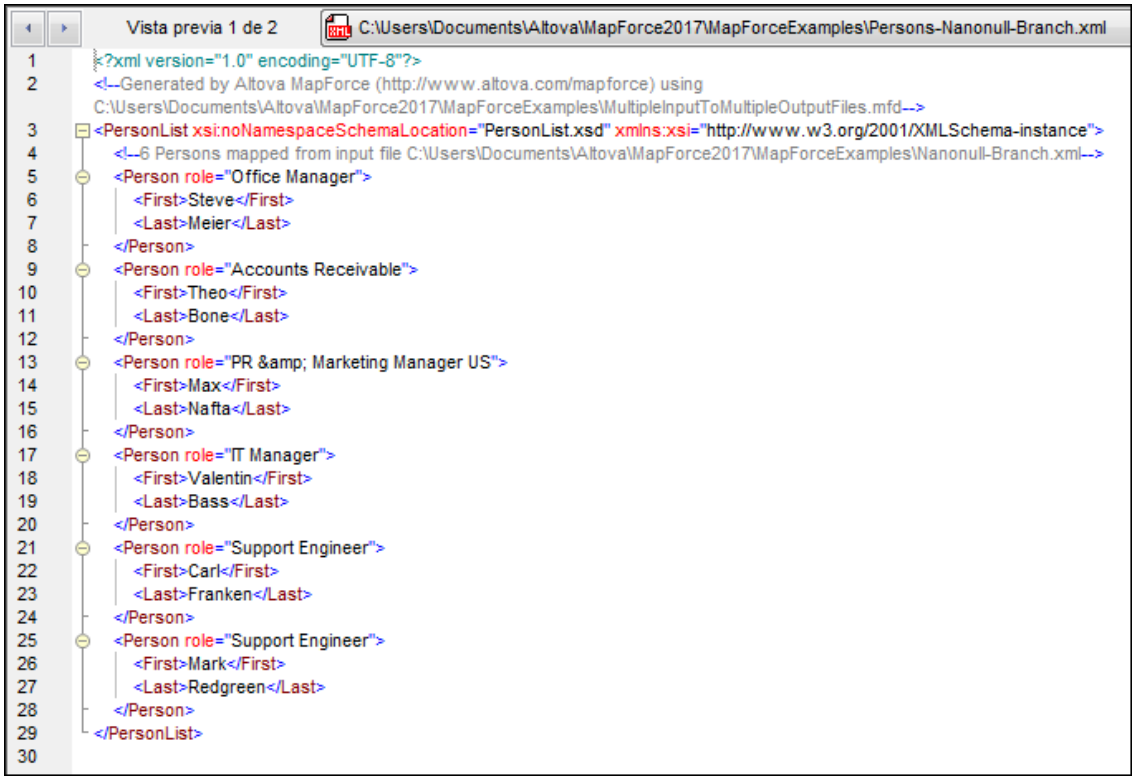
entrada). Para más información consulte el apartado Agregar componentes de entrada simples.

2. Haga clic en el botón Archivo o Archivo/Cadena del componente de origen y seleccione **Usar nombres de archivo dinámicos dados por la asignación** en el menú contextual.
3. Conecte el componente de entrada **input** con el nodo raíz del componente de origen.

Para ver un ejemplo más detallado consulte el apartado Ejemplo: usar nombres de archivo como parámetros de asignación.

5.4.4 Vista previa de varios archivos de salida

Haga clic en el panel *Resultados* para ver el resultado de la asignación. Si la asignación produce varios archivos de salida, cada archivo contará con un panel numerado en el panel *Resultados*. Haga clic en los botones en forma de flecha para navegar por los distintos archivos de salida.



MultipleInputToMultipleOutputFiles.mfd

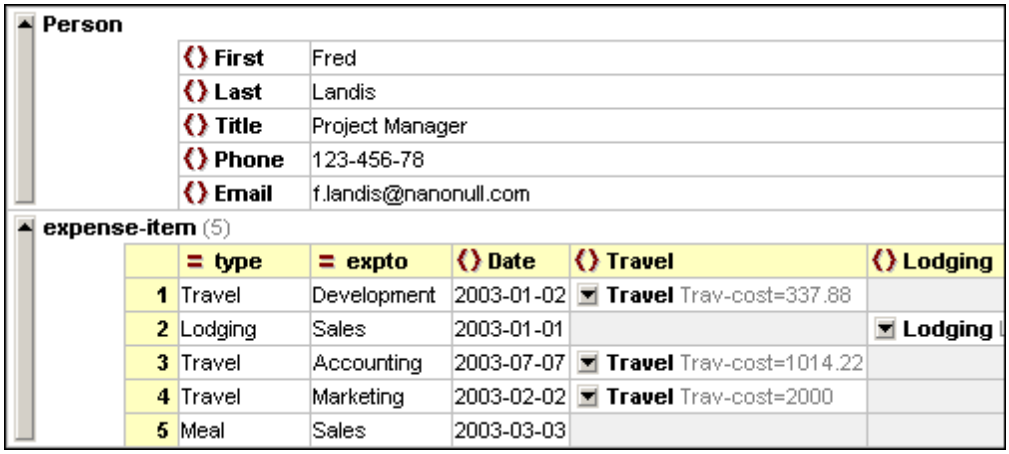
Hay dos maneras de guardar los archivos de salida generados:

- En el menú **Resultados** haga clic en el comando **Guardar todos los archivos de salida** ( ).
- Haga clic en el botón **Guardar todos** ( ) de la barra de herramientas.

5.4.5 Ejemplo: dividir un archivo XML en varios archivos

Este ejemplo demuestra cómo se pueden generar varios archivos XML de forma dinámica a partir de un solo archivo XML de origen. Este ejemplo es el archivo de muestra **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial\Tut-ExpReport-dyn.mfd**.

El archivo XML de origen (situado en la misma carpeta que la asignación) está compuesto por un informe de gastos de alguien llamado *Fred Landis* y contiene cinco conceptos de gastos de diferentes tipos. El objetivo de esta asignación de datos es generar un archivo XML por cada concepto de gasto.



Person		
First	Fred	
Last	Landis	
Title	Project Manager	
Phone	123-456-78	
Email	f.landis@nanonull.com	

expense-item (5)

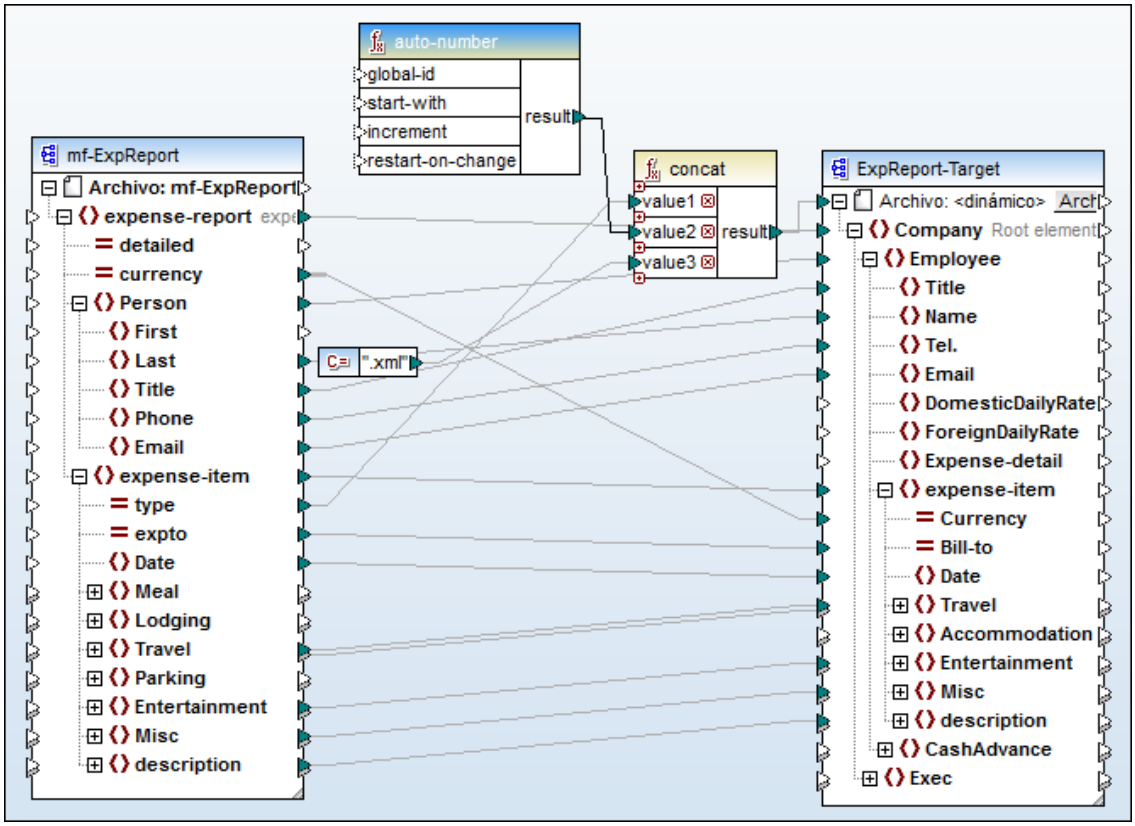
	= type	= expto	Date	Travel	Lodging
1	Travel	Development	2003-01-02	**Travel** Trav-cost=337.88	
2	Lodging	Sales	2003-01-01		**Lodging**
3	Travel	Accounting	2003-07-07	**Travel** Trav-cost=1014.22	
4	Travel	Marketing	2003-02-02	**Travel** Trav-cost=2000	
5	Meal	Sales	2003-03-03		

mf-ExpReport.xml (en la vista Cuadrícula de Altova XMLSpy)

Como el atributo `type` define el tipo de gasto, este será el elemento que usaremos para dividir el archivo de origen. Para conseguir nuestro objetivo debemos seguir estos pasos:

1. Insertamos una función **`concat`** (se puede arrastrar desde la biblioteca de funciones **core | string functions** de la ventana Bibliotecas).
2. Insertamos una constante (clic en el comando **Insertar | Constante**) e introducimos el valor ".xml".
3. Insertamos la función **`auto-number`** (se puede arrastrar desde la biblioteca de funciones **core | generator functions** de la ventana Bibliotecas).
4. Hacemos clic en el botón Archivo o Archivo/Cadena del componente de destino y seleccionamos **Usar nombres de archivo dinámicos dados por la asignación**.
5. Para terminar creamos las conexiones que se ven en la imagen siguiente y hacemos clic en el panel *Resultados* para ver el resultado de la asignación.

© 2018 Altova GmbH



Tut-ExpReport-dyn.mfd (MapForce Basic Edition)

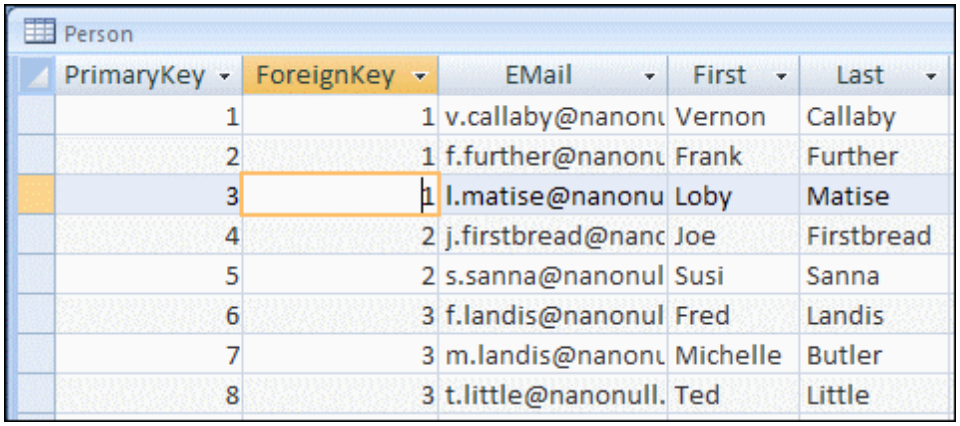
Tenga en cuenta que los archivos de salida resultantes tendrán nombres dinámicos construidos de la siguiente manera:

- El atributo `type` aporta la primera parte del nombre de archivo (p. ej. "Travel").
- La función **`auto-number`** aporta el número secuencial del archivo (p. ej. "Travel1", "Travel2" y así sucesivamente).
- La constante aporta la extensión del archivo, que es ".xml". Por tanto, el nombre del primer archivo de salida es "Travel1.xml".

5.4.6 Ejemplo: dividir una tabla de BD en varios archivos XML

Este ejemplo demuestra cómo se pueden generar varios archivos XML de forma dinámica a partir de cada registro de una tabla de BD. Este ejemplo es el archivo de muestra **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\PersonDB-dyn.mfd**.

El archivo de BD de origen (situado en la misma carpeta que la asignación) contiene una tabla llamada Person que tiene 21 registros. El objetivo de esta asignación es generar un archivo XML de salida por cada registro de la tabla Person.



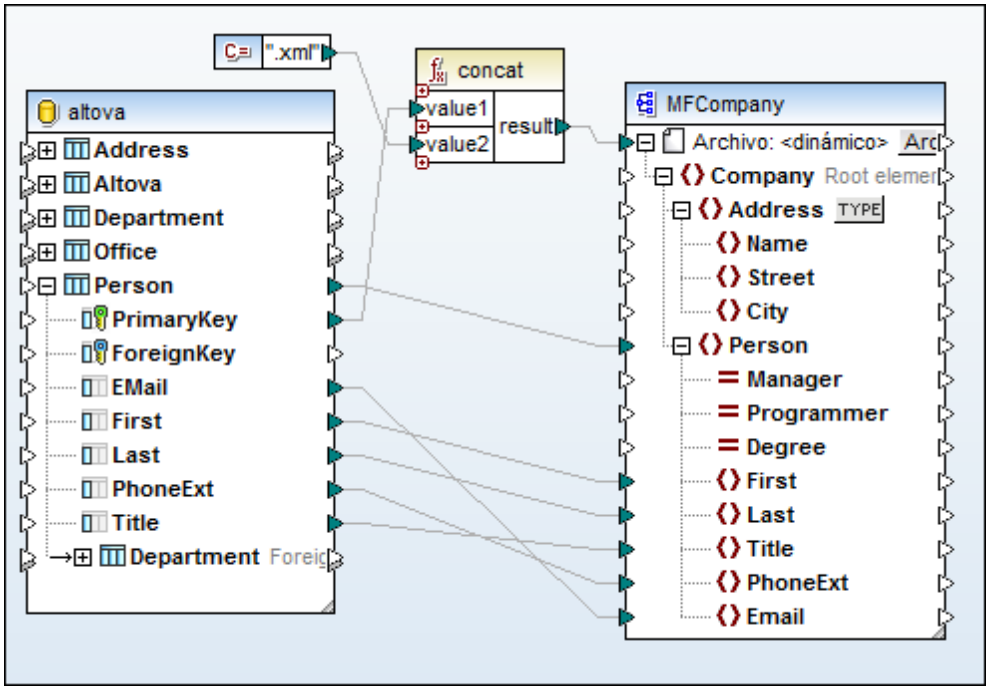
Person

PrimaryKey	ForeignKey	EMail	First	Last
1	1	v.callaby@nanonu	Vernon	Callaby
2	1	f.further@nanonu	Frank	Further
3	1	l.matise@nanonu	Loby	Matise
4	2	j.firstbread@nanc	Joe	Firstbread
5	2	s.sanna@nanonul	Susi	Sanna
6	3	f.landis@nanonul	Fred	Landis
7	3	m.landis@nanonu	Michelle	Butler
8	3	t.little@nanonull.	Ted	Little

Como el campo "PrimaryKey" identifica a todas las personas de la tabla con una clave única, lo utilizaremos para dividir la base de datos de origen en varios archivos. Para conseguir nuestro objetivo debemos seguir estos pasos:

1. Insertamos una función **`concat`** (se puede arrastrar desde la biblioteca de funciones **core | string functions** de la ventana Bibliotecas).
2. Insertamos una constante (clic en el comando **Insertar | Constante**) e introducimos el valor ".xml".
3. Hacemos clic en el botón Archivo o Archivo/Cadena del componente de destino y seleccionamos **Usar nombres de archivo dinámicos dados por la asignación**.
4. Para terminar creamos las conexiones que se ven en la imagen siguiente y hacemos clic en el panel *Resultados* para ver el resultado de la asignación.

© 2018 Altova GmbH



PersonDB-dyn.mfd (MapForce Professional Edition)

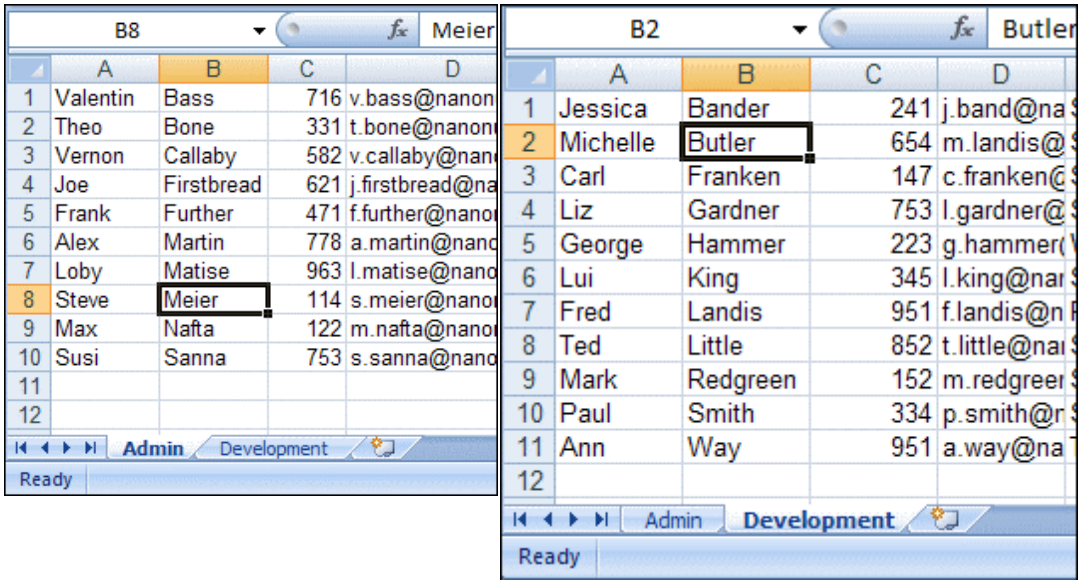
Tenga en cuenta que los archivos de salida resultantes tendrán nombres dinámicos construidos de la siguiente manera:

- El campo **PrimaryKey** aporta la primera parte del nombre de archivo (p. ej. "1").
- La constante aporta la extensión de archivo (".xml"). Por tanto, el nombre del primer archivo de salida es "1.xml".

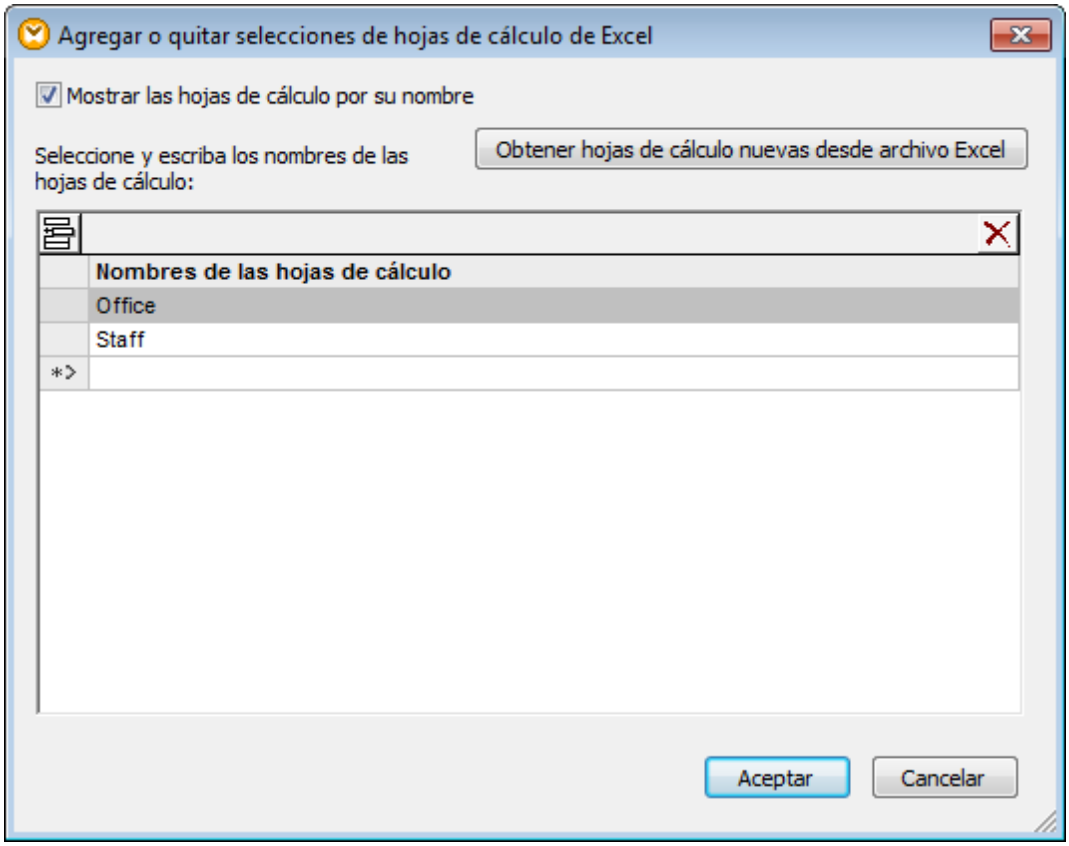
5.4.7 Varios archivos XML a partir de filas Excel

En las imágenes que aparecen a continuación puede ver el contenido del archivo Excel **altova.xlsx** (disponible en la carpeta ...\MapForceExamples\Tutorial). Como puede ver, este archivo está formado por dos hojas de cálculo: la hoja Admin, que tiene 10 filas de datos, y la hoja Development, que tiene 11 filas de datos. El ejemplo de asignación que usaremos para este apartado es el archivo **Excel-Mapping-dyn.mfd** situado en la carpeta ...\MapForceExamples\Tutorial.

Hoja de cálculo Admin Hoja de cálculo Development



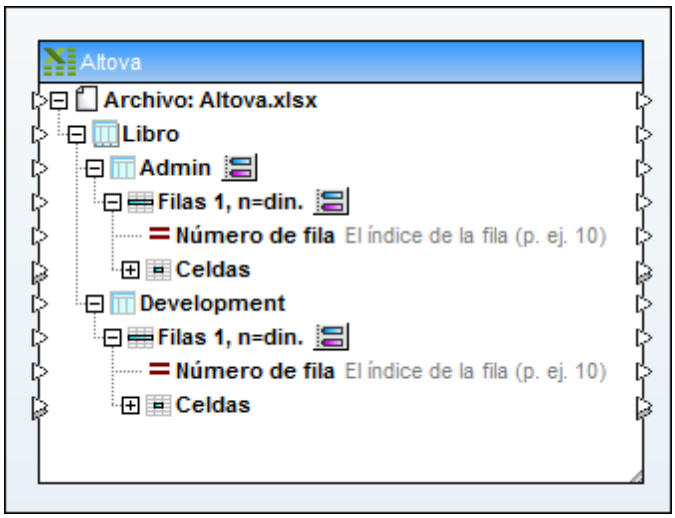
MapForce es capaz de presentar y asignar componentes Excel de dos formas distintas, dependiendo de la configuración del componente. En el cuadro de diálogo que aparece a continuación puede ver la configuración predeterminada. La casilla *Mostrar las hojas de cálculo por su nombre* se activa cuando se inserta el componente Excel.



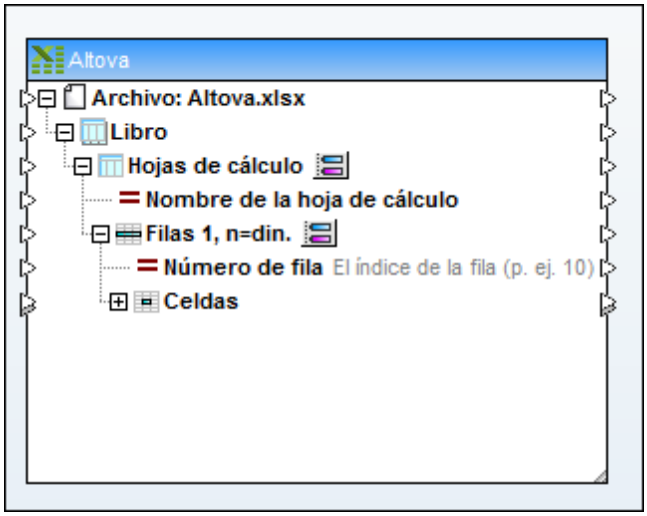
Para acceder al libro Excel como si fuera una sola hoja de cálculo:

1. Haga clic en el icono situado junto a **Admin** en el componente Excel.

 © 2018 Altova GmbH



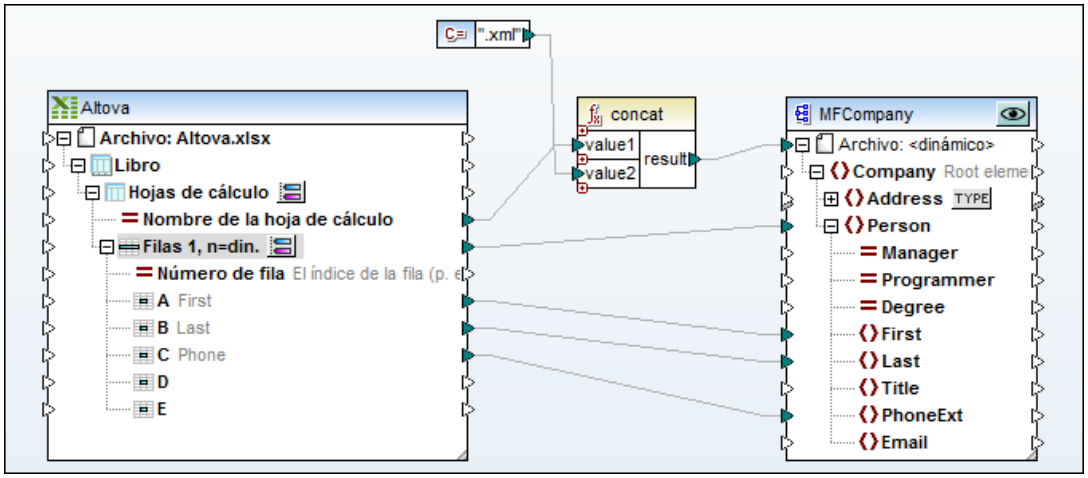
2. Haga clic en la casilla *Mostrar las hojas de cálculo por su nombre* para desactivarla. Las hojas de cálculo que tengan nombre dejan de ser visibles (*imagen siguiente*) pero ahora aparece un elemento llamado **`Nombre de la hoja de cálculo`**.



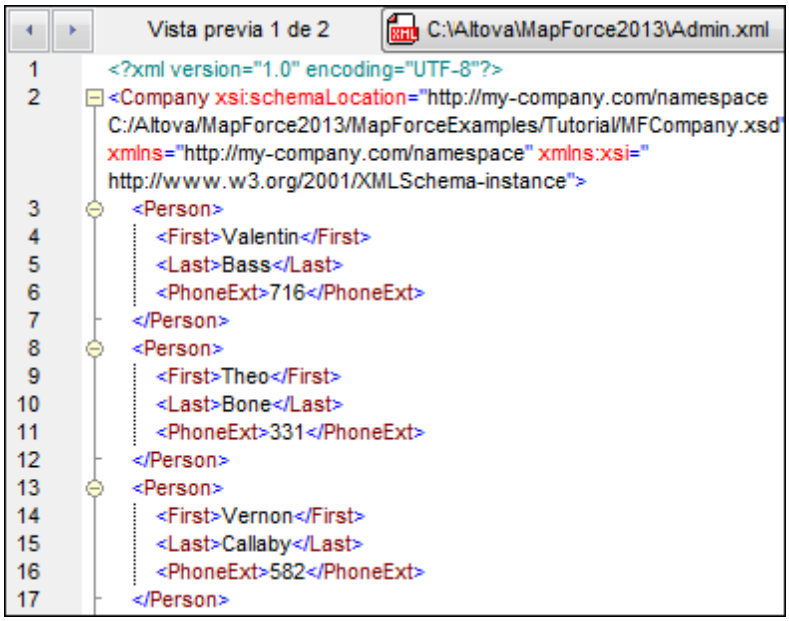
Objetivo nº1: generar archivos de salida distintos por cada hoja de cálculo.

El elemento **`Nombre de la hoja de cálculo`** determina las diferentes hojas de cálculo del libro. Por ello usaremos este elemento para dividir el libro de origen en varios archivos de salida.

1. Insertamos una función **`concat`** y una constante.



2. Creamos las conexiones que aparecen en la imagen anterior (**`Nombre de la hoja de cálculo`** se conecta a `value1` y la constante se conecta con `value2`).
3. Conectamos el parámetro **result** de la función **`concat`** con el elemento **`Archivo:`** del componente de destino. Observe que ahora aparece **`Archivo: <dinámico>`** en el componente.
4. Definimos el resto de conexiones que necesitamos.
5. Para terminar hacemos clic en el panel *Resultados* para ver el resultado de la asignación.



Ahora cada registro tiene una pestaña propia en el panel *Resultados* (*imagen anterior*). Puede consultar todos los archivos de salida con las flechas **Siguiente/Anterior** del panel *Resultados*.

Nota:

- El campo **`Nombre de la hoja de cálculo`** aporta la primera parte del nombre del archivo de salida (p. ej. Admin).
- La constante aporta la extensión de archivo (**.xml**). Por tanto, el nombre del primer archivo es Admin.xml. Este archivo contiene todas las filas de la pestaña Excel. El

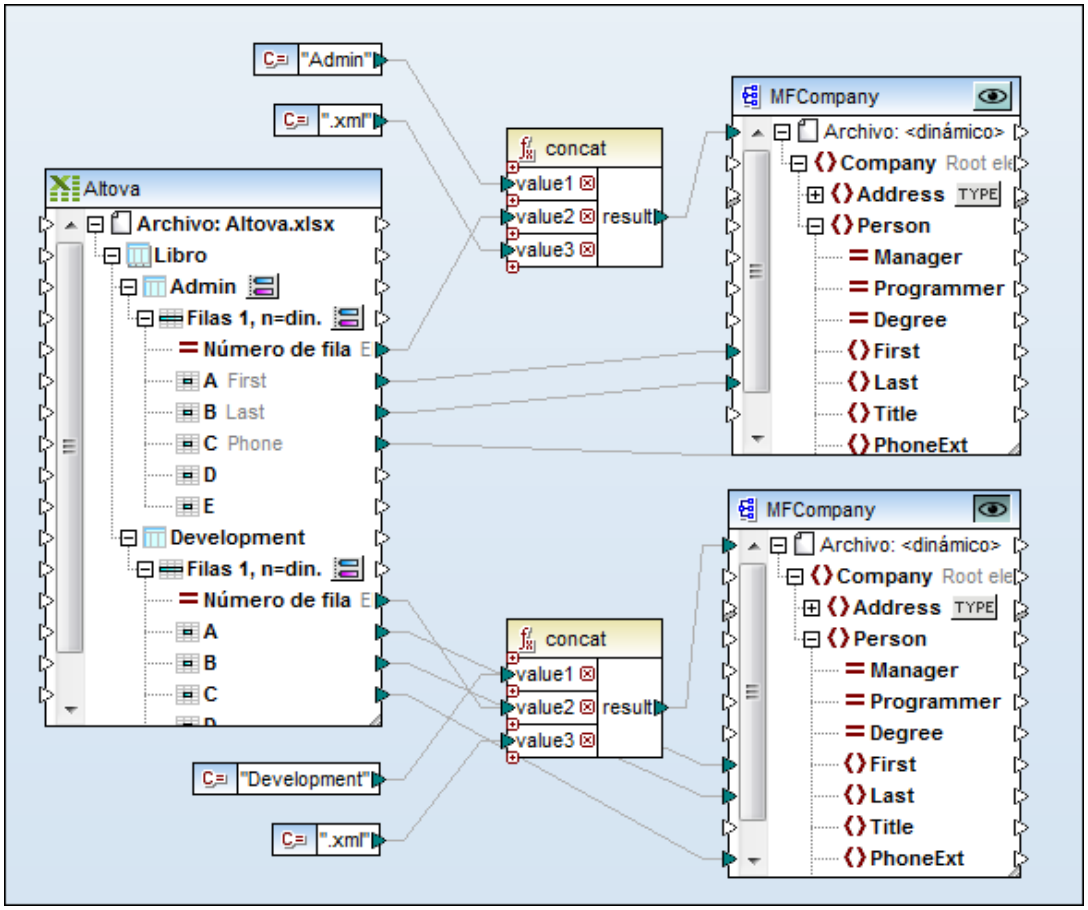
 © 2018 Altova GmbH

archivo Development.xml contiene las demás filas.

- Puede hacer clic en el icono **Guardar todos** para guardar todos los archivos de salida sin necesidad de generar código.

Objetivo nº2: generar archivos de salida distintos por cada registro Person de cada hoja de cálculo

1. Hacemos clic en el icono situado junto a Libros y marcamos la casilla *Mostrar las hojas de cálculo por su nombre*.
   Ahora se pueden ver todos los libros en el componente: Admin y Development.



2. Insertamos una función **concat** y una constante tal y como aparecen en la imagen anterior.
3. Insertamos otro archivo de esquema XML con el mismo nombre y creamos las conexiones que aparecen en la imagen anterior.

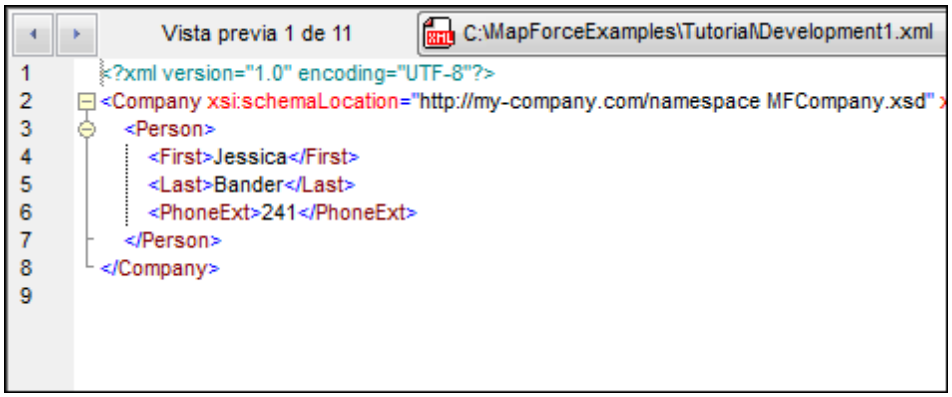
   Como el elemento **Número de fila** determina la fila de cada persona en la hoja de cálculo, usaremos este elemento para dividir cada hoja de cálculo en varios archivos.

Para el elemento de hoja de cálculo **Admin** del componente XLSX llamado **Altova**:

1. Creamos las conexiones que aparecen en la imagen anterior: la constante **Admin** se conecta con `value1`, **Número de fila** se conecta con `value2` y la constante **.xml** se

conecta con `value3`.

2. Conectamos el parámetro **`result`** de la función **`concat`** con el elemento **`Archivo:`** del componente de destino. Observe que ahora aparece **`Archivo: <dinámico>`** en el componente.
3. Definimos el resto de conexiones que necesitamos.
4. Para terminar hacemos clic en el panel *Resultados* para ver el resultado de la asignación.



Cada fila cuenta con una pestaña propia en el panel *Resultados* (*imagen anterior*). Los 10 registros de la hoja de cálculo Admin se dividieron en archivos de salida diferentes.

Puede consultar todos los archivos de salida con las flechas **Siguiente/Anterior** del panel *Resultados*.

Notas:

- La **constante** Admin aporta la primera parte del nombre del archivo de salida (p. ej. Admin).
- El elemento **`Número de fila`** aporta el número de fila de la hoja de cálculo Excel (p. ej. 1).
- La **constante** .xml aporta la extensión de archivo que se utiliza para guardar el archivo de salida (es decir, Admin1.xml).
- Puede hacer clic en el icono **Guardar todos** para guardar todos los archivos de salida sin necesidad de generar código.

Para ver el resultado de la hoja de cálculo Development haga clic en el botón **Vista previa** de ese componente y después en el panel *Resultados*. En este caso los 11 registros de la hoja de cálculo se dividieron en archivos de salida diferentes.

© 2018 Altova GmbH

5.5 Pasar parámetros a la asignación

Puede pasar valores simples a una asignación por medio de componentes de entrada simples. En el área de asignación los componentes de entrada simple desempeñan el papel de componente de origen que tiene un tipo de datos simple (p. ej. cadena, entero, etc.) en lugar de una estructura de elementos y secuencias. Por consiguiente, puede crear un componente de entrada simple en lugar de (o además de) un componente de origen basado en un archivo.

Puede usar componentes de entrada simples con cualquier lenguaje de transformación de MapForce:

- BUILT-IN (desde el panel *Resultados* para obtener la vista previa de la transformación de la asignación en MapForce directamente)
- BUILT-IN (cuando se ejecuta un archivo de ejecución de MapForce Server compilado)
- XSLT 1.0, XSLT 2.0
- XQuery
- C++
- C#
- Java

Cuando se trate de asignaciones ejecutadas con MapForce Server o por medio de código generado, los componentes de entrada simples se convierten en parámetros de la línea de comandos. Cuando se trate de asignaciones generadas como transformaciones XSLT, los componentes de entrada simples corresponden a parámetros de hojas de estilos en el archivo XSLT que se genera.

Cada componente de entrada simple (o parámetro) se puede crear como componente opcional u obligatorio (*véase Configurar componentes de entrada simples*). Si fuera necesario, también podrá crear valores predeterminados para los parámetros de entrada de la asignación (*véase Crear un valor de entrada predeterminado*). Esto le permitirá ejecutar la asignación de forma segura aunque no aporte explícitamente un valor de parámetro en tiempo de ejecución de la asignación.

Los parámetros de entrada añadidos en el área de asignación principal no se deben confundir con los parámetros de entrada de las funciones definidas por el usuario (*véase Funciones definidas por el usuario*). En esta tabla puede ver en qué se parecen y en qué se diferencian:

Parámetros de entrada de la asignación	**Parámetros de entrada de funciones definidas por el usuario**
Se añaden con el comando **Función | Insertar componente de entrada**.	Se añaden con el comando **Función | Insertar componente de entrada**.
Pueden tener tipos de datos simples (cadena, entero, etc.).	Puede tener tipos de datos simples y complejos.
Afectan a toda la asignación.	Afectan solo al contexto de la función donde se definen.

Cuando cree una asignación invertida (con el comando de menú **Herramientas | Crear asignación inversa**), los componentes de entrada simples se convertirán en componentes de

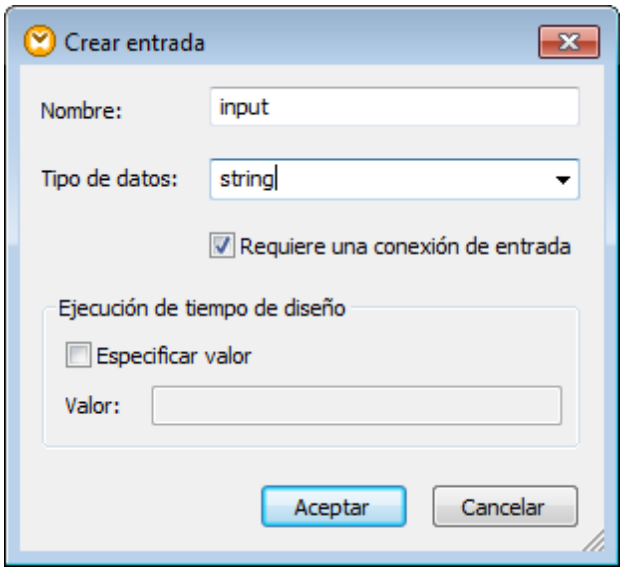
salida simples.

Puede consultar un ejemplo en el apartado Ejemplo: usar nombres de archivo como parámetros de asignación.

5.5.1 Agregar componentes de entrada simples

Para agregar un componente de entrada simple a la asignación:

1. Asegúrese de que en la ventana de asignación está activa la asignación principal (y no una función definida por el usuario).
2. En el menú **Función** haga clic en **Insertar componente de entrada**.
3. Introduzca un nombre y seleccione el tipo de datos que requiere esta entrada. Si la entrada debe tratarse como un parámetro obligatorio de la asignación, marque la casilla Requiere una conexión de entrada. (Para más información sobre las demás opciones de configuración consulte el apartado Configurar componentes de entrada simples.)
4. Haga clic en **Aceptar**.



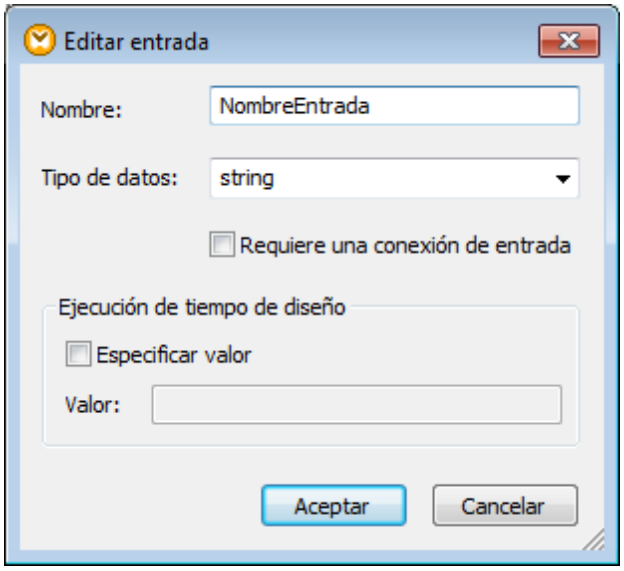
Cuadro de diálogo "Crear entrada"

Las demás opciones de configuración se describen en el apartado Configurar componentes de entrada simples.

5.5.2 Configurar componentes de entrada simples

Los componentes de entrada simples se pueden configurar en el momento en el que se añaden al área de asignación o más tarde. Esto se hace en el cuadro de diálogo "Crear entrada" o "Editar entrada" respectivamente.

© 2018 Altova GmbH



Cuadro de diálogo "EdEdit Input dialog box

Para abrir el cuadro de diálogo "Editar entrada" tiene tres opciones:

- Seleccione el componente y seleccione el comando de menú **Componente | Propiedades**.
- Haga doble clic en el componente.
- Haga clic con el botón derecho en el componente y después elija **Propiedades** en el menú contextual.

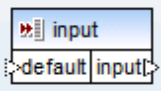
Estas son las opciones de configuración que ofrece el cuadro de diálogo:

Nombre	Introduzca un nombre descriptivo para el parámetro de entrada que corresponde a este componente. En tiempo de ejecución el valor que se introdujo en este campo pasa a ser el nombre del parámetro que se pasa a la asignación. Por tanto, no está permitido el uso de espacios ni caracteres especiales.
Tipo de datos	Todos los parámetros de entrada se tratan por defecto como tipo de datos de cadena. Si el parámetro debe tener otro tipo de datos, seleccione el valor pertinente en la lista. Cuando la asignación se ejecute, MapForce convertirá el tipo del parámetro de entrada en el tipo de datos que se seleccionó aquí.
Requiere una conexión de entrada	Si marca esta casilla, el parámetro de entrada será obligatorio (es decir, la asignación no se podrá ejecutar a no ser que se aporte un valor de parámetro). Desactive esta casilla si desea especificar un valor predeterminado para el parámetro de entrada (*véase* *Crear un valor de entrada predeterminado*).
Especificar valor	Esta casilla solo es relevante cuando se ejecuta la asignación en tiempo de diseño (cuando se hace clic en el panel *Resultados*). Permite introducir

	directamente en el componente el valor que se debe usar como entrada de la asignación.
Valor	Este campo solo es relevante cuando se ejecuta la asignación en tiempo de diseño (cuando se hace clic en el panel *Resultados*). Para introducir el valor que debe usar MapForce como entrada de la asignación, marque la casilla *Especificar valor* y después escriba el valor en este campo.

5.5.3 Crear un valor de entrada predeterminado

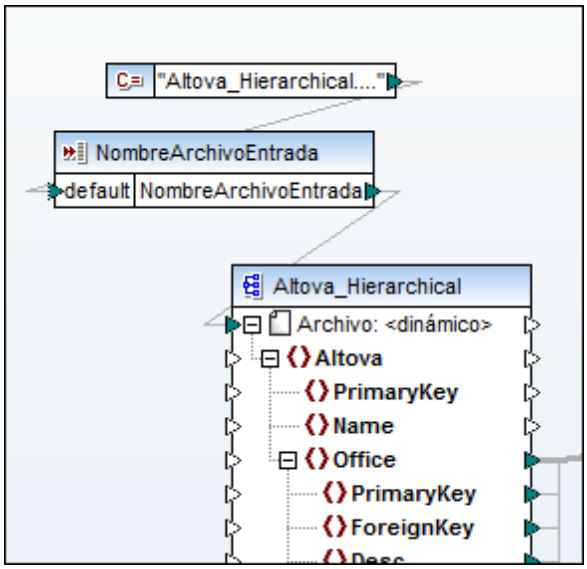
Tras añadir el componente de entrada al área de asignación, debemos fijarnos en el elemento **`default`** situado en el lado izquierdo del componente.



Componente de entrada simples

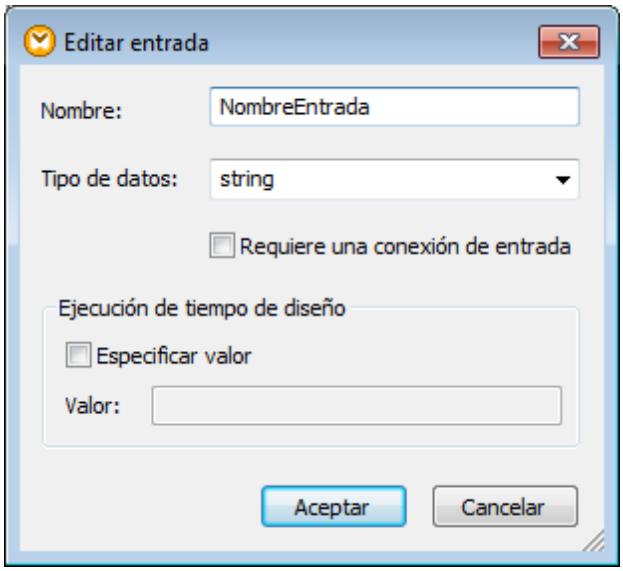
El elemento **`default`** permite conectar un valor predeterminado opcional a este componente de entrada. Esto se hace de la siguiente manera:

1. Añada un componente de constante (con el comando de menú **Insertar | Constante**) y después conéctelo con el elemento **`default`** del componente de entrada.



2. Haga doble clic en el componente de entrada y compruebe que la casilla *Requiere una conexión de entrada* está desactivada. Cuando se crea un valor de entrada predeterminado, esta casilla no es relevante y causa advertencias de validación.

 © 2018 Altova GmbH



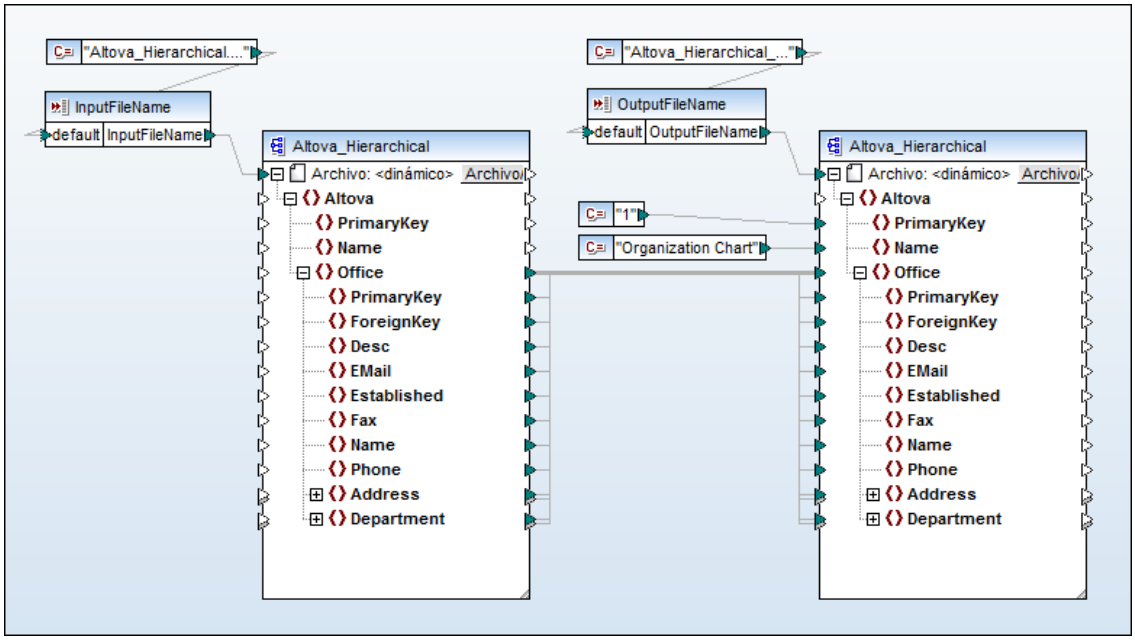
3. Haga clic en **Aceptar** para terminar.

Nota: si marca la casilla *Especificar valor* e introduce un valor en el campo adyacente, dicho valor tendrá prioridad sobre el valor predeterminado cuando consulte la vista previa de la asignación (es decir, cuando ejecute la asignación en tiempo de diseño). Sin embargo, ese valor no tendrá efecto alguno en el código generado.

5.5.4 Ejemplo: usar nombres de archivo como parámetros de asignación

En este ejemplo se explica paso a paso cómo ejecutar una asignación que toma parámetros de entrada en tiempo de ejecución. El archivo de diseño de asignación que utilizamos en este ejemplo está en `<Documentos>\Altova\MapForce2018\MapForceExamples\FileNamesAsParameters.mfd`.

La asignación utiliza dos componentes de entrada: **InputFileName** y **OutputFileName**. Estos componentes aportan el nombre de archivo de entrada (y el nombre de archivo de salida respectivamente) del archivo XML de origen y de destino. Por este motivo, están conectados con el nodo `Archivo: <dinámico>`.



FileNamesAsParameters.mfd (MapForce Basic Edition)

Ambos componentes (**InputFileName** y **OutputFileName**) son componentes de entrada simples, así que podrá suministrarlos como parámetros de entrada a la hora de ejecutar la asignación. A continuación explicamos cómo hacerlo en estos lenguajes de transformación:

- XSLT 2.0, con ayuda de RaptorXML Server
- Built-in (archivo de ejecución de MapForce Server), con ayuda de MapForce Server
- Java

XSLT 2.0

Si genera código en XSLT 1.0 o XSLT 2.0, los parámetros de entrada se escriben en el archivo por lotes `DoTransform.bat` que se ejecutará con RaptorXML Server (*véase Automatización con RaptorXML Server*). Para usar otro archivo de entrada (o de salida), puede pasar los parámetros necesarios en la línea de comandos cuando llame al archivo `DoTransform.bat` o editar este último para incluir los parámetros necesarios.

Siga estos pasos para suministrar un parámetro de entrada personalizado al archivo `DoTransform.bat`:

1. Genere código XSLT 2.0 (**Archivo | Generar código en | XSLT 2.0**) a partir del diseño de asignación `FileNamesAsParameters.mfd`.

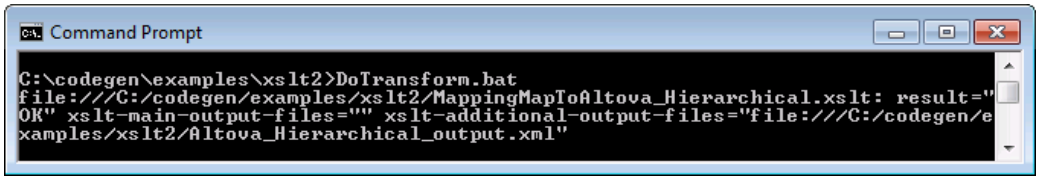
2. Copie el archivo `Altova_Hierarchical.xml` del directorio `<Documentos>\Altova \MapForce2018\MapForceExamples\` al directorio donde generó el código XSLT 2.0 (en este ejemplo usamos el directorio de salida `c:\codegen\examples\xslt2\`). Este archivo hará las veces de parámetro personalizado.

3. Edite el archivo por lotes `DoTransform.bat` para incluir el parámetro de entrada personalizado antes o después de `%*` (ver texto resaltado más abajo). Observe que el

valor de parámetro va entre comillas simples. Los parámetros de entrada disponibles se enumeran en la sección **rem** (Remark).

```
@echo off

RaptorXML xslt --xslt-version=2 --
input="MappingMapToAltova_Hierarchical.xslt" --
param=InputFileName:'Altova_Hierarchical.xml' %*
"MappingMapToAltova_Hierarchical.xslt"
rem --param=InputFileName:
rem --param=OutputFileName:
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```

Cuando ejecute el archivo por lotes `DoTransform.bat`, RaptorXML Server finaliza la transformación usando `Altova_Hierarchical.xml` como parámetro de entrada.



Archivo de ejecución de MapForce Server

Siga estos pasos para suministrar un parámetro de entrada personalizado a un archivo de ejecución de MapForce Server:

1. Compile el diseño de asignación `FileNamesAsParameters.mfd` en un archivo de ejecución de MapForce Server (*véase* *Compilar asignaciones en archivos de ejecución de MapForce Server*). Cuando llegue el momento, guarde el archivo de ejecución `.mfx` en un directorio del equipo (en este ejemplo usamos `c:\codegen\examples\mfx\`).

2. Copie el archivo `Altova_Hierarchical.xml` del directorio `<Documentos>\Altova \MapForce2018\MapForceExamples\` al directorio donde guardó el archivo de ejecución `.mfx`. Este archivo hará las veces de parámetro personalizado para el archivo de ejecución de la asignación.

3. Ejecute MapForce Server con este comando:

```
MapForceServer.exe run "C:\codegen\examples\mfx
\FileNamesAsParameters.mfx" -p=InputFileName:"C:\codegen\examples\mfx
\Altova_Hierarchical.xml" -p=OutputFileName:"C:\codegen\examples\mfx
\OutputFile.xml"
```

Observe que, en este comando de MapForce Server, `-p=InputFileName` y `-p=OutputFileName` son los parámetros de entrada para la asignación. Como valor de `OutputFileName` puede usar cualquier nombre de archivo. Sin embargo, el nombre de archivo dado en `InputFileName` debe existir como archivo físico para que la ejecución pueda realizarse correctamente.

Nota: si recibe el mensaje "MapForceServer.exe no se reconoce como un comando interno o externo, programa o archivo por lotes ejecutable", cambie el directorio actual por el directorio donde está instalado el ejecutable de MapForce Server. Si no quiere tener que cambiar de directorio cada vez que ejecute una asignación, basta con añadir la ruta de acceso del directorio donde está instalado MapForce Server a la variable de entorno PATH del sistema operativo (p. ej. `C:\Archivos de programa (x86)\Altova\MapForceServer2018\bin`) .

Java

Siga estos pasos para suministrar un parámetro de entrada personalizado a una aplicación Java .jar:

1. Genere código Java (**Archivo | Generar código en | Java**) a partir del diseño de asignación `FileNamesAsParameters.mfd`.

2. Compile el código Java en un archivo ejecutable JAR (encontrará instrucciones en este apartado: Ejemplo: generar una aplicación Java con Eclipse y Ant).

3. Copie el archivo `Altova_Hierarchical.xml` del directorio `<Documentos>\Altova\MapForce2018\MapForceExamples\` al directorio donde está el archivo `.jar`. Este archivo hará las veces de parámetro personalizado para la aplicación de asignación Java.

4. En la línea de comandos introduzca `java -jar Mapping.jar /InputFileName "InputFile.xml"`

   Si usa comodines a la hora de pasar parámetros a archivos .jar, debe poner los parámetros comodín entre comillas. Por ejemplo:

   ```
   java -jar Mapping.jar /InputFileName "altova-*.xml"
   ```


© 2018 Altova GmbH

5.6 Obtener valores de cadena de una asignación

Cuando necesite devolver un valor de cadena de una asignación, puede usar un componente de salida simple. En el área de asignación los componentes de salida simples desempeñan el papel de componente de destino que tiene un tipo de datos de cadena en lugar de una estructura de elementos y secuencias. Por tanto, puede crear un componente de salida simple en lugar de (o además de) un componente de destino basado en un archivo. Por ejemplo, puede usar un componente de salida simple para probar y obtener una vista previa del resultado de una función (*véase* *Ejemplo: vista previa de resultados de una función*). Esta técnica también puede ser muy práctica cuando se trabaja con asignaciones que usan serialización de cadenas (*véase* *Analizar y serializar cadenas de texto*). No obstante, debe destacarse que el objetivo principal de los componentes de salida simples es recuperar una cadena cuando se llama a la API de MapForce Server sin necesidad de escribir ningún archivo.

Los componentes de entrada simples no se deben confundir con los parámetros de salida de las funciones definidas por el usuario (*véase* *Funciones definidas por el usuario*). En esta tabla puede ver en qué se parecen y en qué se diferencian:

Componentes de salida	**Parámetros de salida de funciones definidas por el usuario**
Se añaden desde el comando de menú **Función | Insertar componente de salida**.	Se añaden desde el comando de menú **Función | Insertar componente de salida**.
Tienen el tipo de datos "string".	Puede tener tipos de datos simples y complejos.
Afectan a toda la asignación.	Afectan solo al contexto de la función donde se definen.

Si lo necesita, puede añadir varios componentes de salida simples a la asignación. También puede usar componentes de salida simples junto con componentes de destino de base de datos y basados en archivos. Si la asignación contiene varios componentes de destino, puede consultar la vista previa de los datos que devuelve cada componente haciendo clic en el botón **Vista previa** () de la barra de título del componente correspondiente y haciendo clic en el panel *Resultados* de la ventana de asignación.

Los componentes de salida simples se pueden usar en estos lenguajes de transformación de MapForce:

Lenguaje	**Funcionamiento**
BUILT-IN (cuando se genera la vista previa de la transformación de la asignación)	Puede consultar la vista previa de componentes de salida igual que los resultados de una asignación basada en archivos (haciendo clic en el panel *Resultados* de la ventana de asignación.
BUILT-IN (cuando se ejecuta el archivo de ejecución de MapForce	Cuando ejecute un archivo de ejecución de MapForce Server compilado (*véase* *Compilar una asignación de MapForce*), el resultado de la asignación se devuelve en la secuencia de datos

Server)	de salida estándar (stdout). De este modo podrá ver los resultados o redirigirlos a un archivo. Por ejemplo, si imaginamos que el nombre del archivo de ejecución de MapForce Server es **MiAsignación.mfx**, puede usar esta sintaxis para redirigir el resultado de la asignación al archivo **resultado.txt** y los errores al archivo **log.txt**: `MapForceServer.exe run MiAsignación.mfx >resultado.txt 2>log.txt`
XSLT 1.0, XSLT 2.0	En los archivos XSLT que se generan, el componente de salida simple definido en la asignación se convierte en el resultado de la transformación XSLT. Si usa un servidor RaptorXML Server, puede ordenar al servidor que escriba el resultado de la asignación en el archivo que se pasa como valor del parámetro `--output`. Para escribir el resultado en un archivo, añada o edite el parámetro `--output` en el archivo **DoTransform.bat**. Por ejemplo, el archivo **DoTransform.bat** que aparece en el comando siguiente se editó para que se escriba el resultado de la asignación en el archivo **Resultado.txt** (véase el texto resaltado). `RaptorXML xslt --xslt-version=2 --input="MappingMapToResult1.xslt" --output="Resultado.txt" %* "MappingMapToResult1.xslt"` Si no se definió el parámetro `--output`, el resultado de la asignación se escribirá en la secuencia de datos de salida estándar (stdout) cuando se ejecute la asignación.
C++, C#, Java	En el código C++, C# y Java que se genera, el resultado de la asignación se escribe en el resultado estándar de la aplicación generada. Si la asignación contiene varios componentes de destino, la aplicación generada concatena el resultado estándar de cada componente de salida y lo devuelve como resultado estándar unificado.

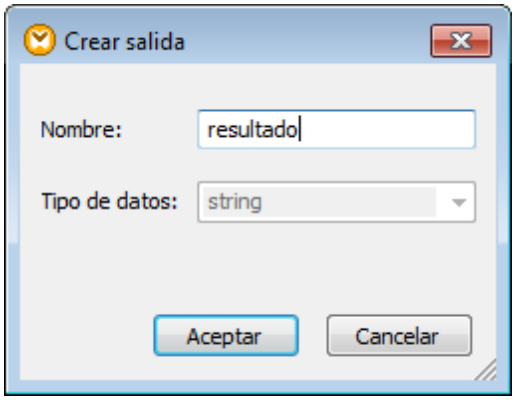
Cuando cree una asignación invertida (con el comando de menú **Herramientas | Crear asignación inversa**), los componentes de salida simples se convertirán en componentes de entrada simples.

© 2018 Altova GmbH

5.6.1 Agregar componentes de salida simples

Para agregar un componente de salida en el área de asignación:

1. Asegúrese de que en la ventana de asignación está activa la asignación principal (y no una función definida por el usuario).
2. En el menú **Función** haga clic en el comando **Insertar componente de salida**.
3. Introduzca el nombre del componente.
4. Haga clic en **Aceptar**.



Cuadro de diálogo "Crear salida"

El nombre del componente se puede cambiar más adelante de varias maneras:

- Seleccionando el componente y haciendo clic en **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haga clic con el botón derecho en el título del componente y después elija **Propiedades**.

5.6.2 Ejemplo: vista previa de resultados de una función

Este ejemplo sirve para explicar cómo consultar la vista previa de los resultados que devuelven funciones de MapForce con la ayuda de componente de salida simples. Antes de consultar este ejemplo es conveniente conocer el funcionamiento básico de las funciones en general y de las funciones de MapForce en particular. Puede encontrar más información en la sección Funciones.

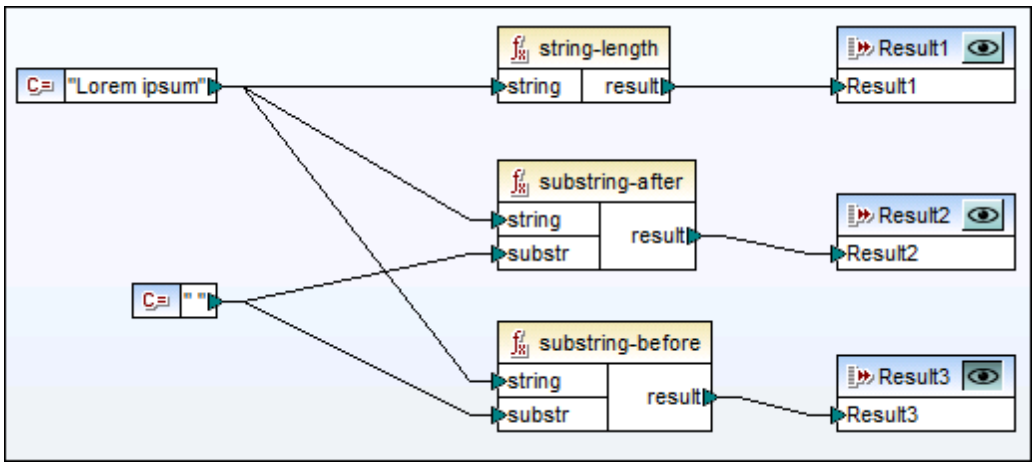
Nuestro objetivo en este ejemplo es añadir varias funciones al área de asignación y aprender a generar la vista previa de los resultados con ayuda de componentes de salida simples. En concreto, este ejemplo utiliza varias funciones simples de la biblioteca principal **core** de MapForce. Aquí resumimos el uso de estas funciones:

`string-length`	Devuelve el número de caracteres de la cadena que se dio como argumento. Por ejemplo, si pasamos a esta función el valor "Lorem ipsum", se obtiene el resultado "11" porque es el número de caracteres que toma el texto "Lorem ipsum".

`substring-after`	Devuelve la parte de la cadena que aparece después del separador que se dio como argumento. Por ejemplo, si pasamos a esta función el valor "Lorem ipsum" y el carácter de espaciado (" "), el resultado es "ipsum".
`substring-before`	Devuelve la parte de la cadena que aparece antes del separador que se dio como argumento. Por ejemplo, si pasamos a esta función el valor "Lorem ipsum" y el carácter de espaciado (" "), el resultado es "Lorem".

Siga estos pasos para probar todas estas funciones con un valor de texto (p. ej. "Lorem ipsum"):

1. Añada una constante con el valor "Lorem ipsum" al área de asignación (con el comando de menú **Insertar | Constante**). La constante será el parámetro de entrada para cada una de las funciones que vamos a probar.
2. Añada las funciones `string-length`, `substring-after` y `substring-before` al área de asignación (arrastrándolas desde la biblioteca **core**, sección **string functions**, hasta el área de asignación).
3. Añada una constante con un carácter de espaciado como valor (" "). Este será el parámetro separador que requieren las funciones `substring-after` y `substring-before`.
4. Añada tres componentes de salida simples (con el comando de menú **Función | Insertar componente de salida**). En nuestro ejemplo estos componentes se llaman **Resultado1**, **Resultado2** y **Resultado3** respectivamente.
5. Conecte los componentes tal y como aparece en la siguiente imagen.



Probando resultados de funciones con componentes de salida simples

Tal y como puede ver en la imagen anterior, la cadena "Lorem ipsum" hace de parámetro de entrada para las funciones `string-length`, `substring-after` y `substring-before`. Además, las funciones `substring-after` y `substring-before` toman como segundo parámetro de entrada un valor de espaciado. Los componentes **Resultado1**, **Resultado2** y **Resultado3** pueden utilizarse para generar la vista previa del resultado de cada función.

Para obtener la vista previa de resultados de una función:

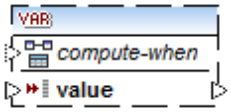
- Haga clic en el botón **Vista previa** ( ) en la barra de título del componente correspondiente y después haga clic en el panel *Resultados* de la ventana de asignación.

© 2018 Altova GmbH

5.7 Usar variables

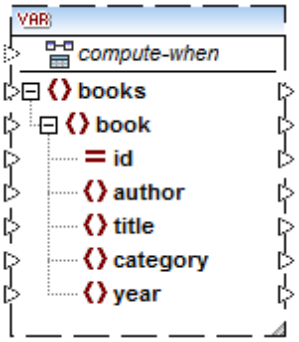
Las variables son un tipo especial de componente utilizado para almacenar un resultado de asignación intermedio y poder seguir procesándolo. Pueden ser necesarias cuando hace falta recordar temporalmente ciertos datos de la asignación y procesarlos de alguna manera (p. ej. filtrándolos o aplicándoles funciones) antes de copiarlos en el componente de destino.

Las variables pueden ser de tipo simple (p. ej. cadena, entero, valor booleano, etc) o de tipo complejo (estructuras jerárquicas).



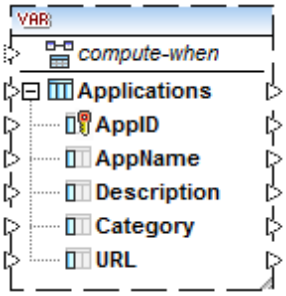
Variable simple

Puede crear una variable de tipo complejo suministrando un esquema XML que exprese la estructura de la variable. Si el esquema define elementos globalmente, podrá elegir cuál de ellos debe ser el nodo raíz de la estructura de la variable. Recuerde que una variable no tiene ningún archivo XML de instancia asociado sino que sus datos se calculan cuando se ejecuta la asignación.



Variable compleja creada a partir de un esquema XML

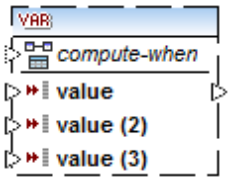
También se pueden crear variables de tipo complejo a partir de bases de datos, archivos EDI y plantillas FlexText (*véase* MapForce FlexText). En el caso de las bases de datos podrá elegir una tabla concreta de la base de datos como elemento raíz de la estructura de la variable.



Variables complejas creadas a partir de una tabla de BD

En las imágenes anteriores observará que cada variable tiene un elemento llamado `compute-when`. Conectar este elemento es opcional y sirve para controlar cómo se debe calcular el valor de la variable en la asignación (*véase* Cambiar el contexto y el ámbito de las variables).

Si es necesario, los elementos de la estructura de una variable se pueden duplicar para que acepten datos de más de una conexión de origen, igual que se hace con componentes estándar (*véase* Duplicar entradas). Sin embargo, esto no se puede hacer en las variables creadas a partir de tablas de BD.



Variable simple con entradas duplicadas

Una de las cosas más importantes de las variables es que son secuencias y que pueden utilizarse para crear secuencias. En este sentido el término secuencia significa "una lista de cero o más elementos" (*véase* Reglas y estrategias de asignación de datos). Esto permite a las variables poder procesar varios elementos durante todo el ciclo de vida de la asignación. No obstante, también puede asignar un valor a una variable una sola vez y conservarlo para el resto de la asignación (*véase* Cambiar el contexto y ámbito de las variables).

Hasta cierto punto las variables pueden compararse con los componentes intermedios de una asignación en cadena (*véase* Asignaciones en cadena). Sin embargo, son mucho más flexibles y prácticas cuando no se necesita generar archivos intermedios en cada fase de la asignación. A continuación se explican las diferencias que existen entre las variables y las asignaciones en cadena.

Asignaciones en cadena	Variables
Las asignaciones en cadena se desarrollan en dos fases totalmente independientes. Por ejemplo, imaginemos que una asignación tiene tres componentes llamados A, B y C. La ejecución de la asignación se desarrolla en	Mientras se ejecuta la asignación, las variables se evalúan en función de su contexto y su ámbito. Su contexto y ámbito puede modificarse (*véase* Cambiar el contexto y ámbito de las variables).

 © 2018 Altova GmbH

dos fases: la ejecución de la asignación desde A hasta B y la ejecución de la asignación desde B hasta C.	
Cuando se ejecuta la asignación, los resultados intermedios se almacenan de forma externa en archivos.	Cuando se ejecuta la asignación, los resultados intermedios se almacenan de forma internan. No se producen archivos externos con los resultados de la variable.
Con el botón  se puede consultar la vista previa del resultado intermedio.	No se puede consultar la vista previa del resultado de una variable porque éste se calcula en tiempo de ejecución. Sin embargo, se puede usar el depurador para ver los datos generados por la variable (*véase* *Depurar asignaciones*).

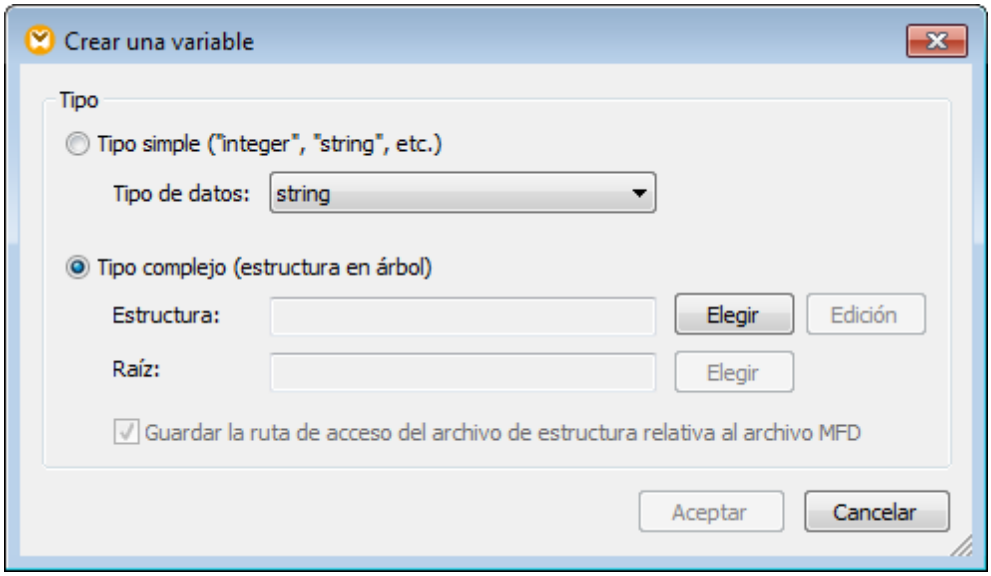
Nota: no se admite el uso de variables si el lenguaje de transformación elegido es XSLT 1.0.

5.7.1 Agregar variables

Hay varias maneras de agregar variables a una asignación de datos.

Con un comando de menú o de la barra de herramientas

1. En el menú **Insertar** haga clic en el comando **Variable** (o haga clic en el botón **Variable**  de la barra de herramientas).

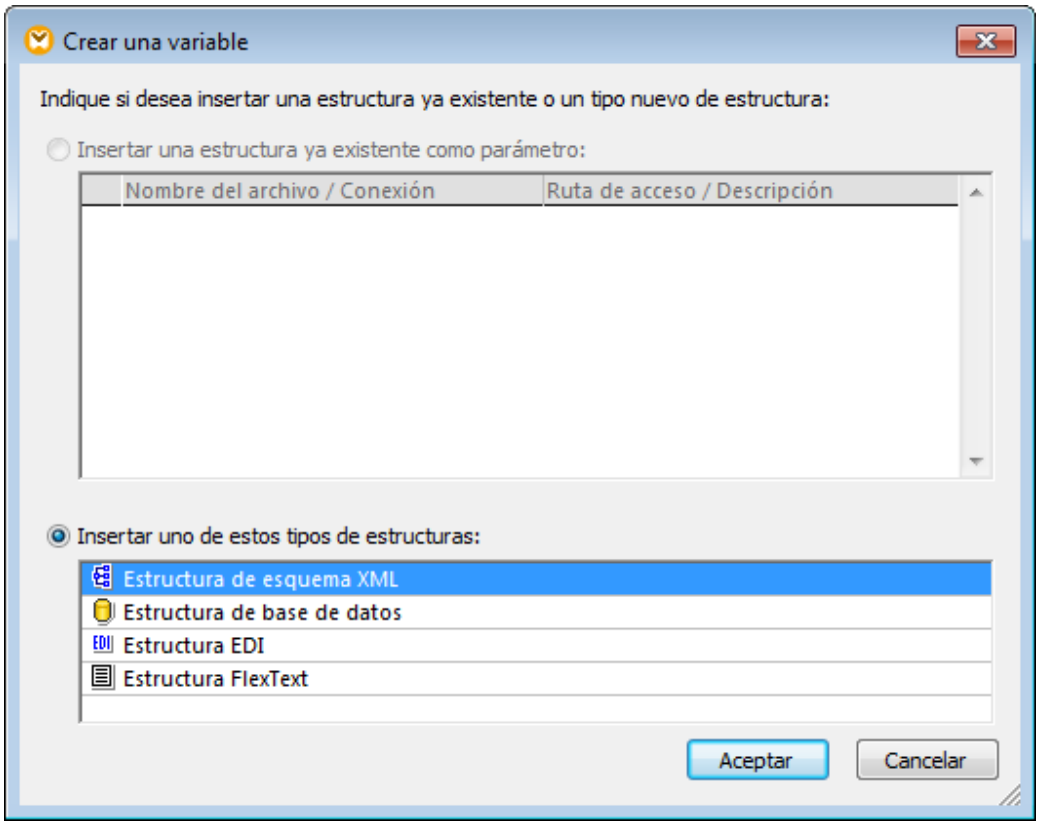


2. Seleccione el tipo de variable que desea insertar (de tipo simple o complejo).

   Si selecciona *Tipo complejo* debe seguir algunos pasos más:

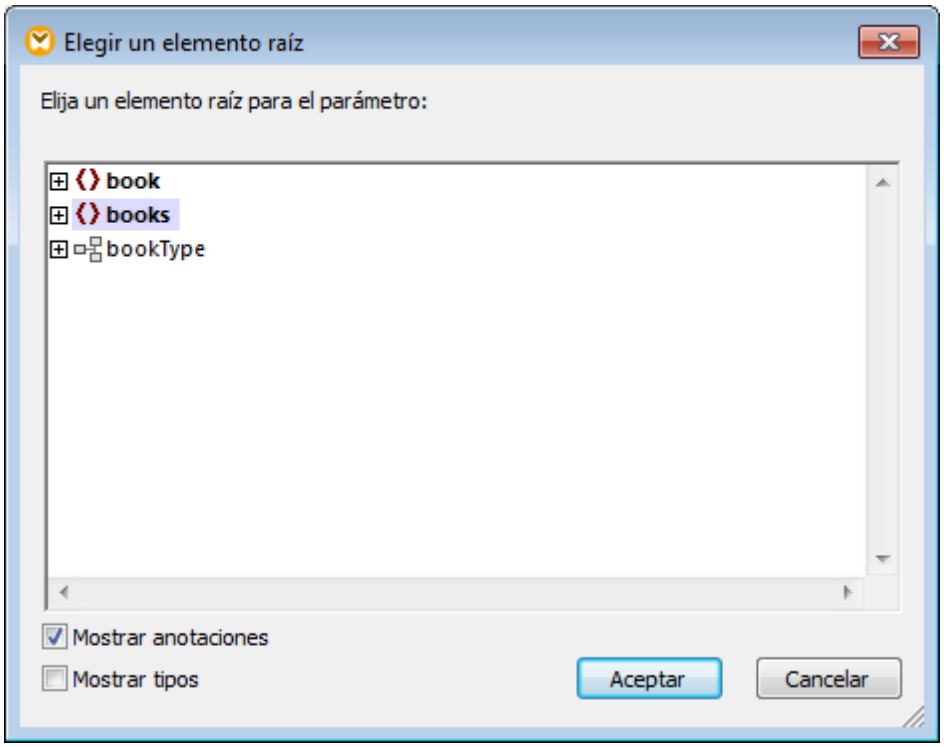
3. Haga clic en **Elegir** para seleccionar la fuente que aportará la estructura de la variable (p.

ej. un esquema XML, una base de datos, un archivo EDI o una plantilla FlexText).



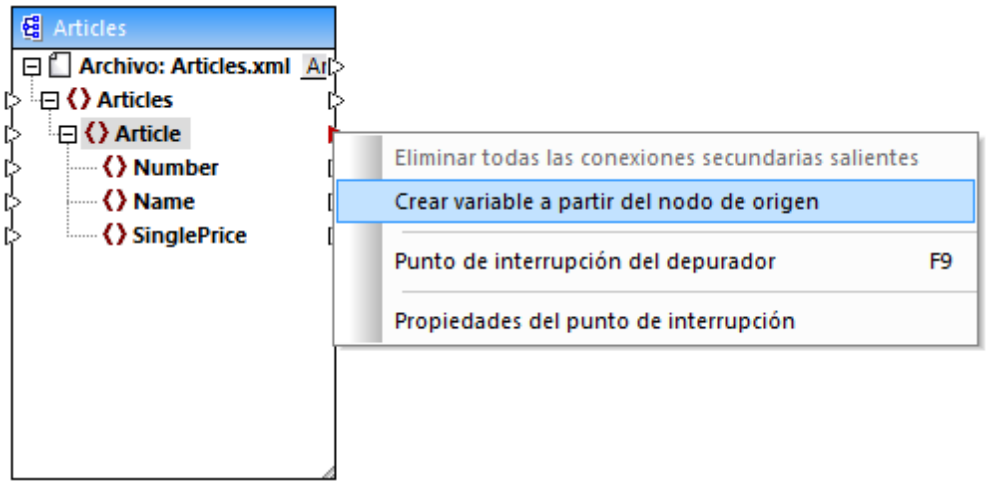
4. Cuando se solicite, especifique un elemento raíz de la estructura. En el caso de los esquemas XML, el elemento raíz puede ser cualquier elemento que esté definido globalmente. En el caso de las bases de datos, el elemento raíz puede ser cualquier tabla.

© 2018 Altova GmbH

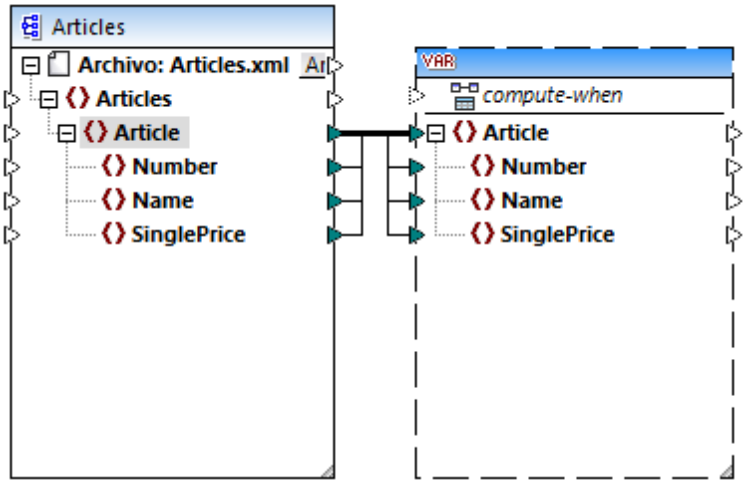


Con un menú contextual

- Haga clic con el botón derecho en el conector de salida de un componente (p. ej. "Article") y seleccione **Crear variable a partir del nodo de origen**.



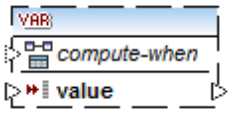
Esto crea una variable compleja que usa el mismo esquema de origen y conecta automáticamente todos los elementos con una conexión de copia total.



- Haga clic con el botón derecho en el conector de entrada de un componente de destino y seleccione **Crear variable para nodo de destino**. Esto crea una variable compleja que usa el mismo esquema que el componente de destino y conecta automáticamente todos los elementos con una conexión de copia total.
- Haga clic con el botón derecho en el conector de salida de un componente de filtro (`on-true`/`on-false`) y seleccione **Crear variable a partir del nodo de origen**. Esto crea un componente complejo que usa el mismo esquema de origen y usa automáticamente como elemento raíz del componente intermedio el elemento vinculado a la entrada del filtro.

5.7.2 Cambiar el contexto y ámbito de las variables

Todas las variables tienen un elemento de entrada `compute-when` que sirve para controlar el ámbito de la variable, es decir, en qué momento y con qué frecuencia se calcula el valor de la variable cuando se ejecuta la asignación. En muchos casos este elemento de entrada puede dejarse sin conectar, pero en ocasiones puede ser imprescindible a la hora de reemplazar el contexto predeterminado o para optimizar el rendimiento de la asignación de datos.



Elemento "compute-when"

El término ***subestructura*** utilizado en esta documentación hace referencia al conjunto de elementos/nodos de un componente de destino y a todos sus descendientes (p. ej. un elemento `<Persona>` con sus elementos secundarios `<Nombre>` y `<Apellido>`.

El término ***valor de variable*** sirve para denominar los datos disponibles en el lado saliente del componente de variable.

- En el caso de las variables simples se trata de una secuencia de valores atómicos cuyo

 © 2018 Altova GmbH

tipo de datos está especificado en las propiedades del componente.+
- En el caso de las variables complejas se trata de una secuencia de nodos raíz (del tipo especificado en las propiedades del componente), cada uno de ellos con sus nodos descendientes.

La secuencia de valores atómicos (o nodos) puede contener un elemento e incluso cero. Esto depende de cómo esté conectado el lado entrante de la variable y de con qué elementos principales del componente de origen/destino esté conectado.

Si "Compute-when" no está conectado (configuración predeterminada)

Si el elemento de entrada `compute-when` no está conectado (a ningún nodo de salida de un componente de origen), entonces el valor de la variable se calcula *cuando se use por primera vez en una subestructura de destino* (ya sea por conexión directa entre el componente de variable y un nodo del componente de destino o por conexión indirecta a través de funciones). El mismo valor de variable se utiliza después para todos los nodos secundarios de destino incluidos dentro de la subestructura.

El valor real de la variable dependerá de las conexiones que existan entre los elementos principales del componente de origen y de destino.

Este comportamiento predeterminado es idéntico al de los resultados complejas de funciones definidas por el usuario y de llamadas a función de servicio web.

Si el resultado de la variable se conecta a varios nodos de destino que no guardan relación entre sí, el valor de la variable se calcula por separado para cada uno de ellos. Esto puede dar lugar a resultados distintos porque las distintas conexiones principales influyen en el contexto en el que se evalúa el valor de la variable.

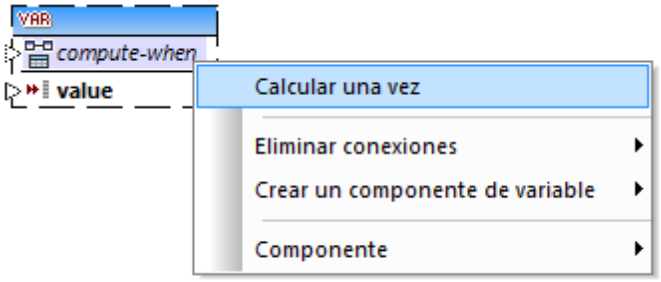
Si "Compute-when" está conectado

Si se conecta un conector de salida de un componente de origen con el elemento `compute-when`, el valor de la variable se calcula *cuando dicho elemento de origen se use por primera vez en una subestructura de destino*.

En realidad la variable hace las veces de elemento secundario del elemento que está conectado a `compute-when`. Esto permite vincular la variable con un elemento de origen concreto. Es decir, en tiempo de ejecución la variable se vuelve a evaluar cada vez que se lea un elemento nuevo de la secuencia del componente de origen. Se trata de la regla general que rige las conexiones en MapForce: *por cada elemento de origen, se crea uno de destino*. En el caso del elemento de entrada `compute-when` esto significa que *por cada elemento de origen, se calcula el valor de variable* (*véase Reglas y estrategias de asignación de datos*).

"Compute-once"

Si fuera necesario, puede solicitar que el valor de variable se calcule una sola vez antes de cada componente de destino, lo cual convierte a la variable en una constante global para el resto de la asignación. Esto se consigue haciendo clic con el botón derecho en el elemento `compute-when` y seleccionando el comando **Calcular una vez** en el menú contextual (*imagen siguiente*).



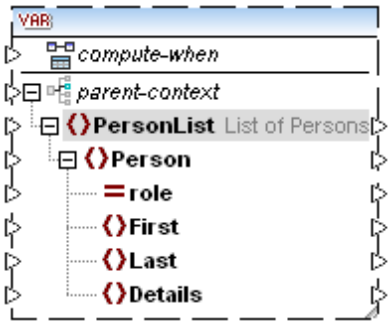
Cuando se cambia el ámbito de una variable por `compute-when=once`, el conector de entrada se elimina del elemento `compute-when` (porque dicha variable solamente se evaluará una vez).

En funciones definidas por el usuario la variable `compute-when=once` se evalúa cada vez que se llama a la función y antes de que el resultado de la función se evalúe.

Contexto primario

En ocasiones puede ser necesario agregar un contexto primario. Por ejemplo, si la asignación usa varios filtros y necesita recorrer un nodo primario adicional (*véase* *Reemplazar el contexto de la asignación*).

Para agregar un contexto primario a una variable basta con hacer clic con el botón derecho en el nodo raíz (p. ej. "PersonList") y seleccionar el comando b en el menú contextual. Como resultado se añade un nodo nuevo llamado `parent-context` a la jerarquía de la variable.



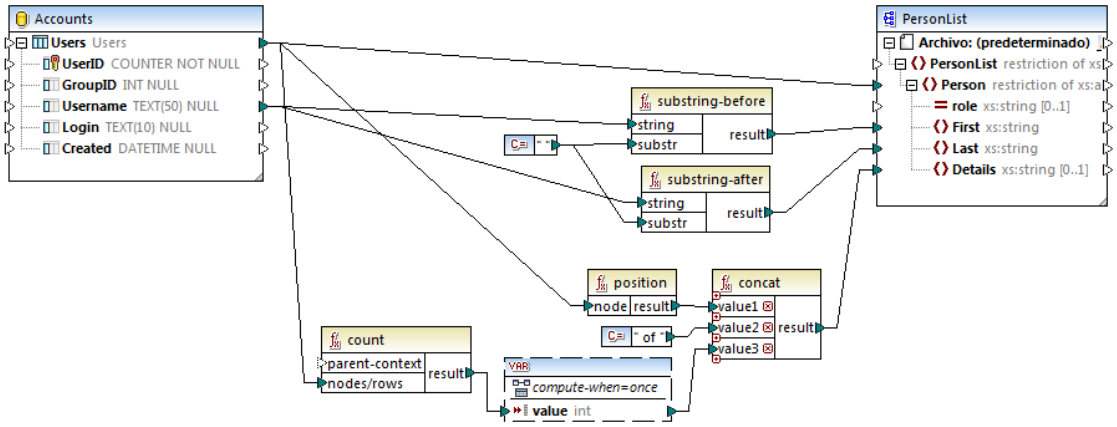
El contexto primario añade un nodo primario virtual a la jerarquía del componente, lo cual permite recorrer otro nodo más en el mismo componente de origen o en otro distinto.

5.7.3 Ejemplo: contar filas de tabla de BD

El diseño de asignación que aparece más abajo corresponde al archivo **DB_UserList.mfd** situado en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**. Esta asignación extrae registros de usuarios de una tabla de base de datos llamada "Users" y los escribe en un archivo XML. La columna de base de datos "Username" contiene el nombre y el apellido de una persona (p. ej. "Vernon Callaby"). Esta asignación tiene varios objetivos:

© 2018 Altova GmbH

1. Crear un elemento `Person` nuevo por cada registro de la tabla "Users" en el archivo XML de destino.
2. Dividir el valor extraído del campo de base de datos "Username" en dos campos diferentes del archivo XML (`First` y `Last`).
3. Encontrar el número secuencial de cada registro comparado con el número total de registros presentes en la base de datos (p. ej. "Registro 1 de 4") y escribir esta información en el elemento `Details`.



DB_UserList.mfd

Como indica el diseño, para poder conseguir el primer objetivo debemos dibujar una conexión entre la tabla de origen "Users" y el elemento `Person` del archivo XML de destino. Esto permite crear un elemento nuevo `Person` por cada registro de la tabla de origen.

El valor del campo "Username" se pasa a las funciones **`substring-before`** y **`substring-after`** , que se encargan de extraer el texto situado antes y después del carácter de espaciado (" ") respectivamente. Así se consigue el segundo objetivo del diseño.

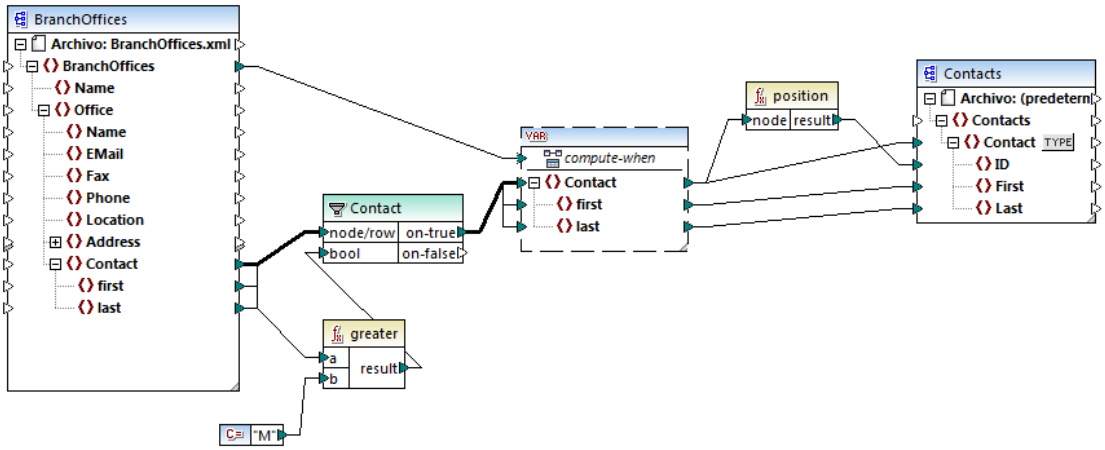
Y, por último, para conseguir el tercer objetivo es fundamental la función **`count`**. El resultado de esta función se pasa a una variable, que se encarga de almacenar este resultado en la asignación y servirlo a la hora de escribir el elemento `Details` de cada personal en el archivo XML de destino. Recuerde que por razones de eficacia los registros de base de datos se deberían contar una sola vez, así que definimos el ámbito de variable `compute-when=once` (*véase* *Cambiar el contexto y ámbito de las variables*).

5.7.4 Ejemplo: filtrar y numerar nodos

El diseño de asignación que aparece más abajo corresponde al archivo **PositionInFilteredSequence.mfd** situado en la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples**.

Esta asignación lee un archivo XML que contiene datos de contacto de varias persona, los filtra y los escribe en un archivo XML de destino. El objetivo del diseño es filtrar datos del archivo XML de origen de las personas cuyo apellido empiece a partir de la letra M. Además, los contactos extraídos deben estar numerados en el XML de destino. Este número hará de identificador único

en el XML de destino.



PositionInFilteredSequence.mfd

Para conseguir este objetivo se añadieron varios componentes al diseño de asignación:

- Un filtro (*véase Filtros y condiciones*)
- Una variable compleja (*véase Agregar variables*)
- Las funciones **`greater`** y **`position`** (*véase Trabajar con funciones*)
- Una constante (para agregar una constante seleccione el comando de menú **Insertar | Constante**).

La variable usa el mismo esquema que el componente de origen. Si hacemos clic con el botón derecho en la variable y seleccionamos Propiedades en el menú contextual, veremos que el nodo raíz que está seleccionado para esta estructura de variable es **BranchOffices/Office/Contact**.

Primero se pasan al filtro los datos del componente de origen. Después el filtro pasa a la variable los registros que cumplan la condición de filtrado. Es decir, el filtro está configurado para obtener los nodos `Contact` cuyo nombre sea igual o mayor que "M". Para ello se usa la función **`greater`** que compara cada elemento `last` con el valor de constante "M".

La variable tiene la entrada `compute-when` conectada con el elemento raíz del componente de origen (`BranchOffices`). En tiempo de ejecución esto hace que la variable se vuelva a evaluar cada vez que se lea un elemento nuevo de la secuencia del componente de origen. Sin embargo, en esta asignación no influye en modo alguno el conectar o no conectar el elemento `compute-when`. El motivo es que la variable está conectada al elemento de origen `Contact` (de forma indirecta a través del filtro) y se calcularía tantas veces como instancias de `Contact` cumplan la condición de filtrado.

La función **`position`** devuelve (por cada iteración de la variable) el número de la secuencia actual. Solo ocho contactos cumplen la condición. Por tanto, si consultamos la vista previa del resultado de la asignación, veremos que se escribieron los identificadores 1 - 8 en el elemento `ID` del componente de destino.

La variable se necesita porque necesitamos numerar todos los registros. Si se hubiera conectado el resultado del filtro al componente de destino directamente, no habría manera de numerar cada instancia de `Contact`. La función de la variable es, por tanto, almacenar temporalmente cada

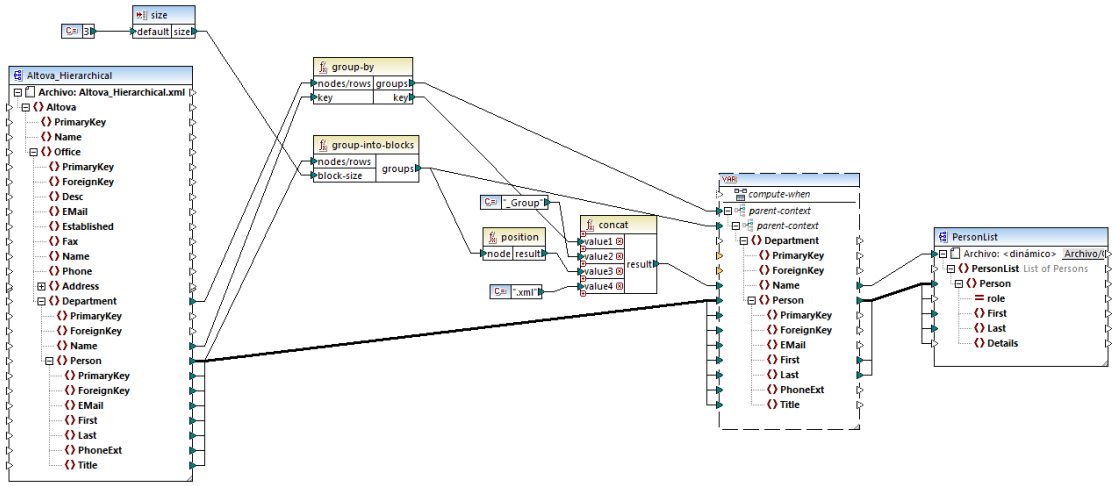
 © 2018 Altova GmbH

instancia de `Contact` para poder numerarlas antes de escribirlas en el destino.

5.7.5 Ejemplo: crear grupos y subgrupos de registros

El diseño de asignación que aparece más abajo corresponde al archivo **DividePersonsByDepartmentIntoGroups.mfd** situado en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**.

Este diseño procesa un archivo XML que contiene registros de empleados de una compañía ficticia. La compañía tiene dos oficinas: "Nanonull, Inc." y "Nanonull Partners, Inc". Cada oficina tiene varios departamentos (p. ej. "IT", "Marketing", etc.) y cada departamento tiene como mínimo un empleado. El objetivo de esta asignación de datos es crear grupos formados por un tres personas como máximo de cada departamento, sin importar la oficina. El tamaño predeterminado de cada grupo es 3, pero esto se puede cambiar. Además cada grupo debe guardarse en un archivo XML distinto, cuyo nombre seguirá el patrón "<Nombre Departamento>_NºGrupo" (p. ej. **Marketing_Group1.xml**, **Marketing_Group2.xml**, etc.).



DividePersonsByDepartmentIntoGroups.mfd

Como puede ver, la asignación cuenta con una variable compleja y varios componentes más (funciones sobre todo). La variable tiene la misma estructura que el elemento `Department` del archivo XML de origen. Si hacemos clic con el botón derecho en la variable para ver sus propiedades, observaremos que utiliza el mismo esquema que el componente de origen y que su elemento raíz es `Department`. Y lo que es más importante, la variable tiene dos elementos `parent-context` anidados, que garantizan que la variable se calcule primero en el contexto de cada departamento y después en el contexto de cada grupo dentro de cada departamento (*véase* *Cambiar el contexto y ámbito de las variables*).

En un principio la asignación recorre todos los departamentos para obtener el nombre de cada uno de ellos (el nombre de los departamentos se necesita para crear el nombre de archivo que corresponde a cada grupo). Esto se consigue conectando la función **`group-by`** al elemento de origen `Department` y pasando el nombre del departamento como clave de agrupación.

Después, dentro del contexto de cada departamento, se lleva a cabo otra agrupación: la asignación llama a la función **`group-into-blocks`** para crear los grupos necesarios de

empleados. El tamaño de cada grupo viene dado por un componente de entrada simple cuyo valor predeterminado es "3". El valor predeterminado viene dado por una constante. En este ejemplo, para cambiar el tamaño de cada grupo basta con modificar el valor de la constante según corresponda. Sin embargo, también se puede modificar el componente de entrada **size** para que, si la asignación se ejecuta con código generado o con MapForce Server, el tamaño de cada grupo pueda especificarse como parámetro (*véase Pasar parámetros a la asignación*).

A continuación, el valor de la variable se pasa al componente XML de destino PersonList. El nombre de archivo de cada grupo creado se calcula con ayuda de la función **concat**, mediante la concatenación de:

1. el nombre de cada departamento,
2. la cadena "_Group",
3. el número que tiene el grupo en la secuencia actual (p. ej. "1" si se trata del primer grupo del departamento) y
4. la cadena ".xml"

El resultado de la concatenación se almacena en el elemento `Name` de la variable y después se pasa como nombre de archivo dinámico al componente de destino. Por tanto, se crea un nombre de archivo nuevo por cada valor recibido. En este ejemplo, la variable calcula 8 grupos en total, así que se crean 8 archivos de salida cuando se ejecuta la asignación. Para más información consulte la sección Procesar varios archivos de entrada o salida simultáneamente.

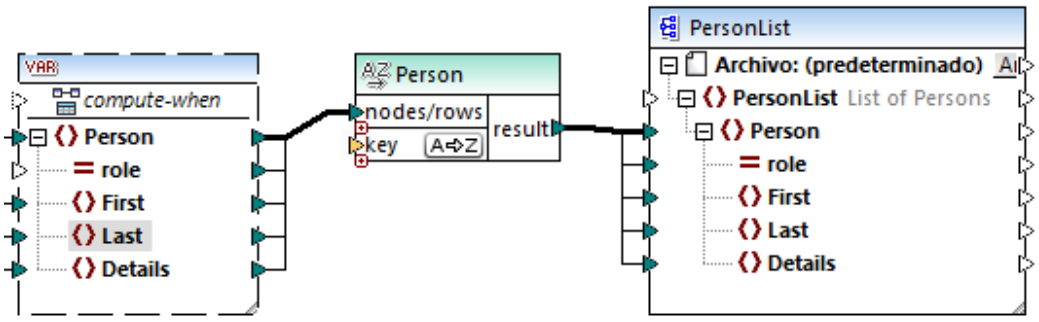
© 2018 Altova GmbH

5.8 Ordenar datos

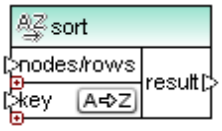
Para ordenar datos de entrada siguiendo un criterio de ordenación concreto basta con usar un componente de ordenación. El componente de ordenación es compatible con estos lenguajes de destino: XSLT2, XQuery y el motor de ejecución integrado. Si elige el lenguaje de transformación integrado, podrá usar el componente de ordenación para ordenar datos de tabla de base de datos. Sin embargo, conseguirá un mejor rendimiento con el componente WHERE/ORDER de SQL (*véase* *Filtrar y ordenar datos de base de datos (WHERE/ORDER de SQL)*).

Para agregar un componente de ordenación a la asignación:

- Haga clic con el botón derecho en una conexión y seleccione **Insertar componente de ordenación: nodos/filas** en el menú contextual. Esto inserta un componente de ordenación y lo conecta automáticamente a los componentes de origen y destino. Por ejemplo, en la imagen siguiente el componente de ordenación se introdujo entre una variable y un componente XML. Lo único que falta por conectar a mano es el criterio de ordenación (el campo por el que desea ordenar los elementos).



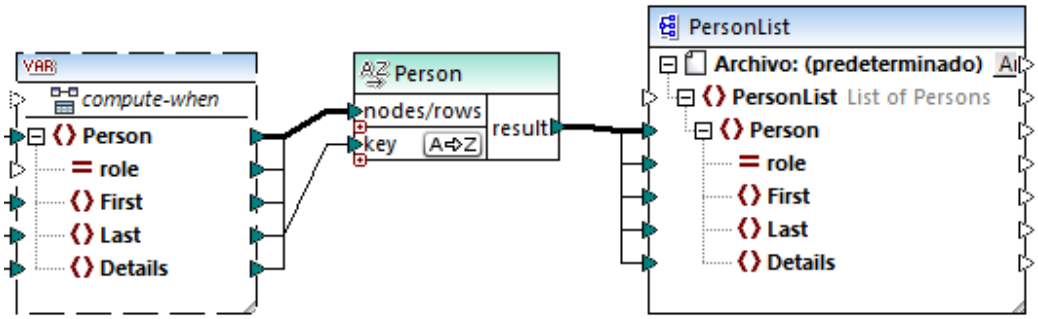
- En el menú **Insertar** seleccione el comando **Componente de ordenación: nodos/filas** (o haga clic en el botón **Insertar componente de ordenación** de la barra de herramientas). Esto inserta el componente de ordenación pero sin conectarlo.



En cuanto conectemos el componente de ordenación con el componente de origen, en la barra de título del componente de ordenación aparecerá el nombre del elemento que está conectado a `nodes/rows`.

Para definir por qué elemento se deben ordenar los nodos/filas:

- Conecte el elemento por el que desea ordenar los nodos/filas al parámetro `key` del componente de ordenación. Por ejemplo, en la imagen siguiente los nodos/filas `Person` se ordenan por el campo `Last`.

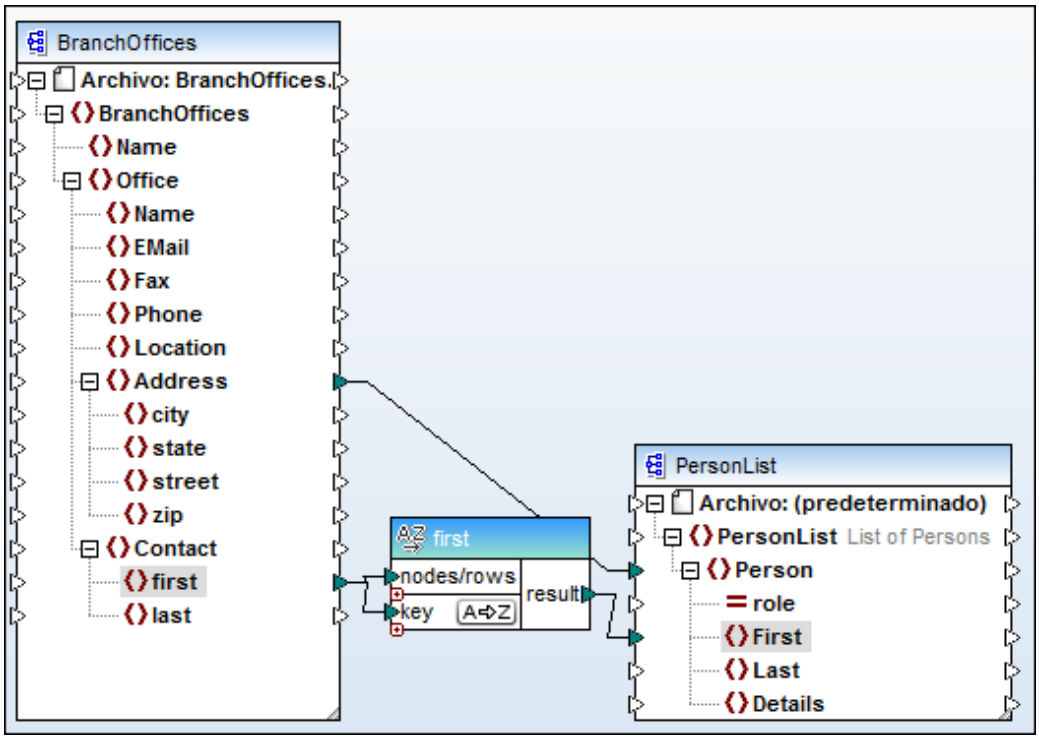


Para cambiar el orden:

- Haga clic en el icono A⇨Z del componente de ordenación. Se convertirá en el icono Z➔A para mostrar que el orden se invirtió.

Para ordenar datos de entrada compuestos por elementos de tipo simple:

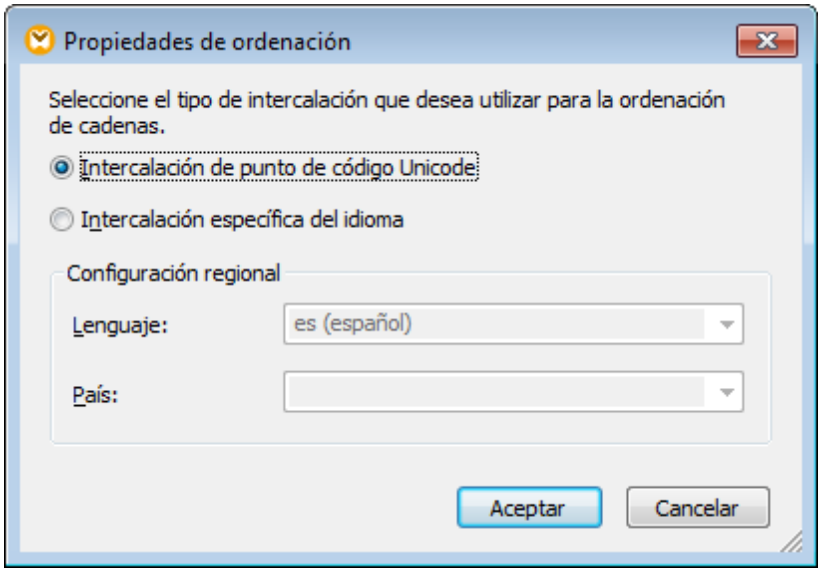
- Conecte el elemento tanto al parámetro `nodes/rows` como al parámetro `key` del componente de ordenación. En la imagen siguiente, por ejemplo, se ordena el elemento de tipo simple `first`.



 © 2018 Altova GmbH

Para ordenar cadenas usando reglas de un idioma concreto:

- Haga doble clic en el título del componente de ordenación para abrir el cuadro de diálogo "Propiedades de ordenación" (*imagen siguiente*).

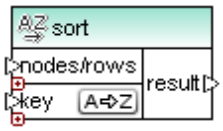


Intercalación de punto de código: esta opción predeterminada compara/ordena las cadenas basándose en valores de punto de código. Los valores de punto de código son enteros que se asignaron a caracteres abstractos del conjunto de caracteres universal adoptado por el consorcio Unicode. Esta opción permite ordenar datos en muchos idiomas y scripts.

Intercalación específica del idioma: esta opción permite definir el idioma y el país que debe utilizarse para la ordenación. Esta opción es compatible con el motor de ejecución integrado. Para XSLT la compatibilidad dependerá del motor que se utilice para ejecutar el código.

5.8.1 Ordenar según varias claves

Cuando se añade un componente de ordenación a la asignación, MapForce crea por defecto un criterio de ordenación llamado `key`.

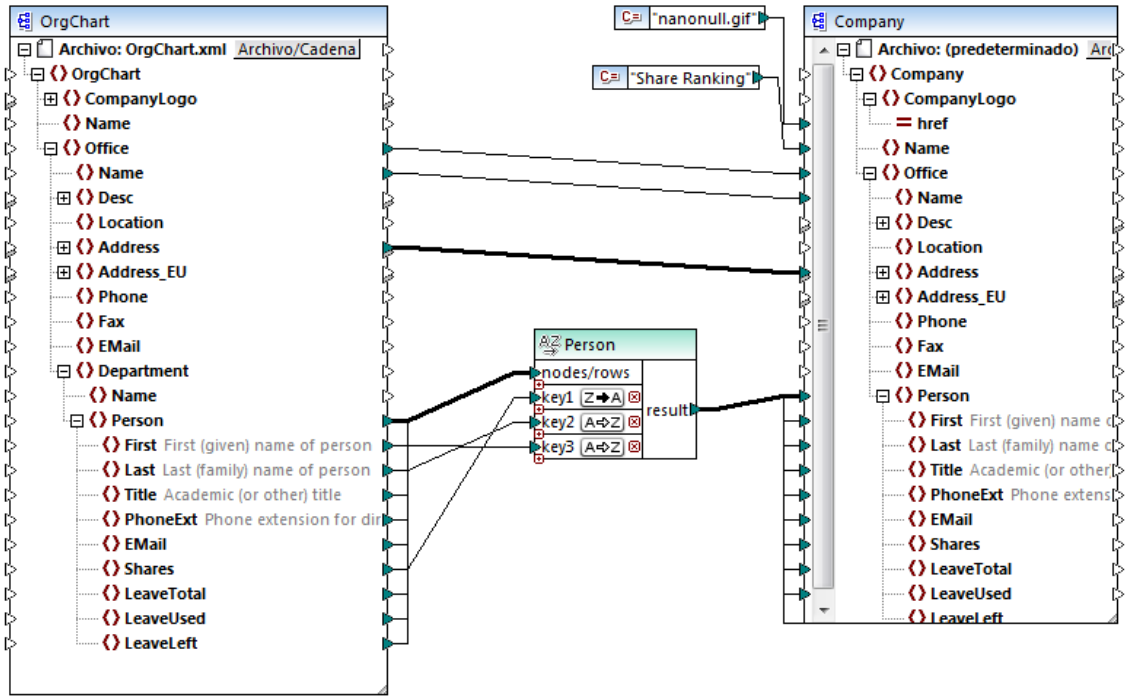


Componente de ordenación

Si desea usar varios criterios de ordenación, deberá realizar varios ajustes en el componente:

- Haga clic en el icono **Agregar criterio** (⊙) para agregar un criterio nuevo (p. ej. `key2`).
- Haga clic en el icono **Eliminar criterio** (⊗) para eliminar un criterio de ordenación.
- Arrastre y coloque una conexión encima del icono ⊙ para agregar un criterio y al mismo tiempo conectarlo.

El diseño de asignación **<Documentos>\Altova\MapForce2018\MapForceExamples \SortByMultipleKeys.mfd** contiene un componente de ordenación que usa varios criterios.



SortByMultipleKeys.mfd

En esta asignación de datos los registros `Person` se ordenan siguiendo tres criterios:

1. `Shares` (número de acciones en propiedad de cada persona)
2. `Last` (apellido)
3. `First` (nombre)

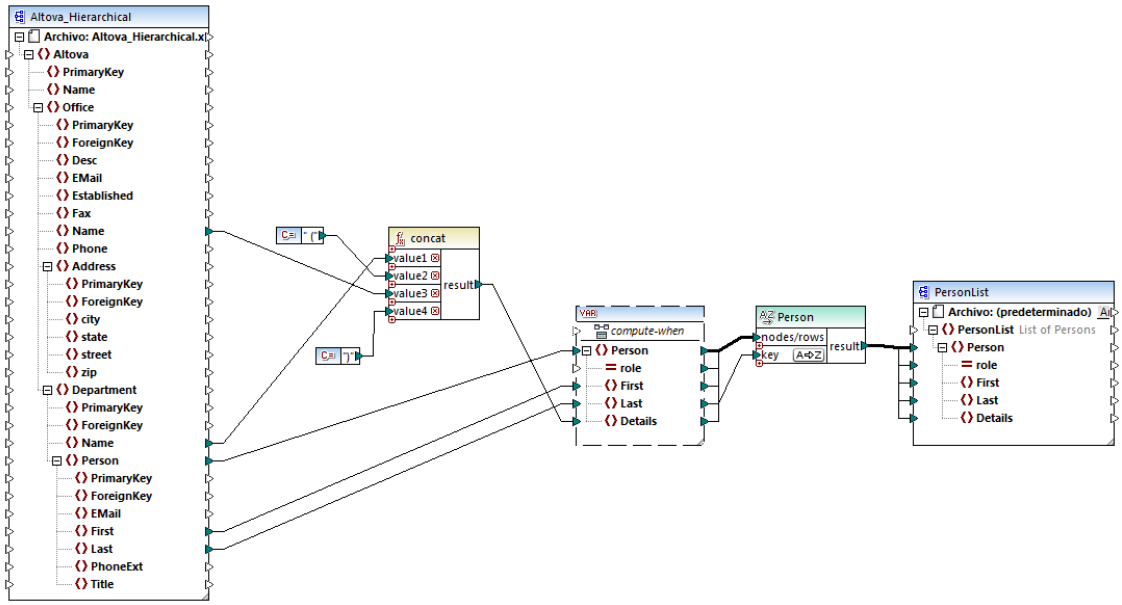
Observe que la posición del criterio de ordenación en el componente determina su prioridad. Por ejemplo, en la imagen puede ver que los registros se orden primero en función del número de acciones. Se trata del criterio de ordenación con mayor prioridad. Si el número de acciones fuera el mismo, las personas se ordenan por apellido. Y, por último, si varios personas tuvieran el mismo apellido, se ordenarán por nombre.

El orden de cada criterio puede ser distinto. Por ejemplo, en la imagen anterior puede ver que el criterio `Shares` indica un orden descendiente (Z-A), mientras que los otros dos criterios siguen un orden ascendente (A-Z).

© 2018 Altova GmbH

5.8.2 Ordenar con variables

En algunos casos puede ser necesario agregar variables intermedias para conseguir ciertos resultados. En el diseño de asignación que aparece a continuación se extraen registros de un archivo XML y éstos se orden con ayuda de variables intermedias. Se trata del diseño de asignación **<Documentos>\Altova\MapForce2018\MapForceExamples \Altova_Hierarchical_Sort.mfd**.



Altova_Hierarchical_Sort.mfd

Esta asignación lee datos de un archivo XML de origen llamado **Altova_Hierarchical.xml** y los escribe en un archivo XML de destino. Como puede ver, el XML de origen contiene información sobre una compañía ficticia. La compañía se divide en oficinas y éstas en departamentos. A su vez, los departamentos están compuestos por empleados.

El componente XML de destino `PersonList` contiene una lista de registros `Person`. El elemento `Details` tiene la función de almacenar información sobre la oficina y el departamento al que pertenece cada empleado.

El objetivo de esta asignación es extraer todos los empleados del XML de origen y ordenarlos por apellido y por orden alfabético. Además, el nombre de la oficina y del departamento al que pertenece cada empleado debe escribirse en el elemento `Details`.

Estos son los componentes necesarios para conseguir el objetivo de la asignación:

1. La función **`concat`**, que devuelve una cadena con el patrón `Office(Department)`. Toma como entrada el nombre de la oficina, el nombre del departamento y dos constantes que aportan el paréntesis de apertura y de cierre respectivamente (*véase Trabajar con funciones*).
2. Una variable intermedia. El papel que desempeña es recopilar todos los datos de cada empleado en el mismo contexto de asignación. La variable consigue que la asignación

busque el departamento y la oficina de cada empleado, en el contexto de cada empleado. Es decir, la variable *recuerda* el nombre de la oficina y del departamento al que pertenece cada empleado. Sin la variable, el contexto sería incorrecto y la asignación produciría resultados no deseados (*véase Reglas y estrategias de asignación de datos*). Tenga en cuenta que la variable reproduce la estructura del archivo XML de destino (usa el mismo esquema XML). Esto permite conectar el resultado del componente de ordenación con el componente de destino por medio de una conexión de copia total. Consulte las secciones Usar variables y Conexiones de copia total para obtener más información.

3. Un componente de ordenación que ordena los datos. Observe que el criterio de ordenación del componente está conectado al elemento `Last` de la variable, lo cual permite ordenar todos los registros por apellido.

© 2018 Altova GmbH

5.9 Filtros y condiciones

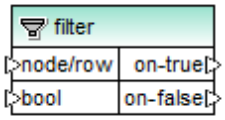
Cuando necesite filtrar datos o conseguir un valor de forma condicional, utilice uno de estos componentes:

- Filtro: nodos/filas ( )
- WHERE/ORDER de SQL ( )
- Condición If-Else ( )

Estos componentes se pueden añadir en la asignación desde el menú Insertar o desde la barra de herramientas Insertar componente. A continuación vamos a explicar las diferencias entre estos componentes, su comportamiento y sus requisitos.

Filtrar nodos o filas

Cuando necesite filtrar datos, incluidos nodos XML o filas CSV, utilice un componente **Filtro: nodos/filas**. Este componente permite recuperar un subconjunto de nodos de un conjunto de datos de mayor tamaño, usando una condición true o false. Esta es su estructura en el área de asignación:



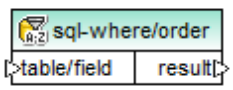
En la estructura de filtro anterior, la condición que está conectada a bool determina si la entrada **node/row** que está conectado se pasa a la salida **on-true** o a la salida **on-false**. Es decir, si la condición se cumple (es true), entonces **node/row** se redirige a la salida **on-true**. Por el contrario, si la condición no se cumple (es false), entonces **node/row** se redirige a la salida **on-false**.

Cuando necesite que la asignación solamente consuma elementos que *cumplan* la condición de filtrado, puede dejar sin conectar la salida **on-false**. Si necesita procesar los elementos que no cumplan la condición de filtrado, entonces conecte la salida **on-false** al nodo de destino donde se deben redirigir dichos elementos. Si quiere añadir una excepción para cuando la condición de filtrado no se cumpla, será obligatorio conectar la salida **on-false** (*véase Añadir excepciones*).

Para ver un ejemplo con instrucciones paso a paso consulte el apartado Ejemplo: filtrar nodos.

Filtrar datos de base de datos

Los componentes **Filtro: nodos/filas** pueden filtrar datos de cualquier otra estructura compatible con MapForce, incluidas las bases de datos. Sin embargo, si desea filtrar datos de una base de datos, lo más recomendado es usar un componente **WHERE/ORDER de SQL**. Este componente está optimizado para trabajar con bases de datos y ofrece un mayor rendimiento que el componente **Filtro: nodos/filas**.



Para más información consulte el apartado Componente WHERE/ORDER de SQL.

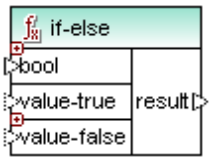
Devolver un valor de forma condicional

Si necesita obtener un solo valor (en lugar de un nodo o de una fila) de forma condicional, lo mejor es usar la **condición If-Else**. Recuerde que estas condiciones no son adecuadas para el filtrado de nodos o filas. A diferencia de los componentes **Filtro: nodos/filas**, las **condiciones If-Else** devuelven un valor de tipo simple (como una cadena o un entero). Por tanto, estas condiciones solamente deben utilizarse cuando sea necesario procesar un valor simple de forma condicional. Por ejemplo, imagine que tiene una lista de las temperaturas medias de cada mes en este formato:

```
<Temperatures>
   <data temp="19.2" month="2010-06" />
   <data temp="22.3" month="2010-07" />
   <data temp="19.5" month="2010-08" />
   <data temp="14.2" month="2010-09" />
   <data temp="7.8" month="2010-10" />
   <data temp="6.9" month="2010-11" />
   <data temp="-1.0" month="2010-12" />
</Temperatures>
```

Una **condición If-Else** permite devolver por cada elemento de la lista el valor "alto" si la temperatura supera los 20 grados centígrados y el valor "bajo" si la temperatura es inferior a los 5 grados centígrados.

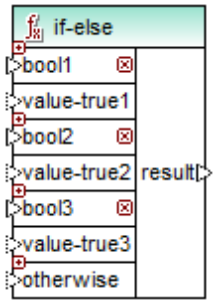
En la asignación la estructura de la **condición If-Else** sería:



Si la condición que está conectada a **bool** es `true`, entonces el valor que está conectado a **value-true** se devuelve como **result**. Si la condición es `false`, el valor que está conectado a **value-false** se devuelve como **result**. El tipo de datos de **result** se conoce previamente y depende del tipo de datos del valor que está conectado a **value-true** o **value-false**. Lo importante es que siempre sea de tipo simple (cadena, entero, etc.). Las **condiciones If-Else** no admiten el uso de valores de entrada de tipo complejo (como nodos o filas).

Además, las **condiciones If-Else** son extensibles. Esto significa que puede añadir varias condiciones al componente con solo hacer clic en el botón **Agregar** ⊡. Para eliminar una condición añadida previamente, haga clic en el botón **Eliminar** ⊠. Esta característica permite comprobar si se cumplen varias condiciones y devolver un valor distinto para cada condición.

© 2018 Altova GmbH



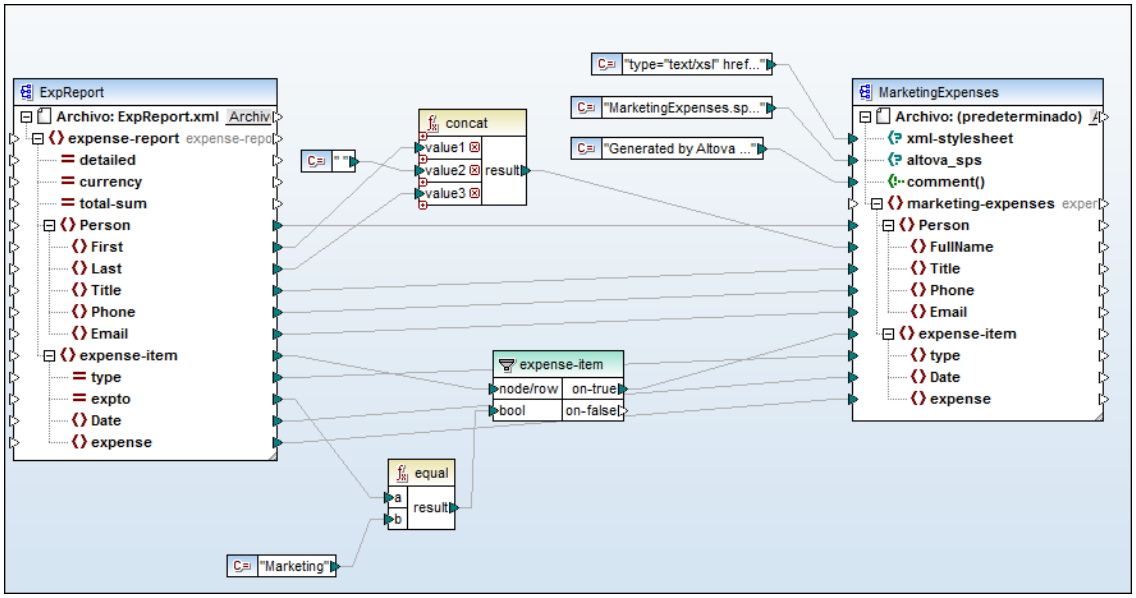
Las **condiciones If-Else** extendidas, se evalúan de arriba a abajo (se comprueba en primer lugar las condiciones situadas más arriba). Si desea devolver un valor aunque no se cumpla ninguna condición (es decir, aunque ninguna condición sea true), conecte el valor a **`otherwise`**.

Para ver un ejemplo con instrucciones paso a paso consulte el apartado Ejemplo: devolver un valor de forma condicional.

5.9.1 Ejemplo: filtrar nodos

Este ejemplo explica cómo filtrar nodos basándose en una condición true/false. Para conseguirlo se utiliza un componente **Filtro: nodos/filas** ( ). La técnica que se utiliza en este ejemplo funciona igual para componentes XML y otros tipos como CSV o texto. A la hora de trabajar con bases de datos, sin embargo, se recomienda el uso de componentes WHERE/ORDER de SQL, que permiten un mayor rendimiento (véase Componente WHERE / ORDER de SQL).

La asignación de este ejemplo es **<Documentos>\Altova\MapForce2018\MapForceExamples\MarketingExpenses.mfd**.



Como puede verse en la imagen, la asignación lee datos de un XML de origen que contiene un informe de gastos ("ExpReport") y escribe datos en un XML de destino ("MarketingExpenses").

Entre el componente de origen y de destino existen varios componentes más. El más relevante en este ejemplo es el componente de filtrado **expense-item** ( ).

El objetivo de la asignación es filtrar los gastos que pertenecen al departamento de Marketing. Para conseguirlo se añadió un componente de filtrado a la asignación. (Para añadir un filtro haga clic en el menú Insertar y elija el comando **Filtro: nodos/filas**.)

Para identificar qué gastos pertenecen al departamento de Marketing, la asignación busca el valor del atributo "expto" del XML de origen. Este atributo tiene el valor "Marketing" si el gasto pertenece a dicho departamento. Por ejemplo, en el siguiente fragmento de código, puede verse que el primer gasto y el tercero pertenecen a Marketing, el segundo a Development (desarrollo) y el cuarto a Sales (ventas):

```
...
   <expense-item type="Meal" expto="Marketing">
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item type="Lodging" expto="Development">
      <Date>2003-01-02</Date>
      <expense>122.12</expense>
   </expense-item>
   <expense-item type="Lodging" expto="Marketing">
      <Date>2003-01-02</Date>
      <expense>299.45</expense>
   </expense-item>
   <expense-item type="Entertainment" expto="Sales">
      <Date>2003-01-02</Date>
      <expense>13.22</expense>
   </expense-item>
...
```

XML de entrada antes de ejecutarse la asignación

En el área de asignación, la entrada **node/row** del filtro está conectado con el nodo **expense-item** del componente de origen. Esto permite al filtro obtener la lista de nodos que debe procesar.

Para agregar la condición en la que se debe basar el filtrado, se añadió la función **equal** de la biblioteca **core** de MapForce (*véase Trabajar con funciones*). La función **equal** compara el valor del atributo "type" con una constante cuyo valor es "Marketing". (Para agregar una constante haga clic en el menú **Insertar** y elija el comando **Constante**.)

Como nuestro objetivo es filtrar los elementos que cumplen esta condición, conectamos solamente la salida **on-true** del filtro con el componente de destino.

Cuando consultamos la vista previa del resultado de la asignación (en el panel *Resultados*), MapForce evalúa la condición conectada a la entrada **bool** del filtro en cada nodo expense-item. Si la condición se cumple (es true), el nodo expense-item se pasa al destino. De lo contrario, se pasa por alto. Como resultado se obtienen los gastos que cumplen los criterios:

```
...
```


© 2018 Altova GmbH

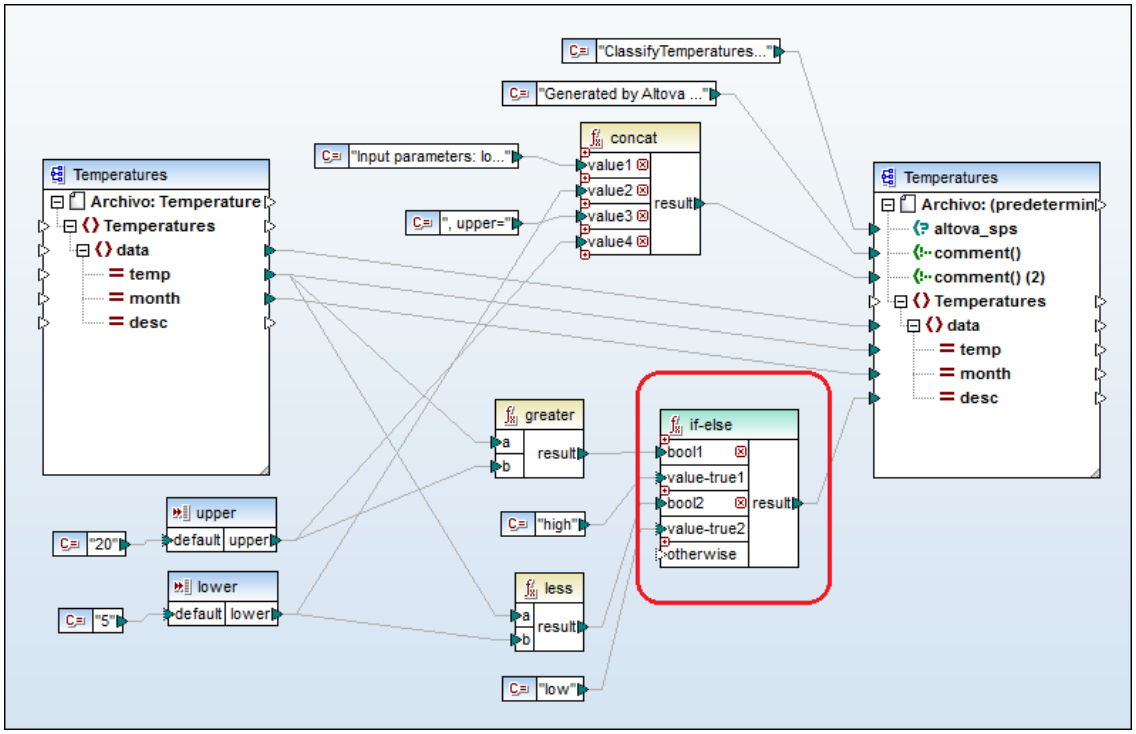
```
   <expense-item>
      <type>Meal</type>
      <Date>2003-01-01</Date>
      <expense>122.11</expense>
   </expense-item>
   <expense-item>
      <type>Lodging</type>
      <Date>2003-01-02</Date>
      <expense>299.45</expense>
   </expense-item>
...
```

XML de salida después de ejecutarse la asignación

5.9.2 Ejemplo: devolver un valor de forma condicional

Este ejemplo explica cómo obtener un valor simple de un componente con ayuda de una condición true/false. Para conseguirlo se utiliza una **condición If-Else** ( ). Las **condiciones If-Else** no se deben confundir con los componentes de filtrado. Las primeras son adecuadas para procesar valores simples de forma condicional (cadenas, enteros, etc.). Los componentes de filtrado, sin embargo, deben utilizarse para filtrar valores complejos como nodos (*véase* *Ejemplo: filtrar nodos*).

La asignación de este ejemplo es **<Documentos>\Altova\MapForce2018\MapForceExamples\ClassifyTemperatures.mfd**.



Esta asignación lee datos de un XML de origen que contiene datos sobre temperaturas ("Temperatures") y escribe datos en un XML de destino que cumple el mismo esquema. Entre el destino y el origen existen varios componentes más. De ellos el más relevante es la condición if-else (marcada en rojo).

El objetivo de la asignación es agregar una breve descripción a cada registro de temperatura en el documento de destino. Concretamente, si la temperatura supera los 20 grados centígrados, la descripción será "high" (alta). Si, por el contrario, la temperatura no supera los 5 grados centígrados, la descripción será "low" (baja). En el resto de los casos, no se incluirá ninguna descripción.

Para conseguir nuestro objetivo es necesario un procesamiento condicional y, por tanto, se añadió una condición if-else a la asignación. (Para añadir una condición if-else haga clic en el menú **Insertar** y después elija el comando **Condición If-Else**). En esta asignación la condición if-else se amplió (con ayuda del botón ⊡) para que incluyera dos condiciones: **bool1** y **bool2**.

Las condiciones propiamente dichas vienen dadas por las funciones **greater** y **less**, que se añadieron desde la biblioteca **core** de MapForce (*véase Trabajar con funciones*). Estas funciones evalúan los valores que aportan dos componentes de entrada llamados "upper" y "lower". (Para añadir un componente de entrada haga clic en el menú Insertar y elija el comando Insertar componente de entrada. Consulte el apartado Supplying Parameters to the Mapping para obtener más información.)

Las funciones **greater** y **less** devuelven el valor true o false. El resultado de la función determina qué texto se escribe en la instancia de destino. Es decir, si el valor del atributo "temp" del XML de origen es superior a 20, entonces se pasa el valor de constante "high" a la condición if-else. Si el valor del atributo "temp" del XML de origen es inferior a 5, entonces se pasa el valor de constante "low" a la condición **if-else**. La entrada **otherwise** de la condición se deja sin conectar. Por tanto, si no se cumple ninguna de las condiciones anteriores, no se pasará nada al conector de salida **result**.

Por último, el conector de salida **result** suministra este valor (uno por cada registro de temperatura) al atributo "desc" del componente de destino.

Para consultar una vista previa del resultado de la asignación haga clic en el panel *Resultados*. Observe que el XML de salida resultante incluye ahora el atributo "desc" cuando la temperatura es superior a 20 o inferior a 5.

```
...
   <data temp="-3.6" month="2006-01" desc="low"/>
   <data temp="-0.7" month="2006-02" desc="low"/>
   <data temp="7.5" month="2006-03"/>
   <data temp="12.4" month="2006-04"/>
   <data temp="16.2" month="2006-05"/>
   <data temp="19" month="2006-06"/>
   <data temp="22.7" month="2006-07" desc="high"/>
   <data temp="23.2" month="2006-08" desc="high"/>
...
```

XML de salida después de ejecutarse la asignación

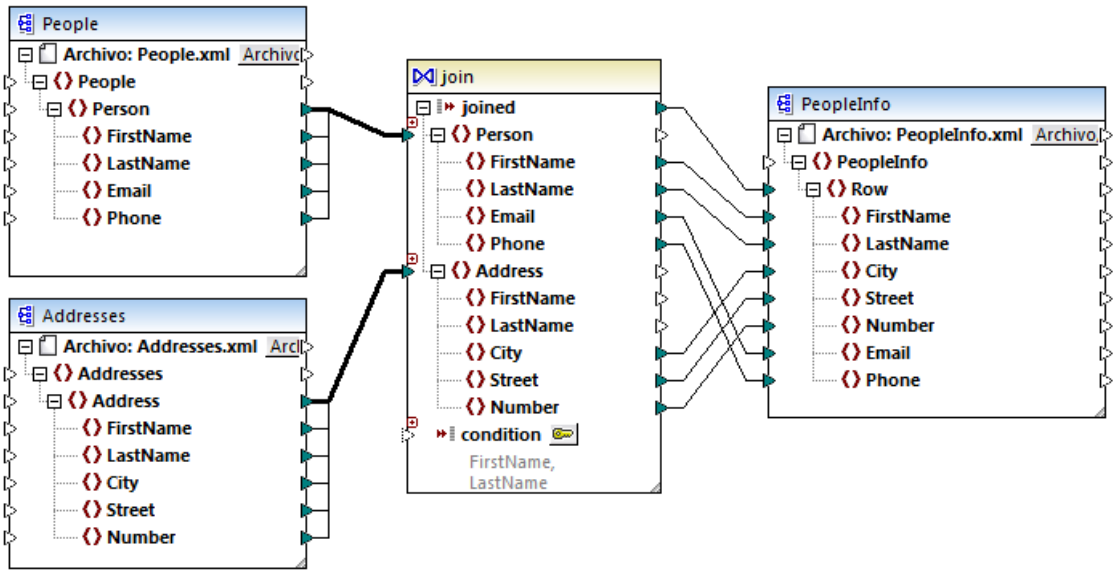
© 2018 Altova GmbH

5.10 Combinar datos

A veces es necesario combinar datos de dos o más estructuras en base a una condición (p. ej. si el campo A de la primera estructura tiene el mismo valor que el campo B de la segunda estructura). Para este tipo de diseños puede usar un componente de combinación.

Un componente de combinación es un componente de MapForce que permite unir varias estructuras en la asignación de acuerdo con las condiciones definidas por el usuario. Devuelve la asociación (conjunto combinado) de elementos que cumplen con la condición. Las combinaciones son de gran ayuda a la hora de combinar datos de dos estructuras que tienen un campo en común (p. ej. una identidad).

Por ejemplo, en la asignación que aparece más abajo, el componente del medio es un componente de combinación. En este diseño de asignación se combinan dos estructuras XML (una lista de personas y una lista de direcciones). El objetivo es recopilar todos los datos personales de cada persona en un archivo XML de destino. Los campos `FirstName` y `LastName` hacen de claves de combinación. Es decir, si `FirstName` y `LastName` (secundarios de `Person`) tienen el mismo valor que `FirstName` y `LastName` (secundarios de `Address`), entonces los datos de la dirección pertenecen a la misma persona y, por tanto, se combinan. Los nodos de la estructura combinada se pueden seguir procesando en un componente de destino (en este caso un archivo XML). La condición de combinación propiamente dicha se define en las propiedades del componente de combinación con el botón **Definir condición de combinación** . Este ejemplo viene con una asignación de muestra y se explica con más detalla en el apartado Ejemplo: combinar estructuras XML.



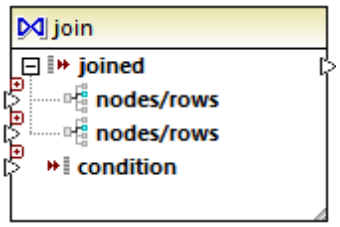
JoinPeopleInfo.mfd

Como puede ver en la imagen, las estructuras de origen y el componente de combinación están conectadas por medio de una conexión de copia total, para reducir la acumulación de elementos en el área de diseño. En general, este tipo de conexiones se crean automáticamente en MapForce si el contexto es relevante (*véase Conexiones de copia total*).

Las estructuras que se deben combinar pueden ser de componentes distintos (como en el ejemplo anterior) o pueden pertenecer al mismo componente. Las estructuras que se deben combinar también pueden ser de tipos distintos (p. ej. una estructura XML y una tabla de BD). Consulte el apartado Combinar datos de base de datos para aprender cómo funciona la combinación en las bases de datos.

Para agregar un componente de combinación:

1. Active el lenguaje de transformación BUILT-IN (haciendo clic en el botón BUILT IN de la barra de herramientas o con el comando de menú **Resultados | Motor de ejecución integrado**).
2. Haga clic en el comando de menú **Insertar | Combinación** o en el botón **Combinación** de la barra de herramientas. El componente de combinación aparece en la asignación y por defecto acepta datos de dos estructuras, así que tiene dos entradas `nodes/rows`. Si lo necesita, puede agregar entradas nuevas en la combinación con ayuda del botón **Agregar entrada** (*véase Combinar tres o más estructuras*).



3. Conecte las estructuras que se deben combinar con las entradas `nodes/rows` del componente de combinación.
4. Añada la condición para la combinación (o varias condiciones). Esto se consigue haciendo clic con el botón derecho en el componente de combinación y seleccionando **Propiedades**. También puede añadir condiciones de combinación desde la asignación directamente: conecte el resultado booleano de una función con la entrada `condition` del componente de combinación. Cuando se están combinando tablas de base de datos, MapForce puede crear condiciones de combinación automáticamente (*véase Agregar condiciones de combinación*).

Notas:

- Se admite el uso de componentes de combinación cuando el lenguaje de destino de la asignación es el motor de ejecución integrado (BUILT-IN). No es compatible con la generación de código C#, C++ o Java.
- Cuando una estructura no es una fuente de entrada válida ni compatible para la combinación, MapForce emite sugerencias en la asignación directamente o, cuando se valide la asignación, en la ventana Mensajes (*véase Validar asignaciones*).
- No se deberían conectar componentes de combinación con parámetros de entrada ni resultados de funciones definidas por el usuario inline. Si existen dichas conexiones, se producirán errores de validación.
- Cuando conecte componentes de base de datos (tablas o vistas, por ejemplo) a un componente de combinación directamente, en la esquina superior derecha del componente de combinación se habilitará el botón **Modo SQL** SQL. Si se activa, este botón ofrece funciones SQL especiales para trabajar con la operación de combinación (*véase Combinaciones en modo SQL*).

© 2018 Altova GmbH

- No se puede conectar la salida del nodo `joined` con otro componente de combinación, pero si lo necesita puede conectar un resultado parcial de una combinación con otra.

Diferencias entre componentes de combinación y otros tipos de componentes

En algunos casos se puede conseguir el mismo resultado usando variables o filtros complejos en lugar de componentes de combinación (*véase Usar variables y Filtros y condiciones*). Sin embargo, a diferencia de otros componentes, los de combinación facilitan la comprensión de la asignación porque permiten distinguir a simple vista qué datos se combinan. Además, si se habilita el modo SQL en el componente de combinación, el rendimiento de la asignación aumenta considerablemente cuando se trata de combinaciones de bases de datos (*véase Combinar datos de base de datos*).

Agregar un contexto primario

En casos especiales para poder conseguir un resultado concreto se podrá ofrecer explícitamente un contexto de asignación (el denominado *contexto primario*) para los datos que están conectados con el componente de combinación. Para agregar un contexto primario haga clic con el botón derecho en el nodo `joined` del componente de combinación y seleccione **Agregar contexto primario** en el menú contextual. A partir de ese momento, el componente de combinación incluirá una entrada nueva llamada `parent-context` donde podrá conectar el nodo de origen correspondiente. Consulte Reemplazar el contexto de la asignación para obtener más información.

5.10.1 Agregar condiciones de combinación

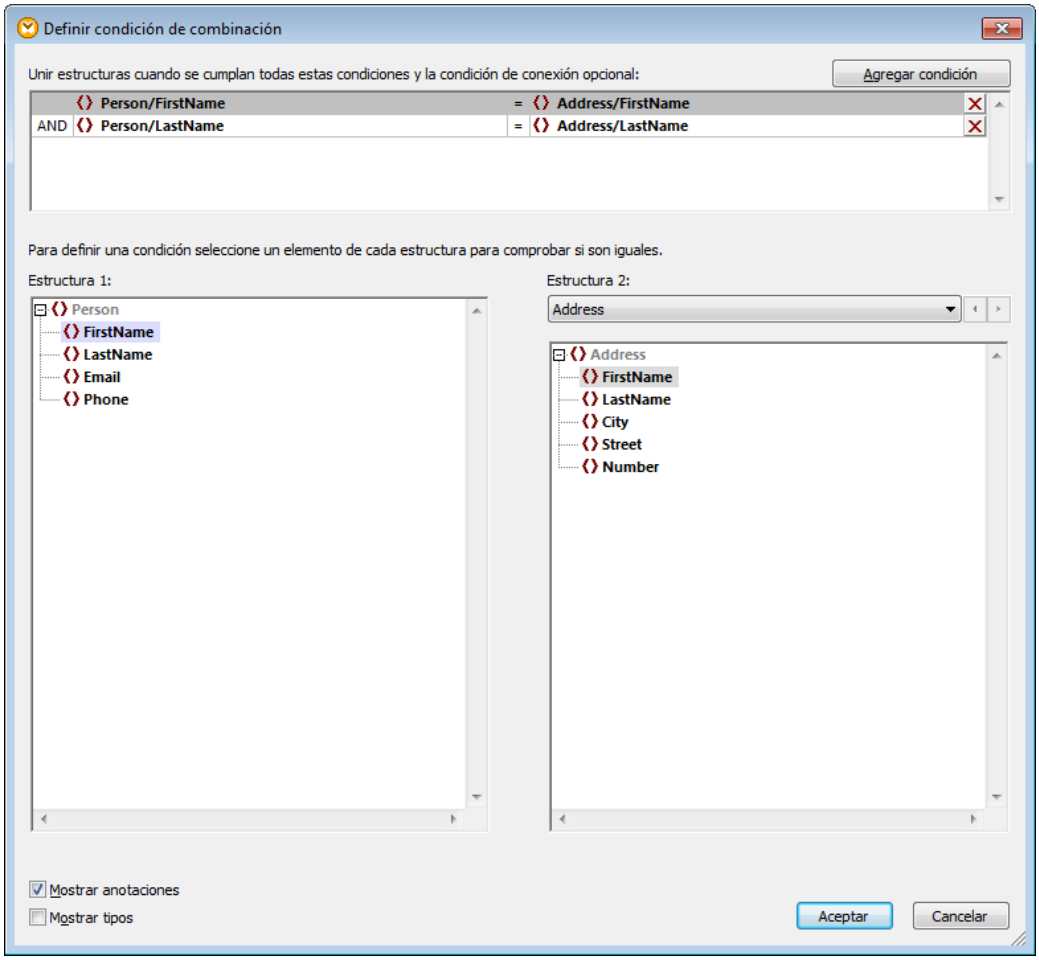
Una combinación une elementos de dos o más estructuras en base a una condición así que una combinación siempre necesita una condición como mínimo. Hay varias maneras de agregar condiciones de combinación, tal y como explicamos más abajo.

Nota: cuando se combinan tablas de base de datos en modo SQL, MapForce crea las condiciones de combinación automáticamente a partir de las relaciones de clave foránea detectadas entre las tablas. Para que MapForce pueda crear condiciones de combinación automáticas las tablas de la base de datos deben estar en una relación secundario/primario en el componente de MapForce (es decir, una tabla debe ser el primario o secundario de otra tabla en el componente). Consulte el apartado Ejemplo: combinar tablas en modo SQL para obtener más información.

Método 1: agregar una condición de combinación desde las propiedades del componente

1. En la asignación primero debe comprobar que como mínimo hay dos estructuras o tablas de BD conectadas al componente de combinación. (El componente de combinación de este ejemplo forma parte de la asignación **JoinPeopleInfo.mfd**, disponible en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**. Esta asignación se describe más detalladamente en el apartado Ejemplo: combinar estructuras XML.
2. Haga clic en el botón **Definir condición de combinación** del componente de combinación o haga clic con el botón derecho en el título del componente y seleccione **Propiedades**.
3. Seleccione un elemento de la estructura de la izquierda y otro de la estructura de la

derecha (es decir, cuando la comparación de este par dé true como resultado, las estructuras se combinarán).



Si necesita agregar más condiciones haga clic en **Agregar condición** y después seleccione un nuevo par de elementos. Por ejemplo, en la imagen anterior se definieron dos condiciones de combinación:

1. `FirstName` en Estructura 1 debe ser igual a `FirstName` en Estructura 2 y
2. `LastName` en Estructura 1 debe ser igual a `LastName` en Estructura 2.

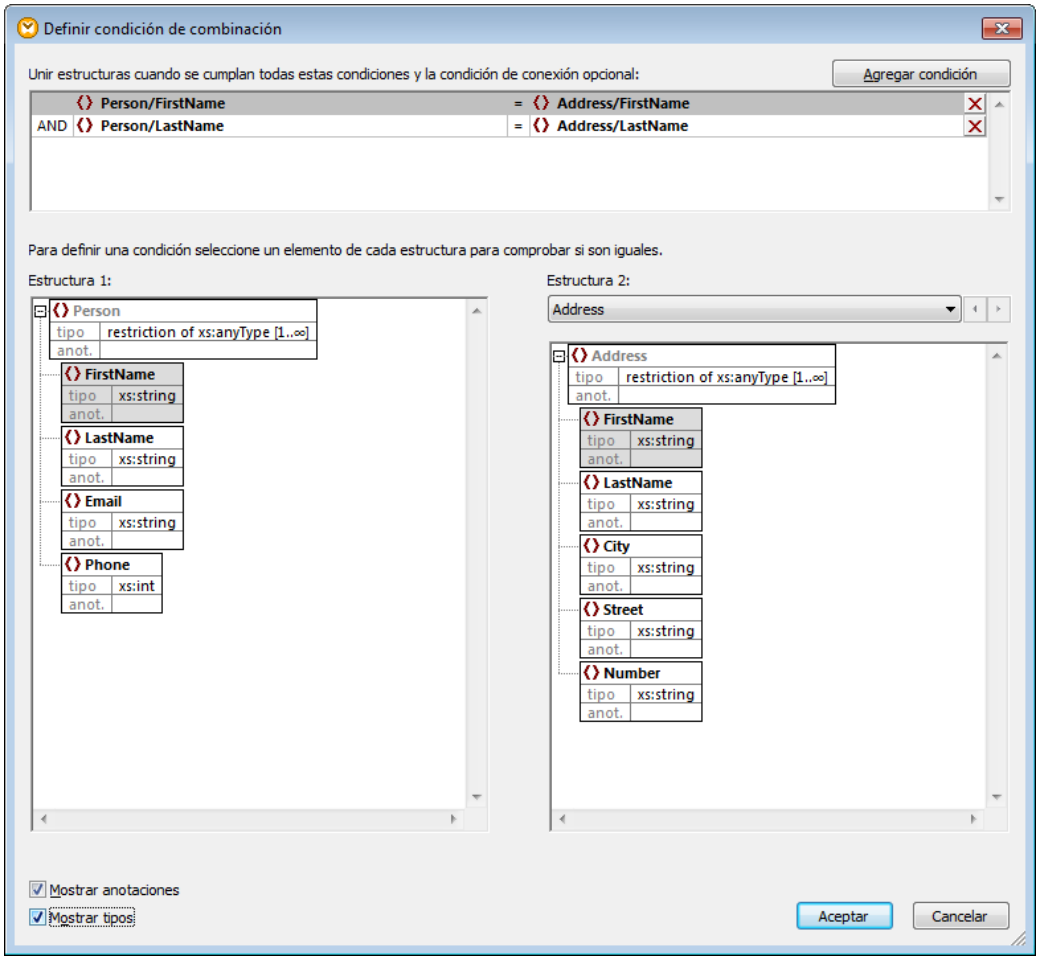
Para eliminar una condición de combinación haga clic en el botón **Eliminar** de la condición.

Notas:

- Cuando existen varias condiciones de combinación, todas deben cumplirse o de lo contrario no se combinarán las dos estructuras. En otras palabras, cuando existen varias condiciones, éstas se unen con la operación lógica AND. Esto también incluye condiciones opcionales que se añadieran desde la asignación (ver método 2 más abajo).
- Si hay más de dos estructuras conectadas al componente de combinación, esas estructuras adicionales aparecerán en la lista desplegable situada debajo de *Estructura 2.* Cuando seleccione una de estas estructuras adicionales en la lista desplegable, el panel izquierdo mostrará todas las estructuras que aparecen antes de ella en el

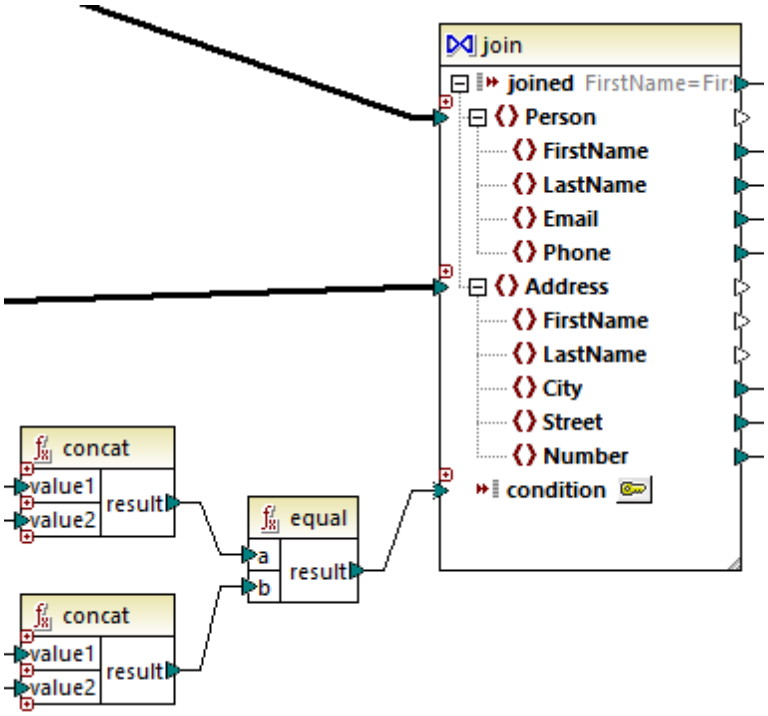
componente de combinación. Así podrá definir condiciones de combinación entre cualquier estructura. Para ver un ejemplo consulte Ejemplo: crear un informe CSV a partir de varias tablas.

- Para ver el tipo de datos de los elementos de cada estructura marque la casilla *Mostrar tipos*. La opción *Mostrar anotaciones* sirve para ver información adicional sobre los elementos (siempre y cuando dicha información exista en el esquema o base de datos subyacente). Si marca ambas casillas, la presentación de los datos cambia ligeramente en el cuadro de diálogo:



Método 2: agregar una condición de combinación desde la asignación

- En la asignación añada componentes que producen un valor booleano y después conecte la salida booleana a la entrada del nodo `condition`. Por ejemplo, la función **`equal`** puede comparar un valor con un nodo de la asignación y aportar el resultado booleano como entrada del nodo `condition` del componente de combinación.



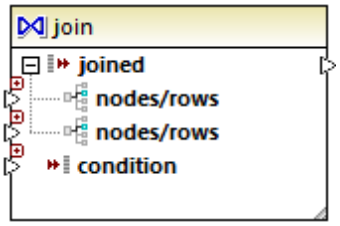
Nota: si no se define ninguna condición desde las propiedades del componente (método 1), debe conectarse el nodo `condition` del componente de combinación (método 2).

Método 3: método mixto

En la misma asignación se pueden definir algunas condiciones de combinación desde las propiedades del componente (método 1) y combinarlas con la condición definida en la asignación (método 2). No obstante, si intenta combinar tablas de BD en modo SQL, las condiciones solamente se podrán definir con el método 1 (*véase* *Combinaciones en modo SQL*).

5.10.2 Combinar tres o más estructuras

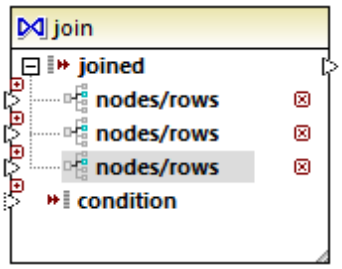
Si se añade un componente de combinación a la asignación con el comando de menú Insertar **| Combinación**, el componente acepta dos estructuras por defecto (es decir, el componente solo contiene dos entradas `nodes/rows`).



Si necesita combinar más de dos estructuras, haga clic en el botón **Agregar entrada** y cree tantos `nodes/rows` como necesite. Si necesita eliminar una entrada `nodes/rows`, haga clic en el

 © 2018 Altova GmbH

botón **Eliminar entrada** ⊠. Tenga en cuenta que un componente de combinación necesita dos estructuras como mínimo (y, por tanto, el botón ⊠ solo está disponible si existen más de dos entradas).



Cuando un componente de combinación tiene varias entradas, las condiciones de combinación deben tener en cuenta cada una de las entradas (*véase* *Agregar condiciones de combinación*). Para ver un ejemplo con instrucciones para aprender a combinar varias tablas de base de datos consulte el apartado Ejemplo: crear un informe CSV a partir de varias tablas.

5.10.3 Ejemplo: combinar estructuras XML

Con este ejemplo aprenderá a combinar datos de dos estructuras XML de forma condicional y con ayuda de un componente de combinación. La asignación utilizada para el ejemplo está en esta ruta de acceso: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\JoinPeopleInfo.mfd**.

El objetivo de esta asignación es recopilar en un archivo XML de destino datos personales (nombre, apellido, dirección postal, dirección de correo electrónico y teléfono) procedentes de dos archivos XML de origen.

El primer archivo XML de origen contiene el nombre y apellido de cada persona, así como su dirección de correo electrónico y teléfono (observe que para simplificar se han omitido la declaración XML, los espacios de nombres y algunos registros):

```
<People>
   <Person>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <Email>m.bailey@nanonull.com</Email>
      <Phone>555323698</Phone>
   </Person>
   <Person>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <Email>t.rea@nanonull.com</Email>
      <Phone>555598653</Phone>
   </Person>
</People>
```

People.xml

El segundo archivo XML de origen contiene el nombre y apellido de cada persona, así como su

dirección postal:

```
<Addresses>
   <Address>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <City>Bridgedell</City>
      <Street>Olive Street</Street>
      <Number>4</Number>
   </Address>
   <Address>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <City>Roseford</City>
      <Street>Evergreen Lane</Street>
      <Number>34</Number>
   </Address>
</Addresses>
```

Addresses.xml

El objetivo de la asignación es combinar la información de `<Person>` del primer archivo con la información de `<Address>` del segundo archivo, siempre y cuando coincidan el nombre y el apellido. Concretamente, por cada `<Person>` del primer archivo y por cada `<Address>` del segundo archivo, deben compararse `FirstName` y `LastName`. Si tienen valores idénticos, entonces los registros `<Person>` y `<Address>` correspondientes se refieren a la misma persona y se deben combinar. La estructura XML de destino tendría este aspecto:

```
<PeopleInfo>
   <Row>
      <FirstName>Marquita</FirstName>
      <LastName>Bailey</LastName>
      <City>Bridgedell</City>
      <Street>Olive Street</Street>
      <Number>4</Number>
      <Email>m.bailey@nanonull.com</Email>
      <Phone>555323698</Phone>
   </Row>
   <Row>
      <FirstName>Totie</FirstName>
      <LastName>Rea</LastName>
      <City>Roseford</City>
      <Street>Evergreen Lane</Street>
      <Number>34</Number>
      <Email>t.rea@nanonull.com</Email>
      <Phone>555598653</Phone>
   </Row>
</PeopleInfo>
```

PeopleInfo.xml

Esto se consigue fácilmente añadiendo un componente de combinación a la asignación.

© 2018 Altova GmbH

Recuerde que puede conseguir el mismo resultado usando otro tipo de componente, pero en este ejemplo usaremos un componente de combinación.

Paso nº1: agregar los archivos XML de origen en la asignación

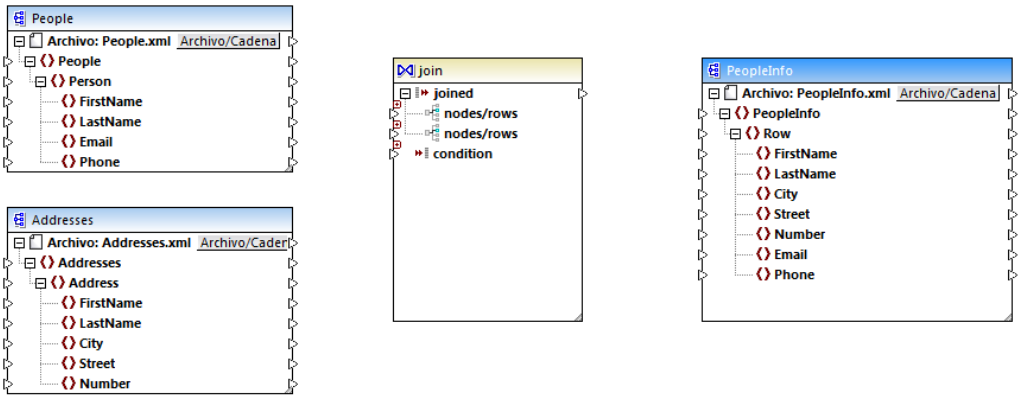
1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta este archivo de origen: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\People.xml**.
2. Repita el paso anterior e inserte este archivo **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Addresses.xml**.

Paso nº2: agregar el archivo de esquema de destino en la asignación

1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta este archivo: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\PeopleInfo.xsd** (el archivo de esquema de destino).
2. Cuando la aplicación solicite un archivo XML de muestra, haga clic en **Omitir**.
3. Cuando la aplicación solicite un elemento raíz, seleccione `PeopleInfo` como elemento raíz.

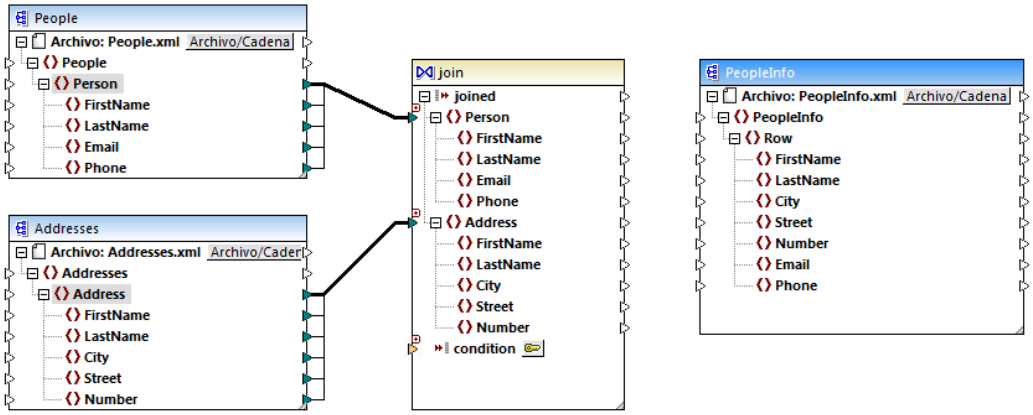
Paso nº3: agregar el componente de combinación

1. En el menú **Insertar** haga clic en el comando **Combinación** (o haga clic en el botón **Combinación** de la barra de herramientas). Llegados a este punto la asignación debería tener este aspecto (deberá arrastrar los componentes y ajustar su tamaño para conseguir este diseño):

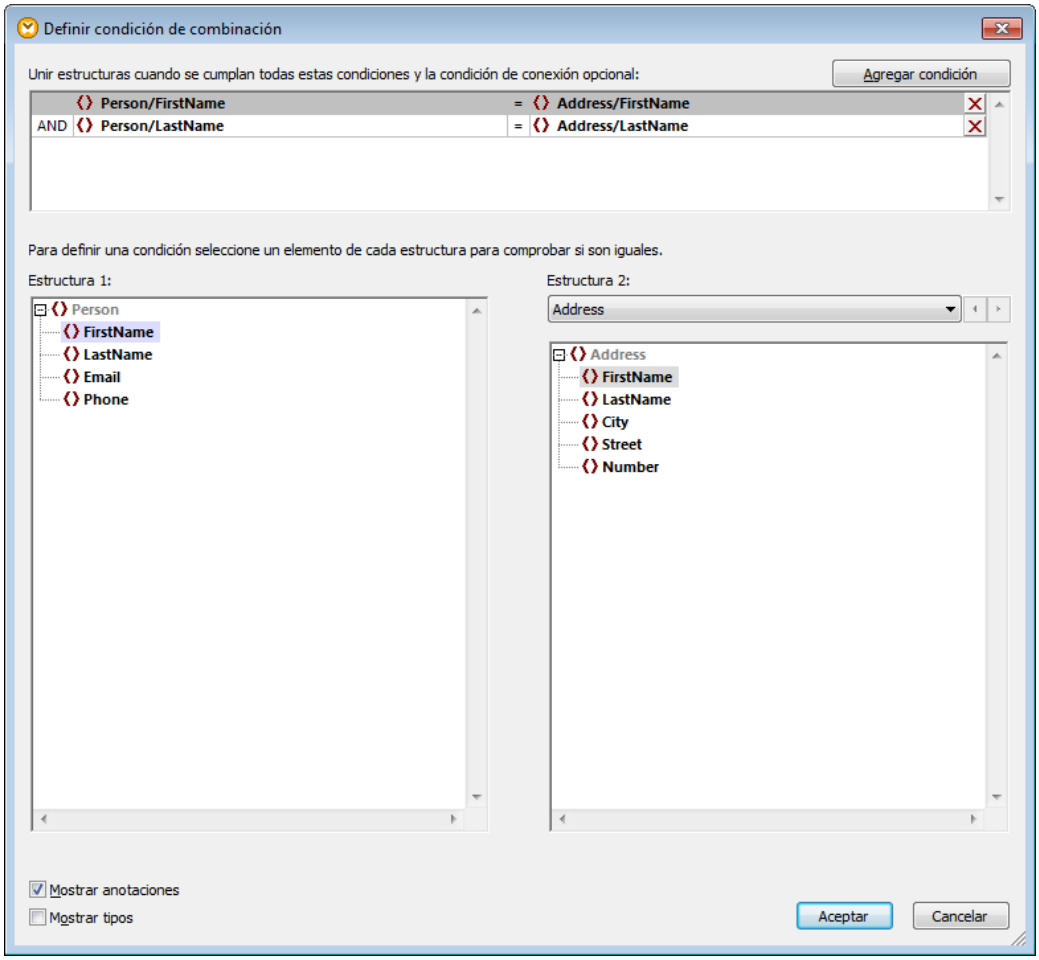


Fíjese en la estructura del componente de combinación. Tiene dos entradas `nodes/rows` para que podamos conectarlo a las dos estructuras que deseamos comparar (las estructuras `Person` y `Address` en este caso).

2. Dibuje una conexión entre `Person` y la primera entrada `nodes/rows` del componente de combinación. Igualmente, conecte `Address` con la otra entrada `nodes/rows`.



3. Como dijimos anteriormente, la combinación solo debería tener lugar si `FirstName` y `LastName` tienen valores idénticos en ambas estructuras. Para definir esta condición haga clic en el botón **Definir condición de combinación** del componente.
4. Seleccione el par de elementos que definen la primera condición de combinación (`FirstName` en Estructura 1 y `FirstName` en Estructura 2).
5. Haga clic en **Agregar condición** y repita el paso anterior con `LastName` esta vez.



 © 2018 Altova GmbH

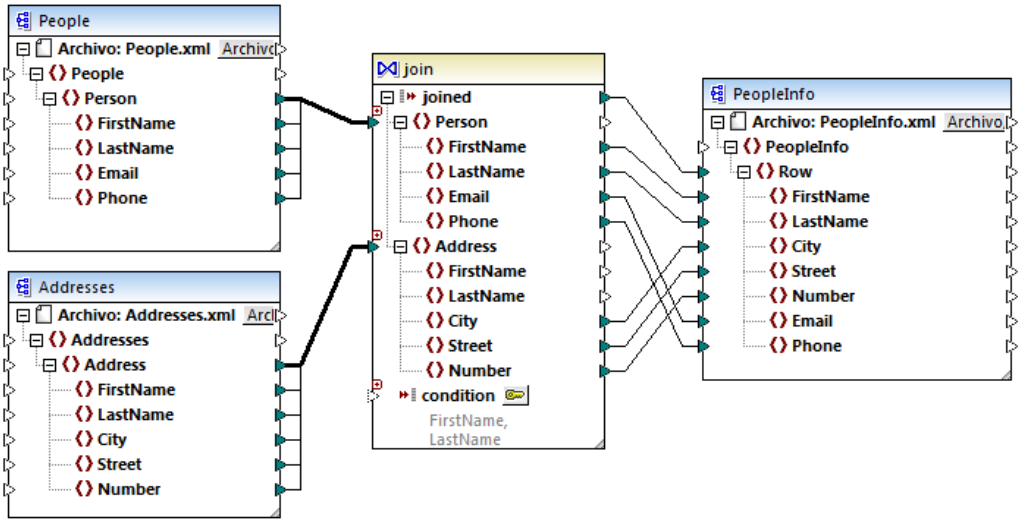
En algunas asignaciones puede ser suficiente una condición formada por una comparación. Sin embargo, en este ejemplo es importante crear dos comparaciones:

1) `FirstName` en Estructura 1 = `FirstName` en Estructura 2
2) `LastName` en Estructura 1 = `LastName` en Estructura 2.

Cuando se definen varias condiciones, todas ellas deben cumplirse. De lo contrario no tiene lugar la combinación. Por tanto, en este ejemplo, la combinación solo tendrá lugar si las dos comparaciones dan como resultado true. Si, por el contrario, solo se definiera una comparación, podrían combinarse los datos de personas que tienen el mismo nombre pero distinto apellido.

Paso nº4: asignar el componente de combinación al esquema de destino

Ahora que se combinaron las dos estructuras, ya puede definir qué elementos de la estructura combinada se deben asignar al componente de destino. Esto se hace creando conexiones entre los elementos de la estructura combinada y los del componente de destino (*imagen siguiente*). La conexión que une `joined` con `Row` tiene este propósito: cuando se cumpla la condición de combinación, se crea un elemento `Row` nuevo en el componente de destino.



Para consultar una vista previa de los resultados de la asignación abra el panel *Resultados*. Como se esperaba, ahora los registros de las personas (`<Row>`) incluyen también la dirección postal.

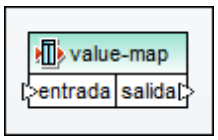
5.11 Usar asignaciones de valores

El componente Asignación de valores sirve para transformar un valor de entrada en un valor de salida distinto por medio de una tabla de consulta. Es un componente muy práctico a la hora de convertir diferentes tipos de enumeración. Solamente está compuesto por un elemento de entrada y otro de salida.

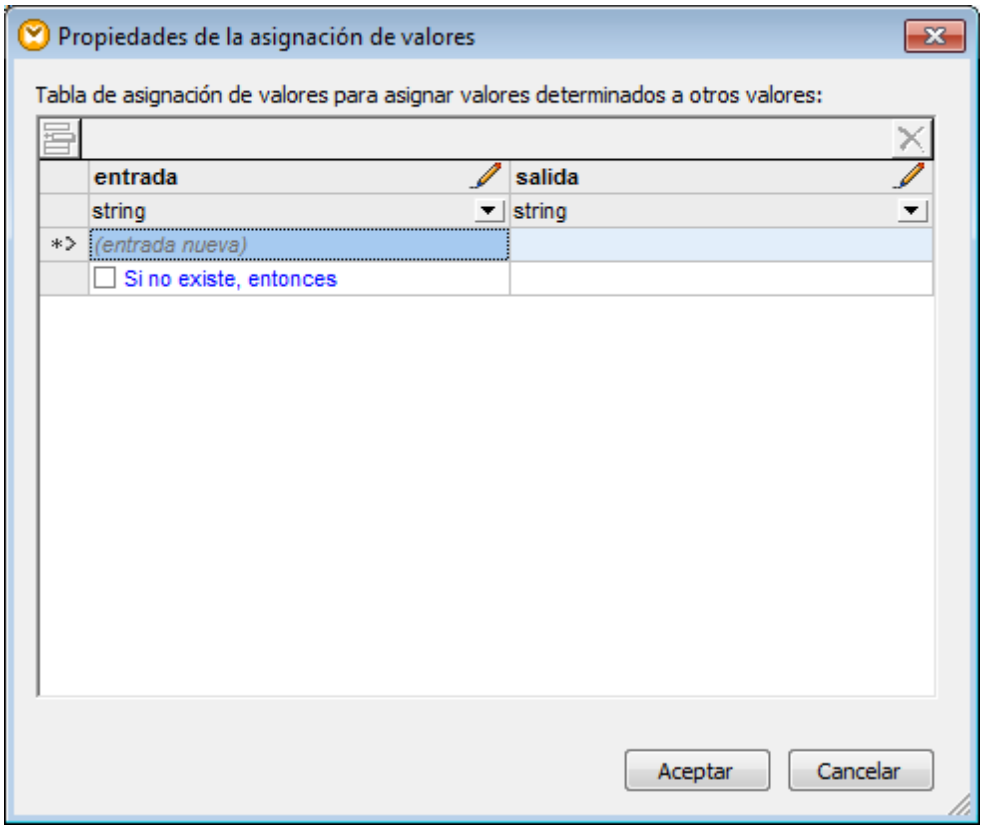
Nota: si desea recuperar/filtrar datos en base a determinados criterios, se recomienda el componente Filtro (*véase* *Filtros y condiciones*).

Para usar un componente Asignación de valores:

1. Seleccione el comando de menú **Insertar | Asignación de valores** o haga clic en el icono **Asignación de valores** de la barra de herramientas.

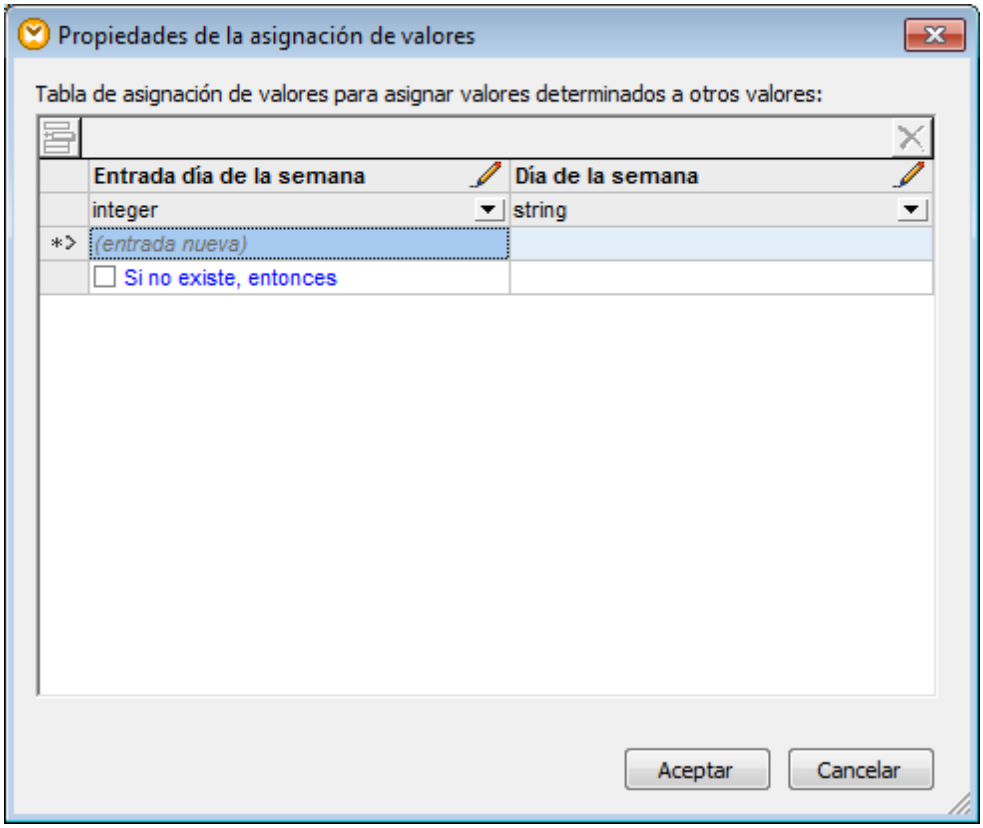


2. Haga doble clic en el componente **value-map** para abrir la tabla de asignación de valores.

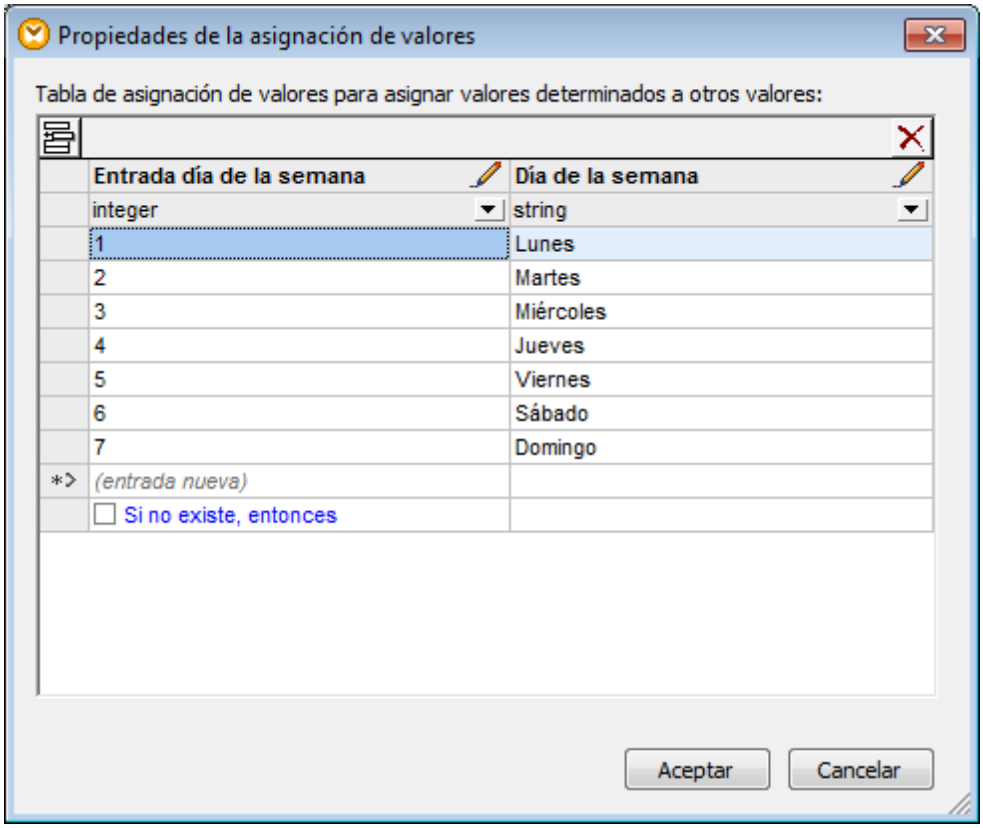


3. Haga clic en los encabezados de columna e introduzca el texto `Entrada día de la semana` en el primer encabezado y `Día de la semana` en el segundo.

© 2018 Altova GmbH



4. Ahora introduzca el valor de entrada que desea transformar en la columna `Entrada día de la semana`.
5. En la columna `Día de la semana` introduzca el valor de salida al que se debe transformar el valor de entrada.
6. Para introducir un par de valores nuevo introduzca texto en el campo de entrada *`(entrada nueva)`*.
7. Haga clic en el cuadro combinado de tipo de datos (situado justo debajo del encabezado de columna) para seleccionar los tipos de datos de entrada y salida (p. ej. `integer` y `string`).

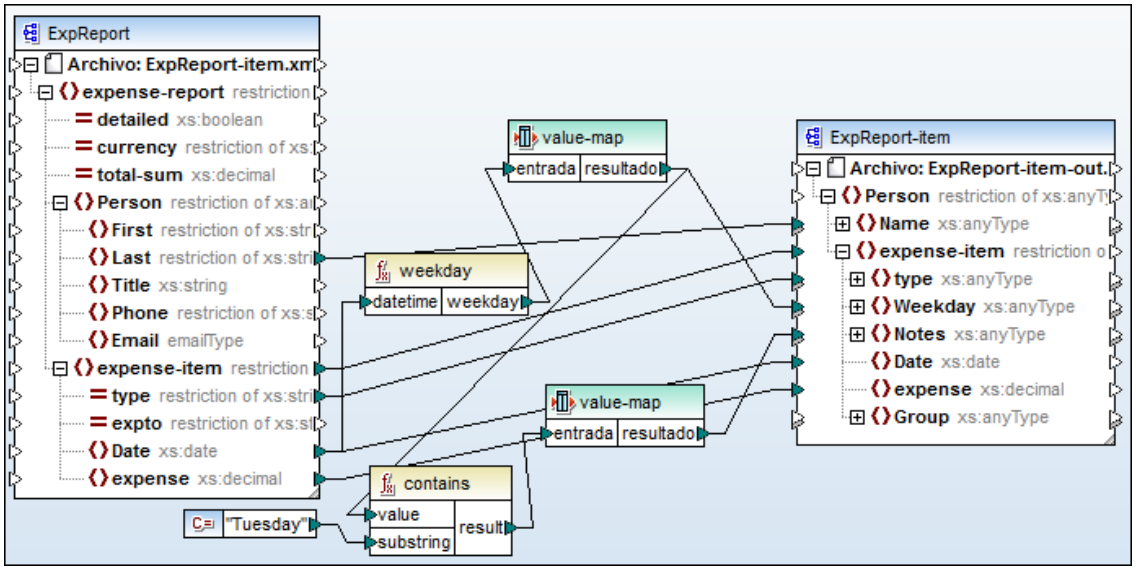


Nota: para definir un valor de salida alternativo si los valores dados no existen en la entrada, marque la casilla `Si no existe, entonces` e introduzca el valor alternativo. Para aprender a pasar datos de origen sin modificarlos consulte el apartado Pasar datos sin modificarlos.

8. Para cambiar el nombre de las columnas (que también aparecen en el diseño de asignación) haga clic en los iconos de edición de las filas de encabezado. Esto le permitirá identificar con facilidad qué papel desempeña el componente en la asignación de datos.

El archivo **Expense-valmap.mfd** de la carpeta ...\MapForceExamples\Tutorial\ contiene un componente **Asignación de valores**.

© 2018 Altova GmbH



¿Qué hace el diseño de asignación de datos **Expense-valmap.mfd**?
Extrae el día de la semana del elemento **Date** del origen de datos, convierte el valor numérico en texto y lo coloca en el elemento **Weekday** del componente de destino (es decir, Sunday, Monday, etc.).

- La función **weekday** extrae el número del día de la semana del elemento **Date** del archivo de origen **ExpReport**. El resultado de esta función son enteros comprendidos entre el 1 y el 7.
- La asignación de valores **value-map** transforma los enteros en días de la semana (es decir, Sunday, Monday, etc.) tal y como se explica al principio de este apartado.
- Si la salida contiene `Tuesday`, entonces se asigna el valor de salida correspondiente al elemento **Notes** del componente de destino.
- Si hacemos clic en el panel *Resultados* podemos ver el archivo XML de destino con los datos ya transformados.

```
3     <Name>Landis</Name>
4     <expense-item>
5       <type>Meal</type>
6       <Weekday>Tuesday</Weekday>
7       <Notes>-- Prepare financial reports -- !</Notes>
8       <Date>2003-01-01</Date>
9       <expense>122.11</expense>
10    </expense-item>
11    <expense-item>
12      <type>Lodging</type>
13      <Weekday>Monday</Weekday>
14      <Notes> --</Notes>
15      <Date>2003-01-14</Date>
16      <expense>122.12</expense>
17    </expense-item>
```

Nota: si pasamos el puntero por encima del componente value-map, aparece información rápida con los valores que están definidos.

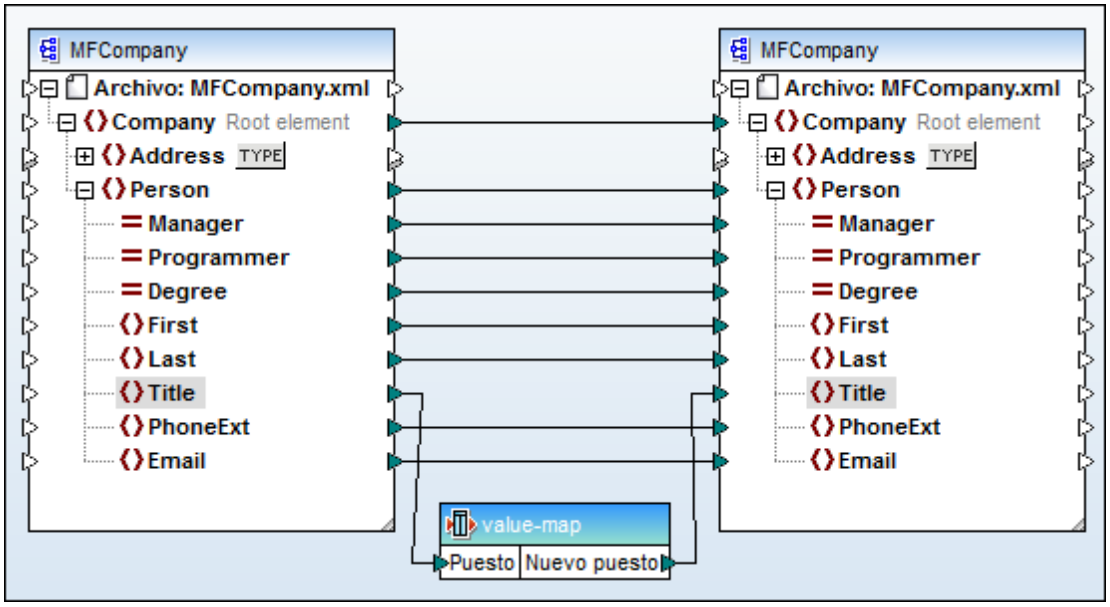
El resultado de varias funciones de cadena y funciones lógica solamente puede ser el valor

booleano **true** o **false**. Por tanto, el valor que se desea comprobar debe introducirse en el campo entrada de la tabla de la asignación de valores (p. ej. true).

5.11.1 Pasar datos sin modificarlos

En algunos casos es necesario transformar datos concretos de un nodo, mientras que el resto de datos del nodo deben pasarse al nodo de destino sin modificaciones.

Por ejemplo, imaginemos que en una filial de una compañía cambian los nombres de los puestos de los empleados. En nuestro ejemplo cambian dos nombres de puestos pero el resto de puestos no cambian.

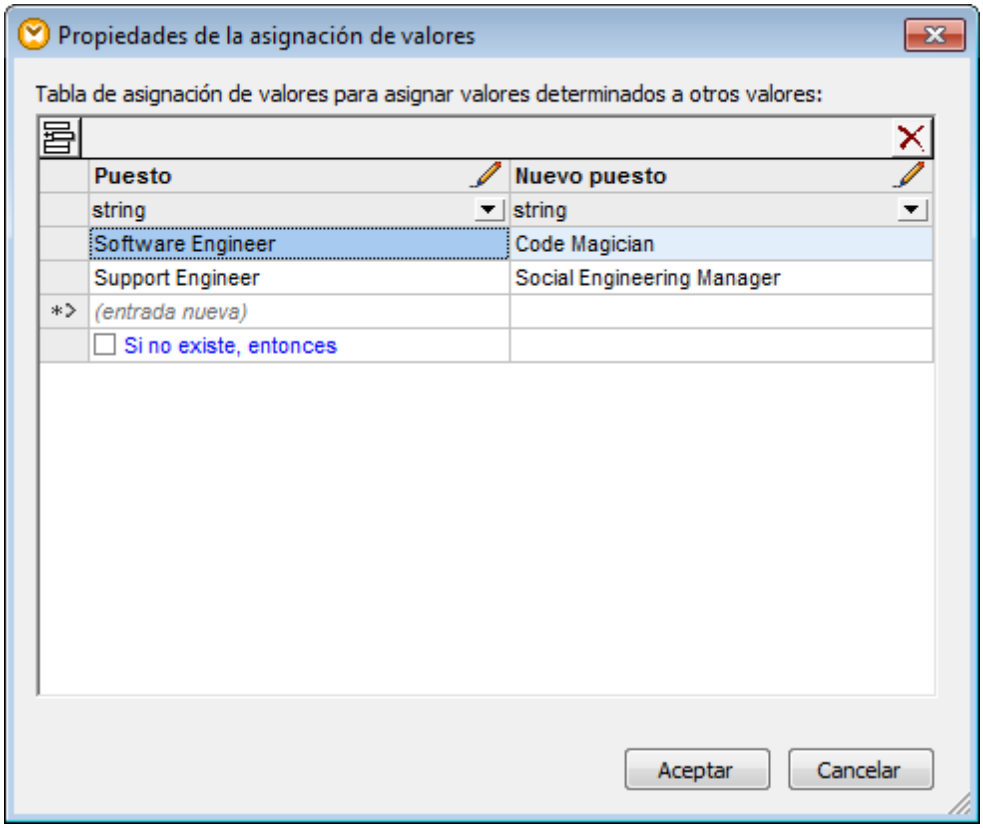


La asignación más obvia sería la que aparece en la imagen anterior, que utiliza una asignación de valores para transformar el nombre de los puestos.
Si abrimos el panel *Resultados* podremos ver el resultado de la asignación:

```
33  <Person>
34    <First>Fred</First>
35    <Last>Landis</Last>
36    <PhoneExt>951</PhoneExt>
37    <Email>f.landis@nanonull.com</Email>
38  </Person>
39  <Person>
40    <First>Michelle</First>
41    <Last>Butler</Last>
42    <Title>Code Magician</Title>
43    <PhoneExt>654</PhoneExt>
44    <Email>m.landis@nanonull.com</Email>
45  </Person>
```

En el caso de los empleados que no tengan ninguno de los dos tipos que muestra la asignación de valores, el elemento **Title** se eliminará en el documento de salida.

© 2018 Altova GmbH

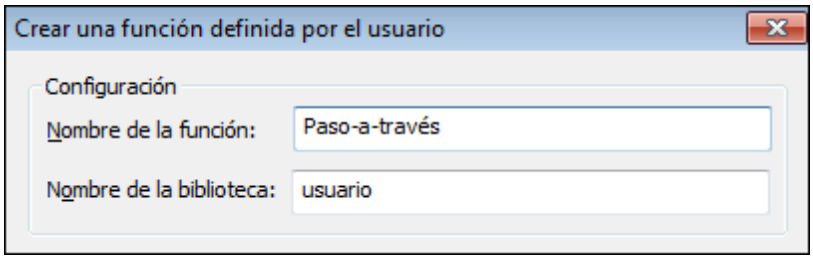


Otra manera de hacer esto sería marcar la casilla *Si no existe, entonces* e introducir un término de sustitución. El nodo **Title** volvería a aparecer en el documento de salida, pero ahora incluiría el mismo `Nuevo puesto` que los demás empleados de la compañía.

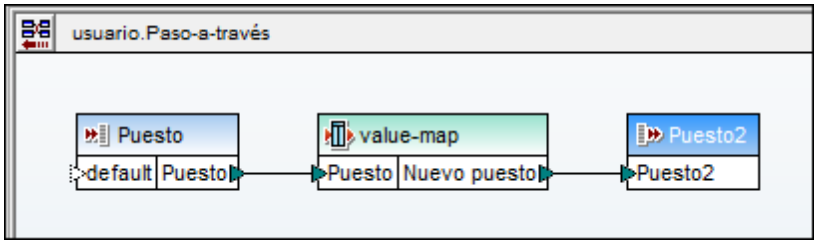
Solución:

Cree una función definida por el usuario que contenga un componente de asignación de valores y utilice la función **substitute-missing** para suministrar los datos iniciales para los nodos vacíos.

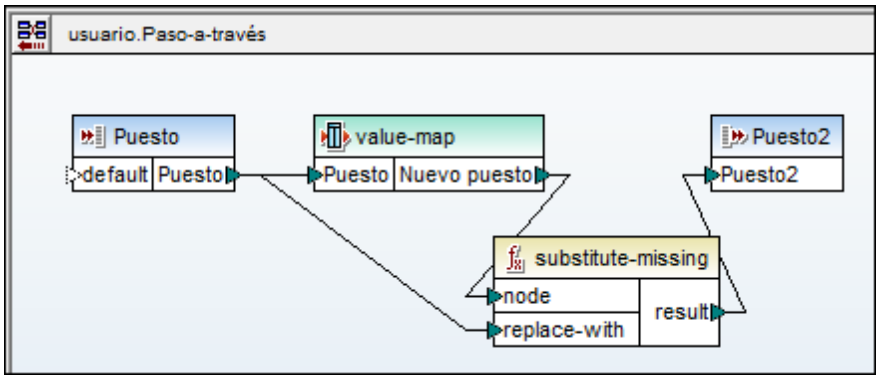
1. Haga clic en la asignación de valores **value-map** y seleccione **Función | Crear una función definida por el usuario a partir de la selección**.



2. Introduzca un nombre para esta función (p. ej. Paso-a-través) y haga clic en **Aceptar**.



3. Inserte una función **substitute-missing** desde la sección **core | node function** del panel Bibliotecas y cree las conexiones que aparecen en esta imagen:



4. Ahora puede abrir el panel *Resultados* para ver el resultado.

Resultado de la asignación:

- Las dos designaciones Title de la asignación de valores value-map se transformaron en Nuevo puesto.
- Los demás nodos Title del archivo de origen conservan sus datos iniciales en el archivo de destino.

```
38  <Person>
39    <First>Fred</First>
40    <Last>Landis</Last>
41    <Title>Program Manager</Title>
42    <PhoneExt>951</PhoneExt>
43    <Email>f.landis@nanonull.com</Email>
44  </Person>
45  <Person>
46    <First>Michelle</First>
47    <Last>Butler</Last>
48    <Title>Code Magician</Title>
49    <PhoneExt>654</PhoneExt>
50    <Email>m.landis@nanonull.com</Email>
51  </Person>
```

¿Por qué ocurre esto?
La asignación de valores value-map evalúa los datos de entrada de la siguiente manera:

- Si los datos entrantes **coinciden con una** de las entradas de la primera columna, los datos se transforman y se pasan al parámetro `node` de la función **substitute-missing** y después a `Puesto2`.

 © 2018 Altova GmbH

- Si los datos entrantes **no coinciden con ninguna** entrada de la columna izquierda, no se pasa ningún dato de la asignación de valores al parámetro `node` (es decir, es un nodo vacío).

  Cuando esto ocurre, la función **substitute-missing** recupera el nodo y los datos iniciales del nodo `Title` y los pasa a través del parámetro **replace-with** hasta `Puesto2`.

5.11.2 Propiedades de las asignaciones de valores

Acciones:

 Este icono **inserta** una fila nueva antes de la fila activa.

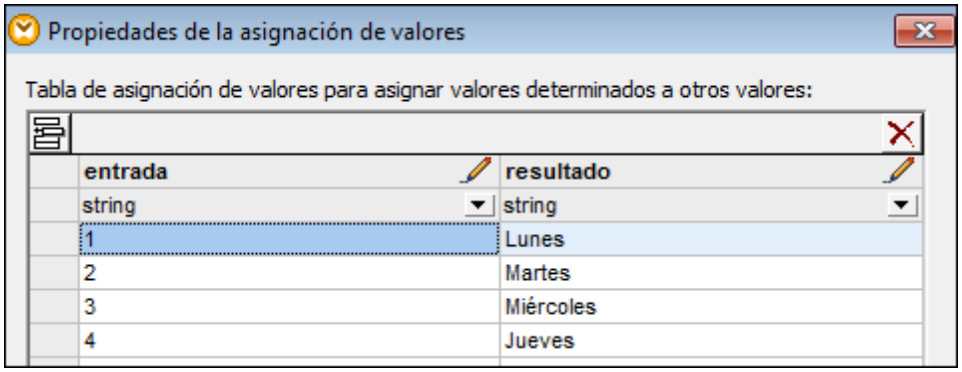
 Este icono **elimina** la fila activa.

 Este icono sirve para **editar** el encabezado de columna.

También puede cambiar el orden de las filas arrastrándolas y colocándolas en una nueva posición.

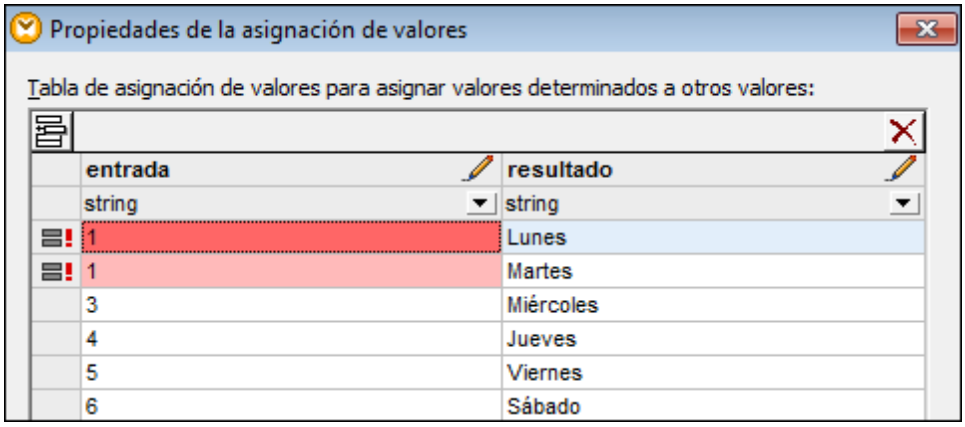
Cambiar el encabezado de columna:

Haga doble clic en el encabezado de columna (o clic en el icono ) para editar el nombre de la columna. Los nombres de las columnas aparecen en el diseño de asignación y así podrá identificar con facilidad qué papel desempeña el componente.



Usar valores de entrada únicos:

Los valores que introduzca en la columna de entrada deben ser valores únicos. Si introduce dos valores idénticos, ambos se resaltan en rojo para que los corrija.

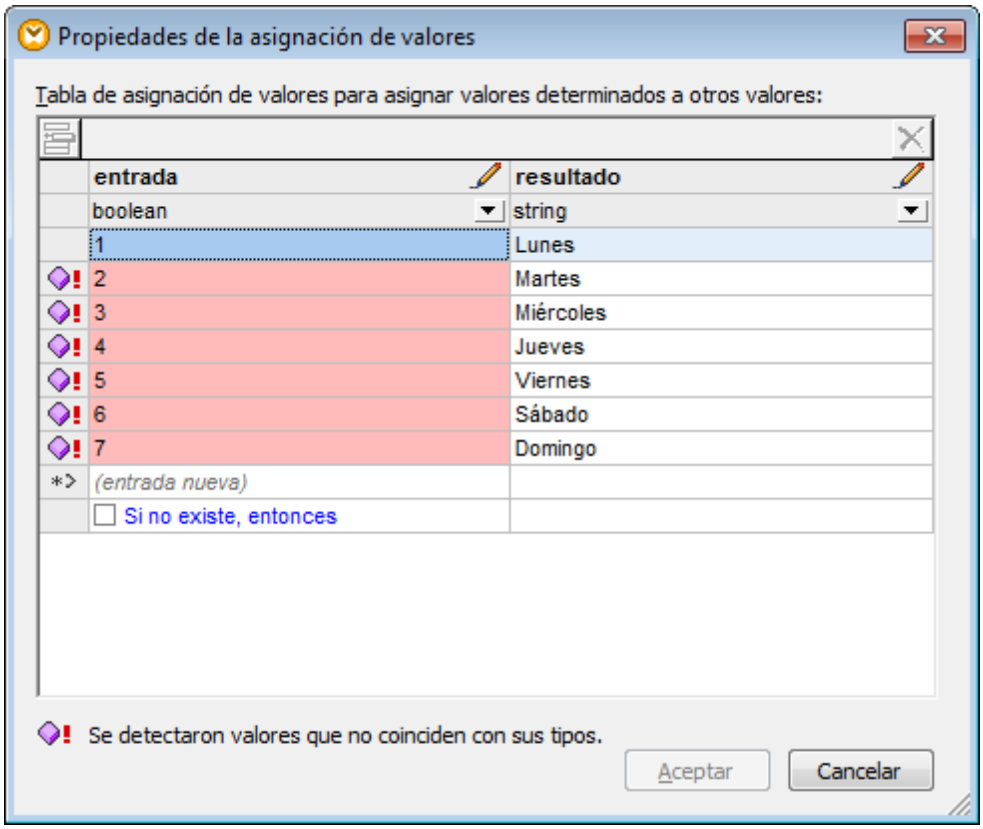


Una vez corregidos los valores, el botón **Aceptar** del cuadro de diálogo se habilita de nuevo.

Tipos de datos de entrada y salida

Los tipos de datos de entrada y salida se verifican automáticamente cuando se selecciona una entrada en el cuadro combinado. Si hay un error de coincidencia, se resaltan en rojo los campos pertinentes y se deshabilita el botón **Aceptar** del cuadro de diálogo.

En la imagen siguiente, por ejemplo, se seleccionaron los tipos de datos `boolean` y `string`.



 © 2018 Altova GmbH

5.12 Añadir excepciones

Una excepción es un tipo especial de componente que permite detener el proceso de asignación de datos y emitir un error cuando se cumple una condición devuelta por un filtro. Puede añadir una excepción si su asignación de datos incluye un filtro que comprueba si se cumple una condición true/false (*véase Filtros y condiciones*). Por ejemplo, puede diseñar la asignación para que emita una excepción si el valor de algún elemento de la asignación es superior a un valor predefinido.

También puede añadir excepciones en los diseños de servicios web WSDL de MapForce (*véase Errores de servicio web*).

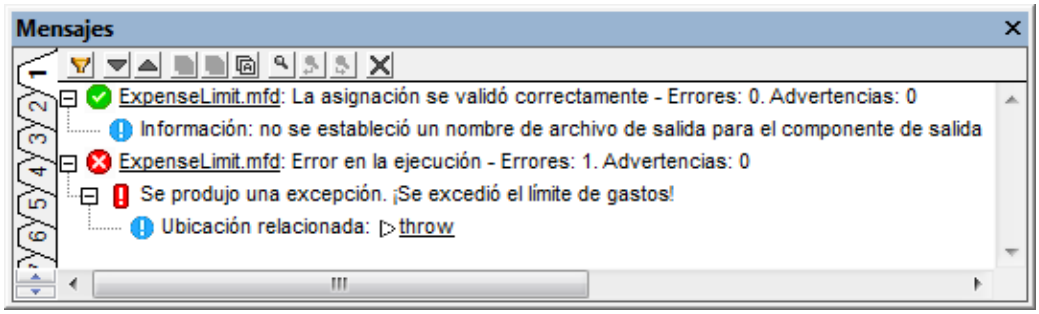
Para añadir una excepción a la asignación:

1. En el menú **Insertar** haga clic en el comando **Excepción** (o haga clic en el icono **Insertar excepción** de la barra de herramientas).
2. Conecte la entrada **throw** de la excepción con la salida **on-true** o con la salida **on-false** de un filtro.
3. Si lo desea, puede conectar la entrada **error-text** de la excepción con otro componente (por lo general una constante) que suministre el texto del error que debe aparecer cuando se emita la excepción.

Nota: las dos salidas del filtro (**on-true** y **on-false**) deben estar conectadas. Concretamente, una de estas dos salidas debe estar conectada con la excepción directamente (sin funciones ni otros componentes de por medio). La otra salida debe estar conectada con el componente de destino, directamente o a través de componentes intermedios.

Cuando la asignación se encuentra con una excepción, aparece una notificación:

- En MapForce aparece un error en la ventana Mensajes, acompañado del texto de la excepción (en este caso "Se excedió el límite de gastos".).



  Si el lenguaje de la asignación es XSLT 2.0 o XQuery, en la ventana Mensajes aparece el error *Error en la ejecución* y se abre el panel correspondiente (*XSLT 2.0* o *XQuery*). La línea del error se resalta en la ventana Mensajes.

- Si ejecuta la asignación con MapForce Server, se devuelve el error "Se produjo una excepción" seguido del texto que se definió en MapForce ("Se excedió el límite de gastos").

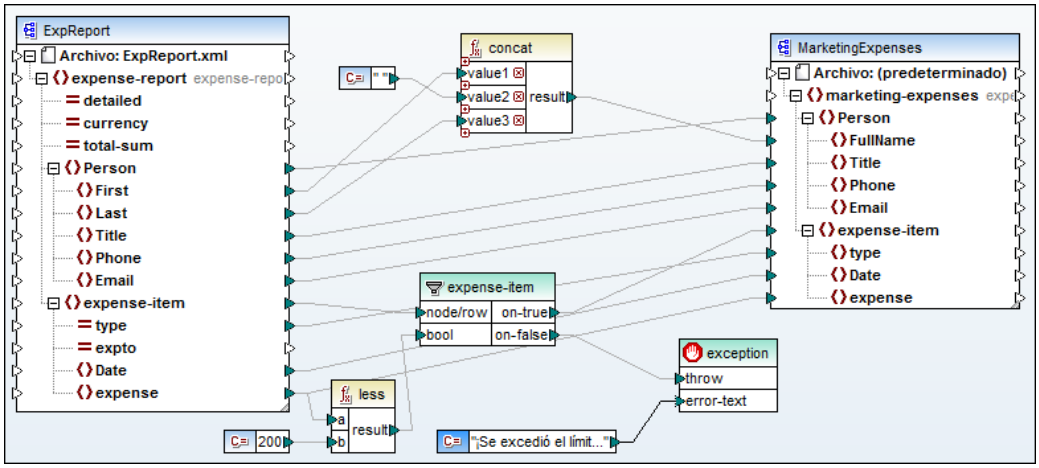
```
c:\codegen\mfx\ExpenseLimit>"C:\Program Files (x86)\Altova\MapForceServer2016\bi
n\MapForceServer.exe" run ExpenseLimit.mfx
Exception was thrown! Expense limit exceeded!
```

- Si ejecuta la asignación a partir del código C#, C++ o Java generado con MapForce, se devuelve el error "EXCEPCIÓN DE USUARIO" seguido del texto que se definió en MapForce ("Se excedió el límite de gastos").

```
C:\codegen\cs\ExpenseLimit\Mapping\bin\Debug>Mapping.exe
Mapping Application
USER EXCEPTION: Expense limit exceeded!
```

5.12.1 Ejemplo: excepción en la condición Greater than

En este ejemplo podemos ver una asignación de datos que emite una excepción cuando se cumple una condición Greater than. Se trata del diseño de asignación de datos **<Documentos>\Altova\MapForce2018\MapForceExamples\ExpenseLimit.mfd**.



Esta asignación de datos emite una excepción si el elemento **expense** de la instancia XML de origen tiene un valor superior a 200. El valor "200" viene dado por una constante. La función **less** se utiliza para comparar los dos valores. Si el valor de **expense** es inferior a 200, entonces se pasa su elemento primario (**expense-item**) al filtro y la asignación no emite ninguna excepción. Si por el contrario el valor de **expense** es superior a 200, se emite una excepción con el texto predefinido "Se excedió el límite de gastos".

Como puede ver en la imagen la excepción se identifica con el icono y se compone de dos elementos: **throw** y **error-text**. El elemento **throw** debe estar conectado con la salida **on-false** o **on-true** de un filtro. El elemento **error-text** está conectado con una constante que suministra el texto de la excepción.

Lo más importante es saber que las dos salidas del filtro deben estar conectadas. De lo contrario no se podrá emitir una excepción. En nuestro ejemplo, la salida **on-false** está conectada con la excepción, mientras que la salida **on-true** está conectada con el componente de destino.

© 2018 Altova GmbH

5.12.2 Ejemplo: excepción cuando un nodo no existe

En este ejemplo podemos ver una asignación de datos que emite una excepción cuando no existe un nodo concreto en el esquema XML de origen. Para simplificar el ejemplo, la asignación usa el mismo esquema XML en el componente de origen y de destino.

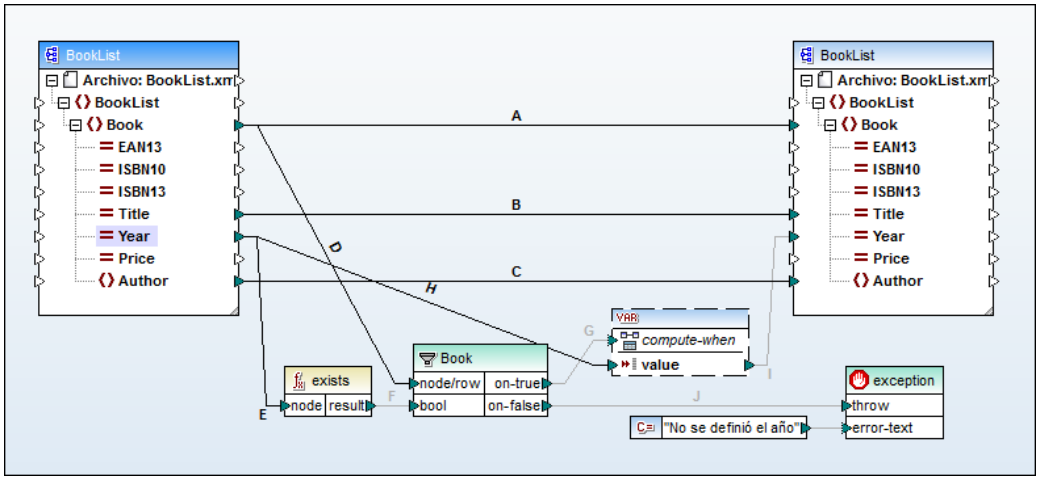
Para añadir el esquema de origen a la asignación:

1. En el menú **Insertar** haga clic en **Archivo o esquema XML...** y navegue hasta el archivo **<Documentos>\Altova\MapForce2018\MapForceExamples\BookList.xsd**.
2. Cuando la aplicación solicite un archivo de instancia, haga clic en **Omitir**.
3. Cuando la aplicación solicite un elemento raíz, seleccione el elemento raíz **BookList**.

Para añadir el esquema de destino siga los mismos pasos. Después, con los comandos correspondientes del menú **Insertar** (o con los botones correspondientes de la barra de herramientas), añada estos componentes:

- Un **Filtro: nodos/filas** (*véase* Filtros y condiciones)
- Una variable intermedia del tipo simple "string"
- Una constante con el texto "No se definió el año"
- Una excepción

Por último, arrastre la función **exists** del panel Bibliotecas hasta el área de asignación y cree las conexiones que aparecen en esta imagen:



Según lo definido en el esquema XML, todos los atributos del elemento Book son opcionales, excepto el título del libro (Title). Por tanto, en una instancia XML válida el atributo Year puede existir o no. El propósito de la asignación es procesar correctamente una instancia XML en la que todos los libros tienen el atributo Year. Si un libro no contiene este atributo, la asignación emitirá una excepción.

Para probar la ejecución de la asignación:

1. Haga doble clic en el título del componente de origen y en el campo *Archivo XML de entrada* navegue hasta este archivo: **<Documentos>\Altova\MapForce2018**

\MapForceExamples\BookList.xml.
2. Haga clic en el panel *Resultados* para ejecutar la asignación.

Para probar la excepción:

1. En el mismo directorio cree una copia del archivo **BookList.xml** llamada **BookListInvalid.xml**.
2. Edite el archivo **BookListInvalid.xml** eliminando el atributo Year de algún elemento Book.
3. Haga doble clic en el título del componente de origen y en el campo *Archivo XML de entrada* navegue hasta el archivo **BookListInvalid.xml**.
4. Haga clic en el panel *Resultados* para ejecutar la asignación.

Vemos ahora cómo funciona la asignación. A diferencia del ejemplo anterior, esta asignación utiliza una variable intermedia (*véase Usar variables*). El propósito de la variable intermedia es almacenar temporalmente el valor del atributo "Year" (cuando exista) en el contexto de cada libro y escribirlo después en el archivo de destino. El resultado Year del componente de origen no se conectó con la entrada Year del componente de destino a propósito (porque el valor Year puede faltar y eso es lo que queremos comprobar).

Por cada libro de la instancia de origen, la **conexión A** crea un libro en la instancia de destino.

Por cada libro del componente de origen, las **conexiones B y C** se encargan de copiar Title y Author en el componente de destino.

La **conexión D** pasa **Book** al filtro de forma condicional. Es decir, el filtro solo recibe los libros que tienen el atributo "Year". Para conseguirlo se usa la conexión **E**, que desencadena la función **`exists`**, encargada de comprobar si existe o no el atributo "Year". La **conexión F** pasa al filtro el resultado de la función (`true` o `false`). Si el resultado es `true`, significa que el atributo "Year" existe el libro se pasa al filtro. Además, la variable se rellena con el valor del año (por medio de la **conexión H**).

> Observe que el filtro no está conectado directamente con la salida **Year** del componente de origen. Si así fuera, el filtro filtraría el año sin más y la excepción no se emitiría.

La **conexión G** se encarga de calcular la variable si la condición del filtro se cumple (si es, es decir, si existe el atributo "Year"). Como dijimos anteriormente, el valor de la variable viene dada por la **conexión H** de la salida **Year** del componente de origen. Posteriormente, la **conexión I** pasa el valor al componente de destino.

La **conexión J** existe porque la excepción debe estar conectada con una salida **on-false** o **on-true** de un filtro. Y, por último, la **conexión K** pasa el texto del error desde la constante hasta el componente de excepción.

© 2018 Altova GmbH

5.13 Redistribuir y serializar cadenas de texto

La redistribución y serialización de cadenas de texto es una técnica avanzada de asignación de datos que le permitirá configurar un componente para redistribuir datos de una cadena o serializar datos en una cadena. Esta técnica es una manera alternativa de leer datos (o escribir datos) en archivos. Los componentes de MapForce que redistribuyen cadenas o serializan datos en cadenas pueden ser muy prácticos en determinados casos, por ejemplo:

- Si necesita insertar estructuras XML, de texto o archivos JSON en campos de BD o en celdas de hojas de cálculo Excel.
- Si necesita convertir fragmentos XML almacenados en campos de BD en archivos XML independientes.
- Si tiene datos heredados en formato texto (p. ej. contenido de longitud fija en un solo campo de BD) y desea pasarlos a una estructura basada en campos (con la aplicación FlexText, por ejemplo)

La redistribución y serialización de cadenas de texto es compatible con este tipo de componentes de MapForce:

- Componentes de texto (texto CSV, texto de longitud fija, EDI y plantillas de MapForce FlexText)
- Archivos de esquema JSON
- Archivos de esquema XML

La redistribución y serialización de cadenas de texto es compatible con estos lenguajes de destino de MapForce:

Lenguaje	Lectura	Escritura
BUILT-IN (vista previa de la transformación)	Sí	Sí
BUILT-IN (ejecución de archivo de ejecución de MapForce Server)	Sí	Sí

Esta sección incluye varios apartados:

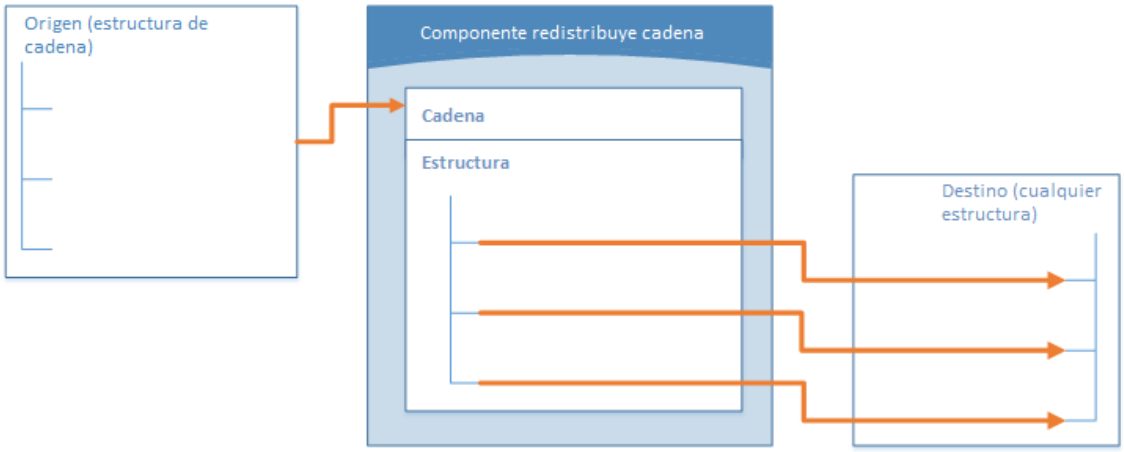
- Información sobre el componente de análisis/serialización
- Ejemplo: redistribuir cadena (de texto de longitud fija a Excel)
- Ejemplo: serializar datos a una cadena (de XML a BD)

5.13.1 Información sobre el componente de análisis/serialización

En MapForce un componente de redistribución/serialización es un componente híbrido, que no es componente de origen ni de destino. Debido al papel que desempeñan en el diseño de asignación, este tipo de componentes debe colocarse entre componentes de origen y de destino.

Puede utilizar un componente de redistribución/serialización de cadenas cuando necesite pasar a otro formato una cadena de texto que tiene una estructura (p. ej. datos XML guardados como

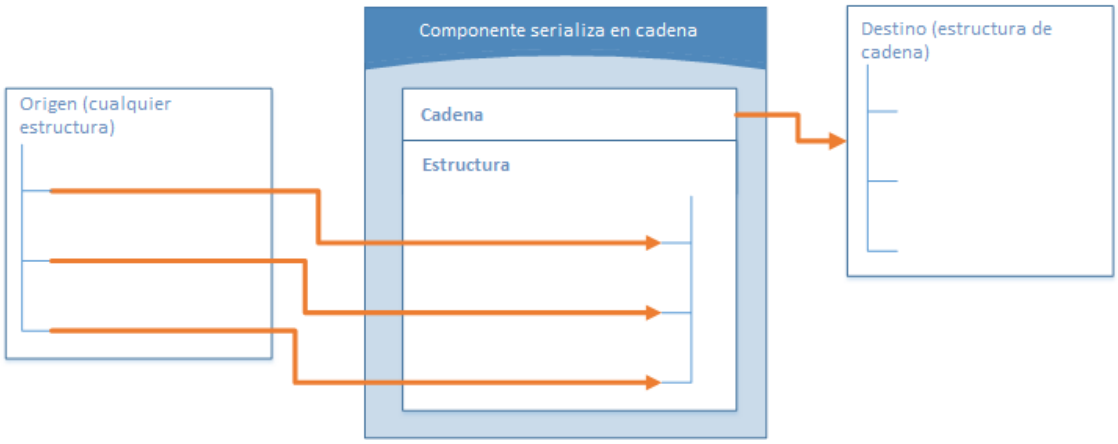
cadena de texto en una BD). La redistribución de la cadena de origen a través del componente de redistribución/serialización implica que la cadena de origen se convierte en una estructura de MapForce y, por tanto, será posible acceder a los elementos y atributos del XML de origen guardado como cadena.



Componente de redistribución de cadena genérico

En el diagrama anterior puede verse la estructura típica de un componente de MapForce que redistribuye una cadena. Observe que el componente de redistribución/serialización de cadenas está situado entre el componente de origen y el de destino. Lo que hace este componente es aceptar como entrada una estructura de cadena (por medio de un conector de MapForce que está conectado su nodo de nivel superior **Cadena**). La estructura de salida puede ser cualquier destino de datos compatible con MapForce. Puede ver un ejemplo en Ejemplo: redistribuir cadena (de texto de longitud fija a Excel).

Cuando se serializan datos de un componente a una cadena ocurre lo contrario. Toda la estructura del componente de MapForce se convierte en una estructura de cadena que se puede manipular. Por ejemplo, puede escribir un archivo XML (o fragmento XML) en un campo de BD o en una sola celda de una hoja de cálculo Excel.

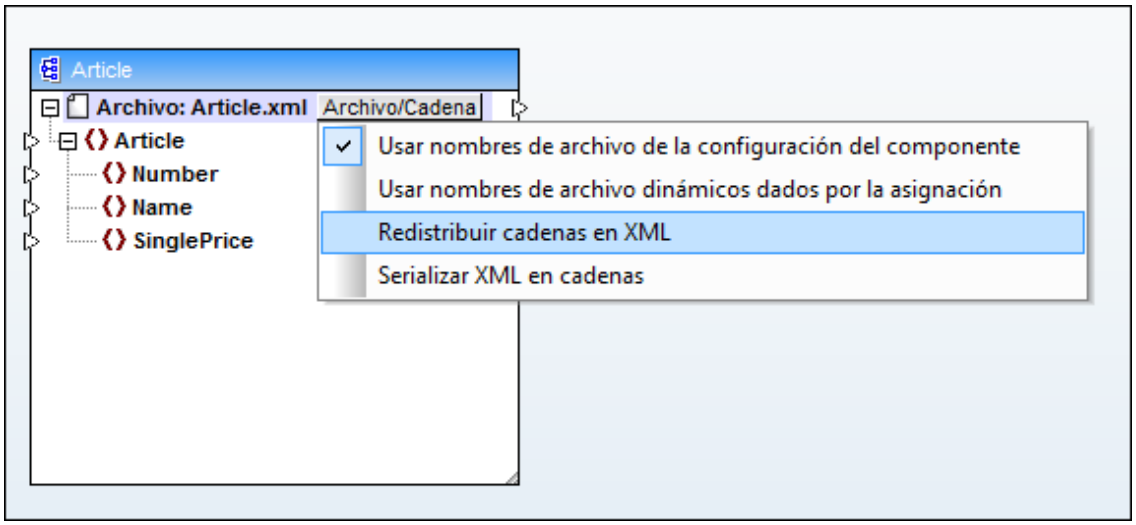


Componente de serialización de cadena genérico

 © 2018 Altova GmbH

En el diagrama anterior puede verse un componente de serialización de cadena genérico. Lo que hace este componente es aceptar como entrada cualquier origen de datos compatible con MapForce (por medio de conectores estándar de MapForce). La estructura de salida es una cadena que se puede pasar dibujando un conector entre el nodo de nivel superior **Cadena** del componente y un elemento del componente de destino (p. ej. una celda de una hoja de cálculo). Puede ver un ejemplo en Ejemplo: serializar datos a una cadena (de XML a BD).

Para configurar un componente para la redistribución o serialización de cadenas basta con hacer clic en el botón Archivo/Cadena situado junto al nodo raíz de un componente y seleccionar la opción correspondiente.



Cambiar el modo del componente

Nota: un componente de redistribución/serialización de cadenas no puede leer datos de una cadena y escribirlos en otra simultáneamente. Por tanto, el nodo raíz puede tener un conector de entrada o uno de salida (pero no ambos). MapForce genera un error si se intenta usar el mismo componente para ambas operaciones.

Cuando un componente se configura para la redistribución o serialización de cadenas, su aspecto cambia:

- El título del componente aparece con el prefijo **parse** (redistribución) o **serialize** (serialización).
- El color de fondo de la barra de título es amarillo (igual que la barra de título de las funciones).
- El nodo de nivel superior incluye el prefijo `Cadena:` y aparece junto al icono .
- Si el componente redistribuye una cadena, el conector de salida del nodo raíz no es relevante y, por tanto, no está disponible.
- Si el componente serializa datos en una cadena, el conector de entrada del nodo raíz no es relevante y, por tanto, no está disponible.

Cuando un componente está en modo de redistribución/serialización de cadenas, su configuración se puede cambiar como si se tratase de un componente basado en un archivo (*véase* *Cambiar configuración de los componentes*). Recuerde que no todas las opciones de configuración están disponibles si el componente está en modo de redistribución/serialización de

cadenas.

5.13.2 Ejemplo: redistribuir cadena (de texto de longitud fija a Excel)

Este ejemplo explica paso a paso cómo crear un diseño de asignación que redistribuye datos de cadena. El ejemplo viene con un archivo de muestra. Si prefiere consultar el archivo de muestra antes de empezar con el ejemplo, puede hacerlo en esta ubicación: **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial\ParseString.mfd**.

Imaginemos que tenemos datos de texto heredados y guardados en un solo campo de BD. Los datos de texto son una lista de empleados, almacenados en la columna RESOURCE y con formato de campo de longitud fija:

ID	RESOURCE					TYPE
1	P00001Callaby	Vernon	582Office Manager	v.callaby@nanonull.com		Employees
	P00002Further	Frank	471Accounts Receivable	f.further@nanonull.com		
	P00003Matise	Loby	963Accounting Manager	l.matise@nanonull.com		
	P00004Firstbread	Joe	621Marketing Manager Europe	j.firstbread@nanonull.com		
	P00005Sanna	Susi	753Art Director	s.sanna@nanonull.com		
	P00006Landis	Fred	951Program Manager	f.landis@nanonull.com		
	P00007Butler	Michelle	654Software Engineer	m.landis@nanonull.com		
	P00008Little	Ted	852Software Engineer	t.little@nanonull.com		
	P00009Way	Ann	951Technical Writer	a.way@nanonull.com		
	P00010Gardner	Liz	753Software Engineer	l.gardner@nanonull.com		
	P00011Smith	Paul	334Software Engineer	p.smith@nanonull.com		

Como los datos de texto están guardados en un solo campo de BD, no es fácil acceder a los registros de cada empleado y manipularlos. Es decir, no es fácil añadir ni quitar empleados en la lista ni ordenar los datos. En este ejemplo, nuestro objetivo es extraer los datos de texto del campo de BD RESOURCE y dividirlos en una estructura para poder procesar todos los registros con facilidad.

Esto se consigue con un componente de redistribución/serialización de cadenas. Lo primero que hará este componente será tomar los datos de texto como entradas. Después los analizará y los convertirá en una estructura. Por último, escribirá la estructura en un formato de destino. En este ejemplo, el formato de destino es una hoja de cálculo Excel (pero recuerde que podría ser cualquier formato de salida compatible con MapForce).

En resumen, la asignación de datos de este ejemplo convertirá el contenido del campo de BD RESOURCE en una tabla. Una vez ejecutada la asignación, cada fila de la tabla corresponderá a un empleado y cada columna corresponderá a uno de estos campos de longitud fija: ID, Apellido, Nombre, Extensión, Puesto, Correo electrónico.

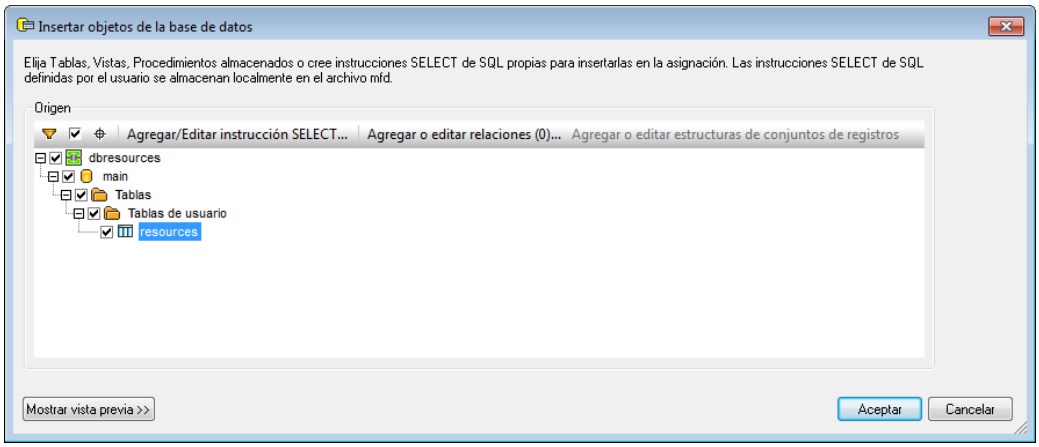
© 2018 Altova GmbH

	A	B	C	D	E	F	G
1	P00001	Callaby	Vernon	582	Office Manager	v.callaby@nanonull.com	
2	P00002	Further	Frank	471	Accounts Receivable	f.further@nanonull.com	
3	P00003	Matise	Loby	963	Accounting Manager	l.matise@nanonull.com	
4	P00004	Firstbread	Joe	621	Marketing Manager Europe	j.firstbread@nanonull.com	
5	P00005	Sanna	Susi	753	Art Director	s.sanna@nanonull.com	
6	P00006	Landis	Fred	951	Program Manager	f.landis@nanonull.com	
7	P00007	Butler	Michelle	654	Software Engineer	m.landis@nanonull.com	
8	P00008	Little	Ted	852	Software Engineer	t.little@nanonull.com	
9	P00009	Way	Ann	951	Technical Writer	a.way@nanonull.com	
10	P00010	Gardner	Liz	753	Software Engineer	l.gardner@nanonull.com	
11	P00011	Smith	Paul	334	Software Engineer	p.smith@nanonull.com	
12							
13							

Resultado esperado de la redistribución de cadenas

Ahora veamos las instrucciones paso a paso:

1. Añada la base de datos de origen al área de asignación. La base de datos es un archivo de BD SQLite independiente ubicado en esta ruta de acceso: **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial\\dbresources.db**. (Para añadir el componente de base de datos haga clic en el comando **Insertar | Base de datos** y consulte el apartado Conectarse a una base de datos).
2. Cuando MapForce solicite los objetos de la BD, seleccione la tabla **resources**.

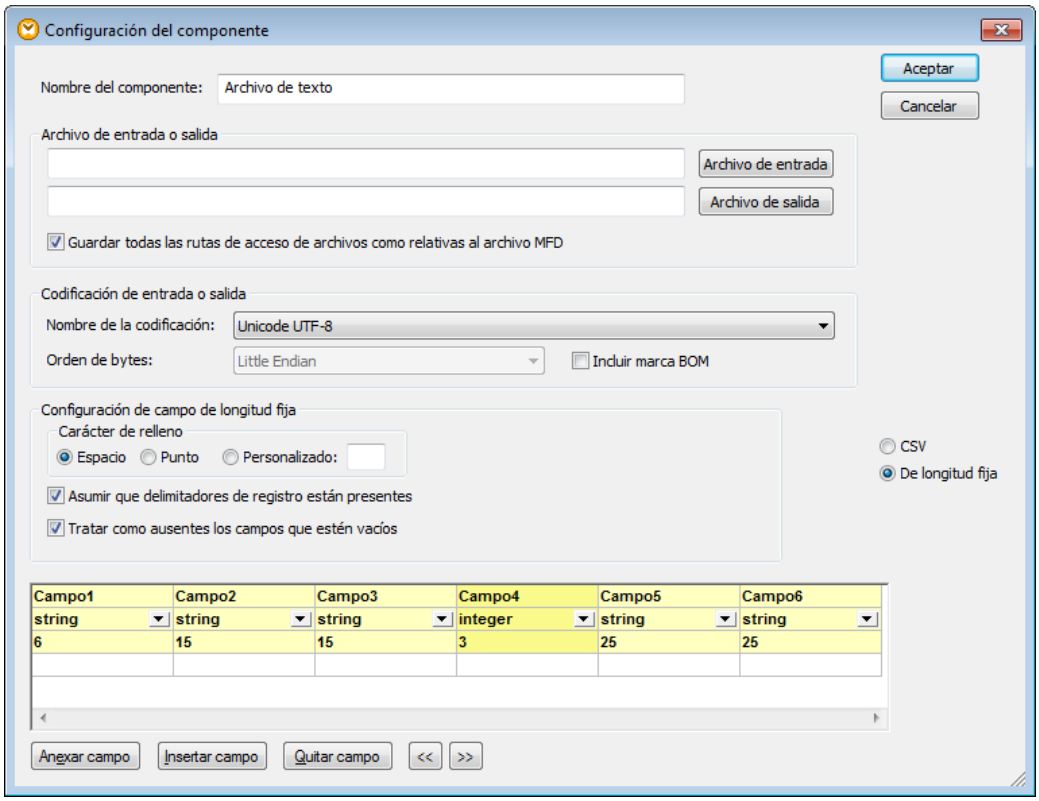


3. Añada un componente de texto al área de asignación (con el comando de menú **Insertar | Archivo de texto**). Como los datos de origen son texto delimitado por campos, elija la opción **Utilizar procesamiento básico**.
4. Configure la estructura del componente de texto para asignarla a la estructura de los datos de texto de origen almacenados en la BD. Como puede ver, en este ejemplo, el texto de origen está compuesto por 6 campos de longitud fija de tamaño fijo:

Tamaño en caracteres	Campo
6	ID

Tamaño en caracteres	Campo
15	Surname
15	First Name
3	Extension
25	Job Title
25	Email

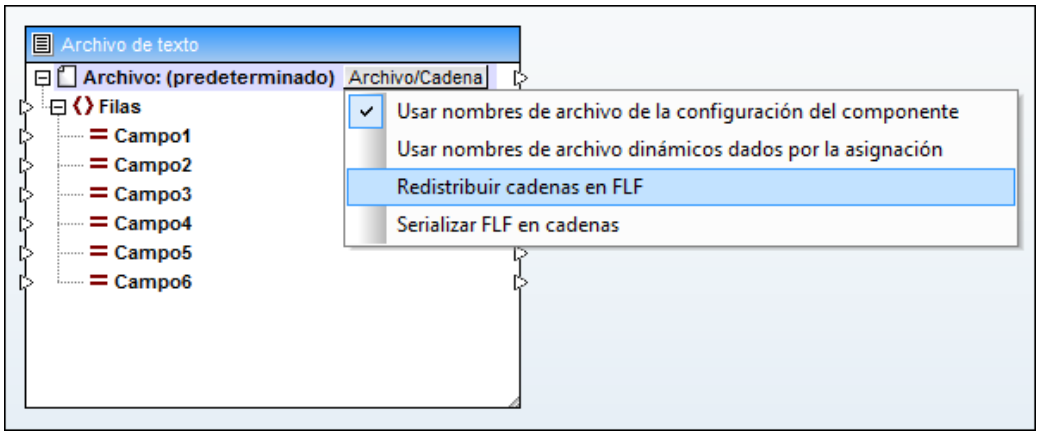
Primero declaramos el componente Texto como **De longitud fija** y le añadimos seis campos que se correspondan con la tabla anterior. (Para abrir el cuadro de diálogo "Configuración del componente" haga clic con el botón derecho en el componente y elija el comando **Propiedades** en el menú contextual).



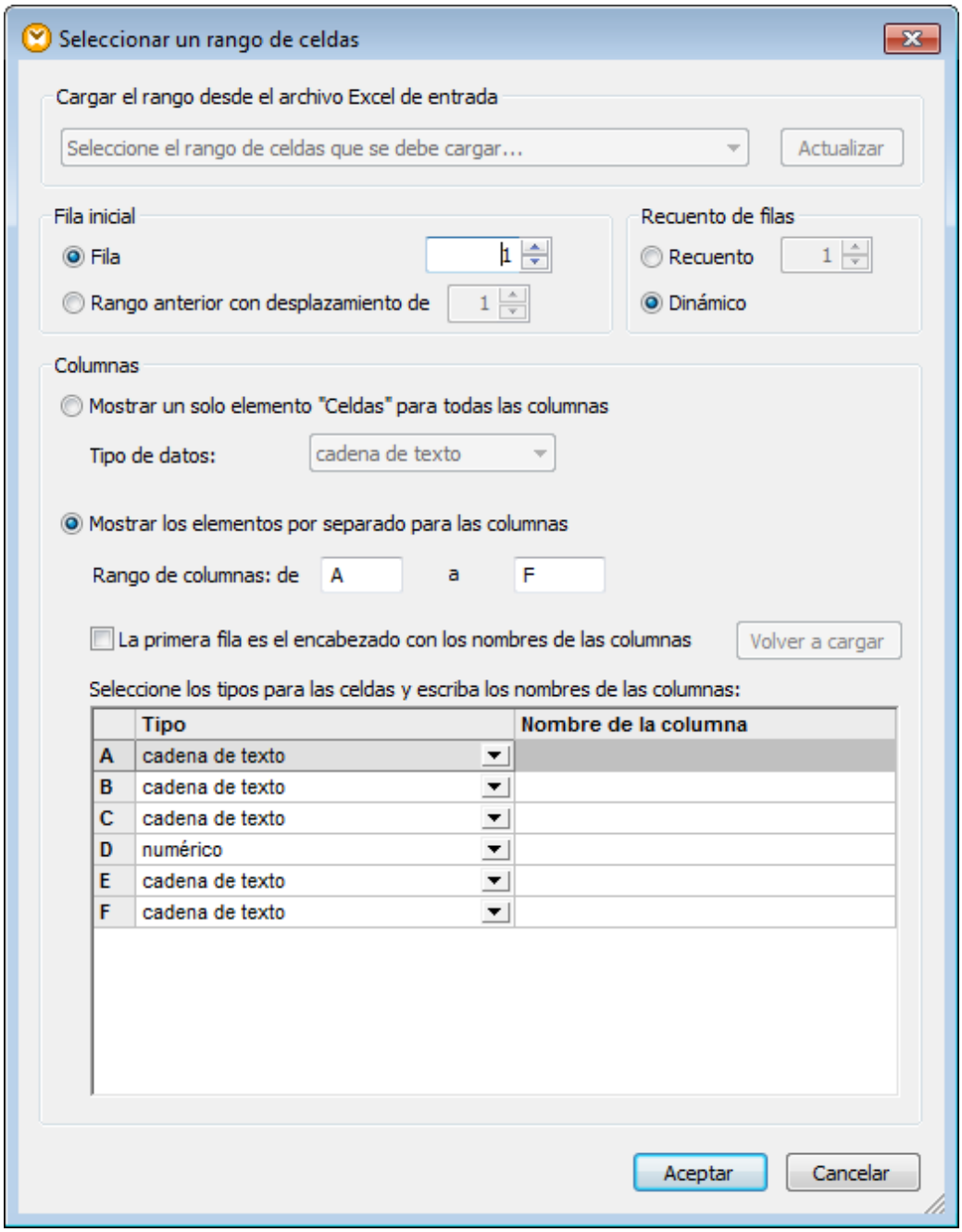
Observe que el **Campo4** utiliza el tipo de datos **integer**. Aunque declarar el **Campo4** como tipo numérico es opcional en este ejemplo, así nos aseguramos de que la extensión de teléfono (**Campo4**) que se extrae del texto de origen sea un valor numérico válido.

5. Haga clic en el botón Archivo/Cadena y seleccione el comando **Redistribuir cadenas en FLF** del menú contextual. Esto sirve para indicar que este componente redistribuirá una cadena al formato FLF (de longitud fija).

© 2018 Altova GmbH

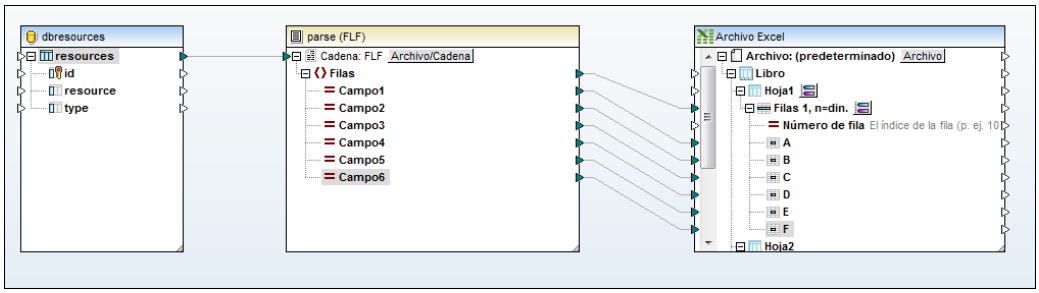


6. Ahora añada el componente Excel 2007+ de destino al área de asignación. Cuando MapForce solicite un archivo de muestra, haga clic en **Omitir**. (Puede añadir el componente Excel con el comando de menú **Insertar | Excel 2007+** y consultar el apartado Agregar archivos Excel 2007+ como componentes de la asignación).
7. Haga clic en el botón  situado junto al nodo **Filas 1, n=dyn** y configure el componente Excel para que escriba una fila por cada campo de texto, empezando por la primera fila (ver configuración en la imagen siguiente). Para más información sobre componentes Excel 2007+ y su configuración consulte el apartado Información sobre componentes Excel 2007+.



8. Dibuje las conexiones entre componentes que aparecen a continuación:

 © 2018 Altova GmbH



En el lado izquierdo de la asignación, el contenido de la columna de BD resource pasa de ser (se redistribuye) un valor de cadena a una estructura MapForce. En el lado derecho de la asignación, los elementos del componente **parse (FLF)** se conectan con las diferentes columnas Excel para dividir la cadena de origen en celdas separadas.

El diseño de asignación de datos de MapForce ya está terminado: los datos de cadena se redistribuyen y se crea una estructura a partir de ellos. Si ahora hacemos clic en el panel *Resultados*, los datos de texto heredados pasan a ser filas y columnas de la hoja de cálculo Excel.

5.13.3 Ejemplo: serializar datos a una cadena (de XML a BD)

Este ejemplo explica paso a paso cómo crear un diseño de asignación que serializa datos en una cadena de texto. El ejemplo viene acompañado de un archivo de muestra. Si desea consultar el archivo de muestra antes de empezar con el ejemplo, puede hacerlo en esta ubicación: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\SerializeToString.mfd**.

Imaginemos que tenemos un archivo XML (y su correspondiente esquema) compuesto por varios elementos `<Person>`. Cada elemento `<Person>` describe el nombre de una persona, su apellido, su puesto, su extensión de teléfono y su dirección de correo electrónico:

```
<Person>
      <First>Joe</First>
      <Last>Firstbread</Last>
      <Title>Marketing Manager Europe</Title>
      <PhoneExt>621</PhoneExt>
      <Email>j.firstbread@nanonull.com</Email>
</Person>
```

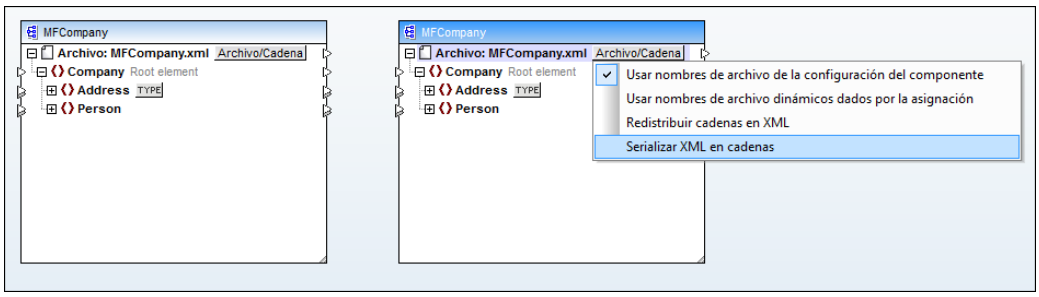
Nuestro objetivo es extraer cada elemento `<Person>` del archivo XML e insertarlo literalmente (incluidas las etiquetas XML) en un nuevo registro de BD de la tabla `PEOPLE` de una base de datos SQLite. La tabla `PEOPLE` solamente contiene dos columnas: `ID` y `PERSON`. Esta sería su definición completa:

```
CREATE TABLE PEOPLE (ID INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL, PERSON
TEXT);
```

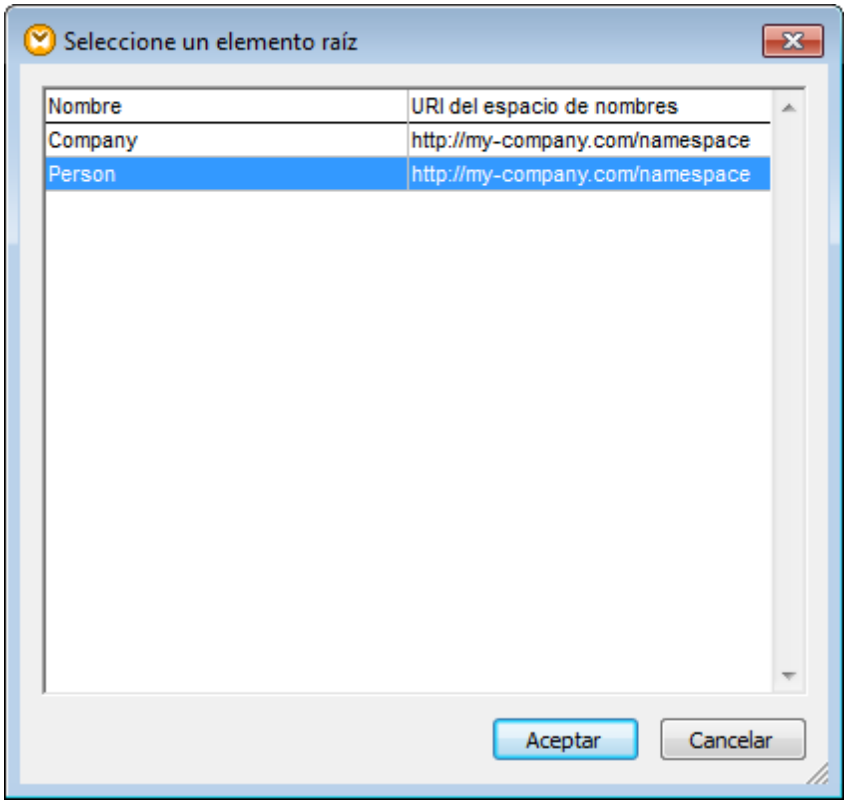
Una vez ejecutada la asignación, el resultado esperado es que la tabla `PEOPLE` tenga tantas filas como elementos `<Person>` contenía el archivo XML.

Para conseguir este objetivo debe seguir estos pasos:

1. Añada el componente XML de origen (con el comando de menú **Insertar | Archivo o esquema XML**). El archivo de muestra está en esta ubicación: **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial\MFCompany.xml**.
2. Cree una copia del componente XML (mediante copia y pega).
3. En la segunda copia del componente XML haga clic en el botón Archivo/Cadena y seleccione **Serializar XML en cadenas** en el menú contextual.



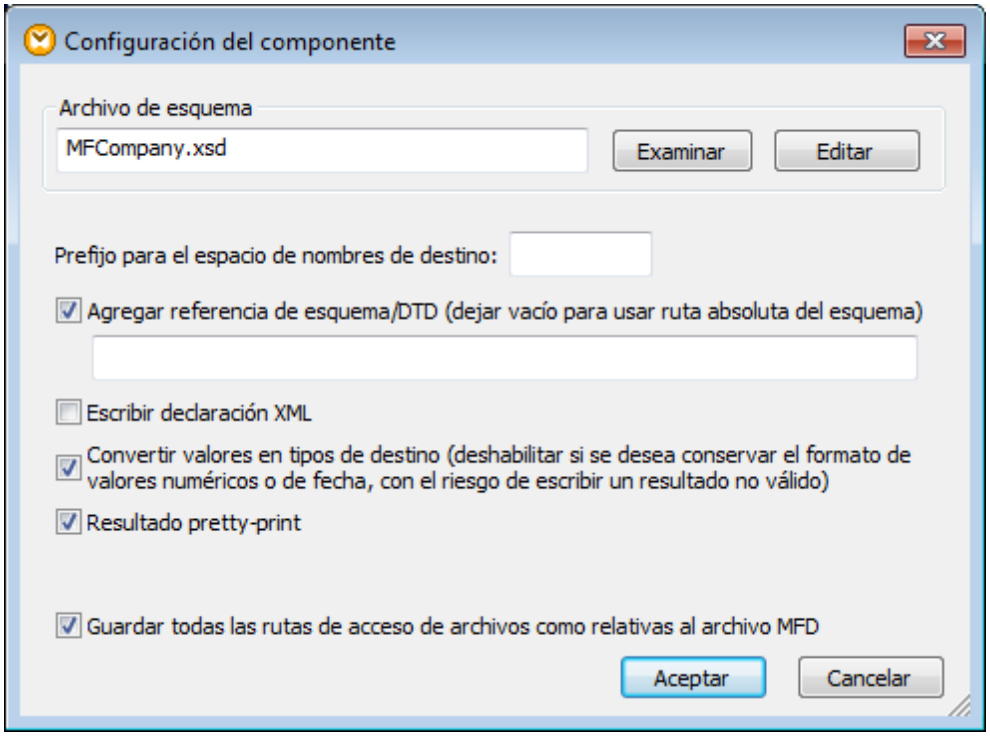
4. Haga clic con el botón derecho en la copia del componente XML y seleccione **Cambiar de elemento raíz** en el menú contextual. Ahora cambie el elemento raíz por `<Person>`.



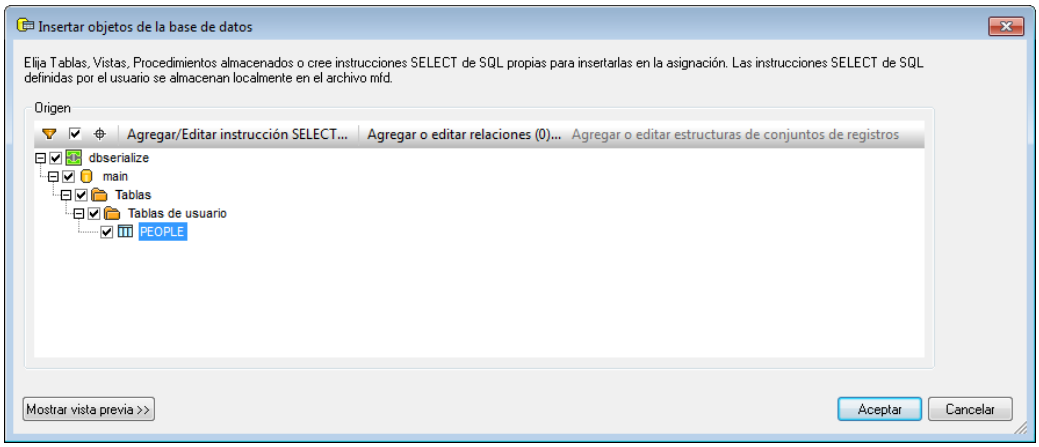
Por lo general, el elemento raíz puede cambiarse por cualquier elemento que tenga una declaración global (no local) en el esquema XML. Los elementos que no estén definidos globalmente en el esquema no aparecerán en el cuadro de diálogo "Seleccione un elemento raíz".

© 2018 Altova GmbH

5. Haga doble clic en el componente y deshabilite la casilla *Escribir declaración XML*. Esto sirve para impedir que la declaración XML se escriba por cada elemento `<Person>`.

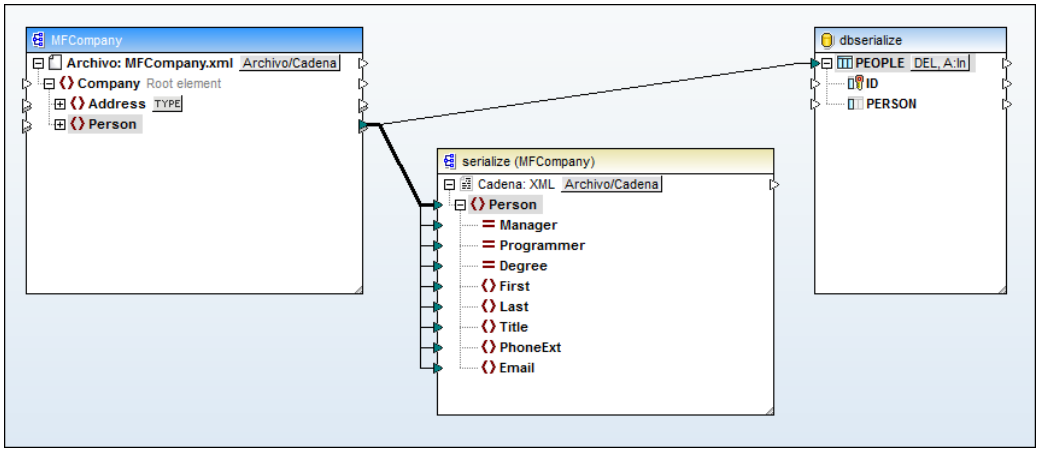


6. Añada el componente de base de datos SQLite de destino al área de asignación (la base de datos está en esta ubicación: **<Documentos>\Altova\MapForce2018 \MapForceExamples\Tutorial\\dbserialize.db**). Para añadir el componente de base de datos seleccione el comando de menú **Insertar | Base de datos** (*véase también* *Conectarse a una base de datos*). Cuando solicite un objeto de base de datos, seleccione la tabla `PEOPLE`.

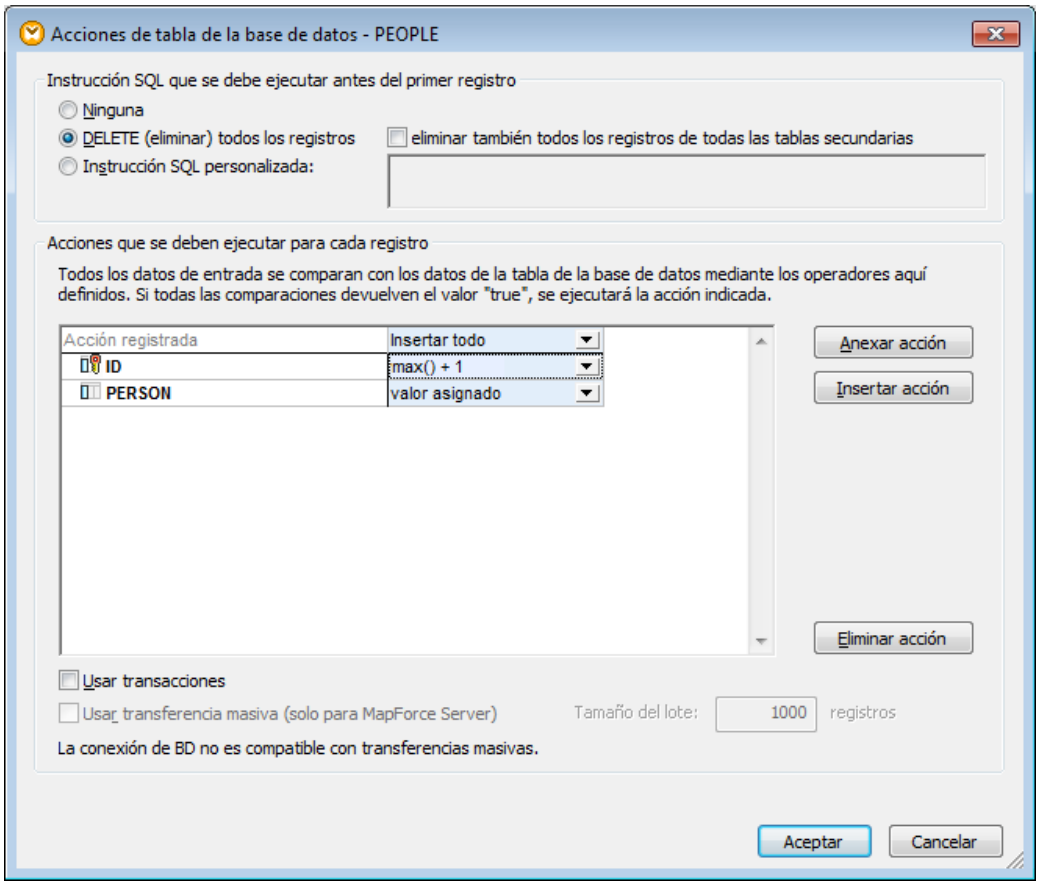


7. Conecte los componentes tal y como puede ver en la imagen siguiente. En el lado izquierdo de la asignación, el elemento `<Person>` se asigna al componente de serialización. En el lado derecho el valor de cadena serializado se inserta en la columna `PERSON` de la tabla de BD `PEOPLE`. Por último, el conector que une `<Person>` con la tabla

`PEOPLE` indica que se debe crear un registro nuevo por cada elemento `<Person>` que se detecte.



8. Haga clic en el botón **A:In** del componente de base de datos y configúrelo para que MapForce lleve a cabo estas acciones cada vez que la transformación se ejecute:
   a. Eliminar todos los registros de la tabla;
   b. Incrementar el valor de ID en 1.



Observe que está seleccionada la acción **max()+1** para la columna ID. Esto indica

 © 2018 Altova GmbH

que MapForce debe analizar cuál es el valor de ID máximo que ya existe en la BD y que debe insertar el entero siguiente, incrementado en 1.

Ya ha terminado de crear un diseño de asignación que serializa datos a una cadena. Si hace clic en el panel *Resultados*, la vista previa de la consulta SQL indica que en la BD se insertarán registros distintos para cada elemento `<Person>` del archivo XML.

5.14 Asignar nombres de nodos

En la mayoría de las asignaciones de datos de MapForce el objetivo es leer *valores* de un componente de origen y escribir *valores* en un componente de destino. Sin embargo, en algunos casos el objetivo no es solo acceder a los valores de los nodos de origen, sino al nombre de los nodos. Por ejemplo, puede tratarse de una asignación que lea el nombre de los elementos o atributos (y no sus valores) de un archivo XML de origen y convertir esos nombres en valores de elemento o atributo en el XML de destino.

Imaginemos que tenemos un archivo XML que contiene una lista de productos. Cada producto tiene este formato:

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

Nuestro objetivo es convertir información sobre cada producto en pares nombre/valor. Por ejemplo:

```
<product>
   <attribute name="id" value="1" />
   <attribute name="color" value="red" />
   <attribute name="size" value="10" />
</product>
```

En este caso, necesitaremos acceder al nombre de los nodos de forma *dinámica* para realizar conversiones de datos.

Nota: la transformación del ejemplo también puede conseguirse con las funciones **node-name** y **static-node-name** de la biblioteca **core**. No obstante, en nuestro ejemplo necesitamos conocer exactamente los nombres de elemento del componente de origen y necesitamos conectar cada uno de los elementos manualmente con el componente de destino. Además, puede que las funciones mencionadas no sean suficiente para filtrar o agrupar nodos por nombre o para manipular el tipo de datos de los nodos.

No solo puede acceder al nombre de los nodos de forma dinámica cuando necesite leer nombres de nodo, sino también cuando necesite escribirlos. En una asignación normal y corriente el nombre de los atributos o elementos del componente de destino siempre se conoce antes de que se ejecute la asignación. Estos nombres de atributo o elemento vienen del esquema subyacente del componente. Sin embargo, con los nombres de nodo dinámicos podrá crear atributos o elementos nuevos cuyo nombre no se conoce antes de que se ejecute la asignación. Concretamente, el nombre del atributo o elemento viene dado por la asignación propiamente dicha (por el componente de origen).

Para poder acceder a elementos o atributos secundarios de forma dinámica el nodo debe tener elementos o atributos secundarios y el nodo principal no puede ser el nodo XML raíz.

© 2018 Altova GmbH

Puede trabajar con nombres de nodo dinámicos si en la asignación de datos participan componentes de este tipo:

- XML
- CSV/FLF*

* Solo compatible con MapForce Professional or Enterprise Edition.

Nota: si se trata de componentes CSV/FLF el acceso dinámico implica poder acceder a los *campos* en lugar de a los *nodos* porque las estructuras CSV/FLF no tienen nodos.

Cuando el destino de la asignación es un archivo CSV o FLF, los campos deben estar definidos en la configuración del componente (y el nombre, el orden y el número de los campos de destino no se puede cambiar). Al contrario de los archivos XML, el formato de los archivos de texto es fijo así que solamente se puede manipular el valor de los campos (no se puede modificar el nombre, el número ni el orden de los campos).

Puede trabajar con nombres de nodo dinámicos si selecciona estos lenguajes de transformación: motor integrado BUILT-IN*, XSLT2, XQuery*, C#*, C++*, Java*.

* Solo compatible con MapForce Professional o Enterprise Edition.

Para más información consulte el apartado Obtener acceso al nombre de los nodos o el ejemplo del apartado Ejemplo: asignar nombres de elemento a valores de atributo.

5.14.1 Obtener acceso al nombre de los nodos

Cuando un nodo de un componente XML (o un campo de un componente CSV/FLF) tiene nodos secundarios, podemos obtener tanto el nombre como el valor de cada nodo secundario en la asignación directamente. Esta técnica se denomina *nombres de nodo dinámicos.* El adjetivo *dinámicos* hace referencia al hecho de que el procesamiento se realiza *sobre la marcha,* durante la ejecución de la asignación, y no se basa en la información estática del esquema que se conoce antes de que se ejecute la asignación. En este apartado explicamos cómo habilitar el acceso dinámico a nombres de nodo y cómo utilizarlo.

Cuando se leen datos de un componente de origen, los *nombres de nodo dinámicos* permiten:

- Obtener una lista de nodos o atributos secundarios de un nodo en forma de secuencia. En MapForce una *secuencia* es una lista de cero o más elementos que se pueden conectar con un componente de destino y que crea tantos elementos en el componente de destino como elementos existen en el componente de origen. Por ejemplo, si un nodo tiene cinco atributos en el componente de origen, puede crear cinco elementos nuevos en el componente de destino (cada uno de ellos corresponderá a un atributo).
- Leer no solo los valores de los nodos secundarios (como hacen las asignaciones normales y corrientes) sino también sus nombres.

Cuando se escriben datos en un componente de destino, los *nombres de nodo dinámicos* permiten:

- Crear nodos nuevos usando nombres que vienen dados por la asignación (nombres *dinámicos*), en lugar de los nombres que vienen dados por la configuración del componente (nombres *estáticos*).

A continuación veremos un ejemplo basándonos en el esquema XML **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Products.xsd**. Este esquema viene acompañado de un documento de instancia llamado **Products.xml**. Para agregar tanto el esquema como el archivo de instancia al área de asignación seleccione **Insertar | Archivo o esquema XML** y navegue hasta el archivo **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Products.xml**. Cuando la aplicación solicite un elemento raíz, seleccione `products`.

Para habilitar los nombres de nodo dinámicos para el nodo `product` haga clic con el botón derecho en ese nodo y seleccione uno de estos comandos en el menú contextual:

- **Mostrar atributos con nombre dinámico** si desea tener acceso a los atributos del nodo o
- **Mostrar elementos secundarios con nombre dinámico** si desea tener acceso a los elementos secundarios del nodo.

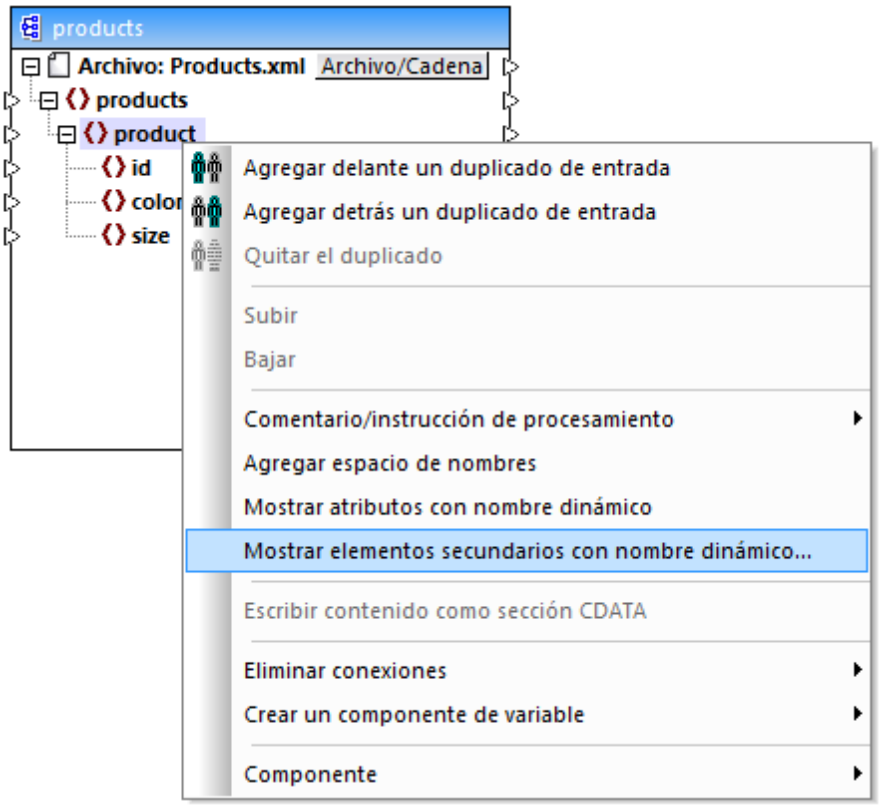


Fig. 1: habilitar nombres de nodo dinámicos (para elementos secundarios)

Nota: los comandos que aparecen en la imagen anterior solamente están disponibles cuando se trata de nodos con nodos secundarios. Además, estos comandos no están disponibles cuando se trata del nodo raíz.

© 2018 Altova GmbH

Cuando se cambia un nodo al modo dinámico, aparece un cuadro de diálogo como el de la imagen siguiente. Siguiendo con nuestro ejemplo, ahora configuramos las opciones que puede ver en la imagen (estas opciones se definen en detalle en el apartado Obtener acceso a determinado tipo de nodos).

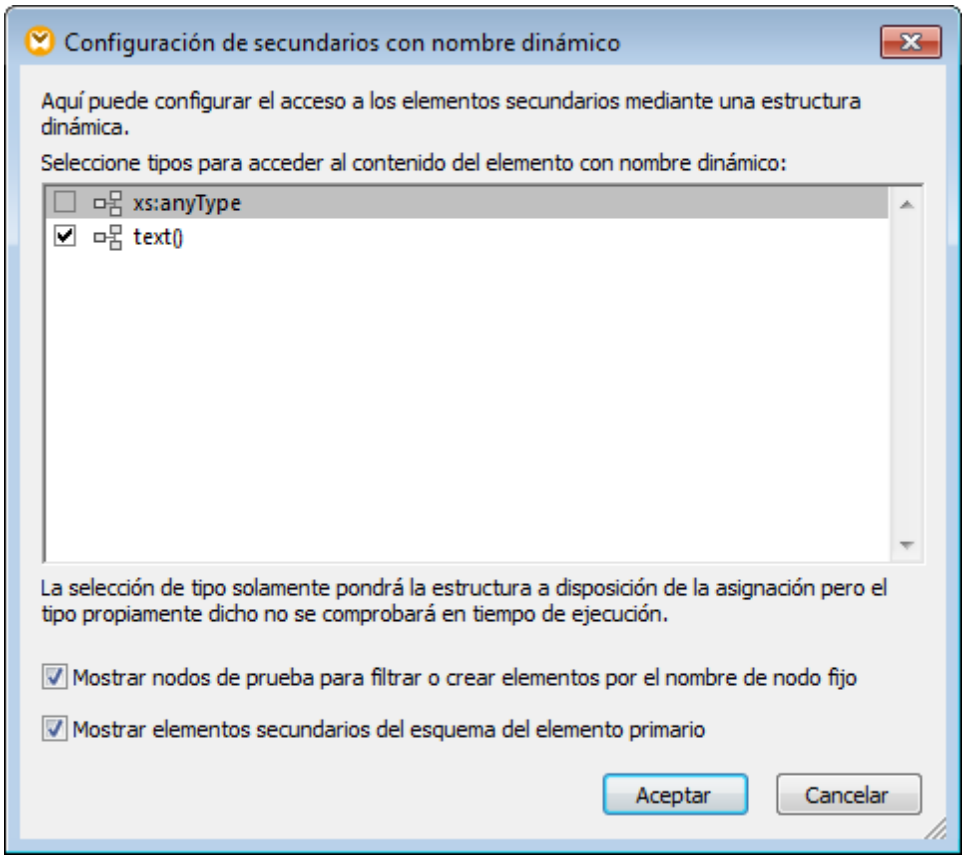


Fig. 2: cuadro de diálogo "Configuración de secundarios con nombre dinámico"

A continuación explicamos lo que hace el componente cuando se habilitan los nombres de nodo dinámicos para el nodo `product`. Observe que el aspecto del componente cambia significativamente.

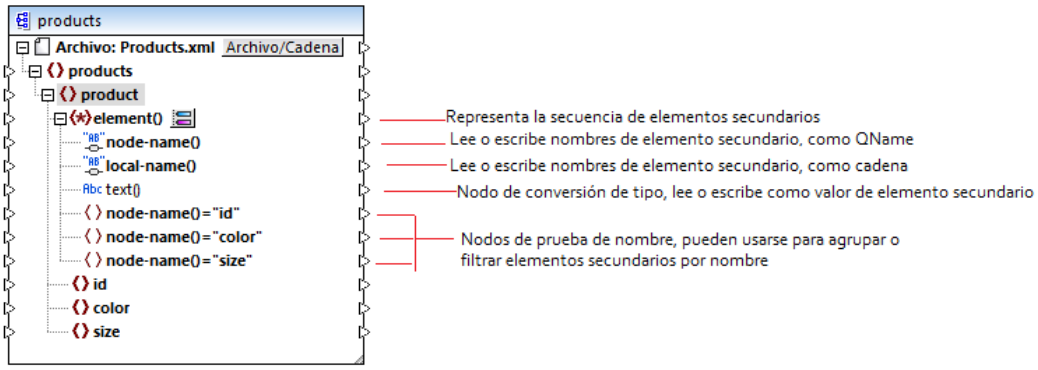


Fig. 3: nombres de nodo dinámicos habilitados (para elementos secundarios)

Para restaurar el modo estándar del componente haga clic con el botón derecho en el nodo `product` y desactive el comando **Mostrar elementos secundarios con nombre dinámico** en el menú contextual.

En la imagen siguiente podemos ver el aspecto que tiene el componente cuando se habilita el acceso dinámico a los atributos de un nodo (haciendo clic con el botón secundario en el elemento `product` y seleccionando el comando **Mostrar atributos con nombre dinámico** en el menú contextual).

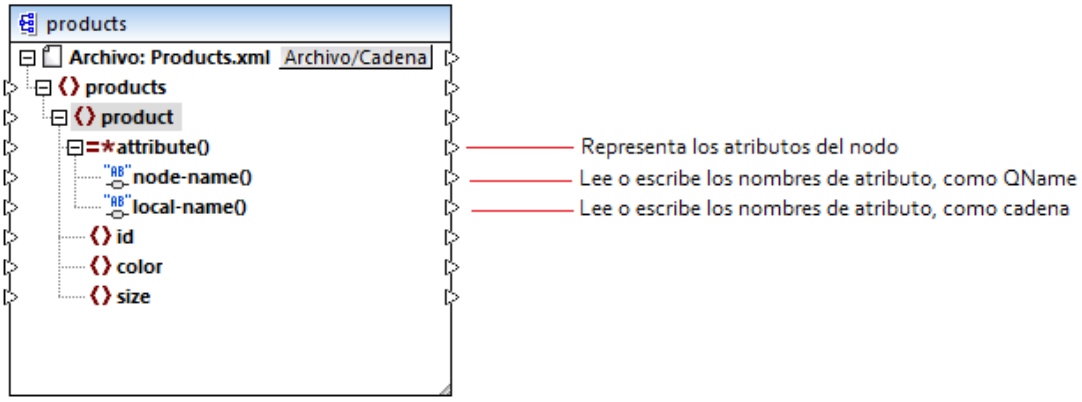


Fig. 4: nombres de nodo dinámicos habilitados (para atributos)

Para restaurar el modo estándar del componente haga clic con el botón derecho en el nodo `product` y desactive el comando **Mostrar atributos con nombre dinámico** en el menú contextual.

Como puede ver en las dos imágenes anteriores, el aspecto del componente cambia cuando se habilita el modo *nombres de nodo dinámicos* en uno de los nodos (en nuestro ejemplo se trata del nodo `product`). El nuevo aspecto del componente ofrece varias opciones nuevas porque permite:

- Leer o escribir una lista con todos los elementos secundarios o atributos de un nodo (que vienen dados por el elemento `element()` o `attribute()` respectivamente).

 © 2018 Altova GmbH

- Leer o escribir el nombre de cada elemento secundario o atributo (que viene dado por el elemento `node-name()` o `local-name()` respectivamente).
- Si se trata de elementos, leer o escribir el valor de cada elemento secundario, en el tipo de datos que corresponda. Este valor viene dado por el nodo de conversión de tipo (el elemento `text()`). Recuerde que solamente los elementos pueden tener nodos de conversión de tipo. Los atributos siempre tienen el tipo "string".
- Agrupar o filtrar elementos secundarios por nombre (*véase* *Ejemplo: agrupar y filtrar nodos según su nombre*).

A continuación explicamos con qué tipos de nodos puede trabajar cuando use el modo *nombres de nodo dinámicos.*

element()

Este nodo diferentes comportamientos, dependiendo de si está en el componente de origen o en el componente de destino. Si está en el componente de origen, aporta los elementos secundarios del nodo en forma de secuencia. En la Fig. 3 (ver más arriba) el elemento `element()` ofrece una lista (secuencia) con todos los elementos secundarios de `product`. Por ejemplo, la secuencia que se crea a partir de este XML contendría tres elementos (porque `product` tiene tres elementos secundarios):

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

Observe que el nombre y el tipo de cada elemento de la secuencia es el que nos da el nodo `node-name()` y el nodo de conversión de tipo respectivamente. Imagine que necesita pasar datos de un archivo XML a otro XML de destino de la siguiente manera:

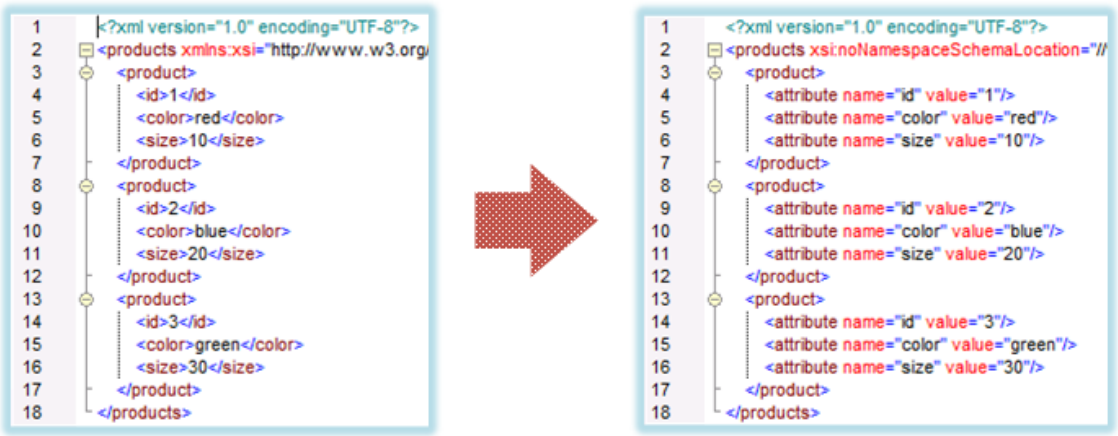


Fig. 6: asignación entre nombres de elemento XML y valores de atributo

Para conseguir este objetivo debemos diseñar esta asignación de datos:

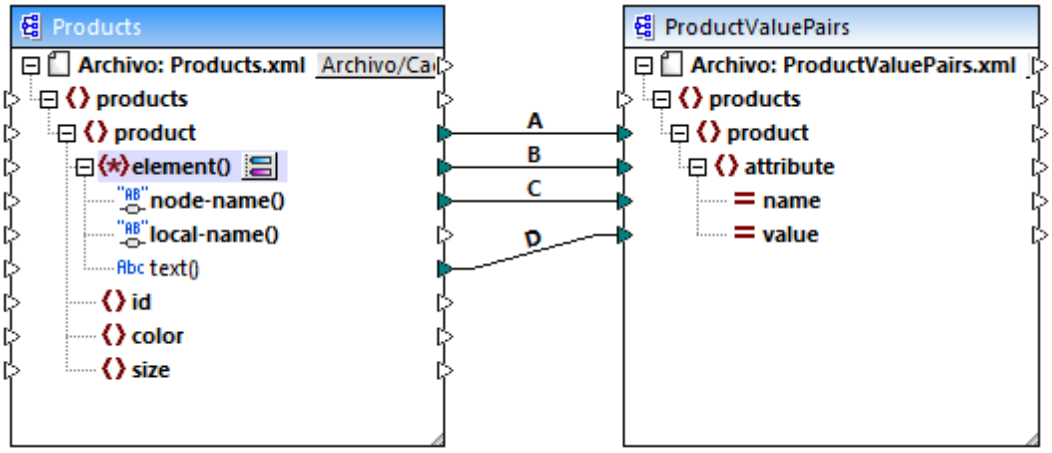


Fig. 7: asignar nombres de elemento XML a valores de atributo

El papel que `element()` desempeña aquí es aportar la secuencia de elementos secundarios de `product`, mientras que `node-name()` y `text()` aportan el nombre y el valor real de cada elemento de la secuencia. Esta asignación se describe más detalladamente en el apartado Ejemplo: asignar nombres de elemento a valores de atributo.

En el componente de destino `element()` no crea nada por sí mismo, lo cual constituye una excepción de la regla básica de asignación (por cada elemento del origen, crear un elemento de destino). Los elementos propiamente dichos los crean los nodos de conversión de tipo (usando el valor de `node-name()`) y por los nodos de prueba de nombre (usando su propio nombre).

attribute()

Como puede verse en la figura nº4, este elemento permite acceder a todos los atributos del nodo en tiempo de ejecución de la asignación. En un componente de origen aporta, en forma de secuencia, los atributos del nodo de origen que está conectado. Por ejemplo, en este archivo XML, la secuencia incluiría dos elementos (porque `product` tiene dos atributos):

```
<product id="1" color="red" />
```

Observe que el nodo `attribute()` solo aporta el valor de cada atributo en la secuencia, siempre como tipo string. El nombre de cada atributo viene dado por el nodo `node-name()`.

En un componente de destino este nodo procesa una secuencia conectada y crea un valor de atributo por cada elemento de la secuencia. El nombre de atributo viene dado por `node-name()`. Por ejemplo, imagine que necesita pasar datos de un archivo XML a otro archivo XML de la siguiente manera:

© 2018 Altova GmbH

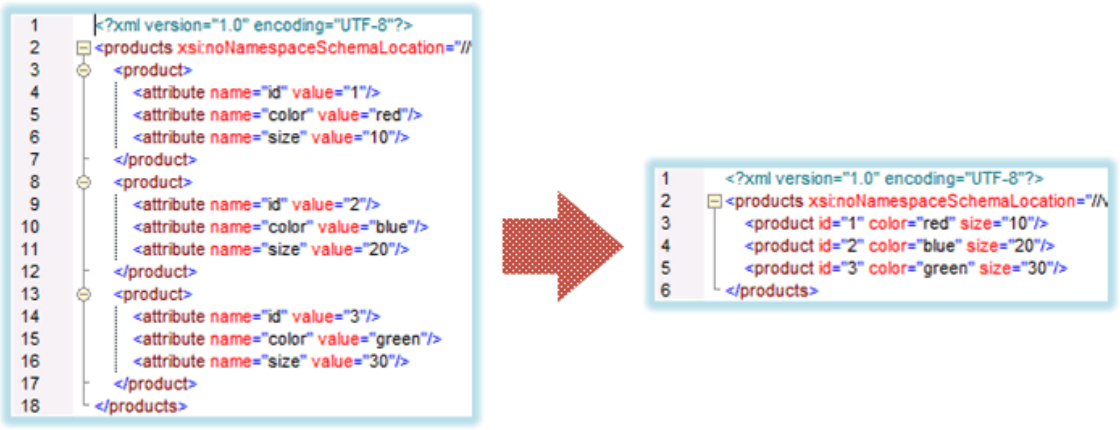


Fig. 8: asignar valores a nombres de atributo

Para conseguir este objetivo debemos diseñar esta asignación de datos:

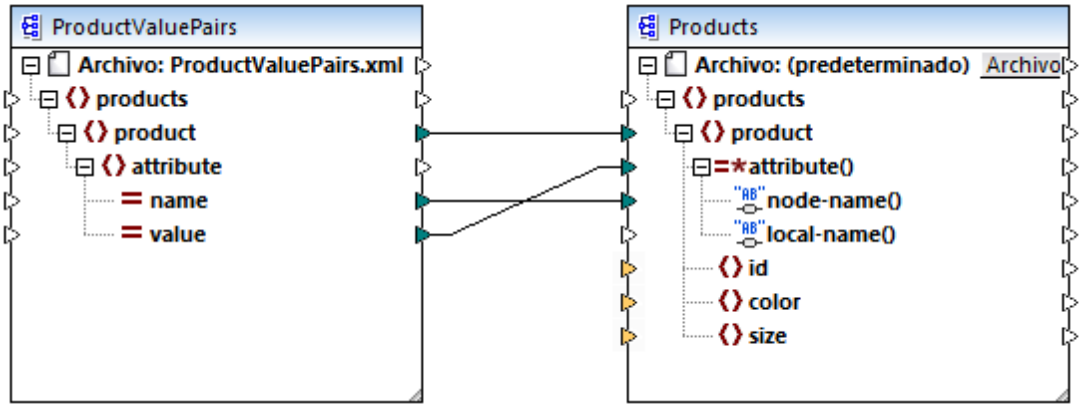


Fig. 9: asignar valores de atributo a nombres de atributo

Nota: esta transformación puede conseguirse sin habilitar el acceso dinámico a los atributos del nodo, pero aquí usamos el acceso dinámico para explicar cómo funciona `attribute()` en un componente de destino.

Si quiere reconstruir esta asignación, tenga en cuenta que usa los mismos componentes XML que la asignación **ConvertProducts.mfd** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**. La única diferencia es que el destino ahora es el origen y el origen ahora es el destino. Como datos de entrada del componente de origen deberá utilizar una instancia XML que contenga valores de atributo, por ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<products>
   <product>
      <attribute name="id" value="1"/>
      <attribute name="color" value="red"/>
```

```
      <attribute name="size" value="big"/>
   </product>
</products>
```

Nota: para simplificar el ejemplo se omitió la declaración de espacio de nombres y de esquema en el fragmento de código anterior.

node-name()

En un componente de origen `node-name()` aporta el nombre de cada elemento secundario de `element()` o el nombre de cada atributo de `attribute()` respectivamente. Por defecto el nombre que aporta `node-name()` es de tipo `xs:QName`. Para obtener el nombre como tipo string debe utilizarse el nodo `local-name()` (*véase la figura nº3*) o utilice la función QName-as-string.

En un componente de destino `node-name()` escribe el nombre de cada elemento o atributo que exista en `element()` o `attribute()`.

local-name()

Este nodo funciona igual que el nodo `node-name()`, pero su tipo es `xs:string` en lugar de `xs:QName`.

Nodo de conversión de tipo

En un componente de origen el nodo de conversión de tipo aporta el valor de cada elemento secundario que incluye `element()`. El nombre y la estructura de este nodo dependerá del tipo qeu esté seleccionado en el cuadro de diálogo "Configuración de secundarios con nombre dinámico" (figura nº2).

Para cambiar el tipo del nodo haga clic en el botón **Cambiar selección** y seleccione un tipo de la lista de tipos disponibles, incluido el comodín de esquema (`xs:any`). Consulte Obtener acceso a determinado tipo de nodos para obtener más información.

En un componente de destino el nodo de conversión de tipo escribe el valor de cada elemento secundario que incluye `element()`, en el tipo de datos correspondiente. El tipo de datos deseado se selecciona haciendo clic en el botón **Cambiar selección**.

Nodos de prueba de nombre

En un componente de origen los nodos de prueba de nombre permiten agrupar o filtrar elementos secundarios de una instancia de origen por nombre. Por ejemplo, si necesita filtrar elementos secundarios por nombre para que la asignación acceda a los datos de instancia usando el tipo correcto (*véase Obtener acceso a determinado tipo de nodos*). Para ver un ejemplo consulte Ejemplo: agrupar y filtrar nodos según su nombre.

En general los nodos de prueba de nombre funcionan casi igual que los nodos de elemento normales para leer y escribir valores y subestructuras. Sin embargo, como la semántica de la asignación es distinta cuando está habilitado el acceso dinámico, existen algunas restricciones.

© 2018 Altova GmbH

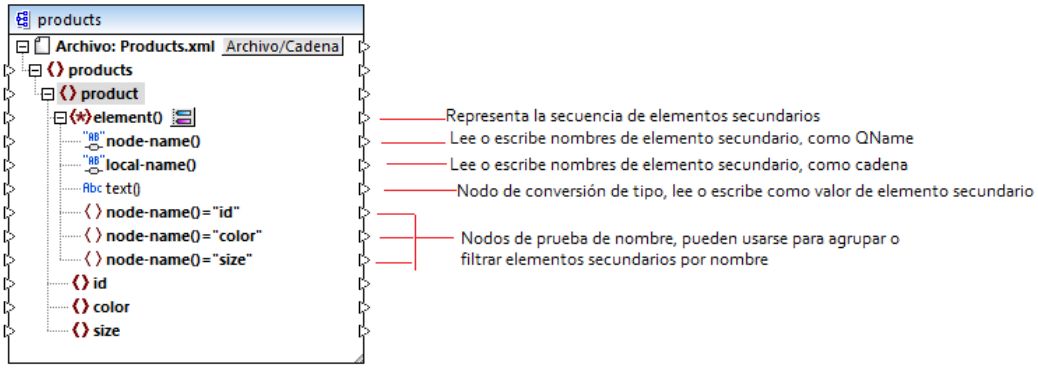
Por ejemplo, no se puede concatenar el valor de dos nodos de prueba de nombre.

En un componente de destino los nodos de prueba de nombre crean tantos elementos en el resultado como elementos existen en la secuencia de origen conectada. Su nombre reemplaza el valor que esté asignado a `node-name()`.

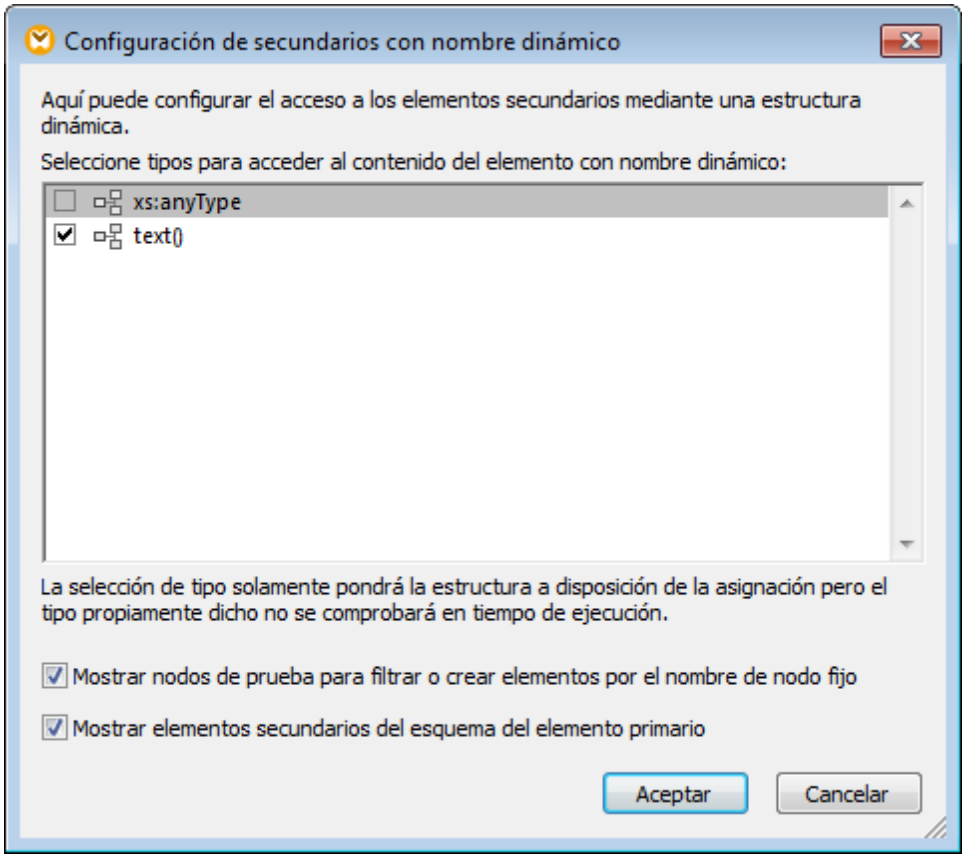
Si es necesario, puede ocultar los nodos de prueba de nombre en el componente. Esto se hace haciendo clic con el botón **Cambiar selección** del nodo `element()` y desactivando la casilla *Mostrar nodos de prueba....* en el cuadro de diálogo "Configuración de secundarios con nombre dinámico" (figura nº2).

5.14.2 Obtener acceso a determinado tipo de nodos

Como decíamos en el apartado anterior, podemos acceder a todos los elementos secundarios de un nodo haciendo clic con el botón derecho en el nodo y seleccionando el comando **Mostrar elementos secundarios con nombre dinámico** en el menú contextual. En tiempo de ejecución esto permitirá acceder al nombre de cada elemento secundario a través del nodo `node-name()`, mientras que el valor estará disponible a través de un nodo de conversión de tipo especial. En la imagen siguiente podemos ver que el nodo de conversión de tipo es el nodo `text()`.



Es importante señalar que el tipo de datos de cada elemento secundario no se conoce antes de que se ejecute la asignación. Además, el tipo de datos de cada elemento secundario puede ser distinto. Por ejemplo, un nodo `product` del archivo de instancia XML puede tener un elemento secundario `id` de tipo `xs:integer` y un elemento secundario `size` de tipo `xs:string`. Para poder acceder al contenido del nodo de un tipo específico el cuadro de diálogo (*imagen siguiente*) se abre cada vez que se habilita el acceso dinámico a los elementos secundarios de un nodo. Este cuadro de diálogo se puede abrir en cualquier momento con el botón **Cambiar selección** situado junto al nodo `element()`.



Cuadro de diálogo "Configuración de secundarios con nombre dinámico"

Para acceder al contenido de cada elemento en tiempo de ejecución tenemos varias opciones:

1. Acceder al contenido como cadena: marque la casilla **text()** del cuadro de diálogo. En este caso, se crea un nodo `text()` en el componente cuando se cierra el cuadro de diálogo. Esta opción es la más adecuada si el contenido es de tipo simple (`xs:int`, `xs:string`, etc.) y se describe en el apartado Ejemplo: asignar nombres de elemento a valores de atributo. Recuerde que el nodo **text()** solamente aparece si un nodo secundario del nodo actual puede contener texto.
2. Acceder al contenido como cierto tipo complejo permitido por el esquema. Cuando el esquema permite para el nodo seleccionado tipos complejos personalizados definidos globalmente, éstos también estarán disponibles en el cuadro de diálogo y podremos marcar sus casillas. En la imagen anterior, por ejemplo, no hay tipos complejos definidos globalmente por el esquema así que no hay ninguno en el cuadro de diálogo.
3. Acceder al contenido como cualquier tipo. Esto puede ser muy práctico en asignaciones de datos complejas y se consigue marcando la casilla de **xs:anyType**.

Debe tener en cuenta que en tiempo de ejecución MapForce no dispone de información (a través del nodo de conversión de tipo) sobre el tipo real del nodo de instancia. Por tanto, su asignación debe acceder al contenido del nodo usando el tipo correcto. Por ejemplo, si se espera que el nodo de una instancia XML de origen tenga nodos secundarios de varios tipos complejos, deberá hacer lo siguiente:

© 2018 Altova GmbH

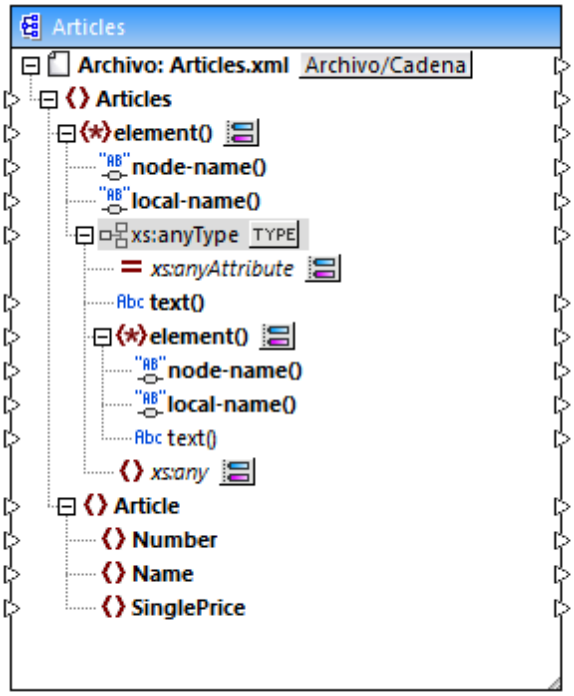
a) Asignar al nodo de conversión de tipo el tipo complejo que debe corresponderle.
b) Agregar un filtro para leer de la instancia solamente el tipo complejo que debe corresponder.

Esta técnica se describe con más detalle en Ejemplo: agrupar y filtrar nodos según su nombre.

Acceder a estructuras más profundas

En asignaciones más complejas es posible acceder a nodos de niveles más profundos del esquema que el nivel de secundarios inmediatos de un nodo. En las asignaciones más sencillas como las del apartado Ejemplo: asignar nombres de elemento a valores de atributo esta técnica no es necesaria porque la asignación solo accede a los secundarios inmediatos de un nodo XML. Sin embargo, cuando lo necesite también podrá acceder de forma dinámica a estructuras más profundas (p. ej. a *nietos*, *bisnietos*, etc.) si sigue estos pasos:

1. Cree una asignación nueva.
2. Seleccione el comando de menú **Insertar | Archivo o esquema XML** y navegue hasta el archivo XML de instancia (p. ej. **Articles.xml** de la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial**).
3. Haga clic con el botón derecho en el nodo `Articles` y seleccione **Mostrar elementos secundarios con nombre dinámico** en el menú contextual.
4. Marque la casilla **xs:anyType** en el cuadro de diálogo "Configuración de secundarios con nombre dinámico".
5. Haga clic con el botón derecho en el nodo `xs:anyType` y seleccione otra vez **Mostrar elementos secundarios con nombre dinámico** en el menú contextual.
6. Marque la casilla **text()** en el cuadro de diálogo "Configuración de secundarios con nombre dinámico".



En la imagen puede ver que el componente tiene dos nodos `element()`. El segundo de ellos ofrece acceso dinámico a los nietos del nodo `<Articles>` de la instancia **Articles.xml**.

```
<?xml version="1.0" encoding="UTF-8"?>
<Articles xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Articles.xsd">
   <Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>
   <Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>2.30</SinglePrice>
   </Article>
   <Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>
   <Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>57.50</SinglePrice>
   </Article>
</Articles>
```


© 2018 Altova GmbH

Articles.xml

Por ejemplo, para obtener los nombres de elementos *nietos* (`Number`, `Name`, `SinglePrice`) debemos dibujar una conexión entre el nodo `local-name()` del segundo nodo `element()` y un elemento de destino. Igualmente para obtener los valores de elementos *nietos* (`1`, `T-Shirt`, `25`), debemos dibujar una conexión desde el nodo `text()`.

Aunque en este ejemplo no es relevante, MapForce ofrece la posibilidad de seguir habilitando nombres de nodo dinámico para los siguientes nodos `xs:anyType` y así poder alcanzar niveles aún más profundos de la estructura.

Debe tener en cuenta que:

- El botón TYPE sirve para seleccionar cualquier tipo derivado del esquema actual y mostrarlo en un nodo independiente. Esto puede ser práctico a la hora de crear asignaciones entre tipos de esquema derivados (*véase* *Tipos XML Schema derivados*).
- El botón **Cambiar selección** situado junto a un nodo `element()` abre el cuadro de diálogo "Configuración de secundarios con nombre dinámico".
- El botón **Cambiar selección** situado junto a `xs:anyAttribute` permite seleccionar cualquier atributo definido globalmente en el esquema. Igualmente el botón **Cambiar selección** situado junto al elemento `xs:any` permite seleccionar cualquier elemento definido globalmente en el esquema. Se trata del mismo funcionamiento de las asignaciones de comodines de esquema (*véase* *Comodines: xs:any / xs:anyAttribute*). Si usa esta opción, asegúrese de que el esquema permite que el atributo o elemento exista de verdad.

5.14.3 Ejemplo: asignar nombres de elemento a valores de atributo

Este ejemplo demuestra cómo asignar nombres de elemento de un documento XML a valores de atributo de un documento XML de destino. El ejemplo viene acompañado del diseño de asignación **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\ConvertProducts.mfd**.

Para comprender cómo funciona el ejemplo primero debemos imaginar que tenemos un archivo XML que contiene una lista de productos. Cada producto tiene este formato:

```
<product>
   <id>1</id>
   <color>red</color>
   <size>10</size>
</product>
```

Nuestro objetivo es convertir la información disponible sobre cada producto en pares de nombre/valor. Por ejemplo:

```
<product>
   <attribute name="id" value="1" />
   <attribute name="color" value="red" />
```

```
    <attribute name="size" value="10" />
</product>
```

Para conseguirlo el diseño de asignación del ejemplo utiliza la característica de MapForce conocida como "acceso dinámico a nombres de nodo". El término *dinámico* hace referencia al hecho de que, cuando se ejecute la asignación, se podrán leer los nombres de nodo (no solo los valores) y estos nombres se podrán usar como valores. A continuación explicamos cómo diseñar esta asignación.

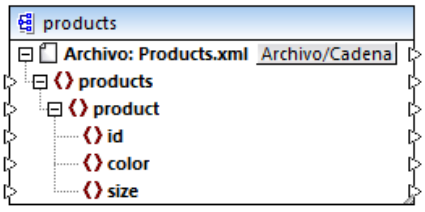
Paso nº1: agregar el componente XML de origen a la asignación

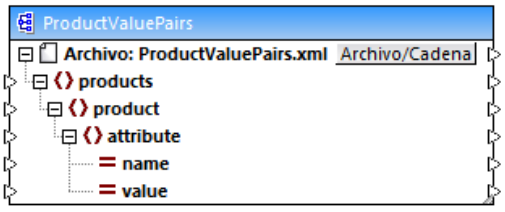
- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Products.xml**. Este archivo XML apunta al esquema **Products.xsd** que está situado en la misma carpeta.

Paso nº2: agregar el componente XML de destino a la asignación

- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo de esquema **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\ProductValuePairs.xsd**. Cuando la aplicación solicite un archivo de instancia, haga clic en **Omitir**. Cuando solicite un elemento raíz, seleccione `products`.

Llegados a este punto el diseño tendrá este aspecto:

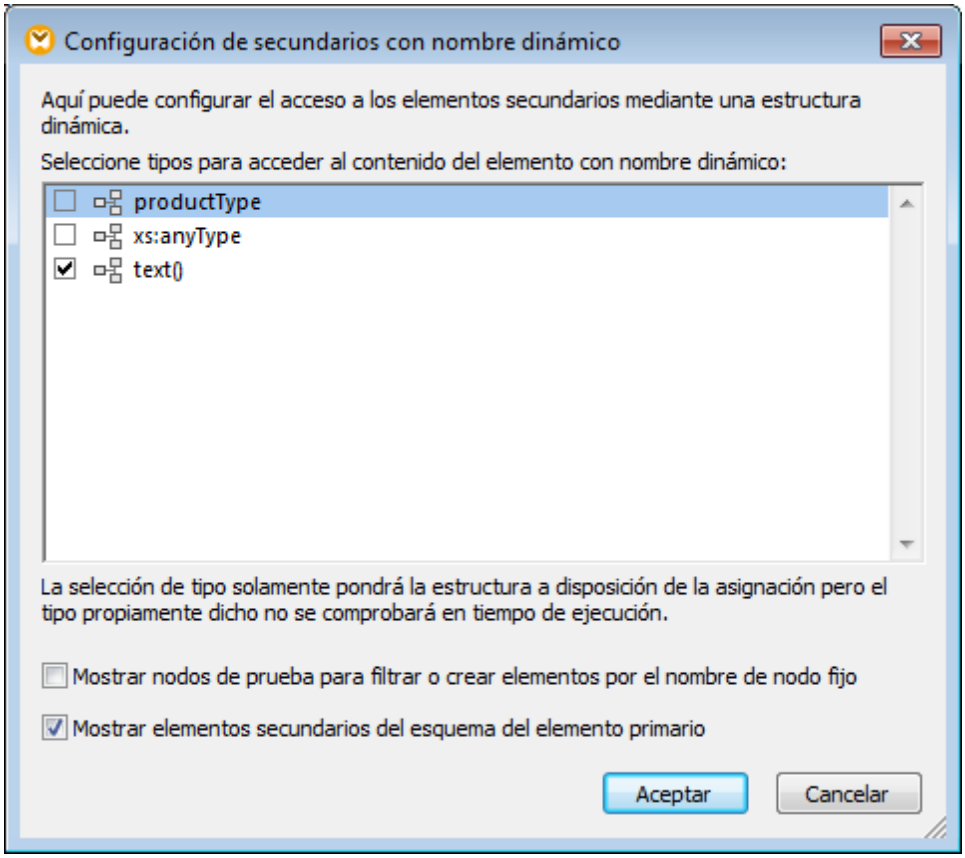




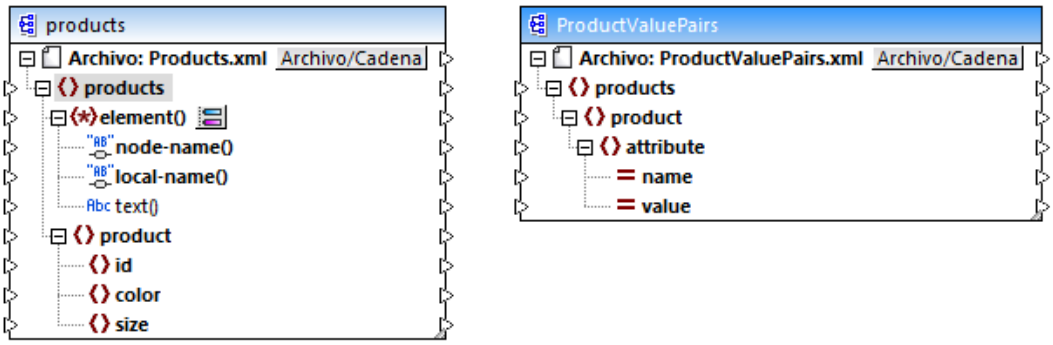
Paso nº3: habilitar el acceso dinámico a los nodos secundarios

1. Haga clic con el botón derecho en el nodo `products` del componente de origen y seleccione **Mostrar elementos secundarios con nombre dinámico** en el menú contextual.
2. En el cuadro de diálogo que aparece seleccione el tipo **text()** y deje las demás opciones como están.

© 2018 Altova GmbH

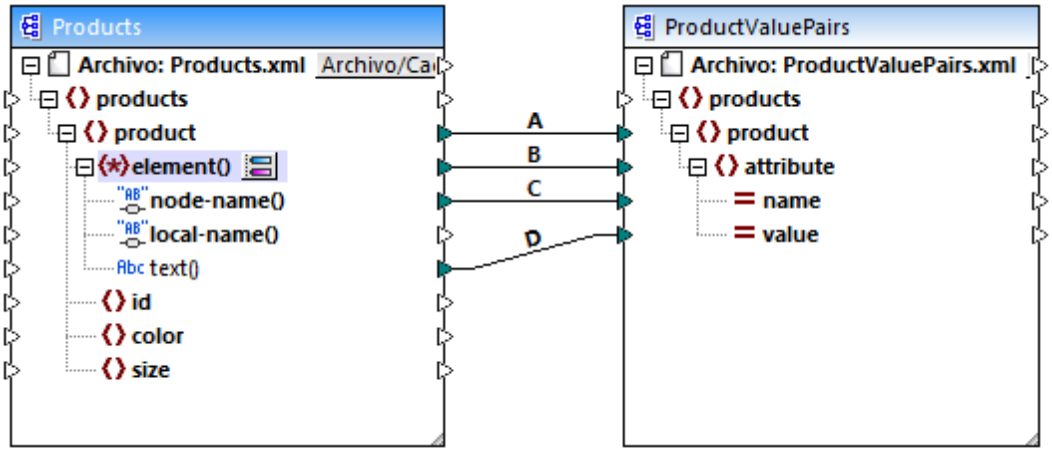


Observe que esto añade un nodo llamado `text()` al componente de origen. Este nodo se encargará de suministrar el contenido de los elementos secundarios a la asignación (en este caso, el valor de "id", "color" y "size").



Paso nº4: dibujar las conexiones de asignación de datos

Por último, debemos dibujar las conexiones de asignación de datos A, B, C y D que pueden verse en la imagen siguiente (si quiere, haga doble clic en cada línea de conexión, empezando por la primera, e introduzca el texto "A", "B", "C" y "D" respectivamente en el cuadro *Descripción*.



ConvertProducts.mfd

En el diseño de asignación anterior, la conexión A crea en el componente de destino un producto por cada producto del componente de origen. Se trata de una conexión estándar de MapForce que no se ocupa de los nombres de los nodos. Sin embargo, la conexión B crea en el componente de destino un elemento nuevo llamado `attribute` por cada elemento secundario de `product` del componente de origen.

> La conexión B es una conexión crucial en la asignación pues se encarga de transferir una *secuencia* de elementos secundarios de `product` desde el componente de origen al componente de destino. No transfiere los nombres ni valores propiamente dichos. Por tanto, debe entenderse de la siguiente forma: si el **element()** de origen tiene X secundarios, entonces crea X instancias de dicho elemento en el componente de destino. En este caso concreto, `product` tiene tres secundarios en el componente de origen (`id`, `color` y `size`). Esto significa que cada `product` del componente de destino tendrá tres secundarios llamados `attribute`.
>
> Aunque ahora no es el caso, la misma regla puede utilizarse para asignar elementos secundarios de **attribute()**: si el elemento **attribute()** de origen tiene X atributos secundarios, la conexión creará X instancias de dicho elemento en el componente de destino.

Después, la conexión C copia el nombre real de cada elemento secundario de `product` en el componente de destino (literalmente "id", "color" y "size").

Por último, la conexión D copia el valor de cada elemento secundario de `product` (como tipo `string`) en el componente de destino.

Para consultar una vista previa de los resultados de la asignación abrimos el panel *Resultados* y consultamos el XML que se genera. Tal y como esperábamos, el resultado contiene varios productos cuyos datos están almacenados como pares nombre/valor.

```
<?xml version="1.0" encoding="UTF-8"?>
<products xsi:noNamespaceSchemaLocation="ProductValuePairs.xsd"
```


© 2018 Altova GmbH

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <product>
      <attribute name="id" value="1"/>
      <attribute name="color" value="red"/>
      <attribute name="size" value="10"/>
   </product>
   <product>
      <attribute name="id" value="2"/>
      <attribute name="color" value="blue"/>
      <attribute name="size" value="20"/>
   </product>
   <product>
      <attribute name="id" value="3"/>
      <attribute name="color" value="green"/>
      <attribute name="size" value="30"/>
   </product>
</products>
```

Resultado de la asignación

5.14.4 Ejemplo: agrupar y filtrar nodos según su nombre

Este ejemplo demuestra cómo diseñar una asignación que lee pares de clave/valor de una lista de propiedades XML (o XML plist) y los escribe en un archivo CSV. Las listas de propiedades XML permiten almacenar información de objetos OS X e iOS en formato XML (véase https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/PropertyLists/UnderstandXMLPlist/UnderstandXMLPlist.html.) Este ejemplo viene acompañado de un diseño de asignación: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\ReadPropertyList.mfd**.

A continuación podemos ver un fragmento de código del archivo XML de origen.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist SYSTEM "https://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
   <dict>
      <key>First Name</key>
      <string>William</string>
      <key>Last Name</key>
      <string>Shakespeare</string>
      <key>Birthdate</key>
      <integer>1564</integer>
      <key>Profession</key>
      <string>Playwright</string>
      <key>Lines</key>
      <array>
         <string>It is a tale told by an idiot,</string>
         <string>Full of sound and fury, signifying nothing.</string>
      </array>
   </dict>
</plist>
```

El objetivo de la asignación es crear una línea nueva en el archivo CSV a partir de ciertos pares de clave/valor del nodo `<dict>` del archivo de lista de propiedades. Concretamente, la asignación solamente filtrará los pares **<key>** - **<string>** y el resto de pares clave/valor (p. ej. **<key>** - **<integer>**) se pasarán por alto. En el archivo CSV la nueva línea debe almacenar el nombre de la propiedad, separado del valor de la propiedad por una coma. En otras palabras, el resultado de la asignación debe tener este aspecto:

```
First Name,William
Last Name,Shakespeare
Profession,Playwright
```

Para conseguir el objetivo de la asignación será necesario tener acceso dinámico a los nodos secundarios del nodo `dict`. Además, necesitaremos la función group-starting-with para agrupar los pares de clave/valor recuperados del archivo XML. Y, por último, deberemos usar un filtro para filtrar los nodos cuyo nombre sea "string".

A continuación explicamos cómo diseñar esta asignación paso a paso.

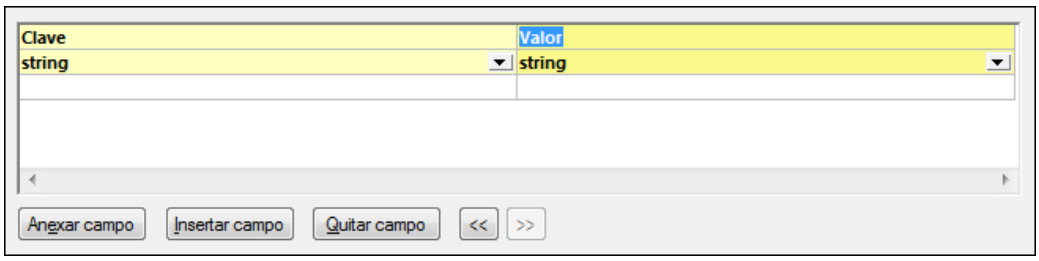
Paso nº1: agregar el componente XML de origen a la asignación

1. Elija el lenguaje de transformación integrado (*véase Seleccionar un lenguaje de transformación*).
2. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\plist.xml**. Este archivo XML apunta al esquema **plist.xsd** que está situado en la misma carpeta.

© 2018 Altova GmbH

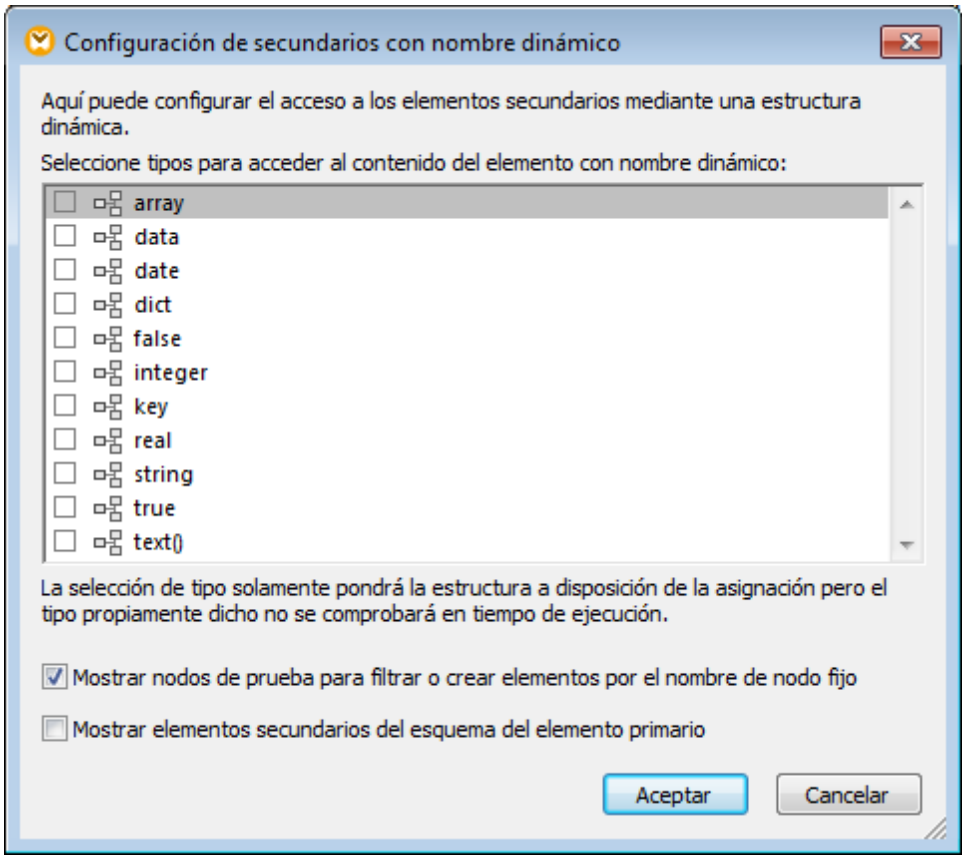
Paso nº2: agregar el componente CSV de destino a la asignación

1. En el menú **Insertar** haga clic en el comando **Archivo de texto**. Ahora seleccione la opción *Utilizar procesamiento básico para archivos CSV...* en el cuadro de diálogo.
2. Añada un campo CSV más al componente con el botón **Anexar campo**.
3. Haga doble clic en el nombre de cada campo e introduzca **`Key`** para el nombre del primer campo y **`Value`** para el nombre del segundo. El campo **`Key`** almacenará el nombre de la propiedad, mientras que el campo **`Value`** almacenará su valor. Consulte la sección Archivos CSV y archivos de texto para obtener más información.

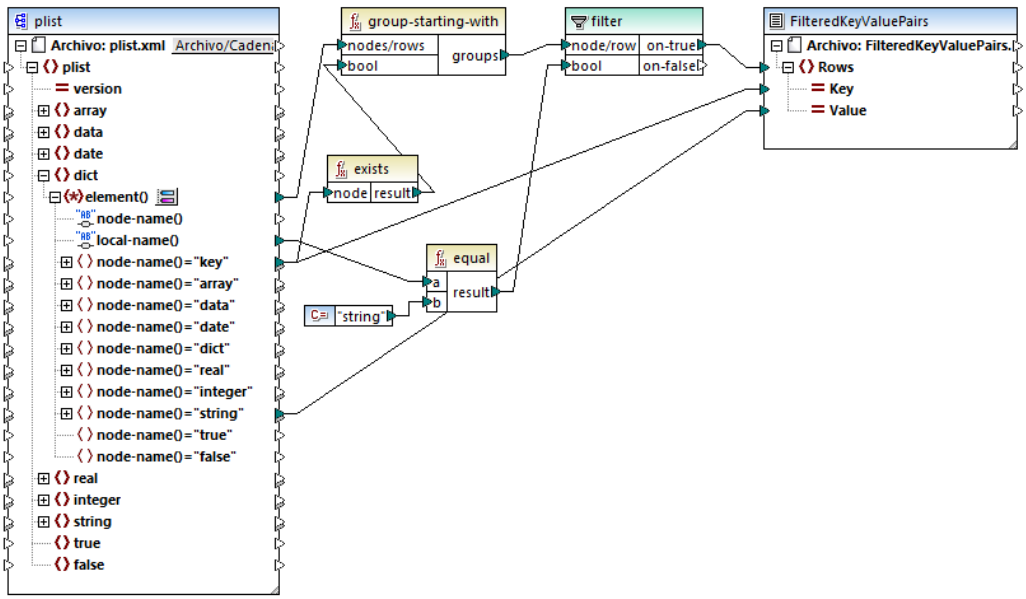


Paso nº3: agregar el filtro y funciones

1. Arrastre las funciones **`equal`**, **`exists`** y **`group-starting-with`** desde la ventana Bibliotecas hasta el área de asignación (*véase Trabajar con funciones*).
2. Para agregar el filtro haga clic en el menú **Insertar** y seleccione el comando **Filtro: nodos/filas** (*véase Filtros y condiciones*).
3. En el menú **Insertar** seleccione el comando **Constante** e introduzca el texto "string".
4. En el componente de origen haga clic con el botón derecho en el nodo `dict` y seleccione **Mostrar elementos secundarios con nombre dinámico** en el menú contextual. En el cuadro de diálogo que aparece compruebe que la casilla *Mostrar nodos de prueba para filtrar o crear elementos...* está marcada.



5. Dibuje las conexiones que aparecen en la imagen siguiente.



ReadPropertyList.mfd

 © 2018 Altova GmbH

Funcionamiento de la asignación

El elemento `element()` del componente de origen suministra todos los secundarios del nodo `dict` en forma de secuencia a la función **`group-starting-with`**. La función **`group-starting-with`** crea un grupo nuevo cada vez que se detecta un nodo llamado `key`. La función **`exists`** comprueba si se cumple esta condición y devuelve el valor binario true/false a la función de agrupación.

Por cada grupo el filtro comprueba si el nombre del nodo actual equivale a "string" (con ayuda de la función **`equal`**). El nombre propiamente dicho se lee de `local-name()`, que suministra el nombre del nodo en forma de cadena.

Las conexiones con el componente de destino se encargan de lo siguiente:

- Cuando la condición de filtrado se cumple, entonces se crea una fila nueva en el componente CSV de destino.
- `Key` (nombre de la propiedad) se toma del valor del elemento `key` del componente de origen.
- `Value` (valor de la propiedad) se toma del nodo de prueba `string`.

5.15 Reglas y estrategias de asignación de datos

Por lo general MapForce crea asignaciones de datos de forma intuitiva pero en algunas ocasiones puede que los resultados tengan demasiados elementos o muy pocos. En este tema ofrecemos consejos para evitar este tipo de problemas.

Regla general

Por lo general, cada conexión entre un elemento de origen y uno de destino significa que, por cada elemento de origen, se creará un elemento de destino. Si el nodo de origen incluye contenido simple (p. ej. una cadena o un entero) y el nodo de destino acepta contenido simple, MapForce copia el contenido en el nodo de destino y, si hace falta, convierte el tipo de datos.

Esta es la regla general para todas las conexiones excepto en estos casos:

- El elemento raíz XML de destino se crea una sola vez. Si conecta una secuencia con el elemento raíz XML de destino, solo se repetirá el contenido del elemento pero no el elemento raíz propiamente dicho y puede que el resultado no sea válido según el esquema. Si también se conectan los atributos del elemento raíz, la serialización XML fallará en tiempo de ejecución, así que debe evitarse conectar una secuencia con el elemento raíz. Si lo que desea es crear varios archivos de salida, conecte la secuencia con el nodo `Archivo`, por medio de algún tipo de función que genere nombres de archivo.
- Algunos nodos aceptan un solo valor y no una secuencia (p. ej. los atributos XML, los campos de BD y los componentes de destino de las funciones definidas por el usuario.

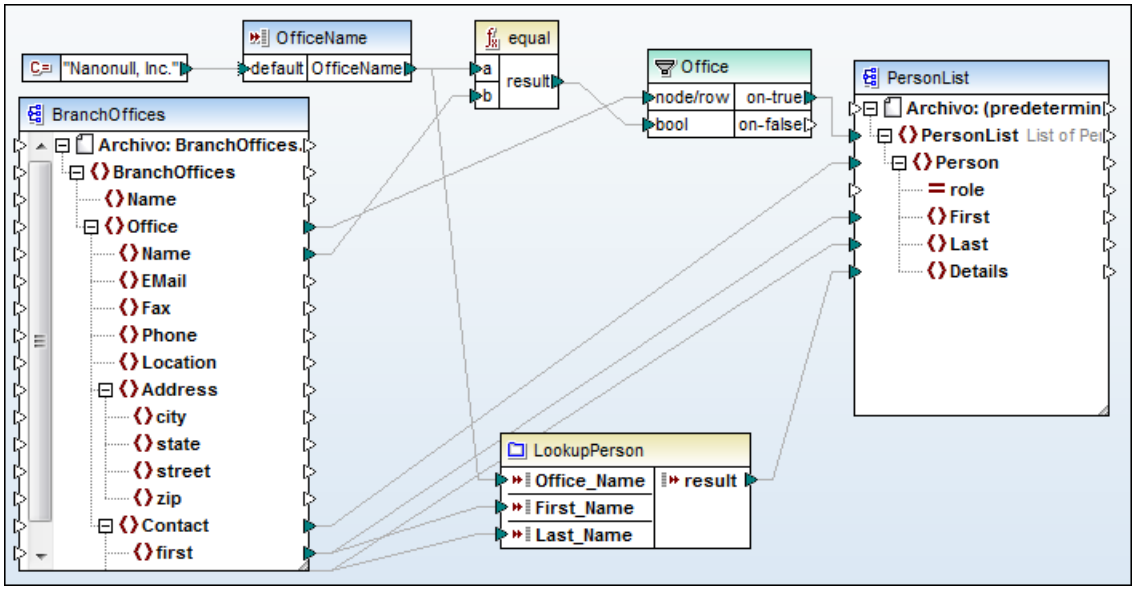
Los elementos "context" y "current"

MapForce muestra la estructura de un archivo de esquema, base de datos o EDI en el componente como una jerarquía con elementos que se pueden asignar. Cada uno de estos nodos puede tener varias instancias (o ninguna) en el archivo de instancia o de la base de datos.

Ejemplo: en el componente de origen de la asignación **PersonListByBranchOffice.mfd** solo hay un nodo **first** (dentro de **Contact**). En el archivo de instancia **BranchOffices.xml**, hay varios nodos **first** y **Contact** con contenido distinto (dentro de los nodos primarios **Office**).

Dependiendo de cuál sea el nodo de **contexto** actual (del nodo de **destino**), se seleccionarán unos nodos de origen u otros y sus datos se copiarán, por medio del conector, en el componente/elemento de destino.

© 2018 Altova GmbH



PersonListByBranchOffice.mfd

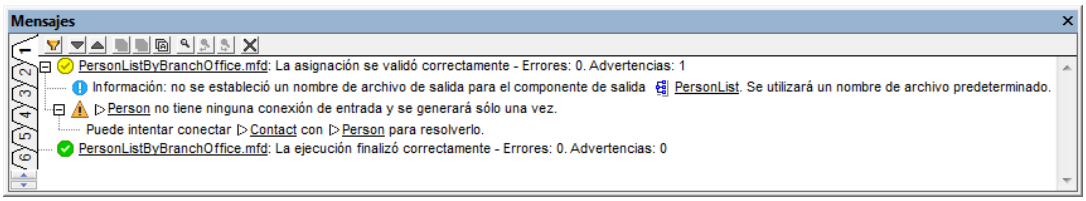
Este contexto viene definido por el **nodo de destino actual** y por las conexiones con sus antecesores:

- Al principio el contexto solo contiene los componentes de origen, pero ningún nodo concreto. Cuando evalúa la asignación MapForce procesa primero el nodo **raíz de destino** (PersonList) y después sigue recorriendo la jerarquía.
- El conector que está unido al nodo de **destino** se sigue hasta todos los elementos de origen a los que está conectado (directa o indirectamente, incluso a través de funciones). Los elementos de origen y los resultados de las funciones se añaden al contexto para este nodo.
- Por cada nodo de destino nuevo se establece un contexto nuevo, que inicialmente contiene todos los elementos del contexto del nodo primario. Los nodos de destino del mismo nivel, por tanto, son independientes entre sí pero tienen acceso a todos los datos de origen de sus nodos primarios.

Por ejemplo, en el caso de la asignación (**PersonListByBranchOffice.mfd**):

- La conexión que une a **Office** con **PersonList** a través del filtro (Office) define una sola oficina como contexto para todo el documento de destino (porque PersonList es el elemento raíz del componente de destino). El nombre de la oficina viene dado por el componente de entrada, que tiene el valor predeterminado "Nanonull, Inc."
- Todos los datos/todas las conexiones de los **descendientes** del elemento raíz PersonList se ven afectados automáticamente por la condición de filtrado porque la oficina seleccionada está en el contexto.
- La conexión que une a **Contact** con **Person** crea un Person de destino por cada elemento Contact del XML de origen (la regla general se aplica). Por cada Person se añade un Contact concreto al contexto, a partir del cual se crearán los secundarios de Person.
- El conector que une a **first** con **First** selecciona el nombre del Contact actual y lo escribe en el elemento First de destino.

Si se obvia el conector que une **Contact** con **Person**, se crearía solo un Person con varios nodos First, Last y Detail, pero eso no es lo que queremos en este caso. Así que cuando eso ocurre MapForce emite una advertencia y una sugerencia para solucionar el problema: "Puede intentar conectar Contact con Person para resolverlo.":



Secuencias

MapForce muestra la estructura de un archivo de esquema, base de datos o EDI en el componente como una jerarquía de elementos que se pueden asignar.

Dependiendo de cuál sea el **contexto (de destino**), cada elemento asignable de un componente de origen puede representar:

- un **solo nodo de instancia** del archivo de entrada asignado (o de la BD asignada)
- una **secuencia de cero a varios nodos de instancia** del archivo de entrada (o de la BD)

Si hay una secuencia conectada a un nodo de destino, se crea un bucle para crear tantos nodos de destino como nodos de origen existan.

Si se coloca un filtro entre la secuencia y el nodo de destino, se verifica la condición binaria por cada nodo de entrada (es decir, por cada elemento de la secuencia). Más concretamente se comprueba si en cada secuencia hay como mínimo un booleano que dé true como resultado. La configuración de contexto prioritario puede influir en el orden de evaluación (ver más abajo).

Como se dijo anteriormente, las condiciones de filtrado afectan automáticamente a todos los nodos descendientes.

Nota: si el esquema de origen especifica que determinado nodo aparece una sola vez, MapForce puede eliminar el bucle y tomar solo el primer elemento, que sabe que existe. Esta optimización se puede deshabilitar en el cuadro de diálogo "Configuración del componente" (casilla *Optimización de procesamiento de datos de entrada basada en min/maxOccurs"*).

Las **entradas de las funciones** (normales, no secuenciales) funcionan de forma similar: si una secuencia está conectada a una entrada de función, se crea un bucle alrededor de la llamada a función para que produzca **tantos resultados** como elementos hay en la secuencia.

Si una secuencia está conectada a más de una entrada de función, MapForce crea bucles anidados que procesarán el **producto cartesiano** de todas las entradas. Esto no suele ser lo más indicado, así que no conecte más de una secuencia formada por varios elementos a una función.

Nota: si hay una secuencia vacía conectada a una función (p. ej. **`concat`**), recibirá como resultado una secuencia vacía, que no producirá ningún nodo de destino. Si no hay resultados en los documentos de salida porque no hay datos de entrada, puede usar la función **`substitute-missing`** para insertar valores sustitutos.

 © 2018 Altova GmbH

Las funciones con **entradas secuenciales** son las únicas funciones que pueden producir un resultado si la secuencia de entrada está **vacía**:

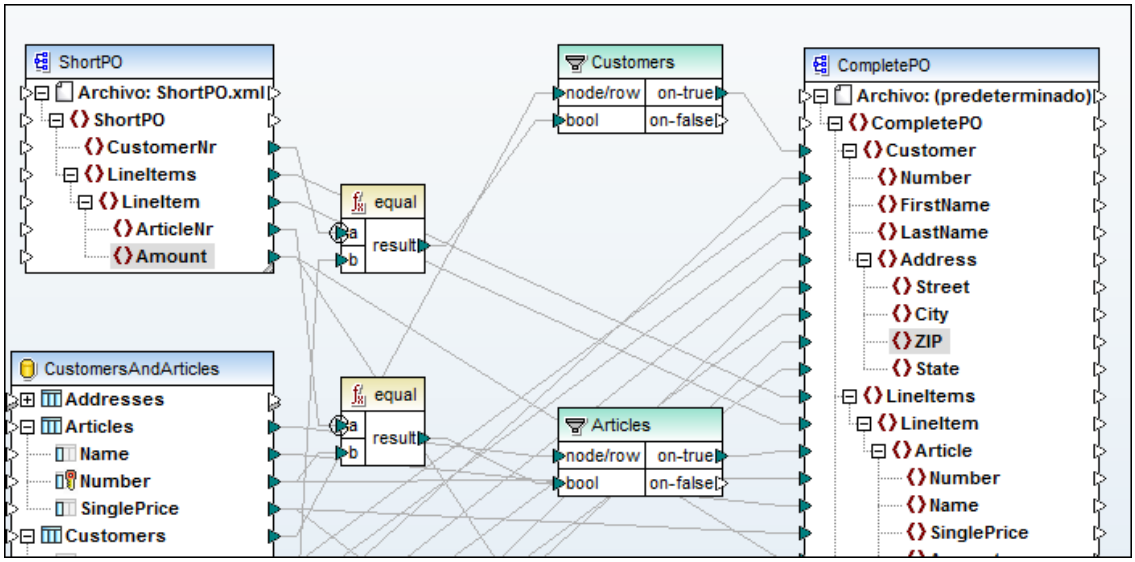
- **`exists`**, **`not-exists`** y **`substitute-missing`** (también, **`is-not-null`**, **`is-null`** y **`substitute-null`**, que son los alias para las tres primeras)
- funciones de agregado (**`sum`**, **`count`**, etc.)
- funciones definidas por el usuario que acepten secuencias (es decir, funciones no insertadas).

La entrada secuencia de dichas funciones siempre se evalúa independientemente del nodo de destino actual en el contexto de sus antecesores. Esto también significa que cualquier componente de filtrado o Where SQL que esté conectado a dichas funciones no tendrán efecto alguno en las demás conexiones.

Contexto prioritario

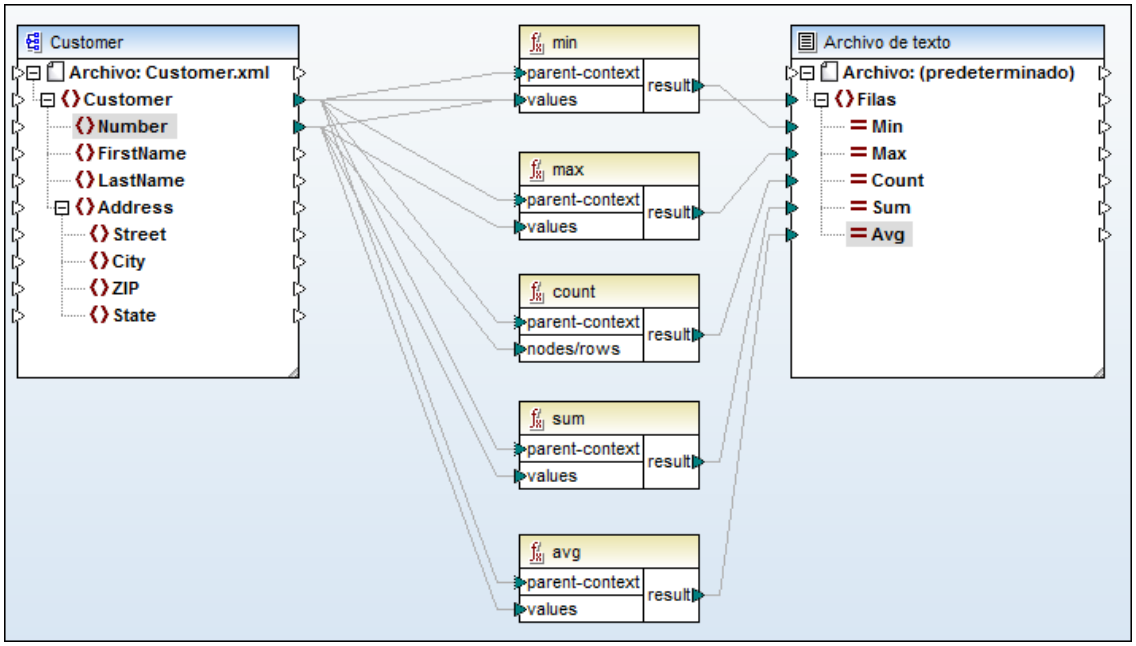
Por lo general los parámetros de función se evalúan de arriba a abajo, pero también se puede definir que un parámetro se evalúe antes que los demás. Esto se hace estableciendo un **contexto prioritario**.

En las funciones que están conectadas a la entrada bool de condiciones de **filtrado**, el contexto prioritario no solo afecta a la función de comparación sino a la evaluación del filtro, así que se pueden unir dos secuencias de origen (*véase CompletePO.mfd: CustomerNo y Number*). Consulte el apartado Nodo de contexto prioritario para obtener más información.



Reemplazar el contexto

Algunas funciones de agregado tienen una entrada opcional llamada “parent-context”. Si esta entrada se deja sin conectar, no tiene efecto alguno y la función se evalúa en el contexto normal de para entradas secuencias (es decir, en el contexto del primario del nodo de destino).



Si por el contrario se conecta la entrada `parent-context` a un nodo de origen, la función se evalúa por cada nodo `parent-context` y producirá un resultado distinto por cada instancia. Consulte el apartado Reemplazar el contexto de la asignación para obtener más información.

Traer varios nodos del mismo componente de origen al contexto

Esto puede ser necesario en algunos casos y se puede hacer con ayuda de variables intermedias.

5.15.1 Cambiar el orden de procesamiento de los componentes

En MapForce pueden diseñarse asignaciones con varios componentes de destino. Cada uno de ellos cuenta con un botón de vista previa que permite consultar el resultado que genera dicho componente.

Si la asignación se ejecuta desde la línea de comandos o desde el código generado, entonces se ejecuta toda la asignación (independientemente de qué botón de vista previa esté activo) y se genera el resultado de todos los componentes de destino.

El orden en el que se procesan los componentes de destino se puede modificar con solo cambiar la posición de los componentes de destino. La posición de un componente viene definida por la posición de su **esquina superior izquierda**.

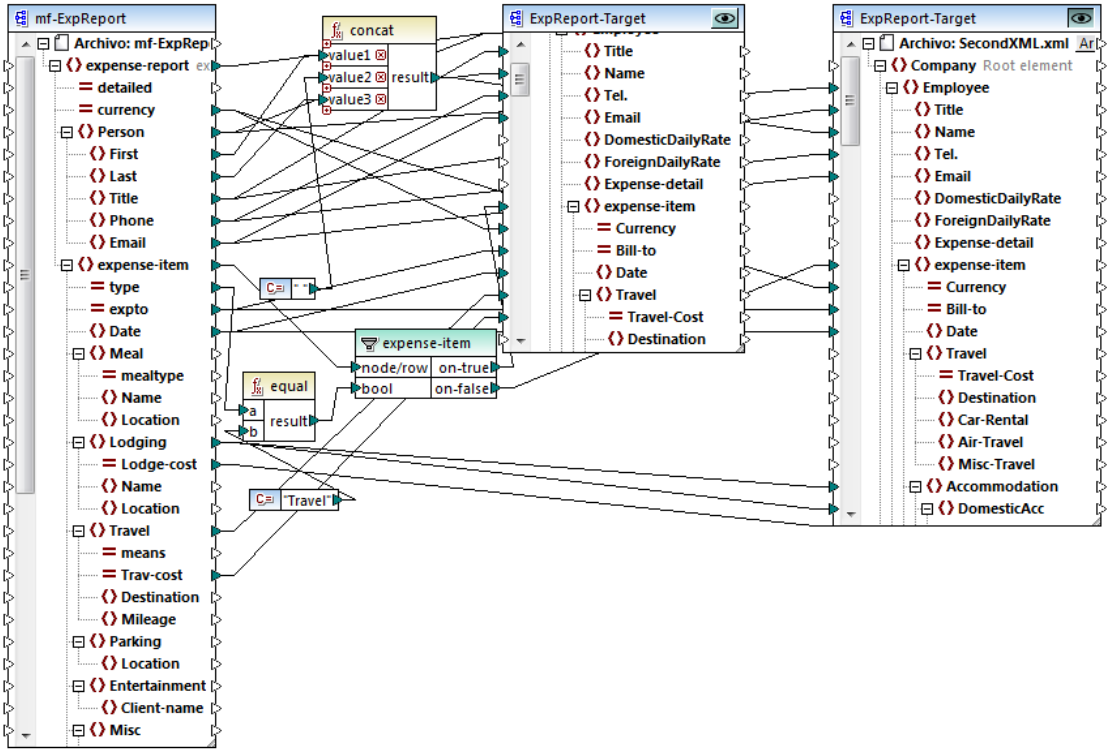
Los componentes de destino se procesan de abajo a arriba y de izquierda a derecha:

- Si hay dos componentes con la misma posición vertical, se procesa primero el situado a la izquierda.
- Si hay dos componentes con la misma posición horizontal, se procesa el situado más arriba.
- En el caso improbable de que dos componentes tengan una posición idéntica, se usa

© 2018 Altova GmbH

automáticamente un identificador de componente interno y único que garantiza un orden de procesamiento bien definido, pero inalterable.

En la imagen siguiente aparece el tutorial **Tut-ExpReport-multi.mfd**, situado en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**, que tiene dos componentes de destino. Ambos componentes tienen la misma posición **vertical** y el componente de destino de la derecha tiene activado el botón de vista previa.



Tut-ExpReport-multi.mfd (MapForce Enterprise Edition)

Si seleccionamos XSLT2 y generamos el código:

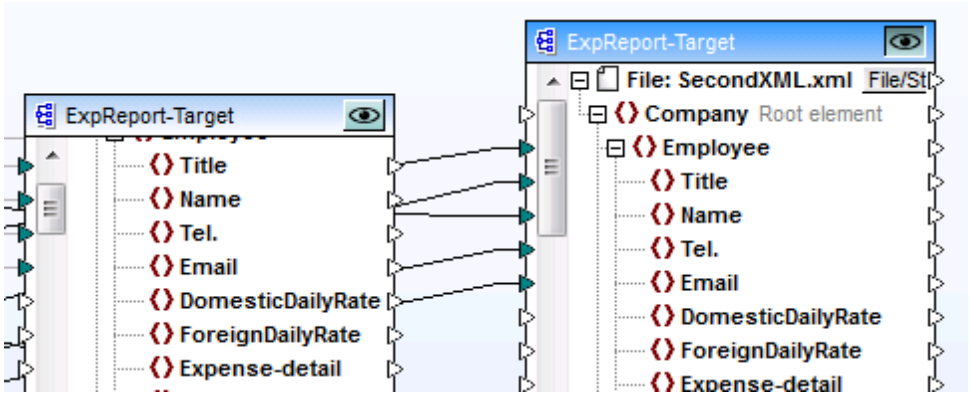
- Primero se procesa el componente de destino de la izquierda y se genera el archivo **ExpReport.xml**.
- Después se procesa el componente de destino de la derecha y se genera el archivo **SecondXML.xml**.

Para comprobar si éste es el orden de procesamiento abra el archivo **DoTransform.bat** (situado en la carpeta de salida que se especificara) y consulte la secuencia de los archivos de salida generados. El primer archivo que genera el archivo de procesamiento por lotes es **ExpReport-Target.xml** y el segundo es **SecondXML.xml**.

```
@echo off

RaptorXML xslt --xslt-version=2 --
input="C:\Users\me\Documents\Altova\MapForce2013\MapForceExamples\Tutorial\m
f-ExpReport.xml" --
output="C:\Users\me\Documents\Altova\MapForce2013\MapForceExamples\Tutorial\
ExpReport-Target.xml" %* "MappingMapToExpReport-Target.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --
input="C:\Users\me\Documents\Altova\MapForce2013\MapForceExamples\Tutorial\m
f-ExpReport.xml" --
output="C:\Users\me\Documents\Altova\MapForce2013\MapForceExamples\Tutorial\
SecondXML.xml" %* "MappingMapToExpReport-Target2.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```

Cambiar la secuencia de procesamiento de la asignación:

1. Haga clic en el componente de destino situado a la izquierda y muévalo debajo del situado a la derecha.



2. Vuelva a generar el código y consulte otra vez el archivo **DoTransform.bat**.

```
@echo off

RaptorXML xslt --xslt-version=2 --
input="C:\Users\alp\Documents\Altova\MapForce2013\MapForceExamples\Tutorial\
mf-ExpReport.xml" --
output="C:\Users\alp\Documents\Altova\MapForce2013\MapForceExamples\Tutorial
\SecondXML.xml" %* "MappingMapToExpReport-Target.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
RaptorXML xslt --xslt-version=2 --
input="C:\Users\alp\Documents\Altova\MapForce2013\MapForceExamples\Tutorial\
mf-ExpReport.xml" --
output="C:\Users\alp\Documents\Altova\MapForce2013\MapForceExamples\Tutorial
\ExpReport-Target.xml" %* "MappingMapToExpReport-Target2.xslt"
IF ERRORLEVEL 1 EXIT/B %ERRORLEVEL%
```

Ahora el primer archivo que genera el archivo de procesamiento por lotes es **SecondXML.xml** y el segundo es **ExpReport-Target.xml**.

Asignaciones en cadena

La misma secuencia de procesamiento que acabamos de describir es la que se sigue en las asignaciones en cadena. El grupo de la asignación en cadena se entienda como unidad indisoluble y, por tanto, si cambiamos la posición del componente intermedio o de destino final de una asignación en cadena, la secuencia de procesamiento no cambia.

El único caso en el que la posición de los componentes de destino finales de cada grupo determina qué se procesa primero es cuando una asignación tiene varias cadenas o componentes de destino.

- Si dos componentes de destino finales tienen la misma posición vertical, se procesa primero el situado a la izquierda.
- Si dos componentes de destino finales tienen la misma posición horizontal, se procesa primero el situado más arriba.
- En el caso improbable de que dos componentes tengan una posición idéntica, se usa automáticamente un identificador de componente interno y único que garantiza un orden de procesamiento bien definido, pero inalterable.

5.15.2 Nodo de contexto prioritario

Cuando aplicamos una función a distintos elementos del esquema o de la BD, MapForce necesita saber qué nodo será el nodo de contexto. Los demás nodos se procesan en relación al nodo de contexto. Esto se consigue designando un elemento (o nodo) como contexto prioritario.

El contexto prioritario se utiliza para clasificar por orden de prioridad la ejecución de las asignaciones de los elementos no relacionados entre sí.

Las asignaciones siempre se ejecutan de arriba a abajo. Si recorre/realiza una búsqueda en dos tablas, entonces se procesa primero un bucle y después otro. Cuando se realizan asignaciones entre elementos no relacionados entre sí, sin establecer el contexto prioritario, MapForce no sabe qué bucle debe ejecutarse primero. Por tanto, cuando desconoce el contexto prioritario, MapForce selecciona automáticamente la primera tabla o el primer origen de datos.

Solución:
Decida en qué tabla o en qué origen de datos debe realizarse la búsqueda primero o qué tabla/origen de datos debe recorrerse primero y después establezca el contexto prioritario en su conector.

Más abajo puede ver una imagen que muestra una versión simplificada del archivo **DB_CompletePO.mfd** de la carpeta **...\MapForceExamples**.

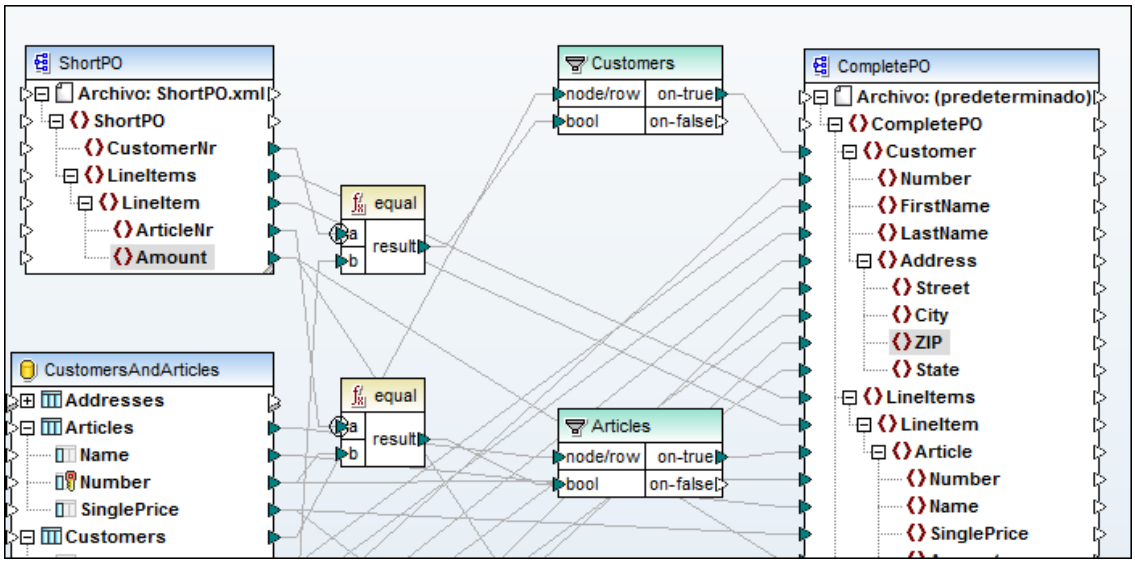
Observe que esta asignación tiene varios componentes de origen. **ShortPO** es un esquema que tiene asociado un archivo XML de instancia. Por su parte, el componente **CustomersAndArticles** es una base de datos. Los datos de ambos componentes se asignan al archivo/esquema XML **CompletePO**. El nodo de contexto prioritario aparece señalado con un círculo.

Al designar el parámetro **a** de la función **equal** como **contexto prioritario**, conseguimos esto:

- El nodo **CustomerNr** del esquema de origen **ShortPO** se compara con el elemento **Number** de la BD de origen.
- El nodo **CustomerNr** se convierte en contexto prioritario y se coloca en el parámetro **a** de la función **equal**.
- En la base de datos **CustomersAndArticles** se busca (una vez) el mismo número. El parámetro **b** contiene el elemento **Number** de la base de datos.
- Si se encuentra el número, entonces el resultado se pasa al parámetro **bool** del componente de filtrado **Customers**.
- El parámetro **node/row** del filtro **Customers** pasa a **on-true** cuando el parámetro **bool**

es **true**, es decir, cuando se encontró el mismo número.

- Los demás datos de **Customer** (como **Number**, **FirstName** y **LastName**) se pasan y se conectan a los elementos correspondientes del esquema de destino.

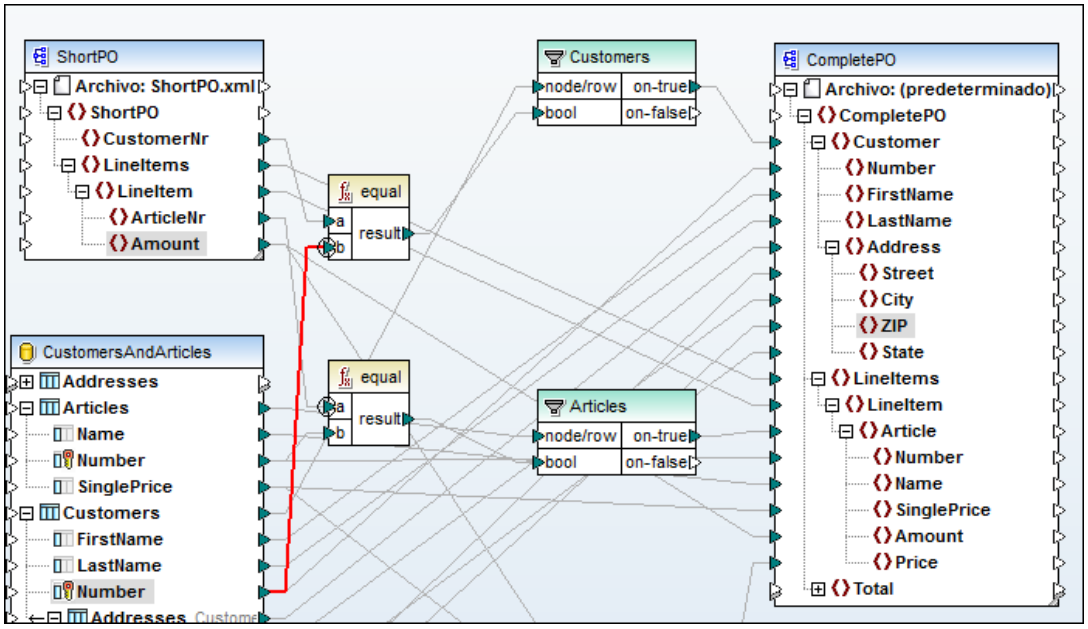


Esto significa que la búsqueda se realiza una sola vez en la BD por cada **CustomerNr** suministrado por **ShortPO**.

Al designar el parámetro **b** de la función **equal** como **contexto prioritario**, conseguimos esto:

- MapForce busca el primer **Number** en la base de datos y lo carga en el parámetro **b** de la función **equal**
- Se compara con el nodo **CustomerNr** del parámetro **a**, procedente del esquema de origen **ShortPO**
- Si no son iguales, la búsqueda recorre todos los nodos **CustomerNr** del esquema **ShortPO**
- MapForce busca el siguiente **Number** en la base de datos y lo carga en el parámetro **b** y lo compara con **a**
- Se itera en cada **Number** de la base de datos para intentar buscar ese número en el esquema **ShortPO**.

 © 2018 Altova GmbH

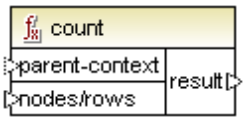


Esto significa que se genera una consulta de BD por cada **`Number`** y el resultado se compara después con cada elemento **`CustomerNr`** de **ShortPO**.

Contexto prioritario en funciones definidas por el usuario
Si una función definida por el usuario se define como de tipo *inline* (configuración predeterminada), entonces no se puede definir un contexto prioritario en ninguno de los parámetros de la función definida por el usuario. La función definida por el usuario puede tener otras funciones definidas por el usuario no *inline* que tenga contextos prioritarios.

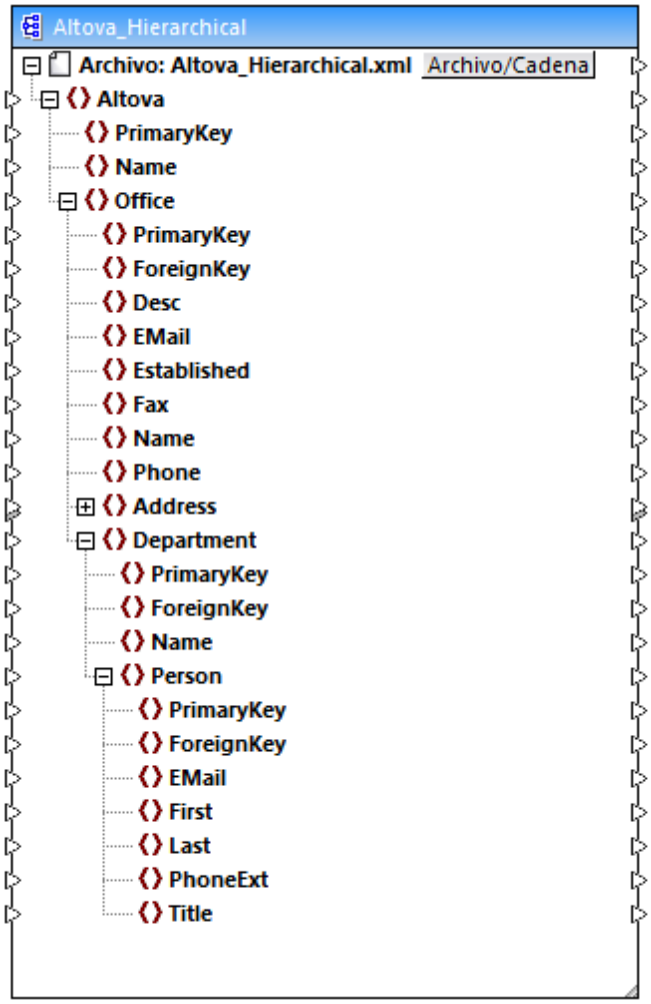
5.15.3 Reemplazar el contexto de la asignación

En algunas asignaciones, para poder conseguir el resultado deseado, puede ser necesario reemplazar el contexto de la asignación. Por este motivo la estructura de algunos componentes ofrece un elemento opcional llamado `parent-context` que permite determinar el contexto de la asignación. Por ejemplo, las funciones de agregado, las variables y los componentes de combinación cuentan con el elemento opcional `parent-context`.



Función de agregado con el elemento opcional parent-context

Para entender lo importante que es el contexto de la asignación añadiremos a la asignación un archivo XML que contiene nodos anidados con varios niveles. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el archivo **<Documentos>\Altova \MapForce2018\MapForceExamples\Altova_Hierarchical.xml**.



Altova_Hierarchical.xml

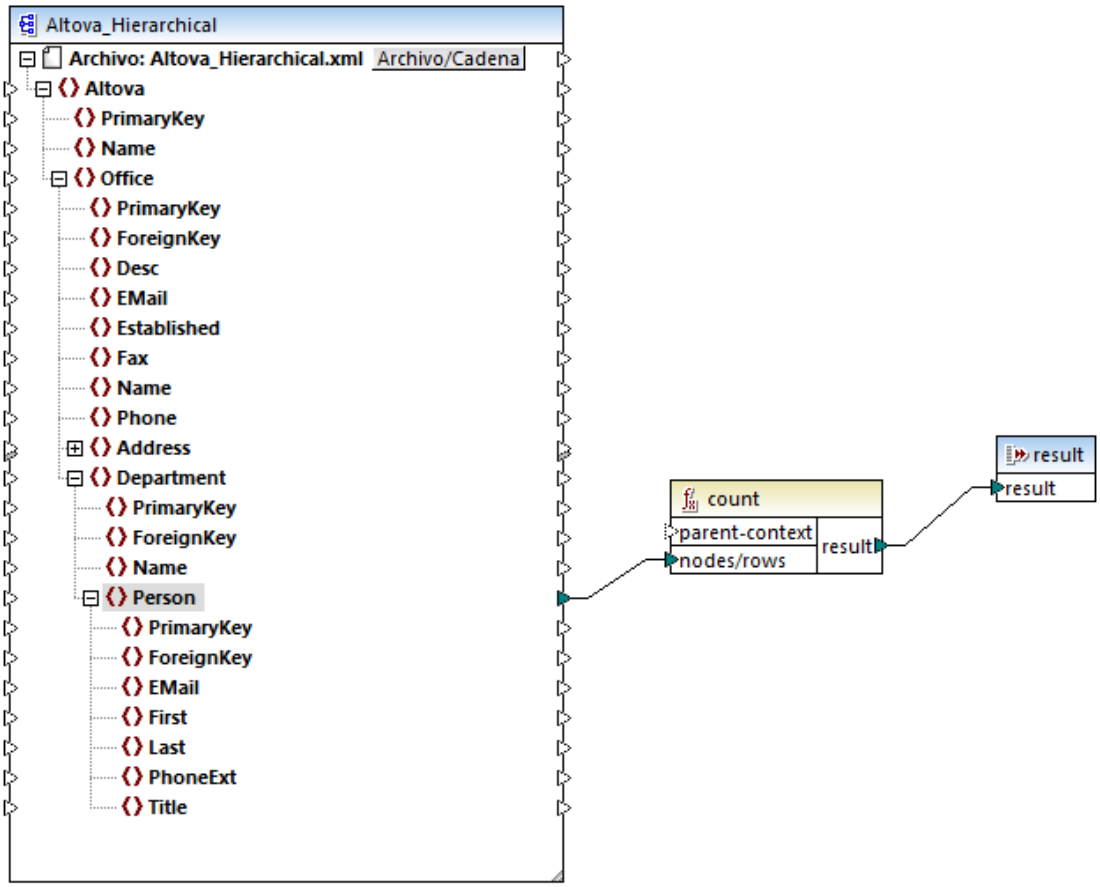
Lo más importante es observar que, en el archivo XML, el nodo primario `Office` contiene varios nodos `Department` que a su vez contienen varios nodos `Person`. Si abrimos el archivo XML en un editor XML podremos ver la distribución de empleados por oficina y por departamento:

Oficina	Departamento	Nº de empleados
Nanonull, Inc.	Administration	3
	Marketing	2
	Engineering	6
	IT & Technical Support	4
Nanonull Partners, Inc.	Administration	2

© 2018 Altova GmbH

Oficina	Departamento	Nº de empleados
	Marketing	1
	IT & Technical Support	3

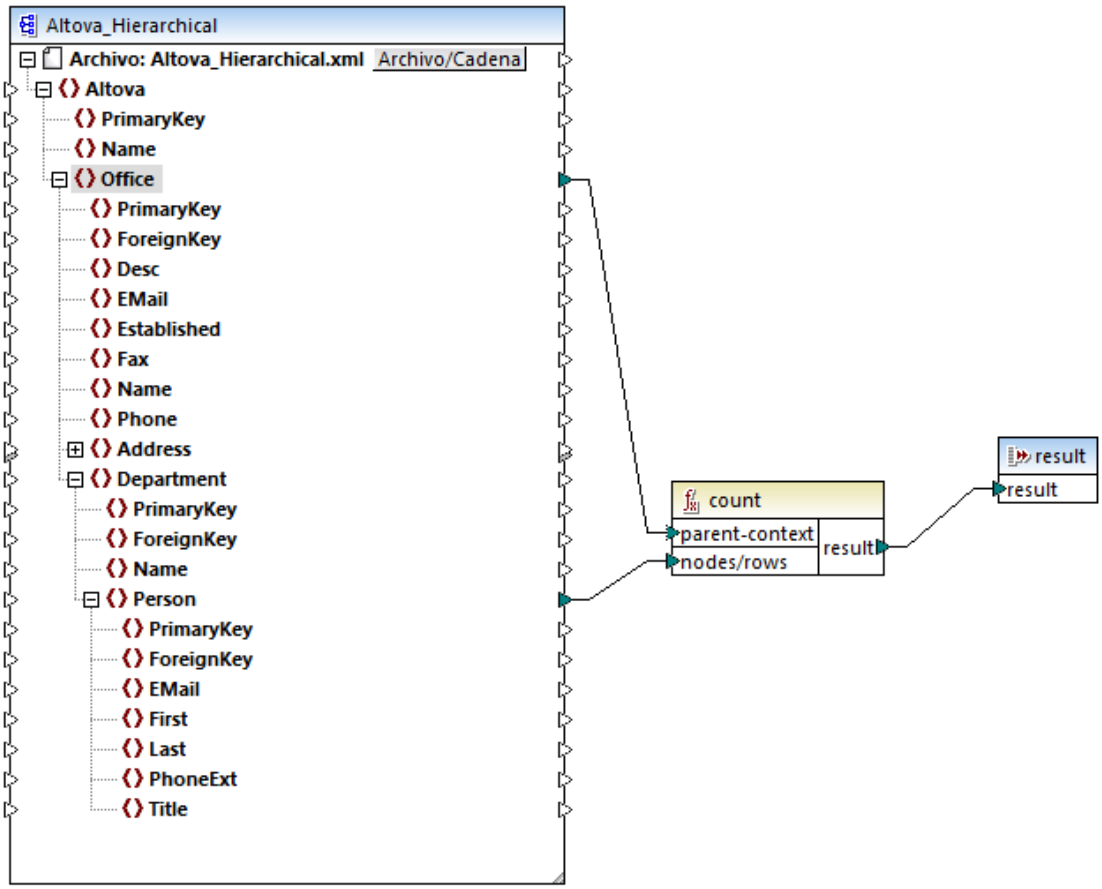
Ahora imaginemos que la asignación debe contar todos los empleados de todos lo departamentos. Para conseguir este objetivo podemos añadir la función **count** de la biblioteca de funciones core | aggregate functions y crear estas asignaciones de datos:



Si consultamos la vista previa del resultado en este momento, veremos que el resultado es `21`, es decir, el número total de empleados de todos los departamentos juntos. Observe que la función **count** tiene un elemento opcional llamado `parent-context` que no hemos utilizado por ahora. Como resultado, el contexto primario de la función **count** es el nodo raíz predeterminado del componente de origen (que, en este caso, es el elemento `Altova`). Esto significa que se tienen en cuenta todos los empleados de todos los departamentos para el ámbito de la función **count**. Este es el funcionamiento predeterminado del contexto de la asignación (*véase Reglas y estrategias de asignación de datos*) y suele ser el funcionamiento adecuado para la mayoría de asignaciones.

Sin embargo, en MapForce se puede reemplazar el contexto predeterminado de la asignación.

Esto se hace creando una conexión entre el nodo `Office` y el elemento opcional `parent-context` de la función **`count`**.

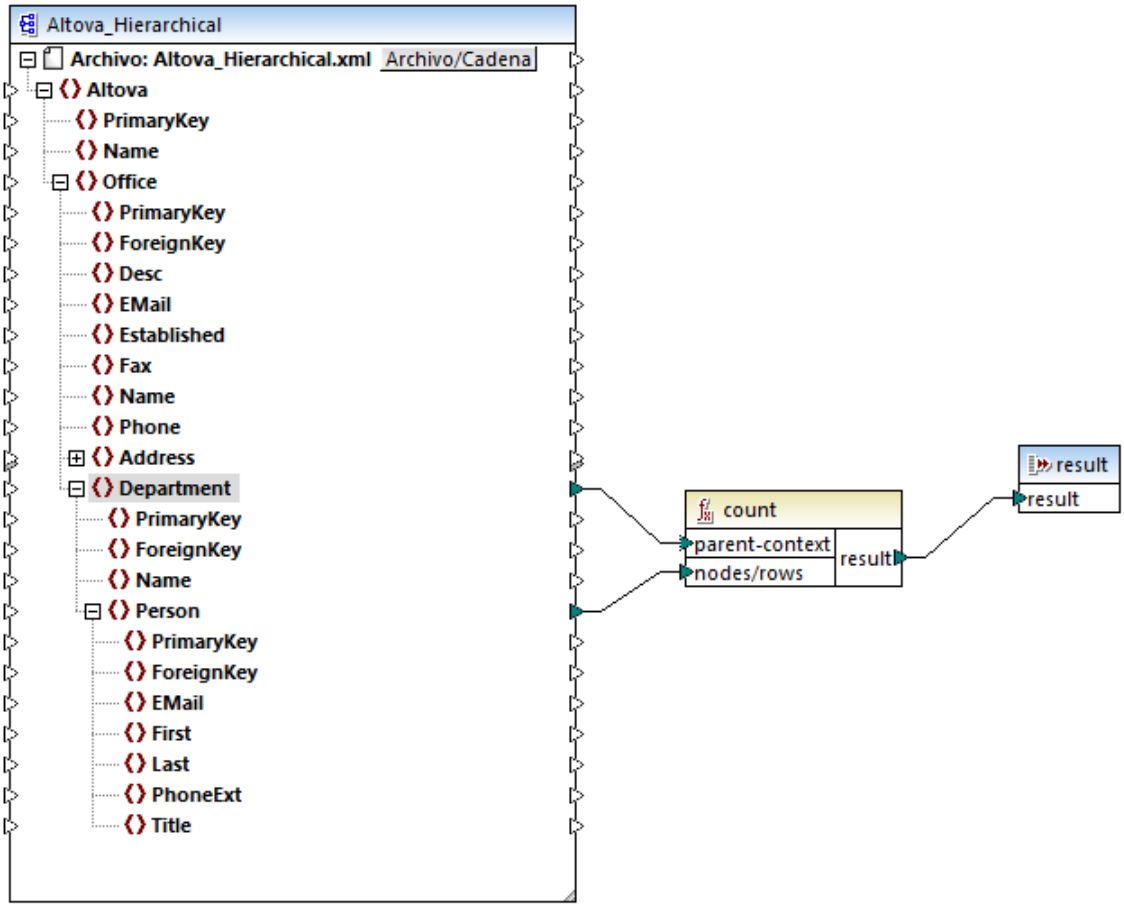


Al crear esta conexión estamos indicando que la asignación debe recorrer los registros de empleados *en el contexto de cada oficina*. Es decir, si ahora consultamos la vista previa del resultado de la asignación, el resultado será `15*`. Se trata del número de empleados de la primera oficina ("Nanonull, Inc."). El motivo es que esta vez los nodos `Person` se contaron dos veces (una vez por cada oficina). El número de empleados de las oficinas es 15 y 6 respectivamente, pero solamente se devuelve el primer resultado (porque la función no puede devolver una secuencia de valores sino un solo valor).

** Siempre y cuando el lenguaje de destino elegido para la asignación no sea XSLT 1.0.*

Puede seguir modificando la asignación y su contexto. Por ejemplo, ahora cambiamos el contexto por `Department` (*imagen siguiente*). Esta vez los registros de empleados se contarán en el contexto de cada departamento (es decir, 7 veces, tantas como departamentos). Una vez más, solamente se devuelve el primero de los resultados. Así que el resultado de la asignación es `3`, que es el número de empleados del primer departamento de la primera oficina.

 © 2018 Altova GmbH



Aunque esta asignación no es de gran utilidad, nos sirve para explicar cómo influye el elemento opcional `parent-context` en el resultado de la asignación. Ahora puede tener esto en cuenta y reemplazar el contexto primario `parent-context` en sus asignaciones (p. ej. en las que contienen variables o componentes de combinación.

Temas relacionados: *Ejemplo: crear grupos y subgrupos de registros*

Altova MapForce 2018 Enterprise Edition

Depuración de asignaciones

6 Depuración de asignaciones

MapForce viene con un depurador de asignaciones integrado para el lenguaje de transformación integrado. El depurador de asignaciones le permitirá:

- Ver y analizar los valores que produce la asignación por cada uno de los conectores.
- Resaltar en la asignación el contexto (conjunto de nodos) responsable de producir un valor determinado.
- Ejecutar paso a paso una asignación de datos para ver cómo se procesa o calcula cada valor en tiempo real y generar una vista previa del resultado de la asignación a medida que se produce.
- Establecer puntos de interrupción en los que se debe detener la ejecución de la asignación y ver los valores que se están procesando en ese momento.
- Ver el historial de valores procesados por un conector desde que se empezó a ejecutar la asignación hasta la posición actual de la ejecución.

El depurador de asignaciones solamente está disponible si el lenguaje de transformación de la asignación es BUILT-IN. Si comienza a depurar una asignación configurada para otro tipo de lenguaje, MapForce solicitará que cambie el lenguaje por BUILT-IN. También puede convertir una asignación al lenguaje BUILT-IN seleccionando el comando de menú **Resultados | Motor de ejecución integrado**. En ambos casos la conversión a BUILT-IN solo será posible si la asignación carece de componentes incompatibles con el lenguaje BUILT-IN (como las funciones XSLT por ejemplo).

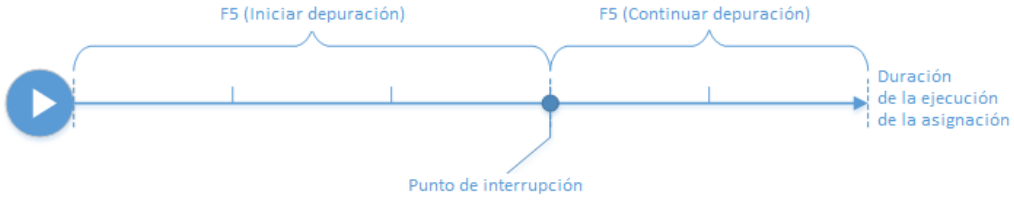
El depurador de MapForce se diferencia de los depuradores tradicionales en que no recorre el código de programa línea por línea (ya que en MapForce no es necesario escribir código). Por el contrario, el depurador de MapForce expone los resultados del código generado a partir de las asignaciones diseñadas. Concretamente, el depurador registra los valores que se pasan entre los componentes de la asignación a través de los conectores de entrada y salida. Los valores registrados se pueden analizar después en la asignación directamente o en ventanas especiales.

A continuación explicamos algunos casos de uso para trabajar con el depurador de asignaciones.

Depuración con puntos de interrupción

Cuando necesite detener la ejecución de la depuración en una posición determinada de la asignación, podrá establecer puntos de interrupción (tal y como se hace en los entornos de desarrollo tradicionales). La diferencia es que en MapForce los puntos de interrupción no se añaden a una línea de código sino a un conector de entrada o salida de un componente de la asignación. También puede añadir condiciones a los puntos de interrupción (lo cual puede ser de gran ayuda si solamente queremos que la ejecución se detenga solamente si se cumple cierta condición).

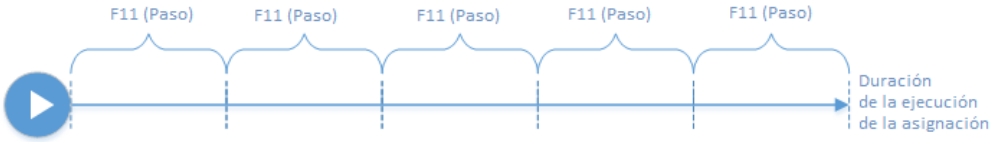
© 2018 Altova GmbH



Puede definir puntos de interrupción en los conectores pertinentes y ejecutar la asignación hasta que se encuentre el primer punto de interrupción y después continuar hasta el siguiente y así sucesivamente. De este modo podrá analizar el contexto de la asignación y los valores asociados a los conectores elegidos. También podrá acelerar o ralentizar la ejecución con ayuda de los comandos **Paso a paso por instrucciones**, **Paso a paso para salir**, **Paso a paso por procedimientos** y **Paso mínimo** que ofrece el depurador. Estos comandos sirven para pasar por alto partes de la asignación o ejecutar partes de la asignación más detalladamente.

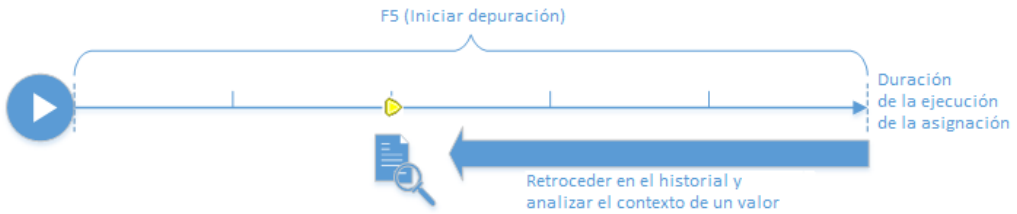
Depuración por pasos

Puede depurar una asignación por pasos y analizar el contexto de la asignación y los valores asociados a cada paso. Al igual que en el tipo de depuración anterior podrá acelerar o ralentizar la ejecución con los comandos **Paso a paso por instrucciones**, **Paso a paso para salir**, **Paso a paso por procedimientos** y **Paso mínimo** que ofrece el depurador.



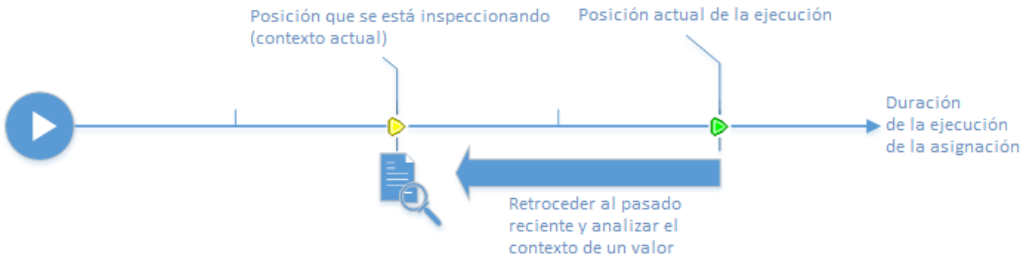
Analizar el registro de valores

Puede configurar MapForce para que recuerde el registro de todos los valores (historial de seguimiento) que fueron procesados por todos lo conectores mientras depura una asignación de datos. El historial de seguimiento puede deshabilitarse ya que puede no ser una opción adecuada para las asignaciones que hagan un uso intensivo de lo datos. Si se deshabilita el historial de seguimiento, podrá analizar el registro de todos los valores procesados por cada conector hasta la posición actual de la ejecución. También puede configurar MapForce para que recree el contexto de asignación asociado con un valor determinado (lo cual puede ayudarle a comprender el motivo por el que se produjo dicho valor).



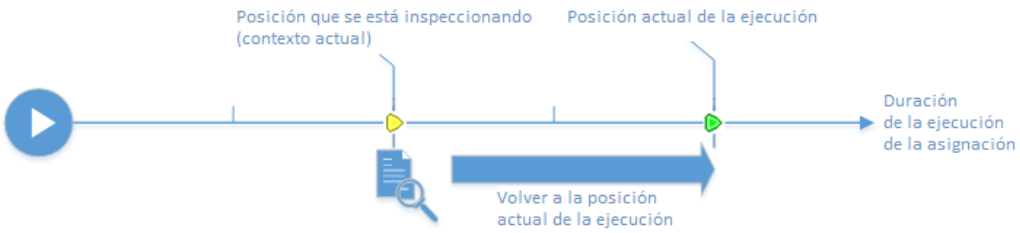
Establecer el contexto en un valor relacionado con la posición actual de la ejecución

Cuando el depurador se encuentra en una posición determinada de la ejecución, se puede analizar el contexto de un valor pasado relacionado con la posición actual de la ejecución (como si se retrocediera un poco en la ejecución):



El sentido del contexto es explicar por qué se calcula un valor. En otras palabras, describe cómo se generó un valor determinado de la asignación. El contexto suele ser la posición actual de la ejecución, aunque también puede ser un contexto del pasado reciente que MapForce permite definir. Cuando el contexto se establece en un valor determinado, MapForce resalta en la asignación directamente los nodos que son relevantes, ofrece consejos junto a los conectores y expone información adicional en las ventanas del depurador (ventanas Valores, Contexto y Puntos de interrupción).

Cuando termine de inspeccionar un contexto de la asignación que no sea igual a la posición actual de la ejecución, podrá restablecer el contexto en la posición actual de la ejecución:



Restricciones

- Cuando MapForce ejecuta una asignación, a veces optimiza el código de forma interna (por ejemplo, almacenando datos en caché o calculando resultados intermedios en puntos aleatorios). Esto puede provocar que sea imposible acceder a ciertos conectores (y, por tanto, a ciertos puntos de interrupción) durante la depuración. Cuando esto ocurre,

 © 2018 Altova GmbH

MapForce muestra una notificación. Nota: las optimizaciones de código de MapForce (y por consiguiente el comportamiento que expone el depurador) pueden diferir en las diferentes versiones del software aunque el resultado de la asignación sea el mismo.

- El depurador puede depurar la generación resultante para cada componente de destino uno a uno. Si la asignación incluye varios componentes de destino, deberá seleccionar qué componente de destino se ejecuta en el depurador.
- Actualmente la depuración no es compatible con las acciones de tabla de BD (como "Insertar todo", "Actualizar si", etc.) de los componentes de base de datos.
- No se pueden establecer puntos de interrupción en estas entidades: constantes, la función **`core | position`**, elementos descendientes de las conexiones de copia total y parámetros de funciones definidas por el usuario insertadas.

6.1 Preparar la depuración

Preparar la depuración puede ser fundamental en asignaciones masivas de datos que puedan necesitar mucha memoria de sistema a la hora de ejecutarse. Nos referimos a las asignaciones de datos que procesan archivos de entrada o salida de gran tamaño o que recorren reiteradamente colecciones de datos de gran tamaño.

Para completar la depuración más rápido y reducir el uso de memoria, recomendamos seguir estos pasos antes de iniciar la depuración:

- Si se trata de una asignación compleja, elimine o desconecte las partes de la asignación que no necesita depurar.
- Si la asignación utiliza archivos de entrada de gran tamaño, sustitúyalos por archivos de menor tamaño.
- Asegúrese de deshabilitar la opción **Guardar historial de seguimiento completo** (*véase* *Configurar el depurador*).

Además, para garantizar que el depurador analiza los resultados correctos deberá repasar algunos aspectos:

- Si la asignación tiene varios componentes de destino, seleccione el componente de destino que desea depurar haciendo clic en el botón **Vista previa** ().
- Si la asignación es una asignación en cadena (*véase* *Asignaciones en cadena*), desactive el botón **Paso a través** () del componente intermedio. MapForce no permite depurar componentes de paso a través.

Por último, tiene la opción de agregar puntos de interrupción en conectores importantes cuyos valores desee analizar. El depurador se detendrá en estos conectores a medida que avance por la asignación (*véase* *Agregar y quitar puntos de interrupción*).

© 2018 Altova GmbH

6.2 Comandos del depurador

Los comandos del depurador están disponibles:

- En el menú **Depuración**
- Como teclas de acceso rápido
- En la barra de herramientas **Depuración**.

Comando	Tecla de acceso rápido	Botón de la barra de herramientas	Descripción
Depuración | Iniciar depuración	**F5**		Inicia o continúa la depuración hasta llegar a un punto de interrupción o hasta que finaliza la asignación.
Depuración | Detener depuración	**Mayús + F5**		Detiene la depuración. Esto comando cierra el modo depurador y rehabilita el modo estándar de MapForce.
Depuración | Paso a paso por instrucciones	**F11**		Ejecuta la asignación hasta que se termina un solo paso en cualquier posición de la asignación. En el depurador el término ***paso*** se usa para designar a un grupo lógico de cálculos dependientes que normalmente producen un solo elemento de una secuencia. Dependiendo del contexto de la asignación, este comando puede leerse como "ir a la izquierda / ir al secundario de destino / ir al primario de origen".
Depuración | Paso a paso por procedimientos	**F10**		Continúa con la ejecución hasta que finaliza el paso actual (o se termina otra vez para otro elemento de la secuencia) o hasta que finaliza un paso no relacionado. Este comando se salta los cálculos que son entradas del paso actual.
Depuración | Paso a paso para salir	**Mayús + F11**		Continúa con la ejecución hasta que el resultado del paso actual se consume o hasta que se ejecuta un paso que no es una entrada ni un secundario del consumo. Este comando sale del cálculo actual. Dependiendo del contexto de la

Comando	Tecla de acceso rápido	Botón de la barra de herramientas	Descripción
			asignación, este comando puede leerse como "ir a la derecha / ir al primario de destino / ir al secundario de origen".
Depuración | Paso mínimo	**Ctrl + F11**		Continúa con la ejecución hasta que se produce o consume un valor. Este comando subdivide un paso y se detiene dos veces por cada conexión: cuando su origen produce un valor y cuando su destino lo consume. MapForce no siempre calcula los valores en el orden que sugiere la asignación, así que los eventos de producción y consumo no siempre van de la mano.



Barra de herramientas Depuración

© 2018 Altova GmbH

6.3 Información sobre el modo depurador

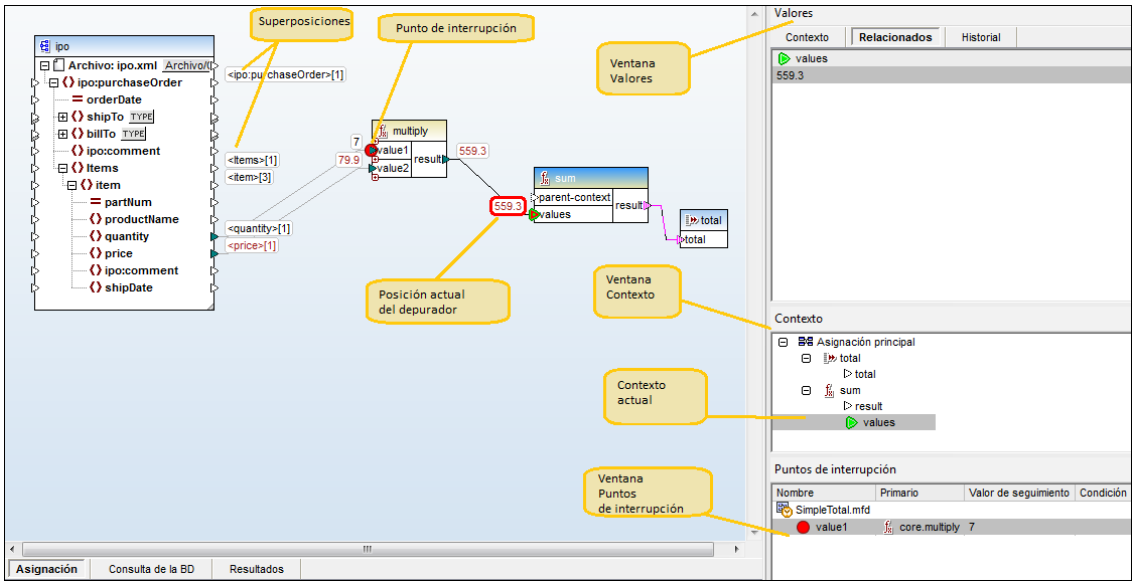
Cuando se inicia la depuración (pulsando **F5**, **F11** o **Ctrl + F11**), MapForce ejecuta la asignación en modo depurador.

> Mientras MapForce esté en modo depurador, la asignación es de solo lectura. Aunque los componentes se pueden mover por el área de asignación, la mayoría de los comandos estarán deshabilitados. Entre ellos los comandos de validación e implementación, de generación de código y documentación y los comandos para agregar componentes nuevos al área de asignación o para volver a cargar los componentes actuales.

El modo depurador permite analizar el contexto responsable de producir un valor determinado. La información que produce el depurador aparece en la asignación directamente, pero también en las ventanas Valores, Contexto y Puntos de interrupción. Estas tres ventanas se abren por defecto nada más iniciar la depuración y desaparecen cuando la depuración se detiene.

MapForce estará en modo depurador (y la asignación será de solo lectura) hasta que se detenga la depuración, pulsando **Mayús + F5** (o haciendo clic en el botón **Detener depuración** de la barra de herramientas).

En la siguiente imagen puede ver una asignación que se está depurando por pasos (pulsando **F11** para avanzar por los pasos de la asignación). La imagen corresponde a la asignación de muestra **SimpleTotal.mfd**, disponible en el directorio **<Documentos>\Altova\MapForce2018\MapForceExamples**.



Entorno de desarrollo MapForce en modo depurador

A continuación describimos las características de la interfaz del depurador de MapForce.

Panel de asignación

Durante la depuración el panel de asignación incluye información adicional:

- Superposiciones de datos (ver más abajo) que muestran el valor actual y valores relacionados junto a cada conector.
- El contexto actual (la estructura que aparece en la ventana Contexto) se resalta de la siguiente manera:
  - Los conectores que están dentro de contexto son de rayas magenta ( ).
  - Los conectores que están en contexto ambiguo son de puntos magenta ( ).
  - Las conexiones que están dentro de contexto son de rayas magenta.
  - Las conexiones que están en contexto ambiguo son de rayas magenta más claro.
- La ubicación actual de la ejecución aparece marcada con un icono de conector verde ( ).

Superposiciones de datos

Los valores que procesa cada conector aparecen representados en forma de superposiciones de datos (rectángulos pequeños) junto al conector correspondiente. Una superposición de datos que está seleccionada tiene el borde sólido y rojo. Los valores que se modificaran desde el último paso aparecen en rojo oscuro. Para los nodos con contenido simple la superposición de datos combina dos valores: el nombre del nodo y el valor. Si el nombre de nodo se recorrió varias veces antes de la posición actual de la ejecución, el índice de la iteración actual se indica con un número que aparece entre corchetes.

Las superposiciones de datos funcionan de la siguiente manera:

- Si se pasa el puntero por encima de una superposición de datos, esta pasa a primer plano temporalmente. Si se hace clic en la superposición, esta pasa a primer plano permanentemente y además se selecciona el conector correspondiente.
- Pueden arrastrarse y colocarse en otra posición.
- Se mueven cuando se mueve un componente. Por tanto, si aparecen apiladas porque los componentes están demasiado cerca unos de otros, arrastre los componentes hasta otra posición y las superposiciones se moverán junto con su componente.
- Si se hace clic en una superposición de datos, su valor aparece en la ventana Valores.
- Si se hace clic en un conector también se selecciona su superposición de datos.

Puntos de interrupción

Los puntos de interrupción son acontecimientos señalados en los que la asignación debe detenerse durante la ejecución en modo depurador. Este término es utilizado en otros entornos de desarrollo integrado. Pero si en los entornos de desarrollo los puntos de interrupción se añaden en líneas de código, en MapForce se pueden añadir en conectores de entrada o salida (los iconos en forma de triángulo situados en los extremos de las conexiones). En el panel de asignación los puntos de interrupción se representan en forma de círculos rojos. Además, los puntos de interrupción definidos aparecen en la ventana Puntos de interrupción. Consulte el apartado Agregar y quitar puntos de interrupción para obtener más información.

© 2018 Altova GmbH

Posición actual del depurador

El triángulo verde () indica la posición actual del depurador. Esta posición puede ser un conector de entrada o salida de cualquier componente.

El valor que se esté procesando en ese momento también aparece en la pestaña *Contexto* de la ventana Valores.

El conjunto de conexiones y conectores de rayas magenta indican el contexto actual de la asignación. La misma información aparece representada de forma jerárquica en la ventana Contexto (véase el apartado Ventana Contexto).

Cuando defina a mano el contexto de un valor, la posición actual del depurador será una posición del pasado en relación a la posición más actual de la ejecución. Para ayudarle a distinguir entre la posición más actual de la ejecución y la posición del pasado, el conector de la *posición actual* aparece con estos colores en la interfaz del depurador.

	El triángulo verde es *el presente*. Indica la posición actual de la ejecución (*véase Ver el valor actual de un conector*).
	El triángulo amarillo es *el pasado*. Indica que se trata de un conector del pasado, relativo a la posición actual de la ejecución. Esto puede ocurrir después de definirse un contexto a mano (*véase Establecer un contexto en un valor*).

Ventana Valores

La ventana Valores ofrece información sobre los valores que procesa la asignación. Permite ver qué procesa la asignación en la posición actual de la ejecución o en un contexto determinado que se puede establecer a mano. Consulte el apartado Ventana Valores para obtener más información.

Ventana Contexto

La ventana Contexto ofrece una vista jerárquica del conjunto de nodos y funciones que son relevantes para la posición actual del depurador. Consulte el apartado Ventana Contexto para obtener más información.

Ventana Puntos de interrupción

La ventana Puntos de interrupción muestra la lista de puntos de interrupción que se crearon desde que se inició MapForce. Si definió puntos de interrupción en varias asignaciones diferentes, todos ellos aparecerán en esta ventana. Consulte el apartado Ventana Puntos de interrupción para obtener más información.

6.4 Agregar y quitar puntos de interrupción

Los puntos de interrupción son acontecimientos señalados en los que la asignación se debe interrumpir durante la ejecución en modo depurador. Los puntos de interrupción creados por el usuario se almacenan de forma global para todas la asignaciones y pueden consultarse en la ventana Puntos de interrupción. Los puntos de interrupción definidos son válidos hasta que el usuario los elimina o hasta que se cierra MapForce

Nota: no se pueden agregar puntos de interrupción en estas entidades: constantes, funciones `core | position`, elementos descendientes de conexiones de copia total y parámetros de funciones definidas por el usuario inline.

Los puntos de interrupción pueden ser simples o condicionales. Los simples detienen la ejecución de la asignación sin condiciones. Los condicionales detienen la ejecución de la asignación cuando se cumple una condición asignada previamente al punto de interrupción. Las condiciones pueden ser funciones de la biblioteca integrada de MapForce a las que se aportan valores personalizados. En otras palabras, si la condición devuelve el valor `true`, el punto de interrupción detendrá la ejecución de la asignación.

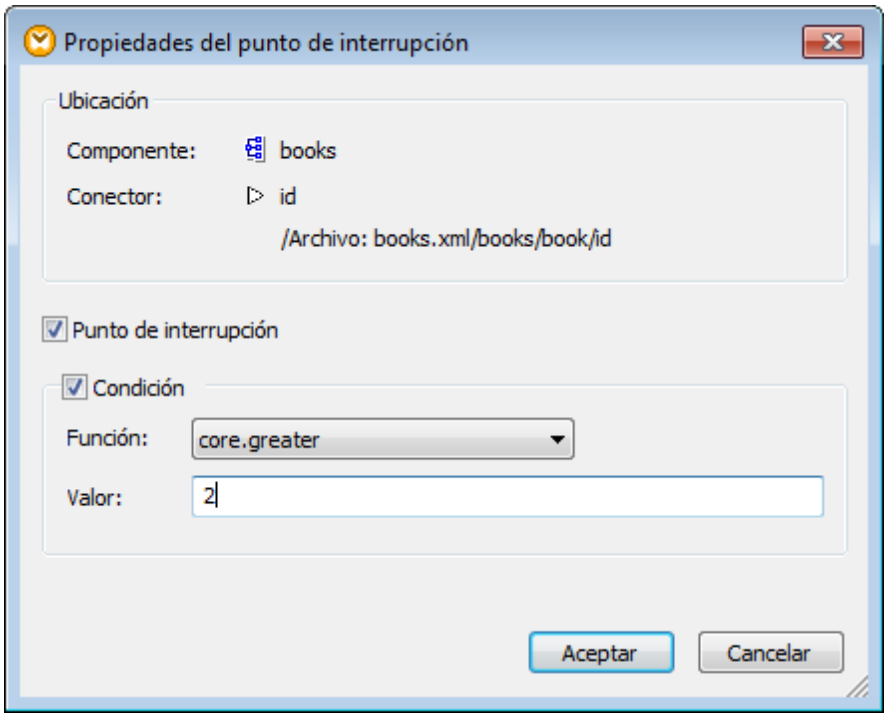
Hay dos maneras de crear puntos de interrupción simples:

- Haciendo clic con el botón derecho en un conector de entrada o salida (los triángulos situados a la izquierda o derecha de un componente) y seleccionando **Punto de interrupción del depurador** en el menú contextual.
- Haciendo clic en un conector de entrada o salida y pulsando **F9**.

Para crear un punto de interrupción condicional:

1. Haga clic con el botón derecho en un conector y seleccione **Propiedades del punto de interrupción** en el menú contextual.

© 2018 Altova GmbH



2. En el cuadro de diálogo que aparece marque las casillas *Punto de interrupción* y *Condición*.
3. Seleccione la función pertinente en el cuadro combinado *Función* e introduzca el valor de la función (si procede). Por ejemplo, en la imagen anterior se definió un punto de interrupción que detendrá la ejecución de la asignación si el valor que pasa por el punto es superior a 2.

Si el tipo de datos del conector donde se añade el punto de interrupción condicional no coincide con los tipos esperados por la función, MapForce intentará convertir el tipo de datos automáticamente. Si la conversión automática no es posible, la ejecución de la asignación fallará. Para evitarlo, asegúrese de usar tipos de datos compatibles. Por ejemplo, la función `core.starts-with` espera un valor de cadena, de modo que el conector del punto de interrupción debería tener el mismo tipo.

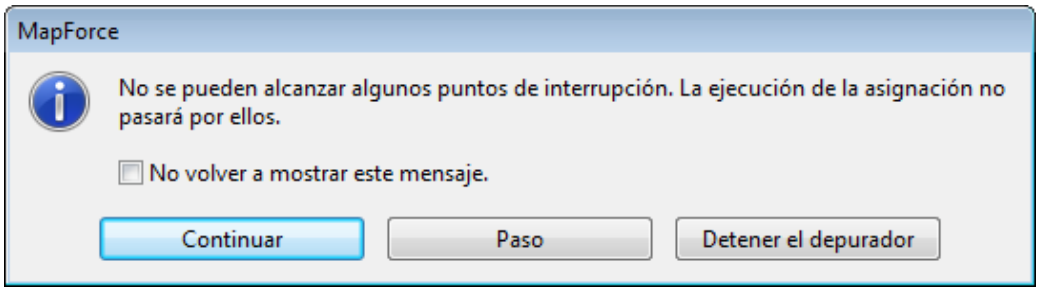
Quitar puntos de interrupción

Para quitar un punto de interrupción haga clic con el botón derecho en el conector que lo contiene y seleccione **Punto de interrupción del depurador**. Otra opción es hacer clic en el conector de entrada o salida donde está definido el punto de interrupción y pulsar **F9**.

También puede eliminar puntos de interrupción desde la ventana Puntos de interrupción.

Puntos de interrupción inalcanzables

En algunos casos MapForce emite este mensaje:



Este mensaje indica que el depurador no puede alcanzar algunos puntos de interrupción. Esto puede deberse a dos motivos distintos:

- Se definió un punto de interrupción en un conector que no participa en la asignación.
- MapForce no puede alcanzar el punto de interrupción debido a optimizaciones de la ejecución (consulte la información sobre restricciones).

Haga clic en Continuar para proceder al siguiente punto de interrupción (o para llegar al final de la ejecución de depuración). Haga clic en **Paso** para iniciar la depuración por pasos.

Este tipo de mensajes se puede deshabilitar marcando la casilla *No volver a mostrar este mensaje* o siguiendo estas instrucciones:

1. Haga clic en el comando de menú **Herramientas | Opciones**.
2. Haga clic en *Mensajes*.
3. Desactive la casilla *Informar sobre puntos de interrupción que no se puedan alcanzar*.

© 2018 Altova GmbH

6.5 Ventana Valores

La ventana Valores ofrece información sobre los valores que procesa la asignación cuando MapForce está en modo depurador. La información que aparece en esta ventana dependerá de la posición actual del depurador y de los elementos de la interfaz en los que haga clic el usuario. Estas son las pestañas de la ventana Valores:

Pestaña *Contexto*

La pestaña *Contexto* muestra el valor que se está procesando actualmente (el mismo valor cuyo contexto aparece en la ventana Contexto). Se trata del valor que está en la posición actual de la ejecución del depurador o del valor de un conector que se procesó en el pasado. Para ayudarle a distinguir entre estos dos tipos de valores, MapForce usa iconos de colores distintos:

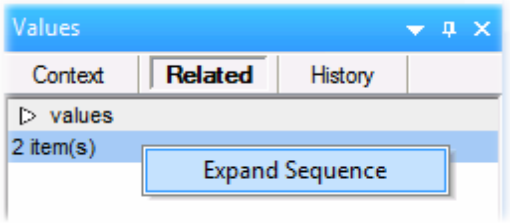
	El triángulo verde es *el presente*. Indica la posición actual de la ejecución (*véase* *Ver el valor actual de un conector*).
	El triángulo amarillo es *el pasado*. Indica que se trata de un conector del pasado, relativo a la posición actual de la ejecución. Esto puede ocurrir después de definirse un contexto a mano (*véase* *Establecer un contexto en un valor*).

Pestaña *Relacionados*

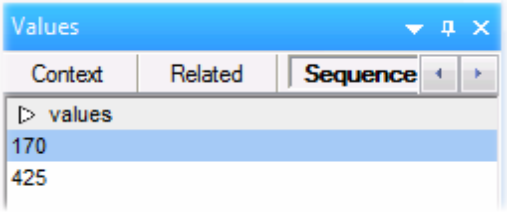
La pestaña *Relacionados* muestra los valores que están relacionados con el valor que se está procesando (o que representan su pasado reciente). En general no necesitará hacer clic en esta pestaña porque MapForce cambia a esta pestaña automáticamente cuando se hace clic en la superposición de datos de un conector que está relacionado con la posición actual de la ejecución del depurador (*véase* *Retroceder en el pasado reciente*).

Pestaña *Secuencia*

Cuando está presente, esta pestaña sirve para acceder a los valores de un conector que procesa una secuencia. Esta pestaña solamente está presente si un conector procesó una secuencia de elementos (por ejemplo, una función de agregado como `sum` o `count`). Cuando se hace clic en la superposición de datos de un conector que procesó una secuencia de elementos, la ventana Valores incluirá la entrada "`n elementos`", siendo `n` el número de elementos que procesa el conector. Para acceder a cada valor, haga doble clic en esta entrada (o clic con el botón derecho y seleccione **Expandir secuencia** en el menú contextual).



Los valores aparecen después en la pestaña *Secuencia*.



Pestaña *Historial*

La pestaña *Historial* muestra valores procesados por un nodo determinado desde que se inició la depuración y hasta que se llegó a la posición actual de la ejecución (*véase* *Ver historial de valores procesados por un conector*).

© 2018 Altova GmbH

6.6 Ventana Contexto

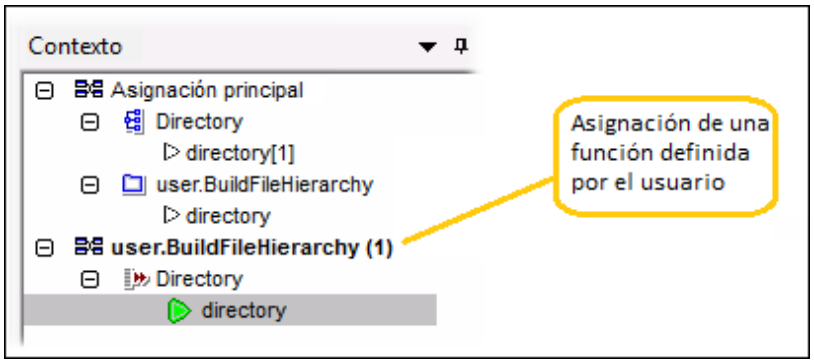
Cuando MapForce está en modo depurador, la ventana Contexto muestra una estructura de conectores que corresponden a la posición actual del depurador. En otras palabras, ofrece el contexto de asignación responsable de producir el valor actual de la asignación.

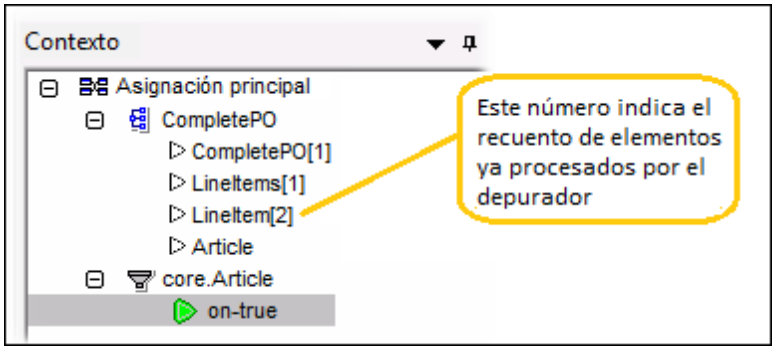
MapForce genera el contexto actual de la siguiente manera:

1. Empezar por el nodo raíz de la estructura de destino.
2. Descender hasta el nodo de destino actual.
3. Desde el nodo de destino actual, avanzar hacia la izquierda por los componentes que conduzcan hasta la posición actual. Estos componentes pueden ser filtros o componentes de ordenación, funciones definidas por el usuario o integradas, variables, etc.

La ventana Contexto ofrece información pero también es un asistente para la navegación. Para seleccionar un nodo determinado de la asignación desde el contexto actual directamente, haga clic con el botón derecho en el nodo en la ventana Contexto y elija **Seleccionar en la asignación**. Con esto puede evitar tener que desplazarse todo el rato por la asignación, si se trata de un diseño muy grande.

La ventana Contexto puede incluir estos iconos y anotaciones:

Icono	Descripción
	Representa la asignación a la que pertenece el contexto. Puede ser la asignación principal o la asignación de una función definida por el usuario. 
▷	Representa un conector. Los nodos de destino procesados hasta el momento aparecen entre corchetes.

Icono	Descripción
	
	Representa el conector actual (la posición más reciente de la ejecución). Se trata del origen del valor actual que aparece en la ventana Valores. En casos muy raros, es posible que un valor calculado se utilice para varios conectores. Cuando esto ocurra, podrían aparecer varios iconos verdes.
	Representa el conector actual cuando el depurador está en una posición del pasado relativo al puesto más reciente de la ejecución. Esto puede ocurrir después de que se establezca el contexto en un valor (*véase* *Establecer el contexto en un valor*).

Además de estos iconos y anotaciones, la ventana Contexto incluye los iconos estándar correspondientes a los tipos de componentes presentes en la asignación de datos.

Ventana Contexto y funciones definidas por el usuario

Si el contexto actual incluye funciones definidas por el usuario, éstas aparecerán también en la ventana Contexto. Recuerde que si el contexto actual es para calcular un valor de entrada de una función definida por el usuario, entonces el contexto se determinará de la siguiente manera:

1. Desde el destino del conector de salida de la función definida por el usuario hasta el conector de entrada de la función definida por el usuario.
2. Desde esa posición hacia la izquierda.

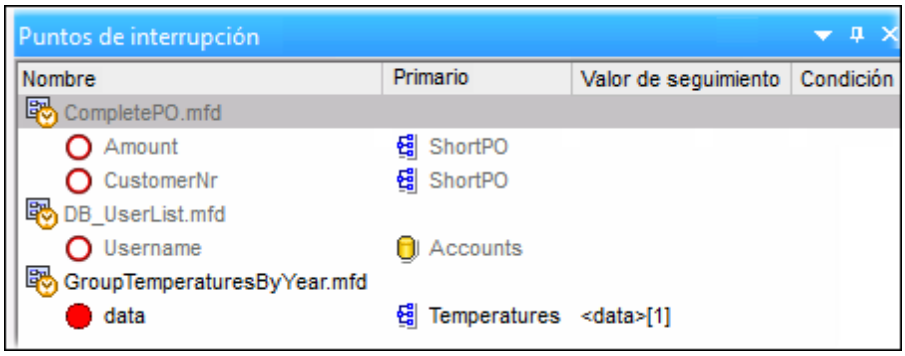
Nota: en el contexto se puede dar la misma función definida por el usuario varias veces. Esto ocurre porque hay varias llamadas a función encadenadas o porque la función definida por el usuario está definida como recursiva.

 © 2018 Altova GmbH

6.7 Ventana Puntos de interrupción

La ventana Puntos de interrupción permite consultar y gestionar los puntos de interrupción de forma global. Esta ventana aparece en cuanto MapForce entra en el modo depurador. Para poder ver la ventana Puntos de interrupción en todo momento, seleccione el comando de menú **Vista | Ventanas del depurador | Puntos de interrupción**.

Esta ventana muestra todos los puntos de interrupción creados desde que se inició MapForce y agrupados según el archivo de asignación al que pertenecen. Mientras esté en ejecución, MapForce recuerda todos los puntos de interrupción asociados con cada asignación y los muestra en la ventana Puntos de interrupción (aunque cierre una asignación, sus puntos de interrupción siguen apareciendo en la ventana). La asignación que se esté depurando en ese momento aparece en el color de texto estándar, mientras que el resto de asignaciones (las que están cerradas o inactivas) aparecen atenuadas.



Haga doble clic en una asignación de la ventana Puntos de interrupción (o en sus puntos de interrupción) para abrirla rápidamente.

Nota importante: cuando cierre o reinicie MapForce, se eliminarán todos los puntos de interrupción.

La información de la ventana Puntos de interrupción está organizada en cuatro columnas:

Columna	Descripción
Nombre	El nombre del nodo al que pertenece el punto de interrupción.
Primario	El nombre del componente de asignación al que pertenece el punto de interrupción.
Valor de seguimiento	El valor que pasa por el conector donde está situado el punto de interrupción. El valor de seguimiento aparece durante la ejecución de depuración.
Condición	Si el punto de interrupción es condicional, muestra la condición que se debe cumplir.

Estos son los iconos que se utilizan para describir los puntos de interrupción en la ventana:

Icono	Descripción.
	Punto de interrupción **activo**: denota un punto de interrupción de la asignación que se está depurando en ese momento.
	Punto de interrupción **inactivo**: denota un punto de interrupción de una asignación que está abierta, pero que no se está depurando en ese momento.
	Punto de interrupción **inaccesible**: denota un punto de interrupción que el depurador no puede alcanzar.
	Punto de interrupción **condicional**: denota un punto de interrupción que tiene adjunta una condición.

Para ver o cambiar las propiedades de un punto de interrupción:

- Haga clic con el botón derecho en el punto de interrupción y elija **Propiedades del punto de interrupción** en el menú contextual.

Para quitar un punto de interrupción:

- Haga clic con el botón derecho en el punto de interrupción y elija **Eliminar punto de interrupción** en el menú contextual.
- Haga clic en el punto de interrupción y pulse **Supr**.

El comando del menú contextual **Eliminar todos los puntos de interrupción** quita todos los puntos de interrupción de la ventana, independientemente de la asignación de datos a la que pertenezcan.

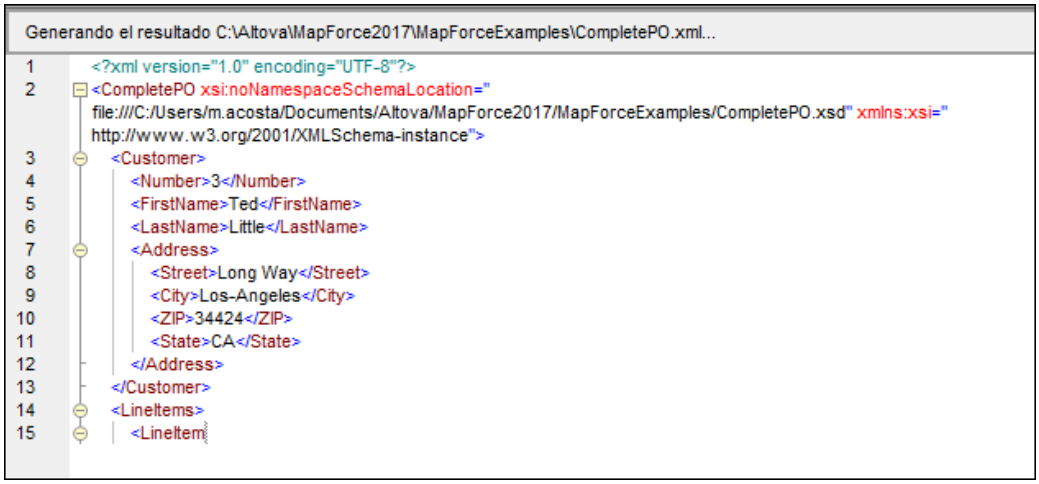
Consulte también el apartado Agregar y quitar puntos de interrupción.

© 2018 Altova GmbH

6.8 Vista previa parcial de resultados generados

Cuando se depura una asignación por pasos o por medio de puntos de interrupción, se puede ver el resultado generado hasta la posición actual del depurador. La vista previa de resultados parcialmente generados es compatible con componentes de destino XML, EDI y de texto plano.

Cuando pulse **F5** (sin haber definido ningún punto de interrupción), MapForce ejecuta por defecto toda la asignación en modo depurador, después cambia a el panel *Resultados* y muestra el resultado final generado. Sin embargo, si define puntos de interrupción o depura la asignación por pasos (con **F11** o **Ctrl + F11**), la ejecución del depurador se interrumpe mientras se genera el resultado de la asignación. Aunque llegados a este punto el resultado de la asignación solamente es parcial, podrá hacer clic en el panel *Resultados* y obtener una vista previa.



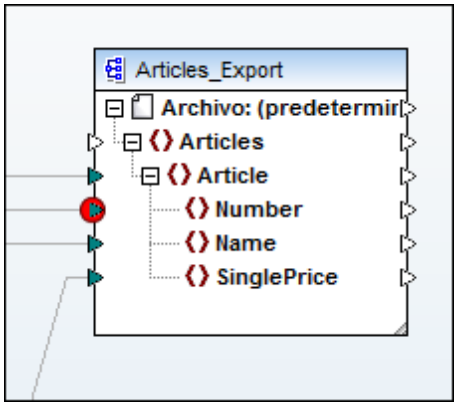
Restricciones

- El nodo de destino que se está calculando en ese momento no siempre aparece en el panel *Resultados*. Por ejemplo, los atributos XML se recopilan de forma interna y se escriben de una sola vez.
- Si el resultado produce varios archivos de salida, solamente se puede obtener una vista previa del archivo que se escribe en ese momento dado. Es decir, está deshabilitada la opción de cambiar a otro archivo de salida.

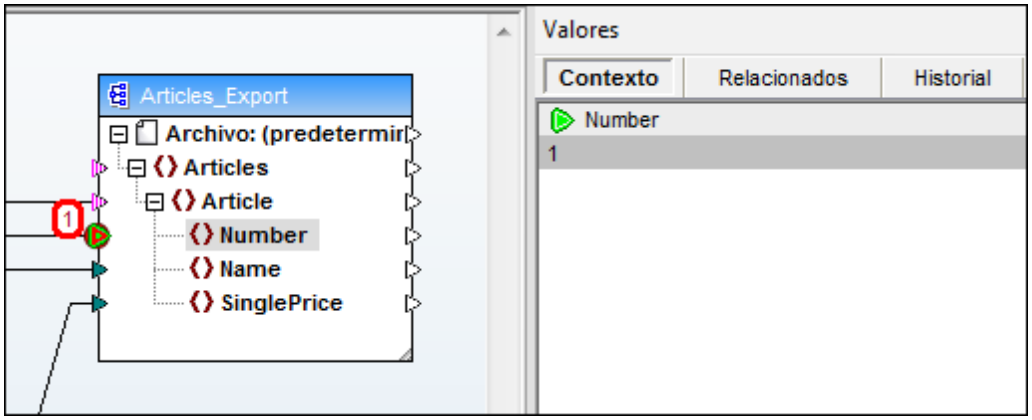
6.9 Ver el valor actual de un conector

Cuando la posición actual de la ejecución del depurador ( ) está en determinado conector (porque se depure por pasos o porque haya definido un punto de interrupción en el conector), el valor que está procesando en ese momento el conector aparece en la pestaña *Contexto* de la ventana Valores. Este valor es el que está a punto de escribirse en el resultado, es decir, *el presente.* También es el valor cuyo contexto aparece en la ventana Contexto.

Para comprender mejor este funcionamiento del depurador abra la asignación **PreserveFormatting.mfd** del directorio **<Documentos>\Altova\MapForce2018\MapForceExamples**. Haga clic en el conector de entrada del nodo `Number` del componente de destino y pulse **F9** para agregar un punto de interrupción en este conector.



Ahora pulse **F5** para iniciar la depuración y observar qué resultados se obtienen.



Tal y como se aprecia en la imagen anterior, la posición actual del depurador (y el punto de interrupción) está en el nodo `Number` del componente de destino. La ventana Valores indica que este nodo procesa el valor "1" (este valor también se resalta con un borde grueso rojo en la asignación).

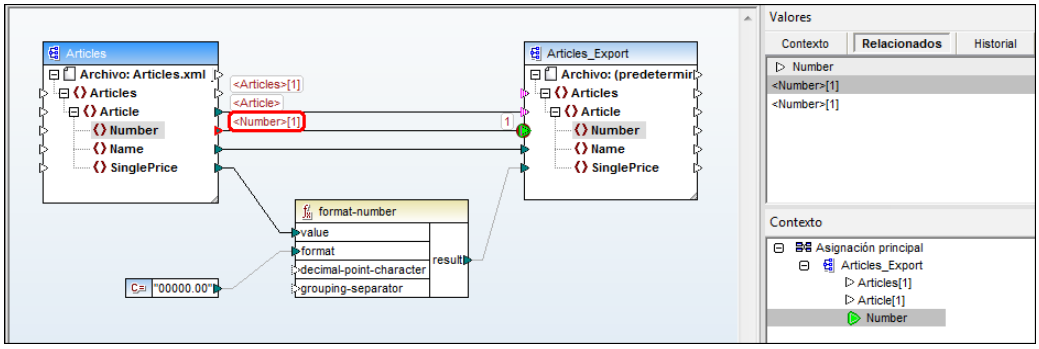
© 2018 Altova GmbH

6.10 Retroceder en el pasado reciente

Cuando se hace clic en una superposición de datos (los pequeños recuadros de la asignación) situada junto a un conector, en la ventana Valores aparece el nombre y, de forma opcional, el valor asociados al conector seleccionado. Ahora lo que importa no es la posición actual del depurador, sino la superposición de datos seleccionada. Esta vista puede entenderse como una manera de retroceder ligeramente en el historial de depuración. Se trata del *pasado reciente* porque la asignación muestra superposiciones de datos para los conectores recientes relacionados con la posición actual del depurador. Cuando se hace clic en una superposición de datos "relacionada", se abre automáticamente la pestaña *Relacionados* de la ventana Valores.

Para comprender mejor este funcionamiento del depurador abra la asignación **PreserveFormatting.mfd** del directorio **<Documentos>\Altova\MapForce2018\MapForceExamples**.

Haga clic en el conector situado junto al nodo `Number` del componente de destino y pulse **F9** para agregar un punto de interrupción. Acto seguido pulse **F5** para iniciar la depuración. Después haga clic en la superposición de datos (el pequeño recuadro) situada junto al nodo `Number` del componente de origen.



El depurador suele recorrer un conector varias veces mientras dure la asignación. Por este motivo, el índice actual de la iteración aparece entre corchetes (en la ventana Valores, en la superposición de datos y en la ventana Contexto): `<Number>[1]`. Además, como el conector lleva un valor, su valor también se representa tras el signo igual: **<Number>[1]=1**. En la ventana Valores aparece el mismo valor en una fila nueva (*imagen anterior*).

Si necesita conocer más información sobre un valor determinado, recuerde que puede recrear el contexto que lo produjo (*véase* *Establecer el contexto en un valor*).

6.11 Ver historial de valores procesados por un conector

Si habilitó la opción *Guardar historial de seguimiento completo* (*véase* *Configurar el depurador*), podrá ver un historial con todos los valores que procesó cada conector (hasta la posición actual de la ejecución).

El historial aparece cuando se hace clic en un conector y después en la pestaña *Historial* de la ventana Valores. Recuerde que esta operación solo es relevante si se trata de conectores que procesaron valores desde el comienzo de la ejecución hasta la posición actual de la depuración.

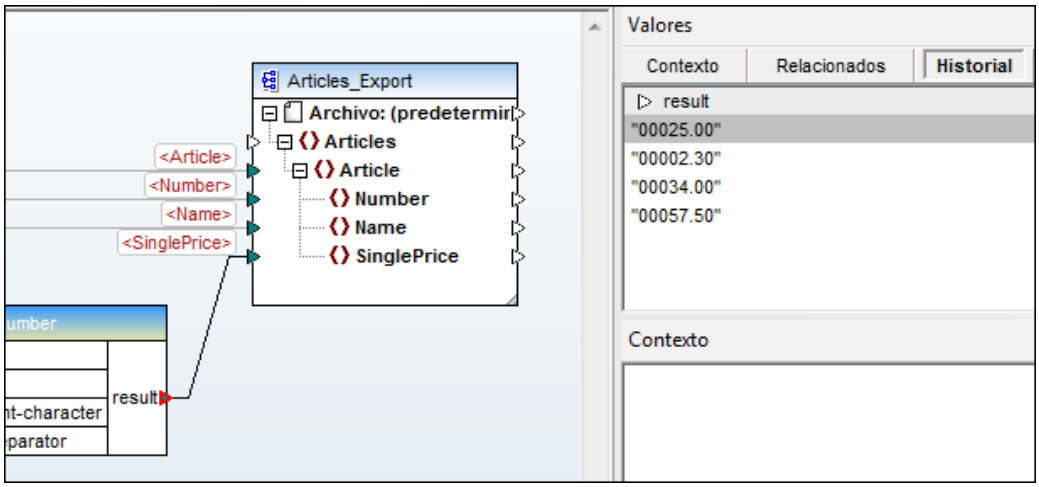
Para comprender mejor este funcionamiento del depurador vamos a depurar de principio a fin y sin puntos de interrupción una asignación y después veremos el historial de valores que fueron procesados por un conector determinado.

Para empezar abra la asignación **PreserveFormatting.mfd** del directorio **<Documentos>\Altova\MapForce2018\MapForceExamples**. Si ya tiene abierta esta asignación, asegúrese de:

- Eliminar todos los puntos de interrupción previamente definidos (*véase* *Agregar y quitar puntos de interrupción*).
- Detener la depuración si está en proceso (pulsando **Mayús + F5**).

Una vez detenida la depuración y eliminados todos los puntos de interrupción previamente definidos:

1. Pulse **F5** para iniciar una nueva sesión de depuración. Cuando pulse **F5**, MapForce ejecutará la asignación en modo depurador y cambiará automáticamente al panel *Resultados*.
2. Ahora haga clic en el panel *Asignación* para volver a la ventana de asignación principal.
3. Haga clic en el nodo `result` de la función `format-number` (marcado en rojo en la imagen siguiente).
4. Para terminar haga clic en la pestaña *Historial* de la ventana Valores y observe qué valores aparecen.

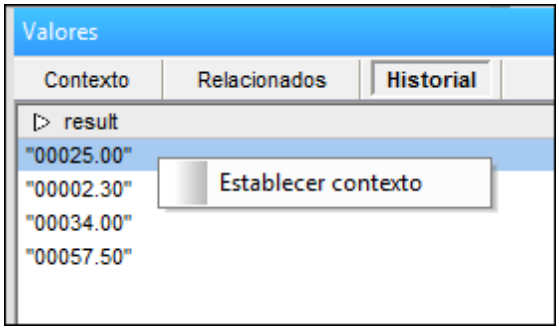


 © 2018 Altova GmbH

Como se aprecia en la imagen anterior, este nodo concreto (`result`) procesó un total de cuatro valores. Si necesita conocer más información sobre alguno de ellos, recuerde que puede recrear el contexto que lo produjo (*véase* *Establecer el contexto en un valor*).

6.12 Establecer el contexto en un valor

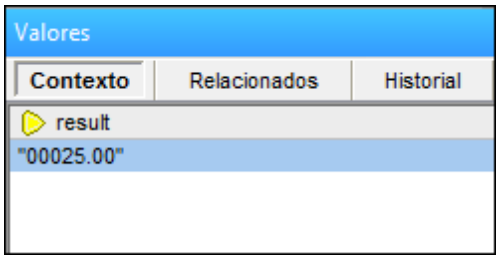
Establecer el contexto en un valor es una acción comparable con retroceder un paso hacia el pasado con el objetivo de obtener más información sobre el contexto de asignación que produjo dicho valor. Puede establecer el contexto en cualquier valor de los que aparecen en la ventana Valores (en cualquiera de sus pestañas). Si habilitó la opción *Guardar historial de seguimiento completo* (*véase* Configurar el depurador), la pestaña *Historial* mostrará todos los valores que fueron procesados por el conector seleccionado. Por tanto, en este caso podrá establecer el contexto en cualquier valor del pasado de dicho conector.



Hay dos maneras de establecer el contexto en un valor desde la ventana Valores:

- Haciendo clic con el botón derecho en un valor y seleccionando **Establecer contexto** en el menú contextual.
- Haciendo doble clic en el valor.

Cuando se establece el contexto en un valor, MapForce resalta el área de asignación para recrear la situación que produjo dicho valor y rellena la ventana Valores y la ventana Contexto en función del contexto seleccionado. En el apartado Información sobre el modo depurador se describen los iconos y señales visuales que se utilizan en el área de asignación. Para más información sobre el término *contexto* propiamente dicho consulte el apartado Ventana Contexto.



El conector de un contexto que se estableció a mano se señala en amarillo ( ). Esto indica que ya no estamos en la posición más reciente de la ejecución. Para volver a la posición más reciente de la ejecución (si procede), haga clic en el botón **Restaurar con actual** que aparece en la pestaña *Contexto* de la ventana Valores.

© 2018 Altova GmbH

6.13 Configurar el depurador

Para acceder a las opciones de configuración del depurador de MapForce seleccione el comando de menú **Herramientas | Opciones** y haga clic en *Depurador*. Estas son las opciones disponibles:

Longitud máxima de almacenamiento de valores

Define la longitud de cadena de los valores que aparecen en la ventana Valores (15 caracteres como mínimo). Recuerde que si define una longitud de almacenamiento alta, puede que se agote la memoria del sistema.

Guardar historial de seguimiento completo

Determina si MapForce debe guardar el historial de todos los valores procesados por todos los conectores de todos los componentes de la asignación mientras dure la depuración. Si activa esta opción, todos los valores procesados por MapForce desde el principio de la ejecución de depuración se almacenarán en memoria y se podrán analizar en la ventana Valores hasta que se detenga la depuración. No se recomienda activar esta opción si tiene pensado depurar asignaciones que hagan un uso intensivo de datos porque podría ralentizar la ejecución de depuración y agotar la memoria del sistema. Si desactiva esta opción, MapForce guardará solamente el historial de seguimiento más reciente de los nodos relacionados con la posición actual de la ejecución.

Altova MapForce 2018 Enterprise Edition

Orígenes y destinos de datos

7 Orígenes y destinos de datos

En esta sección describimos todos los tipos de componente de origen y destino compatibles con MapForce.

- XML y esquemas XML
- Bases de datos
- Archivos CSV y archivos de texto
- MapForce FlexText
- EDI (esto incluye formatos basados en EDI o relacionados, como ASC X12, HIPAA X12, HL7 Versión 2, IATA PADIS, NCPDP SCRIPT, SAP IDoc, TRADACOMS, UN/EDIFACT)
- JSON
- Microsoft OOXML Excel 2007+
- XBRL
- HL7 Versión 3

© 2018 Altova GmbH

7.1 XML y esquemas XML

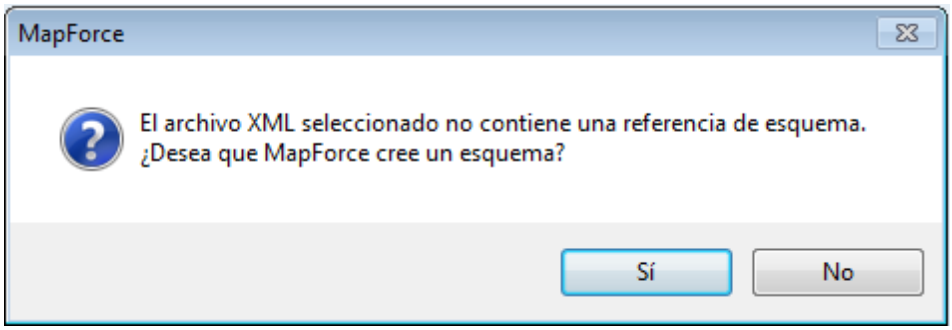
Sitio web de Altova: Asignaciones XML

En los temas de introducción de esta documentación se presentaron ejemplos de asignaciones de datos simples que utilizaban archivos XML y de esquema XML como componentes de origen y destino. En esta sección encontrará más información sobre este tipo de componentes, organizada por temas:

- Configuración de componentes XML
- Usar documentos DTD como componentes de esquema
- Asignación de datos de tipos XML Schema derivados
- Compatibilidad con nombres QName
- Valores Nil / Nillable
- Comentarios e instrucciones de procesamiento
- Secciones CDATA
- Comodines xs:any

7.1.1 Generar un esquema XML

MapForce puede generar automáticamente un esquema XML a partir de un archivo XML. Cada vez que se añade un archivo XML que carece de esquema al área de asignación (con el comando **Insertar | Archivo o esquema XML**), MapForce muestra este mensaje:



Haga clic en **Sí** para generar el esquema. Después deberá seleccionar el directorio donde desea guardar el esquema resultante.

Cuando MapForce genera un esquema a partir de un archivo XML, los tipos de datos para elementos y atributos deben inferirse del documento de instancia XML y puede que el resultado no sea exactamente lo que se esperaba. Por eso, recomendamos que terminada la operación compruebe si el esquema generado representa correctamente los datos de instancia.

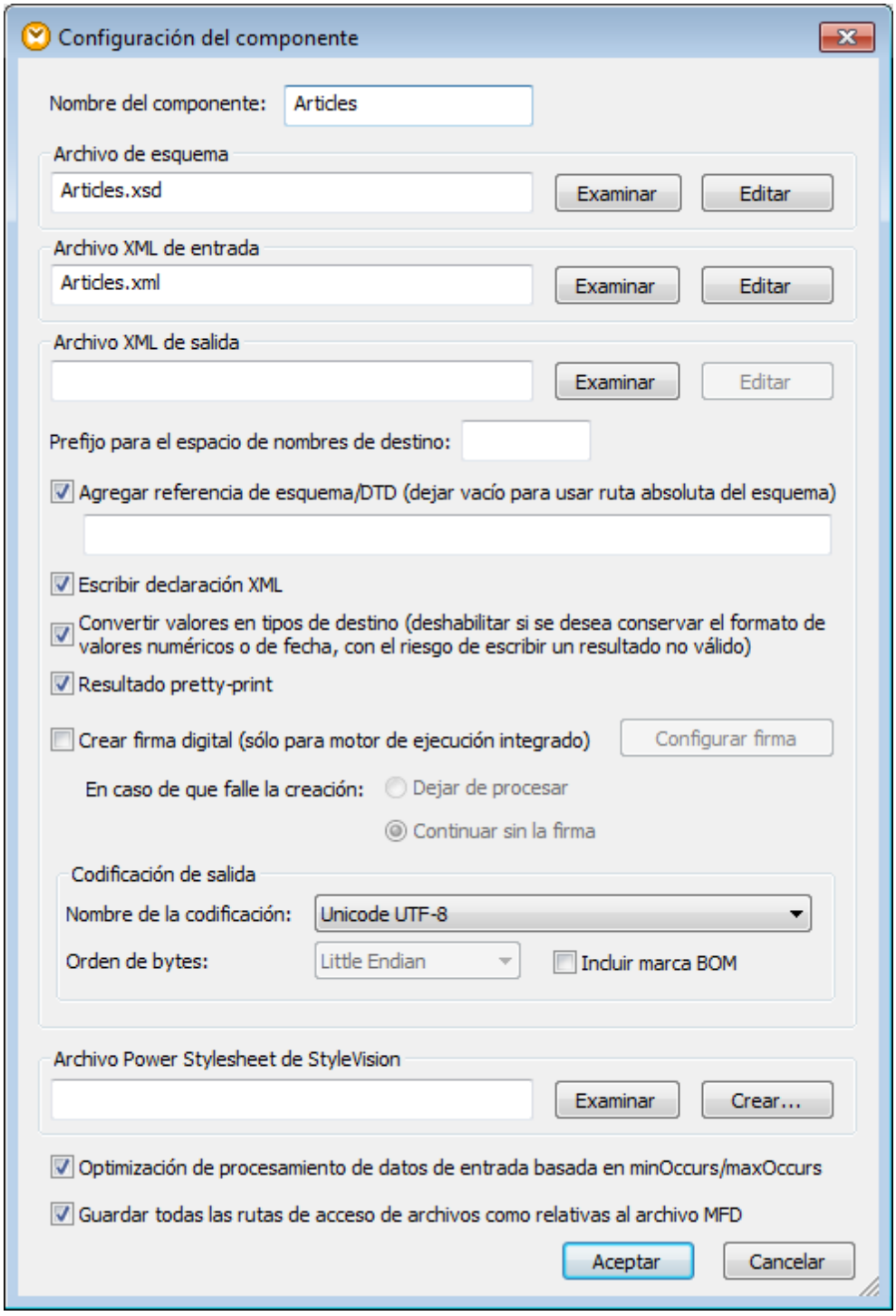
Si existen elementos o atributos en más de un espacio de nombres, MapForce genera un esquema XML distinto por cada espacio de nombres (es decir, puede que se creen varios archivos en el disco).

7.1.2 Configuración de componentes XML

Tras añadir un componente XML al área de asignación, podrá configurarlo desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de tres maneras diferentes:

- Seleccionando el componente en el área de asignación y haciendo clic en el comando de menú **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y eligiendo **Propiedades** en el menú contextual.

© 2018 Altova GmbH



Cuadro de diálogo "Configuración del componente" para un componente XML en MapForce Enterprise y Professional Edition

Esta son las opciones que ofrece el cuadro de diálogo:

Nombre del componente	El nombre del componente se genera automáticamente cuando se crea el componente. Sin embargo, puede cambiar el nombre si lo desea. Si el nombre del componente se generó automáticamente y después selecciona un archivo de instancia, MapForce le preguntará si desea actualizar también el nombre del componente. El nombre del componente puede contener espacios (p. ej. Archivo XML de origen) y puntos (p. ej. Pedidos.EDI). No puede contener barras diagonales, dos puntos, comillas dobles ni espacios iniciales ni finales. Además, cuando cambie el nombre del componente tenga en cuenta que: • Si tiene pensado implementar la asignación en FlowForce Server, el nombre del componente debe ser único. • Se recomienda usar caracteres que se puedan introducir en la línea de comandos. Los caracteres regionales pueden tener distinta codificación en Windows y en la línea de comandos.
Archivo de esquema	Especifica el nombre o la ruta de acceso del archivo de esquema XML que MapForce usará para validar y asignar datos. Para cambiar de archivo de esquema haga clic en **Examinar** y seleccione el nuevo archivo. Para editarlo en XMLSpy haga clic en **Editar**.
Archivo XML de entrada	Especifica el archivo XML de instancia donde MapForce leerá datos. Este campo es relevante para los componentes de origen y se rellena automáticamente cuando se crea el componente por primera vez y se le asigna un archivo XML de instancia. En un componente de origen el nombre del archivo de instancia también se usa para detectar el elemento XML raíz y el esquema al que hace referencia, además de para validar los datos con el esquema seleccionado. Para cambiar la ubicación del archivo haga clic en **Examinar** y seleccione el nuevo archivo. Para editarlo en XMLSpy haga clic en **Editar**.
Archivo XML de salida	Especifica el archivo XML de instancia donde MapForce escribirá los datos. Este campo es relevante para los componentes de destino. Para cambiar la ubicación del archivo haga clic en **Examinar**

© 2018 Altova GmbH

<table>
<tr><td></td><td>y seleccione el nuevo archivo. Para editarlo en XMLSpy haga clic en Editar.</td></tr>
<tr><td>Prefijo para el espacio de nombres de destino</td><td>Permite introducir un prefijo para el espacio de nombres de destino. Antes de asignar el prefijo, asegúrese de que en el esquema de destino está definido el espacio de nombres de destino.</td></tr>
<tr><td>Agregar referencia de esquema/DTD</td><td>Agrega la ruta de acceso del esquema XML referenciado al elemento raíz del archivo XML de salida. La ruta de acceso del esquema introducido en este campo se escribe en los archivos de instancia de destino que se generan (en el atributo <code>xsi:schemaLocation</code> o en la declaración <code>DOCTYPE</code> si se usa una DTD.

Recuerde que, si genera código en XQuery o C++, la opción de agregar una referencia de DTD no estará disponible.

Al introducir una ruta de acceso en este campo podrá definir dónde está ubicado el archivo de esquema al que hace referencia el archivo XML de instancia. Esto garantiza que la instancia de salida se pueda validar en el destino de la asignación cuando se ejecute la asignación. En este campo puede introducir tanto una dirección http:// como una ruta de acceso absoluta o relativa.

Si desactiva esta opción, podrá desvincular la instancia XML del esquema XML o DTD de referencia (por ejemplo, si desea enviar el documento XML de salida a alguien que no tiene acceso al esquema XML subyacente).</td></tr>
<tr><td>Escribir declaración XML</td><td>Esta opción permite suprimir la declaración XML del documento de salida que se genera. Esta opción está habilitada por defecto, es decir, la declaración XML se describe por defecto en los resultados.

Esta característica no es compatible con todos los lenguajes de destino y motores de ejecución de MapForce, como puede verse en la tabla siguiente.

<table>
<tr><th>Lenguaje de destino / motor de ejecución</th><th>Si el resultado es un archivo</th><th>Si el resultado es una cadena de texto</th></tr>
<tr><td>Integrado</td><td>Sí</td><td>Sí</td></tr>
<tr><td>MapForce Server</td><td>Sí</td><td>Sí</td></tr>
<tr><td>XSLT, XQuery</td><td>Sí</td><td>No</td></tr>
<tr><td>Generador de código (C++, C#, Java)</td><td>Sí</td><td>Sí</td></tr>
</table>
</td></tr>
</table>

Convertir valores en tipos de destino	Permite definir si los tipos del esquema XML de destino deben utilizarse durante la asignación o si todos los datos asignados al componente de destino deben considerarse valores de **cadena**. Esta opción está habilitada por defecto. Si desactiva esta opción podrá conservar el formato preciso de los valores. Por ejemplo, si se desea satisfacer una faceta de patrón de un esquema que requiere un número concreto de dígitos decimales en un valor numérico. Puede usar funciones de asignación de datos para dar el formato de cadena necesario al número y después asignar esta cadena al destino. Recuerde que si deshabilita esta opción, también se deshabilita la detección de valores no válidos (p. ej. si se escriben letras en campos numéricos).
Resultado pretty-print	Ajusta el formato del documento XML de salida para que tenga un aspecto estructurado. Cada nodo secundario se desplaza una tabulación con respecto a su primario.
Crear firma digital	Permite agregar una firma digital al archivo XML de instancia de salida. Esto solo es compatible con el lenguaje de transformación integrado (*véase* *Firmas digitales*).
Codificación de salida	Permite especificar estas opciones de configuración para el archivo de instancia de salida: • Nombre de la codificación • Orden de bytes • Si se incluye o no la marca BOM Por defecto, todos los componentes nuevos tienen la codificación definida en la opción *Codificación predeterminada para componentes nuevos*. Esta opción se puede modificar en la pestaña *Generales* de **Herramientas | Opciones**. Si la asignación de datos genera código XSLT 1.0/2.0, el estado de la casilla *Incluir marca BOM* no tiene ningún efecto porque estos lenguajes no son compatibles con marcas BOM.
Archivo Power Stylesheet de StyleVision	Esta opción permite seleccionar o crear un archivo de hoja de estilos de Altova StyleVision. Estos archivos permiten presentar los datos de salida de un archivo XML de instancia en varios formatos, como HTML, RTF, etc. Consulte también el apartado Usar rutas de acceso relativas en un componente.

© 2018 Altova GmbH

Optimización de procesamiento de datos de entrada basada en minOccurs/maxOccurs	Esta opción permite tratar de forma especial las secuencias que contienen un elemento exactamente, como por ejemplo los atributos obligatorios o los elementos secundarios con atributos `minOccurs` y `maxOccurs="1"`. En este caso, MapForce extrae el primer elemento de la secuencia y después procesa el elemento directamente como valor atómico (y no como secuencia). Si los datos de entrada **no son válidos** según el esquema, puede que la asignación incluya una secuencia vacía que detenga la ejecución con un mensaje de error. Para permitir el procesamiento de dicha **entrada no válida** basta con desactivar esta casilla.
Guardar todas las rutas de acceso de archivos como relativas al archivo MFD	Cuando se habilita esta opción, MapForce guarda las rutas de acceso de archivos que aparecen en el cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd). Consulte también el apartado Usar rutas de acceso relativas en un componente.

7.1.3 Usar documentos DTD como componentes de esquema

A partir de la versión 2006 SP2, MapForce es compatible con el uso de documentos DTD preparados para espacios de nombres como componentes de origen y destino. Para ello, los URI de espacio de nombres se extraen de las declaraciones de atributo "xmlns" de la DTD.

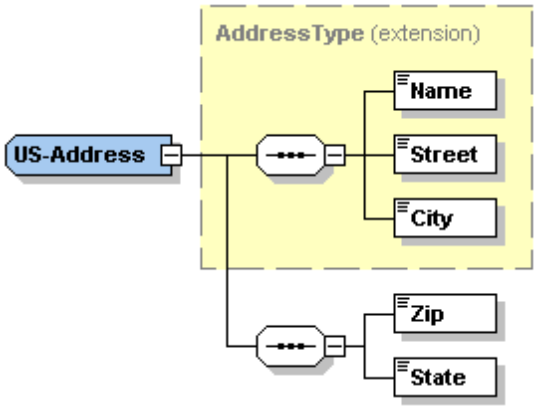
Sin embargo, algunas DTD contienen declaraciones de atributo "xmlns" sin URI de espacio de nombres (p. ej. las DTD que utiliza StyleVision). Estas DTD se deben ampliar para poder utilizarse en MapForce. Concretamente los atributos "xmlns" deben definirse con el URI de espacio de nombres tal y como aparece en este ejemplo:

```
<!ATTLIST fo:root
   xmlns:fo CDATA #FIXED 'http://www.w3.org/1999/XSL/Format'
   ...
>
```

7.1.4 Tipos XML Schema derivados

MapForce permite realizar asignaciones de datos entre tipos derivados de un tipo complejo. Los tipos derivados son tipos complejos de un esquema XML que usan el atributo **xsi:type** para identificar los tipos derivados especificados.

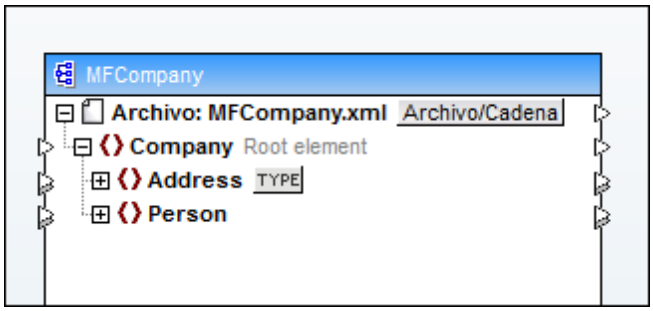
En la imagen siguiente puede ver la definición de un tipo derivado llamado `US-Address` (en XMLSpy). El tipo base (o tipo complejo originario) es `AddressType`. Se añadieron también dos elementos más para crear los tipos derivados `US-Address`: `Zip` y `State`.



Tipo derivado de muestra (en la vista Esquema de XMLSpy)

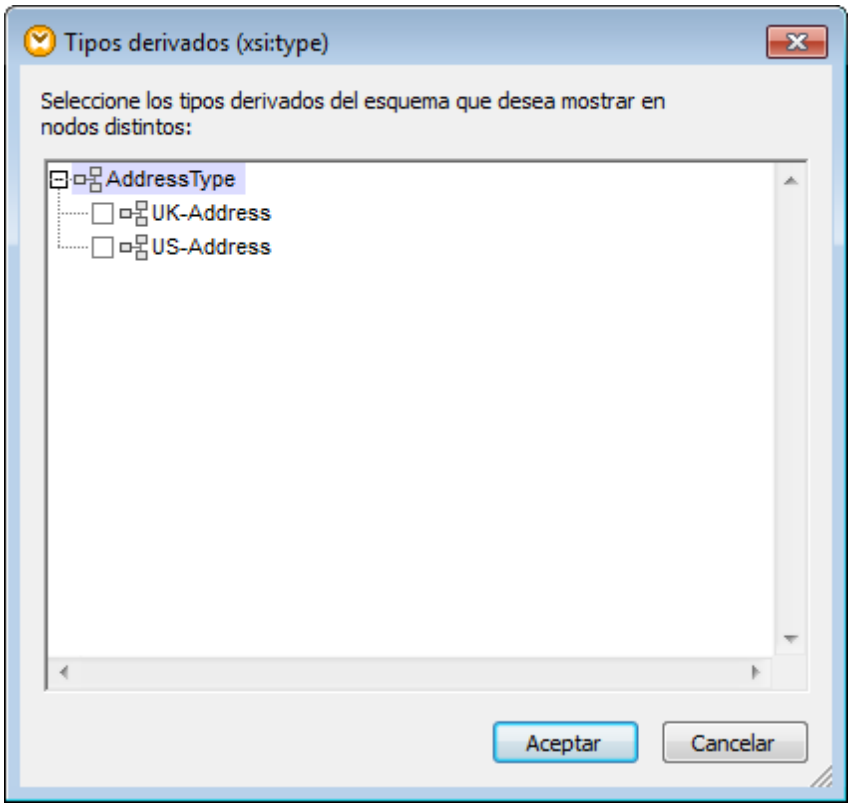
A continuación explicamos cómo realizar asignaciones de datos entre tipos derivados de esquema XML:

1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta este esquema XML: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\MFCompany.xsd**.
2. Cuando MapForce solicite un archivo de instancia haga clic en **Omitir** y seleccione el elemento raíz `Company`.

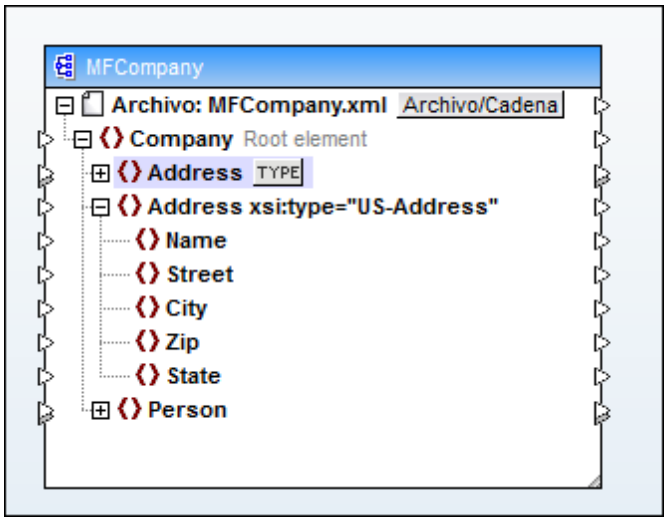


3. Ahora haga clic en el botón TYPE situado junto al elemento `Address`. Este botón indica que para este elemento existen tipos derivados en el esquema.

© 2018 Altova GmbH



4. Marque la casilla del tipo derivado que desea utilizar (en este caso `US-Address`) y haga clic en **Aceptar** para confirmar. Al componente se añade un elemento nuevo llamado `Address xsi:type="US-Address"`.



Ahora puede crear asignaciones de datos entre el tipo derivado `US-Address` y otros elementos.

Recuerde que en el cuadro de diálogo "Tipos derivados" puede seleccionar varios tipos derivados para incluirlos en el componente. En este caso, cada uno de los tipos derivados tendrían su propio elemento `xsi:type` en el componente.

7.1.5 Nombres QName

MapForce resuelve los prefijos de nombres QName (nombre completo) (http://www.w3.org/TR/xml-names/#ns-qualnames) cuando lee datos de archivos XML en tiempo de ejecución de la asignación.

Los nombres QName se usan para hacer referencia a identificadores URI de espacio de nombres de documentos de instancia XML y XBRL y para abreviarlos. Hay dos tipos de QName: con prefijo y sin prefijo:

NombreConPrefijo	Prefijo ':'	ParteLocal
NombreSinPrefijo		ParteLocal

siendo ParteLocal un nombre de elemento o atributo.

Por ejemplo, en el fragmento de código que aparece a continuación `<x:p/>` es un QName:

- el prefijo "x" es una abreviatura del espacio de nombres "**http://miCompañía.com**".
- `p` es la parte local.

```
<?xml version='1.0'?>
<doc xmlns:x="http://miCompañía.com">
   <x:p/>
</doc>
```

MapForce también incluye varias funciones relacionadas con nombres QName en la biblioteca **core | QName functions**.

7.1.6 Valores Nil y Nillable

La especificación XML Schema permite que un elemento sea válido sin tener contenido si el atributo `nillable="true"` se definió para dicho elemento en el esquema. En el documento de instancia XML se puede indicar después que el valor de un elemento es nulo añadiéndole el atributo `xsi:nil="true"`. En este apartado describimos cómo se ocupa MapForce de estos elementos en los componentes de origen y destino.

Diferencias entre 'xsi:nil' y 'nillable'

El atributo `xsi:nil="true"` se define en el documento XML de **instancia**.

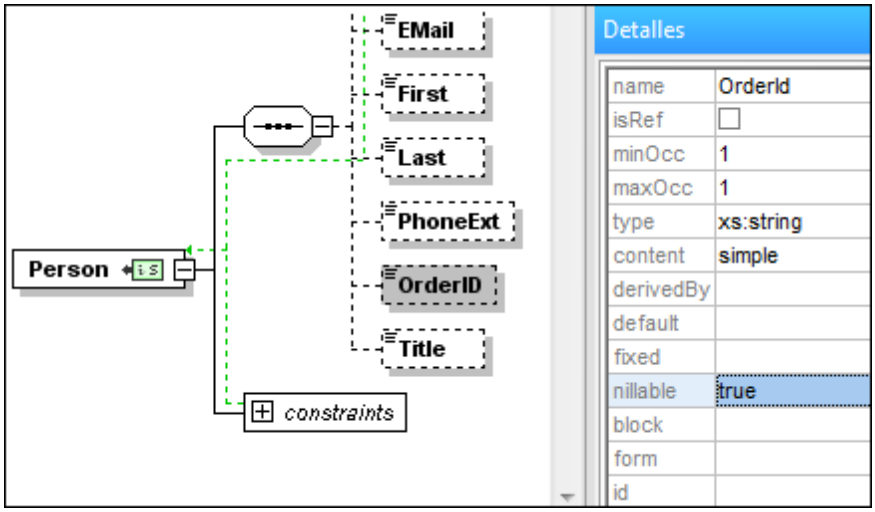
© 2018 Altova GmbH

```
14   <Person>
15     <PrimaryKey>2</PrimaryKey>
16     <ForeignKey>1</ForeignKey>
17     <EMail>biff@amail.com</EMail>
18     <First>biff</First>
19     <Last>bander</Last>
20     <PhoneExt>22</PhoneExt>
21     <OrderID xsi:nil="true"/>
22     <Title>IT services</Title>
23   </Person>
```

El atributo `xsi:nil="true"` indica que, aunque exista, el elemento no tiene contenido. Recuerde que el atributo `xsi:nil="true"` afecta al valor de los elementos y no de los atributos. Un elemento con `xsi:nil="true"` puede tener otros atributos aunque no tenga contenido.

El atributo `xsi:nil` no se representa de forma explícita en la asignación gráfica de MapForce porque se procesa automáticamente en la mayoría de los casos. En concreto, existe un nodo con valor nulo (es decir, un nodo que tiene el atributo `xsi:nil="true"`) pero su contenido no existe.

El atributo `nillable="true"` se define en el **esquema** XML. En MapForce puede estar presente tanto en el componente de origen como en el de destino.



Elementos que pueden ser nulos como origen de la asignación

MapForce revisa automáticamente el atributo `xsi:nil` si la asignación lee datos de elementos XML o XBRL con valor nulo. Si el valor de `xsi:nil` es `true`, el contenido se tratará como inexistente.

Cuando se crea una asignación **basada en el destino** entre un elemento de origen que puede ser nulo y un elemento de destino que puede ser nulo con **contenido simple** (un solo valor con atributos opcionales, pero sin elementos secundarios), donde se definió `xsi:nil` en un elemento de origen, MapForce añade el atributo `xsi:nil` al elemento de destino (p. ej. `<OrderID xsi:nil="true"/>`).

Cuando se crea una asignación de **copia total** entre un elemento de origen que puede ser nulo y un elemento de destino que puede ser nulo, donde se definió `xsi:nil` en un elemento de origen, MapForce añade el atributo `xsi:nil` al elemento de destino (por ejemplo, `<OrderID xsi:nil="true"/>`).

Para comprobar de forma explícita si un elemento de origen tiene un atributo `xsi:nil` con valor `true`, use la función **`is-xsi-nil`**. La función devuelve TRUE para los elementos con valor nulo y FALSE para los demás nodos.

Para sustituir el valor de un elemento de origen nulo (inexistente) con un valor concreto use la función **`substitute-missing`**.

Notas:

- Si se conecta a un elemento de origen con valor nulo, la función **`exists`** devuelve TRUE porque el nodo de elemento existe aunque no tenga contenido.
- Si se usan en elementos donde se definió `xsi:nil`, las funciones que esperan valores simples (como **`multiply`** y **`concat`**) no producen resultados porque no hay ningún contenido de elemento y no se puede extraer ningún valor. Estas funciones se comportan como si el nodo de origen no existiera.

Elementos que pueden ser nulos como destino de la asignación

Cuando se crea una asignación basada en el destino entre un elemento de origen que puede ser nulo y un elemento de destino que puede ser nulo con contenido simple (un solo valor con atributos opcionales pero sin elementos secundarios), donde se definió `xsi:nil` en un elemento de origen, MapForce inserta el atributo `xsi:nil` en el elemento de destino (por ejemplo: `<OrderID xsi:nil="true"/>`). Si el atributo `xsi:nil="true"` no se definió en el elemento XML de origen, entonces el contenido del elemento se asigna al elemento de destino.

Cuando se crea una asignación con un elemento de destino que puede ser nulo con **tipo complejo** (con elementos secundarios), el atributo `xsi:nil` **no** se escribirá automáticamente porque, a la hora de escribir los atributos del elemento, MapForce no puede saber si después vendrán elementos secundarios. En estos casos se recomienda definir una conexión de **copia total** para copiar el atributo `xsi:nil` del elemento de origen.

Cuando se crea una asignación entre una **secuencia vacía** y un elemento de destino, el elemento no se creará, independientemente de si puede ser nulo o no.

Para forzar la creación de un elemento de destino vacío con el atributo `xsi:nil="true"` conecte la función **`set-xsi-nil`** con el elemento de destino directamente. Esto sirve para elementos de destino con tipo complejo y con tipo simple.

Si el nodo tiene tipo simple, use la función **`substitute-missing-with-xsi-nil`** para insertar `xsi:nil` en el destino si no hay ningún valor disponible para la asignación. Esto puede ocurrir si el nodo de origen no existe en absoluto o si en un cálculo (en una multiplicación, por ejemplo) interviene un nodo de origen nulo y, por tanto, no se obtuvo ningún resultado.

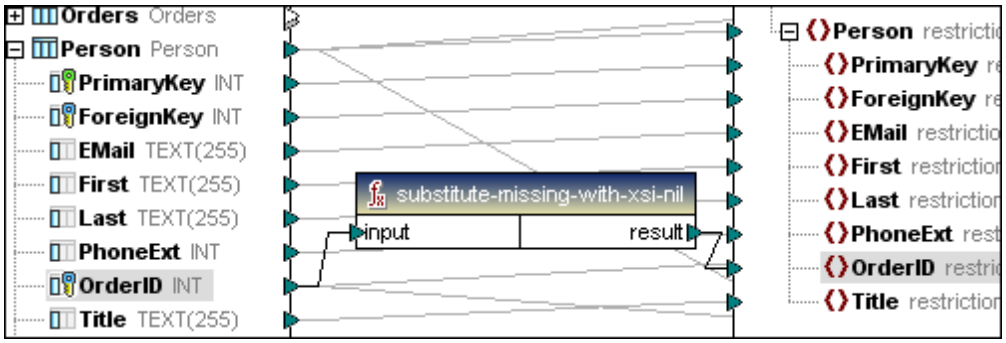
Nota: las funciones que generan `xsi:nil` no pueden pasar por funciones ni componentes que solamente operen en valores (como la función **`if-else`**, por ejemplo).

© 2018 Altova GmbH

Asignar campos de base de datos NULL a xsi:nil

Si crea una asignación entre un campo de base de datos NULL y un elemento que puede ser nulo de un esquema XML, MapForce genera solamente los elementos de destino que contengan datos de BD. Los elementos de campos de base de datos NULL no se crean en el componente de destino. Si se conecta la función de nodo **exists** con un elemento de origen de este tipo, se obtiene el resultado `false` para los campos NULL.

Para forzar la creación de todos los elementos en el componente de destino basta con usar la función **substitute-missing-with-xsi-nil** de biblioteca.



En la imagen anterior puede ver cómo puede usar la función **substitute-missing-with-xsi-nil** para crear elementos de destino para todos los campos de base de datos:

- Todos los campos de base de datos NULL o que faltan contienen <OrderID xsi:nil="true"/> en el elemento de destino.
- Los datos actuales de los campos de BD se asignan directamente al elemento de destino (p. ej. <OrderID>1</OrderID>).

Para ver los campos NULL de un componente de base de datos abra el panel *Consulta de BD* y ejecute una consulta en las tablas de la base de datos. En la ventana de resultados los campos nulos se designan con *[NULL]*.

	EMail	First	Last	PhoneExt	OrderID	Title
1	v.callaby@nanonull.com	Vernon	Callaby	582	*[NULL]*	Office Manager
2	f.further@nanonull.com	Frank	Further	471	*[NULL]*	Accounts Recei
3	l.matise@nanonull.com	Loby	Matise	963	1	Accounting Man
4	j.firstbread@nanonull.com	Joe	Firstbread	621	*[NULL]*	Marketing Manag

Asignar xsi:nil a campos de base de datos NULL

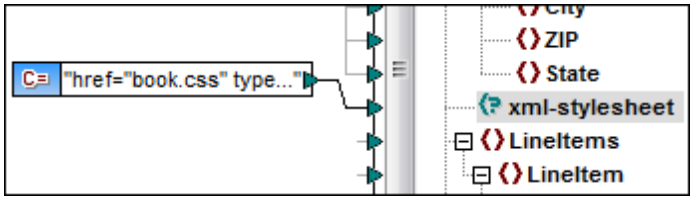
Si crea una asignación entre un elemento XML nulo y una columna de una base de datos, MapForce escribe un valor NULL en la base de datos. También puede usar la función **set-null** si desea que el campo de la base de datos sea nulo de forma incondicional.

7.1.7 Comentarios e instrucciones de procesamiento

En los componentes XML de destino se pueden insertar comentarios e instrucciones de procesamiento. Estas últimas sirven para pasar información a las aplicaciones que continúan con el procesamiento de los documentos XML. Recuerde que no se pueden definir comentarios ni instrucciones de procesamiento en nodos que formen parte de un grupo de asignación de copia total.

Para insertar una instrucción de procesamiento:

1. Haga clic con el botón derecho en un elemento del componente de destino y elija **Comentario/instrucción de procesamiento** en el menú contextual y después elija **Agregar delante una instrucción de procesamiento** o bien **Agregar detrás una instrucción de procesamiento**.
2. Introduzca el nombre (de destino) de la instrucción de procesamiento (p. ej. `xml-stylesheet`) y haga clic en **Aceptar** para confirmar.
   Esto añade un nodo de este nombre a la estructura del componente de destino.

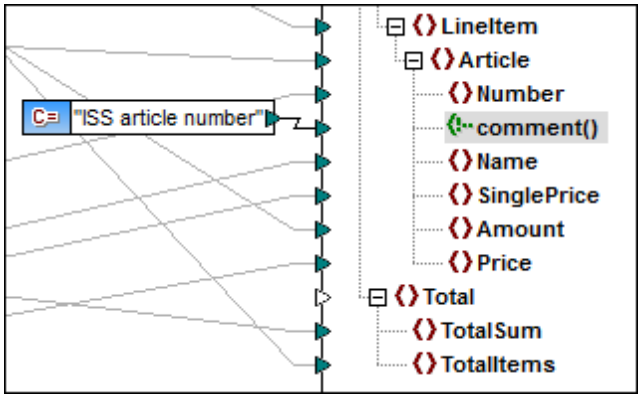


3. Ahora puede usar un componente de constante para aportar el valor del atributo de la instrucción de procesamiento (p. ej. `href="book.css" type="text/css"`).

Nota: puede añadir varias instrucciones de procesamiento delante o detrás de cualquier elemento del componente de destino.

Para insertar un comentario:

1. Haga clic con el botón derecho en un elemento del componente de destino y elija **Comentario/instrucción de procesamiento** en el menú contextual y después elija **Agregar delante un comentario** o bien **Agregar detrás un comentario**.



Esto añade el nodo de comentario (`<!--comment()`) a la estructura del componente de

© 2018 Altova GmbH

destino.
2. Ahora puede usar un componente de constante para aportar el texto del comentario o conectar un nodo de origen al nodo de comentario.

Nota: solamente se puede añadir un comentario antes o detrás de cada nodo de destino. Para crear varios comentarios deberá crear duplicados de entrada.

Para eliminar un comentario o una instrucción de procesamiento:

- Haga clic con el botón derecho en el correspondiente nodo, elija **Comentario/ instrucción de procesamiento** en el menú contextual y después elija **Eliminar**.

7.1.8 Secciones CDATA

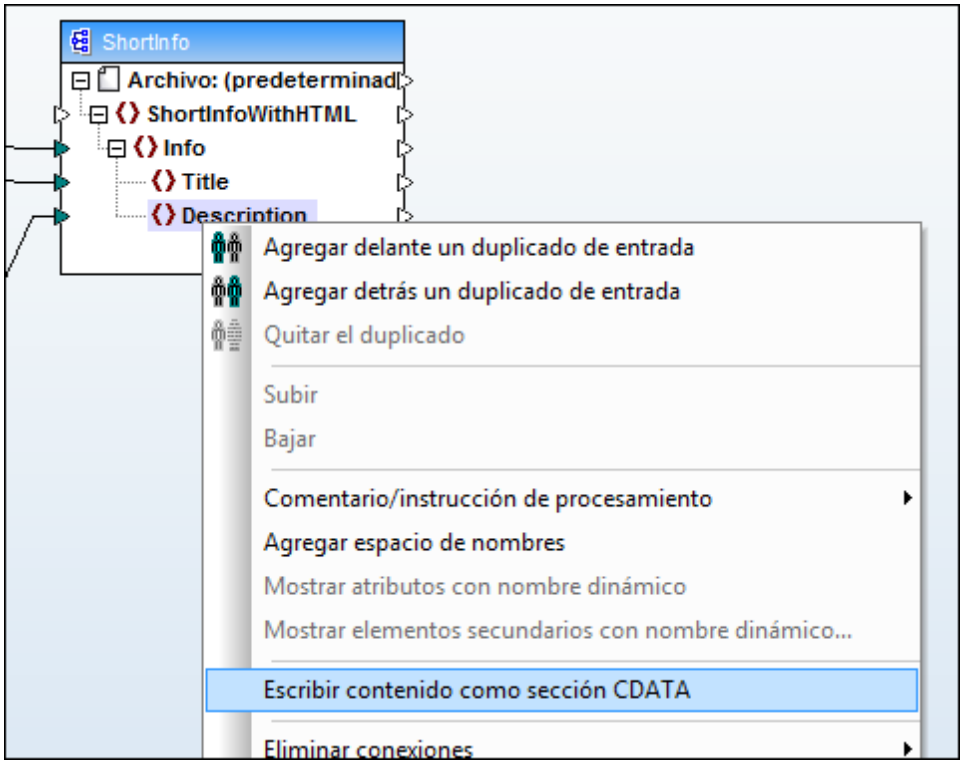
Las secciones CDATA sirven para añadir caracteres de escape a bloques de texto que contienen caracteres que se podrían interpretar como marcado. Las secciones CDATA empiezan con "<![CDATA[" y terminan con "]]>".

Los nodos de destino pueden escribir los datos de entrada que reciben como secciones CDATA. Los componentes de destino pueden ser:

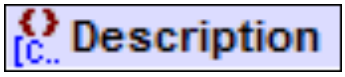
- datos XML
- datos XML incrustados en campos de BD
- elementos secundarios XML de dimensiones con tipo de un componente XBRL de destino

Para crear una sección CDATA:

1. Haga clic con el botón derecho en el nodo de destino que desea definir como sección CDATA y seleccione **Escribir contenido como sección CDATA**.



Aparece entonces una advertencia señalando que los datos de entrada no deben contener el delimitador de cierre de sección CDATA ']]>'. Haga clic en **Aceptar** para cerrar el aviso.
El icono `[C..` que aparece debajo de la etiqueta del elemento indica que este nodo está definido como sección CDATA.



Nota: también se pueden definir secciones CDATA en nodos duplicados y en nodos xsi:type.

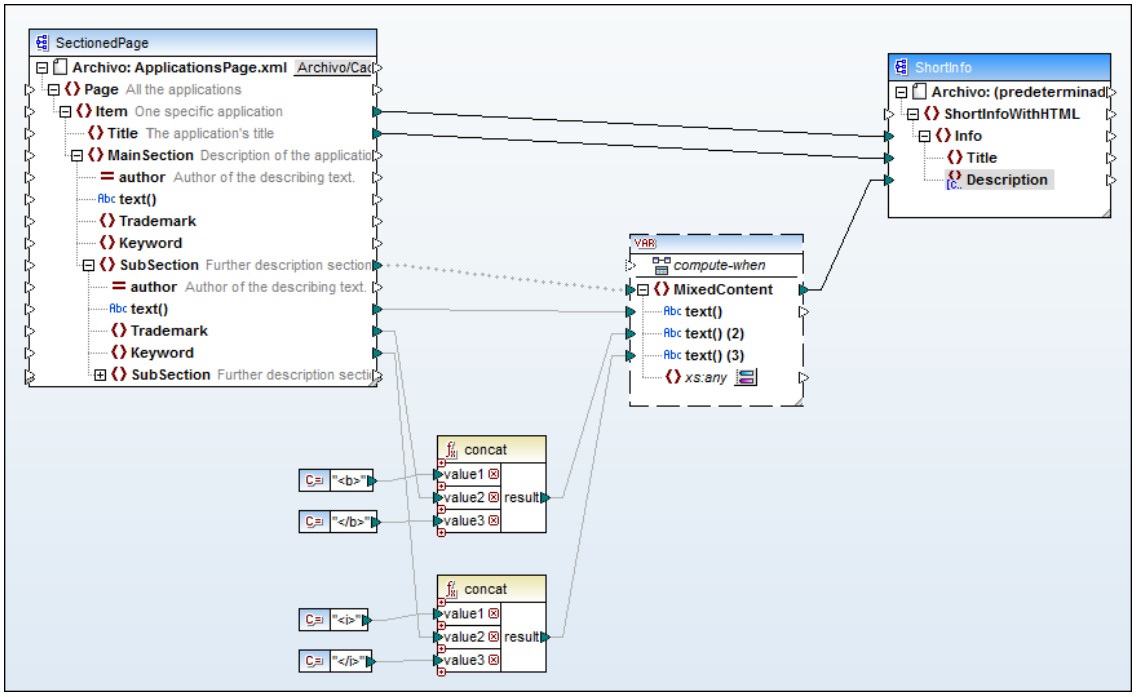
Asignación de ejemplo

El diseño de asignación **HTMLinCDATA.mfd** de la carpeta `...\MapForceExamples` muestra lo útil que pueden ser las secciones CDATA.

En este ejemplo:

- Al contenido del elemento de origen **`Trademark`** se añaden las etiquetas de apertura y cierre y .
- Al contenido del elemento de origen **`Keyword`** se añaden las etiquetas de apertura y cierre y
- Los datos resultantes se pasan a los nodos **`text()`** duplicados en el orden en que aparecen en el documento de origen porque el conector del elemento **`Subsection`** se definió como nodo basado en el origen (de contenido mixto).
- El resultado del nodo **`MixedContent`** se pasa después al nodo **`Description`** del componente de destino **ShortInfo**, que se definió como sección CDATA.

© 2018 Altova GmbH

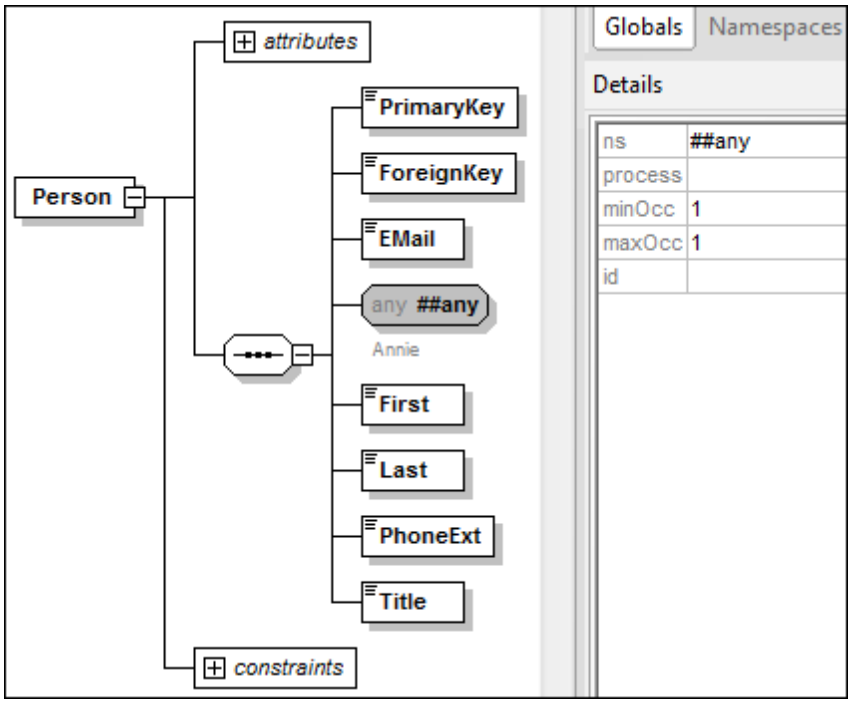


Si abrimos el panel *Resultados* podremos ver la sección CDATA que contiene el texto con marcado.

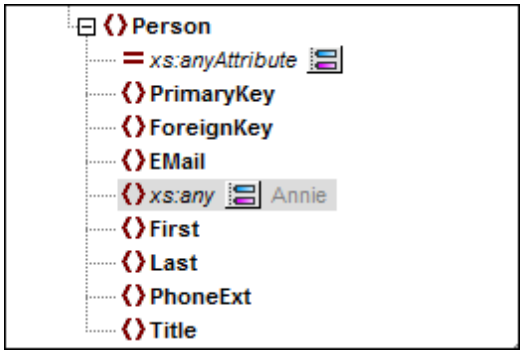
```
7     <Info>
8       <Title>MapForce</Title>
9       <Description><![CDATA[Altova <b>MapForce</b> 2014 Enterprise Edition is the premier <i>XML</i>
/ <i>database</i> / <i>flat file</i> / <i>EDI</i> data mapping tool that auto-generates mapping code in
<i>XSLT</i> 1.0/2.0, <i>XQuery</i>, <i>Java</i>, <i>C++</i> and <i>C#</i>. It is the definitive tool for
data integration and information leverage.]]></Description>
10    </Info>
```

7.1.9 Comodines: xs:any / xs:anyAttribute

Los comodines `xs:any` (y `xs:anyAttribute`) permiten usar cualquier elemento o atributo de los esquemas. En la imagen siguiente puede ver el elemento `any` en la vista Esquema de XMLSpy.

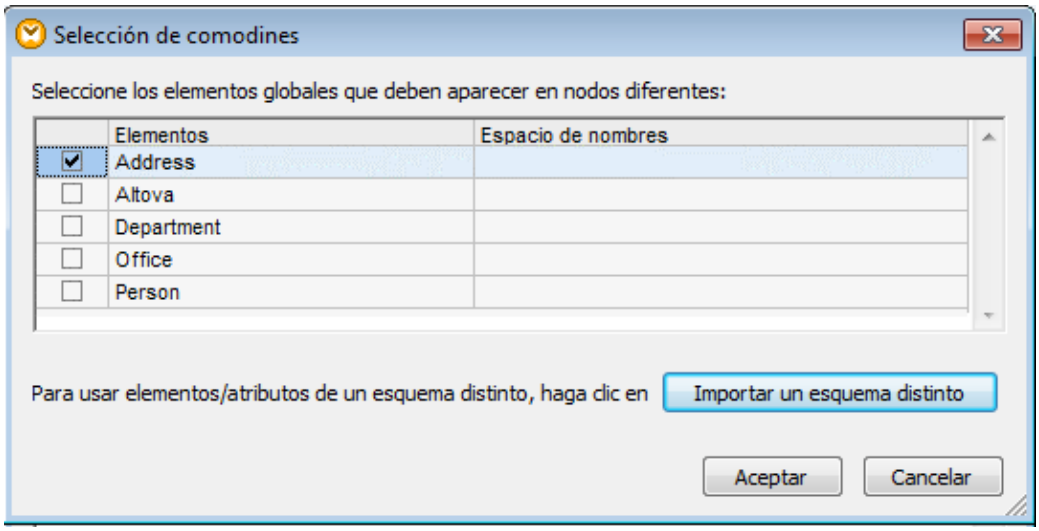


En MapForce los elementos `xs:any` (o `xs:anyAttribute`) aparecen acompañados del botón **Cambiar selección** .

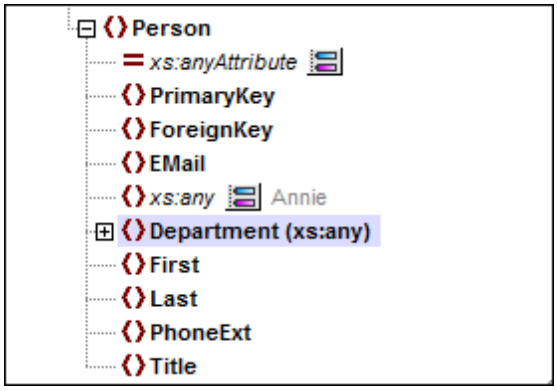


Cuando se hace clic en el botón **Cambiar selección** , aparece el cuadro de diálogo "Selección de comodines". En este cuadro de diálogo se enumeran los elementos y atributos globales declarados en el esquema actual.

 © 2018 Altova GmbH



Si marca una de las casillas y hace clic en **Aceptar** para confirmar, MapForce inserta el elemento/atributo (y sus nodos secundarios) en el componente. Los elementos o atributos comodines se insertan inmediatamente después del nodo cuyo botón **Cambiar selección** accionó el usuario.



Ahora puede crear asignaciones entre estos nodos y cualquier otro elemento.

> En un componente puede reconocer los elementos y atributos comodines por el texto `(xs:any)` que aparece anexado.

Para eliminar un elemento comodín haga clic en el botón **Cambiar selección** y desactive su casilla en el cuadro de diálogo "Selección de comodines".

Comodines y nombres de nodo dinámicos

Por lo general, asignar datos a comodines o viceversa es adecuado si todos los atributos o elementos posibles que aparecen en la instancia XML están declarados por el esquema XML del componente (o si se pueden importar de esquemas externos). Sin embargo, en algunos casos los elementos o atributos que aparecen en la instancia son demasiados. Por ejemplo, en esta

instancia el número de elementos secundarios de <message> es aleatorio:

```
<?xml version="1.0" encoding="UTF-8"?>
<message>
   <line1>1</line1>
   <line2>2</line2>
   <line3>3</line3>
   ...............
   <line999></line999>
</message>
```

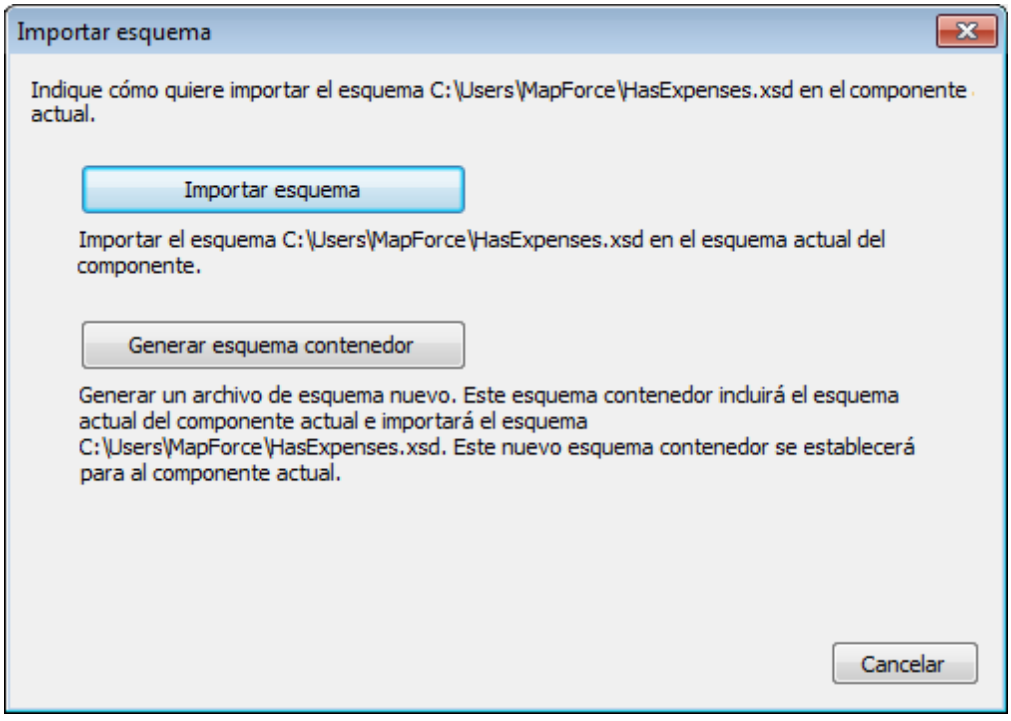
Para casos como este, en lugar de comodines, se recomienda un acceso dinámico a los nombres de nodo (*véase* *Asignar nombres de nodos*).

Agregar elementos de un esquema distinto como comodines

También puede usar como comodines los elementos de un esquema que no sea el asignado. Para poder ver estos elementos en el componente haga clic en el botón **Importar un esquema distinto** del cuadro de diálogo "Selección de comodines". Esto abre un cuadro de diálogo nuevo que ofrece dos opciones:

1. Importar un esquema o
2. generar un esquema contenedor.

Por ejemplo, en la imagen siguiente puede ver lo que ocurre si intenta importar un esquema externo llamado **HasExpenses.xsd** en el esquema actual que está asignado a un componente.

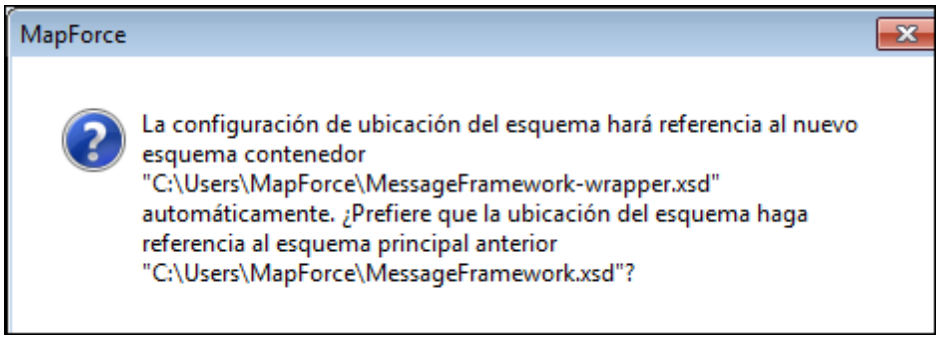


La opción **Importar esquema** importa el esquema externo en el esquema actual que está asignado al componente. Tenga en cuenta que esta opción sobrescribe el esquema actual del

 © 2018 Altova GmbH

componente en el disco. Si el esquema actual es un esquema remoto que se abrió desde una dirección URL (*véase Agregar componentes desde una URL*) y no desde el disco, entonces no se podrá modificar. En este caso debe utilizarse la opción **Generar esquema contenedor**.

La opción **Generar esquema contenedor** crea un archivo de esquema nuevo que se denomina esquema *contenedor*. La ventaja de usar esta opción reside en que el esquema actual del componente no se modifica. Por el contrario, se creará un esquema nuevo (es decir, un esquema contenedor) que incluirá tanto el esquema actual como el esquema que se desea importar. Cuando haga clic en esta opción, MapForce preguntará dónde se debe guardar el esquema contenedor. El nombre predeterminado del esquema contenedor será **archivo-wrapper.xsd**. Una vez guardado, el esquema contenedor se asigna automáticamente y por defecto al componente y aparece este aviso:



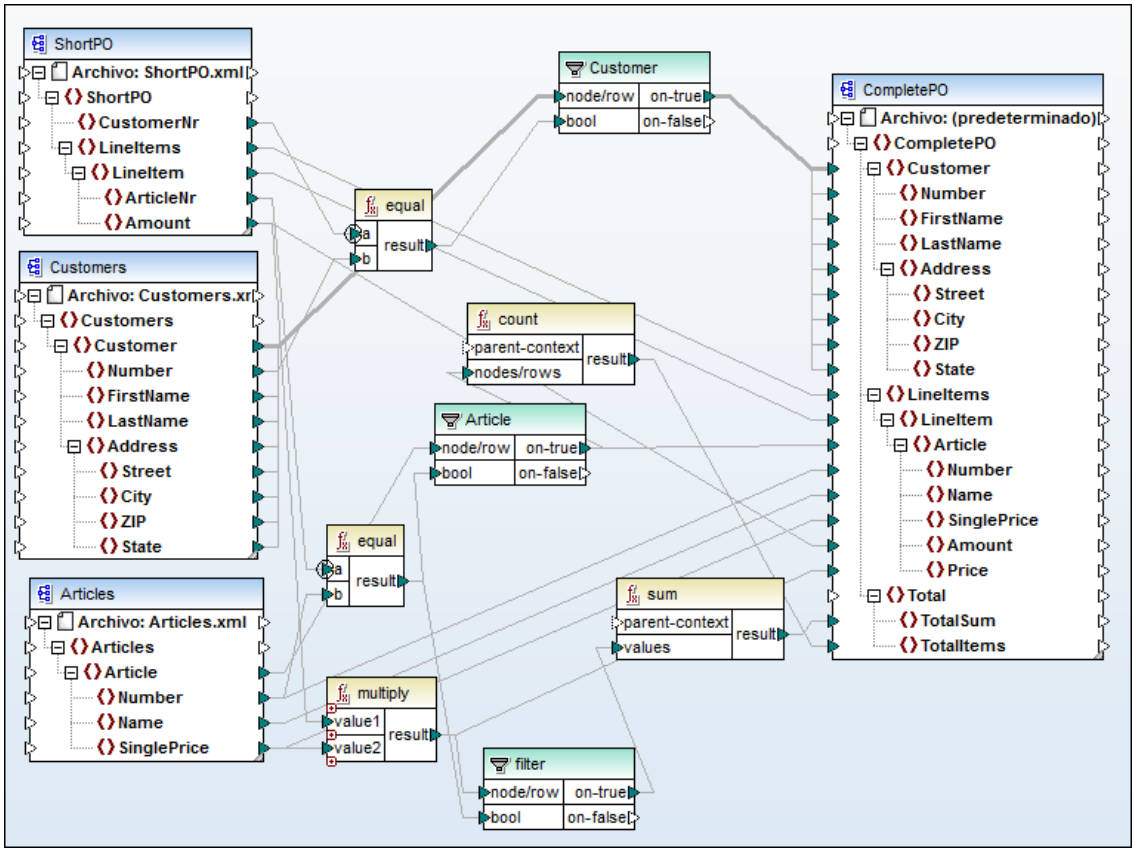
Haga clic en **Sí** para volver al esquema anterior y en **No** para que el esquema contenedor recién creado siga asignado al componente.

7.1.10 Combinar datos de varios esquemas distintos

MapForce permite combinar varios archivos en un solo archivo de destino.

El ejemplo de este apartado combina varios componentes de origen con esquemas distintos en un solo esquema de destino. En la sección Procesar varios archivos de entrada o salida simultáneamente se explica cómo combinar un número aleatorio de archivos usando el mismo esquema.

El archivo **CompletePO.mfd** de la carpeta `...\MapForceExamples` combina tres archivos XML en un archivo XML de orden de compra.



Observe que los datos de varios componentes de origen se combinan en un solo archivo XML de destino (CompletePO).

- **ShortPO** es un esquema que tiene asociado un archivo XML de instancia y contiene los datos sobre los artículos y el número de cliente (en este archivo solo hay un cliente con número de cliente 3).
- **Customers** es un esquema que tiene asociado un archivo XML de instancia y contiene el número de cada cliente e información sobre ellos como el nombre del cliente y su dirección.
- **Articles** es un esquema que tiene asociado un archivo XML de instancia y contiene datos sobre los artículos, como el precio, el nombre y el número de identificación.
- **CompletePO** es un esquema sin archivo de instancia asociado porque todos los datos vienen de los tres archivos XML de instancia. La estructura jerárquica de este archivo permite combinar todos los datos XML y darles salida.

Este archivo de esquema debe crearse en un editor XML como XMLSpy. No se genera con MapForce (aunque se podría crear si tuviera un archivo de instancia CompletePO.xml).

La estructura de CompletePO es una combinación de la estructura de los archivos XML de origen.

El componente de filtrado Customer sirve para buscar/filtrar los datos donde el número de cliente sea idéntico en los archivos XML ShortPO y Customers y después pasar los datos asociados al componente de destino CompletePO.

- El nodo **CustomerNr** de ShortPO se compara con el nodo **Number** de Customers con ayuda de la función "equal".

 © 2018 Altova GmbH

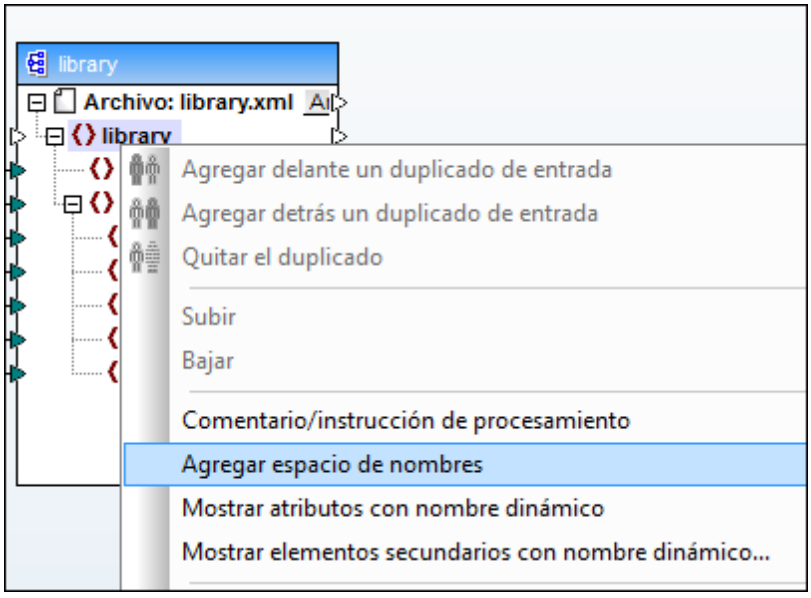
- Como ShortPO solo contiene un cliente (el número 3), solamente se puede pasar al componente de filtrado los datos disponibles para el cliente número 3.
- El parámetro **node/row** del componente de filtrado pasa los datos de **Customer** a "on-true" si el parámetro es `true`, es decir, cuando se encuentra el mismo número (en este caso el número 3).
- El resto de datos sobre el cliente y los artículos se pasan al esquema de destino a través de dos componentes de filtrado más.

7.1.11 Declarar espacios de nombres personalizados

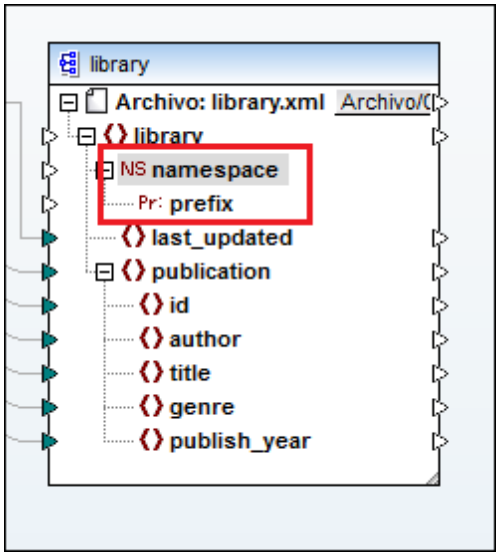
Cuando una asignación produce un resultado XML, MapForce deriva automáticamente el espacio de nombres (o conjunto de espacios de nombres) de cada elemento y atributo a partir del esquema asociado al componente de destino. Este es el comportamiento predeterminado de MapForce y el más apropiado en las asignaciones de datos que implican la generación de resultados XML.

Sin embargo, en otros casos puede ser preferible tener un mayor control sobre el espacio de nombres de los elementos en el código XML resultante. Por ejemplo, en algunos casos puede ser necesario declarar a mano el espacio de nombres de un elemento desde la asignación directamente.

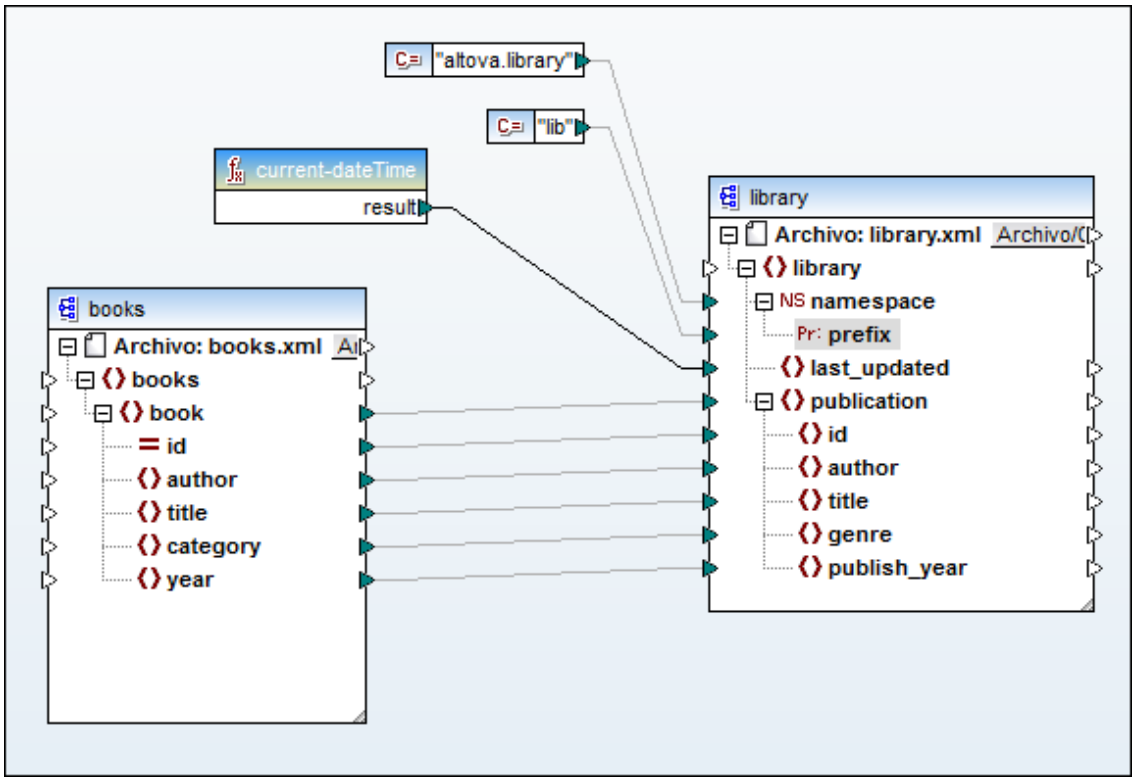
Para comprender mejor este funcionamiento abra la asignación **BooksToLibrary.mfd** de la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`. Haga clic con el botón derecho en el nodo `library` y seleccione Agregar **espacio de nombres** en el menú contextual.



Observe que ahora aparecen dos nodos nuevos bajo el nodo `library`: el nodo `namespace` y el nodo `prefix`.



Ahora puede crear asignaciones entre estos nodos y valores de cadena. Por ejemplo, en la imagen siguiente puede ver que se definieron dos constantes (con el comando de menú **Insertar | Constante**) que aportan el espacio de nombres "altova.library" y el prefijo "lib":



En el resultado podrá ver que el atributo `xmlns:<prefix>="<namespace>"` se añadió al elemento y que `<prefix>` y `<namespace>` son valores procedentes de la asignación (en este caso vienen dados por constantes). El resultado será:

© 2018 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<library xmlns:lib="altova.library" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance" xsi:noNamespaceSchemaLocation="library.xsd">
...
```

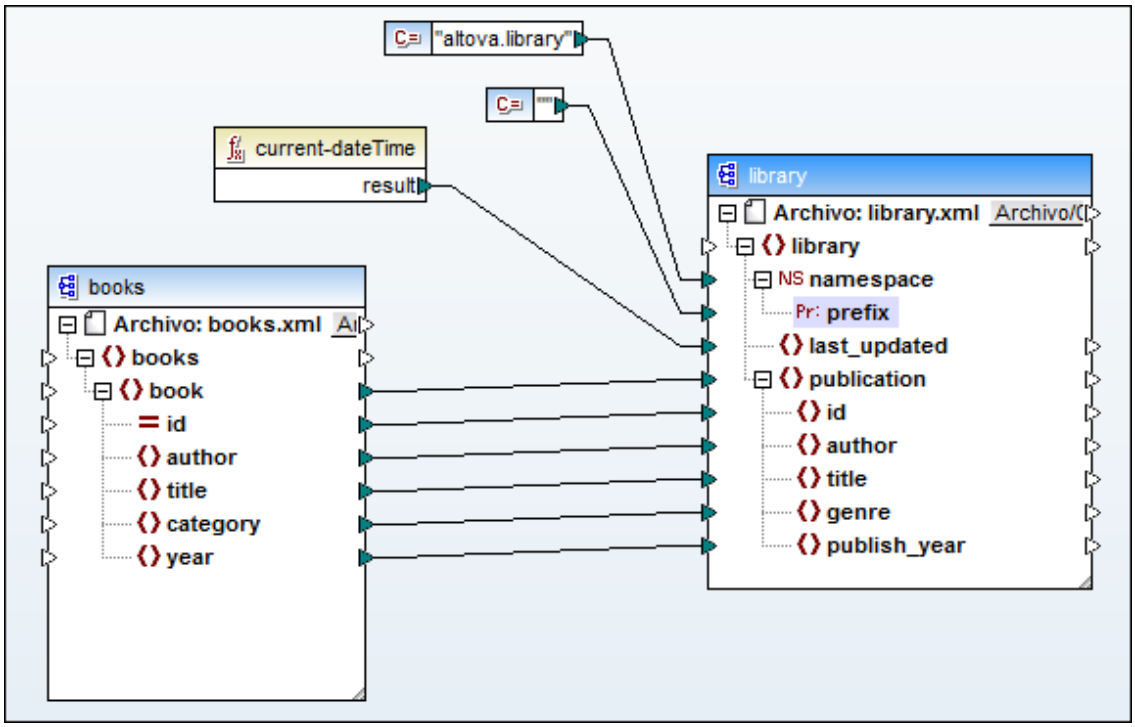
Nota: la declaración de espacios de nombres personalizados (y el comando **Agregar espacio de nombres**) solamente es relevante para componentes de destino XML y solamente para elementos. El comando **Agregar espacio de nombres** no está disponible en el caso de atributos ni nodos comodín. Tampoco está disponible en caso de nodos que reciben datos por medio de conexiones de **copia total**.

También puede declarar varios espacios de nombres para un mismo elemento. Para ello vuelva a hacer clic con el botón derecho en el nodo y seleccione Agregar espacio de nombres en el menú contextual. Bajo el nodo aparecerán dos nodos más para el espacio de nombres y para el prefijo. Ahora podrá conectar dos nuevos valores a estos dos nodos.

Para eliminar una declaración de espacio de nombres haga clic con el botón derecho en el nodo `ns:namespace` y seleccione **Quitar espacio de nombres** en el menú contextual.

Es obligatorio conectar los conectores de entrada de `namespace` y de `prefix` (aunque estén conectados a valores vacíos).

Si desea declarar un espacio de nombres predeterminado (es decir, un espacio de nombres con el formato `xmlns="miEspacioPredeterminado"`), basta con asignar un valor de cadena vacío al nodo `prefix`. Por ejemplo, en la imagen siguiente, el valor de la segunda constante es una cadena vacía.



El resultado sería:

```
<?xml version="1.0" encoding="UTF-8"?>
<library xmlns="altova.library" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="library.xsd">
...
```

Si necesita crear prefijos para nombres de atributo (p. ej. `<number prod:id="prod557">557</number>`) puede habilitar el acceso dinámico a los atributos del nodo (*véase* *Asignar nombres de nodos*) o editar el esquema para que contenga un atributo `prod:id` para `<number>`.

7.1.12 Firmas digitales

Las firmas digitales son una especificación del W3C para la firma digital de documentos XML con código cifrado que puede utilizarse para comprobar si se modificó el documento XML. La característica de firmas digitales de MapForce es compatible con certificados de tipo RSA-SHA1 y DSA-SHA1 solamente.

Para más información sobre firmas digitales consulte la especificación del W3C para firmas XML: http://www.w3.org/TR/xmldsig-core/

MapForce permite crear firmas digitales XML para archivos de salida XML y XBRL. Pero solamente se pueden generar si el lenguaje de transformación es BUILTIN y solamente en la vista previa. MapForce crea la firma para el archivo de resultados que genera cuando se hace clic en el panel *Resultados* y se guarda el archivo de resultados.

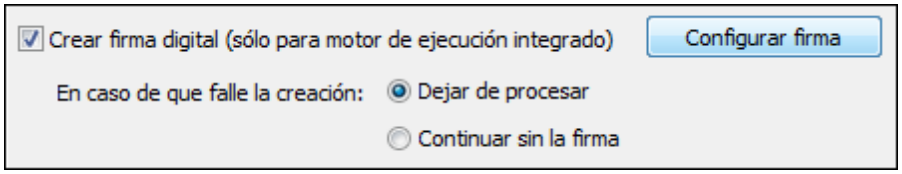
Nota: MapForce Server no es compatible con firmas digitales.

Puede incrustar la firma digital como último elemento del documento de salida o puede almacenarla en un archivo de firma separado:

- Si selecciona la opción *Envuelta*, la firma será el último secundario situado bajo el elemento raíz del archivo XML.
- Si selecciona la opción *Separada*, la firma se genera en un otro documento.

Para activar la generación de firmas digitales:

1. Abra el cuadro de diálogo "Configuración del componente" del componente de destino (haciendo doble clic en el título del componente o seleccionando el comando de menú **Componente | Propiedades**.
2. Marque la casilla *Crear firma digital*.



3. Aparece el cuadro de diálogo "Opciones de la firma XML", donde puede definir las opciones de configuración pertinentes (*véase* *Configurar la firma XML*).

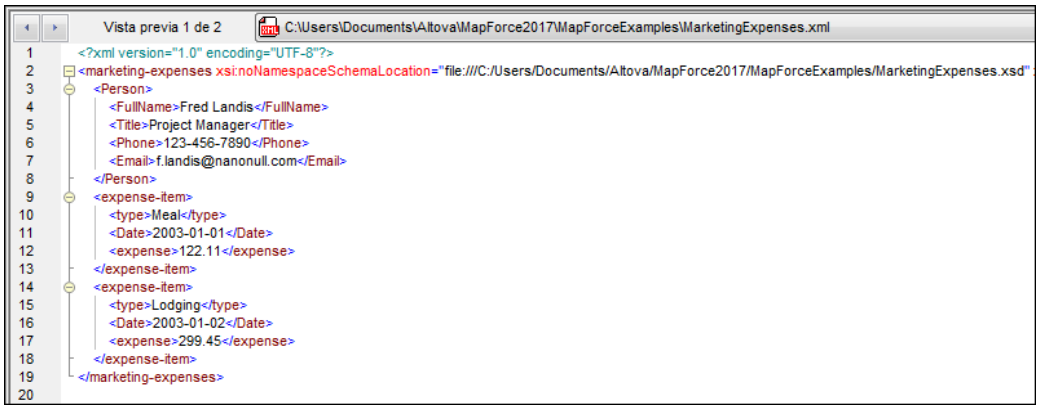
Para cambiar la configuración de una firma digital:

1. Abra el cuadro de diálogo "Configuración del componente" del componente de destino (haciendo doble clic en el título del componente o seleccionando el comando de menú **Componente | Propiedades**.
2. Haga clic en el botón **Configurar firma** para abrir el cuadro de diálogo Opciones de la firma XML.
3. Elija las opciones necesarias y haga clic en **Aceptar**.

Veamos ahora el ejemplo **MarketingExpenses_DetachedSignature.mfd** de la carpeta `...\MapForceExamples`:

1. Haga doble clic en el componente de destino **MarketingExpenses** y después haga clic en el botón **Configurar firma**. En el cuadro de diálogo que aparece puede ver cómo se configuró la firma.
2. Haga clic en **Aceptar** para cerrar el cuadro de diálogo.
3. Abra el panel *Resultados* para ver el resultado de la asignación.

   En la ventana de vista previa se generaron dos archivos. El primer archivo **MarketingExpenses.xml** es el resultado de dicho componente de destino.

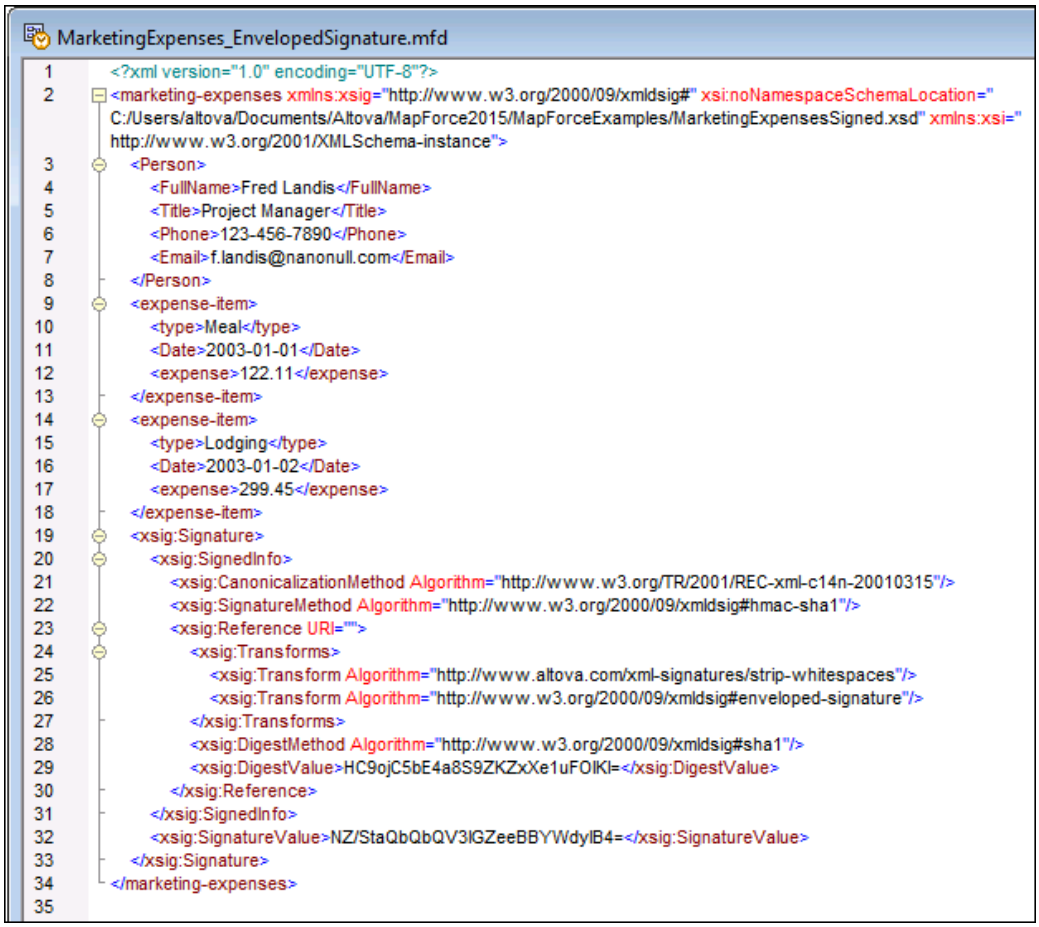


   El segundo archivo **MarketingExpenses.xml.xsig** es el archivo de firma digital temporal que produce el componente de destino.



Para generar el archivo de la firma haga clic en el botón **Guardar todos los resultados generados** de la barra de herramientas.
Esto genera los archivos .xml y .xsig en el directorio de salida.

El diseño de asignación **MarketingExpenses_EnvelopedSignature.mfd** dela carpeta . . . `\MapForceExamples` muestra el resultado que se obtiene cuando la firma está envuelta.



Validez del documento XML

Si la firma XML está incrustada en el documento XML, se añade al documento un elemento `Signature` en el espacio de nombres `http://www.w3.org/2000/09/xmldsig#`. Para que el documento siga siendo válido según el esquema, éste debe contener las declaraciones de elemento adecuadas. MapForce incrusta firmas cuando se elije la opción *Envuelta* en el cuadro de diálogo "Opciones de la firma XML": el elemento `Signature` se crea como último elemento secundario del elemento raíz (o de documento).

Nota: si no desea modificar el esquema del documento XML, la firma XML se puede crear en un archivo externo (basta con elegir la opción *Separada* en el cuadro de diálogo "Opciones de la firma XML").

© 2018 Altova GmbH

A continuación puede ver dos fragmentos de esquemas XML que muestran cómo se permite el elemento `Signature` de una firma envuelta. Puede usar estos ejemplos para guiarse a la hora de modificar sus propios esquemas.

En el primer esquema XML, se importó el esquema XML Signatura al esquema del usuario. El esquema XML Signature está en la dirección web http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd

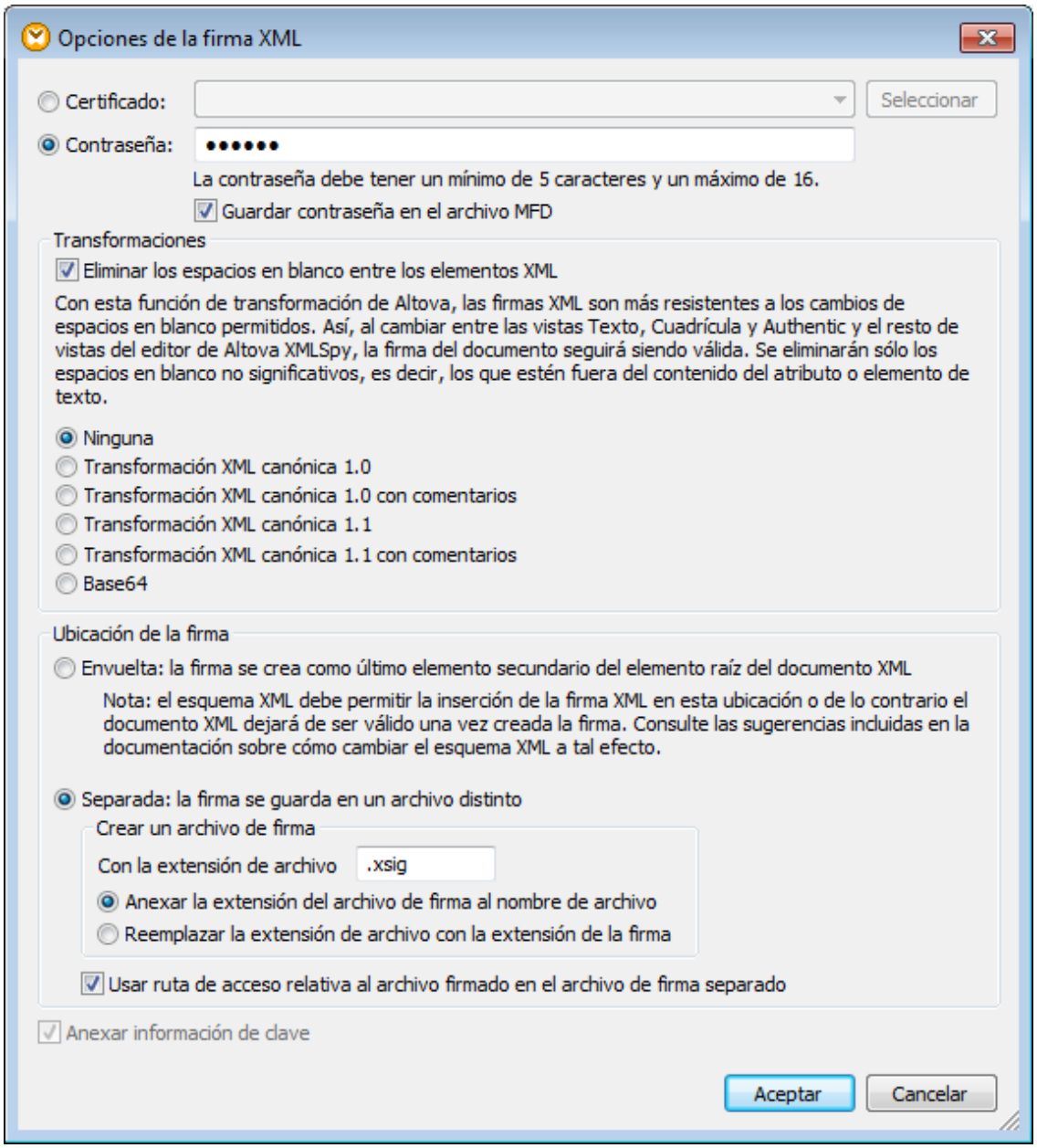
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:xsig="http://www.w3.org/2000/09/xmldsig#"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
   <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
              schemaLocation="http://www.w3.org/TR/xmldsig-core/xmldsig-core-schema.xsd"/>
   <xs:element name="Root">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="FirstChildOfRoot"/>
            <xs:element ref="SecondChildOfRoot" minOccurs="0"/>
            <xs:element ref="ThirdChildOfRoot" minOccurs="0"/>
            <xs:element ref="xsig:Signature" minOccurs="0"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   ...
</xs:schema>
```

En el segundo esquema XML se agregó un elemento comodín genérico que coincide con cualquier elemento de otros espacios de nombres. Al darle el valor `lax` al atributo `processContents`, el validador ignorará este elemento (porque no se encontró ninguna declaración de elemento coincidente). Por tanto, el usuario no necesita hacer referencia al esquema XML Signatures. El inconveniente de esta opción, sin embargo, es que cualquier elemento (no solo el elemento `Signature`) puede agregarse en la ubicación especificada del documento XML sin que ello provoque la invalidez del documento XML.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
   <xs:element name="Root">
      <xs:complexType>
         <xs:sequence>
            <xs:element ref="selection"/>
            <xs:element ref="newsitems" minOccurs="0"/>
            <xs:element ref="team" minOccurs="0"/>
            <xs:any namespace="##other" minOccurs="0" processContents="lax"/>
         </xs:sequence>
      </xs:complexType>
   </xs:element>
   ...
</xs:schema>
```

7.1.12.1 Configurar la firma digital XML

Las opciones de configuración de la firma se almacenan por separado por cada componente en el cuadro de diálogo "Configuración del componente" y todas las firmas se guardan en el archivo MFD.



Cuadro de diálogo "Opciones de la firma XML"

Método de autenticación: certificado o contraseña
La firma puede basarse en un certificado o en una contraseña. Seleccione el botón de opción del

© 2018 Altova GmbH

método que desea usar.

- *Certificado:*
  Si desea usar un certificado, éste debe tener una clave privada y encontrarse en un almacén de certificados accesible. La firma se genera con la clave privada del certificado. Para verificar la firma es necesario tener acceso al certificado (o a su versión de clave pública). La clave pública del certificado sirve para verificar la firma. Para seleccionar el certificado de clave pública/privada que se debe usar haga clic en el botón **Seleccionar** y navegue hasta el certificado.
- *Contraseña:*
  Introduzca una contraseña con una longitud comprendida entre 5 y 16 caracteres. Esta contraseña sirve para crear la firma y será necesaria posteriormente para verificar la firma. El botón **Aceptar** del cuadro de diálogo no se habilita hasta que no se cumple este requisito.
- *Guardar contraseña en el archivo MFD:*
  Si marca esta casilla, la contraseña introducida en el campo *Contraseña* se guardará en formato confuso en el archivo MFD (es decir, la contraseña se cifrará). Recuerde que cualquier usuario con acceso al archivo MFD puede crear firmas con esta contraseña.

Transformaciones

Los datos XML se transforman y el resultado de la transformación sirve para crear la firma. Puede especificar qué algoritmo de transformación canónica se debe aplicar a los datos XML del archivo (el contenido `SignedInfo`) antes de que se lleven a cabo los cálculos de firma. Debe tener en cuenta algunas diferencias entre las diferentes opciones de transformación:

- *Transformación canónica XML con o sin comentarios:*
  Si se incluyen comentarios para los cálculos de la firma y se realizan cambios en los comentarios de los datos XML, entonces se producirá un error de verificación. Si no se incluyen los comentarios, se podrán añadir o modificar comentarios en el documento XML una vez firmado y la firma se considerará válida y auténtica.

  Nota:
  "...con comentarios" es una opción que solo está disponible si se elige la opción de ubicación de la firma *Separada*.
- *Base64:*
  Se entiende que el elemento raíz (o de documento) del documento XML está codificado en base64 y se lee en su formato binario. Si el elemento raíz no es base64, se devuelve un error o el elemento se lee como vacío, dependiendo del tipo de elemento del que se trate.
- *Ninguna:*
  No se lleva a cabo transformación alguna y los datos XML del archivo binario guardado en disco se pasan directamente para la creación de firma. Si posteriormente se modifican los datos la firma no se podrá verificar.

  Sin embargo, si marca la casilla *Eliminar los espacios en blanco entre los elementos XML,* todos los espacios en blanco se eliminarán y los cambios en los espacios en blanco se pasarán por alto.

  La principal diferencia entre la opción *Ninguna* y la opción *Transformación canónica* es que esta última produce una secuencia de datos XML, en las que se normalizan algunas

diferencias, como el orden de los atributos por ejemplo. Por consiguiente, una transformación canónica normalizará cualquier cambio realizado, como el orden de los atributos, por ejemplo. Mientras que con la opción *Ninguna* este tipo de cambios se tendrán en cuenta a la hora de verificar la firma.

Sin embargo, recuerde que si la firma está incrustada (envuelta), MapForce realiza una transformación canónica predeterminada. Así que los datos XML se usarán tal y como están (es decir, sin transformarse) cuando: la firma esté separada, la opción Ninguna esté seleccionada y la casilla *Eliminar los espacios en blanco* esté desactivada.

Ubicación de la firma

La firma se puede ubicar dentro del archivo XML o se puede crear en un archivo separado. Estas son las opciones de configuración:

- *Envuelta:*
  El elemento de firma se crea como último elemento secundario del elemento raíz (o de documento). Nota: el esquema XML asociado debe tener los elementos de definición de firma para que el resultado XML sea válido. Consulte la información dada más arriba.
- *Separada:*
  La firma XML se crea en un archivo separado. En este caso puede especificar la extensión de archivo y si el nombre del archivo se crea: (i) anexando la extensión al nombre del archivo (p. ej. `test.xml.xsig`) o (ii) reemplazando la extensión XML del archivo XML (p. ej. `test.xsig`). También puede elegir si en el archivo de firma la referencia al archivo XML es una ruta de acceso relativa o absoluta.

Notas:

- Las firmas XML para archivos de esquema XML (`.xsd`) y archivos XBRL solamente se pueden crear como archivos de firma externos. Para archivos WSDL las firmas se pueden crear como archivos externos y "envolverse" en el archivo WSDL.
- Si la firma XML se crea en un archivo separado, entonces el archivo XML y la firma XML se asocian por medio de una referencia que está presente en el archivo de la firma. Por tanto, cuando la firma es un archivo externo, su verificación debe hacerse con el archivo de firma activo y no con el archivo XML activo.

Anexar información de clave

La casilla *Anexar información clave* se habilita si la firma está basada en un certificado. Si la firma está basada en una contraseña, esta casilla se deshabilita.

Si se marca esta casilla, la información de clave pública se pone dentro de la firma. Si no se marca esta casilla, esta información de clave no se incluye en la firma. La ventaja de incluir la información de clave en la firma es que no se necesitará el certificado (concretamente la información de clave pública que contiene) para el proceso de verificación (porque la información de clave está presente en la firma).

Configuración de firma no válida

MapForce no puede firmar digitalmente el resultado de la asignación si las opciones de la firma XML no son válidas. La configuración de la firma digital no es válida si:

- No se puede acceder al certificado seleccionado o si el certificado no es adecuado para la firma de documentos XML.

© 2018 Altova GmbH

- No se establece una contraseña (p. ej. no se marca la casilla *Guardar contraseña en el archivo MFD*).

Cuando se abre el panel *Resultados*, MapForce emite un mensaje:

- Para solicitar la contraseña: *Indique la contraseña que desea utilizar para firmar el resultado del componente "MarketingExpensesSigned".*
- Para solicitar el certificado: *Elija el almacén que contiene el certificado que desea utilizar para firmar el resultado del componente "MarketingExpensesSigned".*

Si no se elije contraseña ni certificado algunos, entonces el procesamiento se detiene o se continúa sin firma. Este comportamiento se puede determinar en el cuadro de diálogo "Configuración del componente", con los botones de opción *Dejar de procesar* o *Continuar sin la firma*.

Si la asignación se ejecuta desde la línea de comandos, entonces no aparece ningún aviso. La ejecución de la asignación se detiene y genera un error o continúa sin la firma.

7.2 Bases de datos en MapForce

Sitio web de Altova: Asignación de bases de datos

MapForce 2018 es totalmente compatible con la asignación de datos de bases de datos con archivos XML, archivos planos y otros formatos de base de datos. Además, con la edición Enterprise Edition podrá crear asignaciones de datos entre bases de datos y formatos EDI, Excel 2007+, JSON, XBRL y servicios web.

Las bases de datos compatibles y sus objetos raíz aparecen a continuación. Altova procura ofrecer compatibilidad con otras bases de datos, pero sólo se garantiza una conexión y un procesamiento correctos con las bases de datos de la lista. Si usa la versión de 64 bits de MapForce, compruebe que tiene acceso a los controladores de BD de 64 bits de la BD a la que quiere conectarse.

Base de datos	Objeto raíz	Observaciones
Firebird 2.5.4	database	
IBM DB2 8.x, 9.1, 9.5, 9.7, 10.1, 10.5	schema	
IBM DB2 for i 6.1, 7.1	schema	Los archivos lógicos son compatibles y se muestran en vistas.
IBM Informix 11.70	database	Informix es compatible con conexiones vía ADO, JDBC y ODBC. La implementación no es compatible con tipos de datos de objetos de gran dimensión en cualquiera de los idiomas de generación del código. MapForce generará un mensaje de error (durante el proceso de generación del código) si se usa cualquiera de este tipo de datos.
MariaDB 10	database	
Microsoft Access 2003, 2007, 2010, 2013	database	
Microsoft Azure SQL Database	database	SQL Server 2016 codebase
Microsoft SQL Server 2005, 2008, 2012, 2014, 2016	database	
MySQL 5.0, 5.1, 5.5, 5.6, 5.7	database	
Oracle 9i, 10g, 11g, 12c	schema	
PostgreSQL 8.0, 8.1, 8.2, 8.3, 9.0.10, 9.1.6, 9.2.1, 9.4, 9.5, 9.6	database	Son compatibles todas las conexiones PostgreSQL, tanto

© 2018 Altova GmbH

		nativas como basadas en controladores, a través de interfaces como ODBC o JDBC. Las conexiones nativas no necesitan controladores.
Progress OpenEdge 11.6	database	
SQLite 3.x	database	Las conexiones SQLite son conexiones nativas y directas compatibles con el archivo de base de datos de SQLite. No se precisan controladores separados.
Sybase ASE 15, 16	database	
Teradata16	database	Son compatibles las conexiones vía ADO.NET, JDBC y ODBC. Cuando una asignación inserta datos en una tabla de BD, los campos de identidad generados por la base de datos no son compatibles.

En MapForce puede usar bases de datos como componentes de asignación en todos los lenguajes de transformación, con la excepción de SQLite. Los componentes SQLite solo son compatibles con el lenguaje de transformación integrado.

Si tiene asignaciones con conexiones a bases de datos y ejecuta el archivo de ejecución de la asignación (.mfx) en un equipo servidor con MapForce Server, también necesitará crear conexiones de base de datos en el servidor de destino. Si el sistema operativo de destino es Linux o Mac OS, las conexiones por ADO y ODBC no son compatibles (*véase* *Conexiones de base de datos en Linux y Mac*).

Una opción para ahorrar tiempo y reducir la complejidad de las asignaciones es definir las conexiones de base de datos como recursos globales.

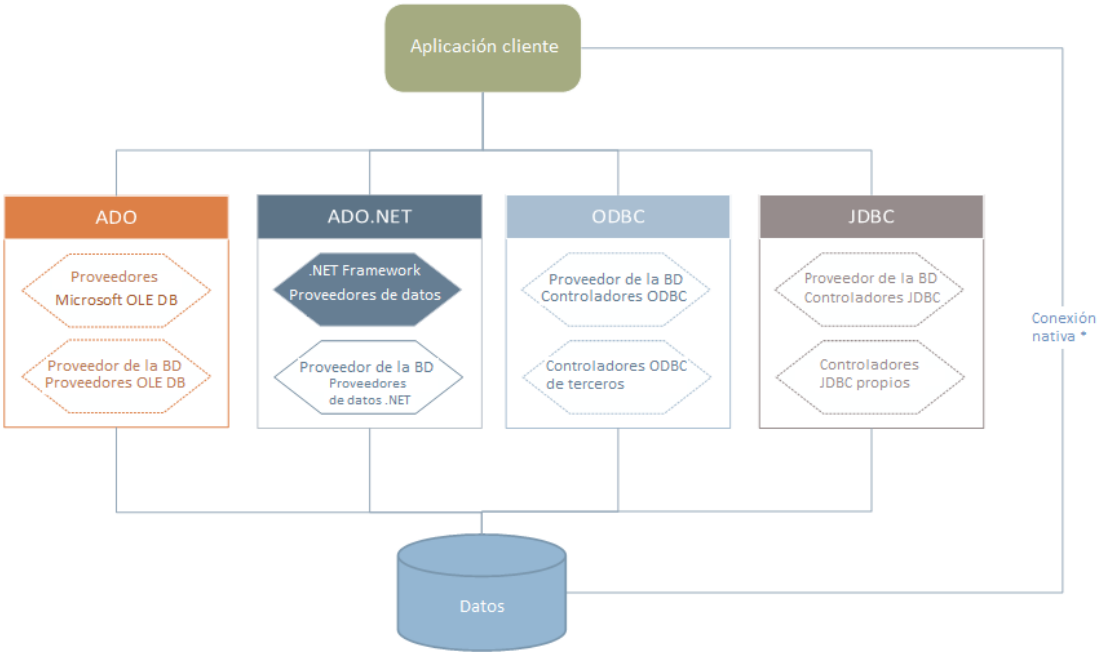
7.2.1 Conectarse a una base de datos

En su definición más sencilla, una base de datos es un archivo local como un archivo de base de datos Microsoft Access o SQLite. En casos más complejos, una base de datos puede residir en un servidor de bases de datos remoto o de la red que no tienen por qué usar el mismo sistema operativo que la aplicación que se conecta a la BD y que consume los datos. Por ejemplo, mientras que MapForce se puede ejecutar en sistemas operativos Windows, puede que la base de datos a la que desea acceder (p. ej. una base de datos MySQL) esté en un equipo Linux.

Para interactuar con los diferentes tipos de bases de datos, MapForce se sirve de las interfaces de conexión de datos y los controladores de BD disponibles en su sistema operativo o publicados

por los principales proveedores de BD. La tecnología de base de datos evoluciona constantemente y por tanto consideramos que este mecanismo garantiza la mayor compatibilidad y flexibilidad en las principales plataformas.

En el siguiente diagrama puede ver un resumen de las opciones de conectividad de datos que pueden existir entre MapForce (*aplicación cliente*) y un almacén de datos (que puede ser un archivo o un servidor de base de datos).



** Las bases de datos SQLite y PostgreSQL admiten conexiones nativas directas. Para conectarse a este tipo de bases de datos no es necesario tener instalado controladores específicos.*

Por tanto, tal y como muestra el diagrama, desde MapForce puede acceder a los principales tipos de BD con estas tecnologías de acceso de datos:

- ADO (Microsoft® ActiveX® Data Objects), que a su vez utiliza OLE DB
- ADO.NET (un conjunto de bibliotecas disponibles en Microsoft .NET Framework que permiten la interacción con datos)
- JDBC (Java Database Connectivity)
- ODBC (Open Database Connectivity)

Algunos proveedores de ADO.NET no son compatibles o tienen una compatibilidad limitada. Consulte el apartado *Notas sobre compatibilidad con ADO.NET* para obtener más información.

La interfaz de conexión de datos que se debe utilizar dependerá de la infraestructura de software con la que cuente. Lo normal es elegir una tecnología de acceso de datos y un controlador de base de datos que se integre bien con el sistema de base de datos al que desea conectarse. Por

ejemplo, para conectarse con una base de datos Microsoft Access 2013, puede generar una cadena de conexión ADO que utilice un proveedor nativo como **Microsoft Office Access Database Engine OLE DB Provider**. Para conectarse a Oracle, por su parte, lo mejor será descargar e instalar las interfaces JDBC, ODBC o ADO .NET más recientes del sitio web de Oracle.

Lo más probable es que los controladores de los productos Windows (como Microsoft Access o SQL Server) ya estén disponibles en su sistema, pero puede que necesite descargar controladores para otros tipos de bases de datos. Los principales proveedores de BD publican software cliente y controladores con frecuencia. Además, puede encontrar otros controladores de otras organizaciones para las tecnologías de acceso de datos mencionadas. En la mayoría de los casos hay varias maneras de conectarse a una base de datos. Las características y el rendimiento de la base de datos dependerán de la tecnología de acceso o de los controladores utilizados.

7.2.1.1 Iniciar el asistente para la conexión de base de datos

Cada vez que lleve a cabo una acción que requiera una conexión a una base de datos, aparecerá un asistente que le guiará durante la conexión.

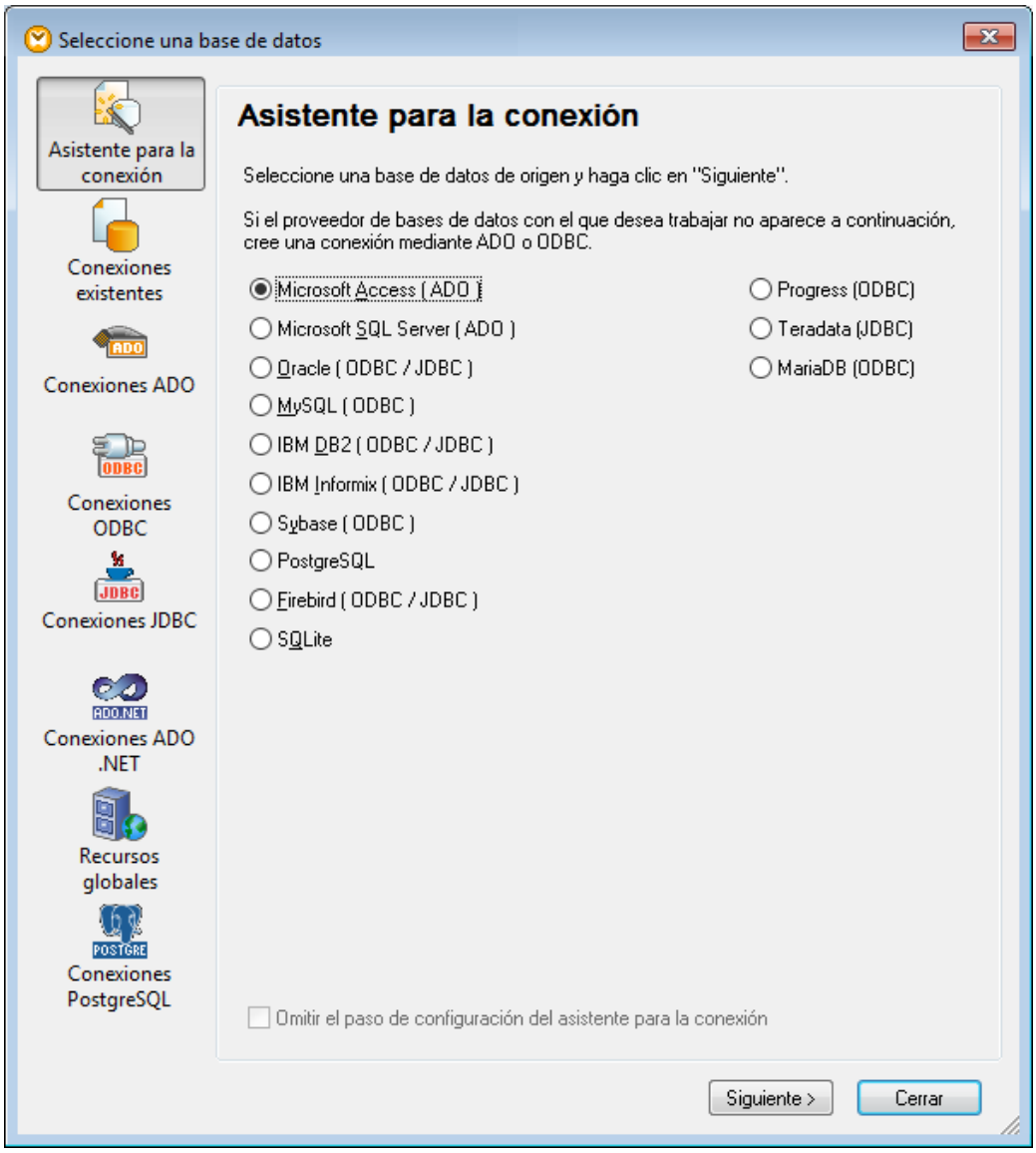
Antes de empezar a seguir las instrucciones del asistente, debe tener en cuenta que algunos tipos de BD requieren instalar y configurar algunas características, como controladores o software cliente. Para ver una lista de controladores ordenados por tipo de base de datos, consulte el apartado Resumen de controladores de base de datos.

Para agregar la base de datos como componente de origen o destino de una asignación:

- En el menú **Insertar** haga clic en **Base de datos**.

Para agregar la base de datos como recurso global reutilizable:

1. En el menú **Herramientas** haga clic en **Recursos globales**.
2. Haga clic en **Agregar** y después en **Base de datos**.
3. Haga clic en **Elegir base de datos**.



Tras seleccionar el tipo de BD haga clic en **Siguiente** y siga las instrucciones que aparecen en pantalla, que dependerán del tipo de BD, de la tecnología de acceso (ADO, ADO.NET, ODBC, JDBC) y del controlador utilizados.

Para ver ejemplos de conexión de cada tipo de BD consulte el apartado Ejemplos de conexión a bases de datos. También puede consultar estos temas:

- Configurar una conexión ADO
- Configurar una conexión ADO.NET
- Configurar una conexión ODBC
- Configurar una conexión JDBC

© 2018 Altova GmbH

7.2.1.2 Resumen de controladores de base de datos

En la tabla que aparece más abajo puede ver una lista de controladores de BD que puede utilizar para conectarse a una base de datos a través de las diferentes tecnologías de acceso de datos. Esta lista no contiene todos los controladores disponibles en la actualidad y, por tanto, puede usar otros controladores que no aparecen en la lista.

Aunque puede que Windows venga con algunos controladores de BD, es posible que necesite descargar más controladores. Por lo general, se recomienda utilizar el controlador más reciente publicado por el proveedor de la BD.

Los proveedores de bases de datos pueden ofrecer controladores para descargar en paquetes o junto con el software cliente de base de datos. En este último caso, el cliente de base de datos suele incluir todos los controladores necesarios u ofrecer una opción durante la instalación para seleccionar los controladores y componentes que el usuario desee instalar. El software cliente de base de datos suele incluir funciones de administración y configuración que permiten simplificar la administración de la BD y documentación que explica cómo instalar y configurar el cliente y sus componentes.

Es muy importante configurar bien el cliente de BD para establecer correctamente la conexión con la BD. Antes de instalar y usar el software cliente de BD, recomendamos que lea detenidamente las instrucciones de instalación y configuración porque pueden variar según la versión de la BD y según la versión de Windows.

Si desea conocer las características y limitaciones de cada tecnología de acceso de datos con respecto a cada tipo de BD, consulte la documentación de la BD correspondiente y pruebe la conexión. Debe tener en cuenta estos puntos para evitar problemas de conexión:

- Algunos proveedores de ADO.NET no son compatibles o tienen una compatibilidad limitada (*véase Notas sobre compatibilidad con ADO.NET*).
- Cuando instale un controlador de BD, recomendamos que tenga la misma plataforma que la aplicación de Altova (32 o 64 bits). Por ejemplo, si usa una aplicación de Altova de 32 bits en un sistema operativo de 64 bits, instale el controlador de 32 bits y configure la conexión de base de datos con ayuda del controlador de 32 bits (*véase Ver los controladores ODBC disponibles*).
- Cuando configure un origen de datos ODBC, recomendamos crear el nombre de origen de datos (DSN) como DSN del sistema en lugar de como DSN de usuario (*véase Configurar una conexión ODBC*).
- Cuando configure un origen de datos JDBC, compruebe que tiene instalado JRE (Java Runtime Environment) o Java Development Kit (JDK) y que la variable de entorno CLASSPATH del sistema operativo está configurado (*véase Configurar una conexión JDBC*).
- Si necesita consultar las instrucciones de instalación e información sobre compatibilidad de los controladores o clientes de BD, consulte la documentación que viene con el paquete de instalación.

Base de datos	Interfaz	Controladores
Firebird	ADO.NET	Proveedor de datos ADO.NET (https://www.firebirdsql.org/en/

Base de datos	Interfaz	Controladores
		additional-downloads/)
	JDBC	Controlador Firebird JDBC (https://www.firebirdsql.org/en/jdbc-driver/)
	ODBC	Controlador Firebird ODBC (https://www.firebirdsql.org/en/odbc-driver/)
IBM DB2	ADO	Proveedor IBM OLE DB para DB2
	ADO.NET	Proveedor IBM Data Server para .NET
	JDBC	Controlador IBM Data Server para JDBC y SQLJ
	ODBC	Controlador IBM DB2 ODBC
IBM DB2 for i	ADO	• Proveedor IBM DB2 para i5/OS IBMDA400 OLE DB • Proveedor IBM DB2 para i5/OS IBMDARLA OLE DB • Proveedor IBM DB2 para i5/OS IBMDASQL OLE DB
	ADO.NET	Proveedor de datos .NET Framework Data Provider para IBM i
	JDBC	Controlador IBM Toolbox para Java JDBC
	ODBC	Controlador iSeries Access ODBC
IBM Informix	ADO	Controlador IBM Informix OLE DB
	JDBC	Controlador IBM Informix JDBC
	ODBC	Controlador IBM Informix ODBC
Microsoft Access	ADO	• **Proveedor de Microsoft Jet OLE DB** • **Proveedor de Microsoft Access Database Engine OLE DB**
	ADO.NET	**Proveedor de datos de .NET Framework para OLE DB**
	ODBC	• **Controlador de Microsoft Access**
MariaDB	ADO.NET	Si falta el conector .NET especial para MariaDB, use **Connector/NET** pararMySQL (https://dev.mysql.com/downloads/connector/net/).
	JDBC	MariaDB Connector/JDBC (https://downloads.mariadb.org/)
	ODBC	MariaDB Connector/ODBC (https://downloads.mariadb.org/)
Microsoft SQL Server	ADO	• **Proveedor de Microsoft OLE DB para SQL Server** • **Cliente nativo de SQL Server**
	ADO.NET	• **Proveedor de datos .NET Framework para SQL Server** • **Proveedor de datos .NET Framework para OLE DB**
	JDBC	• Controlador JDBC de Microsoft para SQL Server (https://msdn.microsoft.com/library/mt484311.aspx)

© 2018 Altova GmbH

Base de datos	Interfaz	Controladores
	ODBC	• **Cliente nativo de SQL Server**
MySQL	ADO.NET	Conector/NET (https://dev.mysql.com/downloads/connector/net/)
	JDBC	Conector/JDBC (https://dev.mysql.com/downloads/connector/j/)
	ODBC	Conector/ODBC (https://dev.mysql.com/downloads/connector/odbc/)
Oracle	ADO	• Proveedor de datos Oracle para OLE DB • **Proveedor de Microsoft OLE DB para Oracle**
	ADO.NET	Proveedor de datos Oracle para .NET (https://www.oracle.com/technetwork/topics/dotnet/index-085163.html)
	JDBC	• Controlador JDBC Thin • Controlador JDBC Oracle Call Interface (OCI) Estos controladores se suelen instalar durante la instalación del cliente de base de datos Oracle. Conéctese con el controlador OCI (no con Thin) si usa el componente Oracle XML DB.
	ODBC	• **Microsoft ODBC para Oracle** • Controlador Oracle ODBC (por lo general se instala durante la instalación del cliente de base de datos Oracle)
PostgreSQL	JDBC	Controlador PostgreSQL JDBC (https://jdbc.postgresql.org/download.html)
	ODBC	psqlODBC (https://odbc.postgresql.org/)
	Conexión nativa	Disponible. No necesita instalar ningún controladores si usa una conexión nativa.
Progress OpenEdge	JDBC	Conector JDBC (https://www.progress.com/jdbc/openedge)
	ODBC	Conector ODBC (https://www.progress.com/odbc/openedge)
SQLite	Conexión nativa	Disponible. No necesita instalar ningún controladores si usa una conexión nativa.
Sybase	ADO	Proveedor de Sybase ASE OLE DB
	JDBC	jConnect™ para JDBC
	ODBC	Controlador Sybase ASE ODBC
Teradata	ADO.NET	Proveedor de datos .NET para Teradata (https://downloads.teradata.com/download/connectivity/net-data-provider-for-teradata)
	JDBC	Controlador JDBC de Teradata (https://downloads.teradata.com/download/connectivity/jdbc-driver)
	ODBC	Controlador ODBC de Teradata para Windows (https://

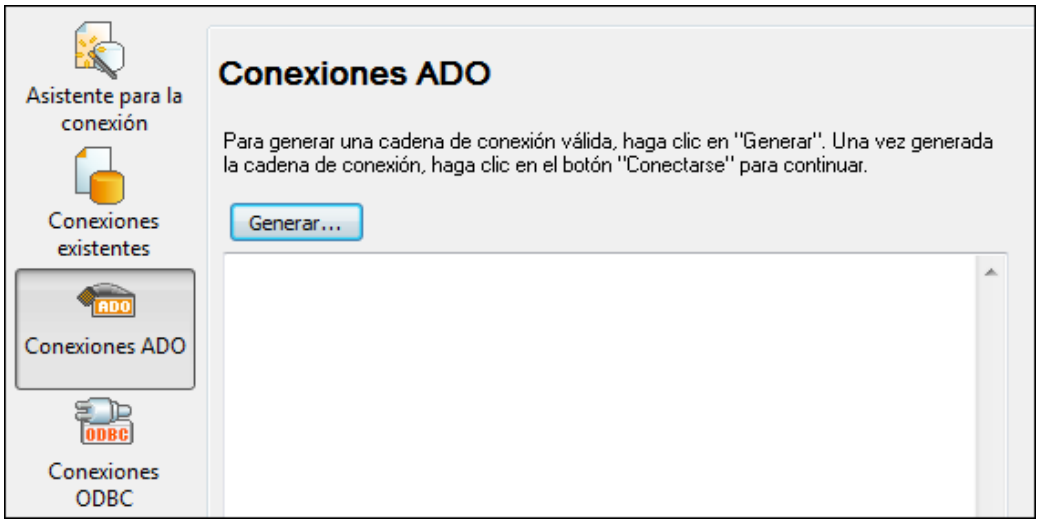
Base de datos	Interfaz	Controladores
		downloads.teradata.com/download/connectivity/odbc-driver/windows)

7.2.1.3 Configurar una conexión ADO

Microsoft ActiveX Data Objects (ADO) es una tecnología de acceso de datos que permite conectarse a gran variedad de orígenes de datos con OLE DB. OLE DB es una interfaz alternativa a ODBC y JDBC. Ofrece acceso uniforme a los datos en un entorno COM (Component Object Model). ADO suele utilizarse para conectarse a bases de datos Microsoft nativas como Microsoft Access o SQL Server, aunque también puede usar ADO para otros orígenes de datos.

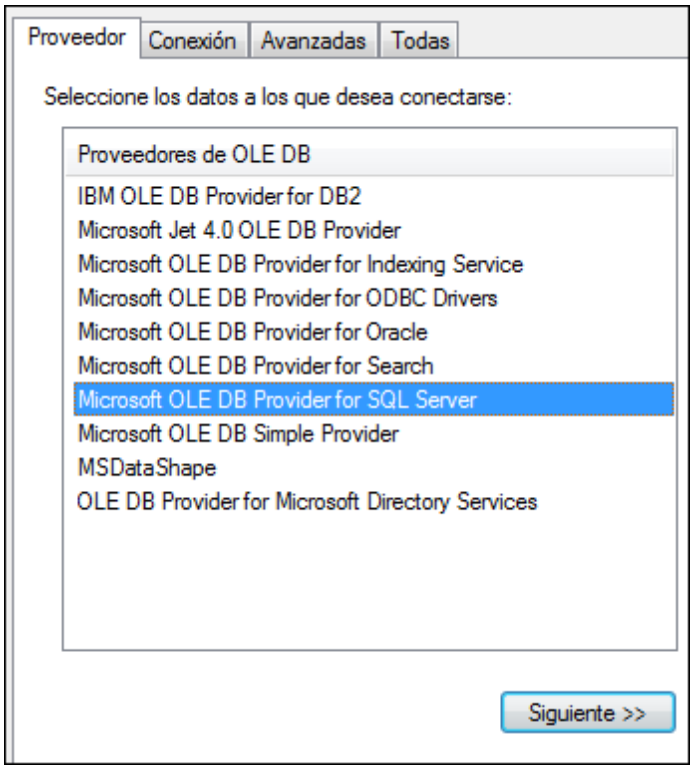
Para configurar una conexión ADO:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones ADO**.



3. Haga clic en **Generar**.

© 2018 Altova GmbH



4. Seleccione el proveedor de datos que desea utilizar para establecer la conexión. La tabla que aparece a continuación enumera las combinaciones más comunes.

Para conectarse a esta BD...	**Utilice este proveedor...**
Microsoft Access	• **Microsoft Office Access Database Engine OLE DB Provider** Si se conecta a Microsoft Access 2003, también puede usar **Microsoft Jet OLE DB Provider**.
SQL Server	• **SQL Server Native Client** • **Microsoft OLE DB Provider for SQL Server**
Otras bases de datos	Seleccione el proveedor correspondiente. Si no hay un proveedor OLE DB para su BD, instale el controlador necesario que ofrece el proveedor de la BD (consulte el Resumen de controladores de base de datos). Otra opción es configurar una conexión ODBC o JDBC. Si el sistema operativo tiene un controlador ODBC para la BD, también puede usar **Microsoft OLE DB Provider for ODBC Drivers**.

5. Haga clic en **Siguiente**.

Las instrucciones de las siguientes pantallas dependen del proveedor elegido. Para SQL Server deberá dar o seleccionar el nombre de host del servidor de BD y el nombre de usuario y la contraseña de la base de datos. Para Microsoft Access, deberá dar o seleccionar la ruta de acceso del archivo de BD.

La lista de propiedades de inicialización (parámetros de conexión) aparece en la pestaña Todas del cuadro de diálogo de conexión. Estas propiedades dependen del proveedor elegido. Consulte estos apartados para aprender a configurar las propiedades básicas de inicialización para bases de datos Microsoft Access y SQL Server:

- Configurar las propiedades de vínculo de datos de SQL Server
- Configurar las propiedades de vínculo de datos de Microsoft Access

7.2.1.3.1 Conectarse a una base de datos Microsoft Access

Siga estas instrucciones si desea conectarse a una base de datos Microsoft Access que no requiere contraseña. Si la base de datos está protegida con contraseña, entonces consulte el apartado Conectarse a Microsoft Access (ADO).

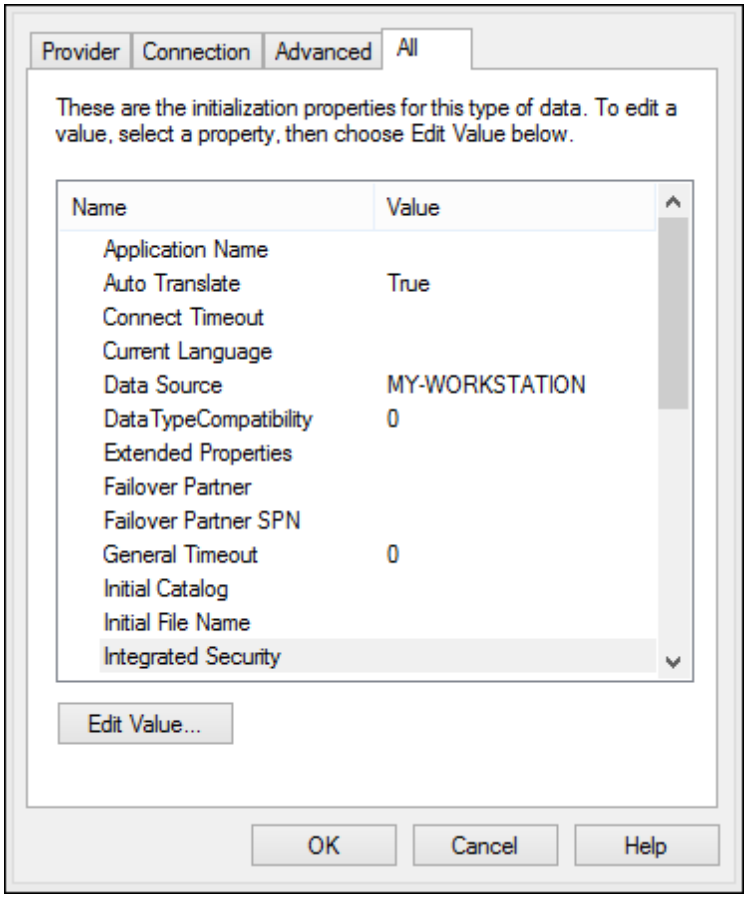
Para conectarse a una BD Microsoft Access:

1. Inicie el asistente para la conexión de base de datos.
2. Seleccione el botón de opción *Microsoft Access (ADO)* y haga clic en **Siguiente**.
3. Busque el archivo de BD o introduzca su ruta de acceso (relativa o absoluta).
4. Haga clic en **Conectarse**.

7.2.1.3.2 Configurar las propiedades de vínculo de datos de SQL Server

Cuando se conecte a una base de datos Microsoft SQL Server por ADO (consulte Configurar una conexión ADO), asegúrese de configurar correctamente estas propiedades de vínculo de datos en la pestaña *Todas* del cuadro de diálogo "Propiedades de vínculo de datos".

© 2018 Altova GmbH

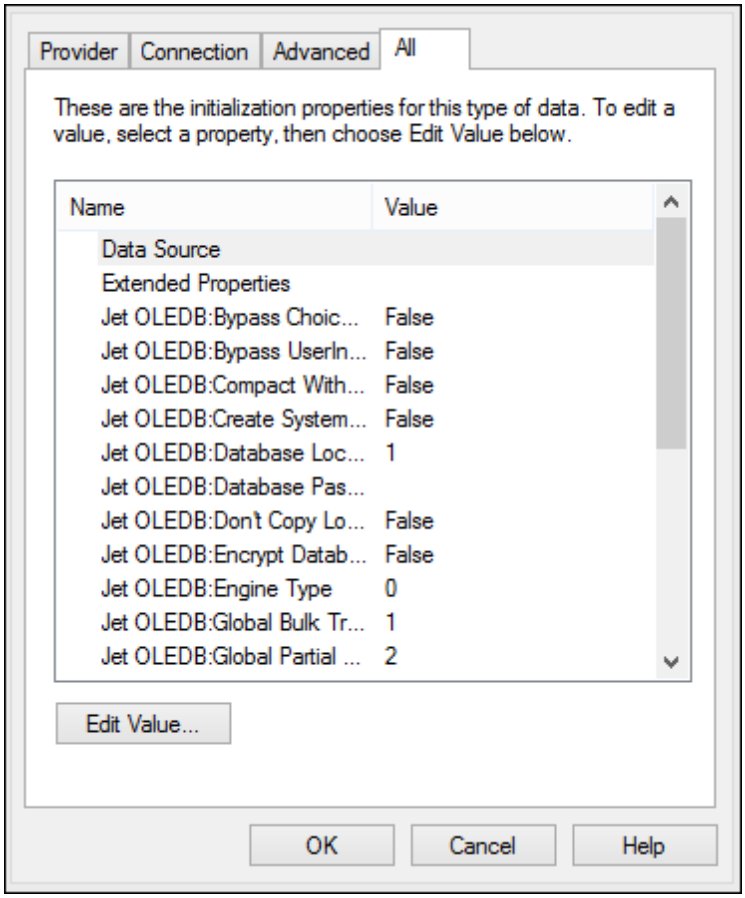


Data Link Properties dialog box

Propiedad	Notas
Seguridad integrada	Si seleccionó el proveedor de datos **SQL Server Native Client** en la pestaña *Proveedor*, esta propiedad debe ser un carácter de espacio en blanco.
Almacenar información de seguridad	Esta propiedad debe tener el valor **True**.

7.2.1.3.3 Configurar las propiedades de vínculo de datos de Microsoft Access

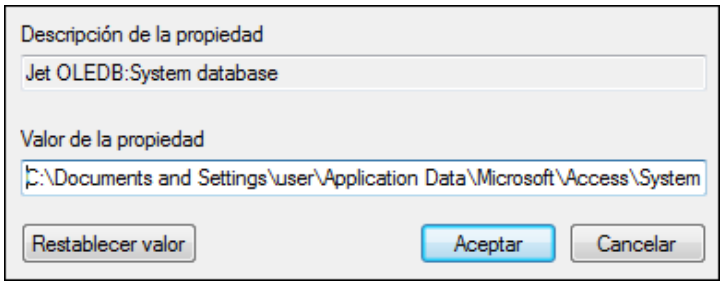
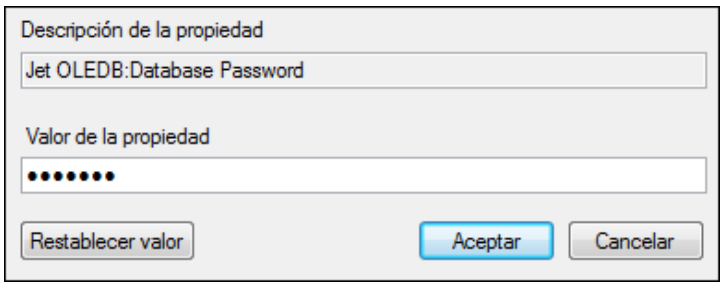
Cuando se conecte a una base de datos Microsoft Access por ADO (consulte Configurar una conexión ADO), asegúrese de configurar correctamente estas propiedades de vínculo de datos en la pestaña *Todas* del cuadro de diálogo "Propiedades de vínculo de datos".



Data Link Properties dialog box

Propiedad	Notas
Origen de datos	Esta propiedad almacena la ruta de acceso del archivo de BD Microsoft Access. Para evitar problemas de conexión, recomendamos usar el formato de ruta de acceso UNC. Por ejemplo: `\\servidor\compartir$\rutaArchivo`
Base de datos OLEDB:System Database	Esta propiedad almacena la ruta de acceso del archivo de información de grupo de trabajo. Quizás sea necesario configurar el valor de esta propiedad para poder establecer la conexión con la BD Microsoft Access. Si se produce un error relacionado con el archivo de información de grupo de trabajo, busque el archivo de información de grupo de trabajo (**System.MDW**) que corresponda a su perfil de usuario e introduzca la ruta de acceso del archivo **System.MDW** como valor de esta propiedad.

© 2018 Altova GmbH

Propiedad	Notas
	
Contraseña Jet OLEDB:Database	Si la base de datos está protegida con contraseña, el valor de esta propiedad debe ser la contraseña de la base de datos. 

7.2.1.4 Configurar una conexión ADO.NET

ADO.NET es un conjunto de bibliotecas de Microsoft .NET Framework diseñado para interactuar con datos, incluidos datos de bases de datos. Para conectarse a una base de datos desde MapForce por ADO.NET es necesario tener instalado Microsoft .NET Framework 4 o superior. Como puede ver más abajo, la conexión a la base de datos a través de ADO.NET se hace seleccionando un proveedor .NET y aportando una cadena de conexión.

Un proveedor de datos .NET es una colección de clases que permite conectarse a un tipo concreto de origen de datos (p. ej. un servidor SQL Server o una base de datos Oracle), ejecutar comandos en él y recuperar sus datos. En otras palabras, con ADO .NET las aplicaciones como MapForce interactúan con una base de datos a través de un proveedor de datos. Cada proveedor de datos está optimizado para poder trabajar con el tipo concreto de origen de datos para el que está diseñado. Hay dos tipos de proveedores .NET:

1. El proveedor que viene por defecto con Microsoft .NET Framework.
2. Proveedores que ofrecen los principales proveedores de bases de datos como extensión para .NET Framework. Este tipo de proveedores ADO.NET deben instalarse por separado y por lo general se pueden descargar del sitio web del correspondiente proveedor de base de datos.

Nota: ciertos proveedores ADO.NET son incompatibles o tienen una compatibilidad limitada. Consulte el apartado Notas sobre compatibilidad con ADO.NET.

Para configurar una conexión ADO.NET:

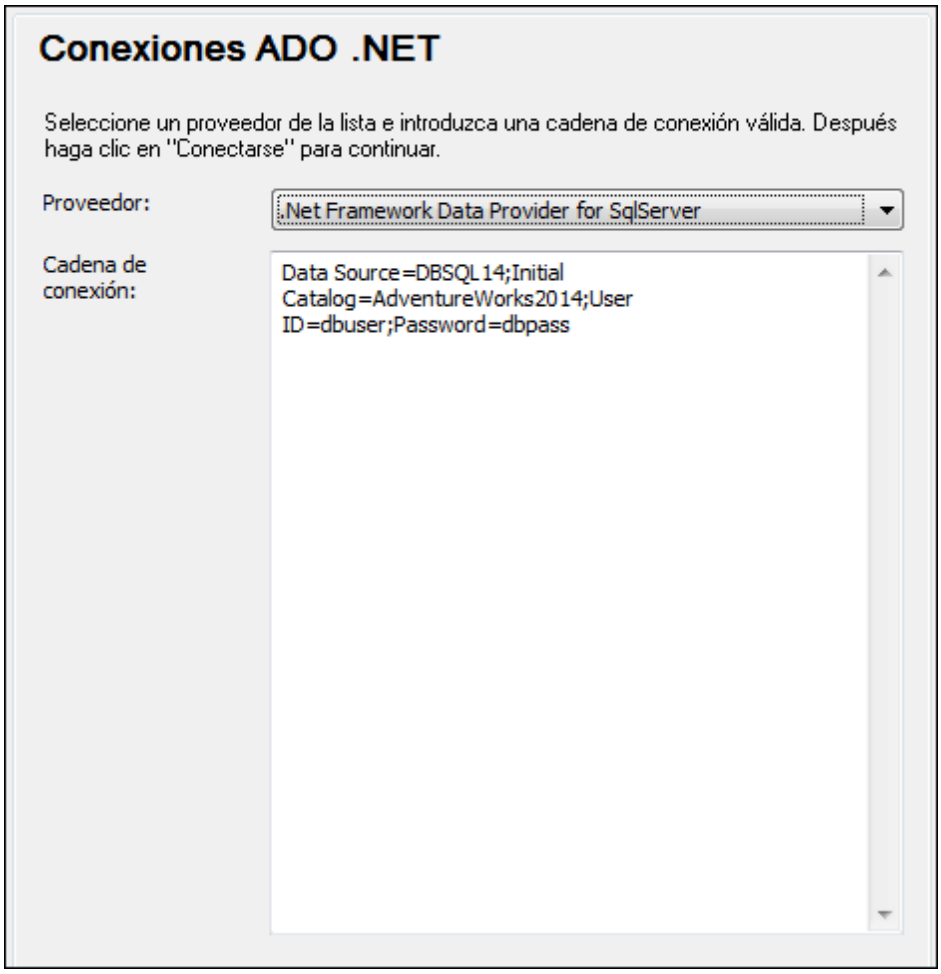
1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones ADO.NET**.
3. Seleccione un proveedor de datos .NET de la lista.

> En la lista *Proveedores* aparecen todos los proveedores que vienen por defecto con .NET Framework. Los proveedores de datos .NET propios del proveedor de la base de datos solo aparecen en la lista si ya están instalados en el sistema. Deben instalarse en el caché global de ensamblados (GAC) con ayuda del archivo .msi o .exe que ofrece el proveedor de la base de datos.

4. Introduzca la cadena de conexión con la base de datos. Una cadena de conexión defina la información de conexión con la base de datos y está formada por pares clave/valor de parámetros de conexión delimitados por caracteres de punto y coma. Por ejemplo, la cadena de conexión **`Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;Password=dbpass`** permite conectarse a la base de datos SQL Server `ProductsDB` del servidor `DBSQLSERV`, con el nombre de usuario `dbuser` y la contraseña `dbpass`. Puede crear la cadena de conexión tecleando los pares clave/valor directamente en el cuadro de texto *Cadena de conexión*, pero también puede crearla con Visual Studio (*véase Crear una cadena de conexión en Visual Studio*).

> La sintaxis de la cadena de conexión depende del proveedor que se seleccione en la lista *Proveedores*. Para ver un ejemplo consulte el apartado Ejemplo: cadenas de conexión ADO.NET.

© 2018 Altova GmbH



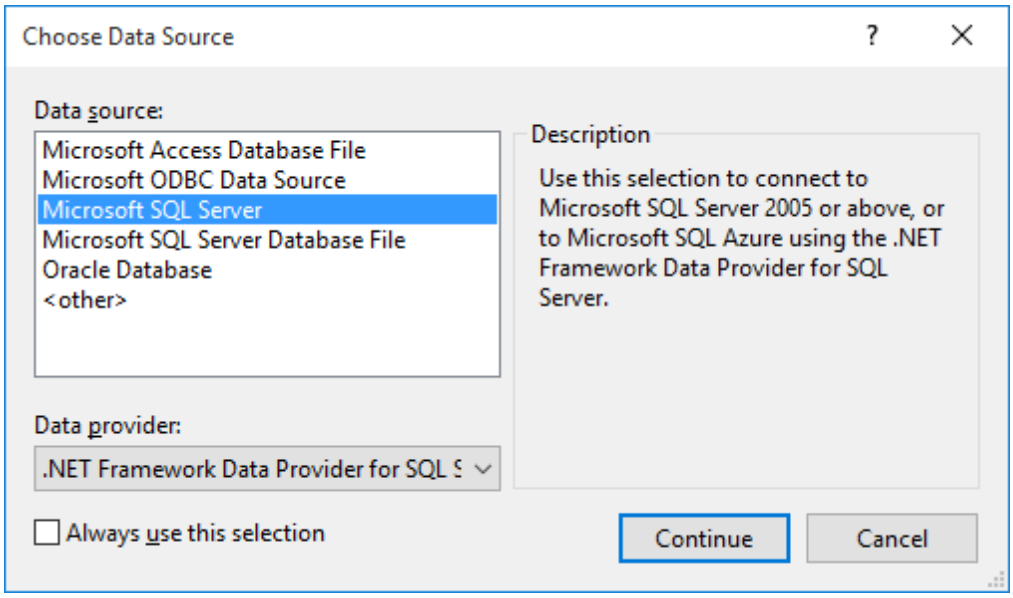
5. Haga clic en **Conectarse** para terminar.

7.2.1.4.1 Crear una cadena de conexión en Visual Studio

Para conectarse a un origen de datos usando ADO.NET se necesita una cadena de conexión válida. A continuación explicamos cómo crear una cadena de conexión desde Visual Studio.

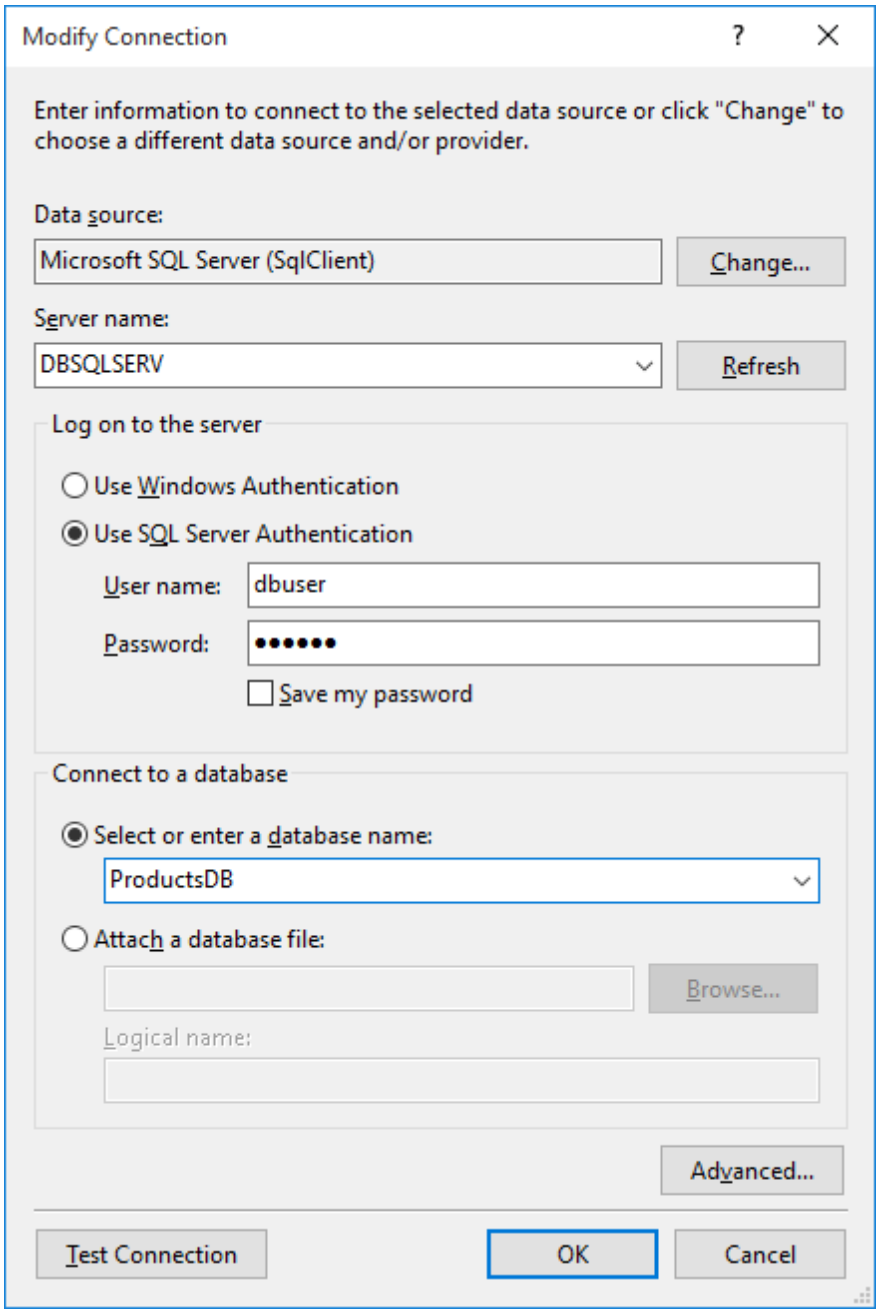
Para crear una cadena de conexión en Visual Studio:

1. En el menú **Herramientas** haga clic en **Conectar a base de datos**.
2. Seleccione un origen de datos de la lista (en este ejemplo Microsoft SQL Server). El proveedor de datos se rellena automáticamente en función de la opción elegida.



3. Haga clic en **Continuar**.

© 2018 Altova GmbH



4. Introduzca el nombre de host del servidor, el nombre de usuario y la contraseña de la base de datos. En este ejemplo nos conectamos a la base de datos `ProductsDB` en el servidor `DBSQLSERV`, usando SQL Server para la autenticación.
5. Para terminar haga clic en **Aceptar**.

Si la conexión se establece correctamente, aparecerá en la ventana Explorador de servidores. Para abrir esta ventana puede usar el comando **Vista | Explorador de servidores**. Para obtener la cadena de conexión con la base de datos, haga clic con el botón derecho en la conexión en la ventana Explorador de servidores y seleccione el comando **Propiedades**. Ahora aparece la cadena de conexión en la ventana Propiedades de Visual Studio. Recuerde que debe reemplazar los asteriscos con la contraseña antes de pegar la cadena en el cuadro de texto *Cadena de*

conexión de MapForce.

7.2.1.4.2 Ejemplo: cadenas de conexión ADO.NET

Para configurar una conexión ADO.NET deberá seleccionar un proveedor ADO.NET en el cuadro de diálogo de conexión a la base de datos y deberá introducir una cadena de conexión (*véase Configurar una conexión ADO.NET*). A continuación ofrecemos ejemplos de cadenas de conexión ADO.NET para varias bases de datos.

Proveedor de datos .NET para Teradata

Este proveedor se puede descargar del sitio web de Teradata (https://downloads.teradata.com/download/connectivity/net-data-provider-for-teradata). Un ejemplo de cadena de conexión sería:

```
Data Source=DirecciónServidor;User Id=usuario;Password=contraseña;
```

Proveedor de datos .NET Framework Data Provider para IBM i

Este proveedor se instala con *IBM i Access Client Solutions - Windows Application Package*. Un ejemplo de cadena de conexión sería:

```
DataSource=DirecciónServidor;UserID=usuario;Password=contraseña;DataCompression=True;
```

Para más información consulte el archivo de ayuda llamado ".NET Provider Technical Reference" que viene con el paquete de instalación.

Proveedor de datos .NET Framework Data Provider para MySQL

Este proveedor se puede descargar del sitio web de MySQL (https://dev.mysql.com/downloads/connector/net/). Un ejemplo de cadena de conexión sería:

```
Server=127.0.0.1;Uid=root;Pwd=12345;Database=test;
```

Véase también https://dev.mysql.com/doc/connector-net/en/connector-net-programming-connecting-connection-string.html

Proveedor de datos .NET Framework Data Provider para SQL Server

Un ejemplo de cadena de conexión sería:

```
Data Source=DBSQLSERV;Initial Catalog=ProductsDB;User ID=dbuser;Password=dbpass
```


© 2018 Altova GmbH

Véase también https://msdn.microsoft.com/en-us/library/ms254500(v=vs.110).aspx

Proveedor de datos IBM DB2 Data Provider 10.1.2 para .NET Framework 4.0

```
Database=PRODUCTS;UID=usuario;Password=contraseña;Server=localhost:50000;
```

Nota: por lo general este proveedor se instala con el paquete de IBM DB2 Data Server Client. Si después de instalar el paquete de IBM DB2 Data Server Client el proveedor no aparece en la lista de proveedores ADO.NET, consulte esta nota técnica: https://www-01.ibm.com/support/docview.wss?uid=swg21429586.

Véase también https://www.ibm.com/support/knowledgecenter/en/SSEPGG_10.1.0/com.ibm.swg.im.dbclient.adonet.ref.doc/doc/DB2ConnectionClassConnectionStringProperty.html

Proveedor de datos Oracle para .NET (ODP.NET)

El paquete de instalación que incluye el proveedor ODP.NET se puede descargar del sitio web de Oracle (http://www.oracle.com/technetwork/topics/dotnet/downloads/index.html). Un ejemplo de cadena de conexión sería:

```
Data Source=DSORCL;User Id=usuario;Password=contraseña;
```

En esta cadena de conexión, `DSORCL` es el nombre del origen de datos que apunta a un nombre de servicio Oracle que está definido en el archivo **tnsnames.ora** (tal y como se describe en el apartado Conectarse a Oracle (ODBC)).

Si prefiere establecer la conexión sin configurar un nombre de servicio en el archivo **tnsnames.ora**, entonces use una cadena de conexión parecida a esta:

```
Data Source=(DESCRIPTION=(ADDRESS_LIST=(ADDRESS=(PROTOCOL=TCP)(HOST=host)
(PORT=puerto)))(CONNECT_DATA=(SERVER=DEDICATED)
(SERVICE_NAME=MyOracleSID)));User Id=usuario;Password=contraseña;
```

Véase también https://docs.oracle.com/cd/B28359_01/win.111/b28375/featConnecting.htm

7.2.1.4.3 Notas sobre compatibilidad con ADO.NET

En esta tabla aparecen controladores de base de datos ADO.NET conocidos que por ahora no son compatibles con MapForce o que tienen una compatibilidad limitada.

Base de datos	Controlador	Notas
Todas las bases de datos	**Proveedor de datos .Net Framework para ODBC**	Compatibilidad limitada. Existen problemas conocidos con las conexiones Microsoft Access. Es mejor utilizar conexiones directas

Base de datos	Controlador	Notas
		ODBC (véase *Configurar una conexión ODBC*).
	Proveedor de datos .Net Framework para OleDb	Compatibilidad limitada. Existen problemas conocidos con las conexiones Microsoft Access. Es mejor utilizar conexiones directas ADO (véase *Configurar una conexión ADO*).
Firebird	**Proveedor de datos Firebird ADO.NET**	Compatibilidad limitada. Es mejor utilizar ODBC o JDBC (véase *Conectarse a Firebird (ODBC)* y *Conectarse a Firebird (JDBC)*).
Informix	**Proveedor de datos IBM Informix para .NET Framework 4.0**	Incompatible. Utilice el proveedor **DB2 Data Server Provider**.
IBM DB2 for i (iSeries)	**Proveedor de datos .Net Framework para i5/OS**	Incompatible. utilice el **proveedor de datos .Net Framework para IBM i**, que se instala con el paquete *IBM i Access Client Solutions - Windows Application Package*.
Oracle	**Proveedor de datos .Net Framework para Oracle**	Compatibilidad limitada. Aunque este controlador viene con .NET Framework, Microsoft no recomienda su uso porque es obsoleto.
PostgreSQL	**-**	Para este proveedor no hay controladores ADO.NET compatibles.
Sybase	**-**	Para este proveedor no hay controladores ADO.NET compatibles.

7.2.1.5 Configurar una conexión ODBC

ODBC (Open Database Connectivity) es una tecnología de acceso de datos muy popular con la que se puede conectar a bases de datos desde MapForce. Puede utilizarse como método principal de conexión con bases de datos o como alternativa a las conexiones OLE DB o JDBC.

Para conectarse a una base de datos por ODBC primero es necesario contar con un DSN de ODBC en el sistema operativo. El DSN describe de manera uniforme la conexión de BD a todas las aplicaciones cliente compatibles con ODBC que estén en el sistema, incluido MapForce. Los DSN pueden ser de varios tipos:

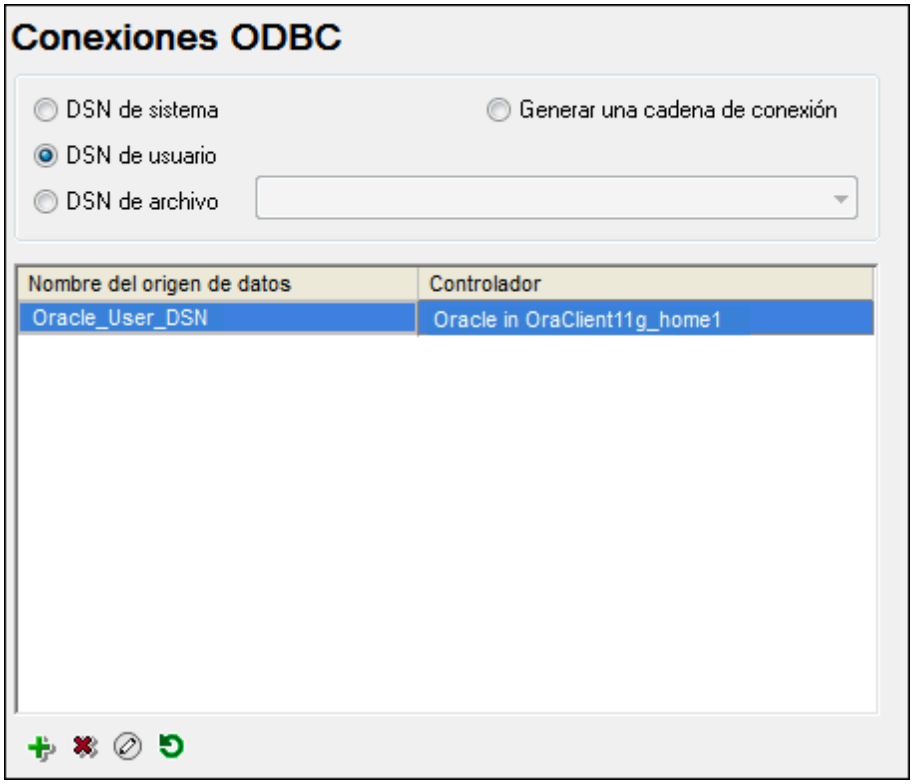
- DSN de sistema
- DSN de usuario
- DSN de archivo

A un origen de datos de sistema pueden acceder todos los usuarios que tengan privilegios en el

© 2018 Altova GmbH

sistema operativo. A un origen de datos de usuario solo puede acceder el usuario que lo creó. Y, por último, si crea un DSN de archivo, el origen de datos se creará como archivo con extensión `.dsn` que podrá compartir con otros usuarios (siempre que tengan instalados los controladores que utiliza el origen de datos).

Los DSN que estén en el equipo aparecen en el cuadro de diálogo de conexión de base de datos al hacer clic en **Conexiones ODBC**.



Si el DSN de la base de datos no existe, el asistente para la conexión de base de datos de MapForce le ayudará a crearlo. Si lo prefiere puede crearlo en Windows directamente. En ambos casos, antes de continuar, compruebe que el controlador ODBC para la base de datos está en la lista de controladores disponibles del sistema operativo (consulte el apartado Ver los controladores disponibles).

Para conectarse usando un DSN nuevo:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en el botón **Conexiones ODBC**.
3. Seleccione un tipo de origen de datos (DSN de usuario, de sistema o de archivo).

> **Nota:** para crear un DSN de sistema necesitará derechos de administrador en el sistema y UModel debe ejecutarse como administrador.

4. Haga clic en **Agregar** .
5. Seleccione un controlador y haga clic en **DSN de usuario** o **DSN de sistema**. Si el

controlador correspondiente a la BD no aparece en la lista, descárguelo e instálelo.
6. En el cuadro de diálogo que aparece debe rellenar la información de conexión para terminar de configurar la conexión.

Para que la conexión se establezca correctamente deberá dar el nombre de host (o dirección IP) del servidor de la BD, así como el nombre de usuario y la contraseña. Quizás sean necesarios otros parámetros de conexión. Para más información consulte la documentación que ofrece el proveedor de la base de datos. Una vez creado, el DSN estará en la lista de nombres de orígenes de datos. Así podrá volver a usar los datos de la conexión cada vez que desee conectarse a la BD.

Para conectarse por medio de un DSN:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en el botón **Conexiones ODBC**.
3. Elija el tipo de origen de datos (DSN de usuario, de sistema o de archivo).
4. Haga clic en el DSN y después en **Conectarse**.

Para generar una cadena de conexión basada en un archivo .dsn:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en el botón **Conexiones ODBC**.
3. Seleccione *Generar una cadena de conexión* y después haga clic en **Generar**.
4. Si quiere generar la conexión a partir de un DSN de archivo, haga clic en la pestaña *Origen de datos de archivo.* Si no es así, haga clic en la pestaña *Origen de datos de equipo* (los DSN de sistema y de usuario se denominan orígenes de datos de equipo)
5. Seleccione el archivo `.dsn` correspondiente y haga clic en **Aceptar**.

Para conectarse a través de una cadena de conexión preparada previamente:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en el botón **Conexiones ODBC**.
3. Seleccione *Generar una cadena de conexión*.
4. Pegue la cadena de conexión en el cuadro de texto y haga clic en **Conectarse**.

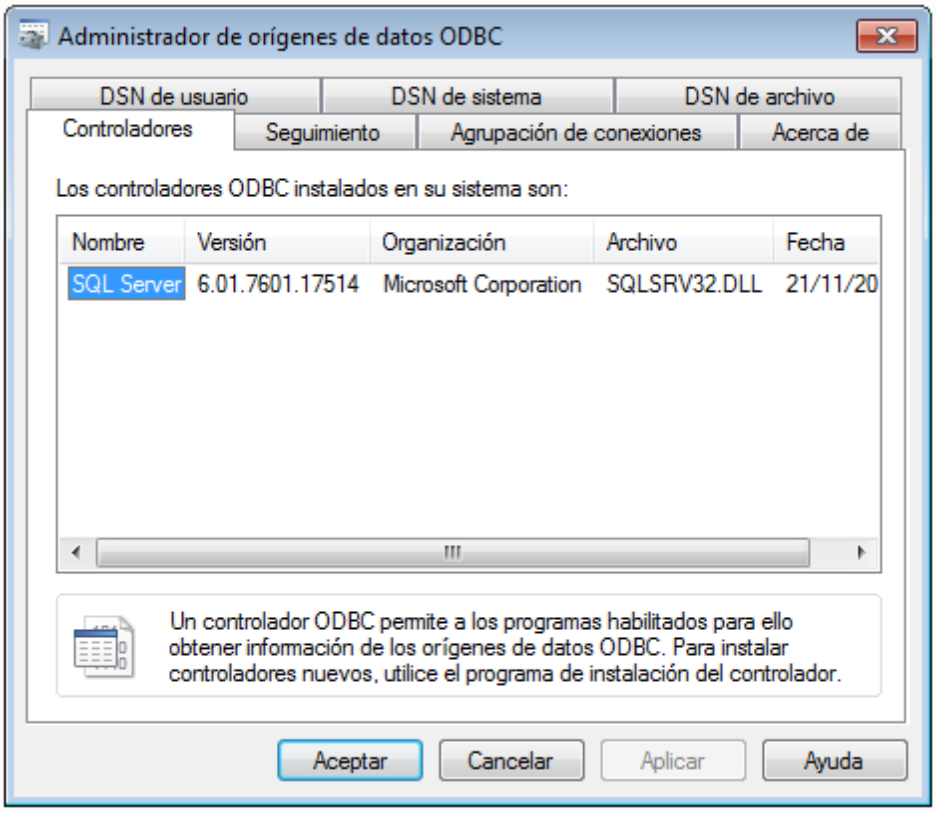
7.2.1.5.1 Ver los controladores ODBC disponibles

En el administrador de orígenes de datos ODBC puede ver qué controladores ODBC están disponibles en su sistema operativo. El administrador (**Odbcad32.exe**) se puede abrir desde el panel de control de Windows (desde **Herramientas administrativas**). En sistemas operativos de 64 bits encontrará dos versiones de este ejecutable:

- La versión de 32 bits está en el directorio **C:\Windows\SysWoW64** (siempre y cuando **C:** sea su unidad de sistema).
- La versión de 64 bits está en el directorio **C:\Windows\System32**.

Los controladores de BD de 43 bits aparecerán en la versión de 32 bits del administrador de orígenes de datos ODBC, mientras que los controladores de 64 bits aparecerán en la versión de 64 bits. Esto debe tenerse en cuenta a la hora de consultar los controladores en el administrador.

© 2018 Altova GmbH



Si el controlador de la base de datos de destino no está en la lista del administrador o si desea agregar algún controlador, deberá descargar el controlador (Resumen de controladores de base de datos). Cuando el controlador esté disponible en el sistema podrá crear conexiones ODBC con él.

7.2.1.6 Configurar una conexión JDBC

JDBC (Java Database Connectivity) es una interfaz de acceso a base de datos que forma parte de la plataforma de software Java de Oracle. Las conexiones JDBC suelen consumir más recursos que las conexiones ODBC pero pueden ofrecer más características.

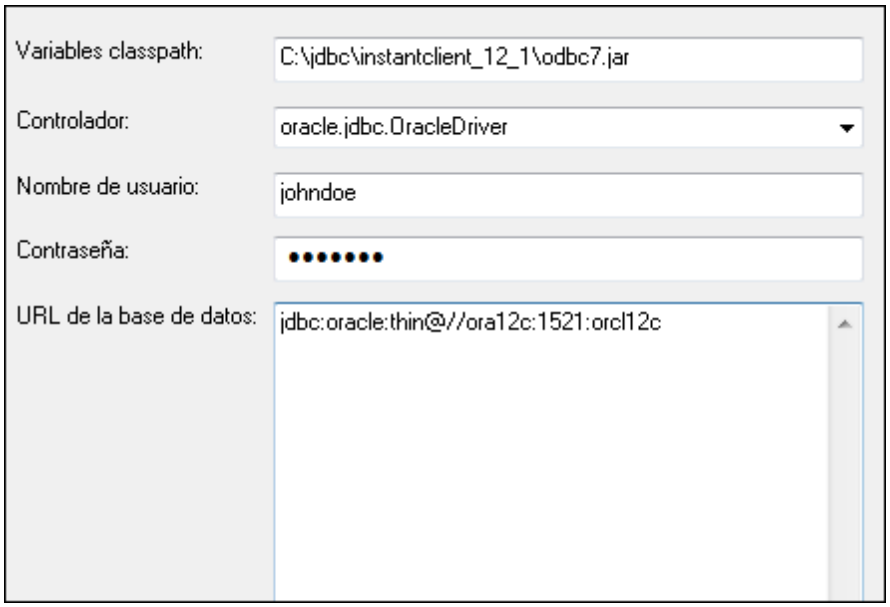
Requisitos

- Tener instalado Java Runtime Environment (JRE) o Java Development Kit (JDK). Consulte el sitio web oficial de Java para descargar estos productos.
- Tener instalados los controladores JDBC del proveedor de la BD. Si quiere conectarse a una base de datos Oracle, tenga en cuenta que algunos controladores Oracle solo funcionan con determinadas versiones de JRE. Consulte la documentación de su producto de Oracle para obtener más información.
- La variable de entorno `PATH` del sistema operativo debe incluir la ruta de acceso del directorio `bin` del directorio de instalación de JRE o JDC. P. ej.: `C:\Program Files (x86)\Java\jre1.8.0_51\bin`.
- La variable del entorno `CLASSPATH` debe incluir la ruta de acceso del controlador JDBC (que puede ser un archivo .jar o varios). Algunos clientes de base de datos configuran

esta variable automáticamente durante la instalación. Para más información consulte el apartado Configurar la variable CLASSPATH.

Para configurar una conexión JDBC:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en el botón **Conexiones JDBC**.
3. Si quiere, introduzca una lista de rutas de archivo .jar separadas por caracteres de punto y coma en el cuadro de texto *Variables classpath*. Las bibliotecas .jar que se introduzcan aquí se cargarán en el entorno junto a las que ya estén definidas en la variable de entorno CLASSPATH. Cuando termine de editar el cuadro de texto Variables classpath, todos los controladores JDBC encontrados en las bibliotecas .jar de origen se cargan automáticamente a la lista del cuadro combinado *Controlador*.



4. En el cuadro combinado *Controlador* seleccione un controlador JDBC de la lista o introduzca un nombre de clase Java. Observe que esta lista contiene todos los controladores JDBC configurados a través de la variable de entorno CLASSPATH (*véase Configurar la variable CLASSPATH*), así como los controladores encontrados en el campo *Variables classpath*.

> Las rutas de acceso del controlador JDBC definidas sen la variable CLASSPATH, así como las rutas de acceso de los archivos .jar introducidos directamente en el cuadro de diálogo de conexión a la base de datos se envían a Java Virtual Machine (JVM). JVM decide qué controladores se utilizan para establecer la conexión. Se recomienda realizar un seguimiento de las clases Java que se cargan en JVM para evitar conflictos y resultados inesperados a la hora de conectarse a la base de datos.

5. Introduzca el nombre de usuario y la contraseña de la BD.
6. En el cuadro de texto *URL de la base de datos:* introduzca la URL de la conexión JDBC en el formato propio del tipo de base de datos utilizado. En la siguiente tabla puede ver la sintaxis de las URL de conexión JDBC para los tipos de base de datos más frecuentes.

© 2018 Altova GmbH

Base de datos	URL de conexión JDBC
Firebird	`jdbc:firebirdsql://<host>[:<puerto>]/<ruta de acceso o alias de la BD>`
IBM DB2	`jdbc:db2://nombreHost:puerto/nombreBaseDatos`
IBM Informix	`jdbc:informix-sqli://nombreHost:puerto/nombreBaseDatos:INFORMIXSERVER=miservidor`
MariaDB	`jdbc:mariadb://nombreHost:puerto/nombreBaseDatos`
Microsoft SQL Server	`jdbc:sqlserver://nombreHost:puerto;nombreBaseDatos=name`
MySQL	`jdbc:mysql://nombreHost:puerto/nombreBaseDatos`
Oracle	`jdbc:oracle:thin:@//nombreHost:puerto:servicio`
Oracle XML DB	`jdbc:oracle:oci:@//nombreHost:puerto:servicio`
PostgreSQL	`jdbc:postgresql://nombreHost:puerto/nombreBaseDatos`
Progress OpenEdge	`jdbc:datadirect:openedge://host:puerto;databaseName=nombre_bd`
Sybase	`jdbc:sybase:Tds:nombreHost:puerto/nombreBaseDatos`
Teradata	`jdbc:teradata://nombreServidorBaseDatos`

Nota: en algunos casos puede modificarse el formato sintáctico (p. ej. la URL de la base de datos puede excluir el puerto o puede incluir el nombre de usuario y la contraseña). Consulte la documentación del proveedor de BD para obtener más información.

7. Haga clic en **Conectarse**.

7.2.1.6.1 Configurar la variable CLASSPATH

La variable de entorno CLASSPATH es utilizada por Java Runtime Environment (JRE) para encontrar las clases Java y otros archivos de recursos del sistema operativo. Cuando se conecte a una BD con JDBC, esta variable debe incluir la ruta de acceso del controlador JDBC del sistema y, en algunos casos, la ruta de acceso de otros archivos de biblioteca relacionados con el tipo de BD que esté utilizando.

En la tabla que aparece a continuación puede ver ejemplos de variables CLASSPATH. Lo más importante que debe tener en cuenta es que quizás deba adaptar esta información dependiendo de la ubicación y del nombre del controlador JDBC y de la versión JRE que esté en su sistema operativo. Para evitar problemas de conexión recomendamos leer detenidamente las instrucciones de instalación del controlador JDBC que esté instalado en el sistema.

Base de datos	Ejemplo de variable CLASSPATH
Firebird	`C:\Archivos de programa\Firebird\Jaybird-2.2.8-JDK_1.8\jaybird-full-2.2.8.jar`
IBM DB2	`C:\Archivos de programa (x86)\IBM\SQLLIB\java\db2jcc.jar;C:\Archivos de programa (x86)\IBM\SQLLIB\java\db2jcc_license_cu.jar;`
IBM Informix	`C:\Informix_JDBC_Driver\lib\ifxjdbc.jar;`
MariaDB	**<directorio de instalación>**`\mariadb-java-client-2.2.0.jar`
Microsoft SQL Server	`C:\Archivos de programa\Microsoft JDBC Driver 4.0 for SQL Server\sqljdbc_4.0\enu\sqljdbc.jar`
MySQL	**<directorio de instalación>**`\mysql-connector-java-`*version*`-bin.jar;`
Oracle	**ORACLE_HOME**`\jdbc\lib\ojdbc6.jar;`
Oracle (with XML DB)	**ORACLE_HOME**`\jdbc\lib\ojdbc6.jar;`**ORACLE_HOME**`\LIB\xmlparserv2.jar;`**ORACLE_HOME**`\RDBMS\jlib\xdb.jar;`
PostgreSQL	**<directorio de instalación>**`\postgresql.jar`
Progress OpenEdge	`%DLC%\java\openedge.jar;%DLC%\java\pool.jar;` **Nota:** siempre y cuando Progress OpenEdge SDK esté instalado en el equipo, `%DLC%` es el directorio donde está instalado OpenEdge.
Sybase	`C:\sybase\jConnect-7_0\classes\jconn4.jar`
Teradata	**<directorio de instalación>**`\tdgssconfig.jar;`**<directorio de instalación>**`\terajdbc4.jar`

- Los cambios en la configuración de la variable CLASSPATH pueden afectar al comportamiento de las aplicaciones Java del equipo. Consulte la documentación de Java antes de continuar.
- Las variables de entorno pueden ser del sistema o del usuario. Para cambiar las variables de entorno del sistema es necesario tener derechos de administrador.
- Tras modificar la variable de entorno, reinicie los programas que estén en ejecución para que los cambios surtan efecto. También puede cerrar sesión o reiniciar el sistema.

© 2018 Altova GmbH

Para configurar CLASSPATH en Windows 7:

1. Abra el menú **Inicio** y haga clic con el botón derecho en **Equipo**.
2. Haga clic en **Propiedades**.
3. Haga clic en **Configuración avanzada del sistema**.
4. En la pestaña *Avanzadas*, haga clic en **Variables de entorno**.
5. Busque la variable CLASSPATH del sistema o del usuario y haga clic en **Editar**. Si no existe, haga clic en **Nueva** para crearla.
6. Edite el valor de la variable e incluya la ruta de acceso del controlador JDBC. Utilice un punto y coma para separar la ruta de acceso del controlador de las demás partes de la variable.

Para configurar CLASSPATH en Windows 10:

1. Pulse la tecla Windows y teclee "variables de entorno".
2. Haga clic en la sugerencia **Editar las variables de entorno del sistema**.
3. Haga clic en **Variables de entorno**.
4. Busque la variable CLASSPATH del sistema o del usuario y haga clic en **Editar**. Si no existe, haga clic en **Nueva** para crearla.
5. Edite el valor de la variable e incluya la ruta de acceso del controlador JDBC. Utilice un punto y coma para separar la ruta de acceso del controlador de las demás partes de la variable.

7.2.1.7 Configurar una conexión PostgreSQL

Las conexiones a bases de datos PostgreSQL pueden configurarse como conexiones nativas o a través de ODBC, JDBC y otros controladores. La ventaja de configurar una conexión nativa es que no es necesario tener controladores instalados en el sistema.Además, si tiene pensado implementar archivos para ejecutarlos en servidores macOS o Linux, tampoco será necesario instalar controladores en el servidor de destino (*véase* *Conexiones de BD en Linux y macOS*).

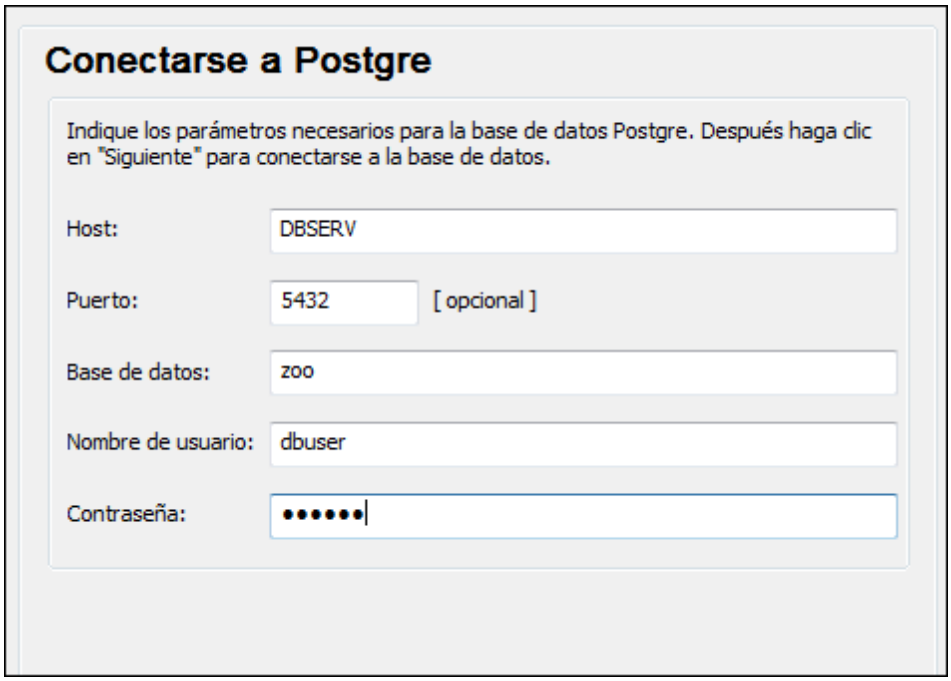
Si prefiere establecer la conexión por medio de un controlador no nativo, consulte estos temas de la documentación:

- Configurar una conexión JDBC
- Conectarse a PostgreSQL (ODBC)

Si lo que quiere es configurar una conexión nativa con PostgreSQL, siga las instrucciones que aparecen más abajo. El único requisito es conocer el nombre de host, el puerto, el nombre de la base de datos, el nombre de usuario y la contraseña.

Para configurar una conexión PostgreSQL nativa:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en el botón **Conexiones PostgreSQL**.
3. Ahora introduzca el host (localhost si PostgreSQL se ejecuta en el mismo equipo), el puerto (es opcional, pero suele ser 5432), el nombre de la BD, el nombre de usuario y la contraseña.



4. Para terminar haga clic en **Conectarse**.

Si el servidor de base de datos PostgreSQL está en otro equipo, debe tener en cuenta lo siguiente:

- El servidor de base de datos PostgreSQL debe estar configurado para aceptar conexiones de clientes. Concretamente, debe configurar el archivo **pg_hba.conf** para que permita conexiones no locales. Además debe configurar el archivo **postgresql.conf** para que escuche determinadas direcciones IP y puertos. Para más información consulte la documentación de PostgreSQL (https://www.postgresql.org/docs/9.5/static/client-authentication-problems.html).
- El equipo servidor debe estar configurado para aceptar conexiones en el puerto correspondiente (suele ser 5432) a través del servidor de seguridad. Por ejemplo, imagine que tiene un servidor de BD que se ejecuta en un equipo Windows. Deberá crear una regla que permita conexiones en el puerto 5432 a través del servidor de seguridad (desde **Panel de control > Firewall de Windows > Configuración avanzada > Reglas de entrada**).

7.2.1.8 Configurar una conexión SQLite

SQLite (https://www.sqlite.org) es un tipo de base de datos basado en archivos y con almacenamiento. Las bases de datos SQLite son compatibles de forma nativa con MapForce, así que no es necesario instalar ningún controlador para poder conectarse a este tipo de BD.

Notas sobre compatibilidad con bases de datos SQLite

- Las bases de datos SQLite son compatibles con el lenguaje de transformación integrado de MapForce (tanto para obtener una vista previa de la asignación de datos como para

 © 2018 Altova GmbH

ejecutar archivos de ejecución de MapForce Server).
- Las bases de datos SQLite no se pueden usar en funciones definidas por el usuario.
- En Linux no hay un tiempo de espera de ejecución de instrucciones para bases de datos SQLite.
- No son compatibles con búsquedas de texto completo en tablas.
- SQLite admite valores de diferentes tipos de datos en cada fila de una tabla. En MapForce todos los valores procesados deben ser compatibles con el tipo de columna declarado. Por tanto, pueden darse errores en tiempo de ejecución si la base de datos SQLite tiene valores de fila que no coinciden con el tipo de columna declarado.
- Si la asignación de datos debe escribir datos en una base de datos SQLite y no tiene preparado el archivo de BD de destino, entonces deberá crearlo por separado. Este archivo se puede crear con una herramienta como Altova DatabaseSpy (http://www.altova.com/es/databasespy.html) o puede descargar la shell de comandos de SQLite del sitio web oficial y crear el archivo desde la línea de comandos (*véase Ejemplo: asignar datos XML a SQLite*). Para más información consulte la documentación de SQLite.

7.2.1.8.1 Conectarse a una base de datos SQLite

Para conectarse a una base de datos SQLite:

1. Inicie el asistente para la conexión de base de datos
2. Seleccione el botón de opción *SQLite* y después haga clic en **Siguiente**.
3. Busque el archivo de BD o introduzca su ruta de acceso (relativa o absoluta). El botón **Conectarse** se habilita una vez introducida la ruta de acceso del archivo.
4. Ahora haga clic en **Conectarse**.

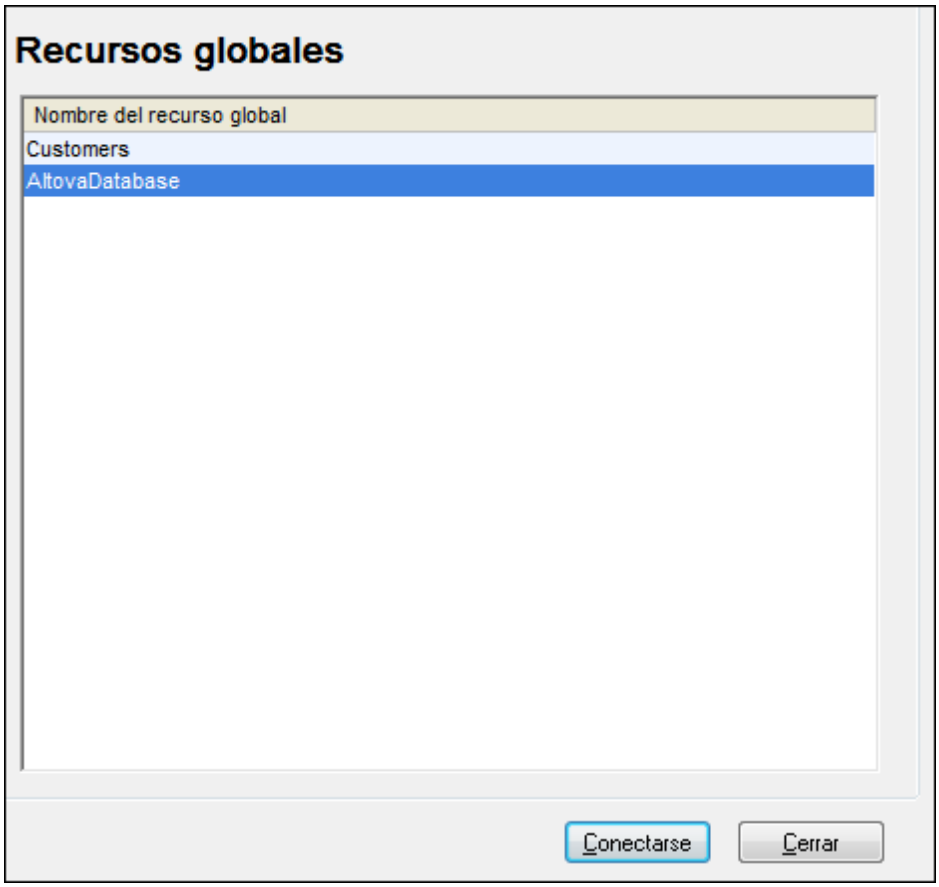
7.2.1.9 Usar una conexión a través de recursos globales

Los recursos globales de Altova es una característica que permite hacer referencia a archivos, carpetas y bases de datos y poder reutilizar este tipo de recursos, configurarlos y ponerlos a disposición de otras aplicaciones de Altova.

Si ya configuró una conexión de base de datos como recurso global, podrá volver a utilizar la conexión siempre que quiera e incluso desde aplicaciones de Altova diferentes.

Para usar una conexión de base de datos disponible como recurso global:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Recursos globales** en el panel izquierdo. En pantalla aparecen ahora las conexiones de base de datos configuradas previamente como recurso global.



3. Seleccione la conexión de base de datos correspondiente y haga clic en **Conectarse**.

Nota: para obtener información sobre un recurso global pase el puntero por encima de la entrada.

7.2.1.10 Ejemplos de conexión a bases de datos

Esta sección incluye procedimientos de muestra para conectarse a una base de datos desde MapForce. Recuerde que es probable que su equipo Windows, el entorno de red y el cliente de BD o software servidor no tengan la misma configuración que la utilizada en estos ejemplos.

Nota: en la mayoría de los casos se puede establecer la conexión por medio de varias tecnologías de acceso (ADO, ADO.NET, ODBC, JDBC) o controladores. El rendimiento de la conexión de BD, así como sus características y limitaciones, dependerá del controlador, software cliente y parámetros de conexión seleccionados.

7.2.1.10.1 Conectarse a Firebird (ODBC)

Este ejemplo explica cómo conectarse a una base de datos Firebird 2.5.4 de un servidor Linux.

 © 2018 Altova GmbH

Requisitos

- El servidor de BD Firebird está configurado para aceptar conexiones TCP/IP desde clientes.
- El controlador ODBC de Firebird está instalado en el sistema. Este ejemplo usa la versión 2.0.3.154 del controlador (https://www.firebirdsql.org/).
- El cliente Firebird está instalado en el sistema. Recuerde que el cliente forma parte del paquete de instalación del servidor Firebird. Puede descargar este paquete del sitio web de Firebird (https://www.firebirdsql.org/) y buscar el instalador para Windows (Windows executable installer for full Superclassic/Classic or Superserver). Para instalar los archivos del cliente solamente elija la opción **Minimum client install - no server, no tools** del asistente para la instalación.

  Nota:

  - La plataforma del controlador ODBC de Firebird y del cliente (de 32 o 64 bits) debe coincidir con la plataforma de MapForce.
  - La versión del cliente Firebird debe coincidir con la versión del servidor Firebird al que desea conectarse.

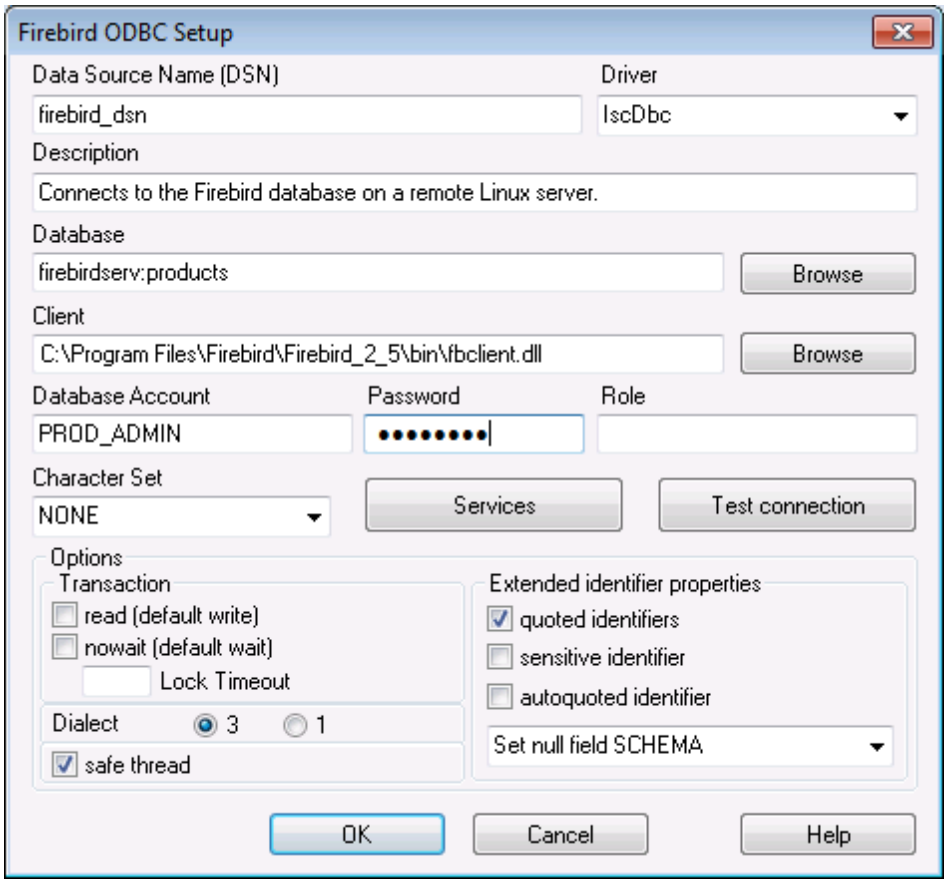
- Disponer de los datos de conexión: nombre de host o dirección IP, ruta de acceso (o alias) de la base de datos en el servidor, nombre de usuario y contraseña.

Para conectarse a Firebird por ODBC:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones ODBC**.
3. Seleccione la opción *DSN de usuario* (o *DSN de sistema* si tiene privilegios de administrador) y después haga clic en **Agregar** .



4. Seleccione el controlador Firebird y después haga clic en *DSN de usuario* (o *DSN de sistema* dependiendo de la opción seleccionada en el paso anterior). Si el controlador Firebird no aparece en la lista, compruebe que está instalado en el sistema.



5. Introduzca los datos de conexión:

Nombre del origen de datos (DSN)	Introduzca un nombre para el origen de datos que desea crear.

Base de datos	Introduzca el nombre de host o dirección IP del servidor, seguida de dos puntos, seguido del alias (o ruta de acceso) de la base de datos. En este ejemplo el nombre de host es `firebirdserv` y el alias de la BD es `products`:
	
	`firebirdserv:products`
	
	Usamos el alias de la BD porque damos por hecho que en el lado servidor el administrador de la BD ha configurado el alias *products* para apuntar al archivo de BD Firebird (.fdb) en el servidor.
	
	En lugar del nombre de host también puede usar la dirección IP del servidor. Y en lugar del alias puede usar una ruta de acceso. Por tanto, también podría usar estas otras dos cadenas de conexión:
	
	`firebirdserver:/var/Firebird/databases/butterflies.fdb`

© 2018 Altova GmbH

	`127.0.0.1:D:\Misc\Lenders.fdb` Si la base de datos está en el equipo Windows local, haga clic en **Examinar** y seleccione el archivo de base de datos directamente.
Cliente	Introduzca la ruta de acceso del archivo `fbclient.dll`. Su ubicación predeterminada es el subdirectorio `bin` del directorio de instalación de Firebird.
Cuenta de base de datos	Introduzca el nombre de usuario que recibió del administrador de la BD (en este ejemplo es `PROD_ADMIN`).
Contraseña	Introduzca la contraseña de la BD que recibió del administrador.

6. Haga clic en **Aceptar**.

7.2.1.10.2 Conectarse a Firebird (JDBC)

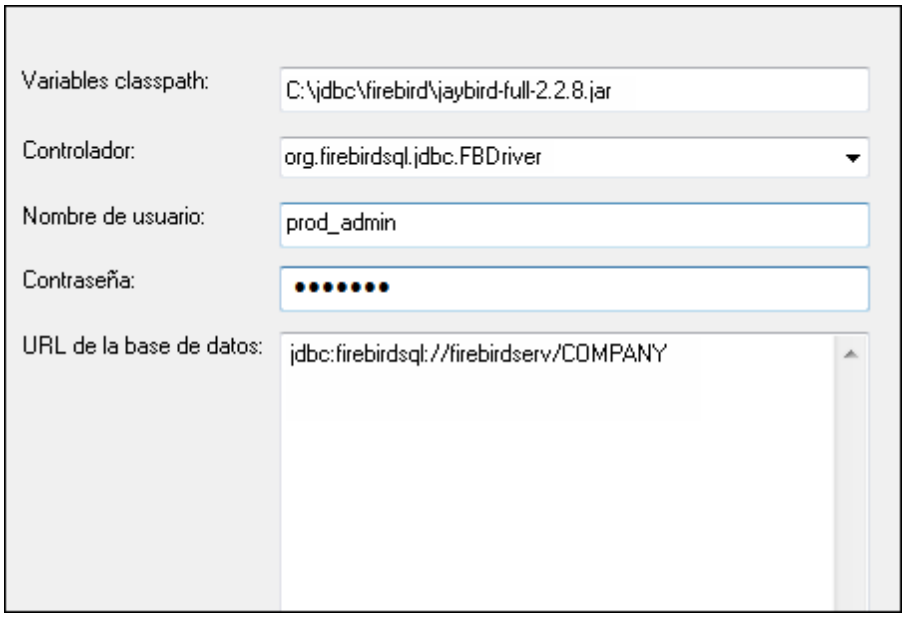
Este ejemplo explica cómo conectarse a una base de datos Firebird por JDBC.

Requisitos

- Java Runtime Environment (JRE) o Java Development Kit (JDK) está instalado en el sistema.
- La variable de entorno `PATH` del sistema incluye la ruta al directorio `bin` del directorio de instalación de JRE o JDK (p. ej. `C:\Archivos de programa (x86)\Java\jre1.8.0_51\bin`).
- El controlador Firebird JDBC debe estar disponible en el sistema operativo (se trata de un archivo .jar que ofrece conectividad con la base de datos). El controlador se puede descargar del sitio web de Firebird (https://www.firebirdsql.org/). En este ejemplo usamos el controlador *Jaybird 2.2.8*.
- Disponer de los datos de conexión: host, ruta de acceso (o alias) de la base de datos, nombre de usuario y contraseña.

Para conectarse a Firebird por JDBC:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones JDBC**.
3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, el archivo .jar está ubicado en esta ruta de acceso: `C:\jdbc\firebird\jaybird-full-2.2.8.jar`. Tenga en cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase Configurar la variable CLASSPATH*).
4. En el campo *Controlador* seleccione **org.firebirdsql.jdbc.FBDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.



5. Introduzca el nombre de usuario y la contraseña de la base de datos.
6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su base de datos).

```
jdbc:firebirdsql://<host>[:<puerto>]/<ruta o alias de la BD>
```

7. Haga clic en **Conectarse**.

7.2.1.10.3 Conectarse a IBM DB2 (ODBC)

Este ejemplo explica cómo conectarse a una base de datos IBM DB2 por ODBC.

Requisitos

- El cliente IBM Data Server está instalado y configurado en el sistema (en este ejemplo usamos IBM Data Server Client 9.7). Después de instalar el cliente IBM Data Server compruebe que los controladores ODBC están disponibles en el equipo (consulte Ver los controladores ODBC disponibles).
- Tiene un alias para la base de datos. El alias se puede crear de varias maneras:
  - Con el asistente de configuración de IBM DB2
  - Con el procesador de línea de comandos de IBM DB2
  - Con el asistente para orígenes de datos ODBC (instrucciones más abajo).
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

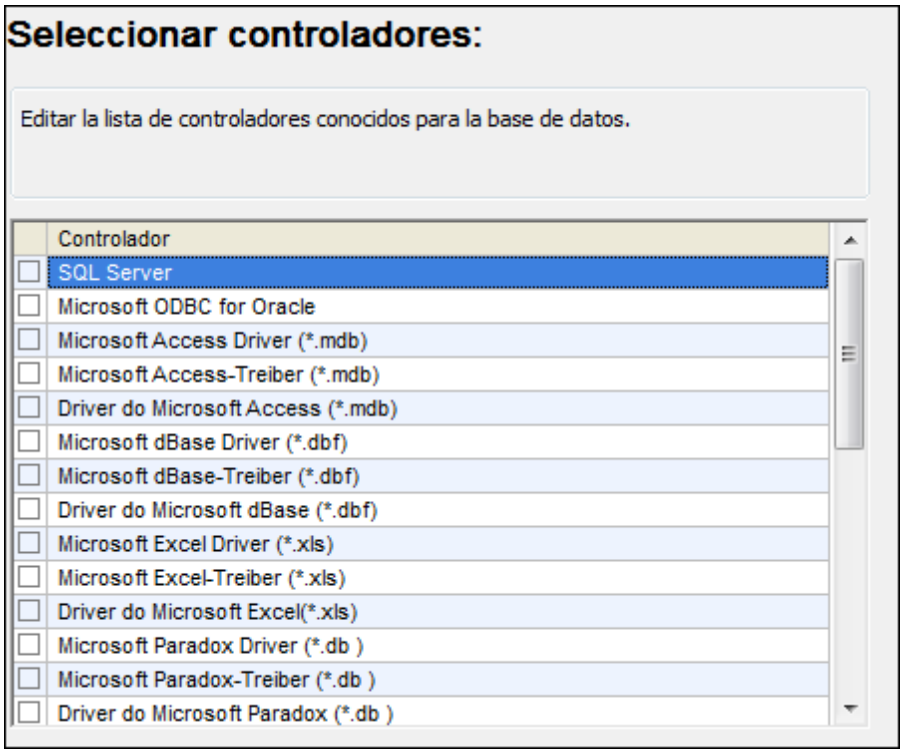
Para conectarse a IBM DB2:

1. Inicie el asistente para la conexión de base de datos y seleccione la opción *IBM DB2 (ODBC/JDBC)*.
2. Haga clic en **Siguiente**.

© 2018 Altova GmbH



3. Seleccione la opción *ODBC* y haga clic en **Siguiente**. Si necesita editar la lista de controladores conocidos para la base de datos, seleccione los controladores que corresponden a IBM DB2 (ver Requisitos) y haga clic en **Siguiente**.

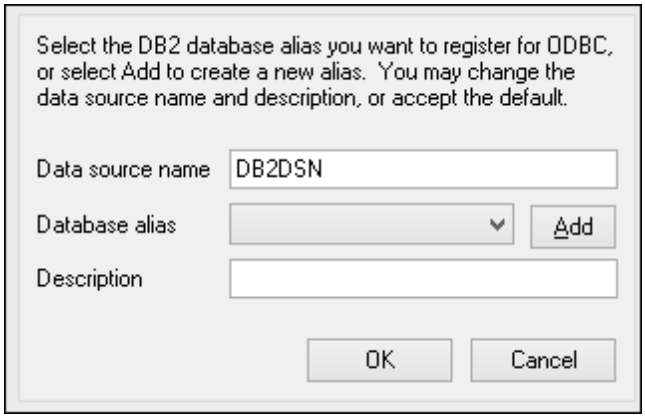


4. Seleccione el controlador de la lista y haga clic en **Conectarse**. (Para editar la lista de controladores disponibles haga clic en **Editar controladores** y active/desactive los controladores que desea agregar o eliminar)

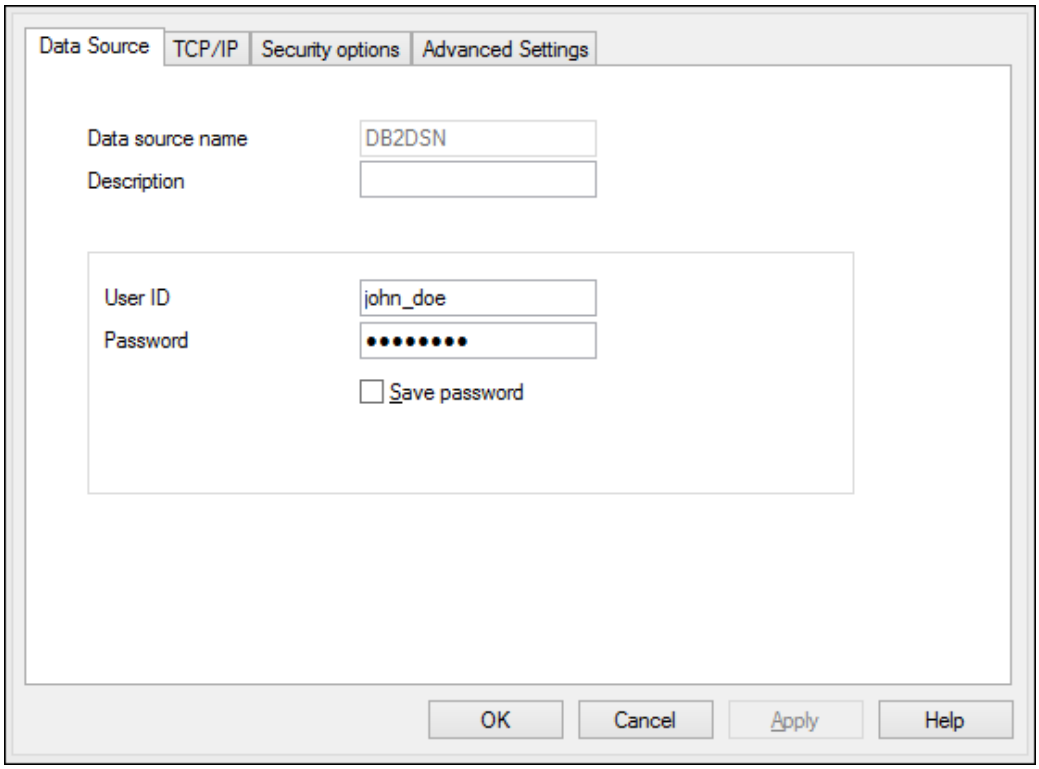
© 2018 Altova GmbH



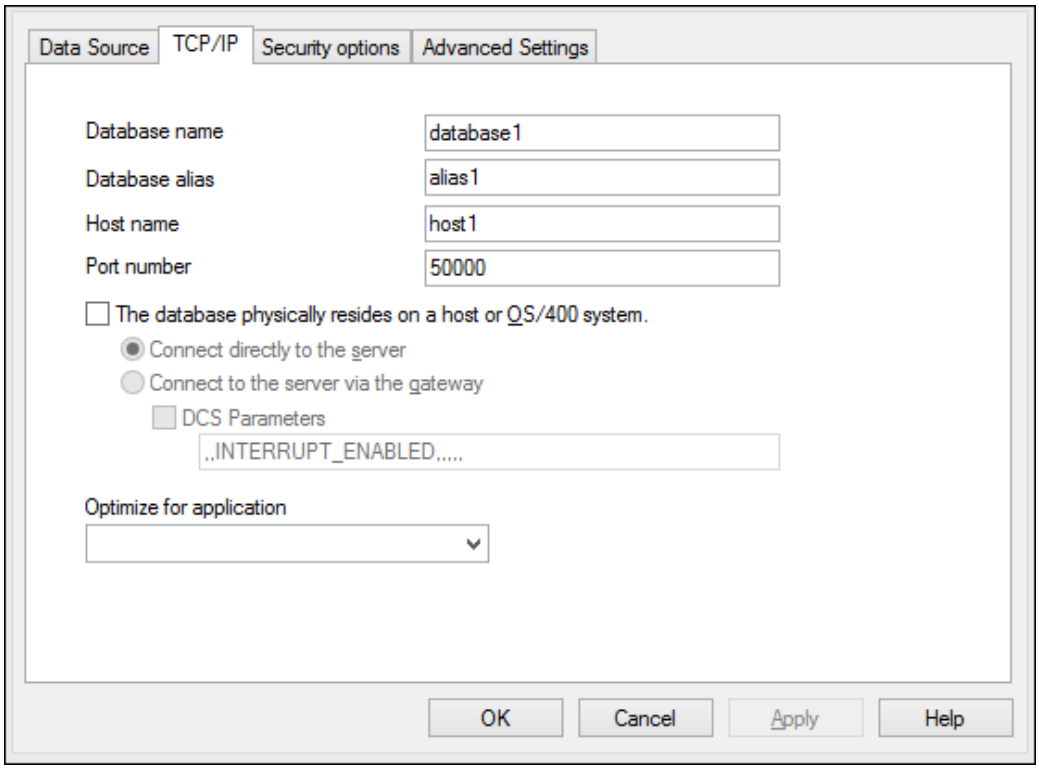
5. Introduzca el DSN (**DB2DSN**) y haga clic en **Agregar**.



6. En la pestaña *Origen de datos* introduzca el nombre de usuario y la contraseña de la base de datos.

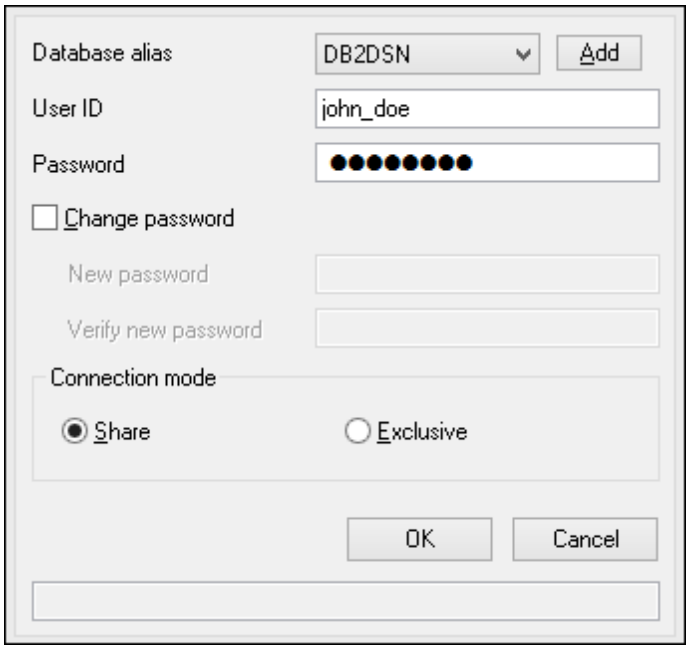


7. En la pestaña **TCP/IP** introduzca el nombre de la base de datos, un nombre para el alias, el nombre de host y el número de puerto. Después haga clic en **Aceptar**.



© 2018 Altova GmbH

8. Vuelva a introducir el nombre de usuario y la contraseña y haga clic en **Aceptar**.

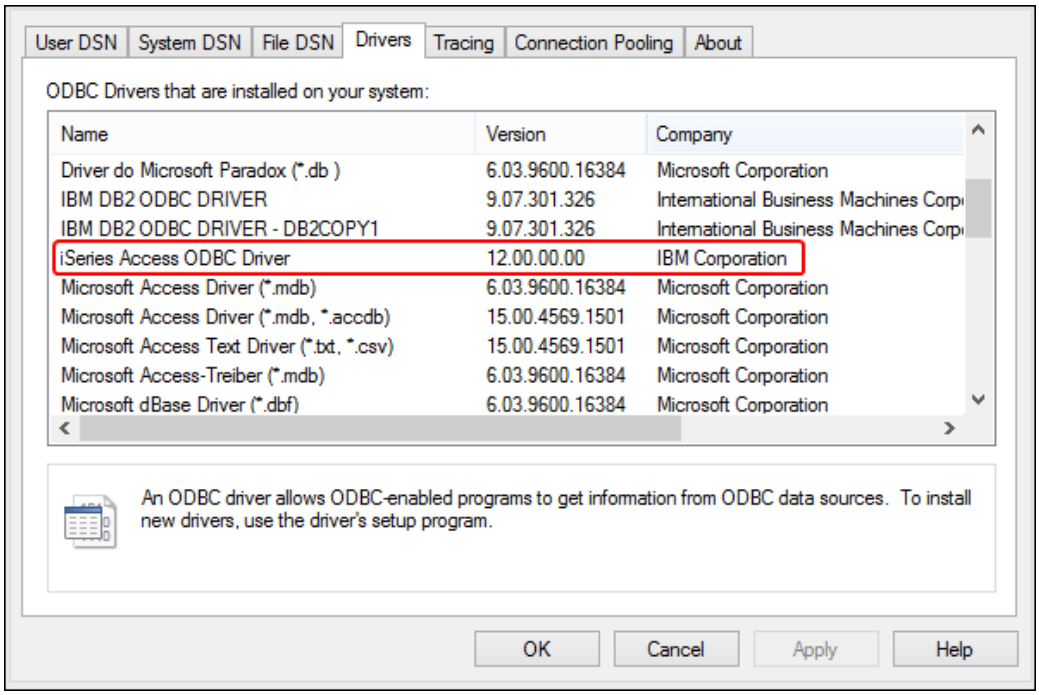


7.2.1.10.4 Conectarse a IBM DB2 for i (ODBC)

Este ejemplo explica cómo conectarse a una base de datos IBM DB2 for i por ODBC.

Requisitos

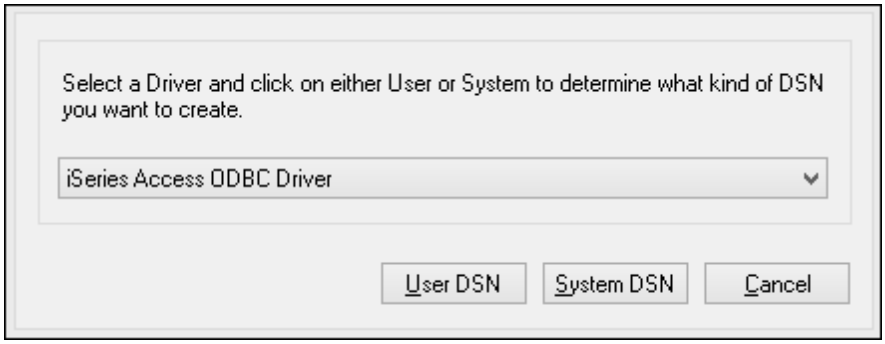
- *IBM System i Access for Windows* está instalado en el sistema (para este ejemplo usamos *IBM System i Access for Windows V6R1M0*). Compruebe que el controlador ODBC está en el equipo (consulte Ver los controladores ODBC disponibles).



- Disponer de los datos de conexión: dirección IP del servidor de BD, nombre de usuario y contraseña.
- Ejecute *System i Navigator* y siga las instrucciones que aparecen en pantalla para crear una conexión nueva. Llegado el momento de especificar un sistema, introduzca la dirección IP del servidor de BD. Tras crear la conexión se recomienda verificarla (haga clic en la conexión y seleccione **Archivo** > **Diagnóstico** > **Verificar conexión**). Si recibe errores de conexión, póngase en contacto con el administrador del servidor de BD.

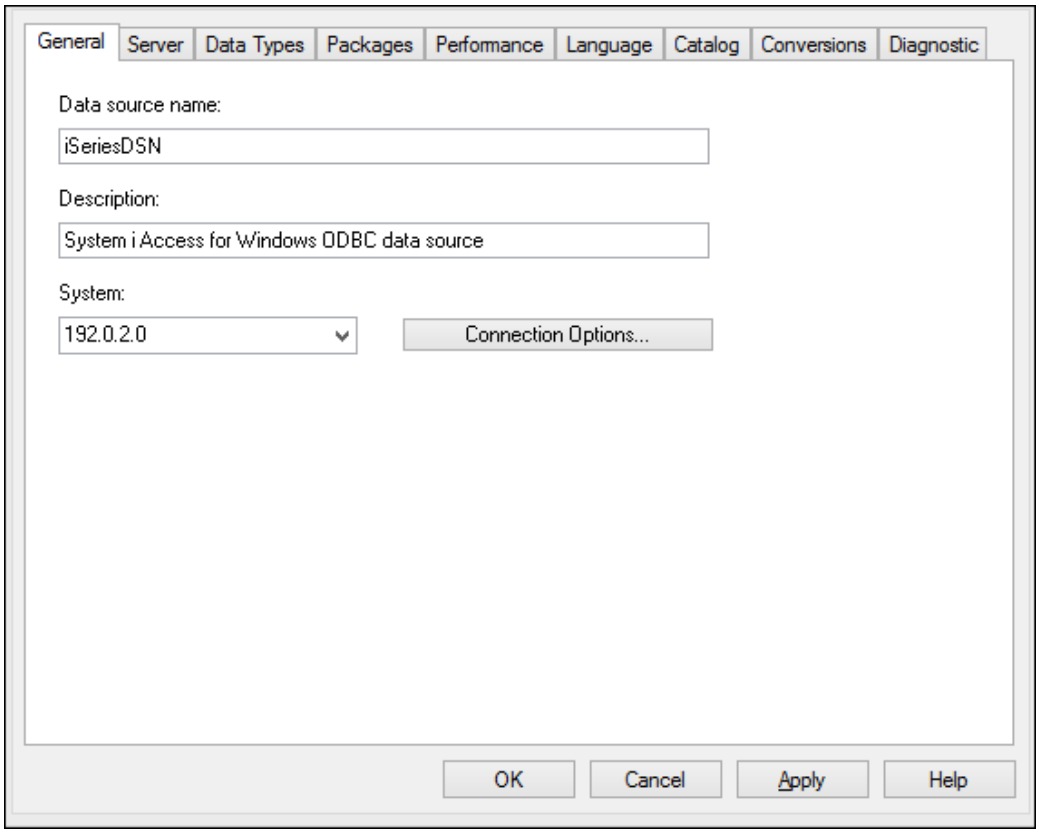
Para conectarse a IBM DB2 for i:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones ODBC**.
3. Seleccione *DSN de usuario* (o *DSN de sistema* o *DSN de archivo*).
4. Haga clic en **Agregar** .
5. Seleccione el controlador **iSeries Access ODBC Driver** de la lista y después haga clic en **DSN de usuario** (o **DSN de sistema**).



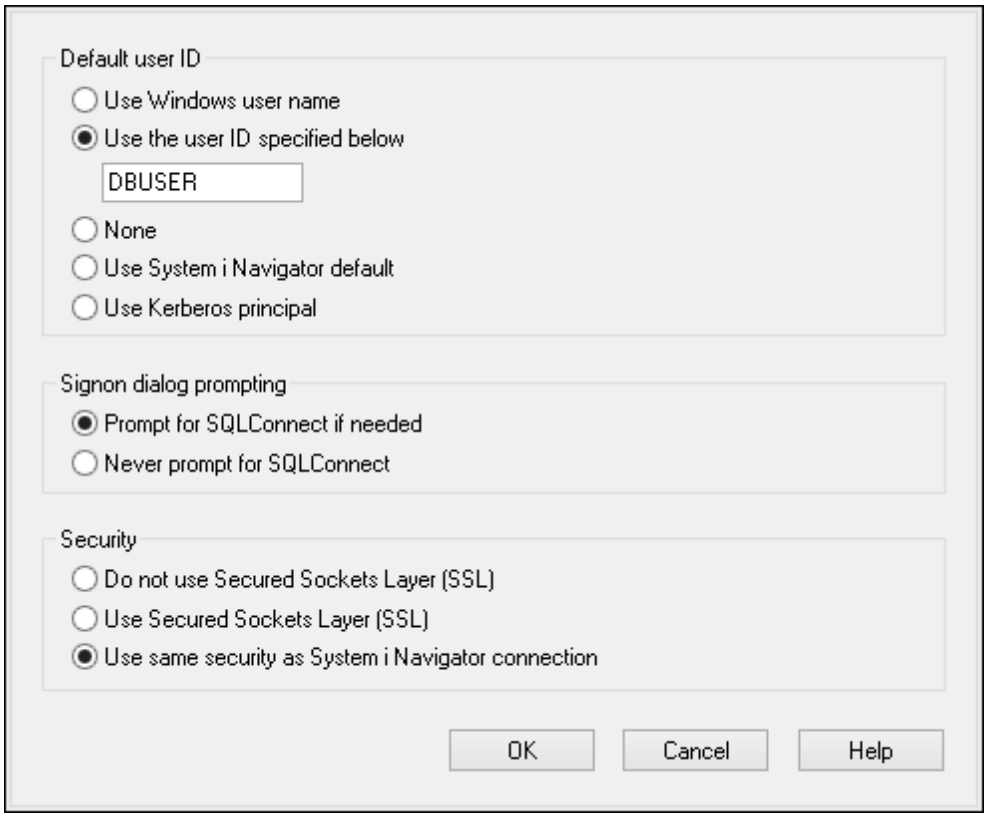
6. Introduzca el DSN y seleccione la conexión del cuadro combinado *Sistema*. En este ejemplo el DSN es **iSeriesDSN** y el sistema es **192.0.2.0**.

 © 2018 Altova GmbH



Nota: cuando se añade un origen de datos ODBC para una base de datos *IBM DB2 for i*, se establece una marca predeterminada que permite tiempos de espera de consulta. Esta opción de configuración se puede deshabilitar para que los archivos de asignación de MapForce carguen correctamente. Para deshabilitar esta opción haga clic en la pestaña de rendimiento Performance, después en el botón de opciones avanzadas **Advanced** y desactive la casilla **Allow query timeout** (permitir tiempo de espera de consulta).

7. Haga clic en el botón **Opciones de conexión** y seleccione *Utilizar el Id. de usuario indicado* y escriba el nombre del usuario de la BD (en este ejemplo es **DBUSER**).



8. Haga clic en **Aceptar**. El origen de datos nuevo aparece ahora en la lista de DSN.
9. Haga clic en **Conectarse**.
10. Introduzca el nombre de usuario y la contraseña de la BD cuando sea necesario y después haga clic en **Aceptar**.

7.2.1.10.5 Conectarse a IBM Informix (JDBC)

Este ejemplo explica cómo conectarse a una base de datos IBM por JDBC.

Requisitos

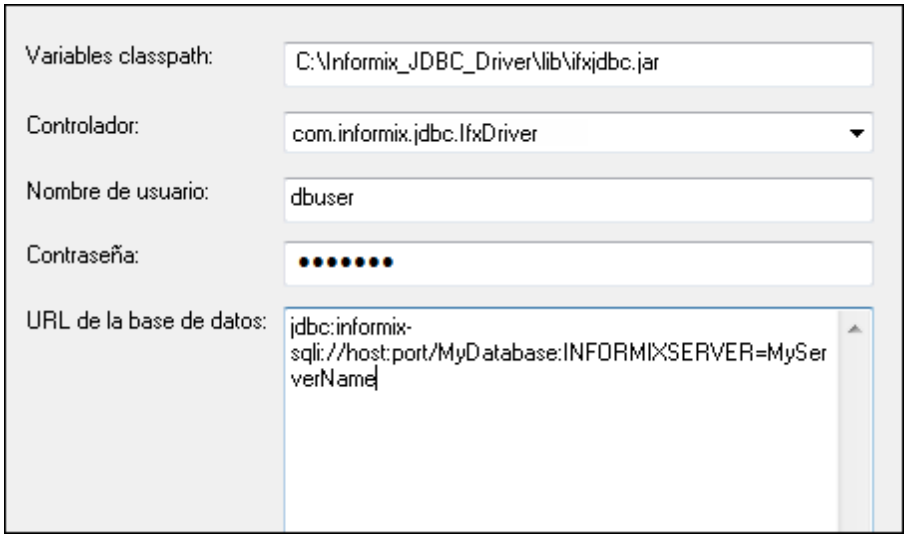
- Java Runtime Environment (JRE) está instalado en el sistema.
- El controlador JDBC (se trata de un archivo .jar o varios archivos .jar que ofrecen conectividad con la base de datos) debe estar disponible en el sistema operativo. En este ejemplo se utiliza el controlador JDBC IBM Informix versión 3.70. Consulte la documentación del controlador para ver las instrucciones de instalación o la guía *IBM Informix JDBC Driver Programmer's Guide*.
- Disponer de los datos de conexión: host, nombre del servidor Informix, base de datos, puerto, nombre de usuario y contraseña.

Para conectarse a IBM Informix por JDBC:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones JDBC**.

© 2018 Altova GmbH

3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, el archivo .jar está ubicado en esta ruta de acceso: `C:\Informix_JDBC_Driver\lib\ifxjdbc.jar`. Tenga en cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase* *Configurar la variable CLASSPATH*).
4. En el campo *Controlador* seleccione **com.informix.jdbc.IfxDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.



5. Introduzca el nombre de usuario y la contraseña de la base de datos.
6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su base de datos).

```
jdbc:informix-sqli://nombreHost:puerto/
nombreBD:INFORMIXSERVER=myserver;
```

7. Haga clic en **Conectarse**.

7.2.1.10.6 Conectarse a MariaDB (ODBC)

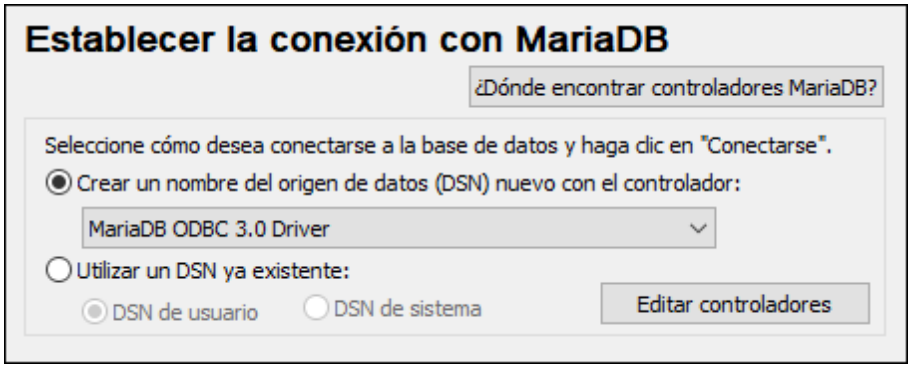
Este ejemplo explica cómo conectarse a un servidor de base de datos MariaDB por ODBC.

Requisitos

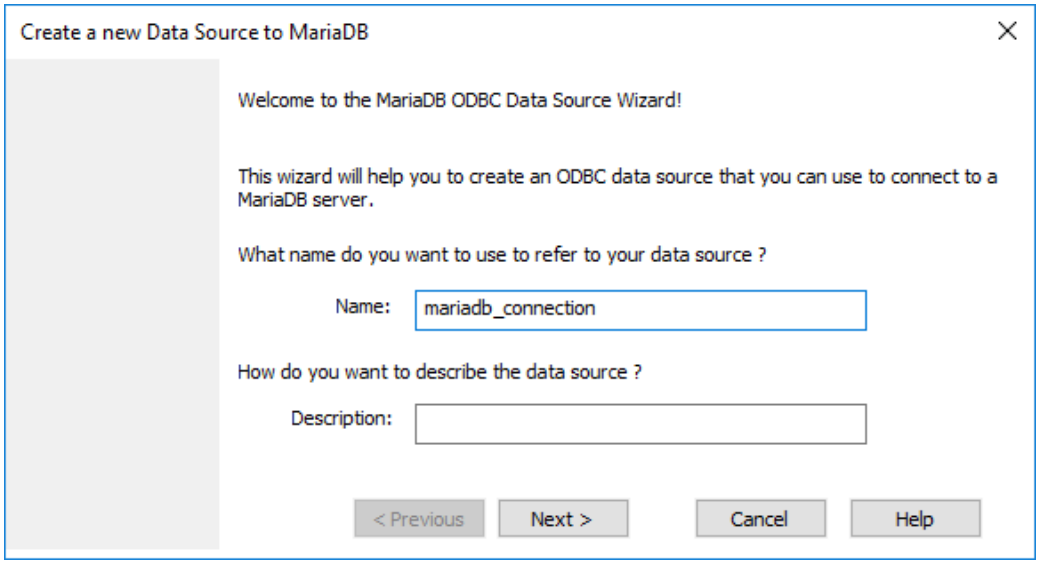
- Debe tener instalado el conector de MariaDB para ODBC (https://downloads.mariadb.org/connector-odbc/).
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

Para conectarse a MariaDB por ODBC:

1. Inicie el asistente para la conexión de base de datos.
2. Seleccione *MariaDB (ODBC)* y después haga clic en **Siguiente**.

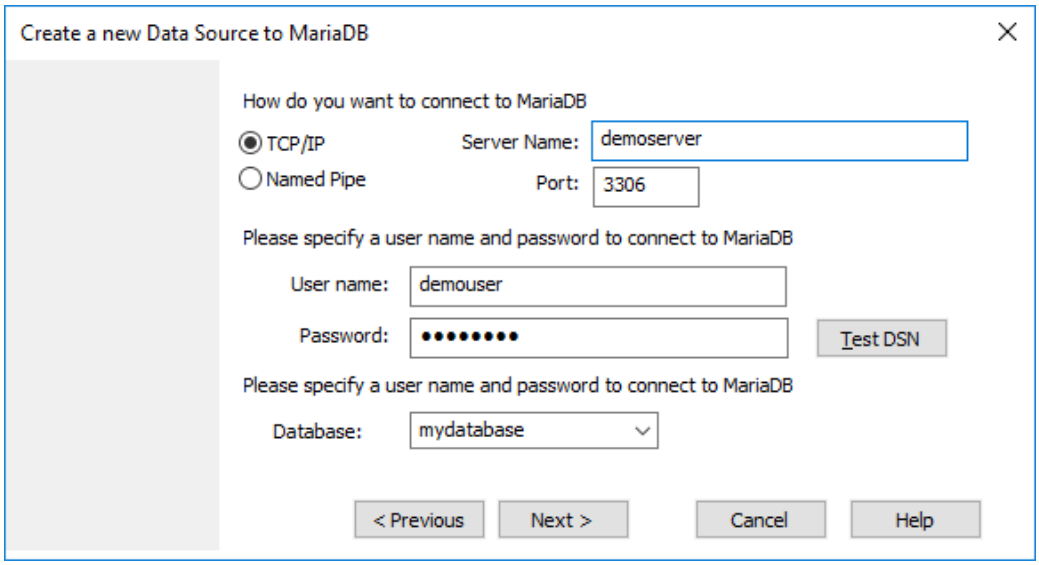


3. Seleccione *Crear un nombre del origen de datos (DSN) nuevo con el controlador* y elija el controlador **MariaDB ODBC 3.0 Driver**. Si este controlador no aparece en la lista, haga clic en **Editar controladores** y seleccione cualquier controlador MariaDB (la lista contiene todos los controladores ODBC que están instalados en el sistema operativo).
4. Haga clic en **Conectarse**.

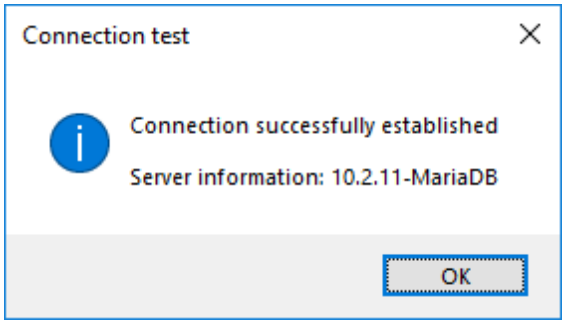


5. Introduzca el nombre y, si quiere, una descripción que le ayude a identificar este origen de datos ODBC más adelante.

© 2018 Altova GmbH



6. Rellene las credenciales de la conexión de base de datos (servidor TCP/IP, usuario, contraseña), seleccione una base de datos y después haga clic en **Probar DSN**. Cuando se establezca la conexión aparecerá este mensaje:



7. Haga clic en **Siguiente** y siga los pasos del asistente hasta el final. Dependiendo del caso, puede que sean necesarios más parámetros (p. ej. certificados SSL si se conecta a MariaDB a través de una conexión segura).

Nota: si el servidor de base de datos es remoto, deberá estar configurado por el administrador para que acepte conexiones remotas desde la dirección IP de su equipo.

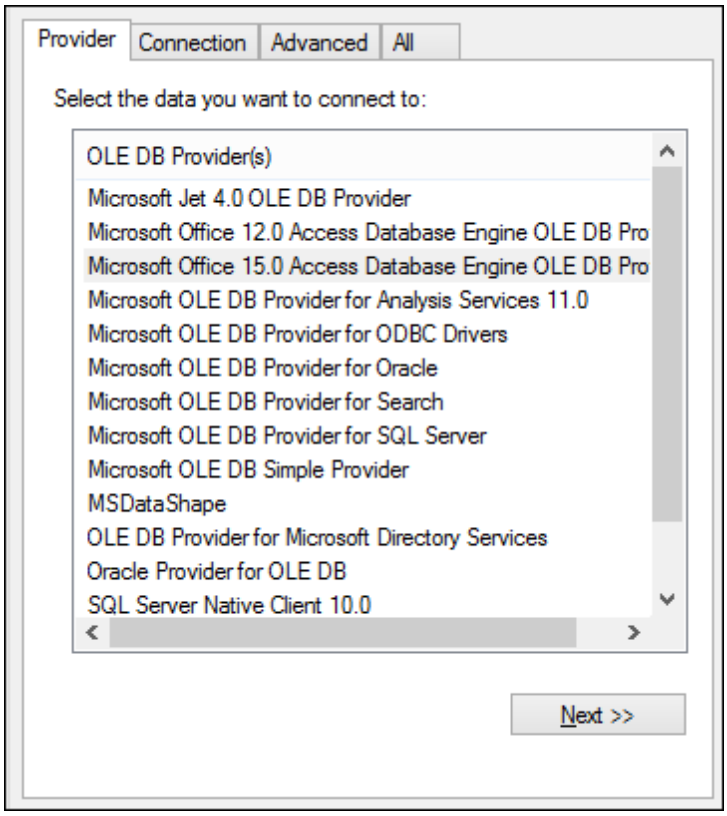
7.2.1.10.7 Conectarse a Microsoft Access (ADO)

Lo más sencillo para conectarse a una base de datos Microsoft Access es seguir las instrucciones del asistente para la conexión de base de datos y buscar el archivo de base de datos. También puede configurar explícitamente una conexión ADO, como muestra este ejemplo. Esta segunda opción se recomienda si la base de datos está protegida con contraseña.

También puede conectarse a Microsoft Access por ODBC, pero esto implica algunas restricciones por lo que recomendamos evitar este tipo de conexión.

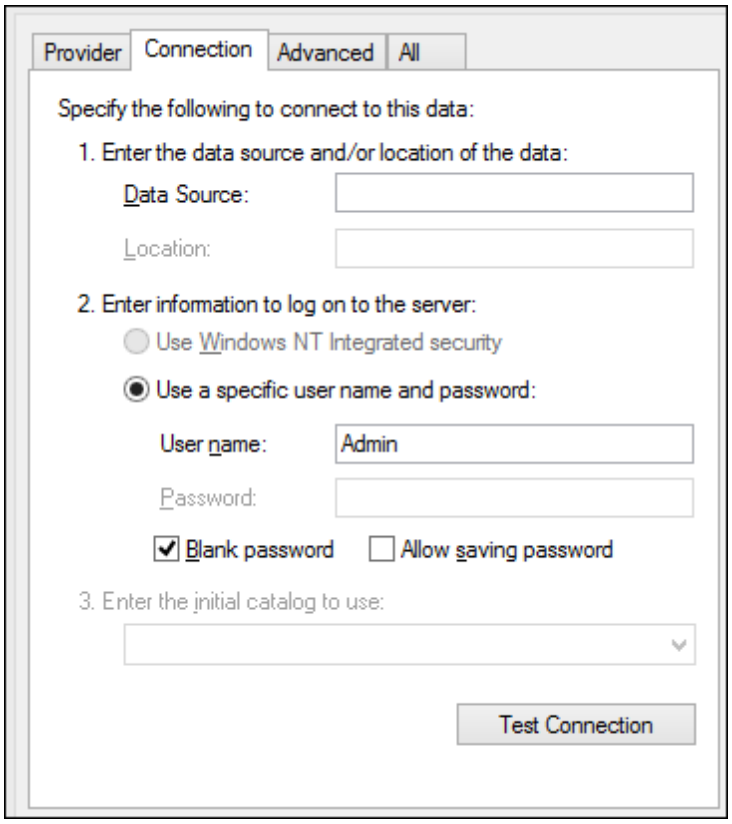
Para conectarse a una base de datos Microsoft Access protegida con contraseña:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones ADO**.
3. Ahora haga clic en **Generar**.

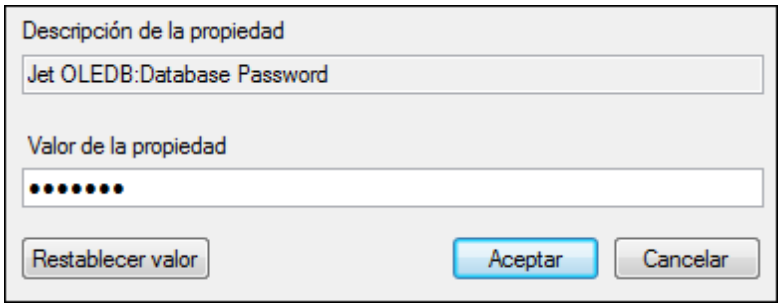


4. Seleccione **Microsoft Office 15.0 Access Database Engine OLE DB Provider** y haga clic en **Siguiente**.

© 2018 Altova GmbH



5. En el cuadro de texto Origen de datos introduzca la ruta de acceso del archivo de Microsoft Access. Como el archivo está en la red local, comparta `U:\Departments\Finance\Reports\Revenue.accdb`. Lo pasamos al formato UNC y tenemos `\\server1\dfs\Departments\Finance\Reports\Revenue.accdb`, siendo `server1` el nombre del servidor y `dfs` el nombre del recurso compartido de red.
6. En la pestaña *Todos* haga doble clic en la propiedad **Jet OLEDB:Database Password** e introduzca la contraseña de la base de datos.



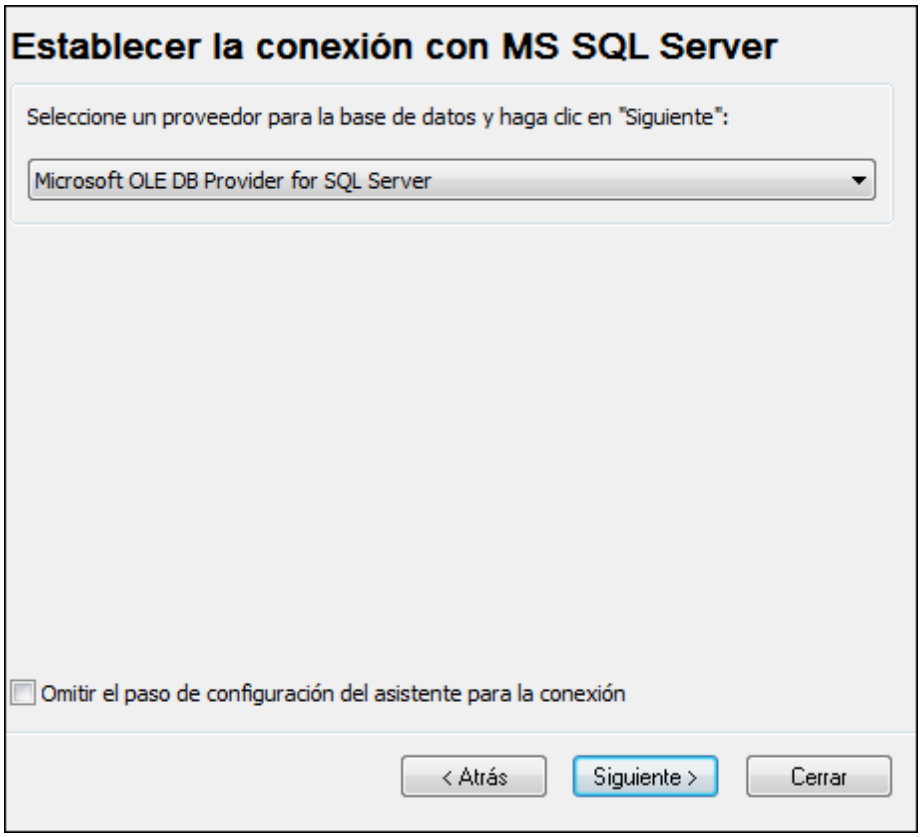
Nota: si no es capaz de establecer la conexión, busque el archivo de información del grupo de trabajo (`System.MDW`) de su perfil de usuario y establezca el valor de la propiedad **Jet OLEDB: System database** en la ruta de acceso del archivo `System.MDW` (consulte https://support.microsoft.com/es-es/kb/305542 para obtener más información).

7.2.1.10.8 Conectarse a Microsoft SQL Server (ADO)

Este ejemplo explica cómo conectarse a una base de datos SQL Server por ADO.

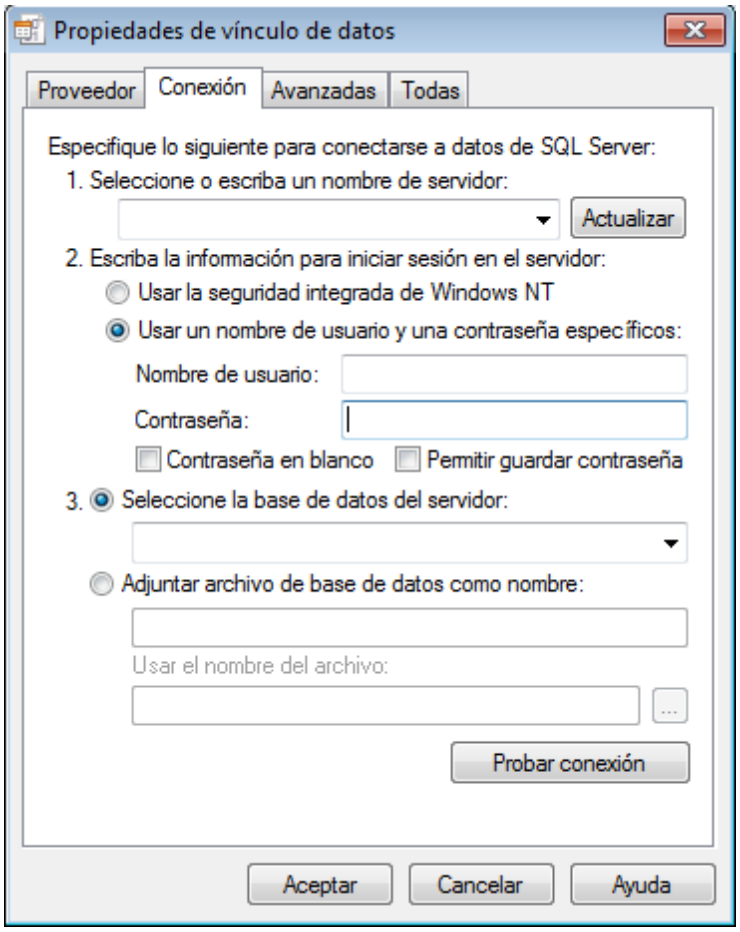
Para conectarse a SQL Server con Microsoft OLE DB Provider:

1. Inicie el asistente para la conexión de base de datos.
2. Seleccione la opción *Microsoft SQL Server (ADO)* y haga clic en **Siguiente**. Ahora aparece la lista de controladores ADO disponibles.

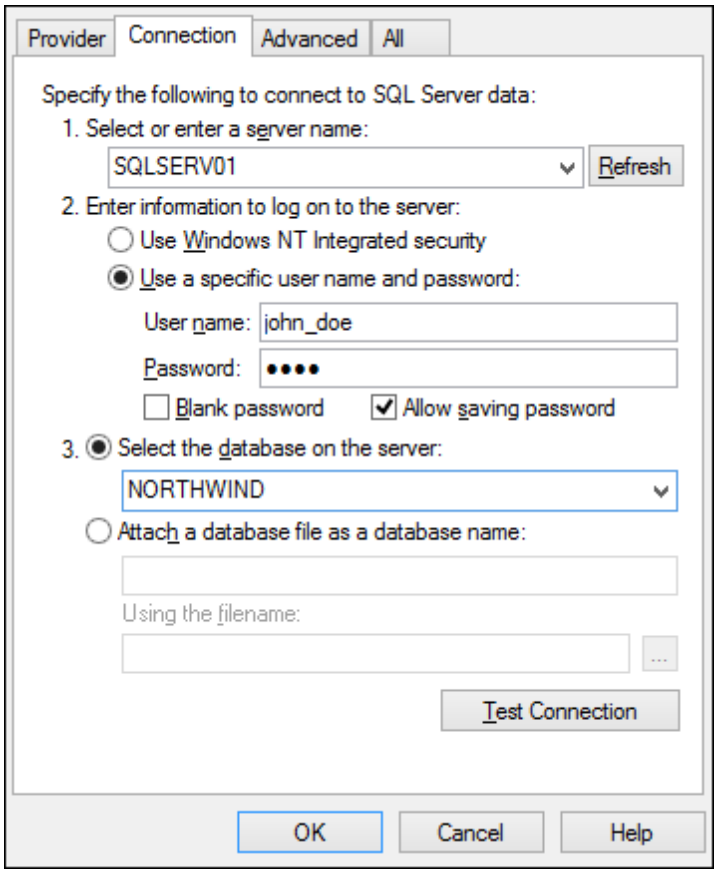


3. Seleccione el controlador **Microsoft OLE DB Provider for SQL Server** y haga clic en **Siguiente**.

© 2018 Altova GmbH



4. Seleccione o introduzca el nombre del servidor de BD (en este ejemplo es SQLSERV01). Expanda la lista desplegable para ver la lista de servidores disponibles en la red.
5. Si el servidor de BD se configuró para admitir conexiones desde usuarios autenticados en el dominio de Windows, seleccione Utilizar seguridad integrada de Windows NT. De lo contrario, seleccione Utilizar un nombre de usuario y una contraseña específicos e introduzca los datos donde corresponda.
6. Seleccione la base de datos a la que desea conectarse (en este ejemplo es NORTHWIND).
7. Para probar la conexión haga clic en Probar conexión. Este paso se recomienda en este momento.
8. Ahora tiene dos opciones:
   a. Marcar la casilla *Permitir guardar la contraseña*.
   b. En la pestaña Todos puede cambiar el valor de la propiedad *Almacenar información de seguridad* por `True`.



9. Haga clic en **Aceptar**.

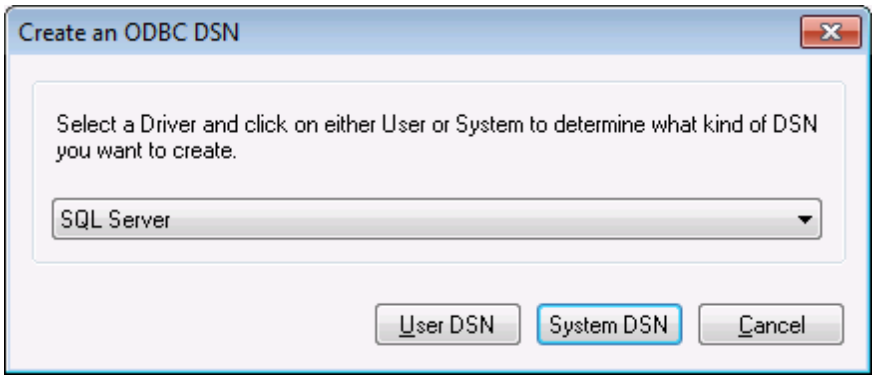
7.2.1.10.9 Conectarse a Microsoft SQL Server (ODBC)

Este ejemplo explica cómo conectarse a una base de datos SQL Server por ODBC.

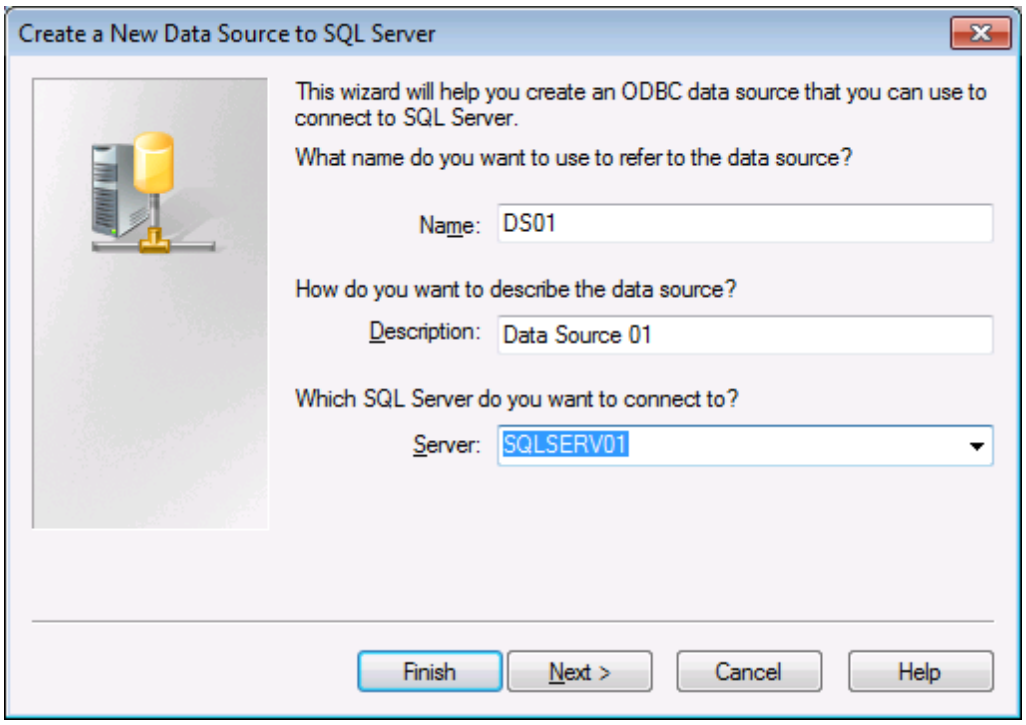
Para conectarse a SQL Server con ODBC:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones ODBC**.
3. Seleccione *DSN de usuario* (o *DSN de sistema* si tiene privilegios de administrador) y después haga clic en **Agregar** .

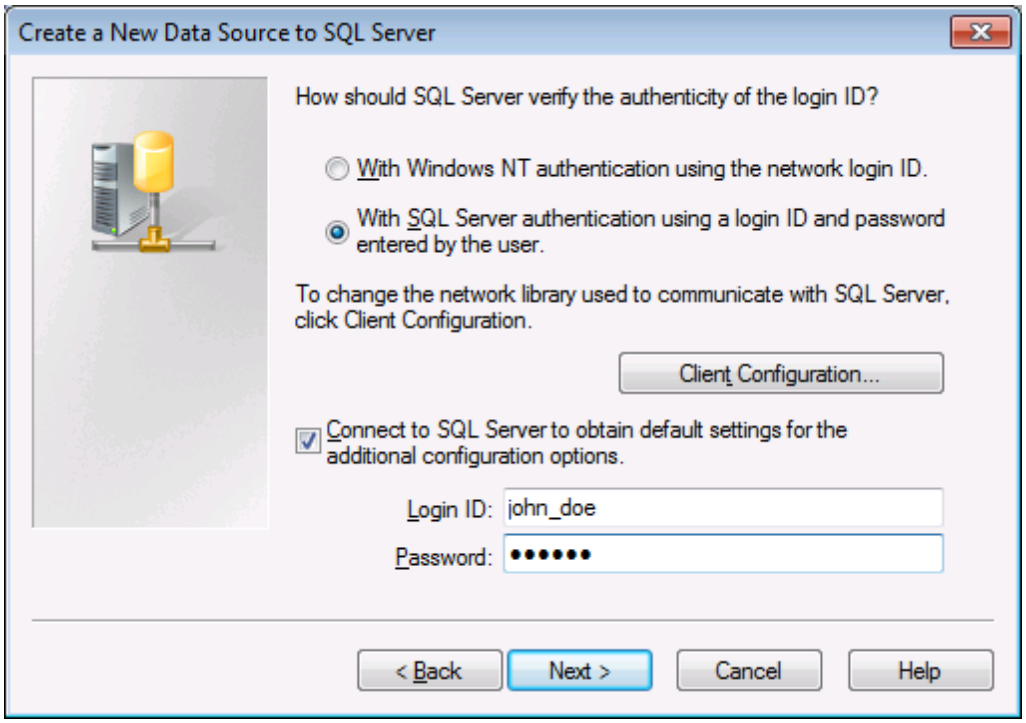
© 2018 Altova GmbH



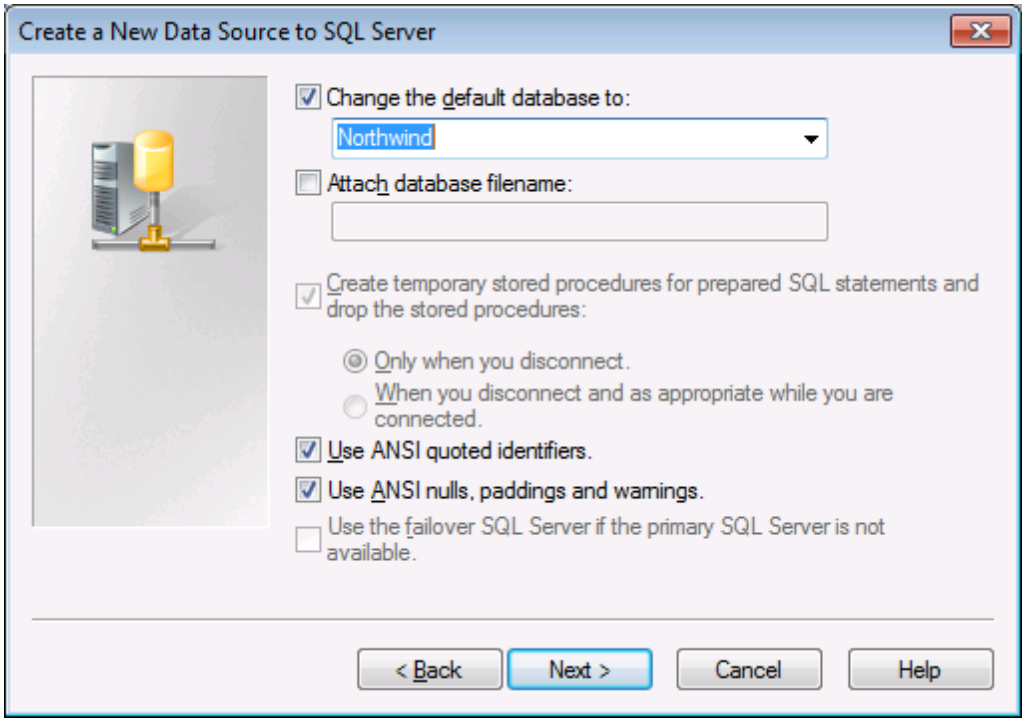
4. Seleccione **SQL Server** (o **SQL Server Native Client**) y después haga clic en *DSN de usuario* (o *DSN de sistema*).



5. Introduzca un nombre y una descripción para poder identificar esta conexión. Después seleccione el SQL Server al que desea conectarse (en este ejemplo es **SQLSERV01**).



6. Si el servidor de BD se configuró para admitir conexiones desde usuarios autenticados en el dominio de Windows, seleccione *Utilizar seguridad integrada de Windows NT*. De lo contrario, seleccione *Con autenticación SQL Server* e introduzca los datos donde corresponda.



7. Seleccione la base de datos a la que desea conectarse (en este ejemplo es **Northwind**).

© 2018 Altova GmbH

8. Haga clic en **Finalizar**.

7.2.1.10.10 Conectarse a MySQL (ODBC)

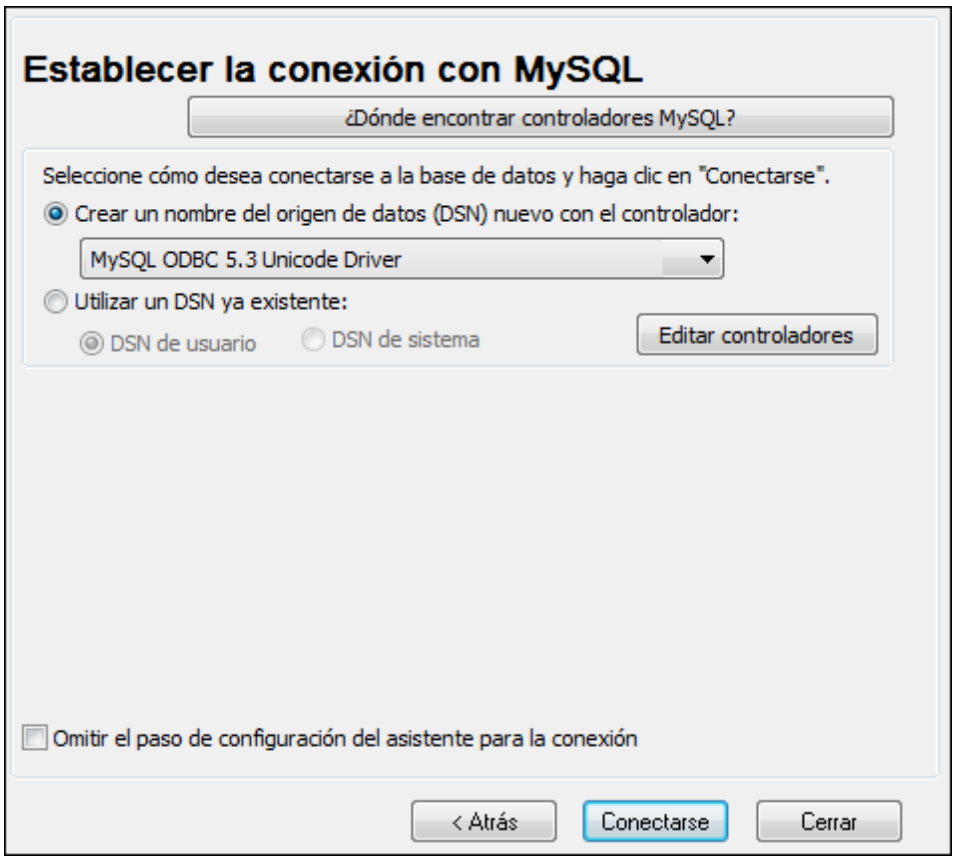
Este ejemplo explica cómo conectarse a un servidor de base de datos MySQL desde un equipo Windows con el controlador ODBC. El controlador ODBC MySQL no está disponible en Windows así que deberá descargarlo e instalarlo por separado. En este ejemplo usamos la versión 5.3.4 del controlador, que se puede descargar del sitio web oficial (consulte el apartado Resumen de controladores de base de datos).

Requisitos

- El controlador MySQL ODBC está instalado en el sistema.
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

Para conectarse a MySQL por ODBC:

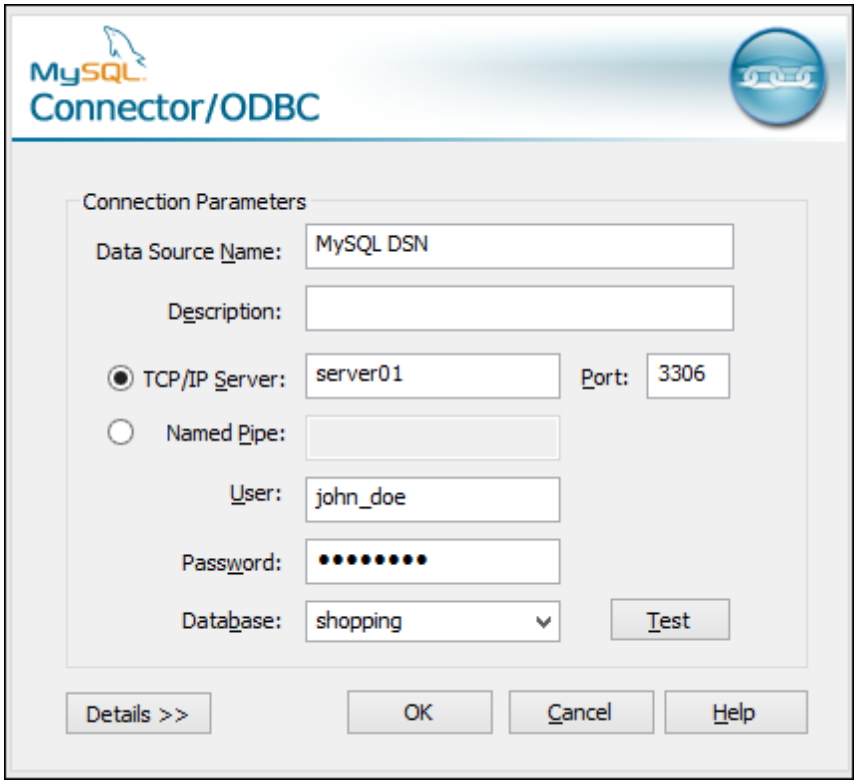
1. Inicie el asistente para la conexión de base de datos.
2. Seleccione la opción *MySQL (ODBC)* y después haga clic en **Siguiente**.



3. Seleccione la opción *Crear un DNS nuevo con el controlador* y seleccione un controlador MySQL. Si no hay ningún controlador MySQL en la lista, haga clic en **Editar controladores** y seleccione uno (la lista contiene todos los controladores ODBC que

están instalados en el sistema).

4. Haga clic en **Conectarse**.



5. En el cuadro de texto *Nombre del origen de datos* introduzca un nombre que le ayude a identificar este origen de datos ODBC más adelante.
6. Rellene las credenciales de la conexión de BD (servidor TCP/IP, usuario, contraseña), seleccione una base de datos y haga clic en **Aceptar**.

Nota: si el servidor de BD es remoto, el administrador del servidor debe configurarlo para que acepte conexiones remotas desde la dirección IP de su equipo. Además, si hace clic en **Detalles>>**, podrá configurar algunos parámetros más. Consulte la documentación del controlador antes de cambiar los valores predeterminados.

7.2.1.10.11 Conectarse a Oracle (ODBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Oracle desde MapForce. En este ejemplo la base de datos Oracle está en un equipo de la red y la conexión se establece a través de un cliente de base de datos Oracle instalado en el equipo local.

En este ejemplo ofrecemos instrucciones para configurar un DSN ODBC con el asistente para la conexión de base de datos de MapForce. Si ya tiene un DSN o prefiere crear uno desde el administrador de orígenes de datos ODBC de Windows, puede seleccionarlo desde el asistente. Para más información consulte el apartado Configurar una conexión ODBC.

© 2018 Altova GmbH

Requisitos

- El cliente de BD Oracle (que incluye el controlador ODBC Oracle) está instalado y configurado en el sistema. Para más información consulte la documentación del software de Oracle.
- El archivo `tnsnames.ora` ubicado en el directorio de inicio de Oracle contiene una entrada que describe los parámetros de conexión de la base de datos:

```
ORCL =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = server01)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SID = orcl)
      (SERVER = DEDICATED)
    )
  )
```

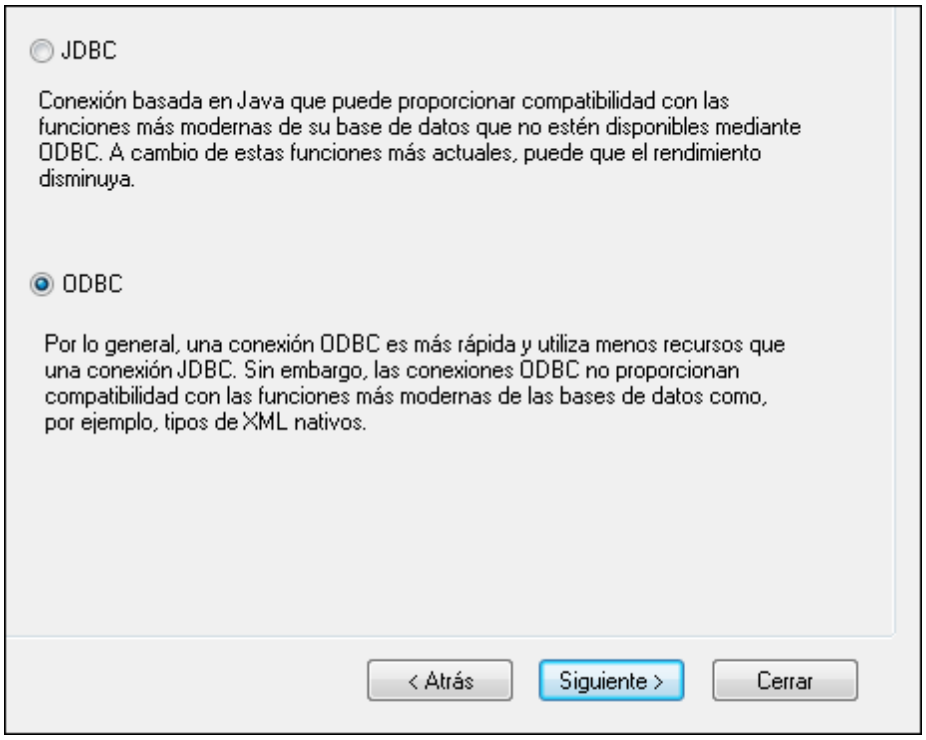
La ruta de acceso del archivo `tnsnames.ora` depende de la ubicación del directorio de inicio de Oracle. Por ejemplo, en el caso del cliente de base de datos Oracle 11.2.0, la ruta de acceso predeterminada del directorio de inicio podría ser:

```
C:\app\nombreUsuario\product\11.2.0\client_1\network\admin\tnsnames.ora
```

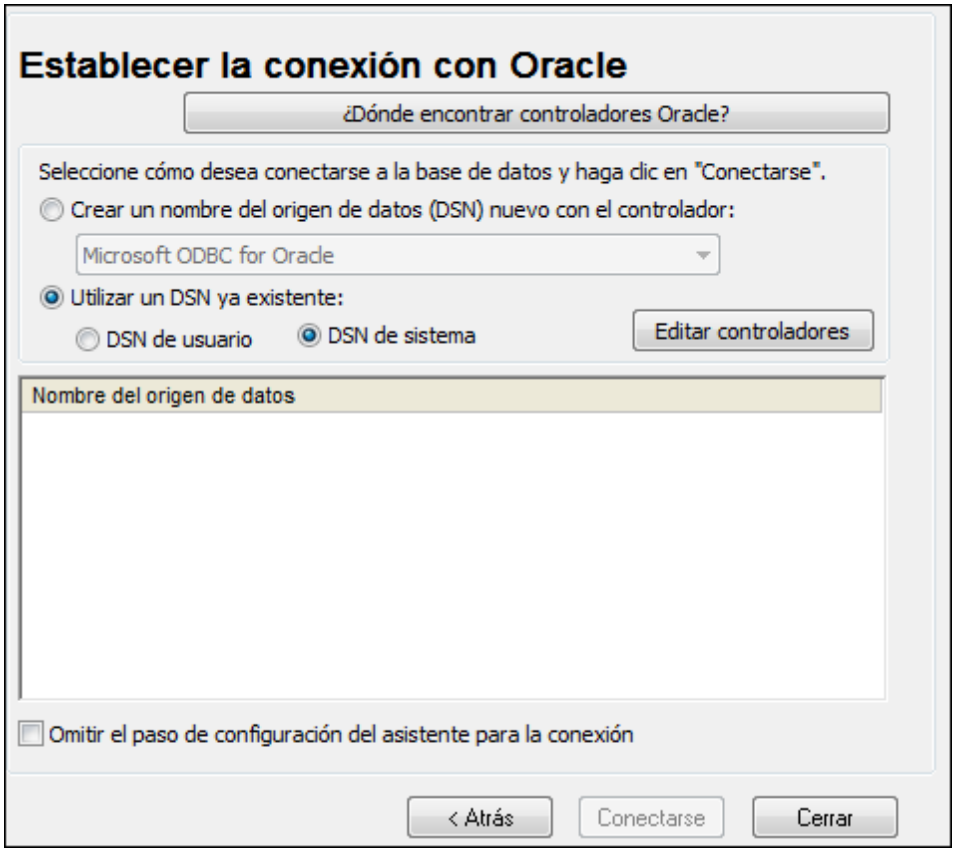
En el archivo `tnsnames.ora` puede introducir entradas nuevas, pegando los datos de conexión y guardando el archivo o ejecutando el asistente *Net Configuration Assistant* de Oracle (si está disponible).

Para conectarse a Oracle por ODBC:

1. Inicie el asistente para la conexión de base de datos.
2. Seleccione la opción *Oracle (ODBC / JDBC)* y después haga clic en **Siguiente**.

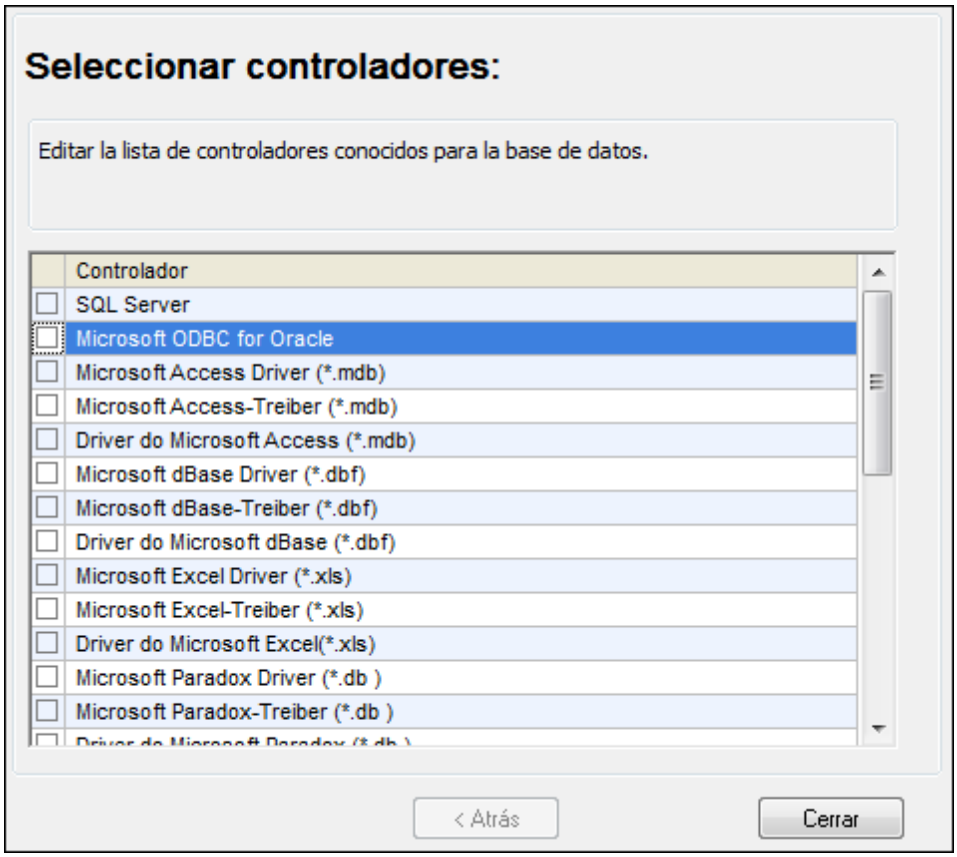


3. Seleccione el botón de opción *ODBC*.

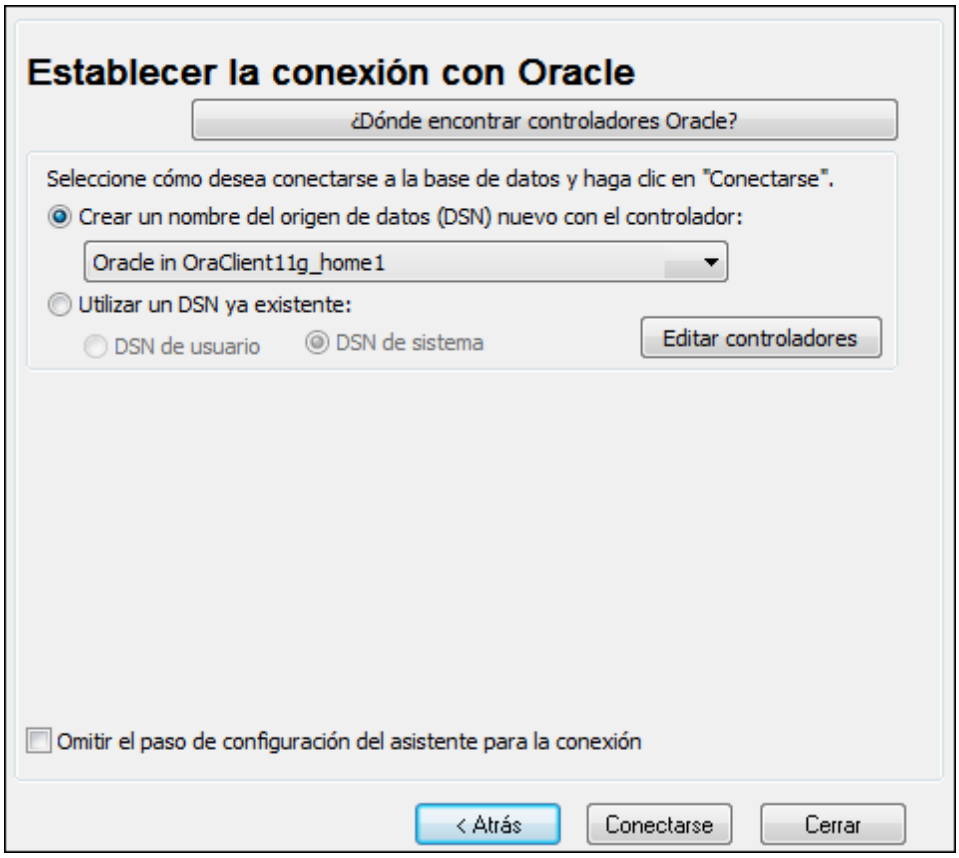


 © 2018 Altova GmbH

4. Haga clic en **Editar controladores**.



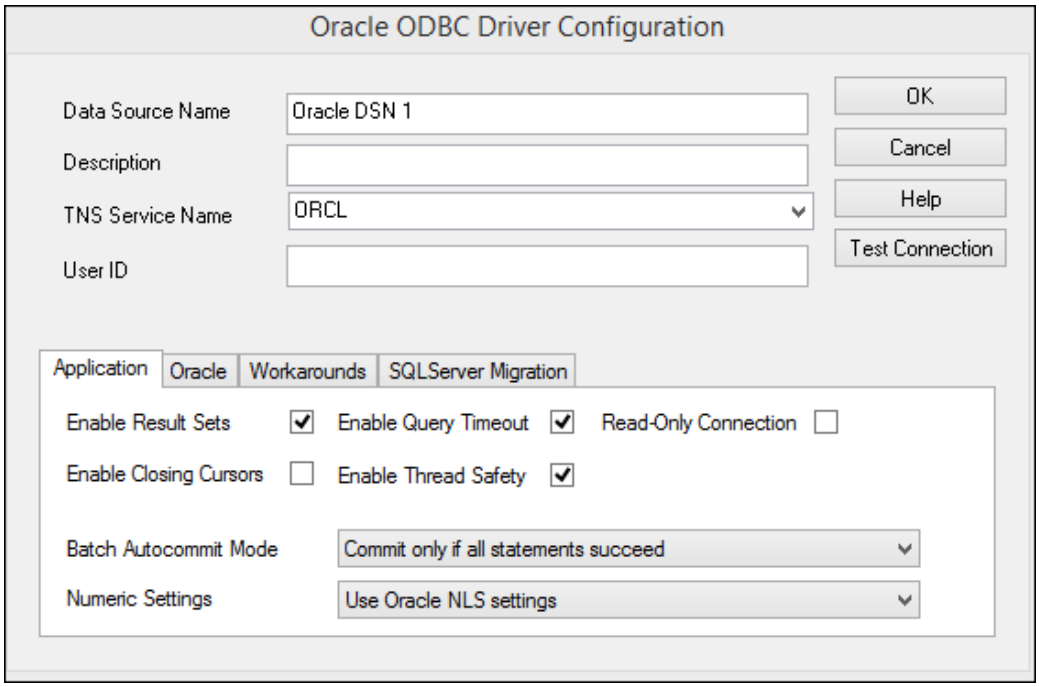
5. Seleccione los controladores Oracle que desea usar (en este ejemplo usamos **Oracle in OraClient11g_home1**). La lista incluye todos los controladores Oracle que están disponibles en el sistema después de instalar el cliente Oracle.
6. Haga clic en **Atrás**.
7. Seleccione la opción *Crear un DSN nuevo con el controlador* y después seleccione el controlador de Oracle elegido en el paso nº 4.



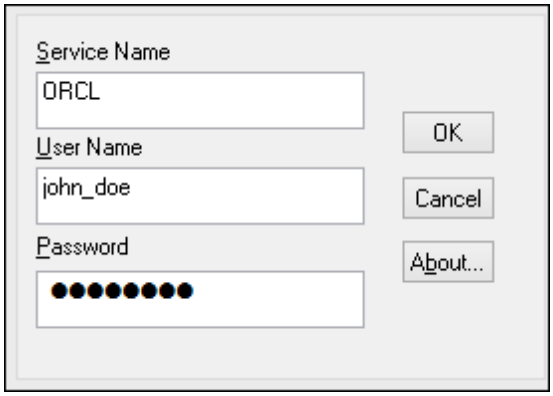
Recomendamos no utilizar el controlador **Microsoft ODBC for Oracle**. Microsoft recomienda utilizar el controlador ODBC que ofrece Oracle (consulte http://msdn.microsoft.com/en-us/library/ms714756%28v=vs.85%29.aspx)

8. Haga clic en **Conectarse**.

© 2018 Altova GmbH



9. En el cuadro de texto *Nombre del origen de datos* introduzca un nombre que le ayude a identificar el origen de datos más adelante (en este ejemplo **Oracle DSN 1**).
10. En el cuadro de texto *Nombre del servicio TNS* introduzca el nombre de la conexión tal y como se define en el archivo `tnsnames.ora` (ver Requisitos). En este ejemplo el nombre de la conexión es **ORCL**.
11. Haga clic en **Aceptar**.



12. Escriba el nombre de usuario y la contraseña de la base de datos y haga clic en **Aceptar** para terminar.

7.2.1.10.12 Conectarse a Oracle (JDBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Oracle desde un equipo cliente, usando la interfaz JDBC. La conexión se crea como una conexión Java pura, usando el paquete **Oracle Instant Client Package (Basic)** que se puede descargar del sitio web de

Oracle. La ventaja de este tipo de conexión es que solamente exige el entorno Java y las bibliotecas .jar que vienen con el paquete Oracle Instant Client Package. Es decir, no es necesario instalar ni configurar clientes de base de datos más complejos.

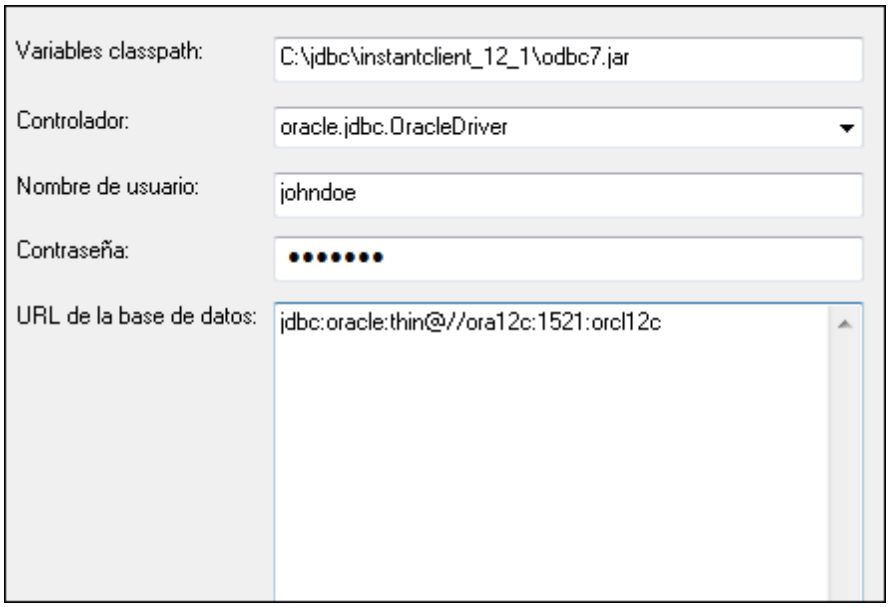
Requisitos

- Java Runtime Environment (JRE) o Java Development Kit (JDK) está instalado en el sistema.
- La variable de entorno `PATH` del sistema incluye la ruta al directorio `bin` del directorio de instalación de JRE o JDK (p. ej. `C:\Archivos de programa (x86)\Java\jre1.8.0_51\bin`).
- El paquete **Oracle Instant Client Package (Basic)** está disponible en el sistema operativo. El paquete se puede descargar del sitio web oficial de Oracle. En este ejemplo utilizamos la versión 12.1.0.2.0 para Windows de 32 bits.
- Disponer de los datos de conexión: host, puerto, nombre del servicio, nombre de usuario y contraseña.

Para conectarse a Oracle a través del paquete Instant Client Package:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones JDBC**.
3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, el archivo .jar está ubicado en esta ruta de acceso: `C:\jdbc\instantclient_12_1\odbc7.jar`. Tenga en cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase Configurar la variable CLASSPATH*).
4. En el campo *Controlador* seleccione **oracle.jdbc.OracleDriver** o **oracle.jdbc.driver.OracleDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.
5. Introduzca el nombre de usuario y contraseña de la base de datos.

© 2018 Altova GmbH



6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su servidor de base de datos).

```
jdbc:oracle:thin:@//host:puerto:servicio
```

7. Haga clic en **Conectarse**.

7.2.1.10.13 Conectarse a PostgreSQL (ODBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos PostgreSQL desde un equipo Windows con el controlador ODBC. El controlador ODBC PostgreSQL no está disponible en Windows así que deberá descargarlo e instalarlo por separado. En este ejemplo usamos la versión 09_03_300-1 del controlador psqlODBC, que se puede descargar del sitio web oficial (consulte el apartado Resumen de controladores de base de datos).

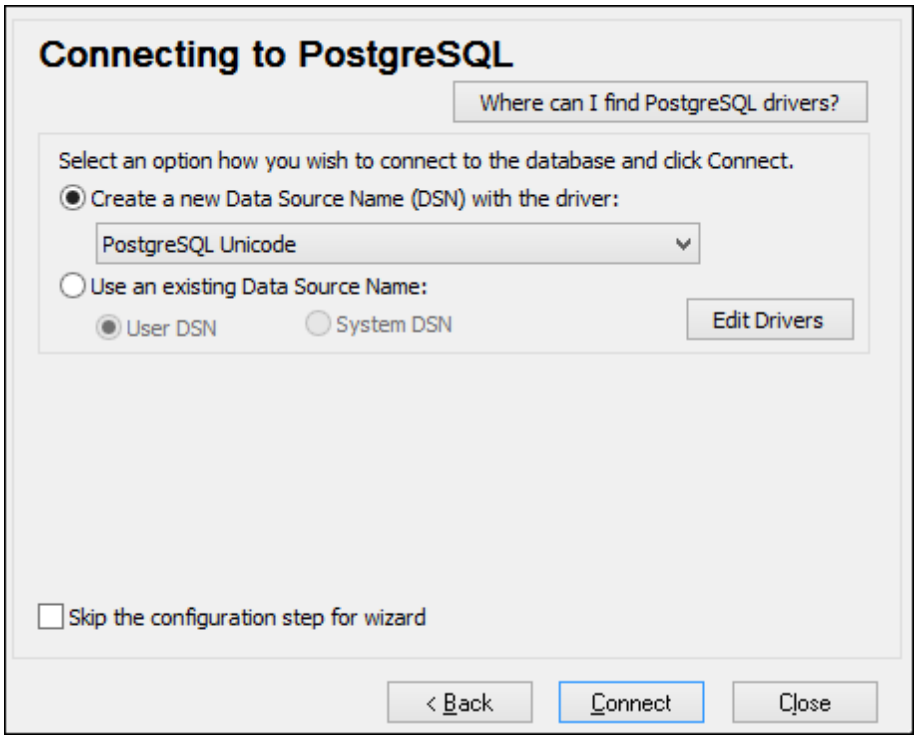
Nota: también se puede conectar a un servidor de base de datos PostgreSQL directamente (es decir, sin el controlador ODBC). Consulte el apartado Configurar una conexión PostgreSQL para obtener más información.

Requisitos

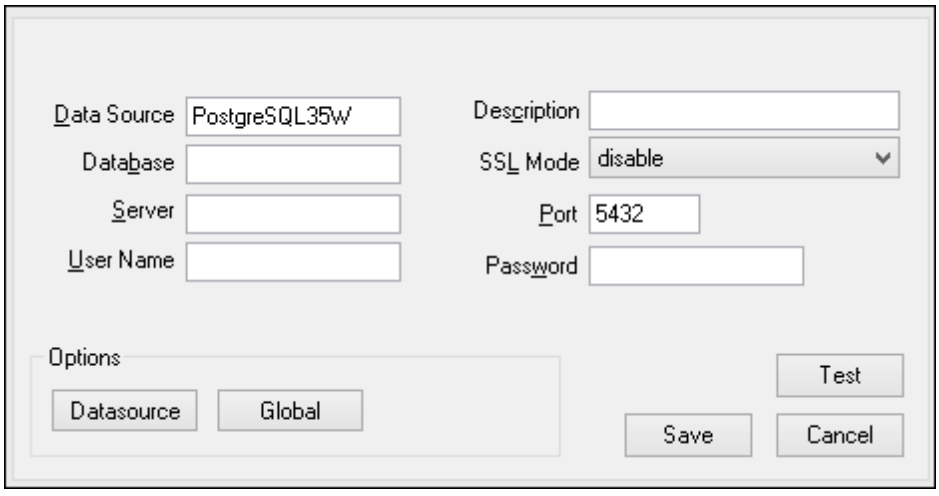
- El controlador *psqlODBC* está instalado en el sistema.
- Disponer de los datos de conexión: servidor, puerto, base de datos, nombre de usuario y contraseña.

Para conectarse a PostgreSQL por ODBC:

1. Inicie el asistente para la conexión de base de datos.
2. Seleccione la opción *PostgreSQL (ODBC)* y haga clic en **Siguiente**.



3. Seleccione la opción *Crear un DSN nuevo con el controlador* y seleccione el controlador PostgreSQL. Si no hay ningún controlador PostgreSQL en la lista, haga clic en Editar controldores y seleccione uno (la lista incluye todos los controladores ODBC que están instalados en el sistema).
4. Haga clic en **Conectarse**.



5. Rellene las credenciales de la conexión de base de datos (base de datos, servidor, puerto, nombre de usuario y contraseña) y haga clic en **Aceptar**.

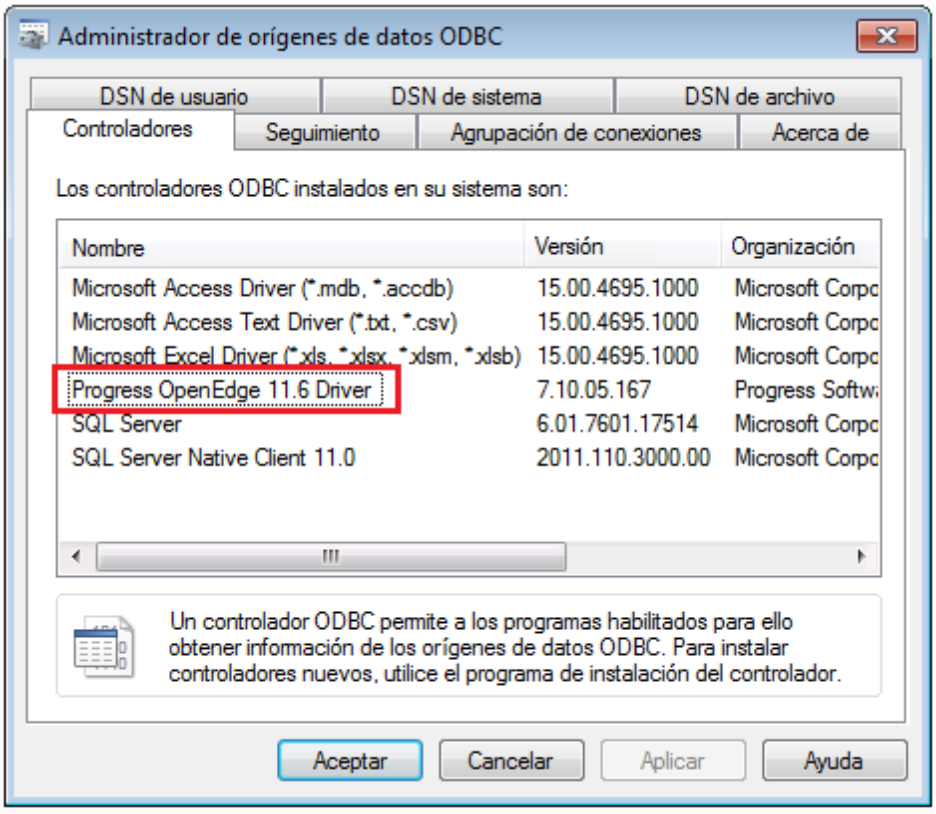
© 2018 Altova GmbH

7.2.1.10.14 Conectarse a Progress OpenEdge (ODBC)

En este apartado encontrará instrucciones para conectarse a un servidor de base de datos Progress OpenEdge por medio del controlador ODBC Progress OpenEdge 11.6.

Requisitos

- El controlador *ODBC Connector for Progress OpenEdge* está instalado en el sistema operativo. Este controlador se puede descargar del sitio web del proveedor (consulte la lista del apartado Resumen de controladores de base de datos). Si trabaja con la versión de 32 bits de MapForce, descargue el controlador de 32 bits. Por el contrario, si usa la versión de 64 bits, descargue el controlador de 64 bits. Una vez finalizada la instalación, compruebe que el controlador ODBC está disponible en el equipo (*véase Ver los controladores ODBC disponibles*).



- Disponer de los datos de conexión: nombre de host, número de puerto, nombre de la base de datos, id. de usuario y contraseña.

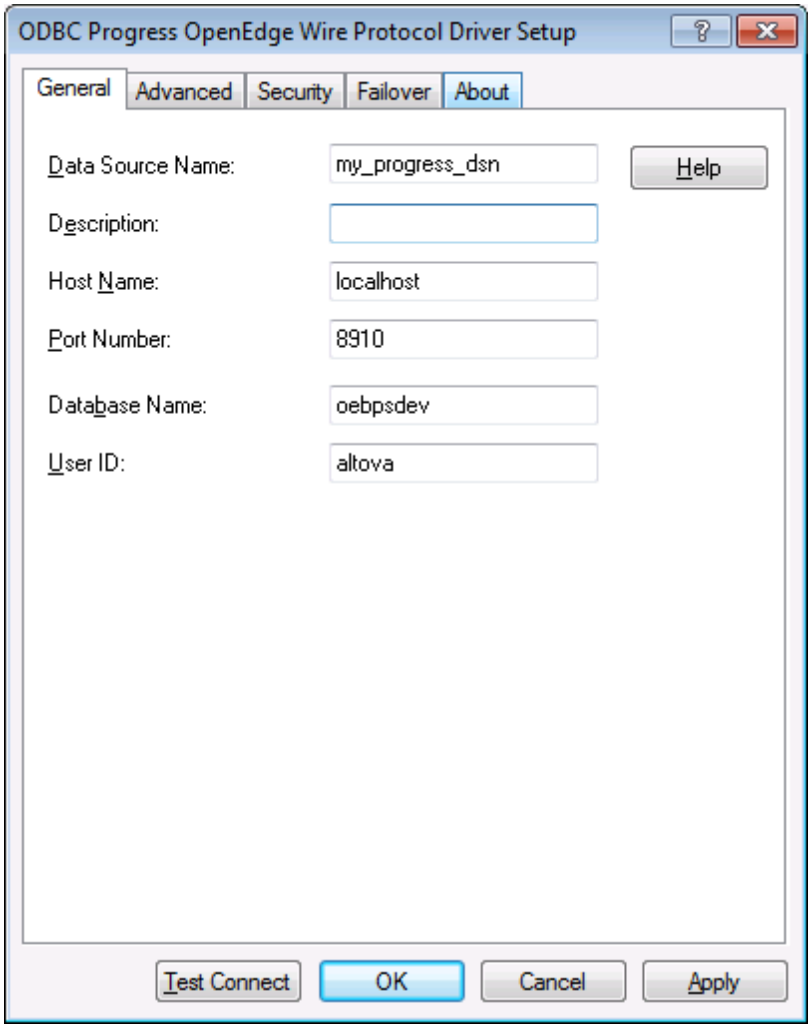
Para conectarse a Progress OpenEdge por ODBC:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en el botón Conexiones ODBC.
3. Haga clic en *DSN de usuario* (o en *DSN de sistema* o *DSN de archivo*).

4. Ahora haga clic en el icono **Agregar** .
5. Seleccione el controlador **Progress OpenEdge** de la lista y haga clic en el botón **DSN de usuario** (o **DSN de sistema** según corresponda).

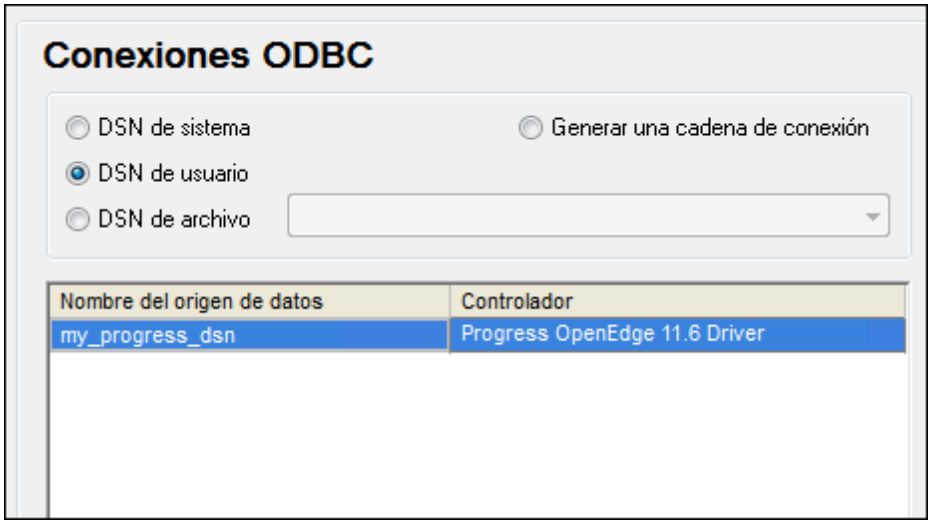


6. Rellene las credenciales para la conexión con la base de datos (base de datos, servidor, puerto, nombre de usuario y contraseña) y haga clic en **Aceptar**. Para probar la conexión antes de guardar los datos introducidos haga clic en el botón **Probar conexión**.



© 2018 Altova GmbH

7. Haga clic en **Aceptar**. El origen de datos nuevo aparece ahora en la lista de orígenes de datos ODBC.



8. Para terminar haga clic en **Conectarse**.

7.2.1.10.15 Conectarse a Progress OpenEdge (JDBC)

En este apartado encontrará instrucciones para conectarse a un servidor de base de datos Progress OpenEdge 11.6 por JDBC.

Requisitos

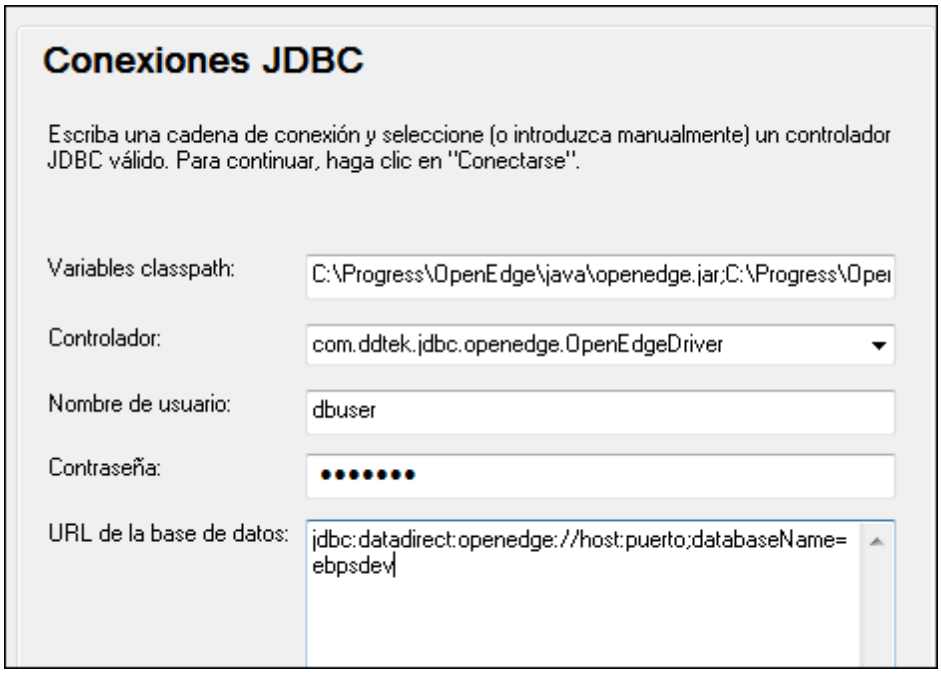
- Java Runtime Environment (JRE) o Java Development Kit (JDK) está instalado en el sistema. La plataforma de MapForce (32 o 64 bits) debe coincidir con la de JRE/JDK.
- La variable de entorno `PATH` del sistema incluye la ruta al directorio `bin` del directorio de instalación de JRE o JDK (p. ej. `C:\Archivos de programa (x86)\Java\jre1.8.0_51\bin`).
- El controlador JDBC Progress OpenEdge está disponible en el sistema operativo. En este ejemplo la conexión por JDBC se consigue con los archivos de controlador **openedge.jar** y **pool.jar** disponibles en `C:\Progress\OpenEdge\java` y que se instalan con OpenEdge SDK.
- Disponer de los datos de conexión: host, puerto, nombre de la base de datos, nombre de usuario y contraseña.

Para conectarse a Progress OpenEdge por JDBC:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones JDBC**.
3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, los archivos .jar están ubicados en esta ruta de acceso: **`C:\Progress\OpenEdge\java\openedge.jar;C:\Progress\OpenEdge\java\pool.jar;`**. Tenga en

cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase Configurar la variable CLASSPATH*).

4. En el campo *Controlador* seleccione **com.ddtek.jdbc.openedge.OpenEdgeDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.



5. Introduzca el nombre de usuario y la contraseña de la base de datos.
6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su base de datos).

```
jdbc:datadirect:openedge://host:puerto;databaseName=nombre_BD
```

7. Haga clic en **Conectarse**.

7.2.1.10.16 Conectarse a Sybase (JDBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Sybase por JDBC.

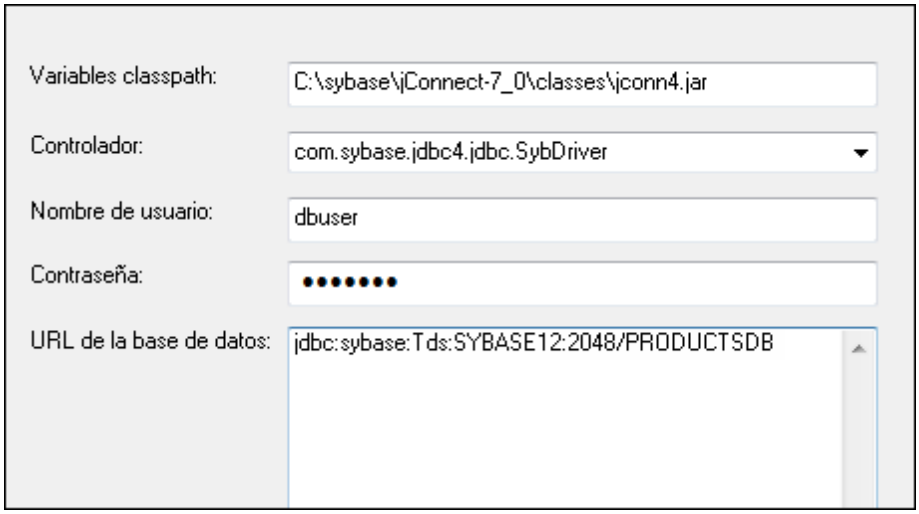
Requisitos

- Java Runtime Environment (JRE) está instalado en el sistema.
- El componente Sybase *jConnect* está instalado en el sistema (en este ejemplo usamos *jConnect 7.0* que se instala con el cliente *Sybase Adaptive Server Enterprise PC Client*. Consulte la documentación de Sybase para obtener más información.
- Disponer de los datos de conexión: host, puerto, nombre de la base de datos, nombre de usuario y contraseña.

© 2018 Altova GmbH

Para conectarse a Sybase por JDBC:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones JDBC**.
3. En el campo *Variables classpath* debe introducir la ruta de acceso del archivo .jar que ofrece conectividad con la base de datos. Si fuera necesario, también puede introducir una lista de rutas de archivo .jar separadas por caracteres de punto y coma. En este ejemplo, el archivo .jar está ubicado en esta ruta de acceso: `C:\sybase\jConnect-7_0\classes\jconn4.jar`. Tenga en cuenta que este campo puede dejarse en blanco si añadió la ruta de acceso de los archivos .jar a la variable de entorno CLASSPATH del sistema operativo (*véase Configurar la variable CLASSPATH*).
4. En el campo *Controlador* seleccione **com.sybase.jdbc4.jdbc.SybDriver**. Recuerde que esta entrada solo estará disponible si se encuentra una ruta de archivo .jar válida en el campo Variables *classpath* o en la variable de entorno CLASSPATH del sistema operativo.



5. Introduzca el nombre de usuario y la contraseña de la base de datos.
6. Introduzca la cadena de conexión para el servidor de BD en el cuadro de texto *URL de la base de datos* (reemplace lo valores resaltados con los de su base de datos).

```
jdbc:sybase:Tds:nombreHost:puerto/nombreBD
```

7. Haga clic en **Conectarse**.

7.2.1.10.17 Conectarse a Teradata (ODBC)

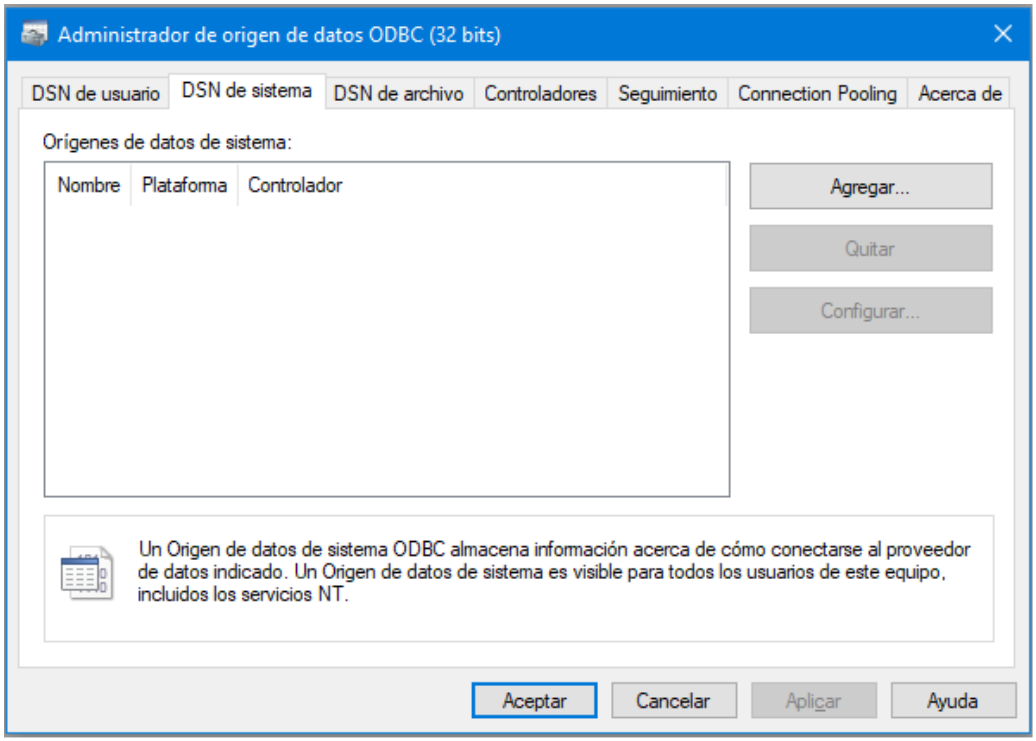
Este ejemplo explica cómo conectarse a un servidor de base de datos Teradata por ODBC.

Requisitos

- Debe tener instalado el controlador ODBC de Teradata (https://downloads.teradata.com/download/connectivity/odbc-driver/windows). Para este ejemplo se utiliza el controlador ODBC de Teradata para Windows versión 16.20.00.
- Disponer de los datos de conexión: host, nombre de usuario y contraseña.

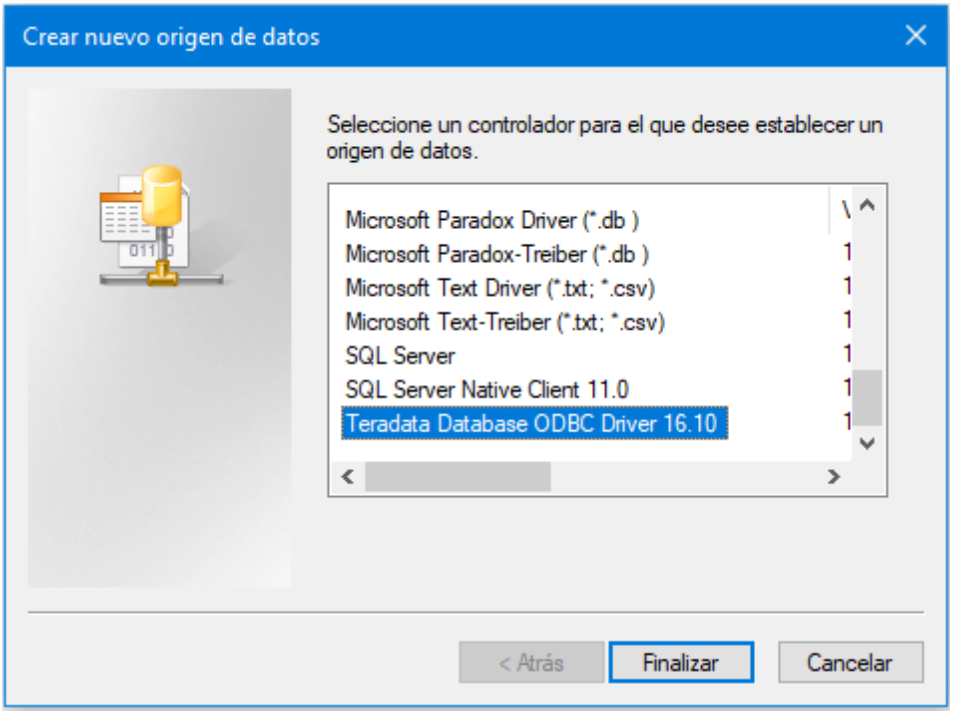
Para conectarse a Teradata por ODBC:

1. Pulse la tecla Windows, teclee "ODBC" y seleccione **Configurar orígenes de datos ODBC (32 bits)** en la lista de sugerencias. Si tiene un controlador ODBC de 64 bits, seleccione la opción **Configurar orígenes de datos ODBC (64 bits)** y utilice MapForce de 64 bits a partir de ese momento.

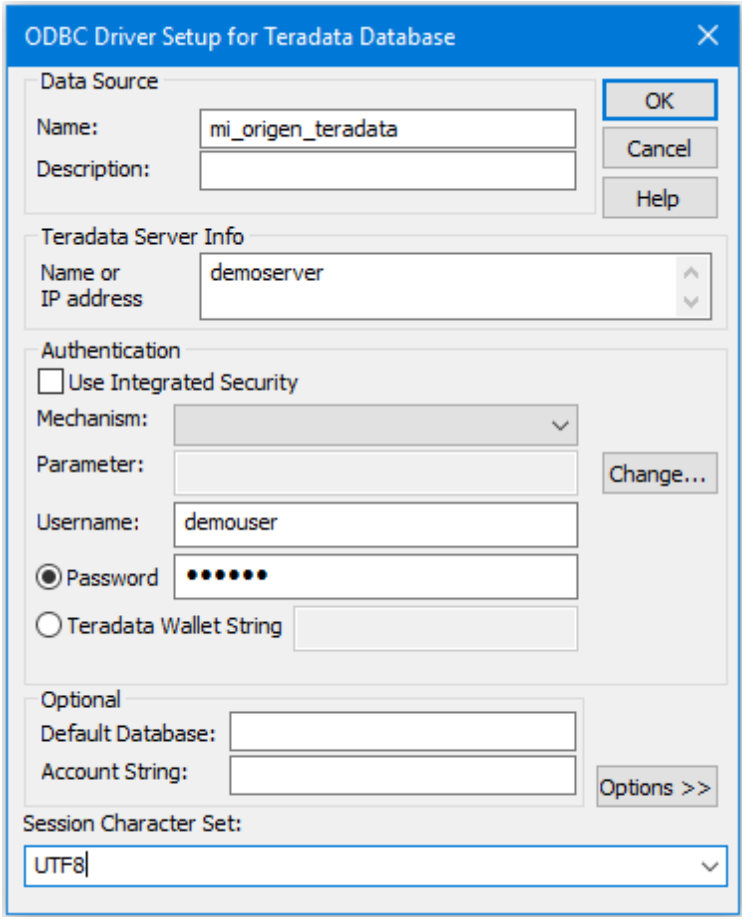


2. Haga clic en la pestaña *DSN de sistema* y después haga clic en **Agregar**.

© 2018 Altova GmbH

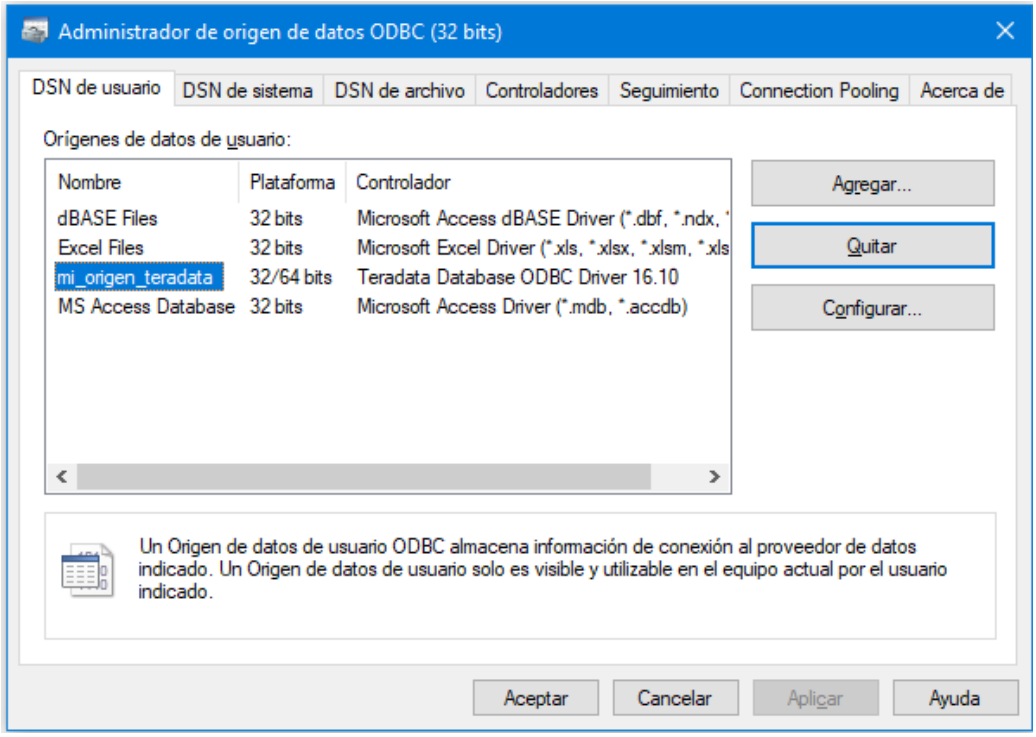


3. Seleccione **Teradata Database ODBC Driver** y después haga clic en **Finalizar**.

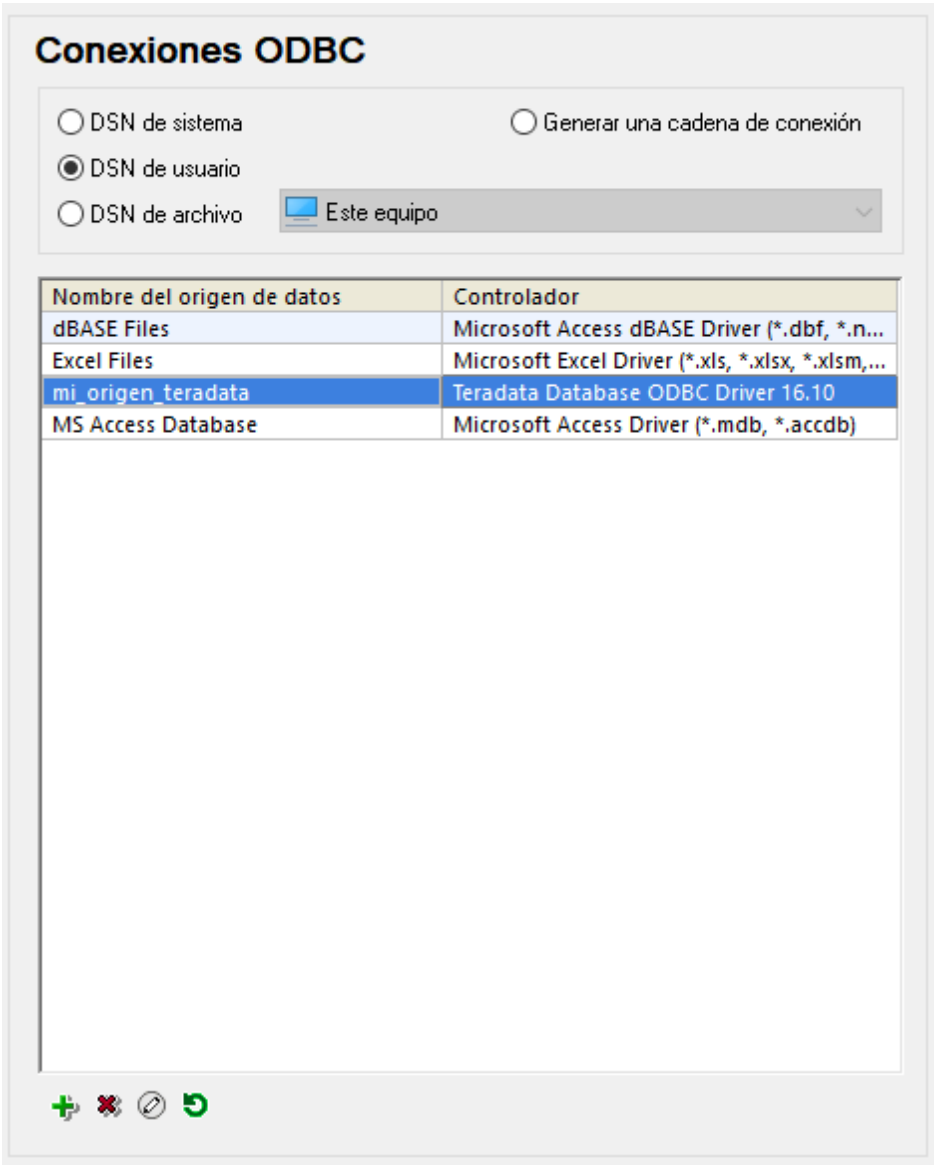


4. Introduzca el nombre y, si quiere, una descripción que le ayude a identificar este origen de datos ODBC más adelante. Además, deberá introducir las credenciales de la conexión de base de datos (servidor de BD, usuario y contraseña) y, si quiere, seleccione una base de datos.
5. Haga clic en **Aceptar**. El origen de datos aparecerá en la lista.

© 2018 Altova GmbH



6. Ejecute MapForce e inicie el asistente para la conexión de base de datos.
7. Haga clic en **Conexiones ODBC**.



8. Haga clic en *DSN de sistema,* seleccione el origen de datos creado en los pasos anteriores y después haga clic en el botón **Conectarse**.

Nota: si recibe el mensaje de error "Controlador devuelto no válido (o error en la devolución) SQL_DRIVER_ODBC_VER: 03.80", compruebe que la ruta de acceso del cliente ODBC (p. ej. **C:\Archivos de programa\Teradata\Client\16.10\bin**) existe en su variable de entorno PATH del sistema. Si falta esta ruta de acceso, entonces deberá añadirla a mano.

7.2.1.10.18 Conectarse a Teradata (JDBC)

Este ejemplo explica cómo conectarse a un servidor de base de datos Teradata por JDBC.

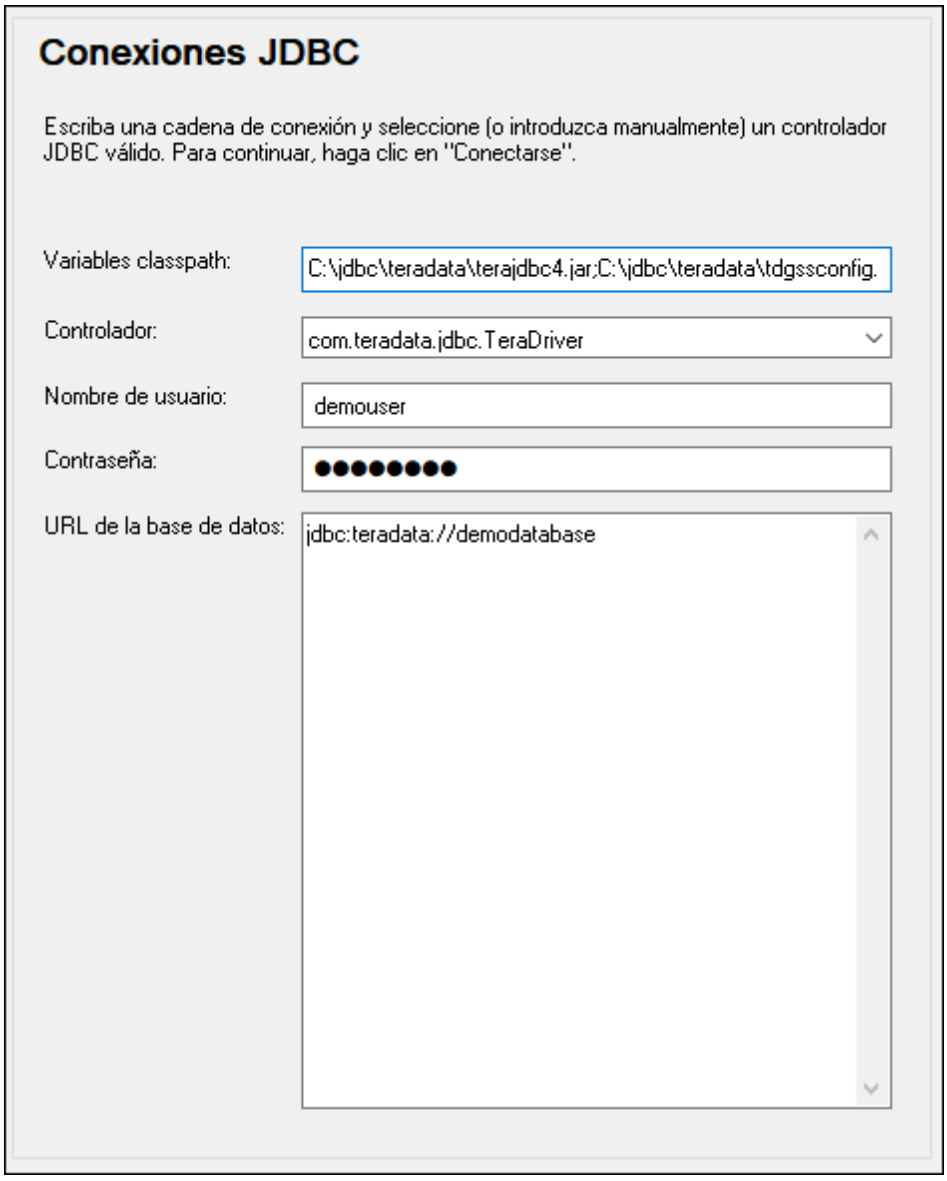
© 2018 Altova GmbH

Requisitos

- Debe tener instalado Java Runtime Environment (JRE) o Java Development Kit (JDK) en el sistema operativo.
- Debe contar con el controlador JDBC (archivos .jar que permiten conectarse a la base de datos) en el sistema operativo. Para este ejemplo usamos el controlador JDBC de Teradata 16.20.00.02 (https://downloads.teradata.com/download/connectivity/jdbc-driver).
- Disponer de los datos de conexión: host, base de datos, puerto, nombre de usuario y contraseña.

Para conectarse a Teradata por JDBC:

1. Inicie el asistente para la conexión de base de datos.
2. Haga clic en **Conexiones JDBC**.
3. Junto a *Variables Classpath* introduzca la ruta de acceso del archivo .jar que permite conectarse a la base de datos. Si es necesario, también puede introducir una lista de rutas de archivo .jar separadas por punto y comas. Para este ejemplo los archivos .jar están ubicados en esta ruta de acceso: **C:\jdbc\teradata**. Recuerde que puede dejar vacío el cuadro *Variables Classpath* si añadió las rutas de archivo .jar a la variable de entorno CLASSPATH del sistema operativo (véase Configurar la variable CLASSPATH).
4. En el cuadro *Controlador* seleccione **com.teradata.jdbc.TeraDriver**. Recuerde que esta entrada está disponible si en el cuadro *Variables Classpath* o en la variable de entorno CLASSPATH del sistema operativo se encuentra una ruta de archivo .jar válida (ver paso anterior).



5. Introduzca el nombre de usuario y la contraseña de la base de datos en las casillas correspondientes.
6. Introduzca la cadena de conexión en el cuadro de texto *URL de la base de datos* (reemplazando el valor resaltado con el valor correspondiente).

```
jdbc:teradata://nombreServidorBaseDatos
```

7. Por último, haga clic en **Conectarse**.

7.2.1.11 Conexiones de BD en Linux y macOS

Si trabaja con productos servidor de Altova (como MobileTogether Server, MapForce Server o StyleVision Server), es probable que diseñe soluciones de MobileTogether, asignaciones de

© 2018 Altova GmbH

datos de MapForce y transformaciones de StyleVision en un equipo de escritorio Windows y después los implemente en un equipo servidor (Windows, Linux o macOS) para automatizar su ejecución.

En esta documentación, el término *archivos de ejecución en servidor* se utiliza en referencia a este tipo de archivos:

- Archivos de ejecución de MapForce Server (.mfx)
- Archivos de diseño de MobileTogether (.mtd)
- Transformaciones de StyleVision (.sps) empaquetadas como formularios XML portátiles (.pxf).

Cuando implemente los archivos de ejecución en servidor pueden darse dos casos:

1. **Diseño y ejecución en Windows:** los diseños de MobileTogether, asignaciones de MapForce y transformaciones de StyleVision se diseñan en Windows y sus archivos de ejecución correspondientes se ejecutan también en un sistema Windows (bien el mismo equipo, bien un servidor Windows remoto).
2. **Diseño en Windows y ejecución en Linux o macOS:** los archivos se diseñan en Windows y sus archivos de ejecución correspondientes se implementan en Linux o macOS para ser ejecutados.

En el caso nº 1 (*Diseño y ejecución en Windows*), puede utilizar tecnologías y controladores de bases de datos ADO, ODBC y JDBC, así como conexiones SQLite (ver apartado Resumen de controladores de base de datos).

En el caso nº 2 (*Diseño en Windows y ejecución en Linux o macOS*) no podrá utilizar conexiones ADO ni ODBC. Solamente podrá usar conexiones SQLite (consulte Conexiones SQLite) y JDBC (consulte Conexiones JDBC).

Cuando implemente archivos de ejecución en un servidor, las bases de datos no se incluyen en el paquete implementado (tampoco las bases de datos basadas en archivos como SQLite y Microsoft Access). Por tanto, también deberá configurar una conexión a las bases de datos desde el servidor de implementación. En otras palabras, deberá utilizar la misma configuración de BD en el sistema donde diseñó los archivos y en el servidor donde los implemente. Hay una excepción a esta regla: las conexiones PostgreSQL nativas (no basadas en controladores). Este tipo de conexiones no requieren ninguna configuración fuera de MapForce. Consulte el apartado Configurar una conexión PostgreSQL para obtener más información.

Por lo general, el caso nº 2 es más complejo porque exige la misma configuración de BD en ambos equipos. Por eso recomendamos utilizar la característica de recursos globales de Altova, disponible en MapForce, MobileTogether Designer y StyleVision.

Por ejemplo, puede definir dos configuraciones de recurso global distintas para conectarse a la misma base de datos: una configuración incluirá la configuración de conexión con rutas de acceso de tipo Windows y la otra tendrá rutas de acceso tipo Linux. La primera conexión puede utilizarse para probar los archivos durante la fase de diseño y la segunda para ejecutar el archivo de ejecución en el servidor Linux.

7.2.1.11.1 Conexiones SQLite en Linux y macOS

No es necesario instalar SQLite en Linux y macOS porque los productos servidor de Altova incluyen compatibilidad con este tipo de base de datos. Por tanto, si sus archivos de ejecución en servidor incluyen llamadas a una base de datos SQLite, podrá ejecutarlos sin necesidad de instalar SQLite. Sin embargo, deberá asegurarse de que los archivos de ejecución en servidor utilizan la ruta de acceso correcta al archivo de BD. Es decir, antes de ejecutar los archivos de ejecución en el servidor Linux o macOS, compruebe que al archivo de BD se le hace referencia por medio de una ruta de acceso compatible con POSIX (Portable Operating System Interface). La ruta de acceso no incluirá letras de unidad de tipo Windows y los directorios estarán separados por barras diagonales. Por ejemplo, la ruta de acceso `/usr/local/mydatabase.db` es compatible con POSIX, pero `c:\sqlite\mydatabase.db` no lo es.

7.2.1.11.2 Conexiones JDBC en Linux y macOS

Para configurar una conexión JDBC en Linux o macOS:

1. Descargue el controlador JDBC que ofrece el proveedor de la BD e instálelo en el sistema. Recuerde que debe seleccionar la versión de 32 bits si su sistema es de 32 bits y la versión de 54 bits si el sistema es de 64 bits.
2. Configure las variables de entorno con la ubicación donde está instalado el controlador JDBC. Por lo general será necesario configurar la variable `CLASSPATH`, pero quizá alguna más. Consulte la documentación del controlador JDBC para obtener más información.

Nota: en macOS, el sistema espera que las bibliotecas JDBC instaladas estén en el directorio `/Library/Java/Extensions`. Por tanto, se recomienda desempaquetar el controlador JDBC en esta ubicación. De lo contrario, deberá configurar el sistema para que busque la biblioteca JDBC en una ruta de acceso diferente.

7.2.1.11.3 Conexiones Oracle en OS X Yosemite

En OS X Yosemite puede conectarse a bases de datos Oracle con el cliente **Oracle Database Instant Client**. Recuerde que si usa macOS con una versión de Java anterior a Java 8, también se podrá conectar con la biblioteca **JDBC Thin for All Platforms**, en cuyo caso puede ignorar las instrucciones que aparecen en este tema.

El cliente Oracle Instant Client se puede descargar del sitio web oficial de Oracle. Tenga en cuenta que en la página de descargas hay varios paquetes y deberá seleccionar un paquete con compatibilidad con Oracle Call Interface (OCI), como el paquete Instant Client Basic, por ejemplo. Además, recuerde que debe seleccionar la versión de 32 bits si su sistema es de 32 bits o la versión de 64 bits si su sistema es de 64 bits.

Tras descargar y desempaquetar el cliente Oracle Instant Client, edite el archivo de lista de propiedades (`.plist`) que viene con el instalador para que las variables de la tabla apunten a la ruta de acceso del controlador correspondiente. Por ejemplo:

 © 2018 Altova GmbH

Variable	Valor de muestra
CLASSPATH	/opt/oracle/instantclient_11_2/ojdbc6.jar:/opt/oracle/instantclient_11_2/ojdbc5.jar
TNS_ADMIN	/opt/oracle/NETWORK_ADMIN
ORACLE_HOME	/opt/oracle/instantclient_11_2
DYLD_LIBRARY_PATH	/opt/oracle/instantclient_11_2
PATH	$PATH:/opt/oracle/instantclient_11_2

Nota: los valores de las variables deben ajustarse a la ruta de acceso donde están instalados los archivos del cliente Oracle Instant Client.

7.2.2 Introducción a las asignaciones de datos de BD

Esta sección explica a grandes rasgos cómo trabajar con bases de datos en MapForce y cómo trabajar con una base de datos después de establecer la conexión con la base de datos correctamente (*véase* *Conectarse a una base de datos*). Más concretamente, encontrará información sobre cómo seleccionar los objetos de la BD que deben aparecer en la asignación, cómo gestionar las relaciones de base de datos y cómo configurar la base de datos para el proceso de asignación. Además encontrará varios ejemplos prácticos.

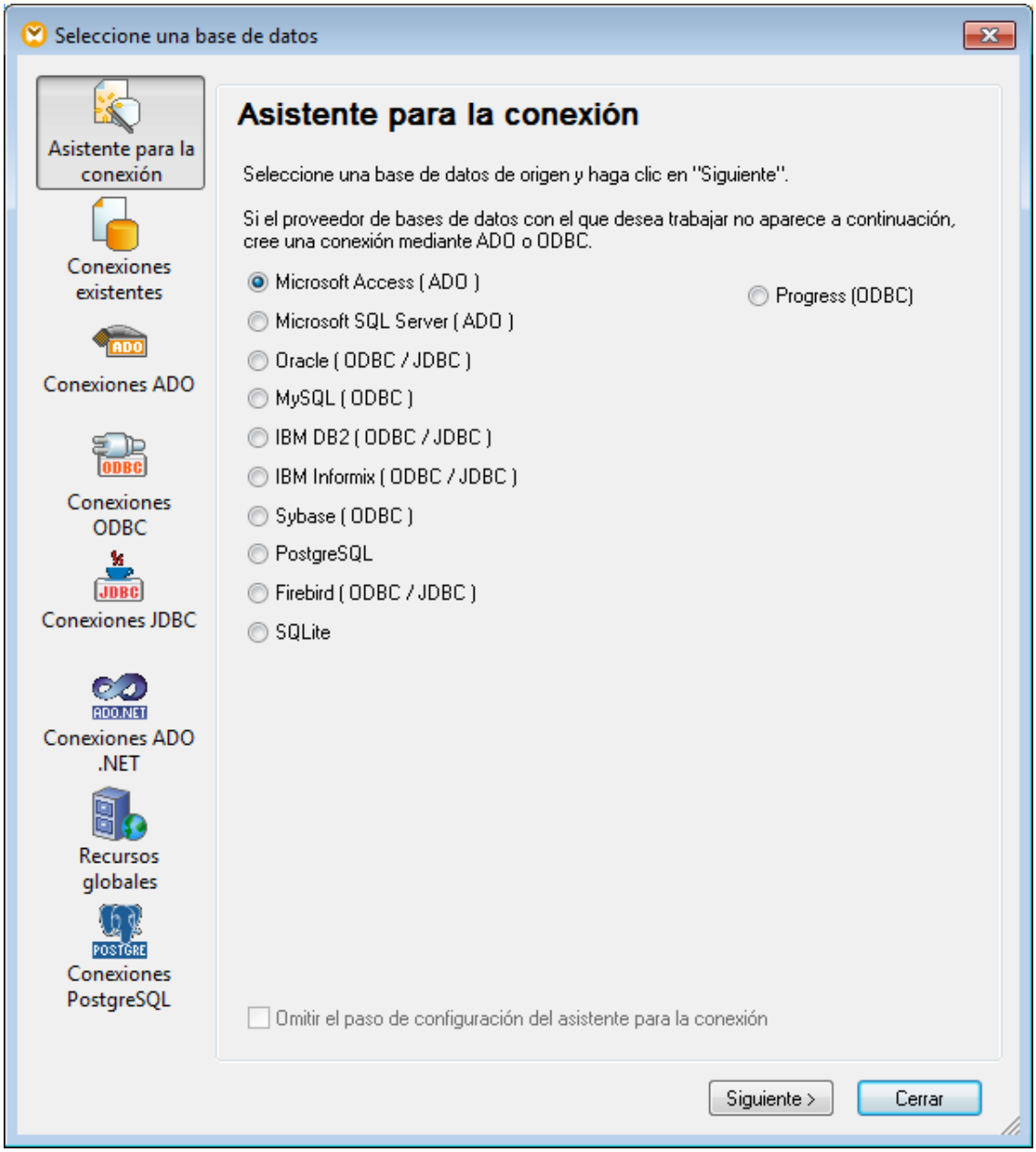
7.2.2.1 Agregar bases de datos a la asignación

Antes de agregar una base de datos a la asignación, asegúrese de que selecciona un lenguaje de transformación compatible con las asignaciones de datos de base de datos: el motor de transformación integrado o C++, C# o Java (*véase* *Seleccionar un lenguaje de transformación*). Recuerde que, si tiene pensado implementar la asignación en FlowForce Server o ejecutarla con MapForce Server o si tiene pensado usar características como la transferencia masiva o los procedimientos almacenados, entonces deberá seleccionar el motor de transformación integrado (BUILT-IN).

Tras seleccionar el lenguaje de transformación adecuado podrá agregar la base de datos a la asignación. Esto puede hacerse de dos maneras:

- Con el comando **Base de datos** del menú **Insertar**.
- Con el botón Insertar base de datos de la barra de herramientas.

Esto abre el asistente para la conexión a bases de datos, que le guiará en el proceso de conexión.



Asistente para la conexión a bases de datos

Nota: las bases de datos también se pueden agregar a la asignación en forma de variable (*véase Usar variables*). Cuando se añade una estructura de base de datos como variable, aparece el mismo asistente de conexión.

Para aprender a trabajar con este asistente y configurar una conexión con una de las bases de datos compatibles con MapForce debe consultar la sección Conectarse a una base de datos. Además, encontrará ejemplos con instrucciones paso a paso en estos apartados:

- Conectarse a Firebird (ODBC)
- Conectarse a IBM DB2 (ODBC)

© 2018 Altova GmbH

- Conectarse a una base de datos Microsoft Access
- Conectarse a Microsoft SQL Server (ADO)
- Conectarse MySQL (ODBC)
- Conectarse a Oracle (ODBC)
- Conectarse a PostgreSQL (ODBC)
- Conectarse a Progress OpenEdge (JDBC)
- Conectarse a una base de datos SQLite
- Conectarse a Sybase (JDBC)

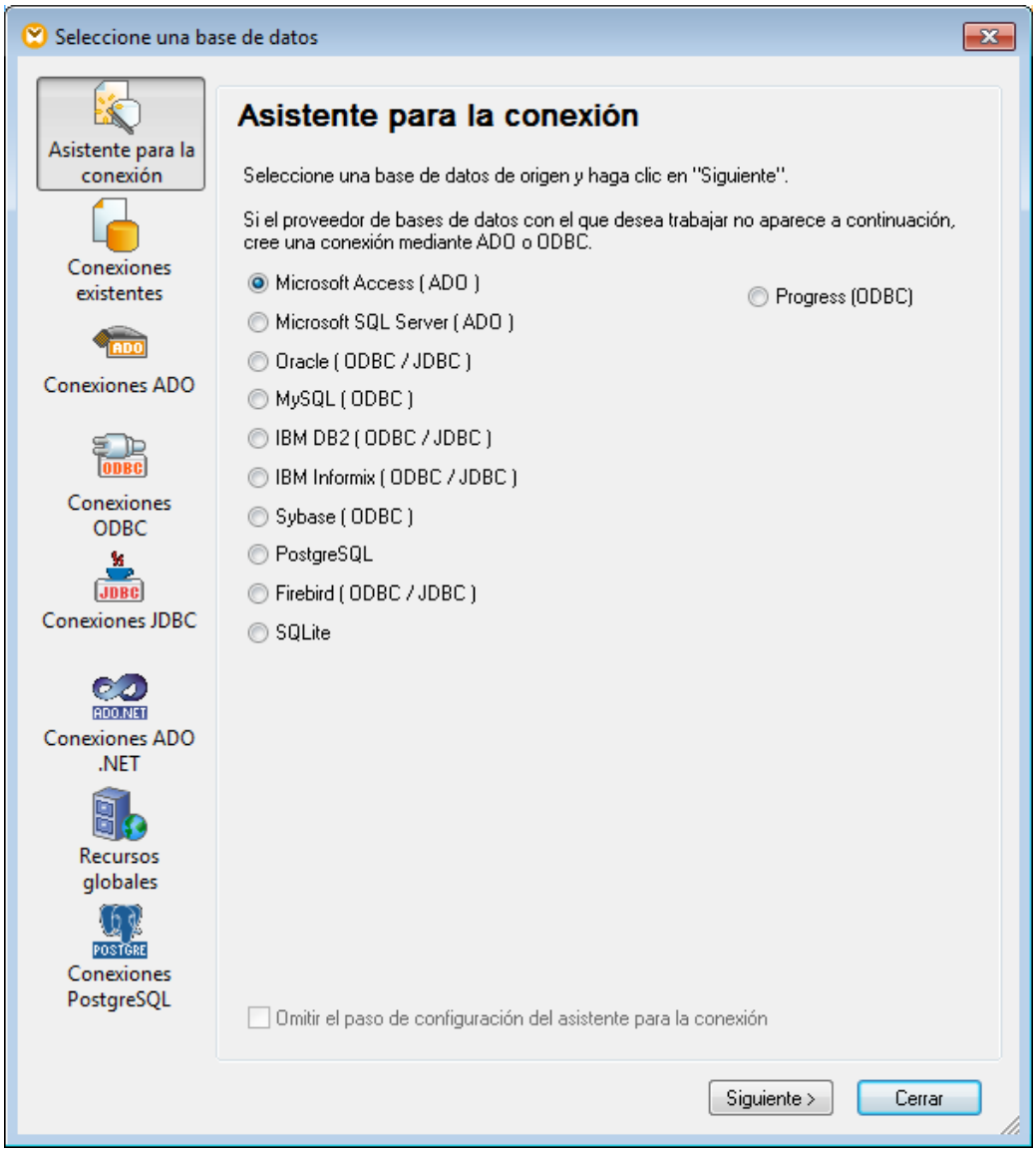
Una vez establecida la conexión con la base de datos, deberá seleccionar qué objetos de la base de datos deben aparecer en la asignación (*véase Agregar, editar y eliminar objetos de base de datos*).

7.2.2.2 Ejemplo: agregar la base de datos "altova.mdb" a la asignación

Este ejemplo explica cómo agregar una base de datos Microsoft Access a la asignación. La base de datos se llama **altova.mdb** y está en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**. La base de datos **altova.mdb** permite trabajar con varias acciones y conceptos descritos en esta documentación.

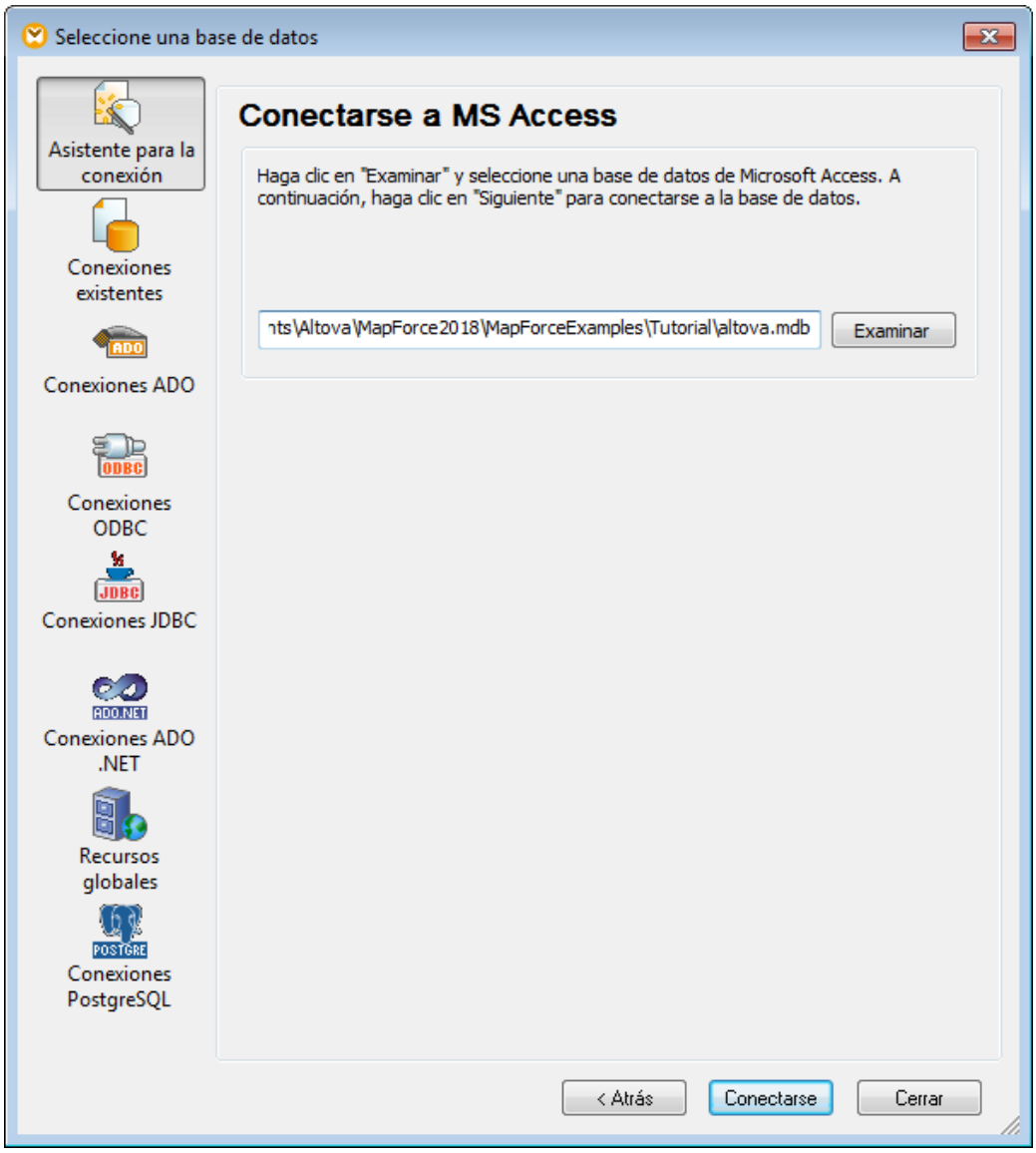
Para agregar la base de datos altova.mdb a la asignación:

1. En el menú **Insertar** haga clic en el comando **Base de datos** (o haga clic en el botón **Insertar base de datos** de la barra de herramientas).

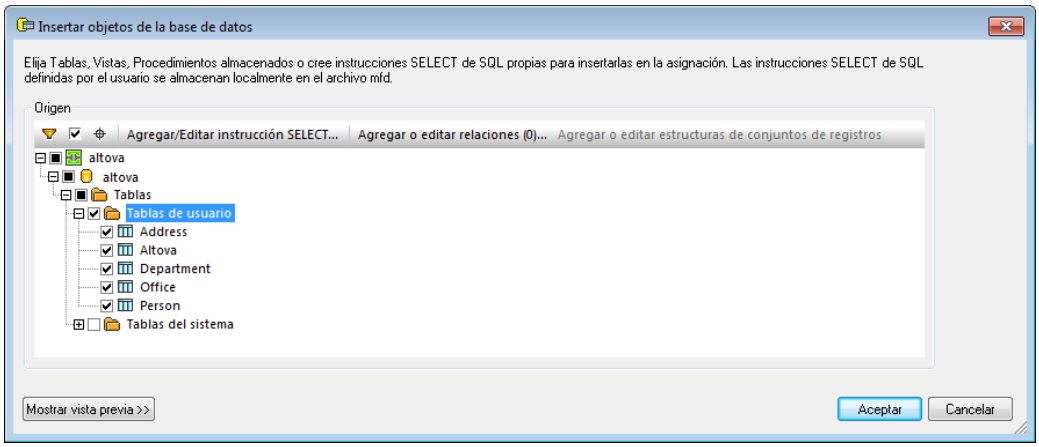


2. En el asistente para la conexión elija el botón de opción *Microsoft Access (ADO)* y haga clic en **Siguiente**.

© 2018 Altova GmbH



3. Navegue hasta el archivo **altova.mdb** de la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples** y haga clic en **Conectarse**.
4. Cuando la aplicación le pida que seleccione objetos de la base de datos, marque la casilla de `Tablas de usuario`.

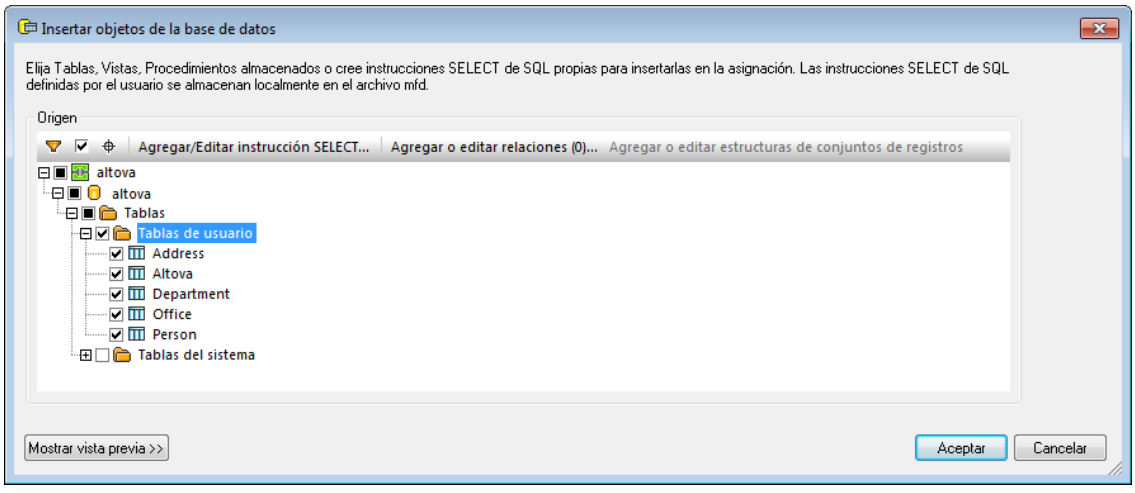


7.2.2.3 Agregar, editar y quitar objetos de BD

Algunas bases de datos pueden tener gran número de objetos (esquemas, tablas, vistas, etc.). En este apartado explicamos cómo incluir en la asignación solamente los objetos que necesitamos. Para ello utilizaremos una base de datos Access, pero puede seguir las mismas instrucciones con cualquier tipo de base de datos.

1. En el menú **Insertar** haga clic en el comando **Base de datos**.
2. En el asistente para la conexión seleccione el botón de opción *Microsoft Access (ADO)*.
3. Haga clic en **Siguiente** y navegue hasta el archivo **altova.mdb** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**.

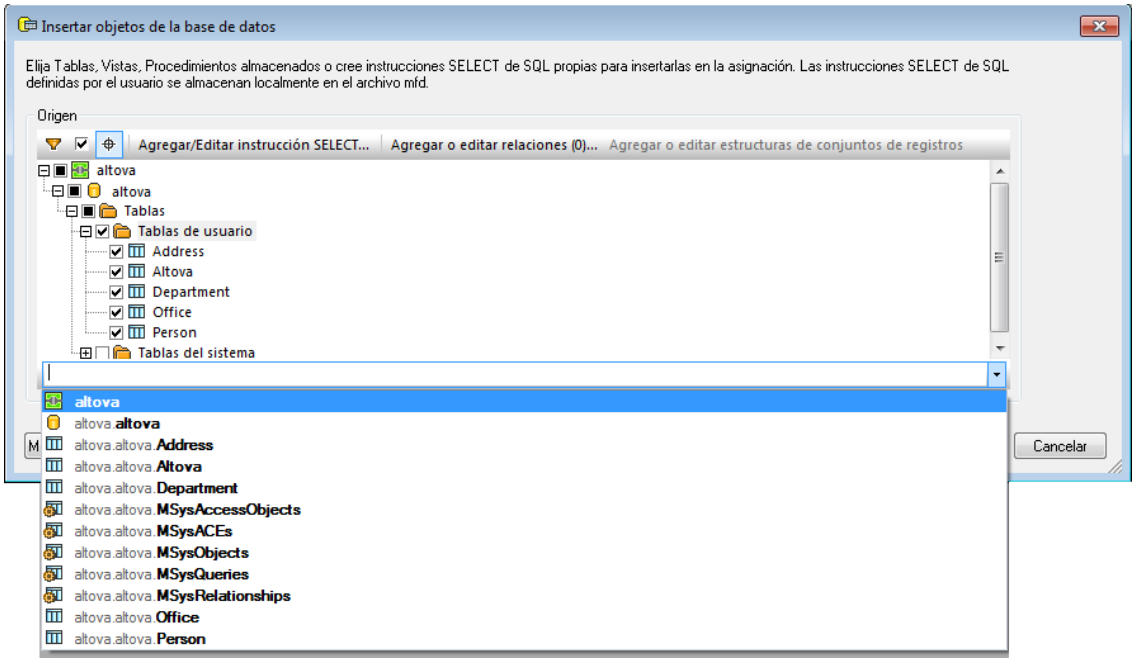
Aparece un cuadro de diálogo donde puede seleccionar qué objetos de la base de datos se incluyen en la asignación.



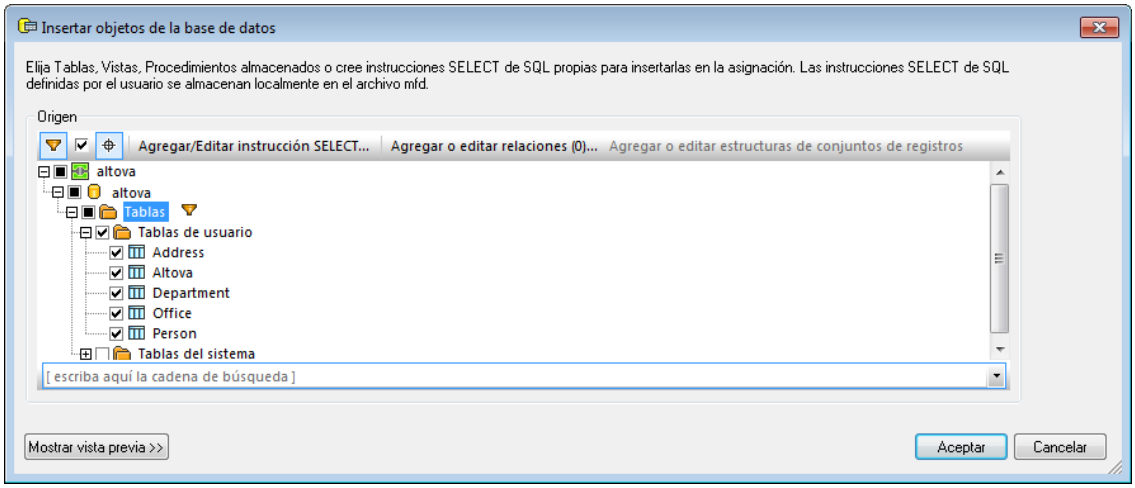
Para incluir un objeto de la base de datos (p. ej. una tabla) en la asignación basta con marcar su casilla. Por ejemplo, vamos a marcar la casilla de `Tablas de usuario`.

El botón **Localizador de objetos**  del cuadro de diálogo permite buscar elementos concretos

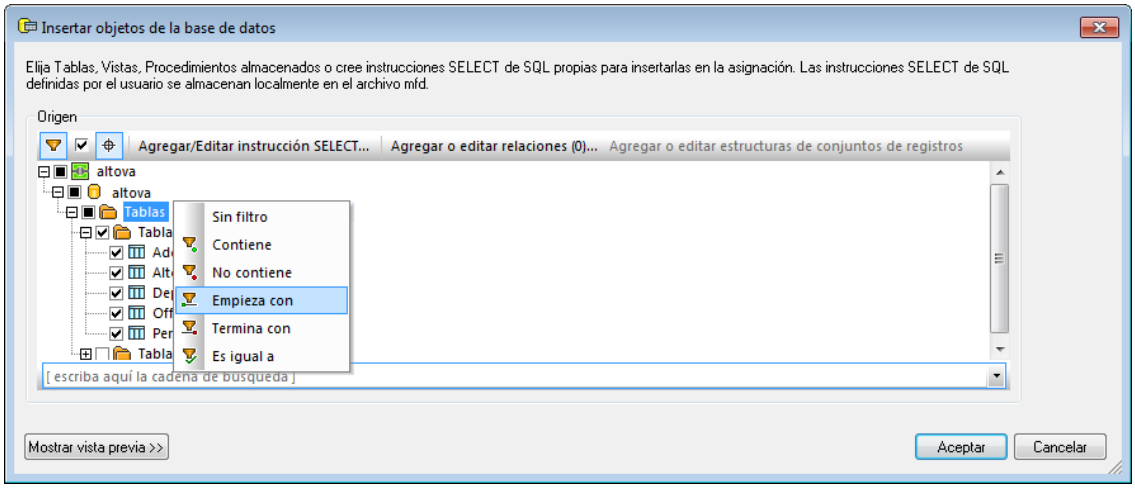
de la base de datos. Seleccione un objeto (o escriba su nombre) en el cuadro combinado que aparece en la parte inferior del cuadro de diálogo.



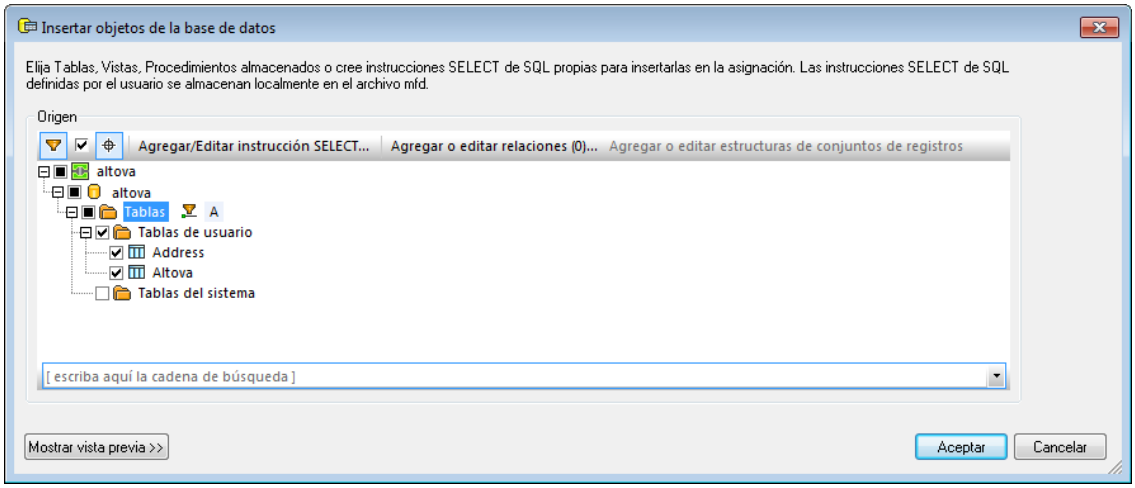
El botón **Filtrar**  sirve para filtrar objetos por nombre. Cuando se hace clic en el botón **Filtrar**, aparece un icono en forma de embudo junto a los objetos que se pueden filtrar (p. ej. `Tablas`).



Haga clic en el icono en forma de embudo para elegir si el nombre del objeto debe empezar o terminar con la cadena de búsqueda o ser igual o contener la cadena de búsqueda.



Ahora puede introducir la cadena de búsqueda junto al filtro (p. ej. "A"):



El botón **Mostrar objetos seleccionados** sirve para ver solamente los elementos cuya casilla esté marcada.

El botón **Agregar/Editar instrucción SELECT** sirve para agregar o editar instrucciones SELECT personales para la base de datos actual. Los datos que devuelvan dichas instrucciones podrán utilizarse como componentes de origen en la asignación (*véase* *Instrucciones SQL SELECT como tablas virtuales*).

El botón **Agregar o editar relaciones** sirve para definir relaciones de clave principal o foránea entre los campos de la base de datos (*véase* *Definir relaciones locales*).

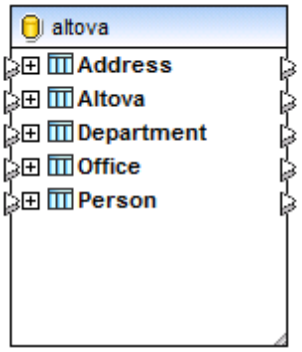
El botón **Agregar o editar estructuras de conjuntos de registros** solamente se habilita si la base de datos admite procedimientos almacenados y si se seleccionó un procedimiento almacenado en el cuadro de diálogo (*véase* *Procedimientos almacenados*).

El botón **Mostrar vista previa** sirve para obtener una vista previa de los datos de la tabla o vista

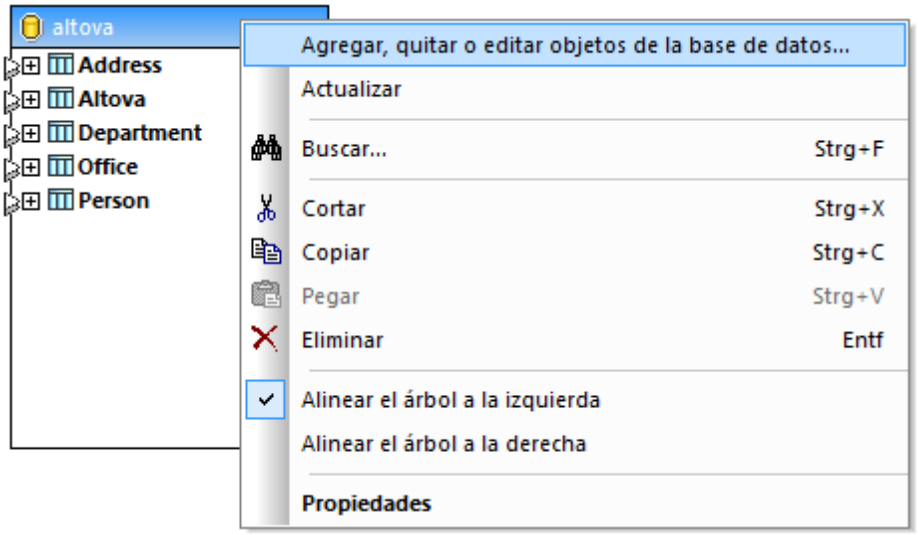
 © 2018 Altova GmbH

que está seleccionada en ese momento. Recuerde que con el Explorador de bases de datos podrá examinar y consultar bases de datos fuera del proceso de asignación (*véase Examinar y consultar bases de datos*).

Cuando esté todo listo para agregar los objetos de la base de datos a la asignación, haga clic en **Aceptar**. En el componente de base de datos solamente aparecerán las tablas, vistas, etc. seleccionadas y podrá dibujar conexiones de asignación entre ellas y los demás componentes de MapForce.



Cuando desee cambiar los objetos de la base de datos que incluye el componente, haga clic con el botón derecho en el componente y seleccione **Agregar, quitar o editar objetos de la base de datos**.



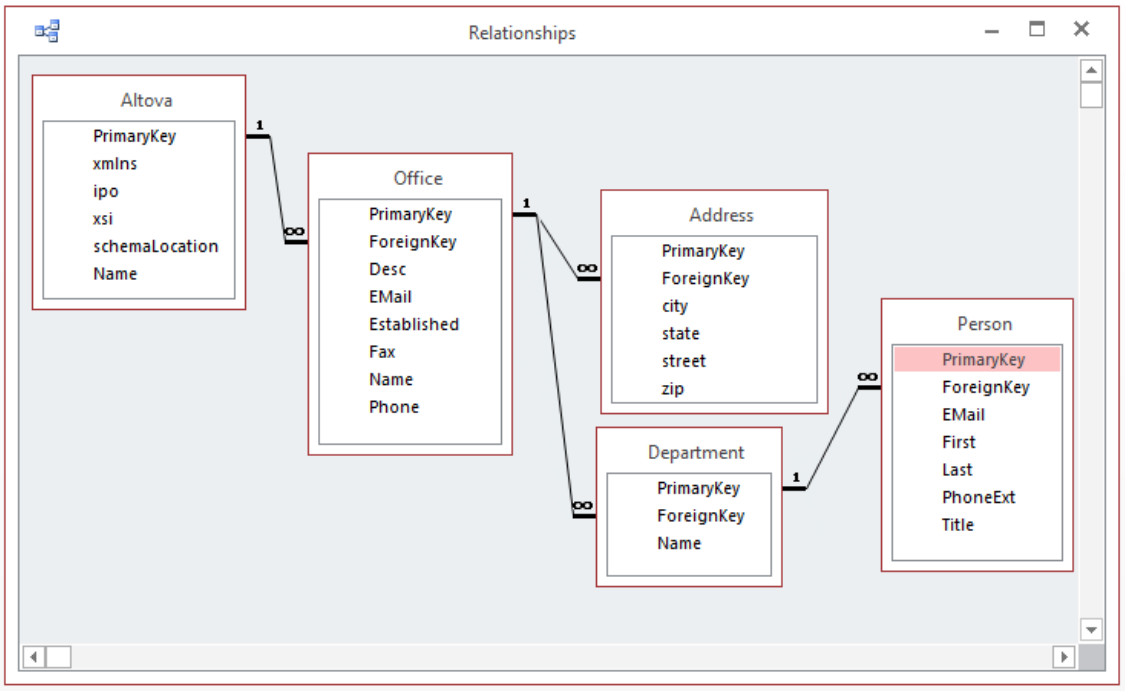
7.2.2.4 Gestionar relaciones de BD

Las bases de datos relacionales, tal y como indica su nombre, suelen tener relaciones definidas entre sus tablas. Por ejemplo, la base de datos **altova.mdb** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial** tiene estas relaciones:

- La compañía del ejemplo (correspondiente a la tabla "Altova") está compuesta como

mínimo por una oficina (p. ej. Brenton y Vereno). En las bases de datos esto se denomina *relación de uno a varios* y este tipo de relación existe entre las tablas "Altova" y "Office". En otras palabras por cada registro **PrimaryKey** de la tabla "Altova" puede haber varios registros **ForeignKey** en la tabla "Office". Por tanto, los registros de "Office" donde el valor de ForeignKey se corresponda con el valor de PrimaryKey de "Altova" se considerarán oficinas de "Altova".

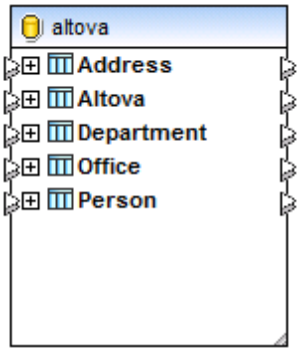
- Cada oficina está compuesta como mínimo por un departamento (p. ej. "Marketing", "IT", "Development"). Una vez más, podemos decir que existe una *relación de uno a varios* entre las tablas "Office" y "Department".
- Por último, cada departamento está compuesto como mínimo por un empleado. Por tanto, existe una *relación de uno a varios* entre las tablas "Department" y "Person".



Relaciones de tablas en la base de datos altova.mdb (vista "Relationships" de Microsoft Access)

Las relaciones entre las tablas de la base de datos son muy importantes en las asignaciones de datos porque MapForce controla dichas relaciones de base de datos cuando la base de datos se añade al diseño de asignación. Esto permite conservar las relaciones mientas se crean las asignaciones de datos. Veamos un ejemplo para comprenderlo mejor: añada la base de datos **altova.mdb** a la asignación (con el comando de menú **Insertar | Base de datos**). A las tablas que aparecen a continuación las llamaremos tablas *raíz*:

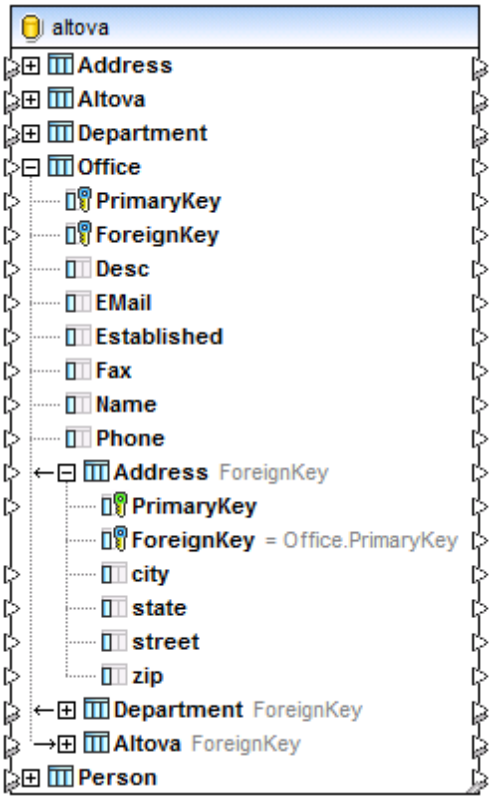
© 2018 Altova GmbH



Tablas raíz

Si expandimos cualquier tabla raíz, aparecen todas las tablas relacionadas. Por ejemplo, si expandimos la tabla **Office**, aparece la jerarquía de tablas relacionadas con esta tabla:

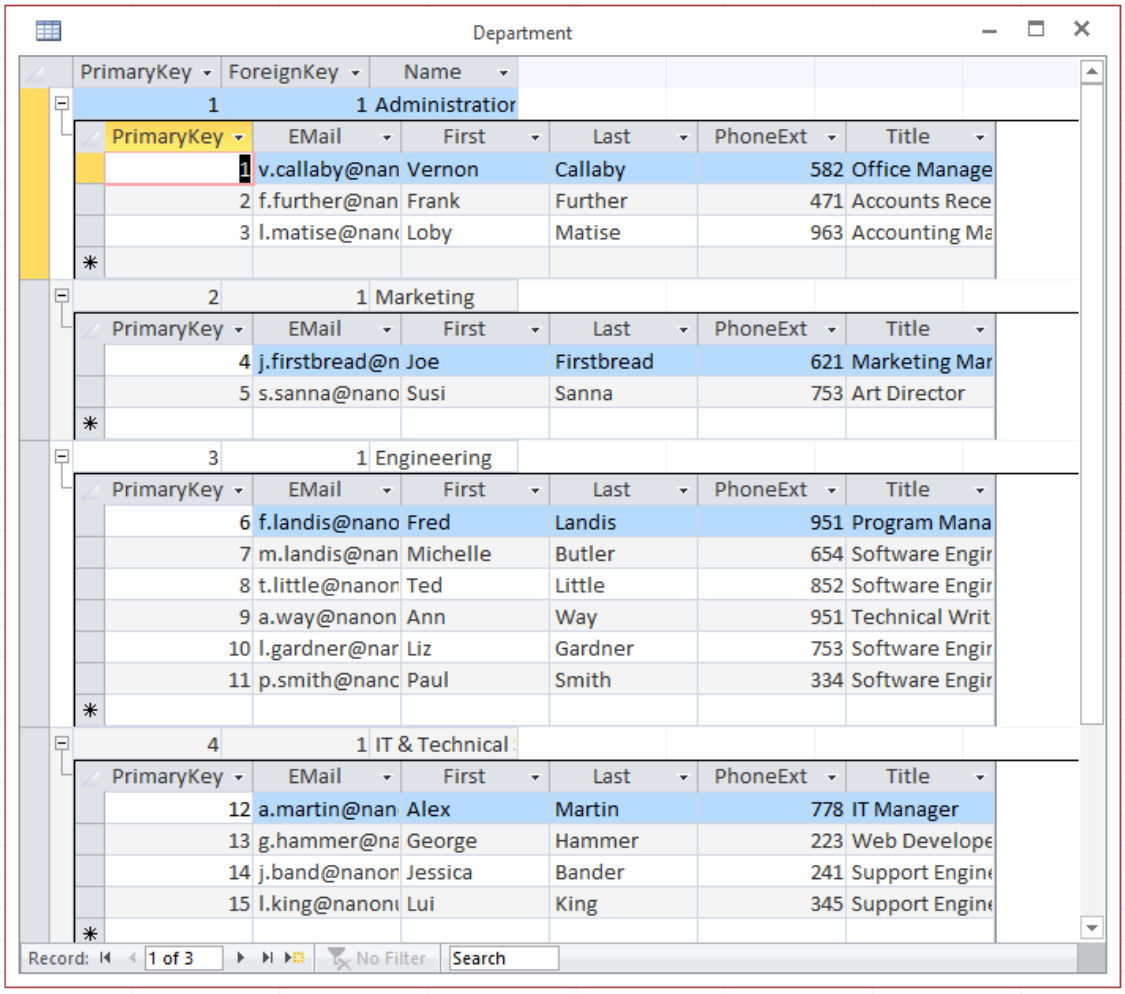
- La flecha ← de una tabla indica que la tabla es una tabla secundaria. Por ejemplo, **Address** es una tabla secundaria de **Office** y **Department** es una tabla secundaria de **Office**, así como una tabla hermana de **Address**, por lo que ambas tienen el mismo nivel de sangría. Como puede ver, la relación en la asignación de datos se corresponde con el diagrama que aparece al principio de este apartado.
- La flecha → indica que la tabla es una tabla primaria. Por ejemplo, **Altova** es la tabla primaria de **Office**.



Relaciones de tabla en MapForce (base de datos altova.mdb)

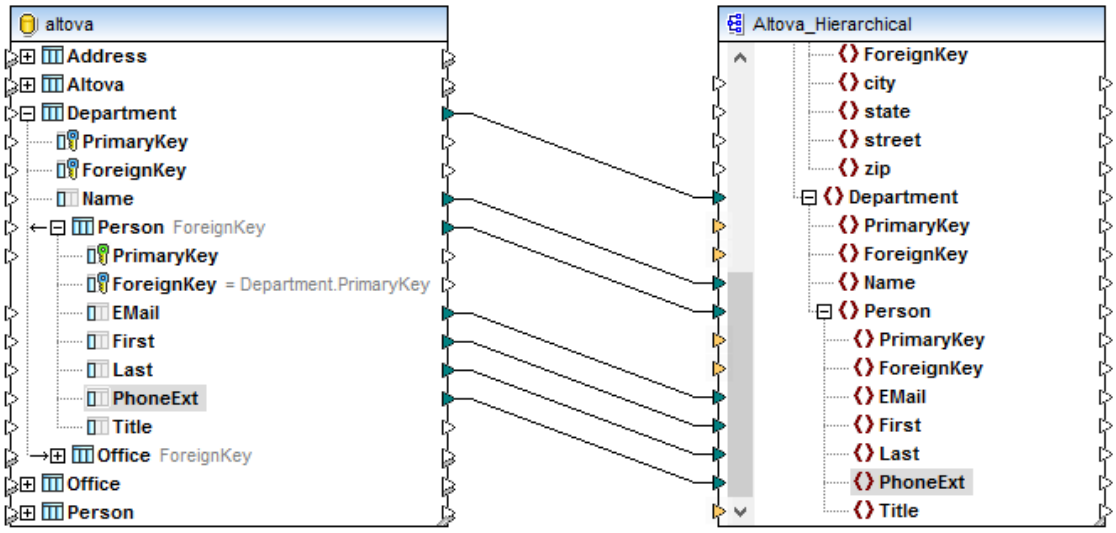
Esta representación jerárquica de las tablas ayuda a conservar las relaciones que existen en la base de datos cuando la asignación lee o escribe datos en una base de datos. Por ejemplo, imaginemos que queremos obtener todos los registros de la tabla **Person** y escribirlos en un archivo XML agrupados por departamento. Concretamente, el archivo XML debería vincular todos los empleados con un departamento (más o menos como en la base de datos **altova.mdb** utilizada en este ejemplo):

© 2018 Altova GmbH



Como muestra la imagen, el departamento "Administration" tiene tres empleados, "Marketing" tiene dos, "Engineering" tiene seis, etc.

Si queremos que cada empleado se asigne al departamento correcto, debemos usar como tabla raíz la tabla **Department** y después crear asignaciones de datos desde la tabla **Person**, que es una tabla secundaria de **Department**.

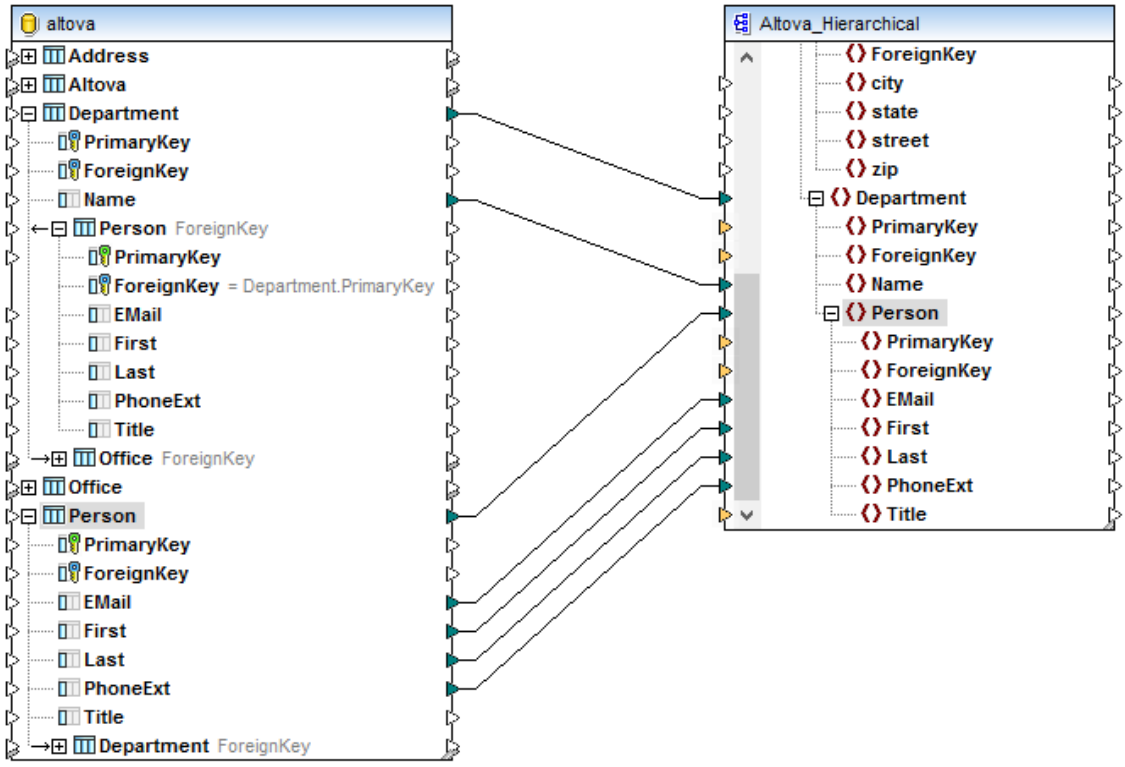


La asignación de la imagen anterior es una versión modificada del archivo **DB_Altova_Hierarchical.mfd** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**. En la vista previa de resultados veríamos que los empleados se agrupan por departamentos tal y como queríamos. Es decir, "Administration" tiene tres empleados, "Marketing" tiene dos, "Engineering" tiene seis, etc.

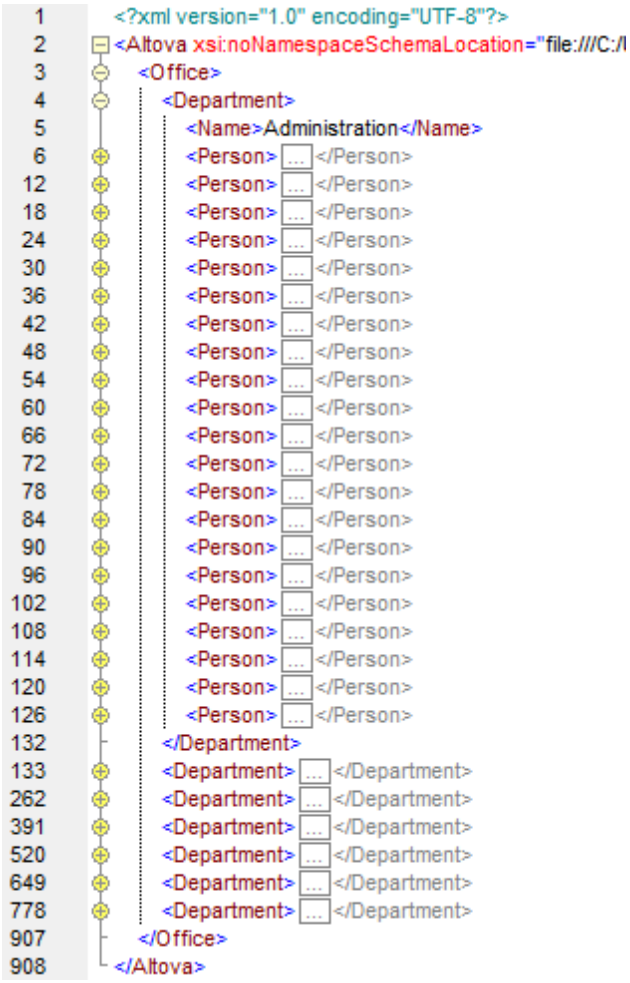
```
<?xml version="1.0" encoding="UTF-8"?>
<Altova xsi:noNamespaceSchemaLocation="file:///C:
  <Office>
    <Department>
      <Name>Administration</Name>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
    </Department>
    <Department>
      <Name>Marketing</Name>
      <Person> ... </Person>
      <Person> ... </Person>
    </Department>
    <Department>
      <Name>Engineering</Name>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
      <Person> ... </Person>
    </Department>
    <Department> ... </Department>
    <Department> ... </Department>
    <Department> ... </Department>
    <Department> ... </Department>
  </Office>
</Altova>
```

Ahora veamos otra versión de la misma asignación en la que las conexiones dibujadas convierten las tablas **Department** y **Person** en tablas raíz.

 © 2018 Altova GmbH



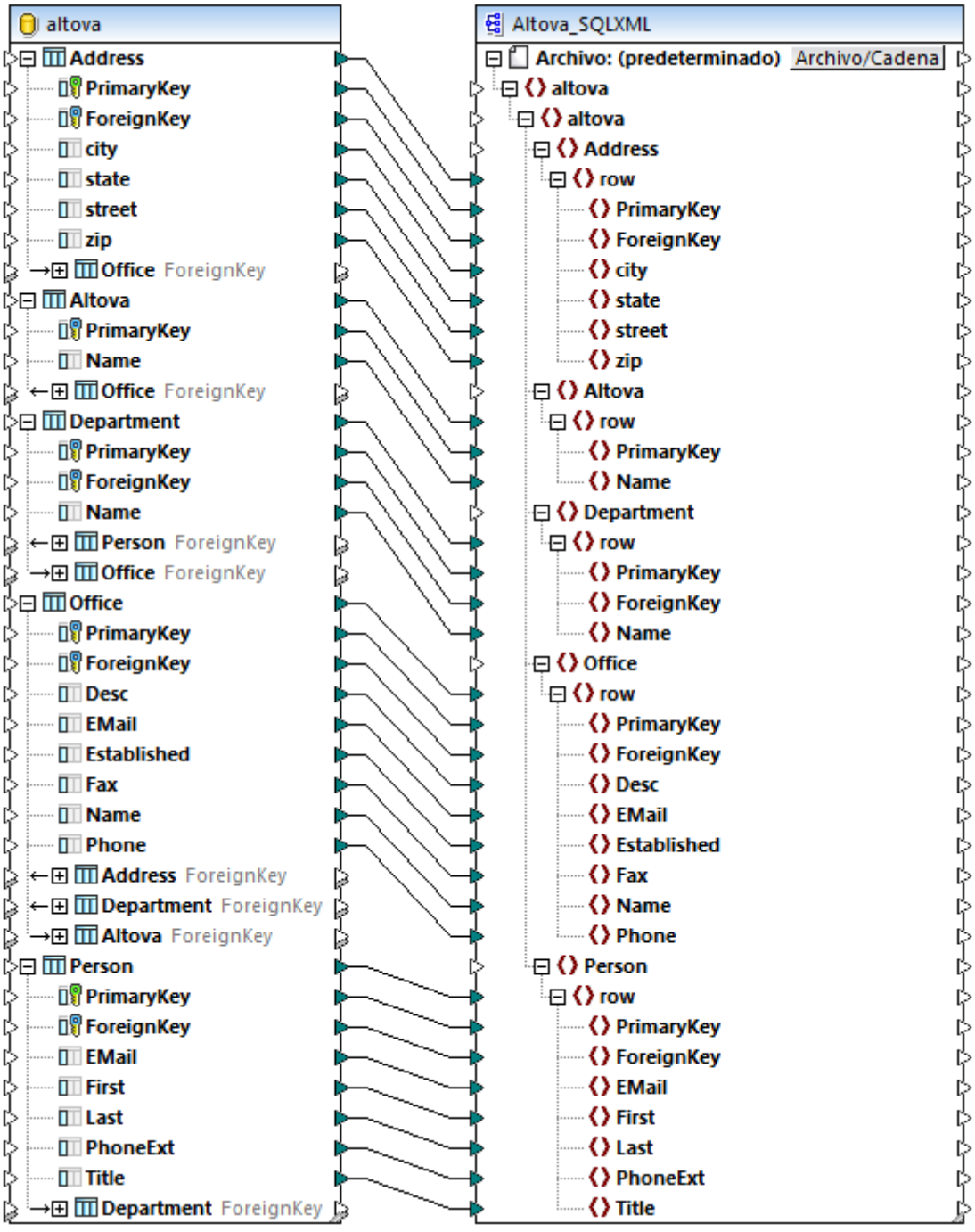
En esta ocasión, en la vista previa de resultados aparecen todos los empleados (sin importar su departamento de origen) en cada departamento de destino. Es decir, "Administration" tiene 21 empleados, "Marketing" tiene 21, "Engineering" tiene 21, etc.



En este segundo ejemplo, debido al modo en que se dibujaron las conexiones, MapForce no tiene en cuenta las relaciones de la base de datos.

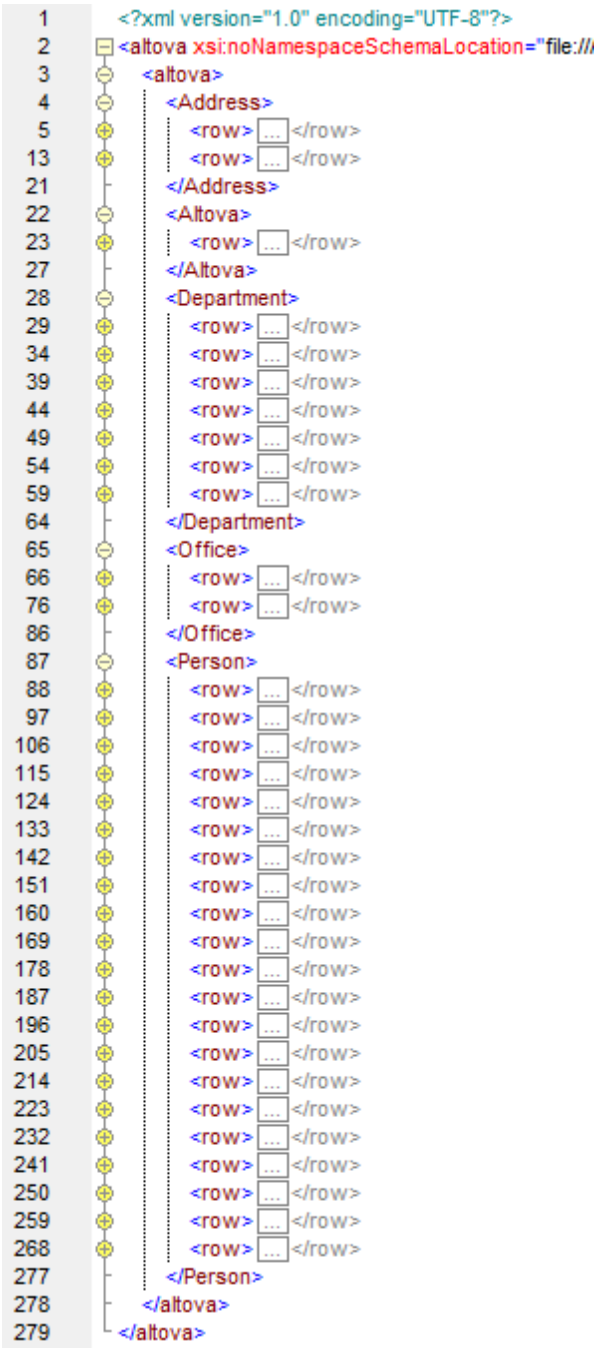
Por tanto, si queremos conservar las relaciones de la base de datos, debemos asegurarnos de dibujar conexiones entre las mismas tablas raíz, pues estas contienen las tablas secundarias cuyas relaciones deseamos conservar, tanto si se trata de la base de datos de origen como de la de destino. Para ver ejemplos de asignaciones de datos que conservan las relaciones de las bases de datos consulte los archivos **DB_Altova_Hierarchical.mfd** y **Altova_Hierarchical_DB.mfd** de la carpeta **<Documentos>\Altova\MapForce2018 \MapForceExamples** (*véase* *Insertar datos en tablas vinculadas*).

También puede darse el caso de que no sea necesario o conveniente conservar las relaciones de la base de datos. Por ejemplo, imaginemos que queremos exportar todos los datos de la base de datos **altova.mdb** a un archivo XML plano conforme a la especificación SQL/XML (parte 14 de la especificación SQL). Este tipo de asignación puede verse en el diseño **DB_Altova_SQLXML.mfd** de la carpeta **<Documentos>\Altova\MapForce2018 \MapForceExamples**. El objetivo de la asignación es obtener los datos de la base de datos en forma de archivo XML plano. El esquema SQL/XML de destino se generó con XMLSpy (con el comando **Conversión | Crear esquema XML a partir de la estructura de la BD**).



DB_Altova_SQLXML.mfd

Como muestra la imagen, a cada tabla de la base de datos le corresponde un elemento en el XML de destino. En la vista previa de resultados podemos ver que las filas de cada tabla de la base de datos se escriben en elementos "row" en el XML de destino.


<?xml version="1.0" encoding="UTF-8"?>
<altova xsi:noNamespaceSchemaLocation="file:///
<altova>
<Address>
<row> ... </row>
<row> ... </row>
</Address>
<Altova>
<row> ... </row>
</Altova>
<Department>
<row> ... </row>
<row> ... </row>
<row> ... </row>
<row> ... </row>
<row> ... </row>
<row> ... </row>
<row> ... </row>
</Department>
<Office>
<row> ... </row>
<row> ... </row>
</Office>
<Person>
<row> ... </row>
<row> ... </row>
<row> ... </row>
<row> ... </row>
<row> ... </row>
<row> ... </row>
<row> ...

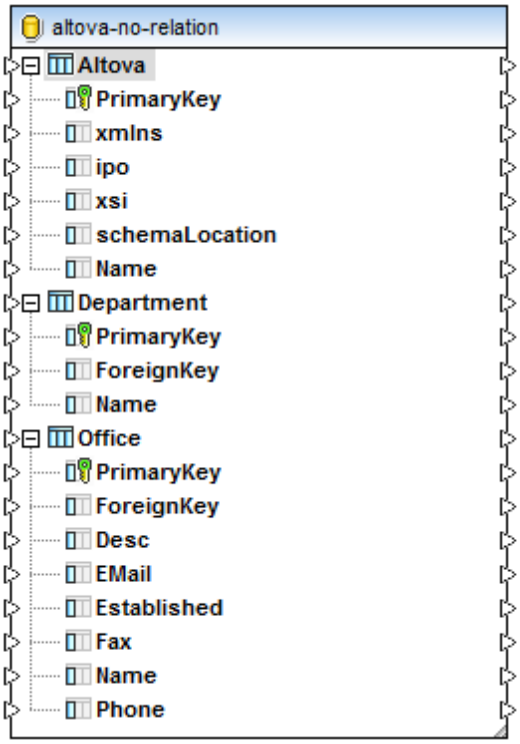
7.2.2.5 Definir relaciones locales

Cuando las tablas de una base de datos no tienen definidas relaciones explícitas, podemos definirlas en MapForce directamente. Concretamente, podemos crear relaciones de clave principal y foránea entre columnas de distintas tablas sin que ello afecte a la base de datos. Como claves principales y foráneas se puede usar cualquier columna de la base de datos. Además, se pueden crear relaciones nuevas además de las que ya existen en la base de datos. Las relaciones definidas localmente desde MapForce se guardan con la asignación.

En MapForce estas relaciones instantáneas se denominan ***relaciones locales***. Podemos definir relaciones locales para estos objetos de base de datos:

- Tablas
- Vistas
- Procedimientos almacenados y funciones
- Instrucciones SELECT definidas por el usuario

La base de datos **altova-no-relation.mdb** utilizada en este ejemplo es una versión simplificada de la base de datos **altova.mdb** que viene con MapForce. Las tablas "Person" y "Address", al igual que todas las relaciones de tabla, se han eliminado en Microsoft Access. Como podemos ver en la imagen siguiente, ninguna tabla del componente **altova-no-relation** tiene tablas secundarias. Es decir, todas están en el mismo nivel raíz. El contenido de cada tabla se limita a los campos que contiene.

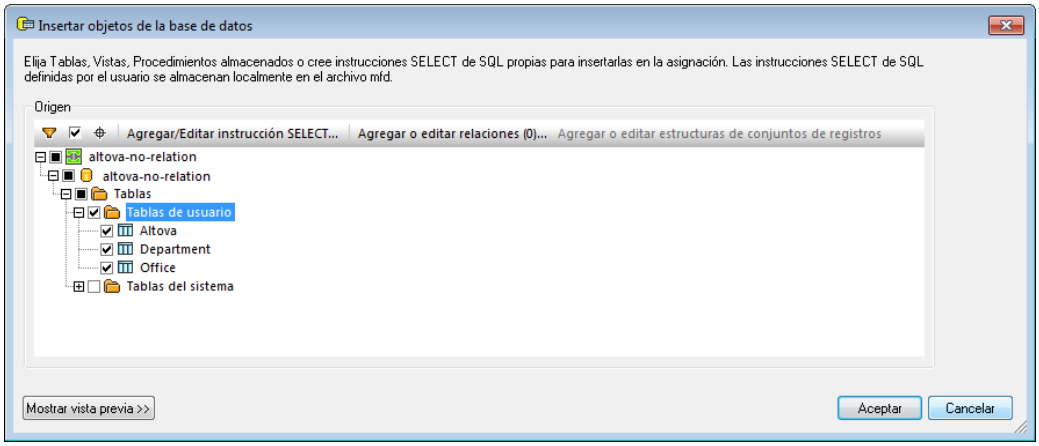


Estructura de base de datos sin relaciones explícitas

Imaginemos que queremos ver las oficinas de "Altova" y los departamentos de cada oficina. Recuerde que en **altova-no-relation.mdb** no existen relaciones de clave principal y foránea explícitas, pero existen de forma lógica (implícitamente), así que las recrearemos localmente en MapForce para alcanzar nuestro objetivo.

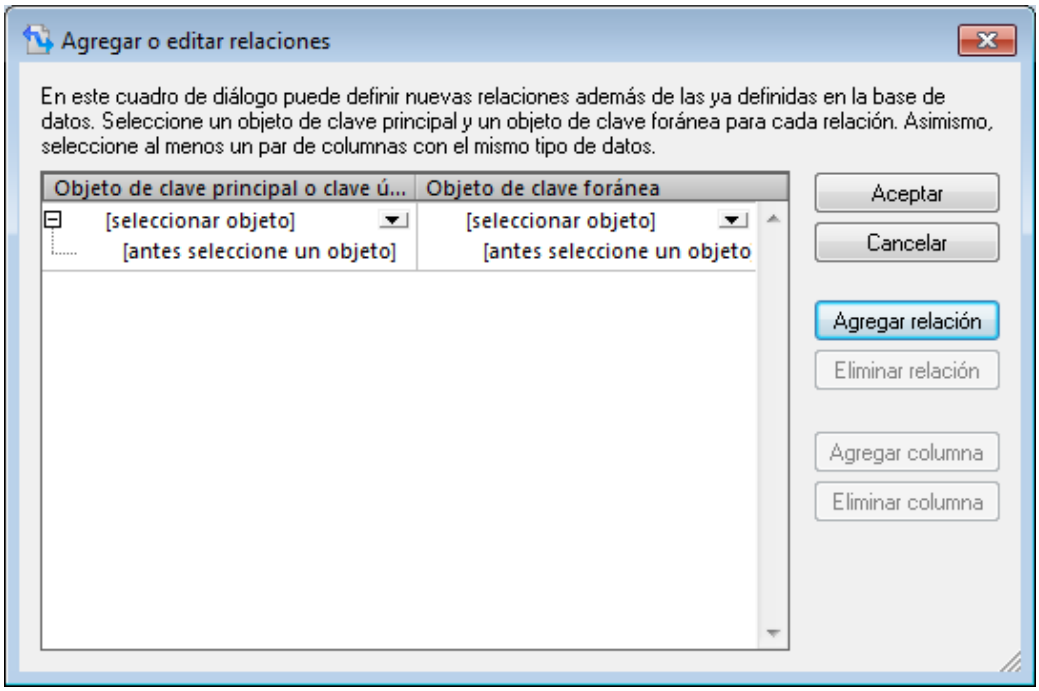
Podemos definir relaciones locales mientras añadimos la base de datos a la asignación o haciendo clic con el botón derecho en un componente de base de datos y eligiendo el comando **Agregar, quitar o editar objetos de la base de datos** del menú contextual:

1. En el menú **Insertar** haga clic en el comando **Base de datos**.
2. En el asistente para la conexión elija la opción *Microsoft Access (ADO)* y haga clic en **Siguiente**.
3. Navegue hasta la base de datos **altova-no-relation.mdb** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial** y haga clic en **Conectarse**.
4. Marque la casilla de `Tablas de usuario`.

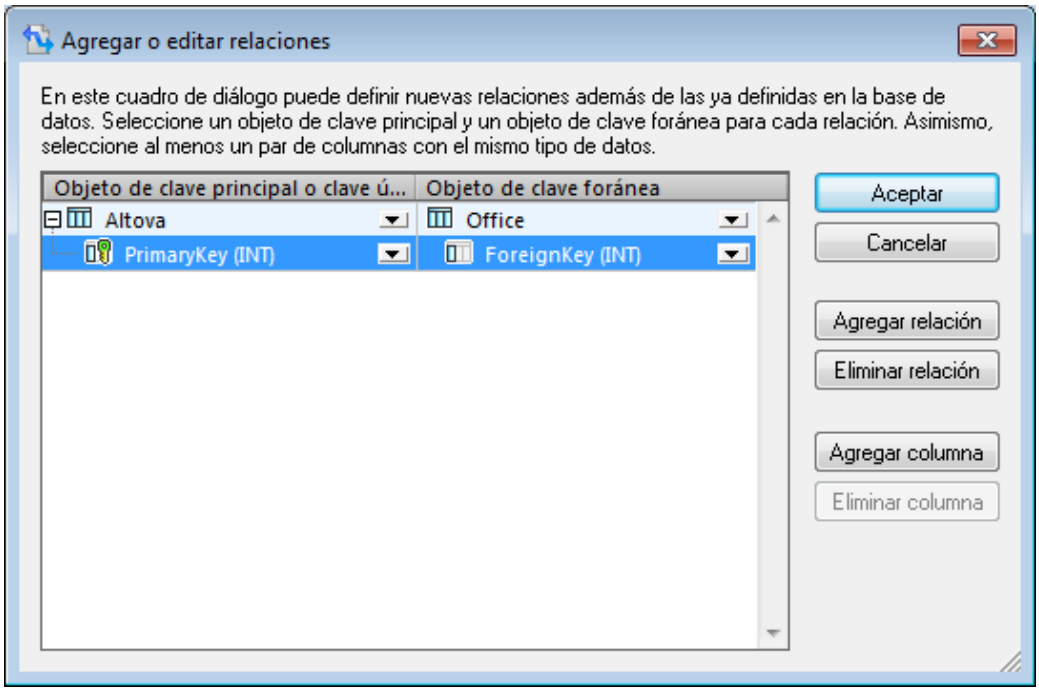


5. Haga clic en el botón **Agregar o editar relaciones** de la barra de herramientas.
6. Aparece el cuadro de diálogo "Agregar o editar relaciones". Haga clic en el botón **Agregar relación**.

© 2018 Altova GmbH

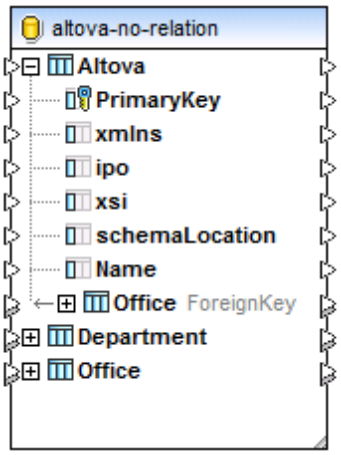


7. Seleccione valores en las dos listas desplegables para crear una relación de clave principal y foránea entre las tablas "Altova" y "Office" (*imagen siguiente*). En las dos listas desplegables puede seleccionar las tablas u objetos de base de datos para los que deben crearse relaciones. La lista de la izquierda especifica el objeto que almacena la clave principal/única, mientras que la derecha especifica el objeto de clave foránea. El objeto de clave principal/única será el objeto primario en MapForce, mientras que el objeto de clave foránea aparecerá como secundario en el componente de base de datos (*véase* Gestionar relaciones de BD).



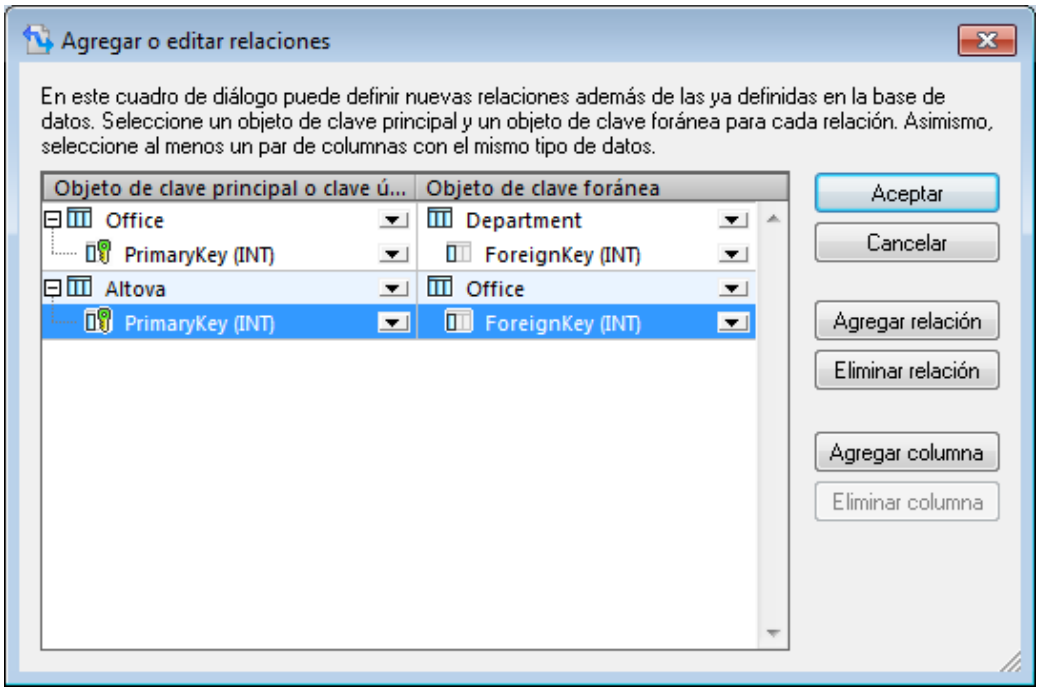
8. Haga clic en **Aceptar** para terminar de crear la definición de relación local y después haga clic en el botón **Insertar** para insertar la base de datos en el área de asignación.

Llegados a este punto, tenemos una relación local entre la columna **PrimaryKey** de la tabla "Altova" y la columna **ForeignKey** de la tabla "Office". Como muestra la imagen siguiente, la tabla raíz "Altova" ahora es la tabla primaria de la tabla "Office". Es decir, la tabla "Office" está relacionada con la tabla "Altova" y aparece con un icono de expansión.



Sin embargo, todavía no hemos terminado de crear relaciones. Todavía debemos crear una relación entre las tablas **Office** y **Department** (*imagen siguiente*).

© 2018 Altova GmbH

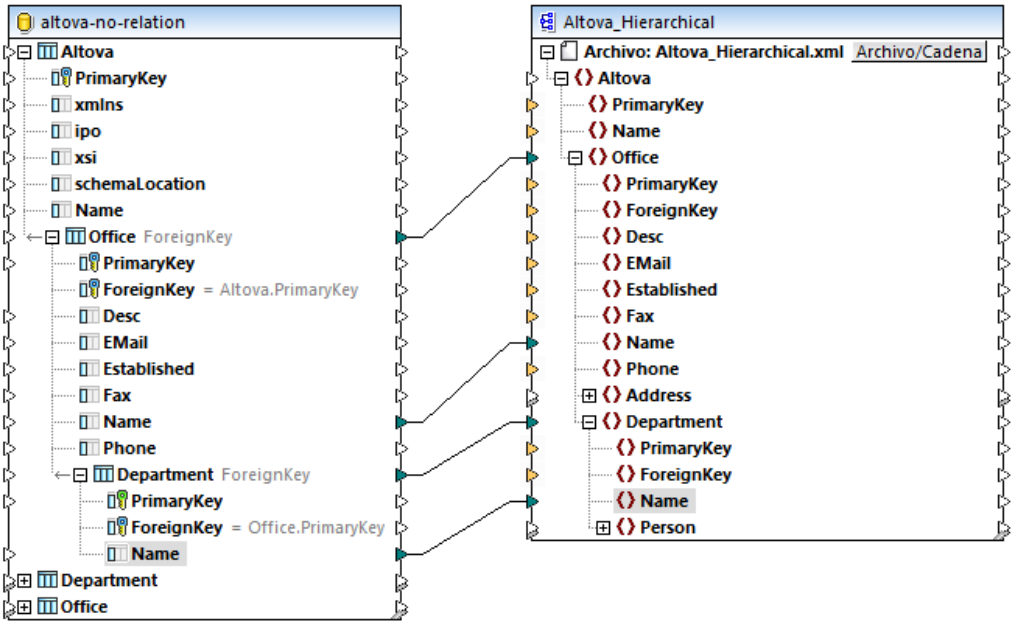


Para volver a abrir el cuadro de diálogo "Agregar o editar relaciones" haga clic con el botón derecho en el componente de base de datos y seleccione **Agregar, quitar o editar objetos de la base de datos** en el menú contextual.

Para terminar solo falta agregar el esquema de destino a la asignación:

1. En el menú **Insertar** haga clic en el comando **Insertar archivo o esquema XML**.
2. Navegue hasta el archivo **Altova_Hierarchical.xsd** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**. Cuando la aplicación solicite un archivo XML, haga clic en **Omitir**. Cuando la aplicación solicite un elemento raíz, seleccione `Altova`.

Observe que, para poder conservar las relaciones que existen entre las tablas en el XML de destino, todas las conexiones se dibujaron desde la misma tabla raíz y jerárquicamente (desde la tabla "Altova". Para más información consulte el apartado Gestionar relaciones de BD.



Si abrimos el panel *Resultados*, veremos que los elementos Department aparecen anidados en la oficina correspondiente, tal y como queríamos.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Altova xsi:noNamespaceSchemaLocation="file:///C:/U
3    <Office>
4      <Name>Nanonull, Inc.</Name>
5      <Department>
6        <Name>Administration</Name>
7      </Department>
8      <Department>
9        <Name>Marketing</Name>
10     </Department>
11     <Department>
12       <Name>Engineering</Name>
13     </Department>
14     <Department>
15       <Name>IT &amp;amp; Technical Support</Name>
16     </Department>
17   </Office>
18   <Office>
19     <Name>Nanonull Partners, Inc.</Name>
20     <Department>
21       <Name>Administration</Name>
22     </Department>
23     <Department>
24       <Name>Marketing</Name>
25     </Department>
26     <Department>
27       <Name>IT &amp;amp; Technical Support</Name>
28     </Department>
29   </Office>
30 </Altova>
```

 © 2018 Altova GmbH

7.2.2.6 Ejecutar asignaciones que modifican BD

Cuando una asignación de datos modifica los datos de una BD (p. ej. insertando, actualizando o eliminando registros), los cambios los aplica el motor que ejecuta la asignación. El motor que ejecuta la asignación puede ser MapForce, MapForce Server (independientemente o bajo la dirección de FlowForce Server) o el entorno de ejecución del código generado para C++, C# o Java.

Cuando se consulta la vista previa de resultados de una asignación en MapForce (en el panel *Resultados*), se obtiene un script de actualización que incluye las instrucciones SQL que se aplicarán en la base de datos. El script no se llega a ejecutar en la base de datos. Se trata de una simple vista previa.

```
/*
Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar y no se pueden ejecutar en otra herramienta de consultas SQL.
Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
Conéctese a la base de datos usando esta cadena de conexión:
Data Source=C:\Users\Documents\Altova\MapForce2018\MapForceExamples\AltovaTarget.mdb;Provider=Microsoft.Jet.OLEDB.4.0
*/

-- begin transaction

DELETE FROM [Person]

DELETE FROM [Department]

DELETE FROM [Address]

DELETE FROM [Office]

DELETE FROM [Altova]

SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Altova]
-->>> %PrimaryKey1%

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Organization Chart', '%PrimaryKey1%')

SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Office]
-->>> %PrimaryKey2%
```

Vista previa de resultados de una asignación que modifica una base de datos (Altova_Hierarchical_DB.mfd)

Tenga en cuenta que este script solamente se puede aplicar manualmente a la base de datos con los motores de ejecución mencionados anteriormente. El script puede contener formatos de valor desconocidos para editores SQL externos. Además, si se definen varias acciones en una tabla (p. ej. "Actualizar si... Insertar el resto"), en la vista previa solo aparecerá la primera acción porque la segunda acción se ejecuta de forma condicional.

Si quiere aplicar los cambios de la asignación a la base de datos desde MapForce directamente, haga clic en el comando **Ejecutar script SQL** del menú **Resultados**. Recuerde que esta acción modificará realmente la base de datos inmediatamente.

Cuando la asignación se ejecuta con MapForce Server (independientemente o bajo la dirección de FlowForce Server), los cambios se aplican inmediatamente en la base de datos. Los mismo ocurre en el código generado: los cambios de la base de datos se aplican cuando se compila y ejecuta el código (p. ej. con el comando **Ejecutar** de Visual Studio).

Su instalación de MapForce incluye varios archivos de base de datos de ejemplo (Microsoft Access o SQLite), que están disponibles en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**. No se recomienda aplicar cambios en las bases de datos de ejemplo desde MapForce con el comando **Ejecutar script SQL** porque puede que

tras aplicar los cambios estas bases de datos sean inservibles. Lo más recomendable es crear una copia de seguridad de la carpeta **<Documentos>\Altova\MapForce2018 \MapForceExamples** antes de actualizar ningún archivo de esta carpeta.

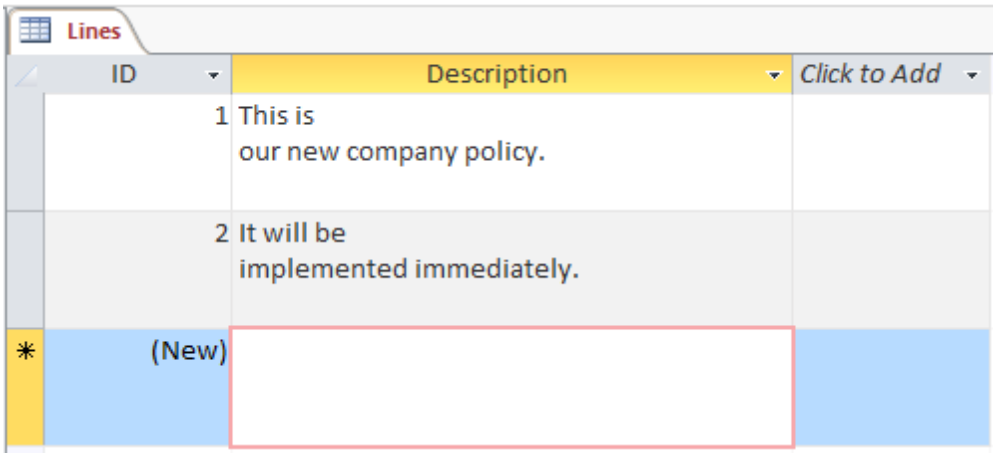
En estos apartados de la documentación encontrará información sobre la ejecución de asignaciones de datos en entornos de ejecución externos a MapForce:

- Implementar asignaciones en FlowForce Server
- Compilar asignaciones en archivos de ejecución de MapForce Server
- Generador de código

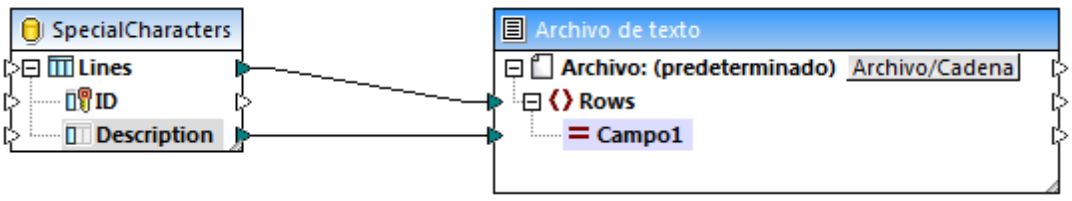
7.2.2.7 Reemplazar caracteres especiales

Cuando quiera transformar datos de una base de datos, primero puede ser necesario quitar ciertos caracteres especiales (como los caracteres de retorno de carro o de salto de línea) de la fuente de datos. Esto se puede hacer con la función **`char-from-code`** de MapForce.

Imagine que tiene una base de datos Microsoft Access compuesta por una tabla llamada "Lines" que tiene dos columnas: "ID" y "Description".



El objetivo es extraer cada descripción a un archivo CSV (donde hay una descripción por línea). La asignación de datos que necesitamos para conseguir este objetivo es:



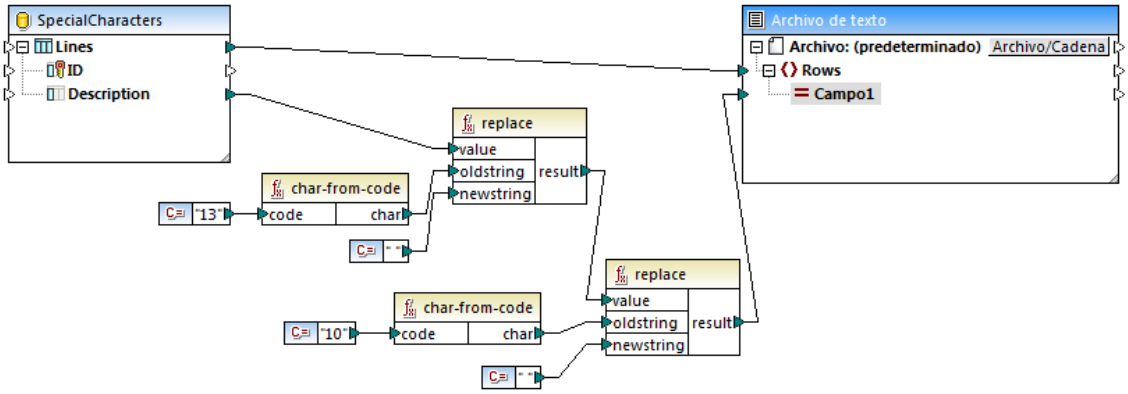
Sin embargo, como cada fila "Description" de Access contiene varias líneas separadas por caracteres CR/LF, el resultado de la asignación incluye también saltos de línea y este no es el resultado que queremos.

 © 2018 Altova GmbH

```
"This is
our new company policy."
"It will be
implemented immediately."

```

Para resolver este problema añadiremos las funciones **char-from-code** y **replace** de la biblioteca de MapForce (*véase* Trabajar con funciones). Cada descripción debe procesarse de modo que, cuando se detecten caracteres especiales, éstos se reemplazarán con un espacio.

En el gráfico Unicode (http://www.unicode.org/charts/), los caracteres LF y CR corresponden a los caracteres **hex 0A | dec 10** y **hex 0D | dec 13** respectivamente. Por tanto, la asignación debe modificarse para convertir los valores Unicode decimales 13 y 10 en una cadena para que la función **replace** los pueda procesar.



En la vista previa de resultados de esta asignación podemos ver que los caracteres CR/LF de cada campo de la base de datos se reemplazaron con un espacio.

```
This is  our new company policy.
It will be  implemented immediately.

```

7.2.2.8 Gestionar valores null

Para comprobar en tiempo de ejecución de la asignación si un campo de la base de datos es NULL basta con usar las funciones **is-null** y **is-not-null** de la biblioteca de MapForce. Para ver en MapForce si una tabla tiene campos NULL basta con consultarla en el Explorador de bases de datos (*véase* Examinar y consultar bases de datos).

Para establecer un campo de base de datos como NULL basta con usar la función **set-null**.

Para reemplazar valores de base de datos NULL con una cadena podemos usar la función **substitute-null** (puede ver un ejemplo de uso en el archivo **DB_ApplicationList.mfd** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**).

Para aprender a trabajar con comparaciones de valores NULL en asignaciones que actualizan bases de datos consulte Gestionar valores null en acciones de tabla de BD.

Para obtener información sobre cómo gestionar valores NULL en asignaciones entre bases de datos y documentos XML consulte Valores Nil y Nillable.

7.2.2.9 Generar valores secuenciales y únicos

Cuando se insertan datos o se actualiza una base de datos, puede ser necesario crear valores secuenciales o únicos de forma instantánea para los campos de base de datos que no reciben datos de ningún componente de origen. En este tipo de casos podemos usar estas funciones de la biblioteca integrada de MapForce:

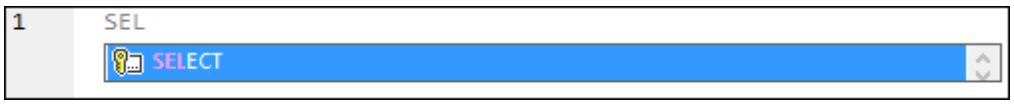
- `auto-number` (de la biblioteca **core | generator functions**): suele utilizarse para generar valores de clave principal para un campo numérico.
- `create-guid` (de la biblioteca **lang | generator functions**): esta función crea un identificador único global (como una cadena con codificación hexadecimal) para el campo especificado.

Tenga en cuenta que los valores para campos de base de datos se pueden escribir usando valores generados con la base de datos. Esta opción está disponible en el cuadro de diálogo "Acciones de tabla de la base de datos" (*véase Configurar acciones de tabla de BD*) y puede ser muy práctica a la hora de generar claves principales.

7.2.2.10 Sugerencias de finalización automática SQL

Cuando se teclean instrucciones SQL en determinados contextos, MapForce puede sugerir entradas de texto automáticamente. La finalización automática está disponible en estos contextos:

- En el editor SQL (*véase Examinar y consultar bases de datos*).
- En el cuadro de texto *Instrucción SQL personalizada* del cuadro de diálogo "Acciones de tabla de la base de datos" (*véase Configurar acciones de tabla de BD*).
- En el cuadro de diálogo "Introducir una instrucción SQL SELECT" (*véase Crear instrucciones SELECT*).



Finalización automática

Con las teclas **Arriba** y **Abajo** del teclado podrá navegar por la lista de sugerencias de la finalización automática. Para usar una entrada haga clic en ella o pulse **Entrar**.

Para deshabilitar las sugerencias de finalización automática:

1. En el menú **Herramientas** haga clic en el comando **Opciones** (o pulse **Ctrl+Alt+O**).
2. Dentro de **Base de datos** haga clic en **Editor SQL**.

 © 2018 Altova GmbH

3. En el grupo de opciones *Ayudantes de entrada* desactive la casilla *Abrir automáticamente*.

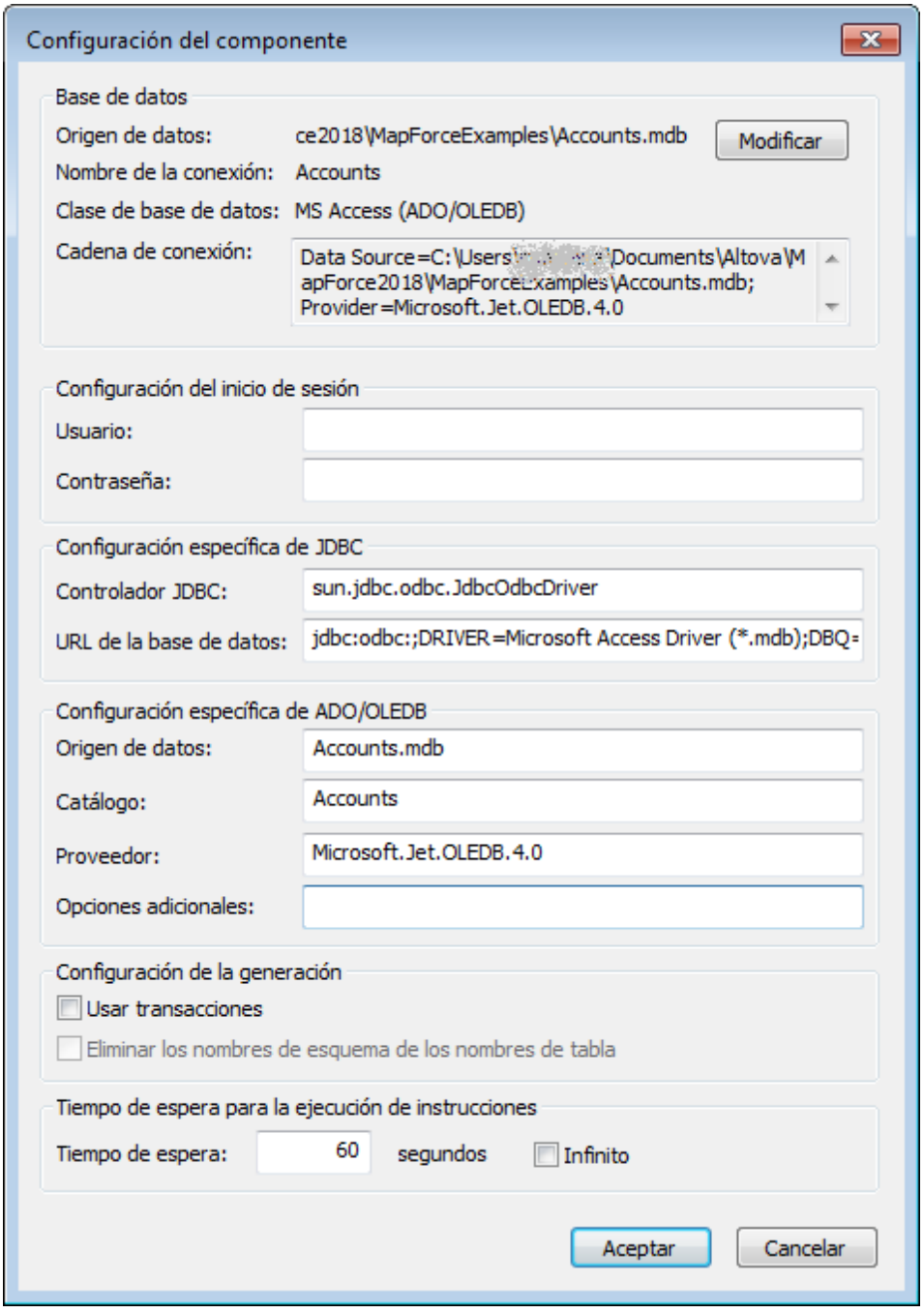
Para invocar las sugerencias de finalización automática a mano:

- Pulse **Ctrl+Espacio**.

7.2.2.11 Configuración de componentes de BD

Tras agregar un componente de base de datos a la asignación, podrá editar su configuración desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de varias maneras:

- Seleccionando el componente y después ejecutando el comando **Propiedades** del menú **Componentes**.
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el componente y eligiendo **Propiedades** en el menú contextuales.



Cuadro de diálogo "Configuración del componente" de un componente de base de datos

Estas son las opciones que se pueden configurar en este cuadro de diálogo.

Base de datos

En este panel de opciones aparece información de conexión de la base de datos. Haga clic en **Modificar** para seleccionar una base de datos distinta o para volver a definir los objetos actuales

© 2018 Altova GmbH

del componente de base de datos. Tras la modificación se conservarán los conectores de las tablas que tengan el mismo nombre. También pueden modificarse las tablas del componente (haciendo clic con el botón derecho en el componente y seleccionando **Agregar, quitar o editar objetos de la base de datos** del menú contextual).

Origen de datos	Especifica el nombre del origen de datos actual. Si se trata de una base de datos basada en archivos, puede ser una ruta de acceso del sistema de archivos. Este campo indica si la base de datos basada en archivos se agregó a la asignación usando una ruta de acceso absoluta o relativa. Si se usó una ruta relativa, este campo contiene una ruta de acceso. Si se usó una ruta absoluta, el campo contiene el nombre de archivo de la base de datos.
Nombre de la conexión	Especifica el nombre de la conexión (si la base de datos usa un origen de datos, el nombre de la conexión es el mismo que el nombre del origen de datos).
Clase de base de datos	Especifica el tipo de base de datos.
Cadena de conexión	Muestra la cadena de conexión de la base de datos actual. Es un campo de solo lectura que se genera a partir de la información indicada durante la fase de creación o modificación de la conexión.

Configuración del inicio de sesión

Estas opciones se usan para los destinos de la generación de código y para el motor de ejecución integrado.

Usuario	Permite cambiar el nombre de usuario que se utiliza para conectarse a la base de datos. Es obligatorio si la base de datos exige un nombre de usuario.
Contraseña	Permite cambiar la contraseña que se utiliza para conectarse a la base de datos. Es obligatorio si la base de datos exige una contraseña.

Configuración específica de JDBC

Estas opciones sirven para conectarse a la base de datos si la asignación contiene una conexión JDBC y se ejecuta con el código Java generado o con MapForce Server.

Nota: las conexiones ADO, ADO.NET y ODBC se convierten en JDBC (y, por tanto, se aplican estas opciones de configuración) cuando la asignación se ejecuta en un equipo Linux o OS X / macOS (*véase* *Bases de datos en MapForce*).

Controlador JDBC	Muestra el controlador que está activo para el componente

	de base de datos. Cuando se define un componente de base de datos se introduce automáticamente el controlador predeterminado, pero el valor de este campo se puede modificar si es necesario. Asegúrese de que la sintaxis del campo *URL de la base de datos* concuerda con el controlador elegido.
URL de la base de datos	URL de la base de datos que está seleccionada. Asegúrese de que su sintaxis concuerda con la del controlador JDBC (el controlador que aparece en el campo *Controlador JDBC*).

Configuración específica de ADO/OLEDB

Estas opciones sirven para conectarse a la base de datos si la asignación contiene una conexión ADO y se ejecuta con código C# o C++ o con MapForce Server desde Windows (*véase Bases de datos en MapForce*). Las opciones *Origen de datos* y *Catálogo* no se utilizan con el motor de ejecución integrado.

Origen de datos	Muestra el nombre del origen de datos ADO.
Catálogo	Muestra el nombre del catálogo ADO.
Proveedor	Muestra el proveedor que está activo para el componente de base de datos.
Opciones adicionales	Muestra opciones adicionales de la base de datos.

Configuración de la generación

Estas opciones son relevantes para todos los destinos de generación de código, así como para el motor de ejecución integrado.

Usar transacciones	Habilita el procesamiento de transacciones cuando se usa una base de datos como destino. Cuando se detecta un error aparece un cuadro de diálogo donde el usuario puede elegir cómo se procede. Si se marca esta casilla, el procesamiento de transacciones se habilita para todas las tablas del componente de base de datos. Para más información consulte Revertir transacciones.
Eliminar los nombres de esquema de los nombres de tabla	Permite eliminar los nombres de esquema de base de datos en el código generado y conservar solamente los nombres de tabla, lo cual permite una mayor flexibilidad. Recuerde que esta opción solamente funciona con las instrucciones SQL Select generadas con MapForce. Las instrucciones SQL definidas por el usuario, mientras se crean tablas virtuales, no se modificarán.

© 2018 Altova GmbH

Tiempo de espera para la ejecución de instrucciones

Cuando se usa una base de datos como componente de destino, pueden superarse los tiempos de espera de ejecución debido a problemas de disponibilidad del servidor o de tráfico, a disparadores de larga ejecución, etc. Estas opciones permiten definir el tiempo de espera máximo antes de que se cierre la conexión. La configuración definida en este grupo de opciones tienen efecto durante las consultas de datos de la base de datos y en el código C#, Java y C++ generado.

Tiempo de espera	Define el plazo de tiempo máximo (en segundos) que debe esperar el motor de ejecución hasta recibir una respuesta de la base de datos antes de anular la ejecución de la instrucción. El valor predeterminado es 60 segundos.
Infinito	Si se marca esta casilla, el motor de ejecución no agotará el tiempo de espera nunca.

Nota: el tiempo de espera para la ejecución de instrucciones no afecta a las bases de datos SQLite.

7.2.2.11.1 Asignar datos a bases de datos

Esta sección incluye instrucciones y ejemplos para transferir datos desde cualquier origen de datos compatible con MapForce (p. ej. archivos XML) a una base de datos de destino. A continuación puede ver un resumen de todas las opciones disponibles.

Si quiere...	Consulte este tema...
Insertar datos en una tabla de base de datos de destino basada en datos suministrados por la asignación de datos...	• Insertar datos en una tabla
Controlar cómo se crean los valores de clave principal...	• Insertar datos en una tabla Table • Insertar datos en varias tablas vinculadas
Ejecutar una instrucción SQL preliminar que se debe ejecutar antes de que la asignación modifique una tabla (p. ej. eliminar todos los registros de la tabla o una instrucción SQL personalizada)...	• Insertar datos en varias tablas vinculadas • Configurar acciones de tabla de BD
Conservar la relación jerárquica existente entre los registros de tablas vinculadas por claves foráneas...	• Insertar datos en varias tablas vinculadas
Actualizar una tabla de forma condicional...	• Actualizar una tabla
Combinar registros en una tabla de BD (actualizar algunos registros e insertar otros en la misma tabla) basándose en una condición...	• Acción "Actualizar si... insertar restantes" • Instrucciones de combinación MERGE

Si quiere...	Consulte este tema...
Preservar la integridad de la BD cuando se actualizan tablas que están vinculadas a otras tablas a través de relaciones de clave foránea...	• Opciones para tablas secundarias cuando se actualiza la tabla principal
Definir varias acciones en la misma tabla (p. ej. eliminar un registro si se cumple una condición o, de lo contrario, insertar un registro nuevo)...	• Acción "Eliminar si..." • Acción "Omitir si..."
Preservar la integridad de los datos en caso de que falle la ejecución de la asignación...	• Revertir transacciones
Insertar varios registros en una tabla de BD de forma masiva (combinando varias instrucciones INSERT en una sola consulta)...	• Inserciones masivas (con MapForce Server)
Evitar resultados no deseados cuando se asignen datos a tablas de BD de destino que contengan valores null...	• Gestionar valores null en acciones de tabla de BD
Ver un ejemplo de cómo crear una asignación que actualice una base de datos...	• Ejemplo: asignar datos XML a SQLite

7.2.2.11.1.1 Insertar datos en una tabla

Una asignación de datos puede insertar datos en una tabla de BD desde cualquier tipo de componente de origen compatible con MapForce, incluidas otras bases de datos. Además puede configurar cómo se crea la clave principal de registros nuevos recién insertados. Por ejemplo, la clave principal se puede tomar de la asignación, generada por la base de datos, o se puede calcular a partir de valores de clave ya existentes en la tabla de BD.

A continuación explicamos cómo insertar registros nuevos en una tabla de BD desde un archivo XML. También explicamos cómo se configura la generación de claves principales. Para ello nos serviremos de estos archivos:

- **altova-cmpy.xml**: que contiene los datos de origen que se deben insertar en la base de datos.
- **Altova_Hierarchical.xsd**: el esquema utilizado para validar el archivo de instancia anterior.
- **altova.mdb**: la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

> La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en

© 2018 Altova GmbH

este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos.

El objetivo de la asignación de datos que nos ocupa es insertar compañías que se encuentran en el archivo **Altova-cmpy.xml** como registros nuevos en la tabla "Altova" dela base de datos **altova.mdb**. Si abre el archivo XML de origen, observará que solamente contiene una compañía llamada "Microtech OrgChart". Por tanto, la asignación de datos debe insertar un registro nuevo en la tabla "Altova" que se llamará "Microtech OrgChart". Además será necesario generar su clave principal.

Para conseguir este objetivo debemos seguir estos pasos:

Paso nº1: insertar el componente XML de origen

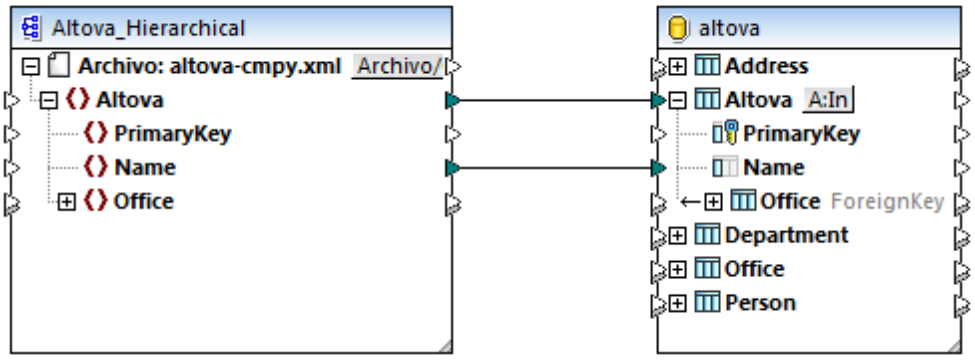
- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el esquema **Altova_Hierarchical.xsd**. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo **Altova-cmpy.xml**.

Paso nº2: insertar la base de datos de destino

- En el menú **Insertar** haga clic en el comando **Base de datos** y siga los pasos del asistente para conectarse a **altova.mdb** (*véase Ejemplo: agregar la base de datos "altova.mdb" a la asignación*).

Paso nº3: dibujar las conexiones

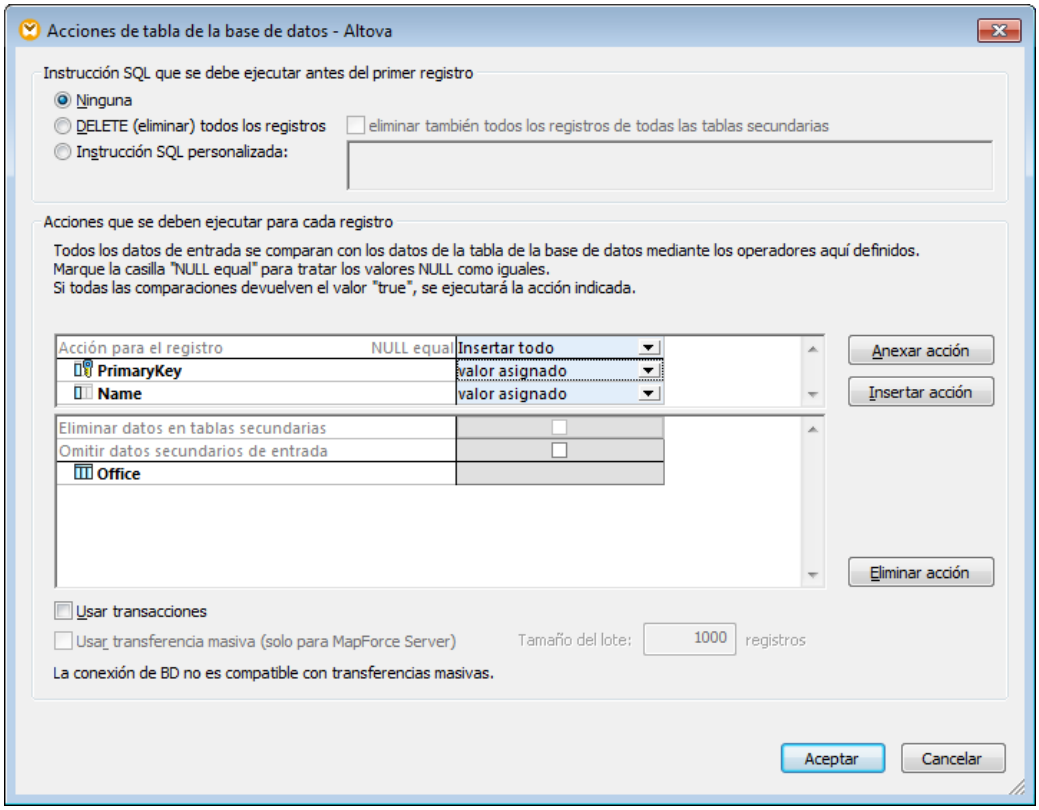
- Dibuje las conexiones de asignación de datos que aparecen a continuación.



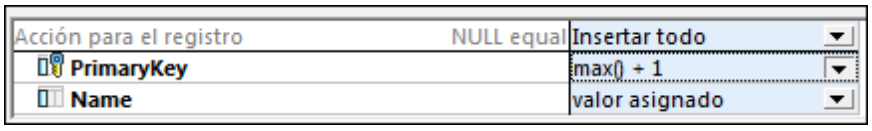
Nota: si se dibujan conexiones no deseadas automáticamente para los elementos descendientes, esto significa que está activa la opción *Conectar automáticamente los secundarios equivalentes*. Si esto as así, para deshacer la última acción, debe seleccionar el comando Edición **| Deshacer**. Para deshabilitar la opción de conexión automática de secundarios, haga clic en el comando **Conexión | Conectar automáticamente los secundarios equivalentes**.

Paso nº4: configurar la acción Insertar

1. Observe que el componente de destino tiene un botón **Acción: Insertar** (A:In). Este botón aparece junto a cada tabla que tiene una conexión con el componente de origen (en este caso, la tabla "Altova"). Haga clic en este botón para configurar más detalladamente la acción de base de datos que se debe ejecutar (en este caso, la acción de inserción). Aparece el cuadro de diálogo "Acciones de tabla de la base de datos".



2. En el cuadro de diálogo, debajo del campo *Insertar todo* y junto a **`PrimaryKey`**, seleccione la opción **max() + 1**.



Las opciones que ofrecen estas listas desplegables son:

Opción	Descripción
valor asignado	Permite asignar los datos de origen al campo de base de datos directamente y se trata de la opción estándar para todos los campos de BD. También se puede usar un procedimiento almacenado para suministrar un valor de clave definiendo una relación (*véase* *Usar procedimientos*

© 2018 Altova GmbH

	almacenados para generar claves principales).
max() + 1	Genera los valores de clave a partir de las claves que ya existen en la base de datos. Por ejemplo, si la tabla tiene tres registros con calves principales 1, 2 y 3, entonces **max() + 1** es 4. En este ejemplo, la tabla "Altova" solo tiene un registro con clave principal 1, así que **max() + 1** es 2, que es el valor esperado de la nueva clave principal.
generada por la BD	La base de datos usa la función de identidad para generar valores de clave.

La opción valor asignada elegida para **Name** significa que esta columna recibirá el valor de la asignación directamente. Para más información sobre las demás opciones consulte Configurar acciones de tabla de BD.

Paso nº5: vista previa de la asignación y actualizar la base de datos

Abra el panel *Resultados* para consultar una vista previa de los resultados de la asignación. La asignación genera un script SQL que contiene las acciones que se deben ejecutar en la base de datos. El script no ha modificado todavía la base de datos. Solamente se trata de una vista previa.

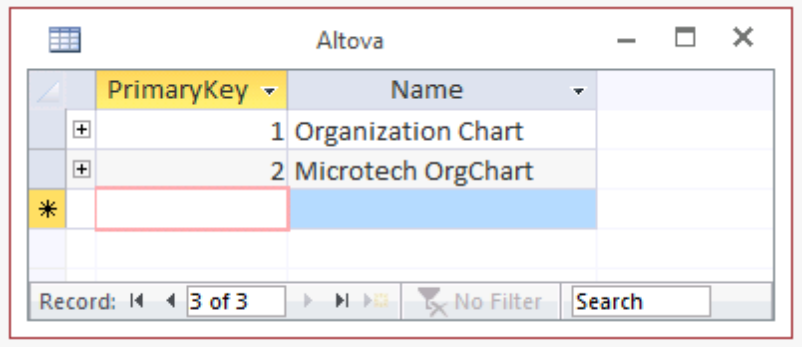
```
1   /*
2   Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar y no se pueden ejecutar en otra herramienta de consultas SQL.
3   Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
4   Conéctese a la base de datos usando esta cadena de conexión:
5   Data Source=S:\MapForce\DB Insert\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Altova]
9   --> > > %PrimaryKey1%
10
11  INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', '%PrimaryKey1%')
12
```

Para ejecutar el script en la base de datos:

- En el menú **Resultados** haga clic en **Ejecutar script SQL**.

Nota: el script SQL se puede ejecutar desde MapForce directamente pero hay otras maneras de actualizar la base de datos (*véase* *Ejecutar asignaciones que modifican bases de datos*).

Para ver el resultado abra la base de datos **altova.mdb** en DatabaseSpy o Access y observe que se añadió el registro "Microtech OrgChart" a la tabla "Altova" con la nueva clave principal 2. Los datos de este registro proceden de la instancia XML de entrada.



Hemos conseguido crear una asignación de datos que inserta datos en una tabla de base de datos. Para ver un ejemplo de asignación que inserte datos tanto en la tabla actual como en una tabla secundaria dependiente puede consultar el apartado Insertar datos en tablas vinculadas.

7.2.2.11.1.2 Insertar datos en tablas vinculadas

Las tablas de base de datos pueden ser tablas primarias, es decir, pueden ser referente de otras tablas de la base de datos, que le harán referencia mediante relaciones de clave foránea. Cuando la BD contenga este tipo de tablas podrá configurar la asignación de datos para que inserte registros en la tabla primaria pero también en sus tablas secundarias dependientes. Por ejemplo, cuando quiera insertar un registro "company" nuevo en una tabla de base de datos, también podrá insertar registros para las oficinas vinculadas a esta compañía, así como sus departamentos secundarios, sus empleados, etc.

A continuación explicamos cómo insertar datos en varias tablas sin perder las relaciones de la base de datos. Se trata de una versión algo más elaborada del ejemplo anterior (véase Insertar datos en una tabla. Para ello nos serviremos de estos archivos:

- **Altova_Hierarchical.mfd**: el archivo de asignación de datos propiamente dicho.
- **Altova_Hierarchical.xml**: el archivo XML que contiene los datos de origen que se deben insertar en la base de datos.
- **Altova_Hierarchical.xsd**: el esquema utilizado para validar el archivo de instancia anterior.
- **AltovaTarget.mdb**: la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2018 \MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

> La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos.

El objetivo de la asignación de datos que nos ocupa es reemplazar datos de la base de datos de

destino (**AltovaTarget.mdb**) con datos del archivo XML de origen. La estructura del archivo XML se corresponde con la estructura jerárquica de las tablas de la base de datos. Se trata de un organigrama que está estructurado de la siguiente manera: el elemento de nivel superior es una compañía compuesta por dos oficinas. Cada oficina está formada por varios departamentos y cada departamento por múltiples empleados. La misma jerarquía existe en la base de datos **AltovaTarget.mdb**, cuya tabla "Altova" corresponde a la compañía. Esta tabla está vinculada por medio de una relación de clave foránea a los registros de la tabla "Office". Igualmente "Office" está vinculada a "Department" y "Department" a "Person". Para ver un diagrama de relaciones de la base de datos **AltovaTarget.mdb**, ábrala en la vista "Relationships" de Access (*véase Gestionar relaciones de BD*).

Para conseguir este objetivo debemos seguir estos pasos:

Paso nº1: insertar el componente XML de origen

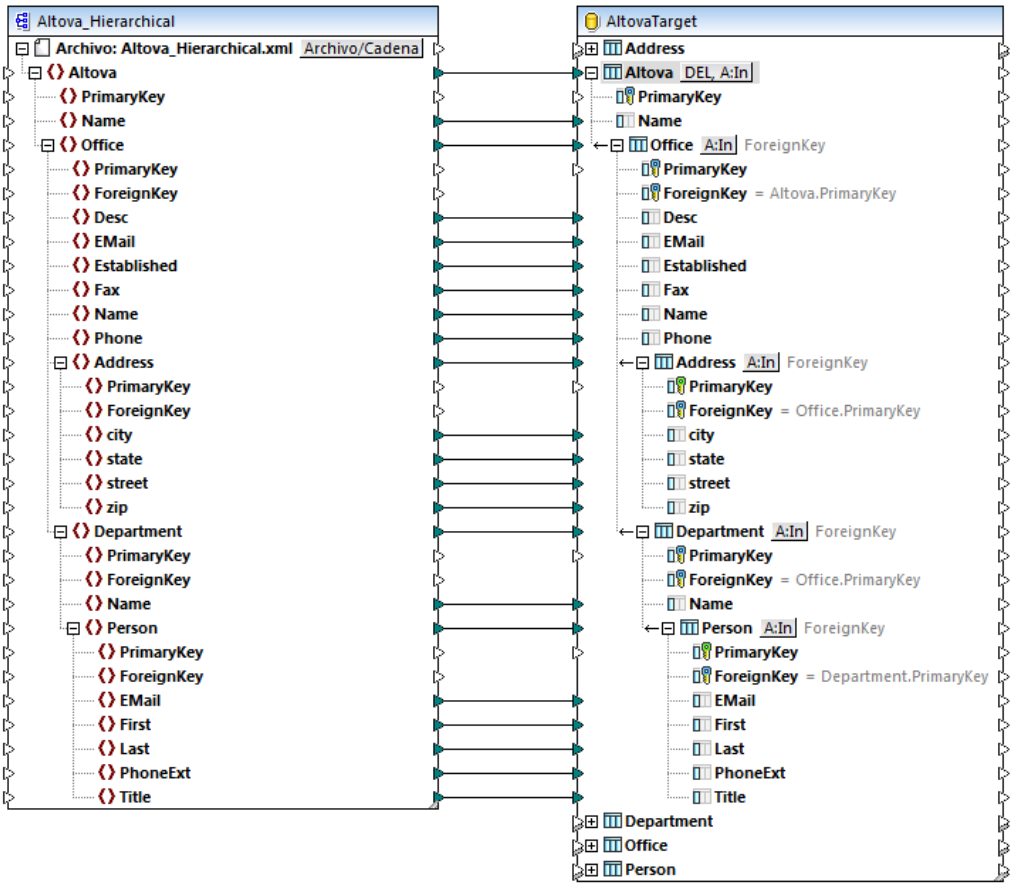
- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el esquema **Altova_Hierarchical.xsd**. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo **Altova_Hierarchical.xml**.

Paso nº2: insertar la base de datos de destino

- En el menú **Insertar** haga clic en el comando **Base de datos** y siga los pasos del asistente para conectarse a **AltovaTarget.mdb** (*véase Ejemplo: agregar la base de datos "altova.mdb" a la asignación*).

Paso nº3: dibujar las conexiones

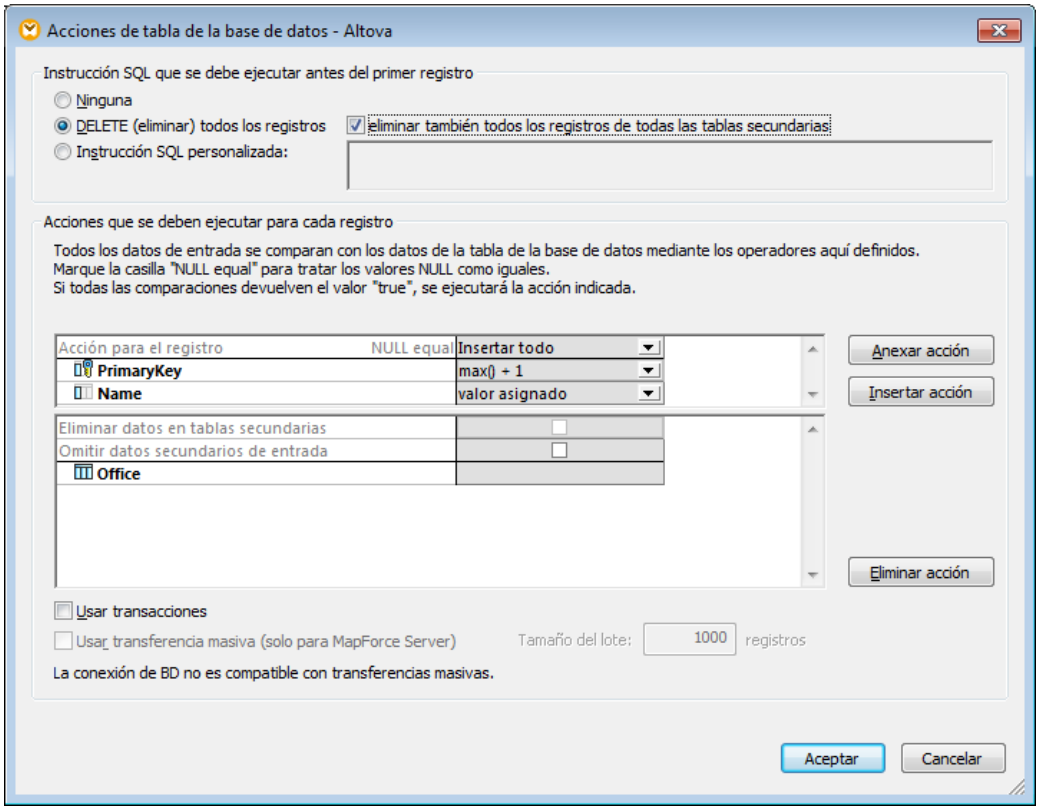
- Dibuje las conexiones de asignación de datos que aparecen en la imagen. Observe que las claves principal y foránea no están asignadas porque se generarán de forma instantánea (ver más abajo).



Nota: si se dibujan conexiones no deseadas automáticamente para los elementos descendientes, esto significa que está activa la opción *Conectar automáticamente los secundarios equivalentes*. Si esto as así, para deshacer la última acción, debe seleccionar el comando Edición **| Deshacer**. Para deshabilitar la opción de conexión automática de secundarios, haga clic en el comando **Conexión | Conectar automáticamente los secundarios equivalentes**.

Paso nº4: configurar las acciones Insertar

1. En el componente de destino haga clic en el botón **Acción: Insertar** (A:In) situado junto a la tabla "Altova" y configure la opción **max() + 1** de la clave principal (*imagen siguiente*). Esta opción se explica con más detalle en el apartado anterior (*véase Insertar datos en una tabla*).

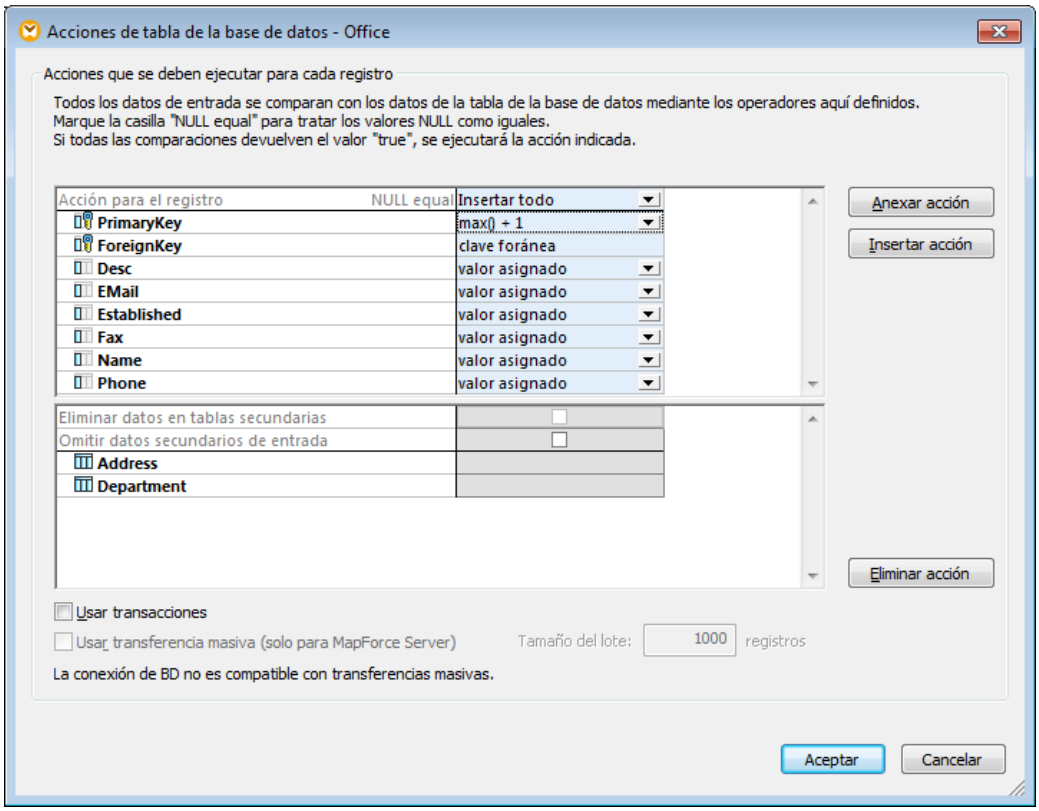


Observe que la opción *ELIMINAR todos los registros* está habilitada. Esto borra todos los registros que existen en la tabla antes de introducir los nuevos, tal y como queremos. Si deshabilita esta opción, los nuevos registros (con una clave principal nueva) se añadirán a la base de datos junto a los que ya existen cada vez que se ejecute la asignación. Pero esto no es lo que queremos.

En nuestro ejemplo también está habilitada la opción *eliminar también todos los registros de todas las tablas secundarias*. Esto garantiza que solamente se eliminen los registros de la tabla "Altova", pero también todos los registros que estén vinculados a la tabla "Altova" por medio de relaciones de clave foránea. Si las tablas secundarias tienen tablas secundarias propias, sus registros también se eliminarán y así sucesivamente hasta llegar al final de la jerarquía. Si intenta eliminar los registros de la tabla raíz "Altova" solamente, se comprometerá la integridad de la base de datos y la asignación no se podrá ejecutar.

Para obtener más información sobre las demás opciones del cuadro de diálogo "Acciones de tabla de la base de datos" consulte el apartado Configurar acciones de tabla de BD.

2. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Observe que en el área de asignación el aspecto del botón es DEL, A:In. Esto indica que se configuró una instrucción Delete para que la eliminación tenga lugar antes de la acción Insert.
3. Haga clic en el botón A:In de la tabla "Office" y configure la opción **max() + 1** de la clave principal.



4. Repita el paso nº3 para cada tabla a partir de la tabla "Office", es decir, en las tablas "Address", "Department" y "Person". Asegúrese de que todas estas tablas son descendientes inmediatos de la tabla raíz "Altova". Si necesita información sobre qué es una tabla raíz y por qué se necesita, consulte Gestionar relaciones de BD.

Paso nº5: vista previa de la asignación y actualizar la base de datos

Abra el panel *Resultados* para consultar una vista previa de los resultados de la asignación. La asignación genera un script SQL que contiene las acciones que se deben ejecutar en la base de datos. El script no ha modificado todavía la base de datos. Solamente se trata de una vista previa.

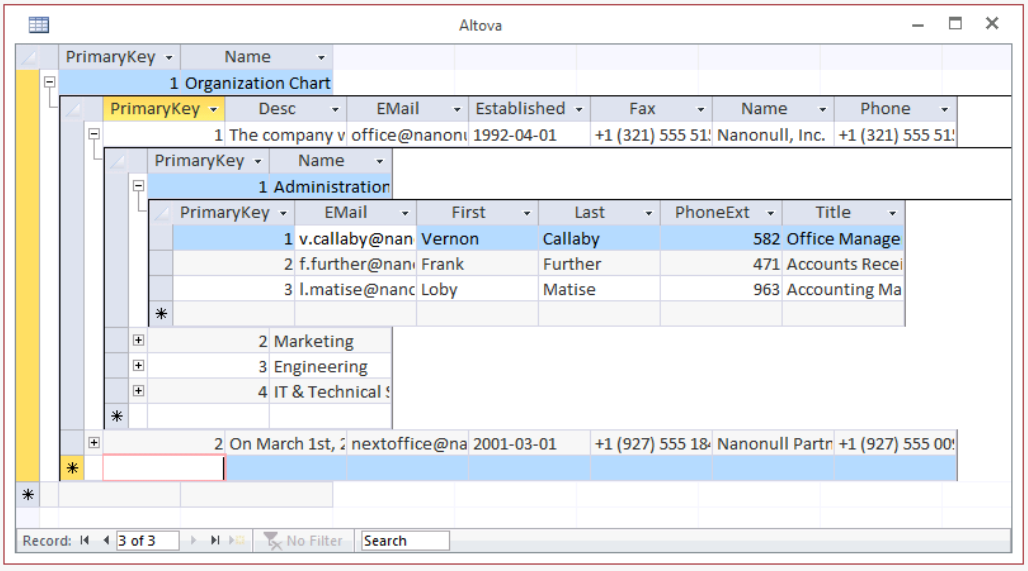
Para ejecutar el script en la base de datos:

- En el menú **Resultados** haga clic en **Ejecutar script SQL**.

Nota: el script SQL se puede ejecutar desde MapForce directamente pero hay otras maneras de actualizar la base de datos (*véase Ejecutar asignaciones que modifican bases de datos*).

Para ver el resultado abra la tabla "Altova" en Microsoft Access y observe cómo las relaciones del archivo XML se han copiado en la base de datos, desde la tabla "Altova" hasta la tabla "Person".

© 2018 Altova GmbH



Hemos conseguido crear una asignación de datos que inserta datos en varias tablas de la base de datos sin comprometer la integridad de las relaciones existentes entre las tablas.

7.2.2.11.1.3 Actualizar una tabla

Con este ejemplo explicamos cómo actualizar datos de una tabla de base de datos con datos procedentes de un XML de origen. Para ello nos serviremos de estos archivos:

- **altova-cmpy.xml**, que contiene los datos de origen que se deben insertar en la base de datos.
- **Altova_Hierarchical.xsd**, el esquema que se utilizará para validar el archivo de instancia anterior.
- **altova.mdb**, la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

> La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos.

El objetivo de la asignación de datos que nos ocupa es actualizar todos los registros de la tabla "Person" con instancias de "Person" del documento XML. Cada persona del archivo XML tiene un elemento secundario llamado `PrimaryKey`. Cada persona de la tabla "Person" tiene una columna llamada `PrimaryKey`. Solamente se deben actualizar los registros que correspondan a personas del archivo XML con un secundario `PrimaryKey` que coincida con la columna `PrimaryKey` de una

persona de la base de datos.

Para conseguir este objetivo debemos seguir estos pasos:

Paso nº1: insertar el componente XML de origen

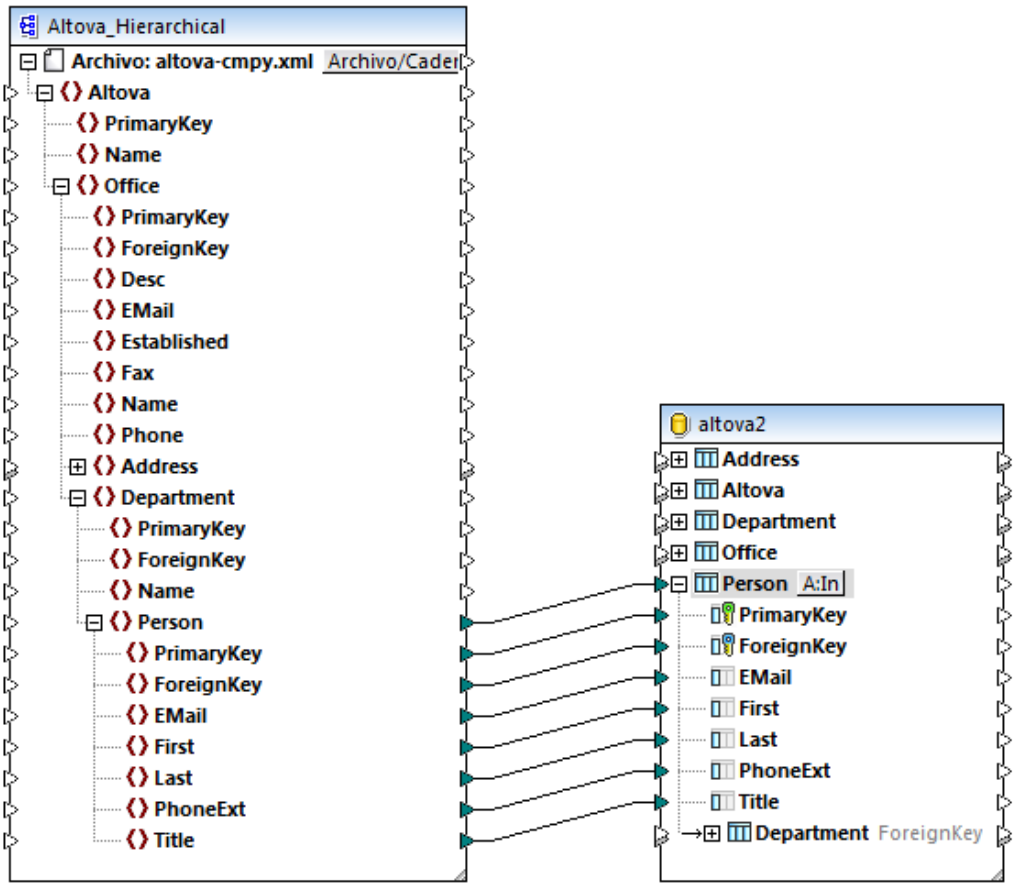
- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el esquema **Altova_Hierarchical.xsd**. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo **altova-cmpy.xml**.

Paso nº2: insertar la base de datos de destino

- En el menú **Insertar** haga clic en el comando **Base de datos** y siga los pasos del asistente para conectarse a **altova.mdb** (*véase* *Ejemplo: agregar la base de datos "altova.mdb" a la asignación*).

Paso nº3: dibujar las conexiones

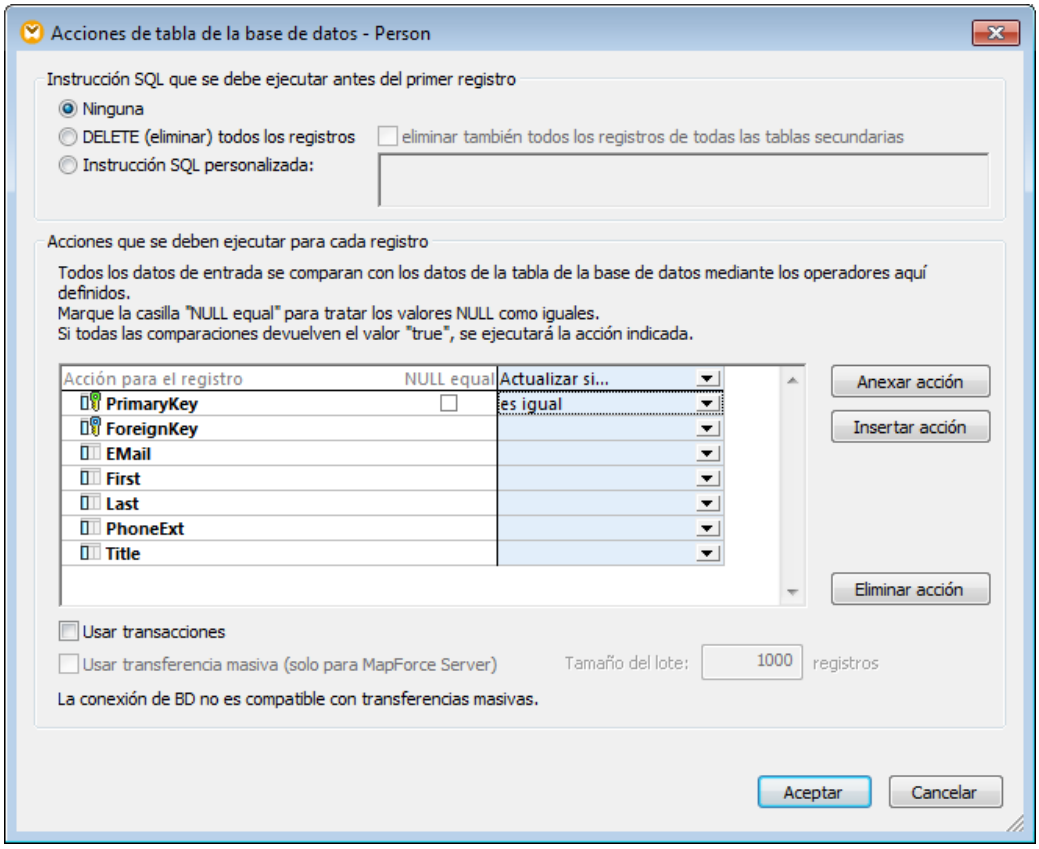
- Dibuje las conexiones de asignación de datos que aparecen en la imagen.



 © 2018 Altova GmbH

Paso nº4: configurar la acción Actualizar

1. En el componente de destino haga clic en el botón **Acción: Insertar** A:In situado junto a la tabla "Person".
2. Junto al campo *Acción para el registro* seleccione la opción **Actualizar si...**, que cambiará la acción de tabla de base de datos a una acción de actualización condicional. Es decir, el registro actual solamente se actualizará si se cumple determinada condición (ver paso siguiente).
3. Junto al campo **`PrimaryKey`**, seleccione el valor **es igual** (*imagen siguiente*). Esto define la condición de actualización: es decir, el registro de la base de datos solamente se actualizará cuando el valor de su **`PrimaryKey`** sea igual al valor de **`PrimaryKey`** que procede de la asignación.



En este ejemplo el operador de igualdad se aplica al campo **`PrimaryKey`**, que suele ser el caso en tareas de actualización de bases de datos. Tenga en cuenta que también se pueden definir condiciones en otros campos y no solo en claves principales. Por ejemplo, si selecciona el valor **es igual** de los campos **`First`** y **`Last`**, podría actualizar los registros cuyo nombre y apellido coincida con los del XML de origen.

4. Haga clic en Aceptar para cerrar el cuadro de diálogo. Observe que en la asignación el botón **Acción: Insertar** A:In ahora se ha transformado en el botón **Acción: Actualizar** A:Up para indicar que se definió una acción de actualización para esta tabla.

Paso nº5: vista previa de la asignación y actualizar la base de datos

Abra el panel *Resultados* para consultar una vista previa de los resultados de la asignación. La asignación genera un script SQL que contiene las acciones que se deben ejecutar en la base de datos. El script no ha modificado todavía la base de datos. Solamente se trata de una vista previa.

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar
    y no se pueden ejecutar en otra herramienta de consultas SQL.
3   Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
4   Conéctese a la base de datos usando esta cadena de conexión:
5   Data Source=S:\Documentation\Public\ExampleFiles\EN\MapForce\DB Update
    if\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] =
     'Aldrich', [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
9
10  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] =
    'Bander', [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] =
    'Clovis', [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
13
14  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'd.Durnell@microtech.com', [First] = 'Dave', [Last] =
    'DurnellI', [PhoneExt] = 621, [Title] = 'Marketing Manager Europe' WHERE ([Person].[PrimaryKey] = 4)
15
16  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'e.ellas@microtech.com', [First] = 'Eve', [Last] =
    'Ellas', [PhoneExt] = 753, [Title] = 'Art Director' WHERE ([Person].[PrimaryKey] = 5)
17
18  UPDATE [Person] SET [ForeignKey] = 3, [EMail] = 'f.fortunas@microtech.com', [First] = 'Fred', [Last] =
    'Fortunas', [PhoneExt] = 951, [Title] = 'Program Manager' WHERE ([Person].[PrimaryKey] = 6)
19
20  UPDATE [Person] SET [ForeignKey] = 3, [EMail] = 'g.gundall@microtech.com', [First] = 'Gerry', [Last] =
     'Gundall', [PhoneExt] = 654, [Title] = 'Software Engineer' WHERE ([Person].[PrimaryKey] = 7)
```

Para ejecutar el script en la base de datos:

- En el menú **Resultados** haga clic en **Ejecutar script SQL**.

Nota: el script SQL se puede ejecutar desde MapForce directamente pero hay otras maneras de actualizar la base de datos (*véase* *Ejecutar asignaciones que modifican bases de datos*).

7.2.2.11.1.4 Acción "Actualizar si... Insertar restantes"

A veces no solo es necesario actualizar registros de una tabla sino insertar registros nuevos en la misma tabla. En casos así podemos usar la acción **Actualizar si... Insertar restantes** de MapForce, que funciona de la siguiente manera:

- Si se cumple la condición **Actualizar si**, el registro de la base de datos se actualiza con los datos de la asignación.
- Si no se cumple la condición **Actualizar si** y existe una condición **Insertar restantes**, se inserta un registro nuevo.
- Si en la base de datos existen registros que no tienen equivalentes en el archivo de origen, estos registros no se modifican.

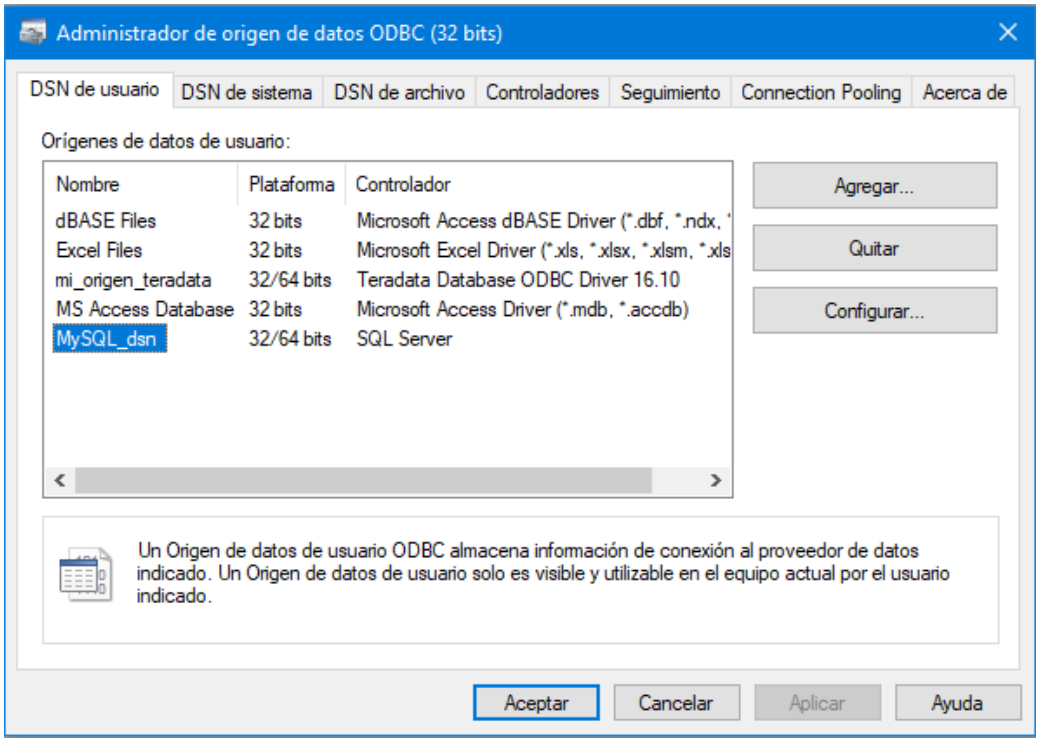
© 2018 Altova GmbH

Nota sobre MySQL ODBC

Si la base de datos de destino es MySQL con conexión por ODBC, deberá habilitar la opción *Return matched rows instead of affected rows* (Devolver filas coincidentes en lugar de filas afectadas) de la pestaña **Cursor/Results** del conector MySQL ODBC. Si lo prefiere e introduce la cadena de conexión a mano en el asistente de conexión para bases de datos, añada `Option=2` a la cadena de conexión (p. ej. `Dsn=mydsn;Option=2;`)

Para habilitar esta opción desde el conector MySQL ODBC:

1. Pulse la tecla **Windows** y teclee "ODBC".
2. Ejecute el administrador de fuentes de datos ODBC (versión de 32 bits o de 64 bits, dependiendo de la plataforma del conector MySQL ODBC instalado).
3. Haga clic en el nombre de origen de datos (DSN) utilizado por la asignación de MapForce y después haga clic en el botón **Configure** para modificar la configuración.



4. Haga clic en **Details >>** para acceder a las opciones avanzadas de configuración.
5. Haga clic en la pestaña **Cursors/Results** y después marque la casilla *Return matched rows instead of affected rows*.

Ejemplo

El siguiente ejemplo explica cómo combinar (actualizar e insertar) datos de un XML de origen en una tabla de una base de datos. Para ello se usan estos archivos:

- **altova-cmpy-extra.xml**, que contiene los datos de origen que se deben insertar en la

base de datos.
- **Altova_Hierarchical.xsd**, que es el esquema utilizado para validar el archivo de instancia anterior.
- **altova.mdb**, que es la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2018 \MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos.

El objetivo de la asignación que nos ocupa es combinar todos los registros de un documento XML de origen en una tabla de destino llamada "Person". Es decir, por cada registro de los datos XML de origen, la asignación deberá hacer lo siguiente:

- Si el campo `PrimaryKey` de la persona en el archivo XML se corresponde con el campo `PrimaryKey` de la persona en la base de datos, la asignación deberá actualizar el registro.
- Los registros de la tabla "Person" que no cumplan esta condición no se modificarán.
- Si el campo `PrimaryKey` de la persona en el archivo XML no tiene equivalentes en la tabla de base de datos de destino, la asignación añadirá un registro nuevo en la tabla.

Para conseguir este objetivo debemos seguir estos pasos:

Paso nº1: insertar el componente XML de origen

- En el menú **Insertar** haga clic en el comando **Archivo o esquema XML** y navegue hasta el esquema **Altova_Hierarchical.xsd**. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo **altova-cmpy-extra.xml**.

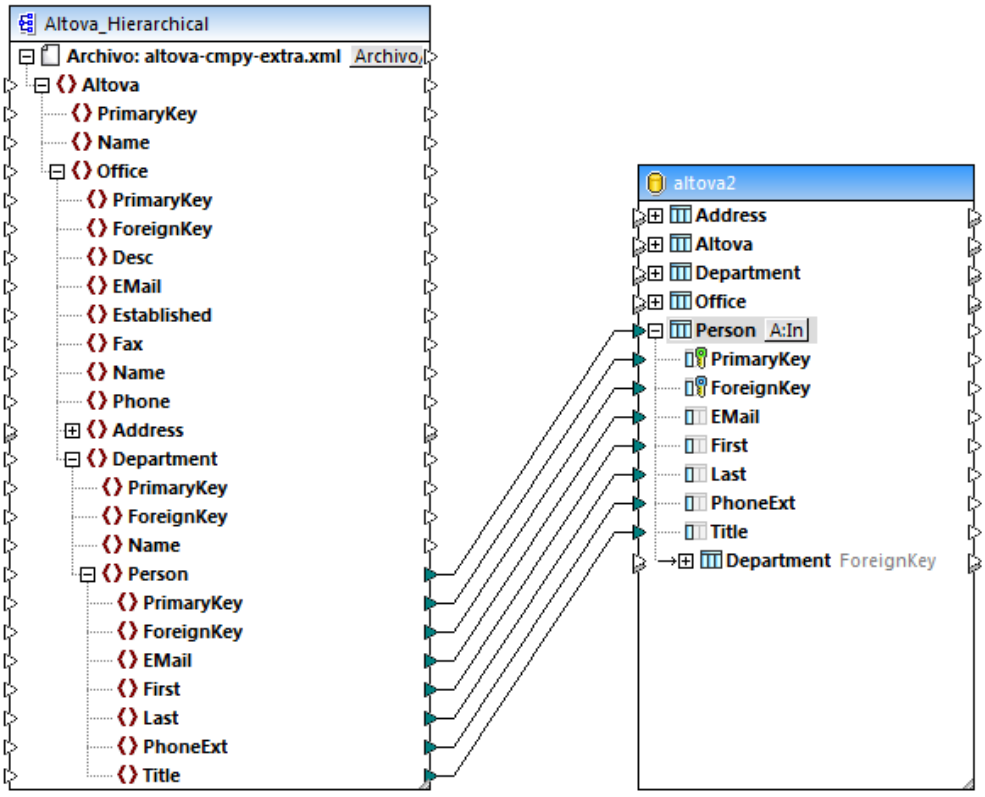
Paso nº2: insertar la base de datos de destino

- En el menú **Insertar** haga clic en el comando **Base de datos** y siga los pasos del asistente para conectarse a **altova.mdb** (*véase Ejemplo: agregar la base de datos "altova.mdb" a la asignación*).

Paso nº3: dibujar las conexiones

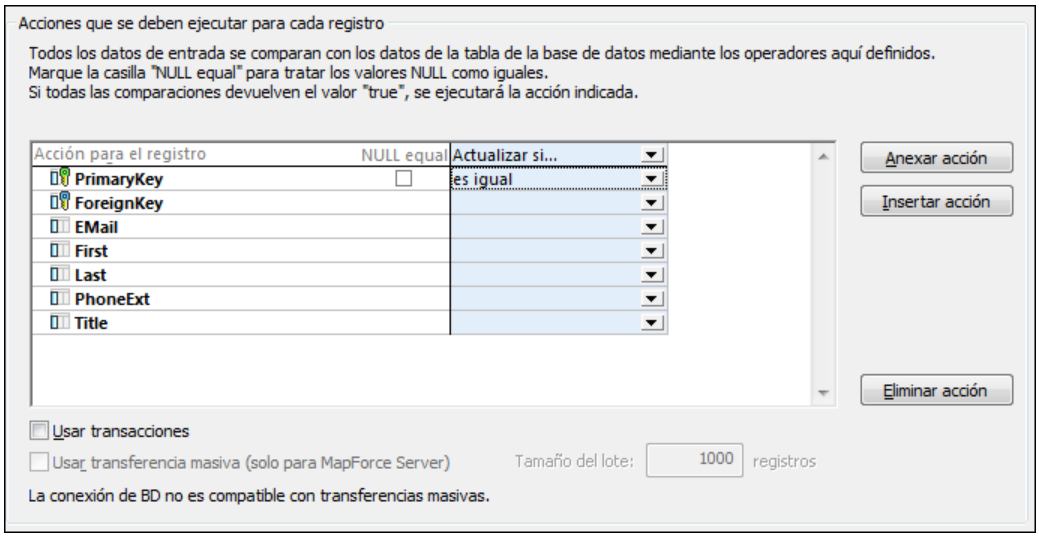
- Dibuje las conexiones de asignación de datos que aparecen en la imagen.

© 2018 Altova GmbH

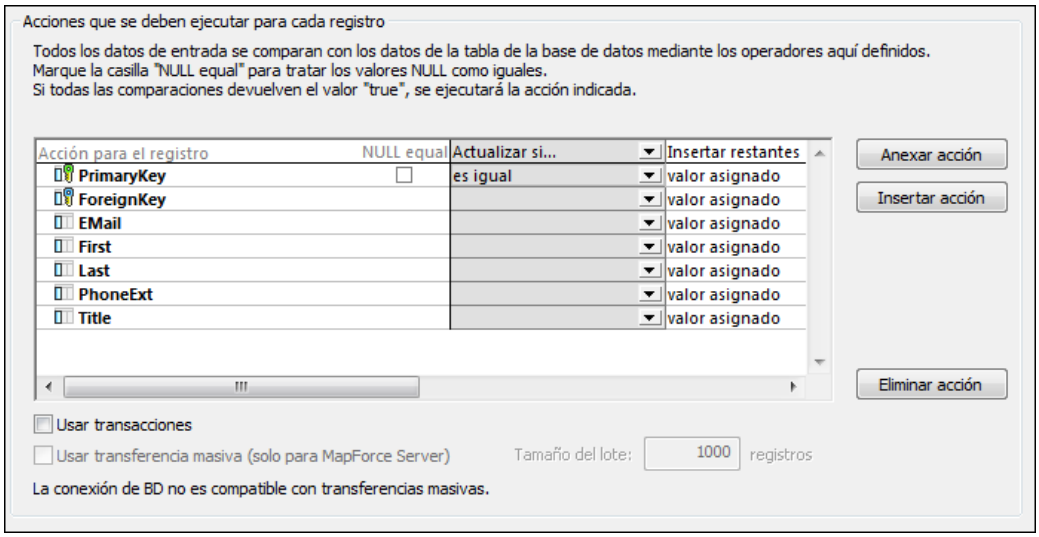


Paso nº4: configurar las acciones "Actualizar si... Insertar restantes"

1. En el componente de destino haga clic en el botón **Acción: Insertar** A:In situado junto a la tabla "Person".
2. Junto al campo *Acción para el registro* seleccione **Actualizar si...**, lo cual cambia la acción de tabla de base de datos por una acción de actualización condicional. Es decir, el registro actual solamente se actualizará si se cumple una condición concreta (paso siguiente).
3. Junto al campo **`PrimaryKey`**, seleccione el valor es igual (*imagen siguiente*). Esto define la condición de actualización: el registro de la base de datos solamente se actualizará si su valor **`PrimaryKey`** es igual al valor **`PrimaryKey`** procedente de la asignación.



4. Haga clic en el botón **Anexar acción**. Esto añade una acción nueva a la derecha de la acción **Actualizar si...** actual. Configure esta acción nueva como **Insertar restantes**.



En la imagen anterior, las acciones de tabla de base de datos están configuradas de acuerdo con los objetivos de esta asignación. Es decir, el registro solamente se actualizará si se cumple la condición **Actualizar si**. De lo contrario, el registro se insertará. La opción valor asignado especifica que los valores de la asignación se usarán para rellenar todos los campos del registro.

También se pueden definir más de dos acciones en la misma tabla de base de datos (aunque en nuestro ejemplo esto no es necesario). Cuando se ejecute la asignación, las acciones se ejecutarán de izquierda a derecha. Se entiende que la última acción **Insertar** es la última acción y las acciones que se añadan después se pasarán por alto.

Tenga en cuenta que el botón **Anexar acción** del cuadro de diálogo añade una acción nueva *después* de la acción seleccionada. El botón **Insertar acción** añade una acción nueva *antes* de la acción seleccionada. Para eliminar una acción haga clic en ella y después en el botón **Eliminar acción**.

 © 2018 Altova GmbH

5. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. Observe cómo, en la asignación, el botón **Acción:Insertar** A:In se ha convertido en **Acción: Actualizar; Insertar** A:Up,In. Esto indica que se configuró una acción de actualización y de inserción para esta tabla.

Paso nº5: vista previa de la asignación y actualizar la base de datos

Abra el panel *Resultados* para consultar una vista previa de los resultados de la asignación. La asignación genera un script SQL que contiene las acciones que se deben ejecutar en la base de datos. El script no ha modificado todavía la base de datos. Solamente se trata de una vista previa.

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar y no
    se pueden ejecutar en otra herramienta de consultas SQL.
3   Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
4   Conéctese a la base de datos usando esta cadena de conexión:
5   Data Source=S:\ExampleFiles\EN\MapForce\DB Update if... Insert
    Rest\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
6   */
7
8   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] =
    'Aldrich', [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
9
10  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] =
    'Bander', [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis'
    , [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
13
14  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.Cicada@microtech.com', [First] = 'Camilla ', [Last] =
    'Cicada', [PhoneExt] = 765, [Title] = 'HR' WHERE ([Person].[PrimaryKey] = 30)
15
16  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.corrigan@microtech.com', [First] = 'Carol', [Last] =
    'Corrigan', [PhoneExt] = 629, [Title] = 'Admin' WHERE ([Person].[PrimaryKey] = 31)
17
18  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'd.Durnell@microtech.com', [First] = 'Dave', [Last] =
    'Durnell', [PhoneExt] = 621, [Title] = 'Marketing Manager Europe' WHERE ([Person].[PrimaryKey] = 4)
19
20  UPDATE [Person] SET [ForeignKey] = 2, [EMail] = 'e.ellas@microtech.com', [First] = 'Eve', [Last] = 'Ellas',
    [PhoneExt] = 753, [Title] = 'Art Director' WHERE ([Person].[PrimaryKey] = 5)
21
22  UPDATE [Person] SET [ForeignKey] = 4, [EMail] = 'n.newbury@microtech.com', [First] = 'Nira', [Last] =
    'Newbury', [PhoneExt] = 241, [Title] = 'Support Engineer' WHERE ([Person].[PrimaryKey] = 14)
23
24  UPDATE [Person] SET [ForeignKey] = 4, [EMail] = 'o.origone@microtech.com', [First] = 'Olanda', [Last] =
    'Origone', [PhoneExt] = 345, [Title] = 'Support Engineer' WHERE ([Person].[PrimaryKey] = 15)
```

Script SQL (vista parcial) antes de actualizarse la base de datos

Puede que en el script de vista previa no aparezca ninguna instrucción INSERT. Esto es totalmente normal y se debe a que los registros se insertan de forma condicional y a que las instrucciones INSERT dependen del resultado de la acción **Actualizar si** (lo cual no ocurre hasta que se ejecuta la asignación).

Nota: para determinadas bases de datos MapForce crea instrucciones de combinación MERGE en lugar de instrucciones UPDATE. Para más información consulte el apartado Instrucciones de combinación MERGE.

Para ejecutar el script en la base de datos:

- En el menú **Resultados** haga clic en **Ejecutar script SQL**.

Una vez ejecutada la asignación de datos y aplicado el script a la base de datos, podemos observar cómo las instrucciones INSERT ahora aparecen en el panel *Resultados*.

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se ejecutaron durante la función "Generar
    resultados".
3   Cada resultado aparece escrito a la derecha de la cadena "-->>>".
4   Estas instrucciones se presentan únicamente de forma preliminar y no se pueden ejecutar en otra
    herramienta de consultas SQL.
5   La base de datos se conectó usando esta cadena de conexión:
6   Data Source=S:\ExampleFiles\EN\MapForce\DB Update if... Insert
    Rest\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
7   */
8
9   UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] =
    'Aldrich', [PhoneExt] = 582, [Title] = 'Manager' WHERE ([Person].[PrimaryKey] = 1)
10  -->>> OK. 1 fila(s).
11
12  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'b.bander@microtech.com', [First] = 'Bert', [Last] =
    'Bander', [PhoneExt] = 471, [Title] = 'Accounts Receivable' WHERE ([Person].[PrimaryKey] = 2)
13  -->>> OK. 1 fila(s).
14
15  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.clovis@microtech.com', [First] = 'Clive', [Last] = 'Clovis'
    , [PhoneExt] = 963, [Title] = 'Accounting Manager' WHERE ([Person].[PrimaryKey] = 3)
16  -->>> OK. 1 fila(s).
17
18  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.Cicada@microtech.com', [First] = 'Camilla ', [Last] =
    'Cicada', [PhoneExt] = 765, [Title] = 'HR' WHERE ([Person].[PrimaryKey] = 30)
19  -->>> OK. 0 fila(s).
20
21  INSERT INTO [Person] ([PrimaryKey], [ForeignKey], [EMail], [First], [Last], [PhoneExt], [Title]) VALUES (30, 1,
    'c.Cicada@microtech.com', 'Camilla ', 'Cicada', 765, 'HR')
22  -->>> OK. 1 fila(s).
23
24  UPDATE [Person] SET [ForeignKey] = 1, [EMail] = 'c.corrigan@microtech.com', [First] = 'Carol', [Last] =
    'Corrigan', [PhoneExt] = 629, [Title] = 'Admin' WHERE ([Person].[PrimaryKey] = 31)
25  -->>> OK. 0 fila(s).
26
27  INSERT INTO [Person] ([PrimaryKey], [ForeignKey], [EMail], [First], [Last], [PhoneExt], [Title]) VALUES (31, 1,
    'c.corrigan@microtech.com', 'Carol', 'Corrigan', 629, 'Admin')
```

Script SQL (vista parcial) después de actualizarse la base de datos

Nota: el script SQL se puede ejecutar desde MapForce directamente pero hay otras maneras de actualizar la base de datos (*véase* *Ejecutar asignaciones que modifican bases de datos*).

Si abrimos la tabla "Person" en el panel *Consulta de BD* de MapForce (*véase* *Examinar y consultar bases de datos*), podrá ver el resultado de la asignación:

- Se actualizaron todos los registros de la base de datos que tenían claves principales equivalentes en el archivo XML (p. ej. los registros con clave principal 1, 2, 3, 4 y 5).
- No se modificaron los registros de la base de datos que no tenían claves equivalentes en el archivo XML (p. ej. los registros con clave principal 6, 7, 8 y 9).
- Se insertaron registros nuevos en la tabla "Person", cuando la clave no existía en la base de datos (p. ej. los registros con clave principal 30 y 31).

© 2018 Altova GmbH

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	1	1	A.Aldrich@microtech.com	Albert	Aldrich	582	Manager
2	2	1	b.bander@microtech.com	Bert	Bander	471	Accounts Receivable
3	3	1	c.clovis@microtech.com	Clive	Clovis	963	Accounting Manager
4	4	2	d.Durnell@microtech.com	Dave	Durnelll	621	Marketing Manager Europe
5	5	2	e.ellas@microtech.com	Eve	Ellas	753	Art Director
6	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
7	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
8	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
9	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
10	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
11	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
12	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
13	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
14	14	4	n.newbury@microtech.com	Nira	Newbury	241	Support Engineer
15	15	4	o.origone@microtech.com	Olanda	Origone	345	Support Engineer
16	16	5	s.meier@microtech.com	Paul	Ponetti	114	Office Manager
17	17	5	t.bone@microtech.com	Quang	Qui	331	Accounts Receivable
18	18	6	m.nafta@microtech.com	Rita	Rotter	122	PR & Marketing Manager US
19	19	7	v.bass@microtech.com	Silla	Semua	716	IT Manager
20	20	7	c.franken@microtech.com	Tom	Tinker	147	Support Engineer
21	21	7	m.redgreen@microtech.com	Uri	Ugoner	152	Support Engineer
22	30	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
23	31	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

Tabla "Person" después de actualizarse la base de datos

7.2.2.11.1.5 Instrucciones de combinación MERGE

En algunas asignaciones de datos que actualizan e insertan datos en una tabla de base de datos (*véase Acción "Actualizar si... Insertar restantes"*). MapForce genera instrucciones de combinación MERGE que se ejecutarán en la base de datos cuando se ejecute la asignación. El motor de ejecución no tiene por qué ser MapForce (*véase Ejecutar asignaciones que modifican BD*).

Las instrucciones de combinación MERGE son compatibles con estas bases de datos:

- SQL Server 2008 y superior
- Oracle
- DB2
- Firebird

Las instrucciones MERGE reducen el número de llamadas al servidor de la base de datos porque compaginan instrucciones INSERT y UDPATE en una sola instrucción. Además, en el caso de las instrucciones MERGE la comprobación de coherencia la lleva a cabo la propia base de datos. Recuerde que MapForce crea instrucciones MERGE automáticamente cuando detecta un tipo de base de datos compatible con estas instrucciones. Es decir, se trata de un comportamiento

automático de la aplicación que no se puede configurar.

Para comprobar si la asignación ejecutará instrucciones MERGE en tiempo de ejecución (en lugar de instrucciones INSERT y UPDATE):

1. Cree una asignación que utilice acciones **Actualizar si...** y acciones **Insertar restantes** (*véase* *Acción "Actualizar si... Insertar restantes"*).
2. Consulte la vista previa de resultados de la asignación (abriendo el panel *Resultados*).

Si la base de datos admite instrucciones MERGE, el script SQL que genera MapForce incluirá instrucciones MERGE. Por ejemplo:

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se presentan únicamente de forma preliminar y no
    se pueden ejecutar en otra herramienta de consultas SQL.
3   Para ejecutar estas instrucciones, utilice la función "Ejecutar script SQL" del menú "Resultados".
4   Conéctese a la base de datos usando esta cadena de conexión:
5   Data Source=Persist Security Info=true;Provider=SQLOLEDB.1
6   */
7
8   SET QUOTED_IDENTIFIER ON
9
10  MERGE INTO [Production].[Location] AS T USING ( VALUES ( (CAST(NULL AS nvarchar(50))) ) ) AS S ( [Name] )
     ON ( (S.[Name] = T.[Name]) ) WHEN MATCHED THEN UPDATE SET [Name] = S.[Name] WHEN NOT
    MATCHED THEN INSERT ( [Name] ) VALUES ( S.[Name] );
```

Si la base de datos no admite instrucciones MERGE, el script SQL generado incluirá instrucciones UPDATE solamente (la vista previa no incluirá instrucciones INSERT porque estas instrucciones solamente se ejecutan si no se cumple la condición **Actualizar si...** (y esto no se sabe antes de ejecutar la asignación).

Restricciones:

- Con las instrucciones de combinación MERGE la opción *Transferencia masiva* (*véase* *Inserciones masivas*) solamente es compatible con conexiones ODBC y JDBC.

7.2.2.11.1.6 Opciones para tablas secundarias cuando se actualiza la tabla principal

Cuando la asignación actualiza una tabla que es tabla *principal* (es decir, una tabla que tiene relaciones de clave foránea con otras tablas), podrá configurar cómo ocuparse de los registros dependientes tanto en el origen de datos como en la tabla de destino. Por ejemplo, imaginemos que desea actualizar la tabla "Department" de la base de datos **altova.mdb**. Como cada persona está vinculada a un departamento por medio de una clave foránea, lo más probable es que desee ejecutar acciones en la tabla "Person" también (p. ej. inserciones, actualizaciones o eliminaciones). Esto le ayudaría a mantener la integridad de la base de datos y prevenir errores de asignación.

En este tema se describen las opciones disponibles para la tabla "Person" cuando se actualiza la tabla principal "Department" y para ello utilizamos estos archivos de ejemplo:

- **altova-cmpy-extra.xml**: contiene el origen de datos que se debe insertar en la base de datos.
- **Altova_Hierarchical.xsd**: el esquema utilizado para validar el archivo de instancia anterior.

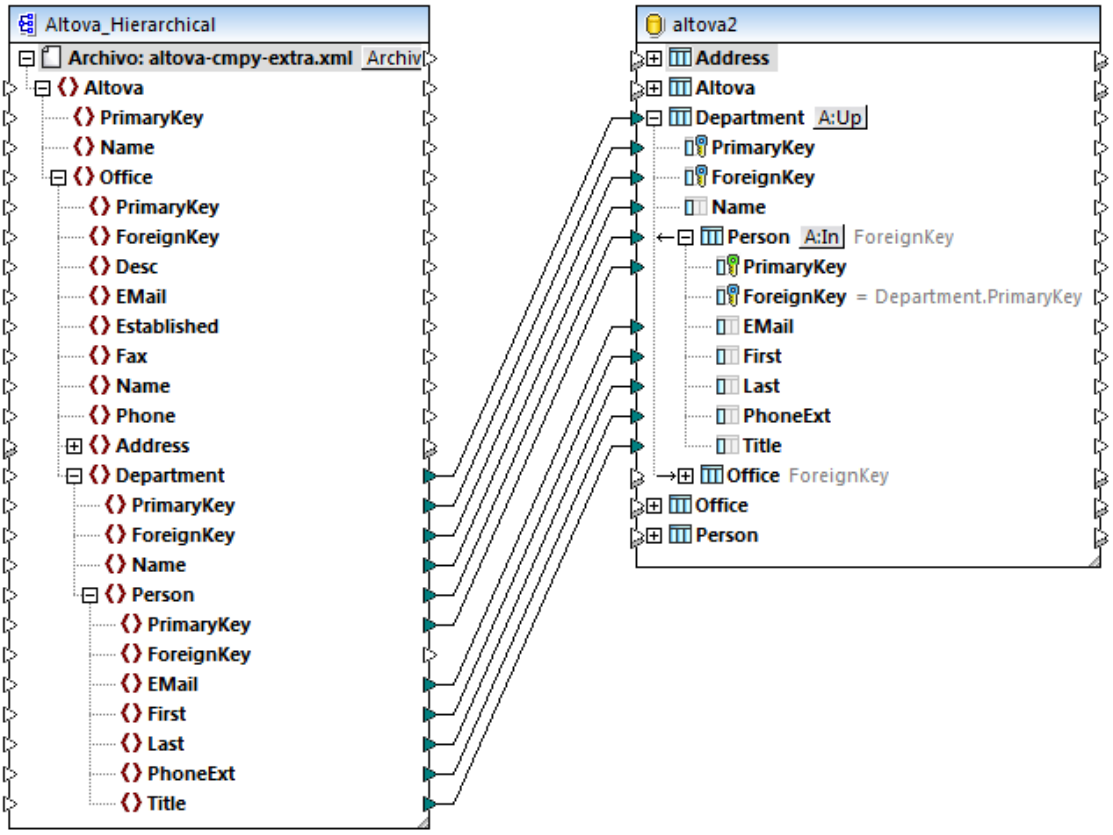
 © 2018 Altova GmbH

- **altova.mdb**: la base de datos de destino que se debe actualizar.

Todos estos archivos están en la carpeta **<Documentos>\Altova\MapForce2018 \MapForceExamples** y, para simplificar, en las instrucciones omitimos su ruta de acceso.

> La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos.

En primer lugar, añada el esquema y la instancia XML de origen así como la base de datos de destino a la asignación de datos (*véase Ejemplo: agregar la base de datos "altova.mdb" a la asignación*). Siga los mismos pasos que aparecen en el apartado Acción "Actualizar si... Insertar restantes". En segundo lugar, dibuje las conexiones que aparecen en esta imagen:

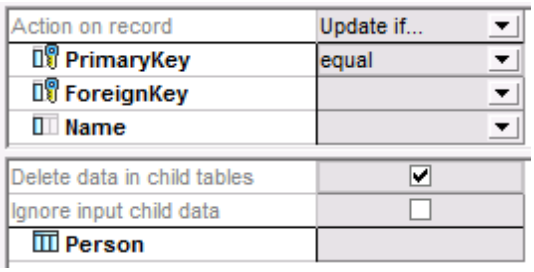
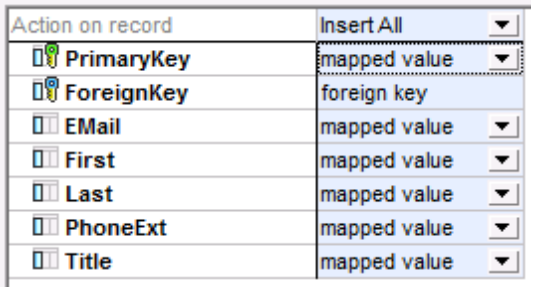


Como puede ver en la imagen, la asignación actualiza la tabla "Department" en la base de datos de destino. Esta tabla se eligió como tabla raíz. Para más información sobre las tablas raíz y por qué son necesarias consulte Gestionar relaciones de BD. La acción que debe ejecutarse en la tabla secundaria "Person" es el tema que nos ocupa en este apartado.

Las tablas que aparecen más abajo ilustran varias opciones de configuración y el resultado correspondiente. Estas opciones se pueden seleccionar en el cuadro de diálogo "Acciones de

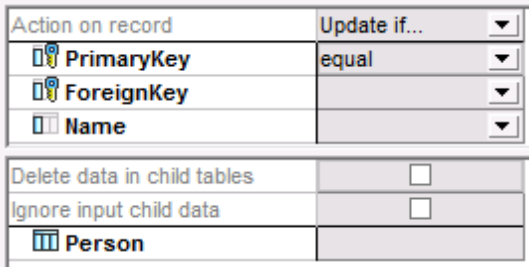
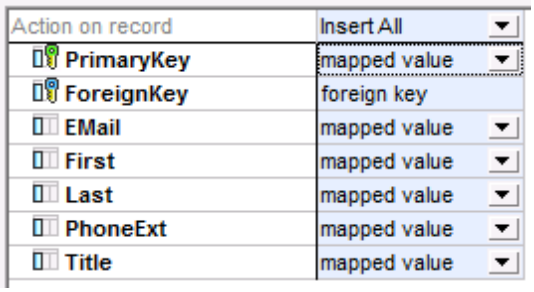
tabla de la base de datos" de la tabla principal "Department" y de la tabla secundaria "Person".

Configuración A

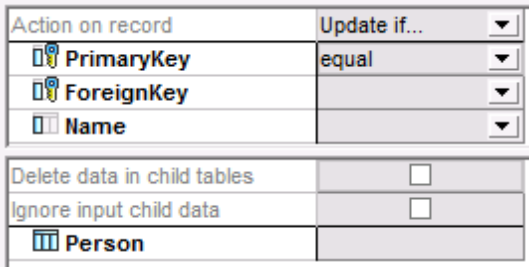
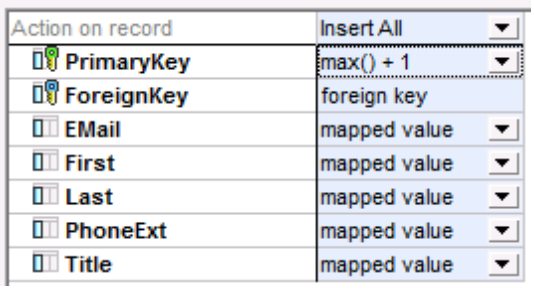
Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" donde `PrimaryKey` corresponde en el XML de origen a `PrimaryKey` en la tabla de BD. • No actualiza los registros de "Department" que no tengan un equivalente en el archivo XML de entrada (dicha clave no existe en el origen). • Inserta, desde la instancia XML de entrada, todos los registros de "Person" que todavía no existen en la base de datos. • Elimina los datos secundarios (registros de "Person") de los registros de "Department" que cumplen con la condición **Actualizar si...** definida.

© 2018 Altova GmbH

Configuración B

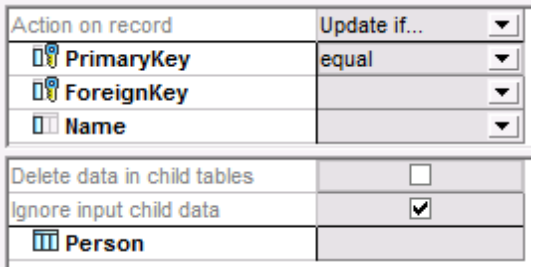
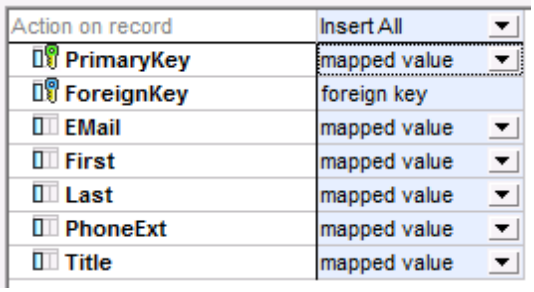
Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	La asignación falla y emite un mensaje de error de ejecución SQL. El motivo es que la asignación intenta insertar registros nuevos de "Person" con la misma clave principal que los registros de "Person" que ya existen. Si desea insertar registros desde el XML de entrada y añadirlos a los que ya existen en la base de datos, debe consultar la siguiente configuración.

Configuración C

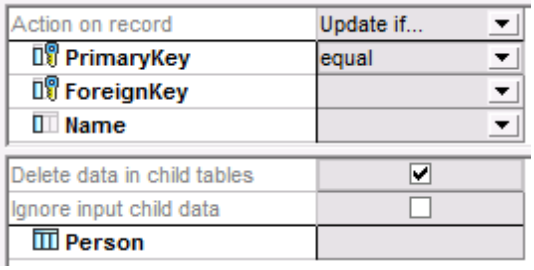
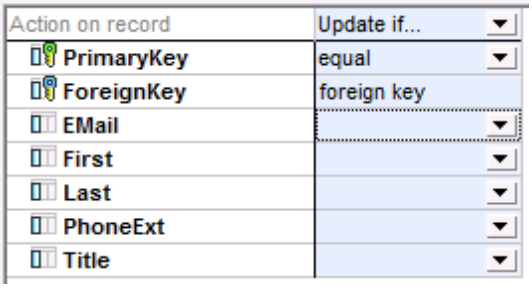
Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" cuyo registro `PrimaryKey` corresponde en el XML de origen al registro `PrimaryKey` de la tabla de la BD. • No actualiza los registros de "Department" que no tengan equivalente en el XML de entrada (dicha clave no existe en el origen). • Se insertan registros nuevos de "Person" (con claves principales generadas automáticamente) en la tabla "Person".

© 2018 Altova GmbH

Configuración D

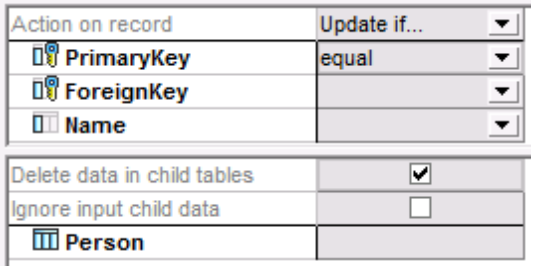
Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" cuyo `PrimaryKey` corresponda en el XML de origen al `PrimaryKey` en la tabla de la BD. • No actualiza los registros de "Department" que no tengan equivalente en el XML de entrada (dicha clave no existe en el origen). • No se insertan registros en la tabla "Person" porque la opción *Omitir datos secundarios de entrada* está habilitada para la tabla "Departments" principal.

Configuración E

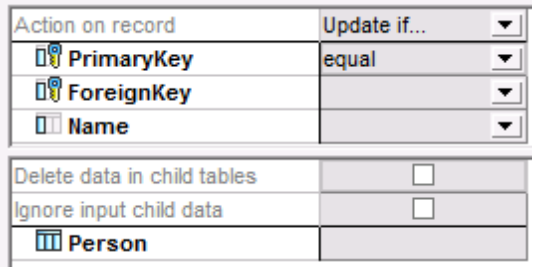
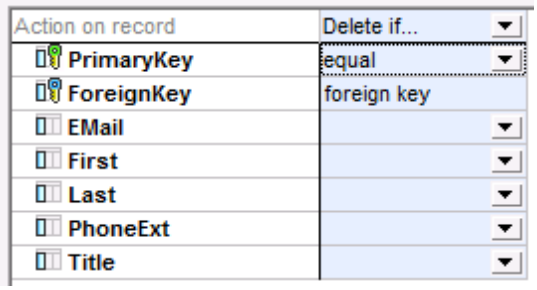
Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Actualiza los registros de "Department" cuyo `PrimaryKey` corresponda en el XML de origen al `PrimaryKey` en la tabla de la BD. • No actualiza los registros de "Department" que no tengan equivalente en el XML de entrada (dicha clave no existe en el origen). • Elimina todos los registros de "Person" vinculados a un "Department" que tenga el correspondiente `PrimaryKey` en el XML de origen. El motivo es que la opción *Eliminar datos en tablas secundarias* está habilitada para la tabla principal "Department". • Se conservan en la base de datos los registros de "Person" vinculados a un departamento que no cumpla con la condición **Actualizar si...** definida. • No se actualiza ningún registro de la tabla "Person".

© 2018 Altova GmbH

Configuración F

Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Updates Department records where `PrimaryKey` in the source XML corresponds to the `PrimaryKey` in the database table. • Does not update existing Department records which do not have a counterpart in the input XML file (no such key exists in the source). • Deletes all Person records linked to a Department which has a corresponding `PrimaryKey` in the source XML. The reason is that the **Delete data in child tables** option is enabled for the parent Department table. • Person records linked to a department that did not meet the **Update if...** condition remain in the database.

Configuración G

Opciones	Resultado de la asignación
 *Tabla "Department"*  *Tabla "Person"*	• Updates Department records where `PrimaryKey` in the source XML corresponds to the `PrimaryKey` in the database table. • Does not update existing Department records which do not have a counterpart in the input XML file (no such key exists in the source). • Deletes all Person records which satisfy both of the following conditions: a. The Person record is linked to a Department which has a corresponding `PrimaryKey` in the source XML, and b. The Person record has a corresponding `PrimaryKey` in the source XML.

7.2.2.11.1.7 Acción "Eliminar si..."

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

The table action **Delete if..**. is used to delete data from a database table conditionally. You can define this action from the Database Table Actions dialog box (see Database Table Actions Settings). For example, when mapping data from a source XML to a target database, you can configure a **Delete if...** condition to check whether a certain field in the source XML is equal to a field in the target database record (typically, a primary key value). If the **Delete if...** condition is true (that is, the two fields are equal), the database record will be deleted when the mapping runs.

Note: The **Delete if...** table action should not be confused with the **Delete data in child tables** option available in the Database Table Actions dialog box. The **Delete if...** table action only affects the table for which the action is defined; no other tables are affected.

This example shows you how to delete data from a database table conditionally, and also insert records into the same database table based on the outcome of the delete condition.

This example uses the following files:

 © 2018 Altova GmbH

- **altova-cmpy-extra.xml** — contains the source data to be inserted into the database.
- **Altova_Hierarchical.xsd** — the schema used to validate the instance file above.
- **altova.mdb** — the target database to be updated.

All files are available in the **<Documentos>\Altova\MapForce2018\MapForceExamples** folder. Below, the complete path to them will be omitted, for simplicity.

> La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos.

The goal of the mapping is as follows:

- If any person records with the same `PrimaryKey` exist both in the source XML and the target Person table, they must be deleted from the Person table.
- All other records from the source XML must be inserted into the Person table.

To achieve the mapping goal, we will take the steps below.

Step 1: Insert the source XML component

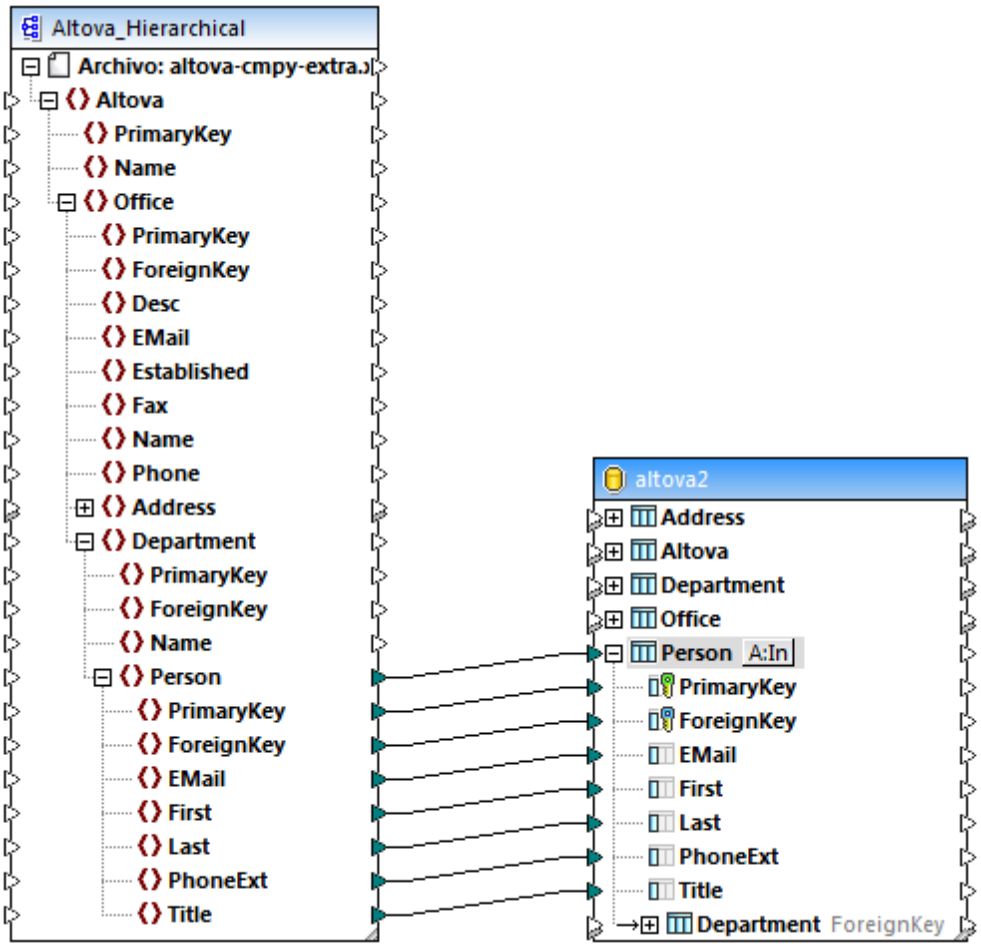
- On the **Insert** menu, click **XML Schema/File**, and browse for **Altova_Hierarchical.xsd**. When prompted to supply an instance file, browse for **altova-cmpy-extra.xml**.

Step 2: Insert the target database

- On the **Insert** menu, click **Database**, and go through the wizard steps to connect to **altova.mdb** (see Example: Adding the "altova.mdb" Database to the Mapping).

Step 3: Draw the connections

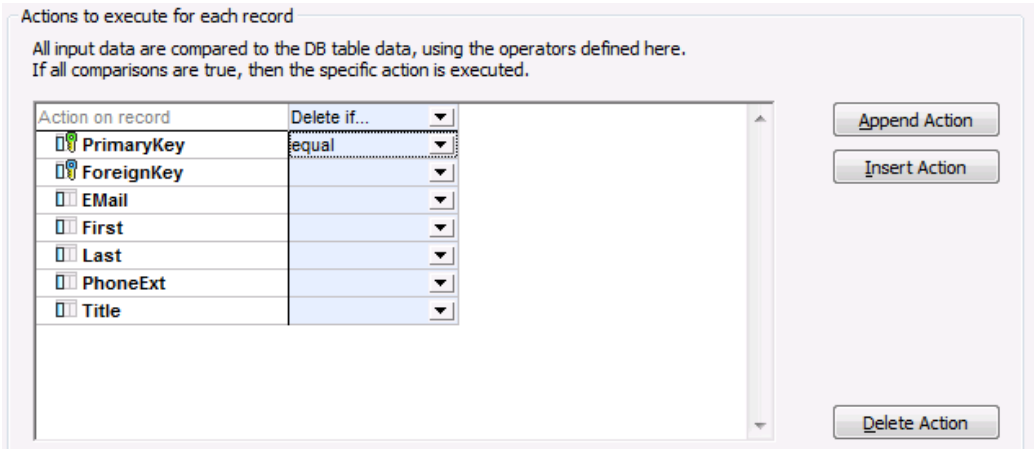
- Draw the mapping connections as shown below.



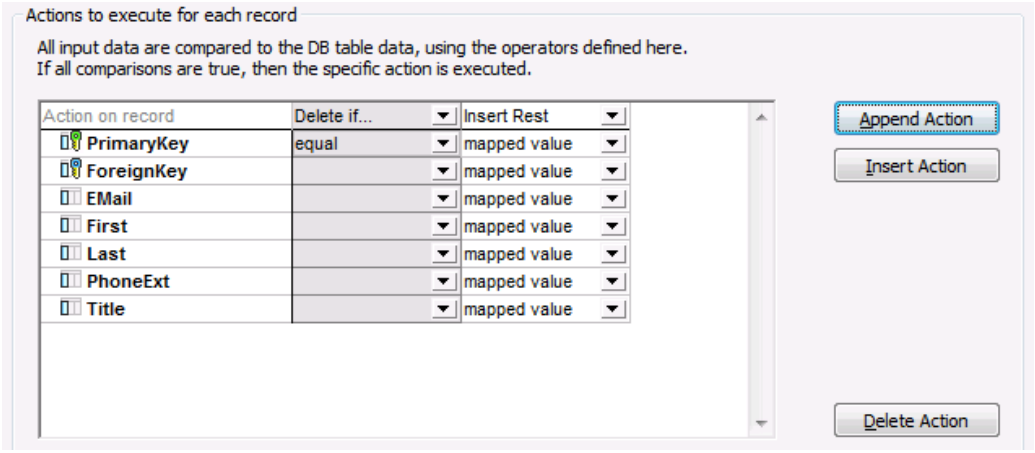
Step 4: Configure the "Delete if... Insert Rest" actions

1. On the target component, click the **Action: Insert** (A:In) button next to the "Person" table.
2. Next to **Action on record**, select **Delete if...** . This changes the database table action to a conditional delete action. That is, the current record will only be deleted when a condition is satisfied (see next step).
3. Next to **PrimaryKey**, select the value **equal**, as shown below. This defines the update condition: that is, the database record will be deleted only when its **PrimaryKey** value is equal to the **PrimaryKey** value coming from the mapping.

© 2018 Altova GmbH



4. Click **Append Action**. This adds a new action to the right of the existing **Delete If** action. Configure the new action as **Insert Rest**:



In the image above, the database table actions have been configured in accordance with the goals of the mapping. That is, only when the **Delete If**... condition is satisfied will the record be deleted; otherwise, it will be inserted. The option "mapped value" specifies that values from the mapping will be used to populate all fields of the record.

5. Click **OK** to close the dialog box. Notice that, back on the mapping, the **Action: Insert** A:In button has now changed to an **Action: Delete; Insert** (A:De,In) button. This indicates that both a delete and an insert action is configured for this table.

Step 5: Preview the mapping and update the database

Click the **Output** tab to preview the mapping. A SQL script is generated, containing actions to be executed against the database. The script has not modified the database yet; it is only for preview. To run the script against the database:

- On the **Output** menu, click **Run SQL-Script**.

Note: Running the SQL script directly from MapForce is just one of the ways to update the

database, see also Executing Mappings Which Modify Databases.

If you open the "Person" table in the DB query tab of MapForce (see Browsing and Querying Databases), you can see the result of the mapping as follows:

- All database records which had corresponding primary keys in the XML file have been deleted. Examples are records with primary key 1, 2, 3, 4, and 5.
- All database records which had no corresponding keys in the XML file remained unaffected. Examples are records with primary key 6, 7, 8, 9, 10, 11, 12, and 13.
- New records have been inserted to the "Person" table (where key did not already exist in the database). Examples are records with primary key 30 and 31.

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
2	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
3	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
4	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
5	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
6	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
7	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
8	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
9	30	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
10	31	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

The "Person" table after updating the database

7.2.2.11.1.8 Acción "Omitir si..."

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

The table action **Ignore if..**. is used to prevent certain records in a database table from being updated, based on a defined condition. The **Ignore if...** action is only meaningful when used in combination with another database table action (such as the **Insert Rest** action). For example, when mapping data from a source XML to a target database, you can configure an **Ignore if...** condition to check whether a certain field in the source XML is equal to a field in the target database record (typically, a primary key value). If the **Ignore if...** condition is true (that is, the two fields are equal), the database record will be ignored when the mapping runs, and the next defined action (**Insert Rest**, for example) will be executed.

This example shows you how insert records into a database table based on the outcome of the **Ignore if...** condition. It uses the following files:

- **altova-cmpy-extra.xml** — contains the source data to be inserted into the database.
- **Altova_Hierarchical.xsd** — the schema used to validate the instance file above.
- **altova.mdb** — the target database to be updated.

 © 2018 Altova GmbH

All files are available in the **<Documentos>\Altova\MapForce2018\MapForceExamples** folder. Below, the complete path to them will be omitted, for simplicity.

> La asignación de este ejemplo modifica un archivo de base de datos de ejemplo. Lo más recomendable es crear una copia de seguridad de la base de datos original y empezar con una copia nueva antes de seguir las instrucciones que aparecen más abajo. Así se evitarán posibles modificaciones en los ejemplos originales y obtendrá el mismo resultado que en este ejemplo. Para más información consulte Ejecutar asignaciones que modifican bases de datos.

The goal of the mapping is as follows:

- If any person records with the same `PrimaryKey` exist both in the source XML and the target Person table, no action must be taken against them (that is, they must be ignored).
- If any person records which do not meet the above condition exist in the Person table, no action must be taken against them either.
- Records from the source XML which do not have a counterpart (no primary key) in the Person table must be treated as new and inserted into the Person table with a new primary key.

To achieve the mapping goal, we will take the steps below.

Step 1: Insert the source XML component

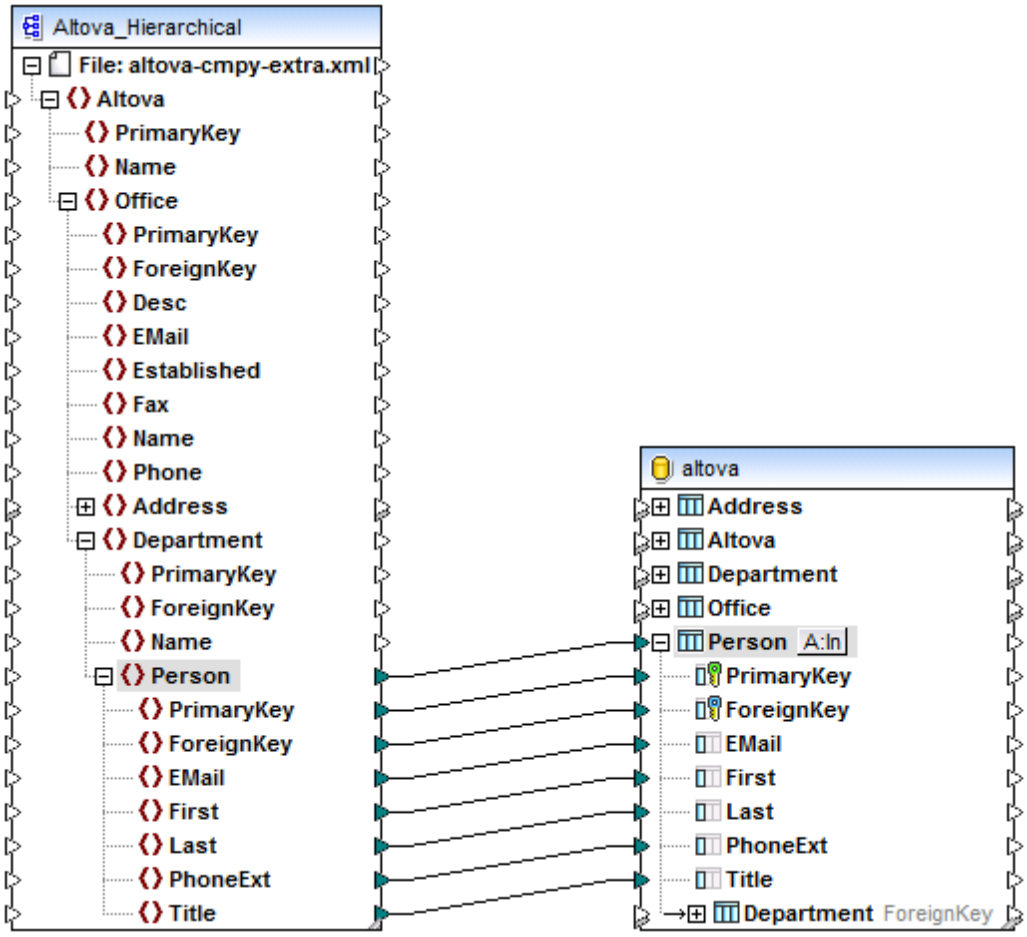
- On the **Insert** menu, click **XML Schema/File**, and browse for **Altova_Hierarchical.xsd**. When prompted to supply an instance file, browse for **altova-cmpy-extra.xml**.

Step 2: Insert the target database

- On the **Insert** menu, click **Database**, and go through the wizard steps to connect to **altova.mdb** (see Example: Adding the "altova.mdb" Database to the Mapping).

Step 3: Draw the connections

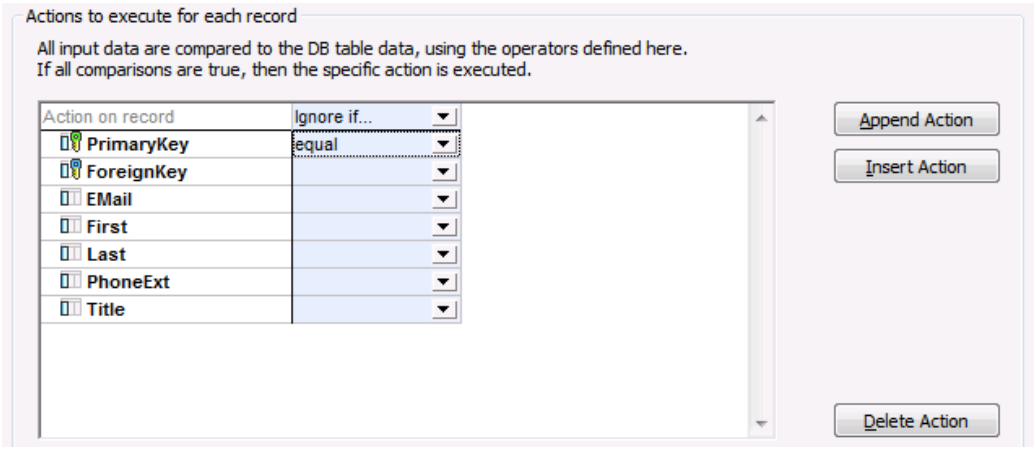
- Draw the mapping connections as shown below.



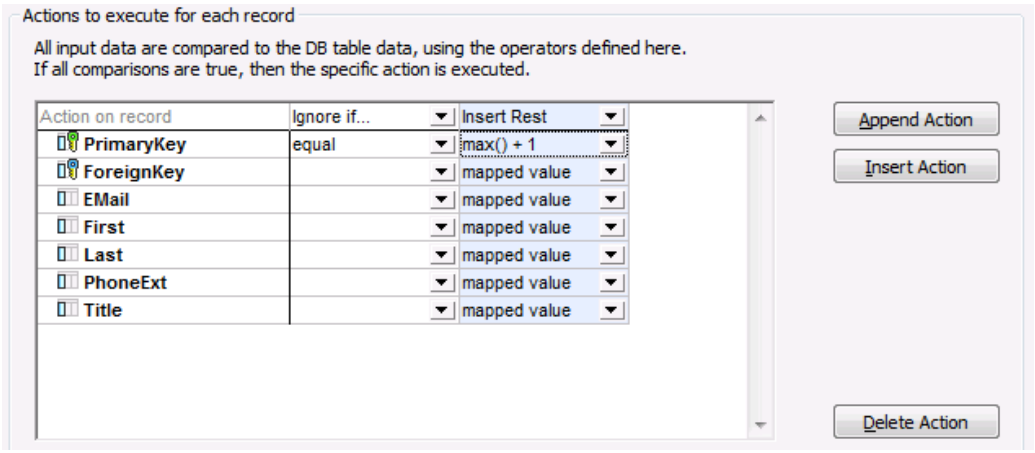
Step 4: Configure the "Ignore if... Insert Rest" actions

1. On the target component, click the **Action: Insert** (A:In) button next to the "Person" table.
2. Next to **Action on record**, select **Ignore if...** . This changes the database table action to a conditional ignore action. That is, the current record will only be ignored when a condition is satisfied (see next step).
3. Next to **PrimaryKey**, select the value **equal**, as shown below. This defines the ignore condition: that is, the database record will be ignored only when its `PrimaryKey` value is equal to the `PrimaryKey` value coming from the mapping.

© 2018 Altova GmbH



4. Click **Append Action**. This adds a new action to the right of the existing **Ignore If** action. Configure the new action as **Insert Rest**, with the primary key set to **max() + 1**, as shown below:



In the image above, the database table actions have been configured in accordance with the goals of the mapping. That is, only when the **Ignore If**... condition is satisfied will the record be skipped; otherwise, it will be inserted. The option "mapped value" specifies that values from the mapping will be used to populate all fields of the record. The option **max() + 1** generates a unique, new primary key value for the record.

5. Click **OK** to close the dialog box. Notice that, back on the mapping, the **Action: Insert** A:In button has now changed to an **Action: Ignore; Insert** ( ) button. This indicates that both the ignore and insert actions are configured for this table.

Step 5: Preview the mapping and update the database

Click the **Output** tab to preview the mapping. A SQL script is generated, containing actions to be executed against the database. The script has not modified the database yet; it is only for preview. To run the script against the database:

- On the **Output** menu, click **Run SQL-Script**.

Note: Running the SQL script directly from MapForce is just one of the ways to update the database, see also Executing Mappings Which Modify Databases.

If you open the "Person" table in the DB query tab of MapForce (see Browsing and Querying Databases), you can see the result of the mapping as follows:

- All database records which had corresponding primary keys in the XML file satisfied the **Ignore if...** and remained unaffected. Examples are records with primary key 1, 2, 3, 4, and 5.
- All database records which had no corresponding keys in the XML file did not satisfy the **Ignore if...** condition but nevertheless remained unaffected. Examples are records with primary key 6, 7, 8, 9, 10, 11, 12, and 13.
- New records have been inserted to the "Person" table (where key did not already exist in the database). Examples are records with primary key 30 and 31 in the source XML file. These were inserted into the database with the new primary key 22 and 23, respectively.

	PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt	Title
1	1	1	v.callaby@nanonull.com	Vernon	Callaby	582	Office Manager
2	2	1	f.further@nanonull.com	Frank	Further	471	Accounts Receivable
3	3	1	l.matise@nanonull.com	Loby	Matise	963	Accounting Manager
4	4	2	j.firstbread@nanonull.com	Joe	Firstbread	621	Marketing Manager Europe
5	5	2	s.sanna@nanonull.com	Susi	Sanna	753	Art Director
6	6	3	f.landis@nanonull.com	Fred	Landis	951	Program Manager
7	7	3	m.landis@nanonull.com	Michelle	Butler	654	Software Engineer
8	8	3	t.little@nanonull.com	Ted	Little	852	Software Engineer
9	9	3	a.way@nanonull.com	Ann	Way	951	Technical Writer
10	10	3	l.gardner@nanonull.com	Liz	Gardner	753	Software Engineer
11	11	3	p.smith@nanonull.com	Paul	Smith	334	Software Engineer
12	12	4	a.martin@nanonull.com	Alex	Martin	778	IT Manager
13	13	4	g.hammer@nanonull.com	George	Hammer	223	Web Developer
14	14	4	j.band@nanonull.com	Jessica	Bander	241	Support Engineer
15	15	4	l.king@nanonull.com	Lui	King	345	Support Engineer
16	16	5	s.meier@nanonull.com	Steve	Meier	114	Office Manager
17	17	5	t.bone@nanonull.com	Theo	Bone	331	Accounts Receivable
18	18	6	m.nafta@nanonull.com	Max	Nafta	122	PR & Marketing Manager US
19	19	7	v.bass@nanonull.com	Valentin	Bass	716	IT Manager
20	20	7	c.franken@nanonull.com	Carl	Franken	147	Support Engineer
21	21	7	m.redgreen@nanonull.com	Mark	Redgreen	152	Support Engineer
22	22	1	c.Cicada@microtech.com	Camilla	Cicada	765	HR
23	23	1	c.corrigan@microtech.com	Carol	Corrigan	629	Admin

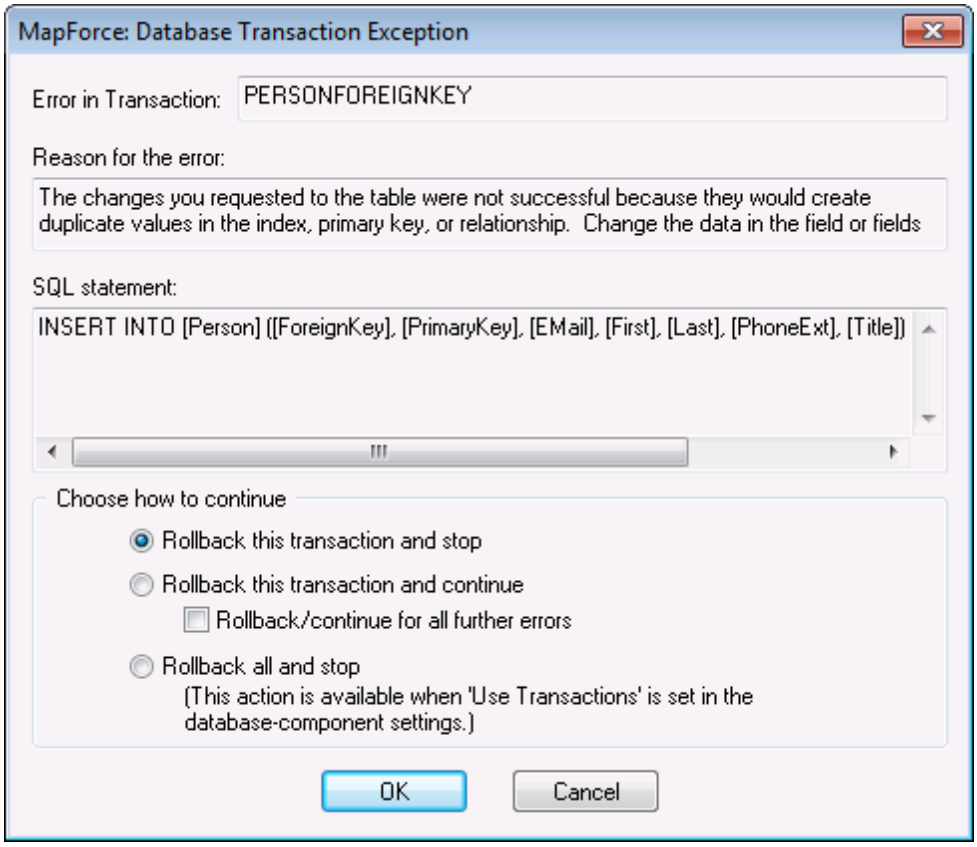
The "Person" table after updating the database

© 2018 Altova GmbH

7.2.2.11.1.9 Revertir transacciones

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

Transaction rollback is a feature that enables you to decide what should happen when a database exception occurs for whatever reason. Namely, when you attempt to run a SQL script from MapForce, and an error occurs, a dialog box such as the one below opens, prompting you to choose how to continue. Because there might be SQL statements (transactions) that were already executed successfully before the exception occurred, you can choose to roll back (undo) all previously executed transactions, or only the current one.



Database Transaction Exception dialog box

To be able to roll back unsuccessful database actions, you must enable transactions first. You can enable database transactions as follows:

- At database level. To enable transactions at database level, select the **Use transactions** check box from the Database Component Settings dialog box (see also Database Component Settings).
- For each individual table. To enable transactions for each individual table, select the **Use**

transactions check box from the Database Table Actions dialog box (see also Database Table Actions Settings).

You can enable transactions at database level without enabling them at table level, and vice versa. Be aware that, when a database exception occurs, the available rollback options depend on how transactions were enabled:

- If transactions were not enabled at all, and an error occurs, execution stops at the point the error occurs. All previously successful SQL statements are executed and the results are stored in the database. It is not possible to roll back any transactions.
- If transactions were enabled at table level but not at database level, execution stops at the point the error occurs. In this case, the option **Rollback all and stop** is disabled. You can roll back only the current transaction for that specific table, and either continue with the mapping execution or stop running the mapping completely.
- If transactions were enabled at the database level only, execution stops at the point the error occurs. All previously successful SQL statements are rolled back. No changes are made to the database.
- If transactions were enabled at both the database level and table level, execution stops at the point the error occurs. In this case, select **Rollback all and stop** to roll back all previously successful SQL statements for the entire database. To roll back only the last transaction, and continue executing the mapping, select **Rollback this transaction and continue**.

Note that the Database Transaction Exception dialog box is displayed only when you run the mapping directly from MapForce, using the **SQL | Run SQL-Script** menu command. When the mapping is executed in another environment (either by the generated code, or by MapForce Server), and an error is encountered, an automatic rollback of the erroneous transaction occurs. In this case, any successful SQL statements that were previously executed are rolled back only if you enabled transactions at the database level, as explained above.

If you click **Cancel** on the dialog box, the execution rolls back the current SQL statement and stops.

7.2.2.11.1.10 Inserciones masivas

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

The **Use Bulk Transfer** option allows you to insert data at very high speed from a MapForce component (TXT, CSV, DAT, etc.) into a database table. Using this option dramatically speeds up the Insert process, as only one statement needs to be executed instead of many.

The **Use Bulk Transfer** option can be enabled in MapForce, at mapping design time, as shown below. A mapping where this option is enabled can be executed in MapForce, but no bulk insert applies at this stage. The actual bulk transfer of data occurs when the mapping is run by MapForce Server.

Bulk transfer is supported when the following conditions are true:

© 2018 Altova GmbH

- The mapping transformation language is set to BUILT-IN. For further information, see Selecting a Transformation Language.
- The mapping is run by MapForce Server (either standalone or under FlowForce Server management). This means that the mapping must be either compiled to .mfx format or deployed to FlowForce Server. For further information, see Compiling Mappings to MapForce Server Execution Files and Deploying Mappings to FlowForce Server.
- The MapForce Server license is not limited to "single thread execution" on a multi-core machine. That is, the **Limit to single thread execution** check box in the "Server Management" tab of Altova LicenseServer must be inactive.
- The database action is "Insert all", see also Database Table Actions Settings.
- The table into which the data is to be bulk loaded must be a "leaf" table, that is, on the lowest hierarchy of the database. There should not be any related tables, views, or stored procedures referencing the table in the mapping.
- The database driver supports bulk insert on WHERE conditions.

The following table summarizes support for bulk inserts depending on the database kind and the driver used.

	ADO	ODBC	JDBC	ADO.NET	Native
Access	No	No	n/a	n/a	n/a
DB2	No	Yes	Yes	Yes	n/a
Firebird	n/a	Yes	Yes	No	n/a
Informix	No	Yes	Yes	Yes	n/a
iSeries	No	Yes	Yes	Yes	n/a
MySQL	n/a	Yes*	Yes	Yes	n/a
Oracle	No	Yes	Yes	Yes	n/a
PostgreSQL	n/a	Yes	Yes	n/a	Yes
Progress	n/a	Yes	Yes	n/a	n/a
SQL Server	Yes	Yes	Yes	Yes	n/a
SQLite	n/a	n/a	n/a	n/a	No
Sybase	No	Yes	Yes	n/a	n/a

** MySQL version 5 or later is required.*

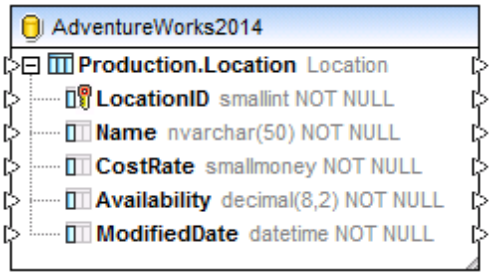
This example shows you how create a mapping which bulk loads data from a sample **source.txt** file into a target database. The example uses SQL Server 2014 and the AdventureWorks 2014 database. The latter can be downloaded from the CodePlex website (https://msftdbprodsamples.codeplex.com/).

```
Location A,15.3,39
Location B,46,34
Location C,56.33,0
Location D,0,399
Location E,0,97.43
```

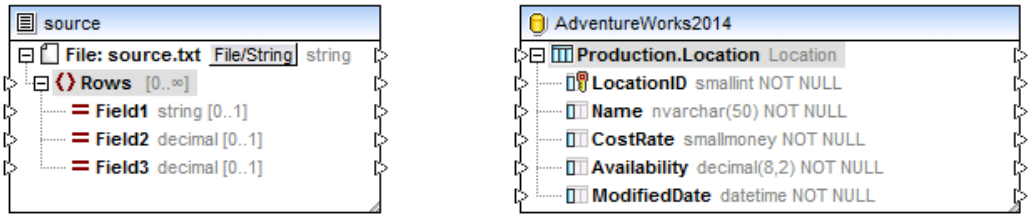
source.txt

To define a bulk insert:

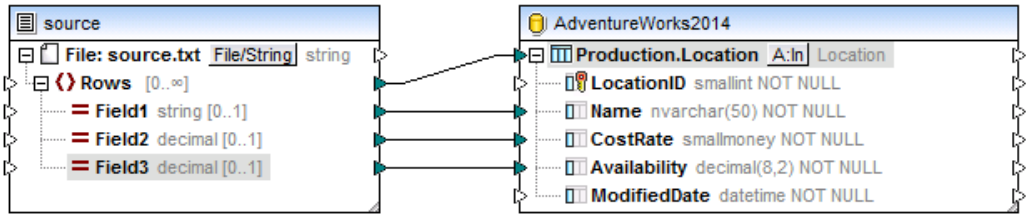
1. Set the transformation language to BUILT-IN ( ).
2. Connect to the AdventureWorks 2014 database and add the "Production.Location" table to the mapping. For more information, see Adding Databases to the Mapping.



3. On the **Insert** menu, click **Text**, and add a source text file (such as the **source.txt** sample above) to the mapping. For more information, see CSV and Text Files. Make sure that the data types of both the source and the target components are compatible. Data types are visible on component when the **Show Data Types** ( ) toolbar button must be enabled.

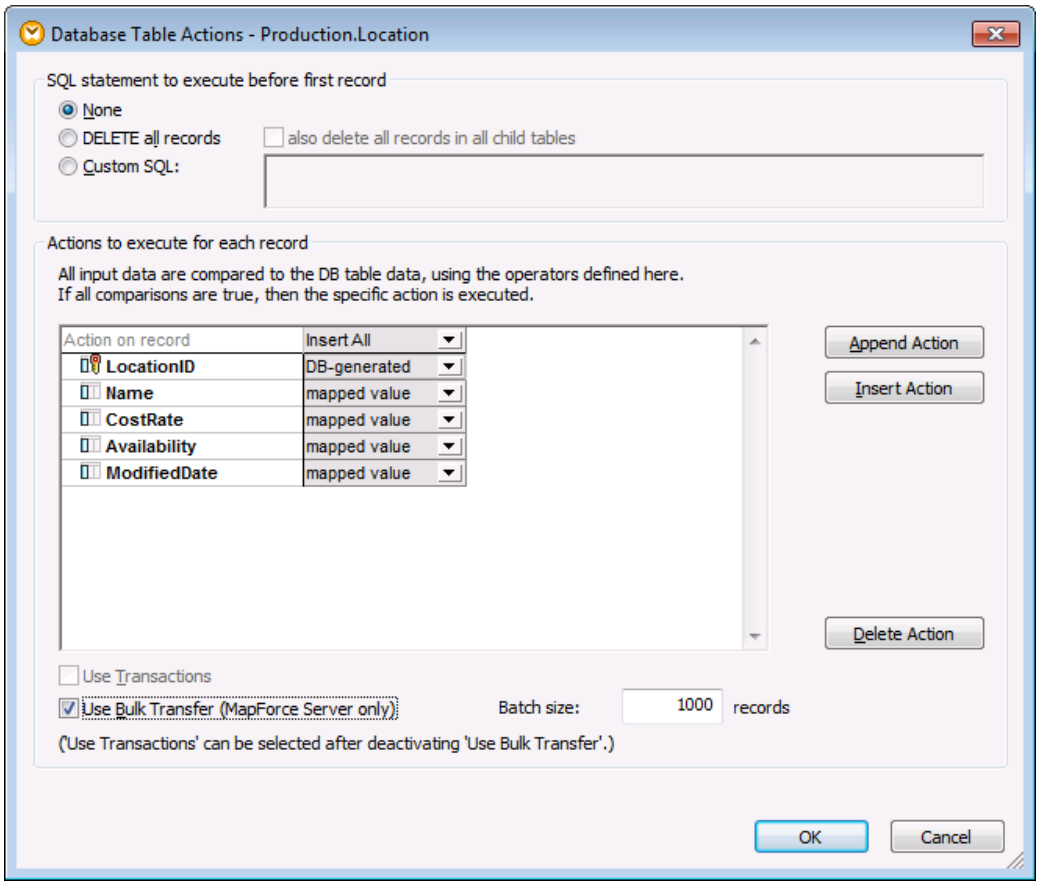


4. Draw the mapping connections as shown below. Note that the **Database Actions** button A:In is now visible to the right of the table name.



5. Click the **Database Actions** button (A:In), select the **Use Bulk Transfer** check box, and click OK to confirm.

 © 2018 Altova GmbH



In the dialog box above, notice that the "Action on record" is "Insert All". The **Batch size** field defines the number of records to be inserted per action.

Note: When the **Use Bulk Transfer** option is enabled, the **Use Transactions** option becomes disabled, and vice versa. If you want to enable transaction processing, click to clear the **Use Bulk Transfer** check box.

Now that bulk insert is enabled, the next step is to execute the mapping in MapForce Server (either standalone or under FlowForce Server management). For further information, see Compiling Mappings to MapForce Server Execution Files and Deploying Mappings to FlowForce Server.

7.2.2.11.1.11 Gestionar valores null en acciones de tabla de BD

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

When a mapping updates a target database by means of table actions such as "Ignore If", "Update If", "Delete If", MapForce compares the source data against the target data and generates internal database update queries as a result. (These internal queries are available for

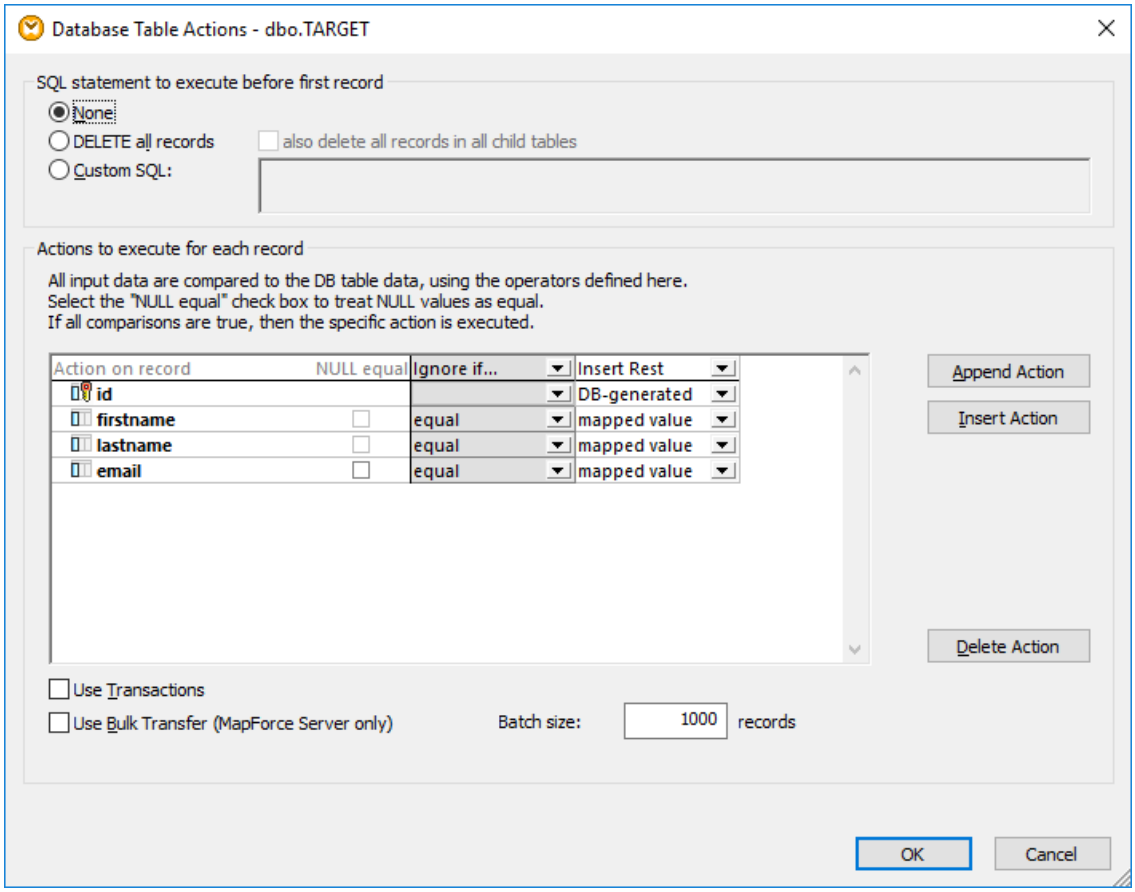
preview in the **Output** pane of MapForce, see Executing Mappings Which Modify Databases). The generated queries reflect the comparison conditions that were defined from the "Database Table Actions" dialog box (see also Database Table Actions Settings).

Null comparisons are a complex subject in the context of SQL and databases, in the sense that there is no commonly accepted way to compare null values across various database types. From a MapForce perspective, it is possible to configure a database mapping so that data comparison is done in a NULL-aware manner, according to rules applicable to the database kind involved in the mapping. "NULL-awareness" means that any `NULL` values will be treated as such for the scope of data comparison (otherwise, you may get undesired results from the mapping). NULL-awareness should be enabled if:

1. The "Database Table Actions" dialog box contains "Ignore if", "Update if", "Delete if" actions, and
2. These actions are taken against records that may contain `NULL` values, and
3. NULL values in the source table must be treated as equal with NULL values in the target table.

By default, NULL-awareness is disabled. If the conditions above are true and NULL-awareness is disabled, there may be instances where the target database table is not updated as expected (for example, more rows are inserted or updated than necessary). This happens because `NULL` values affect the data comparison and could produce undesired results. To prevent this from happening, select the check box next to each nullable field (`email`, in the image below) from the "Database Table Actions" dialog box. Be aware that the check box can be selected only for fields which are nullable, and when at least one table action has an "equal" or "equal (ignore case)" condition.

© 2018 Altova GmbH



Database Table Actions dialog box

Example

To better understand NULL awareness in mappings, let's analyze an example where comparison of null data occurs. This example uses a Microsoft SQL Server database; however, it is also applicable for any other supported database type. Optionally, if you have Microsoft SQL Server, you can create the tables and data used in this example by running the following database script: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\CreateNullableFields.sql**.

For convenience, the database tables are illustrated below. Both tables store people data and have the same columns. Also, the column `email` can contain null data in both tables.

```
+----+-----------+-----------+--------------------------+
| id | firstname | lastname  | email                    |
+----+-----------+-----------+--------------------------+
| 1  | Toby      | Hughey    | t.hughey@nanonull.com    |
| 2  | Mia       | Dahill    | NULL                     |
| 3  | Fred      | Weinstein | f.weinstein@nanonull.com |
+----+-----------+-----------+--------------------------+
```

The SOURCE table

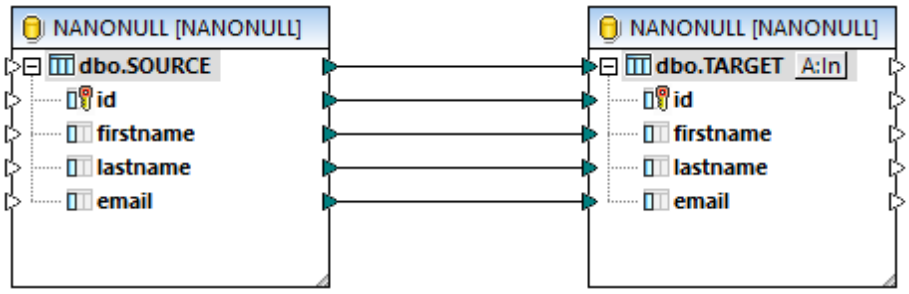
```
+----+-----------+-----------+--------------------------+
| id | firstname | lastname  | email                    |
+----+-----------+-----------+--------------------------+
| 1  | Mia       | Dahill    | NULL                     |
| 2  | Fred      | Weinstein | f.weinstein@nanonull.com |
+----+-----------+-----------+--------------------------+
```

The TARGET table

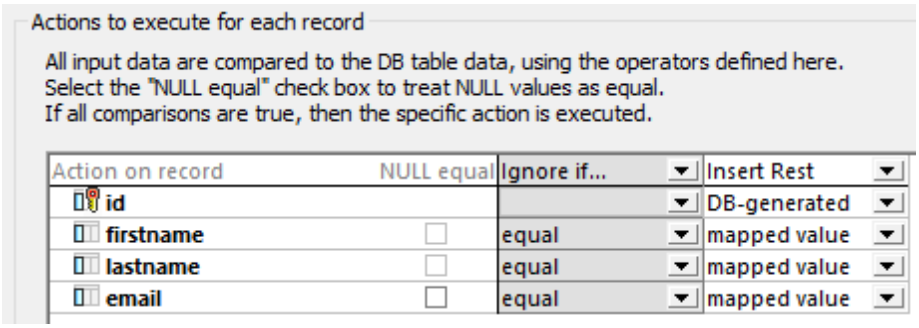
Let's suppose your task is to merge data from the SOURCE table into the TARGET table. Only the new records must be inserted into the TARGET table (in this example, "Tobie Hughey"). The records which exist in both tables ("Mia Dahill" and "Fred Weinstein") must be ignored.

The task can be accomplished as follows.

1. On the **Insert** menu, select **Database**. Follow the wizard steps to connect to the database (see also Connecting to a Database). When prompted to add database objects, select the table SOURCE.
2. On the **Insert** menu, select **Database**. Connect to the database again and add the table TARGET to the mapping.
3. Draw the mapping connections between the source and target components.



4. Click the **Action:Insert** A:In button and configure the database table actions as follows:



As illustrated above, a combination of "Ignore if.. Insert Rest" actions are defined. This configuration means that, for each record, the mapping checks if:

- `firstname` in the source is equal to `firstname` in the target, AND

 © 2018 Altova GmbH

- `lastname` in the source is equal to `lastname` in the target, AND
- `email` in the source is equal to `email` in the target.

If all the conditions above are true, the record is ignored (according to the requirement). Otherwise, a new record is inserted into the target table. The `id` of the new record is generated by the database, while the other fields (`firstname`, `lastname`, `email`) are populated with values mapped from the source.

Importantly, the check box next to `email` enables or disables NULL-aware comparison for this field. This check box must be selected, because `email` can contain NULL values (namely, "Mia Dahill" has a NULL email address). To see the role played by this check box, try updating the database two times: first time, with the check box selected, and a second time with the cleared check box.

To update the database, click the **Output** tab and run the menu command **Output | Run SQL-Script**.

If the check box is selected, MapForce has explicit indication that you want to treat the NULL fields as equal. Therefore, the record "Mia Dahill" is not inserted in the target table, which is the intended result.

If the check box is not selected, the record "Mia Dahill" is inserted in the target table (despite that fact that it exists already), which is not the intended result. The reason is that no explicit indication was given to MapForce that you want to treat NULL values as equal. A similar situation would occur if you ran the following query against the database (this query retrieves no records because the NULL value is compared with the "=" operator so it is not NULL aware):

```
SELECT firstname, lastname, email FROM TARGET WHERE firstname = 'Mia' AND
lastname = 'Dahill' AND email = NULL;
```

In order to be NULL aware, the query above would have to be rewritten as follows:

```
SELECT firstname, lastname, email FROM TARGET WHERE firstname = 'Mia' AND
lastname = 'Dahill' AND email IS NULL;
```

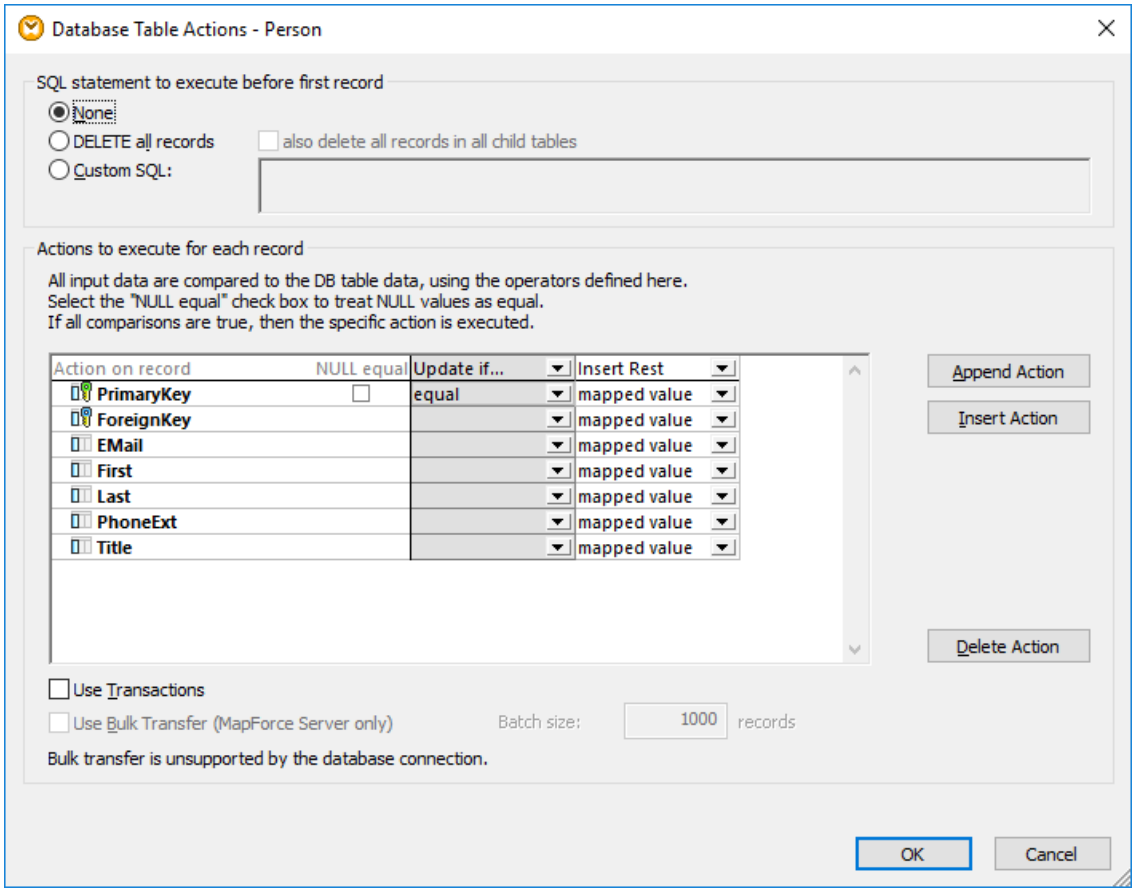
Note: The queries above are only for illustrative purposes and do not reflect the actual syntax of internal queries generated by MapForce. When NULL awareness is enabled, MapForce adapts the syntax of generated queries according to the database type (since various database vendors have different approaches to handling null comparisons).

7.2.2.11.1.12 Configurar acciones de tabla de BD

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

Whenever you create a mapping connection to a database table, a Database Actions button appears next to the affected table. Clicking this button opens Database Table Actions dialog box, from where you can configure the database insert, update, and delete actions, as well as other

options.



Database Table Actions dialog box

Below is a description of the settings available on the Database Table Actions dialog box.

SQL statement to execute before first record

In this group box, you can define SQL statements that are executed before any actions defined under **Actions to execute for each record**. Select the desired radio button:

- **None** — No action is carried through. This is the default setting.
- **DELETE all records** — All records from the selected table are deleted before any specific table action defined in the **Actions to execute for each record** group box is performed. Activate the **also delete all records in all child tables** check box if you also want to get rid of the data stored in child tables of the selected table. For an example, see Inserting Data into Multiple Linked Tables.
- **Custom SQL** — Write a custom SQL statement to affect the complete table. Note that support for multiple SQL statements in one query depends on the database, connection method, and the driver used.

© 2018 Altova GmbH

Actions to execute for each record

This group of settings specify the database actions that are to take place against this table when the mapping runs. To manage table actions, click the **Append Action**, **Insert Action**, or **Delete Action** buttons. Multiple actions can be defined if necessary (for example, an "Update if..." action followed by an "Insert Rest" action.

The defined table actions are processed from left to right. In the example above, the "Update if..." action is processed first. If the update condition is not satisfied then the following action is processed (in this example, the "Insert Rest" action). Note the following:

- All the defined conditions of one action must be satisfied for the table action to be executed. When this is the case, all those fields are updated where a connection exists between the source and target items on the mapping. Any subsequent table actions (to the right of an action whose condition matched) are ignored for that record.
- If the defined condition is not satisfied, then the table action is skipped, and the next action (to the right) is processed.
- If none of the conditions are satisfied, no table action takes place.

> Any table actions defined after "Insert All" or "Insert Rest" actions will never be executed, because no column conditions exist for insert actions. A dialog box appears if this is the case, stating that the subsequent table action columns will be deleted.

In the "NULL Equal" column, the check box next to each record, where applicable, enables you to explicitly instruct MapForce that the column may contain NULL values and should be treated as such (see also Handling Nulls in Database Table Actions).

When the mapping updates a table which has foreign key relationships to other tables, the following options can be used:

Delete data in child tables	This option is meaningful when you select the "Update if..." action for a parent table. It might be necessary if the number of records in the source file is different from the number of records in the target database, and you want to keep the database synchronized (no orphaned data in child tables). See also Options for Child Tables When Updating a Parent Table.
Ignore input child data	Use this option when you want to update a target parent table, without affecting any of the child tables/records of that table. See also Options for Child Tables When Updating a Parent Table.

For examples which illustrate various combinations of actions, see:

- Inserting Data into a Table
- Inserting Data into Multiple Linked Tables
- Updating a Table
- Options for Child Tables When Updating a Parent Table
- "Update if... Insert Rest" Action
- "Delete if..." Action
- "Ignore if..." Action

Use Transactions

Enables database transactions for this particular table action. For more information, see Using Transaction Rollback.

Use Bulk Transfer

Enables bulk transfer (multiple INSERT statements as one query). Bulk transfer is supported if the mapping is executed by MapForce Server and the database action is "Insert All". For more information, see Bulk Inserts (MapForce Server).

7.2.2.11.1.13 Ejemplo: asignar datos XML a datos SQLite

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

This example walks you through the steps required to create a MapForce mapping which reads data from an XML file and writes it to a SQLite database. The example is accompanied by a sample mapping design (.mfd) file. If you want to look at the sample file before starting this example, you can open it from the following path: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\XMLtoSQLite.mfd**.

The goal of the example is to insert data from an XML file into a SQLite database. To accomplish the goal of the example, you will need an empty SQLite database to which data will be written. As illustrated below, you can create and explore the SQLite database either with Altova DatabaseSpy, or with the command-line shell available from the official SQLite website.

To create the SQLite database:

If DatabaseSpy is installed on your computer (either standalone or as part of Altova MissionKit), you can create the new SQLite database as follows:

1. Run DatabaseSpy.
2. On the **File** menu, click **Create a Database Connection**.
3. Click **Connection Wizard**, and then click **SQLite**.
4. Click **Create a new SQLite database**, enter **c:\sqlite\articles.sqlite** as path, and then click **Connect**.
5. When prompted to set a data source name, leave the default name as is.
6. Open a new SQL editor (**Ctrl+N**) and run the following query against the database:

```
create table articles (number smallint, name varchar(10), singleprice
real);
```

Otherwise, follow the steps below to create the database:

© 2018 Altova GmbH

1. Download the SQLite command-line shell for Windows from the SQLite download page (http://www.sqlite.org/download.html) and unpack the .zip archive to a directory on your local machine (for the scope of this example, use **c:\sqlite**).
2. Run **c:\sqlite\sqlite3.exe** and enter the following statement:

   ```
   create table articles (number smallint, name varchar(10), singleprice
   real);
   ```

   This creates the table `articles` in the in-memory database. The table `articles` consists of three columns: *number*, *name*, and *singleprice*. The purpose of these columns is to store data from the elements with the same name defined in the source XML schema. Each column is declared with a data type suitable for the data expected to be stored in that column.
3. Run the command:

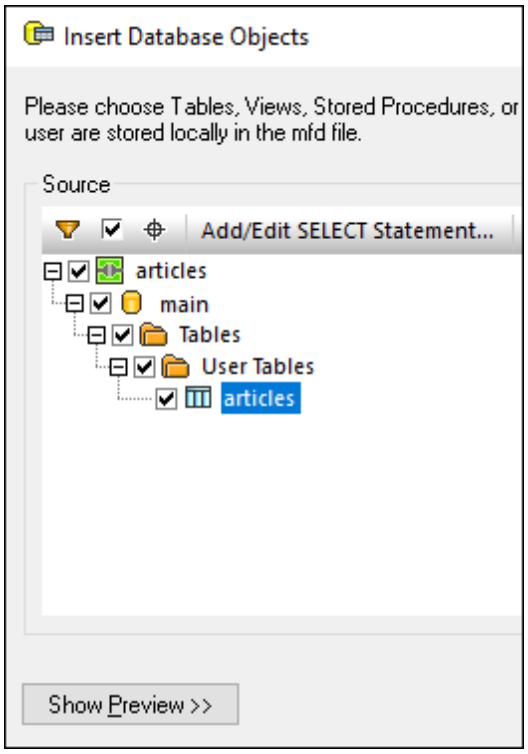
   ```
   .save articles.sqlite
   ```

   This saves the in-memory database to the current working path: **c:\sqlite\articles.sqlite**. Note that you will need to refer to this path in subsequent steps.

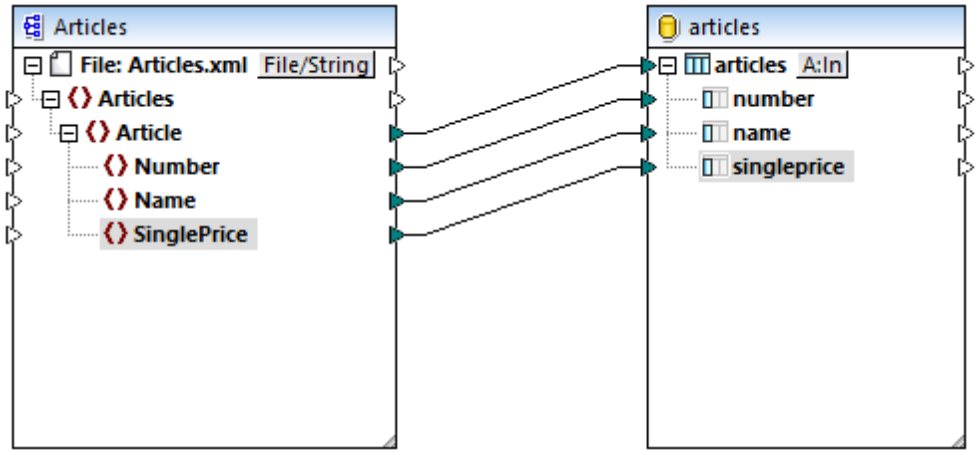
You have now finished creating the sample SQLite database required for this example.

To create the XML to SQLite mapping design:

1. Run MapForce and make sure that the transformation language is set to BUILT-IN (use the menu command **Output | Built-in Execution Engine**).
2. Add to the mapping area the file **Articles.xml** located in the **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial** folder (use the menu command **File | Insert XML Schema/File**).
3. Add to the mapping area the database **articles.sqlite** created in previous steps (use the menu command **File | Insert Database**), and then select SQLite.
4. Click **Connect**. When prompted to choose the database objects, select the `articles` table.

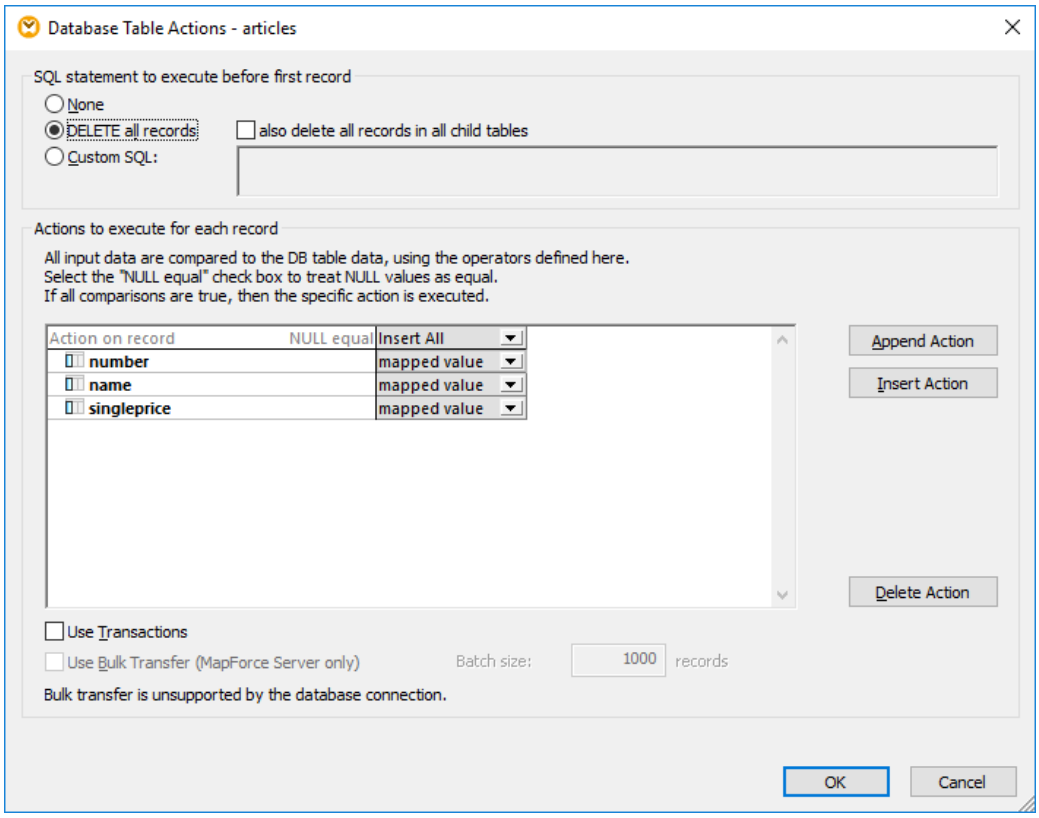


5. Draw the connections as shown below:



6. Click the **A:In** button on the database component and select the **Delete All records** option. This ensures that, every time the mapping is executed, all existing database rows are first deleted, in order to prevent duplication.

 © 2018 Altova GmbH



7. Click the **Output** tab of the main mapping window. MapForce executes the mapping using the built-in execution engine and displays the create SQL query in the Output window.

```
1   /*
2   The following SQL statements are only for preview and may not be executed in another SQL query tool!
3   To execute these statements use function "Run SQL-script" from menu "Output".
4   Connect to database using the following connection-string:
5   C:/sqlite/articles.sqlite
6   */
7
8   PRAGMA foreign_keys = ON;
9
10  DELETE FROM "articles"
11
12  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (1, 'T-Shirt', 25)
13
14  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (2, 'Socks', 2.3)
15
16  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (3, 'Pants', 34)
17
18  INSERT INTO "articles" ("number", "name", "singleprice") VALUES (4, 'Jacket', 57.5)
19
```

8. Run the SQL script to populate the database (use the menu command **Output | Run SQL-Script)**. If MapForce does not encounter any runtime errors, the records are inserted into the SQLite database.

```
/*
The following SQL statements were executed during "Generate output" function.
Every single result is written right to the "-->>>" string.
These statements are only for preview and may not be executed in another SQL query tool!
The database was connected using the following connection-string:
C:/sqlite/articles.sqlite
*/

PRAGMA foreign_keys = ON;
-->>> OK. 0 row(s).

DELETE FROM "articles"
-->>> OK. 0 row(s).

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (1, 'T-Shirt', 25)
-->>> OK. 1 row(s).

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (2, 'Socks', 2.3)
-->>> OK. 1 row(s).

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (3, 'Pants', 34)
-->>> OK. 1 row(s).

INSERT INTO "articles" ("number", "name", "singleprice") VALUES (4, 'Jacket', 57.5)
-->>> OK. 1 row(s).

```

To check whether data was correctly inserted into the SQLite database:

1. Click the **DB Query** tab of MapForce.
2. Select the **articles** entry from the drop-down list at the top of the **DB Query** pane.
3. Enter the following query in the SQL Editor:

```
select * from articles;
```

Alternatively, follows the steps below:

1. Run the file **c:\sqlite\sqlite3.exe** and open the database with the command:

```
.open articles.sqlite
```

2. Run the following select statement:

```
select * from articles;
```

Regardless of the approach you choose to select the table data (MapForce or SQLite shell), the query should returns four rows now. This corresponds to the number of records in the source XML file, which was the intended goal of this example.

7.2.3 Combinar datos de BD

En las asignaciones de datos que leen datos de bases de datos podrá combinar objetos de la base de datos, como tablas o vistas, con solo añadir un componente de combinación a la

© 2018 Altova GmbH

asignación. Por ejemplo, puede combinar datos de dos o más tablas vinculadas por medio de relaciones de clave foránea, la manera más corriente de almacenar datos en bases de datos relacionales. El resultado sería idéntico al que se obtendría si se ejecutara una consulta SQL que uniera dos o más tablas por medio de una operación INNER JOIN en la base de datos.

Dependiendo del tipo de datos que se conecten al componente de combinación, la operación de combinación puede tener lugar en el modo estándar (no SQL) o en el modo SQL. Las combinaciones en modo no SQL las lleva a cabo MapForce, mientras que las combinaciones en modo SQL las lleva a cabo la base de datos en la que lee datos la asignación.

Las combinaciones en modo no SQL son más flexibles porque admiten más tipos de componentes de entrada (p. ej. la combinación puede ser entre tablas de bases de datos distintas o entre estructuras XML y tablas de base de datos). Para ver un ejemplo de combinación no SQL consulte el apartado Ejemplo: combinar estructuras XML. Por otro lado, las combinaciones no SQL exige al motor de asignación comparaciones que requieren un gran uso de memoria (porque el número total de comparaciones representa la combinación cruzada o producto cartesiano de todas las estructuras combinadas). Este proceso se lleva a cabo a gran velocidad por lo general y no se aprecia en las asignaciones que no hacen un uso intensivo de los datos. Sin embargo, si los orígenes de datos combinados están compuestos por un gran número de registros, puede que la asignación necesite mucho tiempo para ejecutarlos. Si la asignación que tiene entre manos debe procesar un gran número de registros, considere obtener una licencia de MapForce Server Advanced Edition, que optimiza especialmente las combinaciones para acelerar la ejecución de asignaciones de datos.

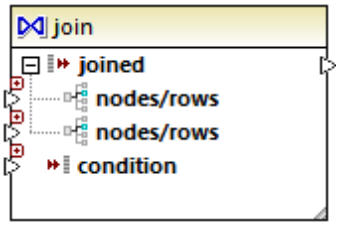
Las combinaciones en modo SQL solamente aceptan determinados objetos de base de datos como entrada (p. ej. tablas o vistas), así que no son tan flexibles como las combinaciones no SQL. Sin embargo, ofrecen un mayor rendimiento de la asignación porque la base de datos las ejecuta de forma nativa. Para más información consulte el apartado Combinaciones en modo SQL.

Nota: los componentes de combinación no son el único método posible para combinar tablas o vistas de base de datos. También puede realizar combinaciones de datos de base de datos con ayuda de instrucciones SQL SELECT (*véase Instrucciones SQL SELECT como tablas virtuales*). La principal diferencia entre las instrucciones SQL SELECT y los componentes de combinación es que aquellas se escriben a mano y permiten mayor flexibilidad. Los componentes de combinación son una alternativa sencilla para quienes no dominen SQL.

Para agregar un componente de combinación:

1. Active el lenguaje de transformación BUILT-IN (haciendo clic en el botón 

de la barra de herramientas o con el comando de menú **Resultados | Motor de ejecución integrado**).
2. Haga clic en el comando de menú **Insertar | Combinación** o en el botón **Combinación**  de la barra de herramientas. El componente de combinación aparece en la asignación y por defecto acepta datos de dos estructuras, así que tiene dos entradas `nodes/rows`. Si lo necesita, puede agregar entradas nuevas en la combinación con ayuda del botón **Agregar entrada** (*véase Combinar tres o más estructuras*).



3. Conecte las estructuras que se deben combinar con las entradas `nodes/rows` del componente de combinación.
4. Añada la condición para la combinación (o varias condiciones). Esto se consigue haciendo clic con el botón derecho en el componente de combinación y seleccionando **Propiedades**. También puede añadir condiciones de combinación desde la asignación directamente: conecte el resultado booleano de una función con la entrada `condition` del componente de combinación. Cuando se están combinando tablas de base de datos, MapForce puede crear condiciones de combinación automáticamente (*véase Agregar condiciones de combinación*).

Notas:

- Se admite el uso de componentes de combinación cuando el lenguaje de destino de la asignación es el motor de ejecución integrado (BUILT-IN). No es compatible con la generación de código C#, C++ o Java.
- Cuando una estructura no es una fuente de entrada válida ni compatible para la combinación, MapForce emite sugerencias en la asignación directamente o, cuando se valide la asignación, en la ventana Mensajes (*véase Validar asignaciones*).
- No se deberían conectar componentes de combinación con parámetros de entrada ni resultados de funciones definidas por el usuario inline. Si existen dichas conexiones, se producirán errores de validación.
- Cuando conecte componentes de base de datos (tablas o vistas, por ejemplo) a un componente de combinación directamente, en la esquina superior derecha del componente de combinación se habilitará el botón **Modo SQL** SQL. Si se activa, este botón ofrece funciones SQL especiales para trabajar con la operación de combinación (*véase Combinaciones en modo SQL*).
- No se puede conectar la salida del nodo `joined` con otro componente de combinación, pero si lo necesita puede conectar un resultado parcial de una combinación con otra.

7.2.3.1 Combinaciones en modo SQL

Cuando se conectan los componentes de base de datos apropiados (p. ej. tablas o vistas) a un componente de combinación, en la esquina superior derecha del componente de combinación aparece el botón **Modo SQL** SQL. Cuando se habilita el modo SQL, la operación de combinación la lleva a cabo la base de datos de la que lee datos la asignación. En otras palabras, MapForce enviará una consulta a la base de datos compuesta con la sintaxis SQL adecuada para seleccionar y combinar datos de todas las tablas que participan en la combinación. Y, lo que es más importante, no será necesario escribir código SQL porque la consulta se produce en base al diseño visual que tenga el componente de combinación en la asignación de datos (*ver ejemplos más abajo*).

Nota: desde el punto de vista de SQL y de la base de datos, las combinaciones generadas con MapForce siempre son combinaciones internas. Es decir, que el componente de

 © 2018 Altova GmbH

combinación solamente devuelve los registros que cumplen con la condición en ambos conjuntos de datos de entrada.

Para poder habilitar el modo SQL deben cumplirse estos requisitos:

1. Los dos objetos que se deben combinar (tablas o vistas) deben ser de la misma base de datos.
2. Los dos objetos que se deben combinar deben proceder del mismo componente de MapForce (recuerde que puede agregar/quitar objetos de base de datos de un componente haciendo clic con el botón derecho en el componente y seleccionando **Agregar, quitar o editar objetos de la base de datos** en el menú contextual).
3. La condición (o las condiciones) de combinación debe definirse desde las propiedades del componente solamente (haciendo clic con el botón derecho en el componente de combinación y seleccionando **Propiedades**) y no en la asignación (*véase* *Agregar condiciones de combinación*).

Nota: cuando se combinan tablas de base de datos en modo SQL, MapForce crea las condiciones de combinación automáticamente a partir de las relaciones de clave foránea detectadas entre las tablas. Para que MapForce pueda crear condiciones de combinación automáticas las tablas de la base de datos deben estar en una relación secundario/primario en el componente de MapForce (es decir, una tabla debe ser el primario o secundario de otra tabla en el componente). Consulte el apartado Ejemplo: combinar tablas en modo SQL para obtener más información.

4. Las tablas de la base de datos no pueden estar todavía en el contexto de destino actual. Por ejemplo, si en la asignación las tablas se consultan antes de que se realice la operación de combinación, puede que la combinación no sea posible. Para más información consulte Reglas y estrategias de asignación de datos.

El modo SQL se puede controlar y consultar con ayuda del botón **SQL**  situado en la esquina superior derecha del componente de combinación:

 El modo SQL está deshabilitado (la combinación la ejecutará MapForce o, si procede, MapForce Server).

 El modo SQL está habilitado (la combinación la ejecutará la base de datos).

Si falta el botón  , significa que el modo SQL no es relevante o no es compatible con los datos que se pretenden combinar.

En algunos casos es necesario deshabilitar el modo SQL explícitamente, por ejemplo:

- Cuando la asignación necesite condiciones de combinación fuera de las propiedades del componente de combinación (es decir, condiciones definidas en la asignación y conectadas al elemento `condition` del componente de combinación).
- Cuando necesite combinar tablas de bases de datos distintas. En este caso deberá usar una combinación estándar (no SQL).

A menudo las tablas o vistas de base de datos combinadas contienen nombres de campo idéntico en ambas estructuras. Cuando se habilita el modo SQL, estos elementos aparecen en el componente con el prefijo "AS". Por ejemplo, si dos tablas combinadas contienen un campo llamado "id", este campo aparecerá como "id" en la primera tabla combinada y como "id AS id2"

en la segunda tabla combinada. Las tablas combinadas también pueden dar lugar a nombres de alias (p. ej. cuando se combina una tabla consigo misma).

Los nombres de alias de campo o de tabla son importantes para hacerles referencia más adelante en la asignación. Por ejemplo, imagínese que quiere filtrar u ordenar el resultado de la combinación. Para conseguirlo deberá conectar el resultado del componente de combinación a un componente WHERE/ORDER de SQL, donde deberá introducir las cláusulas WHERE y ORDER BY de SQL.

Para hacer referencia al campo desde la cláusula WHERE deberá escribir el nombre de la tabla seguido de un punto y del nombre del campo. Para hacer referencia al alias de una tabla deberá usar el nombre de alias tal y como aparece en el componente de combinación. En la cláusula ORDER BY podrá usar la misma técnica (`tabla.campo`) o escribir solo el nombre de alias de campo (el nombre que aparece después de "AS").

Para ver una asignación que utilice cláusulas WHERE/ORDER de SQL consulte el apartado Ejemplo: combinar tablas en modo SQL.

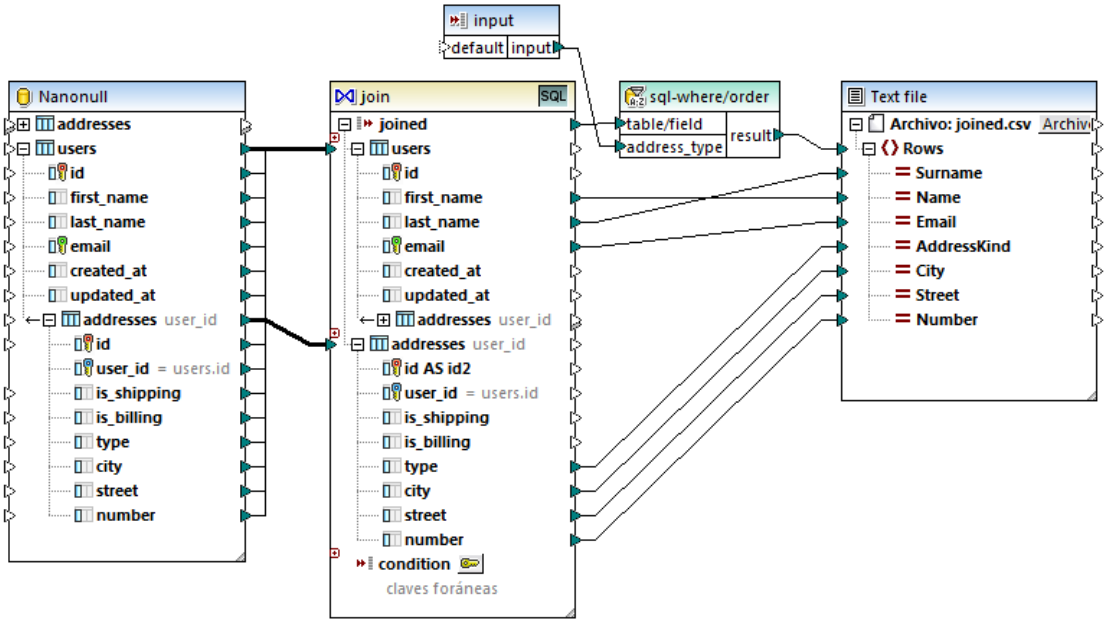
Nota: no está permitido crear componentes WHERE/ORDER de SQL entre una tabla de base de datos y el componente de combinación. Es decir, solamente se pueden añadir después (pero no antes) del componente de combinación. Para más información consulte el apartado Filtrar y ordenar datos de BD (SQL WHERE/ORDER).

7.2.3.2 Ejemplo: combinar tablas en modo SQL

Este ejemplo demuestra cómo combinar datos de dos tablas de base de datos con ayuda de un componente de combinación de MapForce. La operación de combinación se lleva a cabo en modo SQL (*véase Combinaciones en modo SQL*). Recuerde que también se pueden combinar tres o más tablas de una forma muy parecida (*véase Ejemplo: crear un informe CSV a partir de varias tablas*).

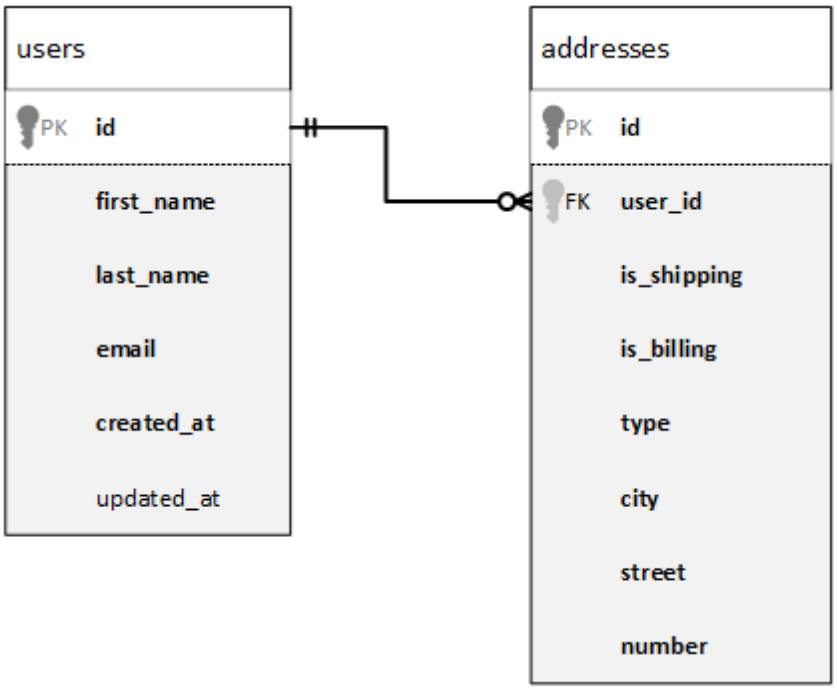
El ejemplo que aparece a continuación se sirve de la asignación de datos **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\JoinDatabaseTables.mfd**.

© 2018 Altova GmbH



JoinDatabaseTables.mfd

El objetivo de la asignación es combinar datos de dos tablas de base de datos de origen en un solo archivo CSV de destino. Como puede ver en el esquema que aparece a continuación, la primera tabla (`users`) almacena direcciones de personas y la segunda (`addresses`) almacena nombres y direcciones de correo de personas. Estas dos tablas están enlazadas por medio de un campo común (el campo `id` de `users` corresponde al campo `user_id` de `addresses`). En bases de datos este tipo de relación recibe el nombre de *relación de clave foránea*.



A continuación puede ver los datos que contienen las dos tablas.

users

id	first_name	last_name	email	created_at	updated_at
1	Marquita	Bailey	m.bailey@nanonull.com	2016-12-29 14:37:14	[NULL]
2	Sharda	Junker	s.junker@nanonull.com	2016-12-29 14:37:14	[NULL]
3	Totie	Rea	t.rea@nanonull.com	2016-12-29 14:37:14	[NULL]
4	Tobie	Hughey	t.hughey@nanonull.com	2016-12-29 14:37:14	[NULL]
5	Eadith	Lafreniere	e.lafreniere@nanonull.com	2016-12-29 14:37:14	[NULL]
6	Yehudi	Sponga	y.sponga@nanonull.com	2016-12-29 14:37:14	[NULL]
7	Laurianne	Huisman	l.huisman@nanonull.com	2016-12-29 14:37:14	[NULL]
8	Fred	Weinstein	f.weinstein@nanonull.com	2016-12-29 14:37:14	[NULL]
9	Mia	Dahill			
10	June	Leiker			
11	Benedick	Kocyk			
12	Andrej	Hildebrand			
13	Ariel	Phelan			
14	Matthaeus	Hulick			
15	Lotta	Mendes			
16	Jessey	Decelles			
17	Hilda	Lees			
18	Mark	Marzolla			
19	Dannie	Vignola			
20	Lanita	Krysiak			

addresses

id	user_id	is_shipping	is_billing	type	city	street	number
1	1	1	0	work	Bridgedell	Maple Lane	1
2	1	0	1	home	Bridgedell	Olive Street	6
3	3	1	1	home	Roseford	Evergreen Lane	34
4	4	1	1	work	Beardale	Route 44	9
5	6	1	1	home	Johnson City	Franklin Avenue	11
6	7	1	1	home	North Kingstown	Beach Alley	5
7	8	1	1	home	Merrowmeadow	Freybeach Street	85
8	10	1	1	work	Barrowedge	Penn Street	8
9	12	1	1	home	Elfville	Creek Road	3
10	13	1	1	home	Roseford	Bowman Ave.	853
11	14	1	1	work	Beardale	Iroquois Street	98
12	17	1	1	home	Bridgedell	Smith Road	7
13	18	1	0	home	Roseford	Wood Street	7
14	18	0	1	work	Johnson City	Thorne Lane	9677
15	20	1	1	home	Mechanicsville	Vine Street	9065

Cada registro de usuario de la tabla `users` puede tener cero o más direcciones en la tabla

© 2018 Altova GmbH

addresses. Por ejemplo, un usuario puede tener una dirección de tipo "home" o dos direcciones (una de tipo "home" y otra de tipo "work") o no tener dirección alguna.

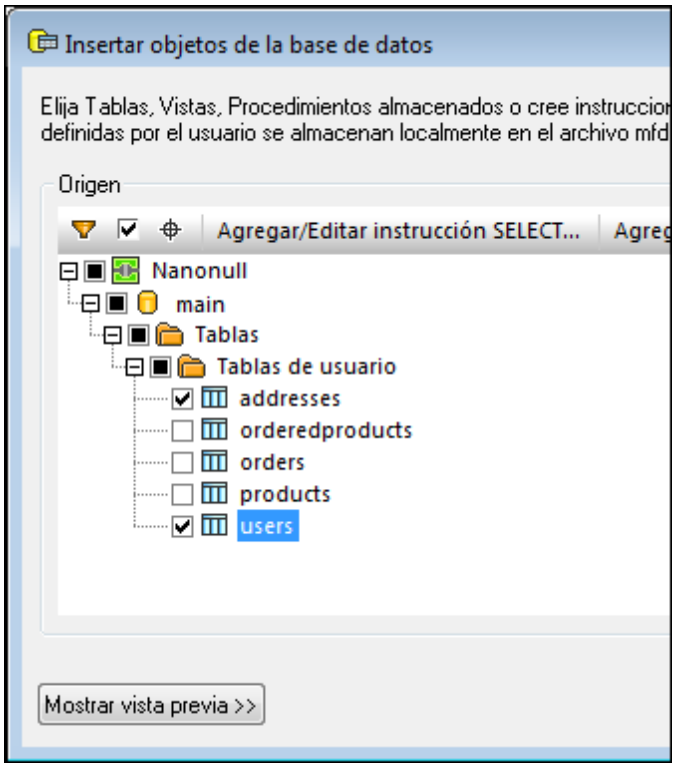
El objetivo de la asignación es recuperar todos los datos (nombre, apellido, correo electrónico, ciudad, calle y número) de todos los usuarios que tengan como mínimo una dirección en la tabla addrresses. También debería ser posible recuperar solamente cierto tipo de direcciones (p. ej. solo las de tipo "home" o solo las de tipo "work"). El tipo de dirección que se debe recuperar ("home" o "work") debe especificarse como parámetro en la asignación. Los registros recuperados deberán ordenarse alfabéticamente según el apellido de la persona.

Todo esto se conseguirá con ayuda de un componente de combinación.

Nota: los componentes de combinación no son el único método posible para combinar tablas o vistas de base de datos. También puede realizar combinaciones de datos de base de datos con ayuda de instrucciones SQL SELECT (*véase* *Instrucciones SQL SELECT como tablas virtuales*). La principal diferencia entre las instrucciones SQL SELECT y los componentes de combinación es que aquellas se escriben a mano y permiten mayor flexibilidad. Los componentes de combinación son una alternativa sencilla para quienes no dominen SQL.

Paso nº1: agregar la base de datos de origen

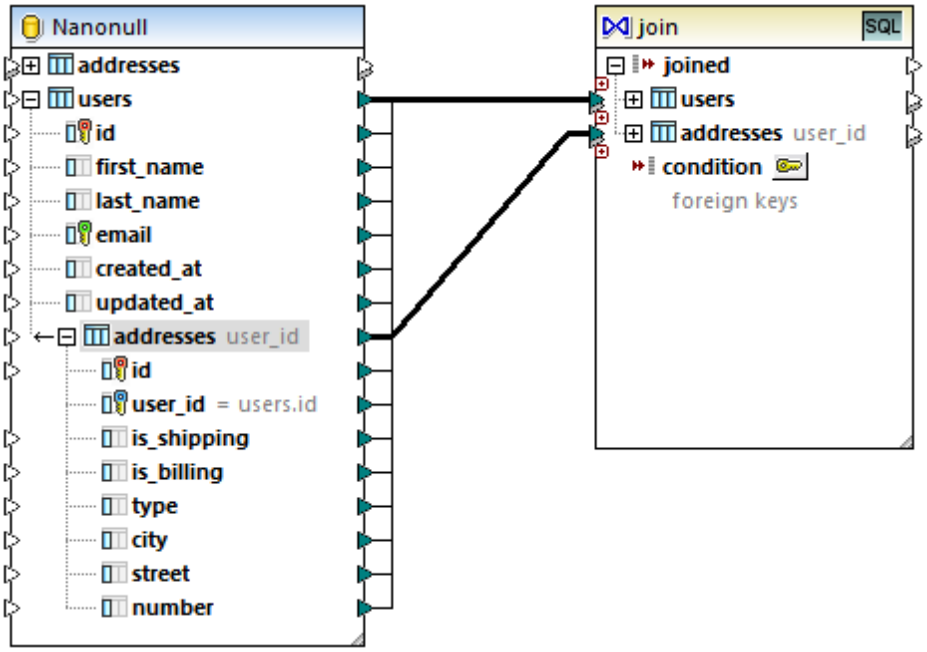
1. En el menú **Insertar** haga clic en el comando **Base de datos** (o en el botón **Insertar base de datos** de la barra de herramientas).
2. Seleccione el tipo de base de datos *SQLite* y haga clic en **Siguiente**.
3. Navegue hasta el archivo **Nanonull.sqlite** de la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial** y haga clic en **Conectarse**.
4. Cuando la aplicación lo solicite, seleccione las tablas addresses y users.



Paso nº2: agregar el componente de combinación

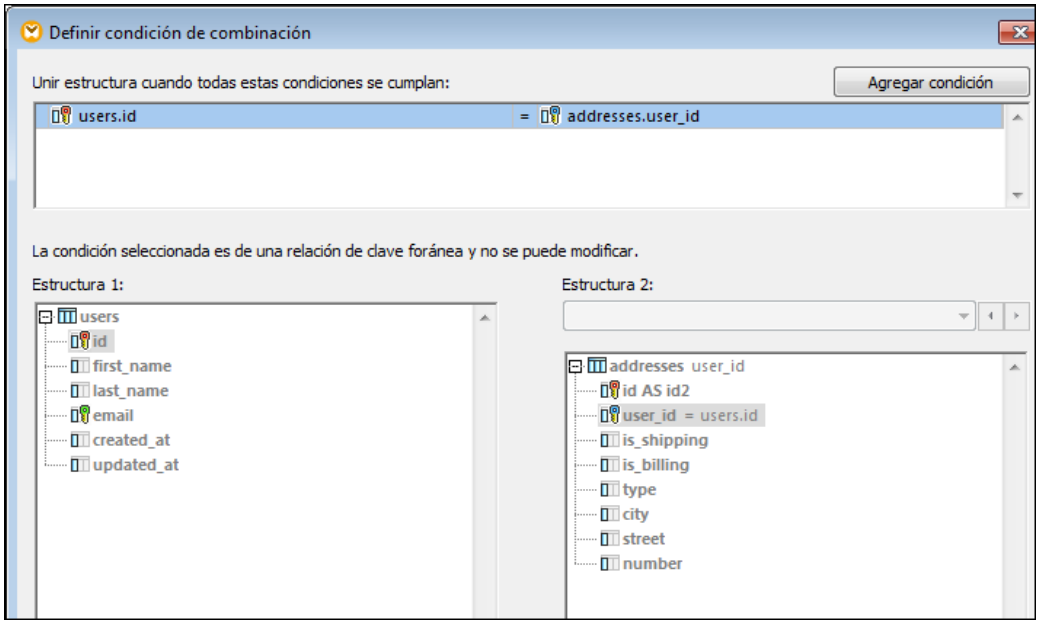
1. En el menú **Insertar** haga clic en **Combinación** (o en el botón **Insertar componente de combinación** de la barra de herramientas).
2. Dibuje una conexión entre la tabla `users` y la primera entrada del componente de combinación.
3. Expanda la tabla `users` y dibuje una conexión entre la tabla `addresses` (secundaria de `users`) y la segunda entrada del componente de combinación. El botón sirve para añadir más tablas si las necesita. No obstante, en este ejemplo solamente combinaremos dos tablas.

© 2018 Altova GmbH



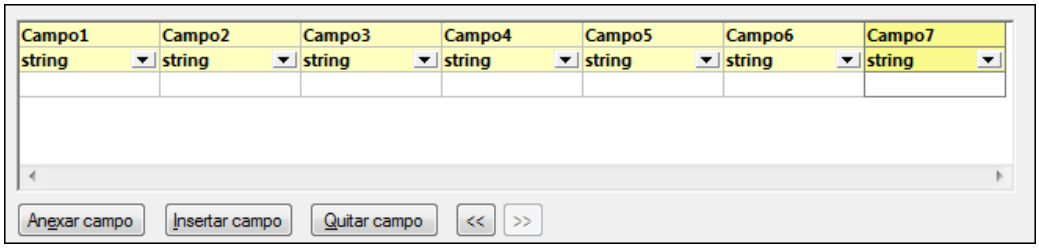
Nota: también se puede agregar la conexión desde la tabla `addresses` directamente (la que no es tabla secundaria de `users`). Sin embargo, en el ejemplo que nos ocupa las condiciones de combinación deberían definirse a mano (*véase* *Agregar condiciones de combinación*). Para este ejemplo asegúrese de crear las conexiones que aparecen en la imagen anterior. Eso garantizará que la condición de combinación necesaria se crea de forma automática.

4. Haga clic en el botón **Definir condición de combinación** del componente de combinación. Observe que la condición de combinación se crea de forma automática (`users.id = addresses.user_id`).

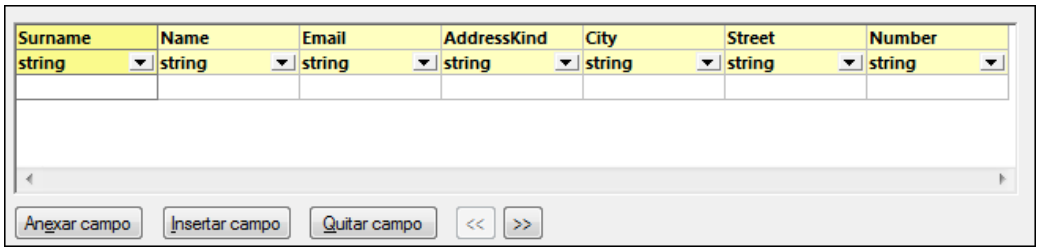


Paso nº3: agregar el componente CSV de destino

1. En el menú **Insertar** haga clic en **Archivo de texto** (o en el botón **Insertar archivo de texto** de la barra de herramientas).
2. Cuando la aplicación le pida que elija un modo de procesamiento, seleccione la opción *Utilizar procesamiento básico...* en el cuadro de diálogo.
3. Haga clic en **Anexar campo** varias veces para crear siete campos CSV. Deje la demás opciones del cuadro de diálogo como están.

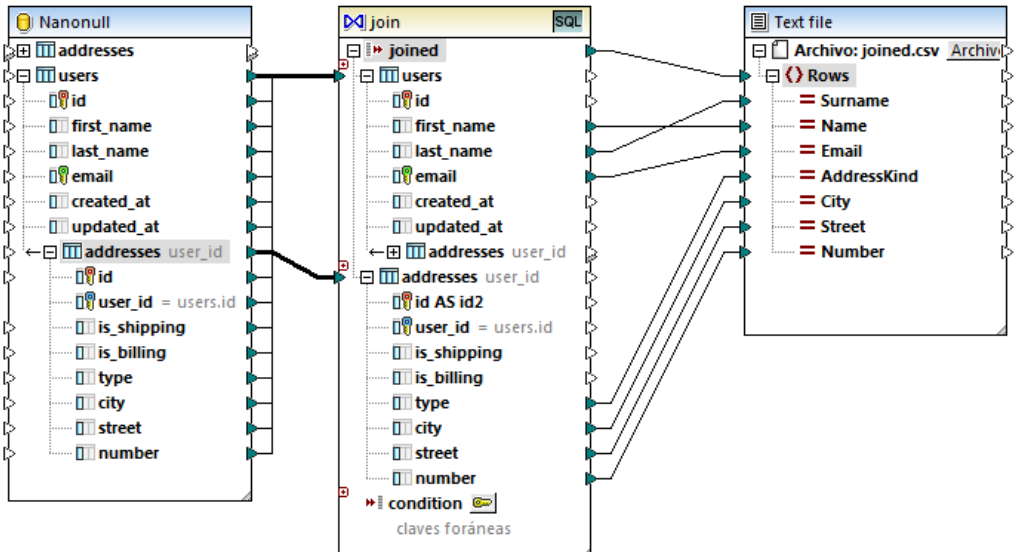


4. Ahora haga doble clic en la celda que contiene el título de los campos recién creados y modifique el nombre de las celdas (*imagen siguiente*).



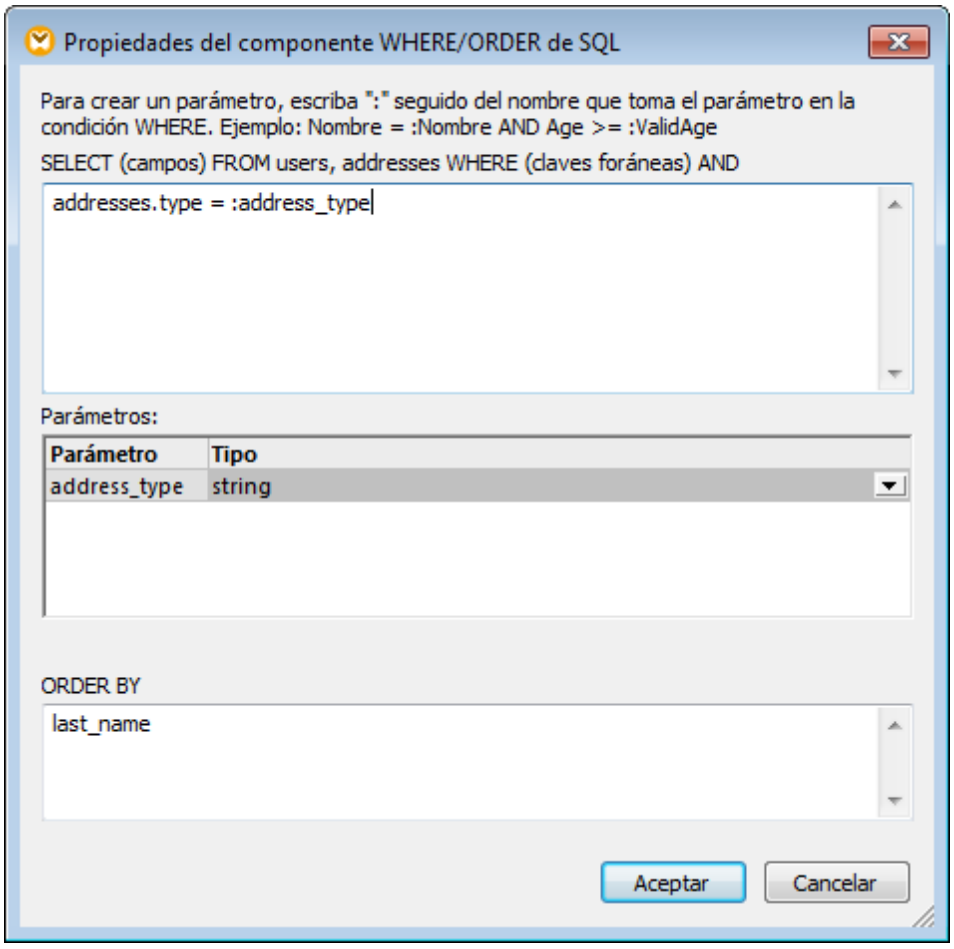
5. Dibuje las conexiones que pueden verse en la siguiente imagen entre el componente de combinación y el componente CSV. La conexión que une el elemento `joined` del componente de combinación y el elemento `Rows` del componente CSV de destino da la orden de *crear tantos registros (filas) en el destino como registros existan que cumplan la condición de combinación*.

© 2018 Altova GmbH

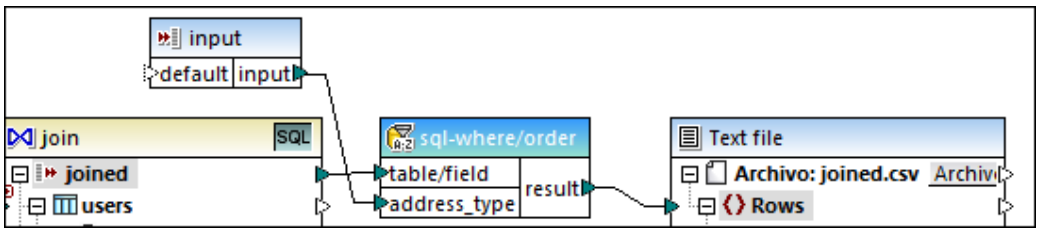


Paso nº4: agregar la condición WHERE/ORDER de SQL y el parámetro de entrada

1. Haga clic con el botón derecho en la conexión que une el elemento `joined` del componente de combinación y el elemento `Rows` del componente CSV de destino y después seleccione **Insertar WHERE/ORDER de SQL**.
2. Introduzca las cláusulas WHERE y ORDER BY que pueden verse en la siguiente imagen.

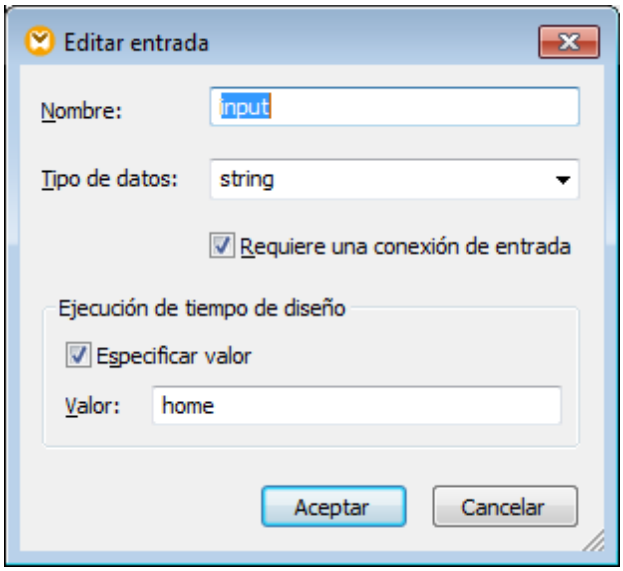


3. En la asignación añada un componente de entrada (con el comando de menú **Insertar | Insertar componente de entrada**) y conecte su resultado con el parámetro `address_type` creado en el paso anterior.



4. Haga doble clic en el componente de entrada y configúrelo como muestra la siguiente imagen. Para poder consultar la vista previa de resultados de la asignación en MapForce es necesario tener un valor en tiempo de diseño (en este caso "home"). Si quiere que la vista previa recupere las direcciones de tipo "work", reemplace este valor con "work".

 © 2018 Altova GmbH



Funcionamiento de esta asignación de datos

La condición de combinación creada de forma automática en el paso nº2 se asegura de que en el destino solo se copien los registros que cumplan la condición de combinación `users.id = addresses.user_id`. La condición de combinación se añadió automáticamente porque las dos tablas están unidas por una relación de clave foránea y porque se dibujaron las correspondientes conexiones de asignación de datos (*véase* *Gestionar relaciones de BD*). Aunque en nuestro caso concreto no sea relevante, también se pueden definir condiciones de combinación a mano (*véase* *Ejemplo: crear un informe CSV a partir de varias tablas*).

Las dos tablas de origen son de la misma base de datos y del mismo componente así que esta combinación puede hacerse en modo SQL. Como el modo SQL está habilitado, la operación de combinación la lleva a cabo la base de datos y no MapForce. En otras palabras, MapForce genera una instrucción de combinación interna INNER JOIN y la envía a la base de datos para que la ejecute.

El componente WHERE/ORDER de SQL que se añadió en el paso nº4 permite filtrar (recuperar las direcciones de tipo "home" o "work") y ordenar el conjunto de registros. Tenga en cuenta que la cláusula WHERE creó un parámetro llamado **`:address_type`** de tipo `string`. Este parámetro permite suministrar el tipo de dirección (home o work) desde la asignación. Para más información consulte Filtrar y ordenar datos de BD (SQL WHERE/ORDER).

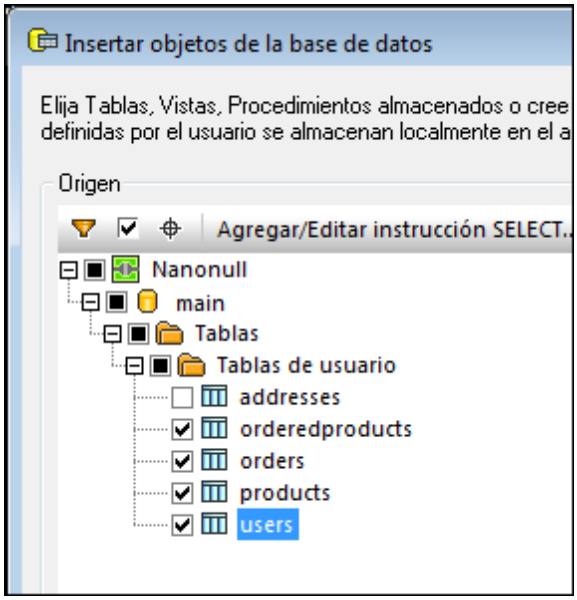
Por último, el componente de entrada permite suministrar el valor de parámetro actual cuando se ejecuta la asignación. Recuerde que, cuando la asignación se ejecuta fuera de MapForce (p. ej. cuando se ejecuta con MapForce Server en otro equipo), la entrada debe suministrarse en tiempo de ejecución como parámetro de la línea de comandos, en cuyo caso se omitirá el valor mencionado. Para más información consulte Pasar parámetros a la asignación.

7.2.3.3 *Ejemplo: crear un informe CSV a partir de varias tablas*

Este ejemplo demuestra cómo combinar varias tablas de base de datos para extraer datos y guardarlos en un solo informe en formato CSV. La base de datos utilizada en este ejemplo se llama **Nanonull.sqlite** y está guardada en este directorio: **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial**. Esta base de datos almacena información sobre una empresa ficticia (pedidos, productos, usuarios y direcciones de los usuarios). Como en cualquier otra base de datos relacional, la información de esta base de datos está normalizada y repartida en varias tablas. Por ejemplo, la tabla `users` almacena datos personales de los usuarios (su nombre, apellido y correo electrónico). La base de datos también almacena información sobre los productos adquiridos por los usuarios en dos tablas distintas: `orders` (que incluye el identificador único del pedido y la hora en la que se realizó el pedido) y `orderedproducts` (que incluye una lista de productos pedidos y su cantidad). Además, el nombre de los productos se almacena en una tabla llamada `products`.

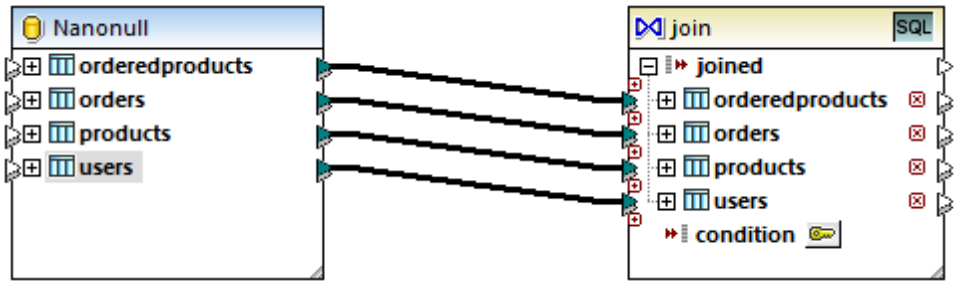
El objetivo de nuestra asignación de datos será producir un informe basado en los datos extraídos de varias tablas y así conocer quién pidió ciertos productos, cuándo y en qué cantidad. Para alcanzar este objetivo debemos seguir estos pasos:

1. En el menú **Insertar** haga clic en el comando **Base de datos**.
2. Cuando llegue el momento de seleccionar un tipo de base de datos, haga clic en la opción *SQLite* y después en el botón **Siguiente**.
3. Navegue hasta la base de datos **Nanonull.sqlite** que mencionamos al principio y haga clic en **Conectarse**.
4. En el cuadro de diálogo que aparece seleccione las tablas `orderedproducts`, `orders`, `products` y `users` y haga clic en **Aceptar**.



5. Añada un componente de combinación y después cree cuatro elementos `nodes/rows` (haciendo clic en el botón **Agregar entrada** del componente de combinación).
6. Conecte las cuatro tablas del componente de base de datos con los elementos de entrada del componente de combinación.

© 2018 Altova GmbH

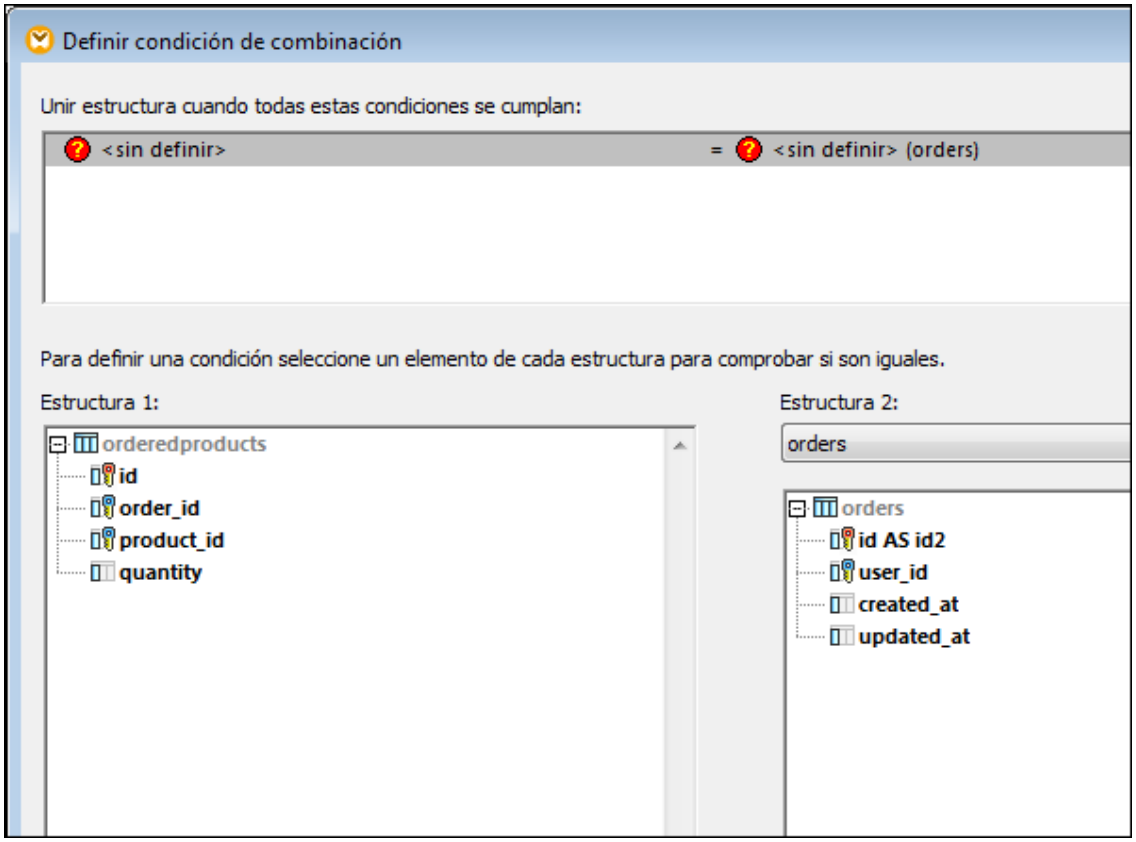


Nota: **otro método alternativo sería conectar la tabla** `orderedproducts` al componente de combinación, después la tabla `orders` (la que tiene anidada y no la que está al mismo nivel) y así sucesivamente hasta que todas las tablas combinadas estén anidadas bajo la misma tabla raíz (*véase Gestionar relaciones de BD*). El resultado de la asignación sería idéntico al que se obtendría al combinar las tablas de esta forma. La diferencia es que en este ejemplo las condiciones de combinación deben crearse a mano (ver más abajo), mientras que con el método alternativo recién descrito las condiciones de combinación las crearía MapForce automáticamente. Para ver un ejemplo en el que se combinan tablas sin necesidad de definir condiciones de combinación a mano consulte el apartado Ejemplo: combinar tablas en modo SQL. También puede consultar la asignación de datos **<Documentos>\Altova\MapForce2018\MapForceExamples\DB_Denormalize.mfd**, donde todas las tablas combinadas se encuentran bajo la misma tabla raíz.

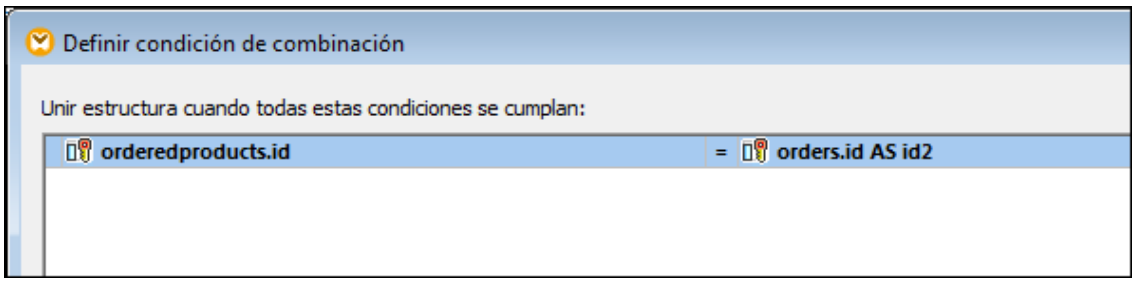
En el ejemplo que nos ocupa las tablas conectadas al componente de combinación siguen este orden:

1. `orderedproducts`
2. `orders`
3. `products`
4. `users`

Este orden afecta a la presentación de las correspondientes estructuras en el cuadro de diálogo "Definir condición de combinación" (*imagen siguiente*), que se abre al hacer clic en el botón **Definir condición de combinación** . Es decir, la primera tabla (`orderedproducts`) aparece por defecto bajo *Estructura 1* y la tabla inmediatamente posterior a ella (`orders`) aparece bajo *Estructura 2*.

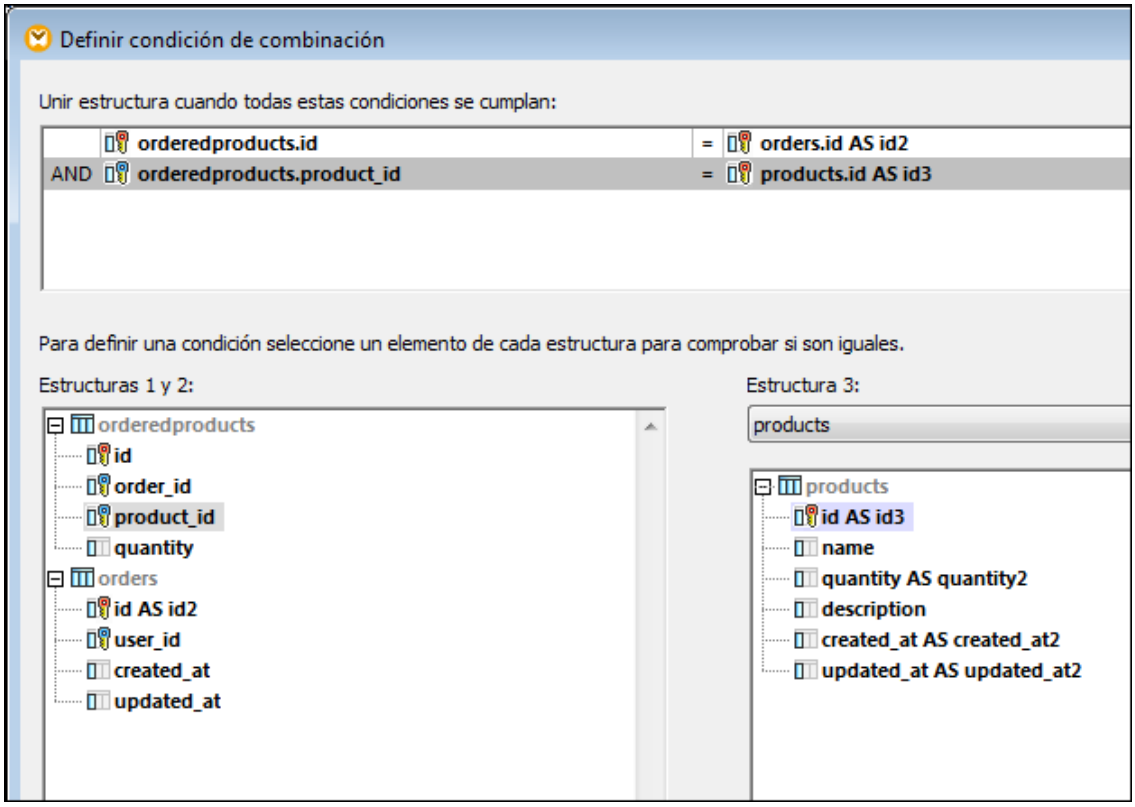


Para definir la primera condición de combinación haga clic en el elemento `order_id` del panel izquierdo y en el elemento `id` del panel derecho. Esto empareja los campos `orderedproducts.order_id` y `orders.id`:

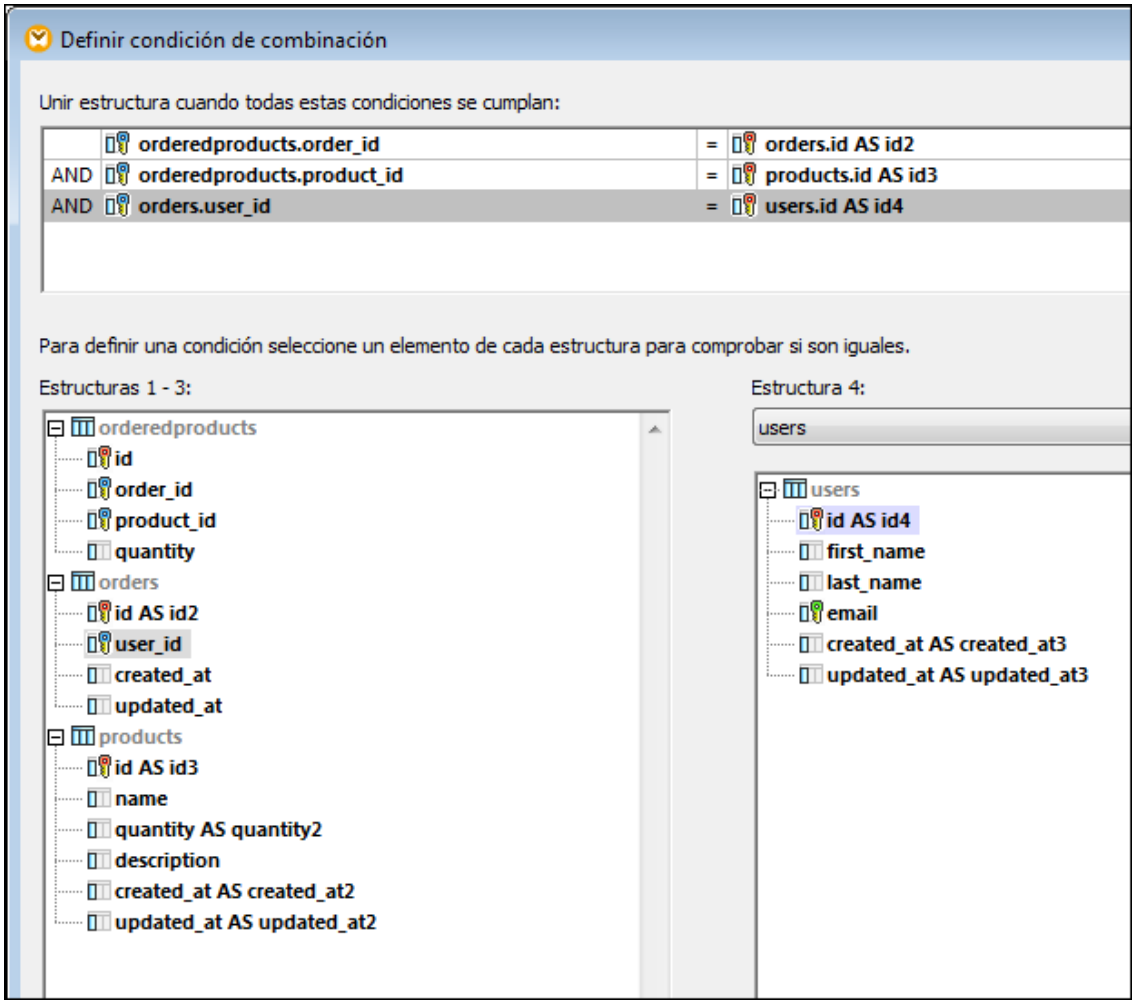


Hasta ahora hemos combinado solamente dos tablas. Para definir condiciones de combinación que comprendan una tercera tabla es necesario seleccionar la tabla correspondiente en la lista desplegable situada sobre el panel derecho. En este caso el panel izquierdo muestra todas las tablas que aparecen después de dicha tabla en el componente de combinación. Por ejemplo, si seleccionamos `products` en el panel derecho, el panel izquierdo presentará `orderedproducts` y `orders` (porque estas dos tablas aparecen antes de `products` en el componente de combinación). Ahora podemos emparejar campos de la tabla `products` con campos de tablas precedentes (en este caso `orderedproducts.product_id` y `products.id`).

© 2018 Altova GmbH

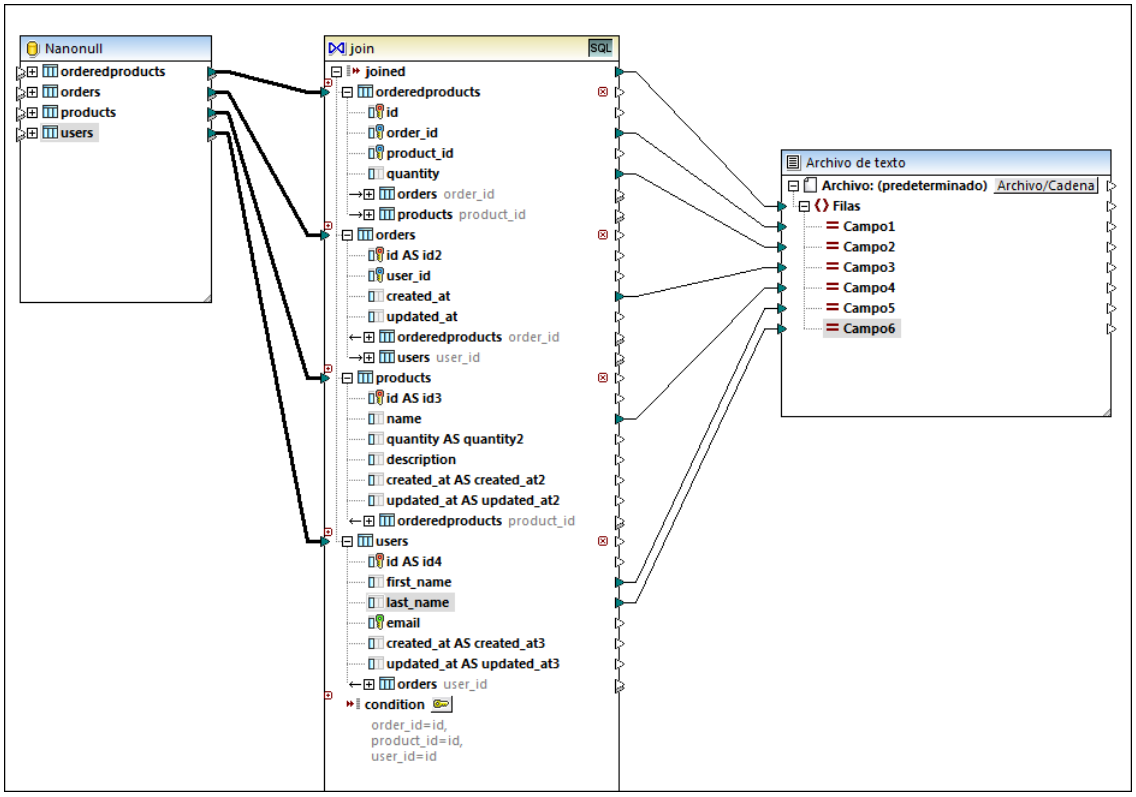


Para combinar la cuarta tabla (`users`) basta con seleccionar la tabla `users` en la lista desplegable y podremos emparejar los campos `orders.user_id` con los campos `users.id`.



Ahora que todas las condiciones de combinación están definidas, podremos crear asignaciones de datos entre los elementos de componente de combinación y los del componente de destino. Para terminar de diseñar la asignación de datos añada un componente CSV (*véase* *Archivos CSV y archivos de texto*) y conecte los elementos del componente de combinación con el componente CSV de destino tal y como muestra la imagen siguiente.

© 2018 Altova GmbH



Esta asignación de datos genera un informe en formato CSV a partir de las cuatro tablas que participan en la combinación y que incluye:

- el identificador del pedido (procedente de la tabla `orderedproducts`)
- la cantidad de artículos pedidos (procedente de la tabla `orderedproducts`)
- la hora a la que se realizó el pedido (procedente de la tabla `orders`)
- el nombre del producto que se pidió (procedente de la tabla `products`)
- el nombre y el apellido del usuario que hizo el pedido (procedente de la tabla `users`)

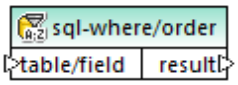
7.2.4 Filtrar y ordenar datos de BD (SQL WHERE/ORDER)

Cuando necesite filtrar y ordenar datos de base de datos, utilice un componente WHERE/ORDER de SQL. Esto le permitirá introducir a mano (desde la interfaz gráfica del usuario de MapForce) una cláusula WHERE de SQL que filtre los datos. También tiene la opción de especificar una cláusula ORDER BY si desea ordenar el conjunto de registros según cierto campo de la base de datos, en orden ascendente o descendente.

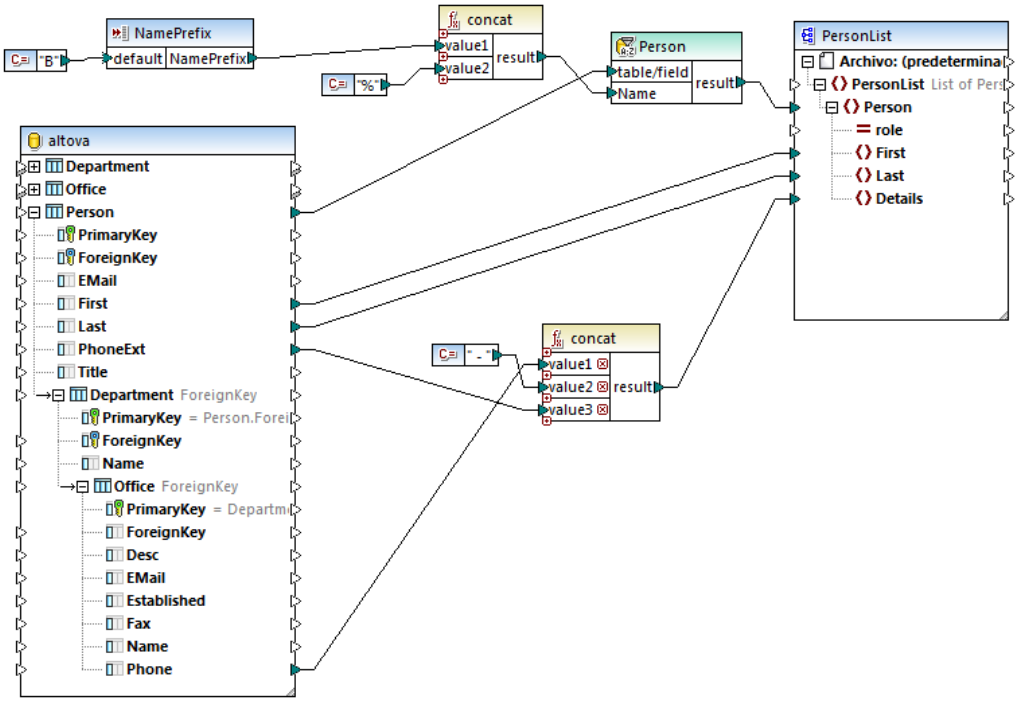
El componente WHERE/ORDER de SQL debe estar conectado a una tabla o campo de un componente de base de datos. También puede conectarse a un componente de combinación cuando se necesita filtrar un conjunto de registros o registros combinados previamente (*véase* *Combinar datos de BD*).

Agregar un componente WHERE/ORDER de SQL en la asignación

1. En el menú Insertar seleccione el comando **WHERE/ORDER de SQL**. Este componente presenta esta estructura predeterminada:

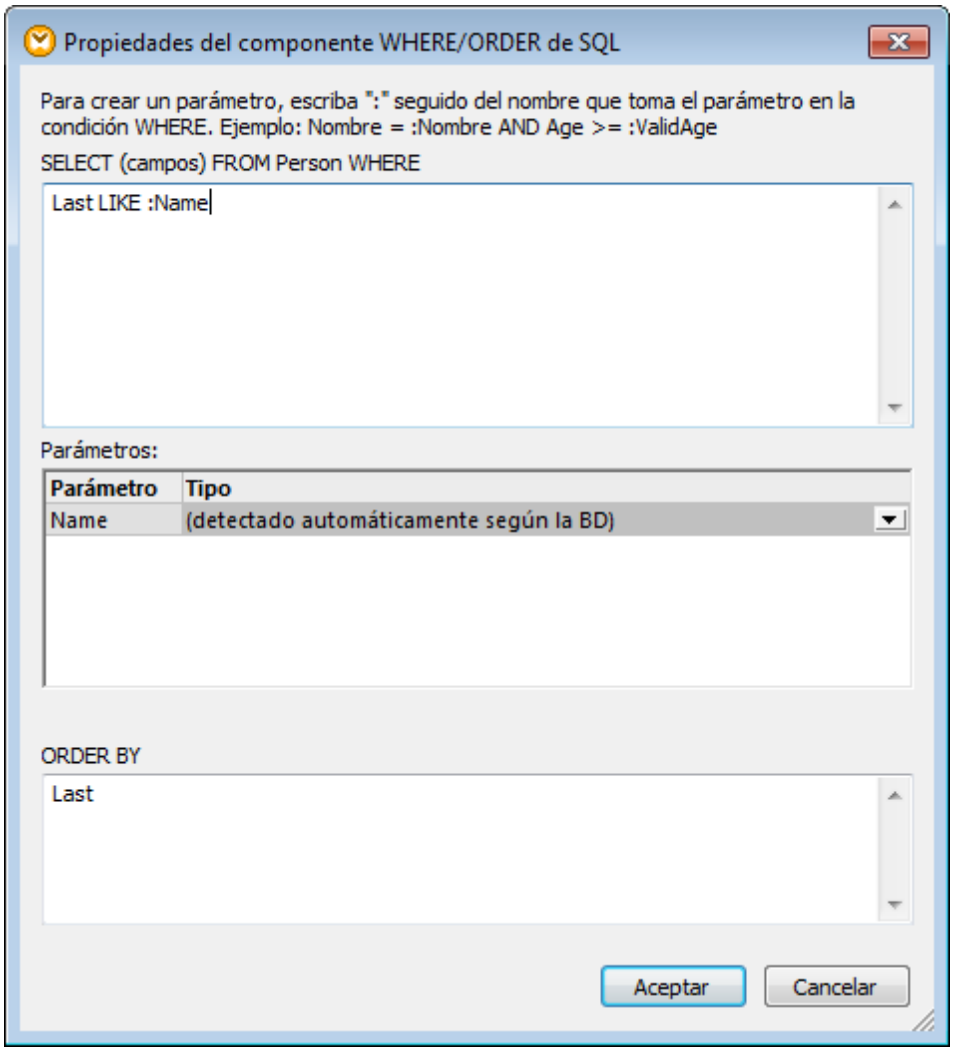


2. Conecte una tabla o un campo de la tabla de base de datos de origen con el nodo `table/field` del componente WHERE/ORDER de SQL. Por ejemplo, el ejemplo de asignación **DB_PhoneList.mfd** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples** incluye un componente WHERE/ORDER de SQL que sirve para filtrar los registros de la tabla de origen `Person` cuyo apellido empiece por la letra B.



3. Haga doble clic en el título del componente WHERE/ORDER de SQL (o clic con el botón derecho y después clic en **Propiedades**). Esto abre el cuadro de diálogo "Propiedades del componente WHERE/ORDER de SQL".

© 2018 Altova GmbH



4. Escriba la cláusula WHERE de SQL en el cuadro de texto superior. También tiene la opción de introducir la cláusula ORDER BY. En la imagen anterior puede ver las cláusulas WHERE y ORDER BY definidas en la asignación **DB_PhoneList.mfd**. Encontrará más ejemplos en el apartado Crear cláusulas WHERE y ORDER BY.

Suministrar parámetros a componentes WHERE/ORDER de SQL

El componente componente WHERE/ORDER de SQL de la asignación **DB_PhoneList.mfd** define esta cláusula WHERE:

```
Last LIKE :Name
```

"Last" se refiere al nombre de un campo de la BD de la tabla que está conectada. "LIKE" es un operador SQL y ":Name" crea un parámetro llamado "Name" en la asignación.

Los parámetros son opcionales en los componentes WHERE/ORDER de SQL. Son prácticos si se quiere pasar un valor a la cláusula WHERE desde la asignación. La misma cláusula WHERE

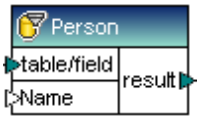
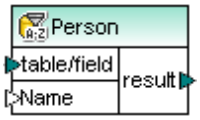
se puede escribir sin parámetros de la siguiente manera:

```
Last LIKE "B%"
```

Esto recuperaría todas las personas cuyo apellido empiece con la letra B. Para poder hacer esta consulta más flexible añadimos un parámetro en lugar de "B%". Esto nos permitirá usar cualquier letra desde la asignación (p. ej. "C" para recuperar las personas cuyo apellido empiece por la letra "C" con solo cambiar una constante o un parámetro de entrada).

Aspecto de los componentes WHERE/ORDER de SQL

Algo importante que se debe tener en cuenta sobre los componentes WHERE/ORDER de SQL es que su aspecto cambia dependiendo de su configuración. Esto permite saber a simple vista qué papel desempeña el componente WHERE/ORDER de SQL en la asignación. Por ejemplo:

	Se definió una cláusula WHERE.
	Se definió una cláusula WHERE con un parámetro. El parámetro "Name" aparece debajo del nodo "table/field".
	Se definió una cláusula WHERE con un parámetro y una cláusula ORDER BY. El icono A-Z indica que se definió una ordenación.

Con solo pasar el puntero del ratón por encima del título del componente podrá ver qué cláusulas se han definido exactamente.

7.2.4.1 Crear cláusulas WHERE y ORDER BY

Tras añadir un componente WHERE/ORDER de SQL a la asignación deberá añadir una condición WHERE para especificar cómo se deben filtrar los datos que están conectados al componente exactamente. la condición WHERE debe introducirse en el cuadro de diálogo "Propiedades del componente WHERE/ORDER de SQL".

En MapForce las condiciones WHERE se escriben de forma similar a como se escriben las mismas cláusulas SQL fuera de MapForce, usando la sintaxis que corresponda al dialecto SQL de la base de datos. Por ejemplo, puede usar operadores, comodines, subselecciones o funciones de agregado. Para crear parámetros que se puedan pasar desde la asignación introduzca un carácter de dos puntos (:) seguido del nombre del parámetro-

Nota: cuando termine de escribir la cláusula WHERE y haga clic en **Aceptar**, MapForce validará la integridad de la instrucción SQL final y, si se detectan errores sintácticos, emitirá mensajes en un cuadro de diálogo.

En esta tabla puede ver los operadores más frecuentes que se pueden usar en la cláusula

 © 2018 Altova GmbH

WHERE:

Operador	Descripción
=	Igual
<>	No igual
<	Menor que
>	Mayor que
>=	Mayor o igual
<=	Menor o igual
IN	Recupera un valor conocido de una columna
LIKE	Busca un patrón determinado
BETWEEN	Busca dentro de un intervalo

Use el comodín `%` (porcentaje) para representar cualquier número de caracteres de un patrón. Por ejemplo, para recuperar todos los registros que empiecen por R de un campo llamado `lastname`, debe usar esta expresión:

```
lastname = "%r"
```

Cuando consulte bases de datos compatibles con la ordenación y consulta de datos de base de datos XML (p. ej. IBM DB2, Oracle, SQL Server), podrá usar funciones y palabras clave XML que correspondan a dicho tipo de base de datos. Por ejemplo:

```
xmlexists('$c/Client/Address[zip>"55116"]' passing
USER.CLIENTS.CONTACTINFO AS "c")
```

Consulte también el apartado Ejemplo: extraer datos de columnas tipo XML de IBM DB2.

Además, si quiere ordenar el conjunto de registros recuperado usando un campo como criterio de ordenación, añada una cláusula ORDER BY en el cuadro correspondiente del cuadro de diálogo "Propiedades del componente WHERE/ORDER de SQL". Para ordenar los registros usando varios campos como criterio de ordenación, separe los nombres de campo por comas. Para cambiar el orden (ascendente/descendente) use las palabras clave `ASC` y `DESC`. Por ejemplo, la cláusula ORDER BY que aparece a continuación recupera registros ordenados usando el campo `lastname` y después el campo `firstname` como criterios de ordenación en orden descendente:

```
lastname, firstname DESC
```

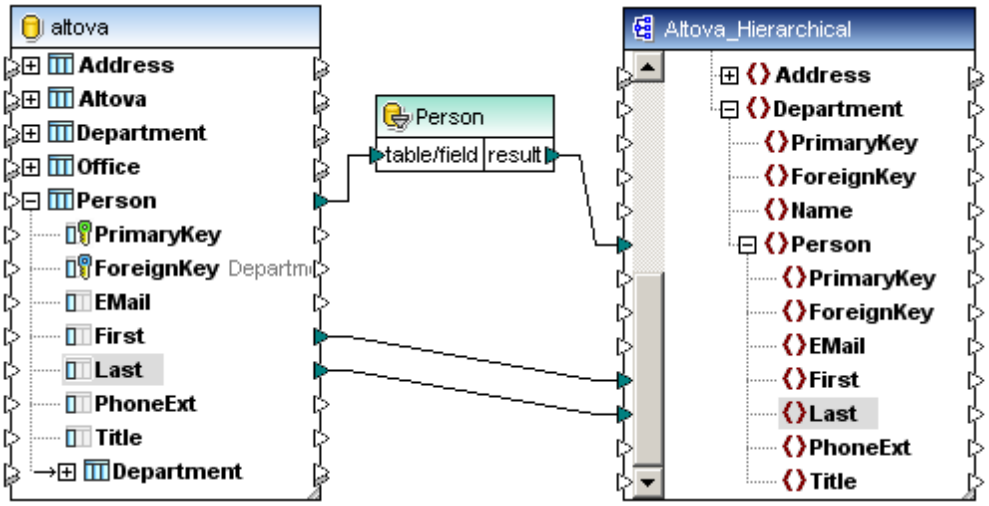
Ejemplo nº1

La condición WHERE que aparece a continuación está unida a la tabla `Person` del componente de base de datos **altova.mdb**. Esta condición se encarga de recuperar los registros cuyos campos `First` y `Last` sean superiores a la letra "C". En otras palabras recupera todos los nombres a partir de Callaby.

```
First > "C" AND Last > "C"
```

Observe cómo se realizaron las conexiones:

- La conexión con **table/field** se origina en la tabla que se debe consultar (en este caso, **Person**).
- El resultado **result** se conecta con un elemento "parent" de los campos que se consultan/filtran (en este caso, el elemento `Person`).

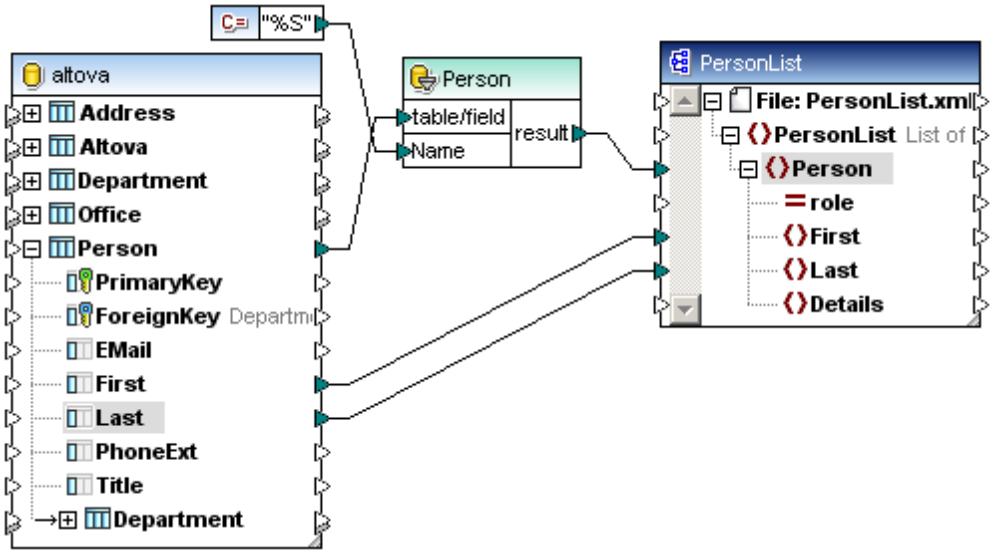


Ejemplo nº2

La condición WHERE que aparece a continuación crea un parámetro **Name** que aparece en el componente WHERE/ORDER de SQL de la asignación.

```
Last LIKE :Name
```


© 2018 Altova GmbH

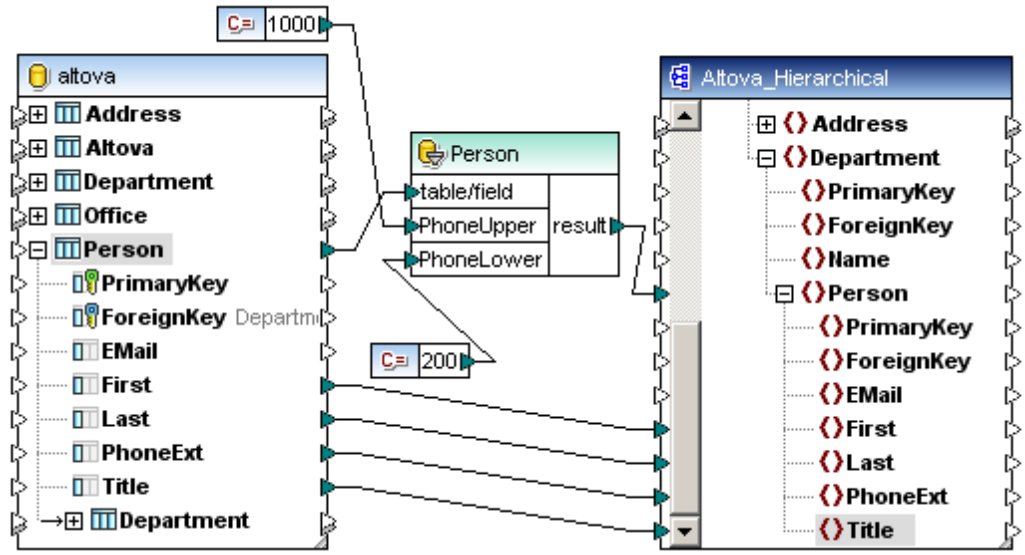


El componente de constante `%S` aporta el valor del parámetro **Name**. El comodín % denota cualquier número de caracteres. Esto hace que la asignación busque un parámetro en la columna "Last" (todos los apellidos que acaben en "S").

Ejemplo nº3

La condición WHERE que aparece a continuación crea dos parámetros: `PhoneUpper` y `PhoneLower`. Los valores actuales de `PhoneExt` se comparan con estos dos parámetros. El valor superior (`Upper`) e inferior (`Lower`) se indican por medio de constantes.

```
PhoneExt < :PhoneUpper and PhoneExt > :PhoneLower
```



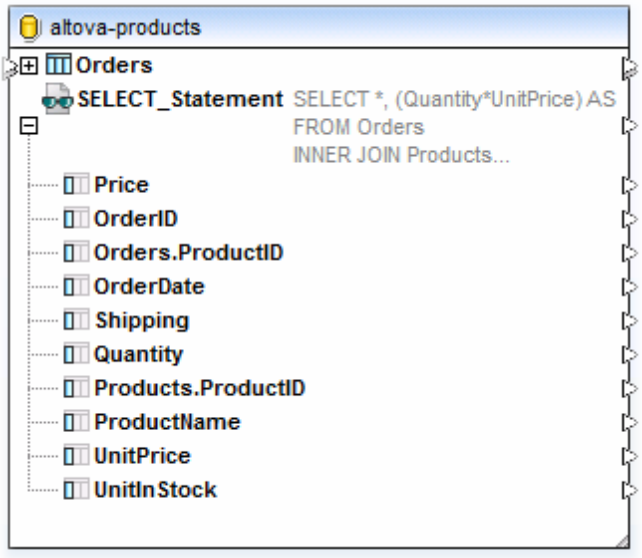
La condición WHERE de este ejemplo también se podría escribir con el operador BETWEEN:

```
PhoneExt BETWEEN :PhoneUpper and :PhoneLower
```

7.2.5 Instrucciones SQL SELECT como tablas virtuales

MapForce permite crear instrucciones SELECT de SQL con parámetros dentro de componentes de BD. Se trata de estructuras tipo tabla que contienen los campos del conjunto de resultados generado por la instrucción SELECT. Estas estructuras se pueden usar como origen de datos de la asignación, igual que cualquier otra tabla o vista definida en la BD.

- Cuando use combinaciones internas/externas (INNER/OUTER JOIN) en la instrucción SELECT, en el componente se incluyen los campos de todas las tablas.
- Las expresiones con nombres de correlación (que usan la palabra clave SQL "AS") también aparecen en el componente como elementos asignables.
- Las vistas de la BD también se pueden usar en la cláusula FROM.
- Las instrucciones SELECT pueden tener parámetros que usan la misma sintaxis que el componente WHERE/ORDER de SQL.



El número de líneas visibles de la instrucción SELECT se puede configurar. Para definir qué número de líneas desea ver en el componente abra la pestaña *Generales* del cuadro de diálogo "Opciones" (**Herramientas | Opciones**) e introduzca el número de líneas en el grupo *Visualización de asignaciones*.

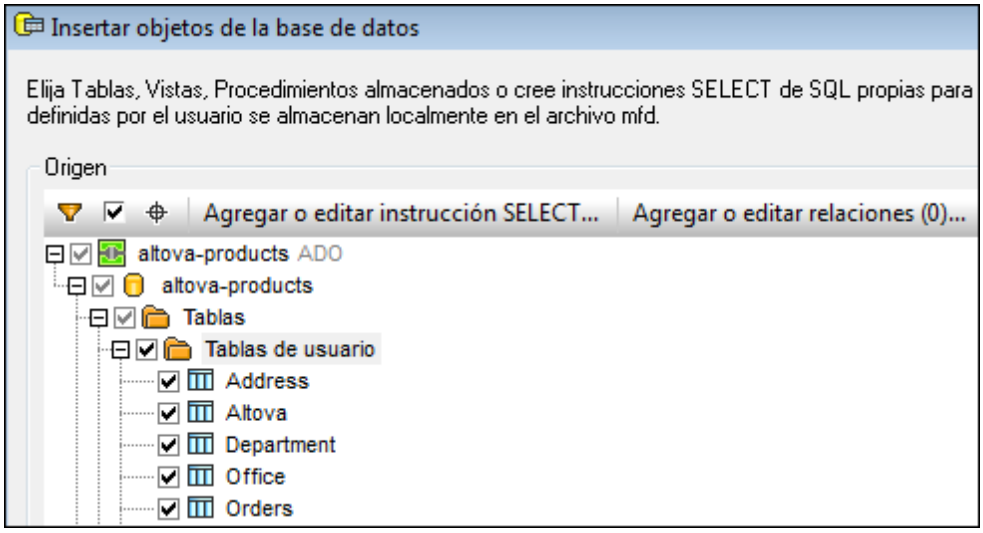
7.2.5.1 Crear instrucciones SELECT

Puede crear instrucciones SELECT en cualquier asignación de datos que tenga un componente de base de datos. Si la asignación no contiene todavía ninguna base de datos, primero deberá añadir una (*véase Conectarse a una base de datos*). Para este ejemplo haga clic en el comando de menú **Insertar | Insertar base de datos** y, siguiendo los pasos del Asistente para la conexión, conéctese al archivo de base de datos **altova-products.mdb** de la carpeta

<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\.

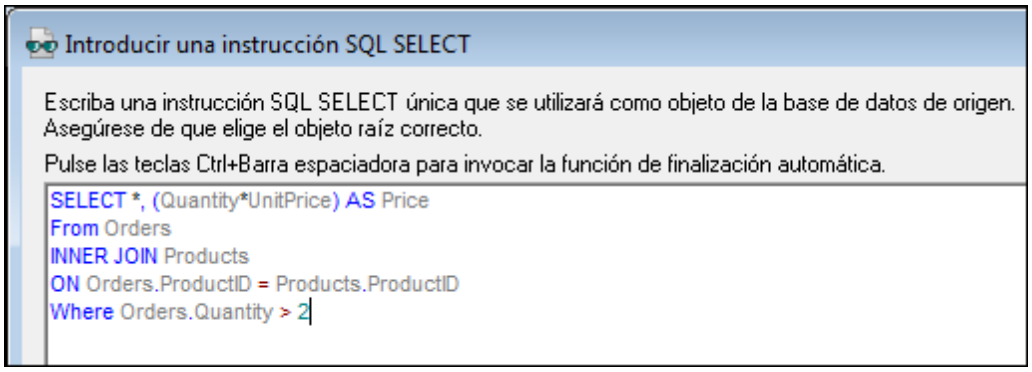
Para crear una instrucción SELECT:

1. Haga clic con el botón derecho en el título del componente de base de datos y elija el comando **Agregar, quitar o editar objetos de la base de datos**. Si lo prefiere, también puede seleccionar el componente de base de datos y después hacer clic en el comando de menú **Componente | Agregar, quitar o editar objetos de la base de datos**.

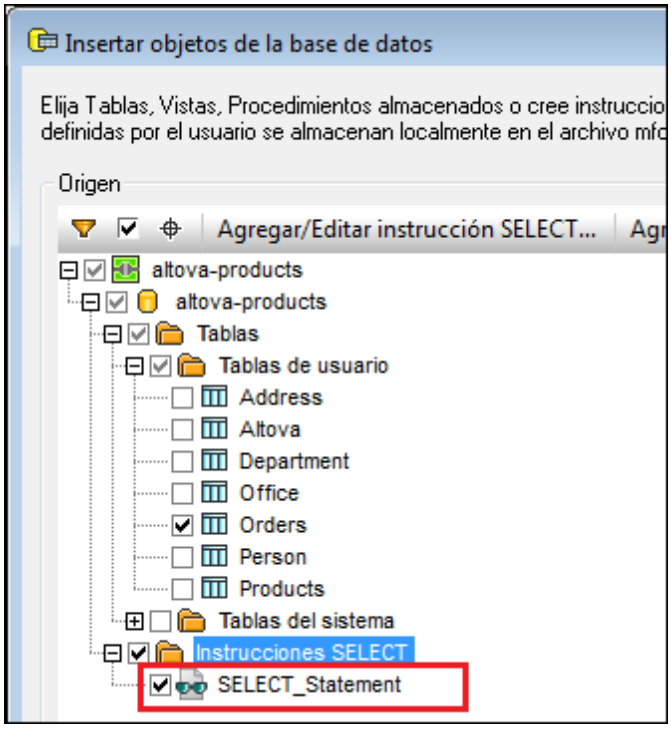


2. Ahora tiene dos opciones:
   - Para generar la instrucción SELECT a partir de una tabla actual: haga clic con el botón derecho en cualquier tabla y seleccione **Generar y agregar una instrucción SELECT...** en el menú contextual. Más tarde podrá editar la instrucción generada.
   - Para escribir una instrucción SELECT personalizada, haga clic en el botón **Agregar/ Editar instrucción SELECT**.
3. Edite o cree la instrucción en función de sus requisitos. Por ejemplo, la instrucción SELECT que aparece a continuación es válida para el archivo **altova-products.mdb** de la carpeta <Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\. El campo **Price** es el producto de los dos campos (**Quantity** y **UnitPrice**) y se declara como nombre de correlación (`AS Price`).

```
SELECT *, (Quantity*UnitPrice) AS Price
From Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
Where Orders.Quantity > 2
```

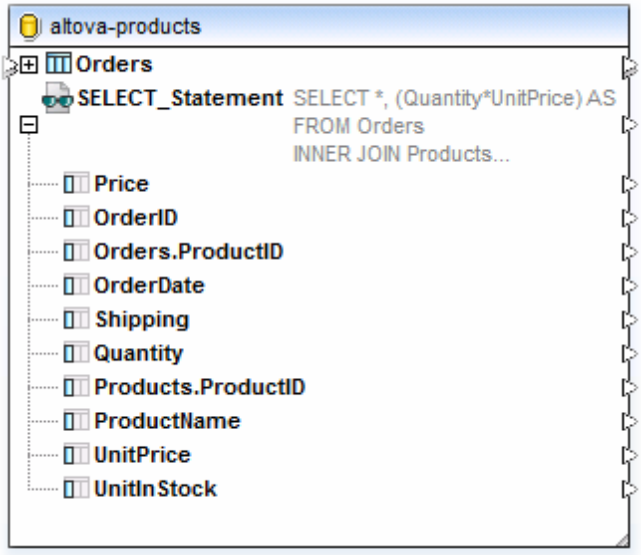


4. Ahora haga clic en el botón **Agregar instrucción SELECT**. Observe que ahora la instrucción SELECT aparece como objeto de la base de datos, como si se tratase de una tabla, una vista o un procedimiento.



5. Para terminar haga clic en **Aceptar**. La instrucción SELECT aparecerá también en el componente de base de datos y podrá crear asignaciones de datos desde cualquier campo que devuelva la consulta SELECT.

© 2018 Altova GmbH



Notas:

- Todas las expresiones calculadas en la instrucción SELECT deben tener un nombre de correlación único (como "AS Price" en el ejemplo anterior). De lo contrario, no estarán disponibles como elemento asignable.
- Si se conecta a una base de datos Oracle o IBM DB2 con JDBC, la instrucción SELECT no puede terminar con un punto y coma.

Para eliminar instrucciones SELECT añadidas previamente:

1. Haga clic con el botón derecho en el título del componente de base de datos y elija el comando **Agregar, quitar o editar objetos de la base de datos**.
2. Haga clic con el botón derecho en la instrucción SELECT que desea eliminar y seleccione el comando **Eliminar instrucción SELECT**.

7.2.5.2 Ejemplo: SELECT con parámetros

Con este ejemplo aprenderá a crear una asignación de MapForce que lee datos de una base de datos Microsoft Access y los escribe en un archivo CSV. En concreto, en este ejemplo utilizamos una consulta SELECT de base de datos personalizada con un parámetro. La instrucción SELECT combina datos de varias tablas diferentes. Y después los resultados se envían a la asignación para seguir procesando los datos.

Este ejemplo viene acompañado de un diseño de asignación, cuya ruta de acceso es **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\select-component.mfd**. Puede abrir este diseño y analizarlo o seguir los pasos que aparecen a continuación para crear el mismo diseño desde cero.

Aunque para este ejemplo usamos una base de datos Microsoft Access, el proceso es idéntico

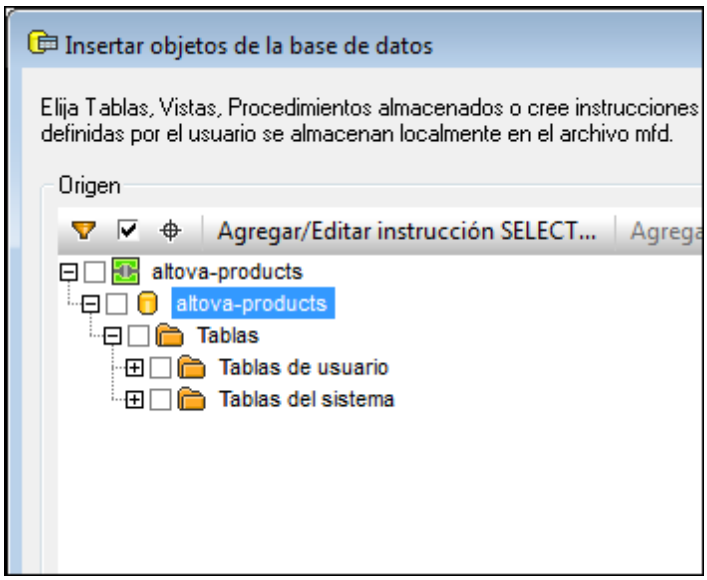
con otras bases de datos. Consulte la sección Conectarse a una base de datos para obtener información sobre cómo conectarse a los distintos tipos de base de datos.

El objetivo de nuestro diseño de asignación es:

1. Seleccionar en la base de datos solamente aquellos pedidos cuyo número de productos pedidos supere determinado valor. Este valor vendrá dado por un parámetro. Para conseguir nuestro objetivo crearemos una instrucción SELECT que tome un parámetro de entrada.
2. En la base de datos Access el formato de fecha es `YYYY-MM-DD HH-MI-SS`. En el archivo CSV de destino la parte de hora debe eliminarse para que el formato sea `YYYY-MM-DD`. Para conseguir nuestro objetivo usaremos la función `date-from-datetime` de MapForce.
3. El archivo CSV resultante debe llamarse **OrdersReport.csv**.

Paso n°1: agregar la estructura SELECT

1. En el menú **Insertar** seleccione el comando **Base de datos**.
2. Seleccione **Microsoft Access (ADO)** y siga los pasos del asistente para conectarse a la base de datos **altova-products.mdb** que está en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`.

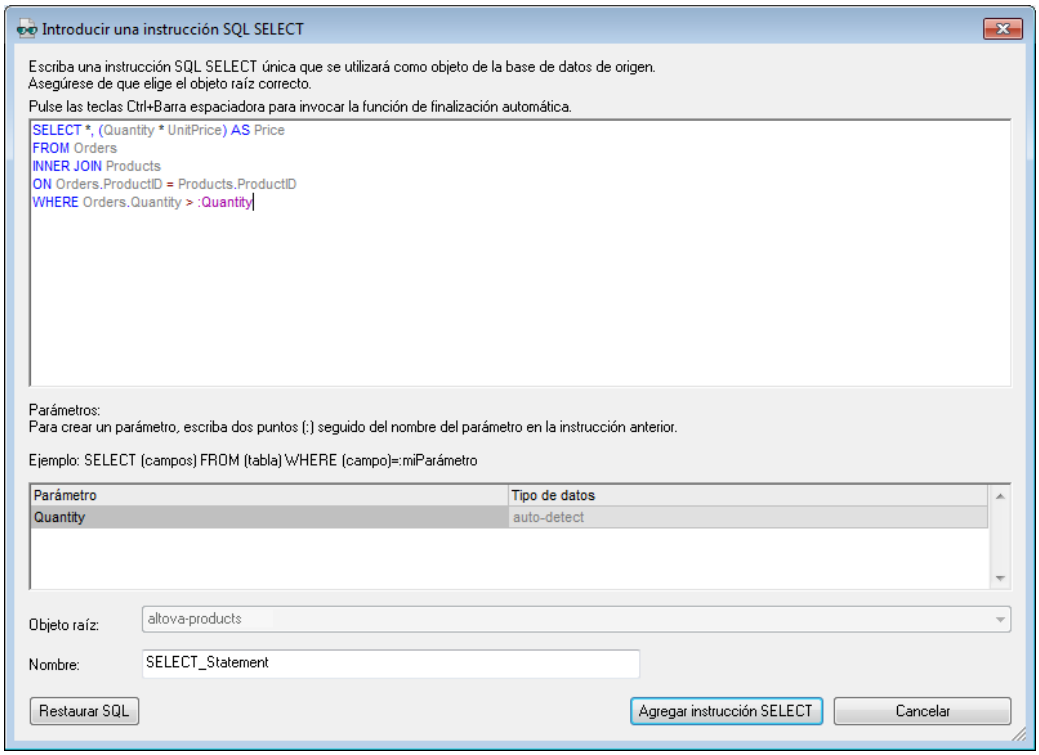


3. En el cuadro de diálogo "Insertar objetos de la base de datos" haga clic en el botón **Agregar/Editar instrucción SELECT** e introduzca esta consulta:

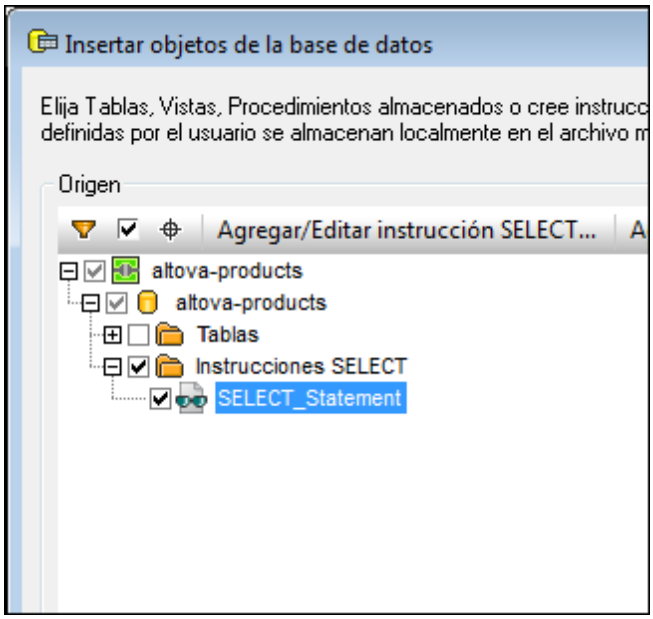
```
SELECT *, (Quantity * UnitPrice) AS Price
FROM Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
WHERE Orders.Quantity > :Quantity
```

Esta consulta usa una combinación interna entre las tablas Orders y Products y recupera todos los campos (*) y un valor calculado (`AS Price`). La consulta también especifica el parámetro **:Quantity** en la cláusula WHERE.

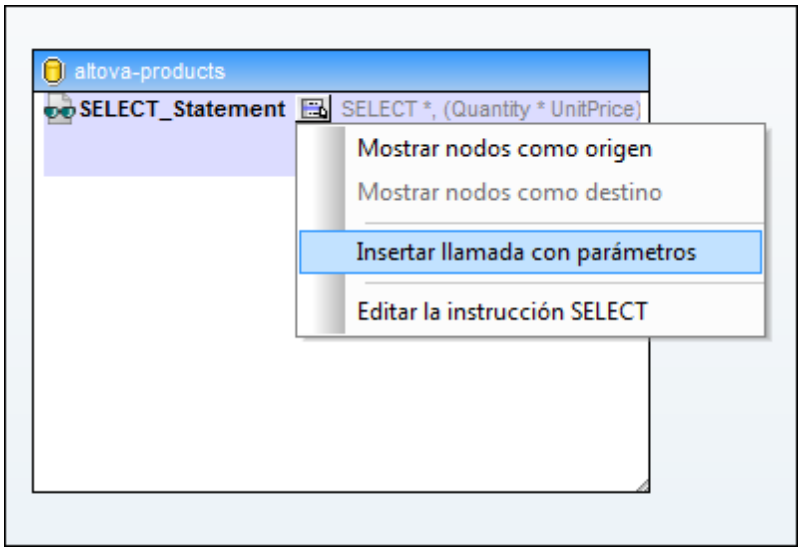
© 2018 Altova GmbH



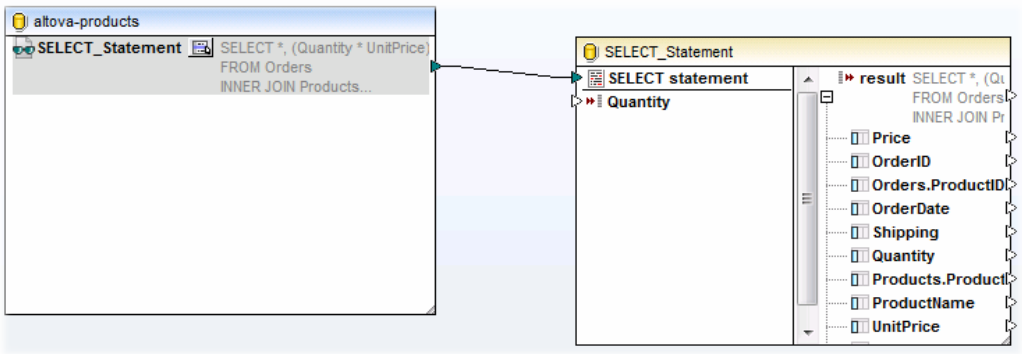
4. Ahora haga clic en el botón **Agregar instrucción SELECT**.



5. Haga clic en **Aceptar** para terminar. El componente **altova-products** se añadió al área de asignación.
6. Ahora haga clic en el botón  de este componente y seleccione **Insertar llamada con parámetros** en el menú contextual.



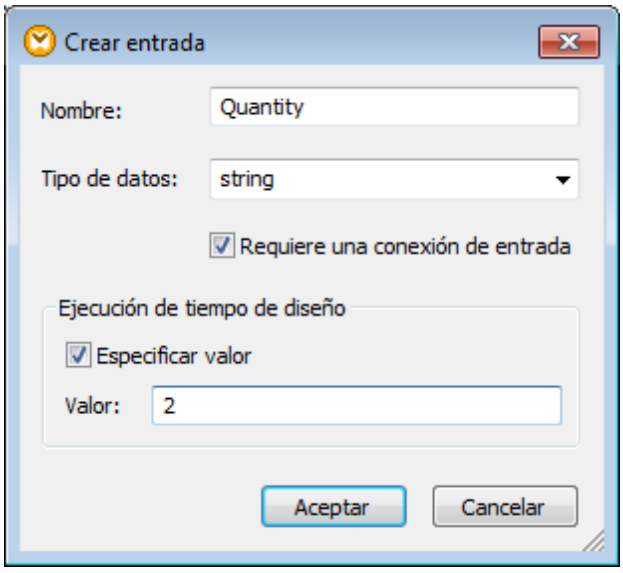
La asignación incluye ahora una estructura nueva (SELECT_Statement) que está dividida en dos paneles: el panel izquierdo aporta los conectores de entrada y el derecho los conectores de salida. Observe que el panel izquierda también incluye el parámetro **`Quantity`** que definimos en pasos anteriores.



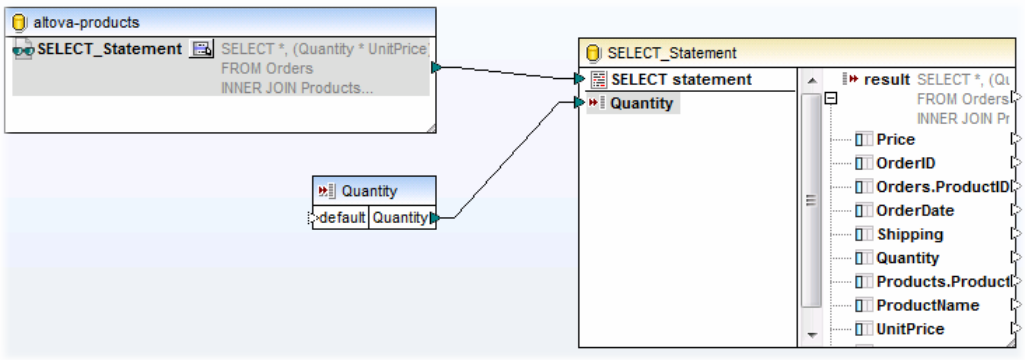
Paso n°2: agregar el parámetro de entrada

1. En el menú **Insertar** haga clic en el comando **Insertar componente de entrada**.
2. Introduzca el nombre "Quantity".
3. En el grupo de opciones *Ejecución de tiempo de diseño* introduzca el valor de parámetro que se debe usar para ejecutar la asignación durante la fase de diseño (p. ej. 2). Para más información consulte la sección Pasar parámetros a la asignación.

© 2018 Altova GmbH

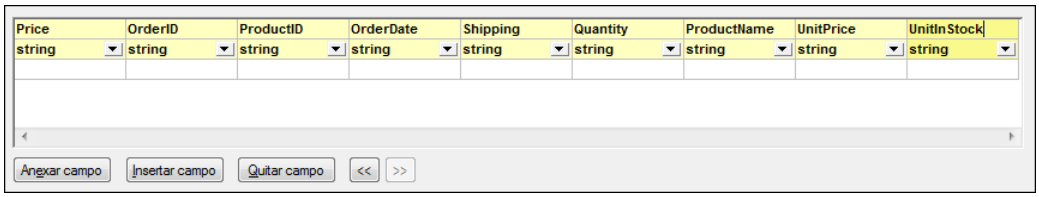


Ahora puede conectar el parámetro de entrada con la estructura de llamada a la base de datos (*imagen siguiente*).

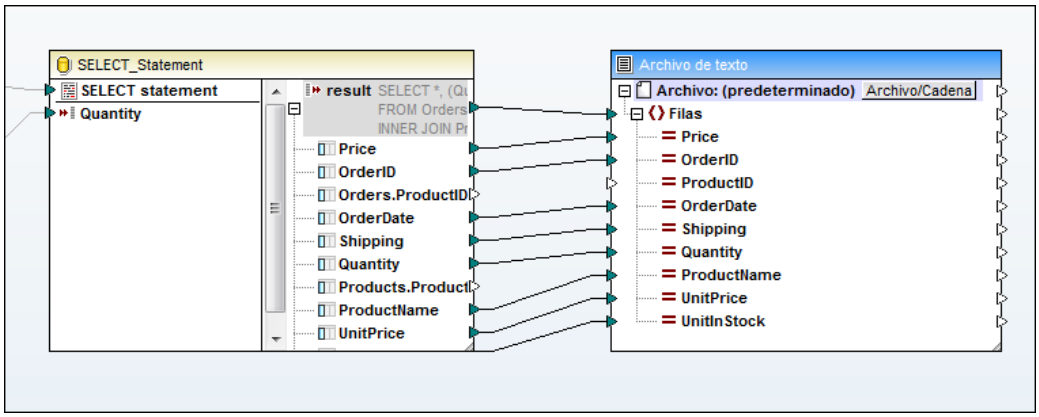


Paso n°3: agregar el componente CSV de destino

1. En el menú **Insertar** haga clic en el comando **Archivo de texto**.
2. Seleccione la opción *Utilizar procesamiento básico para archivos CSV estándar* y después haga clic en **Continuar**.
3. En el cuadro de diálogo "Configuración del componente" haga clic en el botón **Anexar campo** y añada nueve campos nuevos. Recomendamos que los campos CSV tengan el mismo nombre que los campos de la base de datos (*imagen siguiente*). Así podrá ahorrar tiempo a la hora de crear asignaciones. Consulte el apartado Opciones de configuración de componentes CSV para obtener más información.



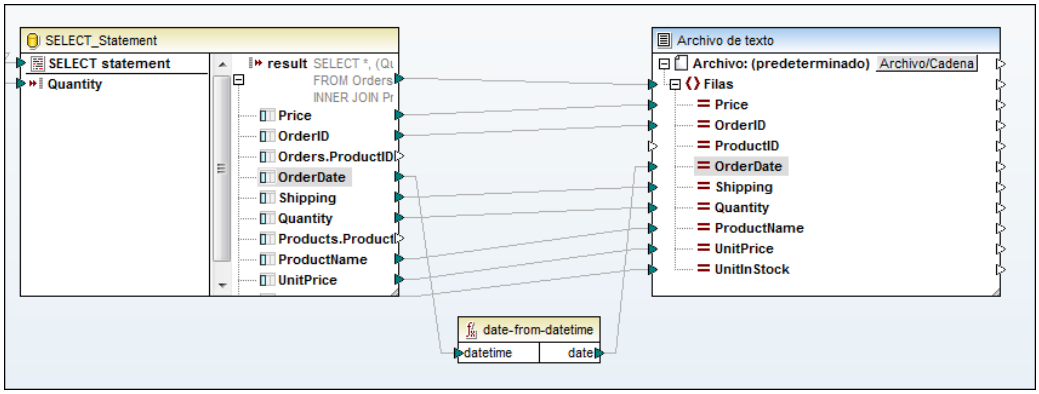
4. Cree una conexión entre el nodo **result** de la estructura SELECT y el nodo **Rows** del componente CSV.



Como la mayoría de los campos del componente CSV se llaman igual que su equivalente de la base de datos, cuando conecte **result** con **Rows**, probablemente MapForce cree las demás conexiones automáticamente. Si no lo hace, seleccione el comando Conexión y habilite la opción *Conectar automáticamente los secundarios equivalentes*. El único elemento que deberá conectar a mano será **ProductID** porque en la estructura SELECT no hay ningún campo con este nombre.

Paso n°4: convertir la fecha

En la ventana Bibliotecas busque la función `date-from-datetime` y arrástrela hasta el área de asignación. Después conecte su entrada y su salida tal y como se aprecia en esta imagen:

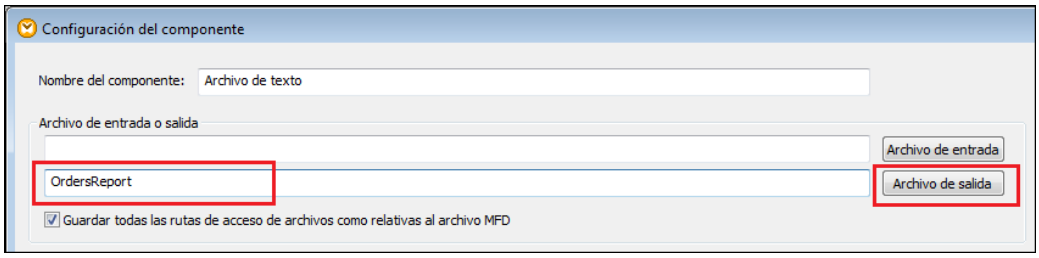


Paso n°5: establecer el nombre del archivo de salida

Para definir el nombre **OrdersReport.csv** para el archivo de salida basta con hacer doble clic en el componente CSV e introducir el valor en el cuadro de texto que corresponde al archivo de

 © 2018 Altova GmbH

salida *(marcado en rojo en la imagen siguiente)*.



7.2.6 Asignaciones entre datos XML y campos de BD

En MapForce puede crear asignaciones de datos con campos (columnas) de BD que almacenen contenido XML. Esto significa que los datos XML almacenados en el campo (columna) de BD se pueden extraer y escribir en cualquier otro tipo de estructura compatible con MapForce. Estas son las asignaciones de datos que se pueden crear en MapForce:

1. Asignaciones de datos con campos de un tipo XML especializado (p. ej. `Xml` en SQL Server y `XMLType` en Oracle). La lectura y escritura de datos XML en campos XML especializados es pertinente en bases de datos con compatibilidad nativa para XML (como IBM DB2, Oracle y SQL Server).
2. Asignaciones de datos con campos de texto que almacenen contenido XML (p. ej. `Text`, `Varchar`). Esto es pertinente en bases de datos con campos de texto de longitud suficiente como para almacenar un documento XML.

En ambos casos es necesario que exista un esquema XML válido para cada columna de BD a la que se desea asignar datos. Cuando una columna de BD almacena XML, MapForce ofrece la opción de asignar un esquema XML desde la BD directamente (si la BD lo admite) o seleccionar un esquema de un archivo externo. Puede asignar un esquema XML por cada columna de BD. Si el esquema tiene varios elementos raíz, podrá seleccionar un solo elemento raíz del esquema.

Cuando XML se almacena como campo de cadena en una BD, la codificación de caracteres del documento XML es la del campo de cadena subyacente. Si el campo de BD no almacena texto en Unicode, algunos caracteres no se podrán representar.

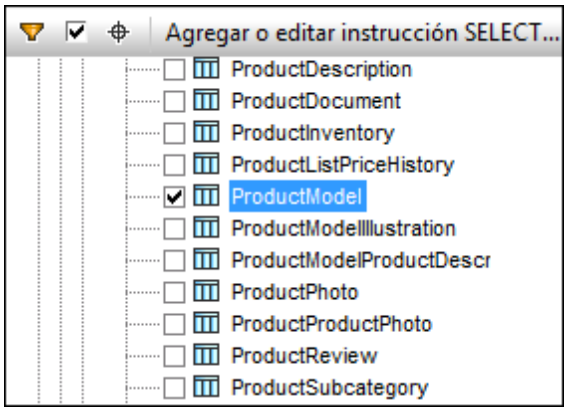
Algunas bases de datos admiten la codificación XML para campos XML (que no es necesariamente la misma que la del conjunto de caracteres de la BD). Si la base de datos lo admite, MapForce entiende que la declaración de codificación del documento XML es la que se declaró en el campo XML. Para más información sobre la compatibilidad de codificación XML consulte la documentación de cada base de datos.

7.2.6.1 Asignar un esquema XML a un campo de BD

En este tema explicamos cómo asignar un esquema a un campo que está definido de forma nativa como tipo XML en la base de datos. Para explicarlo usamos SQL Server 2017 y la base de datos Adventure Works 2014. Esta última se puede descargar del sitio web de CodePlex (https://msftdbprodsamples.codeplex.com/). Recuerde que la asignación de datos a campos XML funciona igual que la asignación a otros tipos de base de datos que admiten campos XML.

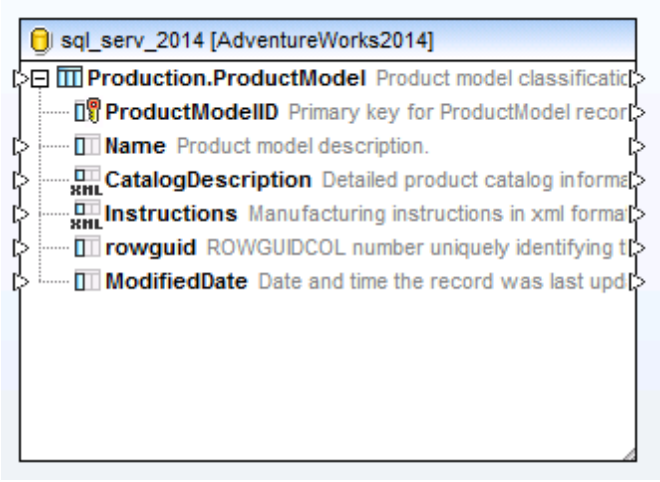
Para agregar la base de datos Adventure Works 2014 como componente de asignación:

1. En el menú Insertar haga clic en el comando Base de datos. Siga los pasos del Asistente para conectarse a la base de datos con ADO o con ODBC. Para más información consulte los temas Conectarse a Microsoft SQL Server (ADO) y Conectarse a Microsoft SQL Server (ODBC). **Nota:** si usa el controlador **SQL Server Native Client**, quizás necesite establecer un carácter de espaciado como valor de la propiedad **Seguridad integrada** (*véase Configurar las propiedades de vínculo de datos de SQL Server*).
2. En el cuadro de diálogo "Insertar objeto de base de datos" expanda el esquema **Production** y después marque la casilla de la tabla **ProductModel**.



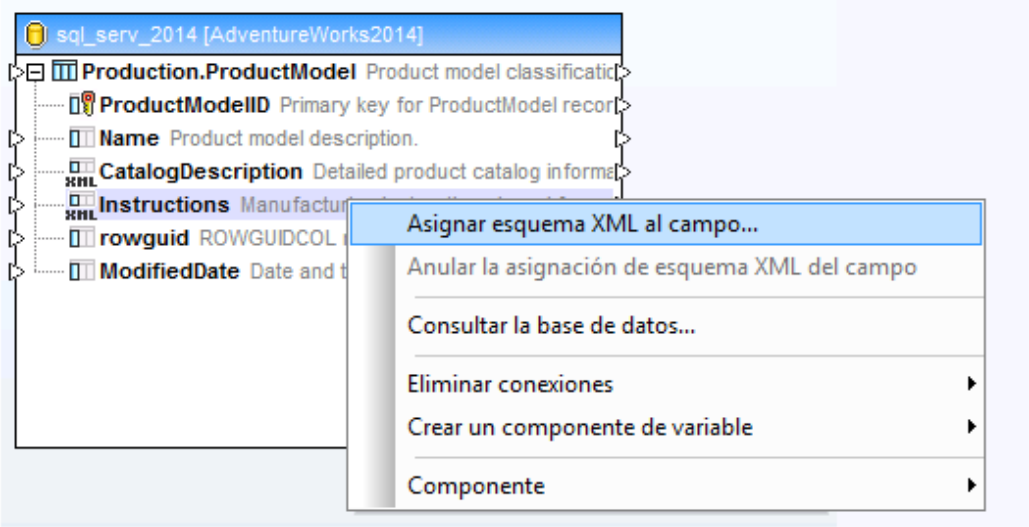
3. Para terminar haga clic en **Aceptar**.

La tabla de base de datos se añade al área de asignación. Observe que esta tabla tiene dos campos de tipo XML, **CatalogDescription** e **Instructions**:

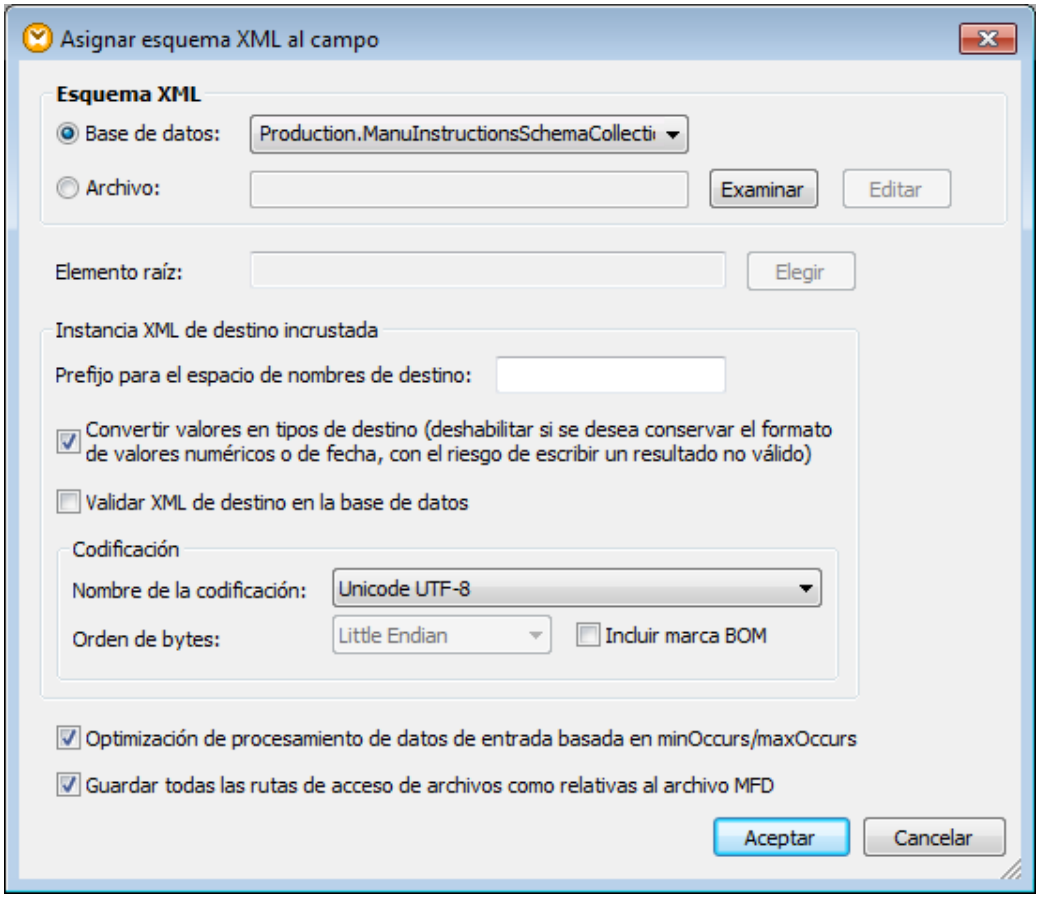


Para que la estructura de los campos XML aparezcan en la asignación es necesario el esquema XML del contenido del campo. Haga clic con el botón derecho en el campo **Instructions** y seleccione **Asignar esquema XML al campo** en el menú contextual.

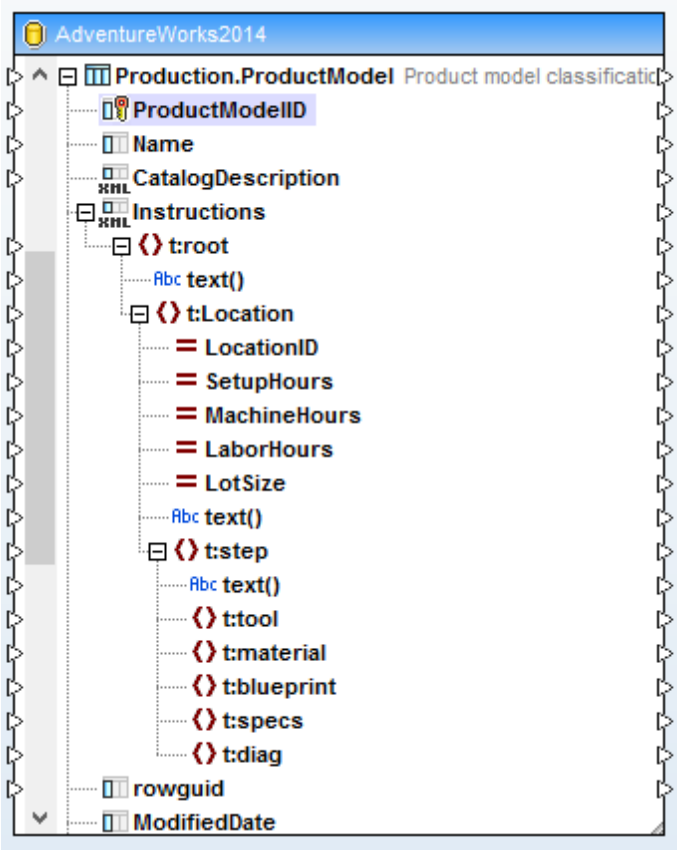
© 2018 Altova GmbH



En este ejemplo concreto asignaremos un esquema al campo **Instructions** desde la BD directamente. Para ello seleccionamos el elemento **Production.ManuInstructionsSchemaCollection** en el cuadro combinado de la opción *Base de datos:* y hacemos clic en **Aceptar**.



La estructura del campo XML ya aparece en el componente y se pueden crear conexiones (y asignaciones de datos) entre este campo y otros elementos.



7.2.6.2 Ejemplo: escribir datos XML a un campo SQLite

Este ejemplo explica paso a paso cómo crear una asignación de MapForce que lee datos de varios archivos XML y los escribe en una base de datos SQLite. El objetivo de la asignación es crear un registro de BD nuevo en la base de datos SQLite por cada archivo XML de origen. Cada registro almacenará el documento XML como campo TEXT.

Todos los archivos utilizados en este ejemplo están en la ruta de acceso `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`:

El archivo de diseño de asignación	• **XmlToSqliteField.mfd**
Los archivos XML de origen	• **bookentry1.xml** • **bookentry2.xml** • **bookentry3.xml**
El esquema XML utilizado para la validación	• **books.xsd**

© 2018 Altova GmbH

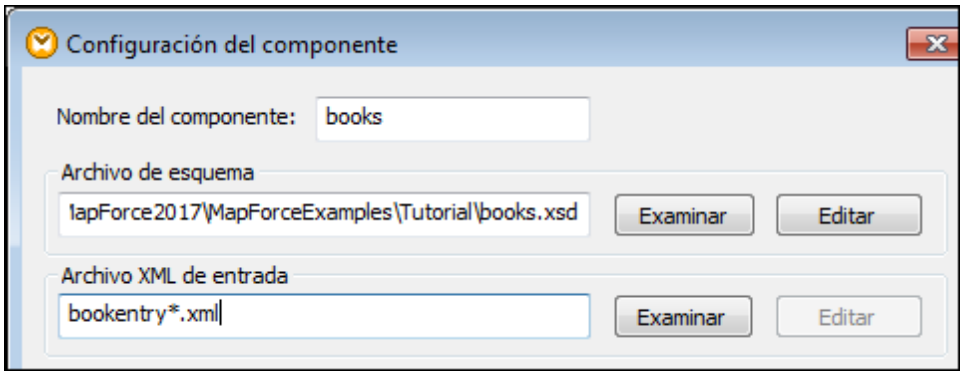
La base de datos SQLite de destino	• **Library.sqlite**

Para conseguir el objetivo de la asignación tendremos que seguir estas instrucciones:

1. Agregar el componente XML y configurarlo para que lea datos de varios archivos.
2. Agregar el componente de BD SQLite y asignar un esquema XML al campo TEXT de destino.
3. Crear las conexiones de asignación y configurar la acción INSERT de la base de datos.

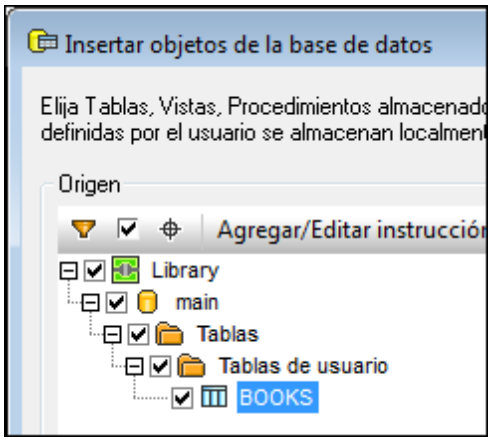
Paso nº1: agregar el componente XML

1. En el menú **Insertar** haga clic en el comando **Archivo o esquema XML...** y navegue hasta el esquema `books.xsd` ubicado en el directorio `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`. Cuando MapForce solicite un archivo XML de muestra, haga clic en **Omitir**. Cuando MapForce solicite un elemento raíz, seleccione **Books**.
2. Haga doble clic en el título del componente e introduzca **bookentry*.xml** en el cuadro de texto *Archivo XML de entrada*. Esto indica que MapForce debe leer todos los archivos XML del directorio de origen cuyo nombre empiece por "bookentry-". Para más información consulte la sección Procesar varios archivos de entrada o salida simultáneamente.

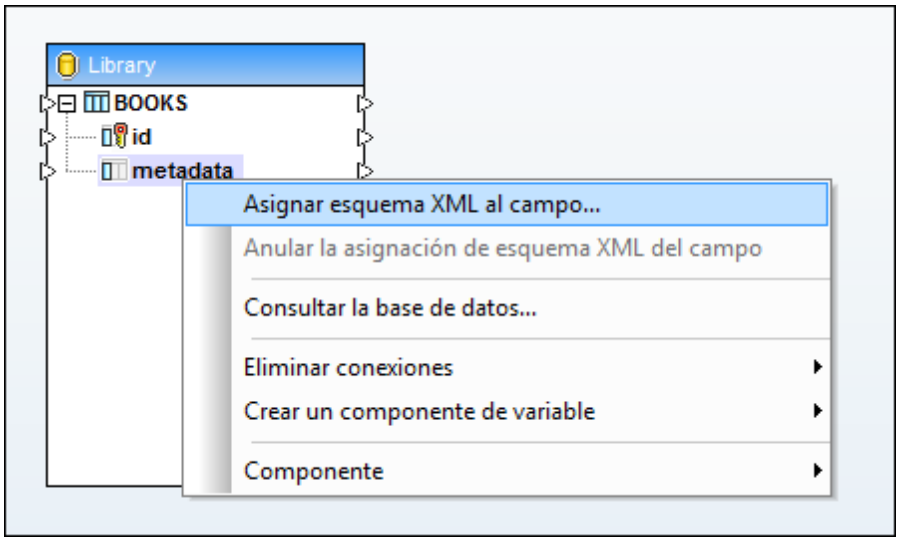


Paso nº2: agregar el componente SQLite

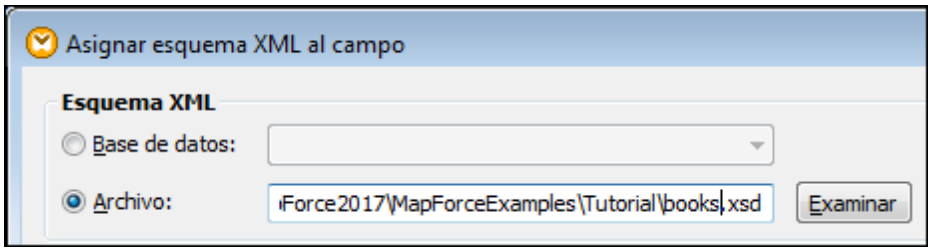
En el menú **Insertar** haga clic en **Base de datos** y siga los pasos del Asistente para conectarse al archivo de base de datos **Library.sqlite** del directorio `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\` directory (*véase Conectarse a una base de datos SQLite*). Llegada la hora de seleccionar objetos de la BD, seleccione la tabla `BOOKS`.



El campo de BD donde se escribirá el contenido XML se llama `metadata`. Para asignar un esquema XML a este campo haga clic con el botón derecho y seleccione **Asignar esquema XML al campo** en el menú contextual.

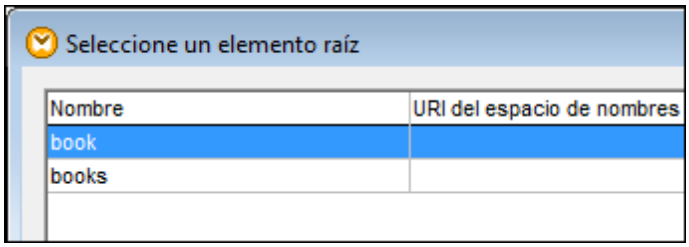


En este tutorial el esquema asignado al campo `metadata` es el mismo que se usa para validar los archivos XML de origen. Haga clic en **Examinar** y seleccione el esquema `books.xsd` del directorio `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`:



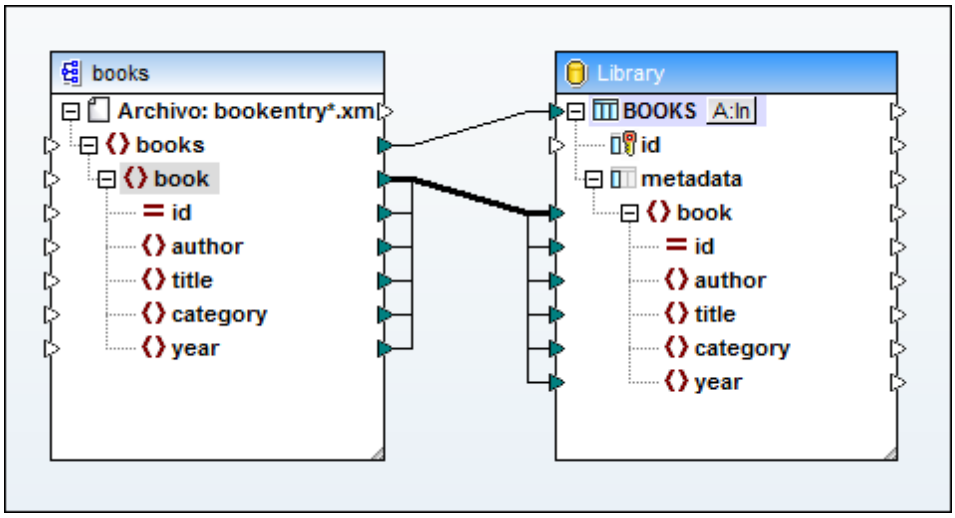
El esquema **books.xsd** tiene dos elementos con declaración global: `book` y `books`. En este ejemplo estableceremos `book` como elemento raíz de los datos XML escritos en el campo de la BD. Haga clic en **Elegir** y seleccione `book` como elemento raíz:

 © 2018 Altova GmbH



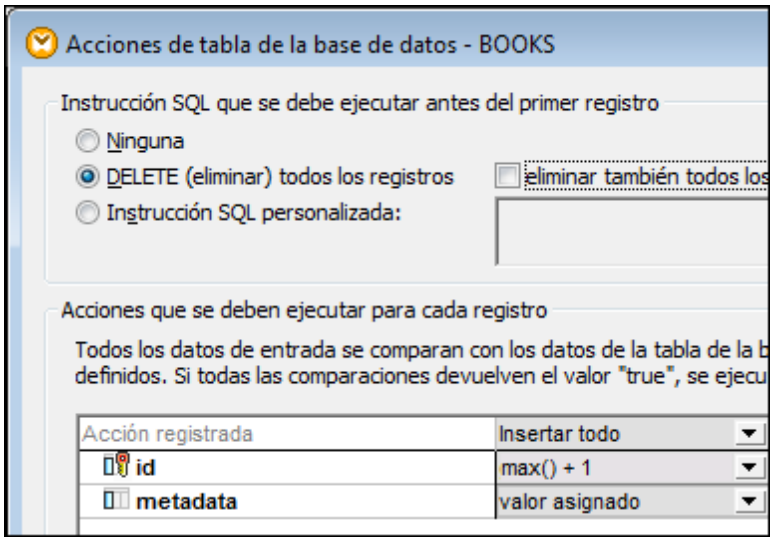
Paso nº3: crear las conexiones de asignación y configurar la acción INSERT

Cree estas conexiones de asignación:



Como se aprecia en la imagen anterior, la conexión entre `book` y `book` es una conexión de copia total porque tanto el origen como el destino usan el mismo esquema y los elementos secundarios tienen nombres idénticos. Para más información consulte el apartado Conexiones de copia total.

La conexión de nivel superior (entre `books` y `BOOKS`) recorre cada elemento `books` del origen y escribe un registro nuevo en la tabla BOOKS. Haga clic en el botón `A:In` del componente de BD y defina estas opciones de configuración para la actualización de la base de datos:



La opción **DELETE (eliminar) todos los registros** indica que MapForce debe eliminar el contenido de la tabla `BOOKS` antes de insertar otros registros.

Las acciones **Insertar todo** especifican que tendrá lugar una consulta `INSERT` de base de datos. El campo `id` se genera a partir de la BD propiamente dicha, mientras que el campo `metadata` se rellenará con el valor que viene dado por la asignación.

Recuerde que antes de ejecutar la asignación deberá guardarla.

Para ejecutar la asignación y ver el resultado generado haga clic en el panel *Resultados*. No olvide que esta acción no actualiza inmediatamente la base de datos. Cuando esté listo para ejecutar el script de base de datos generado, seleccione el comando de menú **Resultados | Ejecutar script SQL** (o haga clic en el botón de la barra de herramientas).

7.2.6.3 Ejemplo: extraer datos de columnas tipo XML de IBM DB2

Este ejemplo explica cómo extraer datos de columnas de bases de datos IBM DB2 de tipo XML y escribirlos en un archivo CSV de destino. También explica cómo usar instrucciones XQuery incrustadas en código SQL para recuperar datos XML de forma condicional. Para seguir este ejemplo se necesita tener acceso a una base de datos IBM DB2 con permiso para crear y rellenar tablas.

Primero debemos preparar la base de datos par que contenga datos XML. Esto se puede hacer desde la herramienta de administración propia de la base de datos o en MapForce directamente. para hacerlo en MapForce debe seguir estas instrucciones:

1. Cree una asignación nueva y haga clic en el panel *Consulta de la BD*.
2. Haga clic en el botón **Conexión rápida** y siga los pasos del asistente para crear una conexión de base de datos nueva (*véase* *Ejemplos de conexión a bases de datos*).
3. Pegue este texto en el editor SQL. Esta consulta SQL crea una tabla de base de datos llamada `ARTICLES` y la rellena con datos.

© 2018 Altova GmbH

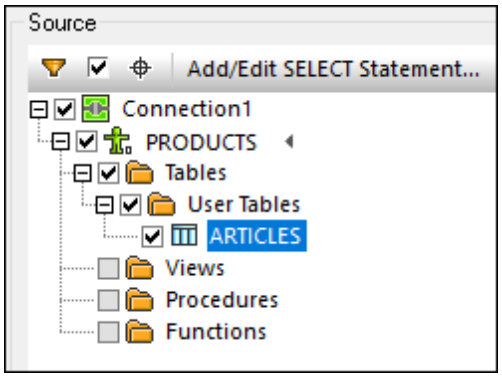
```
-- Create the table
CREATE TABLE
    ARTICLES (
        id INTEGER NOT NULL,
        article XML ) ;
-- Populate the table
INSERT INTO ARTICLES VALUES
   (1, '<Article>
      <Number>1</Number>
      <Name>T-Shirt</Name>
      <SinglePrice>25</SinglePrice>
   </Article>'),
(2, '<Article>
      <Number>2</Number>
      <Name>Socks</Name>
      <SinglePrice>230</SinglePrice>
   </Article>'),
(3, '<Article>
      <Number>3</Number>
      <Name>Pants</Name>
      <SinglePrice>34</SinglePrice>
   </Article>'),
 (4, '<Article>
      <Number>4</Number>
      <Name>Jacket</Name>
      <SinglePrice>5750</SinglePrice>
   </Article>');
```

4. Haga clic en el botón **Ejecutar**  . El resultado de la ejecución de la consulta aparece en la ventana Resultados. Si la consulta se ejecuta correctamente, la tabla recién creada contará con cuatro filas.

Ahora debemos crear una asignación de datos que recupere datos XML de forma condicional de la tabla `ARTICLES` que acabamos de crear. El objetivo es recuperar productos que tengan un importe superior a 100 en la columna `ARTICLES`.

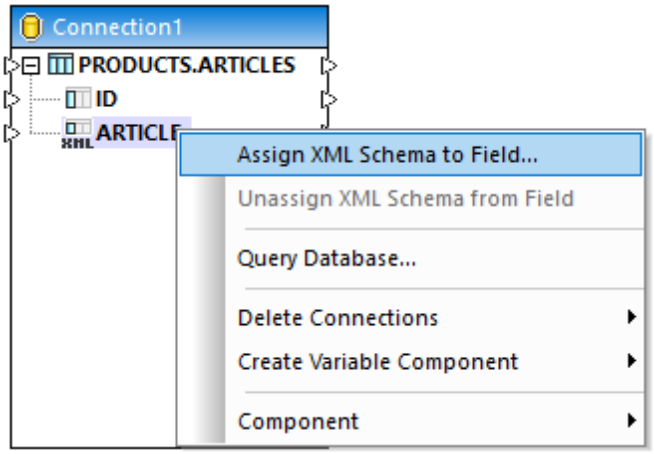
Paso nº1: agregar la base de datos

1. Vuelva al panel *Asignación*.
2. En el menú **Insertar** seleccione el comando **Base de datos** y siga los pasos del asistente para conectarse a la base de datos.
3. Cuando llegue el momento de seleccionar objetos de la base de datos, seleccione la tabla `ARTICLES` que se creó en el paso anterior.



Paso nº2: asignar el esquema al campo de tipo XML

1. Haga clic con el botón derecho en el elemento `ARTICLE` del componente y seleccione **Asignar esquema XML al campo** en el menú contextual.

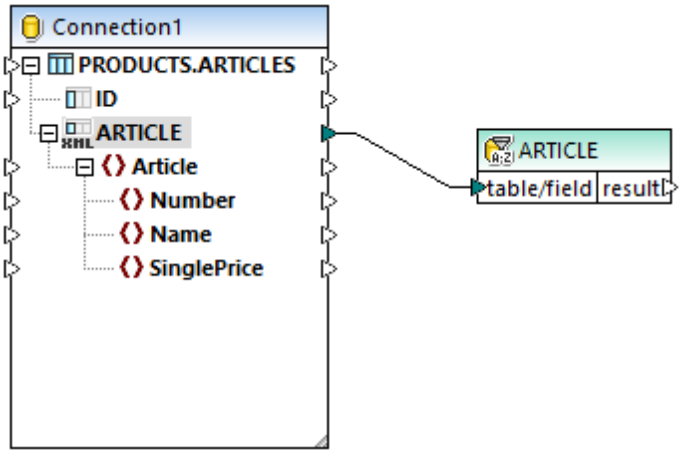


2. Seleccione *Archivo* y navegue hasta este esquema: **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial\DB2xsd.xsd**.

Paso nº3: agregar el componente WHERE/ORDER de SQL

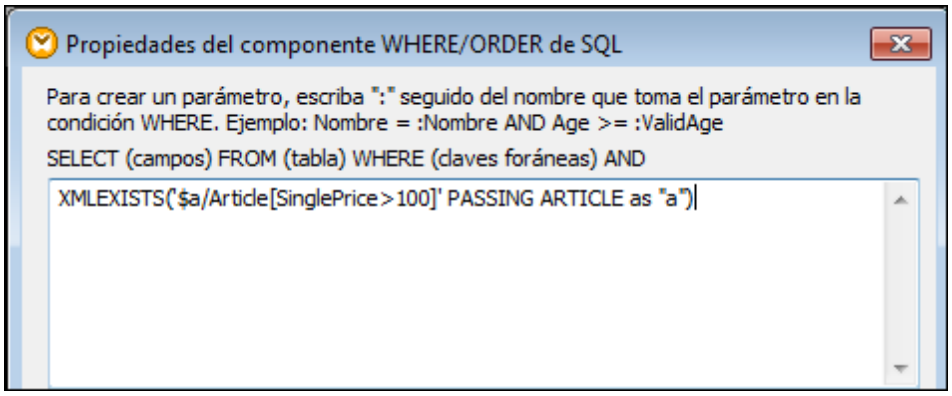
1. En el menú **Insertar** seleccione el comando **WHERE/ORDER de SQL**.
2. Conecte la columna de tipo XML `ARTICLE` a la entrada del componente WHERE/ORDER de SQL.

© 2018 Altova GmbH



3. En el cuadro de diálogo "Propiedades del componente WHERE/ORDER de SQL" introduzca este texto:

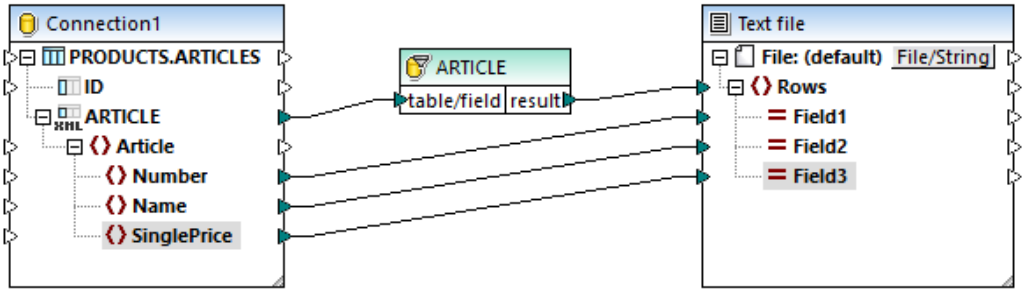
```
XMLEXISTS('$a/Article[SinglePrice>100]' PASSING ARTICLE as "a")
```



Este texto representa la parte WHERE de la consulta SQL. Cuando se ejecute la asignación, la parte WHERE se combinará con la parte SELECT que aparece en el cuadro de diálogo. Esta instrucción usa la función `XMLEXISTS` y sintaxis propia de las bases de datos IBM DB2.

Paso nº4: agregar el archivo CSV de destino

1. En el menú **Insertar** seleccione el comando **Archivo de texto**.
2. Después seleccione la opción *Utilizar procesamiento básico para archivos CSV....* y haga clic en **Continuar**.
3. Haga clic tres veces en **Anexar campo** para agregar tres campos que almacenarán el número de producto, el nombre y el precio respectivamente. Deje las demás opciones de configuración como están.
4. Dibuje las conexiones que aparecen en la imagen siguiente.



Ahora puede generar una vista previa de los resultados de la asignación de datos (abriendo el panel *Resultados*). Como esperábamos, en el resultado solo aparecen los productos con un precio superior a 100.

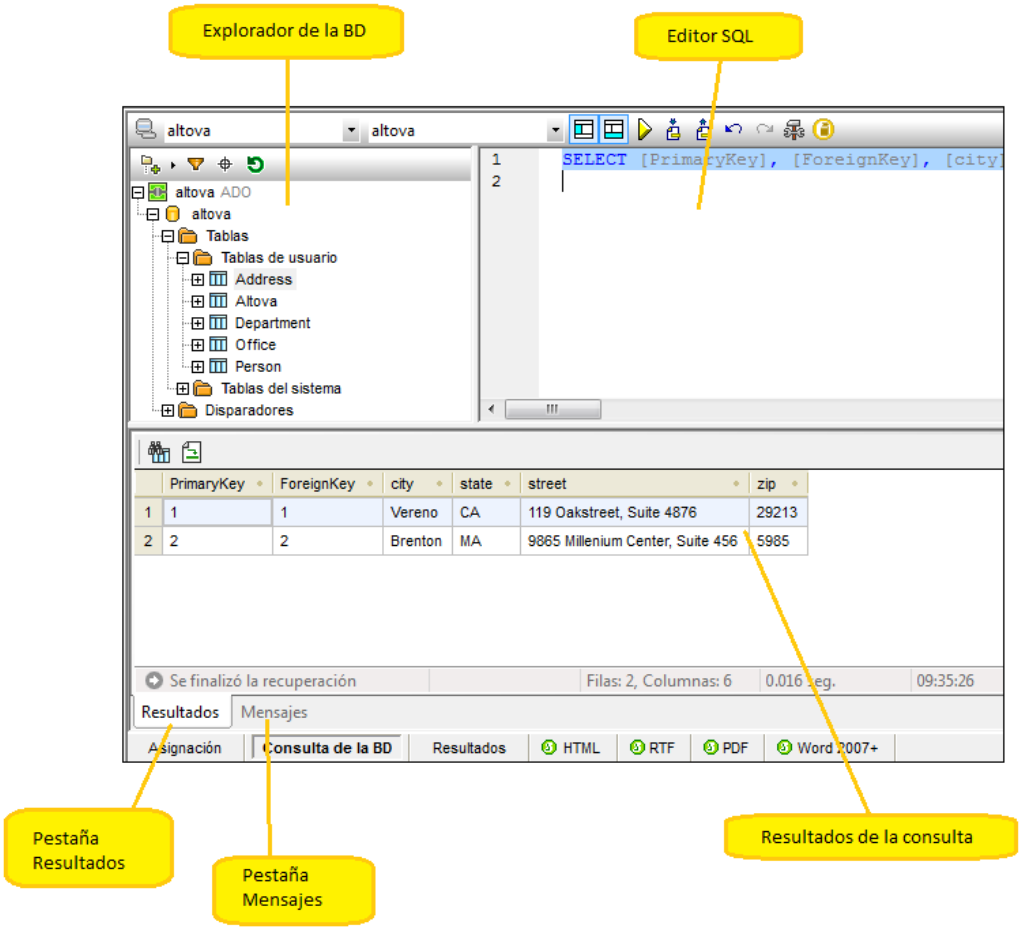
```
1    2,Socks,230
2    4,Jacket,5750
```

7.2.7 Examinar y consultar bases de datos

MapForce tiene un panel especial de consulta de bases de datos (llamado ***Consulta de la BD***) que sirve para consultar bases de datos por separado del proceso de asignación. Estas consultas directas no se guardan con la asignación en el archivo *.mfd, sino que son solo una manera de examinar y modificar el contenido de bases de datos desde MapForce directamente.

MapForce ofrece un panel de consulta de la BD por cada diseño de asignación que esté activo. Puede crear varias conexiones activas con varias bases de datos distintas dentro de cada panel *Consulta de la BD*. Recuerde que las conexiones creadas desde el panel *Consulta de la BD* no forman parte de la asignación y, por tanto, no se conservan cuando se cierra la aplicación, a no ser que las defina como recursos globales (*véase* *Recursos globales*).

© 2018 Altova GmbH



El panel *Consulta de la BD* se divide en varias vistas:

- **Explorador de la BD:** muestra información sobre la conexión y las tablas de la BD.
- **Editor SQL:** aquí puede escribir sus consultas SQL.
- **Pestaña *Resultados*:** muestra los resultados de las consultas en forma de tabla.
- **Pestaña *Mensajes*:** muestra advertencias y mensajes de error.

La parte superior del panel *Consulta de la BD* contiene botones de control para definir las bases de datos de trabajo, la conexión y los esquemas de la base de datos.

7.2.7.1 Seleccionar o conectarse a una base de datos

Por cada base de datos que necesite consultar deberá crear una conexión de base de datos. Si la asignación ya incluye un componente de base de datos, puede seleccionar la conexión de base de datos actual en la parte superior del panel *Consulta de la BD* (por defecto el panel tiene el estado *Sin conexión*) y empezar a explorar los objetos de la BD y a ejecutar consultas.

Si la asignación no incluye ningún componente de base de datos todavía o si desea conectarse a una base de datos nueva, haga clic en el botón **Conexión rápida** y siga los pasos del

asistente para crear una conexión de base de datos nueva (aquí puede ver algunos ejemplos). Si procede, también puede seleccionar una conexión actual de los recursos globales (*véase Recursos globales*).

Tras conectarse a la base de datos podrá crear consultas de base de datos de la siguiente manera:

- Importando la consulta SQL al panel Editor SQL desde un archivo SQL ya existente.
- Escribiendo la consulta en el panel Editor SQL.
- Haciendo clic con el botón derecho en un objeto del panel Explorador y generando una consulta (generalmente SELECT).

Cuando termine de crear la consulta en el panel Editor SQL, haga clic en el botón **Ejecutar**  . Los datos de la base de datos se recuperan y aparecen en el panel *Resultados* en forma de tabla. Recuerde que en la barra de estado aparecerá el mensaje 

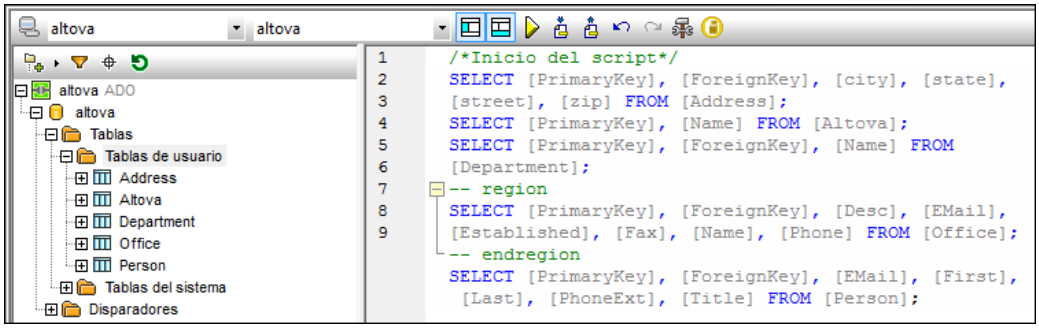
y cualquier otra información relevante sobre los resultados de la consulta.

Después de que aparezca el mensaje 

podrá buscar resultados, ordenarlos y copiarlos en el portapapeles (*véase Consulta de BD: vista de resultados y mensajes*).

7.2.7.2 Crear y editar instrucciones SQL

El Editor SQL sirve para escribir y ejecutar instrucciones SQL, tanto generadas automáticamente, como cargadas desde scripts SQL o escritas a mano. El editor SQL incluye características muy prácticas como la finalización automática (*véase Finalización automática*), las regiones y los comentarios en bloque o en línea.



Estos son los botones de la barra de herramientas del Editor SQL:

	**Mostrar u ocultar Explorador:** muestra u oculta el panel Explorador.
	**Mostrar u ocultar Resultados:** muestra u oculta la pestaña *Resultados*.
	**Ejecutar (F5)**: ejecuta las instrucciones SQL que estén seleccionadas en el editor. Si existen varias instrucciones y no se seleccionó ninguna, entonces se ejecutan todas.
	**Deshacer**: permite deshacer un número ilimitado de operaciones de edición en el Editor SQL.

© 2018 Altova GmbH

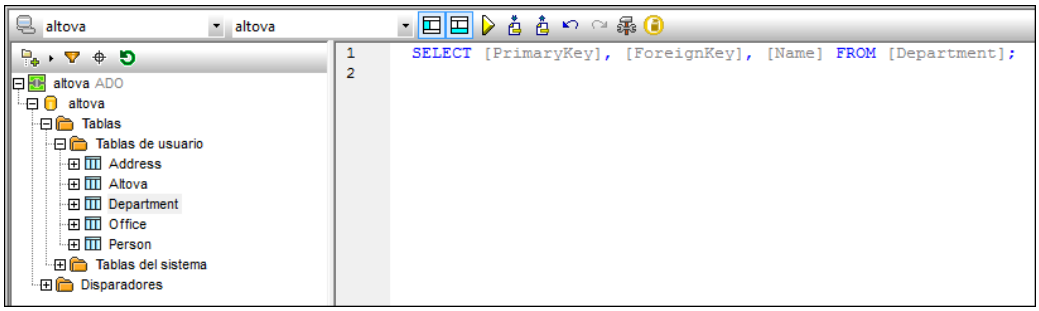
	Rehacer: permite rehacer comandos Deshacer previos. Con estos comandos puede avanzar y retroceder en el historial de operaciones Deshacer.
	Importar archivo SQL: abre un archivo SQL en el Editor SQL. A continuación se podrá ejecutar.
	Exportar archivo SQL: guarda las consultas SQL para poder utilizarlas más tarde.
	Abrir script SQL en DatabaseSpy: abre el script en el editor SQL de Altova DatabaseSpy.
	Opciones: abre el cuadro de diálogo "Opciones", donde se pueden definir opciones de configuración generales sobre la consulta de bases de datos y opciones de configuración del editor SQL.

7.2.7.2.1 Generar instrucciones SQL

Puede generar instrucciones SQL automáticamente desde el panel Explorador, cargarlas desde scripts o introducirlas a mano.

Para generar instrucciones SELECT SQL desde el Explorador:

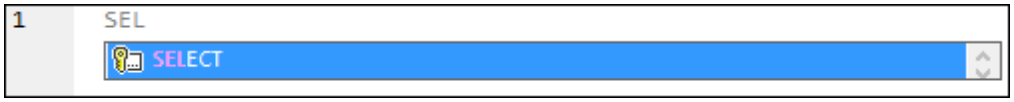
- Haga clic en un objeto de la BD (como una tabla o una vista) o en una carpeta del Explorador y arrástrela hasta el Editor SQL.



- Haga clic con el botón derecho en un objeto de la base de datos en el Explorador y seleccione **Mostrar en Editor SQL | SELECT** en el menú contextual.

Para crear instrucciones SQL a mano:

1. Empiece a teclear la instrucción SQL en el Editor SQL. Si la finalización automática está habilitada, aparecerá una lista desplegable con sugerencias mientras teclea en el editor.



2. Con las teclas de dirección **Arriba** y **Abajo** del teclado puede seleccionar la sugerencia y pulsar Entrar para insertar la opción resaltada (*véase* *Sugerencias de finalización automática SQL*).

7.2.7.2.2 Ejecutar instrucciones SQL

Las instrucciones SQL que aparecen en el Editor SQL se pueden ejecutar en la base de datos y con efecto inmediato. El resultado de la consulta SQL y el número de filas afectadas aparecen en la pestaña *Mensajes* del panel *Consulta de la BD*.

Cuando hay varias instrucciones SQL en el Editor SQL, solamente se ejecutarán las instrucciones que estén seleccionadas. Para seleccionar una instrucción SQL:

- Arrastre el puntero por encima de la instrucción sin dejar de pulsar el botón principal del ratón.
- Haga clic en un número de línea del Editor SQL.
- Haga triple clic en una instrucción.

Para ejecutar una instrucción SQL:

1. Introduzca o seleccione la instrucción SQL en el Editor SQL (*véase Generar instrucciones SQL*).
2. Haga clic en el botón **Ejecutar**  .

7.2.7.2.3 Importar y exportar scripts SQL

Puede guardar el código SQL de la ventana Editor SQL en un archivo y usarlo más tarde.

Para exportar el contenido del Editor SQL a un archivo:

- Haga clic en **Exportar archivo SQL**  e introduzca un nombre para el script SQL.

Para importar un archivo SQL:

- Haga clic en **Importar archivo SQL**  y seleccione el archivo SQL que desea abrir.

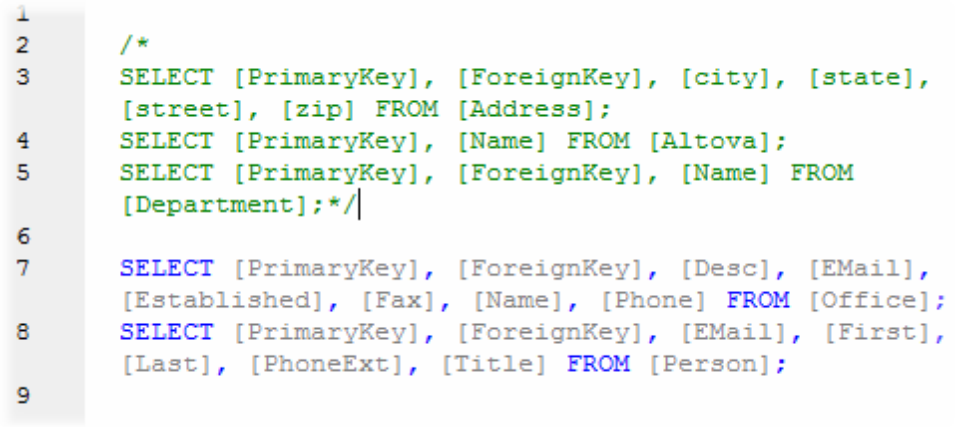
7.2.7.2.4 Agregar y quitar comentarios SQL

En el editor SQL de MapForce puede convertir instrucciones, parte de instrucciones o grupos de instrucciones en comentarios. Las instrucciones convertidas en comentarios se ignoran cuando se ejecuta el script SQL.

© 2018 Altova GmbH

Para convertir una sección del texto en comentario:

1. Seleccione una instrucción o parte de la instrucción.
2. Haga clic con el botón derecho en el texto seleccionado y elija la opción **Convertir el bloque en comentario/quitar comentario**.



Para convertir una línea en comentario:

Haga clic con el botón derecho en la posición que desea convertir en comentario y seleccione la opción **Convertir la línea en comentario/quitar comentario**. A partir de la posición del cursor la instrucción se convierte en comentario.

Para quitar un comentario (de bloque o de línea):

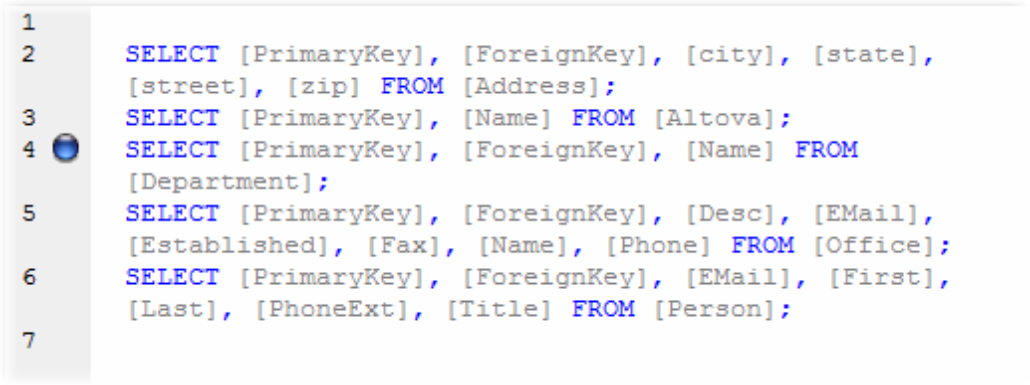
1. Seleccione la parte de la instrucción que es un comentario.
   Si quiere quitar un comentario de línea, basta con borrar las marcas de comentario - - situadas al principio del comentario.
2. Haga clic con el botón derecho y seleccione la opción **Convertir el bloque (o la línea) en comentario/quitar comentario**.

7.2.7.2.5 Uso de marcadores

Los marcadores sirven para marcar puntos de interés en los scripts de gran tamaño.

Para agregar un marcador:

Haga clic con el botón derecho en la línea en la que desea insertar el marcador y seleccione **Insertar o quitar marcador** en el menú contextual.



El icono aparece en el margen de la línea que incluye el marcador.

Para quitar un marcador:

Haga clic con el botón derecho en la línea que incluye el marcador que desea eliminar y seleccione **Insertar o quitar marcador** en el menú contextual.

Para navegar de un marcador a otro:

- Para pasar al siguiente marcador, haga clic con el botón derecho y seleccione **Ir al siguiente marcador**.
- Para ir al marcador anterior, haga clic con el botón derecho y seleccione **Ir al marcador anterior**.

Para quitar todos los marcadores:

Haga clic con el botón derecho en el **Editor SQL** y seleccione **Quitar todos los marcadores**.

7.2.7.2.6 Insertar regiones

Las regiones son partes del texto que se marcan y se declaran como una unidad. Estas unidades sirven para estructurar los scripts SQL. Las regiones se pueden expandir y contraer para mostrar u ocultar partes del script SQL. También se pueden anidar regiones dentro de otras.

Cuando se crea una región, se inserta un icono de expansión/contracción y el comentario - -

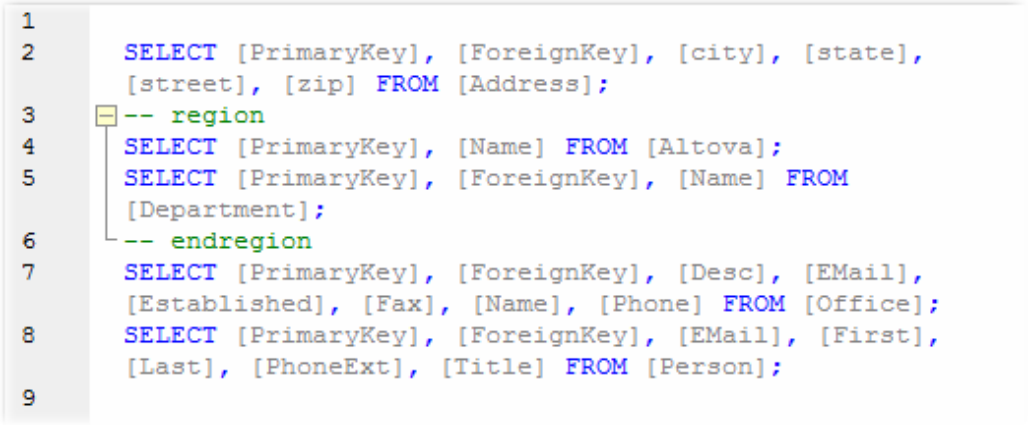
© 2018 Altova GmbH

`region` antes del texto seleccionado.

Nota: puede cambiar el nombre de una región anexando un texto descriptivo al comentario `-- region`. Pero no elimine la palabra `region`. Un ejemplo sería `-- region DB2query`.

Para crear una región:

1. En el **Editor SQL** seleccione las instrucciones que desea insertar en una región.
2. Haga clic con el botón derecho y seleccione la opción **Agregar región** del menú contextual.
   Las instrucciones seleccionadas se convierten en una región que se puede expandir o contraer.



3. Haga clic en el icono **+** o **-** para expandir o contraer la región.

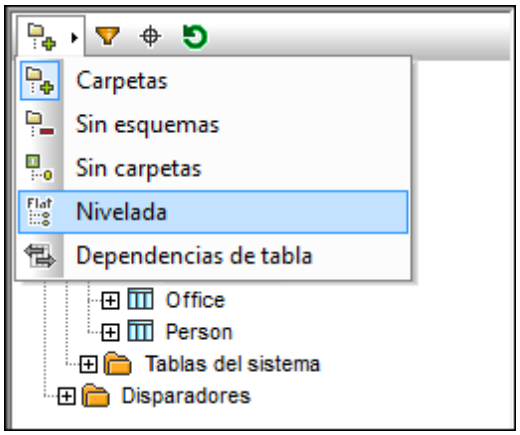
Para quitar una región:

Elimine las marcas de comentario `-- region` y `-- endregion`.

7.2.7.3 Examinar objetos de la base de datos

El panel Explorador muestra con detalle los objetos de las bases de datos que están conectadas, incluyendo sus tablas, vistas, procedimientos, etc. Además, cuando se trata de una base de datos compatible con XML, el Explorador muestra los esquemas XML registrados en una carpeta distinta.

En la parte superior del panel Explorador puede elegir una disposición para todos estos objetos.



Para seleccionar una disposición haga clic en el botón **Disposición en carpetas** y elija una opción de la lista desplegable. Observe que el aspecto del botón cambia dependiendo de la opción elegida.

- La **Disposición en carpetas** organiza los objetos de la BD en carpetas en función del tipo de objeto y estas carpetas aparecen organizadas de forma jerárquica (esta es la disposición predeterminada).
- La **Disposición sin esquemas** es similar a la disposición en carpetas, pero no presenta carpetas para los esquemas de BD. Por tanto, las tablas no se pueden categorizar según el esquema de la BD.
- La **Disposición sin carpetas** muestra todos los objetos de la BD en una jerarquía pero sin usar carpetas.
- La **Disposición nivelada** divide los objetos de la BD por tipos en el primer nivel jerárquico. Por ejemplo, las columnas no aparecen dentro de su tabla, sino que todas las columnas aparecen en una carpeta llamada `Columnas`.
- La **Disposición según dependencias** categoriza las tablas en función de la relación que tengan con las demás tablas. Hay categorías para tablas con claves foráneas, para tablas a las que hagan referencia claves foráneas y para tablas que no tengan relación alguna con otras tablas.

Desde el Explorador también puede:

- Generar instrucciones SQL.
- Filtrar y buscar objetos de la base de datos).
- Dividir las tablas en tablas del sistema y tablas del usuario.
- Actualizar el objeto raíz del origen de datos activo.

Para dividir tablas en tablas del sistema y tablas del usuario:

- En el Explorador haga clic con el botón derecho en la carpeta `Tablas` y seleccione **Ordenar por tablas de usuario y del sistema** en el menú contextual.

  Nota: esta función solo funciona si está elegida la *disposición en carpetas*, la *disposición sin esquemas* o la *disposición nivelada*.

© 2018 Altova GmbH

Para actualizar el objeto raíz del origen de datos activo:

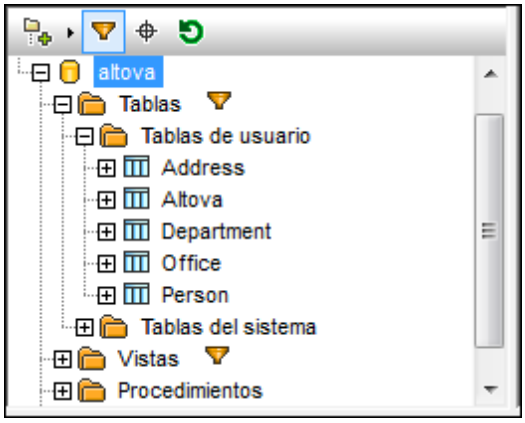
- Haga clic en el botón **Actualizar** de la barra de herramientas del Explorador.

7.2.7.3.1 Filtrar y buscar objetos de la base de datos

En el Explorador puede filtrar los objetos de base de datos (esquemas, tablas, vistas, etc.) por nombre o parte del nombre. Los objetos se filtran a medida que se teclea y el filtrado tiene en cuenta el uso de mayúsculas y minúsculas por defecto. Nota: recuerde que la función de filtrado no funciona con la *disposición sin carpetas*.

Filtrar objetos de base de datos

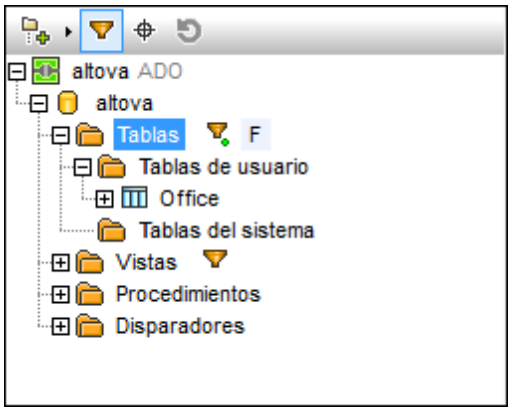
1. En la barra de herramientas del Explorador haga clic en **Filtrar contenido de carpeta** . Ahora aparece un icono en forma de embudo junto a cada una de las carpetas de la jerarquía.



2. Haga clic en el icono en forma de embudo de la carpeta que desea filtrar y seleccione una opción de filtrado en el menú contextual (p. ej. **Contiene**).



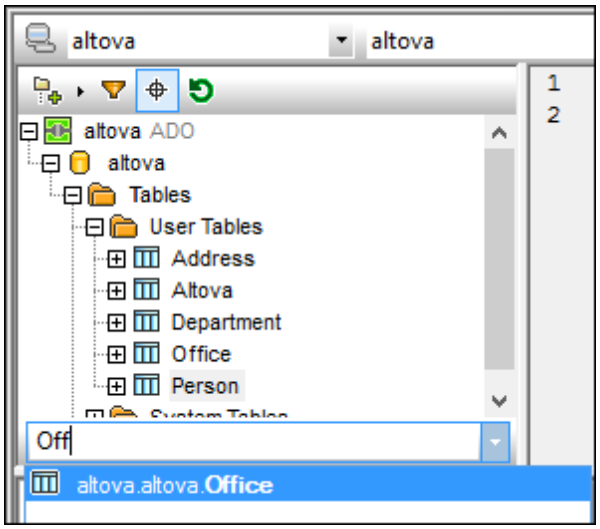
3. En el campo vacío que aparece junto al icono en forma de embudo introduzca el texto de búsqueda (p. ej. "F"). Los resultados aparecen a medida que se teclea.



Buscar objetos de base de datos

Para buscar por su nombre un elemento concreto de la base de datos puede usar funciones de filtrado o el **localizador de objetos**. Para buscar elementos de la base de datos con el localizador de objetos:

1. En la barra de herramientas del Explorador haga clic en **Localizador de objetos** .
2. En la lista desplegable que aparece introduzca el texto de búsqueda (p. ej. "Off").



3. Haga clic en un objeto de la lista de resultados para seleccionarlo en el Explorador.

7.2.7.3.2 Opciones contextuales del explorador

Las opciones del menú contextual del Explorador dependerán del tipo de objeto que esté seleccionado. Por ejemplo:

- Si se hace clic con el botón derecho en un objeto raíz, el menú contextual incluye la opción de actualizar la base de datos.

© 2018 Altova GmbH

- Si se hace clic con el botón derecho en una carpeta, el menú contextual incluye la opción **Expandir | Elementos del mismo nivel/Secundarios** y la opción **Contraer | Elementos del mismo nivel/Secundarios**.
- Si se hace clic con el botón derecho en un objeto de la base de datos, el menú contextual incluye el comando **Mostrar en Editor SQL** y algunos subcomandos (ver más abajo).

Para seleccionar varios objetos de la base de datos utilice **Mayús+Clic** o **Ctrl+Clic**.

Nota: la sintaxis de las instrucciones SQL puede depender de la base de datos que se utilice. La sintaxis utilizada en los siguientes ejemplos corresponde a Microsoft SQL Server 2014.

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual del objeto raíz:

- **CREATE**: crea una instrucción CREATE para el objeto raíz seleccionado. Por ejemplo:
  ```
  CREATE DATABASE [MYDB]
  ```
- **DROP**: crea una instrucción DROP para el objeto raíz seleccionado. Por ejemplo:
  ```
  DROP DATABASE [MYDB]
  ```

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de tablas y vistas:

- **SELECT**: crea una instrucción SELECT que recupera datos de todas las columnas de la tabla de origen. Por ejemplo:
  ```
  SELECT [DepartmentID], [Name], [GroupName], [ModifiedDate] FROM [MYDB].[HumanResources].[Department]
  ```
- **Nombre**: devuelve el nombre de la tabla.
- **Ruta de acceso**: devuelve la ruta de acceso completa de las tablas, con este formato `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla`.

Si seleccionó varias tablas, sus nombres y rutas de acceso se imprimen en líneas distintas y separadas por comas.

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de columnas:

- **SELECT**: crea una instrucción SELECT que recupera datos de todas las columnas seleccionadas de la tabla primaria. Por ejemplo:
  ```
  SELECT [DepartmentID] FROM [MYDB].[HumanResources].[Department]
  ```
- **Nombre**: devuelve el nombre de la columna seleccionada.
- **Ruta de acceso**: devuelve la ruta de acceso completa de la columna, con este formato `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla.NombreColumna`

Si seleccionó varias columnas, sus nombres y rutas de acceso se imprimen en líneas distintas y separadas por comas.

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de restricciones:

- **Nombre**: devuelve el nombre de la restricción seleccionada.
- **Ruta de acceso**: devuelve la ruta de acceso completa de la restricción, con este formato `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla.NombreRestricción`

Si seleccionó varias restricciones, sus nombres y rutas de acceso se imprimen en líneas distintas y separadas por comas.

Estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de índices:

- **Nombre**: devuelve el nombre del índice seleccionado.
- **Ruta de acceso**: devuelve la ruta de acceso completa del índice, con este formato `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla.NombreÍndice`

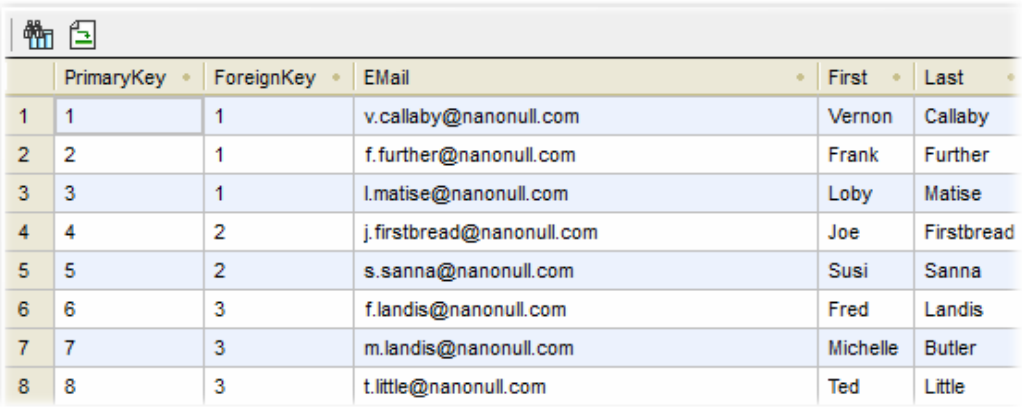
Si seleccionó varios índices, sus nombres y rutas de acceso se imprimen en líneas distintas y separadas por comas.

Si la base de datos admite esquemas XML, estas opciones corresponden al comando **Mostrar en Editor SQL** del menú contextual de los esquemas de la carpeta `Esquemas XML`:

- **Ver en XMLSpy**: abre el esquema de base de datos en XMLSpy (siempre y cuando esté instalado).
- **Administrar esquemas XML**: abre un cuadro de diálogo donde puede agregar o eliminar esquemas XML de base de datos.

7.2.7.4 Copiar, ordenar y buscar resultados de la consulta

La vista de resultados del panel *Consulta de la BD* muestra el conjunto de registros que se recuperó como resultado de la consulta de la base de datos.



	PrimaryKey	ForeignKey	EMail	First	Last
1	1	1	v.callaby@nanonull.com	Vernon	Callaby
2	2	1	f.further@nanonull.com	Frank	Further
3	3	1	l.matise@nanonull.com	Loby	Matise
4	4	2	j.firstbread@nanonull.com	Joe	Firstbread
5	5	2	s.sanna@nanonull.com	Susi	Sanna
6	6	3	f.landis@nanonull.com	Fred	Landis
7	7	3	m.landis@nanonull.com	Michelle	Butler
8	8	3	t.little@nanonull.com	Ted	Little

© 2018 Altova GmbH

Los botones de la barra de herramienta sirven para navegar por los resultados e instrucciones SQL y realizar búsquedas dentro de la tabla de resultados.

	**Buscar:** permite buscar determinado texto dentro de los resultados de la tabla. Pulse **F3** para ir a la siguiente instancia del término de búsqueda.
	**Ir a la instrucción**: salta al Editor SQL y resalta la instrucción SQL que produjo el resultado actual. Esta función puede ser muy práctica si el Editor SQL contiene un gran número de instrucciones.

Para seleccionar celdas de la tabla de resultados:

- Haga clic en el encabezado de columna para seleccionar una columna entera.
- Haga clic en un número de fila para seleccionar una fila entera.
- Haga clic en las celdas una por una o pulse la tecla **Ctrl** mientras hace clic para seleccionar varias celdas. Si una columna o celda contiene datos XML, también podrá copiar estos datos.

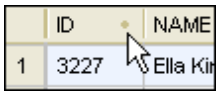
  Nota: también puede usar el menú contextual para seleccionar datos (**Seleccionar | Fila / Columna / Todo**.)

Para copiar las celdas seleccionadas en el portapapeles:

- Haga clic con el botón derecho y seleccione **Copiar celdas seleccionadas** en el menú contextual.

Para ordenar los datos.

- Haga clic con el botón derecho en cualquier parte de la columna que desea ordenar y seleccione **Orden | Ascendente** o **Descendente**.
- Haga clic en el icono de ordenación situado en el encabezado de la columna.



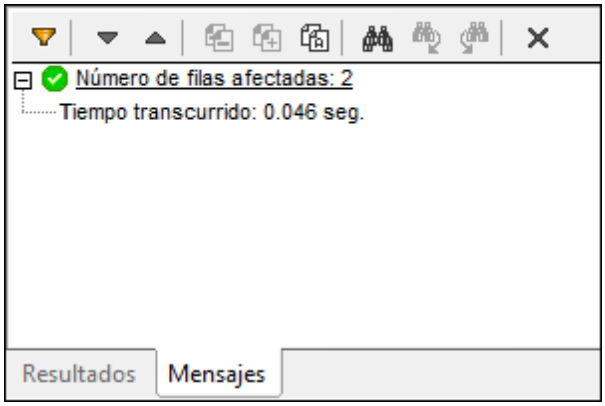
  Los datos se ordenan en función del contenido de la columna.

Para restaurar el orden predeterminado:

- Haga clic con el botón derecho en cualquier parte de la tabla y seleccione **Orden | Restaurar valor predeterminado** en el menú contextual.

7.2.7.5 Ver el estado de las consultas ejecutadas

La pestaña *Mensajes* del panel *Consulta de la BD* muestra información sobre la última instrucción SQL que se ejecutó y notifica errores y advertencias.



Puede personalizar la vista de la pestaña *Mensajes* usando diferentes filtros o navegar por los datos de fila en fila con los botones **Siguiente** y **Anterior**.

Estos son los comandos de la barra de herramientas de la pestaña *Mensajes*.

	**Filtrar**: abre un menú emergente donde puede filtrar el tipo de mensajes (**Resumen**, **Operación correcta**, **Advertencia**, **Error**, **Inserción automática**, **Progreso**). El tipo "Inserción automática" se refiere a aquellos mensajes provocados al insertar instrucciones SQL o construcciones SQL automáticamente en el editor SQL. El tipo "Progreso" indican el resultado de la conexión con la base de datos y el resultado de analizar datos SQL y cargar la estructura de datos. El menú emergente también incluye las opciones **Activar todos** y **Desactivar todos**.
	**Siguiente**: pasa al siguiente mensaje y lo resalta.
	**Anterior**: pasa al mensaje anterior y lo resalta.
	**Copiar el mensaje seleccionado en el portapapeles**
	**Copiar el mensaje seleccionado en el portapapeles (incluyendo sus elementos secundarios)**
	**Copiar todos los mensajes en el portapapeles**
	**Buscar**: abre el cuadro de diálogo "Buscar".
	**Buscar anterior**: pasa a la instancia anterior de la cadena especificada en el cuadro de diálogo "Buscar".
	**Buscar siguiente**: pasa a la instancia siguiente de la cadena especificada en el cuadro de diálogo "Buscar".

© 2018 Altova GmbH

	**Borrar**: elimina todos los mensajes de la pestaña *Mensajes* del Editor SQL.

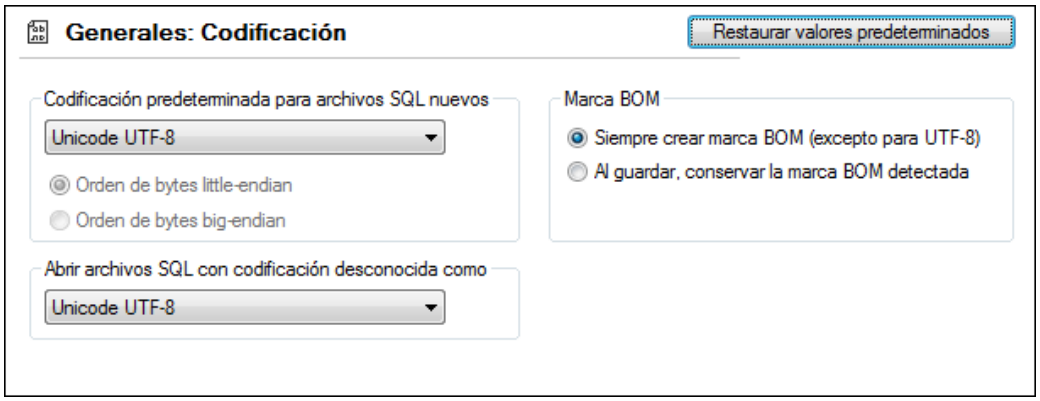
7.2.7.6 Configurar la consulta de la base de datos

En esta sección encontrará información sobre cómo configurar la edición de instrucciones SQL, la carga de scripts en el Editor SQL y la visualización de resultados de consultas.

7.2.7.6.1 Codificación del archivo SQL

Puede configurar las opciones de codificación para los archivos SQL que se crean o abren con el Editor SQL:

1. Haga clic en el panel *Consulta de la BD*.
2. En la parte superior del panel de consultas haga clic en el botón **Opciones**  y después en **Generales | Codificación**.



Codificación predeterminada para archivos SQL nuevos

Define la codificación predeterminada para archivos nuevos para que cada documento nuevo incluya la codificación que se especifique aquí. Si selecciona una codificación de dos o cuatro bytes como codificación predeterminada (p. ej. UTF-16, UCS-2 o UCS-4), también podrá elegir entre el orden de bytes little-endian y big-endian.

La codificación de los archivos que ya existen no se ve afectada por esta opción de configuración.

Abrir archivos SQL con codificación desconocida como

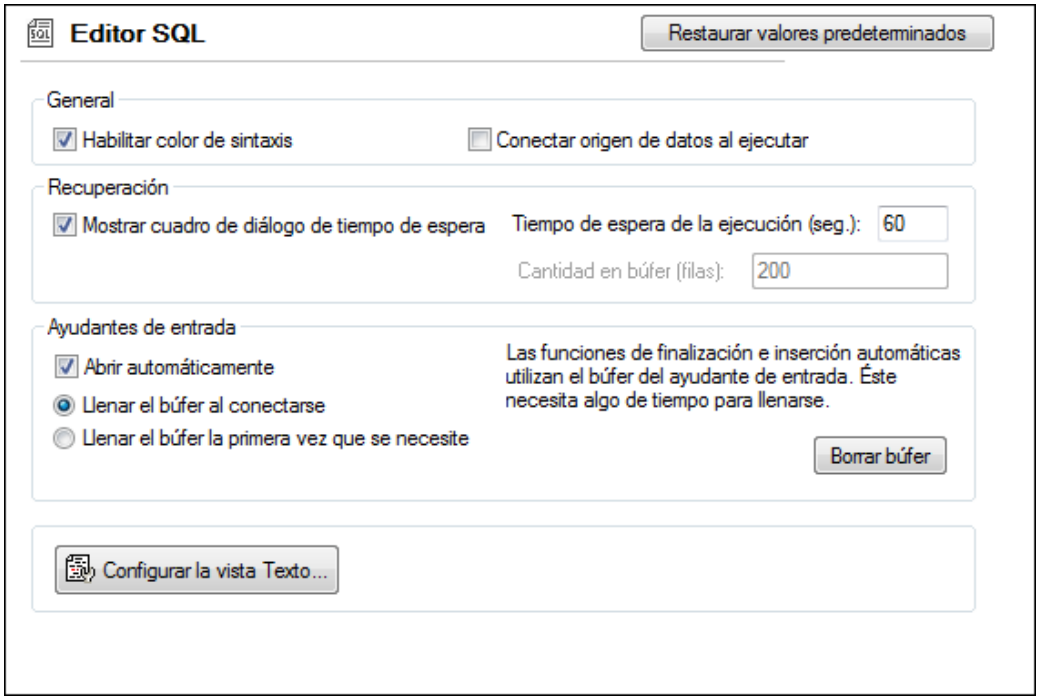
Puede seleccionar con qué codificación se abren los archivos SQL que no tengan especificada ninguna codificación o cuya codificación no se pueda detectar.

Nota: los archivos SQL que no tienen especificación de codificación se guardan en UTF-8.

7.2.7.6.2 Configuración general del editor SQL

Puede cambiar la configuración general del Editor SQL:

1. Haga clic en el panel *Consulta de la BD*.
2. En la parte superior del panel de consultas haga clic en el botón **Opciones** y después en **Editor SQL**.



General

El **color de sintaxis** resalta los elementos de la sintaxis SQL con colores diferentes según su tipo.

Marque la casilla *Conectar origen de datos al ejecutar* para conectarse al origen de datos correspondiente de forma automática cada vez que se ejecute una instrucción SQL y si su origen de datos no está conectado.

Recuperación

Especifique en segundos el tiempo de espera máximo permitido para la ejecución SQL (tiempo de espera de la ejecución).

Marque la casilla *Mostrar cuadro de diálogo de tiempo de espera* para cambiar la configuración de tiempo de espera cuando se supere el plazo de ejecución permitido.

© 2018 Altova GmbH

Búfer del ayudante de entrada

Para que MapForce muestre sugerencias de finalización automática mientras el usuario teclea instrucciones SQL, debe marcar la casilla Abrir automáticamente del grupo de opciones *Ayudantes de entrada* (*véase* *Sugerencias de finalización automática SQL*).

El búfer del ayudante de entrada para la finalización automática se puede llenar cuando el usuario se conecta a un origen de datos o cuando se usa por primera vez. Recuerde que llenar el búfer puede tardar unos minutos. Haga clic en el botón **Borrar búfer** para restaurar el búfer.

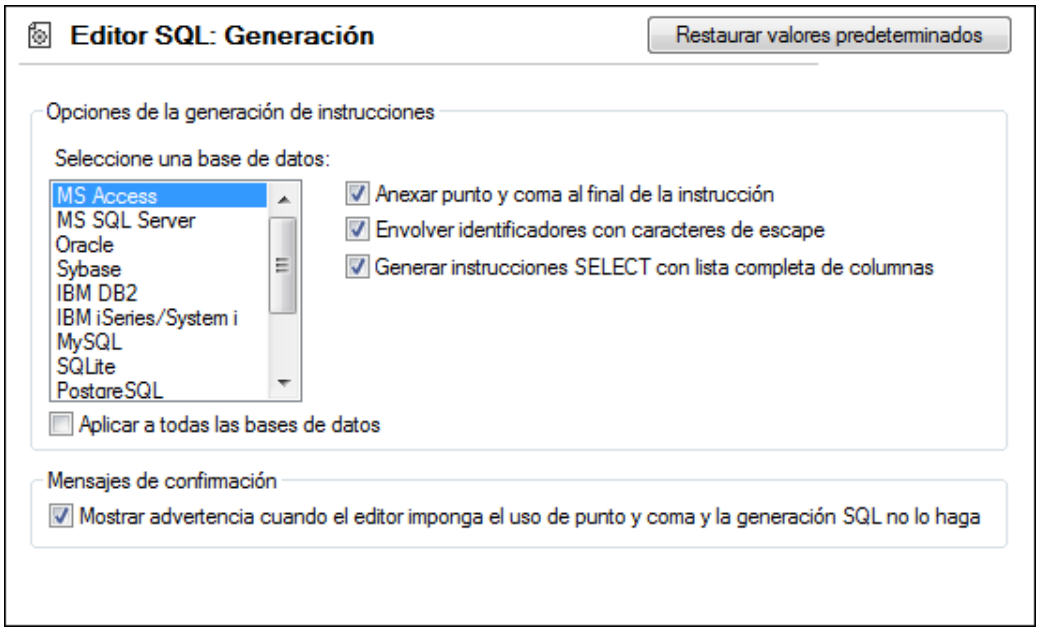
Configurar la vista Texto...

Este botón permite configurar la vista Texto: sus márgenes, pestañas y ayudantes visuales. También permite consultar las teclas de acceso rápido para navegar por la vista Texto.

7.2.7.6.3 Generación de instrucciones SQL

Puede especificar la sintaxis de generación de instrucciones SQL para los diferentes tipos de base de datos:

1. Haga clic en el panel *Consulta de la BD*.
2. En la parte superior del panel de consultas haga clic en el botón **Opciones** y después en **Editor SQL | Generación**.



Para definir las preferencias sintácticas para un tipo de base de datos determinado, seleccione el

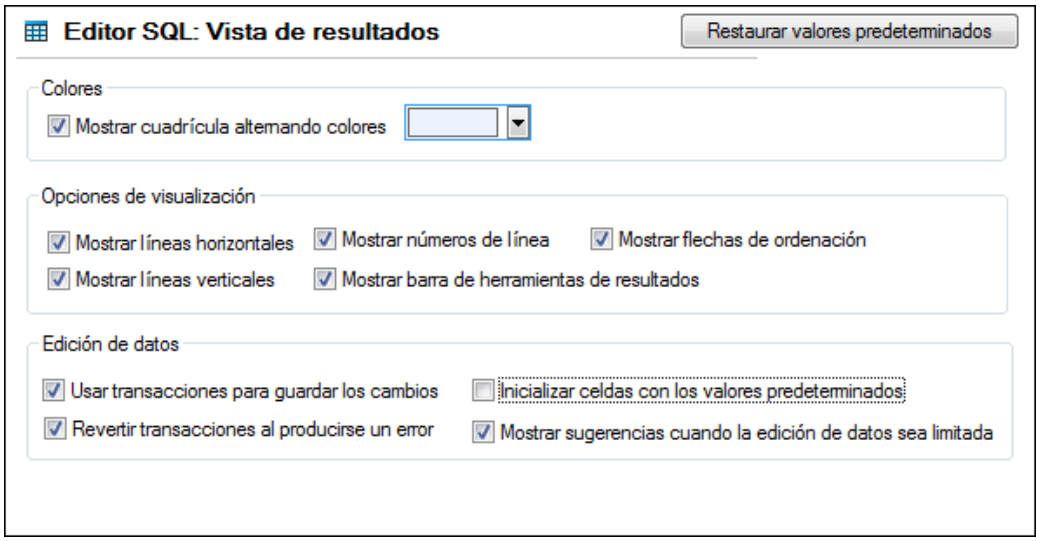
tipo de base de datos en la lista y después active/desactive las casillas correspondientes del panel derecho.

Para definir una sintaxis única para todas las bases de datos marque la casilla *Aplicar a todas las bases de datos* y después habilite/deshabilite las tres casillas del panel derecho. Recuerde que si usa una configuración común para todas las bases de datos, puede que se puedan editar datos en bases de datos Oracle e IBM DB2/iSeries a través de conexiones JDBC.

7.2.7.6.4 Vista de resultados de la consulta

Puede configurar el aspecto de la vista de resultados del panel *Consulta de la BD*:

1. Haga clic en el panel *Consulta de la BD*.
2. En la parte superior del panel de consultas haga clic en el botón **Opciones** y después en **Editor SQL | Vista de resultados**.



Marque la casilla *Mostrar cuadrícula alternando colores* para que las filas de la vista de resultados aparezcan como una cuadrícula simple o como cuadrícula filas blancas y de otro color (que se puede elegir en el cuadro combinado).

El grupo *Opciones de visualización* sirve para configurar el aspecto de las líneas horizontales y verticales de la cuadrícula y para habilitar los números de las líneas y la barra de herramientas de resultados.

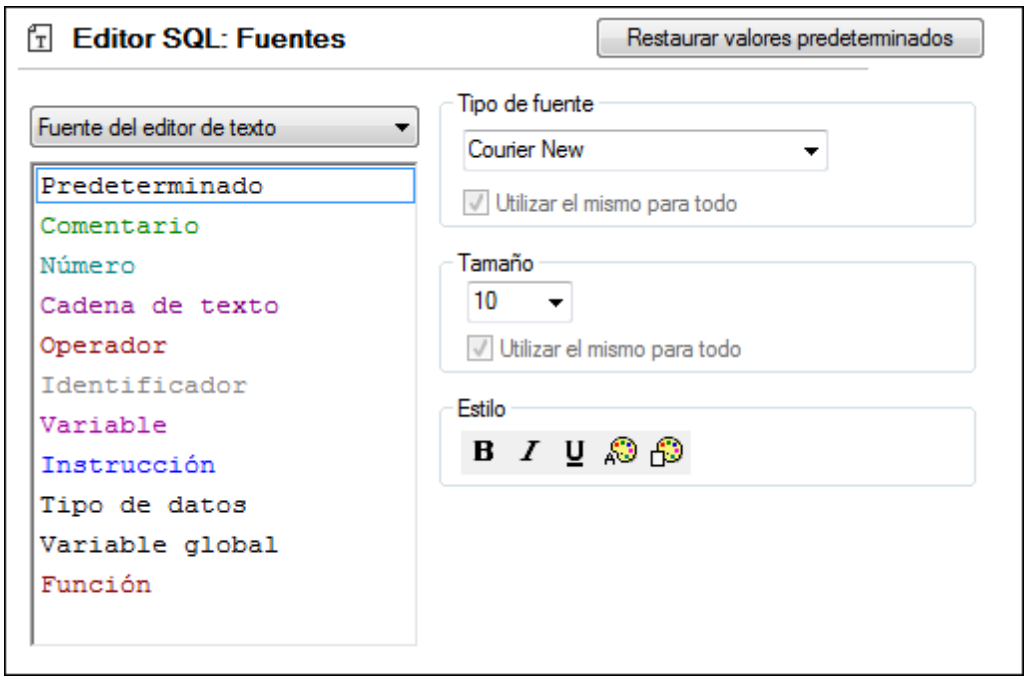
El grupo de opciones *Edición de datos* sirve para configurar las transacciones, para definir si las celdas se deben rellenar con valores predeterminados y para especificar si la aplicación debe mostrar una sugerencia cuando la edición de datos sea limitada.

7.2.7.6.5 Fuentes del editor SQL

Puede configurar el color y las fuentes de las instrucciones SQL que aparecen en el Editor SQL:

© 2018 Altova GmbH

1. Haga clic en el panel *Consulta de la BD*.
2. En la parte superior del panel de consultas haga clic en el botón **Opciones** y después en **Editor SQL | Fuentes**.



El panel de la izquierda enumera los elementos de las instrucciones SQL. A la derecha se puede elegir el tipo de fuente, el estilo y el tamaño del texto que aparece en el Editor SQL. Recuerde que se usará la misma fuente y el mismo tamaño para todos los tipos de texto.

Solamente el estilo se puede cambiar para cada tipo de texto. Se trata de la opción que habilita la función de color de sintaxis. Haga clic en el botón *Restaurar valores predeterminados* para volver a la configuración original.

7.2.8 Procedimientos almacenados

Los procedimientos almacenados son programas que están hospedados y se ejecutan en un servidor de BD. Es lo que se conoce como aplicaciones cliente. A menudo se escriben en algún dialecto extendido de SQL. Algunas bases de datos también admiten implementaciones en Java, .NET CLR y otros lenguajes de programación.

Entre los usos más frecuentes de los procedimientos almacenados se encuentran las consultas de la BD y la devolución de datos al cliente de llamada o la modificación de bases de datos tras la validación de los parámetros de entrada. Los procedimientos almacenados también pueden realizar otras acciones fuera de la base de datos, p. ej. enviar correos electrónicos.

En MapForce los procedimientos almacenados:

- pueden estar presentes (y ser llamados) tanto en componentes de BD de origen como de destino.
- pueden recibir asignaciones de datos desde parámetros de entrada o pueden asignar

datos a parámetros de salida.
- se pueden insertar como llamada de tipo función. Esto sirve para suministrar datos de entrada, ejecutar el procedimiento almacenado y leer/asignar los datos de salida a otros componentes.
- se presentan con su nombre único y un botón interactivo dentro del componente de BD en cuanto se inserta la BD en el área de asignación.
- no se pueden editar desde dentro de MapForce
- solamente se pueden usar con el motor de ejecución integrado (los procedimientos almacenados no son compatibles con la generación de código en C++, C# o Java).

Nota: en esta sección usaremos Microsoft SQL Server 2008 y la base de datos "AdventureWorks" para explicar cómo se implementan los procedimientos almacenados en MapForce. La base de datos "AdventureWorks" se puede descargar del sitio web de CodePlex (http://sqlserversamples.codeplex.com).

Compatibilidad relacionada con procedimientos almacenados:

- Tipos de parámetros de entrada/salida: por lo general no son compatibles los tipos definidos por el usuario, tipos cursor, tipos variant y muchos otros tipos de datos de BD menos comunes (como tipos matriz, geometry, CLR, etc.).

- No es compatible la sobrecarga de procedimientos y funciones (varias definiciones de rutinas con el mismo nombre y parámetros diferentes).

- Algunas bases de datos admiten el uso de valores predeterminados en los parámetros de entrada, pero MapForce no lo admite por ahora. No puede omitir parámetros de entrada en la asignación para usar valores predeterminados.

- Los procedimientos almacenados que devuelven varios conjuntos de registros son compatibles o no dependiendo de la combinación de controlador y API de base de datos (ODBC/ADO/JDBC). Solamente son compatibles los procedimientos que devuelven el mismo número de conjuntos de registro con una estructura de columnas fijas.

- En las bases de datos Microsoft Access los procedimientos almacenados tienen funciones muy limitadas y no son compatibles con MapForce.

Controladores

Recomendamos utilizar siempre que se pueda la versión más reciente del controlador nativo de BD que mantenga el proveedor de la BD. No recomendamos utilizar controladores puente como ODBC to ADO Bridge ni ODBC to JDBC Bridge.

Firebird 2.5

- Compatibles con MapForce: procedimientos almacenados y funciones con valores de tabla.

© 2018 Altova GmbH

IBM DB2 8, 9, 10, IBM DB2 para i 6.1, 7.1

- Compatibles con MapForce: procedimientos almacenados, funciones escalares y funciones con valores de tabla.

Informix 11.7

- Compatibles con MapForce: procedimientos almacenados y funciones con valores de tabla.

Microsoft SQL Server 2005, 2008, 2012, 2014

- Compatibles con MapForce: procedimientos almacenados, funciones escalares y funciones con valores de tabla.
- Recomendamos usar el controlador **SQL Server Native Client** más reciente en lugar de **Microsoft OLE DB Provider for SQL Server**.
- La API de ADO tiene algunas limitaciones de compatibilidad con algunos tipos de datos introducidos con SQL Server 2008 (`datetime2`, `datetimeoffset`). Si detecta problemas de truncamiento de datos con estos tipos temporales cuando use ADO con SQL Server Native Client, puede establecer el argumento de cadena de conexión `DataTypeCompatibility=80` o usar ODBC.

MySQL 5.5, 5.6

- Compatibles con MapForce: procedimientos almacenados y funciones escalares.
- MySQL incluye total compatibilidad para procedimientos almacenados y funciones a partir de la versión 5.5. Si usa una versión anterior, la funcionalidad será limitada en MapForce.

Oracle 9i, 10g, 11g, 12c

- Compatibles con MapForce: procedimientos almacenados, funciones escalares y funciones con valores de tabla.
- Se recomienda usar un controlador Oracle nativo en lugar del proveedor **Microsoft OLE DB Provider for Oracle**.
- Oracle tiene una forma especial de devolver conjuntos de resultados al cliente que consiste en usar parámetros de salida de tipo REF CURSOR. Esto es compatible con MapForce en el caso de los procedimientos almacenados, pero no de las funciones. Por tanto, los nombres y el número de los conjuntos de registros siempre son fijos para procedimientos almacenados de Oracle.

PostgreSQL 8, 9

- Compatibles con MapForce: funciones escalares, funciones con valores de fila y funciones con valores de tabla.
- En PostgreSQL los parámetros de salida definidos en una función describen las columnas del conjunto de resultados. Esta información se utiliza automáticamente en MapForce (no es necesaria la detección por ejecución ni la entrada manual de conjuntos

de registros). Los parámetros de tipo refcursor no son compatibles.

Progress OpenEdge 11.6

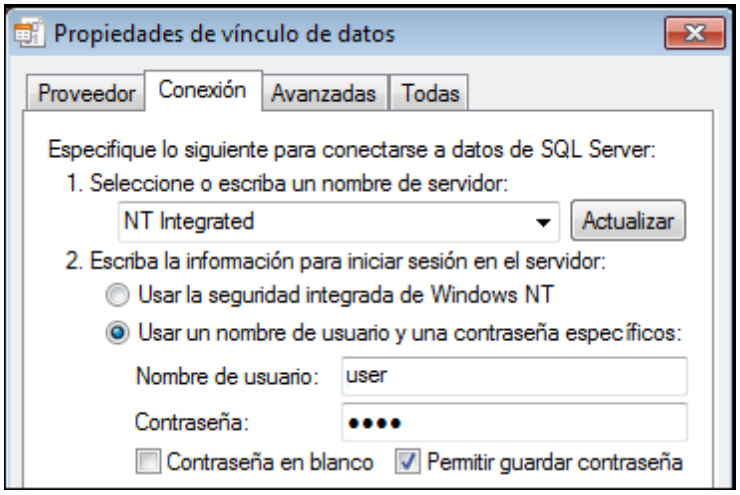
- Compatibles con MapForce: procedimientos almacenados.

7.2.8.1 Insertar procedimientos almacenados en componentes de BD

Los procedimientos almacenados se pueden incorporar a un componente de BD cuando el componente se inserta en el área de asignación.

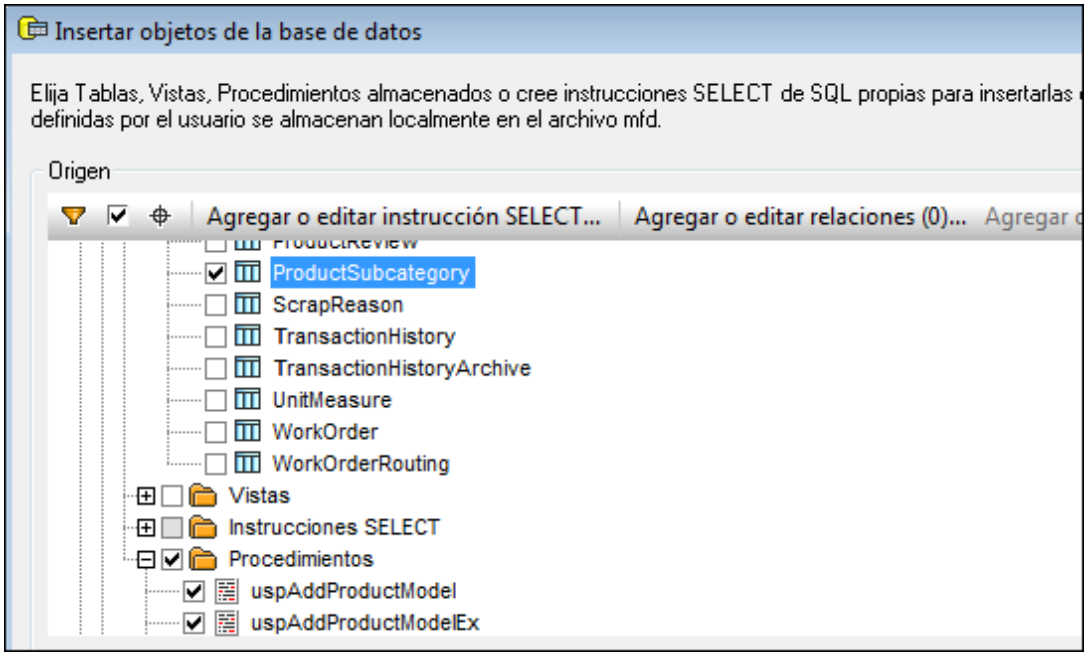
Para insertar un componente de BD que contiene procedimientos almacenados:

1. Haga clic en el icono **Insertar base de datos** o seleccione la opción de menú **Archivo | Insertar base de datos**.
2. Use el Asistente para la conexión para conectarse a la base de datos.
3. Aparece el cuadro de diálogo "Propiedades de vínculo de datos". Rellene los campos de la pestaña *Conexión* y haga clic en **Aceptar**.

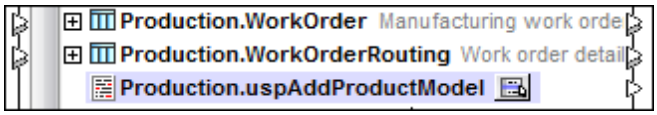


4. Haga clic en el botón de expansión para ver las tablas de la BD y seleccione las que desea insertar.

 © 2018 Altova GmbH

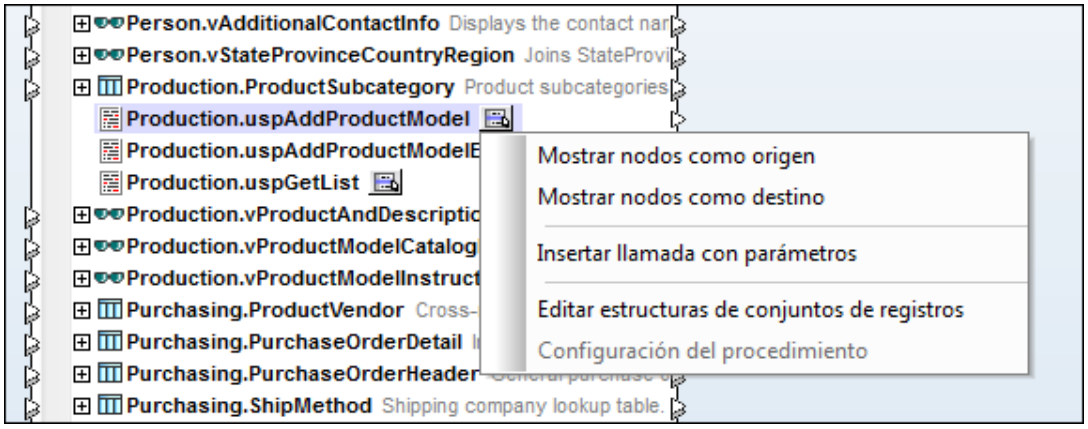


5. Expanda la carpeta *Procedimientos* para seleccionar los procedimientos almacenados que desea insertar y después haga clic en **Aceptar**.



El componente de base de datos se inserta y muestra las tablas seleccionadas en el paso anterior seguidas de los procedimientos almacenados elegidos.

- Las tablas, las vistas y los procedimientos almacenados aparecen por orden alfabético en el componente de base de datos.
- Cada procedimiento almacenado aparece como un elemento, con un nombre y un botón interactivo. Con este botón puede elegir si el procedimiento almacenado se utiliza como origen o como destino. También puede configurar el procedimiento con este botón.
- Llegados a este punto MapForce no sabe si los parámetros de los procedimientos almacenados se deben usar como parámetros de origen o destino. Es necesario hacer clic en el botón del procedimiento almacenado y seleccionar una opción.



7.2.8.2 Casos de uso

A continuación enumeramos los casos de uso más frecuentes relacionados con procedimientos almacenados.

Si tiene pensado...	**Consulte este apartado de la documentación**
... llamar a un procedimiento almacenado para **recuperar** datos de una base de datos y asignarlos a otro componente. P. ej. si desea usar un procedimiento almacenado como origen de datos para escribir los datos resultantes en otro archivo (un archivo XML, TXT, EDI, etc.)	Procedimientos almacenados en componentes de origen
... llamar a un procedimiento almacenado para **modificar** la base de datos o realizar otra acción.	Procedimientos almacenados en componentes de destino
... usar procedimientos almacenados para **generar** valores / claves para una instrucción INSERT en la misma base de datos.	Usar procedimientos almacenados para generar claves principales

7.2.8.3 Procedimientos almacenados y relaciones locales

Puede usar relaciones locales (*véase Definir relaciones locales*) para definir el orden jerárquico en que se llama a los procedimientos almacenados o se realizan ciertas acciones (inserción, actualización, etc.) en las tablas de la BD. Estas relaciones locales se pueden usar tanto en componentes de origen como en componentes de destino.

Una relación local siempre tiene un objeto primario (que contiene una clave principal/única) y un objeto secundario (que contiene una clave foránea).

En una relación se pueden usar estos objetos primarios y sus campos:

- Tabla o vista (columna) de base de datos

 © 2018 Altova GmbH

- Procedimiento almacenado (parámetro de salida o valor devuelto)
- Conjunto de registros de un procedimiento almacenado (columna), pero solamente para componentes de origen y de llamada a procedimientos
- Instrucción SELECT definida por el usuario (columna)

Los objetos secundarios y campos de objetos secundarios que se pueden usar son:

- Tabla o vista (columna, produce una condición WHERE) de base de datos
- Procedimiento almacenado (parámetro de entrada)
- Instrucción SELECT definida por el usuario (parámetro de entrada)

En los procedimientos almacenados esto facilita la lectura de datos de objetos relacionados (como la lectura de Ids de una tabla de base de datos y la llamada a un procedimiento almacenado con estos IDs para recuperar información relacionada). También se puede llamar a un procedimiento almacenado con datos recuperados de otro procedimiento almacenado.

En los componentes de destino las relaciones locales permiten definir con claridad el orden en que se llama a varios procedimientos almacenados (p. ej. uno crea un valor de Id. y otro inserta información relacionada en otra tabla). También se pueden mezclar procedimientos almacenados y tablas en relaciones locales (p. ej. realizar la inserción directamente en la tabla relacionada en lugar de llamar a otro procedimiento).

7.2.8.4 Procedimientos almacenados como origen de datos

El resultado de un procedimiento almacenado puede ser cero o más parámetros de salida/ devolución y cero o más conjuntos de registros de las instrucciones SELECT incrustadas en el procedimiento almacenado. Un conjunto de registros o conjunto de resultados es el resultado de dicha instrucción SELECT y es similar a una tabla o vista. Los parámetros de salida y conjuntos de registros se pueden asignar al componente de destino.

Las estructuras de columna de estos conjuntos de registros no se pueden leer desde el catálogo de la BD directamente. Antes deben ser detectados ejecutando el procedimiento almacenado en tiempo de diseño o definiéndolos manualmente (consultar el apartado Definir conjuntos de registros para más información).

MapForce se ocupa del procedimiento almacenado de varias formas diferentes, dependiendo de si tiene parámetros de entrada o no:

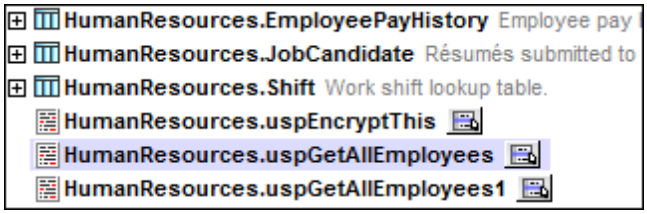
El procedimiento almacenado no tiene parámetros de entrada.	Procedimientos almacenados sin parámetros de entrada
Quiero suministrar los valores para los parámetros de entrada del procedimiento asignando datos de un archivo XML, de texto, etc. o asignándoles parámetros de entrada o constantes.	Llamada con parámetros: entrada/salida
Quiero suministrar los valores para los parámetros de entrada del procedimiento desde una tabla o vista de la misma BD o desde el resultado de otro procedimiento.	Componentes de origen y relaciones locales

7.2.8.4.1 Procedimientos almacenados sin parámetros de entrada

Utilice este método si desea usar el procedimiento almacenado en un componente de origen **sin** parámetros de **entrada**.

Por ejemplo, podría tratarse de un procedimiento almacenado que es una consulta tipo SELECT sin parámetros de entrada donde desea asignar el resultado de la instrucción SELECT a un componente de destino.

Tomemos como ejemplo el procedimiento HumanResources.uspGetAllEmployees de la base de datos **AdventureWorks**.



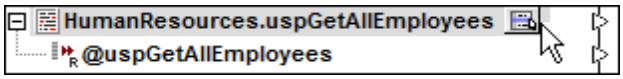
Procedimiento almacenado:

```
PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment;
```

MapForce no puede leer las columnas del conjunto de registros desde el catálogo de la BD directamente. Antes deben ser detectadas ejecutando el procedimiento almacenado una vez o definiéndolas a mano.

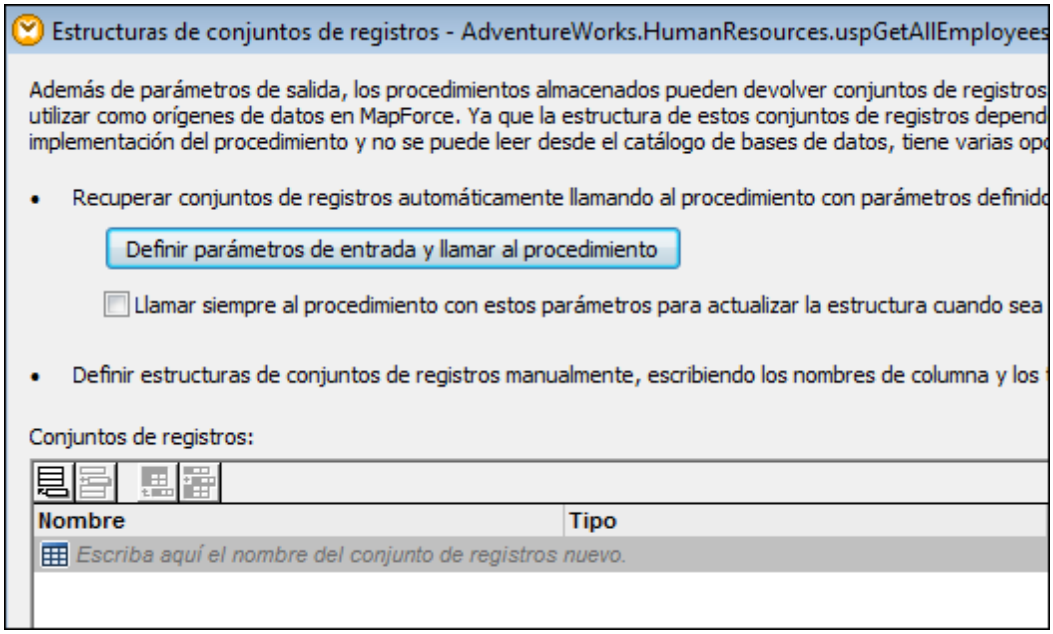
Para definir el conjunto de registros de salida de un componente de origen:
Tras insertar el componente de base de datos **AdventureWorks** y seleccionar las tablas HumanResources y sus procedimientos almacenados:

1. Haga clic en el botón del procedimiento almacenado y seleccione la opción **Mostrar nodos como origen** del menú contextual.

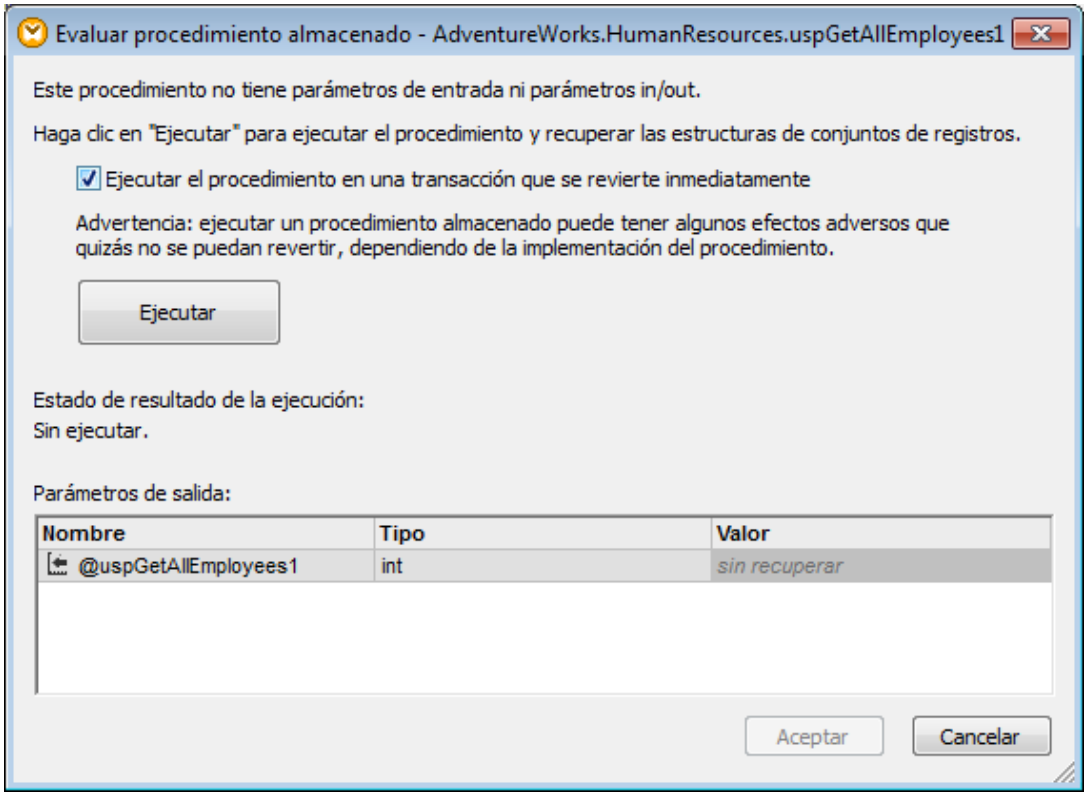
   

   El nodo del valor devuelto se inserta bajo el nombre del procedimiento almacenado. Puesto que queremos leer el conjunto de registros y no el valor devuelto, haga clic otra vez en el botón del procedimiento almacenado y seleccione **Editar estructuras de conjuntos de registros**.
2. Se abre el cuadro de diálogo que aparece en la imagen siguiente. Haga clic en el botón **Definir parámetros de entrada y llamar al procedimiento**.

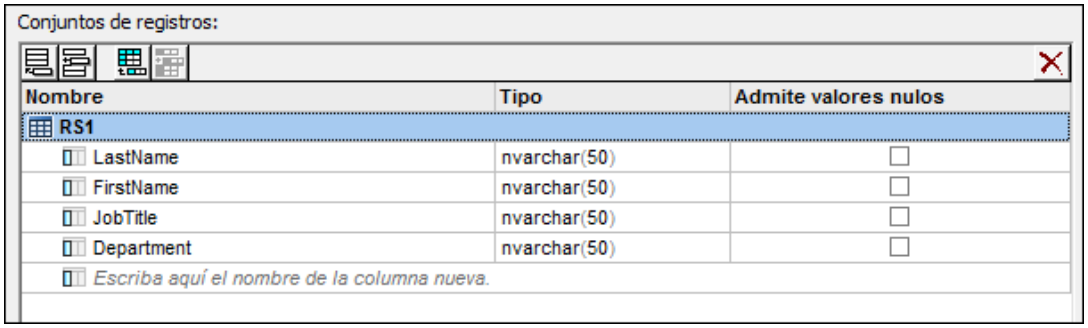
© 2018 Altova GmbH



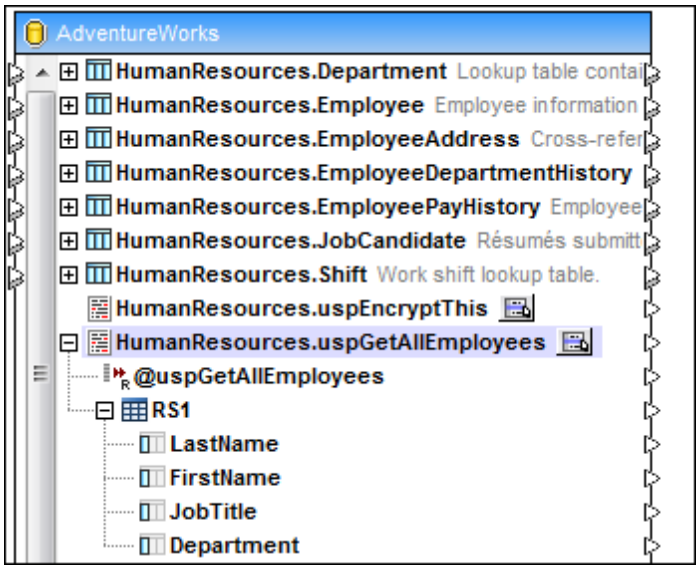
Aparece el cuadro de diálogo "Evaluar procedimiento almacenado".



3. Haga clic en el botón **Ejecutar** y después en **Aceptar**.
   Los campos del conjunto de registros aparecen en el panel *Conjuntos de registros* del cuadro de diálogo.

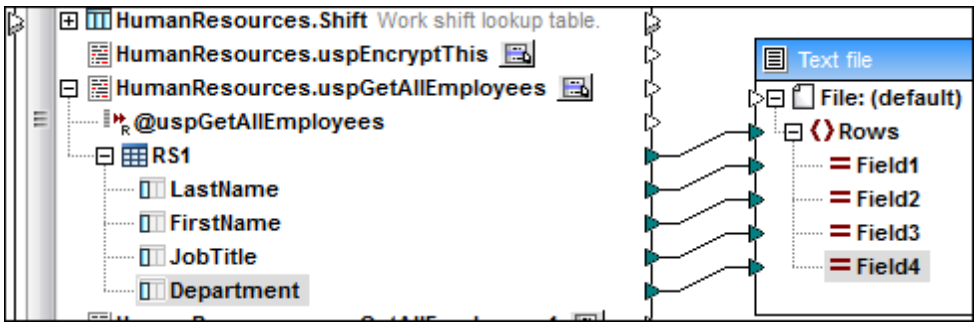


4. Haga clic otra vez en el botón **Aceptar** para terminar de definir el conjunto de registros. Las columnas (LastName, etc.) aparecen en forma de nodos bajo el nodo de conjunto de registros RS1 (*Record Set 1*). (Si no las ve, haga clic en el icono + para expandir el conjunto de registros).



Para terminar la asignación:

1. Inserte un componente de archivo de texto y asigne las columnas de origen al archivo de texto.



2. Abra el panel **Resultados** para ver el resultado de la asignación.

 © 2018 Altova GmbH

```
1    Gilbert,Guy,Production Technician - WC60,Production
2    Brown,Kevin,Marketing Assistant,Marketing
3    Tamburello,Roberto,Engineering Manager,Engineering
4    Walters,Rob,Senior Tool Designer,Tool Design
5    D'Hers,Thierry,Tool Designer,Tool Design
6    Bradley,David,Marketing Manager,Marketing
7    Dobney,JoLynn,Production Supervisor - WC60,Production
```

Nota: si cree que al ejecutar el procedimiento se pueden producir efectos no deseados durante el tiempo de diseño, también puede definir los conjuntos de registros manualmente en el cuadro de diálogo "Estructuras de conjuntos de registros" añadiendo conjuntos de registros y sus columnas. Para ello haga clic en el botón **Agregar conjunto de registros** o **Agregar columna**.

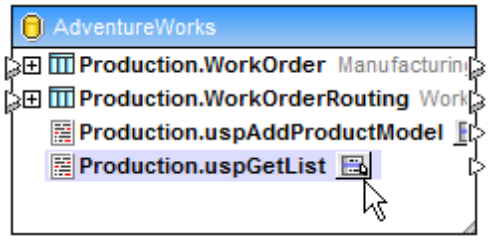
7.2.8.4.2 Llamada con parámetros: entrada/salida

Los procedimientos almacenados también se pueden usar como llamadas de tipo función. De esta manera puede:

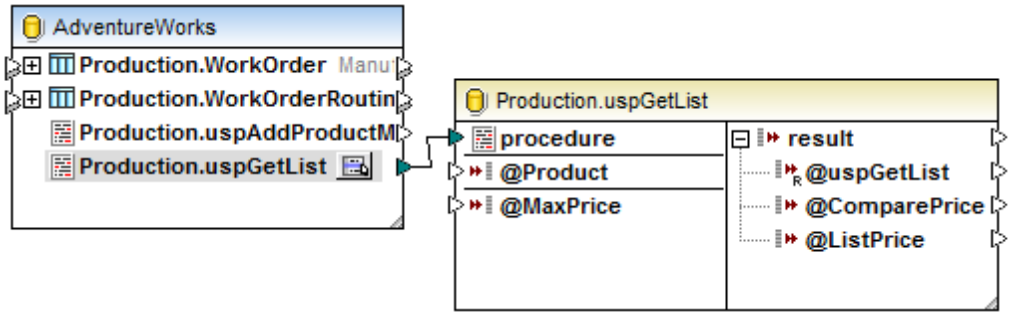
- suministrar datos de entrada al procedimiento
- ejecutar el procedimiento
- asignar los datos de salida del procedimiento a otros componentes

Para usar un procedimiento almacenado como llamada de tipo función:
Tras insertar el componente de base de datos **AdventureWorks**, seleccionar las tablas Production e incluir los procedimientos almacenados:

1. Haga clic en el botón interactivo del procedimiento almacenado Production.uspGetList.



2. Seleccione la opción **Insertar llamada con parámetros** del menú contextual.



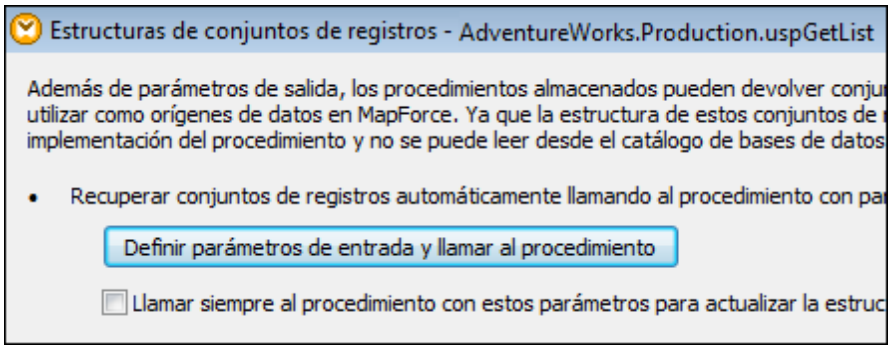
Esto inserta un componente de procedimiento almacenado en la asignación. El componente tiene el mismo aspecto y funciona igual que un componente de servicio web o un componente definido por el usuario. El nombre del procedimiento se conecta

automáticamente al elemento procedure del componente.

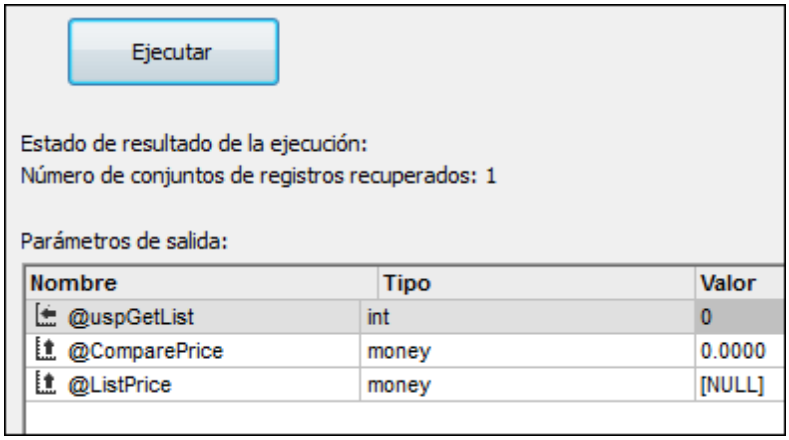
Los parámetros de entrada del procedimiento aparecen a la izquierda y los de salida a la derecha. En concreto, este procedimiento almacenado devuelve parámetros de salida y un conjunto de registros. Sin embargo, debemos definir su estructura antes de poder verla y utilizarla en MapForce.

Para definir la estructura del conjunto de registros:

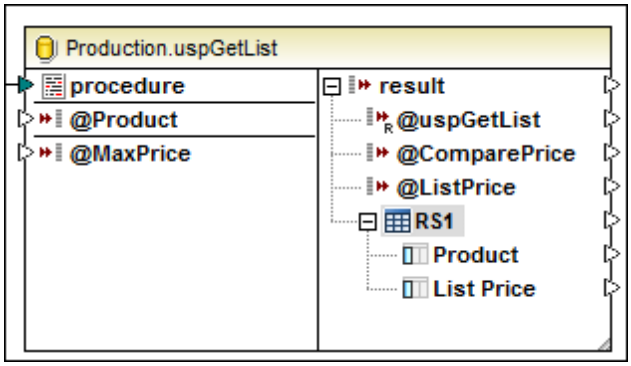
1. Haga clic en el botón interactivo del procedimiento almacenado Production.uspGetList.
2. Haga clic en el botón **Definir parámetros de entrada y llamar al procedimiento**. En el nuevo cuadro de diálogo haga clic en **Ejecutar**.



Los valores de los parámetros de salida devueltos se escriben en la tabla inferior y podemos ver que se recuperó un conjunto de registros.



3. Haga clic **Aceptar** para confirmar y otra vez en **Aceptar** para cerrar el cuadro de diálogo "Estructuras de conjuntos de registros".

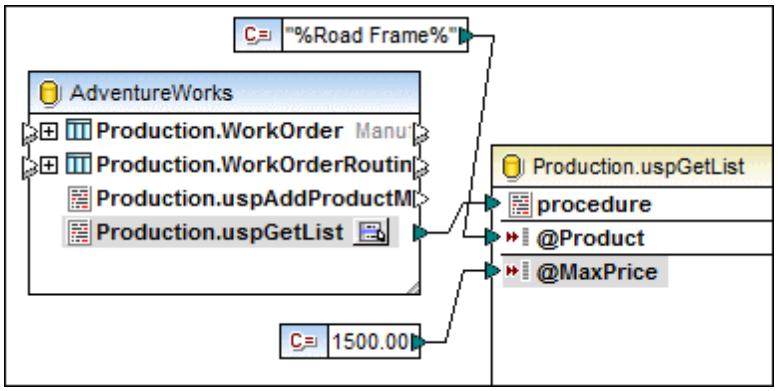


 © 2018 Altova GmbH

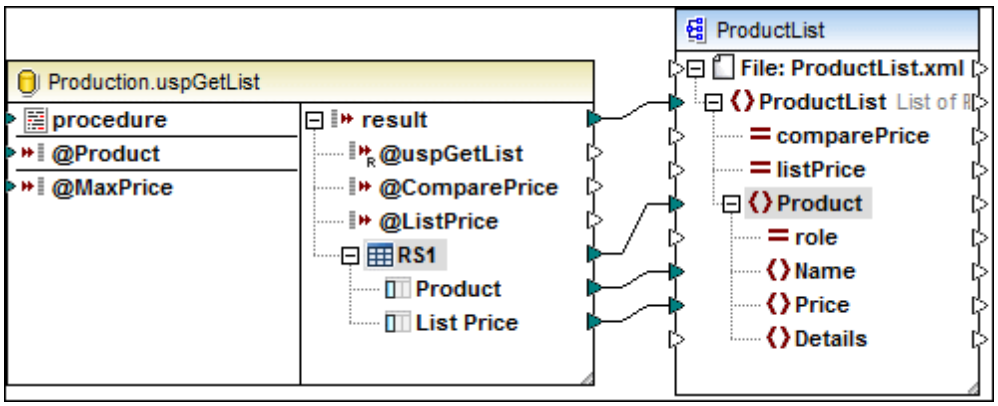
El conjunto de registros RS1 se añadió a la sección derecha del componente de procedimiento.

Para usar el componente de parámetro de llamada:

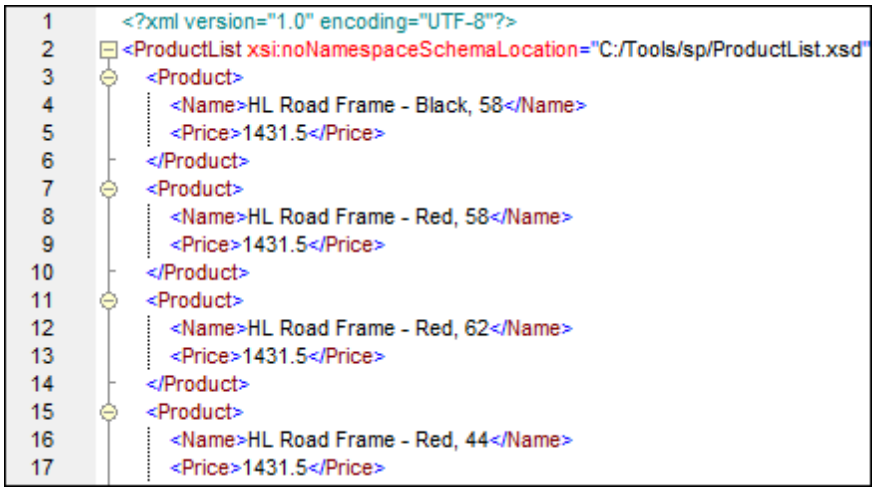
1. Defina qué componentes desea usar para dar un valor a los parámetros de entrada (p. ej. dos constantes) y conéctelos a los parámetros de entrada.



2. Defina e inserte el componente de destino que recibirá el resultado del procedimiento almacenado (p. ej. un documento XML).



3. Abra el panel **Resultados** para ver el resultado de la asignación.



7.2.8.4.3 Componentes de origen y relaciones locales

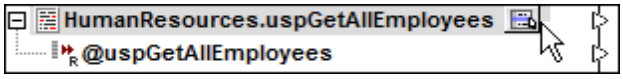
Utilice este método si desea combinar los datos dados por un conjunto de registros de un procedimiento almacenado con los datos de otra tabla con la que no hay ninguna relación directa.

```
PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment;
```

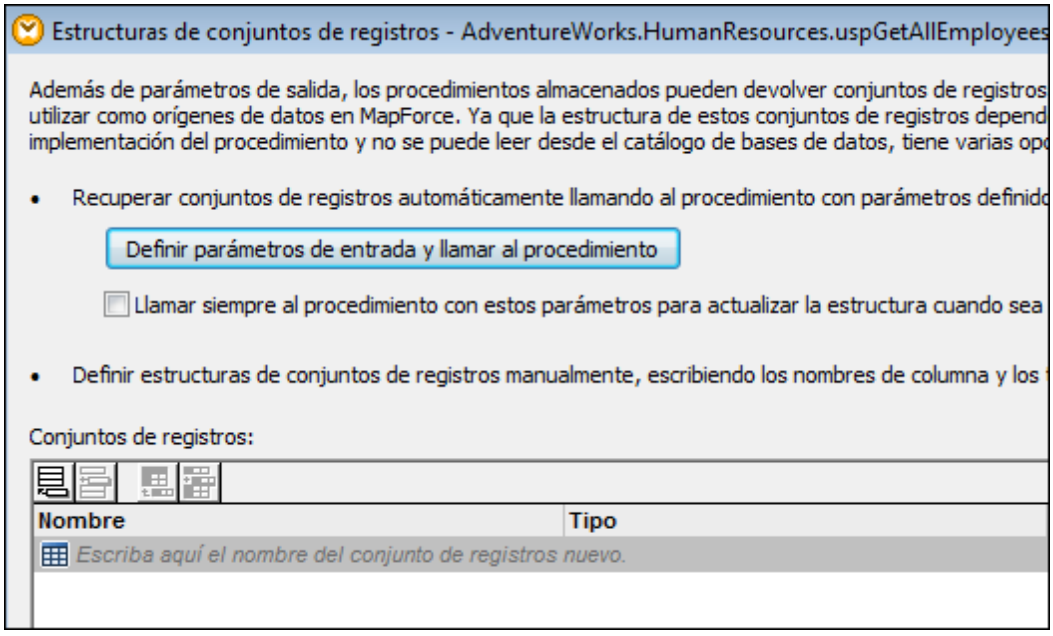
MapForce no puede leer las columnas del conjunto de registros desde el catálogo de la BD directamente. Antes deben ser detectadas ejecutando el procedimiento almacenado una vez o definiéndolas a mano.

Para definir el conjunto de registros de salida de un componente de origen:
Tras insertar el componente de base de datos **AdventureWorks**, seleccionar las tablas HumanResources e incluir los procedimientos almacenados:

1. Haga clic en el botón interactivo del procedimiento almacenado y seleccione la opción **Mostrar nodos como origen**.

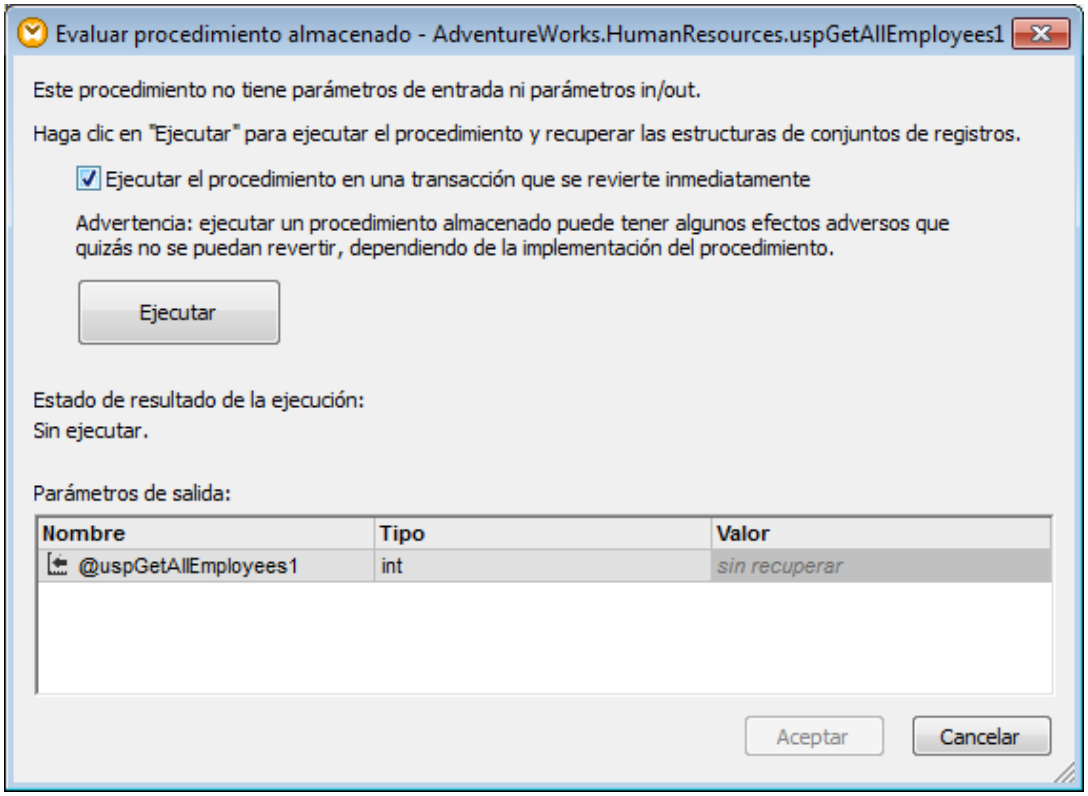


   El nodo del valor devuelto se inserta bajo el nombre del procedimiento almacenado. Puesto que queremos leer el conjunto de registros y no el valor devuelto, haga clic otra vez en el botón del procedimiento y seleccione la opción **Editar estructuras de conjuntos de registros**.
2. Haga clic en el botón **Definir parámetros de entrada y llamar al procedimiento**.

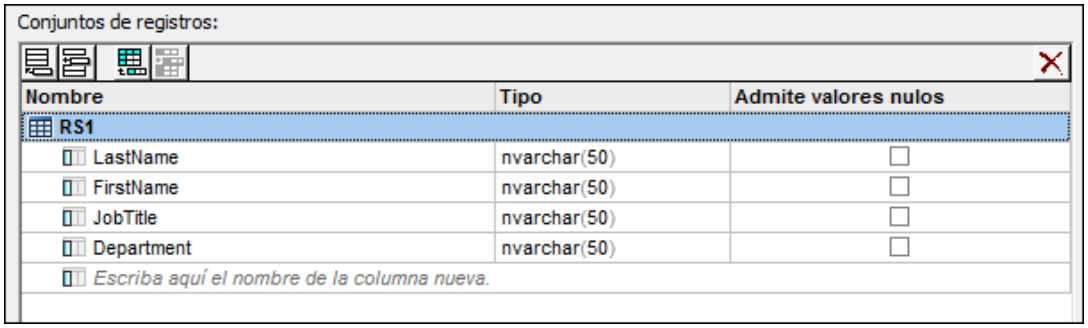


 © 2018 Altova GmbH

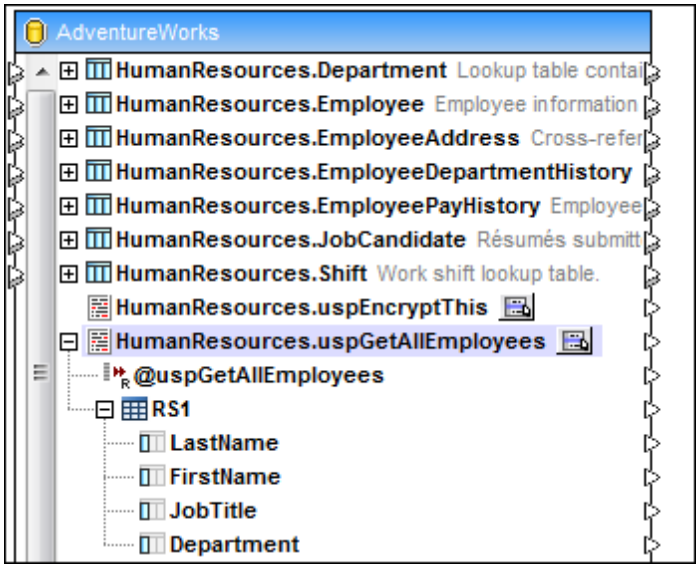
Se abre el cuadro de diálogo "Evaluar procedimiento almacenado".



3. Haga clic en el botón **Ejecutar** y después en **Aceptar**.
   Los campos del conjunto de registros aparecen ahora en el panel *Conjuntos de registros* del cuadro de diálogo.

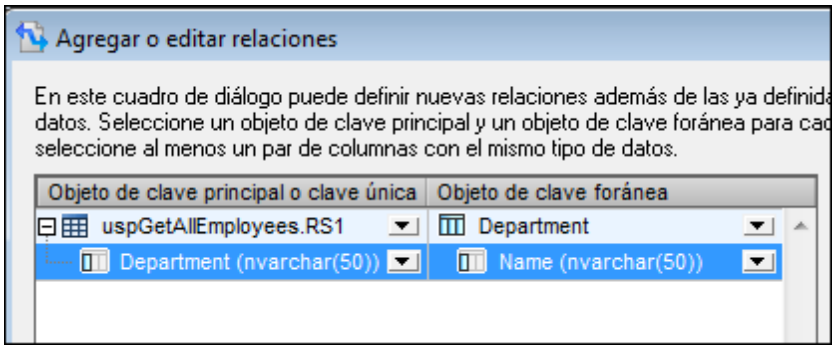


4. Haga clic otra vez en el botón **Aceptar** para terminar de definir el conjunto de registros.
   Las columnas del conjunto de registros (LastName, etc.) aparecen como nodos bajo el conjunto de registros RS1 (*Record Set 1*). (Si no las ve, haga clic en el icono + para expandir el conjunto de registros).



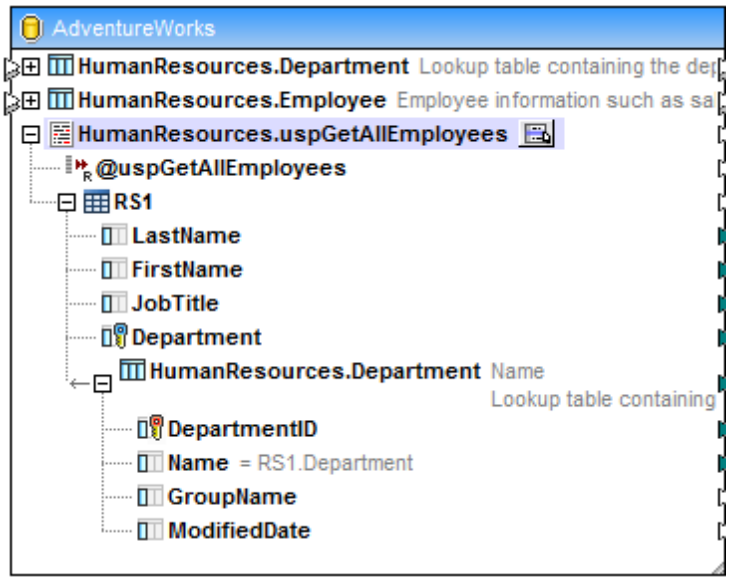
Para definir una relación local con otra tabla:

1. Haga clic con el botón derecho en el título del componente y seleccione **Agregar, quitar o editar objetos de la base de datos**.
2. Haga clic en el botón **Agregar o editar relaciones** y después en el botón **Agregar relación**.
3. Defina el procedimiento almacenado uspGetAllEmployees.RS1 como objeto de clave principal/única y el parámetro @Department como columna.
4. Defina el campo Department como objeto de clave foránea y Name como columna.



5. Haga clic en el botón **Aceptar** para confirmar.

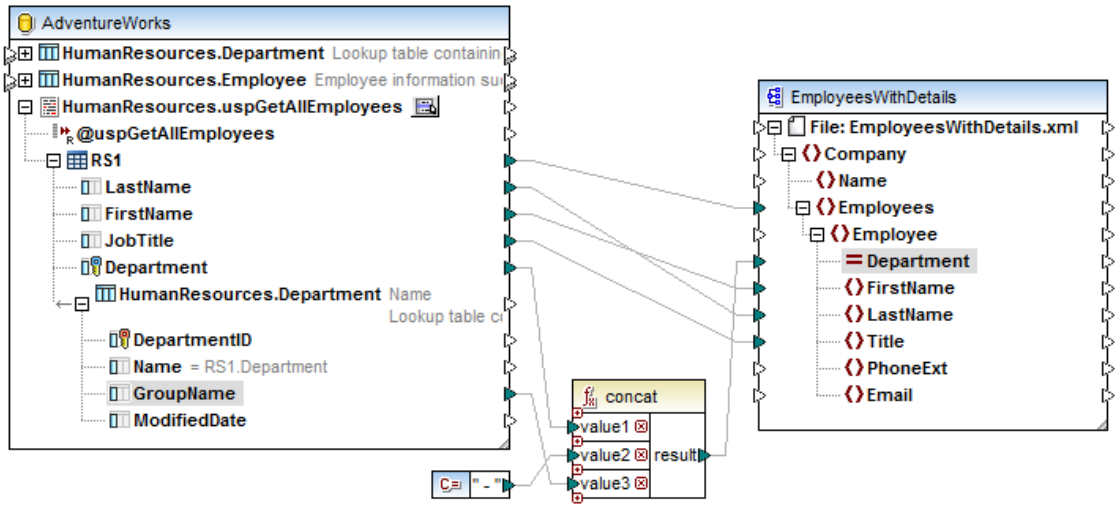
 © 2018 Altova GmbH



La tabla Department aparece ahora como elemento secundario del procedimiento almacenado.

Para terminar la asignación:

1. Inserte el esquema de destino al que desea asignar los datos de la BD de origen y cree las conexiones que aparecen en la imagen siguiente.



2. Abra el panel **Resultados** para ver el resultado de la asignación.

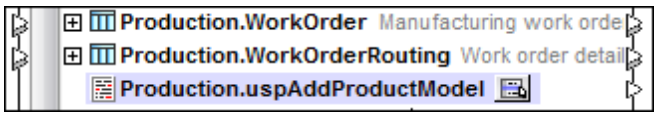


7.2.8.5 Procedimientos almacenados en componentes de destino

Utilice este método si el procedimiento almacenado realiza cambios en la BD (inserciones, actualizaciones, eliminaciones, etc.) y no le interesa ningún resultado del procedimiento almacenado.

Para usar procedimientos almacenados en un componente de destino:
Esta opción añade los nodos secundarios de los parámetros de **entrada** (y los parámetros in/out) bajo el elemento de procedimiento almacenado en el componente de BD de destino.

Por ejemplo, imagine que quiere agregar un modelo de producto nuevo a la BD usando el procedimiento almacenado uspAddProductModel de la BD **AdventureWorks**.



Procedimiento almacenado:

```
PROCEDURE Production.uspAddProductModel
@ModelName nvarchar(50),
@Inst xml

as
INSERT INTO [AdventureWorks].[Production].[ProductModel]
           ([Name]
           --,[CatalogDescription]
           ,[Instructions]
           ,[rowguid]
           ,[ModifiedDate])
     VALUES
```

 © 2018 Altova GmbH

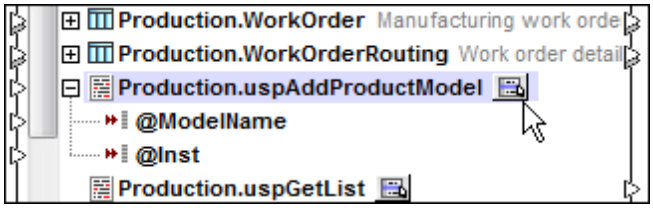
```
(@ModelName
--,<CatalogDescription, ProductDescriptionSchemaCollection,>
,@Inst
,NEWID()
,GETDATE());
```

En tiempo de ejecución MapForce ejecuta el procedimiento almacenado usando todos los parámetros de **entrada** asignados e ignora los resultados del procedimiento almacenado.

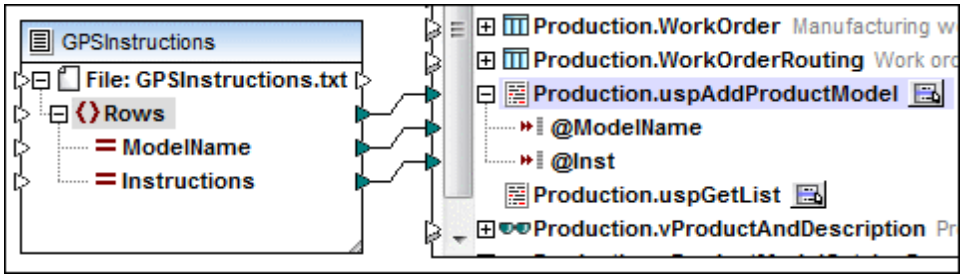
Para crear elementos parámetro de entrada en un componente de destino:

Tras insertar el componente de base de datos **AdventureWorks**, seleccionar las tablas Production e incluir los procedimientos almacenados:

1. Haga clic en el botón interactivo de procedimiento almacenado y seleccione la opción **Mostrar nodos como origen**.

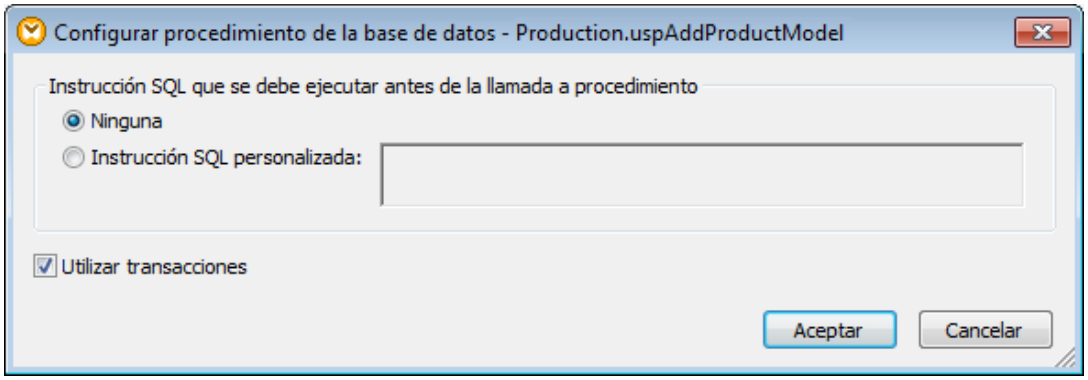


   Esto inserta los parámetros de entrada @ModelName e @Inst bajo el nombre del parámetro de entrada. En el componente de destino solamente están disponibles los iconos de entrada de los parámetros de entrada.
2. Inserte un componente de origen (p. ej. un archivo de texto, un archivo XML, etc.) y asigne a los iconos de entrada del procedimiento almacenado los elementos que deben suministrar los datos a los parámetros de entrada.



Para definir transacciones para un procedimiento almacenado:

1. Haga clic en el botón interactivo del procedimiento almacenado y seleccione la opción **Configuración del procedimiento**.
   Se abre el cuadro de diálogo "Configuración del procedimiento de la base de datos".



2. Marque la casilla *Usar transacciones* y haga clic en **Aceptar** para confirmar. Esta opción garantiza que se puedan revertir los comandos del procedimiento si se produce un error durante la ejecución.
   En este cuadro de diálogo también puede definir las instrucciones SQL que se deben ejecutar antes de llamar al procedimiento almacenado.
3. Abra el panel **Resultados** para ver qué comandos se enviarán a la BD.

```
BEGIN TRANSACTION

EXECUTE NULL = [Production].[uspAddProductModel] 'GPS Navigator',' GPS instructions';

COMMIT TRANSACTION
```

Notas:

- Las opciones **Agregar delante/detrás un duplicado de entrada** del menú contextual se deshabilitan para los **parámetros** de procedimientos almacenados porque cada parámetro es un valor atómico (y también podría admitir valores NULL).
- Sin embargo, las opciones **Agregar delante/detrás un duplicado de entrada** del menú contextual están habilitadas para los **procedimientos** almacenados. Si se crea un duplicado, se llama al procedimiento una vez por cada nodo/elemento duplicado.

7.2.8.5.1 Usar procedimientos almacenados para generar claves principales

Utilice este método si el procedimiento almacenado realiza cambios en la BD y también quiere usar el parámetro de salida del procedimiento para generar una clave principal en otra tabla.

El procedimiento uspAddProductModelEx es una variación del procedimiento uspAddProductModel de la base de datos **AdventureWorks**.

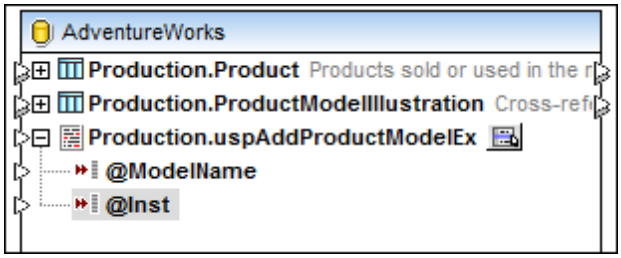
```
procedure Production.uspAddProductModelEx
@ModelName nvarchar(50),
@Inst xml,
@ProductModelID int OUTPUT
as begin
INSERT INTO [AdventureWorks].[Production].[ProductModel]
           ([Name]
           ,[Instructions]
```

 © 2018 Altova GmbH

```
            ,[rowguid]
            ,[ModifiedDate])
      VALUES
            (@ModelName
            ,@Inst
            ,NEWID()
            ,GETDATE());
   SELECT @ProductModelID = SCOPE_IDENTITY()
end;
```

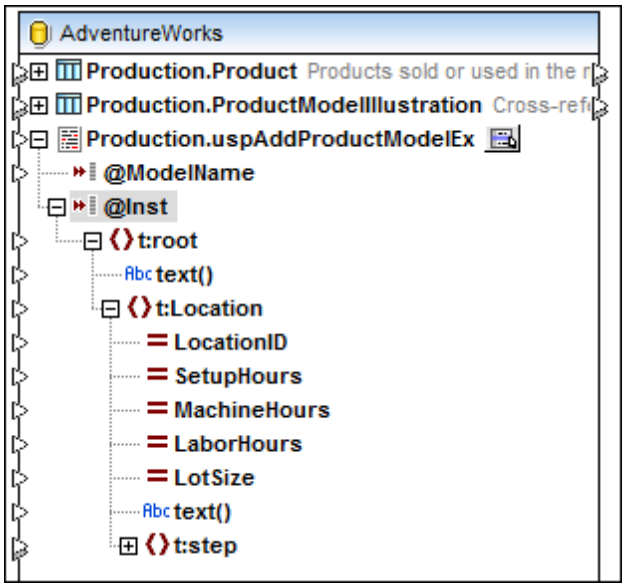
Tras insertar el componente de base de datos **AdventureWorks**, seleccionar las tablas Production e incluir los procedimientos almacenados:

1. Haga clic en el botón interactivo del procedimiento y seleccione **Mostrar nodos como destino** en el menú contextual.



   Esto inserta los parámetros @ModelName e @Inst bajo el nombre de procedimiento. En el componente solamente se ven los parámetros de entrada del procedimiento.

   El parámetro @Inst es de tipo XML y, por tanto, debemos asignarle un esquema XML que aporte los datos XML.
2. Haga clic con el botón derecho en el parámetro @Inst y seleccione **Asignar esquema XML al campo** en el menú contextual.
3. En el cuadro combinado *Baso de datos* seleccione la opción **Production.ManuInstructionsSchemaCollection** y después haga clic en **Aceptar**.

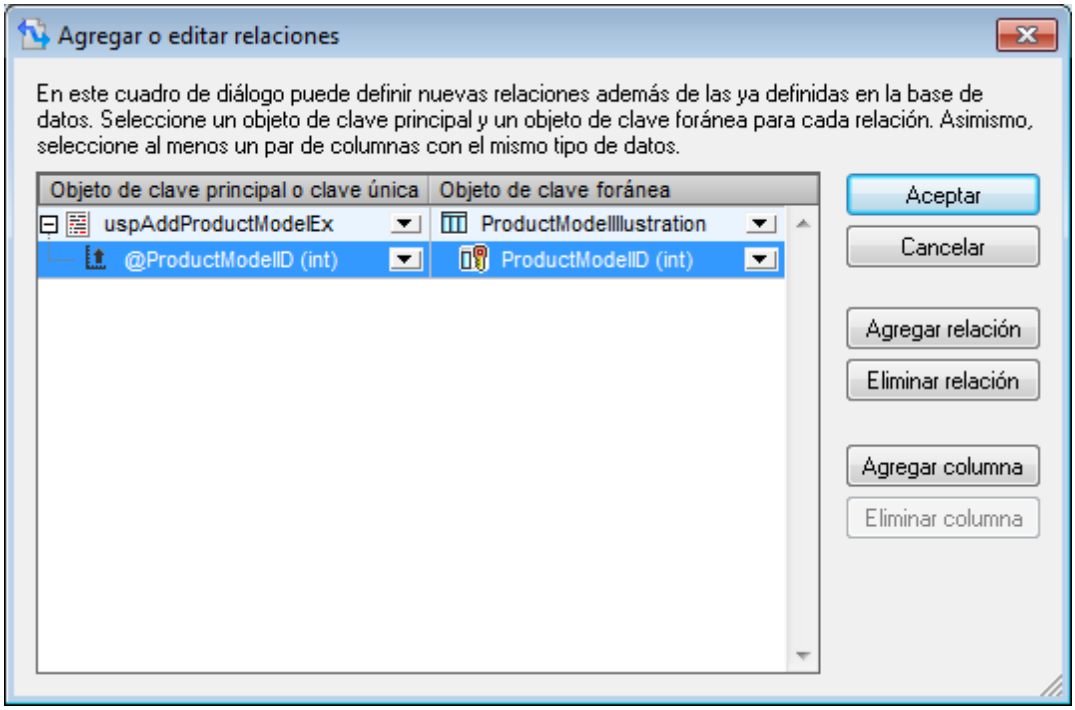


   Esto añade los elementos y atributos del esquema XML al componente. El parámetro @ModelName y todos los parámetros @Inst ya están disponibles en el componente.

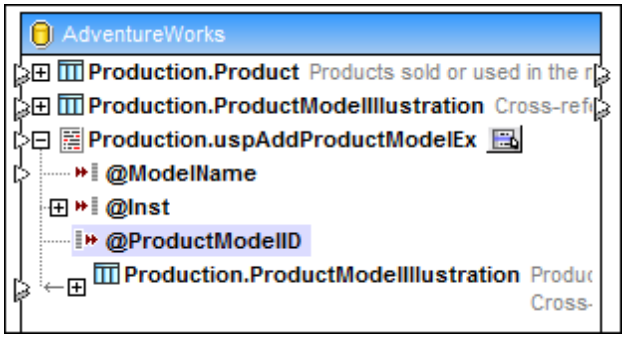
Ahora queremos definir una relación local con una tabla que no tiene conexión directa con la tabla referenciada por los parámetros del procedimiento almacenado (production.product).

Para definir una relación local con una tabla en la que desea generar una clave principal:

1. Haga clic con el botón derecho en el título del componente y seleccione **Agregar, quitar o editar objetos de la base de datos**.
2. Haga clic en el botón **Agregar o editar relaciones**. En el nuevo cuadro de diálogo haga clic en el botón **Agregar relación**.
3. Como objeto de clave principal o única defina el procedimiento almacenado uspAddProductModelEx y el parámetro @ProductModelID como columna.
4. Como objeto de clave foránea defina la tabla ProductModelIllustration y ProductModelID como columna.



5. Haga clic en el botón **Aceptar** para confirmar.

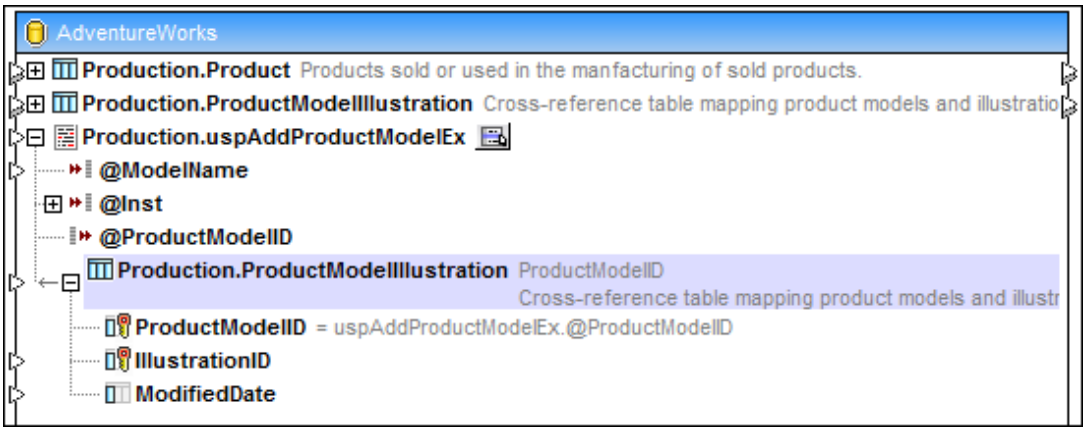


- El parámetro de salida del procedimiento almacenado (ProductModelID) se añadió al

© 2018 Altova GmbH

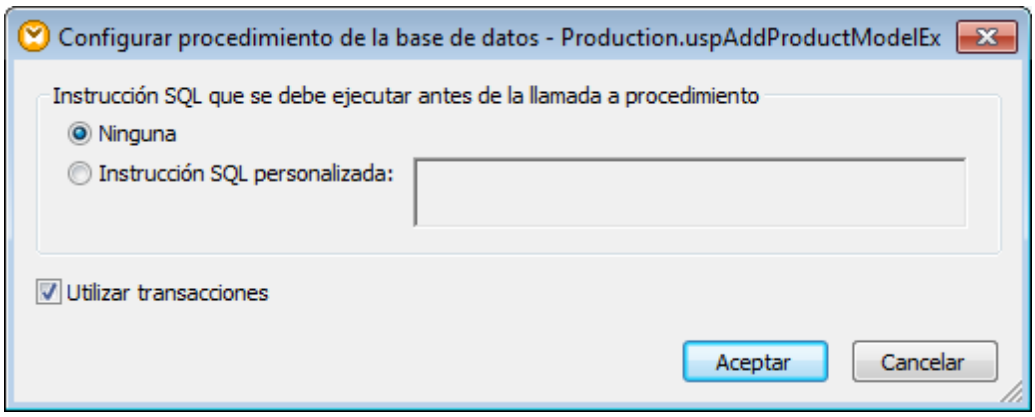
procedimiento almacenado para indicar que se usará en la relación local, pero no tiene iconos de entrada ni salida.

- La tabla ProductModelIllustration también se añadió como elemento secundario del procedimiento almacenado.
- Expanda la tabla pare ver las claves y columnas de la tabla. Observe que la clave ProductModelID muestra el procedimiento almacenado y el nombre de parámetro con el que está relacionado.
- Aquí no se pueden usar las relaciones locales que usan el conjunto de registros (resultados) del procedimiento almacenado.
- Si hace clic en el botón interactivo del procedimiento almacenado y selecciona **Configuración del procedimiento**, podrá definir una instrucción SQL que se ejecutará antes de llamar al procedimiento y podrá activar la opción para transacciones.



Para definir una transacción para un procedimiento almacenado:

1. Haga clic en el botón de procedimiento almacenado y seleccione la opción **Configuración del procedimiento**.
2. Marque la casilla *Usar transacciones* y haga clic en **Aceptar**.

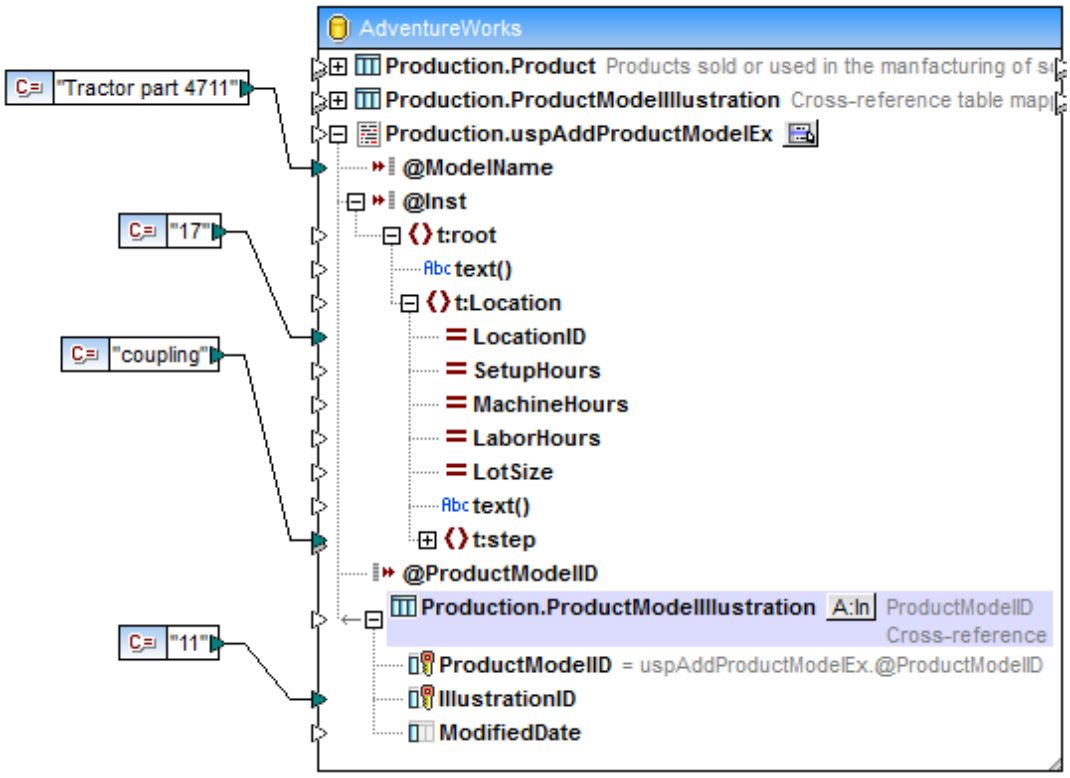


Definir la transacción para el procedimiento almacenado garantiza que tanto la recuperación de la clave como la inserción del registro sucedan durante la misma transacción.

Para terminar la asignación:

La imagen siguiente muestra un subconjunto de los datos que se asignarían.

1. Asigne los elementos de origen (las constantes) a la BD de destino.



2. Abra el panel **Resultados** para ver el SQL que se enviará a la BD.

```
BEGIN TRANSACTION

EXECUTE NULL = [Production].[uspAddProductModelEx] 'Tractor part 4711','<root
xmlns="http://schemas.microsoft.com/sqlserver/2004/07/adventure-works/ProductModelManuInstructions
"><Location LocationID="17"><step>coupling</step></Location></root>',NULL OUT;
-->>> %@uspAddProductModelEx1%
-->>> %@ProductModelID1%

INSERT INTO [Production].[ProductModelIllustration] ([ProductModelID], [IllustrationID]) VALUES (
'%@ProductModelID1%', 11)

COMMIT TRANSACTION
```

Antes de la acción de inserción MapForce llama automáticamente al procedimiento almacenado una vez por cada registro.

© 2018 Altova GmbH

7.3 Archivos CSV y archivos de texto

MapForce admite asignaciones de datos entre formatos de archivo basados en texto como archivos de texto CSV (valores separados por comas) y archivos de texto FLF (campos de longitud fija). Por ejemplo, en MapForce puede crear estas transformaciones de datos:

- transformar esquemas XML en archivos planos y viceversa
- transformar bases de datos en archivos planos y viceversa
- transformar UN/EDIFACT y ANSI X12 en archivos planos o bases de datos y viceversa

Recuerde que, en el caso del formato CSV, los archivos pueden usar delimitadores como comas, tabulaciones, punto y comas, espacios y otros valores.

Además de asignaciones de datos de archivos CSV y FLF, MapForce permite trabajar con estructuras más complejas gracias a la aplicación MapForce FlexText, un módulo disponible con la edición MapForce Enterprise Edition. FlexText permite definir la estructura de los datos de texto (con ayuda de una *plantilla FlexText)* y así poder asignarlos a otros formatos. Consulte la documentación de MapForce FlexText para más información.

La asignación de datos de archivos de texto solamente es compatible con los lenguajes de transformación Java, C#, C++ y BUILT-IN.

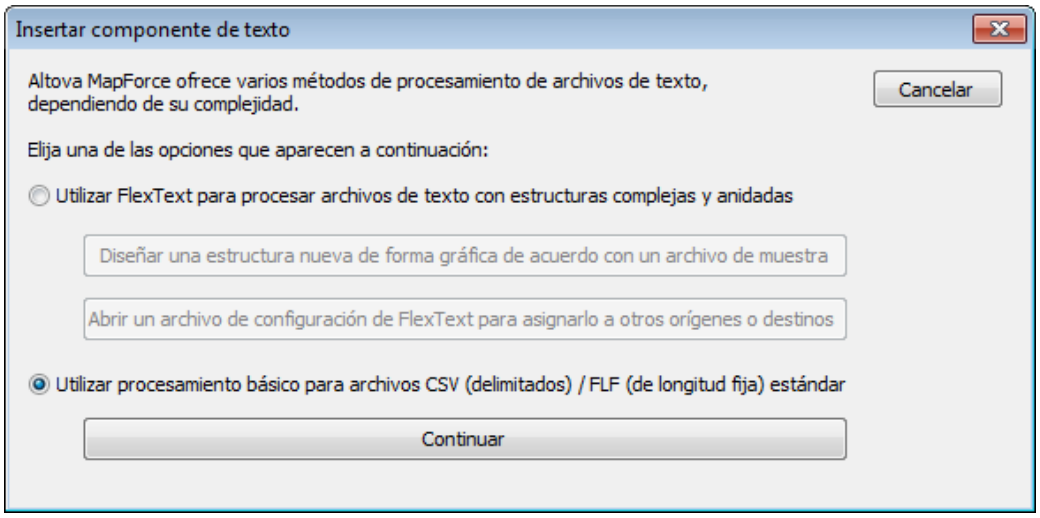
Hay dos maneras de generar resultados para asignaciones datos de archivos planos:

- Abriendo el panel *Resultados*: esto genera una vista previa con el motor de ejecución integrado. También puede guardar el resultado de la asignación con el comando de menú **Resultados | Guardar el archivo de salida...** o con el botón **Guardar resultado generado**  de la barra de herramientas del panel *Resultados*.
- Seleccionando el comando de menú **Archivo | Generar código en | Java/C#/C++** y compilando y ejecutando el código generado.

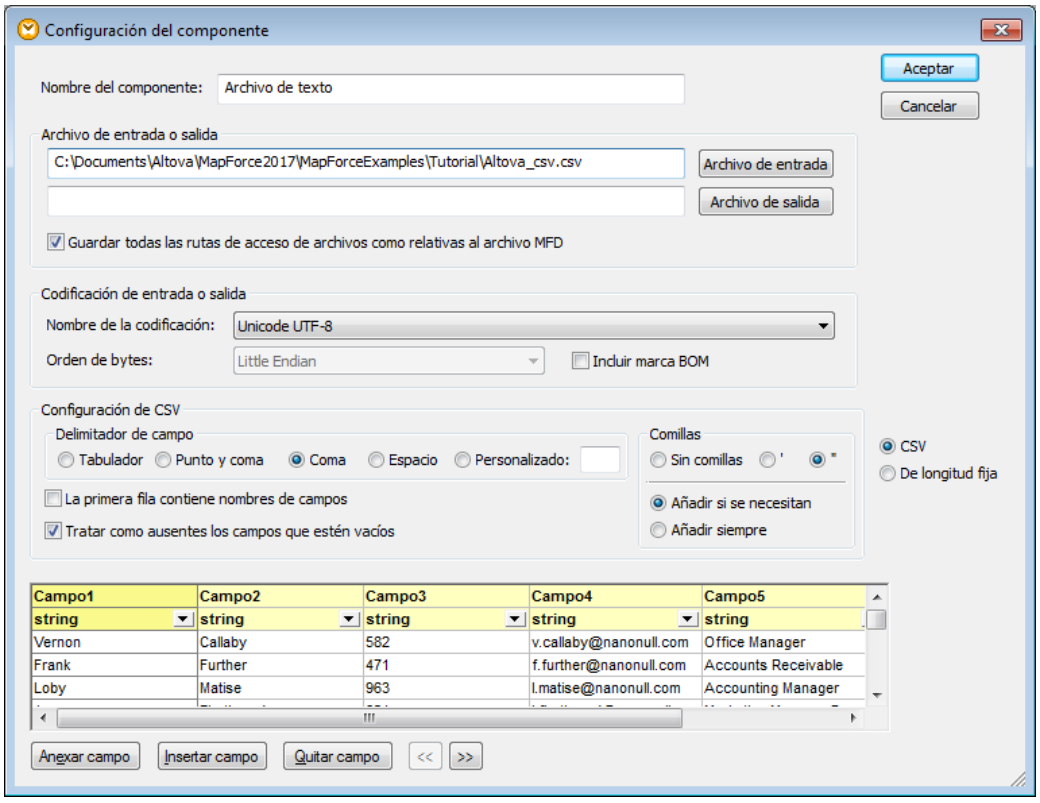
7.3.1 Ejemplo: asignar archivos CSV a XML

El objetivo de este ejemplo es crear una asignación que lea datos de un archivo CSV simple y los escriba en un archivo XML de destino. Los archivos utilizados en este ejemplo se encuentran en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`.

1. Seleccione uno de estos lenguajes de transformación: Java, C#, C++ o BUILT-IN.
2. Añada un componente de archivo de texto al área de asignación (con el comando de menú **Insertar | Archivo de texto** o con el botón **Insertar archivo de texto**  de la barra de herramientas).

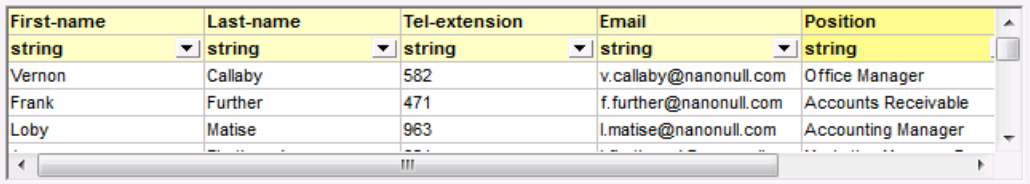


3. Seleccione la opción *Utilizar procesamiento básico...* y después haga clic en **Continuar**.
4. En el cuadro de diálogo "Configuración del componente" haga clic en el botón Archivo de entrada y navegue hasta el archivo **Altova_csv.csv**. El contenido del archivo aparece ahora en la parte inferior del cuadro de diálogo. Observe que en el modo de vista previa solo aparecen las primeras 20 filas del archivo de texto.



5. Haga clic dentro del encabezado Campo1 y cambie el texto por First-name. Haga lo mismo con el resto de campos: Campo 2 => Last-name, Campo 3 =>Tel-extension, Campo 4 => Email, Campo 5 => Position. **Nota:** pulse la tecla **Tabulación** para recorrer los campos.

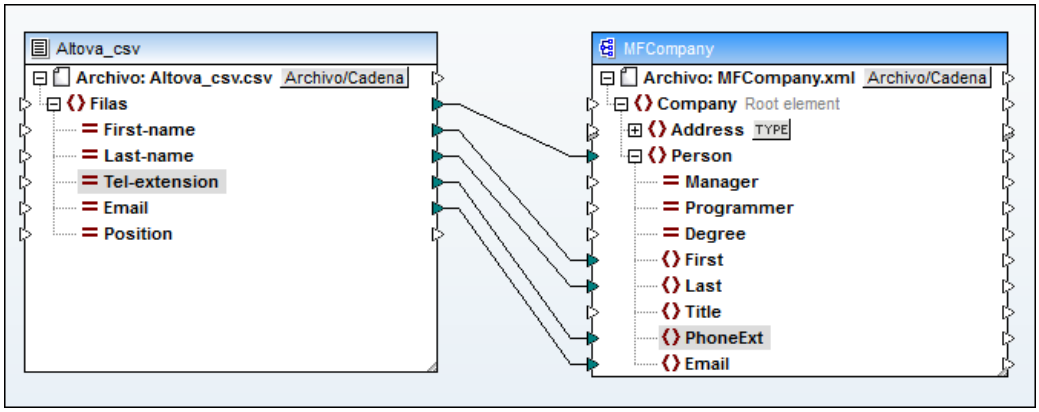
© 2018 Altova GmbH



First-name	Last-name	Tel-extension	Email	Position
string	string	string	string	string
Vernon	Callaby	582	v.callaby@nanonull.com	Office Manager
Frank	Further	471	f.further@nanonull.com	Accounts Receivable
Loby	Matise	963	l.matise@nanonull.com	Accounting Manager

6. Cuando termine, haga clic en **Aceptar**.
7. Cuando la aplicación solicite un nombre nuevo para el componente, haga clic en **Cambiar el nombre del componente**. El componente CSV aparece ahora en la asignación.
8. Añada **MFCompany.xsd** como componente XML de destino de la asignación (con el comando **Insertar | Archivo o esquema XML**).
9. Haga clic en **Omitir** cuando la aplicación solicite un archivo XML de muestra y seleccione el elemento raíz `Company`.
10. Realice las asignaciones correspondientes entre los dos componentes (asegurándose de asignar el elemento `Filas` al elemento `Person` del esquema de destino).

El conector que une el elemento `Filas` del componente CSV con el elemento `Person` del esquema es esencial porque define qué elementos se deben recorrer. Es decir, por cada fila del archivo CSV, se creará un nuevo elemento `Person` en el archivo XML de salida.



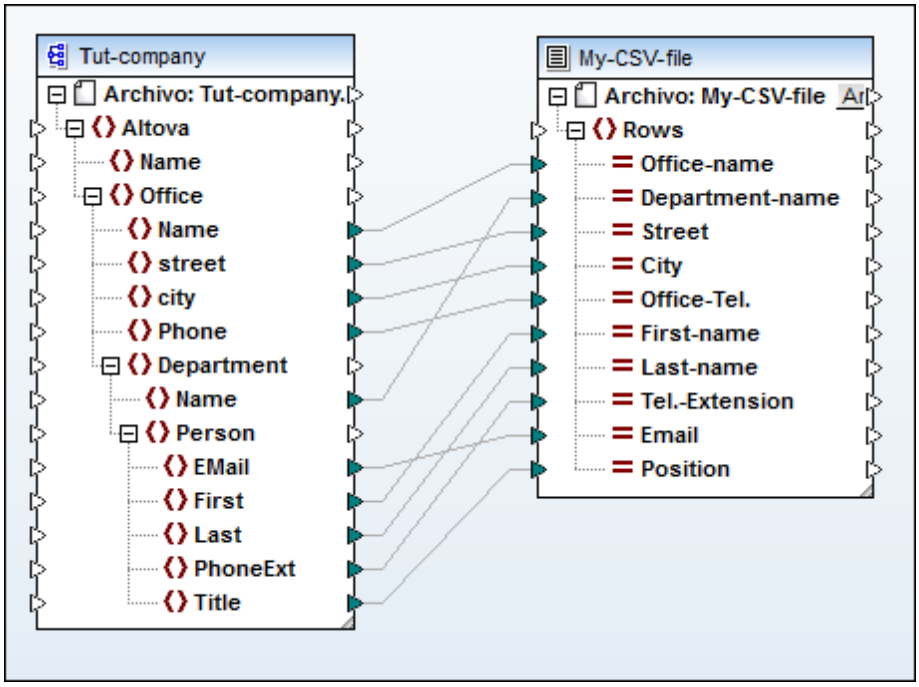
11. Abra el panel *Resultados* para ver el resultado de la asignación.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
3    <Person Manager="true">
4      <First>Vernon</First>
5      <Last>Callaby</Last>
6      <PhoneExt>582</PhoneExt>
7      <Email>v.callaby@nanonull.com</Email>
8    </Person>
9    <Person Manager="true">
10     <First>Frank</First>
11     <Last>Further</Last>
12     <PhoneExt>471</PhoneExt>
13     <Email>f.further@nanonull.com</Email>
14   </Person>
15   <Person Manager="true">
```

Los datos del archivo CSV se asignaron correctamente al archivo XML de destino.

7.3.2 Ejemplo: recorrer elementos

Este ejemplo pretende explicar cómo se crean iteraciones (varias filas) en un archivo CSV de destino. El diseño de asignación que acompaña este ejemplo está en la ruta de acceso `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Tut-xml2csv.mfd`.



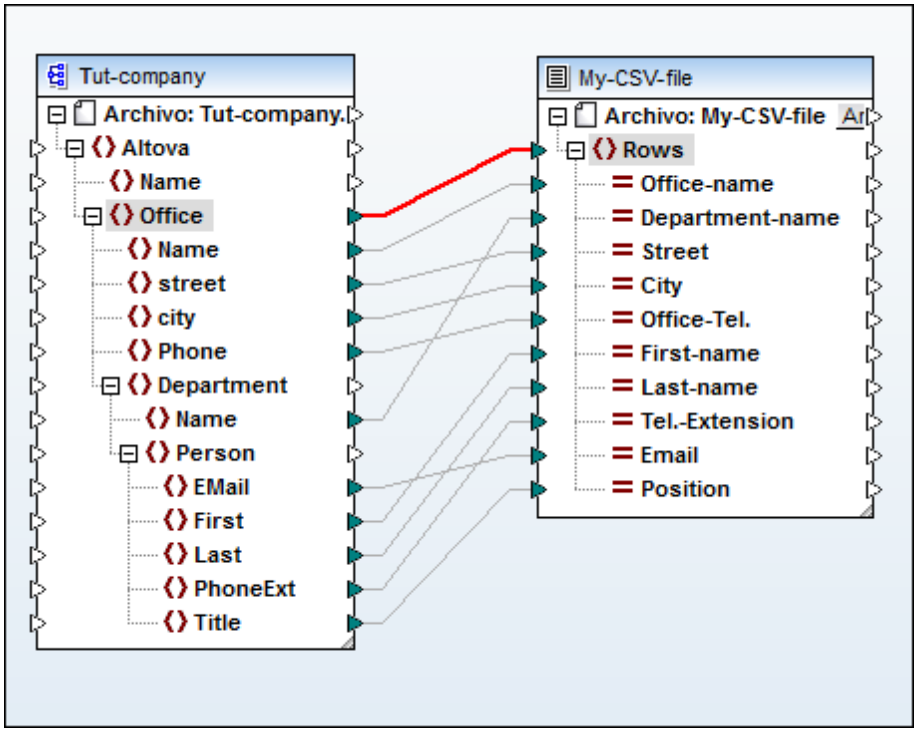
Tut-xml2csv.mfd

Tenga en cuenta que este diseño de asignación no está terminado a propósito. Si intenta validar el archivo de ejemplo con el comando de menú **Archivo | Validar asignación**, recibirá mensajes de advertencia. Además, si abre la vista previa de resultados, verá que solamente se genera una fila.

Imaginemos que lo que usted necesita es crear varias filas en el archivo CSV a partir de una secuencia de elementos del archivo XML. Esto se puede hacer dibujando una conexión con el elemento `Rows` del archivo CSV de destino.

Por ejemplo, para recorrer todas las oficinas y conseguir que el resultado aparezca en el archivo CSV es necesario conectar `Office` con `Rows`. Esta conexión indica que por cada elemento `Office` del archivo XML de origen MapForce debe crear una fila en el archivo CSV de destino.

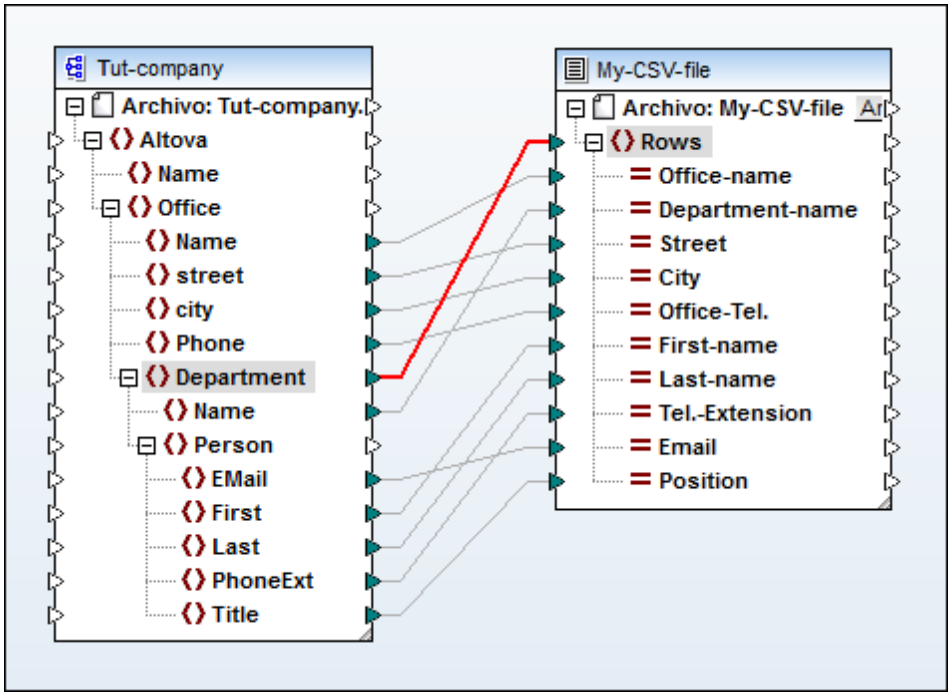
© 2018 Altova GmbH



El elemento `Rows` del componente CSV hace de iterador para la secuencia de elementos a los que está conectado. Por tanto, si conecta el elemento `Office`, el resultado crea una fila por cada oficina que se encontrara en el archivo XML de origen.

```
1   "Microtech, Inc.",Level 1 support,Major Ave 1,Vancouver,558833
2   "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Otto
3
```

Igualmente, si conecta **Department** a **Rows**, se generará una fila por cada departamento que se encuentre en el archivo XML de origen.



Este sería el resultado:

```
1   "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Clive,Clo
2   "Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,558
3   "Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,K
4   "Microtech, Inc.",Level 1 support,Major Ave 1,Vancouver,5588339
5   "Microtech Partners, Inc.",Admin,Perro Bvd 1324,Ottowa,3549202,
6   "Microtech Partners, Inc.",Sales and Marketing,Perro Bvd 1324,O
7   "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Ottow
8
```

Por último, la asignación entre `Person` y `Rows` hace que en el resultado se generen todos los elementos `Person`. En este caso MapForce recorre los registros de esta manera: cada `Person` dentro de cada `Department`, dentro de cada `Office`.

7.3.3 Ejemplo: crear jerarquías a partir de archivos CSV y de texto de longitud fija

Este ejemplo es la asignación `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Tut-headerDetail.mfd` y utiliza un archivo CSV llamado `Orders.csv` que tiene este formato:

- Campo 1: H define un registro de encabezado y D un registro detallado.
- Campo 2: una clave común tanto para los registros de encabezado como para los registros detallados.
- Cada registro de encabezado o detallado está en una línea distinta.

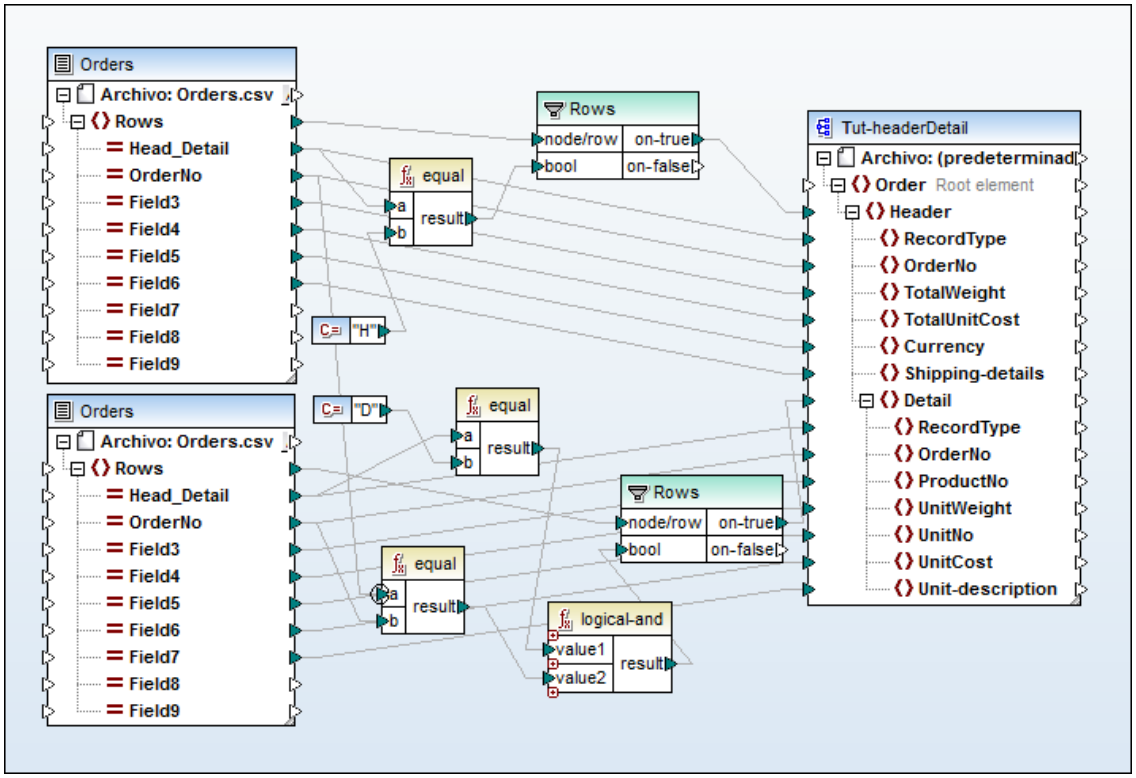
A continuación puede ver el contenido del archivo `Orders.csv`.

© 2018 Altova GmbH

```
H,111,332.1,22537.7,,Container ship,,,
D,111,A-1579-227,10,3,400,Microtome,,
D,111,B-152-427,7,6,1200,Miscellaneous,,
H,222,978.4,7563.1,,Air freight,,,
D,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,
```

Nuestro diseño de asignación tiene varios objetivos:

- Asignar el archivo CSV plano a un archivo XML jerárquico.
- Filtrar los registros de encabezado (señalados con una H).
- Asociar los correspondientes registros detallados (señalados con una D) a cada uno de los registros de encabezado.



tut-headerDetail.mfd

Para conseguir nuestros objetivos primero debemos asegurarnos de que los registros de encabezado y los registros detallados tengan un campo común. En este caso, el campo común (también denominado *clave*) es el segundo campo del archivo CSV (es decir, `OrderNo`). En el archivo CSV, tanto el primer registro de encabezado como los dos registros detallados que le siguen contienen el valor común 111.

El archivo `Orders.csv` se insertó dos veces a fin de hacer el proceso de asignación más intuitivo.

El archivo de esquema **Tut-headerDetail.xsd** tiene una estructura jerárquica: `Order` es el

elemento raíz, `Header` es su elemento secundario y `Detail` es un elemento secundario de `Header`.

El primer archivo `Orders.csv` aporta los registros de encabezado `Header` (y todos los campos asignados) al elemento `Header` del archivo de esquema de destino. El componente de filtrado se usa para filtrar solamente los registros que empiecen por H. El elemento `Rows` aporta estos registros filtrados al elemento `Header` del esquema de destino.

El segundo archivo `Orders.csv` aporta los registros detallados `Detail` (y todos los campos asignados) filtrando los registros `Detail` que coinciden con la clave `OrderNo` del registro `Header`. Este filtrado se hace así:

- Se compara el campo **OrderNo** del registro `Header` con el mismo campo de los registros `Detail` por medio de la función **equal** (el contexto prioritario se estableció en el parámetro **a** para un mayor rendimiento).
- Se usa la función **Logical-and** para aportar solamente los registros `Detail` que contengan el mismo campo `OrderNo` que el registro `Header`.

A través del parámetro **on-true** del componente de filtrado, el elemento **Rows** aporta estos registros filtrados a los elementos Header y Detail del esquema de destino.

En el panel *Resultados* se puede ver el archivo XML de salida. Cada registro de encabezado contiene sus propios datos más los registros detallados asociados que tengan el mismo número de pedido.

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation
  <Header>
    <RecordType>H</RecordType>
    <OrderNo>111</OrderNo>
    <TotalWeight>332.1</TotalWeight>
    <TotalUnitCost>22537.7</TotalUnitCost>
    <Currency/>
    <Shipping-details>Container ship</Shipping-details>
    <Detail>
      <RecordType>D</RecordType>
      <OrderNo>111</OrderNo>
      <ProductNo>A-1579-227</ProductNo>
      <UnitWeight>10</UnitWeight>
      <UnitNo>3</UnitNo>
      <UnitCost>400</UnitCost>
      <Unit-description>Microtome</Unit-description>
    </Detail>
    <Detail>
      <RecordType>D</RecordType>
      <OrderNo>111</OrderNo>
      <ProductNo>B-152-427</ProductNo>
      <UnitWeight>7</UnitWeight>
      <UnitNo>6</UnitNo>
      <UnitCost>1200</UnitCost>
      <Unit-description>Miscellaneous</Unit-description>
    </Detail>
  </Header>
```

Veamos ahora otro ejemplo que usa un archivo CSV algo diferente: la asignación `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Head-detail-inline.mfd`. Estas son

© 2018 Altova GmbH

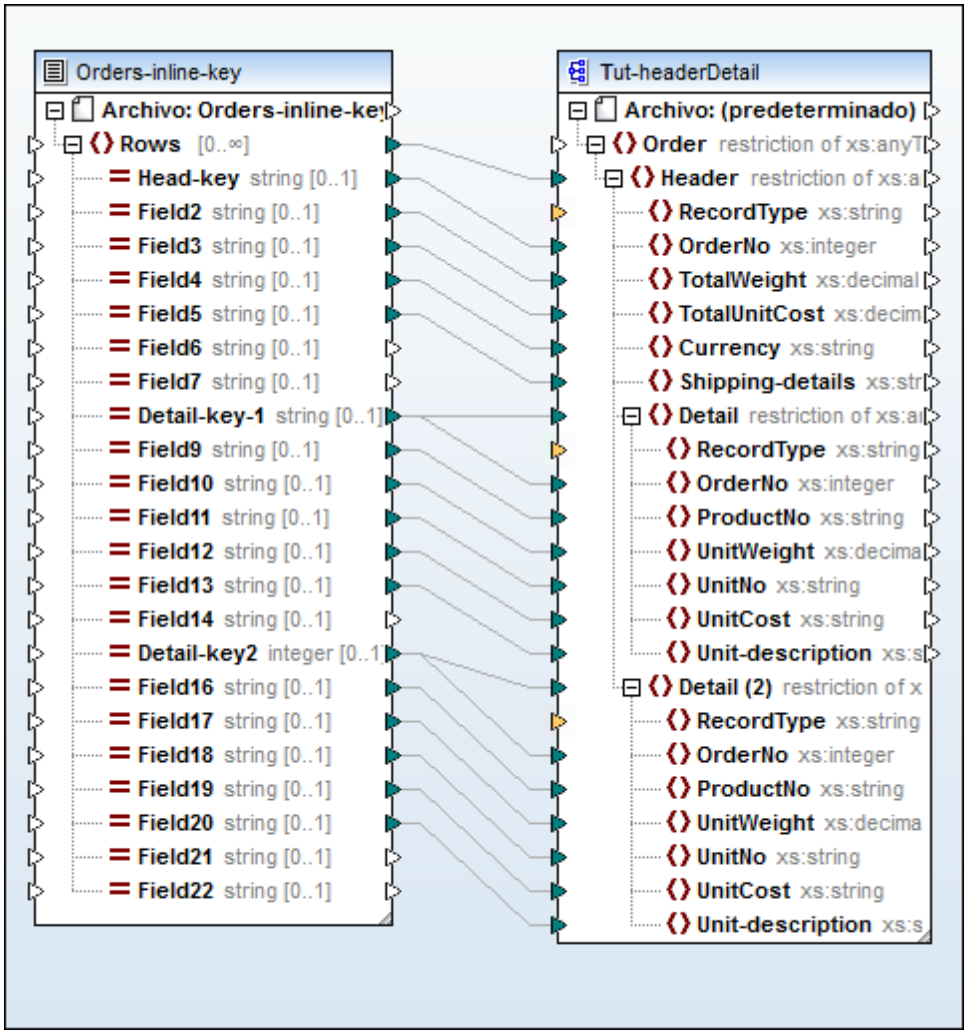
las características de este nuevo archivo CSV:

- No incluye ningún designador de registro (H o D).
- Incluye un campo de clave común (el primer campo del archivo CSV) tanto para los registros de encabezado (Head-key) como para los detallados (Detail-key). Este campo está asignado al elemento OrderNo del esquema de destino.
- El encabezado y sus correspondientes campos detallados están en la misma línea.

```
111,332.1,22537.7,,Container ship,,,111,A-1579-227,10,3,400,Microtome,,111,B-15
222,978.4,7563.1,,Air freight,,,222,ZZ-AW56-1,10,5,10000,Gas Chromatograph,,,,
```

Teniendo en cuenta lo anterior, la asignación se diseñó así:

- Los campos clave se asignaron a los correspondientes elementos `OrderNo` del esquema de destino.
- El elemento `Detail` del esquema de destino se duplicó y aparece como **Detail (2)**. Esto permite asignar el segundo conjunto de registros detallados al elemento correcto.
- El resultado de la asignación es el mismo archivo XML que se generó en el primer ejemplo de este apartado.



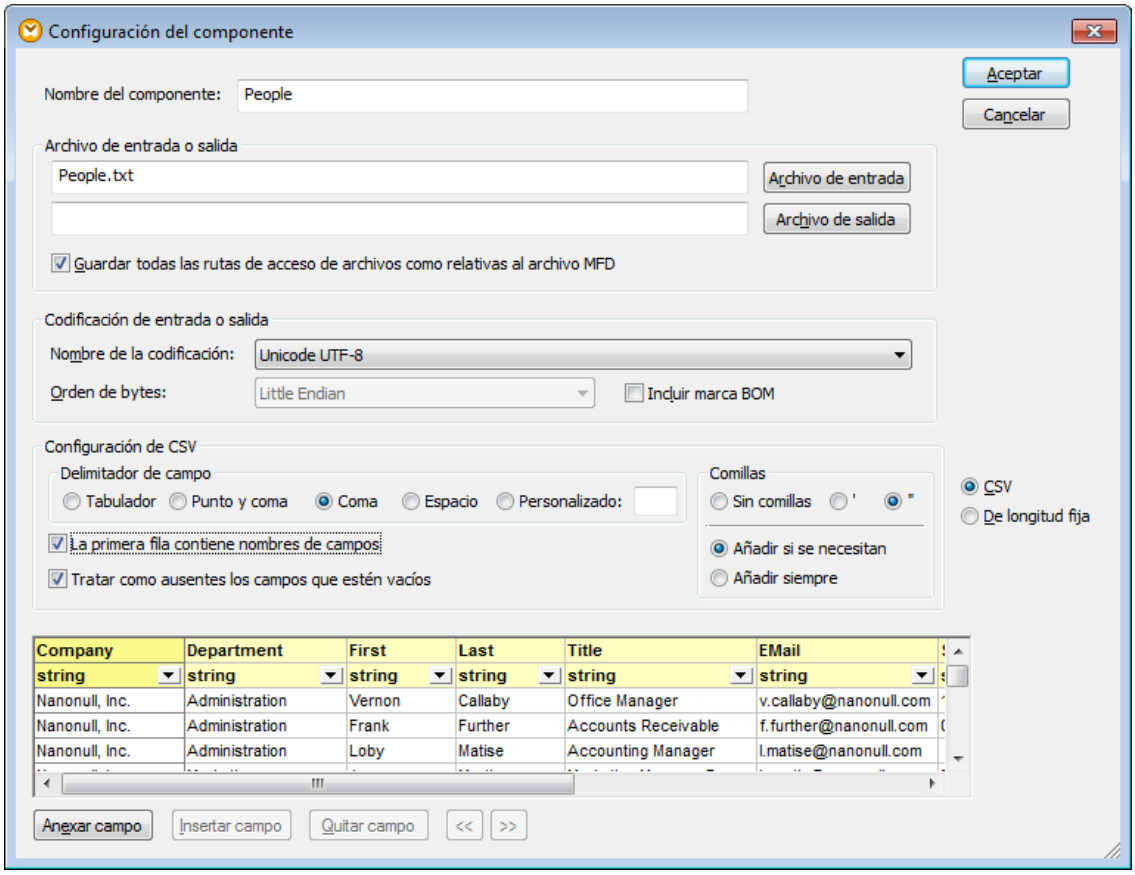
Head-detail-inline.mfd

7.3.4 Opciones de configuración de componentes CSV

Una vez añadido al área de asignación, el componente de texto se puede configurar desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de varias formas distintas:

- Seleccionando el componente y haciendo clic en el comando de menú **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y seleccionando **Propiedades** en el menú contextual.

© 2018 Altova GmbH



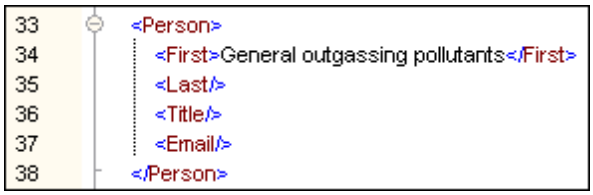
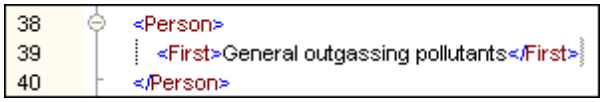
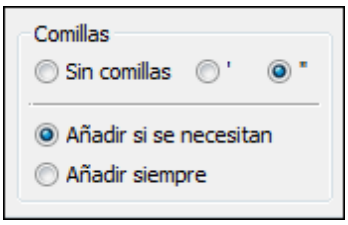
Cuadro de diálogo "Configuración del componente" (en modo CSV)

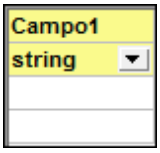
Estas son las opciones de configuración de los componentes CSV:

Nombre del componente	El nombre del componente se genera automáticamente cuando se crea el componente. Sin embargo, puede cambiar el nombre en cualquier otro momento. Si el nombre del componente se generó automáticamente y después seleccionó un archivo de instancia, MapForce preguntará si también desea actualizar el nombre del componente. El nombre del componente puede contener espacios y puntos. El nombre del componente no puede incluir barras diagonales, dos puntos, comillas dobles ni espacios iniciales ni finales. A la hora de cambiar el nombre del componente debe tener en cuenta que: • Si tiene pensado implementar la asignación en FlowForce Server, el nombre del componente debe ser único. • Recomendamos usar caracteres que se puedan usar

	en la línea de comandos. Tenga en cuenta que los caracteres regionales no tienen la misma codificación en Windows y en la línea de comandos.
Archivo de entrada	Especifica el archivo donde MapForce debe leer los datos. Este campo es relevante para componentes de origen y se rellena cuando se crea el componente y se le asigna un archivo de texto. El campo puede dejarse vacío si se usa un componente de archivo de texto como destino de la asignación. En un componente de origen MapForce usa el valor de este campo para leer nombres de columna y obtener una vista previa del contenido del archivo de texto. Haga clic en **Archivo de entrada** para seleccionar un archivo nuevo.
Archivo de salida	Especifica el archivo donde MapForce debe escribir datos. Este campo es relevante para componentes de destino. Haga clic en **Archivo de salida** para seleccionar un archivo nuevo.
Guardar todas las rutas de acceso de archivos como relativas al archivo MFD	Si marca esta casilla, MapForce guarda las rutas de acceso de los archivos que aparecen en el cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd). Esta opción afecta a los archivos de entrada y salida utilizados por el componente de texto (véase también *Usar rutas de acceso relativas en un componente*).
Codificación de entrada / salida	Permite configurar estos aspectos del archivo de instancia de salida: • Nombre de la codificación • Orden de bytes • El carácter de la marca de orden de byte (marca BOM). Por defecto todos los componentes nuevos tienen la codificación definida en la opción *Codificación predeterminada para componentes nuevos*. Para acceder a esta opción seleccione **Herramientas | Opciones | Generales**.
Delimitador de campos	Los archivos CSV están delimitados por comas por defecto. Esta opción permite elegir entre tres tipos de delimitadores: *tabulador*, *punto y coma* o *espacio*. También puede introducir un delimitador personal en el campo *Personalizado*.
La primera fila contiene nombres de campos	Seleccione esta opción para indicar a MapForce que debe tratar los valores del primer registro del archivo de texto como encabezados de columna. Los encabezados de columna

© 2018 Altova GmbH

	aparecerán como nombres de elemento en la asignación.
Tratar como ausentes los campos que estén vacíos	Si marca esta casilla, los campos vacíos del archivo de origen no producirán un elemento correspondiente vacío (elemento o atributo) en el archivo de destino. Por ejemplo, el registro CSV `"General outgassing pollutants,,,,"` está compuesto por cuatro campos y los tres últimos están vacíos. Imaginemos que el destino de la asignación es un archivo XML. Si se desactiva esta opción, los campos vacíos se crearán en el archivo de destino con un valor vacío (p. ej. los elementos `Last`, `Title` y `Email`):  Si se activa esta opción, los campos vacíos no se crearán en el destino de la asignación: 
Comillas	Si en el archivo de entrada los valores de campo están entrecomillados, seleccione qué carácter de comillas existe en el archivo de origen. La misma configuración se usará para los archivos de salida.  Además, los archivos de salida cuentan con dos opciones de configuración más: **Añadir si se necesitan** Añade el carácter de comillas selecci campos cuyo texto contiene el delimi campo o saltos de línea. **Añadir siempre** Añade el carácter de comillas selecci todos los campos del archivo CSV ge

CSV / De longitud fija	Cambia el tipo de componente por CSV o FLF (campo de longitud fija).
Panel de vista previa	La parte inferior del cuadro de diálogo muestra una vista previa de hasta 20 dilas del archivo seleccionado como entrada/salida. Si lo necesita, puede crear la estructura del archivo (o cambiarla) con estos botones: **Anexar campo** Crea un campo nuevo después del últ CSV. **Insertar campo** Crea un campo nuevo inmediatamente registro CSV que está seleccionado. **Quitar campo** Elimina el campo que está seleccion **<<** Desplaza el campo que está seleccio izquierda. **>>** Desplaza el campo que está seleccio derecha. Para cambiar el nombre de un campo haga clic en el encabezado (por ejemplo, **`Campo1`**) y teclee el nuevo valor. Recuerde que los nombres de campo no se pueden editar si está marcada la casilla *La primera fila contiene nombres de campos*.  Para cambiar el tipo de datos de un campo seleccione el valor pertinente en la lista desplegable. MapForce revisa el tipo de datos y, si los datos de entrada y el formato del campo son discordantes, resalta los datos en rojo.  Los tipos de los campos se basan en los tipos de datos predeterminados de XML Schema. Por ejemplo, el tipo de fecha Date está en el formato **AAAA-MM-DD**.

© 2018 Altova GmbH

7.3.5 Ejemplo: asignar archivos de texto de longitud fija a bases de datos

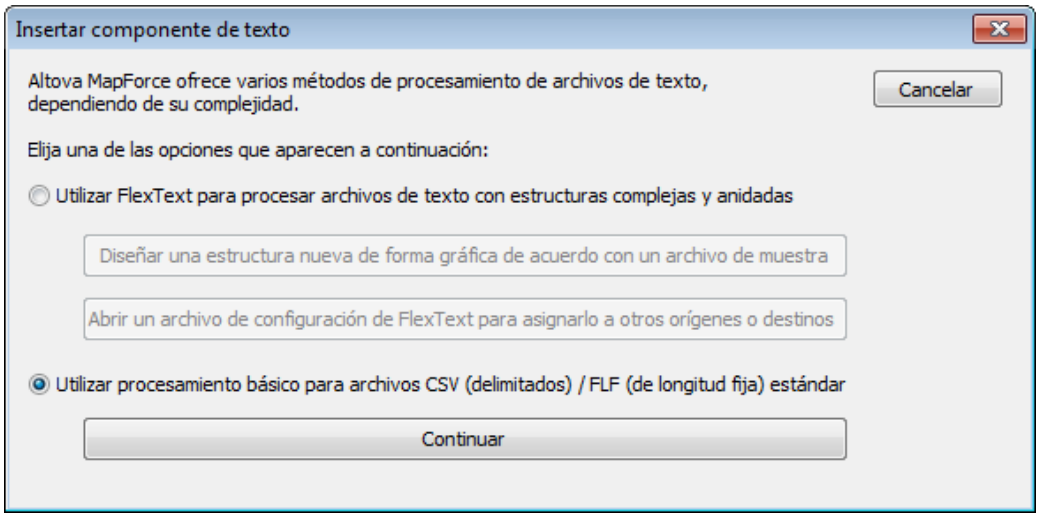
Este ejemplo sirve para explicar una operación de asignación de datos entre un archivo de texto de longitud fija (FLF) y una base de datos Microsoft Access. Los archivos que se utilizan en este ejemplo están en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\`. Tanto el archivo de texto de origen como la base de datos de destino almacenan una lista de empleados. Además, en el archivo de origen los registros están delimitados de forma implícita en función de su tamaño:

Posición y nombre del campo	Tamaño (en caracteres)
Campo 1 (Nombre)	8
Campo 2 (Apellido)	10
Campo 3 (Extensión telefónica)	3
Campo 4 (Correo electrónico)	25
Campo 5 (Posición)	25

El objetivo de la asignación es actualizar la extensión telefónica de cada empleado en la base de datos con la extensión telefónica del archivo de origen, añadiendo además el prefijo 100 a cada extensión.

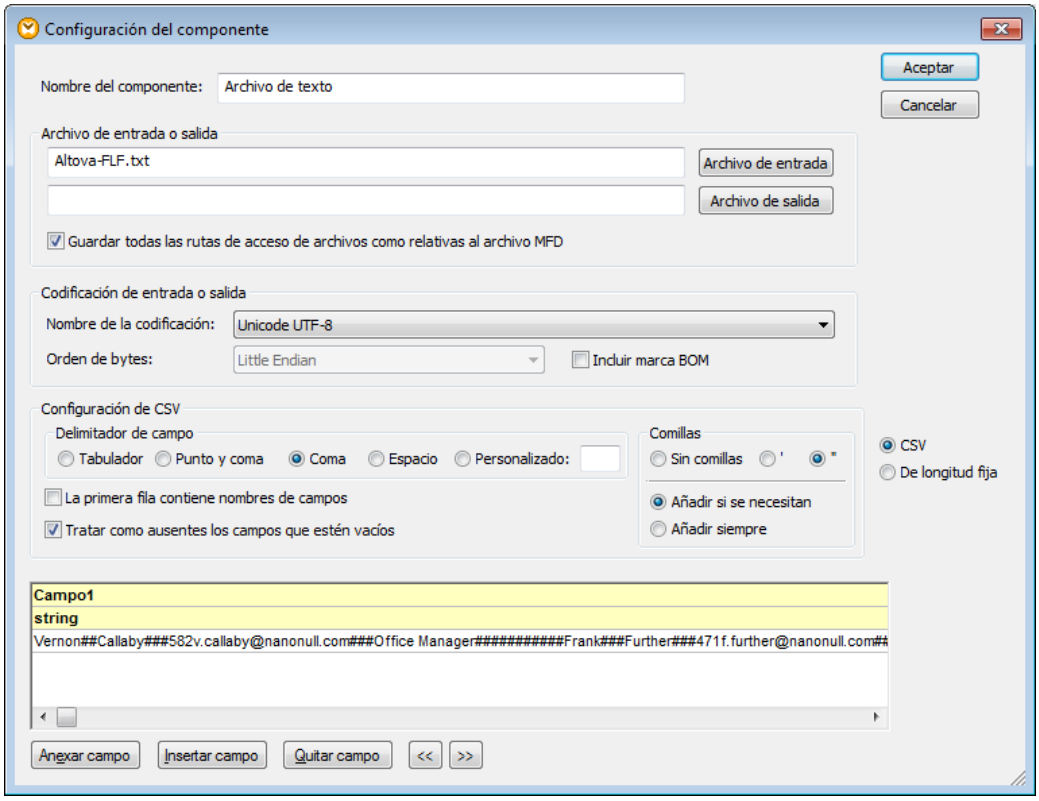
Paso nº1: insertar y configurar el componente de texto

1. Seleccione la opción de menú **Insertar | Archivo de texto** o haga clic en el botón **Insertar archivo de texto**  de la barra de herramientas. Esto abre el cuadro de diálogo "Insertar componente de texto".



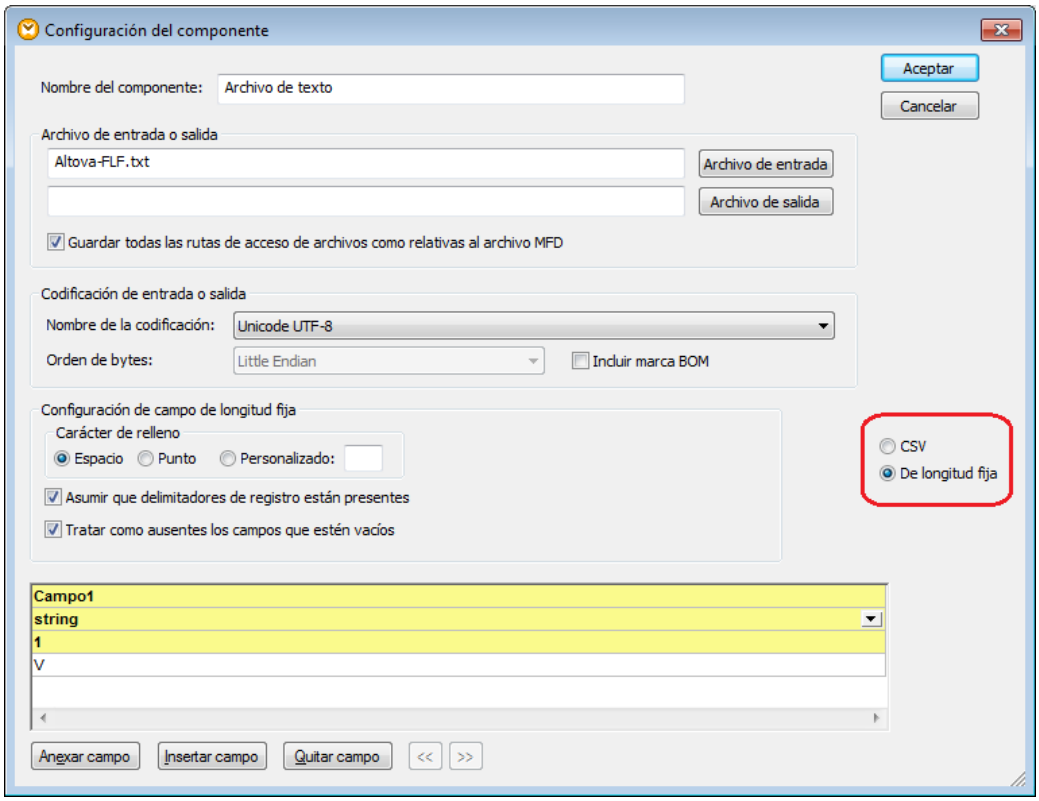
   Seleccione la opción *Utilizar procesamiento básico...* y haga clic en **Continuar**.

2. Haga clic en el botón **Archivo de entrada** y seleccione el archivo `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Altova-FLF.txt`. Observará que el archivo está compuesto por una sola cadena que contiene caracteres de relleno de tipo #.

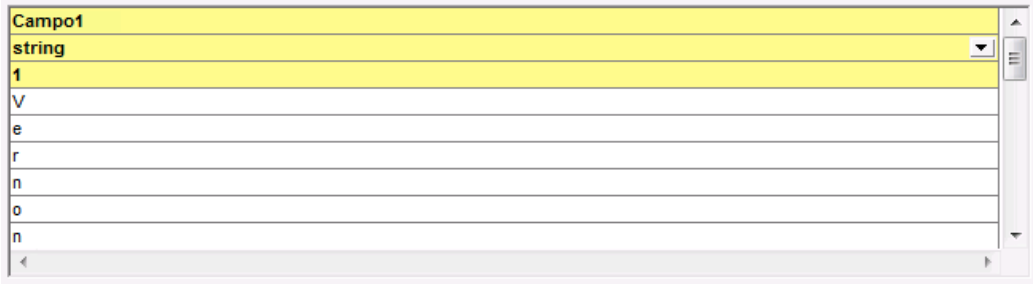


3. Ahora seleccione el botón de opción *De longitud fija*.

© 2018 Altova GmbH



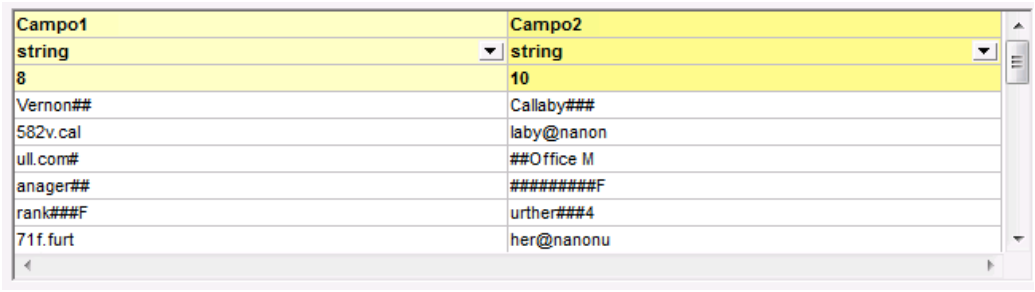
4. Desactive la casilla *Asumir que delimitadores de registro están presentes*.



5. Las tres filas que aparecen en amarillo se pueden editar y en ellas puede especificar: (i) el nombre del campo, (ii) el tipo de datos y (iii) el tamaño del campo. Introduzca ahora el valor **8** para el tamaño del campo y pulse **Entrar**. Ahora pueden verse más datos en la primera columna, que tiene un ancho de 8 caracteres.

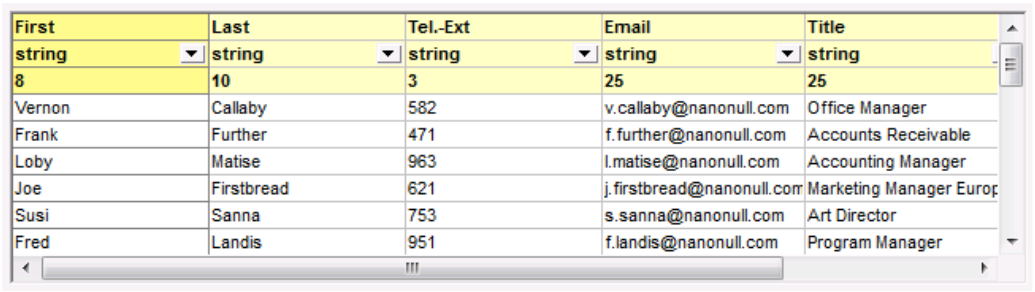


6. Haga clic en **Anexar campo** para añadir un campo nuevo y establezca su longitud en 10 caracteres.



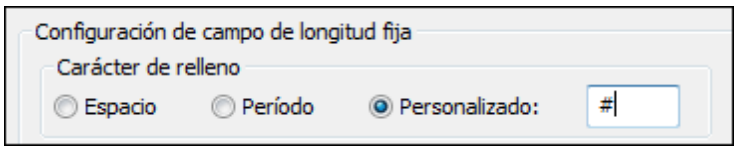
Campo1	Campo2
string	string
8	10
Vernon##	Callaby###
582v.cal	laby@nanon
ull.com#	##Office M
anager##	#########F
rank###F	urther###4
71f.furt	her@nanonu

7. Use el mismo método para crear tres campos más con una longitud de 3, 25 y 25 caracteres respectivamente. Ahora cambie el nombre de los 5 campos por: First, Last, Tel.-Ext, Email, Title. Esta sería la vista previa:

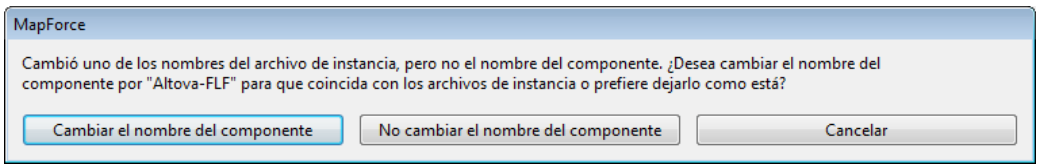


First	Last	Tel.-Ext	Email	Title
string	string	string	string	string
8	10	3	25	25
Vernon	Callaby	582	v.callaby@nanonull.com	Office Manager
Frank	Further	471	f.further@nanonull.com	Accounts Receivable
Loby	Matise	963	l.matise@nanonull.com	Accounting Manager
Joe	Firstbread	621	j.firstbread@nanonull.com	Marketing Manager Europ
Susi	Sanna	753	s.sanna@nanonull.com	Art Director
Fred	Landis	951	f.landis@nanonull.com	Program Manager

8. En el grupo de opciones *Configuración de campo de longitud fija* seleccione la opción *Personalizado* e introduzca una almohadilla (#). Esto le indica a MapForce que debe tratar todos los caracteres # como caracteres de relleno.

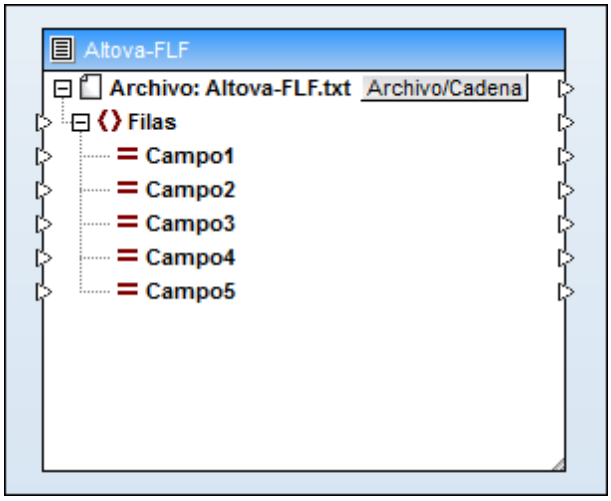


9. Haga clic en **Aceptar** para terminar. MapForce emite este aviso:



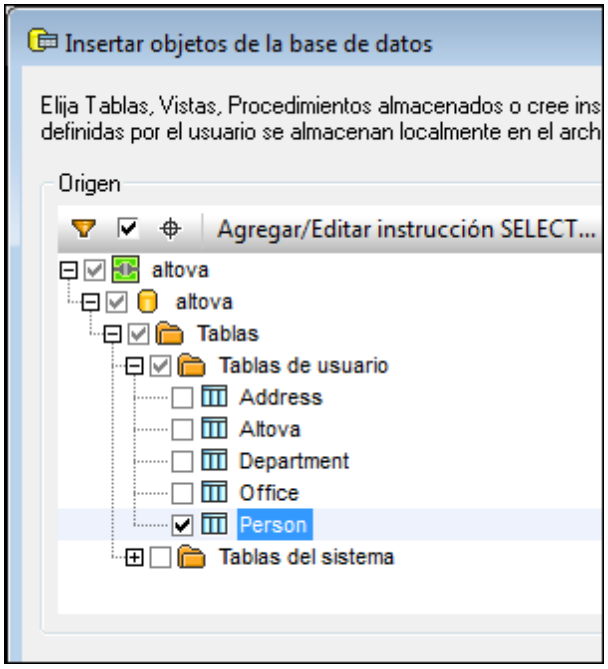
10. Haga clic en **Cambiar el nombre del componente**. El componente de texto aparecerá en el área de asignación, donde podrá crear conexiones con el componente de destino y otros componentes.

 © 2018 Altova GmbH



Paso nº2: insertar el componente de base de datos

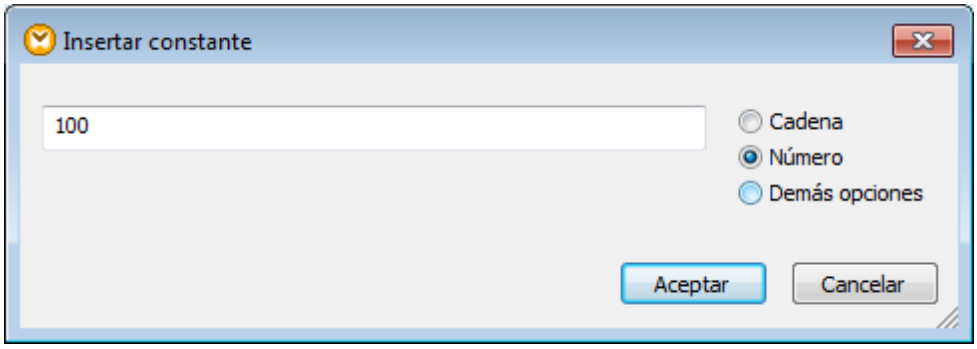
1. Seleccione el comando de menú **Insertar | Base de datos**, seleccione **Microsoft Access** y después haga clic en **Siguiente**.
2. Seleccione la base de datos **altova.mdb** que está en la carpeta `<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\` y después haga clic en **Conectarse**.
3. Seleccione la tabla **`Person`** y haga clic en **Aceptar**.



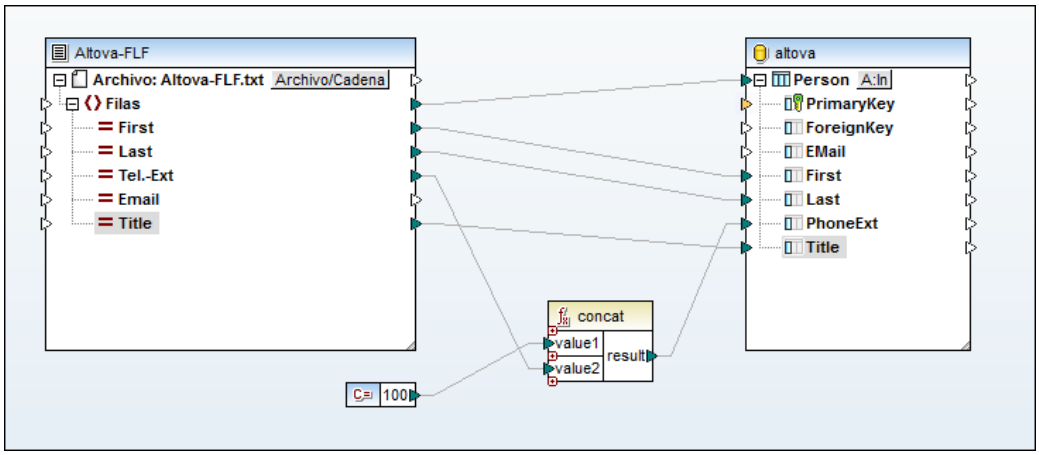
Paso nº3: diseñar la asignación

1. Arrastre la función **`core | concat`** desde la ventana Bibliotecas hasta el área de asignación.

2. Seleccione el comando de menú **Insertar | Constante**, seleccione el tipo *Número* e introduzca el valor `100`. Esta constante almacena el nuevo prefijo de extensión telefónica.

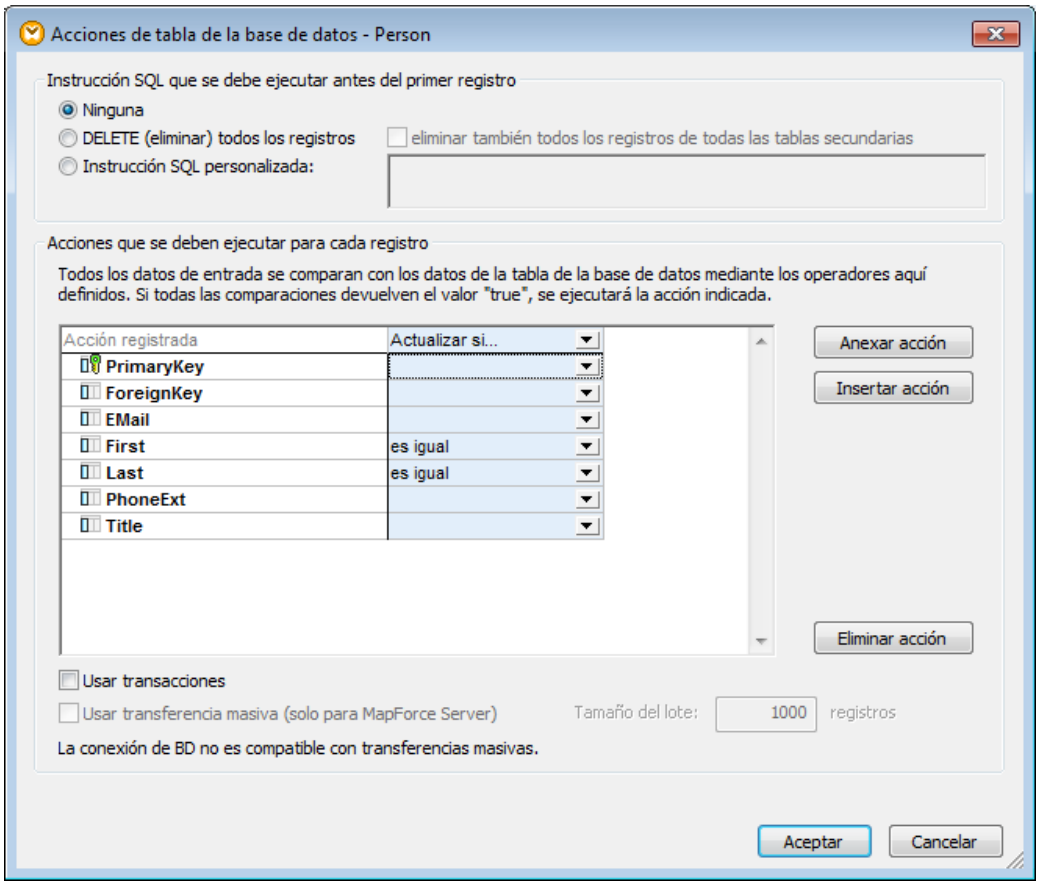


3. Cree las conexiones que aparecen en esta imagen:



4. En el componente de base de datos haga clic en el botón **Acción de tabla** A:In situado junto al elemento **Person**.
5. En el segundo grupo de acciones (*Acciones que se deben ejecutar para cada registro)* seleccione **Actualizar si...** en el primer cuadro combinado de la segunda columna. Ahora para los campos **First** y **Last** debe seleccionar la acción `es igual`. Esto indica que MapForce solo debe actualizar la tabla **Person** si el nombre y el apellido del archivo de origen es igual al campo correspondiente de la base de datos. Si la condición se cumple, la acción que se debe tomar viene dada por la asignación. En este caso, la extensión telefónica recibe el prefijo `100` y se copia en el campo **PhoneExt** de la tabla **Person**.

© 2018 Altova GmbH



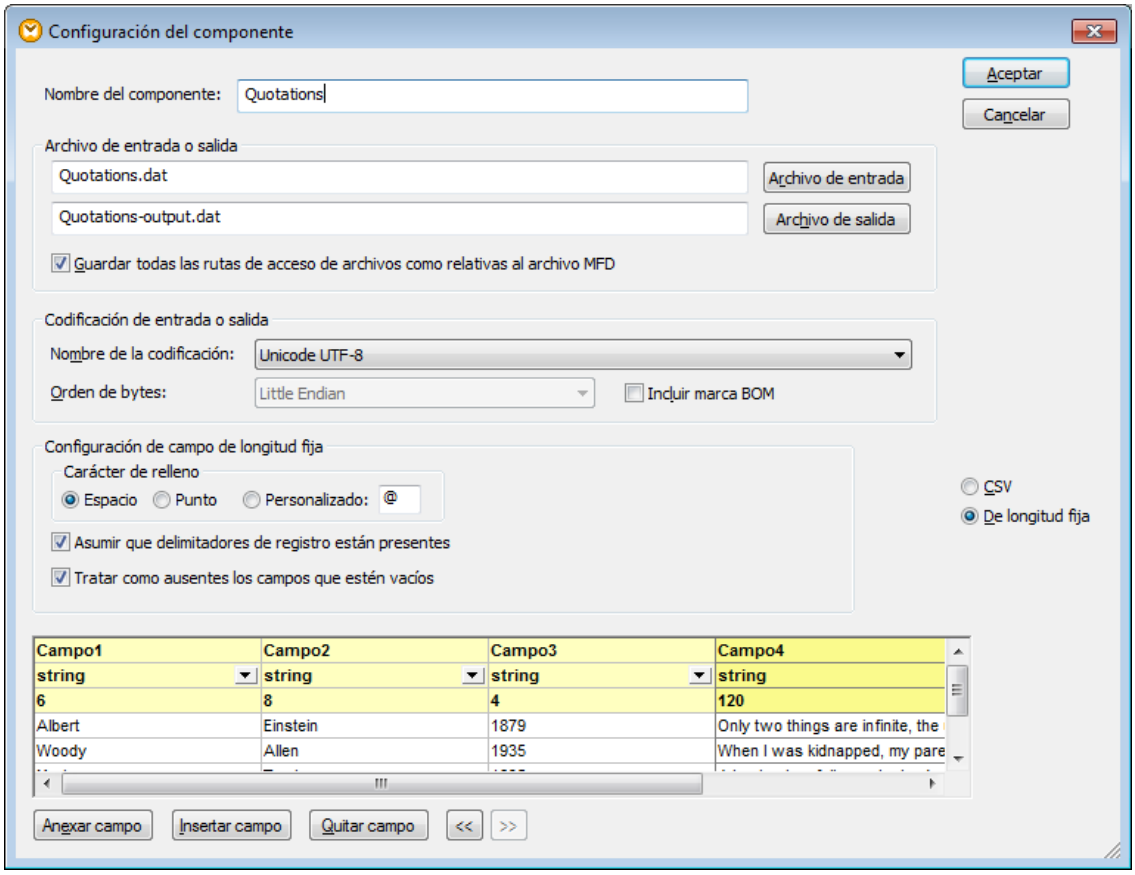
Paso nº4: ejecutar la asignación

Para generar la instrucción SQL (y obtener la vista previa en MapForce) haga clic en el panel *Resultados*. Para ejecutar las instrucciones SQL en la base de datos haga clic en el botón **Ejecutar script SQL**  .

7.3.6 Opciones de configuración de componentes FLF

Una vez añadido al área de asignación, el componente de texto se puede configurar desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de varias formas distintas:

- Seleccionando el componente y haciendo clic en el comando de menú **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y seleccionando **Propiedades** en el menú contextual.



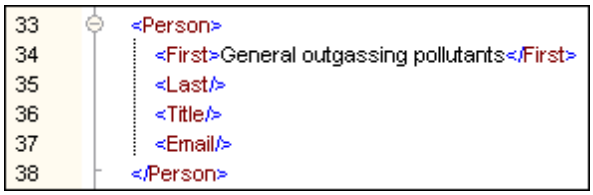
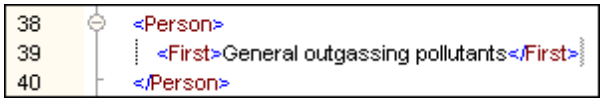
Cuadro de diálogo "Configuración del componente" (en modo de longitud fija)

Estas son las opciones de configuración de los componentes FLF:

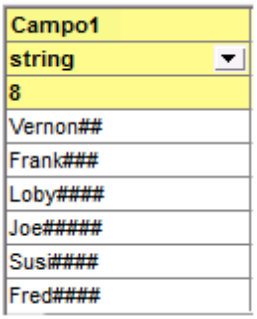
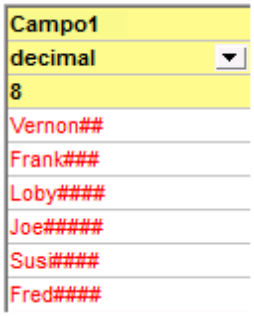
Nombre del componente	El nombre del componente se genera automáticamente cuando se crea el componente. Sin embargo, puede cambiar el nombre en cualquier otro momento. Si el nombre del componente se generó automáticamente y después seleccionó un archivo de instancia, MapForce preguntará si también desea actualizar el nombre del componente. El nombre del componente puede contener espacios y puntos. El nombre del componente no puede incluir barras diagonales, dos puntos, comillas dobles ni espacios iniciales ni finales. A la hora de cambiar el nombre del componente debe tener en cuenta que: • Si tiene pensado implementar la asignación en FlowForce Server, el nombre del componente debe ser único. • Recomendamos usar caracteres que se puedan usar

 © 2018 Altova GmbH

	en la línea de comandos. Tenga en cuenta que los caracteres regionales no tienen la misma codificación en Windows y en la línea de comandos.
Archivo de entrada	Especifica el archivo donde MapForce debe leer los datos. Este campo es relevante para componentes de origen y se rellena cuando se crea el componente y se le asigna un archivo de texto. El campo puede dejarse vacío si se usa un componente de archivo de texto como destino de la asignación. En un componente de origen MapForce usa el valor de este campo para leer nombres de columna y obtener una vista previa del contenido del archivo de texto. Haga clic en **Archivo de entrada** para seleccionar un archivo nuevo.
Archivo de salida	Especifica el archivo donde MapForce debe escribir datos. Este campo es relevante para componentes de destino. Haga clic en **Archivo de salida** para seleccionar un archivo nuevo.
Guardar todas las rutas de acceso de archivos como relativas al archivo MFD	Si marca esta casilla, MapForce guarda las rutas de acceso de los archivos que aparecen en el cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd). Esta opción afecta a los archivos de entrada y salida utilizados por el componente de texto (véase también *Usar rutas de acceso relativas en un componente*).
Codificación de entrada / salida	Permite configurar estos aspectos del archivo de instancia de salida: • Nombre de la codificación • Orden de bytes • El carácter de la marca de orden de byte (marca BOM). Por defecto todos los componentes nuevos tienen la codificación definida en la opción *Codificación predeterminada para componentes nuevos*. Para acceder a esta opción seleccione **Herramientas | Opciones | Generales**.
Carácter de relleno	Esta opción sirve para determinar qué caracteres deben utilizarse para completar o rellenar el resto del campo (fijo) cuando los datos entrantes son más cortos que las respectivas definiciones del campo. Con la opción *Personalizado* puede definir un carácter de relleno propio. Si los datos entrantes ya contiene un carácter de relleno y

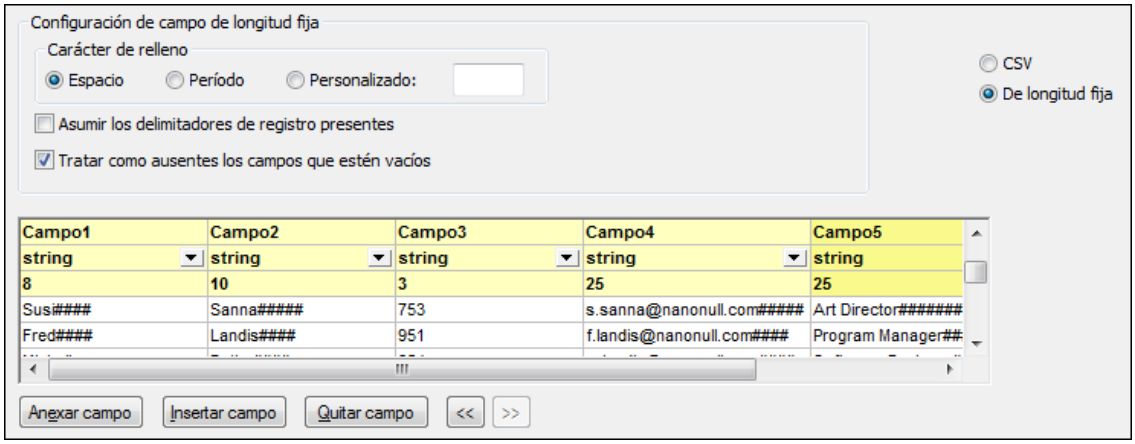
	ese mismo carácter de relleno se introduce en el campo *Personalizado*, entonces los caracteres de relleno se eliminarán en los datos entrantes.
Asumir que delimitadores de registro están presentes	Esta opción puede ser útil si desea leer datos de un archivo plano de origen que no contiene delimitadores de registro como CR/LF o si desea producir un archivo FLF plano de destino sin delimitadores de registros. Consulte el párrafo Más información sobre la opción *Asumir que delimitadores de registro están presentes* que aparece más adelante.
Tratar como ausentes los campos que estén vacíos	Si marca esta casilla, los campos vacíos del archivo de origen no producirán un elemento correspondiente vacío (elemento o atributo) en el archivo de destino. Imaginemos que el destino de la asignación es un archivo XML. Si se desactiva esta opción, los campos vacíos se crearán en el archivo de destino con un valor vacío (p. ej. los elementos `Last`, `Title` y `Email`):  Si se activa esta opción, los campos vacíos no se crearán en el destino de la asignación: 
CSV / De longitud fija	Cambia el tipo de componente por CSV o FLF (campo de longitud fija).
Panel de vista previa	La parte inferior del cuadro de diálogo muestra una vista previa de hasta 20 dilas del archivo seleccionado como entrada/salida. Si lo necesita, puede crear la estructura del archivo (o cambiarla) con estos botones: **Anexar campo** Crea un campo nuevo después del últ **Insertar campo** Crea un campo nuevo inmediatamente registro que está seleccionado. **Quitar campo** Elimina el campo que está seleccion

 © 2018 Altova GmbH

	<< Desplaza el campo que está seleccio izquierda. **>>** Desplaza el campo que está seleccio derecha. Para cambiar el nombre de un campo haga clic en el encabezado (por ejemplo, **Campo1**) y teclee el nuevo valor.  Para cambiar el tipo de datos de un campo seleccione el valor pertinente en la lista desplegable. MapForce revisa el tipo de datos y, si los datos de entrada y el formato del campo son discordantes, resalta los datos en rojo.  Para establecer el tamaño del campo en caracteres basta con introducir el tamaño del campo en la tercera fila de la columna.

Más información sobre la opción *Asumir que delimitadores de registro están presentes*

Veamos un ejemplo para comprender mejor el funcionamiento de esta opción. Abra el archivo **Altova-FLF.txt** que está en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**. Observe que el archivo está compuesto por registros con una longitud de 71 caracteres y que no contiene delimitadores. Si tuviéramos que leer datos de este archivo, primero deberíamos dividirlo en registros. Es decir, crear varios campos cuyo tamaño total sea 71 caracteres y después debemos desactivar la casilla *Asumir que delimitadores de registro están presentes*. Para ver un ejemplo con instrucciones paso a paso consulte el apartado Ejemplo: asignar archivos de texto de longitud fija a bases de datos.



Si necesita escribir datos de este archivo en un archivo de destino que use la misma estructura y marca la casilla *Asumir que delimitadores de registro están presentes*, MapForce creará un registro nuevo cada 71 caracteres.

```
 1  Vernon##Callaby###582v.callaby@nanonull.com###Office Manager##########
 2  Frank###Further###471f.further@nanonull.com###Accounts Receivable######
 3  Loby####Matise####963l.matise@nanonull.com####Accounting Manager#######
 4  Joe#####Firstbread621j.firstbread@nanonull.comMarketing Manager Europe#
 5  Susi####Sanna#####753s.sanna@nanonull.com#####Art Director#############
 6  Fred####Landis####951f.landis@nanonull.com####Program Manager##########
 7  MichelleButler####654m.landis@nanonull.com####Software Engineer########
 8  Ted#####Little####852t.little@nanonull.com####Software Engineer########
 9  Ann#####Way#######951a.way@nanonull.com#######Technical Writer#########
10  Liz#####Gardner###753l.gardner@nanonull.com###Software Engineer########
11  Paul####Smith#####334p.smith@nanonull.com#####Software Engineer########
12  Alex####Martin####778a.martin@nanonull.com####IT Manager###############
13  George##Hammer####223g.hammer@nanonull.com####Web Developer############
14  Jessica#Bander####241j.band@nanonull.com######Support Engineer#########
15  Lui#####King######345l.king@nanonull.com######Support Engineer#########
16  Steve###Meier#####114s.meier@nanonull.com#####Office Manager##########
17  Theo####Bone######331t.bone@nanonull.com######Accounts Receivable######
18  Max#####Nafta#####122m.nafta@nanonull.com#####PR & Marketing Manager US
19  ValentinBass######716v.bass@nanonull.com######IT Manager###############
20  Carl####Franken###147c.franken@nanonull.com###Support Engineer#########
21  Mark####Redgreen##152m.redgreen@nanonull.com##Support Engineer#########
22
```

El resultado de la asignación cuando está activada la opción Asumir que delimitadores de registro están presentes.

Si no marca la casilla *Asumir que delimitadores de registro están presentes*, entonces el resultado de la asignación será una cadena de gran longitud.

© 2018 Altova GmbH

```
1   Vernon##Callaby###582v.callaby@nanonull.com###Office
    Manager###########Frank###Further###471f.further@nanonull.com###Accounts
    Receivable######Loby####Matise####963l.matise@nanonull.com####Accounting
    Manager#######Joe#####Firstbread621j.firstbread@nanonull.comMarketing Manager
    Europe#Susi####Sanna#####753s.sanna@nanonull.com#####Art
    Director#############Fred####Landis####951f.landis@nanonull.com####Program
    Manager##########MichelleButler####654m.landis@nanonull.com####Software
    Engineer########Ted#####Little####852t.little@nanonull.com####Software
    Engineer########Ann#####Way#######951a.way@nanonull.com#######Technical
    Writer#########Liz#####Gardner###753l.gardner@nanonull.com###Software
    Engineer########Paul####Smith#####334p.smith@nanonull.com#####Software
    Engineer########Alex####Martin####778a.martin@nanonull.com####IT
    Manager###############George##Hammer####223g.hammer@nanonull.com####Web
    Developer############Jessica#Bander####241j.band@nanonull.com######Support
    Engineer#########Lui#####King######345l.king@nanonull.com######Support
    Engineer#########Steve###Meier#####114s.meier@nanonull.com#####Office
    Manager###########Theo####Bone######331t.bone@nanonull.com######Accounts
    Receivable######Max#####Nafta#####122m.nafta@nanonull.com#####PR & Marketing Manager
    USValentinBass######716v.bass@nanonull.com######IT
    Manager###############Carl####Franken###147c.franken@nanonull.com###Support
    Engineer#########Mark####Redgreen##152m.redgreen@nanonull.com##Support
    Engineer#########
```

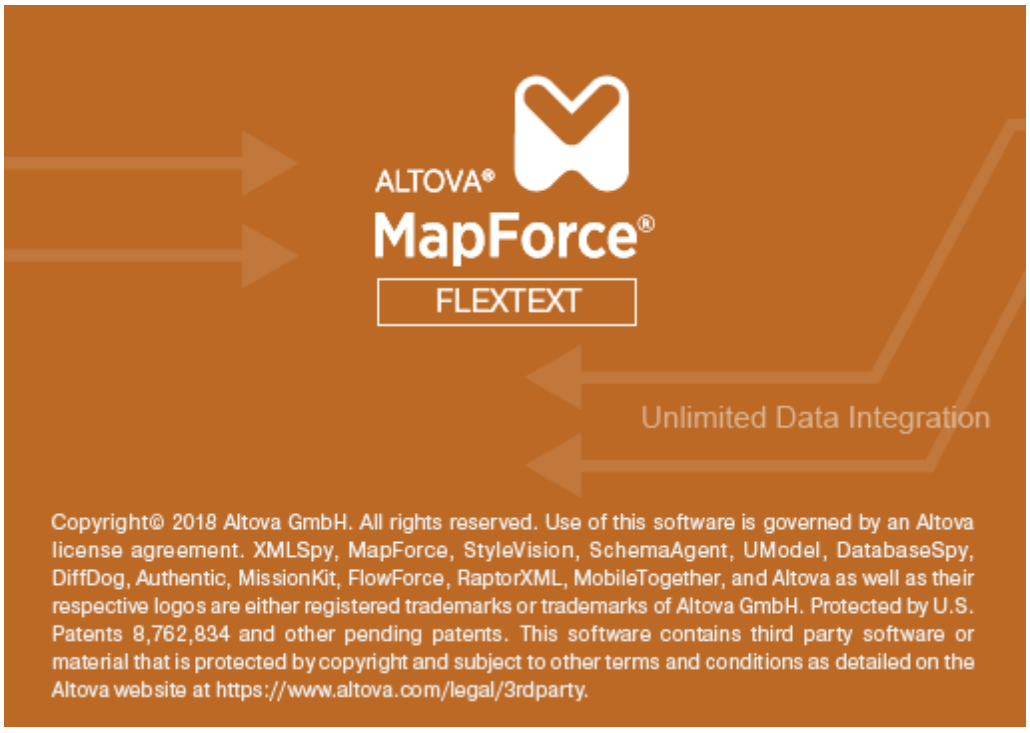
El resultado de la asignación cuando está desactivada la opción Asumir que delimitadores de registro están presentes.

7.4 MapForce FlexText

Sitio web de Altova: Asignación de archivos de texto con FlexText

FlexText es un módulo de MapForce que permite convertir datos de archivos de texto no estándar o heredados y de gran complejidad en otros formatos compatibles con MapForce. Al contrario de los archivos XML los archivos de texto carecen de un esquema o estructura de la cual se puede derivar un esquema, sobre todo cuando tienen una estructura compleja y única que no se ajusta a los patrones CSV o de campo de longitud fija. Además, a veces solo es necesario extraer una sección de datos del archivo de texto. FlexText se encarga de todas estas tareas permitiendo definir y probar de forma visual y en tiempo real las reglas o la plantilla que determinan cómo se dividirán los datos de texto. Las plantillas FlexText definen el modelo estructural de los datos de texto en función de criterios definidos por el usuario con el objetivo de asignar los datos a otros formatos distintos.

Una vez definida, la plantilla FlexText se puede agregar al área de asignación de MapForce en forma de componente de origen o de destino. Esto le permitirá asignar datos de texto complejos y no estándar a cualquier otro formato compatible con MapForce. Los archivos de plantilla FlexText tienen la extensión de archivo .mft (MapForce FlexText). Estas plantillas se pueden reutilizar en tantas asignaciones de datos como desee.

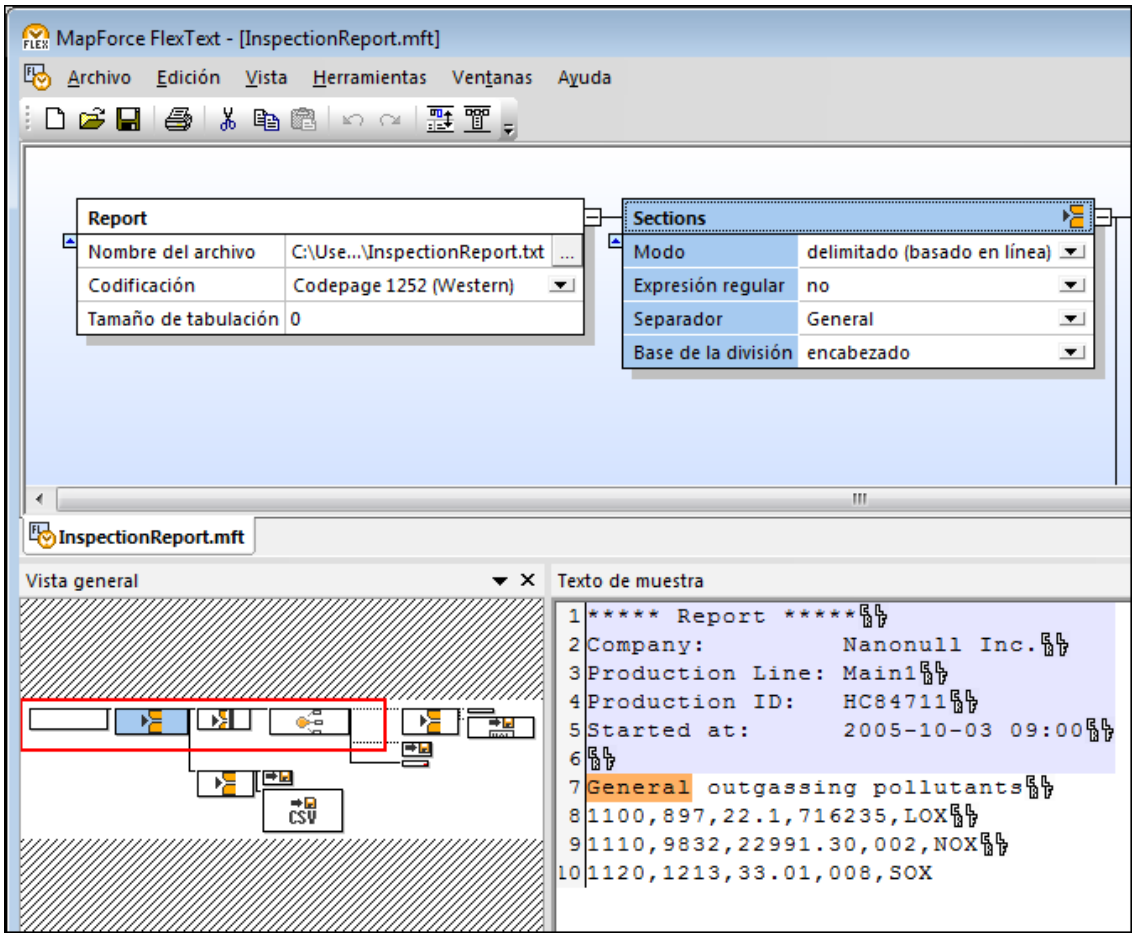


7.4.1 Información general

Puede iniciar FlexText desde MapForce directamente cuando añada archivos de texto como componentes de asignación (con el comando de menú **Insertar | Archivo de texto**). También puede iniciar FlexText como programa independiente, ejecutando el archivo ejecutable *Altova*

MapForce FlexText que encontrará en el directorio de instalación de MapForce.

En la imagen que aparece a continuación puede ver un ejemplo de plantilla FlexText (**InspectionReport.mft**). Encontrará esta y otras plantillas de ejemplo en el directorio **<Documentos>\Altova\MapForce2018\MapForceExamples**.



Interfaz gráfica de FlexText

La interfaz de FlexText está compuesta por varios paneles: Diseño, Vista general y Texto de muestra. Como verá a continuación, podrá personalizar la posición y el aspecto de estos paneles en función de sus preferencias.

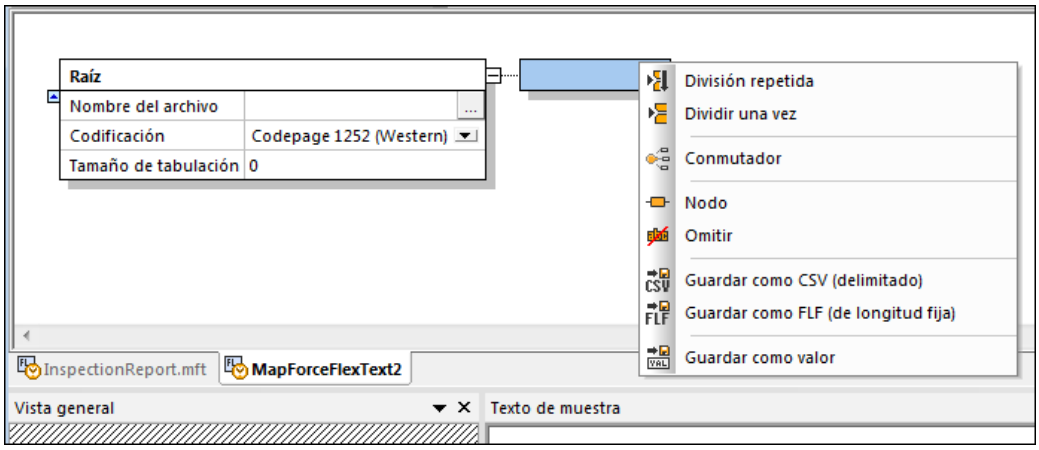
Panel Diseño

Este panel es el área de trabajo donde podrá definir la estructura del texto por medio de contenedores gráficos. Los contenedores definen las reglas que se deben usar para gestionar los datos de texto. El primer contenedor de cualquier plantilla FlexText es el contenedor **Raíz**, que representa todo el archivo de texto. Los demás contenedores se encargan de definir la lógica de procesamiento del resto del texto (normalmente dividiendo el texto en unidades desde las que se podrán asignar datos). Por ejemplo, el contenedor **Dividir una vez** divide un fragmento de texto en dos fragmentos exactamente. El nombre de los contenedores describe por defecto su función (p. ej. **División repetida**). Sin embargo, puede cambiar el nombre del contenedor si lo desea. En FlexText puede definir tantos contenedores como necesite la estructura del archivo de texto

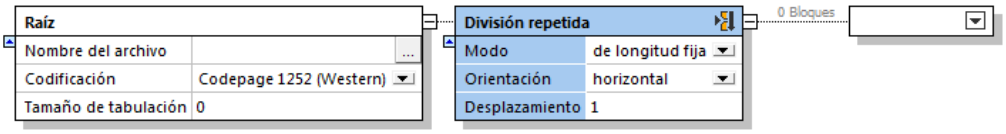
con el que está trabajando.

En la sección Referencia del usuario se describen detalladamente los distintos tipos de contenedor, pero es conveniente detallar ya algunas características de los contenedores:

- Si hace clic en el icono de la esquina superior derecha de un contenedor, aparece un menú donde puede elegir el tipo de contenedor.



- Cada contenedor ofrece varias opciones que indican lo que le ocurre a los datos de texto situados en esa posición exacta de la estructura. Estas opciones determinan el contenido del contenedor y sirven para refinarlo antes de pasar los datos al componente de MapForce.



- Los iconos **Expandir** y **Contraer** sirven para mostrar y ocultar el contenido de los contenedores.
- Para contraer un grupo de contenedores pulse la tecla **Mayús**. Una vez contraídos aparece el icono . Si hace clic en el controlador, la sección situada la derecha del contenedor se contrae.
- Puede obtener una vista previa del texto que genera un contenedor en el panel de diseño directamente. Para ello basta con habilitar el botón **Texto del nodo en vista Diseño** de la barra de herramientas.
- Si habilita el botón **Contraer automáticamente el texto de nodos inactivos** de la barra de herramientas, todo el contenido del contenedor se expande cuando se selecciona. Los demás contenedores se contraen automáticamente.

Panel Vista general

Este panel ofrece una *vista aérea* del panel Diseño. Para navegar por el panel Diseño arrastre el rectángulo rojo por el panel Vista general. Para cambiar la posición del panel Vista general haga clic en su barra de título y arrástrelo hasta la posición deseada.

© 2018 Altova GmbH

Panel Texto de muestra

Este panel muestra el contenido del contenedor que está seleccionado. Recuerde que este panel solo aparece en la interfaz si se habilita el botón **Texto del nodo en vista Diseño**  de la barra de herramientas. (Si no habilita este botón, el contenido del contenedor seleccionado aparece en el panel Diseño directamente.)

Además, a fin de ayudarle a identificar con facilidad los caracteres **Tabulación** y **Retorno de carro/Salto de línea** en el texto de la vista previa, FlexText representa dichos caracteres con los símbolos → y ¶ respectivamente.

7.4.2 Tutorial

Este tutorial explica cómo usar las principales características de FlexText para procesar un archivo de texto y usar el resultado en una asignación de datos de MapForce.

El archivo de ejemplo que utilizaremos es el archivo **Flex-tutorial.txt** (disponible en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**), que tiene este formato:

```
111,332.1,22537.7,5,Container ship,Mega,
111,A1579227,10,3,400,Microtome,
111,B152427,7,6,1200,Miscellaneous,
222,978.4,7563.1,69,Air freight,Mini,
222,ZZAW561,10,5,10000,Gas Chromatograph,,

General outgassing pollutants
1100,897,22.1,716235,LOX
1110,9832,22991.30,002,NOX
1120,1213,33.01,008,SOX
```

Objetivo del tutorial:

- separar los registros que contengan la clave 111 y 222 de los demás registros y guardarlos en elementos asignables.
- descartar los demás registros.
- crear un archivo CSV con los registros sobrantes.

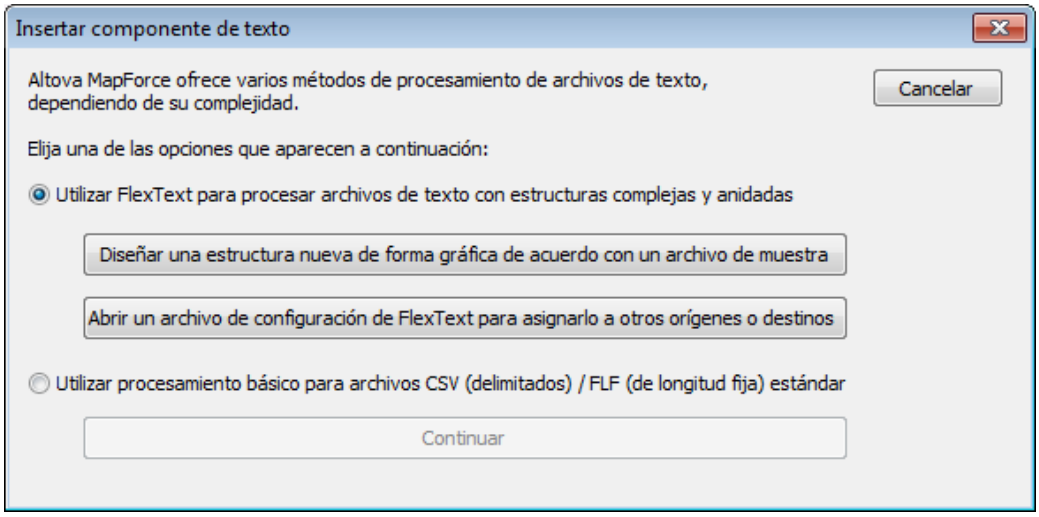
El tutorial se divide en varios apartados:

- Paso nº1: crear la plantilla FlexText
- Paso nº2: definir condiciones de división
- Paso nº3: definir varias condiciones por contenedor
- Paso nº4: crear el componente de destino de MapForce
- Paso nº5: usar plantillas FlexText en MapForce

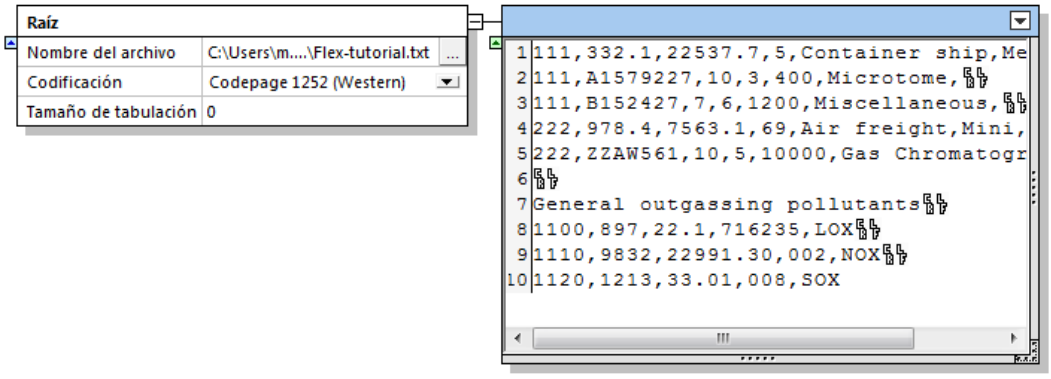
7.4.2.1 Paso nº1: crear la plantilla FlexText

Para crear la plantilla FlexText:

1. Inicie MapForce y abra un archivo de asignación nuevo.
2. Seleccione **Insertar | Archivo de texto** o haga clic en el icono **Insertar archivo de texto** .
3. Haga clic en el botón **Diseñar una estructura nueva de forma gráfica...** en el cuadro de diálogo.



4. Introduzca un nombre para su plantilla FlexText (p.ej. `Flex-tutorial.mft`) y haga clic en **Guardar** para continuar.
   Aparece un diseño vacío en FlexText y el cuadro de diálogo "Abrir".
5. Seleccione el archivo de texto **Flex-tutorial.txt** de la carpeta **...\MapForceExamples\Tutorial** y haga clic en **Abrir**.

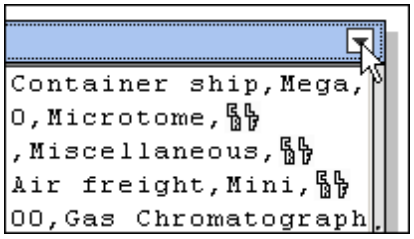


Ahora puede verse el contenido del archivo de texto.

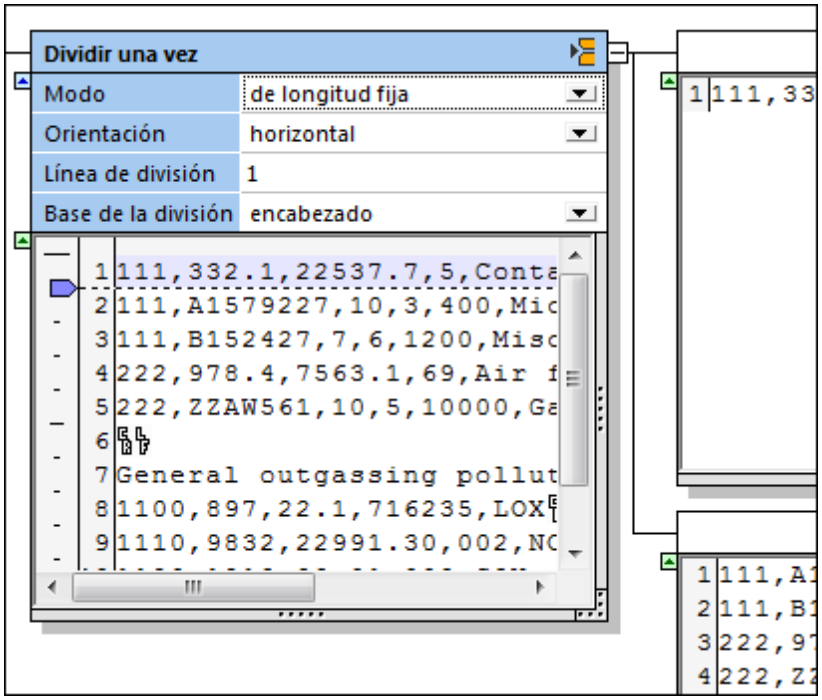
Al desactivar el icono **Texto del nodo en vista Diseño** aparece el contenido del contenedor activo en el panel Texto de muestra.

Al activar el icono **Contraer automáticamente el texto de nodos inactivos** se muestra el contenido del contenedor activo pero los demás contenedores se contraen.

 © 2018 Altova GmbH



6. Haga clic en el icono situado en la esquina superior derecha del contenedor y seleccione **Dividir una vez** en el menú contextual.
   Aparecen dos contenedores nuevos junto al contenedor llamado **Dividir una vez** (para más información consulte la sección *Dividir una vez* de la documentación).



   El contenedor **Dividir una vez** aparece con su configuración predeterminada: `longitud fija`, `orientación horizontal` y `línea de división=1`.

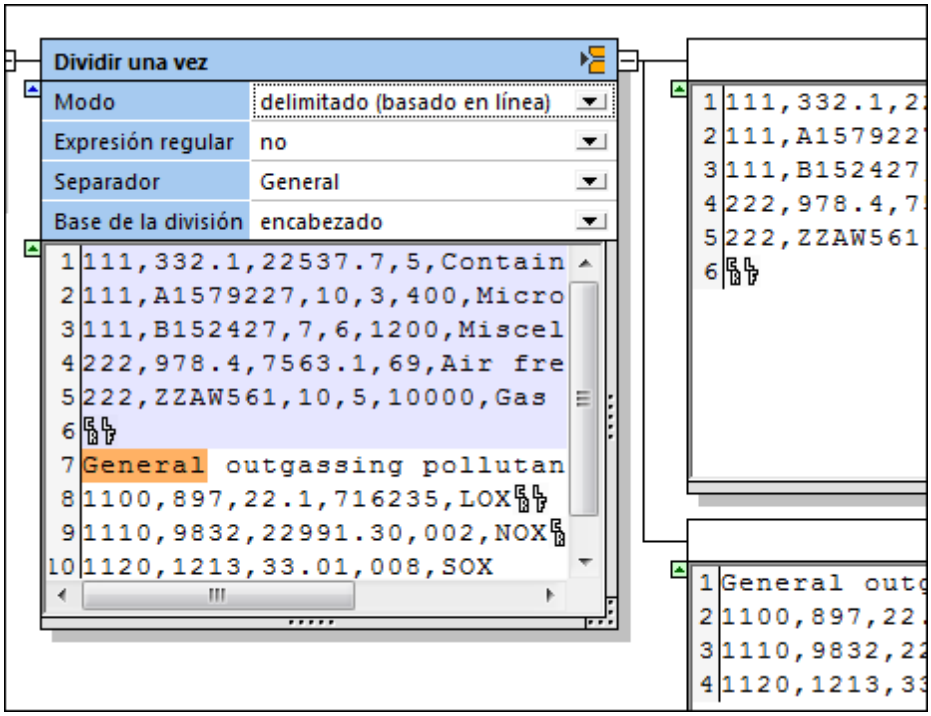
Como resultado de esta configuración:

- el contenedor de la parte superior incluye la primera línea del archivo de texto (que está resaltada en el contenedor **Dividir una vez**).
- el contenedor de la parte inferior contiene el resto del archivo de texto.

7.4.2.2 Paso nº2: definir condiciones de división

Las denominadas *condiciones de división* permiten segmentar fragmentos de texto de varias formas. Para definir condiciones de división:

1. Haga clic en el cuadro combinado *Modo* y seleccione la opción `delimitado (basado en línea)`.
2. Haga doble clic en el campo *Separador* e introduzca `General`.



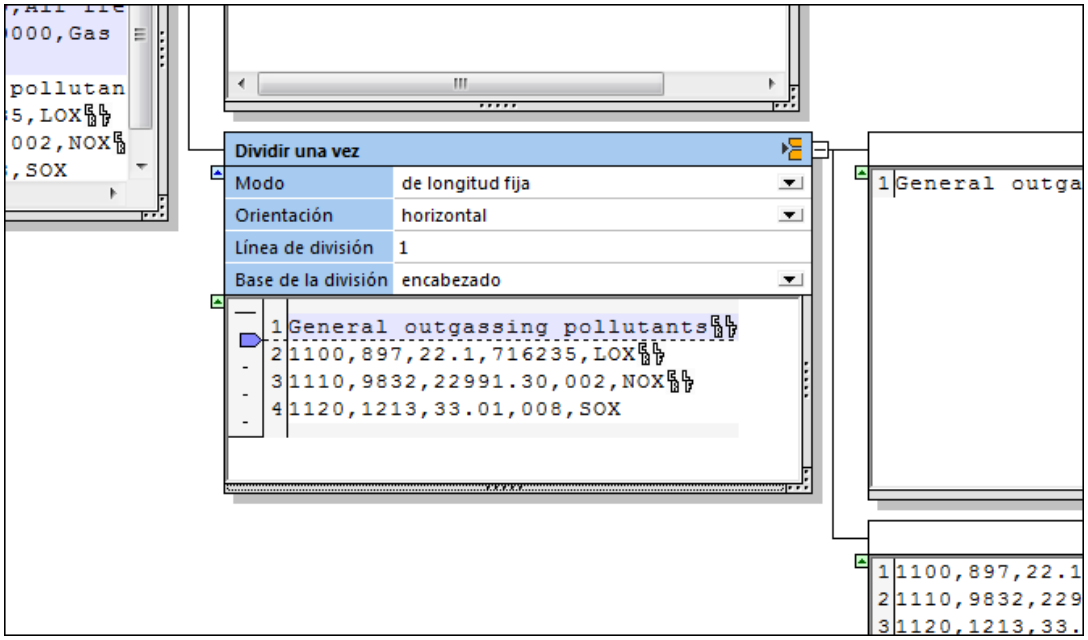
Observe cómo cambian los fragmentos de texto de los contenedores.

Usando el separador `General` y el modo `delimitado (basado en línea)` separamos la sección del texto que contiene la cadena `General` y lo pasamos al contenedor de la parte inferior. El texto que aparece antes del separador `General` se coloca en el contenedor de la parte superior.

A continuación prepararemos el contenedor inferior para producir un archivo CSV que contenga los registros a partir de `1100`.

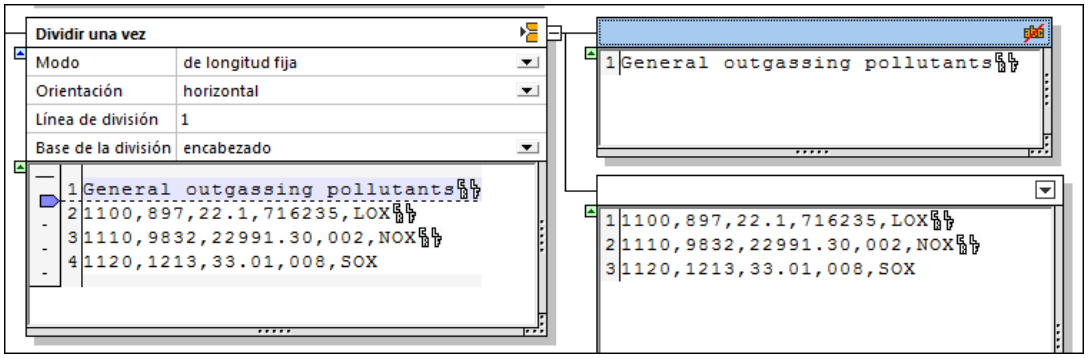
3. Haga clic en el icono interactivo del contenedor inferior y seleccione **Dividir una vez**.

© 2018 Altova GmbH



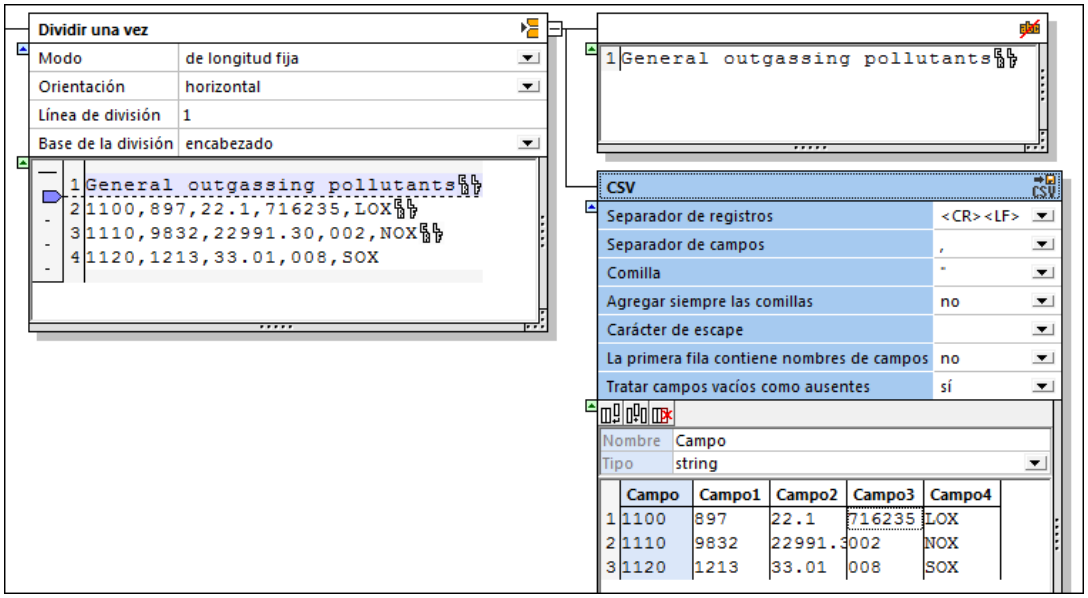
La configuración predeterminada de este nuevo contenedor **Dividir una vez** crea dos contenedores más a su derecha y separa la primera línea del fragmento de texto del resto. Nuestro objetivo es omitir esa primera línea y guardar el resto en un archivo CSV.

4. Haga clic en el icono interactivo del contenedor superior y seleccione **Omitir**.



El fragmento de texto de este contenedor no estará disponible en el componente de la asignación de MapForce.

5. Haga clic en el icono interactivo del contenedor inferior y seleccione la opción **Guardar como CSV**.



Ahora el contenedor muestra el fragmento de texto en forma de tabla y podemos usar su configuración predeterminada.

Configurar el archivo CSV:
Si quiere cambiar el nombre de los campos, haga clic en el campo de la tabla y escriba el nombre nuevo en el campo *Nombre*. También puede anexar, insertar o eliminar columnas en el contenedor. Para más información consulte el apartado Guardar como CSV.

En el siguiente apartado del tutorial continuaremos definiendo el fragmento de texto sobrante.

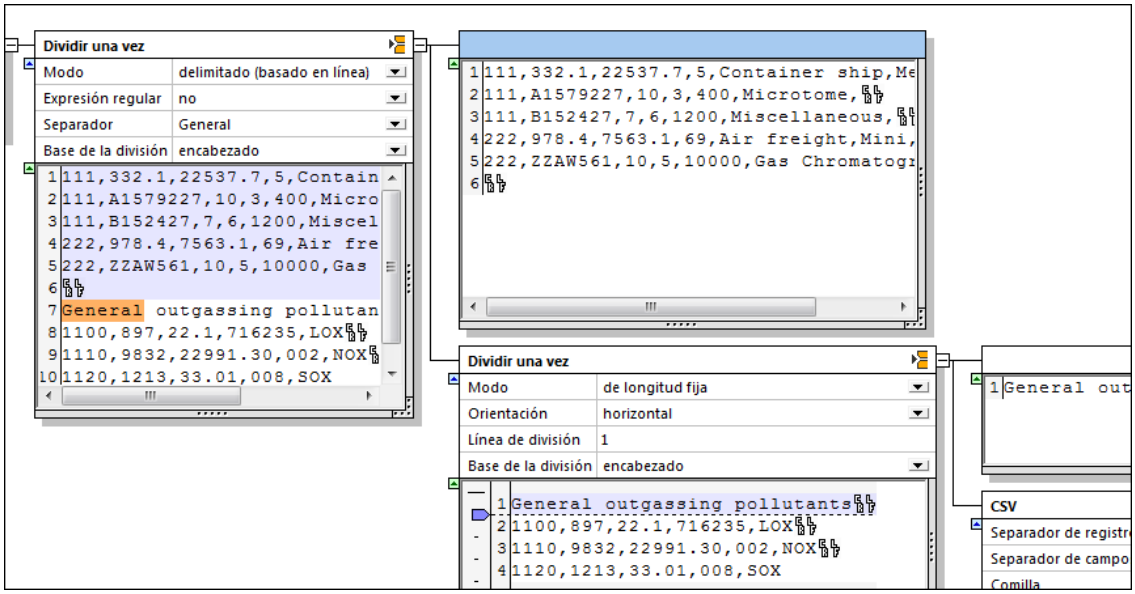
7.4.2.3 Paso nº3: definir varias condiciones por contenedor

En FlexText puede definir varias condiciones por cada fragmento de texto, usando el contenedor **Conmutador**. A cada condición definida se le asigna automáticamente un contenedor asociado.

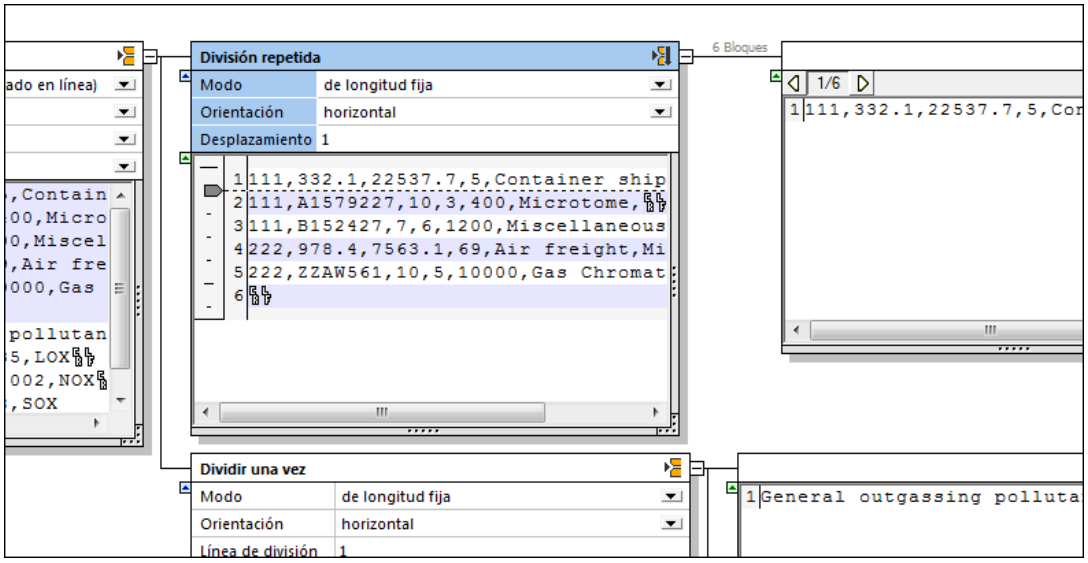
Por ahora hemos definido el fragmento de texto del contenedor inferior (del contenedor **Dividir una vez**):

- el contenedor **Dividir una vez** separa la primera línea y la coloca en un contenedor **Omitir**.
- el segmento sobrante se define/se guarda como archivo CSV (contenedor **CSV**)

© 2018 Altova GmbH

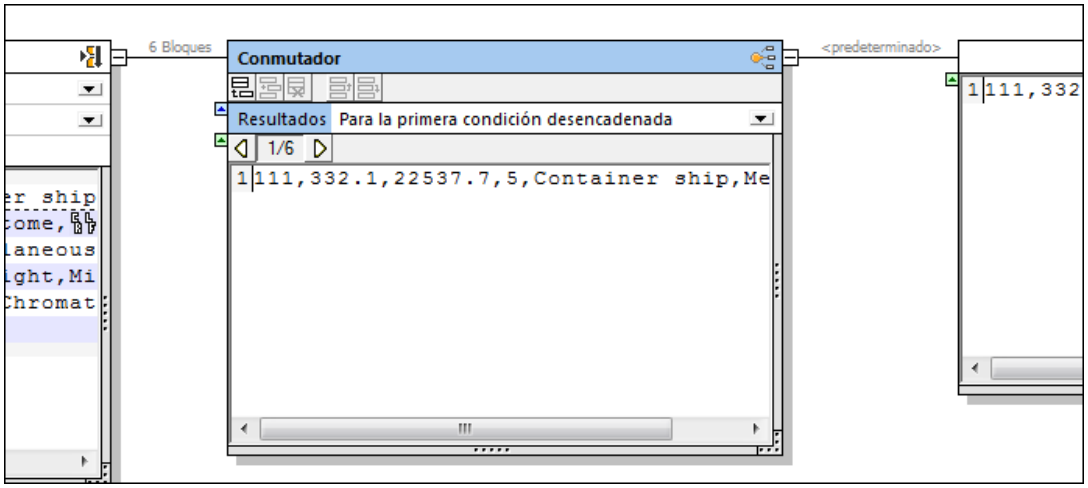


1. Haga clic en el icono del contenedor superior (el que todavía no tiene nombre) y seleccione la opción **División repetida**.



Para este ejemplo usaremos la configuración predeterminada de este nuevo contenedor **División repetida**. El fragmento de texto se divide en varios bloques de texto, cada uno formado por una sola línea. El contenedor asociado muestra una vista previa de cada bloque de texto. Haga clic en el icono **Siguiente bloque de texto** para ver los fragmentos uno por uno (6 en total).

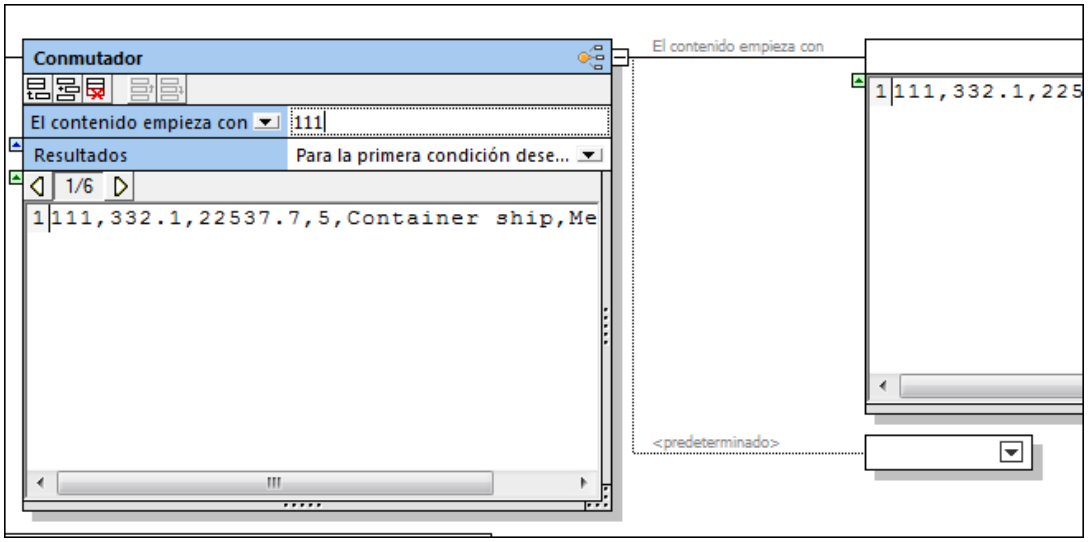
2. Haga clic en el icono de este nuevo contenedor y seleccione la opción **Conmutador**.



El estado inicial del contenedor **Conmutador** es:

- se crea a su derecha un contenedor asociado *predeterminado.*
- el contenido del primer registro (1/6) aparece en el contenedor *predeterminado* situado a la derecha.

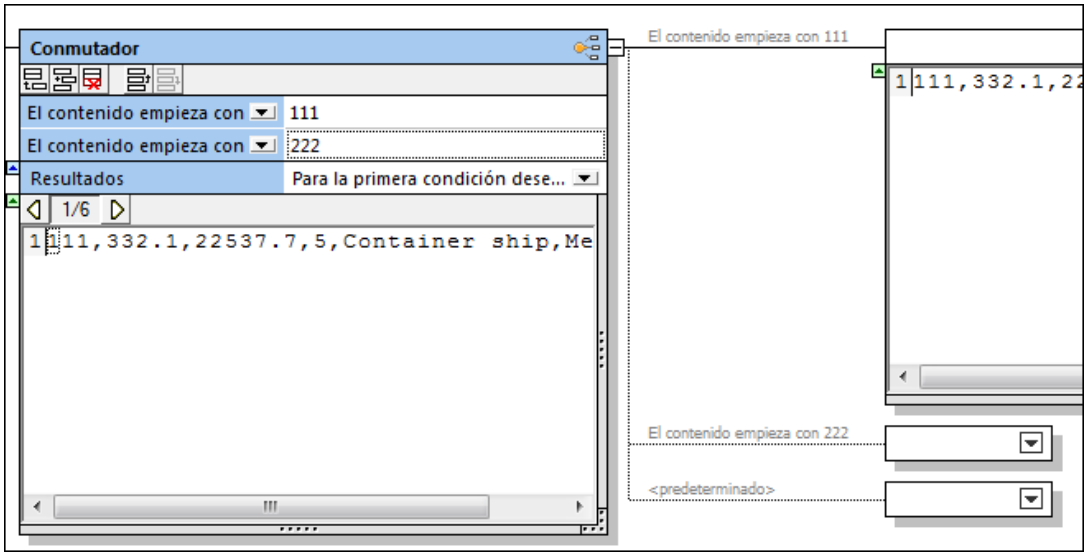
3. Haga clic en el icono **Anexar condición** de la barra de título del contenedor **Conmutador** para añadir una condición nueva.
4. Haga doble clic en el campo *El contenido empieza con* y escriba el número 111.



Esto define la primera condición y se añade un contenedor asociado (*El contenido empieza con 111*) encima del contenedor *predeterminado*.

5. Haga clic otra vez en el icono **Anexar condición** y escriba 222 en el campo *El contenido empieza con*.

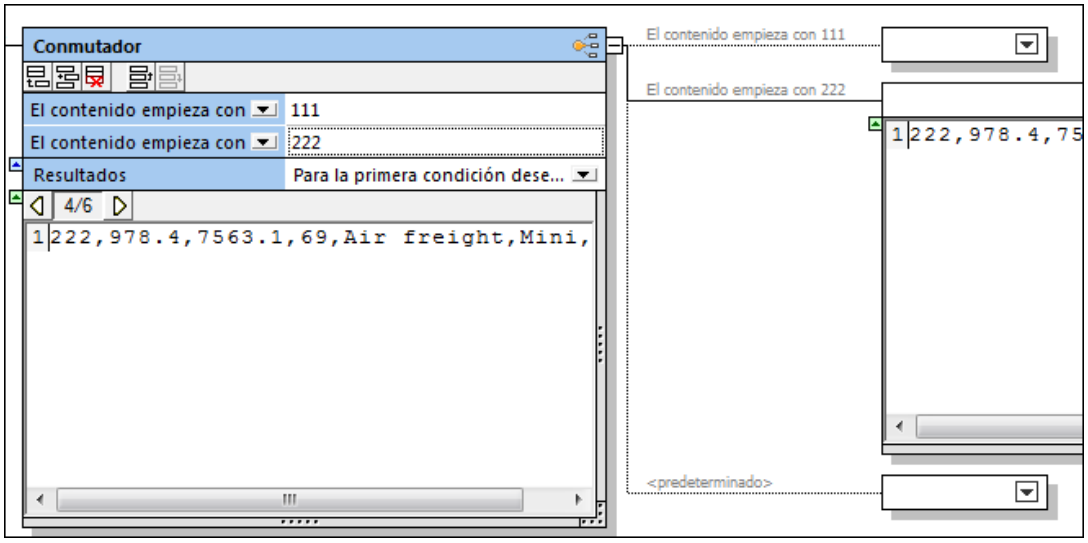
© 2018 Altova GmbH



Se añade un tercer contenedor (*El contenido empieza con 222*).

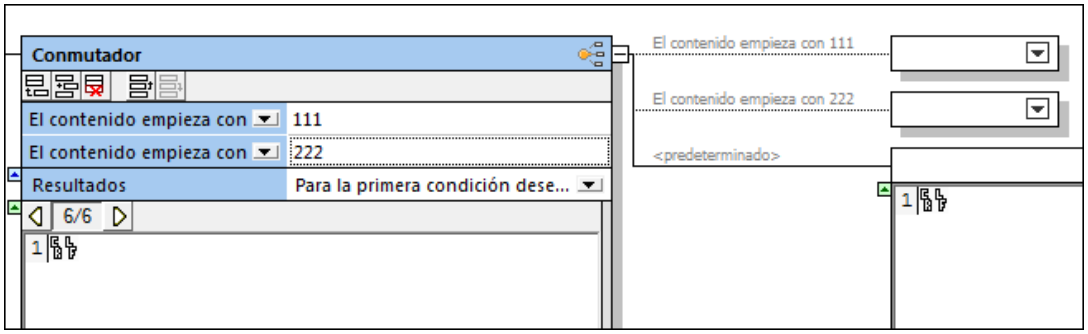
Nota: si hace clic en el cuadro combinado *El contenido empieza con* también puede seleccionar la opción *Contiene* para especificar una cadena que puede aparecer en cualquier parte del fragmento de texto.

6. Haga clic varias veces en el icono **Siguiente bloque de texto** del contenedor **Conmutador** para ver el efecto.



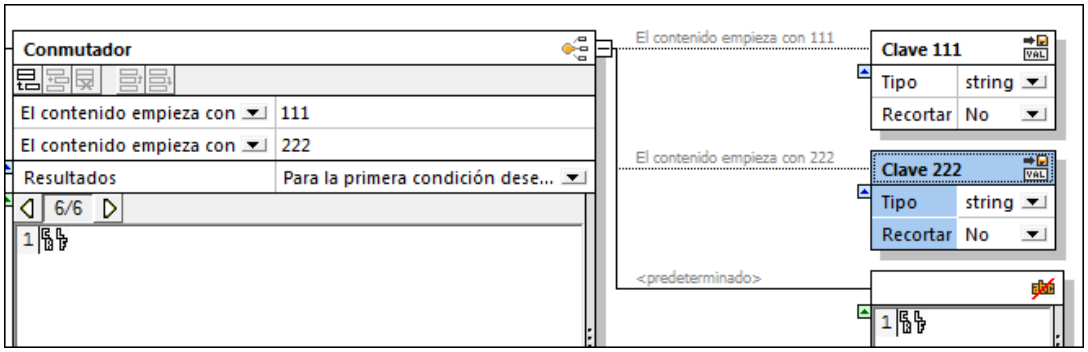
Observe que al llegar al registro 4 de 6 se abre el contenedor *222*.

7. Siga haciendo clic hasta llegar al registro 6 de 6. En el contenedor *predeterminado* aparece solamente el carácter CR / LF.



Por tanto, si un fragmento de datos del bloque actual cumple una condición, los datos enteros del bloque se pasan al contenedor asociado. Los datos no se dividen, simplemente se envían al contenedor asociado. Si no se cumple ninguna condición, los datos se envían al contenedor *predeterminado*.

8. Haga clic en el icono de los dos contenedores 111 y 222 y seleccione la opción **Guardar como valor**.
9. Haga doble clic en el título de estos contenedores y llámelos **Clave 111** y **Clave 222** (para reconocerlos más fácilmente).



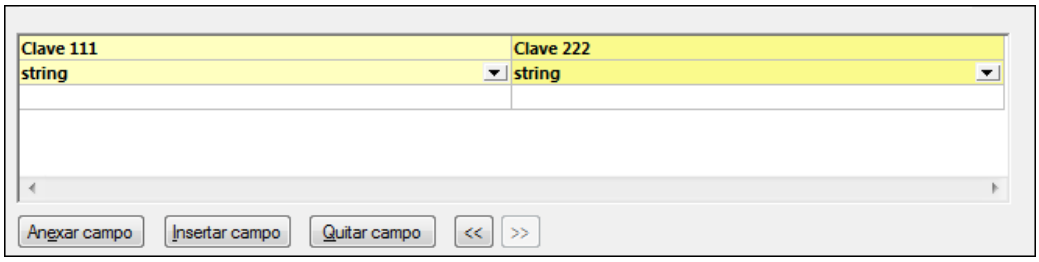
10. Guarde la plantilla FlexText (p. ej. `Flex-Tutorial.mft`).

7.4.2.4 Paso nº4: crear el componente de destino de MapForce

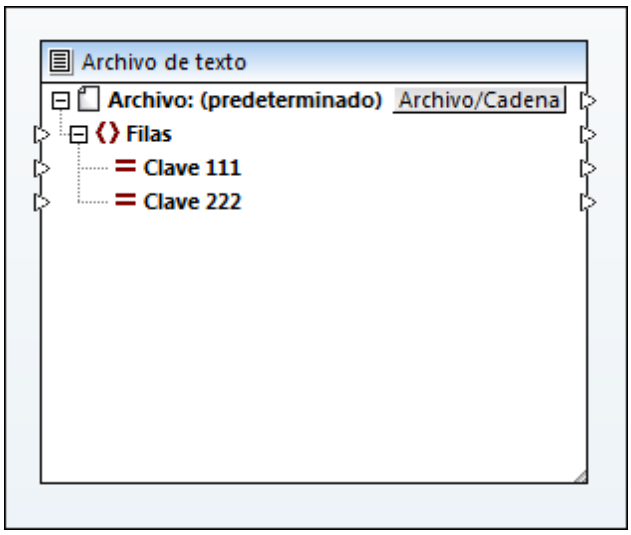
Para crear componentes de destino para el componente FlexText de origen:

1. En MapForce seleccione **Insertar | Archivo de texto** o haga clic en el icono **Insertar texto** .
2. Haga clic en el botón de opción *Utilizar procesamiento básico...* y haga clic en **Continuar**.
   Esto abre el cuadro de diálogo "Configuración del componente".
3. Haga clic en el botón **Anexar campo** para añadir un campo nuevo.
4. Haga doble clic en el nombre del campo **Campo1** y cámbielo por **Clave 111**.

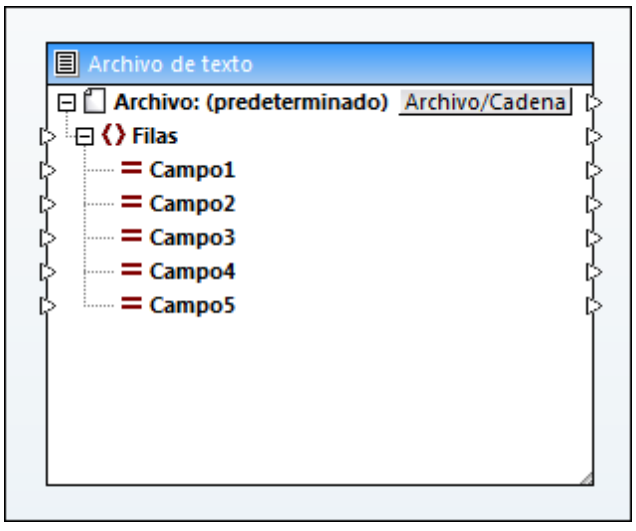
© 2018 Altova GmbH



5. Haga lo mismo en el otro campo y póngale el nombre **Clave 222**. Para terminar haga clic en **Aceptar**.
   Se crea un componente de texto con dos campos.



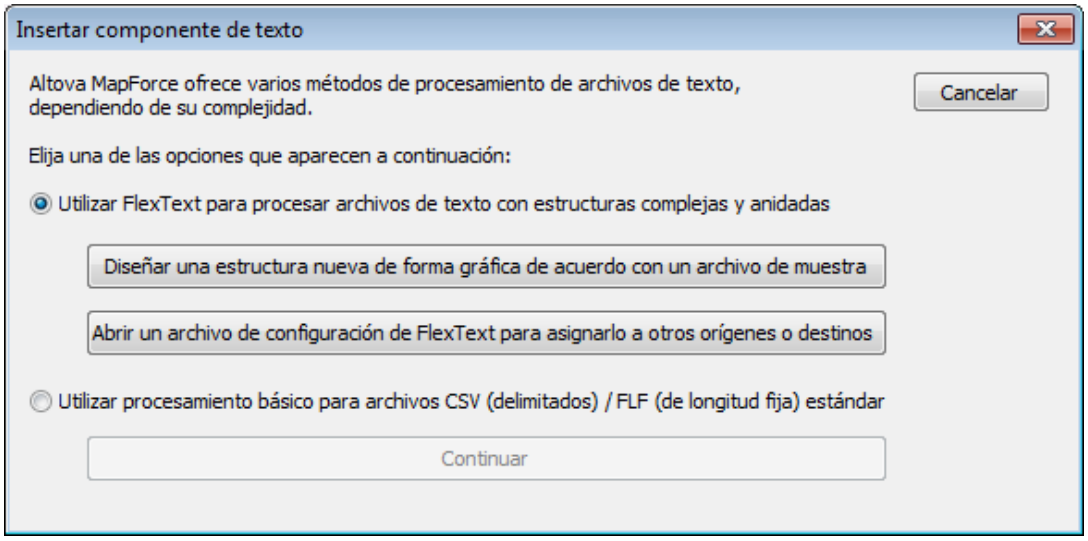
6. Utilice el mismo método para crear otro componente de texto que tenga cinco campos llamados **Campo1**, **Campo2**, etc.



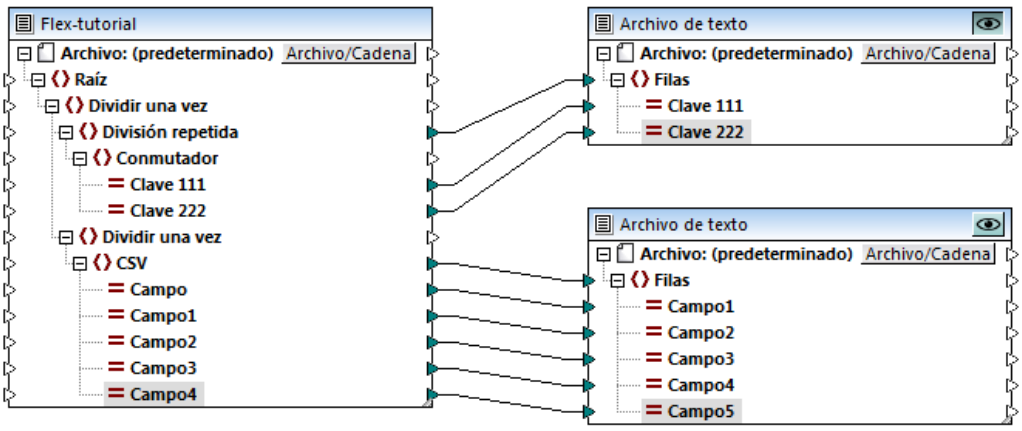
7.4.2.5 Paso nº5: usar plantillas FlexText en MapForce

Para usar la plantilla FlexText en MapForce:

1. En MapForce seleccione **Insertar | Archivo de texto**.



2. Haga clic en el botón **Abrir un archivo de configuración de FlexText...** y seleccione la plantilla FlexText definida previamente (p. ej. `Flex-tutorial.mft`). La estructura del componente de MapForce refleja la estructura de los contenedores que diseñamos en FlexText.



3. Cree asignaciones de datos entre la plantilla FlexText y los componentes de destino definidos en los pasos anteriores. Después abra el panel **Resultados** para ver los resultados de la asignación.

Vista previa de la asignación del componente de texto superior:

 © 2018 Altova GmbH

```
1   "111,332.1,22537.7,5,Container ship,Mega,
2   ",
3   "111,A1579227,10,3,400,Microtome,
4   ",
5   "111,B152427,7,6,1200,Miscellaneous,
6   ",
7   ,"222,978.4,7563.1,69,Air freight,Mini,
8   "
9   ,"222,ZZAW561,10,5,10000,Gas Chromatograph,,
10  "
11  ,
12
```

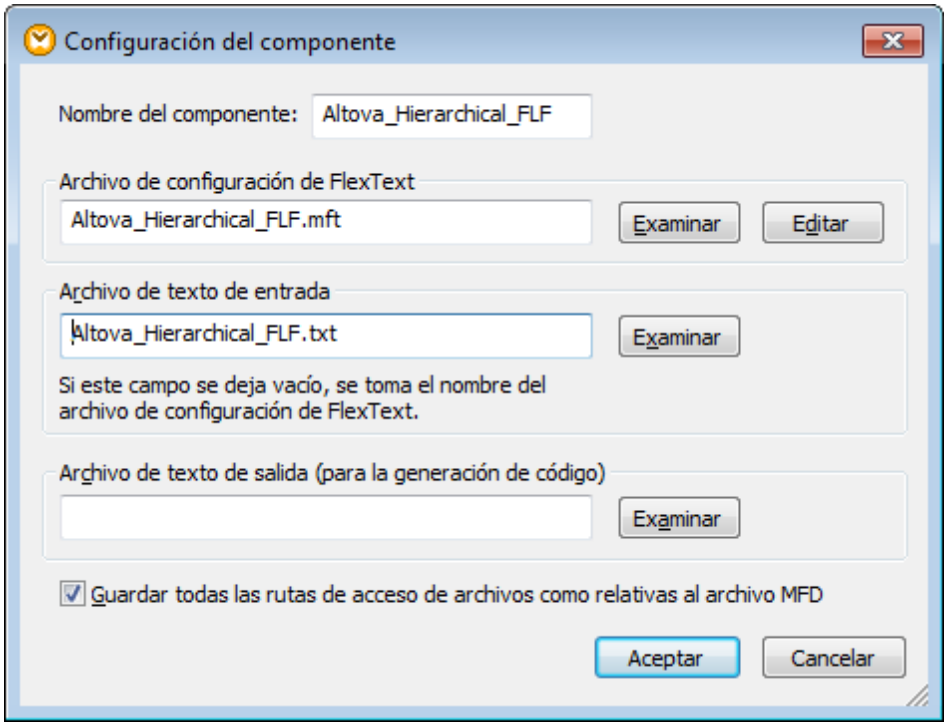
Vista previa del componente de texto inferior:

```
1   1100,897,22.1,716235,LOX
2   1110,9832,22991.30,002,NOX
3   1120,1213,33.01,008,SOX
4
```

7.4.3 Configuración de componentes FlexText

Tras añadir el componente FlexText al área de asignación podrá personalizar su configuración desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de varias maneras:

- Haciendo clic en el comando de menú **Componente | Propiedades** (este comando de menú se habilita cuando está seleccionado un componente).
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y seleccionando **Propiedades** en el menú contextual.



Cuadro de diálogo "Configuración del componente" de un componente FlexText

A continuación describimos las opciones de configuración que ofrece el cuadro de diálogo.

Nombre del componente	El nombre del componente se genera automáticamente cuando se crea, pero se puede cambiar en cualquier momento. Si el nombre del componente se generó automáticamente y selecciona un archivo de instancia después, MapForce le preguntará si quiere actualizar el nombre del componente. El nombre del componente puede incluir espacios y puntos. No puede contener barras diagonales directas ni inversas, dos puntos, comillas dobles ni espacios iniciales ni finales. Cuando cambie el nombre del componente debe tener en cuenta que: • si tiene pensado implementar la asignación en FlowForce Server, el nombre del componente debe ser único. • lo más recomendable es usar caracteres que puedan introducirse en la línea de comandos. La codificación de algunos caracteres nacionales puede ser distinta en Windows y en la línea de comandos.
Archivo de configuración de FlexText	Especifica el nombre o ruta de acceso de la plantilla FlexText (.mft) que utiliza el componente.

© 2018 Altova GmbH

	Para seleccionar una plantilla nueva haga clic en **Examinar**. Para editar la plantilla en FlexText haga clic en **Editar**.
Archivo de texto de entrada	Especifica el archivo de texto del que MapForce leerá los datos. Este campo es relevante para los componentes de origen y se rellena cuando se crea el componente y se le asigna un archivo de texto. El archivo de texto de origen que se especifique en este campo tiene prioridad sobre el que se definió en la plantilla FlexText. Si deja este campo vacío, se usará como entrada el nombre de archivo que está definido en la plantilla FlexText.
Archivo de texto de salida (para la generación de código)	Especifica el nombre o ruta de acceso del archivo de texto en el que MapForce escribirá los datos. Este campo es relevante para los componentes de destino a la hora de generar código. Si introduce una ruta de acceso completa, podrá definir el directorio de destino de manera concreta (p. ej. **c:\misarchivos\secuencia.txt**).
Guardar todas las rutas de acceso de archivos como relativas al archivo MFD	Si marca esta casilla, MapForce guarda las rutas de acceso de archivo del cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd).

7.4.4 Usar FlexText como componente de destino

La ventaja principal de los componentes FlexText es que dividen archivos de texto en fragmentos y permiten asignar elementos a otros componentes de destino. Las plantillas FlexText también se pueden usar como componentes de destino y así reunir varios archivos en uno solo. No obstante, este último uso es algo más complicado y requiere muchas pruebas hasta alcanzar el resultado deseado.

Al usar un componente FlexText como componente de **destino**, se revierten las operaciones previamente definidas en él. En lugar de dividir el archivo en varias secciones, se reconstituye el archivo.

Es decir, por lo general se lleva a cabo (de abajo a arriba) la operación inversa a las operaciones definidas en la plantilla FlexText cuando esta se utiliza como componente de destino:

- una *división* pasa a ser una *combinación* (p. ej. una asignación a una división repetida delimitada por comas pasa a ser una combinación de elementos separados por comas).
- una operación **Guardar** pasa a ser una operación **Cargar**.
- y un *conmutador* pasa a ser una operación "elegir primera coincidencia".

Cuando use plantillas FlexText como componentes de destino en MapForce deberá tener en cuenta que:

- Una vez creada la conexión entre un componente de origen y uno de los elementos de **entrada** de un componente FlexText, el origen de datos del componente FlexText se omite. Los datos suministrados por el componente de origen recién asignado tiene prioridad sobre el origen de

datos original.

- Si se asigna texto a un contenedor **Guardar como CSV/FLF**, el separador se conserva. No obstante, el texto se puede truncar si se produce una división de longitud fija en un nodo situado encima del nodo **Guardar como CSV/FLF**.
- Las divisiones de ancho fijo truncan la sección izquierda/superior si *Base de la división*=`cabeza` o truncan la sección derecha/inferior si *Base de la división*=`cola` a la longitud predefinida. La sección truncada tiene la longitud definida en caracteres. Si el texto es demasiado corto, se insertan caracteres de espacio para rellenar la sección.
- FlexText suele insertar separadores (o espacios en blanco para divisiones fijas) entre los elementos de una operación de división, pero no en las divisiones de tipo `delimitado (basado en línea)`. La operación `delimitado (basado en línea)` no es una operación totalmente reversible. El texto *delimitado* puede aparecer en cualquier parte de la primera línea y se incluye en el texto. Por tanto, su inserción automática no es 100% segura.
  - `Delimitado (basado en línea):` no se añade un separador a la primera línea si no existe.
  - `Delimitado (flotante):` añade un separador entre las dos secciones.
- La operación **Conmutador** no se puede invertir de manera significativa excepto en los casos más básicos. El conmutador busca en sus ramas la primera rama que contiene datos y utiliza/inserta estos datos. Solamente se asigna la primera conexión de una operación conmutador. Para transferir datos a los demás contenedores del conmutador, es necesario definir filtros para los demás conectores. Además es necesario duplicar el elemento primario conmutador para que cada elemento conmutador devuelva un solo elemento, que después se suministra a un elemento de división repetida para combinarlos todos.
- Al crear una asignación a un secundario de un contenedor de división se descartan todos los resultados de la asignación excepto el último elemento. Solamente se conserva un resultado, incluso si se generaron varios resultados.

Cuando se utiliza un componente FlexText como componente de destino su comportamiento es similar al de algunos componentes de XML Schema:

- una **división repetida** es similar a un elemento repetible.
- una **división única** da lugar a un modelo de contenido `sequence`.
- un **conmutador** da lugar a un modelo de contenido `choice`, donde cada caso es un elemento secundario posible.
- una operación **guardar** crea un elemento de tipo simple.

7.4.5 Referencia del usuario

En esta sección se describen con más detalle las características y funciones de FlexText, explicando cómo usarlas para conseguir determinados resultados.

7.4.5.1 División repetida

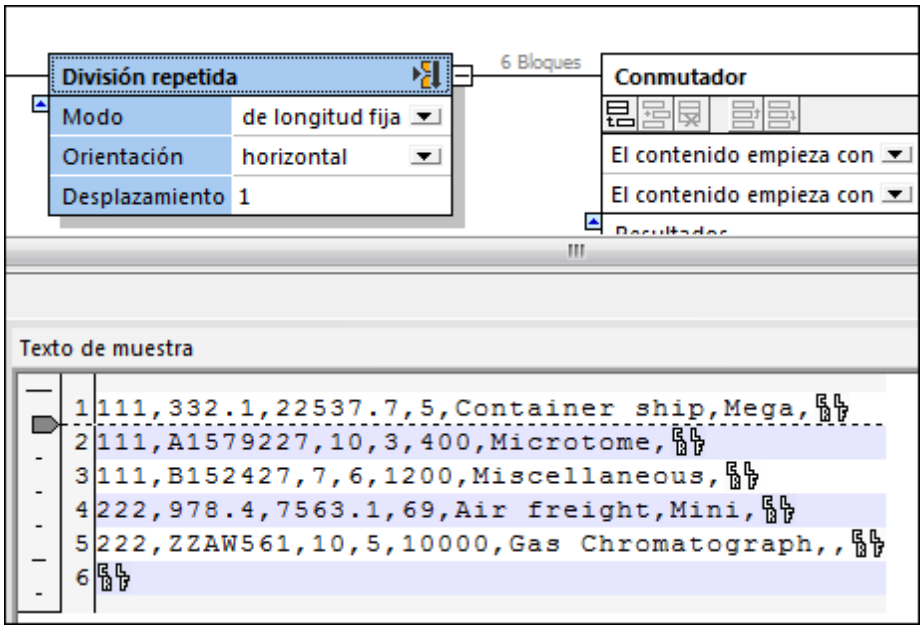


Con esta opción de división se crea un contenedor cuyo contenido es el texto definido por la condición elegida. La opción **División repetida** tiene varios modos: longitud fija, delimitado flotante, delimitado basado en línea y delimitado (la línea empieza con).

Cuando se selecciona la opción **División repetida** por primera vez, se definen automáticamente parámetros predeterminados y los fragmentos resultantes aparecen en el contenedor asociado. Observe que el contenedor **División repetida** está activo en ese momento y que el panel Texto de muestra ofrece una vista previa de todas las líneas (o de todos los registros).

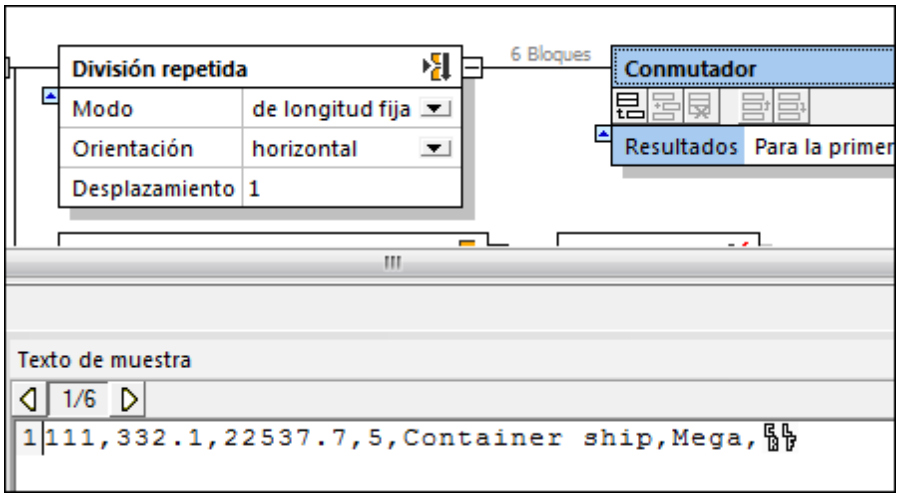
Esta es la **configuración predeterminada** del contenedor **División repetida**:

Modo	`longitud fija`
Orientación	`horizontal`
Desplazamiento	`1`



Resultado predeterminado

Cada línea de texto aparece como una línea/un registro en el contenedor nuevo (porque el desplazamiento es 1). Haga clic en el contenedor nuevo para ver una vista previa de su contenido. Con las flechas de desplazamiento del panel Texto de muestra puede recorrer los 11 bloques/ fragmentos generados con esta configuración.



7.4.5.1.1 Modo: longitud fija

Use el **modo de longitud fija** para la división repetida cuando quiera dividir texto en varios fragmentos horizontales o verticales de longitud fija. A continuación describimos las opciones de configuración para el modo de longitud fija.

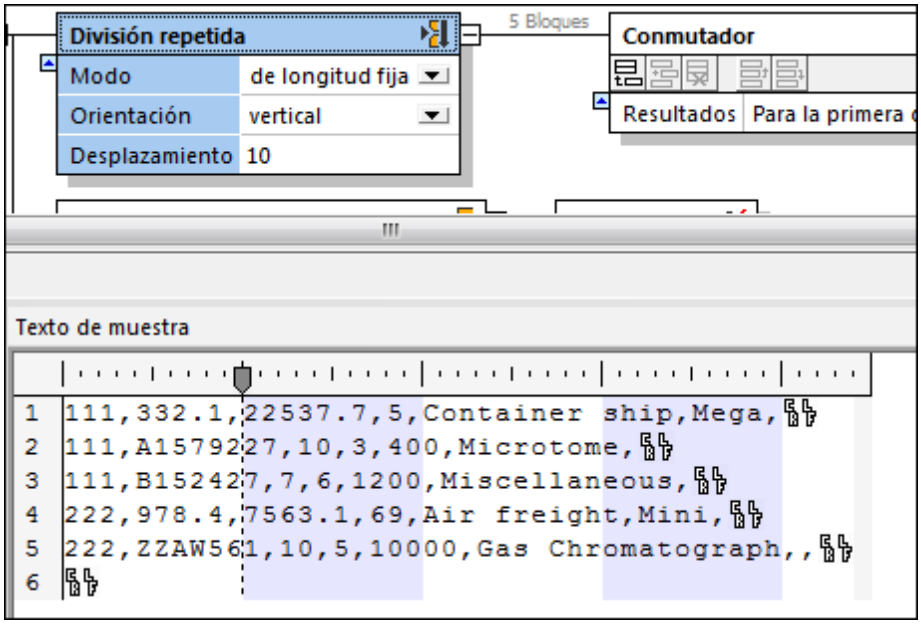
Orientación

Sirve para especificar si el fragmento de texto se debe dividir en horizontal o vertical. Elija la opción `horizontal` para dividir el fragmento en varias secciones horizontales. Elija `vertical` para dividirlo en varias columnas verticales. La orientación predeterminada es `horizontal`.

Desplazamiento

Si la orientación elegida es horizontal, esta opción especifica el número de líneas que debe contener cada fragmento. Si la orientación es vertical, esta opción especifica el ancho en caracteres de cada fragmento. El desplazamiento predeterminado es 1. Hay dos maneras de cambiar el valor de desplazamiento:

- introduciendo un valor en el campo *Desplazamiento* directamente o
- arrastrando la tabulación de la regla horizontal/vertical.



7.4.5.1.2 Modo: delimitado (flotante)

Use el **modo delimitado (flotante)** para la división repetida cuando quiera:

© 2018 Altova GmbH

- dividir texto donde los caracteres separadores especificados deben eliminarse de los fragmentos resultantes y
- dividir texto donde los separadores estén insertados (p. ej. cuando el texto no contenga caracteres CR/LF).

Nota: se entiende como *fragmento* el texto comprendido entre el primer carácter que aparece después del separador y el último carácter que aparece antes de la siguiente instancia del mismo separador. La única excepción es cuando se trata del primer y último fragmento (*ver más abajo*).

A continuación describimos las opciones de configuración para el modo delimitado (flotante).

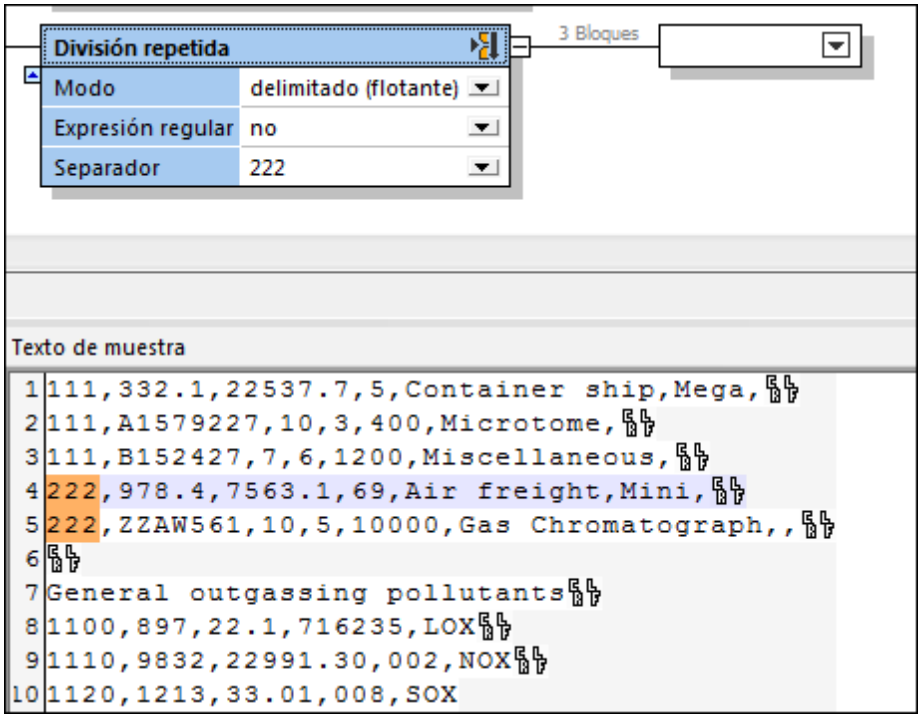
Expresión regular

Se trata de una configuración opcional que sirve para dividir el texto en fragmentos cada vez que una expresión regular encuentre coincidencias (*véase* *Dividir texto con expresiones regulares*). El valor predeterminado es `no`.

Separador

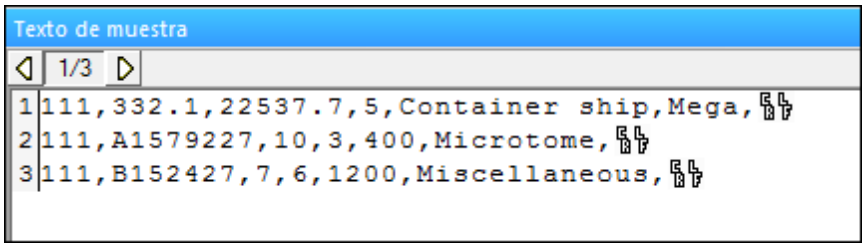
Especifica qué carácter o caracteres deben utilizarse como separador. El valor predeterminado es `ninguno`.

Por ejemplo, si se usa el separador `222` en el texto del ejemplo siguiente, se consigue dividir el texto en tres fragmentos.



El primer fragmento contiene todos los caracteres comprendidos entre el principio del fragmento y el principio del primer separador (`222`). Es decir, el primer fragmento va

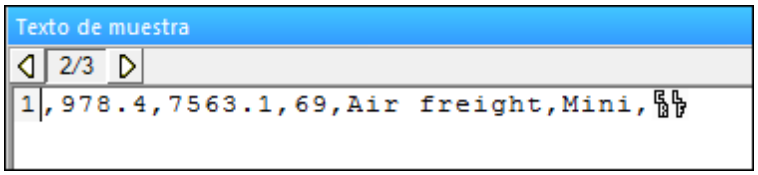
desde `111` hasta `Miscellaneous`.



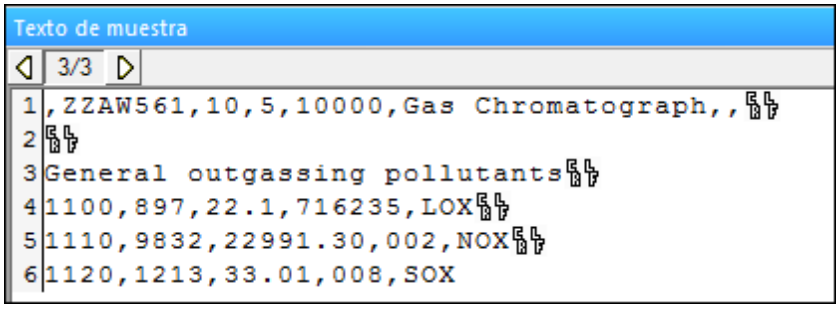
Si el separador no es el primer conjunto de caracteres de la primera línea del fragmento (como en nuestro ejemplo), entonces el primer fragmento incluirá todo el texto hasta la primera instancia del separador (p. ej. `222`).

Si el separador fuera `111`, entonces el primer fragmento sería una cadena de longitud cero porque el separador aparece al principio de la primera línea del fragmento original.

El segundo fragmento de nuestro ejemplo contiene la primera línea que contiene el separador `222`, sin el separador.



El tercer fragmento va desde la siguiente línea que contiene el separador `222`, sin el separador, y hasta el final del archivo de texto/fragmente.



7.4.5.1.3 Modo: delimitado (basado en línea)

Use el **modo delimitado (basado en línea)** para la división repetida cuando quiera dividir el texto en varios fragmentos de la siguiente manera:

- los fragmentos creados vienen definidos por los caracteres separadores que se introducen en el campo *Separador*.
- los caracteres separadores se incluyen en los fragmentos resultantes.
- se entiende como *fragmento* la línea completa que contiene el separador, hasta la siguiente línea que contenga el mismo separador.
- si el separador no aparece en la primera línea, entonces el primer fragmento contiene todo el texto hasta llegar a la primera línea que contenga el separador.

© 2018 Altova GmbH

A continuación describimos las opciones de configuración para el modo delimitado (basado en línea).

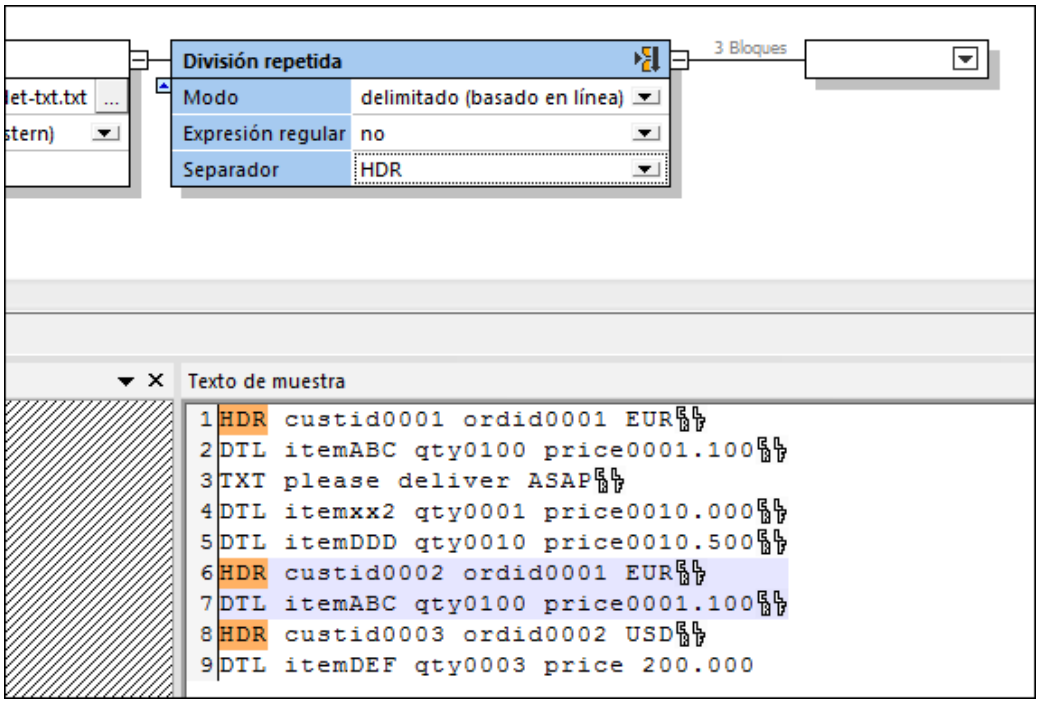
Expresión regular

Se trata de una configuración opcional que sirve para dividir el texto en fragmentos cada vez que una expresión regular encuentre coincidencias (*véase* *Dividir texto con expresiones regulares*). El valor predeterminado es `no`.

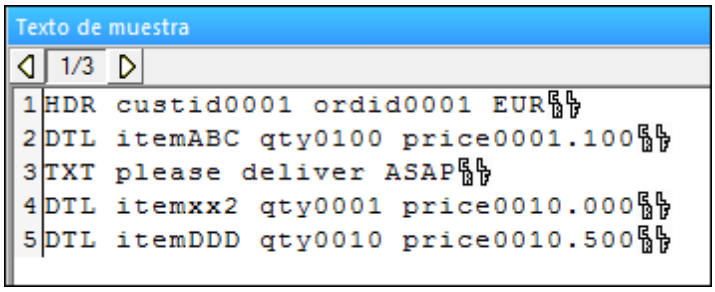
Separador

Especifica qué carácter o caracteres deben utilizarse como separador. El valor predeterminado es `ninguno`.

Por ejemplo, si se usa el separador `HDR` en el texto del ejemplo siguiente, se consigue dividir el texto en tres fragmentos.

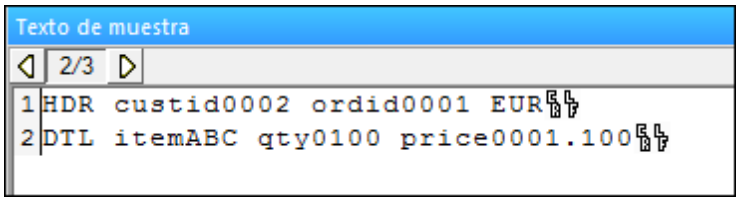


El primer fragmento contiene todos los caracteres desde el principio del archivo/ fragmento, incluidas todas las líneas hasta la siguiente línea que contenga el mismo separador.

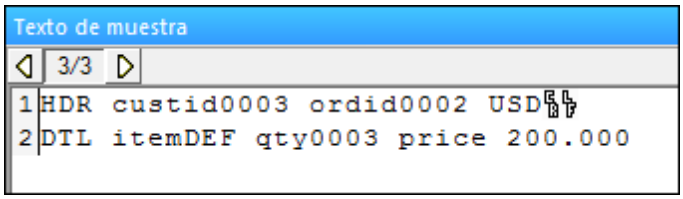


Esto le permite acceder a cualquier número de líneas comprendidas entre dos separadores. Es decir, es muy práctico cuando se trabaja con archivos que tienen tipos de registros opcionales o que no están en secuencia (p. ej. DTL o TXT).

El segundo fragmento contiene todo el texto comprendido entre la segunda instancia de HDR y la siguiente instancia de HDR.



El tercer y último fragmento incluye todo el texto comprendido entre la tercera instancia de HRD y el final del archivo/fragmento.



7.4.5.1.4 Modo: delimitado (la línea empieza con)

Use el **modo delimitado (la línea empieza con)** para la división repetida cuando quiera dividir el texto en varios fragmentos de la siguiente manera:

- los fragmentos creados vienen definidos por los caracteres separadores que se introducen en el campo *Separador*.
- los caracteres separadores se incluyen en los fragmentos resultantes.
- se entiende como *fragmento* la línea completa, empezando por el separador, hasta la siguiente línea que contenga el mismo separador al principio de línea.
- si el separador no aparece en la primera línea, entonces el primer fragmento contiene todo el texto hasta llegar a la primera línea que contenga el separador.

A continuación describimos las opciones de configuración para el modo delimitado (basado en línea).

© 2018 Altova GmbH

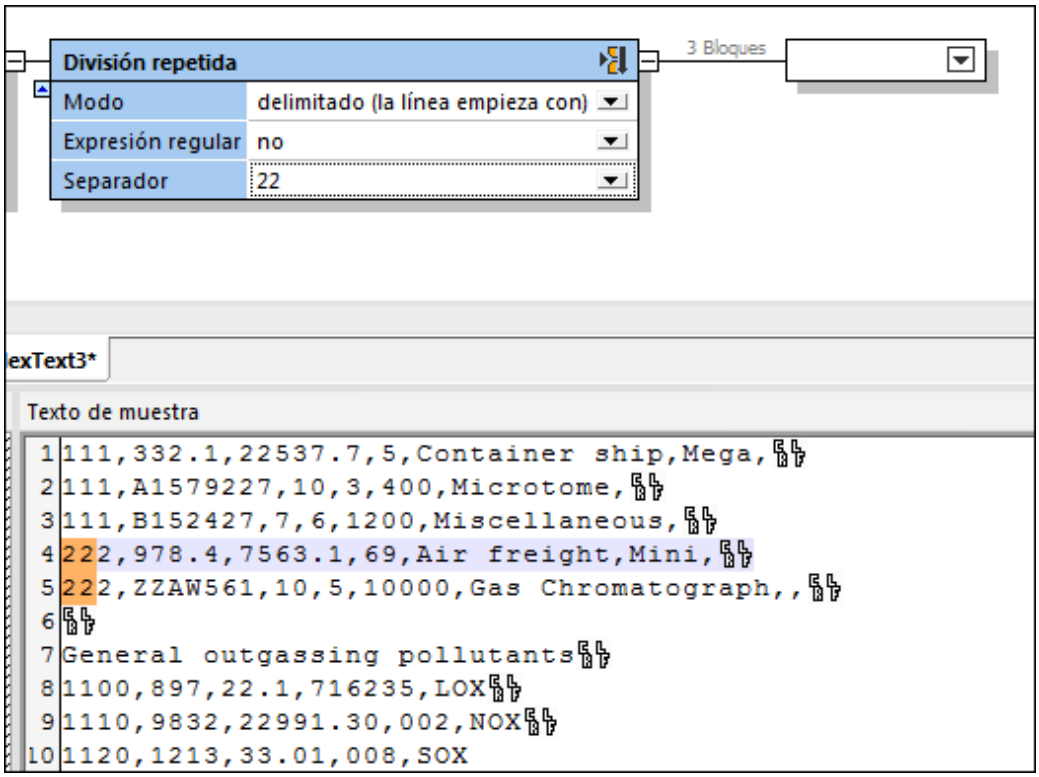
Expresión regular

Se trata de una configuración opcional que sirve para dividir el texto en fragmentos cada vez que una expresión regular encuentre coincidencias (*véase* *Dividir texto con expresiones regulares*). El valor predeterminado es `no`.

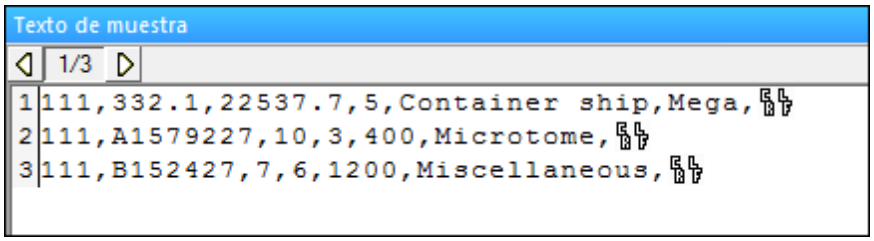
Separador

Especifica qué carácter o caracteres deben utilizarse como separador. El valor predeterminado es `ninguno`.

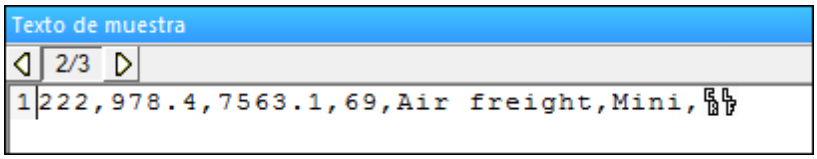
Por ejemplo, si se usa el separador `22` en el texto del ejemplo siguiente, se consigue dividir el texto en tres fragmentos.



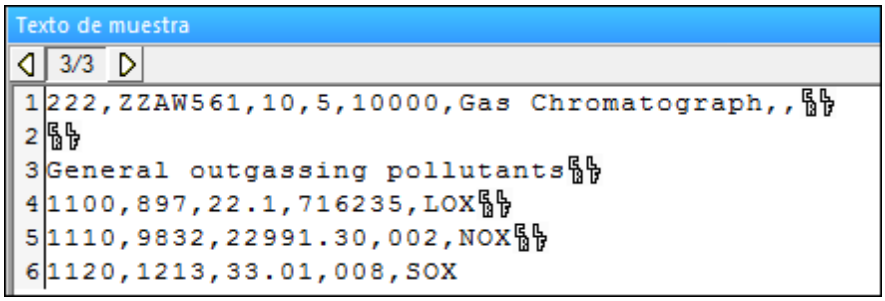
El primer fragmento contiene todas las líneas desde el principio del archivo/fragmento hasta la línea que contiene el separador `22`.



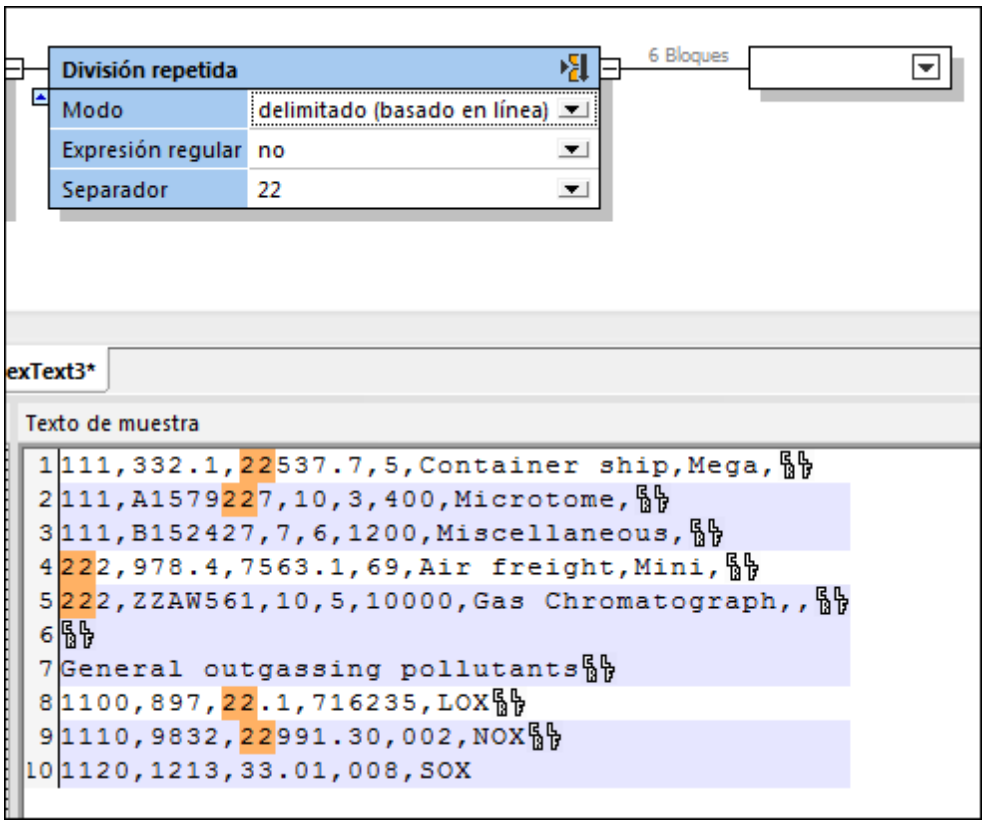
El segundo fragmento contiene todas las líneas comprendidas entre la segunda instancia de `22` hasta la siguiente instancia de `22` (en este caso se trata de una sola línea).



El tercer y último fragmento contiene todas las líneas que aparecen después de la tercera instancia de 22, hasta el final del archivo/fragmento.



Por el contrario, si utilizáramos el modo de división repetida **delimitado (basado en línea)** y el separador 22, obtendríamos estos resultados:



Se conseguirían 6 fragmentos, compuestos por las líneas que contienen el separador 22 en cualquier posición de la línea.

© 2018 Altova GmbH

7.4.5.2 *Dividir una vez*

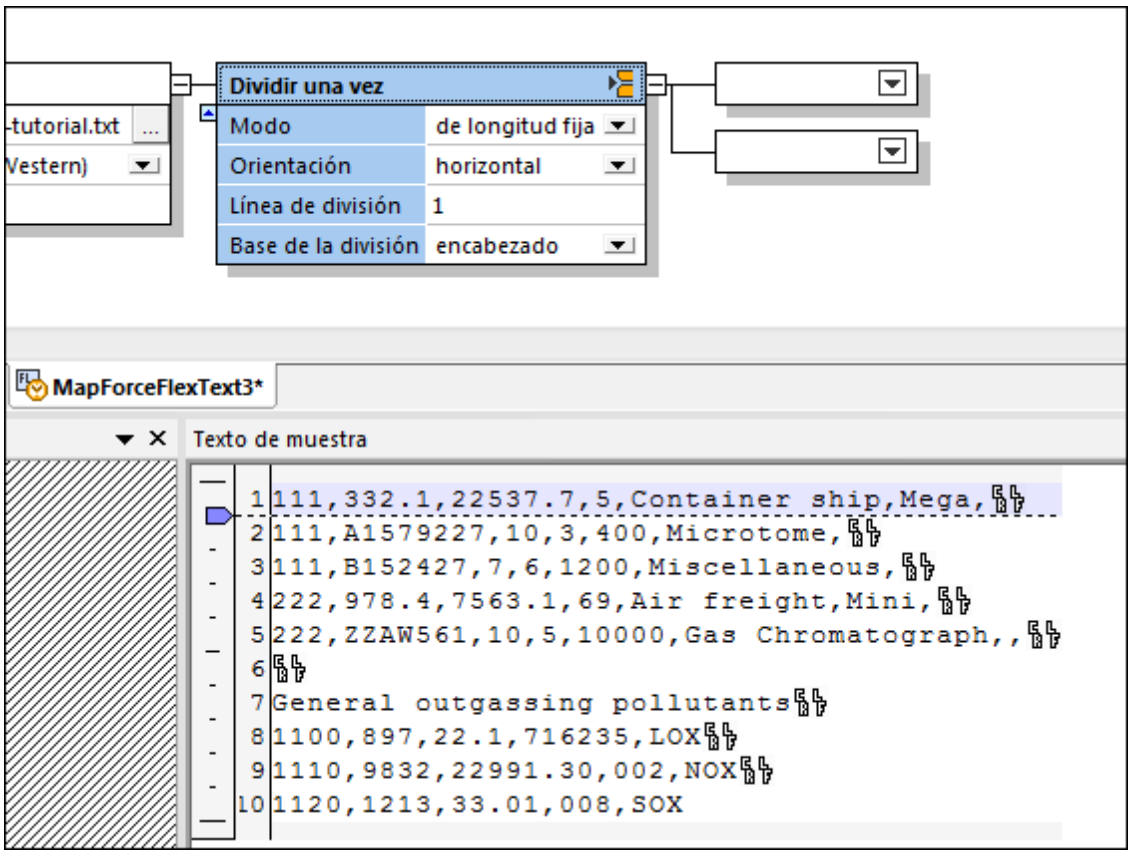


Con esta opción de división puede crear dos contenedores alineados verticalmente. El contenedor superior contiene el texto definido por la condición creada en el contenedor **Dividir una vez**. El contenedor inferior contiene el resto del archivo/fragmento de texto. La opción de división **Dividir una vez** tiene varios modos: longitud fija, delimitado flotante, delimitado basado en línea y delimitado (la línea empieza con).

Cuando se selecciona esta opción de división por primera vez, se establecen automáticamente parámetros predeterminados y los fragmentos resultantes aparecen en ambos contenedores. Observe que el contenedor **Dividir una vez** está activo en ese momento y que el panel Texto de muestra ofrece una vista previa de todas las líneas (o de todos los registros).

Esta es la **configuración predeterminada** del contenedor **Dividir una vez**:

Modo	`longitud fija`
Orientación	`horizontal`
Línea de división	`1`
Base de la división	`encabezado`



Resultado predeterminado

La primera línea del texto aparece en el contenedor superior. El contenedor inferior incluye el resto del archivo/fragmento de texto.

7.4.5.2.1 Modo: longitud fija

Use el **modo de longitud fija** para dividir una vez el texto en dos fragmentos verticales u horizontales de longitud fija, dividiéndolos por una línea o columna en relación al principio o al final del texto. A continuación describimos las opciones de configuración para el modo de longitud fija.

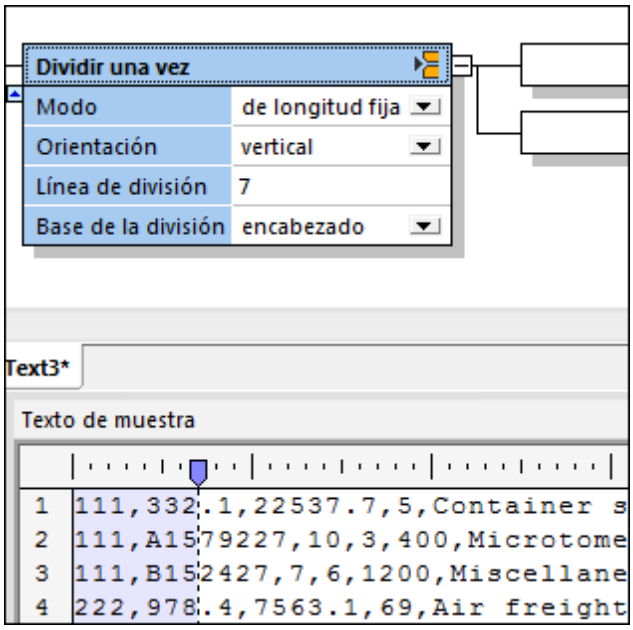
Orientación

Especifica si el fragmento de texto se debe dividir horizontal o verticalmente. Elija la opción `horizontal` para dividir el fragmento en dos secciones horizontales o `vertical` para dividir el fragmento en dos secciones verticales. La orientación predeterminada es `horizontal`.

Línea de división

Especifica el número de líneas (o columnas) tras el cual se debe dividir el fragmento. El desplazamiento predeterminado es 1. Hay dos maneras de cambiar este valor:

- introduciendo un valor en el campo *Línea de división* directamente o
- arrastrando la tabulación de la regla horizontal/vertical.



 © 2018 Altova GmbH

Base de la división

Encabezado	Divide el fragmento por el número de líneas de división indicado, empezando por el principio del texto.
Cola	Divide el fragmento por el número de líneas de división indicado, empezando por el final del texto.

7.4.5.2.2 Modo: delimitado (flotante)

Use el **modo delimitado (flotante)** para dividir una vez el texto en dos fragmentos, usando un separador personalizado que puede estar en cualquier posición del texto. Se trata del modo ideal para trabajar con archivos que no contienen caracteres CR ni LF y para dividir el fragmento en dos, en una posición determinada. Debe tener en cuenta que:

- este modo crea dos fragmentos que vienen definidos por los caracteres separadores que se indican en el campo *Separador.*
- los caracteres separadores no se incluyen en el fragmento.
- el primer fragmento irá del primer carácter del archivo/fragmento hasta el último carácter situado antes del separador.
- el segundo fragmento irá del primer carácter situado después del separador hasta el último carácter del archivo/fragmento.
- si el separador aparece en la primera/última posición del archivo/fragmento, el primero de los dos contenedores resultantes estará vacío.

A continuación describimos las opciones de configuración para el modo delimitado (flotante).

Expresión regular

Se trata de una configuración opcional que sirve para dividir el texto en dos fragmentos cuando la expresión regular encuentre coincidencias (*véase* *Dividir texto con expresiones regulares*). El valor predeterminado es `no`.

Separador

Especifica qué carácter o caracteres deben utilizarse como separador. El valor predeterminado es `ninguno`.

Base de la división

Encabezado	Busca la primera instancia del separador, empezando por el principio del texto.
Cola	Busca la primera instancia del separador, empezando por el final del texto.

7.4.5.2.3 Modo: delimitado (basado en línea)

Use el **modo delimitado (basado en línea)** para dividir el texto en dos fragmentos por la posición del separador. La línea que contiene el separador no se divide, sino que se conserva entera. Se trata del modo ideal para trabajar con archivos que contienen delimitadores de registros (CR/LF) y cuando queremos dividir el texto en dos fragmentos. Cuando use este modo de división debe tener en cuenta que:

- los dos fragmentos creados vienen definidos por los caracteres separadores que se introducen en el campo *Separador*.
- los caracteres separadores se incluyen en los dos fragmentos resultantes.
- el primer fragmento irá desde el principio del texto hasta la línea que contiene el separador.
- el segundo fragmento irá desde el texto y la línea que contiene el separador hasta el final del archivo/fragmento.
- si el separador aparece en la primera o en la última línea del archivo/fragmento, el primero de los dos contenedores resultantes estará vacío.

A continuación describimos las opciones de configuración para el modo delimitado (basado en línea).

Expresión regular

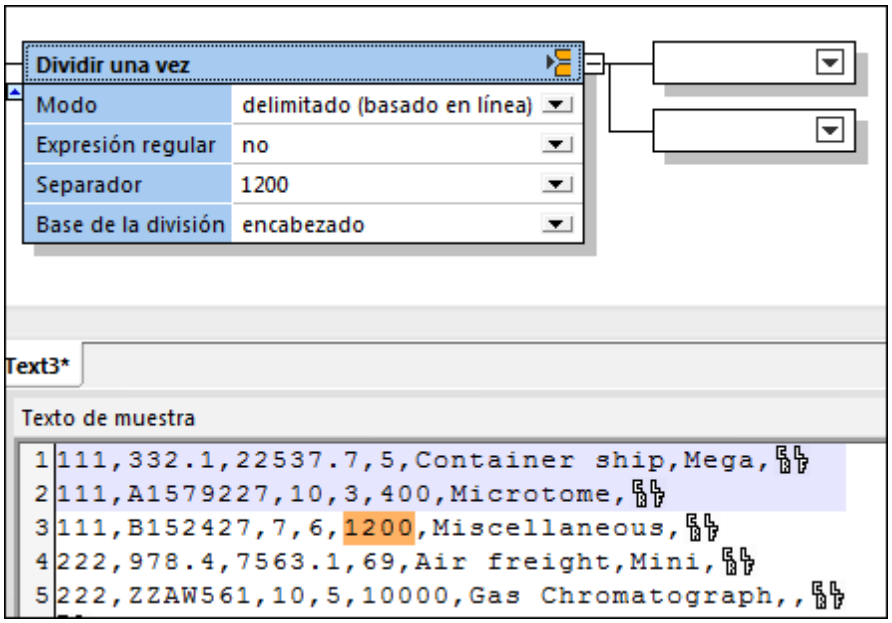
Se trata de una configuración opcional que sirve para dividir el texto en dos fragmentos cuando la expresión regular encuentre coincidencias (*véase Dividir texto con expresiones regulares*). El valor predeterminado es `no`.

Separador

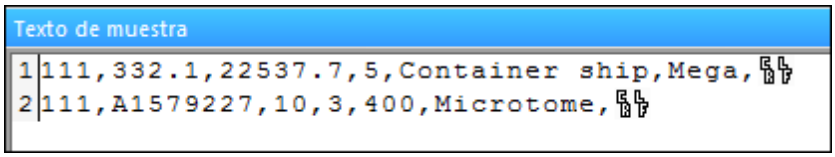
Especifica qué carácter o caracteres deben utilizarse como separador. El valor predeterminado es `ninguno`.

Por ejemplo, si se usa el separador `1200` en el texto del ejemplo siguiente, se consigue dividir el texto en dos fragmentos.

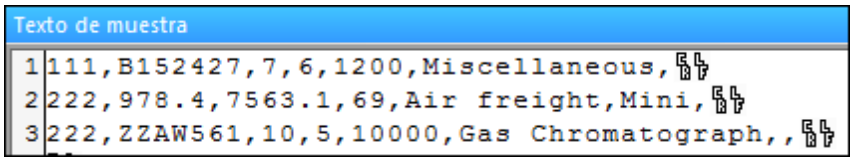
© 2018 Altova GmbH



El primer fragmento está formado por el texto desde el principio hasta la línea que contiene el separador.



El segundo fragmento está formado por la línea que contiene el separador y las demás líneas, hasta el final.



Base de división

Encabezado	Busca la primera instancia del separador, empezando por el principio del texto.
Cola	Busca la primera instancia del separador, empezando por el final del texto.

7.4.5.2.4 Modo: delimitado (la línea empieza con)

Use el **modo delimitado (la línea empieza con)** para dividir el texto en dos fragmentos por la primera línea que empiece con el separador elegido. Si selecciona este modo, se crean dos

fragmentos de texto:

- el primer fragmento está formado por todo el texto hasta la línea donde está el separador.
- el segundo fragmento está formado por el resto del texto, incluida la línea donde está el separador.
- si el separador está al principio de la primera línea, no se lleva a cabo la división (es decir, el primer fragmento resultante estará vacío).

A continuación describimos las opciones de configuración para el modo delimitado (la línea empieza con).

Expresión regular

Se trata de una configuración opcional que sirve para dividir el texto en dos fragmentos cuando la expresión regular encuentre coincidencias (*véase* *Dividir texto con expresiones regulares*). El valor predeterminado es `no`.

Separador

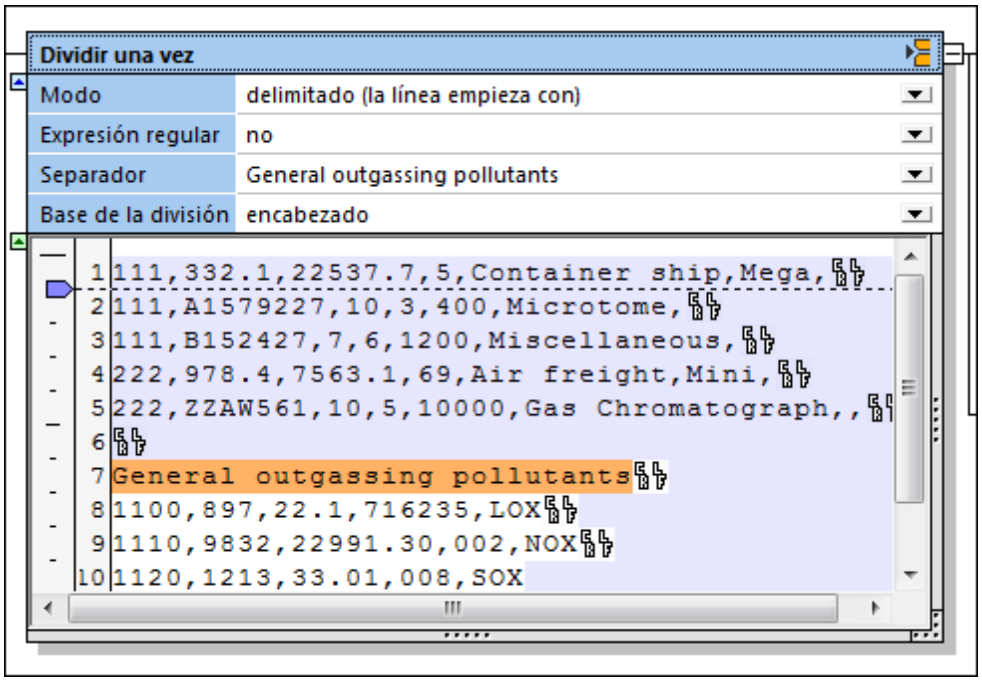
Especifica qué carácter o caracteres deben utilizarse como separador. El valor predeterminado es `ninguno`.

Base de división

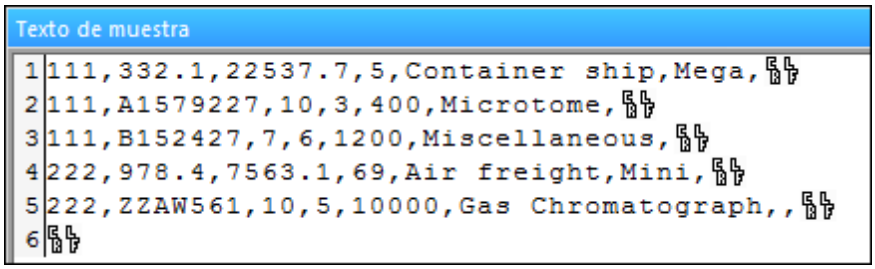
Encabezado	Busca la primera instancia del separador, empezando por el principio del texto.
Cola	Busca la primera instancia del separador, empezando por el final del texto.

Por ejemplo, en la imagen siguiente, la división se produce en la primera línea que empieza con el separador especificado (`General outgassing pollutants`). Si se elige la opción cola para la base de división, entonces la división ocurriría en la primera línea que empieza con el separador especificado, empezando por el final (en este ejemplo concreto se trata de la misma línea).

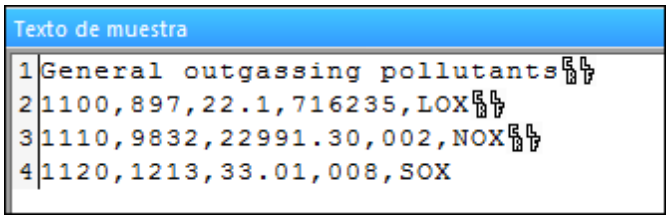
© 2018 Altova GmbH



El primer fragmento sería:



El segundo fragmento sería:



7.4.5.3 Conmutador

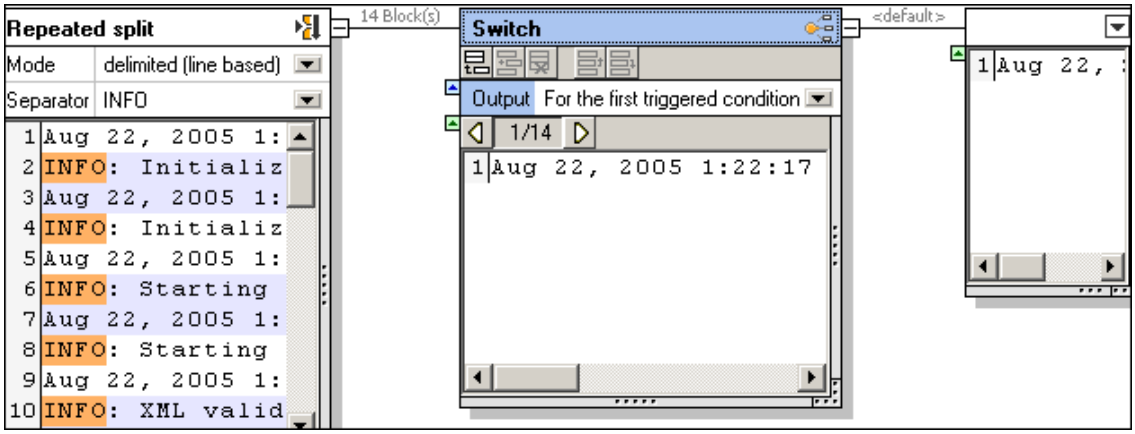


Con la opción Conmutador podrá definir varias palabras clave o condiciones para un solo fragmento de texto. Cada palabra clave que defina tendrá su contenedor asociado, que solamente recibirá datos si se cumple la condición especificada. Si ninguna de las condiciones se cumple, el fragmento se asignará a un contenedor *predeterminado*.

Esta es la **configuración predeterminada** del contenedor **Conmutador**:

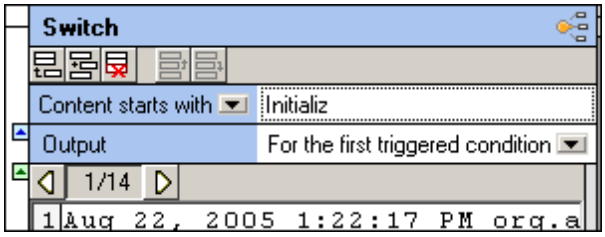
Resultados	`para la primera condición desencadenada`

El ejemplo que aparece a continuación procesa un archivo de registro Tomcat, donde queremos separar los procesos para poder asignarlos. Cuando defina un contenedor **Conmutador** por primera vez, a la derecha del contenedor aparece solamente el contenedor predeterminado. Todos los datos se pasan automáticamente a este contenedor predeterminado.



El contenedor **División repetida**, en modo `delimitado (basado en línea)` separa todas las secciones `INFO` del archivo de registro y las pasa al contenedor **Conmutador**.

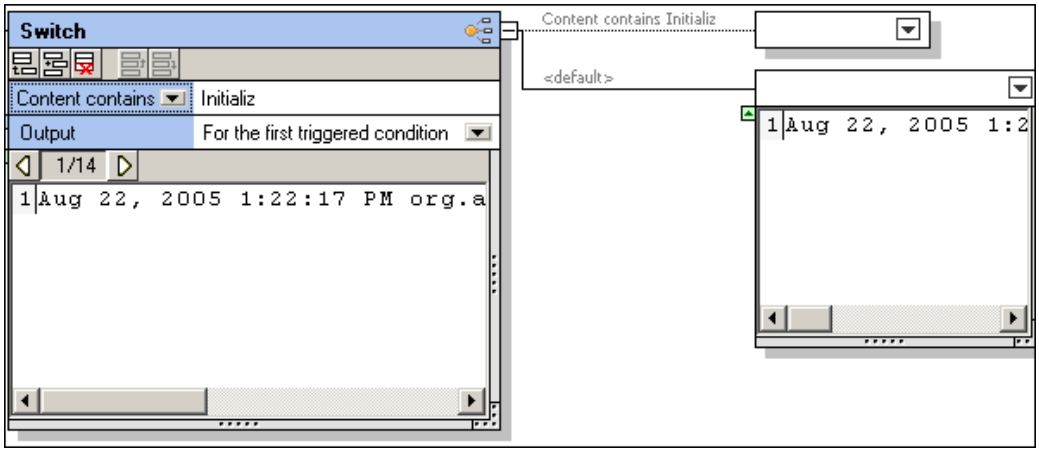
1. Haga clic en el icono **Anexar** para agregar una condición nueva al contenedor **Conmutador**.
2. Haga doble clic en el campo *El contenido empieza con*, introduzca `Initializ` y pulse **Entrar**.



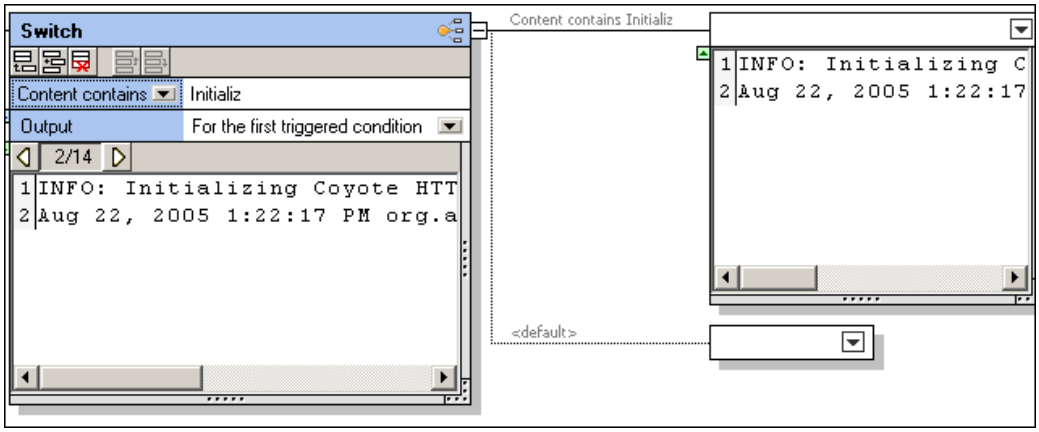
Esto añade un contenedor nuevo al que se reenviarán los datos si la condición se cumple. Si no se cumple, los datos se enviarán al contenedor predeterminado.

3. Haga clic en el cuadro combinado *El contenido empieza con* y cámbielo por la opción *El contenido contiene*.
   La primera condición ya se ha definido y más abajo podemos ver el resultado.
   El primer fragmento no contiene `Initializ` y, por tanto, su contenido se envía al contenedor predeterminado.

 © 2018 Altova GmbH

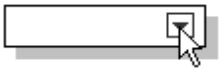


4. Haga clic en el icono **Mostrar bloque siguiente** para ver el siguiente fragmento de texto.

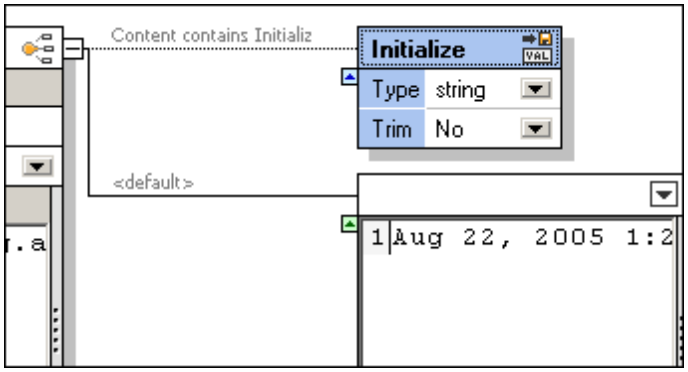


Ahora el fragmento `Initializing...` aparece en su contenedor asociado y el contenedor predeterminado está vacío. Si recorremos los fragmentos uno a uno, obtendremos una vista previa del contenido de cada uno de ellos.

5. Haga clic en el icono interactivo del contenedor y seleccione **Guardar como valor**.

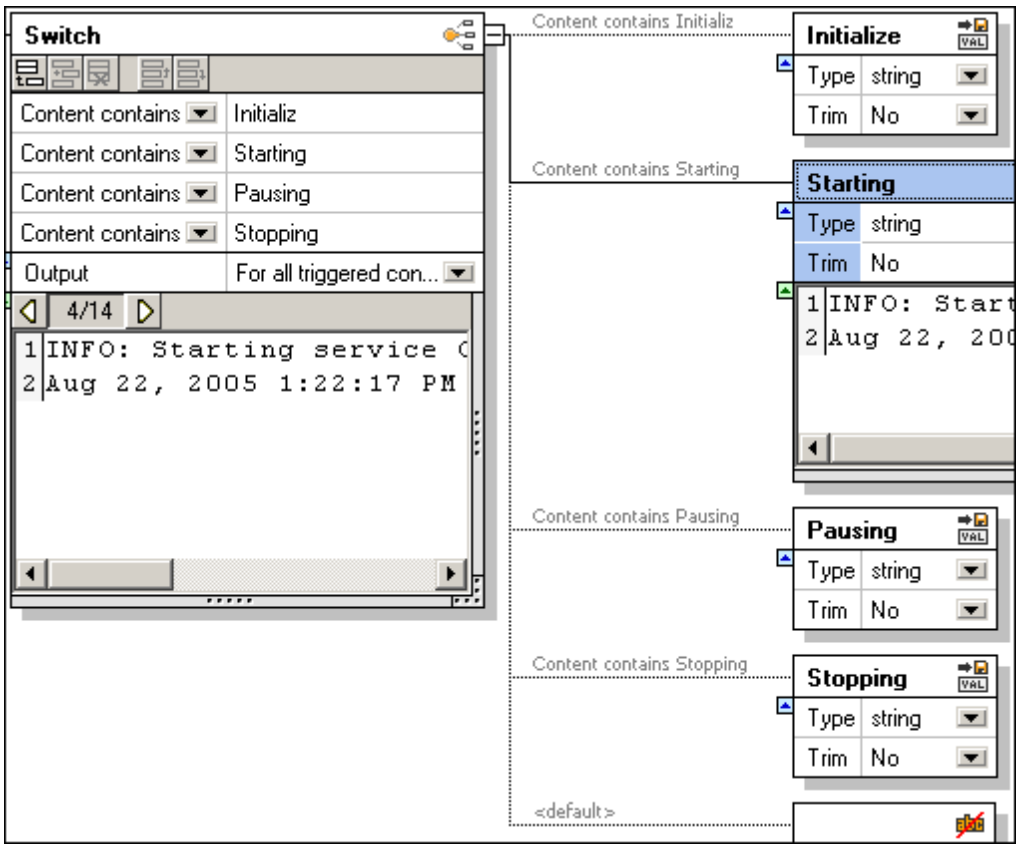


6. Haga doble clic en la barra de título **Guardar** y cambie el texto (p. ej. **Initialize**).



7. Haga clic en el icono **Anexar** para añadir una condición nueva al contenedor **Conmutador**.

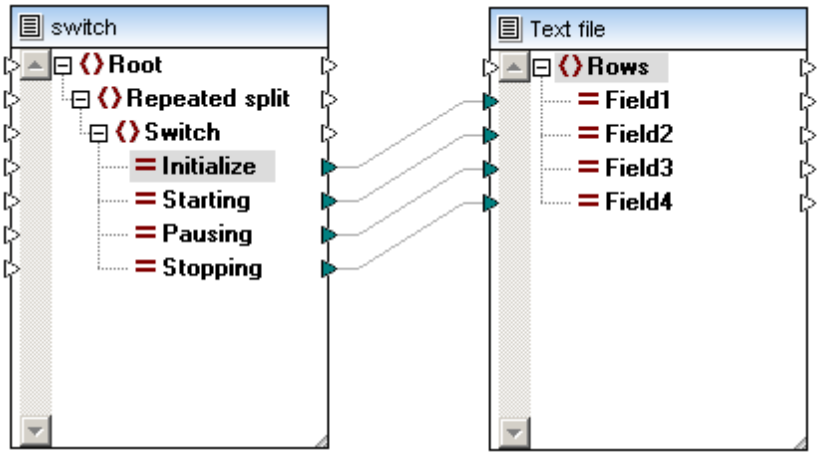
8. Haga doble clic en el campo *El contenido empieza con*, introduzca `Starting` y pulse **Entrar**.
   Puede agregar tantas condiciones como necesite (p. ej. `Pausing` y `Stopping`).
   Debe dar un nombre a cada uno de sus contenedores asociados para poder reconocerlos más fácilmente en MapForce más tarde.



   En la imagen anterior puede ver las cuatro condiciones y el contenido del contenedor **Starting** del bloque/fragmento 4. Los contenedores asociados se renombraron para poder identificarlos fácilmente en el componente MapForce. Recuerde que puede cambiar la posición de las condiciones con los botones **Subir** o **Bajar**.

9. Guarde la plantilla e insértela en MapForce.

© 2018 Altova GmbH



Nota: si un fragmento de texto del fragmento actual cumple una condición, entonces **todos los datos** de ese fragmento se pasan al contenedor asociado. Los datos no se dividen en absoluto, sino que se redirigen a los contenedores asociados o al contenedor predeterminado si no cumple ninguna de las condiciones definidas.

Los contenedores asociados que produce Conmutador se pueden seguir procesando. También puede cambiar estos contenedores por una **División repetida**, **Dividir una vez** o cualquier otro tipo de contenedor.

El contenido empieza con:
Los datos se pasan al contenedor asociado solamente si la cadena de la condición aparece al principio del fragmento de texto.

El contenido contiene:
Los datos se pasan al contenedor asociado si la cadena de la condición aparece en el fragmento de texto.

Para la primera condición desencadenada:
Los datos se pasan al contenedor asociado cuando **se cumple una de las condiciones** de la lista. Las demás condiciones se ignoran, aunque se cumplan, y no se pasan más datos a los demás contenedores asociados.

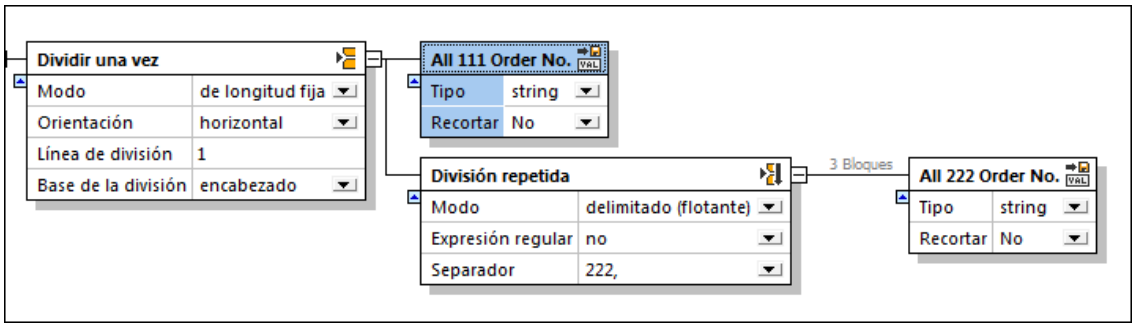
Para todas las condiciones desencadenadas:
Se pasan los datos para todas las condiciones de la lista que se cumplan. Esto permite que los mismos datos/fragmentos aparezcan varias veces en varios contenedores asociados simultáneamente. Por ejemplo, si un fragmento de texto contiene un texto que cumple con dos condiciones a la vez (p. ej. `initializing starting sequence` en el ejemplo anterior).

7.4.5.4 Nodo

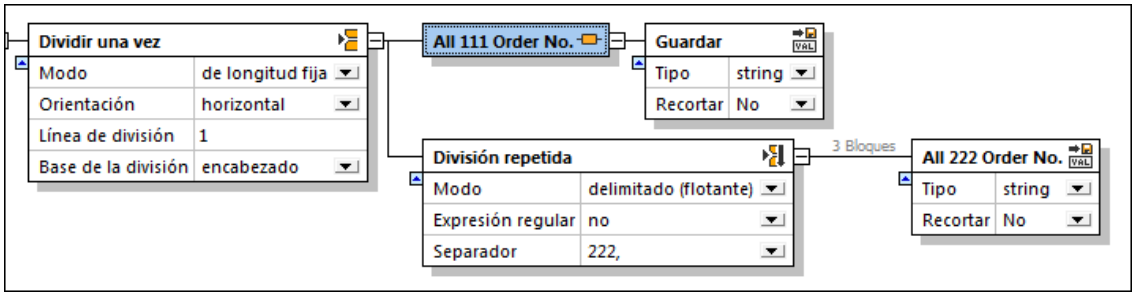


Con esta opción puede añadir un nivel jerárquico nuevo a las estructuras en forma de árbol de FlexText y MapForce. Los datos que contiene el siguiente nodo/contenedor se pasan tal y como están.

En la imagen siguiente, por ejemplo, el contenedor **All 111 Order No.** es el último contenedor de la rama superior.



Haga clic en el icono interactivo del contenedor y seleccione **Nodo** en el menú contextual.

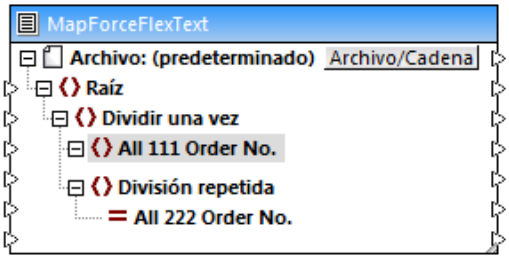


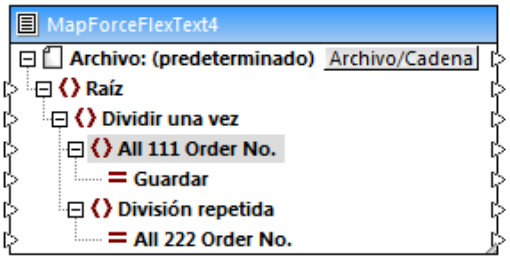
+

Se añade un contenedor nuevo a la derecha del contenedor actual.

Nota: el contenedor anexado automáticamente se definió como contenedor **Guardar como valor**.

La siguiente imagen muestra el aspecto (la estructura) que la plantilla FlexText tiene en MapForce antes y después de crear el contenedor **Nodo**.





El componente de la izquierda muestra la estructura inicial de la plantilla, antes de añadir el contenedor **Nodo** nuevo.

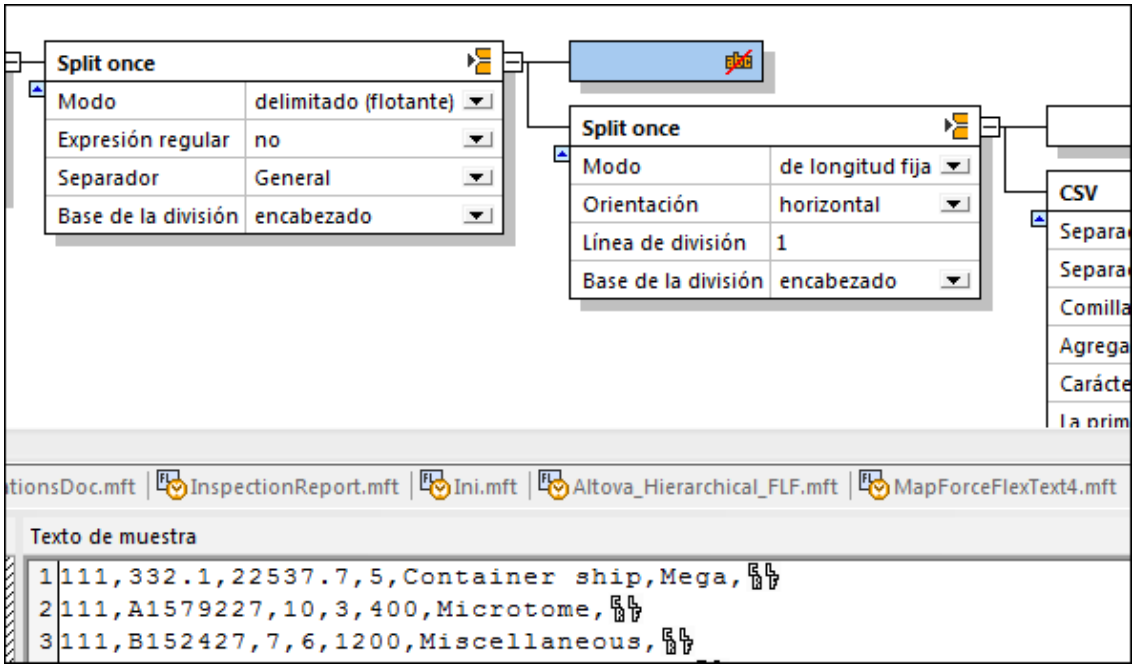
El componente de la derecha muestra la nueva estructura. Ahora **`All 111 Order No.`** es el elemento primario y debajo de él se añadió el elemento secundario **`Guardar`**.

© 2018 Altova GmbH

7.4.5.5 Omitir

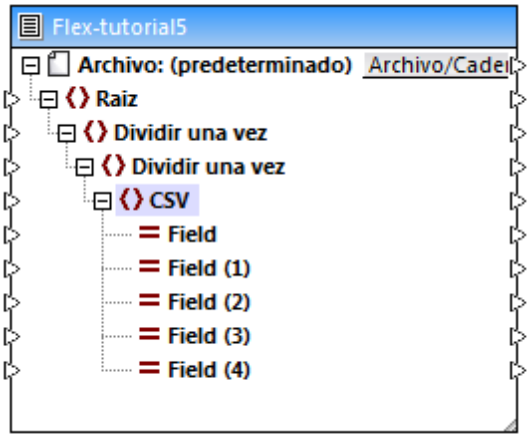


Con esta opción puede suprimir fragmentos de texto concretos. Esto significa que el contenedor y sus datos no estarán disponibles en MapForce como elementos asignables en la plantilla FlexText.



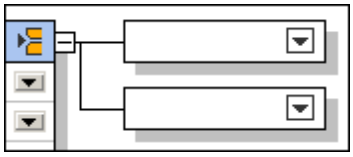
Por ejemplo, en la imagen anterior, el contenedor activo se definió como contenedor **Omitir**. Por tanto, el texto de este contenedor no aparecerá como elemento asignable en MapForce.

A continuación puede ver la estructura de la plantilla FlexText en MapForce. Entre los dos elementos **`Dividir una vez`** no existe ningún elemento que se pueda asignar.



Nota: también existen contenedores **Omitir** predeterminados. Se trata de los contenedores

nuevos que se anexan automáticamente al seleccionar la opción **Dividir una vez** y **División repetida**.



En un principio, el contenido de estos contenedores no está disponible (no se puede asignar) en MapForce cuando se inserta la plantilla. Para poder asignar el contenido es necesario seleccionar en FlexText el tipo de contenedor: **Guardar valor**, **Guardar como CVS**, etc.

7.4.5.6 Guardar como CSV (delimitado)



Con esta opción puede guardar fragmentos de texto como texto CSV y asignar sus columnas en MapForce. En FlexText puede crear un número ilimitado de contenedores/componentes CSV y cada contenedor CSV puede tener varios separadores.

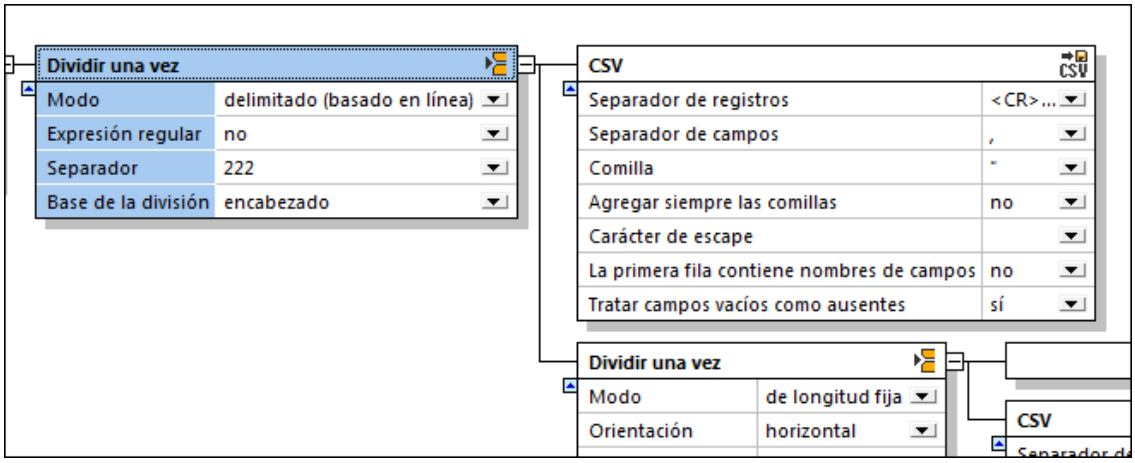
El panel Texto de muestra ofrece una vista previa del fragmento CSV actual y permite especificar los nombres y los tipos de campo. Cada columna constituye un elemento asignable en el componente FlexText en MapForce.

Esta es la configuración predeterminada del contenedor **Guardar como CSV**:

Separador de registros	`CR LF`
Separador de campos	`,`
Comilla	`"`
Agregar siempre las comillas	`no`
Carácter de escape	`<vacío>`
La primera fila contiene nombres de campos	`no`
Tratar campos vacíos como ausentes	`sí`

En el ejemplo que aparece a continuación se divide un archivo de texto pequeño en dos archivos CSV (y después estos dos archivos CSV se asignan a sendos archivos XML en MapForce).

© 2018 Altova GmbH

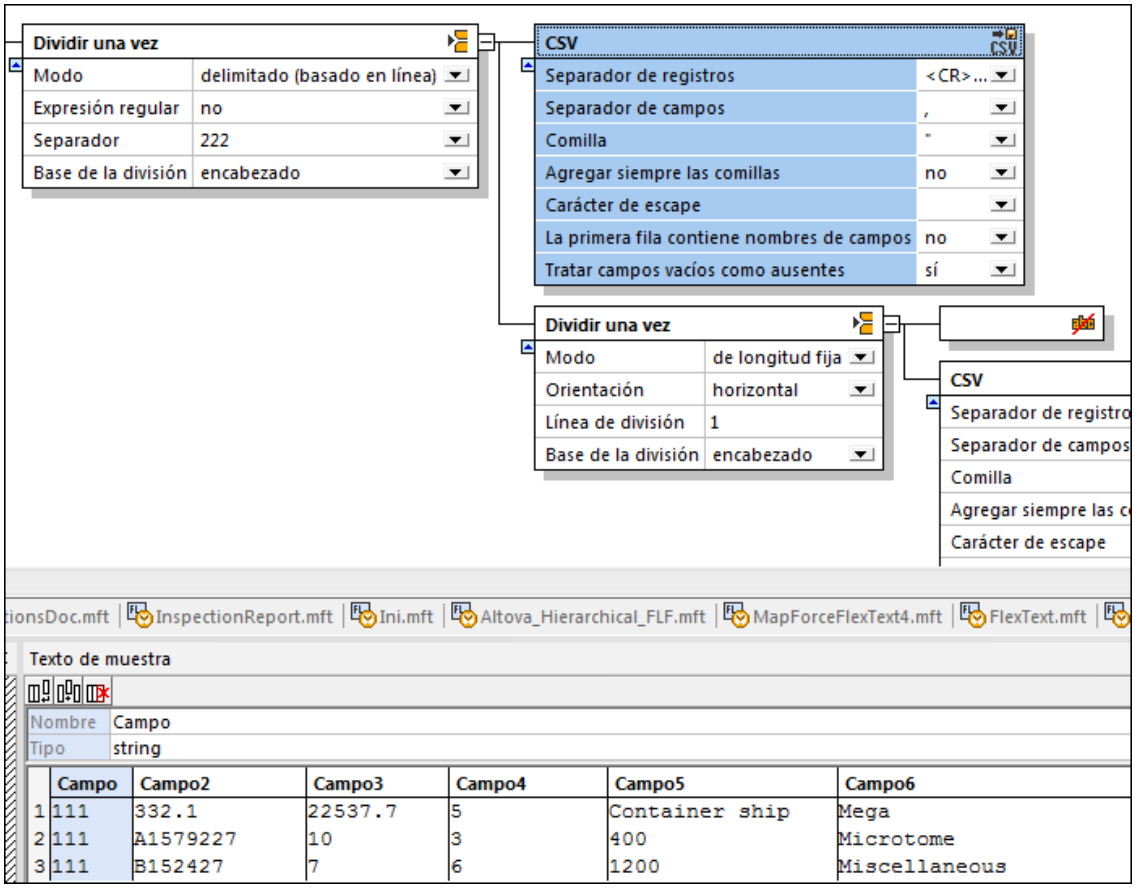


El contenedor **Dividir una vez** (*imagen anterior*) se utiliza para crear **dos** contenedores. Para ello se usa el modo `delimitado (basado en línea)` con el separador `222`. Todos los registros que aparecen antes del primer 222 se pasan al contenedor **CSV**. El primer contenedor, formado por todos los registros que incluyen el texto 111, se define como contenedor **CSV**. El panel Texto de muestra ofrece una vista previa del contenido del contenedor activo, es decir, el contenedor **Dividir una vez**.

Texto de muestra

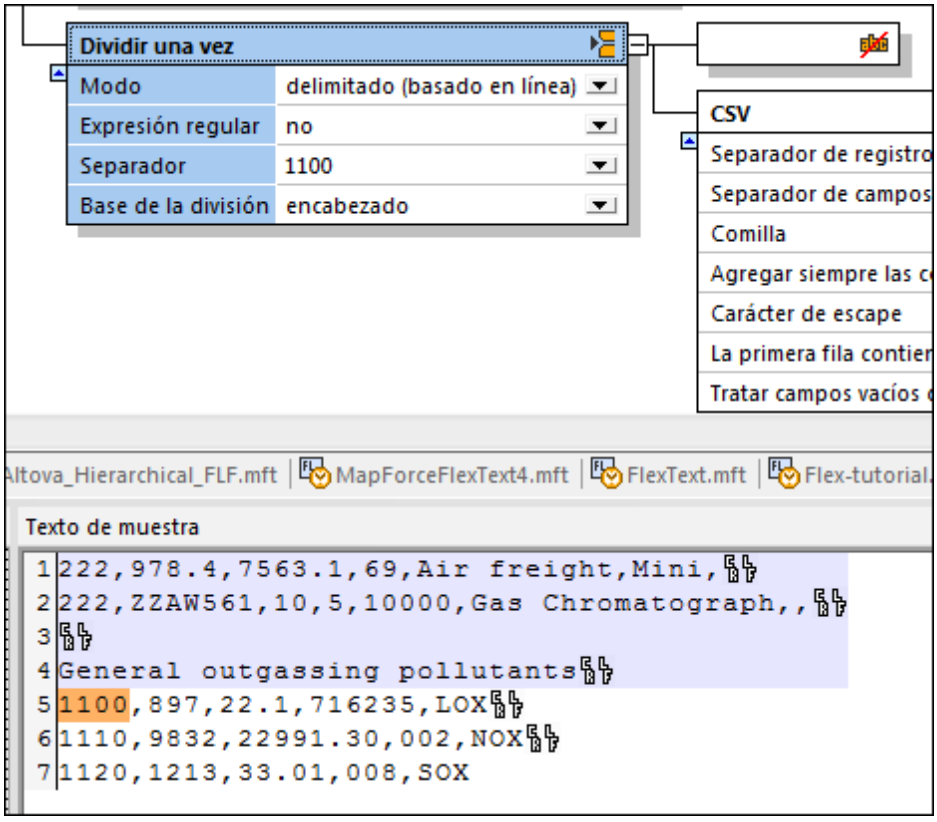
```
1 111,332.1,22537.7,5,Container ship,Mega,
2 111,A1579227,10,3,400,Microtome,
3 111,B152427,7,6,1200,Miscellaneous,
4 222,978.4,7563.1,69,Air freight,Mini,
5 222,ZZAW561,10,5,10000,Gas Chromatograph,,
6
7 General outgassing pollutants
8 1100,897,22.1,716235,LOX
9 1110,9832,22991.30,002,NOX
10 1120,1213,33.01,008,SOX
```

La configuración predeterminada del contenedor **CSV** no se modificó. Haga clic en el contenedor **CSV** para ver su contenido en forma de tabla.



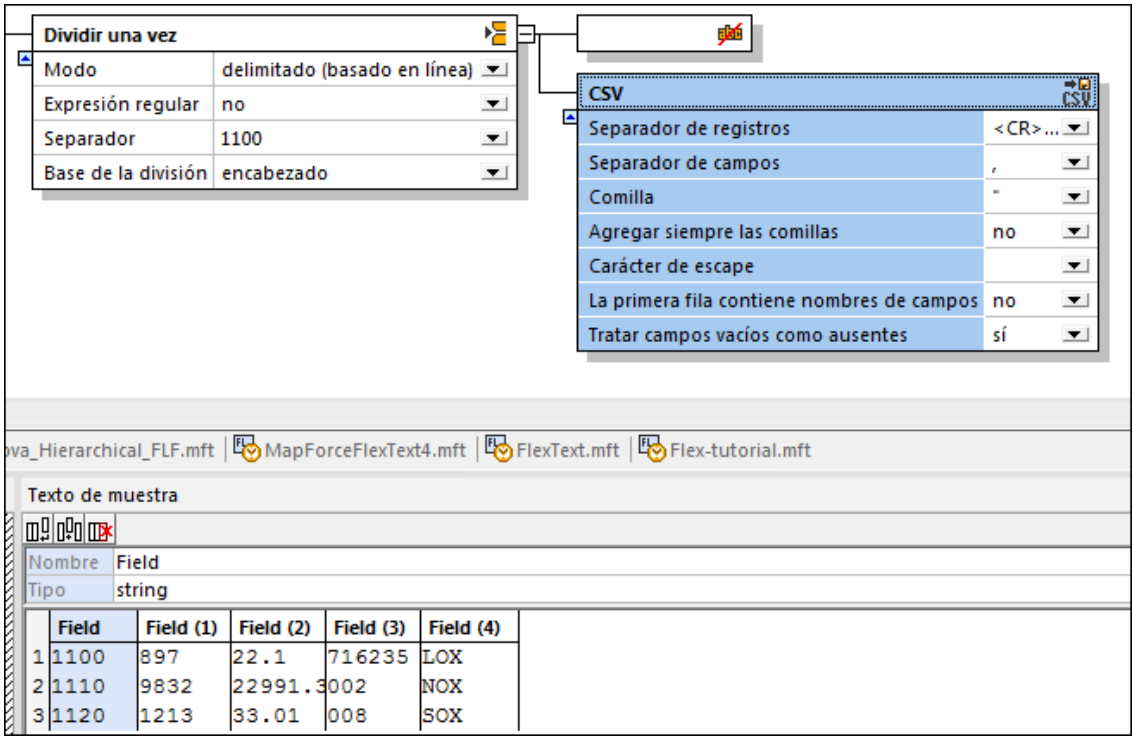
El segundo contenedor incluye los datos sobrantes y se define a su vez como contenedor **Dividir una vez**. Como resultado se crean dos contenedores más, uno de los cuales se define también como contenedor **CSV**. Haga clic en el contenedor **Dividir una vez** para ver su contenido actual.

© 2018 Altova GmbH



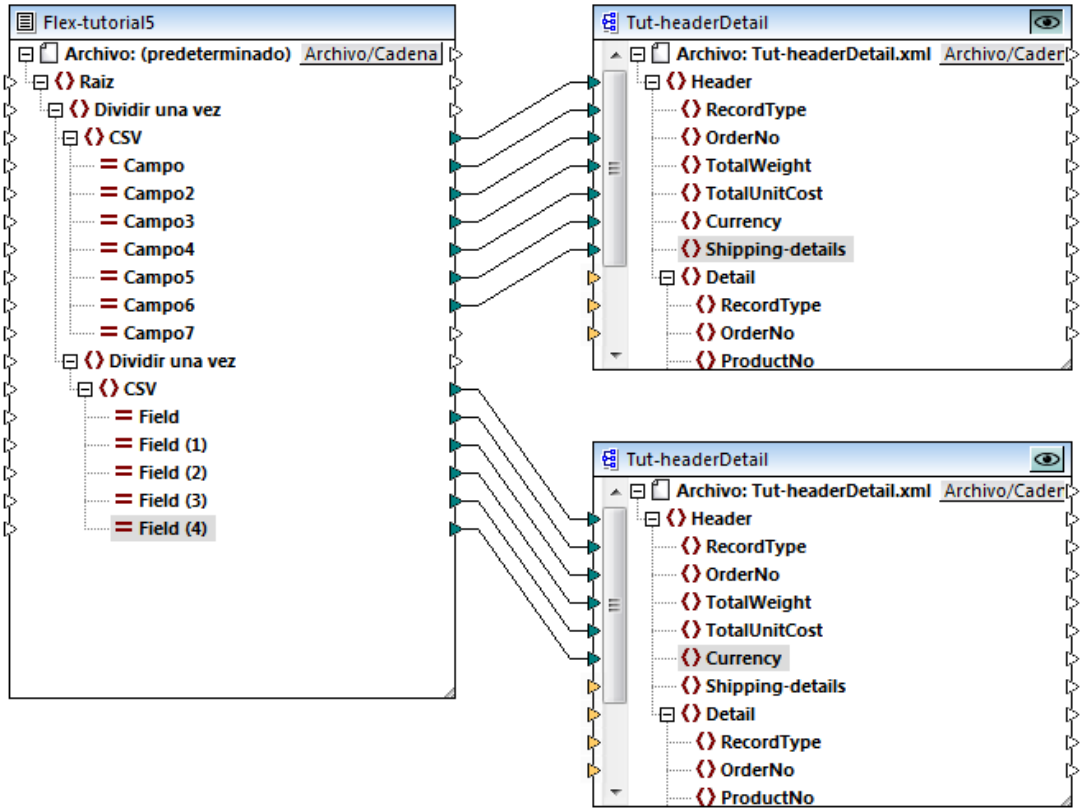
El modo `delimitado (basado en la línea)` con el separador `1100` se utiliza para dividir los datos sobrantes en dos secciones.

- al primer contenedor se pasarán todos los registros que aparecen antes del primer 1100. Este contenedor se definirá como contenedor **Omitir**, para que su contenido no se pueda asignar en MapForce.
- el segundo contenedor se define como contenedor **CSV**. La configuración predeterminada no se cambia. Haga clic en el contenedor **CSV** para ver su contenido en forma de tabla.



Inserte la plantilla FlexText en MapForce para asignar sus datos a los componentes de destino de la asignación. En este ejemplo, cada contenedor **CSV** se asigna a un archivo XML diferente.

© 2018 Altova GmbH



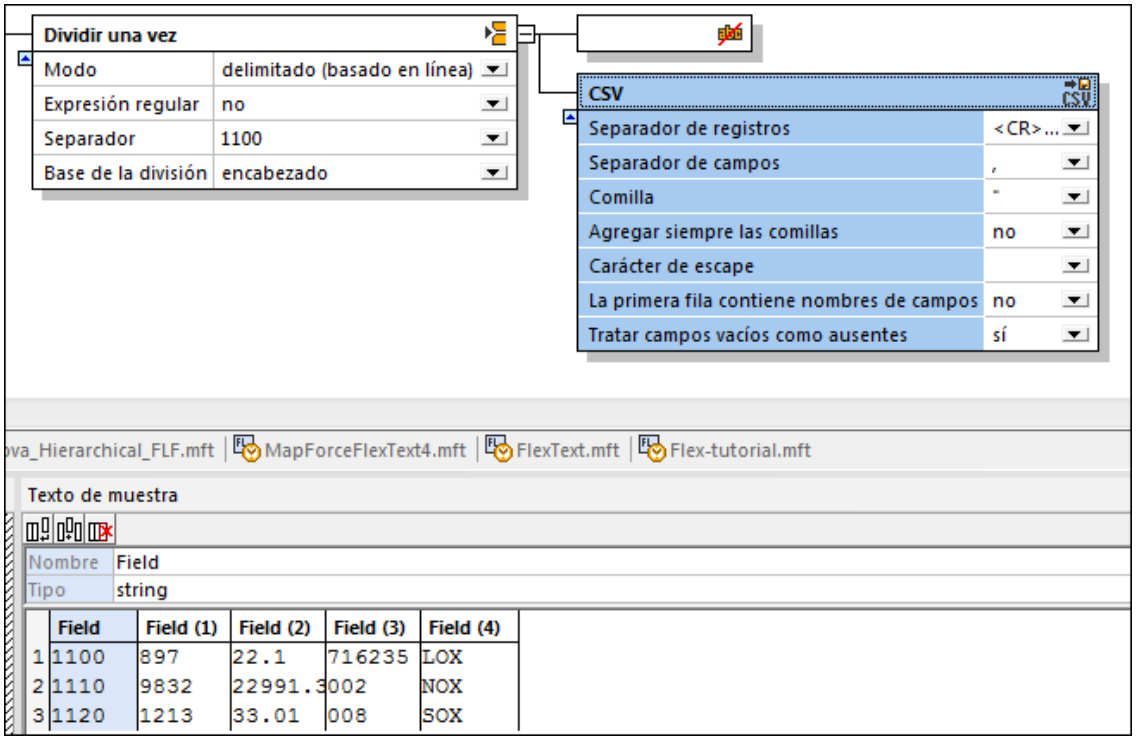
Observe que no todos los elementos de los contenedores **CSV** se asignan a los archivos de destino.
El primer archivo XML contiene todos los registros de tipo 111.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3    <Header>
4      <RecordType>111</RecordType>
5      <OrderNo>332</OrderNo>
6      <TotalWeight>22537.7</TotalWeight>
7    </Header>
8    <Header>
9      <RecordType>111</RecordType>
10     <OrderNo>0</OrderNo>
11     <TotalWeight>10</TotalWeight>
12   </Header>
13   <Header>
14     <RecordType>111</RecordType>
15     <OrderNo>0</OrderNo>
16     <TotalWeight>7</TotalWeight>
17   </Header>
18 </Order>
19
```

El segundo archivo XML contiene todos los registros que empiezan con 1100.

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <Header>
    <RecordType>1100</RecordType>
    <OrderNo>897</OrderNo>
    <TotalWeight>22.1</TotalWeight>
    <TotalUnitCost>716235</TotalUnitCost>
  </Header>
  <Header>
    <RecordType>1110</RecordType>
    <OrderNo>9832</OrderNo>
    <TotalWeight>22991.3</TotalWeight>
    <TotalUnitCost>2</TotalUnitCost>
  </Header>
  <Header>
    <RecordType>1120</RecordType>
    <OrderNo>1213</OrderNo>
    <TotalWeight>33.01</TotalWeight>
    <TotalUnitCost>8</TotalUnitCost>
  </Header>
</Order>
```

Configurar los datos/el contenedor CSV:



Al hacer clic en un campo del panel Texto de muestra, el campo se resalta y se puede configurar:

- El nombre del campo (*Nombre*).
- El tipo de datos del campo (*Tipo*): `string`, `boolean`, `decimal`, etc.

© 2018 Altova GmbH

- Haga clic en el icono **Anexar** para anexar un campo nuevo.
- Haga clic en el icono **Insertar** para insertar un campo antes del campo activo.
- Haga clic en el icono **Eliminar** para eliminar el campo activo.

Nota: puede arrastrar el borde de los campos con el ratón para ver todos los datos que contienen los campos.

Agregar siempre las comillas

Esta opción sirve para definir si el tipo de comillas especificado se añade a todos los campos del archivo CSV de destino.

Tratar campos vacíos como ausentes

Esta opción sirve para especificar que los campos que estén vacíos en el archivo de origen no produzcan elementos vacíos equivalentes (elementos/atributos) en el archivo de destino. Recuerde que, no obstante, en el archivo de origen deben existir los delimitadores para los campos vacíos.

7.4.5.7 Guardar como FLF (longitud fija)

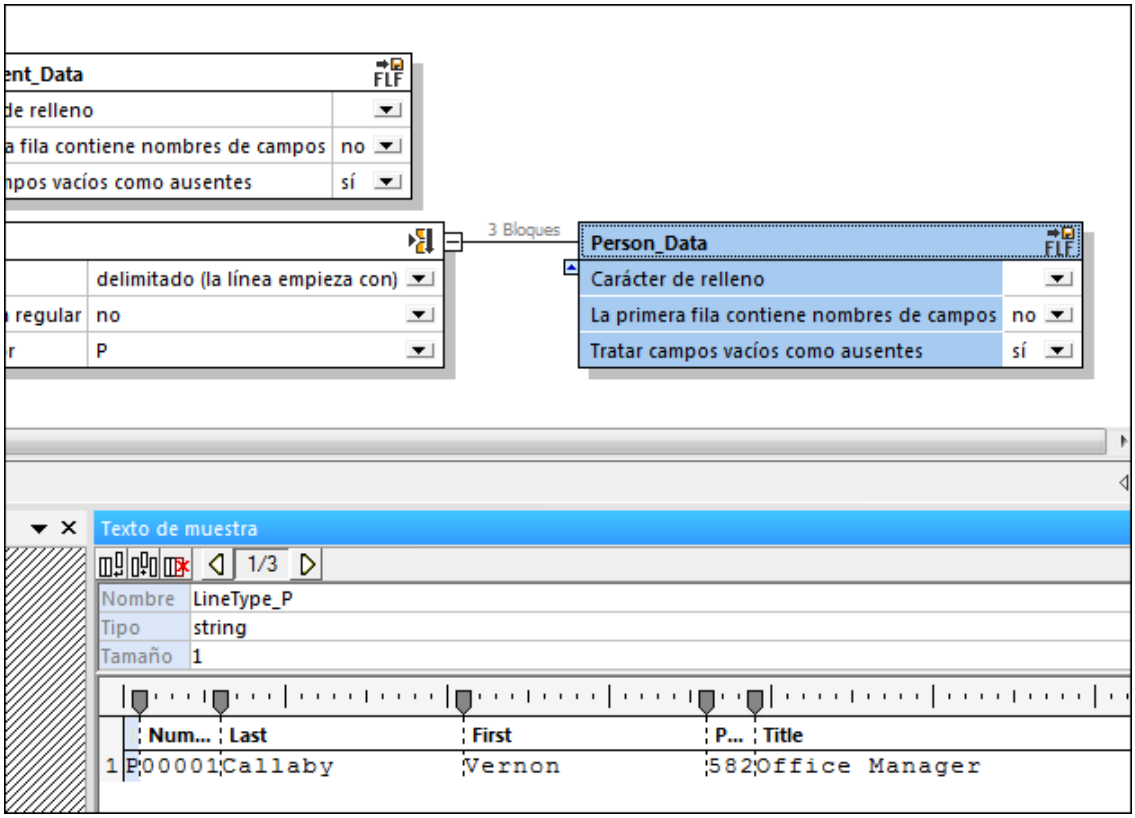


Con esta opción puede guardar fragmentos de texto como texto de longitud fija y realizar asignaciones de datos entre sus columnas y otros componentes en MapForce. En FlexText puede crear un número ilimitado de contenedores/componentes FLF y cada contenedor FLF puede tener caracteres de relleno diferentes.

El panel Texto de muestra ofrece una vista previa del fragmento FLF activo y en él puede especificar el nombre, la longitud y el ancho de sus campos. Cada columna aparece como elemento asignable en el componente de texto de MapForce.

Esta es la **configuración predeterminada** del contenedor FLF:

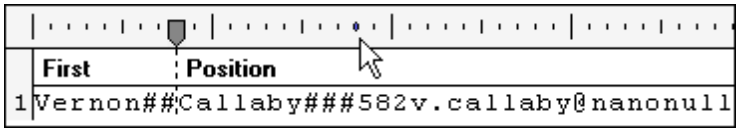
Carácter de relleno	`<vacío>`
La primera fila contiene nombres de campos	`no`
Tratar campos vacíos como ausentes	`sí`



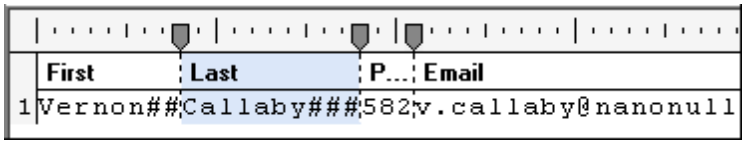
Configurar los datos/el contenedor FLF

Tras definir un contenedor como **Guardar como FLF**, el panel Texto de muestra ofrece el contenido correspondiente (*imagen anterior*). Se inserta automáticamente un campo de 10 caracteres de ancho.

- Haga clic en la tabulación de la regla y arrástrela a la nueva posición. Aparece información rápida sobre la posición actual de la tabulación.
- Al pasar el puntero del mouse por la regla aparece un punto. Al hacer clic en un punto se crea una tabulación nueva en la posición donde se hizo clic.

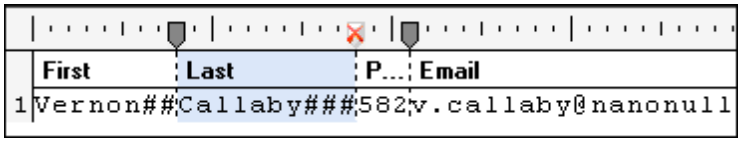


- Una vez definida la nueva posición, haga clic en un campo y edite su nombre en el campo *Nombre*.



- Para **quitar** un campo, haga clic en la tabulación y arrástrela fuera de la regla. Cuando la tabulación aparezca con una cruz roja, suelte el botón del ratón y la tabulación desaparece.

© 2018 Altova GmbH

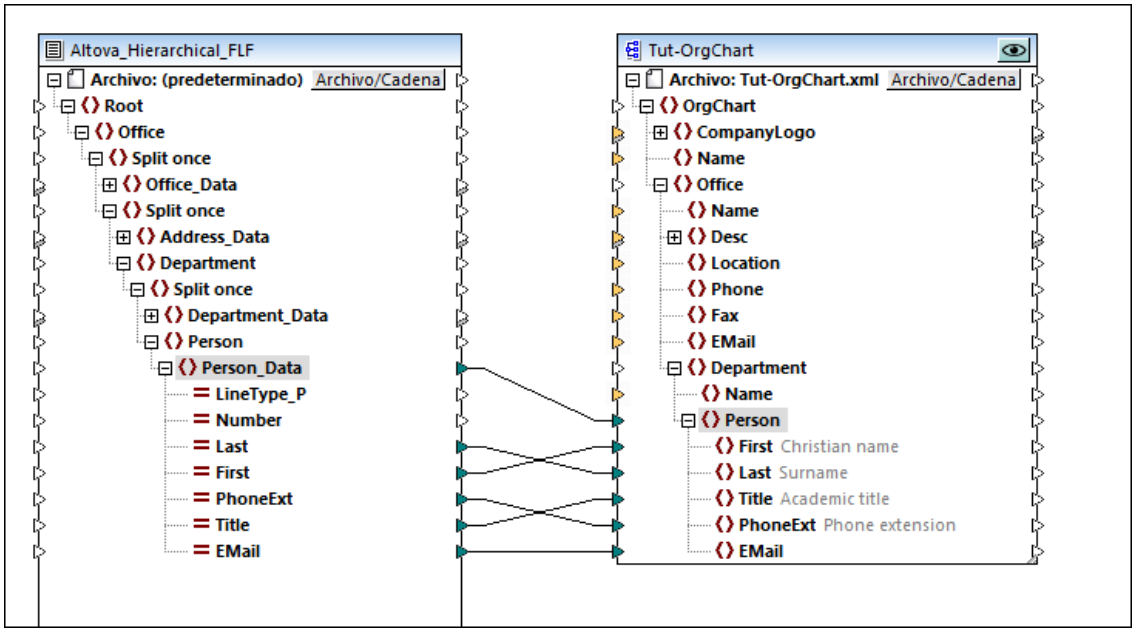


Nota: cuando se arrastra una tabulación por la regla, las tabulaciones situadas a su derecha cambian de posición automáticamente. Para que las demás tabulaciones no cambien de posición, pulse la tecla **Mayús** mientras arrastra la tabulación.

Al hacer clic en un campo del panel Texto de muestra, el campo se resalta y se puede configurar:

- Haga clic en el icono **Anexar** para anexar un campo nuevo de 10 caracteres de longitud.
- Haga clic en el icono **Insertar** para insertar un campo antes del campo activo, con 10 caracteres de longitud.
- Haga clic en el icono **Eliminar** para eliminar el campo activo.
- Haga clic en el campo *Nombre* para editar el nombre predeterminado.
- Haga clic en el campo *Tipo* para definir el tipo de datos del campo: `string`, `boolean`, `decimal`, etc.

Inserte la plantilla FlexText en MapForce para asignar los datos a los archivos de destino. En este ejemplo, los elementos FLF se asignan a elementos XML.



Si la opción ***Tratar campos vacíos como ausentes*** es **sí**, los campos que estén vacíos en el archivo de origen no producirán elementos vacíos equivalentes (elementos/atributos) en el archivo de destino. Se considera que un campo está ausente cuando no hay datos entre dos caracteres de relleno consecutivos.

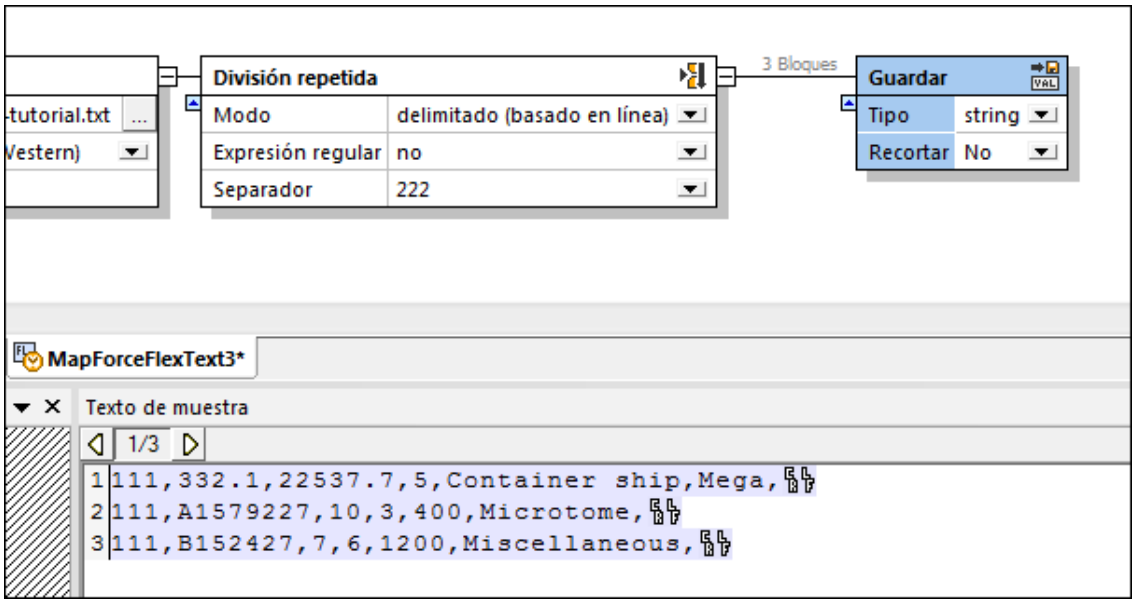
7.4.5.8 Guardar como valor

Con esta opción puede crear un contenedor cuyos datos estarán disponibles en MapForce en forma de elemento asignable. Si no cambia el nombre del contenedor en FlexText, el elemento asignable aparece en MapForce con el nombre **`Guardar`**.

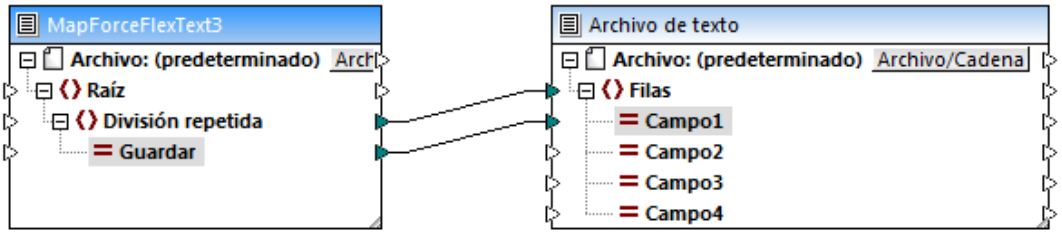
Esta esta la **configuración predeterminada** del contenedor **Guardar**:

Tipo	`string`
Recortar	no

En el ejemplo siguiente puede ver un contenedor **Guardar** y su contenido en el panel Texto de muestra.



Guarde esta plantilla y ábrala en MapForce para asignar el elemento **Guardar** a otros elementos del componente de destino.



Nota: el elemento **`campo1`** del archivo de texto de destino contendrá 3 fragmentos suministrados por el elemento **`Guardar`**.

 © 2018 Altova GmbH

Tipo
En este campo puede definir el tipo de datos de los fragmentos de texto.

Recortar
Define el lado donde se deben recortar los caracteres: **lado izquierdo**, **lado derecho** o **ambos lados**. Si hace clic en **Sí**, aparece el campo *Recortar juego de caracteres*.

Recortar juego de caracteres
Define el número de caracteres que se deben recortar en este fragmento de texto. Los caracteres se quitan del lado elegido en el campo *Recortar*.

7.4.6 Expresiones regulares en FlexText

En MapForce FlexText puede usar expresiones regulares para:

1. Dividir contenedores de texto cuando se encuentren coincidencias (el texto coincidente hace las veces de separador). En este caso, las expresiones regulares están ancladas implícitamente y, por tanto, los caracteres del acento circunflejo (^) y del signo del dólar ($) no se utilizan. Por ejemplo, para buscar una serie de tres dígitos consecutivos debe usar `[0-9]{3}` en lugar de `^[0-9]{3}$`.
2. Redirigir texto de un contenedor Conmutador si la expresión regular devuelve resultados.

Puede usar expresiones regulares en los componentes FlexText en estos lenguajes de destino de MapForce:

- Motor integrado (para la vista previa de resultados de la asignación)
- Motor integrado (para ejecutar archivos de ejecución de MapForce Server)
- Lenguajes de generación de código C++, C# y Java. (Recuerde que en estos lenguajes algunas funciones avanzadas de las expresiones regulares pueden depender de la implementación que haga el lenguaje de este tipo de expresiones.)

La sintaxis y semántica de expresiones regulares de FlexText están basadas en http://www.w3.org/TR/xmlschema-2/#regexs (parecido a lo que sucede con la función **`tokenize-regexp`** de la biblioteca de funciones **core** de MapForce). No obstante, debe tener en cuenta que:

- Si la condición de división encuentra dos secuencias consecutivas, FlexText crea un resultado vacío entre medio (esto ocurre también cuando no se usan expresiones regulares).
- No se admite el uso de indicadores (http://www.w3.org/TR/xquery-operators/#flags).

Esta sección del manual esta compuesta por dos apartados:

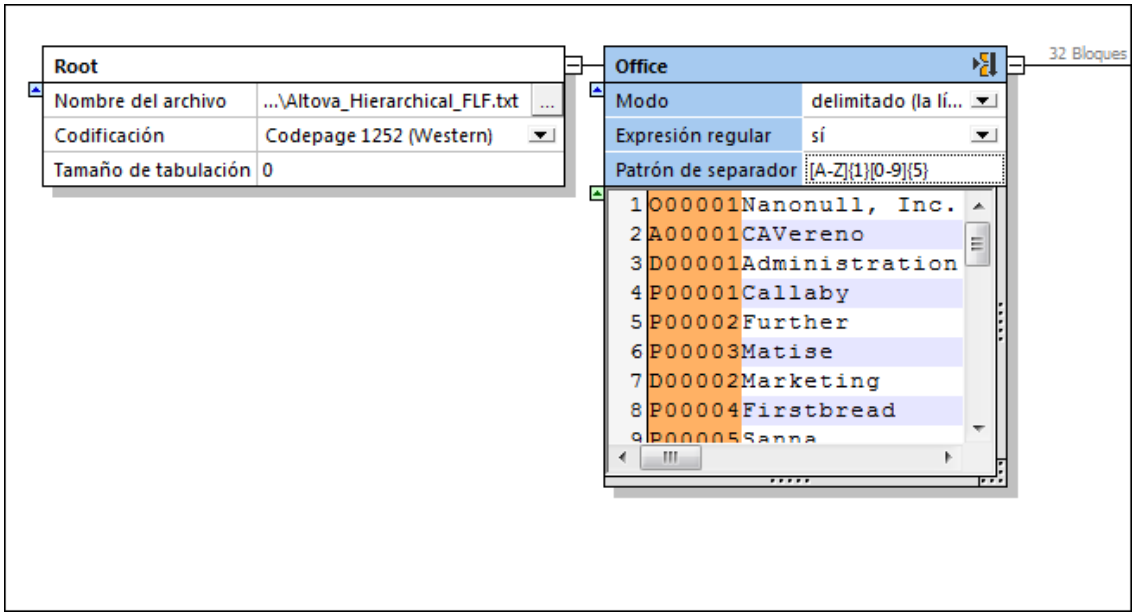
- Dividir texto con expresiones regulares
- Usar expresiones regulares en condiciones del conmutador

7.4.6.1 Dividir texto con expresiones regulares

Cuando necesite dividir texto en dos o más fragmentos, tiene la opción de usar una expresión regular como separador. Se trata de una técnica más avanzada que el simple uso de caracteres para dividir el texto.

La opción de dividir el texto por medio de expresiones regulares está disponible solamente si se cumplen estas condiciones en FlexText:

- El contenedor es de tipo **Dividir una vez** o **División repetida**.
- La opción *Modo* tiene el valor **delimitado (flotante)**, **delimitado (basado en línea)** o **delimitado (la línea empieza con)**.



Ejemplo de plantilla FlexText que usa expresiones regulares para dividir el texto

Las opciones relacionadas con el uso de expresiones regulares son:

Expresión regular	Para usar una expresión regular como separador debe elegir aquí el valor **sí**. Si, por el contrario, prefiere usar texto simple como separador, elija el valor **no** (valor predeterminado).
Patrón de separador	Si eligió el valor **sí** para la opción *Expresión regular,* se habilitará un recuadro con la opción *Patrón de separador.* Aquí puede introducir la expresión regular que debe hacer de separador. Por ejemplo, en la imagen anterior, la expresión regular `[A-Z]{1}[0-9]{5}` devuelve exactamente un carácter alfanumérico seguido de exactamente cinco caracteres numéricos (p. ej. "P00011"). Todas las coincidencias encontradas en el texto se resaltan en la vista previa y hacen de separador para dividir el texto.
Separador para escritura	Esta opción solamente es relevante si se cumplen estas dos condiciones: • El componente FlexText es un componente de destino. • La opción *Modo* es **delimitado (flotante)**.

 © 2018 Altova GmbH

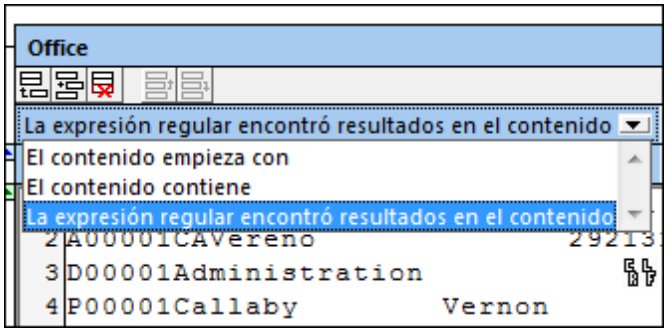
	En este campo debe introducir la cadena que se debe escribir en el componente de destino en lugar del resultado que devuelve la expresión regular.

7.4.6.2 Usar expresiones regulares en condiciones del conmutador

Cuando trabaje con contenedores Conmutador, tiene la opción de crear una condición dentro del Conmutador para poder redirigir el fragmento de texto en caso de que la expresión regular devuelva resultados. Para estos casos puede usar el carácter de acento circunflejo (^) y del signo del dólar ($) para señalar el principio o final del texto donde se deben buscar coincidencias (excepto si C++ es el lenguaje de transformación de destino porque en C++, los caracteres ^ y $ se interpretan como el principio o final de una línea y no del texto entero).

Para usar una expresión regular en una condición del conmutador:

1. Asegúrese de que el tipo de contenedor es Conmutador (si no lo es, haga clic en el icono interactivo del contenedor y elija **Conmutador**).
2. Haga clic en el botón **Anexar condición** para agregar una condición nueva.
3. Elija el tipo de condición **La expresión regular encontró resultados en el contenido** e introduzca a continuación la expresión regular en el recuadro adyacente.



Ejemplo
Imagine que necesita asignar datos de un archivo de registro de BD como el que aparece a continuación (la ruta de acceso de este archivo es **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\SampleDatabaseLog.txt**).

```
Action 18:11:51: INSTALL.
Action start 18:11:51: INSTALL.
Action 18:11:51: WindowsFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action start 18:11:51: WindowsFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: WindowsFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Return value 1.
Action 18:11:51: SystemFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action start 18:11:51: SystemFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: SystemFolder.68B7C6D9_1DF2_54C1_FF1F_C8B3B9A1E18E.
Return value 0.
Action 18:11:51: WindowsFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action start 18:11:51: WindowsFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
```

```
Action ended 18:11:51: WindowsFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Return value 0.
Action 18:11:51: SystemFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action start 18:11:51: SystemFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: SystemFolder.9BAE13A2_E7AF_D6C3_FF1F_C8B3B9A1E18E.
Return value 1.
Action 18:11:51: WindowsFolder.63E949F6_03BC_5C40_FF1F_C8B3B9A1E18E.
Action start 18:11:51: WindowsFolder.63E949F6_03BC_5C40_FF1F_C8B3B9A1E18E.
Action ended 18:11:51: WindowsFolder.63E949F6_03BC_5C40_FF1F_C8B3B9A1E18E.
Return value 1.
```

En este caso sus objetivos son dos:

1. Recopilar una lista de todas las filas que contienen un valor devuelto. Es decir, la lista debe incluir todas las filas que contienen el valor "Return value 1" o el valor "Return value 0" (o cualquier otro valor devuelto expresado en forma de dígito).
2. Recopilar en otra lista todas las filas cuyo texto contenga el valor "Action start".

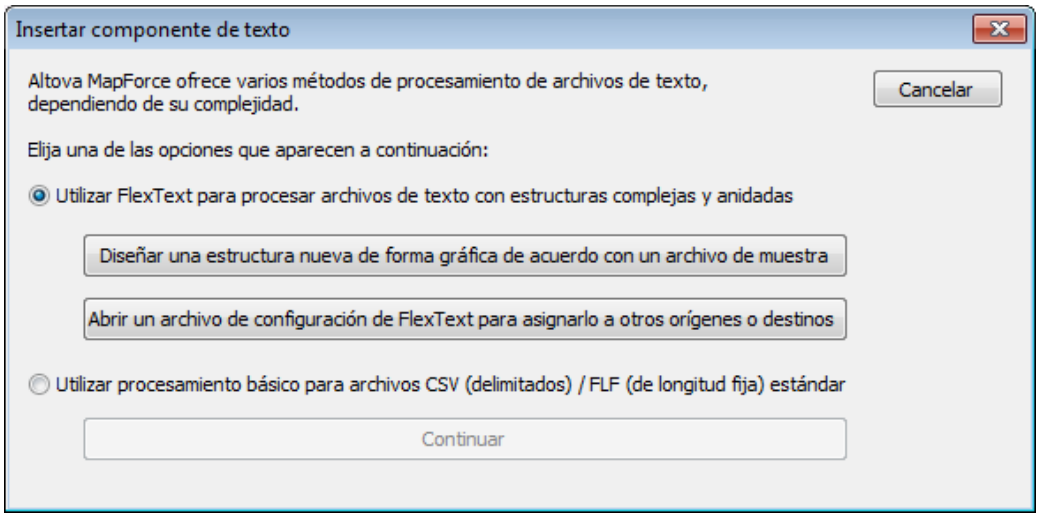
Para alcanzar estos dos objetivos puede usar un contenedor **División repetida** para dividir el texto en filas. Después, puede usar un contenedor **Conmutador** para redirigir cada fila según corresponda. Es decir, el contenedor **Conmutador** estará compuesto por tres condiciones:

1. Redirigir la fila actual al resultado A si contiene el valor "Action started". Estas filas se pueden encontrar con ayuda de una condición de tipo **El contenido empieza con** y asignándole el valor "Action started".
2. Redirigir la fila actual al resultado B si contiene un valor devuelto. Estas filas se pueden encontrar con ayuda de la expresión regular `Return value [0-9]\.` . Esta expresión regular devolverá resultados si la fila contiene el texto "Return value" seguido de un solo dígito y después un punto. La barra diagonal inversa (\ ) detona un carácter de escape (indicando que el punto debe tratarse como un carácter normal y no como metacarácter).
3. Redirigir la fila actual al resultado C (<predeterminado>) si la fila no cumple con ninguna de las condiciones anteriores.

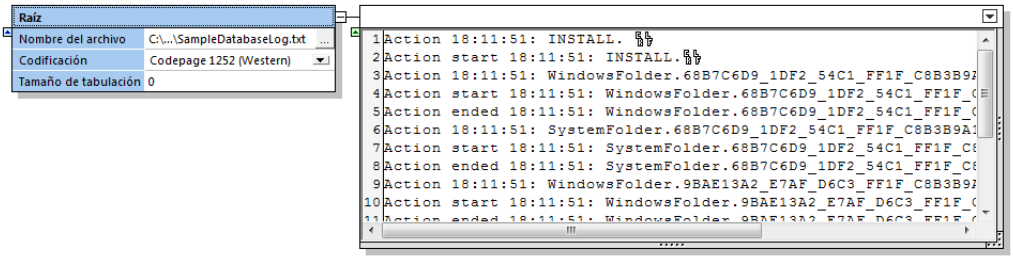
Para crear una plantilla FlexText que se ocupe de todas las tareas descritas anteriormente:

1. En MapForce haga clic en el comando de menú **Insertar | Archivo de texto**.

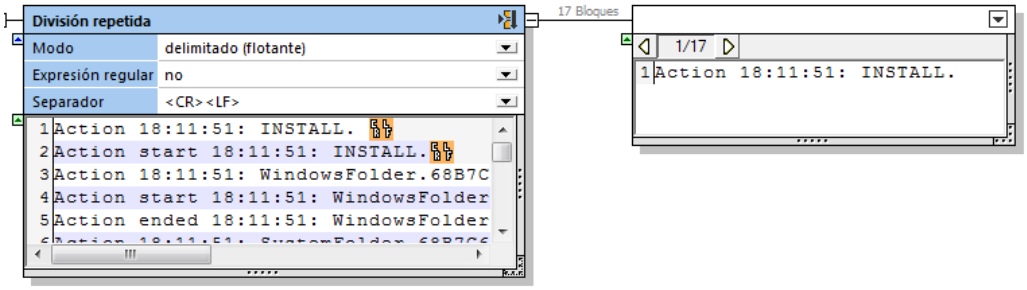
© 2018 Altova GmbH



2. Haga clic en el botón **Diseñar una estructura nueva de forma gráfica de acuerdo con un archivo de muestra** y guarde la plantilla FlexText (archivo .mft) en el directorio que quiera.
3. Cuando la aplicación le pida que abra un archivo, navegue hasta el archivo **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\SampleDatabaseLog.txt**.

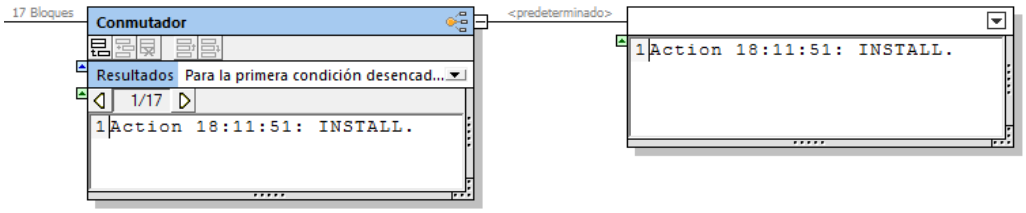


4. En el contenedor de salida haga clic en el botón interactivo y seleccione **División repetida**. Como nuestro carácter de división es el retorno de carro, elija el modo **delimitado (flotante)** e introduzca el separador **<CR><LF>**. Esto da lugar a un nuevo contenedor de salida compuesto por 17 bloques (uno por cada fila).

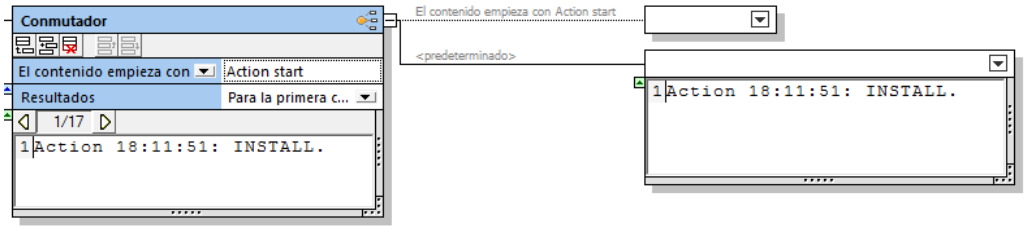


5. En este nuevo contenedor de salida haga clic en el botón interactivo y seleccione **Conmutador**. FlexText tratará el contenido de este contenedor como un conmutador. Como puede verse más abajo, la condición de conmutación `<predeterminado>` se crea automáticamente (esta condición redirige a un contenedor nuevo el texto que no cumpla

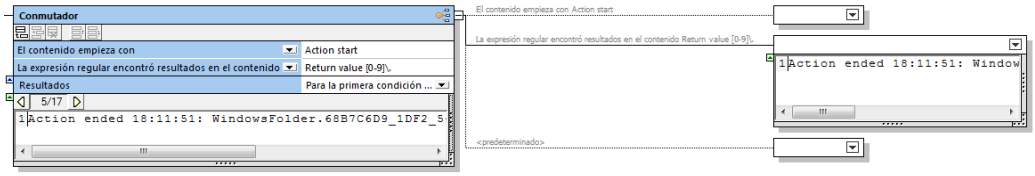
con ninguna de las condiciones). Ahora mismo no hay más condiciones así que todo el texto se redirige al contenedor de salida `<predeterminado>` por ahora.



6. Haga clic en el botón **Anexar condición** y añada una condición de tipo **El contenido empieza con** con el valor *Action start* (*imagen siguiente*). Esta condición redirige a un contenedor nuevo todo el texto que empiece con `Action start`.



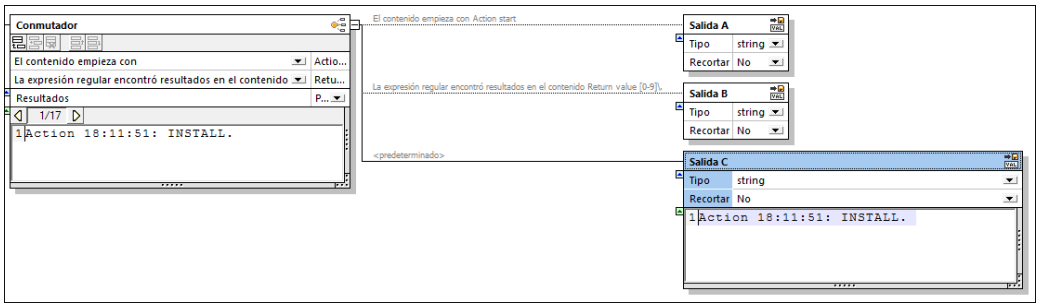
7. Haga clic en el botón **Anexar condición** y añada una condición de tipo **La expresión regular encontró resultados en el contenido** con el valor `Return value [0-9]\.` . Si ahora navegamos al bloque 5, veremos que esta condición redirige dicho bloque a un contenedor nuevo (porque la expresión regular devuelve resultados en este bloque).



El contenedor Conmutador ya está configurado para redirigir texto a diferentes contenedores de salida dependiendo de si se cumplen o no sus condiciones. Hay por tanto tres condiciones de conmutación y tres posibles resultados (uno por condición). Ahora solo nos falta escribir cada resultado en un archivo de texto distinto.

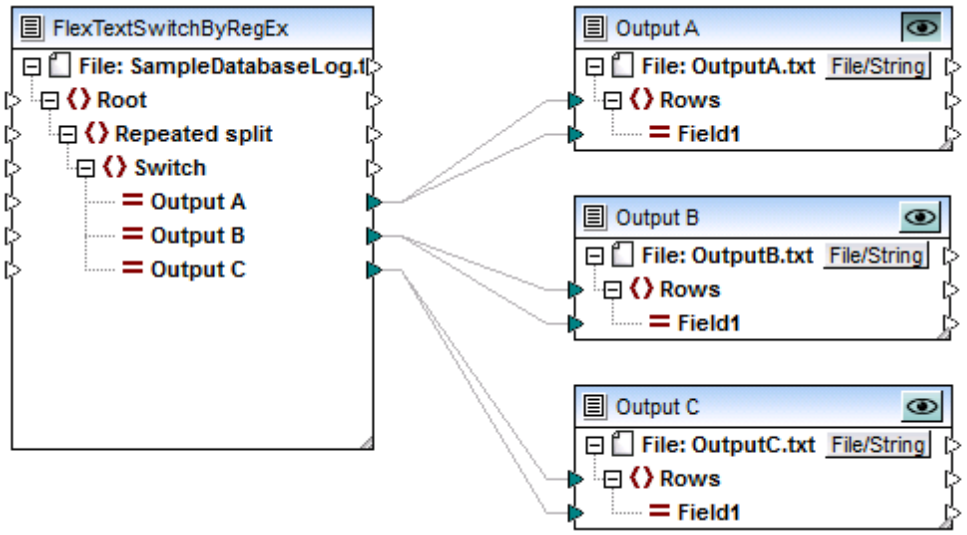
8. Haga clic en el botón interactivo de cada contenedor de salida y elija **Guardar valor**. Haga doble clic en la barra de título de cada contenedor e introduzca un nombre descriptivo como **Salida A**, **Salida B** y **Salida C**.

© 2018 Altova GmbH



9. Guarde y cierre la plantilla FlexText.

La plantilla FlexText que necesitábamos para la asignación de datos ya está lista. Para ver como funciona esta plantilla consulte el archivo del tutorial **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\FlexTextSwitchByRegEx.mfd**.



FlexTextSwitchByRegEx.mfd

La asignación **FlexTextSwitchByRegEx.mfd** que puede verse en la imagen anterior lee datos del archivo **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\SampleDatabaseLog.txt** y los distribuye a tres archivos de texto distintos: Output A, Output B y Output C. Cada archivo de texto de destino contiene solamente las filas que cumplen con una de las tres condiciones definidas previamente en la plantilla FlexText. Para ver el resultado de un componente de destino, active su botón **Vista previa** y después abra la pestaña *Resultados*.

7.5 EDI

EDI (intercambio electrónico de datos) es un grupo de estándares que permiten el intercambio electrónico de datos entre organizaciones y empresas.

MapForce permite traducir datos en formato EDI en estos lenguajes de transformación: BUILT-IN, C#, C++ y Java (*véase Seleccionar el lenguaje de transformación*). En MapForce puede crear asignaciones entre datos EDI y cualquier formato compatible, como bases de datos, archivos planos, documentos XML, etc.

MapForce es compatible con estos estándares EDI:

Estándar	Descripción
ASC X12	ASC X12 es un estándar sectorial para el intercambio de datos. MapForce es compatible con estas versiones: 3040, 3050, 3060, 3070, 4010, 4020, 4030, 4040, 4041, 4042, 4050, 4051, 4052, 4060, 5010, 5011, 5012, 5020, 5030, 5040, 5050, 6010, 6020, 6030 y 6040. **En MapForce la versión predeterminada de ASC X12 es la versión 6040.** En MapForce los componentes ASC X12 tienen nodos virtuales donde se escriben datos/información sobre errores del analizador EDI en función de la configuración seleccionada en el cuadro de diálogo "Configuración de la validación de EDI" (*véase Validación de componentes EDI*). Además se pueden generar confirmaciones funcionales X12 997 a partir de cualquier documento X12. Para más información sobre ASC X12 consulte el sitio web http://www.x12.org/.
HIPAA X12	HIPAA está basado en el estándar X12 EDI 5010 pero tiene versiones especiales con compatibilidad nativa en MapForce 2011 Release 3 o superior. **En MapForce la versión predeterminada de HIPAA X12 es la versión A2 de las especificaciones de implementación HIPAA (TR3).** Puede descargar versiones anteriores desde la página de componentes de MapForce del sitio web de Altova (https://www.altova.com/es/components_mapforce.html).
HL7	HL7 (Health Level Seven) es un estándar sectorial para el intercambio electrónico de información clínica. MapForce es compatible con las versiones 2.2 - 2.6. **En MapForce la versión predeterminada de HL7 es la versión 2.6.** Puede descargar los esquemas XML y archivos de configuración de las versiones HL7 V2.2 - V2.5.1 desde la página de componentes de MapForce

© 2018 Altova GmbH

Estándar	Descripción
	del sitio web de Altova (https://www.altova.com/es/components_mapforce.html). MapForce 2018 es compatible con la versión HL7 3.x por medio de componentes de esquema XML. El proyecto **MapForceExamples** incluye una asignación de ejemplo que asigna datos HL7 V2.6 a un archivo XML HL7 V3 XML (HL7V260_to_HL7V3.mfd). Para más información consulte el sitio web http://www.hl7.org/.
IATA PADIS	PADIS (Passenger and Airport Data Interchange Standards) es un estándar sectorial para el intercambio de datos sobre pasajeros y aeropuertos en documentos EDI. MapForce es compatible con las versiones 00.1 - 08.1. **En MapForce la versión predeterminada de PADIS es la versión 08.1.** Para más información consulte el sitio web http://www.iata.org/Pages/default.aspx.
NCPDP SCRIPT	SCRIPT es un estándar para el intercambio de información electrónica en el sector sanitario de EE UU desarrollado por el NCPDP (Consejo nacional para programas de medicamentos que requieren receta médica). Consulte el sitio web http://www.ncpdp.org. MapForce permite asignar datos entre archivos en formato SCRIPT y cualquier otro formato compatible con MapForce (como XML, CSV, Microsoft Excel, bases de datos, etc.). Las versiones de NCPDP SCRIPT compatibles con MapForce son las versiones 8.1, 10.6, 10.10. **En MapForce la versión predeterminada de NCPDP SCRIPT es la versión 10.10.**
SAP IDocs	Los documentos SAP IDoc (documentos intermedios) sirven para intercambiar datos comerciales entre aplicaciones SAP R/3 y aplicaciones no SAP. Los documentos están en un formato de almacenamiento intermedio que puede intercambiarse entre sistemas distintos. Para más información consulte el sitio web http://help.sap.com/saphelp_nw70/helpdata/en/0b/2a6095507d11d18ee90000e8366fc2/frameset.htm.
TRADACOMS	TRADACOMS (Trading Data Communications) es un estándar de intercambio electrónico de datos del Reino Unido utilizado en el comercio minorista. MapForce implementa la especificación TRADACOMS base siguiendo lo dispuesto en el manual "TRADACOMS Manual of Standards for Electronic Data Interchange", publicado en enero de 1993 por la asociación Article Numbering Association (ANA) UK, que ahora se conoce con el nombre GS1 UK (https://www.gs1uk.org). MapForce se puede personalizar para las demás versiones de TRADACOMS por medio de archivos de configuración, de modo que procese tipos de mensaje, elementos de datos y valores de

Estándar	Descripción
	código nuevos.
UN/EDIFACT	UN/EDIFACT (las reglas de las Naciones Unidas para el intercambio electrónico de datos sobre administración, comercio y transportes) es un conjunto de estándares para el intercambio electrónico de documentos (véase http://www.unece.org/cefact/edifact/welcome.html). MapForce es compatible con los mensajes incluidos en los directorios D93.A - D.17A. **En MapForce la versión predeterminada de UN/EDIFACT es la versión D.17A.** Desde el sitio web de Altova puede descargar un programa de instalación para instalar todas las versiones anteriores (hasta la versión D93.A), incluidos los archivos de configuración necesarios para asignaciones EDI interactivas. Para descargar estos archivos visite https://www.altova.com/es/components_mapforce.html.

La versión predeterminada de cada estándar EDI disponible en MapForce aparece resaltada en negrita en la tabla anterior. No obstante, si lo necesita, puede descargar e instalar versiones anteriores de los archivos de configuración EDI desde la página de componentes de MapForce del sitio web de Altova (https://www.altova.com/es/components_mapforce.html).

Por cada componente EDI que se añada al área de asignación, MapForce puede procesar varios tipos de mensajes de cualquier versión de estándar EDI. Por ejemplo, puede leer datos de un archivo de intercambio que incluya más de un tipo de mensaje o generar un archivo de intercambio que contenga varios tipos de mensajes. También puede procesar datos de varios archivos de instancia que contengan distintos tipos de mensaje. Recuerde que este tipo de operaciones solamente son relevantes si la especificación subyacente las admite.

Si su organización o empresa usa especificaciones EDI que sean personalizaciones de las compatibles con MapForce, puede adaptar la aplicación a su especificación EDI con ayuda de archivos de configuración. Por ejemplo, puede agregar mensajes personalizados, segmentos, elementos de datos, listas de código, etc. (o modificar los que ya tiene), creando así una colección EDI personalizada similar a la de los directorios EDI que ya son compatibles con MapForce. Para más información consulte los apartados *Personalizar estructuras EDI* y *Personalizar la validación de datos EDI* de esta sección.

7.5.1 Terminología EDI

Esta lista describe términos generales utilizados en los estándares EDI.

Mensaje

Un mensaje representa un documento comercial. En ASC X12 los mensajes se denominan *conjuntos de transacciones*. Un mensaje está compuesto por varios bloques que reciben el nombre de *segmentos*. MapForce es compatible con el uso de varios tipos de mensajes distintos en un solo componente EDI (*véase Agregar mensajes EDI como componentes de asignación*).

© 2018 Altova GmbH

Segmento

Un segmento representa un solo registro de un mensaje.

Cada segmento se identifica con un identificador de dos o tres caracteres, que aparece al principio del segmento. Un grupo de elementos relacionados forman una etiqueta de segmento (o Id. de segmento en ASC X12). Los segmentos de una transacción pueden ser de tipo obligatorio o condicional (opcional).

Elemento de datos

Un elemento de datos es un campo de datos específico ubicado dentro de un segmento. Los elementos de datos pueden entenderse como campos porque suelen contener un tipo de datos (p. ej. un nombre o una dirección). Los elementos de datos pueden ser simples o compuestos. Estos últimos están formados por elementos o subelementos componentes.

Separador

Los elementos de datos están delimitados por caracteres separadores. Estos separadores suelen ser caracteres ASCII, que aparecen con menos probabilidad en campos de texto (p. ej. dos puntos, asteriscos, barras verticales, Y comercial, etc.) Los separadores que se utilizan dependen del tipo de estándar EDI. MapForce reconoce por defecto los separadores estándar que corresponden a cada estándar EDI. No obstante, si lo necesita, puede personalizar los separadores (*véase* *Configuración de componentes EDI*).

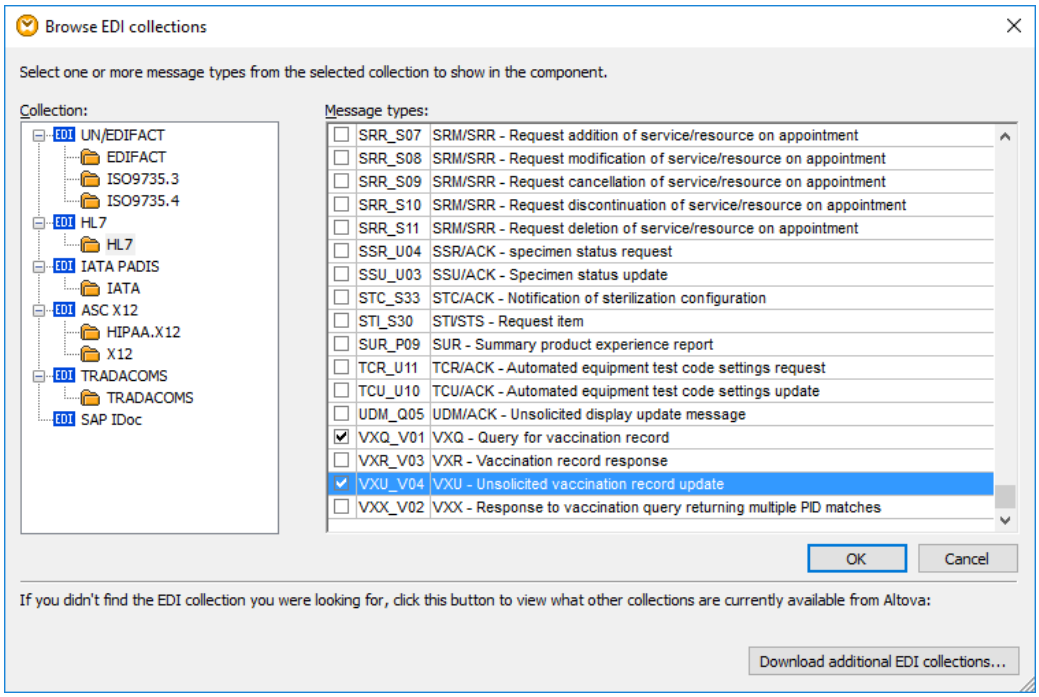
7.5.2 Agregar componentes EDI a la asignación

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

To map data to or from EDI files, the EDI component must be added to the mapping first. The EDI component represents the structure of the EDI message (or messages) on the mapping. Once the EDI component is added to the mapping, you can draw mapping connections to or from it, similar to any other component types supported by MapForce (see Working with Components). When necessary, you may also configure the EDI component to include multiple messages. This would enable you to read data from multiple EDI files (one message per file), as shown in Processing Multiple Input or Output Files Dynamically, or process different message types within the same file.

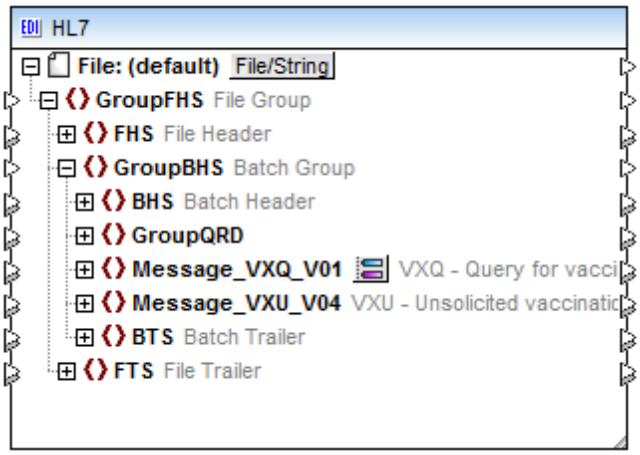
To add the EDI component to the mapping:

1. On the **Insert** menu, click **EDI**.
2. Click the EDI collection, and select the check box next to the message type that you want to include. You can also include multiple messages, by selecting their corresponding check boxes. For example, in the image below, two messages of the HL7 collection are being added to the mapping.



3. Click OK. You are now prompted to supply a sample EDI file.
4. If you want to read data from the EDI component, browse for the sample EDI file; otherwise, click **Skip**.

The component now appears on the mapping, and it includes the structure of both messages selected previously. If you want to add additional messages, or remove existing ones, click the button that appears next to the first message.



Sample HL7 mapping component which includes two messages

You can now draw mapping connections from this EDI component to a target one, or vice versa. Several MapForce mapping design (.mfd) files that illustrate mapping data to or from various EDI flavours are available in the **<Documentos>\Altova\MapForce2018\MapForceExamples** folder:

 © 2018 Altova GmbH

EDI Flavour	Mapping sample
ASC X12	• **X12_To_Xml_Order.mfd** • **Xml_To_X12_Order.mfd**
HIPAA X12	• **HIPAA_837D.mfd**
HL7	• **HL7_MultiMessageTypes.mfd** • **HL7V260_To_HL7V3.mfd**
IATA	• **IATA_FlightInformationReport.mfd**
SAP IDoc	• **IDOC_Order.mfd**
UN/EDIFACT	• **EDI_Order.mfd** • **EDI_Order_DB.mfd** • **EDI_OrderReverse.mfd**

7.5.3 Ejecutar asignaciones EDI

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

Mappings which contain EDI components are supported when the mapping is set to one of the following transformation language: BUILT-IN, C++, C#, Java. Once an EDI mapping has been designed in MapForce, it can be run in the following ways:

- The result of the data transformation can be previewed directly in MapForce, by clicking the **Output** tab. The output generated by the mapping will be saved to the desired location with the desired name, depending on how you configured both MapForce and the target component.
- The mapping can be run on a server machine (Windows, Linux, OS X / macOS) with MapForce Server, through an API call or at the command-line interface. This requires that the mapping be compiled as .mfx file and copied manually to the server machine, see Compiling Mappings to MapForce Server Execution Files.
- The mapping can be deployed to a server machine (Windows, Linux, OS X / macOS) through HTTP (or HTTPS) and run there as a scheduled or on demand job, or as a Web service, see Deploying Mappings to FlowForce Server.
- C++, C#, or Java program code can be generated from the mapping. The generated program performs the same result as if the mapping is previewed in MapForce; it can be run standalone or integrated into your existing code, see Code Generator.

To configure only the settings applicable to an EDI component (either source or target), right-click the component header, and select **Properties** from the context menu, see EDI Component Settings. For settings applicable to other component types, see Changing the Component Settings.

To configure the mapping settings for the entire mapping, right-click the mapping, and select **Mapping Settings** from the context menu, see Changing the Mapping Settings.

To access the code generation settings, go to **Tools | Options**, and then click the **Generation** tab.

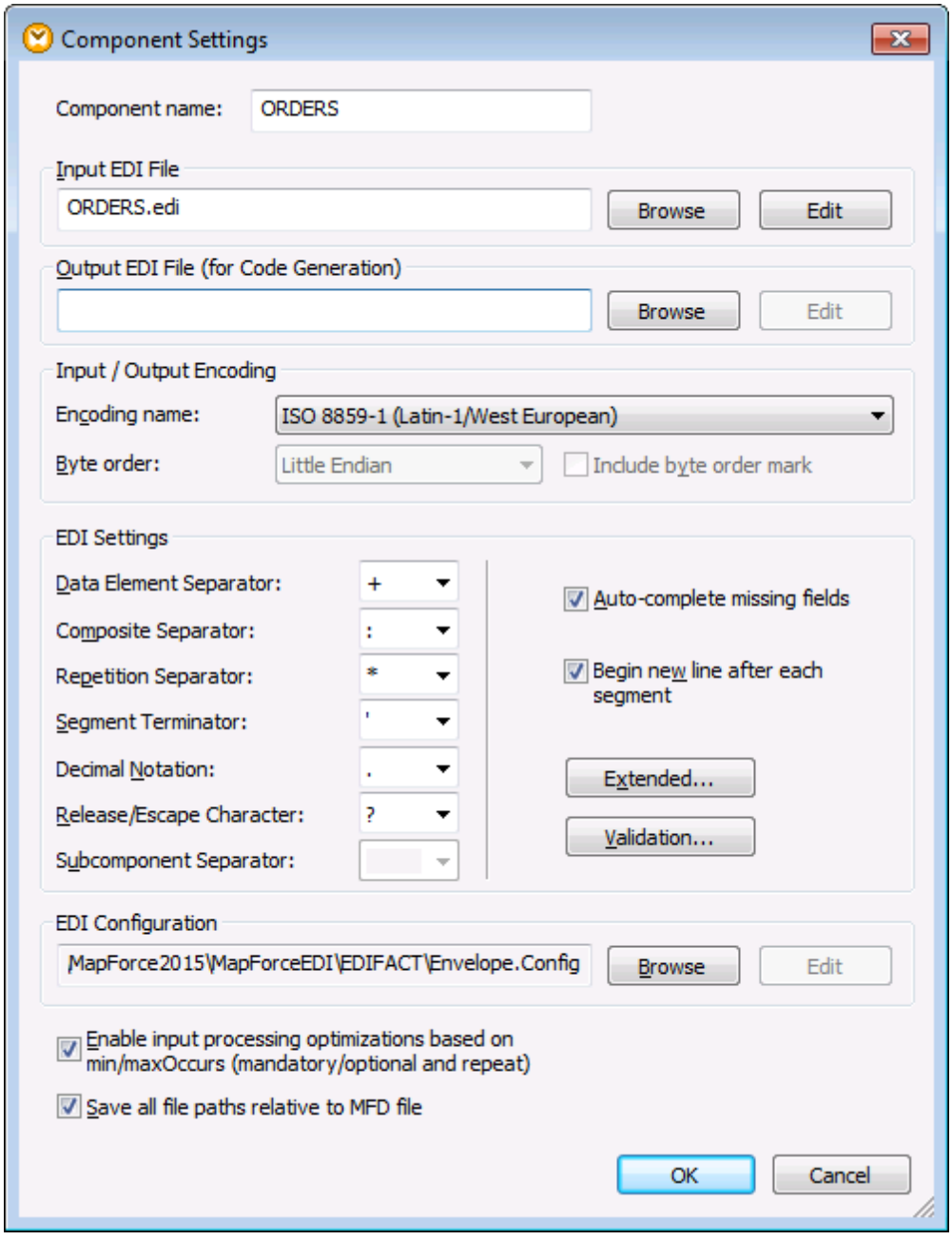
7.5.4 Configuración de componentes EDI

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

After you add an EDI component to the mapping area, you can configure the settings applicable to it from the Component Settings dialog box. You can open the Component settings dialog box in one of the following ways:

- On the **Component** menu, click **Properties** (this command is enabled provided that a component is selected on the mapping).
- Double-click the component header.
- Right-click the component header, and then click **Properties**.

© 2018 Altova GmbH



EDI Component Settings dialog box

The available settings are listed below. Note that some settings are not available if the EDI flavour of the currently selected component does not support them.

Component name	The component name is automatically generated when you create the component. You can however change the name at any time.

	If the component name was automatically generated and you select an instance file after that, MapForce will prompt you to optionally update the component name as well. The component name can contain spaces (for example, "Source XML File") and full stop characters (for example, "Orders.EDI"). The component name may not contain slashes, backslashes, colons, double quotes, leading or trailing spaces. In general, be aware of the following implications when changing the name of the component: • If you intend to deploy the mapping to FlowForce Server, the component name must be unique. • It is recommended to use only characters that can be entered at the command line. National characters may have different encodings in Windows and at the command line.
Input EDI file	Specifies the EDI source file from which MapForce will read data. This field is meaningful for a source component and is filled when you first create the component and assign to it an EDI instance file.
Output EDI file	Specifies the EDI target file to which MapForce will write data. This field is meaningful for a target component.
Input/Output Encoding	Allows you specify the following settings of the output instance file: • Encoding name • Byte order • Whether the byte order mark (BOM) character should be included. By default, any new components have the encoding defined in the **Default encoding for new components** option. You can access this option from **Tools | Options**, General tab.
EDI Settings	This group of settings enable you to define custom EDI delimiters, separators and terminators (note the settings are available only if supported by the EDI format). The EDI separators entered in this dialog box always take effect when writing EDI files. When reading in EDI files, the separators only take effect if the input file does not define/ contain its own separators (for example, EDIFACT files without the UNA "service string advice" segment). If an EDI input component/file contains separator definitions (for example, an X12 file with an ISA segment), then the existing separators override any separators defined in the Component Settings dialog box for that file.

© 2018 Altova GmbH

	You can use non-printable characters as separators by typing "x" followed by the hexadecimal ASCII character code into one of the combo boxes, for example, "x1e" for the RS control character (ASCII record separator, decimal code 30). The **Subcomponent Separator** applies to the HL7 standard. The default value of this separator is ampersand (**&**).
Auto-complete missing fields	This option applies to target EDI components. When this check box is selected, MapForce fills in the values of some data fields automatically. This applies only to those fields where this operation would not contradict the specification of the currently selected EDI format. To disable such behaviour, clear this check box. See also: • ASC X12 Automatic Data Completion Rules • UN/EDIFACT Automatic Data Completion Rules
Begin new line after each segment	This option applies to target EDI components. When the check box is selected, MapForce adds a CR/LF (carriage return / line feed) character after each EDI segment. The EDI standard ignores these lines if present in a message.
Extended	Opens a dialog box where you can define additional settings for the current EDI component. The available settings vary by EDI flavour.
Validation	Opens a dialog box where you can define the validation settings for the current EDI component (see EDI Component Validation).
Enable input processing optimizations based on min/ maxOccurs	This option allows special handling for sequences that are known to contain exactly one item, such as required attributes or child elements with minOccurs and maxOccurs="1". In this case the first item of the sequence is extracted, then the item is directly processed as an atomic value (and not as a sequence). If the input data is **not valid** against the schema, an empty sequence might be encountered in a mapping, which stops the mapping with an error message. To allow the processing of such **invalid input**, clear this check box.
Save all file paths relative to MFD file	When this option is enabled, MapForce saves the file paths displayed on the Component Settings dialog box relative to the location of the MapForce Design (.mfd) file. This setting affects the following files: • The input EDI file (if present)

	• The output EDI file (if present) See also Using Relative Paths on a Component.

7.5.5 Validación de componentes EDI

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

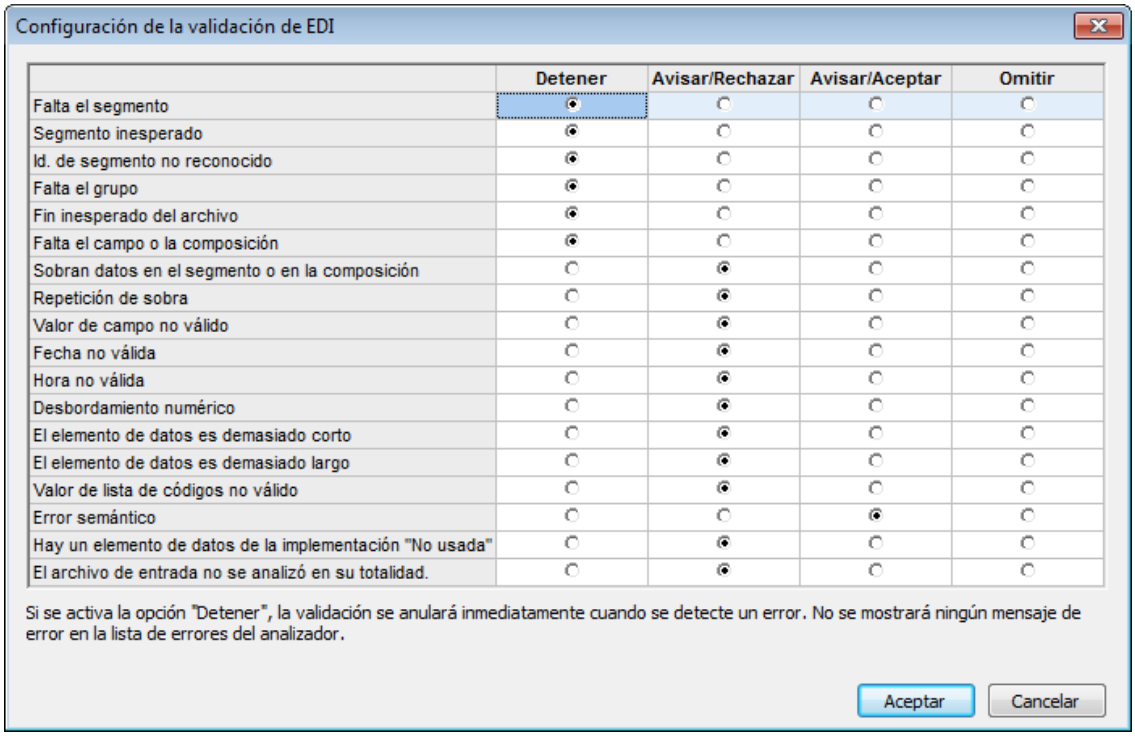
MapForce validates all supported EDI source and target documents when the mapping is executed. This happens either when you preview the mapping in MapForce, by clicking the **Output** tab, or when the mapping is run by MapForce Server, or by a C++, C#, or Java program. You can also validate explicitly the EDI output generated by the mapping. To do this, first click the **Output** tab, and then click the **Validate Output**  toolbar button (see also Validating the Mapping Output).

MapForce performs validation of EDI documents as specified in this documentation for each individual supported format. The validation process is, however, not a full EDI syntax or semantic validation. For example, there may exist instances where the possible values of a field rely on standards external to EDI and cannot be easily expressed as a finite set of values that would foresee all processing possibilities. For such cases, it is possible to customize the validation rules on a case-by-case basis with the help of configuration files (see Customizing EDI Validation).

At a more simple level, some validation options can be configured from the MapForce graphical user interface. For example, you can choose to ignore specific errors, or stop the mapping execution when specific errors are encountered.

To customize the validation settings, double-click an EDI component and click the **Validation** button in the Component Settings dialog box (see also EDI Component Settings). The default validation settings are shown below.

© 2018 Altova GmbH



EDI Validation Settings dialog box

For each validation event listed above, you can take any of the following actions: **Stop**, **Report & Reject**, **Report & Accept**, **Ignore**.

Stop is used to catch fatal errors and stops the execution of the mapping.

Report & Reject provides information about the error and rejects the current message. **Report & Accept** provides information about the error and accepts the current message. The actual error information is visible in the Messages window of MapForce. For source X12 and HIPAA components, the validation errors are also logged inside the mapping component, which enables you to generate X12 997 or X12 999 reports (see Validation Results and X12 Acknowledgement).

Ignore ignores the validation event. If you select this action, no information is provided about the error, so use it cautiously.

If unexpected segments are encountered while processing an EDI mapping, only MapForce and MapForce Server have the capability to recover and attempt to continue parsing. The generated C++, C#, or Java code cannot recover from unexpected segments and will stop processing. Unexpected segments include the following: incorrect segment order, missing mandatory segment, incorrect segment name, additional segments that are not present in the configuration files.

The EDI validation events for which you can configure custom actions (**Stop**, **Ignore**, etc) are as follows.

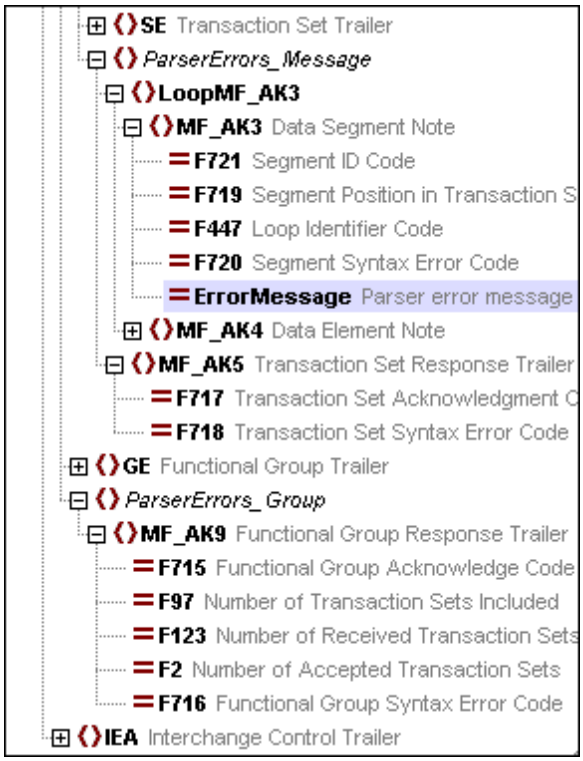
Evento	Descripción	Puede darse un error en
Falta el segmento	Falta un segmento obligatorio o aparece menos veces que el mínimo especificado.	componente de origen
Segmento inesperado	Un segmento está definido en la especificación pero no en este mensaje.	componente de origen
Id. de segmento no reconocido	Se encontró un segmento que no está definido en la especificación.	componente de origen
Falta el grupo	Falta un grupo obligatorio o aparece menos veces que el mínimo especificado.	componente de origen
Fin inesperado del archivo	La instancia no se puede analizar porque faltan datos.	componente de origen
Falta el campo o la composición	Falta un campo o una composición obligatorios o aparecen menos veces que el mínimo especificado.	componente de origen o destino
Sobran datos en el segmento o en la composición	La instancia de entrada contiene datos extra que no se esperaban en base a la descripción de sintaxis.	componente de origen
Repetición de sobra	El número real de campos que forman parte del segmento/de la composición es superior al máximo especificado.	componente de origen o destino
Valor de campo no válido	Un campo numérico contiene un carácter no válido.	componente de origen
Fecha no válida	Un campo de fecha contiene un carácter no válido o el valor del mes/día no es válido.	componente de origen
Hora no válida	Un campo de hora contiene un carácter no válido o el valor de las horas/los minutos no es válido.	componente de origen
Desbordamiento numérico	Un valor numérico desborda su dominio definido. Este error solamente es compatible dentro del código generado.	componente de origen
El elemento de datos es demasiado corto	La longitud del elemento de datos es menor que el valor mínimo especificado.	componente de origen o destino
El elemento de datos es demasiado largo	La longitud de un elemento de datos es mayor que el máximo especificado.	componente de origen o destino
Valor de lista de códigos no válido	El valor del elemento de datos no coincide con ninguno de los valores de código preconfigurados. Ver también *Cambiar*	componente de origen o destino

© 2018 Altova GmbH

Evento	Descripción	Puede darse un error en
	posibles valores de código de elementos de datos.	
Error semántico	Se produjo un error semántico.	componente de origen o destino
Hay un elemento de datos de la implementación "No usada"	En el archivo de entrada existe un elemento que no está permitido por el archivo de configuración HIPAA (maxOccurs="0").	componente de origen
El archivo de entrada no se analizó en su totalidad.	No se terminó de analizar el archivo de entrada completamente.	componente de origen

Validation results and X12 Acknowledgment

When MapForce validates an X12 or HIPAA component, the validation results are placed in "virtual" items at the base of the EDI component (under "ParserError_Message" and "ParserErrors_Group", as shown in the image below). This information can be used to generate X12 997 or X12 999 files that report the status of the interchange (see Generating an X12 997 Functional Acknowledgment and Generating an X12 999 Implementation Acknowledgment).



With X12 or HIPAA.X12 components, the validation actions available on the EDI Validation Settings dialog box (discussed above) have the following meaning:

Stop will stop the execution of the mapping without generating a report message.

Report & Reject and **Report & Accept** will provide information in the "ParserErrors_Message" and "ParserErrors_Group" items of the EDI component which can be mapped further.

Depending on the setting **Reject** or **Accept,** the "Functional Group Acknowledge Code F715" and the "Transaction Set Acknowledgment Code F717" will contain either:
the value 'R' for *'Rejected'* or
the value 'E' for '*Accepted, but errors were noted*'. The errors also appear in the Messages window.

Ignore ignores the specific error. No information is provided within the "ParserErrors_Message" and "ParserErrors_Group" items.

7.5.6 ASC X12

ASC X12 es un estándar sectorial para el intercambio de datos. MapForce es compatible con estas versiones: 3040, 3050, 3060, 3070, 4010, 4020, 4030, 4040, 4041, 4042, 4050, 4051, 4052, 4060, 5010, 5011, 5012, 5020, 5030, 5040, 5050, 6010, 6020, 6030 y 6040.

En MapForce la versión predeterminada de ASC X12 es la versión 6040.

En MapForce los componentes ASC X12 tienen nodos virtuales donde se escriben datos/ información sobre errores del analizador EDI en función de la configuración seleccionada en el cuadro de diálogo "Configuración de la validación de EDI" (*véase Validación de componentes EDI*). Además se pueden generar confirmaciones funcionales X12 997 a partir de cualquier documento X12.

Para más información sobre ASC X12 consulte el sitio web http://www.x12.org/.

7.5.6.1 Reglas de validación ASC X12

Cuando se validan documentos ASC X12, MapForce comprueba:

- si existen los segmentos ISA e IEA.
- si ISA/I01 contiene un calificador de información de autorización legal.
- si ISA/I03 contiene un calificador de información de seguridad legal.
- si los dos segmentos ISA/I05 contienen calificadores de Id. de intercambio legales.
- si ISA/I08 contiene un valor de fecha con un formato correcto.
- si ISA/I09 contiene un valor de hora con un formato correcto.
- si ISA/I13 contiene un valor booleano legal.
- si ISA/I14 contiene un indicador de uso de intercambio legal.
- si ISA/I12 e IEA/I12 contienen el mismo valor.
- si IEA/I16 contiene el número correcto de grupos funcionales del intercambio.

En cada **grupo funcional** se comprueba:

- si contiene el par GS y GE correspondiente.
- si GS/373 contiene un formato de fecha con un formato correcto.
- si GS/337 contiene un formato de hora con un formato correcto.
- si GS/28 y GE/28 contienen el mismo valor.

© 2018 Altova GmbH

- si GE/97 contiene el número correcto de mensajes del grupo funcional.

En cada **mensaje** se comprueba:

- si contiene el par ST y SE correspondiente.
- si ST/143 contiene el identificador de mensaje correcto.
- si ST/329 y SE/329 contienen el mismo valor.
- si SE/96 contiene el número correcto de segmentos del mensaje.

7.5.6.2 Reglas de finalización automática de datos ASC X12

Cuando genera archivos ASC X12, MapForce puede crear automáticamente ciertos campos o valores, de acuerdo con las reglas que se describen más abajo (en adelante "reglas de finalización automática de datos"). Para deshabilitar este comportamiento debe desactivar la casilla *Rellenar automáticamente los campos que falten* en el cuadro de diálogo "Configuración del componente" (*véase* *Configuración de componentes EDI*).

La **finalización automática de datos** para EDI/X12 se asegura:

- de que el par ISA y IEA exista en el nivel de intercambio.
- de que, si existe GS o GE, el otro ID exista también.
- de que haya como mínimo un par ST/SE en el nivel de mensaje.
- de que ISA/I01 y ISA/I03 existan. Si no contienen datos, se inserta 00.
- de que ISA/I02 y ISA/I04 existan. Si no contienen datos, se insertan diez espacios en blanco.
- de que existan ambos segmentos ISA/105. Si no contienen datos, se inserta ZZ.
- de que exista ISA/I08. Si no contiene datos, se inserta la fecha actual en formato EDI.
- de que exista ISA/I09. Si no contiene datos, se inserta la hora actual en formato EDI.
- de que exista ISA/I65. Si no contiene datos, se inserta el separador de repeticiones.
- de que exista ISA/I11. Si no contiene datos, se inserta el número de versión de control del intercambio definido por el usuario. (Botón **Configuración extendida | Número de versión de control del intercambio** del cuadro de diálogo Configuración del componente EDI.)
- de que exista ISA/I12.
- de que exista ISA/I13. Si no contiene datos, se utiliza la opción de solicitud de confirmación. (Botón **Configuración extendida | Solicitar confirmación** del cuadro de diálogo Configuración del componente EDI.)
- de que exista ISA/I14. Si no contiene datos, se inserta P.
- de que exista ISA/I15. Si no contiene datos, se inserta el separador de composiciones definido por el usuario (opción *Separador de composiciones* del cuadro de diálogo Configuración del componente EDI.)
- de que exista IEA/I16. Si no contiene datos, se calcula e inserta el número de grupos funcionales del intercambio.
- de que exista IEA/I12. Si no contiene datos, se copia el valor de ISA/I12.

Nota: los campos que no aparecen en esta lista ni se insertan ni se crean. Los valores correctos no se pueden determinar automáticamente. Dado un elemento primario **A** (en el componente EDI de destino) con los elementos secundarios **x**, **y**, y **z**, siendo **z** obligatorio, el elemento primario **A** solamente se creará en el archivo de salida si el elemento secundario obligatorio **y** tiene una asignación de datos.

En el grupo funcional potencial se comprueba:

- si existe GS/373. Si no contiene datos, se inserta la fecha actual en formato EDI.
- si existe GS/337. Si no contiene datos, se inserta la hora actual en formato EDI.
- si existe GE/97. Si no contiene datos, se calcula e inserta el número de mensajes del grupo funcional.
- si existe GE/28. Si no contiene datos, se copia el valor de GS/28.

En el mensaje se comprueba:

- si existe ST/143. Si no contiene datos, se inserta el nombre del mensaje.
- si existe SE/96. Si no contiene datos, se calcula e inserta el número de segmentos del mensaje.
- si existen ST/329 y SE/329. Si SE/329 no contiene datos, se copia el valor de ST/329.

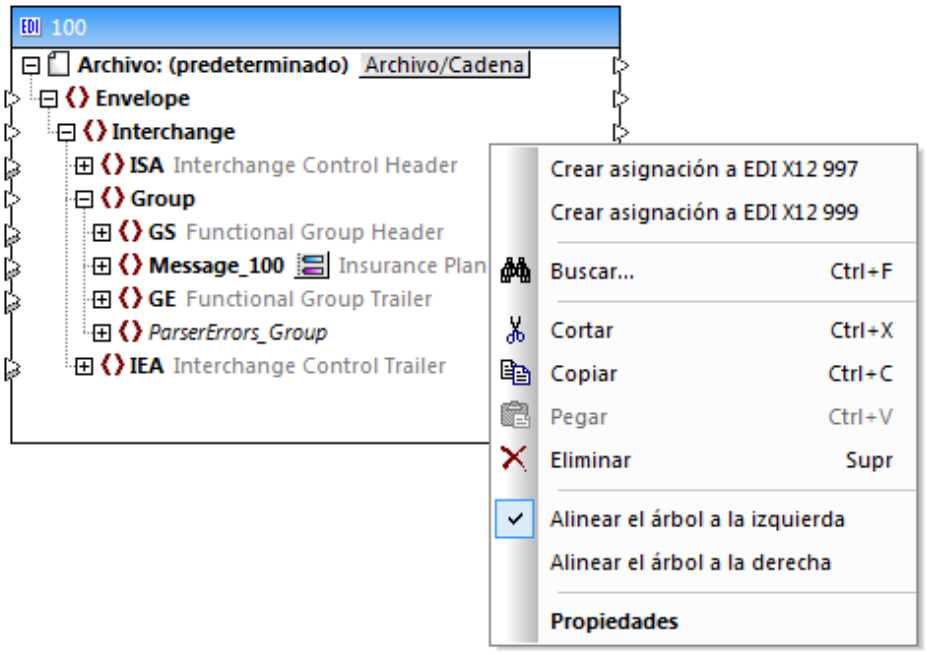
7.5.6.3 Generar una confirmación funcional X12 997

La confirmación funcional 997 de X12 informa del estado del intercambio EDI. En ella se incluyen todos los errores detectados durante el procesamiento del documento. MapForce puede generar automáticamente un componente 997 de X12 en el panel *Asignación* y crear las conexiones necesarias.

Para generar la confirmación funcional X12 997:

1. En el menú **Insertar** haga clic en **EDI** y añada el componente X12 al área de asignación. Cuando la aplicación solicite un archivo de instancia, busque el archivo para el que desea crear la confirmación X12 997.
2. Haga clic con el botón derecho en el componente EDI de origen y seleccione **Crear asignación a EDI X12 997** (este comando solamente se habilita para componentes X12).

© 2018 Altova GmbH



Esto crea un componente de asignación EDI 997 y crea automáticamente todas las conexiones necesarias para generar la confirmación X12 997. Para ver un ejemplo, consulte el archivo de diseño de asignación **X12_To_XML_Order.mfd** de la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples**. Para ver el archivo de confirmación que genera esta asignación haga clic en el botón **Vista previa** del componente EDI 997 y después abra el panel *Resultados*.

```
1   ISA+00+       +00+       +ZZ+ReceiverID    +ZZ+SenderID      +100128+1113+!+00505+000000000+1+P+:'
2   GS+01+ReceiverID+SenderID+20100128+11130544+1+X+05012'
3   ST+997+0001'
4   AK1+PO+1+05012'
5   AK2+850+12345'
6   AK5+A'
7   AK9+A+1+1+1'
8   SE+6+0001'
9   GE+1+1'
10  IEA+1+000000000'
11
```

Para guardar el archivo de confirmación X12 997:

- Abra el panel *Resultados* y después haga clic en el botón **Guardar resultado generado** de la barra de herramientas. Si tiene pensado generar programa de código a partir de la asignación, introduzca un nombre de archivo en el campo *Archivo EDI de salida* del cuadro de diálogo "Configuración del componente" (*véase* *Configuración de componentes EDI*).

7.5.6.4 Generar una confirmación de implementación X12 999

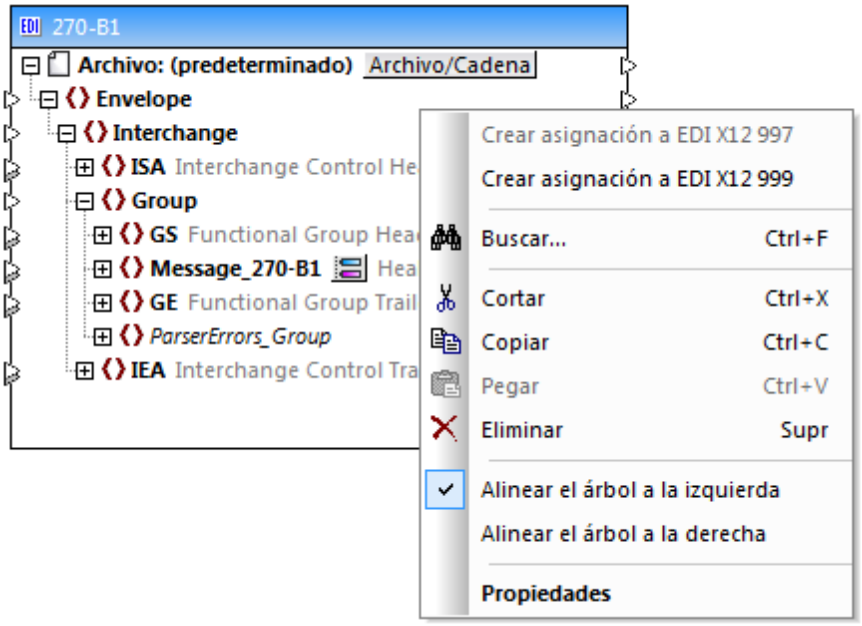
El conjunto de transacciones X12 999 (confirmación de implementación) informa de errores de aplicación o de incumplimiento.

Al tratarse de un supraconjunto de la confirmación funcional 997, la confirmación 999 puede utilizarse en lugar de 997 para aceptar o rechazar conjuntos de transacciones en base a los requisitos sintácticos de la guía de implementación HIPAA o X12.

Todos los errores detectados durante el procesamiento del documento se incluyen en esta confirmación. Por ejemplo, *Required Segment Missing*, *Required Data Element Missing*, *Code Value Not Used in Implementation*, etc. MapForce puede generar automáticamente el componente X12 999 en la asignación y crear las conexiones de asignación necesarias.

Para generar la confirmación de implementación X12 999:

1. En el menú **Insertar** haga clic en **EDI** y añada el componente X12 o HIPAA al área de asignación. Cuando la aplicación solicite un archivo de instancia, busque el archivo para el que desea crear la confirmación X12 999.
2. Haga clic con el botón derecho en el componente EDI de origen y seleccione **Crear asignación a EDI X12 999** (este comando solamente se habilita para componentes X12 e HIPAA).



Esto crea un componente de asignación EDI 999 y crea automáticamente las asignaciones de datos necesarias para generar la confirmación de implementación X12 999. Para ver el archivo de confirmación que genera esta asignación abra el panel *Resultados*. Si la asignación tiene varios componentes de destino, primero haga clic en el botón **Vista previa** del componente EDI 999 y después abra el panel *Resultados*.

© 2018 Altova GmbH

```
1   ISA+00+       +00+       +ZZ+ReceiverID    +ZZ+SenderID      +110502+1533+!+00505+000000000+1+P+:'
2   GS+FA+ReceiverID+SenderID+20110502+15333751+1+X+006020'
3   ST+999+0001'
4   AK1+PO+1+006020'
5   AK2+850+12345'
6   IK5+A'
7   AK9+A+1+1+1'
8   SE+6+0001'
9   GE+1+1'
10  IEA+1+000000000'
11
```

Para guardar el archivo de confirmación 999:

- Abra el panel *Resultados* y después haga clic en el botón **Guardar resultado generado**  de la barra de herramientas. Si tiene pensado generar programa de código a partir de la asignación, introduzca un nombre de archivo en el campo *Archivo EDI de salida* del cuadro de diálogo "Configuración del componente" (*véase* *Configuración de componentes EDI*).

7.5.7 HIPAA X12

HIPAA X12 es la última versión del estándar para historiales clínicos electrónicos. Este estándar fue elaborado por el ministerio de salud y servicios humanos de EE UU para la transacción electrónica de datos clínicos entre empresas aseguradoras y empleados y está basado en EDI X12 versión 5010.

MapForce es compatible con la versión más reciente de HIPAA X12, la versión A2. Puede descargar las versiones más antiguas en un archivo ZIP desde el sitio web de Altova.

Los componentes HIPAA son similares a los componentes ANSI X12 y MapForce es compatible con estas transacciones:

Nombre X12	Nombre del mensaje
X279A1	"Health Care Eligibility Benefit Inquiry (270)"
X279A1	"Health Care Eligibility Benefit Response (271)"
X212	"Health Care Claim Status Request (276)"
X212	"Health Care Information Status Notification (277)"
X214	"Health Care Claim Acknowledgment (277)"
X217	"Health Care Services Review - Request for Review (278)"
X217	"Health Care Services Review - Response (278)"
X218	"Payroll Deducted and Other Group Premium Payment for Insurance Products (820)"

X220A1	"Benefit Enrollment and Maintenance (834)"
X221A1	"Health Care Claim Payment/Advice (835)"
X222A1	"Health Care Claim: Professional (837)"
X224A2	"Health Care Claim: Dental (837)"
X223A2	"Health Care Claim: Institutional (837)"
X231A1	"Implementation Acknowledgment For Health Care Insurance (999)"

La diferencia con los mensajes X12 estándar es que MapForce:

- conserva automáticamente la jerarquía de los segmentos HL
- es compatible con las llamadas estructuras flotantes (de 837 mensajes)
- finaliza automáticamente y valida más campos

MapForce también puede generar confirmaciones de implementación HIPAA X12 999 automáticamente. Esta confirmación de implementación es similar a la confirmación de implementación 999 del estándar X12 (*véase* *Generar una confirmación de implementación X12 999*).

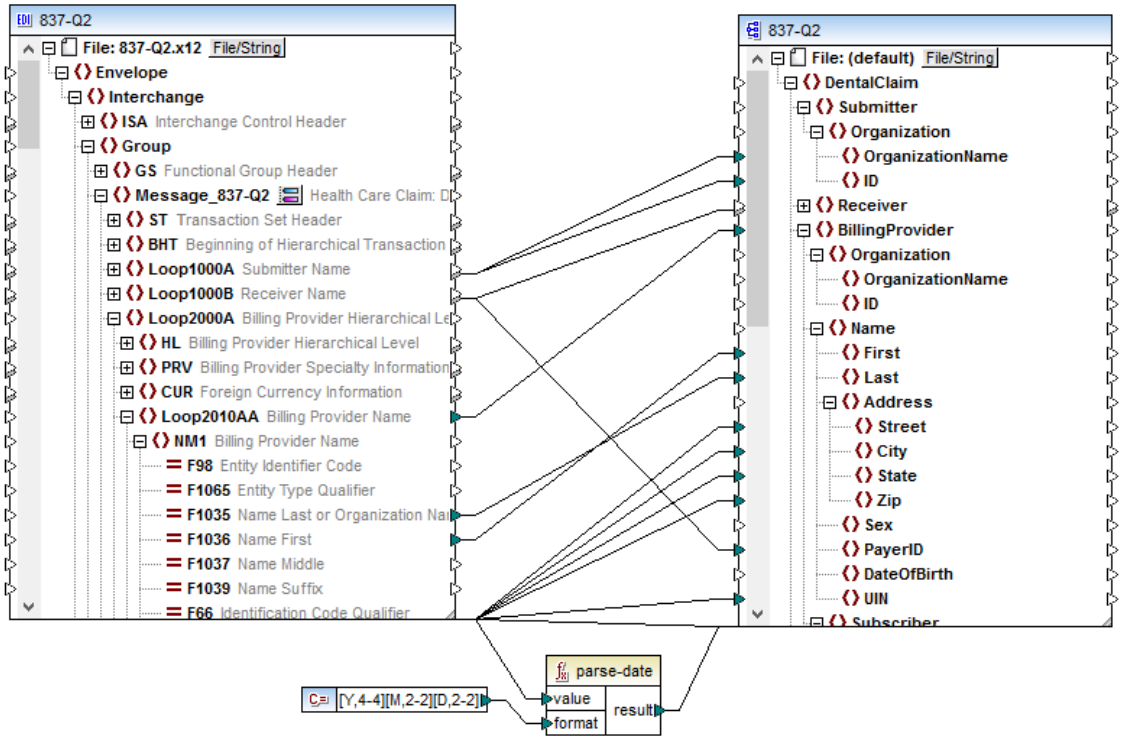
En los componentes HIPAA también se pueden usar varios tipos de mensajes diferentes por cada intercambio.

7.5.7.1 Ejemplo: asignar datos HIPAA X12 a XML

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

This example illustrates how to map a file which contains an "837-Q2 Health Care Claim: Dental" transaction to an XML target file. It is accompanied by a mapping available at the following path: **<Documentos>\Altova\MapForce2018\MapForceExamples\HIPAA_837D.mfd**.

© 2018 Altova GmbH



HIPAA_837D.mfd

The mapping can be created as shown in the steps below.

Step 1: Add the source HIPAA X12 file to the mapping

1. On the **Insert** menu, click **EDI**.
2. Select the collection "HIPAA.X12", and the message type "837-Q2 Health Care Claim: Dental".
3. When prompted to supply an instance file, open the source file available at the following path: **<Documentos>\Altova\MapForce2018\MapForceExamples\837-Q2.x12**.
4. When prompted, adjust the settings of the newly added HIPAA X12 component if necessary (for more information, see EDI Component Settings).

Step 2: Add the target schema to the mapping

1. On the **Insert** menu, click the **XML Schema/File**, and open the **837-Q2.xsd** file from the **<Documentos>\Altova\MapForce2018\MapForceExamples** directory.
2. When prompted to supply a sample XML file, click **Skip** and select **DentalClaim** as the root of the target document.

Step 3: Add a function to format the date and draw the mapping connections

In the source HIPAA file, the DMG segment includes a date, in the format **`19490501`**. To convert

this date to a format suitable for the target XML file `1949-05-01`, we will add a date conversion function, as follows:

1. Drag the **parse-date** function from the Libraries window into the mapping. For more information about functions, see Working with Functions.
2. On the **Insert** menu, click **Constant**, and enter `[Y,4-4][M,2-2][D,2-2]` as constant value.
3. Draw the following connections:

From	To	Purpose
Constant	`format` input item of the **parse-date** function	Supplies to the function the format according to which the date should be parsed.
`F1251`	`value` input item of the **parse-date** function	Supplies to the function the actual value to be formatted.
`result` output item of the **parse-date** function	`DateOfBirth` input item of the XML target	Supplies the value returned by the function to the `DateOfBirth` XML node.

4. Draw the remaining mapping connections as shown in the example mapping (these are too many to be listed here; essentially, each connection copies values from the source HIPAA file to the target XML file. For basic information about mapping connections, see Working with Connections.)

7.5.7.2 Nota sobre asignación de transacciones HIPAA

HL segment hierarchy

HL segments are generated automatically and do need to be mapped manually. It is however possible to map values if there is such a need.

Autocompletion

The configuration files for HIPAA transactions contain many codes and are used for auto-completion of fields whenever possible. Autocompletion is context sensitive, so fields in different places may have different values. To disable auto-completion, clear the **Auto-complete missing fields** check box in the Component Settings dialog box (see EDI Component Settings).

Validation

Validation is similar to auto-completion in that it is also context sensitive. Validation errors are handled in the same way as per X12 validation (see ASC X12 Validation Rules).

Editing of configurations files

HIPAA configuration files uses a newer schema version than the X12 configuration files (see EDI Configuration Schemas). Therefore, HIPAA configuration files are similar to X12 configuration files, with a few additions:

© 2018 Altova GmbH

- Code values at field level
- Conditions
- Completion flags
- “Not Used” Data Elements

Code values

Field instances can have a built-in code list which is used for validation and/or autocompletion. Values can be manually added or removed (see Changing Possible Code Values of Data Elements).

```
<Data ref="F365" info="Communication Number Qualifier" nodeName="F365_1">
<Values>
        <Value Code="EM" />
        <Value Code="FX" />
        <Value Code="TE" />
</Values>
</Data>
```

Conditions

Conditions work in conjunction with values. As there can be multiple repetitions of a segment, or group (loop), they are identified by a specific condition. A Condition requires a value to be present for it to be fulfilled, otherwise the specific group is not found.

For example, Loop1000 is repeated multiple times, but each instance has a different semantic meaning.

To tell which is which, MapForce uses a condition that specifies that Loop1000A must have a Value Code of “41”, in Field F98 of segment NM1. If this is not the case, then Group name Loop1000A is not found.

```
<Group name="Loop1000A" info="Submitter Name">
  <Segment name="NM1" info="Submitter Name">
    <Condition path="F98" />
    <Data ref="F98" info="Entity Identifier Code">
        <Values>
          <Value Code="41" />
        </Values>
    </Data>
(…)
```

Condition codes are auto generated in the target component, if they contain single value.

Configuration files are generated by a method that ensures that condition values are guaranteed to be unique across the sequence of repeating segments/loops.

If there is a need to edit conditions, or their values, the uniqueness constraint must be taken into account.

Completion flags

Autocompletion can be adjusted on the configuration file level. A new element "Completion",

having three attributes, has been introduced to define this:

- singleConditions – auto-complete single conditions for all fields in the target component
- singleValues – auto-complete single values for all fields in the target component
- HL – generate appropriate fields in the HL segment in the target component (for all HL segments)

  (Where "1" means true)

```
<Message>
  <MessageType>837-Q2</MessageType>
  <Completion singleConditions="1" singleValues="1" HL="1" />
  <Description>Health Care Claim: Dental</Description>
(...)
```

"Not Used" Data Elements

HIPAA omits several optional elements that are present in standard X12 transactions. These optional elements are hidden in MapForce components.

The validation engine checks to make sure that these unused fields are not present in source files. Since these fields cannot be mapped to the target component (because they are hidden), they cannot appear in target output files either.

These fields are defined to have the maxOccurs attribute equal to 0, in the configuration files. If necessary, you can manually hide (or unhide) specific fields by using the maxOccurs attribute.

```
<Data ref="F1037" minOccurs="0" info="Submitter Middle Name or Initial" />
<Data ref="F1038" minOccurs="0" maxOccurs="0" />
<Data ref="F1039" minOccurs="0" maxOccurs="0" />
```

Scope of validation

Semantic validation is not supported in MapForce. This means that “situational” fields are simply treated as optional.

7.5.8 HL7 Versión 2

HL7 (Health Level Seven) es un estándar sectorial para el intercambio electrónico de información clínica. MapForce es compatible con las versiones 2.2 - 2.6.

En MapForce la versión predeterminada de HL7 es la versión 2.6.

Puede descargar los esquemas XML y archivos de configuración de las versiones HL7 V2.2 - V2.5.1 desde la página de componentes de MapForce del sitio web de Altova (https://www.altova.com/es/components_mapforce.html).

MapForce 2018 es compatible con la versión HL7 3.x por medio de componentes de esquema XML.

El proyecto **MapForceExamples** incluye una asignación de ejemplo que asigna datos HL7 V2.6 a un archivo XML HL7 V3 XML (HL7V260_to_HL7V3.mfd).

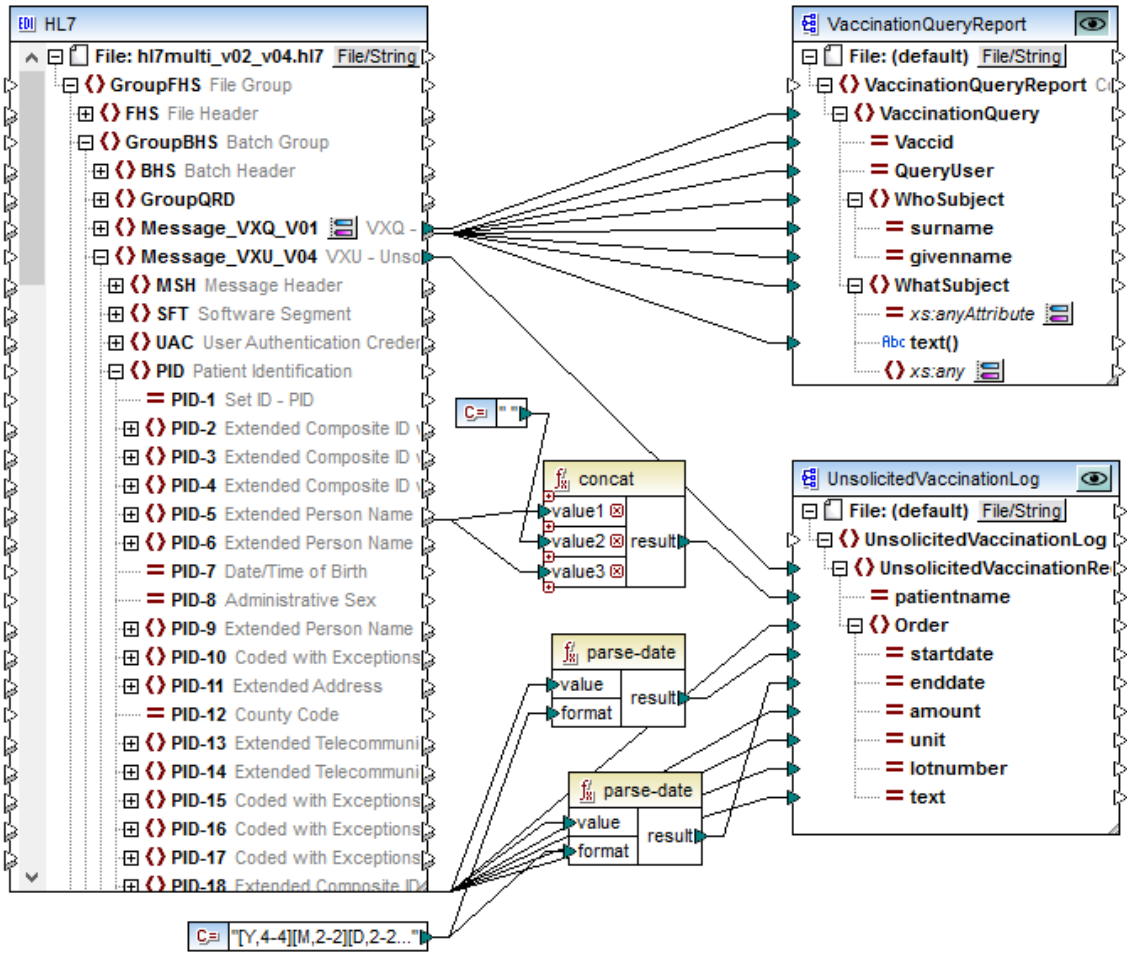
© 2018 Altova GmbH

Para más información consulte el sitio web http://www.hl7.org/.

7.5.8.1 Ejemplo: asignar datos HL7 a XML

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

This example illustrates how to map a file which contains two HL7 messages ("Query for vaccination record" and "Unsolicited vaccination record update") to two separate XML target files. It is accompanied by a mapping available at the following path: **<Documentos>\Altova \MapForce2018\MapForceExamples\HL7_MultiMessageTypes.mfd**.



HL7_MultiMessageTypes.mfd

The mapping can be created as shown in the steps below.

Step 1: Add the source HL7 file to the mapping

1. On the **Insert** menu, click **EDI**.
2. Select the collection "HL7", and then select the check boxes next to the message types "VXQ - Query for vaccination record" and "VXU - Unsolicited vaccination record update".
3. When prompted to supply an instance file, open the source file available at the following path: **<Documentos>\Altova\MapForce2018\MapForceExamples\hl7multi_v02_v04.hl7**.
4. When prompted, adjust the settings of the newly added HL7 component if necessary (for more information, see EDI Component Settings).

Step 2: Add the target schema to the mapping

1. On the **Insert** menu, click the **XML Schema/File**, and open the **VaccinationQueryReport.xsd** file from the **<Documentos>\Altova\MapForce2018\MapForceExamples** directory.
2. When prompted to supply a sample XML file, click **Skip**.

Follow the same steps to add the **UnsolicitedVaccinationLog.xsd** as a mapping component.

Step 3: Add functions and draw the mapping connections

In the source HL7 file, dates have the format `YYYYMMDD`. To convert such dates to a format suitable for the target XML file `YYYY-MM-DD`, we will add a date conversion function, as follows:

1. Drag the `parse-date` function from the Libraries window into the mapping. For more information about functions, see Working with Functions.
2. On the **Insert** menu, click **Constant**, and enter `[Y,4-4][M,2-2][D,2-2]` as constant value.
3. Draw the following connections:

From	To	Purpose
Constant	`format` input item of the `parse-date` function	Supplies to the function the format according to which the date should be parsed.
`RXA-3`	`value` input item of the `parse-date` function	Supplies to the function the actual value to be formatted.
`result` output item of the `parse-date` function	`startdate` input item of the XML target	Supplies the value returned by the function to the `startdate` XML node.

Follow the same steps to a new `parse-date` function, so as to convert the value of `RXA-4` and copy it to the `enddate` item of the target XML.

The `patientname` item in the second target XML is created by concatenating two fields of the

© 2018 Altova GmbH

source HL7 file: `FN-1` (Surname) and `XPN-2` (Given Name). To achieve this, this drag the **`concat`** function from the Libraries window onto the mapping. Also, add a constant that will supply the space character to separate the two names.

Draw the remaining mapping connections as shown in the example mapping (these are too many to be listed here; essentially, each connection copies values from the source HL7 file to the target XML files. For basic information about mapping connections, see Working with Connections.)

To preview the output generated by any of the target XML components, first click its **Preview** ( ) button, and then click the **Output** tab.

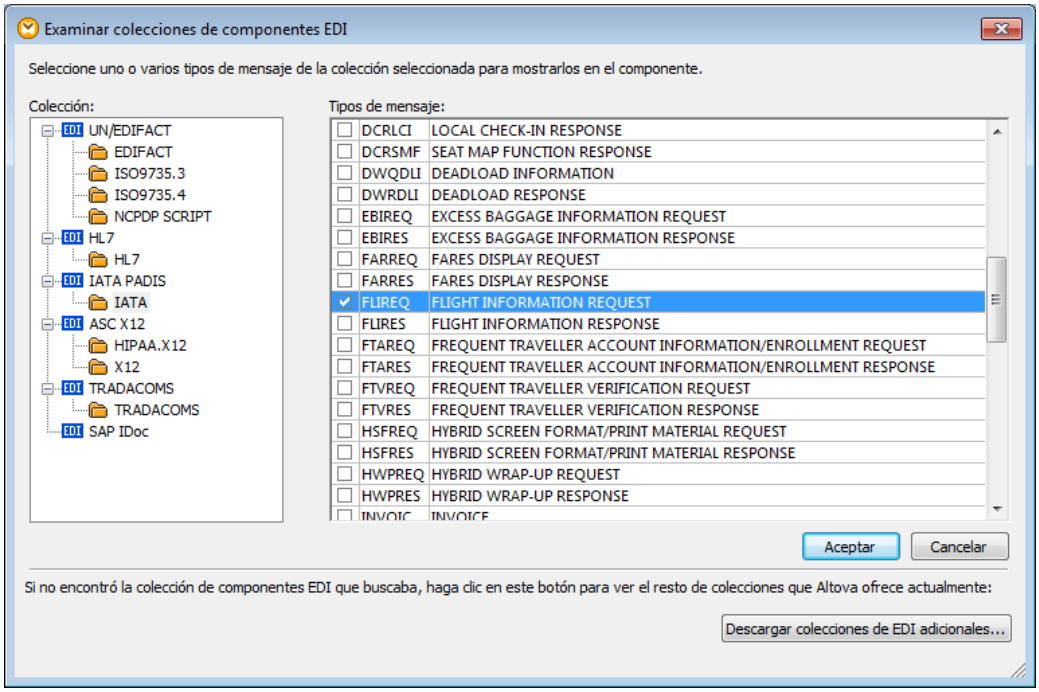
7.5.9 IATA PADIS

PADIS (*estándar de intercambio de datos sobre pasajeros y aeropuertos*) es un conjunto de mensajes que usa la sintaxis EDIFACT (ISO 9735).

- MapForce es compatible con los mensajes que usan el encabezado y el finalizador de mensajes UNH/UNT solamente.
- MapForce es compatible con las colecciones IATA.00.1 - IATA.08.1.

Para agregar un archivo IATA PADIS como componente de asignación:

1. En el menú **Insertar** haga clic en el comando **EDI**.
2. En el panel izquierdo seleccione la colección `IATA` (bajo `IATA PADIS`) y después, en el panel derecho, marque la casilla de los tipos de mensaje que desea utilizar.

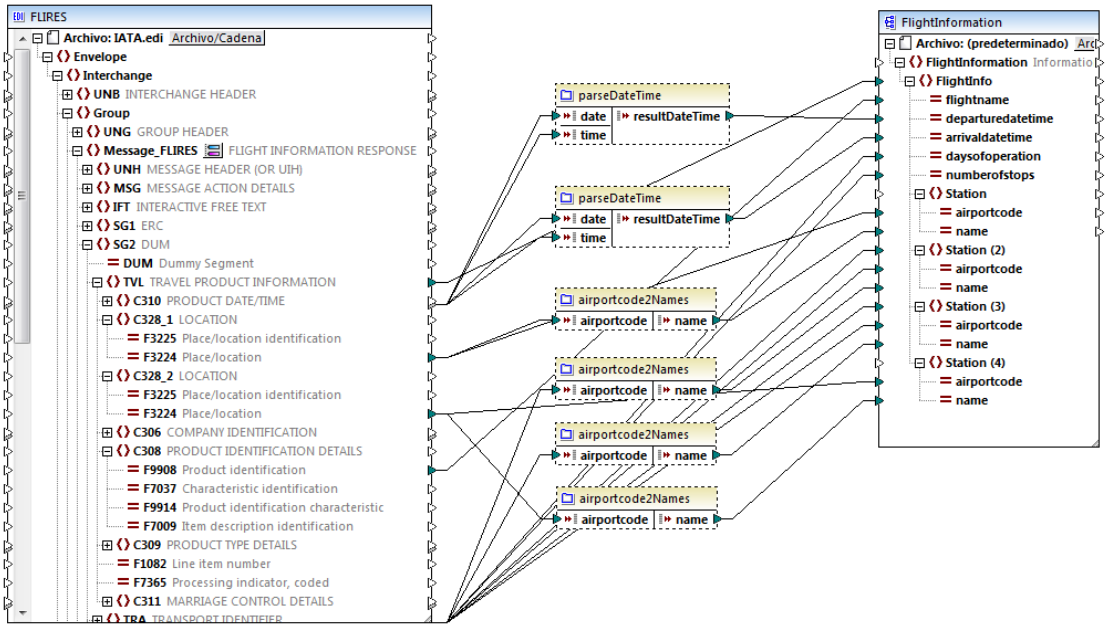


3. Cuando la aplicación solicite un archivo de instancia haga clic en **Examinar** y abra el archivo IATA de entrada que desea utilizar como origen de la asignación. Si el destino de la asignación es un componente IATA PADIS, haga clic en **Omitir**.

El componente IATA se añade a la asignación y ya puede empezar a crear conexiones. Para cambiar su configuración, haga clic con el botón derecho en el componente IATA y después seleccione **Propiedades** en el menú contextual (*véase* *Configuración de componentes EDI*).

7.5.9.1 Ejemplo: asignar datos IATA a XML

El archivo **IATA_FlightInformationReport.mfd** de la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples** es un ejemplo de asignación de datos entre un archivo IATA PADIS y un archivo XML.



IATA_FlightInformationReport.mfd

Esta asignación está formada por un componente IATA de origen que incluye un mensaje FLIRES, por un componente XML de destino y por dos funciones definidas por el usuario (**`parseDateTime`** y **`airportcode2Names`**) que se utilizan varias veces.

Para ver el archivo de instancia que leen o escriben los componentes de origen y de destino haga clic con el botón derecho en el componente correspondiente y seleccione **Propiedades** (*véase* *Configuración de componentes EDI*).

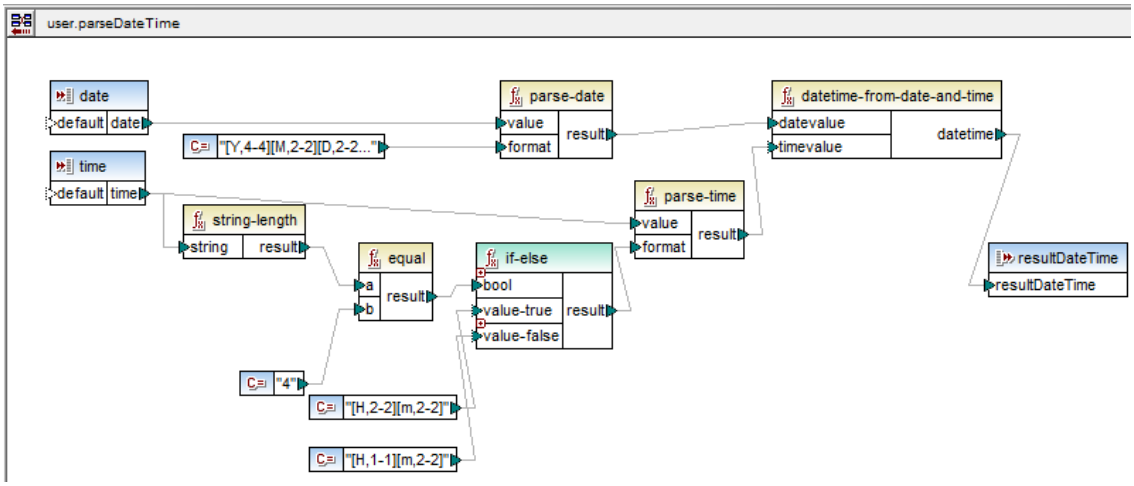
Para ver la estructura de cada función definida por el usuario haga doble clic en la función. Para volver a la asignación principal haga clic en el botón **Volver a la asignación principal** situado en la esquina superior izquierda del área de asignación (*véase* *Funciones definidas por el usuario*).

En esta asignación las funciones definidas por el usuario tienen este objetivo:

1) La función **`parseDateTime`** recibe como argumentos la fecha y la hora de los elementos de fecha y hora IATA y los procesa para producir un valor `xs:dateTime` que es necesario en el

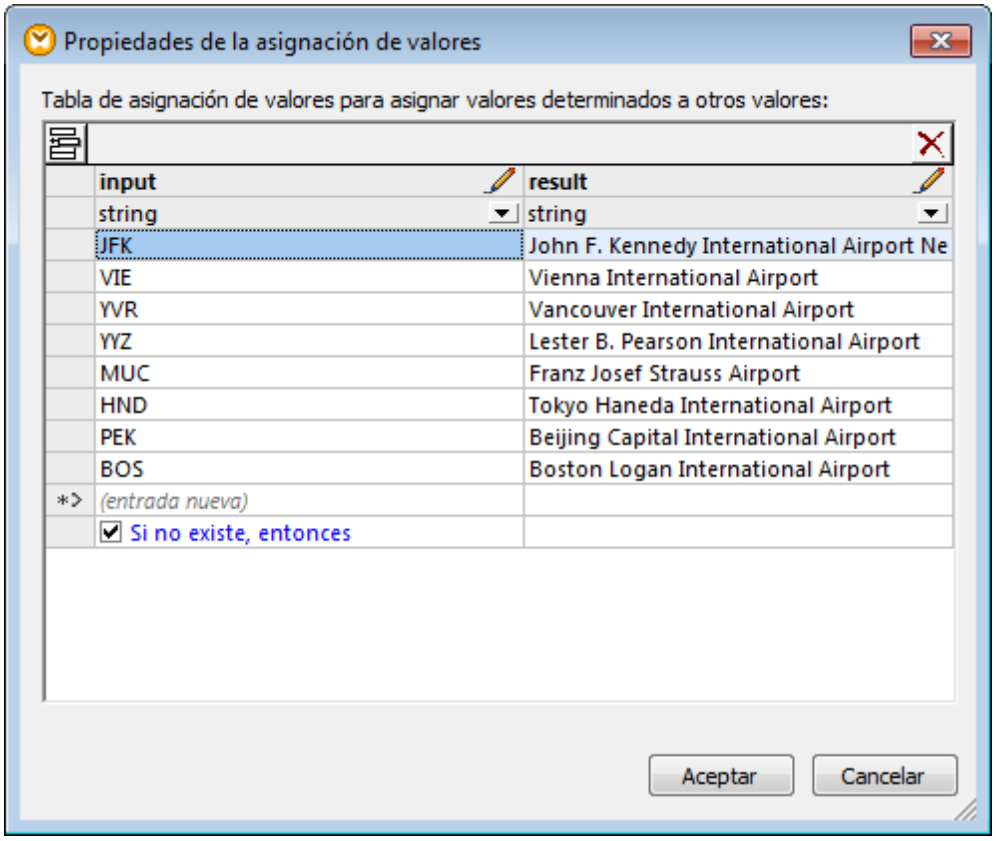
 © 2018 Altova GmbH

archivo XML de destino. Para conseguir este objetivo la función utiliza otras funciones de la biblioteca **core** de MapForce:



Estructura de la función definida por el usuario parseDateTime

2) La función **`airportcode2Names`** recibe como argumento el valor del campo F3223 o F3224 (el código del aeropuerto) y devuelve el nombre completo del aeropuerto. Para ello la función incluye un componente de asignación de valores que asigna cada código de aeropuerto a un nombre de aeropuerto:



Estructura del componente de asignación de valores

Para más información sobre los componentes de asignación de valores consulte el apartado Usar asignaciones de valores.

En el destino de la asignación el componente XML utiliza una técnica llamada *duplicación de nodos.* Es decir, el nodo `Station` se duplica varias veces en el componente de asignación. Esto permite asignar valores de varios elementos de origen al mismo elemento de destino (*véase Duplicar entradas*).

Para generar una vista previa de los resultados de esta asignación abra el panel *Resultados*.

7.5.10 NCPDP SCRIPT

SCRIPT es un estándar para el intercambio de información electrónica en el sector sanitario de EE UU desarrollado por el NCPDP (Consejo nacional para programas de medicamentos que requieren receta médica). Consulte el sitio web http://www.ncpdp.org. MapForce permite asignar datos entre archivos en formato SCRIPT y cualquier otro formato compatible con MapForce (como XML, CSV, Microsoft Excel, bases de datos, etc.). Las versiones de NCPDP SCRIPT compatibles con MapForce son las versiones 8.1, 10.6, 10.10.

En MapForce la versión predeterminada de NCPDP SCRIPT es la versión 10.10.

Nota: para ampliar la compatibilidad de MapForce con otras versiones de NCPDP SCRIPT

© 2018 Altova GmbH

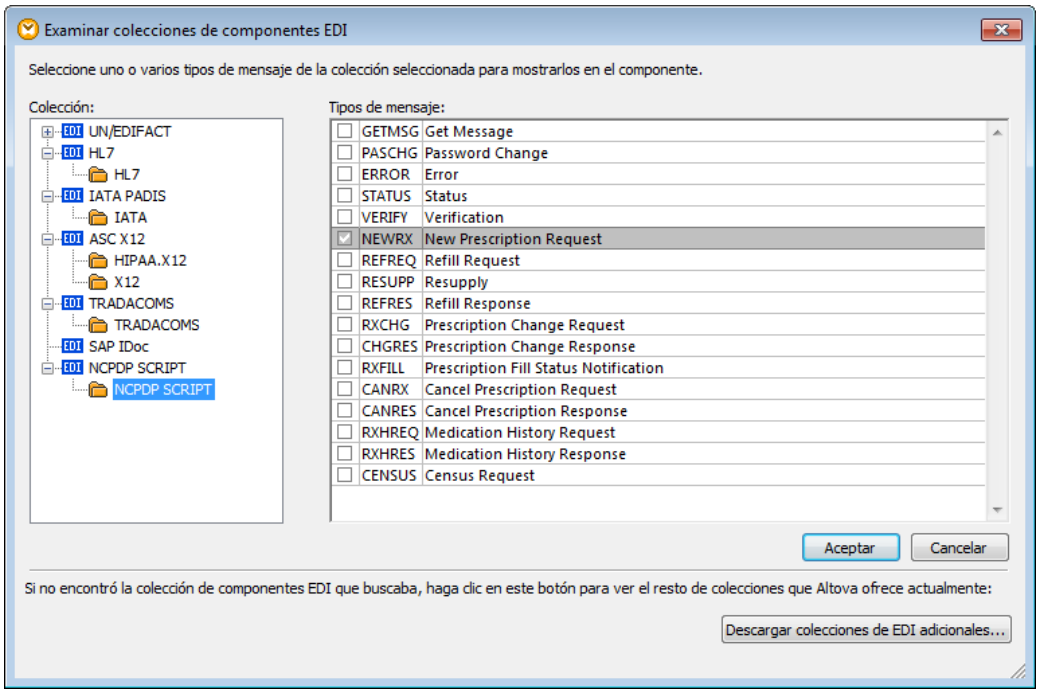
distintas a la predeterminada debe descargar e instalar el paquete "Archivos de configuración NCPDP SCRIPT" del centro de descargas de Altova (https://www.altova.com/es/components_mapforce.html).

NCPDP SCRIPT puede usar el formato XML o el formato UN/EDIFACT. Esta documentación afecta a la versión UN/EDIFACT de NCPDP SCRIPT. Si tiene esquemas XML NCPDP, MapForce le permitirá crear asignaciones entre datos SCRIPT de forma implícita usando los comandos de asignación y validación de datos XML estándar (para agregar archivos XML a la asignación utilice el comando de menú **Insertar | Archivo o esquema XML**).

Recuerde que MapForce no es compatible con el envío ni la recepción de mensajes SCRIPT como parte de las comunicaciones de datos entre los participantes del proceso SCRIPT. La función de MapForce es leer los datos en formato SCRIPT y asignarlos a otros formatos o pasar datos de otros formatos al formato SCRIPT.

Para agregar archivos NCPDP SCRIPT en formato UN/EDIFACT a la asignación:

1. En el menú **Insertar** haga clic en el comando **EDI**.
2. Seleccione la colección **NCPDP SCRIPT** y después marque la casilla del tipo de mensaje que desea usar.



3. Haga clic en **Aceptar**. Ahora la aplicación solicita un archivo de instancia.
4. Tiene dos opciones:
   - si tiene pensado asignar datos de un mensaje NCPDP SCRIPT a otro formato, haga clic en **Examinar** y seleccione el archivo SCRIPT de origen. También puede abrir archivos desde una URL (*véase* *Agregar componentes desde una URL*).
   - si tiene pensado asignar datos a un mensaje NCPDP SCRIPT o prefiere aportar el archivo de origen más tarde, haga clic en Omitir.

Llegados a este punto la estructura del mensaje NCPDP SCRIPT seleccionado aparece en la asignación (se trata del componente EDI). Ahora puede dibujar conexiones con este componente

como origen o como destino. Consulte estos apartados para obtener más información:

- Agregar componentes EDI a la asignación
- Ejecutar asignaciones de datos EDI
- Configuración de componentes EDI
- Validación de componentes EDI

7.5.10.1 Regla de validación NCPDP SCRIPT

Con respecto a mensajes NCPDP SCRIPT, MapForce realiza estas comprobaciones de validación:

- `UIB/S001/F0001` debería ser "UNOA"
- `UIB/S001/F0002` debería ser "0"
- `UIH/S306/F0329` debería contener el tipo de mensaje "SCRIPT"
- `UIH/S306/F0316` debería contener el número de versión de mensaje que está definido en los archivos de configuración seleccionados
- `UIH/S306/F0318` debería contener el número de publicación de mensaje que está definido en los archivos de configuración seleccionados
- `UIH/S306/F0326` debería contener función de mensaje (o el tipo de mensaje si seguimos la terminología de MapForce)
- `UIT/F0062` debería coincidir con el `UIH/F0062` correspondiente (encabezado/finalizador de mensaje) o no estar presente (campo opcional)
- `UIT/F0074` debería contener recuento de segmentos o no estar presente (campo opcional)
- `UIZ/F0036` debería contener recuento de mensajes o no estar presente (campo opcional)

Tema relacionado: Validación de componentes EDI

7.5.10.2 Reglas de finalización automática NCPDP SCRIPT

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

When generating an NCPDP SCRIPT file, MapForce automatically generates certain fields or values, according to the rules below (referred to as "automatic data completion", or "auto-completion" rules). To disable this behaviour, clear the **Auto-complete missing fields** check box in EDI Component Settings.

The following fields will be automatically completed:

Interchange Header

- `UIB/S001/F0001` = controlling agency + syntax version number = "UNOA"

© 2018 Altova GmbH

- UIB/S001/F0002 = syntax version number = "0"
- UIB/S002/F0004 = "Sender"
- UIB/S003/F0010 = "Recipient"
- UIB/S300/F0017 = current date
- UIB/S300/F0114 = current time

Message Header

- UIH/S306/F0329 = "SCRIPT", message type
- UIH/S306/F0316 = "010" for 10.6, message version number from configuration files.
- UIH/S306/F0318 = "006" for 10.6, message release number from configuration files.
- UIH/S306/F0326 = message function (or message type from a MapForce perspective)
- UIH/F0062 = "0", message reference number

Message Trailer

- UIT/F0062 = UIH/F0062
- UIT/F0074 = number of segments in the message.

Interchange Trailer

- UIZ/F0036 = message count

7.5.11 SAP IDoc

Los documentos SAP IDoc (documentos intermedios) se usan para el intercambio de datos comerciales entre aplicaciones SAP y no SAP. Los documentos son un tipo de almacenamiento intermedio de datos que puede intercambiarse con sistemas diferentes.

Los documentos IDoc tienen esta estructura:

- **Registro de control:** contiene datos de control sobre el documento IDoc: el remitente, el destinatario, el tipo de mensaje y el tipo de IDoc. El formato de registro de control es similar en todos los tipos de documentos IDoc.

- **Segmento de datos:** contiene los datos propiamente dichos y metadatos: la cabecera, el número de segmento y el tipo, así como los campos que contienen los datos.

- **Registro de estado:** contiene información sobre el estado actual del documento, es decir, sobre las etapas ya procesadas y las que aún se deben procesar. El formato de estado es el mismo en todos los tipos de documentos IDoc.

El número de versión en la definición de puerto define los sistemas con los que se establece la comunicación. La principal diferencia entre las versiones es la longitud de los nombres utilizada en los elementos, así como el uso de extensiones. A diferencia de las versiones anteriores, SAP R3 versión 4.x admite nombres (y extensiones) largos.

Versión de puerto 1: Versiones 2.1. y 2.2.

Versión de puerto 2: Versiones 3.0, 3.1 y sistemas R/2.
Versión de puerto 3: Versión 4.x (valor predeterminado)

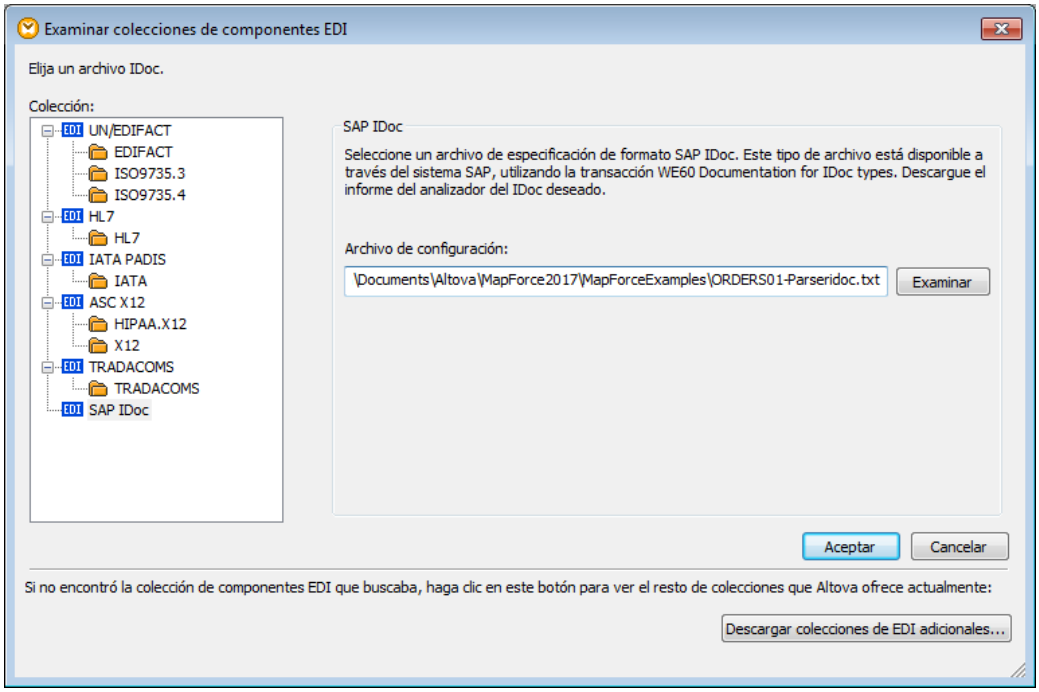
MapForce trata los componentes IDoc como archivos de longitud fija con una longitud de 30 caracteres (para el tipo Mensaje), 30 caracteres (para el tipo IDoc) y 27 caracteres para los campos de segmentos.

Para ver un ejemplo de asignación de datos SAP IDoc a XML consulte el archivo **<Documentos>\Altova\MapForce2018\MapForceExamples\IDoc_Order.mfd**.

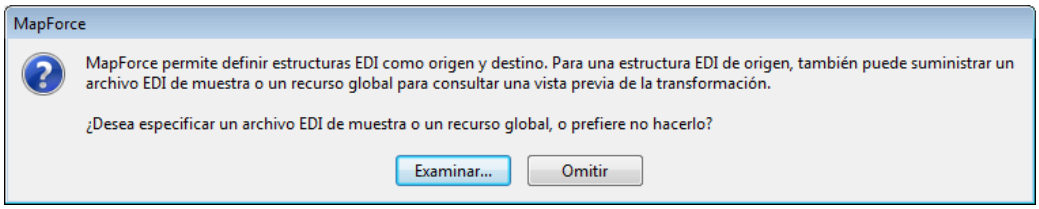
Para agregar un documento SAP IDoc como componente de la asignación:

1. Seleccione el comando de menú **Insertar | EDI**. Esto abre el cuadro de diálogo "Examinar colecciones de componentes EDI".
2. En el cuadro de lista haga clic en la entrada **SAP IDoc**.
3. Haga clic en **Examinar** y seleccione el informe del analizador IDoc (también llamado *archivo de definición IDoc*).

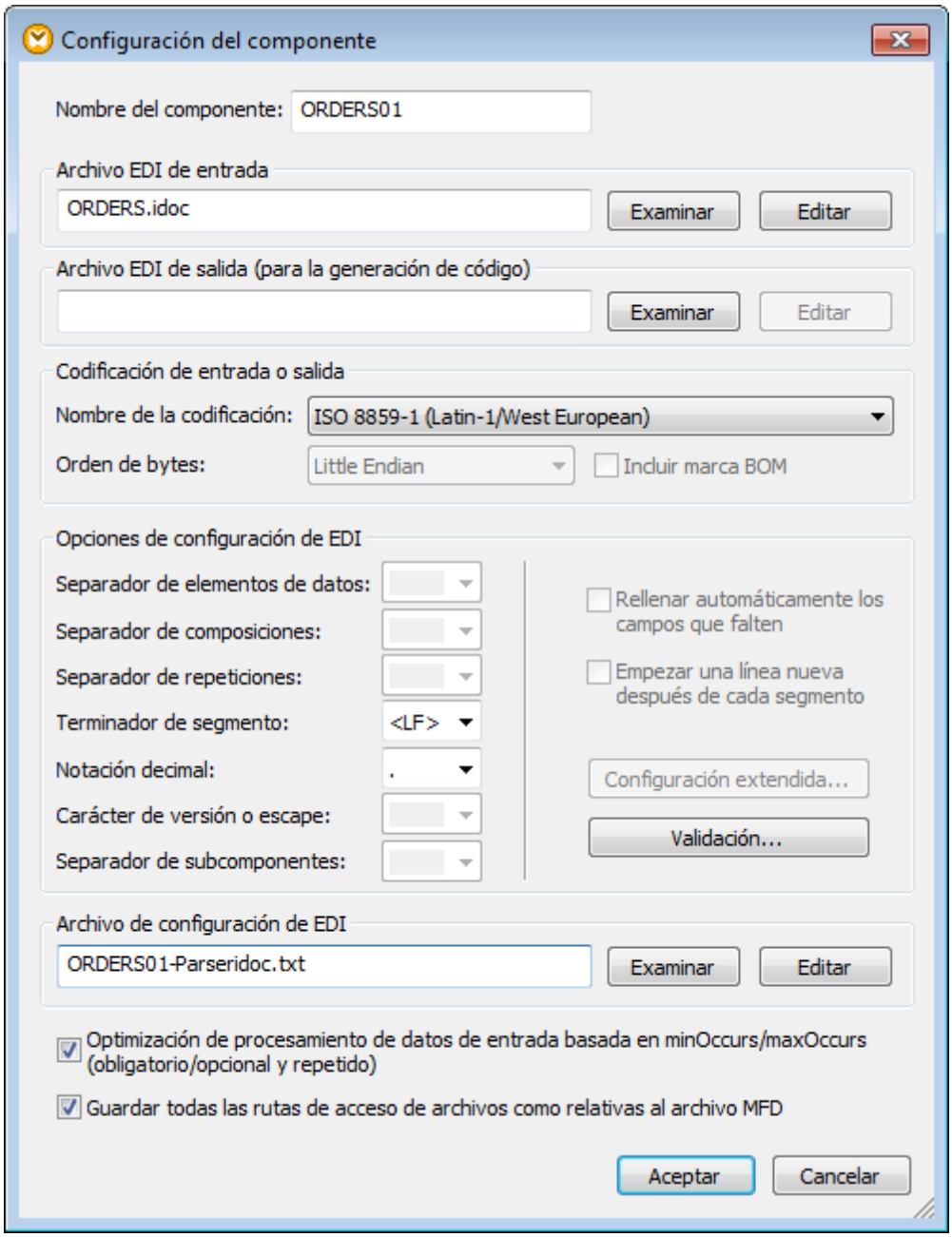
> El informe del analizador IDoc se crea desde el sistema SAP con la transacción WE60 ("Documentación para tipos IDoc"). Recuerde que el archivo se debe exportar de SAP en formato comprimido. Para más información sobre cómo invocar la transacción WE60 y cómo generar el informe del analizador consulte la documentación de SAP IDoc. La carpeta de ejemplos de MapForce incluye un ejemplo de informe del analizador (véase `<Documentos>\Altova\MapForce2018\MapForceExamples\ORDERS01-Parseridoc.txt`).



4. Haga clic en **Aceptar**. MapForce solicitará un archivo de muestra EDI que es opcional.



5. Si desea asignar datos desde un archivo IDoc, haga clic en **Examinar** y seleccione el archivo IDoc (*.idoc) que suministra los datos. En la carpeta de ejemplos de MapForce encontrará un ejemplo de archivo IDoc (`<Documentos>\Altova\MapForce2018\MapForceExamples\ORDERS.idoc`). De lo contrario haga clic en **Omitir**.
6. The Component Settings dialog box is displayed. This enables you to review the settings of the IDoc component before adding it to the mapping. You can change these settings at any time later if required (véase Changing the Component Settings).



7. Haga clic en **Aceptar** para cerrar el cuadro de diálogo "Configuración del componente" y añadir el componente IDoc al área de asignación.

7.5.12 TRADACOMS

TRADACOMS (Trading Data Communications) es un estándar de intercambio de datos electrónicos del Reino Unido utilizado en el comercio minorista. Se utilizó por primera vez en 1982 y fue el primer estándar EDI para la industria y el comercio del Reino Unido. Aunque guarda muchas similitudes con el estándar UN/EDIFACT, TRADACOMS es uno de los antecesores de

© 2018 Altova GmbH

EDIFACT y utiliza diferentes estructuras dentro de los mensajes.

MapForce implementa la especificación TRADACOMS base siguiendo lo dispuesto en el manual "TRADACOMS Manual of Standards for Electronic Data Interchange", publicado en enero de 1993 por la asociación Article Numbering Association (ANA) UK, que ahora se conoce con el nombre GS1 UK (https://www.gs1uk.org). MapForce se puede personalizar para las demás versiones de TRADACOMS por medio de archivos de configuración, de modo que procese tipos de mensaje, elementos de datos y valores de código nuevos.

En MapForce puede trabajar con el formato TRADACOMS de la misma manera que con los demás formatos EDI compatibles:

- Puede crear asignaciones de datos entre archivos TRADACOMS y cualquier otro formato de datos compatible con MapForce.
- Puede crear asignaciones de datos de archivos TRADACOMS en el lenguaje de transformación integrado o BUILT-IN (utilizado para generar la vista previa de resultados o en las ejecuciones con MapForce Server y FlowForce Server) y en los lenguajes de generación de código (C++, C#, Java). Cuando implemente la asignación en un servidor FlowForce Server que esté en otro equipo, el paquete implementado incluirá automáticamente las configuraciones de los tipos de mensaje TRADACOMS seleccionados y todas las listas de códigos que utilizan sus elementos de datos. Igualmente, en el código generado, MapForce genera clases para los grupos, segmentos y elementos de datos de la configuración.
- Puede definir con flexibilidad qué eventos de validación deben detener la conversión de datos, qué eventos deben dar lugar a registros rechazados (o aceptados) y qué eventos se deben pasar por alto.
- Puede habilitar o deshabilitar la función de finalización automática de datos. Si habilita esta función, MapForce rellena algunos valores automáticamente cuando genera archivos TRADACOMS.
- Puede personalizar los tipos de mensaje, los elementos de datos y las listas de código por medio de archivos de configuración, de forma global o local (*véase Archivos de configuración*).
- Puede configurar la codificación de los archivos que se analizan o generan.

Estos son los tipos de mensaje y de archivo TRADACOMS que están disponibles en MapForce por defecto:

Tipo de archivo	Tipo de mensaje	Versión
ARCHIVO DE INFORMACIÓN DEL PRODUCTO	PROHDR (Product File Header)	8
	PROINF (Product Details)	8
	PROTLR (Product File Trailer)	8
ARCHIVO DE INFORMACIÓN SOBRE EL PRECIO	PRIHDR (Price File Header)	8
	PRIINF (Price Details)	8
	PRITLR (Price File Trailer)	8
ARCHIVO DE INFORMACIÓN SOBRE EL CLIENTE	CUSHDR (Customer Information Header)	8
	CUSINF (Customer Information Details)	8

Tipo de archivo	Tipo de mensaje	Versión
	CUSTLR (Customer Information Trailer)	8
ARCHIVO DE PEDIDO	ORDHDR (Order File Header)	9
	ORDERS (Order Details)	9
	ORDTLR (Order File Trailer)	9
ARCHIVO DE INSTRUCCIONES DE RECOGIDA	PICHDR (Picking Instructions File Header)	4
	PICKER (Picking Instructions File Details)	4
	PICTLR (Picking Instructions File Trailer)	4
ARCHIVO DE NOTIFICACIÓN DE ENTREGA	DELHDR (Delivery File Header)	9
	DELIVR (Delivery Details)	9
	DELTLR (Delivery File Trailer)	9
ARCHIVO DE CONFIRMACIÓN DE ENTREGA	DLCHDR (Delivery Confirmation Header)	5
	DLCDET (Delivery Confirmation Details)	5
	DLCTLR (Delivery Confirmation Trailer)	5
ARCHIVO DE FACTURA	INVFIL (Invoice File Header)	9
	INVOIC (Invoice Details)	9
	VATTLR (VAT Trailer)	9
	INVTLR (Invoice File Trailer)	9
ARCHIVO DE NOTA DE ABONO	CREHDR (Credit Note File Header)	9
	CREDIT (Credit Note Details)	9
	VATTLR (File VAT Trailer)	9
	CRETLR (Credit Note File Trailer)	9
ARCHIVO DE EXTRACTO/GIRO	SRMHDR (Statement/Remittance Details File Header)	9
	SRMINF (Statement/Remittance Line Details)	9
	SRMTLR (Statement/Remittance Details File Trailer)	9
ARCHIVO DE INSTRUCCIÓN DE ELEVACIÓN	UPLHDR (Uplift File Header)	4
	UPLIFT (Uplift File Details)	4
	UPLTLR (Uplift File Trailer)	4

© 2018 Altova GmbH

Tipo de archivo	Tipo de mensaje	Versión
ARCHIVO DE CONFIRMACIÓN DE ELEVACIÓN	UCNHDR (Uplift Confirmation Header)	3
	UCNDET (Uplift Confirmation Details)	3
	UCNTLR (Uplift Confirmation Trailer)	3
ARCHIVO DE INSTANTÁNEA DE EXISTENCIAS	SNPHDR (Stock Snapshot Header)	3
	SNPSTS (Stock Snapshot Details)	3
	SNPTLR (Stock Snapshot Trailer)	3
ARCHIVO DE AJUSTE DE EXISTENCIAS	SADHDR (Stock Adjustment Header)	3
	SADDET (Stock Adjustment Details)	3
	SADTLR (Stock Adjustment Trailer)	3
ARCHIVO DE INFORME DE DISPONIBILIDAD	AVLHDR (Availability File Header)	4
	AVLDET (Availability Report Details)	4
	AVLTLR (Availability File Trailer)	4
ARCHIVO DE COMUNICACIÓN GENERAL	GENHDR (General Communications File Header)	3
	GENRAL (General Communications Text)	3
	GENTLR (General Communications Trailer)	3
ARCHIVO DE PEDIDO COMPLEJO	CORHDR (Complex Order File Header)	6
	CORDER (Complex Order Details)	6
	CORTLR (Complex Order Trailer)	6
ARCHIVO DE CONFIRMACIÓN DE PEDIDO	ACKHDR (Acknowledgement File Header)	4
	ACKMNT (Acknowledgement Details)	4
	ACKTLR (Acknowledgement Trailer)	4
ARCHIVO DE INFORME DE PLANIFICACIÓN DE PRODUCTOS	PPRHDR (Product Planning Report Header)	2
	PPRDET (Product Planning Report Details)	2
	PPRTLR (Product Planning Report Trailer)	2
ARCHIVO DE ORDEN DE PAGO	PAYHDR (Payment Order File Header)	3
	PAYINF (Payment Order Line Details)	3
	PAYTLR (Payment Order File Trailer)	3

Tipo de archivo	Tipo de mensaje	Versión
ARCHIVO DE AVISO DE DÉBITO	DRAHDR (Debit Advice File Header)	3
	DRAINF (Debit Advice Line Details)	3
	DRATLR (Debit Advice File Trailer)	3
ARCHIVO DE AVISO DE CRÉDITO	CRAHDR (Credit Advice File Header)	3
	CRAINF (Credit Advice Line Details)	3
	CRATLR (Credit Advice File Trailer)	3
ARCHIVO DE CONDICIÓN DE EXCEPCIÓN	EXCHDR (Exception Condition File Header)	3
	EXCINF (Exception Condition Line Details)	3
	EXCTLR (Exception Condition File Trailer)	3
ARCHIVO DE INFORME DE PLANIFICACIÓN DE UBICACIONES	LPRHDR (Location Planning Report Header)	2
	LPRDET (Location Planning Report Details)	2
	LPRTLR (Location Planning Report Trailer)	2
ARCHIVO DE FACTURA DE SERVICIOS	UTLHDR (Utility Bill File Header)	3
	UTLBIL (Utility Bill File Details)	3
	UVATLR (Utility Bill VAT Trailer)	3
	UTLTLR (Utility Bill File Trailer)	3

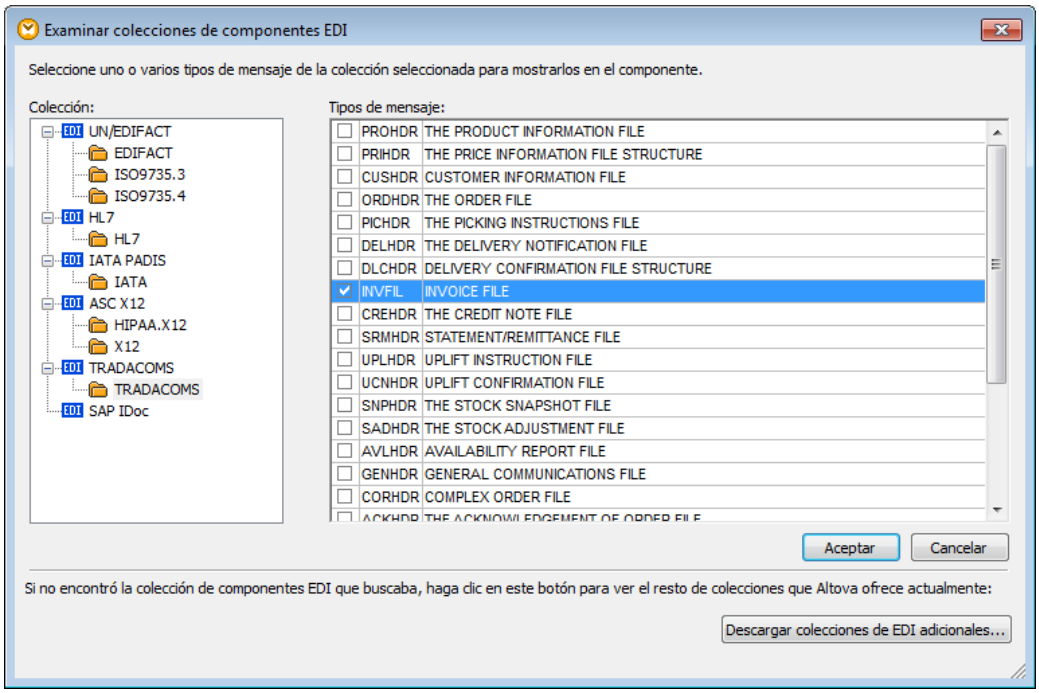
Nota: también es compatible el mensaje de conciliación RSGRSG versión 2. Este tipo de mensaje puede darse en todos los tipos de archivo.

7.5.12.1 Agregar archivos TRADACOMS como componentes

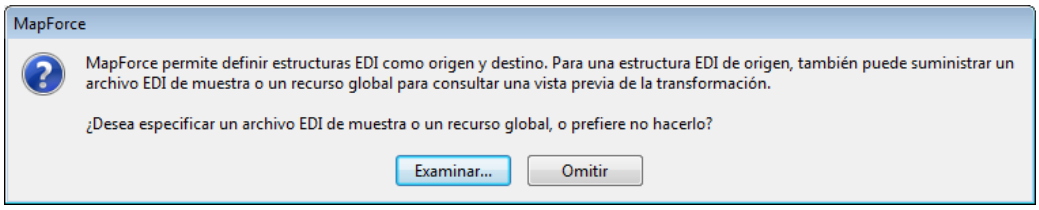
Para agregar un archivo TRADACOMS como origen de la asignación de datos:

1. En el menú **Insertar** haga clic en el comando **EDI** (o haga clic en el botón **Insertar EDI**  de la barra de herramientas).
2. En el cuadro de diálogo "Examinar colecciones EDI" haga clic en **TRADACOMS** y seleccione los tipos de mensaje que se deben incluir en el mensaje.

© 2018 Altova GmbH



3. Ahora haga clic en **Aceptar**. La aplicación solicitará un archivo de instancia.



4. Si prefiere especificar el archivo TRADACOMS más tarde, haga clic en **Omitir**. De lo contrario haga clic en **Examinar** y elija dónde se debe abrir el archivo TRADACOMS:
   - Si desea abrirlo desde la unidad de disco o red local, navegue hasta el archivo y haga clic en **Abrir**.
   - Si desea abrirlo desde una URL, haga clic en **Cambiar a URL** (*véase Agregar componentes desde una dirección URL*).
   - Si el archivo está deifnido como recurso global, haga clic en **Cambiar a recursos globales** (*véase Recursos globales de Altova*).
5. Si procede, modifique la configuración del componente el cuadro de diálogo "Configuración del componente" (véase EDI Component Settings).

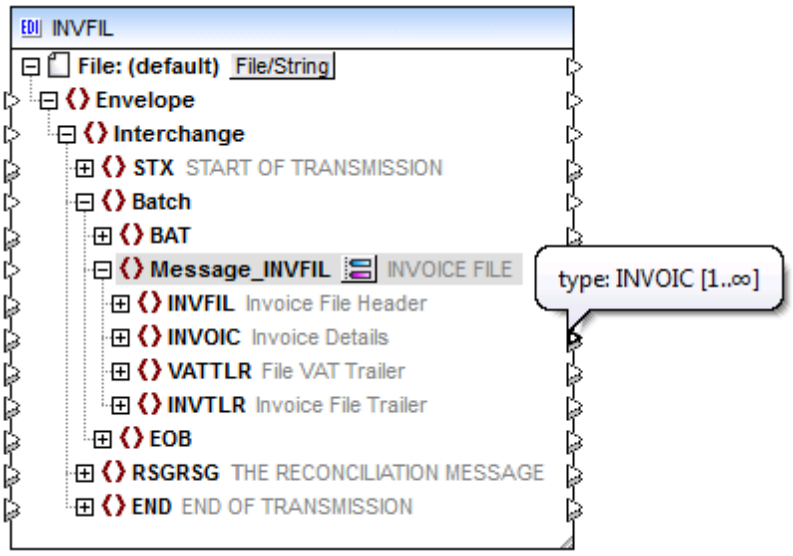
Para agregar un archivo TRADACOMS como destino de la asignación de datos:

1. Siga los pasos nº 1 y 2 de la primera parte de este apartado.
2. Cuando la aplicación solicite un archivo de instancia, haga clic en **Omitir**.

7.5.12.2 Componentes TRADACOMS en MapForce

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

In MapForce, the TRADACOMS component replicates the structure of a generic TRADACOMS transmission (with optional batching feature), while retaining all features common to other MapForce EDI-related components.



A sample TRADACOMS component in MapForce

The topmost item of the component, **File**, displays the name of the TRADACOMS interchange file being processed or generated by the component. "Default" indicates that no file has been assigned yet. The **File/String** (Archivo/Cadena) button displays advanced features common to all file-based MapForce components. It provides, among other options, the ability to read or process multiple files dynamically (for more information, see Processing Multiple Input or Output Files Dynamically).

The "Envelope" and "Interchange" structures are generic for all MapForce EDI-related components. Where applicable, they provide the ability to process multiple interchanges within the same MapForce component.

The STX and END structures are specific to the TRADACOMS format. They denote the "Start of Transmission" and "End of Transmission" segments, respectively.

The BAT and EOB structures are specific to the TRADACOMS format. They denote the "Start of Batch" and "End of Batch" segments, respectively. Such segments are applicable if you are

 © 2018 Altova GmbH

reading data from a TRADACOMS interchange file with batches, or if you want to generate a file with batches. Since there may exist multiple batches in the same transmission, a **Batch** sequence is available as parent of the BAT sequence.

The **Message_*Code*** structure (**Message_INVFIL**, in the sample above) corresponds to the TRADACOMS "file". The **Select EDI Message Types** button ( ) opens a dialog box where you can change the type of the file (for example, "Invoice File", "Orders File", and so on). By virtue of the existing MapForce EDI functionality, you can also include multiple file types in the same component. Note, however, that the TRADACOMS specification recommends one file and one type of file in each interchange transmission.

Any TRADACOMS file ("Invoice File", in this example) contains a Header Message, one or more Details Messages, and a Trailer Message. In the component above, these are the INVFIL (Invoice File Header), INVTLR (Invoice File Trailer), the VATTLR (File VAT Trailer) and multiple INVOIC (Invoice Details) messages.

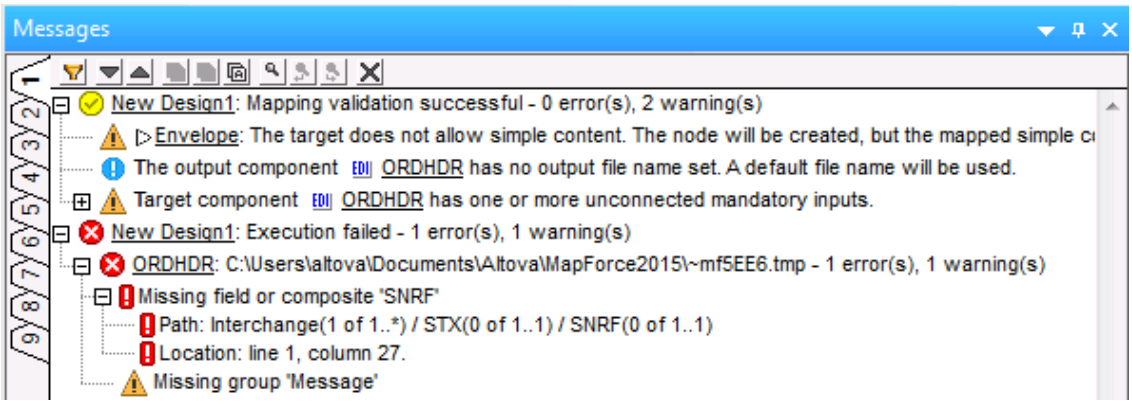
As with any other MapForce component, the input and output connectors (small triangles) displayed on each side of the component provide the ability to map each individual data element or segment to or from other data types or formats supported by MapForce. Likewise, when you move the mouse over any item on the component, you can view additional information about it (such as the minimum and maximum allowed occurrences), provided that tips are enabled from the **View | Show Tips** menu.

Finally, you can change these and other settings by double-clicking the component head and opening the Component Settings dialog box (see EDI Component Settings). Note that this dialog box is generic for all EDI-related component types and, thus, some of the options might not be applicable to TRADACOMS.

7.5.12.3 Validación y finalización automática de datos

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

When you run a mapping that reads data from or writes data to a TRADACOMS structure, MapForce performs structural data validation checks according to the TRADACOMS specification, and displays any validation errors in the Messages window.



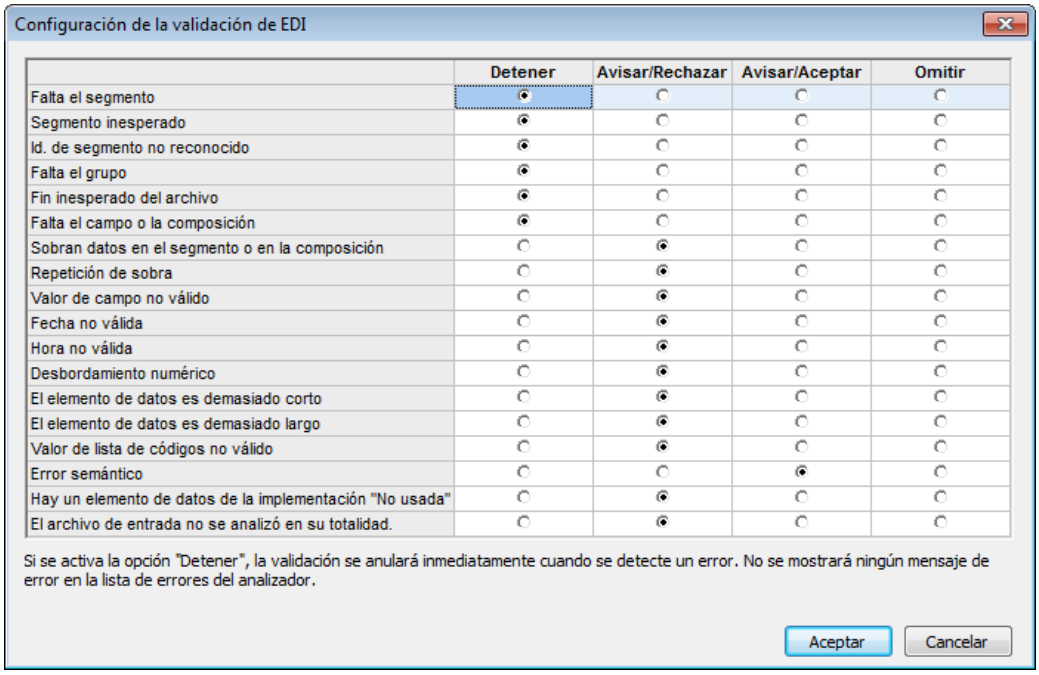
Messages window with validation errors

As shown in the sample above, the validation messages specific to TRADACOMS (information, warnings, and errors) are displayed in addition to the generic validation messages common to any MapForce mapping. For more information about MapForce validation in general, see Validating Mappings.

The following factors determine how MapForce validates the parsed or generated TRADACOMS files:

- The validation constraints defined in the configuration files available in the MapForce installation folder (subfolder **MapForceEDI\TRADACOMS**). These configuration files supply, on one hand, the default validation rules of the TRADACOMS specification. On the other hand, they provide the means to adapt the TRADACOMS format to custom requirements. In particular, it is possible to modify the data elements, segments, or code values defined in the configuration files, and thus influence both the outcome of validation and the mapping execution. For more information about the configuration files, see Configuration Files.
- The validation logic built into MapForce. This includes MapForce internal data integrity checks that may not be enforced by means of configuration files.
- Any custom validation settings that you have defined from the MapForce graphical user interface (see EDI component validation). To view or change the current EDI validation settings of any EDI component, including TRADACOMS, double-click the header of the component, and then click **Validation** on the Component Settings dialog box.

© 2018 Altova GmbH



When writing to a TRADACOMS structure, MapForce automatically fills in the contents of those data elements for which the value can be calculated or is predefined. This is referred to as "automatic completion" (or "auto-completion"). To disable this behavior, clear the **Auto-complete missing fields** check box from the Component Settings dialog box (see EDI Component Settings).

The following TRADACOMS validation rules cause MapForce to either raise validation errors (during file parsing or generation) or to auto-complete missing fields (during file generation):

1. The segments STX (Start of Transmission) and END (End of Transmission) must exist.
2. If STDS-1 has the value 'ANAA' then a Reconciliation Message (RSGRSG) must exist before the end of transmission (END). Otherwise, no Reconciliation Message (RSGRSG) must be present.
3. If STDS-1 has the value 'ANAA' then:
   a. The value of RSGA in the Reconciliation Message must be equal to the value of SNRF in the STX segment
   b. The value of RSGB in the Reconciliation Message must be equal to the value of UNTO-1 in the STX segment.
4. TRDT-1 must contain the date (YYMMDD) and TRDT-2 must contain the time (HHMMSS) of transmission (current date and time).
5. If the Batch Header (BAT) is present then the Batch Trailer (EOB) must also be present, and the number of messages in the batch must be available in the NOLI (Number of Messages in Batch) data element.
6. The MSRF (Message Reference) data element in the Message Header (MHD) must contain the consecutive count of messages within the transmission, starting with 1.
7. The NOSG (Number of Segments in Message) data element in the Message Trailer (MTR) must contain the number of segments, including MHD and MTR.
8. When present, the Reconciliation Message (RSGRSG) must consist of one segment (RSG), except the Message Header and the Message Trailer.
9. The NMST (Number of Messages in Transmission) data element in the END segment must

contain the number of messages in interchange (count of `MHD` segments).
10. In general, when reading a TRADACOMS structure, MapForce expects that the interchange environment is of type "computer to computer" (or, in TRADACOMS terminology, "intelligent terminal to intelligent terminal"). Therefore, a segment such as `MHD = 12 + ORDHDR :3` would trigger a validation error, since it contains extra leading and trailing spaces.
11. String data must be in upper case. When generating TRADACOMS output, MapForce converts string data to upper case.

Additionally, as already mentioned, any validation rules defined in the configuration files will also affect TRADACOMS parsing and generation.

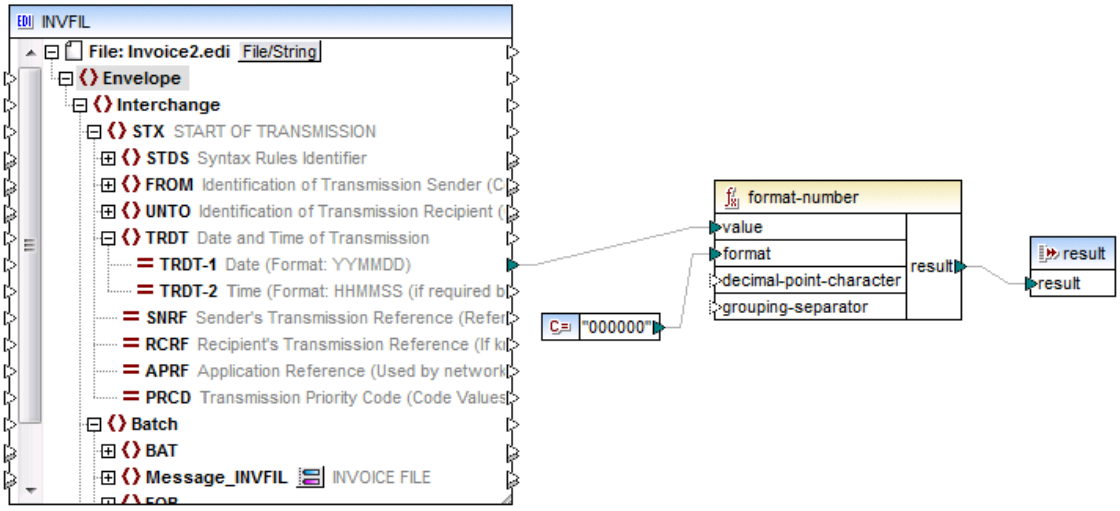
7.5.12.4 Conservar ceros iniciales durante la conversión

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

In the TRADACOMS specification, the data type of the TRDT-1 and TRDT-2 (Date and Time of Transmission) data elements is defined as "decimal". This means that, by default, when you map data from the TRDT-1 or TRDT-2 fields, any leading zeros in these fields would be trimmed during conversion, and thus produce undesired results. This may also be the case of any other items which are defined as "decimal", but store values which are meant to be treated as string.

You can instruct MapForce to treat numeric fields as string (and thus preserve the leading zeros) by applying the **`core | format-number`** MapForce function. In the following example, the value of the TRDT-1 item in the source file is "020312". The normal output of this item would be "20312" (as a result of conversion to decimal), which is not the desired result. Therefore, to keep the leading zero, the **`format-number`** function has been added to the mapping. (For information about working with functions, see Working with Functions.)

© 2018 Altova GmbH



Preserving leading zeros with the format-number function

As illustrated above, the function has the following two input arguments:

1. The value to be formatted (in this case, "020312").
2. The format mask "000000".

To test the output of the function, this example uses a simple output component (see Simple Output). When you click the **Output** button, the output of the mapping (which is the same as the result of the function) is "020312", as intended.

7.5.12.5 Archivos de configuración

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

By default, the MapForce configuration files and the underlying MapForce processing logic reflects the base version of the TRADACOMS standard (931). Therefore, if you use this TRADACOMS version, customization of the TRADACOMS configuration files is not necessary. However, if you want to make changes to the existing configuration for any reason, it is possible to do so. Note that any changes made to the standard TRADACOMS configuration files will take immediate effect and will affect the way MapForce parses or writes TRADACOMS structures.

The MapForce configuration files applicable to TRADACOMS are stored in the following directory: **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI\TRADACOMS**. Should you need to make changes to the TRADACOMS configuration files, you can start by creating a sibling copy of the TRADACOMS directory, for example "TRADACOMS1", edit the directory permissions not to be "read-only", and then edit the files in it according to your custom requirements. All

configuration files adhere to the XML syntax and, so it is recommended that you use an XML editor to modify the files.

File	Purpose
EDI.Collection	The **EDI.Collection** file contains a list of all messages in the current TRADACOMS configuration directory (for example, **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI\TRADACOMS**). Any messages listed in the collection file become available for selection in the "Browse EDI Collections" dialog box when you add a TRADACOMS component to the mapping (**Insert | EDI**). Therefore, if you want to suppress any message type on this dialog box, comment out or remove their respective XML element from the EDI.Collection file. For example, commenting out the following line makes the LPRHDR message type unavailable: `<!--<Message Type="LPRHDR" File="LPRHDR.Config" Description="LOCATION PLANNING REPORT FILE"/>-->`
Envelope.Config	Defines the structure of the TRADACOMS component in MapForce (that is, elements that are surrounded by the **Envelope** sequence). This file is validated against a schema handled by MapForce internally, and it should not be edited.
Tradacoms.Codelist	This file defines the TRADACOMS codes and the values that they may contain, and is used for validation of input/output TRADACOMS files. You can add custom code lists to this file, if required.
Tradacoms.Segment	This file defines the Segment, Composite and Field names of the TRADACOMS files, and is used when parsing a TRADACOMS file. Changes made to this file are global customizations, and apply to all segments and messages.
[Message].Config	Defines the structure of a single TRADACOMS message. Changes made to this file are treated as local (inline) customizations.

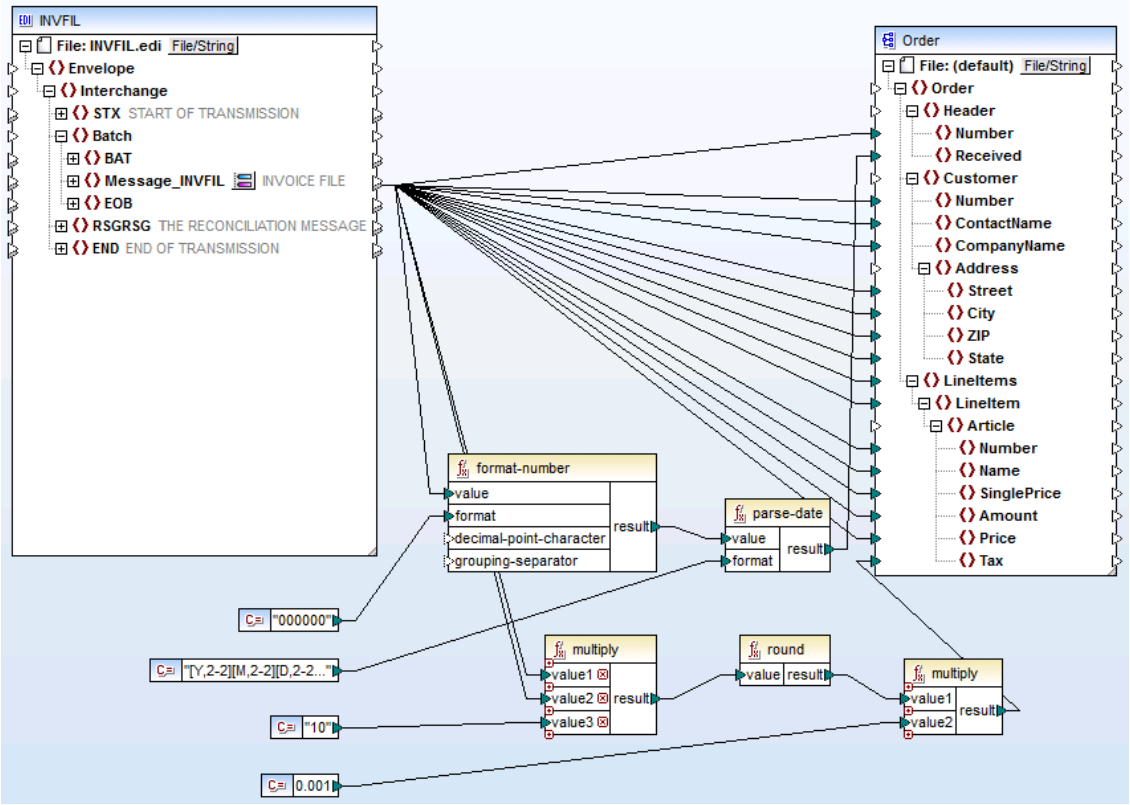
For more information, see Customizing EDI Structure and Customizing EDI Validation.

7.5.12.6 Ejemplo: convertir una factura TRADACOMS en XML

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

© 2018 Altova GmbH

This example provides instructions on how to create a mapping design that converts data from a TRADACOMS invoice file to XML format. The mapping design file created in this example is also available at the following path: **<Documentos>\Altova\MapForce2018\MapForceExamples\TRADACOMS_Invoice.mfd**.



TRADACOMS_Invoice.mfd

The mapping accomplishes the following goals:

1. Convert a source TRADACOMS file (**INVFIL.edi**) to an XML file which is valid against an existing schema (**Order.xsd**). Both the source TRADACOMS file and the schema file (**Order.xsd**) are available in the MapForce Examples folder (**<Documentos>\Altova\MapForce2018\MapForceExamples**).
2. The date in the TRADACOMS file has `decimal` format so it must be converted to `xs:dateTime` format in the target XML file.
3. For each line item in the purchase order, the tax amount must be calculated as a decimal value, rounded to three decimal places (thousandths). For the scope of this example, we will assume that the tax is calculated according to the following formula:

   ```
   Tax = round(LEXC * VATP * 10) * 0.001
   ```

   Where **round** rounds the value to the nearest integer.

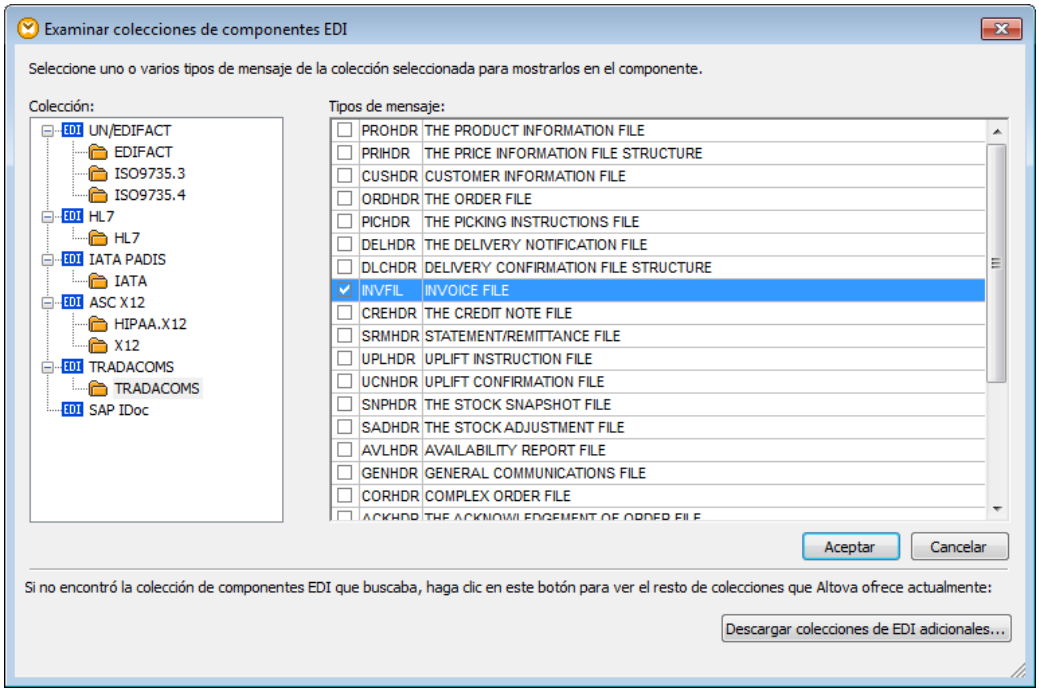
To accomplish the goals, take the following steps:

1. Add the source and target components to the mapping area and draw the mapping connections between them.
2. Use date functions to convert the date to the required format.
3. Use math functions to calculate the tax value.

Step 1: Add the components to the mapping area

To add the source TRADACOMS component:

1. On the **Insert** menu, click **EDI**.
2. Click the TRADACOMS collection folder, and, under Message Types, select the **INVOICE FILE**.



3. When prompted to supply a sample file, browse for the **INVFIL.edi** available in the MapForce Examples folder (**<Documentos>\Altova\MapForce2018\MapForceExamples**).
4. When prompted to enter the component settings, leave the default values as is, and click OK.

To add the target XML component:

1. On the **Insert** menu, click **XML Schema/File**.
2. When prompted to supply a sample file, click **Skip**. (Because this is a target component, the file will be generated, so there is no need to browse for an existing one.)
3. When prompted to select a root element, select "Order".

In this example, the data will be mapped from the source TRADACOMS file to the target XML file

as follows:

Source (TRADACOMS)	Target (XML)
INVN	/Order/Header/Number
IVDT*	/Order/Header/Received
CIDN-2	/Order/Customer/Number
CADD-1	/Order/Customer/ContactName
CNAM	/Order/Customer/CompanyName
CADD-2	/Order/Customer/Address/Street
CADD-3	/Order/Customer/Address/City
CADD-5	/Order/Customer/Address/ZIP
CADD-4	/Order/Customer/Address/State
Group_IRF**	/Order/LineItems
Group_ILD**	/Order/LineItems/LineItem
SPRO-1	/Order/LineItems/LineItem/Article/Number
TDES-1	/Order/LineItems/LineItem/Article/Name
AUCT-1	/Order/LineItems/LineItem/Article/SinglePrice
QTYI-1	/Order/LineItems/LineItem/Article/Amount
LEXC	/Order/LineItems/LineItem/Article/Price
LEXC*, VATP*	/Order/LineItems/LineItem/Article/Tax

* Source data will be processed with functions before being written to the target
** These fields enable iteration through a group of fields of the same type

Using the table above as reference, you can now draw mapping connections from all of the above items in the source component to their destination in the target component. For now, you can omit those fields that require calculated values (that is, the ones marked with * in the table); these will be handled in the next steps. For general information about drawing connections in MapForce, see Working with Connections.

Tip: To search for a field by its name inside the MapForce component, press **Ctrl + F**.

Step 2: Add the date conversion functions

To convert the value of the `IVDT` item from a decimal value to `xs:dateTime` type, the mapping uses the **`format-number`** function from the **`core | conversion functions`** library. For a worked example, see Preserving Leading Zeros During Conversion.

The role of the second function, **parse-date**, is to convert the YYMMDD string value returned by the **format-number** to xs:dateTime type. It has two input arguments: (a) the value to be formatted and (b) the format mask. The format mask essentially instructs MapForce to treat the year, month, and date as having a width of exactly two characters each.

To add the functions above to the mapping, drag them from the Libraries window to the mapping area. To supply an argument to a function, insert a constant (the menu command is **Insert | Constant**). For general information about working with functions in MapForce, see Working with Functions.

Step 3: Add the tax calculation functions

The functions used to calculate tax according to the formula are **multiply** and **round**, available in the **core | math functions** library. The multiply function is extendable (that is, it can take a variable number of arguments). In this example, it takes LEXC, VATP, and the integer 10 as arguments. The first two values are connected from the source TRADACOMS component, while the integer is supplied by a constant. The result of the **multiply** function is then rounded to the nearest integer using the **round** function. Finally, the result of the **round** function is multiplied by 0.001 by means of a second **multiply** function. This result is then connected to the destination item of the XML component (/Order/LineItems/LineItem/Article/Tax).

To preview the output of the mapping design, click the **Output** tab in the lower part of the mapping window.

Step 4 (optional): Style the mapping output with StyleVision

Optionally, you can link the target XML component to a StyleVision Power StyleSheet (.sps) file. (For more information about StyleVision, see https://www.altova.com/stylevision.html.) This would enable you to generate the mapping output as HTML, RTF, PDF, or 2007+ file (provided that StyleVision is installed and the prerequisites required by each format are in place, see Styling Mapping Output with StyleVision). To take this optional step, double-click the header of the XML component, and, next to "StyleVision Power StyleSheet file", browse for the **Order.sps** file available in the MapForce Examples folder. Back on the mapping area, you can now click the HTML, PDF, RTF, and Word 2007+ tabs to view the mapping output in the corresponding format.

7.5.13 UN/EDIFACT

UN/EDIFACT (las reglas de las Naciones Unidas para el intercambio electrónico de datos sobre administración, comercio y transportes) es un conjunto de estándares para el intercambio electrónico de documentos (véase http://www.unece.org/cefact/edifact/welcome.html). MapForce es compatible con los mensajes incluidos en los directorios D93.A - D.17A.

En MapForce la versión predeterminada de UN/EDIFACT es la versión D.17A.

Desde el sitio web de Altova puede descargar un programa de instalación para instalar todas las versiones anteriores (hasta la versión D93.A), incluidos los archivos de configuración necesarios para asignaciones EDI interactivas. Para descargar estos archivos visite https://www.altova.com/es/components_mapforce.html.

© 2018 Altova GmbH

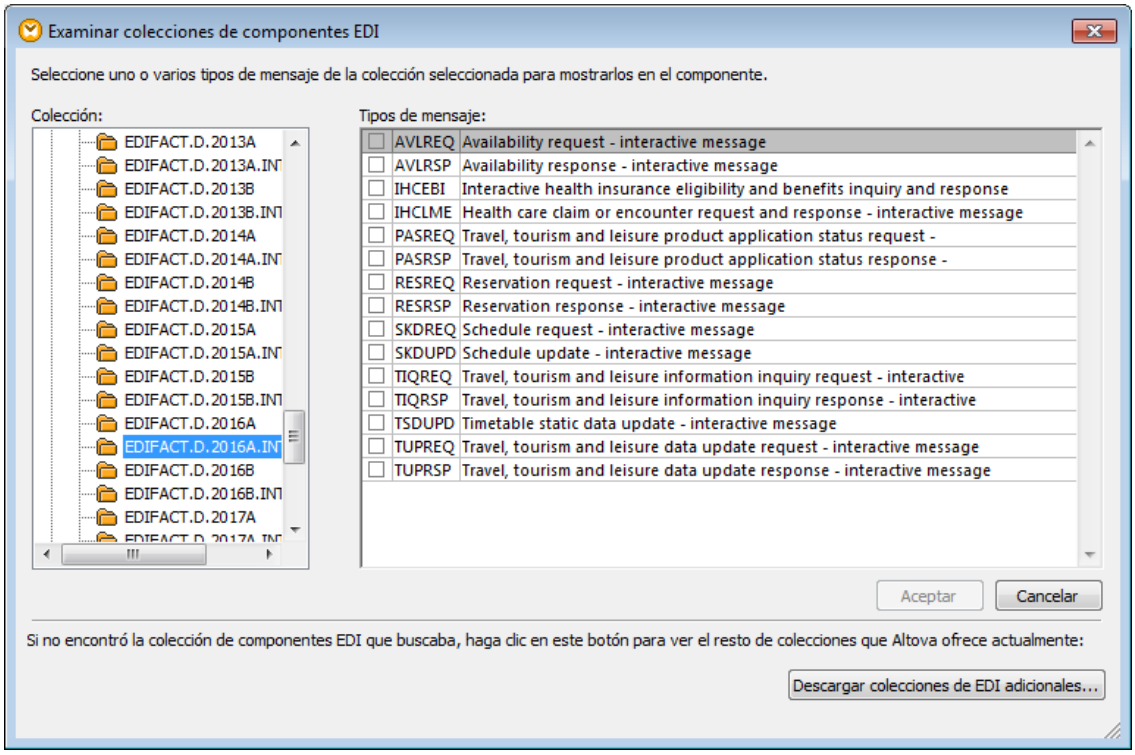
7.5.13.1 Mensajes EDIFACT por lotes e interactivos

En MapForce puede crear asignaciones de datos entre mensajes UN/EDIFACT por lotes e interactivos. Es decir, el componente UN/EDIFACT de MapForce ofrece la estructura necesaria para leer y escribir archivos de formato UN/EDIFACT sintácticamente válidos (para el intercambio por lotes o interactivo). Además, algunas reglas básicas de validación y finalización automática se aplican cuando se leen o escriben archivos de formato UN/EDIFACT, tal y como se describe en esta documentación. Recuerde que MapForce no transmite los archivos a los participantes del proceso de intercambio UN/EDIFACT. El papel de MapForce es generar dichos archivos o pasarlos a otros formatos.

Para poder crear asignaciones de datos entre mensajes EDIFACT por lotes no hay ningún requisito especial. Sin embargo, para crear asignaciones de datos entre mensajes EDIFACT interactivos deben cumplirse estos requisitos:

1. Debe tener instalado MapForce 2018 o superior.
2. Debe instalar por separado los archivos de configuración de MapForce necesarios para trabajar con mensajes EDI interactivos. El archivo de instalación se puede descargar gratis desde la página de bibliotecas de MapForce (https://www.altova.com/es/components_mapforce.html). Tras instalar los archivos de configuración, las colecciones y tipos de mensajes necesarios estarán disponibles en el cuadro de diálogo "Examinar colecciones de componentes EDI" (*imagen siguiente*).

Para añadir un componente UN/EDIFACT por lotes o interactivo a la asignación seleccione la colección correspondiente en el cuadro de diálogo "Examinar colecciones de componentes EDI". Este cuadro de diálogo se abre con el comando de menú **Insertar | EDI** (o con el botón **Insertar archivo EDI** de la barra de herramientas). Consulte el apartado Agregar componentes EDI a la asignación para obtener más información.



Cuadro de diálogo "Examinar colecciones de componentes EDI"

Nota: en el cuadro de diálogo el sufijo "INTERACTIVE" detona una colección relacionada con la estructura Interactive EDIFACT (p. ej. EDIFACT.D.2016A.INTERACTIVE). Las colecciones relacionadas con estructuras por lotes solo incluyen la versión del directorio UN/EDIFACT en su título (p. ej. EDIFACT.D.2016A). La misma convención debe aplicarse a los directorios de configuración (colecciones) disponibles en la carpeta de configuración EDI de MapForce (**C:\Archivos de programa\Altova\MapForce2018\MapForceEDI**).

La estructura de archivos UN/EDIFACT por lotes es distinta a la de los archivos destinados al intercambio interactivo. Por tanto, los componentes EDIFACT de MapForce pueden ser por lotes o interactivos (y, por tanto, pueden incluir mensajes por lotes o interactivos) pero no ambos a la vez. Igualmente, las reglas de validación y finalización automática son distintas en cada caso y se describe en detalle en estos apartados:

- Reglas de validación UN/EDIFACT (por lotes)
- Reglas de finalización automática UN/EDIFACT (por lotes)
- Reglas de validación UN/EDIFACT (interactivo)
- Reglas de finalización automática UN/EDIFACT (interactivo)

7.5.13.2 Reglas de validación UN/EDIFACT (por lotes)

Cuando se valida un documento UN/EDIFACT, MapForce comprueba:

- si existen o no los segmentos UNB y UNZ.
- si UNB/S004 contiene una especificación válida de fecha y hora.

 © 2018 Altova GmbH

- si UNB/0020 y UNZ/0020 contienen el mismo valor.
- si UNZ/0036 contiene el número correcto, que se define como número de grupos funcionales (si existen) o como número de mensajes. Si existen grupos funcionales, debe ser el número de grupos funcionales. De lo contrario, debe ser el número de mensajes que contiene el intercambio.

En cada **grupo funcional** se comprueba:

- si el grupo contiene los pares UNG y UNE correspondientes.
- si UNG/S004 contiene una especificación válida de fecha y hora.
- si UNE/0060 contiene el número correcto de mensajes que contiene el grupo funcional.

En cada **mensaje** se comprueba:

- si el mensaje contiene los pares UNH y UNT correspondientes.
- si UNH/S009/0052 contiene el mismo valor que UNG/S008/0052 del grupo funcional contenedor.
- si UNH/0062 y UNT/0062 contienen el mismo valor.
- si UNH/S009/0065 contiene el especificador de tipo de mensaje correcto.
- si UNT/0074 contiene el número correcto de mensajes que contiene el mensaje.

7.5.13.3 Reglas de finalización automática de datos UN/EDIFACT (por lotes)

Cuando genera archivos UN/EDIFACT, MapForce puede crear automáticamente ciertos campos o valores, de acuerdo con las reglas que se describen más abajo (en adelante "reglas de finalización automática de datos"). Para deshabilitar este comportamiento debe desactivar la casilla *Rellenar automáticamente los campos que falten* en el cuadro de diálogo "Configuración del componente" (*véase* *Configuración de componentes EDI*).

La **finalización automática de datos** para UN/EDIFACT se asegura:

- de que existan los elementos UNB y UNZ.
- de que, si existe UNG o UNE, también exista el otro ID..
- de que existan los segmentos UNH y UNT.
- de que exista UNB/S001. Si no contiene datos, se utiliza el nivel de sintaxis y el número de versión de sintaxis de la configuración definida por el usuario. (Botón **Configuración extendida | Número de versión de la sintaxis** del cuadro de diálogo Configuración del componente EDI.)
- de que existan UNB/S002 y UNB/S003.
- de que exista UNB/S004. Si no contiene datos, se inserta la fecha/hora actual en formato EDI.
- de que exista UNZ/0036. Si no contiene datos, se calcula e inserta el número de grupos funcionales o mensajes.
- de que exista UNZ/0020. Si no contiene datos, se copia el valor de UNB/0020.

Nota: los campos que no aparecen en esta lista ni se insertan ni se crean. Los valores correctos no se pueden determinar automáticamente. Dado un elemento primario **A** (en el componente EDI de destino) con los elementos secundarios **x**, **y**, y **z**, siendo **z** obligatorio, el elemento primario **A** solamente se creará en el archivo de salida si el elemento secundario obligatorio **y** tiene una asignación de datos.

En cada grupo funcional se comprueba:

- si UNG/0038 existe. Si no contiene datos, se inserta el nombre del mensaje.
- si UNG/S006 y UNG/S007 existen.
- si UNG/S004 existe. Si no contiene datos, se inserta la fecha/hora actual en formato EDI.
- si UNG/0051 existe. Si no contiene datos, se insertan los dos primeros caracteres de la agencia de control definida por el usuario. (Botón **Configuración extendida | Agencia de control** del cuadro de diálogo Configuración del componente EDI.)
- si UNE/0060 existe. Si no contiene datos, se calcula e inserta el número de mensajes del grupo.
- si UNE/0048 existe. Si no contiene datos, se copia el valor de UNG/0048.

En cada mensaje se comprueba:

- si UNH/S009/0065 existe. Si no contiene datos, se inserta el nombre del mensaje.
- si existen UNH/S009/0052 y UNH/S009/0054.
- si UNH/S009/0051 existe. Si no contiene datos, se insertan los dos primeros caracteres de la agencia de control definida por el usuario. (Botón **Configuración extendida | Agencia de control** del cuadro de diálogo Configuración del componente EDI.)
- si UNT/0074 existe. Si no contiene datos, se calcula e inserta el número de segmentos del mensaje.
- si UNT/0062 existe. Si no contiene datos, se copia el valor de UNH/0062.
- si UNH/0062 existe. Si no contiene datos, se copia el valor de UNT/0062.
  (Si solamente se asigna el número del segmento de finalizador, se envía el mismo valor al campo correspondiente del segmento de encabezado).

7.5.13.4 Reglas de validación UN/EDIFACT (interactivo)

Estas reglas de validación se aplican a componentes de MapForce que contienen mensajes EDIFACT interactivos:

- Si `UIB` está presente, entonces todos los `UIH/S302` deberían coincidir con `UIB/S302`
- `UIH/S306/F0065` debería contener el tipo de mensaje (validado por el analizador)
- `UIH/S306/F0052` debería contener el número de versión de mensaje que está definido en los archivos de configuración seleccionados
- `UIH/S306/F0054` debería contener el número de publicación de mensaje que está definido en los archivos de configuración seleccionados
- `UIT/F0340` debería coincidir con el `UIH/F0340` correspondiente (encabezado/finalizador de mensaje) o no estar presente (compuesto opcional)
- `UIT/F0074` debería contener recuento de segmentos o no estar presente (compuesto opcional)
- `UIZ/S302` debería coincidir con `UIB/S302` o no estar presente (compuesto opcional)
- `UIZ/F0036` debería contener recuento de mensajes o no estar presente (compuesto opcional)

7.5.13.5 Reglas de finalización automática UN/EDIFACT (interactivo)

Estas reglas de finalización automática se aplican a componentes de MapForce que contienen mensajes EDIFACT interactivos:

- `UIB/S001/F0001` se rellenará con la agencia de control y el número de versión de sintaxis

© 2018 Altova GmbH

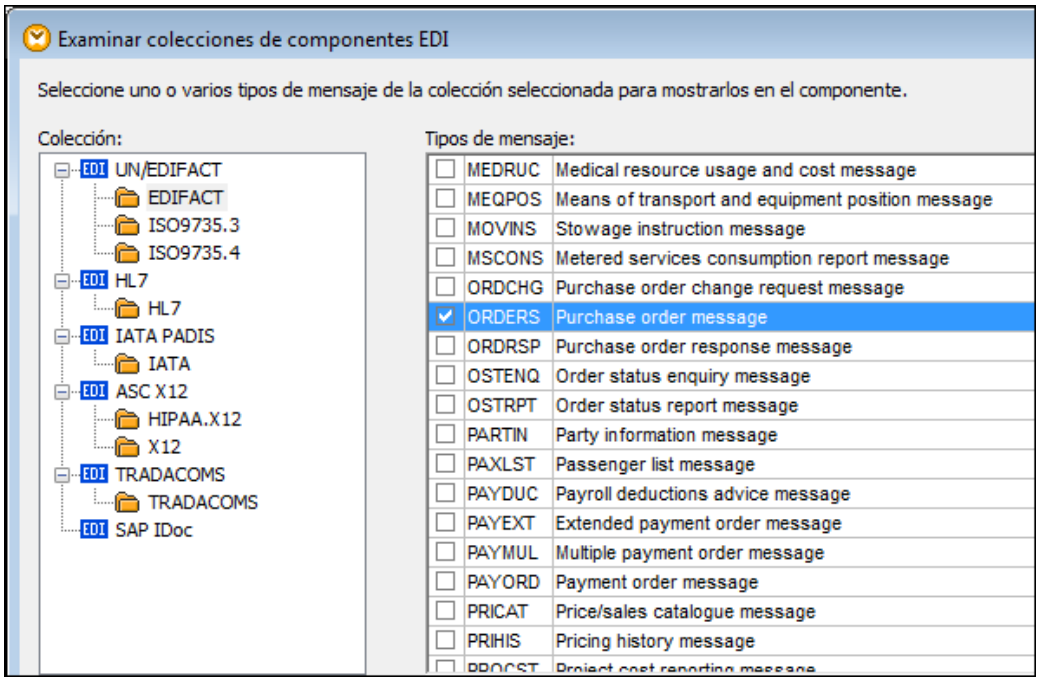
- `UIB/S001/F0002` se rellenará con el número de versión de sintaxis
- `UIB/S002/F0004` se rellenará con el remitente
- `UIB/S003/F0010` se rellenará con el destinatario
- `UIZ/S302/*` se rellenará con `UIB/S302/*`
- `UIZ/F0036` se rellenará con el recuento de mensajes
- `UIH/S302/*` se rellenará con `UIB/S302/*` (si `UIB` y `UIB/S302` están presentes)
- `UIH/S306/F0065` se rellenará con el tipo de mensaje
- `UIH/S306/F0052` se rellenará con el número de versión de mensaje de los archivos de configuración
- `UIH/S306/F0054` ser rellenará con el número de publicación de mensaje de los archivos de configuración
- `UIT/F0340` se rellenará con `UIH/F0340` (es un campo opcional pero se rellenará de todas maneras)
- `UIT/F0074` se rellenará con el número de segmentos del mensaje (es un campo opcional pero se rellenará de todas maneras)

7.5.13.6 Ejemplo: asignar datos UN/EDIFACT a XML

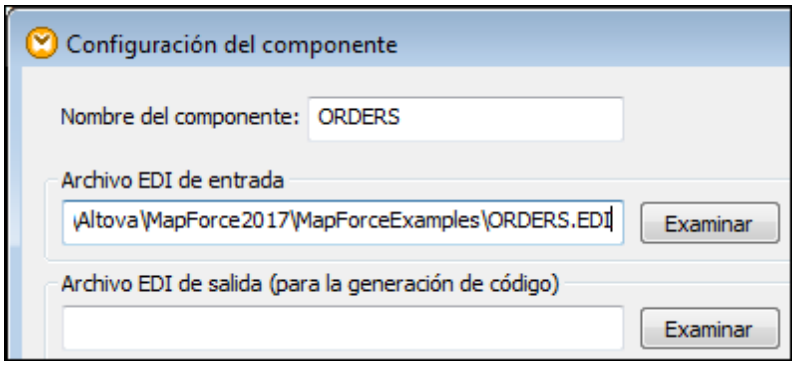
Este ejemplo explica cómo asignar datos de mensajes UN/EDIFACT a un esquema XML para producir una instancia XML que se pueda procesar más adelante. La asignación que se crea siguiendo los pasos de este apartado están en la ruta de acceso **<Documentos>\Altova \MapForce2018\MapForceExamples\EDI_Order.mfd**.

Paso nº1: agregar el componente UN/EDIFACT a la asignación

1. Cree una asignación nueva y seleccione uno de estos lenguajes de transformación: Java, C#, C++ o BUILT-IN. Para este ejemplo usamos el lenguaje de transformación Java.
2. En el menú **Insertar** seleccione el comando **EDI**.
3. En el cuadro de diálogo "Examinar colecciones de componentes EDI" seleccione la colección **EDIFACT** y después el tipo de mensaje **ORDERS**. Haga clic en **Aceptar** para confirmar.

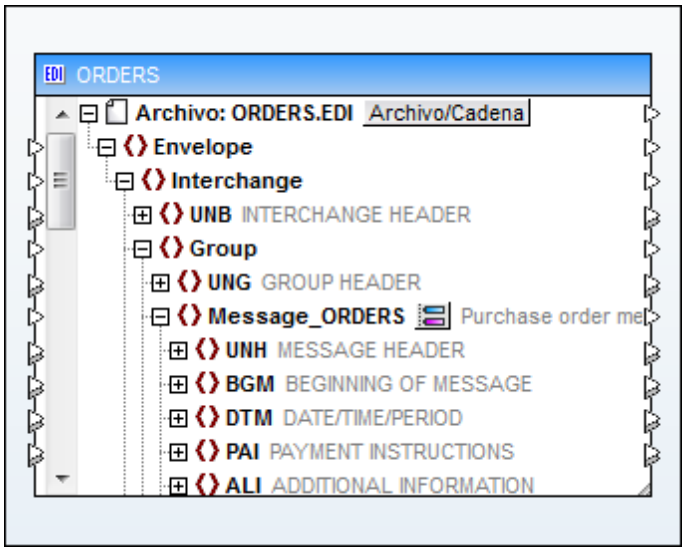


4. Cuando MapForce solicite un archivo EDI de muestra, haga clic en **Examinar** y abra el archivo **ORDERS.EDI** del directorio `<Documentos>\Altova\MapForce2018\MapForceExamples\`. Ahora aparece el cuadro de diálogo "Configuración del componente". Aquí puede revisar la configuración del componente EDI antes de añadirlo al área de asignación. Recuerde que estas opciones de configuración se pueden cambiar en todo momento (*véase* *Configuración de componentes EDI*). Observe que el archivo **ORDERS.EDI** aparece como *Archivo EDI de entrada*.



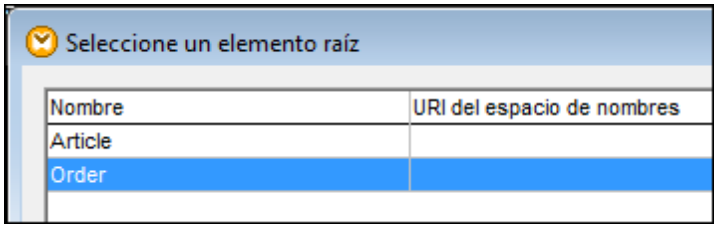
5. Haga clic en **Aceptar**. El componente EDI aparece en el área de asignación. Haga doble clic en el nodo **Message_ORDERS** para ver sus elementos secundarios. Para ajustar el tamaño del componente haga clic en la esquina inferior derecha de la ventana del componente y arrastre el puntero.

 © 2018 Altova GmbH

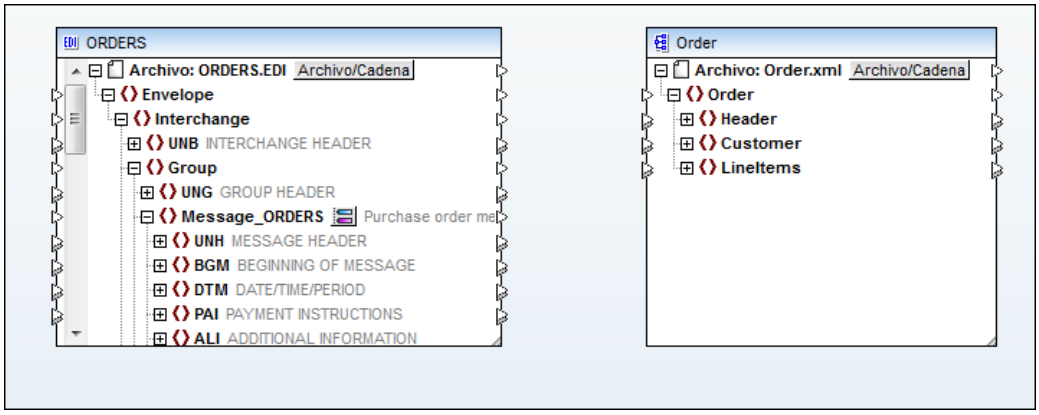


Paso nº2: agregar el componente de esquema de destino a la asignación

1. Seleccione el comando de menú **Insertar | Archivo o esquema XML** y seleccione el archivo **Order.xsd** del directorio `<Documentos>\Altova\MapForce2018\MapForceExamples\`.
2. Cuando MapForce solicite un archivo XML de muestra, haga clic en **Omitir** y seleccione el elemento raíz **Order** del documento de destino.



Llegados a este punto, tanto el componente EDI de origen como el esquema XML de destino aparecen en el área de asignación. De modo que podemos comenzar a crear conexiones entre los componentes.



Paso nº3: asignar los elementos EDI

El componente EDI muestra la estructura de un mensaje basado en la colección que seleccionamos (ORDERS). Por lo general, no todos los nodos contendrán datos así que lo recomendable es familiarizarse con los documentos EDI con los que se trabaja para poder encontrar los nodos relevantes con facilidad.

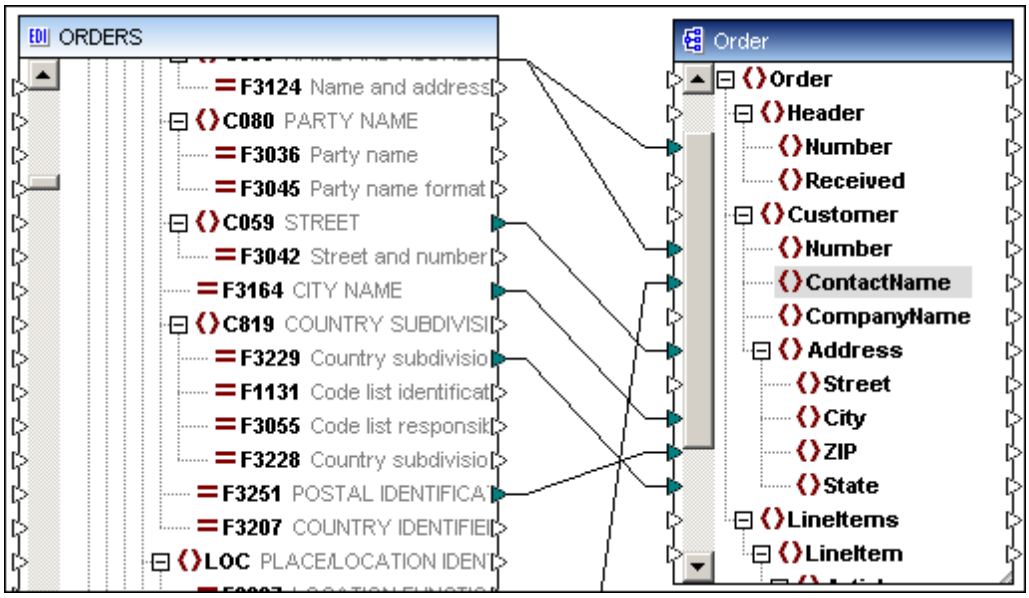
En MapForce las asignaciones de datos se definen conectando un elemento de origen con otro de destino. Para más información consulte la sección Trabajar con conexiones de la documentación.

En este ejemplo lo primero que haremos será conectar estos nodos (empezando por el nodo **Group/Message_ORDERS**):

Origen	Destino
BGM/C106/F1004	Order/Header/Number
SG2/NAD/C082/F3039	Order/Customer/Number
SG2/NAD/C080/F3036	Order/Customer/CompanyName
SG2/NAD/C059/F3042	Order/Customer/Address/Street
SG2/NAD/F3164	Order/Customer/Address/City
SG2/NAD/C819/F3229	Order/Customer/Address/State
SG2/NAD/F3251	Order/Customer/Address/ZIP
SG2/SG5/CTA/C056/F3412	Order/Customer/ContactName

Llegados a este punto la asignación tendrá este aspecto:

© 2018 Altova GmbH



Continúe con la asignación creando conexiones entre estos nodos/elementos:

Origen	Destino
SG29	Order/LineItems
SG29/LIN/C212/F7140	Order/LineItems/LineItem/Article/Number
SG29/IMD/C273/F7008	Order/LineItems/LineItem/Article/Name
SG29/QTY/C186/F6060	Order/LineItems/LineItem/Article/Amount
SG29/SG33/PRI/C509/F5118	Order/LineItems/LineItem/Article/SinglePrice

Paso nº4: dar formato a la fecha

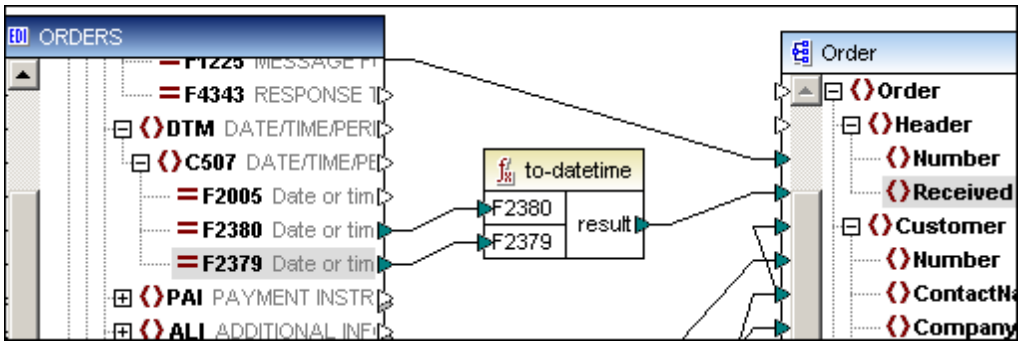
Arrastre la función **to-datetime** desde la biblioteca **edifact** hasta el área de asignación. Para más información consulte la sección Trabajar con funciones.

Al dar los componentes F2380 y F2379 del elemento DTM/C507 como argumentos de la función, podremos crear un valor de fecha hora **Received** con el formato adecuado.

Por tanto, debemos crear conexiones entre estos nodos/elementos:

Origen	Destino
DTM/C507/F2380	La entrada F2380 de la función **to-datetime**
DTM/C507/F2379	La entrada F2379 de la función **to-datetime**

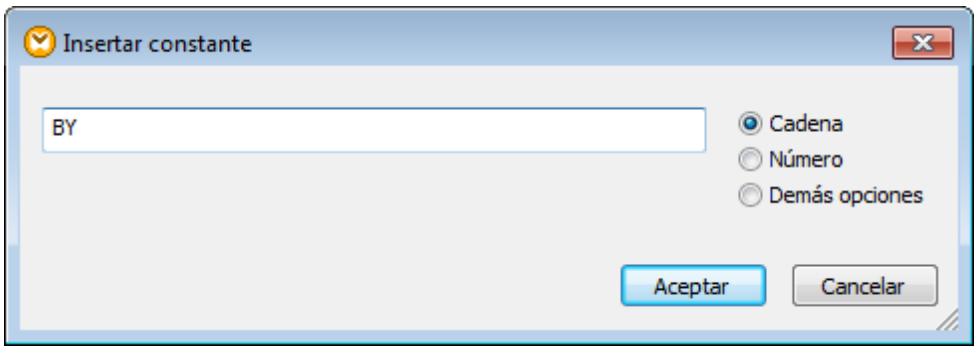
Origen	Destino
La salida **result** de la función **to-datetime**	Order/Header/Received



Paso nº5: filtrar las órdenes de compra del comprador

Llegados a este punto debemos filtrar las órdenes de compra Buyer. Estas se pueden identificar por medio del calificador de código de función de parte del segmento NAD (nombre y dirección). En este caso el valor BY indica que se trata de un comprador o Buyer (la parte a la que se vende la mercancía o servicio).

1. Arrastre la función **equal** de la biblioteca **core** hasta el área de asignación.
2. Añada un filtro a la asignación (con el comando **Insertar | Filtro: nodos/filas**).
3. Añada una constante a la asignación (con el comando **Insertar | Constante**). Asigne el valor BY a la constante:



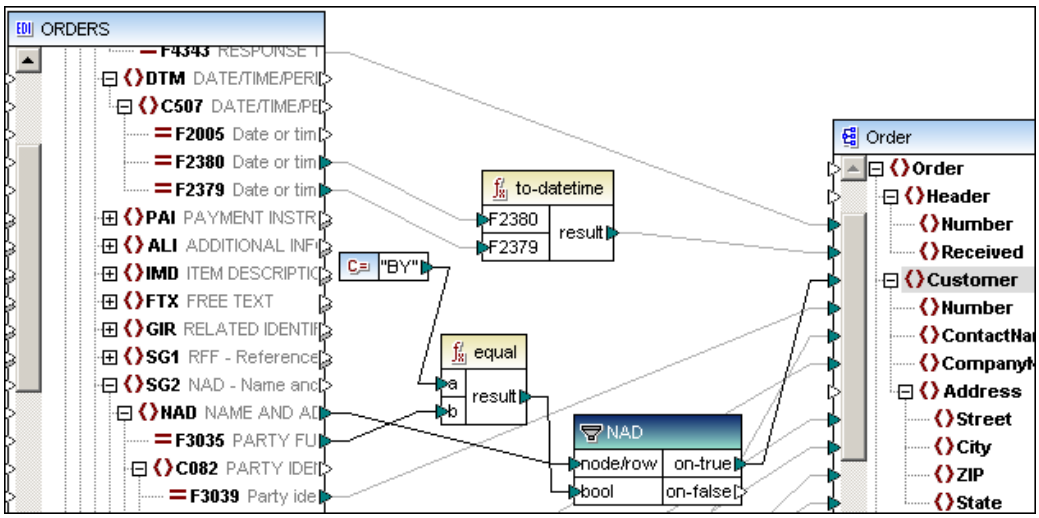
Ahora debemos crear conexiones entre estos nodos/elementos:

Origen	Destino
SG2/NAD/F3035	La entrada **b** de la función **equal**
La constante "BY"	La entrada **a** de la función **equal**

© 2018 Altova GmbH

Origen	Destino
El resultado de la función **equal**	La entrada **bool** del componente de filtrado
SG2/NAD	La entrada **node/row** del componente de filtrado
El resultado **on-true** del componente de filtrado	Order/Customer del esquema

El objetivo es asignar datos solamente si el nodo NAD remite a un comprador (tal y como identifica el calificador de código de función de parte BY).



Paso nº6: calcular el precio y las tasas

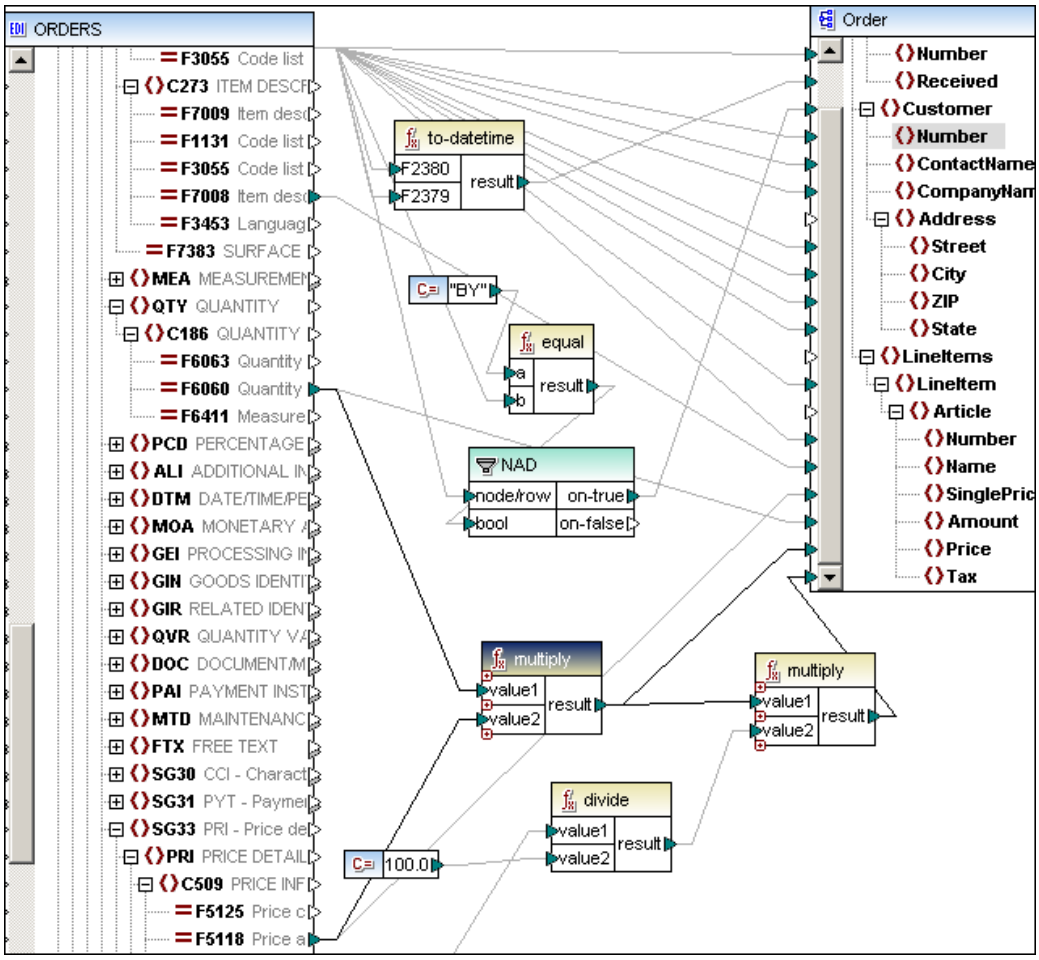
El último paso consiste en calcular el precio y las tasas.

1. Arrastre dos funciones **multiply** y una función **divide** de la biblioteca **core** hasta el área de asignación.
2. Inserte una constante (con el comando **Insertar | Constante**). Asegúrese de seleccionar el botón de opción *Número* y después introduzca el valor 100 en el campo de texto.
3. Ahora debe crear estas conexiones:

Origen	Destino
SG29/QTY/C186/F6060	**value1** de la primera función **multiply**
SG29/SG33/PRI/C509/F5118	**value2** de la primera función **multiply**
El resultado de la primera función **multiply**	Order/LineItems/LineItem/Article/Price
SG29/SG40/TAX/C243/F5278	**value1** de la función **divide**

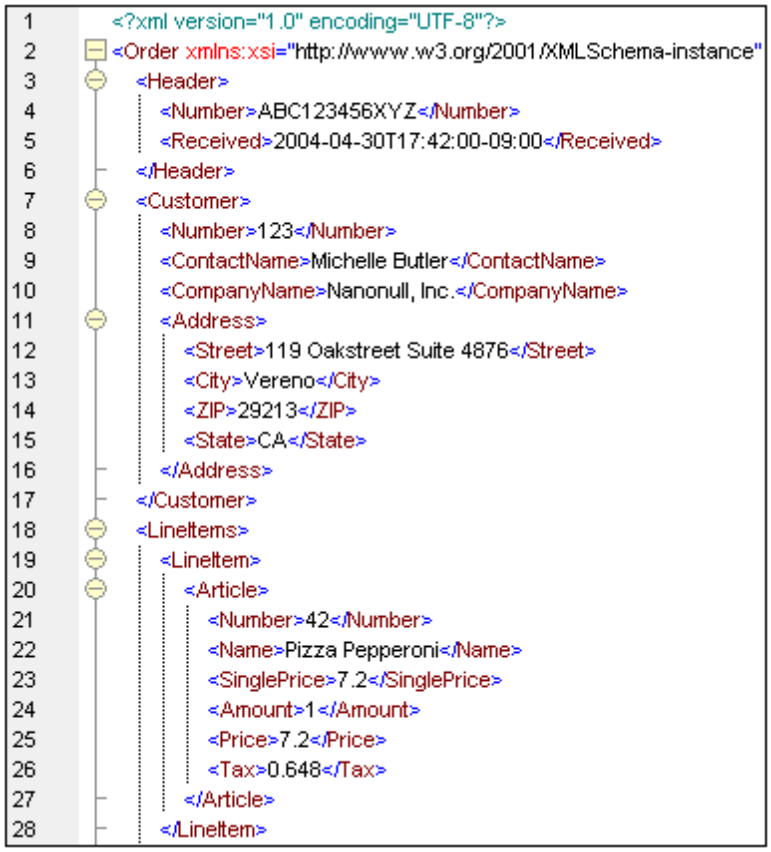
Origen	Destino
La constante "100.0"	**value2** de la función **divide**
El resultado de la primera función **multiply**	**value1** de la segunda función **multiply**
El resultado de la función **divide**	**value2** de la segunda función **multiply**
El resultado de la segunda función **multiply**	`Order/LineItems/LineItem/Article/Tax`

Llegados a este punto la asignación debería tener este aspecto:



Si ahora abrimos el panel *Resultados*, MapForce realizará una transformación instantánea de los datos y generará este documento XML de salida:

© 2018 Altova GmbH



```
<?xml version="1.0" encoding="UTF-8"?>
<Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  <Header>
    <Number>ABC123456XYZ</Number>
    <Received>2004-04-30T17:42:00-09:00</Received>
  </Header>
  <Customer>
    <Number>123</Number>
    <ContactName>Michelle Butler</ContactName>
    <CompanyName>Nanonull, Inc.</CompanyName>
    <Address>
      <Street>119 Oakstreet Suite 4876</Street>
      <City>Vereno</City>
      <ZIP>29213</ZIP>
      <State>CA</State>
    </Address>
  </Customer>
  <LineItems>
    <LineItem>
      <Article>
        <Number>42</Number>
        <Name>Pizza Pepperoni</Name>
        <SinglePrice>7.2</SinglePrice>
        <Amount>1</Amount>
        <Price>7.2</Price>
        <Tax>0.648</Tax>
      </Article>
    </LineItem>
```

7.5.14 Personalizar estructuras EDI

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

By default, MapForce represents the structure of EDI files on the mapping according to their formal specifications. In most cases, it shouldn't be necessary to manually customize the default EDI configuration. However, if you need to process various non-standard EDI formats and variations, then customization is required. For example, you may need to slightly adapt the structure or validation rules of certain data elements. This can be achieved by adding a custom EDI collection to MapForce, and then configuring it according to your business needs, which is the subject of this chapter.

You can create EDI variations (custom EDI collections) for the following standards already supported by MapForce: EDIFACT (ISO 9735), ASC X12, HIPAA X12, HL7, NCPDP SCRIPT, or TRADACOMS. Creating a new EDI standard from scratch is not supported.

7.5.14.1 Crear una colección EDI personalizada

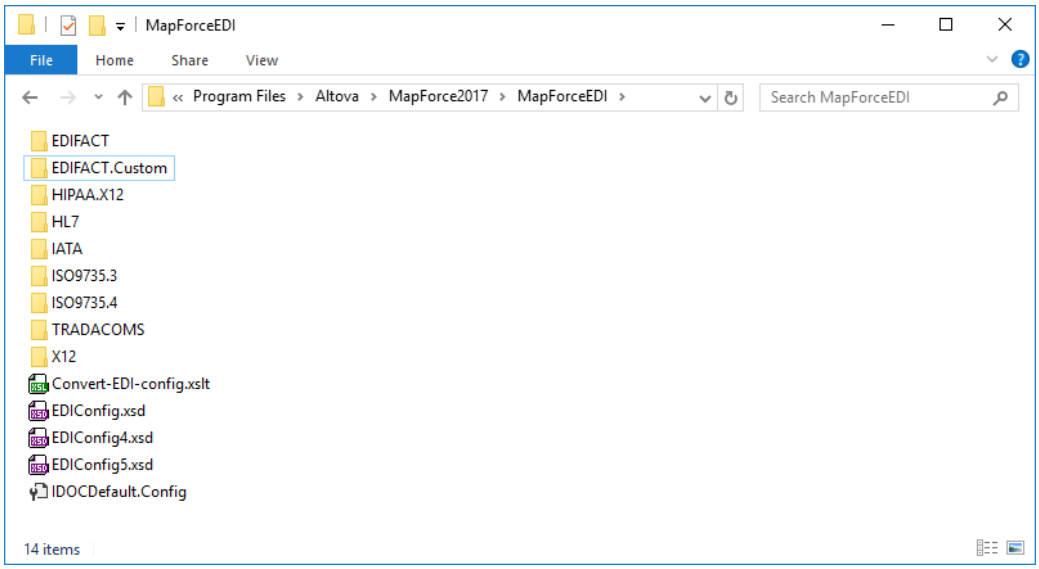
En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

An "EDI Collection" is a directory which contains all configuration files applicable to a specific EDI standard. By default, all built-in EDI collections exist in MapForce at the following path: **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI**. (If you run 32-bit MapForce on a 64-bit operating system, adjust the directory path to **C:\Archivos de programa(x86)\...**).

It is not recommended to alter the contents of original files and folders in the **MapForceEDI** directory, unless you want to override, at your own responsibility, the default EDI validation and processing in MapForce. The recommended approach is to add new custom directories rather than modifying the existing ones, as shown below. This will also make it easier to migrate your custom configuration to a newer version of MapForce.

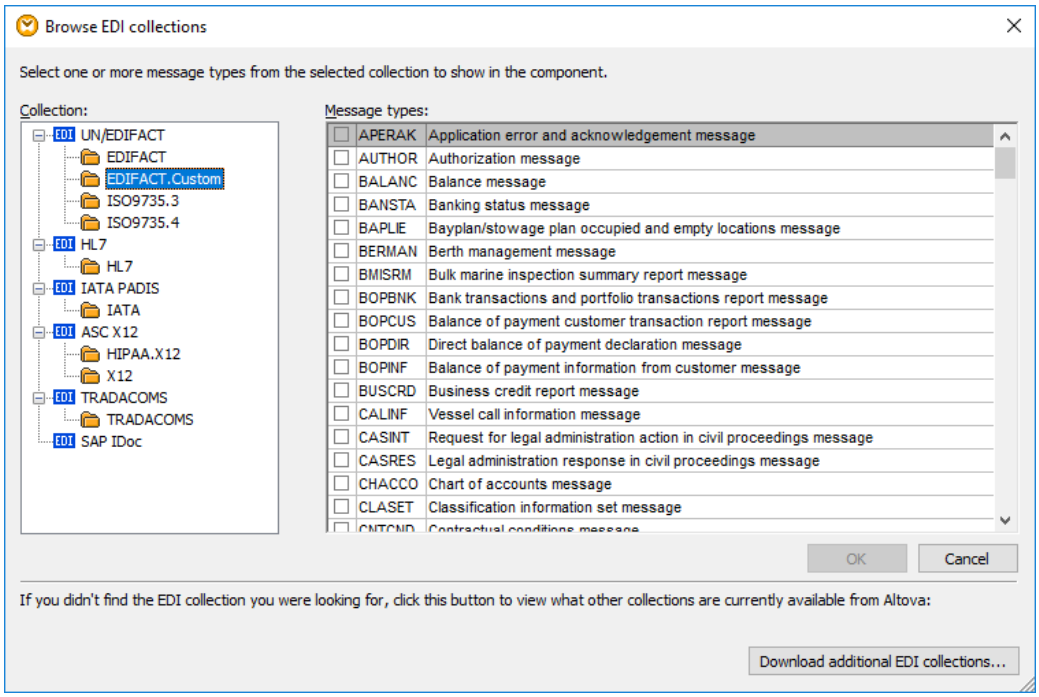
To create a custom EDI collection:

1. In the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI** directory, create a sibling copy of a directory that corresponds to the standard which you want to customize (for example "EDIFACT.Custom").



2. Open MapForce.
3. On the **Insert** menu, click **EDI**. At this stage, your custom EDI collection should appear in the "Browse EDI Collections" dialog box in addition to the default ones.

© 2018 Altova GmbH



Note: You need administrative rights to the C:**Archivos de programa** directory to modify it.

So far, the custom collection that was just created has not been customized yet in any way, so it is not different to the default one. In order to provide you the ability to translate data between your custom EDI format and any other data format supported by MapForce (in both directions), the EDI collection must be customized. This is discussed in the next sections. For now, it is important to remember that a directory must satisfy the following criteria in order to become an EDI collection recognized by MapForce:

1. It must be a subdirectory of **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI**).
2. It must be a copy of any of the following directories: EDIFACT, HIPAA.X12, HL7, NCPDP SCRIPT, TRADACOMS, or X12.

7.5.14.2 Archivos de configuración EDI

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

An EDI collection (directory) may contain any of the following configuration files:

File	Purpose
EDI.Collection	The **EDI.Collection** file defines all message types present in the custom EDI collection that you are implementing. You can edit this file to add or remove message types that should be available for selection in MapForce (on the "Browse EDI Collections" dialog box). See also Adding or Removing Message Types.
Envelope.Config	The **Envelope.config** file defines the structure of the EDI mapping component in MapForce (that is, elements that are surrounded by the **Envelope** sequence). Under normal circumstances, this file should not be edited manually.
<Message>.Config	The EDI collection directory must include one or several **<Message>.Config** files. One such file corresponds to an EDI message type, and contains the group and segment definitions used in that message. The **<Message>.Config** files are referenced from the **EDI.Collection** file. You can modify the **<Message>.Config files** if you want to alter the structure or validation rules applicable to fields of a particular message. Note that changes to these files are considered local customizations (that is, they apply only to that specific message). For examples, see Customizing an EDIFACT message and Customizing an ANSI X12 transaction.
<Filename>.Segment	This file defines the data elements, segments, composites, and subcomposites of the EDI files, and is used when parsing the EDI file. You can modify the .segment files if you want to alter the structure or validation rules applicable to a field (data element) in all messages where it may occur. Changes made to this file are global customizations.
<Filename>.Codelist	This file defines EDI code values used by MapForce to validate the EDI input and output files in MapForce. In general, the code values defined here correspond to those defined by the respective EDI standard. Exception to this rule are some edge cases where the standard relies on external code values and does not fully enforce the list of possible values (see also EDI component validation). You can modify .codelist files if you need to change the default validation rules (for example, in order to relax validation, or add new codes that are not already defined in the EDI code list). For more information, see Relaxing Field Validation and Adapting Code Values from External Sources.
ParserErrors.Config	This file is applicable to the X12 and HIPAA X12 standards only. It defines the structure of "ParserErrors_Message" and "ParserErrors_Group" items on the EDI component in MapForce. These items contain information about parsing errors encountered during file processing and can be used to produce X12 997 (Functional Acknowledgement) and X12 999 (Implementation

© 2018 Altova GmbH

File	Purpose
	Acknowledgement) messages.

All of the files above are XML files and can be edited in an XML editor such as XMLSpy. They all refer to some XML schema (see EDI Configuration Schemas), with the exception of the **EDI.Collection** file. The **EDI.Collection** file has no explicit schema defined but is validated internally by MapForce.

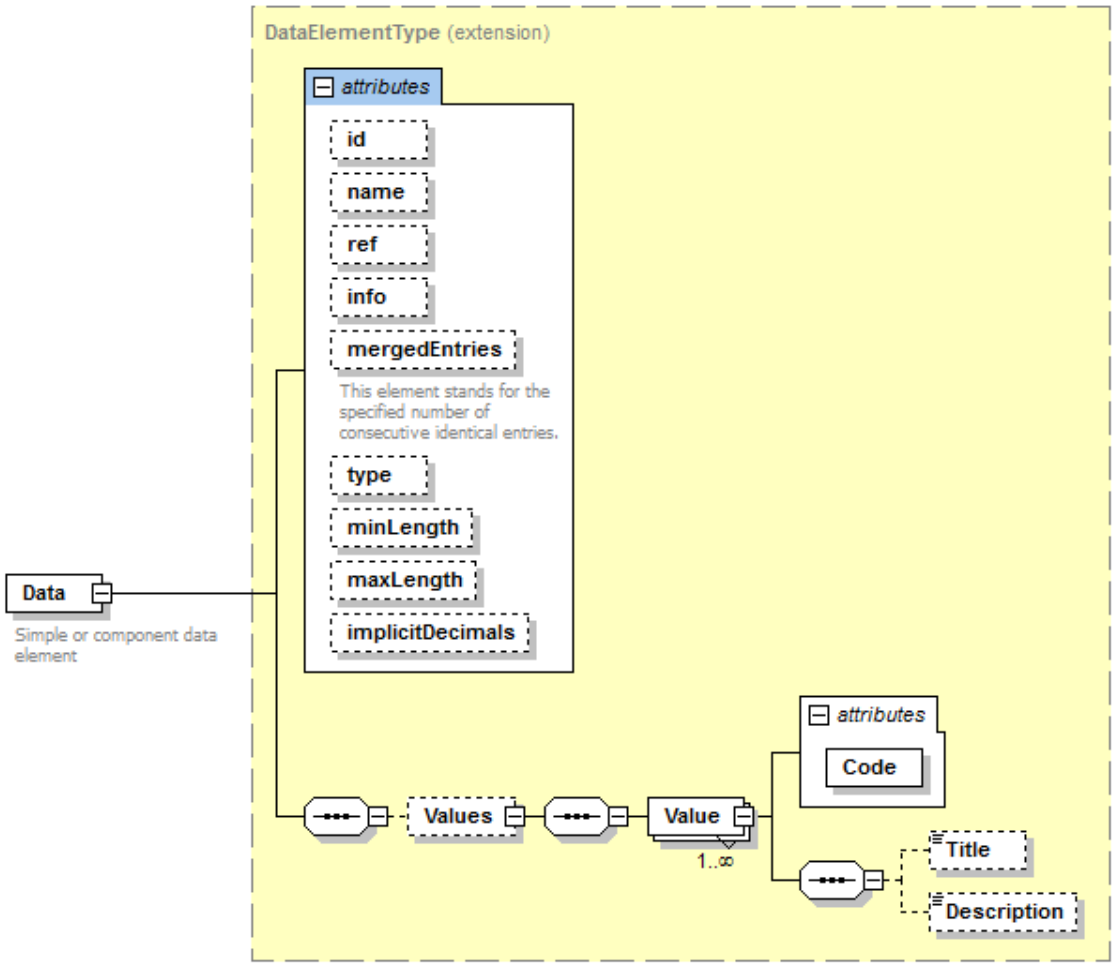
7.5.14.3 Esquemas de configuración EDI

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

Whenever you make a change to a configuration file, it is important to validate it against its applicable XML schema. The schema is available in the directory **C:\Program Files\Altova \MapForce2018\MapForceEDI** and it can be one of the following:

File	Purpose
EDIConfig.xsd (version "3")	This configuration schema is valid starting with MapForce 2011, Release 2. If you have configuration files that are using a schema prior to version 3, they must be upgraded to use this schema (or a newer one). For more information, see Upgrading Older Configuration Files.
EDIConfig4.xsd	This configuration schema is valid starting with MapForce 2011, Release 3. It contains multiple enhancements to the original **EDIConfig.xsd** schema, including the ability to define custom recursive hierarchies which might be necessary when you map data to or from X12 or HL7 formats.
EDIConfig5.xsd	This configuration schema is valid starting with MapForce 2017. It introduces the ability to define a data element that has incomplete validation rules. With this schema, it is possible to define advanced field validation rules at character level (see Configuring Character-Level Validation).
EDIConfig6.xsd	This configuration schema is valid starting with MapForce 2018. It introduces the map data to or from NCPDP SCRIPT format, see NCPDP SCRIPT .

The EDI configuration schema will also offer information about the structure of any elements that may be present in the configuration files. For example, below is a representation of an EDI Data Element structure as it appears in the **EDIConfig4.xsd** schema:



Structure of the Data element (Schema view of XMLSpy)

7.5.14.4 Actualizar archivos de configuración antiguos

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

EDI configuration files contain a reference to (and are validated against) one of the XML schemas available in the **MapForceEDI** directory (see EDI Configuration Schemas). To view the schema and version of a configuration file, open the file in a text editor and check the following line at the beginning:

© 2018 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd" Version="5">
```

MapForce 2018 recognizes custom EDI configurations that are based on **EDIConfig.xsd** schema version 3 or later, introduced in MapForce 2011, Release 3. Compared to previous version, this schema adds the ability to map data to or from multiple message types in the same EDI component. If you were using custom EDI configuration files that point to a schema version prior to version 3, then your configuration files must be upgraded. Otherwise, when you attempt to add the custom EDI collection to the mapping, the following message appears on the "Browse EDI Collections" dialog box:

> "The selected EDI collection uses an old configuration file version that supports only a single message type per component."

The following instructions show you how to upgrade the schema of an EDI configuration file from **EDIConfig.xsd** version 2, to **EDIConfig.xsd** version 3.

To upgrade the configuration files:

1. Copy **Envelope.Config** from the original configuration folder (for example, "EDIFACT") to the folder containing you customized EDI collection (for example, "EDIFACT.Custom").
2. Edit **EDI.Collection**, and change the root element's `Version` attribute from "2" to "3".
3. Add **`<Root File="Envelope.Config"/>`** after the `</Meta>` tag:

```
<?xml version="1.0" encoding="UTF-8"?>
<Messages xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="3">
  <Meta>
    <Version>D</Version>
    <Release>04B</Release>
    <Agency>UN</Agency>
    </Meta>
    <Root File="Envelope.Config"/>
    <Message Type="ORDERS" File="ORDERS.Config" Description="Purchase
order message"/>
</Messages>
```

4. Edit **ORDERS.Config**, and change the root element's `Version` attribute from "2" to "3".
5. Add **`<Format standard="EDIFACT"/>`** (or, depending on the case, X12, or HL7) after the `</Meta>` tag.
6. Rename **`<Group name="Message"...>`** to **`<Group name="Message_ORDERS"...>`** (or whatever the custom message type is), and remove the outer group levels ("Envelope", "Interchange", and their segments) as shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="3">
   <Meta>
      <Version>D</Version>
      <Release>04B</Release>
      <Agency>UN</Agency>
   </Meta>
   <Format standard="EDIFACT"/>
   <Include href="Admin.Segment"/>
   <Include href="EDSD.Segment"/>
   <Include href="UNCL.Codelist"/>
   <Message>
      <MessageType>ORDERS</MessageType>
      <Description>Purchase order message</Description>
      <Revision>14</Revision>
      <Date>2004-11-23</Date>
      <Group name="Envelope">
         <Group name="Interchange" maxOccurs="unbounded">
            <Segment ref="UNA" minOccurs="0"/>
            <Segment ref="UNB" minOccurs="0"/>
            <Group name="Group" maxOccurs="unbounded">
               <Segment ref="UNG" minOccurs="0"/>
               <Group name="Message_ORDERS" maxOccurs="unbounded"
info="UNH - Message header">
                  <Segment ref="UNH"/>
                  <Segment ref="BGM"/>
                        <Segment ref="UNT"/>
               </Group>
               <Segment ref="UNE" minOccurs="0"/>
            </Group>
            <Segment ref="UNZ" minOccurs="0"/>
         </Group>
      </Group>
   </Message>
</Config>
```

If a mapping was open while you were editing the configuration file, it should be reloaded. The connections will be automatically remapped from "Message" to "Message_ORDERS" item.

7.5.14.5 Agregar o eliminar tipos de mensaje

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

To add custom EDI message types (or transaction types, in X12 terminology), follow the steps

© 2018 Altova GmbH

below:

1. In the directory where your custom EDI collection is stored, create a **<Message>.Config** file for each message (transaction) type to be added to the collection.
2. Modify the **EDI.Collection** file to make a reference to each available message (transaction) type.

To remove existing message types:

- Comment out the required `Message` elements in the **EDI.Collection** file.

Adding and configuring the <Message>.Config file

Each **<Message>.config** file you create must be valid according to its configuration schema (see EDI Configuration Files). To avoid problems, pay special attention to the following elements in the body of each message configuration file:

- Meta
- Format
- Include
- Message

The `Meta` element indicates the specification version, release, and agency, for example:

```
<Meta>
   <Version>D</Version>
   <Release>14B</Release>
   <Agency>UN</Agency>
</Meta>
```

The `Format` element indicates the name of the base standard, for example:

```
<Format standard="EDIFACT"/>
```

The following values are valid:

- Fixed
- EDIFACT
- X12
- HL7
- TRADACOMS (this value requires the **EDIConfig4.xsd** schema or later, see EDI Configuration Files and Schemas)

The `Include` element in a **<Message>.Config** file specifies references to any additional configuration files. For example, the code listing below includes a reference to two .segment files and one .codelist file. A .segment file stores all Segment, Composite, and Data Element definitions at a global level (this is to avoid repeating them in each message where they may occur). Likewise, a .codelist file defines validation rules applicable to each field, at a global level.

```
<Include href="Admin.Segment"/>
<Include href="EDSD.Segment"/>
<Include href="UNCL.Codelist"/>
```

Any file referenced by the `Include` element must exist in the same directory as the **<Message>.Config** file.

The `Message` element in each **<Message>.Config** file defines the structure of the message (that is, the hierarchy of groups, segments, and data elements), as well as any code values used for validation (if applicable). For more information about the structure of a message, see Changing the Message Structure. For examples of altering existing messages, see the following topics:

- Customizing an EDIFACT Message
- Customizing an ANSI X12 Transaction
- Defining Code Values at Message Level (Inline)

Modifying the EDI.Collection file

If you add a custom **<Message>.Config** file, a `Message` element must also be added to the **EDI.Collection** file. Each `Message` element must have the following attributes:

Type	Specifies the official code of the message type (for example "BALANC", "999").
File	Specifies the file name containing the specification of this particular message type. The actual file must exist in the same directory.
Description	Provides a description of this message type as it will appear in the MapForce graphical user interface.

For samples of valid **EDI.Collection** files, check any of the MapForce built-in EDI collections in the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI** directory.

7.5.14.6 Cambiar estructura del mensaje

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

In each **<Message>.config** file, the structure of a `Message` element is determined by the XML schema of the EDI standard that you are customizing. This schema is called **EDIConfig.xsd** (or a similar name, see EDI Configuration Schemas). To build or change the structure of the message, you will therefore need to consult both the documentation of your custom EDI standard and the XML schema.

© 2018 Altova GmbH

How to add segments, data elements, and composites

To add data elements, segments, composites, and subcomposites, create `<Data>`, `<Segment>`, `<Composite>`, and `<SubComposite>` elements as follows:

- The .segment file must contain the list of all reusable, globally-defined segments and data elements.
- The .config file corresponding to each message (or transaction) type must contain the actual structure of that message (the hierarchy of segments, composites, and data elements).

This makes it possible to define a data element once (in the .segment file) and reuse it in multiple messages (in .config files).

To refer to a data element, use the `ref` attribute. For example, the code listing below shows how data element F100 is defined in the **X12.Segment** file of the HIPAA.X12 collection.

```
<Data name="F100" type="string" maxLength="3" minLength="3" info="Currency
Code" />
```

X12.Segment file

All segments, in all transaction types that need this data element, can now refer to it. The code listing below shows how the "CUR" segment of the "Health Care: Professional" (837P) transaction refers to the F100 data element previously defined:

```
<Include href="X12.Segment" />

...

<Segment name="CUR" minOccurs="0" info="Foreign Currency Information">
    <Data ref="F100" info="Currency Code" />
</Segment>
```

837P.Config file (X12)

A reference can be in the same file, or in an external file. In the code listing above, the .config file includes a reference to an external .segment file. For this reason, an **`<Include href="X12.Segment" />`** statement is present at the beginning of the **837P.Config** file to add the reference to the external **X12.Segment** file.

For another example that shows how to add a custom segment, see Example: Adding the TA1 Segment.

How to create groups and loops

Creating or changing EDI structures often requires grouping of segments or data elements under some parent node (a group). Even the message on the EDI component is a group, since it groups together all the multiple segments that this message consists of. Groups may be nested (they can have child groups). Also, they may be repeated multiple times (in which case, they become loops). Groups and loops can be created by adding a `<Group>` element. Use the `maxOccurs`

attribute to set the number of allowed occurrences of a loop. For example, the following code listing shows a loop that can occur up to 25 times, is optional, and consists of two segments:

```
<Group name="LoopSAC" maxOccurs="25" minOccurs="0">
   <Segment ref="SAC"/>
   <Segment ref="CUR" minOccurs="0"/>
</Group>
```

850.Config file (X12)

How to specify the type and size of data elements

Use the `type` attribute of a data element or segment to specify its type (string, decimal, etc.). The possible data types (the `type` attribute) are generally any of XML Schema types that are used in the configuration files delivered by default. Other XML Schema simple types can tentatively be used, but cannot be guaranteed. The following simple types are not supported: "anyType", "ENTITIES" and "QName".

Use the `minLength` and `maxLength` attributes of a data element to specify its minimum and maximum length.

How to specify mandatory and optional data elements

To specify that a data element is mandatory, do not set the value of the `minOccurs` attribute (this attribute has "1" as default value, which means the field is mandatory by default). To specify that a data element is optional, set the `minOccurs` attribute to "0". For example, the code listing below defines two mandatory data elements (F1331, F1332) and one optional data element (F1325).

```
<Composite name="C023" info="Health Care Service Location Information">
   <Data ref="F1331"/>
   <Data ref="F1332"/>
   <Data ref="F1325" minOccurs="0"/>
</Composite>
```

7.5.14.7 Varios elementos de datos consecutivos

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

EDI segments often allow the use of multiple consecutive elements of the same name. For example, this is applicable to the N2 segment of the "850 - PURCHASE ORDER" transaction set of the X12 standard. According to the specification, the N2 segment allows for two consecutive alphanumeric "Name" fields (one mandatory and one optional):

© 2018 Altova GmbH

REF	ELE ID	NAME	RPT	ATTRIBUTES
01	93	Name	M	AN 1/60
02	93	Name	O	AN 1/60

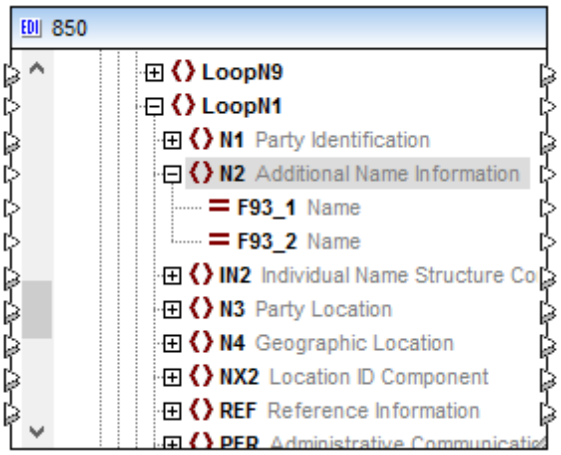
When you create or modify an EDI configuration, there are two possibilities to handle such data elements, as shown below.

Approach 1: Create separate mapping nodes for each consecutive data element

In this approach, you create multiple occurrences of the data element in the segment file. In this case, a new node will appear on the mapping for each occurrence of the data element. Consider this configuration:

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" nodeName="F93_1"/>
   <Data ref="F93" minOccurs="0" nodeName="F93_2"/>
</Segment>
```

On the mapping, this configuration creates two separate F93 nodes, each corresponding to an occurrence of F93:



X12 EDI component with separate F93 nodes

Using this approach is suitable when every occurrence of the data element is meaningful by itself and, consequently, you want a separate node to appear for it on the mapping.

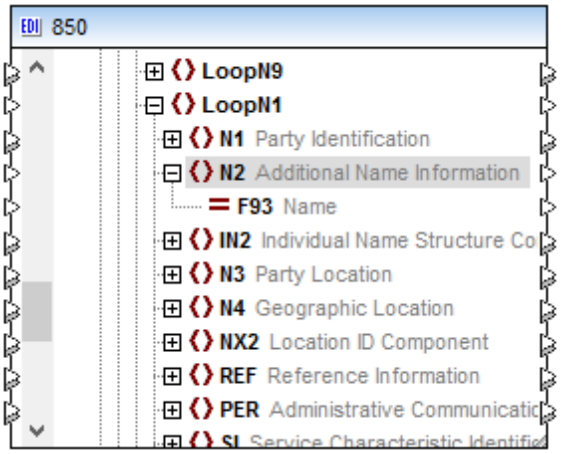
Approach 2: Create a single node for all occurrences of the data element

In this approach, you treat all occurrences of the data element as one single node. To do this, add the `mergedEntries` attribute to the corresponding data element. In the code listing below, the

`mergedEntries` attribute of data element F93 is set to "2". This allows for two consecutive occurrences of this field in the EDI instance file.

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
</Segment>
```

On the mapping, the node corresponding to the F93 data element will be displayed as a single node:



X12 EDI component with separate F93 nodes

Mapping data from a data element which has merged entries will create multiple duplicate elements on the target side, if the source EDI file contains multiple (not empty) occurrences of this element. For example, let's consider a source EDI file where the N2 segment contains two consecutive occurrences of F93. The first occurrence is "Michelle Butler" and the second is "Mrs".

```
N2+Michelle Butler+Mrs+'
```

Mapping data from such an EDI file would create, on the target side, the following output (notice the duplicate occurrences of <ContactName>):

```
<Customer>
      <Number>123</Number>
      <ContactName>Michelle Butler</ContactName>
      <ContactName>Mrs</ContactName>
      <CompanyName>Nanonull, Inc.</CompanyName>
</Customer>
```

Using this approach is suitable when each occurrence of the data element is not meaningful by itself, and you do not want a separate node to appear on the mapping for each occurrence. For example, a data element with multiple occurrences may store multiple line entries that make up an address. For such cases, the "merged entries" approach might be more useful than having individual nodes for each address line.

 © 2018 Altova GmbH

Some configuration files supplied by default with MapForce use merged entries, and therefore, only one node appears on the mapping for multiple consecutive occurrences of a data element. If you need separate nodes for each occurrence of the data element, this can be achieved by customizing the EDI configuration. For an example, see Splitting Merged Entries into Separate Nodes.

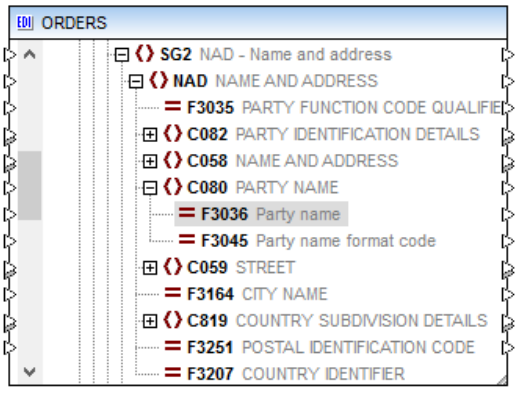
7.5.14.8 *Dividir entradas combinadas en distintos nodos*

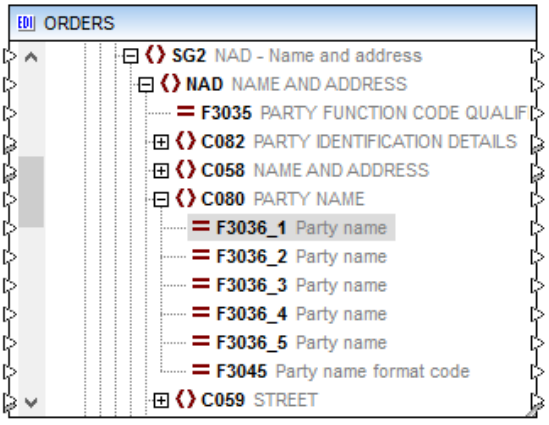
En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

Some data elements in the EDI configuration files supplied by default with MapForce are configured to display only one node on the mapping, even if the corresponding data element may occur multiple times in the source EDI file (see also Handling Multiple Consecutive Data Elements). An example of such data element is F3036 of the UN/EDIFACT standard. According to the specification, F3036 can occur up to five times consecutively in the C080 ("Party name") composite (see http://www.unece.org/fileadmin/DAM/trade/untdid/d15b/trcd/trcdc080.htm).

In the default UN/EDIFACT collection of MapForce, the data element F3036 is represented through a single node on the mapping. If you draw a mapping connection from this node to a target node, then the connection would create as many items in the target as there are occurrences of F3036 in the source. (This follows the basic rule of MapForce connections, for each item in the source, create one item in the target). However, there may be situations where this is not the desired result, and you need separate nodes for each occurrence of the data element.

The goal of the example is, therefore, to create a custom UN/EDIFACT configuration which would display separate mapping nodes for each occurrence of F3036. This would enable you to map each individual occurrence of F3036 to a separate target node. The images below show the EDIFACT mapping component before and after customization.





Before customization *After customization*

The steps required to split the F3036 data element into multiple mapping nodes are as follows:

1. Create a custom EDI collection using the existing UN/EDIFACT collection as starting point (see Creating a Custom EDI Collection). This ensures your changes will not affect in any way the default EDI configuration.
2. Open the **EDSD.Segment** file of your custom collection in an XML editor. (If you want to customize a different EDI flavour, then edit the corresponding .segment file of that particular EDI flavour. For example, for X12, this would be the **X12.Segment** file.)
3. Find the composite C080 and remove the `mergedEntries="5"` attribute of the F3036 data element.

```
<Composite name="C080" info="PARTY NAME">
   <Data ref="F3036" mergedEntries="5"/>
   <Data ref="F3045" minOccurs="0"/>
</Composite>
```

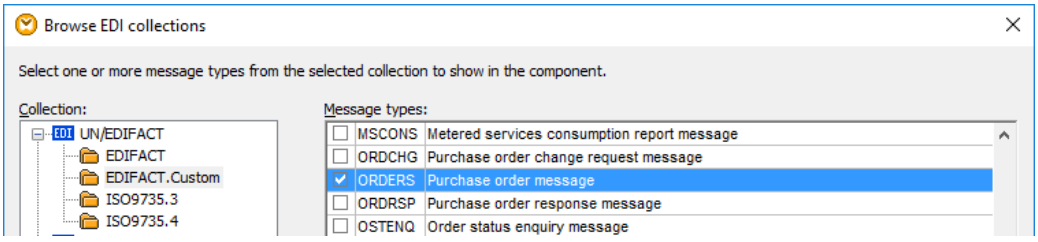
4. Add the `minOccurs="0"` attribute to all fields except for the first one. The reason is that, according to the UN/EDIFACT standard, only the first occurrence of the field is mandatory; the others are conditional.
5. Enter a unique node name for each of the fields (for example, F3036_1, F3036_2, and so on). At this stage, the configuration should look as follows:

```
<Composite name="C080" info="PARTY NAME">
   <Data ref="F3036" nodeName="F3036_1"/>
   <Data ref="F3036" nodeName="F3036_2" minOccurs="0"/>
   <Data ref="F3036" nodeName="F3036_3" minOccurs="0"/>
   <Data ref="F3036" nodeName="F3036_4" minOccurs="0"/>
   <Data ref="F3036" nodeName="F3036_5" minOccurs="0"/>
   <Data ref="F3045" minOccurs="0"/>
</Composite>
```

6. Save the edited **EDSD.Segment** file.

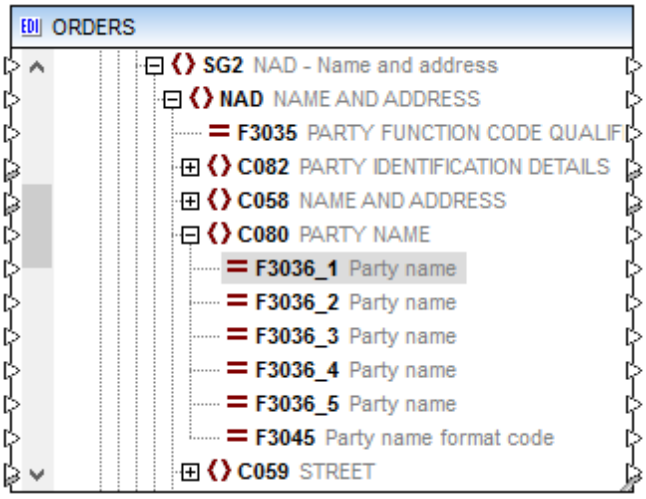
You can now view and work with the modified EDI collection in MapForce, as follows:

1. On the **Insert** menu, click **EDI**.
2. When prompted, select the custom collection created previously (in this example, "EDIFACT.Custom").
3. Select any message type that uses the C080 segment (in this example, "ORDERS - Purchase Order Message").



 © 2018 Altova GmbH

The EDI component now appears modified in MapForce (each occurrence of the F3036 data element has a separate node on the mapping).



The technique described in this example can also be used to split merged entries for other data elements, regardless of the EDI flavour.

7.5.14.9 Cambiar posibles valores de código de elementos de datos

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

The possible values of a data element can be changed by means of configuration files. Similar to other configuration information, you can configure the possible code values at each message level (local customization), or globally (across all messages that may use that particular data element).

Like with any other custom configuration, to customize code values, it is recommended to create a custom EDI collection (see Creating a Custom EDI Collection) and apply all customization to it, rather than altering the default MapForce configuration.

The available customization options depend on the schema version used by your custom EDI collection (see EDI Configuration Schemas). To put it simply, a newer schema means that more advanced configuration options are available. In particular, each schema supports custom code values as follows:

- With **EDIConfig.xsd**, it is possible to change an existing .codelist file in order to add custom code values at a global level.
- With **EDIConfig4.xsd**, custom code values can be added not only at global level, but also at message level.

- With **EDIConfig5.xsd**, all of the options above are available, including the ability to define advanced configuration rules at character level (see Configuring Character-Level Validation). Also, data elements can be declared as having incomplete validation (see Relaxing Field Validation).

Global customization

To add custom code values for a data element globally, edit the .codelist file of your custom EDI collection. First, find the data element (field) whose values you need to change, and then add all the required custom values as new <Value> elements, as shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Include href="ISO6346.Codelist"/>
   <Codelist>
      <Field Name="F1001" Length="3" Type="Alphanumeric">
         <Title>Document name code</Title>
         <Description>Code specifying the document name.</Description>
         <Values>
            <Value Code="1">
               <Title>Certificate of analysis</Title>
               <Description>Certificate providing the values of an analysis.</
Description>
            </Value>

>>> add your custom values here...

         </Values>
      </Field>
   </Codelist>
</Config>
```

UNCL.Codelist file (UN/EDIFACT collection)

As shown in the code listing above, each <Value> element has a Code attribute which specifies the actual value code expected for this data element. The <Title> and <Description> elements provide additional descriptive information about this value code.

The code listing above also has an Include statement which instructs MapForce to look for additional code values defined in an external file (**ISO6346.Codelist**) which uses the same schema as the **UNCL.Codelist** file. Use this technique if you need to adapt various external files containing custom code values.

For an example, see Adapting Code Values from External Sources.

Local customization

To add custom code values at message level, you will need to edit the .config file corresponding to the message type. For more information, see Defining Code Values at Message Level (Inline).

© 2018 Altova GmbH

7.5.14.10 Ejemplo: agregar un segmento TA1

El segmento TA1 es un segmento opcional que se utiliza para confirmar la recepción del intercambio y que la sintaxis de sus segmentos de sobre es correcta. El segmento se puede añadir a los componentes EDI X12 y EDI HIPAA.

Para incluir el segmento TA1 en un componente EDI:

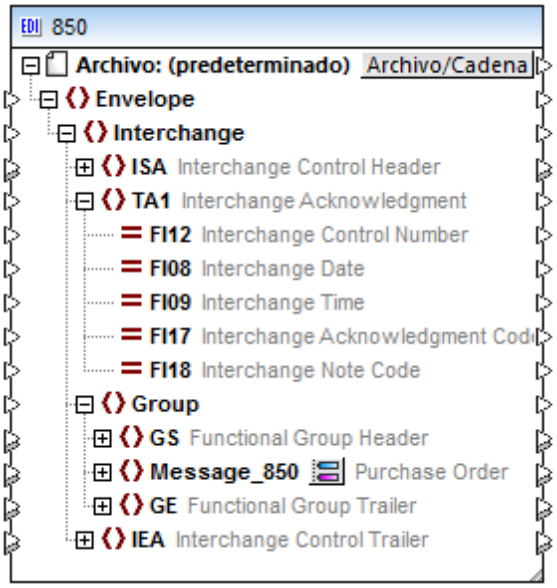
1. Cree una colección EDI personal usando como modelo la colección HIPAA.X12 (*véase* *Crear una colección EDI personalizada*). Así evitaremos que los cambios afecten a la configuración predeterminada original.
2. Abra el archivo **Envelope.config** de la colección EDI personal en un editor XML.
3. Pegue el texto `<Segment ref="TA1" minOccurs="0" maxOccurs="unbounded"/>` entre los segmentos ISA y Group y guarde el archivo. El archivo debería tener este aspecto:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="../EDIConfig.xsd" Version="3">
   <Meta>
      <Release>6020</Release>
      <Agency>X12</Agency>
   </Meta>
   <Format standard="X12"/>
   <Include href="X12.Segment"/>
   <Include href="X12.Codelist"/>
   <Include collection="EDI.Collection"/>
   <Group name="Envelope">
      <Group name="Interchange" maxOccurs="unbounded">
         <Segment ref="ISA" minOccurs="0"/>
         <Segment ref="TA1" minOccurs="0" maxOccurs="unbounded"/>
         <Group name="Group" maxOccurs="unbounded">
            ...
         </Group>
         <Segment ref="IEA" minOccurs="0"/>
      </Group>
   </Group>
</Config>
```

La línea que aparece resaltada en amarillo añade una referencia al segmento TA1 que ya está definido en el archivo **X12.Segment**. Ahora puede ver la estructura del segmento TA1 en MapForce:

1. En el menú **Insertar** haga clic en el comando **EDI**.
2. En el cuadro de diálogo "Examinar colecciones de componentes EDI" seleccione la colección que acaba de crear y un tipo de mensaje (p. ej. `850 - Purchase Order`).

Observe que el componente personalizado ya incluye el segmento TA1. Ahora puede dibujar conexiones de asignación entre este y otros componentes.



Componente X12 EDI personalizado con segmento opcional TA1

7.5.14.11 Ejemplo: personalizar un mensaje EDIFACT

Este ejemplo explica cómo personalizar MapForce para que pueda procesar formatos EDIFACT no estándar o modificados. Para el ejemplo se usa la asignación **Orders-Custom-EDI.mfd** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**. Recuerde que si abre la asignación antes de seguir las instrucciones que aparecen más abajo, MapForce considerará que el archivo no es válido. El motivo es que la asignación hace referencia a una configuración EDIFACT personalizada que no existe todavía en su equipo (se crea durante el ejemplo).

No olvide que, si abre esta asignación antes de seguir las instrucciones de personalización que aparecen más abajo, MapForce considerará que la asignación no es válida. El motivo es que la asignación remite a una configuración EDIFACT personalizada que todavía no existe en el equipo (porque la crearemos en el ejemplo de más abajo).

La asignación de datos **Orders-Custom-EDI.mfd** lee datos del archivo **Orders-Custom.EDI** (que está en el directorio **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**). Si abre este archivo con un editor de texto, verá que utiliza un segmento CTA personalizado. Concretamente, la línea 9 contiene la entrada **Mr** y la línea 11 contiene la entrada **Mrs**:

© 2018 Altova GmbH

```
 1    UNB+UNOB:1+003897733:01:MFGB+PARTNER ID:ZZ:ROUTING
      ADDR+970101:1230+00000000000001++ORDERS++++1'
 2    UNH+0001+ORDERS:S:93A:UN'
 3    BGM+221+ABC123456XYZ+9'
 4    DTM+4:200404301742PDT:303'
 5    FTX+PUR+3++Pizza purchase order'
 6    RFF+CT:123-456'
 7    RFF+CR:1122'
 8    NAD+SE+999::92++24h Pizza+Long Way+San-Francisco+CA+34424+US'
 9    CTA+SR+:Ted Little:Mr'
10    NAD+BY+123::92++Nanonull, Inc.+119 Oakstreet Suite 4876+Vereno+CA+29213+US'
11    CTA+PD+:Michelle Butler:Mrs'
12    NAD+ST+123::92++Nanonull, Inc.+119 Oakstreet Suite 4876+Vereno+CA+29213+US'
13    TAX+9++++++1-12345-6789-0'
14    CUX+2:USD:9'
15    PCD+12:2'
16    TDT+20++++:::24h Pizza Fast Carrier'
17    LOC+16+Nanonull Main Entrance'
```

Para poder asignar datos de archivos EDI personalizados como el anterior primero debemos crear una configuración EDIFACT personalizada. En este ejemplo personalizaremos la configuración EDIFACT predeterminada para ampliar el segmento CTA con un campo nuevo que recibirá el título de la persona. Puede consultar el resultado final del procedimiento de personalización en el directorio **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial** (en el archivo ZIP **EDIFACT.Nanonull.zip**). Para poder seguir las instrucciones que aparecen a continuación se entiende que todavía no se desempaquetó el archivo ZIP en el directorio **..\Archivos de programa\MapForceEDI**.

Configurar los archivos de personalización

1. Abra el directorio de instalación de MapForce (**C:\Archivos de programa\Altova \MapForce2018**). Si ejecuta MapForce de 32 bits en un sistema operativo de 64 bits, utilice la ruta de acceso **C:\Archivos de programa (x86)\Altova\MapForce2018 \MapForceEDI**.
2. Ahora, dentro del directorio **..\MapForceEDI** debe crear un directorio llamado **EDIFACT.Nanonull**.
3. Copie estos archivos de la carpeta ...**MapForceEDI\EDIFACT** dentro de la carpeta **EDIFACT.Nanonull**.

   - **EDI.Collection**
   - **Envelope.Config**
   - **UNCL.Codelist**
   - **Admin.Segment**
   - **EDSD.Segment (Electronic Data Segment Definition)**
   - **ORDERS.Config**

   Para saber qué finalidad tiene cada archivo consulte el apartado Archivos de configuración EDI.

Nota: cambie los atributos de los archivos por **read-write** para poder editarlos.

Configurar el archivo EDI.Collection

1. Abra el archivo **EDI.Collection** en un editor XML como XMLSpy.

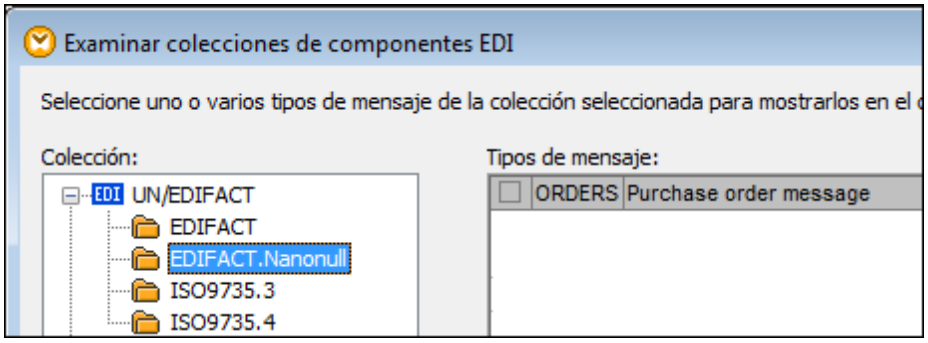
2. Elimine todos los elementos "Message", excepto el mensaje "ORDERS", pero sin eliminar las etiquetas <Messages>.

```
<?xml version="1.0" encoding="UTF-8"?>
<Messages Version="3">
   <Meta>
      <Version>D</Version>
      <Release>16A</Release>
      <Agency>UN</Agency>
   </Meta>
   <Root File="Envelope.Config"/>
   <Message Type="ORDERS" File="ORDERS.Config" Description="Purchase
order message"/>
</Messages>
```

3. Guarde el archivo.

Ahora podrá acceder a este nuevo archivo **EDI.Collection** desde MapForce:

1. Inicie MapForce.
2. En el menú **Insertar** haga clic en el comando **EDI** (o haga clic en el botón **Insertar EDI** de la barra de herramientas). Esto abre un cuadro de diálogo donde aparece una colección nueva llamada "EDIFACT.Nanonull". Si seleccionamos esta colección, en el panel derecho solo aparece un tipo de mensaje: "ORDERS".



Nota: cuando se inicia, MapForce escanea todas las subcarpetas del directorio **...\MapforceEDI** y busca un archivo llamado "EDI.Collection". Cada carpeta que contenga un archivo **EDI.Collection** aparecerá en el cuadro de diálogo de la imagen anterior. La lista de tipos de mensaje muestra el contenido del archivo de colecciones. El archivo de nuestro ejemplo solo contiene un mensaje ORDERS.

Diferencias entre personalización global e insertada (local)

En este ejemplo el objetivo es redefinir el segmento CTA (Contact Information), que está formado por un campo (F3139) y un compuesto (C056). Para almacenar la información sobre el título (p. ej. "Mr" o "Mrs"), añadiremos un campo nuevo al compuesto C056, que llamaremos X1000. Hay varias maneras de realizar esta operación de personalización:

- De forma **global**, personalizando el archivo **EDSD.segment**. Todos los segmentos de todos los mensajes que usen el compuesto C056 incluirán el elemento nuevo o le harán

© 2018 Altova GmbH

referencia.

- Mediante **inserción**, personalizando el archivo **ORDERS.Config**. El único segmento que incluirá el elemento nuevo será el segmento personalizado (CTA) del mensaje actual.

Personalización global

Para que el acceso al nuevo campo X1000 sea global, debemos modificar el archivo **EDSD.Segment**. Todos los segmentos de todos los mensajes que usen el compuesto C056 incluirán el elemento nuevo o le harán referencia.

Para redefinir el compuesto C056 globalmente:

1. Abra el archivo **EDSD.Segment** en un editor de texto o XML y navegue hasta **Config | Elements | Composite | C056**.

```
<Composite name="C056" info="CONTACT DETAILS">
  <Data ref="F3413" minOccurs="0"/>
  <Data ref="F3412" minOccurs="0"/>
</Composite>
```

2. Inserte esta línea en el segmento C056, debajo de "F3412":

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
```

Ahora la definición del compuesto debería tener este aspecto:

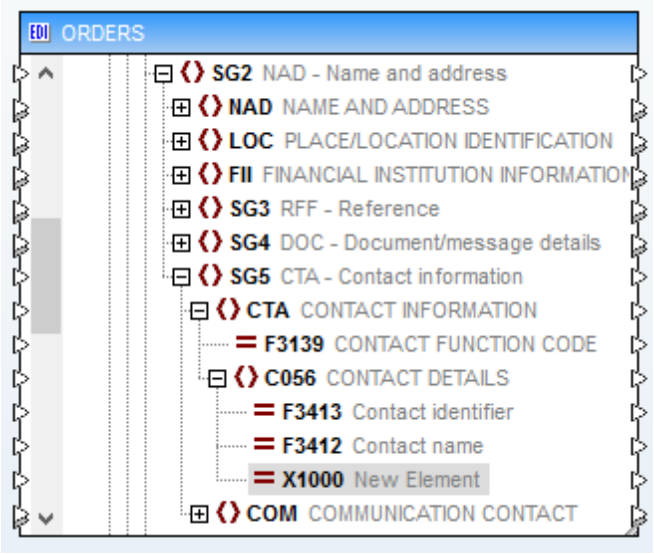
```
<Composite name="C056" info="CONTACT DETAILS">
  <Data ref="F3413" minOccurs="0"/>
  <Data ref="F3412" minOccurs="0"/>
  <Data name="X1000" type="string" maxLength="35" minOccurs="0"
info="New Element"/>
</Composite>
```

En el fragmento de código anterior el nuevo campo X1000 se define con ayuda del atributo "name" al contrario de otros campos del segmento, que utilizan el atributo "ref". Los otros dos campos se definen al principio del archivo y aquí solo se les hace referencia.

El nuevo campo X1000 ya está a disposición de todos los mensajes que usan el compuesto C056. Para obtener una vista previa del campo nuevo:

1. En el menú **Insertar** de MapForce haga clic en el comando **EDI** (o haga clic en el botón **Insertar EDI**  de la barra de herramientas).
2. Haga clic en la colección "EDIFACT.Nanonull" y seleccione el mensaje ORDERS.
3. Cuando la aplicación solicite un archivo EDI de origen, haga clic en **Omitir**. El componente ORDERS se puede ver en la ventana de asignación.
4. Haga clic en el título del componente, pulse **Ctrl+F** y escriba el término de búsqueda "X1000". Haga clic en **Buscar siguiente** para ir a la siguiente instancia del nuevo elemento X1000, que debería aparecer dentro de **Envelope/Interchange/Group/**

Message_ORDERS/SG2/SG5/CTA/C056).



Personalización insertada (local)

Para que el acceso al nuevo campo X1000 sea local (insertado), debemos modificar el archivo **ORDERS.config** solamente. En este caso el único segmento que incluirá el nuevo elemento o que le hará referencia será el segmento CTA redefinido del mensaje **actual**. Es decir, el segmento CTA se redefine de forma local para que contenga un compuesto C056 redefinido, con un campo nuevo personalizado llamado X1000.

Para redefinir el compuesto C056 de forma local:

1. Abra el archivo **ORDERS.Config** en un editor XML o de texto y navegue hasta `Group name="SG5"` (o busque **SG5**).

```
<Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact
information">
   <Segment ref="CTA"/>
   <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
</Group>
```

2. Reemplace la línea `<Segment ref="CTA"/>` por estas líneas:

```
<Segment name="CTA" id="CTA_ORDERS_SG5" info="CONTACT INFORMATION">
    <Data ref="F3139" minOccurs="0"/>
    <Composite name="C056" minOccurs="0" info="DEPARTMENT OR EMPLOYEE
DETAILS">
          <Data ref="F3413" minOccurs="0"/>
          <Data ref="F3412" minOccurs="0"/>
          <Data name="X1000" type="string" maxLength="35"
minOccurs="0" info="New Element"/>
    </Composite>
</Segment>
```

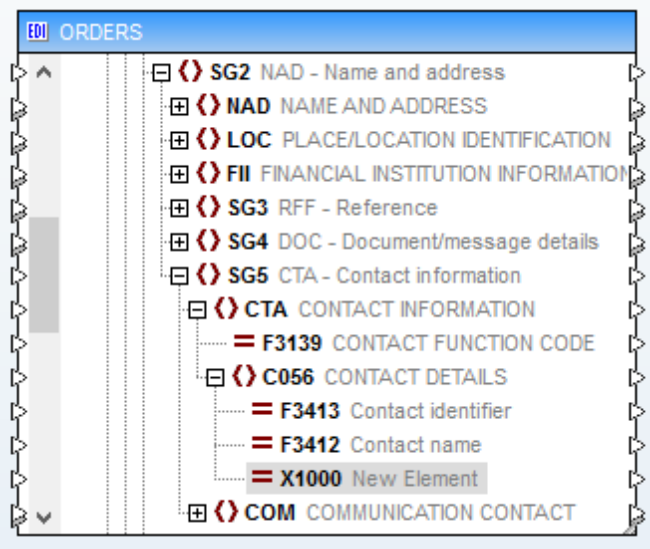

© 2018 Altova GmbH

Ahora la definición `Group` debería tener este aspecto:

```
<Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact
information">
   <Segment name="CTA" id="CTA_ORDERS_SG5" info="CONTACT INFORMATION">
    <Data ref="F3139" minOccurs="0"/>
    <Composite name="C056" minOccurs="0" info="DEPARTMENT OR EMPLOYEE
DETAILS">
           <Data ref="F3413" minOccurs="0"/>
           <Data ref="F3412" minOccurs="0"/>
           <Data name="X1000" type="string" maxLength="35"
minOccurs="0" info="New Element"/>
    </Composite>
   </Segment>
   <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
</Group>
```

Para obtener una vista previa del campo nuevo en MapForce:

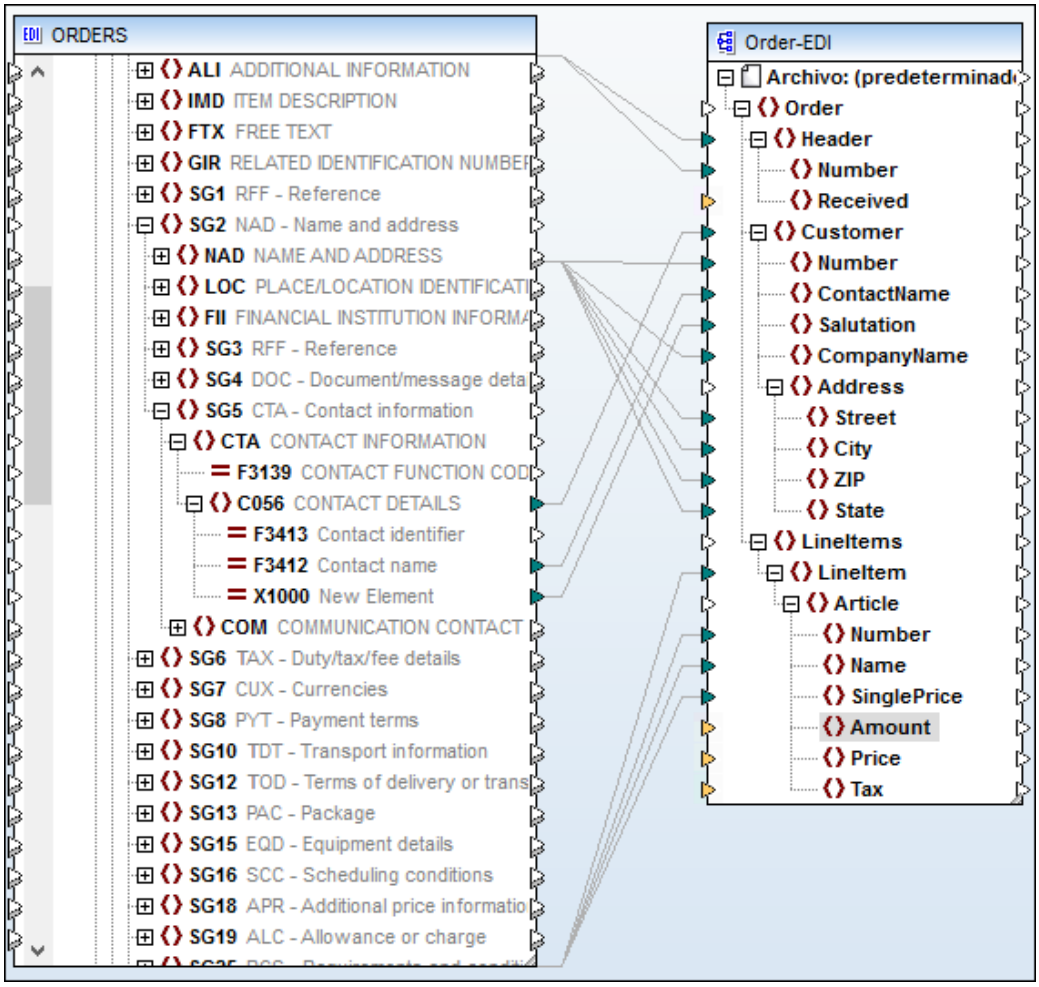
1. En el menú **Insertar** de MapForce haga clic en el comando **EDI** (o haga clic en el botón **Insertar EDI** de la barra de herramientas).
2. Haga clic en la colección "EDIFACT.Nanonull" y seleccione el mensaje ORDERS.
3. Cuando la aplicación solicite un archivo EDI de origen, haga clic en **Omitir**. El componente ORDERS se puede ver en la ventana de asignación.
4. Navegue hasta **Envelope/Interchange/Group/Message_ORDERS/SG2/SG5/CTA/C056** para ver el nuevo elemento X1000.



Usar el mensaje personalizado en asignaciones de datos

Si siguió las instrucciones que aparecen más arriba, ahora podrá usar la configuración EDIFACT personalizada en sus asignaciones de datos. Por ejemplo, el diseño de asignación de datos **Orders-Custom-EDI.mfd** (disponible en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**) utiliza la configuración EDIFACT personalizada. En este diseño

se asignan los datos del archivo **ORDERS-Custom.EDI** al esquema **Order-EDI**. El campo personalizado que se añadió a la estructura EDI (`X1000`) se asigna al elemento `Salutation` del esquema de destino.



Si siguió las instrucciones de este apartado, podrá abrir esta asignación de datos y ejecutarla correctamente. Si no siguió las instrucciones de personalización, deberá ejecutar la asignación de la siguiente manera:

1. Cree un directorio nuevo llamado "**EDIFACT.Nanonull**" dentro de "**C:\Archivos de programa\Altova\MapForce2018\MapForceEDI**". Si ejecuta una versión de 32 bits de MapForce en un sistema operativo de 64 bits, entonces será **C:\Archivos de programa (x86)\Altova\MapForce2018\MapForceEDI**.
2. Descomprima el archivo ZIP **EDIFACT.Nanonull.zip** (situado en el directorio **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**) dentro del directorio nuevo.
3. Abra el diseño de asignación **Orders-Custom-EDI.mfd**.

Para obtener una vista previa del resultado de la asignación abra el panel *Resultados*.

 © 2018 Altova GmbH



```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceS
3    <Header>
4      <Number>ABC123456XYZ</Number>
5    </Header>
6    <Customer>
7      <Number>92</Number>
8      <ContactName>Ted Little</ContactName>
9      <Salutation>Mr</Salutation>
10     <CompanyName>24h Pizza</CompanyName>
11     <Address>
12       <Street>Long Way</Street>
13       <City>San-Francisco</City>
14       <ZIP>34424</ZIP>
15       <State>CA</State>
16     </Address>
17   </Customer>
18   <Customer>
19     <Number>92</Number>
20     <ContactName>Michelle Butler</ContactName>
21     <Salutation>Mrs</Salutation>
22     <CompanyName>Nanonull, Inc.</CompanyName>
```

Al igual que los demás diseños de asignación de MapForce, las asignaciones de datos EDI se pueden ejecutar fuera de MapForce. Por ejemplo, puede implementarlas en otro servidor para ejecutarlas con MapForce Server o FlowForce Server o generar código de programa para ejecutar la asignación. En nuestro ejemplo, si se decide por ejecutar la asignación con código C++, tenga en cuenta que se generará una clase llamada "CX1000Type", a la que se puede tener acceso desde la clase "CC056Type".

7.5.14.12 Ejemplo: personalizar una transacción ANSI X12

Este ejemplo explica cómo personalizar MapForce para que pueda procesar formatos X12 no estándar o modificados. Para el ejemplo se usa la asignación **Orders-Custom-X12.mfd** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**. Recuerde que si abre la asignación antes de seguir las instrucciones que aparecen más abajo, MapForce considerará que el archivo no es válido. El motivo es que la asignación hace referencia a una colección X12 personalizada que no existe todavía en su equipo (se crea durante el ejemplo).

La asignación que se usa en este ejemplo lee datos del archivo **Orders-Custom.X12** está en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial** . Si abre este archivo con un editor de texto observe que usa un segmento N2 personalizado. Concretamente, la línea 6 contiene la entrada **`++Mrs`**:

```
 1    ISA+00+          +00+          +ZZ+SenderID       +ZZ+ReceiverID
 2    GS+PO+SenderID+ReceiverID+20060308+182347+1+UN+050112'
 3    ST+850+12345'
 4    BEG+00+SA+ABC123456XYZ++20040430'
 5    N1+1 +Nanonull, Inc.++123'
 6    N2+Michelle Butler++Mrs'
 7    N3+119 Oakstreet Suite 4876'
 8    N4+Vereno+CA+29213'
 9    PO1++1++7.2'
10    PO3+1 +++7.2++1+US'
11    PID+1++++Pizza Pepperoni'
12    TXI+1 ++9'
13    AMT+1+720'
14    PO1++2++13.2'
15    PO3+1 +++6.6++2+US'
```

Antes de poder asignar datos de archivo X12 personalizados como el anterior, es necesario crear una configuración X12 personal. En este ejemplo aprenderá a personalizar la configuración X12 para ampliar el segmento N2 con un elemento de datos nuevo que recibirá el título de la persona. El resultado final del procedimiento de personalización está disponible, a modo de ejemplo, en el directorio **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**: se trata del archivo ZIP **X12.Nanonull.zip**. Para poder seguir las instrucciones que aparecen a continuación se entiende que todavía no se desempaquetó el archivo ZIP en el directorio **..\Archivos de programa\MapForceEDI**.

Configurar los archivos de personalización

1. Abra el directorio de instalación de MapForce (**C:\Archivos de programa\Altova \MapForce2018**). Si ejecuta MapForce de 32 bits en un sistema operativo de 64 bits, utilice la ruta de acceso **C:\Archivos de programa (x86)\Altova\MapForce2018 \MapForceEDI**.
2. Ahora, dentro del directorio **..\MapForceEDI** debe crear un directorio llamado **X12.Nanonull**.
3. Copie estos archivos de la carpeta ...**MapForceEDI\EDIFACT** dentro de la carpeta **X12.Nanonull**.

   - **EDI.Collection**
   - **Envelope.Config**
   - **850.Config**
   - **X12.Segment**
   - **X12.Codelist**

   Para saber qué finalidad tiene cada archivo consulte el apartado Archivos de configuración EDI.

Nota: cambie los atributos de los archivos por **read-write** para poder editarlos.

Configurar el archivo EDI.Collection

1. Abra el archivo **EDI.Collection** en un editor XML como XMLSpy.
2. Elimine todos los elementos "Message", excepto el mensaje "Purchase Order", pero sin eliminar las etiquetas <Messages>.

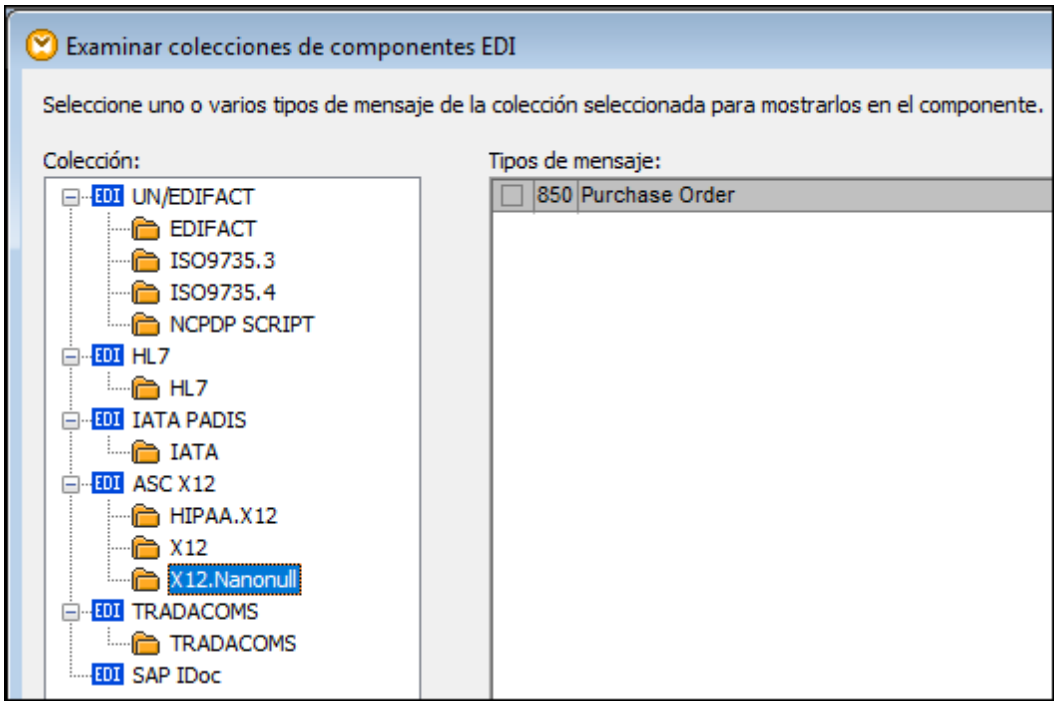
© 2018 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<Messages Version="3">
   <Meta>
      <Release>6040</Release>
      <Agency>X12</Agency>
   </Meta>
   <Root File="Envelope.Config"/>
   <Message Type="850" File="850.Config" Description="Purchase Order"/>
</Messages>
```

3. Guarde el archivo.

El nuevo archivo EDI.Collection ya se puede abrir en MapForce:

1. Inicie MapForce.
2. En el menú **Insertar** haga clic en el comando **EDI** o haga clic en el botón **Insertar EDI** de la barra de herramientas. Se abre el cuadro de diálogo "Examinar colecciones de componentes EDI". En el panel de colecciones, situado a la izquierda, aparece una carpeta nueva llamada `X12.Nanonull`. Si hace clic en esta carpeta, observará que solamente incluye una entrada, el tipo de mensaje "Purchase Order".



Nota: cuando se inicia, MapForce busca el archivo `EDI.Collection` en todas las subcarpetas del directorio **...\MapforceEDI**. El cuadro de lista muestra el contenido actual del archivo de colecciones, que en nuestro ejemplo solamente tiene el mensaje `Purchase Order`.

Diferencias entre personalización global e insertada (local)

En este ejemplo el objetivo es redefinir el segmento N2 "Additional Name Information" para que pueda contener un elemento de datos X1000 personalizado, que se encargará de almacenar el título de una persona (p. ej. `Mrs`). Antes de la personalización, N2 está compuesto por un solo elemento de datos (F93 Name). Hay varias maneras de realizar esta operación de personalización:

- De forma **global**, personalizando el archivo **X12.Segment**. Todos los segmentos de todas las transacciones que usen N2 incluirán el elemento nuevo o le harán referencia.
- Mediante **inserción**, personalizando el archivo **850.Config**. El único segmento que incluirá el elemento nuevo será el segmento personalizado (N2) de la transacción actual.

Personalización global

Para que el acceso al nuevo campo X1000 sea global, debemos modificar el archivo **X12.Segment**. Todos los segmentos de todas las transacciones que usen N2 incluirán el elemento nuevo o le harán referencia.

Para redefinir el segmento N2 globalmente:

1. Abra el archivo **X12.Segment** en un editor de texto o XML y navegue hasta **Config | Elements | Segment name="N2"**.

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
</Segment>
```

2. Inserte esta línea debajo de "F93" y guarde el archivo.

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
Element"/>
```

   Ahora la definición del segmento debería tener este aspecto:

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
   <Data name="X1000" type="string" maxLength="35" minOccurs="0"
info="New Element"/>
</Segment>
```

Notas:

- El nuevo campo X1000 se define con el atributo `name`, al contrario de los demás campos del segmento, que usan el atributo `ref`. El campo F93 se define al principio del archivo **X12.Segment** y aquí solo se le hace referencia.
- El atributo `mergedEntries` de F93 especifica que puede haber dos instancias de este elemento con el mismo nombre. Esto refleja la definición de este campo tal y como se define en la especificación ASC X12. Por este motivo, en el archivo EDI de origen que estamos personalizando el segmento N1 tiene este aspecto:

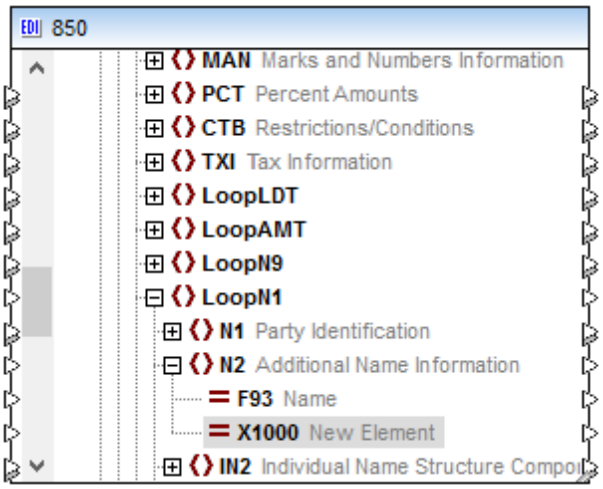
© 2018 Altova GmbH

```
N2+Michelle Butler++Mrs'
```

En la línea anterior puede ver que hay dos separadores + consecutivos entre "Michelle Butler" y "Mrs". Esto se debe a que el segmento N2 está compuesto por un elemento de datos "F93" ("Michelle Butler"), seguido de un separador "+", seguido de otro elemento "F93" (que está vacío), seguido de un separador "+", seguido del elemento de datos personal "X1000". Para más información sobre el atributo `mergedEntries` consulte el apartado Varios elementos de datos consecutivos.

El nuevo campo X1000 ya está a disposición de todos los mensajes que usan el segmento N2. Para obtener una vista previa del campo nuevo:

1. En el menú **Insertar** de MapForce haga clic en el comando **EDI** (o haga clic en el botón **Insertar EDI** de la barra de herramientas).
2. Haga clic en la colección "X12.Nanonull" y seleccione el mensaje "Purchase Order".
3. Cuando la aplicación solicite un archivo EDI de origen, haga clic en **Omitir**. El componente 850 se puede ver en la ventana de asignación.
4. Haga clic en el título del componente, pulse **Ctrl+F** y escriba el término de búsqueda "X1000". Haga clic en **Buscar siguiente** para ir a la siguiente instancia del nuevo elemento X1000, que debería aparecer dentro de **Envelope/Interchange/Group/Message_850/Loop_N1/N2**).



Personalización insertada (local)

Para que el acceso al nuevo campo X1000 sea local (insertado), debemos modificar el archivo **850.config** solamente. En este caso el único segmento que contendrá el nuevo campo X1000 o que le hará referencia será el segmento N2 de la transacción "850 - Purchase Order".

Para redefinir el segmento N2 de forma local:

1. Abra el archivo **850.Config** en un editor XML o de texto y navegue hasta `Group name="LoopN1"` (o busque **LoopN1**).

```
<Group name="LoopN1" maxOccurs="10" minOccurs="0">
```

```
   <Segment ref="N1"/>
   <Segment ref="N2" minOccurs="0" maxOccurs="2"/>

   ... ...

</Group>
```

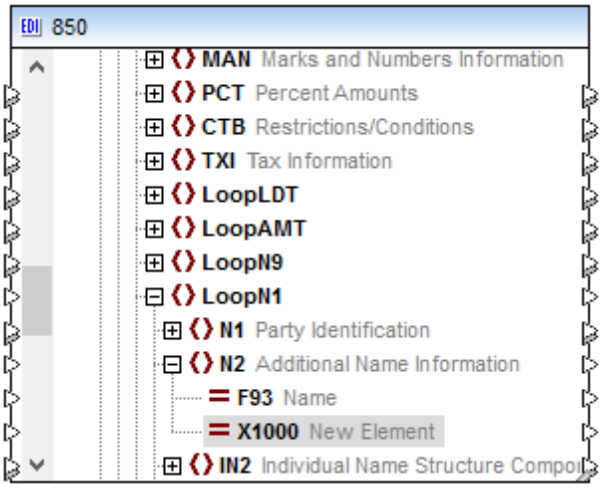
2. Reemplace la línea `<Segment ref="N2" minOccurs="0" maxOccurs="2"/>` con estas líneas:

```
<Segment name="N2" info="Additional Name Information">
   <Data ref="F93" mergedEntries="2"/>
   <Data name="X1000" type="string" maxLength="35" minOccurs="0"
info="New Element"/>
</Segment>
```

Ahora la definición `Group` debería tener este aspecto:

```
<Group name="LoopN1" maxOccurs="10" minOccurs="0">
   <Segment ref="N1"/>
   <Segment name="N2" info="Additional Name Information">
      <Data ref="F93" mergedEntries="2"/>
      <Data name="X1000" type="string" maxLength="35" minOccurs="0"
info="New Element"/>
   </Segment>

   ... ...

</Group>
```

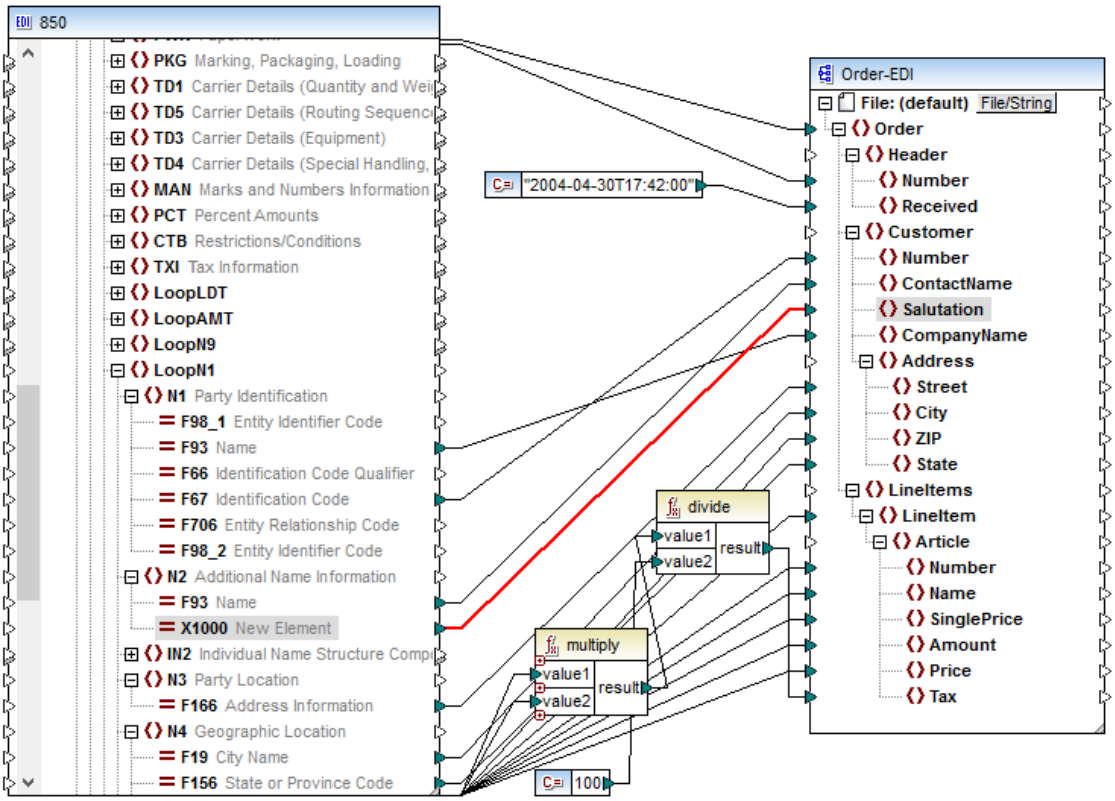
Ya puede consultar la vista previa del nuevo campo en MapForce (ver instrucciones en el apartado sobre personalización global).



© 2018 Altova GmbH

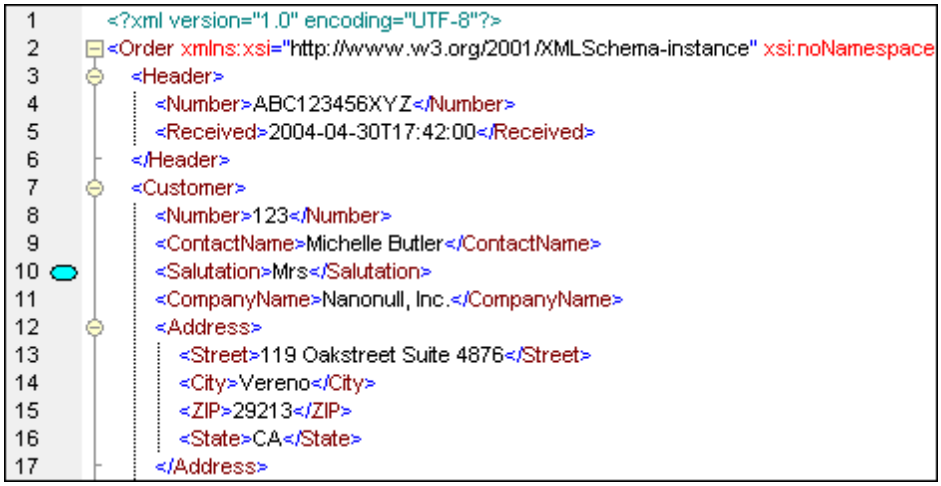
Usar el mensaje personalizado en asignaciones de datos

Si siguió las instrucciones que aparecen más arriba, ahora podrá usar la configuración X12 personalizada en sus asignaciones de datos. Por ejemplo, el diseño de asignación de datos **Orders-Custom-X12.mfd** (disponible en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**) utiliza la configuración X12 personalizada. En este diseño se asignan los datos del archivo **ORDERS-Custom.X12** al esquema **Order-EDI**. El campo personalizado que se añadió (`X1000`) se asigna al elemento `Salutation` del esquema de destino.



Orders-Custom-X12.mfd

Si siguió las instrucciones de este apartado, podrá abrir esta asignación de datos y ejecutarla correctamente. Para obtener una vista previa del resultado de la asignación abra el panel *Resultados*. Tal y como esperábamos, el elemento de destino "Salutation" se rellenó con el valor "Mrs" extraído del elemento de datos personal X1000 del archivo X12.



7.5.15 Personalizar la validación de datos EDI

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

This section illustrates how validation of EDI messages can be customized in MapForce so as to achieve the following goals:

- Relax validation of fields where validation rules are not completely enforced by the standard and may be extended by means of external standards.

  For example, there are EDI data elements that may accept code values specified in external ISO standards or UN/ECE recommendations, in addition to predefined code values. In this case, when processing such EDI fields, you can configure MapForce to return a warning (not a validation error) when an external code is encountered, and resume the mapping process. For further information, see Relaxing Field Validation.

- Perform full validation of fields such as the ones described above, by adapting custom configuration files to MapForce.

  For example, to validate the Data Element 4405 of the UN/EDIFACT D.16A standard, you may add a custom configuration file which contains the validation rules as defined by the UN/ECE Recommendation 24 ("Trade and Transport Status Codes". The code values defined in UN/ECE Recommendation 24 are external to EDIFACT standard (and thus are not part of the default set of EDIFACT configuration files supplied with MapForce). For further information, see Adapting Code Values from External Sources.

- Perform validation globally for all messages where a field is used or locally for a particular

 © 2018 Altova GmbH

message.

It is possible to configure validation to apply globally or locally. Global validation affects all occurrences of the field in all messages. Local (or "inline") validation applies only to a specific message. For further information, see Defining Custom Code Values at Message Level (Inline).

- Define advanced validation rules for fields that require validation not only at field level, but also at character level.

  Certain messages, such as the VERMAS (Verification of Mass) message of UN/EDIFACT D.16A, contain fields that require advanced validation rules (such as Data Element 8155). This field may accept code values specified in ISO 6346. To validate field such as ISO 6346, MapForce can be customized, by means of configuration files, to check if specific characters occur at specific positions within the field. For further information, see Configuring Character-Level Validation. (Note that ISO 6346 is mentioned here only as an example, the MapForce configuration files already include validation rules for ISO 6346.)

7.5.15.1 Relajar la validación de campos

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

There are cases when a standard does not completely enforce the code values that a field may have, allowing code values defined by external standards or recommendations. Taking EDIFACT D16A as example, several fields allow code values defined in external ISO standards, or in UN/ECE recommendations. For example, Data Element 4405 defines 146 code values, but also allows use of external UN/ECE Recommendation 24, which specifies 346 code values.

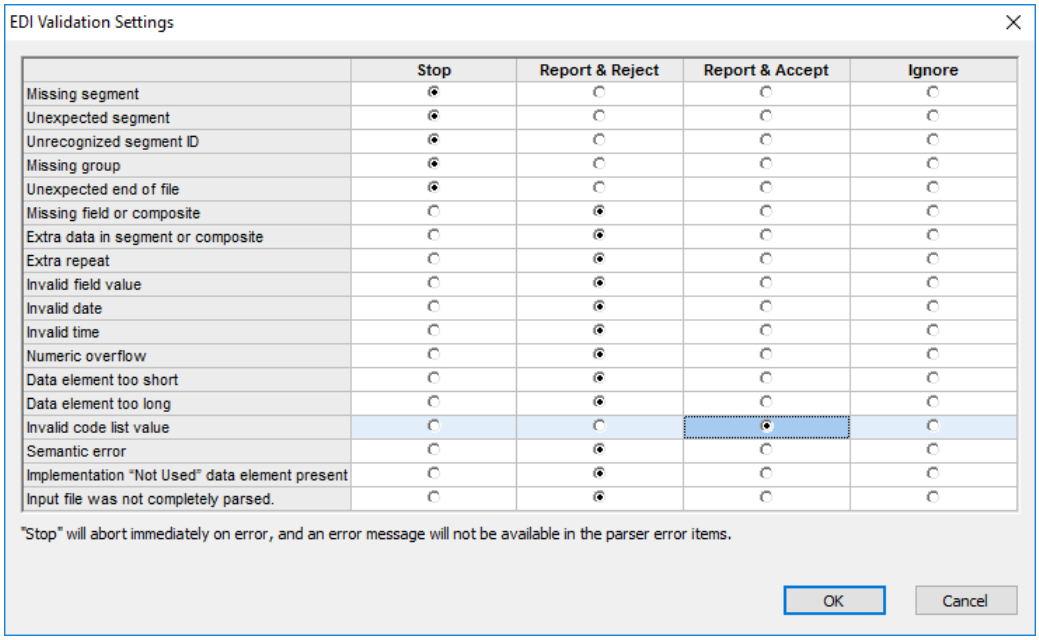
In MapForce, the possible code values for a field are validated against a configuration file having the .codelist extension (hereafter referred to as the ".codelist" file). When MapForce processes an EDI file, it normally produces a validation error if a field does not have any of the code values specified in the .codelist file for that EDI flavour. (This assumes you have not changed the default validation options so as to silently ignore invalid field values, see EDI component validation.) The .codelist files available for each EDI flavour are in the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI** directory. For example, the **UNCL.codelist** file available in **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI\EDIFACT** stores possible code values for fields in UN/EDIFACT fields.

The default .codelist file includes all possible code values for a field, according to the respective EDI standard. When code values are not explicitly defined by the standard but enforced by external standards, validation is considered to be incomplete. Namely, when the EDI standard does not fully enforce the list of code values for a particular field (such as Data Element 4405 of UN/EDIFACT D16A), processing the field during the mapping execution would normally raise validation errors. Specifically, MapForce (or the execution engine running the EDI mapping, be it MapForce Server, or a C#, C++, or Java program) reports an error in the format: **{Value} is not a**

legal value for field {Field}. This error means that the field value was not found in the list of existing code list values defined in the .codelist file.

To prevent the mapping process from being stopped when errors such as above are encountered, you can do the following:

1. Change the validation behavior from EDI Validation Settings (for example, select the check box **Report & Accept** next to "Invalid code list value" on the EDI Validation Settings dialog box. (To configure EDI validation settings, click the **Validation** button on the EDI Component Settings dialog box.)



2. Relax field validation from configuration files, as shown below.

MapForce provides a mechanism by which a field can be marked as having "incomplete" validation rules. This ensures that MapForce (or the execution engine running the EDI mapping, be it MapForce Server, or a C#, C++, or Java program) does not produce an error message when the field is processed; instead, a warning is displayed, and the mapping process resumes. In other words, "incomplete" validation means that only the code values defined in the MapForce default .codelist file should be considered for validation, not those defined externally.

To allow for flexibility without adding too much complexity, the default .codelist configuration files supplied with MapForce treat certain fields as having incomplete validation.

By default, MapForce produces warnings (not errors) if any of the following fields contain code values that are defined in external specifications:

- Data Element 4053 of UN/EDIFACT D.16A
- Data Element 4277 of UN/EDIFACT D.16A
- Data Element 4405 of UN/EDIFACT D.16A
- Data Element 8179 of UN/EDIFACT D.16A

© 2018 Altova GmbH

The warning is produced so as to make you aware that validation of this field is incomplete. Incomplete validation may result in incorrect data being produced by the mapping. To prevent this from happening, it is recommended to perform full field validation, by means of custom .codelist files to MapForce. Custom .codelist files let you define external code values in a format understood by MapForce and perform full validation of the field. For more information, see Adapting Code Values from External Sources.

Note: Data Element 8155 also relies for validation on an external standard (the ISO 6346 standard); however, for this field MapForce provides full validation by means of a separate .codelist file, see Adapting Code Values from External Sources).

If necessary, you can additionally customize other fields to be treated as having incomplete validation rules. To instruct MapForce that a field has incomplete validation rules, do the following:

1. Locate the .codelist file in the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI** directory (subdirectories exist for each EDI flavour), and open it with an XML editor such as XMLSpy.
2. Find the `<Field>` element corresponding to the required data element, and add the attribute `Incomplete="1"` to it (highlighted below), for example:

```
<Field Name="F8179" Length="8" Type="Alphanumeric" Incomplete="1">
   <Title>Transport means description code</Title>
   <Description>Code specifying the means of transport.</Description>
   <Note>1 Use UN/ECE recommendation 28.
</Note>
   <Values>
      <Value Code="1">
         <Title>Box tricycle</Title>
         <Description>Tricycle with a basket for delivering goods.</
Description>
      </Value>
   </Values>
</Field>
```

Note: You must have write permissions to the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI** directory in order to modify its contents.

If your processing requirements demand that an error should be raised instead of a warning for any fields that were already marked as "incomplete", change the attribute `Incomplete` to value "0" instead of "1". This way, field validation will be enforced strictly, and errors (not warnings) will occur if the field value is not part of the set of values defined in the .codelist file.

7.5.15.2 Adaptar valores de código de fuentes externas

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

In the EDI configuration files supplied by default with MapForce, some data elements are configured to have incomplete validation, as explained in Relaxing Field Validation. This is due to the fact that the possible code values for these data elements rely on specifications which are external to the EDI standard itself.

When incomplete (or relaxed) validation of a field is not acceptable or desirable, it is possible to configure MapForce to perform full field validation. This approach requires adding custom .codelist files in addition to the default ones already available in MapForce. The role of the custom .codelist files is to supply any code values that are not defined by default in the EDI standard and consequently are not present in MapForce configuration files. This usually applies to code values that are defined in external standards or recommendations.

Note that the custom .codelist file will require an entry for every possible code value that is to be validated (which may be hundreds of records). An example is the Data Element 4405 of the UN/EDIFACT D16A. In addition to the code values defined in the UN/EDIFACT standard, this field also allows code values that are specified in the UN/ECE Recommendation 24 ("Trade and Transport Status Codes", see http://www.unece.org/fileadmin/DAM/cefact/recommendations/rec24/rec24_ecetrd258e.pdf) which amount to 346 codes. In order to accommodate all these external code values, a custom .codelist file must be created which contains an entry for each of the 346 codes present in the recommendation. Therefore, if that is acceptable by your particular business case, using incomplete validation is a simpler solution (see Relaxing Field Validation). Be aware that incomplete validation involves the risk of allowing invalid data to be produced in the mapping output, so full field validation is recommended instead.

To perform full field validation of fields whose code values are defined in an external standard, take the following steps:

1. Create a copy of the default .codelist file supplied with MapForce. The .codelist files available for each EDI flavour are in the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI** directory. For example, if adding a custom .codelist file to validate UN/EDIFACT messages, use as template the **UNCL.Codelist** file available in **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI\EDIFACT**.

Note: You must have write permissions to the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI** directory in order to modify its contents.

2. Rename the newly created .codelist file to a short and descriptive filename, since the file will be referred from other MapForce configuration files. For the scope of this example, the custom .codelist file will be called **UNECE24.codelist**.
3. Using an XML editor such as XMLSpy, delete from the new file all content included between the `<Codelist>` and `</Codelist>` tags. The file should now look as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="5" xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
```


© 2018 Altova GmbH

```
...

   </Codelist>
</Config>
```

4. Add a new <Field> structure under <Codelist>, as shown below. In this example, we assume to be adding external code values for Data Element 4405 of the UN/EDIFACT. Otherwise, replace the field attributes as required by your particular case. Remember that you can find the correct name, length and type of each field in the default .codelist file supplied by MapForce (in this example, **UNCL.Codelist**).

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="5" xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
      <Field Name="F4405" Length="3" Type="Alphanumeric">
         <Title>Trade and transport status codes</Title>
         <Description>Code specifying a status.</Description>
         <Values>

...

         <Values>
      </Field>
   </Codelist>
</Config>
```

5. Under <Values>, add a new <Value> for each external code, as shown below. In this example, we are adding the first two codes defined in the UN/ECE Recommendation 24 ("Trade and Transport Status Codes", see http://www.unece.org/fileadmin/DAM/cefact/recommendations/rec24/rec24_ecetrd258e.pdf).

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="5" xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
      <Field Name="F4405" Length="3" Type="Alphanumeric">
         <Title>Trade and transport status codes</Title>
         <Description>Code specifying a status.</Description>
         <Values>
            <Value Code="1">
               <Title>Arrival, completed</Title>
               <Description>The goods/consignment/equipment/means of
transport has arrived.</Description>
            </Value>
            <Value Code="2">
               <Title>Loading, authorized</Title>
               <Description>Authorization to load has been given.</
Description>
            </Value>
```

```
        <Values>
      </Field>
   </Codelist>
</Config>
```

At this stage, the custom .codelist file is ready but not yet bound to MapForce. Just be aware that, for the scope of the example, we have only added two external codes ("1" and "2"); however, in a real-life scenario the custom .codelist file is expected to include all code values specified by the external standard.

Finally, before the custom .codelist file can be recognized by MapForce, the following must also be done:

1. Since now the Data Element 4405 does no longer have incomplete validation, MapForce must be instructed to apply full validation. Namely, the `Incomplete="1"` attribute must be modified in the **UNCL.codelist** file for this field (the attribute must either be removed, or set to `Incomplete="0"`). Therefore, modify the **UNCL.codelist** file as shown below:

```
...

<Field Name="F4405" Length="3" Type="Alphanumeric" Incomplete="0">
        <Title>Status description code</Title>
        <Description>Code specifying a status.</Description>
        <Note>1 For transport status, use UN/ECE Recommendation 24.</
Note>

...
```

2. Modify the **UNCL.codelist** file to add a reference to the newly created **UNECE24.codelist** file, as shown below:

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Version="5" xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Include href="UNECE24.codelist"/>
   <Codelist>

....
```

You have now finished adapting MapForce configuration files to validate Field 4405 from code values defined in a custom file. This approach can also be used for other EDI flavours to achieve full field validation based on some external standard.

Although it is possible to define external code lists by editing the existing .codelist file supplied with MapForce (in this example, **UNCL.codelist**), this is not recommended. A good practice is to always define external code lists in custom files and keep changes to the default MapForce configuration files to minimum; this will make it easier to migrate to new versions of EDI formats in future.

© 2018 Altova GmbH

Note that the instructions above cause the field validation to be global, for any segment that may be using Field 4405. In some cases, you may want the custom code lists to apply only to certain messages (in other words, the configuration should not apply globally for all messages, but inline, or locally, for a particular message only). For more information about using the "inline" approach, see Defining Code Values at Message Level (Inline).

7.5.15.3 Definir valores de código a nivel de mensaje (inline)

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

In a previous topic, Adapting Code Values from External Sources, you have seen how to configure field validation through custom code lists *globally* (it applies for any message that may be using a particular field). In the event that you need validation to apply only to a specific message, it is possible to add custom code lists inline (locally for a particular message). When you need to do this, follow the steps below:

1. Open in an XML editor the **{Message}.config** file corresponding to the message of interest. The **{Message}.config** file can be found in the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI** directory, in the subdirectory corresponding to the name of the required EDI flavour.
2. Find the `<Data>` element under the required `<Segment>`, and add the custom code values as children of `<Values>`, for example:

```
<Group name="Message_270-B1" info="Health Care Eligibility Benefit
Inquiry" maxOccurs="unbounded">
      <Segment name="ST" info="Transaction Set Header">
        <Condition path="F1705" value="005010X279A1" />
        <Data ref="F143" info="Transaction Set Identifier Code">
          <Values>
            <Value Code="270" />

[>>>> custom values can be added here...]

          </Values>
        </Data>
        <Data ref="F329" info="Transaction Set Control Number" />
        <Data ref="F1705" info="Implementation Convention Reference" />
      </Segment>
```

Note the following:

- You must have write permissions to the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI** directory in order to modify its contents.
- When a field has both locally-defined values (such as those in the code listing above) and

globally-defined code values (in a .codelist file, as shown in Adapting Code Values from External Sources), then make sure that local code values are also defined as global code values. Otherwise, a validation error*** will be raised: **{Value} is not a legal value for field {Field}**.

- Inline (local) validation can be used if you require validation at field level (that is, the possible values that a field can take can be expressed as a list). Inline validation cannot be used for those fields that require complex validation rules at character level. When the field needs complex validation rules at character level (such as those defined in ISO 6346 applicable to Data Element 8155 of the UN/EDIFACT standard), add the code values globally, as shown in Adapting Code Values from External Sources. You can then configure validation as shown in Configuring Character-Level Validation.

*** Note that the actual validation outcome takes into account the settings configured for the rule "Invalid code list value" in the EDI Validation Settings dialog box. (To open this dialog box, click the **Validation** button on the EDI Component Settings dialog box).

7.5.15.4 Configurar validación a nivel de caracteres

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

In most cases, validation of EDI fields means comparing, at mapping runtime, the field value with a list of possible values defined in a MapForce configuration file. For example, if the field value matches one the possible values defined in the configuration file, it is considered valid.

However, certain fields might need more advanced validation rules, meaning that their validation logic cannot be expressed as a simple list of possible values. An example of a field which requires complex validation is Data Element 8155 used in the VERMAS (Verification of Mass) message of UN/EDIFACT D16A. This field accepts not only code values defined in the UN/EDIFACT standard, but also external codes specified by the ISO 6346 standard. The ISO 6346 standard, in its turn, defines validation rules at character-level, namely:

- First character can be one of these: 1, 2, 3, 4, A, B, C, D, E, F, G, H, K, L, M, N, P
- Second character can be one of these: 0, 2, 4, 5, 6, 8, 9, C, D, E, F, L, M, N, P
- Third and fourth character can be: G0, G1, G2, G3, etc. (values are too many to be listed here).

It is possible to adapt validation logic such as the one above to MapForce as well. In this case, MapForce will validate Data Element 8155 not only according to the UN/EDIFACT code values, but also taking into consideration the rules above applicable to the ISO 6346 standard.

From a MapForce configuration perspective, the rules expressed by the three bullets above can be regarded as three separate lists of values. The first list is one character long, the second list is one character long, and the third list is two characters long. All three lists are finite and their values are known. This information makes it possible to validate the value of the data element by means of MapForce configuration.

© 2018 Altova GmbH

To see an example of how character-level validation must be configured, open the **ISO6346.Codelist** file from the **C:\Archivos de programa\Altova\MapForce2018\MapForceEDI\EDIFACT** directory. This file supplied by default with MapForce is configured to perform validation of Data Element 8155 according to ISO 6346 rules stated above.

```
<?xml version="1.0" encoding="UTF-8"?>
<Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="5"
xsi:noNamespaceSchemaLocation="../EDIConfig5.xsd">
   <Codelist>
      <Field Name="F8155" Type="Alphanumeric" ListName="ISO6346" Length="1"
Offset="1">
         <Title>ISO 6346</Title>
         <Description>Code specifying seagoing container type.</Description>
         <Values>
            <Value Code="1"/>

            ..............

            <Value Code="N"/>
         </Values>
      </Field>
      <Field Name="F8155" Type="Alphanumeric" ListName="ISO6346" Length="1"
Offset="2">
         <Title>ISO 6346</Title>
         <Description>Code specifying seagoing container type.</Description>
         <Values>
            <Value Code="0"/>

            ..............

            <Value Code="F"/>
         </Values>
      </Field>
      <Field Name="F8155" Type="Alphanumeric" ListName="ISO6346" Length="2"
Offset="3">
         <Title>ISO 6346</Title>
         <Description>Code specifying seagoing container type.</Description>
         <Values>
            <Value Code="G0"/>

            ..............

            <Value Code="A0"/>
         </Values>
      </Field>
   </Codelist>
</Config>
```

ISO6346.Codelist configuration file

Note: Some of the values from the sample code listing above were removed, for simplicity.

In the code listing above, notice that, the `<Field>` element is defined three times. This makes it

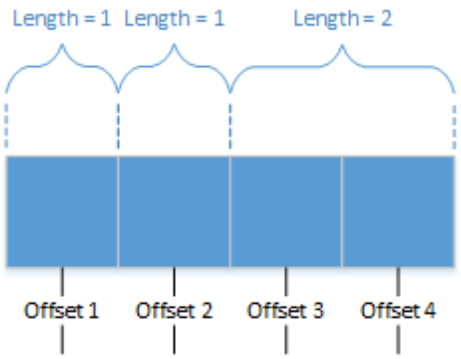
possible to validate the field content at character level. Namely, each occurrence of <Field> validates one of the rules stated in the bullets above at a time. To achieve this, each <Field> has several attributes (highlighted in the code listing above):

- Name - This attribute indicates that the list of values being validated belongs to the same field. In other words, it binds all three lists to the same field (in this case, "F8155").
- ListName - This attribute gives a name to a list of values. Lists of values with the same name are grouped into one code list. Notice that in the example above all three lists have the same ListName="ISO6346". This is needed specifically for character level validation. The actual characters to be validated are specified by the Length and Offset attributes of each list in the group, as explained below.
- Length - This attribute specifies the number of character to be validated (the starting position of characters in the field is defined by the Offset attribute). Notice that character length varies for each <Field>: the first and the second list of values is one character long, while the third list of values contains two characters.
- Offset - This attribute specifies the position at which MapForce (or the external execution engine) must start reading the characters from the data field which is being validated. The first character has offset 1, the second character has offset 2, and so on. The actual number of characters is defined by the Length attribute.

If the Length attribute is defined, the Offset attribute must be defined as well. Also, the value of both Length and Offset must be greater than zero.

In the code listing above, the following Length-Offset combinations were defined:

- Length="1", Offset="1" - this combination will validate one character of the field, starting at offset 1
- Length="1", Offset="2" - this combination will validate one character of the field, starting at offset 2
- Length="2", Offset="3" - this combination will validate two characters of the field, starting at offset 3



Each Length-Offset combination, combined with the code values defined under <Values>, represents a validation condition at character level. When the mapping runs, all character-level conditions must be satisfied (because all have the same ListName attribute value) for field validation to be successful. In other words, when validating a message that includes Data Element 8155, MapForce will apply the following logic:

 © 2018 Altova GmbH

- The field characters are validated against the list of code values defined in the configuration file (in this case, **ISO6346.Codelist**) for a matching Length-Offset combination. For example, the first character of the field must be one of the following: 1, 2, 3, 4, B, C, G, H, L, M, N.
- The field value must satisfy all conditions at character-level (in this case, three conditions must be satisfied) in order for validation to be successful.
- If any of the conditions is not satisfied, the field value is not valid, and a validation error is produced in MapForce (or in the execution engine running the EDI mapping (such as MapForce Server, or a C#, C++, or Java program).
- A field may have more than one list of possible code values. In ordered to be considered valid, the field value must be present in at least one of the lists.

7.6 JSON

Sitio web de Altova: Asignaciones JSON

Puede usar archivos de instancia y de esquema JSON (notación de objetos JavaScript) como componentes de origen y destino de la asignación de datos. MapForce lee y escribe archivos JSON usando el esquema JSON Draft 04 Schema (http://tools.ietf.org/html/draft-zyp-json-schema-04).

Cuando se añade un archivo JSON a una asignación de datos, lo primero que hace MapForce es comprobar si el archivo tiene la propiedad `$schema`. Esto sirve para determinar si se trata de un esquema o de un archivo de instancia. Si el archivo incluye esta propiedad y la propiedad es "http://json-schema.org/schema#" o "http://json-schema.org/draft-04/schema#", entonces MapForce carga el archivo como esquema. Sin embargo, si falta esta propiedad, MapForce pregunta al usuario si el archivo es un esquema o una instancia. Si el usuario confirma que se trata de un archivo de instancia, MapForce le ofrecerá dos opciones: suministrar un esquema para la instancia o generar uno automáticamente. Este esquema JSON se utiliza para generar la estructura del componente de asignación de datos.

MapForce reconoce archivos de esquema JSON con la extensión **schema.json** (p. ej. **Example.schema.json**). En el momento de publicación de este manual del usuario, no existe convención formal alguna para la nomenclatura de archivos de esquema JSON.

En MapForce los componentes JSON suelen tener la estructura de un archivo de instancia JSON. En otras palabras, la estructura de los nodos del componente JSON refleja la estructura del archivo de instancia JSON. En la tabla que aparece a continuación puede ver la representación convencional de los tipos JSON básicos en los componentes JSON de MapForce.

Tipo	Representación en MapForce
Matriz	[]
Booleano	01
Nulo	∅
Número o entero	#
Objeto	{}
Cadena de texto	"AB"

También hay algunos casos especiales:

Caso	Descripción
additionalProperties, patternProperties	El nodo aparece en el componente JSON dentro de todos los objetos que tengan una propiedad `additionalProperties` con valor `true` o que no estén presentes en el esquema. Permite crear asignaciones de datos con propiedades que no están definidas explícitamente en el esquema (*véase Ejemplo: asignación de datos*

© 2018 Altova GmbH

Caso	Descripción
	entre JSON y CSV). Este nodo también puede aparecer en objetos que tengan la propiedad `patternProperties`.
Subtipos	La especificación JSON schema permite definir subtipos para objetos y matrices (`anyOf`, `allOf`, `oneOf`). MapForce muestra estos subtipos por medio de nodos de estructura especiales ( ) que no tienen una representación directa en el archivo de instancia JSON. Si la asignación lee datos de un archivo JSON, estos nodos subtipos aportan un valor si el valor de entrada actual es válido según el esquema del subtipo. Si la asignación escribe datos en un archivo JSON, asegúrese de seleccionar el subtipo correcto. Si se escriben varios subtipos, pueden duplicarse propiedades de objeto, lo cual daría lugar a archivos JSON de salida no válidos.
Varios tipos en la misma ubicación	El esquema JSON permite que haya varios tipos en la misma ubicación. En casos así el componente de MapForce muestra nodos distintos por cada tipo básico que pueda aparecer en dicha ubicación.
Nombres de los tipos	En MapForce las propiedades `title` y `description` de los tipos en el esquema JSON se muestran en los campos `tipo` y `anotación`. Si falta la propiedad `title`, entonces MapForce puede utilizar parte del URI de la propiedad `$ref` como nombre del tipo.
Matrices con varios tipos de elemento	Si una matriz tiene tipos de elementos diferentes en el esquema JSON (p. ej. tanto cadenas como números), MapForce muestra un nodo `item` por cada tipo de elemento. Cuando el destino de la asignación es un archivo JSON, podrá crear matrices que contengan varios tipos diferentes.
Matrices definidas como tuplas	Si una matriz tiene elementos cuyos tipos se asignan dependiendo de la posición que tengan en el esquema JSON, MapForce muestra el índice basado en cero del elemento como diferentes nodos (p. ej. **item[0]**, **item[1]** y así sucesivamente). Cuando el destino de la asignación es un archivo JSON, podrá especificar el tipo de cada elemento de la matriz por medio de su índice basado en cero.

Nota sobre el uso de archivos JSON

Para usar componentes JSON en MapForce es necesario elegir el motor de transformación integrado BUILTIN (véase Seleccionar un lenguaje de transformación). Los componentes JSON no son compatibles con ningún otro lenguaje de generación de código. Además, los componentes JSON no admiten el uso de:

- parámetros complejos en funciones definidas por el usuario ni
- componentes de variable intermedios.

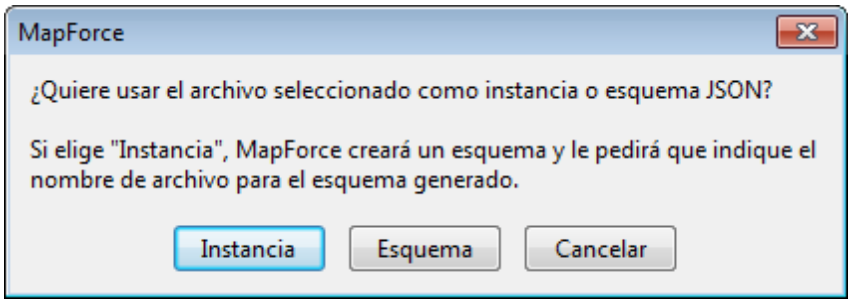
Temas relacionados:

- Agregar archivos JSON como componentes de asignación de datos
- Configuración de componentes JSON
- Ejemplo: asignación de datos entre JSON y CSV

7.6.1 Agregar archivos JSON como componentes de asignación

Para agregar archivos JSON a la asignación de datos:

1. Seleccione el comando **Insertar | Archivo o esquema JSON** o haga clic en el icono correspondiente de la barra de herramientas. Aparece un aviso donde debe seleccionar el tipo de archivo JSON que desea insertar (un esquema o una instancia).

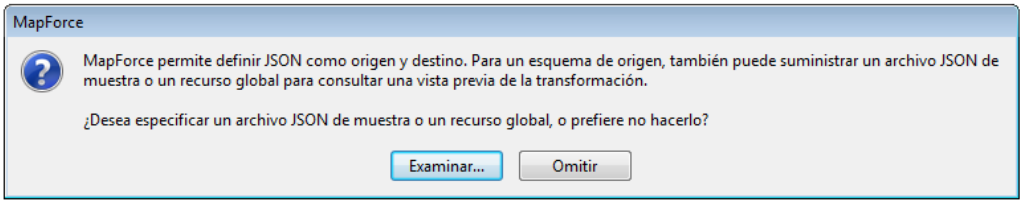


2. Seleccione la opción correspondiente.

> El tamaño del archivo de instancia JSON no puede superar la memoria disponible en el sistema.

Si hace clic en el botón **Instancia**, MapForce genera automáticamente el esquema basándose en el archivo de instancia y le pregunta dónde se debe guardar.

Si hace clic en el botón Esquema, MapForce solicita un archivo JSON de muestra o un recurso global (*véase Recursos globales*). Esto es necesario para generar una vista previa de la transformación.

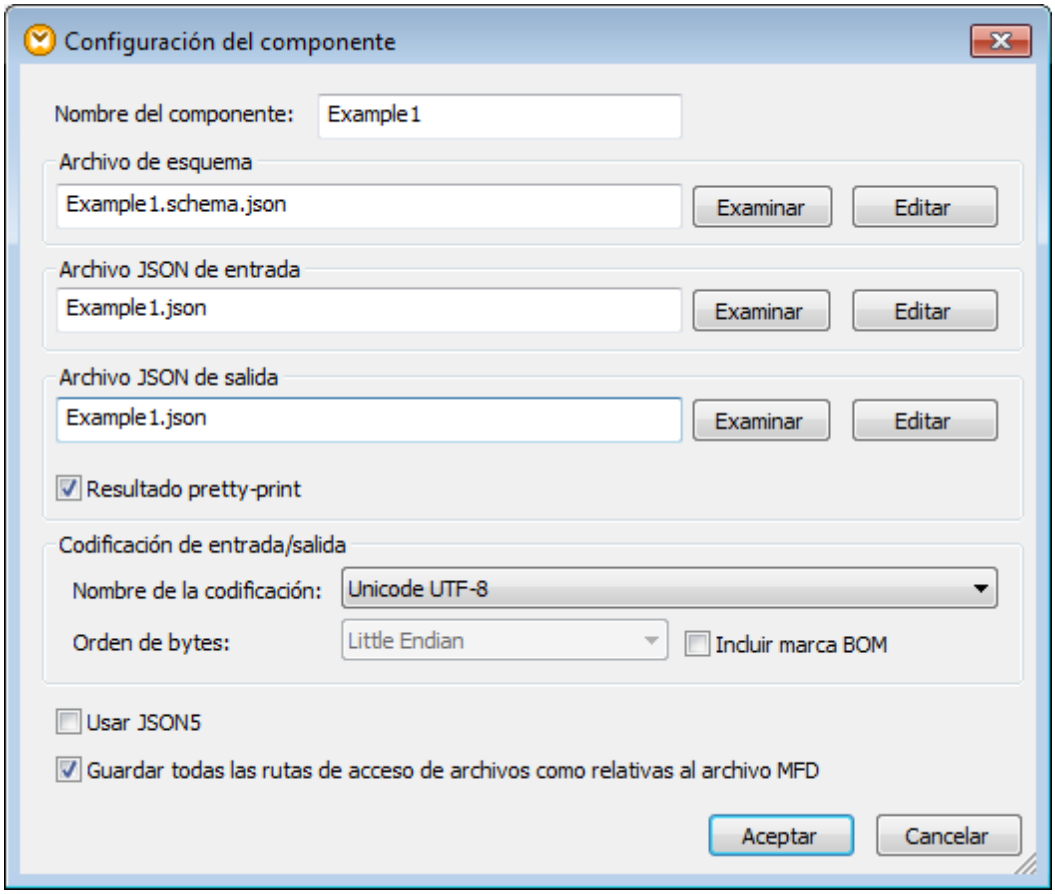


7.6.2 Configuración de componentes JSON

Para cambiar la configuración de un componente JSON haga clic con el botón derecho en la barra de título del componente y elija **Propiedades** en el menú contextual. También puede hacer

© 2018 Altova GmbH

doble clic en la barra de título del componente simplemente.



Cuadro de diálogo "Configuración del componente"

Estas son las opciones de configuración que ofrece el cuadro de diálogo:

Nombre del componente	Aquí puede cambiar el nombre que aparece en pantalla para el componente JSON.
Archivo de esquema	Especifica el nombre y la ruta de acceso del archivo de esquema. Para cambiar la ubicación del archivo haga clic en **Examinar** y seleccione el archivo nuevo. Para editar el archivo en su editor JSON (p. ej. XMLSpy) haga clic en **Editar**.
Archivo de entrada JSON	Especifica el nombre y la ruta de acceso del archivo JSON de entrada. Para cambiar la ubicación del archivo haga clic en **Examinar** y seleccione el archivo nuevo.

	Para editar el archivo en su editor JSON (p. ej. XMLSpy) haga clic en **Editar**.
Archivo de salida JSON	Especifica el nombre del archivo de instancia JSON de destino y la ruta de acceso donde se debe guardar (cuando MapForce Server ejecute la asignación). También es la ubicación predeterminada del archivo de salida cuando se guardan resultados desde el panel *Resultados*. Para cambiar la ubicación del archivo haga clic en **Examinar** y seleccione el archivo nuevo. Para editar el archivo en su editor JSON (p. ej. XMLSpy) haga clic en **Editar**.
Resultado pretty-print	Ajusta el formato del documento JSON cuando se ejecuta la asignación, dándole un formato estructurado. Cada nodo secundario se desplaza de su primario con un carácter de tabulación.
Codificación	Permite configurar estos aspectos del archivo de instancia de salida: • nombre de la codificación • orden de bytes • si se incluye o no el carácter de marca BOM. Los archivos JSON suelen estar codificados en UTF (véase http://tools.ietf.org/html/rfc7159#section-8.1). Las demás codificaciones no se consideran codificaciones estándar.
Usar JSON5	Marque esta casilla si el archivo JSON de entrada del componente usa la sintaxis JSON5. Si marca esta casilla en un componente de destino, MapForce creará un archivo .json5 donde las claves de objeto no tendrán comillas si son identificadores ECMAScript5 válidos (*véase Compatibilidad con JSON5*).
Guardar todas las rutas de acceso de archivos como relativas al archivo MFD	Si habilita esta opción, MapForce guarda las rutas de acceso que aparecen en el cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd). Esta opción de configuración afecta a estos archivos: • el archivo de esquema JSON • el archivo JSON de entrada • el archivo JSON de salida Véase también Usar rutas de acceso relativas en un

© 2018 Altova GmbH

	componente.

7.6.3 Compatibilidad con JSON5

JSON5 es una propuesta de extensión para JSON cuyo objetivo es facilitar la escritura a mano de archivos JSON. Es importante tener en cuenta que JSON5 no es el sucesor oficial de JSON (*véase* *http://json5.org/*).

Los archivos JSON5 tienen la extensión .json5, pero cuando se escribió esta documentación no existía todavía ningún tipo MIME oficial para los archivos .json5. No obstante, MapForce reconoce la extensión .json5 cuando este tipo de archivos se añaden a la asignación. Si implementa la asignación en FlowForce Server (*véase* *Implementar asignaciones en FlowForce Server*), se usará el tipo MIME "`application/json5`" para los archivos .json5.

Requisitos

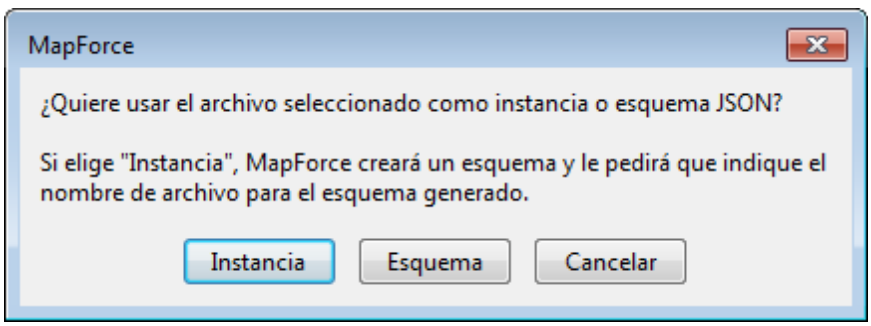
- Para poder asignar datos desde o hasta archivos JSON5, MapForce necesita un esquema JSON válido (que no debería usar la sintaxis JSON5). Si no tiene un esquema JSON para su archivo, MapForce generará uno automáticamente (ver más abajo). El esquema que MapForce genera automáticamente será un esquema JSON (no JSON5). También puede generar un esquema JSON para archivos .json5 con Altova XMLSpy.

Limitaciones

- No se admite la asignación de comentarios desde o hasta archivos JSON5.
- Como MapForce funciona con tipos de datos estrictos, no se puede calificar ni asignar valores como `Infinity`, `-Infinity`, `NaN` o `-NaN`.

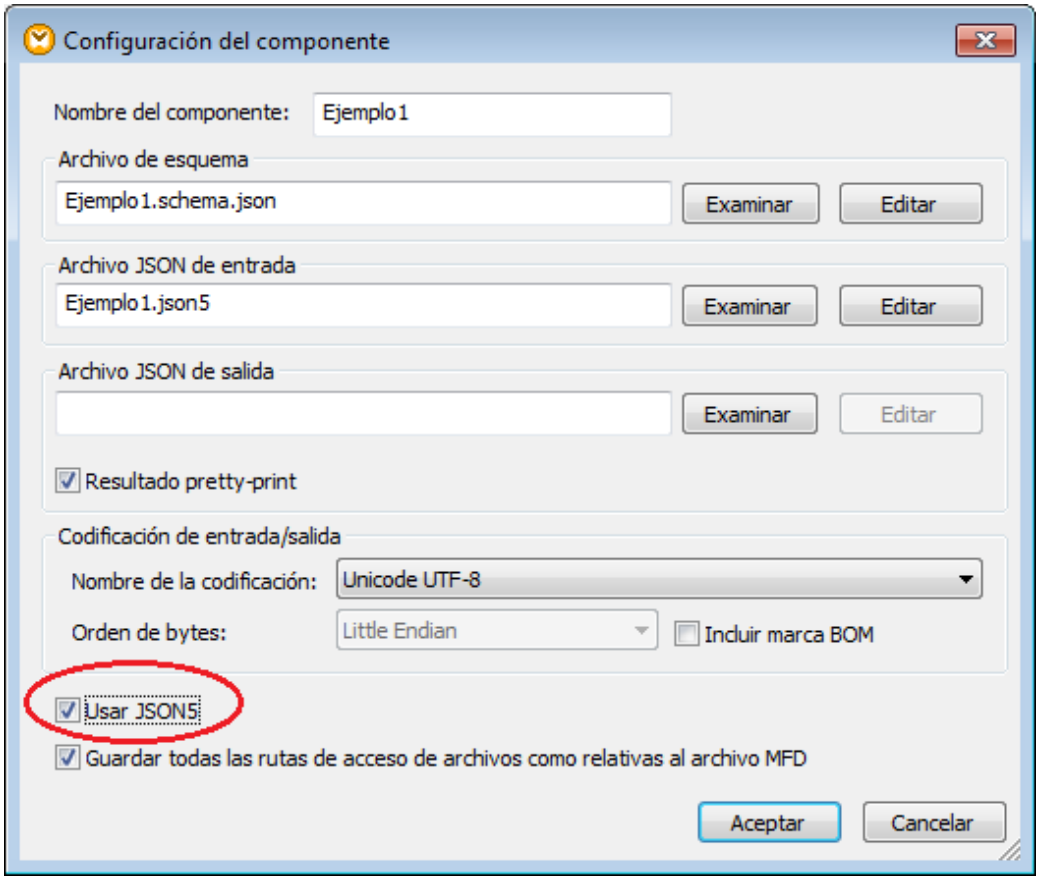
Asignar datos de archivos .json5

1. En el menú **Insertar** seleccione el comando **JSON** y navegue hasta el archivo de instancia .json5 (también puede hacer clic en el botón de la barra de herramientas). Si tiene un archivo de esquema JSON y prefiere aportar el archivo de instancia .json5 más tarde, navegue hasta el esquema JSON (no JSON5). El funcionamiento es el mismo que cuando se añaden archivos JSON planos (*véase* *Agregar archivos JSON como componente de asignación*).



2. Tras añadir el componente JSON a la asignación, haga clic con el botón derecho en su

barra de título y seleccione **Propiedades** en el menú contextual. Ahora, en el cuadro de diálogo "Configuración del componente", marque la casilla *Usar JSON5* para indicar a MapForce que debe tratar los datos de instancia como datos JSON5 (*véase* *Configuración de componentes JSON*).



Para generar un archivo JSON5 con MapForce debe marcar la casilla *Usar JSON5* (*imagen anterior*) y conectar los datos de entrada necesarios al componente JSON en la asignación. Cuando se marca esta casilla, MapForce no pondrá las claves de objeto entre comillas, excepto cuando esto daría lugar a una instancia JSON5 no válida. La extensión de archivo deseada (p. ej. "json5") también se puede configurar desde el cuadro de diálogo "Configuración del componente" (*imagen anterior*), introduciendo el valor correspondiente en el campo *Archivo JSON de salida*.

7.6.4 Ejemplo: asignación de datos entre JSON y CSV

En este ejemplo encontrará una asignación de datos que convierte datos de un archivo de instancia JSON en un archivo de texto separado por comas. En este apartado explicamos también cómo se pueden asignar valores de propiedades adicionales que pueden estar en archivo de instancia JSON, pero no en el esquema.

A continuación puede ver el esquema JSON que se utilizó para este ejemplo. Como indica la palabra clave `$schema`, el esquema valida instancias JSON con el borrador de especificación 04 JSON Schema. Describe una matriz llamada `people` compuesta por varios objetos `person`. El

 © 2018 Altova GmbH

objeto `people` debe contener como mínimo un objeto `person` para poder ser válido. Cada objeto `person` tiene las propiedades `name`, `age` y `email`. Recuerde que `name` y `email` son de tipo `string`, mientras que `age` es de tipo `integer`. Además, las propiedades `name` y `email` son obligatorias, mientras que `age` es opcional.

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "title": "people",
  "type": "array",
  "items": {
       "title": "person",
       "type": "object",
    "required": [
      "name",
      "email"
    ],
    "properties": {
      "name": {
        "type": "string"
      },
      "email": {
        "type": "string",
        "format": "email"
      },
      "age": {
        "type": "integer"
      }
    }
  },
  "minItems": 1
}
```

El archivo de instancia JSON contiene registros de personas y debe pasarse a CSV.

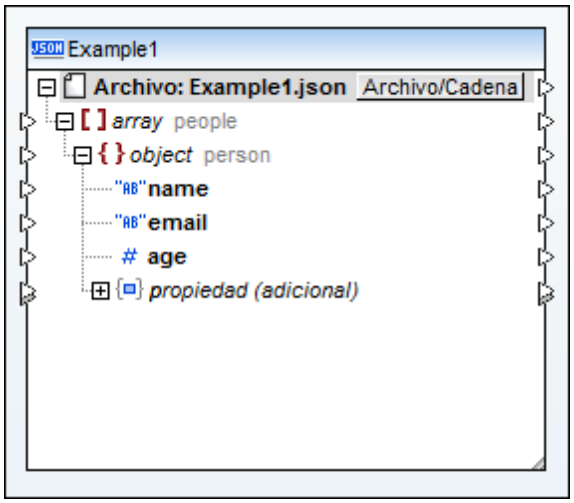
```
[
       {
              "name": "Alethia Alonso",
              "email": "altethia@example.com",
              "age": 35,
              "birthday": "4 July"
       }, {
              "name": "Klaus Mauer",
              "email": "klaus@example.com",
              "age": 57,
              "birthday": "31 August"
       }, {
              "name": "Natsuo Shinohara",
              "email": "natsuo@example.com",
              "age": 29
       }
]
```

El texto resaltado en amarillo muestra que la primera y la segunda persona tienen una propiedad adicional que no está definida en el esquema: la propiedad `birthday`. Sin embargo, la instancia

JSON es válida porque el esquema no contiene una propiedad `additionalProperties` para el objeto `person`. Cuando no está presente en el esquema JSON, la propiedad `additionalProperties` tiene el valor predeterminado **true**, lo cual significa que en la instancia JSON este objeto puede tener tantas propiedades adicionales como sean necesarias y la instancia JSON seguirá siendo válida.

Para crear una asignación entre datos JSON y CSV:

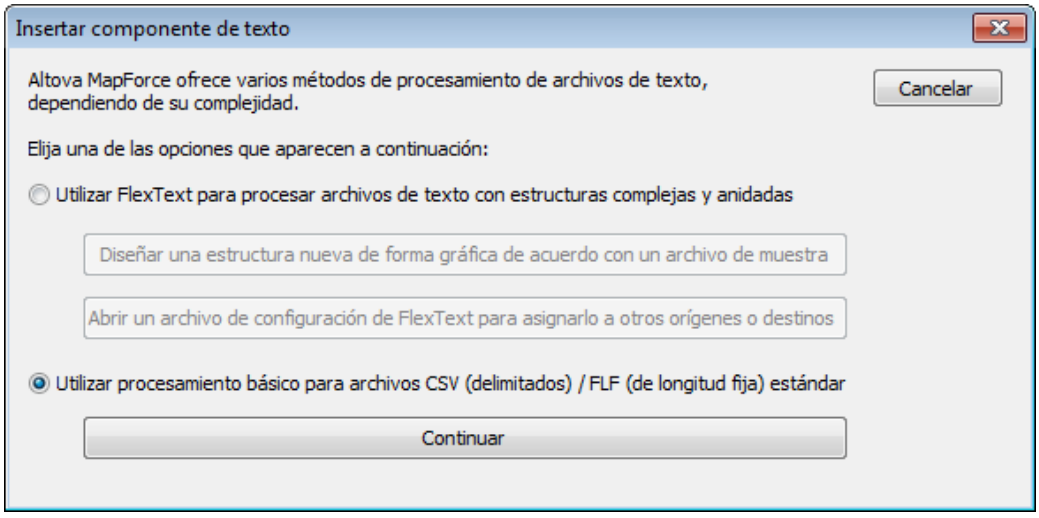
1. Haga clic en el comando de menú **Insertar | Archivo o esquema JSON** y navegue hasta el archivo **Example1.schema.json** de la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial**. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo **Example1.json**. Llegados a este punto el componente de MapForce tendrá este aspecto:



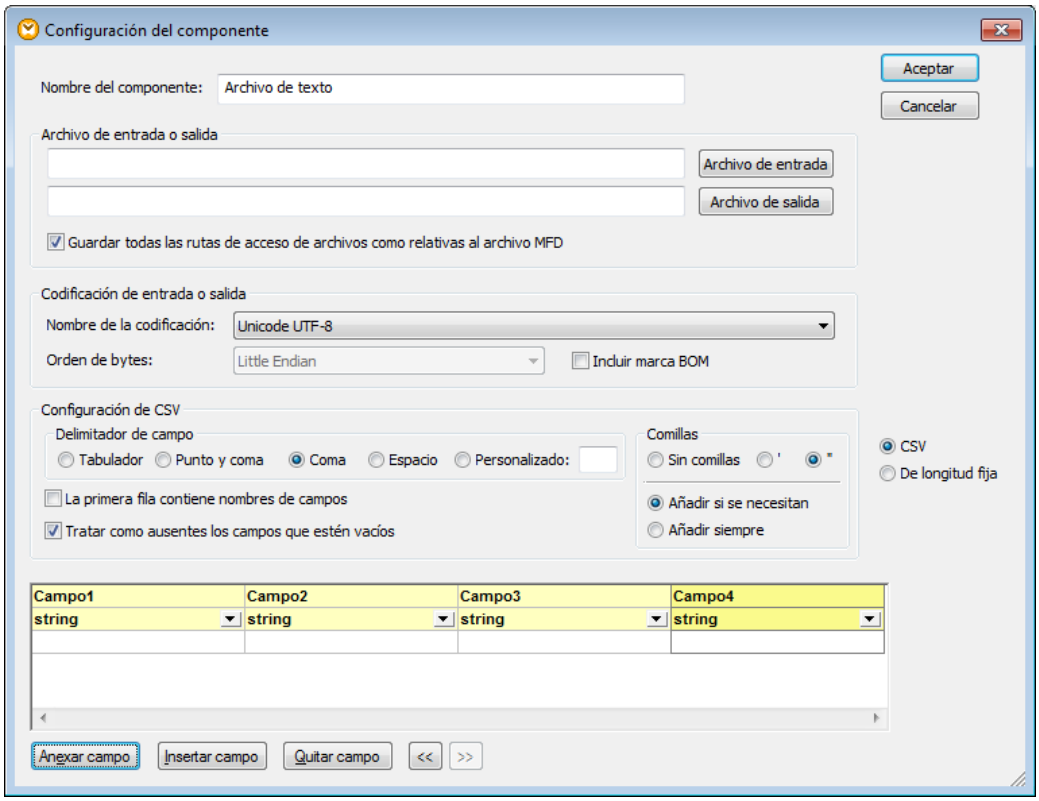
   La estructura del componente de MapForce recuerda a la del archivo JSON propiamente dicho, con la excepción del nodo propiedad (adicional). Este nodo indica que la propiedad `additionalProperties` del objeto `person` no está presente o que tiene el valor **true** en el esquema. Esto significa que el esquema puede contener propiedades adicionales personalizadas. Por eso MapForce presenta este nodo, en caso de que desee asignar datos de propiedades adicionales del objeto (ver pasos siguientes).

2. Haga clic en el comando de menú **Insertar | Archivo de texto**.

© 2018 Altova GmbH



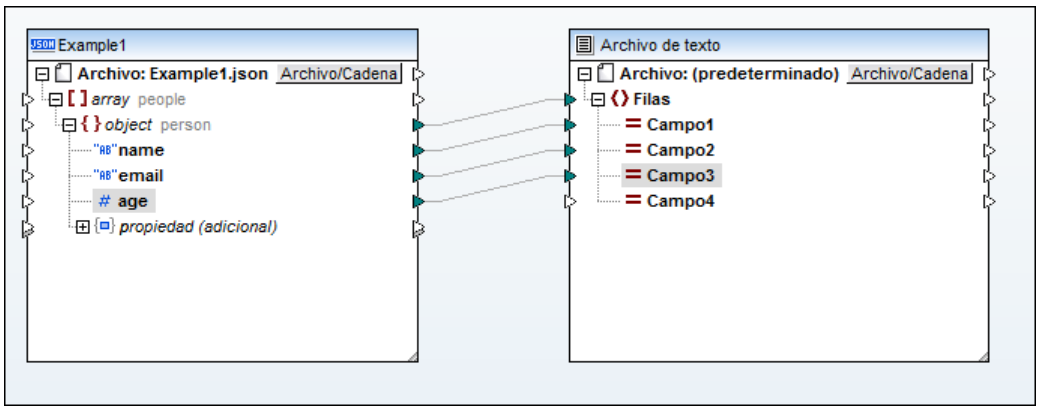
3. Seleccione la opción *Utilizar procesamiento básico para archivos CSV* y después haga clic en el botón **Continuar**.



4. Haga clic cuatro veces en el botón **Anexar campo** (situado al final del cuadro de diálogo) para agregar cuatro campos CSV al componente de texto. Después haga clic en **Aceptar**. Tres campos del componente CSV se asignarán a las propiedades `name`, `email` y `age` del objeto `person` y el cuarto campo se asignará a la propiedad adicional `birthday`.

5. En el área de asignación conecte el objeto `person` del componente JSON con el nodo

`Filas` del componente CSV. Después conecte las propiedades `name`, `email` y `age` del objeto `person` con los tres primeros campos del componente de texto. Llegados a este punto la asignación tendrá este aspecto:



Ahora tiene la opción de dar a los campos CSV el nombre de las propiedades del objeto (es decir, *name*, *email* y *age*). Para eso basta con usar la opción **Conexión automática de secundarios**, que creará todas las conexiones automáticamente (*véase* *Conectar secundarios equivalentes*).

Si ahora consulta una vista previa del resultado de la transformación (en el panel *Resultados*), obtendrá:

```
Alethia Alonso,altethia@example.com,35,
Klaus Mauer,klaus@example.com,57,
Natsuo Shinohara,natsuo@example.com,29,
```

Tal y como muestra el resultado, por cada objeto `person` del archivo JSON, se crea una fila nueva y las propiedades del objeto están separadas por comas, tal y como se pretendía.

Sin embargo, todavía no hemos creado ninguna conexión entre la propiedad adicional `birthday` que existe en el archivo JSON de instancia (aunque no exista en el esquema).

Para crear una asignación desde la propiedad adicional del objeto JSON:

1. Expanda el nodo `propiedad (adicional)` del objeto `person`. Observe que aparece el nodo **`nombre`**. Este nodo permite acceder a la propiedad adicional por su nombre. En nuestro ejemplo, el nombre de la propiedad adicional es "birthday" y su tipo es "string".

2. Inserte estos componentes en el área de asignación:

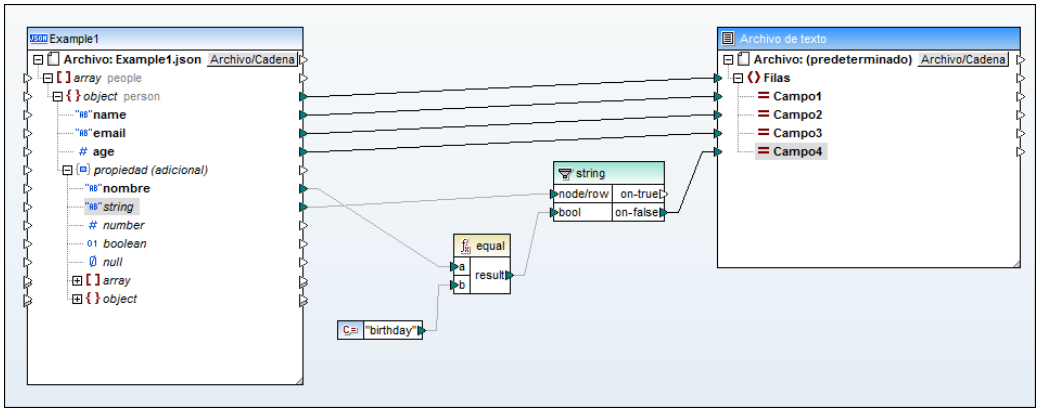
   a. una constante con el valor `birthday`,
   b. un filtro de nodos y filas y
   c. la función lógica **`equal`**.

3. Ahora cree estas conexiones entre los componentes:

© 2018 Altova GmbH

- Desde el nombre de la propiedad adicional hasta la entrada "a" de la función **equal**.
- Desde la constante **birthday** hasta la entrada "b" de la función **equal**.
- Desde la salida de la función **equal** hasta la entrada **bool** del componente de filtrado de nodos y filas.
- Desde el elemento **string** de la propiedad adicional hasta la entrada **node/row** del componente de filtrado de nodos y filas.
- Desde la salida **on-true** del componente de filtrado de nodos y filas hasta el **Campo4** del componente de texto.

Estas conexiones consiguen que MapForce busque una propiedad adicional llamada **birthday** y, si encuentra un valor de cadena, lo copie al **Campo4** del componente de texto.



Si ahora consulta una vista previa del resultado de la transformación (en el panel *Resultados*), obtendrá:

```
Alethia Alonso,altethia@example.com,35,4 July
Klaus Mauer,klaus@example.com,57,31 August
Natsuo Shinohara,natsuo@example.com,29,
```

Tal y como muestra el resultado, el cuarto campo CSV incluye ya el valor de la única propiedad adicional de tipo "string": `birthday`. Además, como la tercera persona no tiene fecha de cumpleaños, no habrá ningún valor en la posición correspondiente del archivo CSV.

Puede usar la misma técnica para crear asignaciones desde otras propiedades adicionales del archivo de instancia JSON. Por lo general, recomendamos que el esquema incluya todas las propiedades que se desean asignar. Sin embargo, si la instancia JSON contiene propiedades que dependen de datos (no fijas), entonces puede usar la técnica que se describe en este apartado.

7.7 Microsoft OOXML Excel 2007+

MapForce puede leer y escribir datos en libros Microsoft Excel 2007+ en el formato predeterminado *Office Open XML (OOXML)*. Este formato se introdujo por primera vez en Microsoft Office 2007 y, en el caso de los libros de Excel, se asocia con la extensión predeterminada .xlsx. En la interfaz de MapForce y en esta documentación este tipo de archivos reciben la denominación general "archivos Excel 2007+".

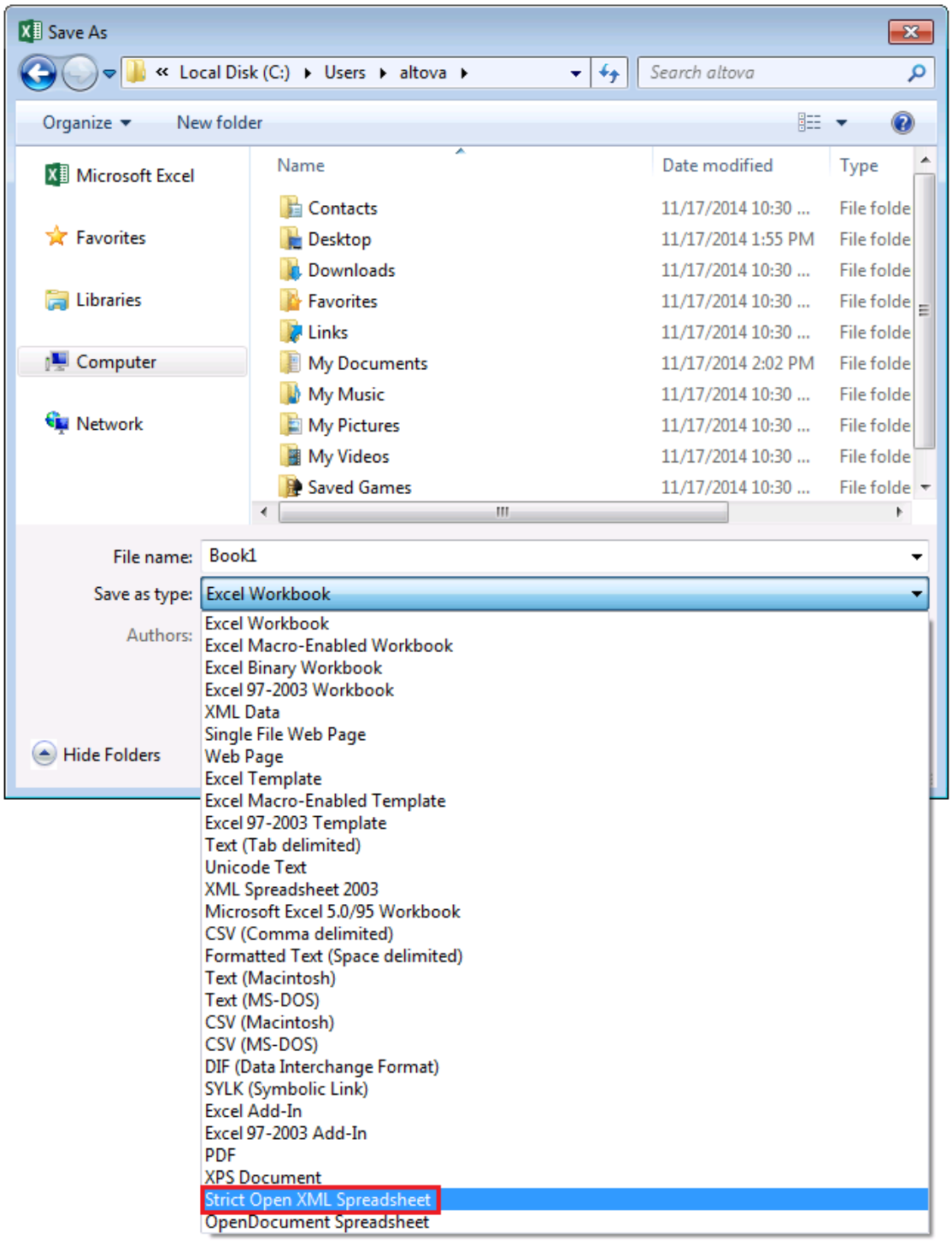
Microsoft Office 2013 incluyó compatibilidad para el formato *Strict Open XML Spreadsheet* format (ISO/IEC 29500 Strict). El formato *Strict Open XML Spreadsheet* también tiene la extensión .xlsx, pero se trata de un formato distinto que sigue reglas de validación más estrictas.

En la tabla que aparece a continuación puede ver la compatibilidad de los lenguajes de transformación de MapForce con los libros de Excel 2007+. Observe las diferencias existentes entre el formato *Office Open XML* y el formato *Strict Open XML Spreadsheet*.

Formato Microsoft Excel	Lenguaje de MapForce	Compatible con lectura	Compatible con escritura
Office Open XML	BUILT-IN	Sí	Sí
	C#	Sí	Sí
	Java	Sí	Sí
	XSLT2	Sí	No
Strict Open XML Spreadsheet	BUILT-IN	Sí	No
	C#	Sí	No
	Java	Sí	No
	XSLT2	Sí	No

Si necesita pasar archivos *Office Open XML* generados previamente con MapForce al formato *Strict Open XML Spreadsheet*, abra el libro en Excel 2013 y guárdelo como *Strict Open XML Spreadsheet*.

© 2018 Altova GmbH



Guardar en formato Strict Open XML Spreadsheet

Estas son las características generales de los componentes de asignación Excel 2007+ en MapForce:

- Puede asignar datos Excel 2007+ a cualquier componente compatible y viceversa, incluso a archivos de taxonomía XBRL (*véase Ejemplo: asignar datos Excel a XBRL*).
- Si tiene Microsoft Excel 2007 o superior instalado en el equipo, podrá ver inmediatamente una vista previa de la transformación en el panel *Resultados* y también podrá guardar la vista previa en un archivo. Si no tiene Excel 2007 o superior instalado, podrá realizar asignaciones de datos Excel 2007+ pero no podrá consultar la vista previa en el panel *Resultados* (aunque podrá guardarlos con el comando **Guardar el archivo de salida** del menú **Resultados**).
- Los documentos Excel 2007+ de salida generados no contienen formato, solamente datos. Si necesita aplicar formato al documento de salida Excel generado por MapForce, puede hacer referencia a los rangos de celdas generados desde otra hoja de cálculo Excel y aplicar formato en la hoja de cálculo referente. Para más información consulte el apartado Ejemplo: suministrar datos a hojas Excel con formato previo.
- Además de generar y guardar los resultados manualmente, puede compilar el diseño de asignación en un archivo de ejecución de MapForce Server o implementarlo en un servidor FlowForce Server y ejecutarlo cuando lo necesite o con ayuda de trabajos de FlowForce. Para más información puede consultar los apartados Compilar una asignación de MapForce e Implementar una asignación de MapForce respectivamente.
- Puede generar código de asignación de datos (ver lenguajes compatibles más arriba) y ejecutar la asignación desde Visual Studio 2008, 2010, 2013, 2015, 2017 o cualquier otra aplicación. Consulte la sección Generador de código para obtener más información.

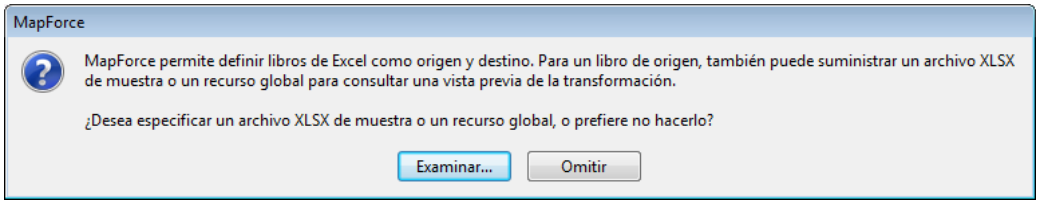
7.7.1 Agregar archivos Excel 2007+ como componentes de la asignación

Para agregar un componente Microsoft Excel 2007+ (*.xlsx) al área de asignación:

1. Para empezar tiene dos opciones:

   - Seleccionar el comando **Insertar | Archivo Excel 2007+**.
   - Seleccionar el botón **Insertar archivo Excel 2007+** de la barra de herramientas.

2. Aparece este aviso:



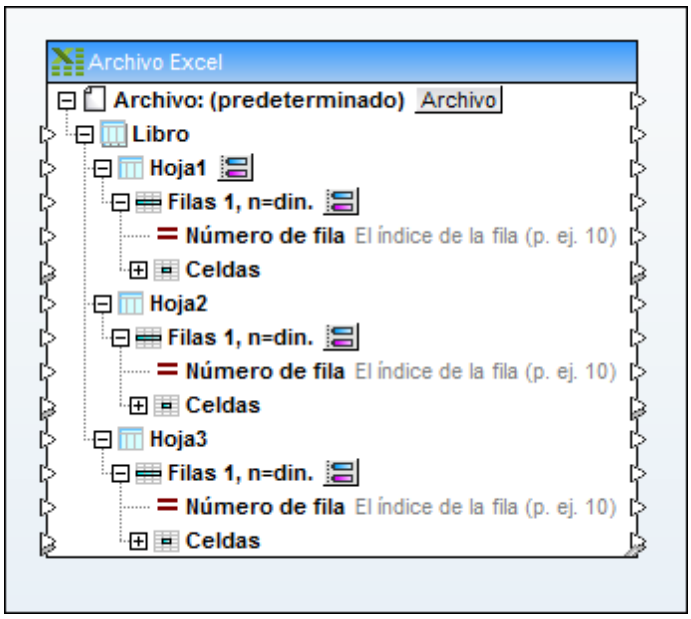
3. Ahora tiene dos opciones:

   - Si tiene pensado asignar datos ***de*** un libro de Microsoft Excel, haga clic en **Examinar** y seleccione el archivo .xlsx cuyos datos desea asignar. MapForce usa el archivo Excel de muestra para leer los nombres de las hojas de cálculo y los rangos de celda. Si tiene archivos Excel definidos como recursos globales, también puede seleccionarlos desde el cuadro de diálogo "Recursos globales" (*véase Recursos globales*). Haga clic en **Omitir** si prefiere elegir el archivo de muestra más tarde.
   - Si tiene pensado asignar datos ***a*** un libro de Microsoft Excel, haga clic en Omitir.

Cuando se ejecute la transformación de la asignación, MapForce generará un archivo Excel de salida llamado **xlsx-mapforce.xlsx** en la carpeta de la asignación. Si es necesario, cambie el nombre del archivo de salida en el cuadro de diálogo "Configuración del componente" (*véase* *Información sobre componentes Excel 2007 +*).

7.7.2 Información sobre componentes Excel 2007+

Si añade un archivo Excel 2007+ al área de asignación sin especificar un archivo de muestra (véase Agregar archivos Excel 2007+ como componentes de la asignación), MapForce crea un componente predeterminado que incluye tres hojas de cálculo (imagen siguiente). Si por el contrario especifica un archivo de muestra, MapForce lee el archivo y solamente crea las hojas de cálculo correspondientes.



Componente Excel predeterminado

En MapForce la estructura del componente Excel 2007+ refleja la estructura de los datos del libro de Excel, con la diferencia de que MapForce representa la estructura de forma jerárquica (lo cual permite crear asignaciones a nivel de celda).

Para poder crear asignaciones entre el componente Excel 2007+ y otros componentes de asignación es necesario indicar qué columnas y filas se deben usar en la asignación de datos. A diferencia de otros componentes de MapForce como los componentes XML o JSON, los archivos Excel 2007+ no tienen un esquema explícito que le sirva a MapForce para inferir la estructura de los datos. Por ello, MapForce necesita que definamos:

- qué datos concretos (hojas de cálculo, rangos con nombre y tablas, columnas, filas...) deben seleccionarse en el libro de Excel y
- en qué hojas de cálculo, columnas y filas se deben escribir los datos (si el destino de la asignación es un libro de Excel).

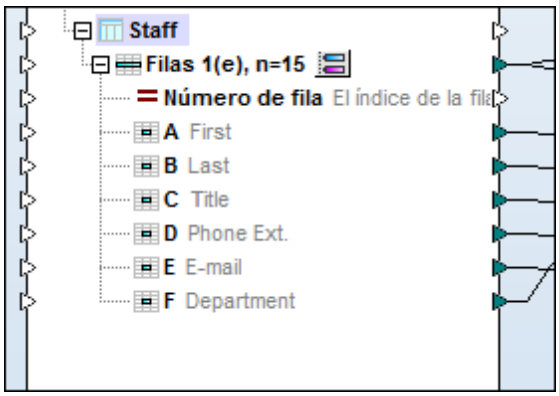
También puede configurar el componente para que lea datos de varias ubicaciones del libro de

Excel o para que escriba datos en varias ubicaciones en la misma operación de asignación de datos.

La configuración se realiza desde el componente directamente y con ayuda de estos iconos:

Elemento del componente	Icono	Descripción
Libro		Representa un libro de Excel.
Hoja		Representa una hoja de cálculo de Excel. El botón  que aparece junto a la primera hoja de cálculo sirve para configurar las hojas de cálculo (*véase* *Agregar y eliminar hojas de cálculo*).
Filas		Representa un rango de filas Excel. Puede añadir varios rangos de filas de la misma hoja de cálculo (*véase* *Agregar y eliminar rangos de filas*). Esto permite leer (y escribir en) varios rangos de celdas en la misma operación de asignación de datos. Por cada rango que se defina, podrá especificar opciones de selección de datos distintas. Por ejemplo, un rango puede empezar en la fila 1 e incluir todas las columnas de esa fila. Y otro rango puede empezar en la fila 3 y estar compuesto por un número dinámico de filas, dependiendo de la cantidad de datos que incluya el archivo Excel de origen. El componente incluye elementos gráficos que le ayudarán a ver la configuración de rangos: **Filas *n*** Indica un rango que empieza en la fila ***n***. **Fila *n*** Indica un rango de una sola línea de la fi **Filas prev+*n*** Indica un rango que empieza ***n*** filas desp del rango anterior. **Filas *n*(h)** Indica un rango que empieza en la fila ***n*** primera fila está definida como fila de encabezado. **n=*n*** Indica un rango que está compuesta por exactamente. **n=dyn** Indica un rango dinámico. Los rangos dinámicos pueden tener un número ilimit de filas. El botón  que aparece junto a cada rango de filas sirve para especificar opciones avanzadas de selección de datos para cada rango.

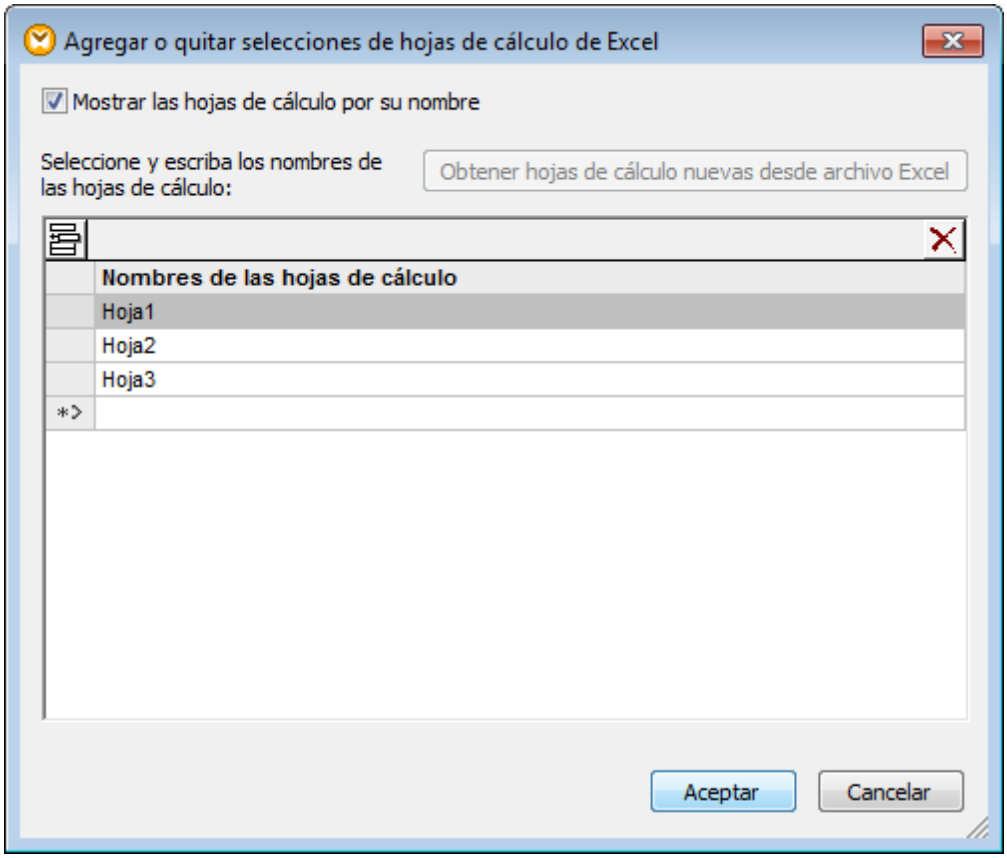
 © 2018 Altova GmbH

Elemento del componente	Icono	Descripción
Celdas		Representa todas las celdas (columnas) de una fila concreta. Este elemento aparece si el componente se configuró para presentar una sola celda para todas las columnas (configuración predeterminada de MapForce) Si lo prefiere, puede configurar el componente para que muestre todas las columnas por separado (*imagen siguiente*) 
Cambiar selección		El botón que aparece junto a cada hoja de cálculo o fila sirve para definir opciones de configuración para ese contexto concreto. Con este botón puede modificar toda la estructura del componente Excel a efectos de la asignación de datos (*véase* *Configuración de componentes Excel 2007+*) Por ejemplo, si la asignación lee datos de un archivo Excel, puede especificar el libro, la fila y la columna donde MapForce debe leer datos. Si por el contrario la asignación escribe datos en un archivo Excel, puede especificar la hoja de cálculo, fila y columna donde MapForce debe escribir los datos.

7.7.3 Agregar y eliminar hojas de cálculo

Puede agregar y eliminar hojas de cálculo en el componente Excel 2007+ a mano o automáticamente, cargándolas desde un archivo .xslx de entrada.

Esto se hace con el botón que aparece junto al nodo `Hoja`.



Cuadro de diálogo "Agregar o quitar selecciones de hojas de cálculo de Excel"

Estas son las opciones disponibles:

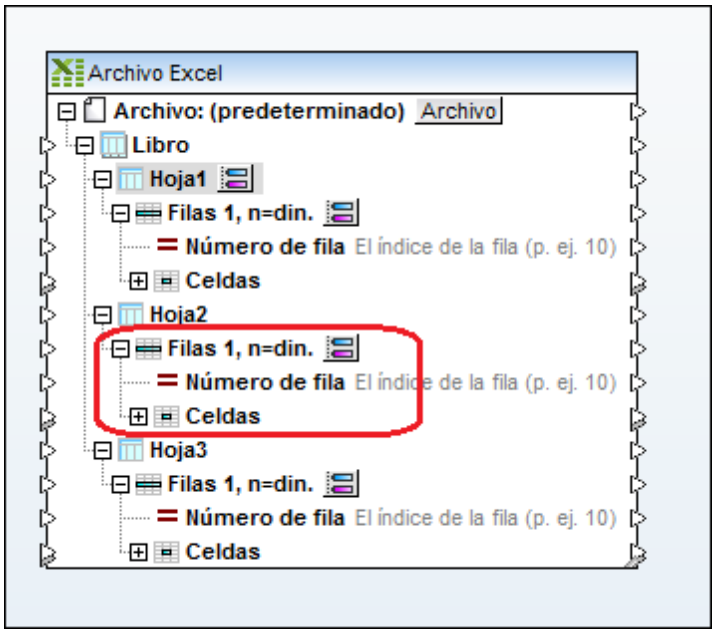
Mostrar las hojas de cálculo por su nombre	Debe seleccionar esta casilla si cada hoja del libro de Excel tiene un diseño distinto y, por tanto, debe aparecer por separado en MapForce. Si todas las hojas de cálculo del libro tienen la misma estructura, entonces puede desactivar esta casilla. Así MapForce podrá contraer los elementos Hoja y mostrar un solo elemento `Hoja` que representa a toda la colección de hojas de cálculo. Si un libro tiene varias hojas de cálculo y la casilla *Mostrar las hojas de cálculo por su nombre* no está activada, entonces MapForce tratará el libro como si fuera una hoja de cálculo. Así, la asignación de datos podrá procesar todas las hojas de una sola vez (siempre y cuando todas tengan la misma estructura).
Insertar ( *)*	Inserta una hoja de cálculo nueva antes de la que está

 © 2018 Altova GmbH

	seleccionada.
Anexar ( *)*	Anexa una hoja de cálculo nueva. Escriba el nombre en el campo de texto situado a la derecha del icono.
Eliminar ( *)*	Elimina la hoja de cálculo que está seleccionada.
Obtener hojas de cálculo nuevas desde archivo Excel	Introduce en el componente las hojas de cálculo nuevas que se detecten en el archivo Excel de entrada. Esta opción es importante en los componentes de origen que tienen un archivo Excel como entrada (*véase* *Configuración de componentes Excel 2007+*).

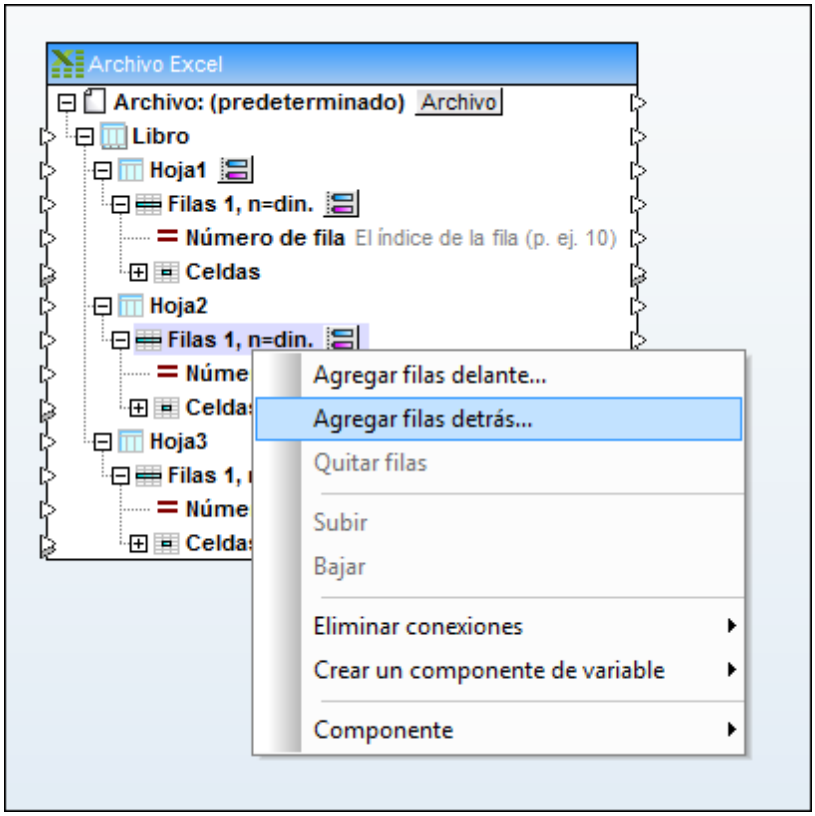
7.7.4 Agregar y eliminar rangos de filas

Puede configurar MapForce para que lea (o escriba en) un rango concreto de celdas en una ubicación concreta de la hoja de cálculo. En la imagen siguiente puede ver un componente con un rango de celdas en la Hoja2 del libro de Excel.



Por defecto, los rangos empiezan en la fila 1 y recorren n filas de forma dinámica (**n=din**). Sin embargo, puede cambiar la selección del rango y configurarlo (*véase* *Seleccionar rangos de celdas*).

Dentro de una hoja de cálculo puede crear tantos rangos de filas como quiera y eliminar los que no necesite. Para agregar un rango de filas nuevo haga clic con el botón derecho en cualquier nodo **`Filas`** y después seleccione **Agregar filas delante** o **Agregar filas detrás**.



Agregar rangos de celdas

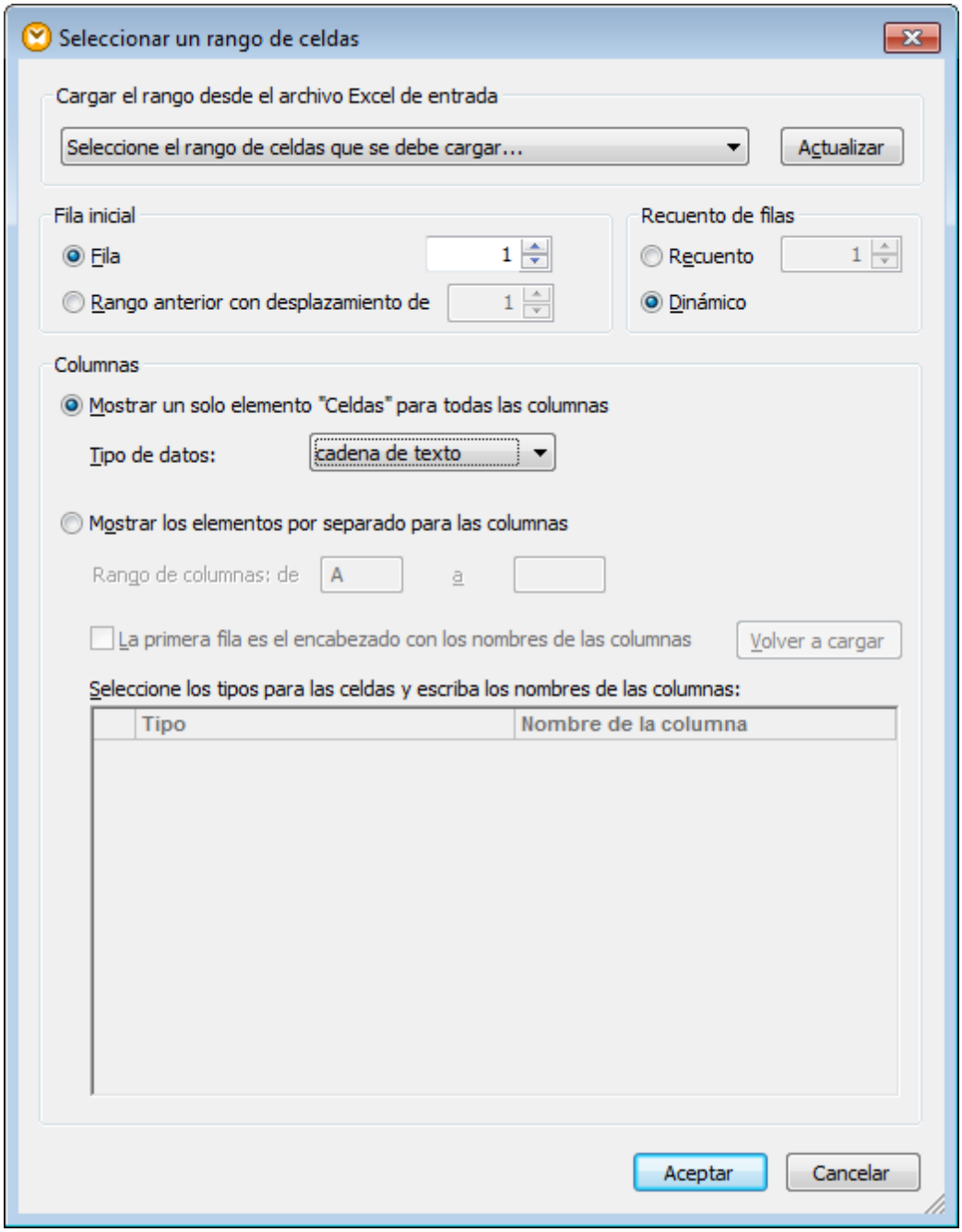
Para eliminar un rango haga clic con el botón derecho en él y después elija **Quitar filas** en el menú contextual.

Para cambiar la posición que tiene un rango dentro del componente haga clic con el botón derecho en el rango y después elija **Subir** o **Bajar** en el menú contextual.

7.7.5 Seleccionar rangos de celdas

Puede definir qué rango de celdas debe leer MapForce (cuando lea datos de un libro de Excel) o en qué rango de celdas debe escribir (cuando escriba datos en un libro de Excel). Esto se hace en el cuadro de diálogo "Seleccionar un rango de celdas". Para abrir este cuadro de diálogo haga clic en el botón que aparece junto a un rango de celdas.

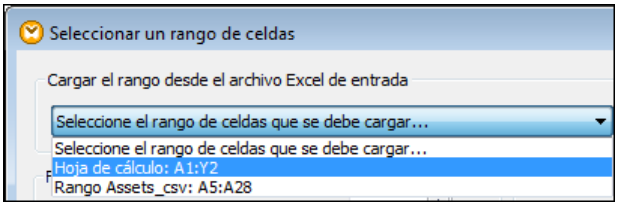
© 2018 Altova GmbH



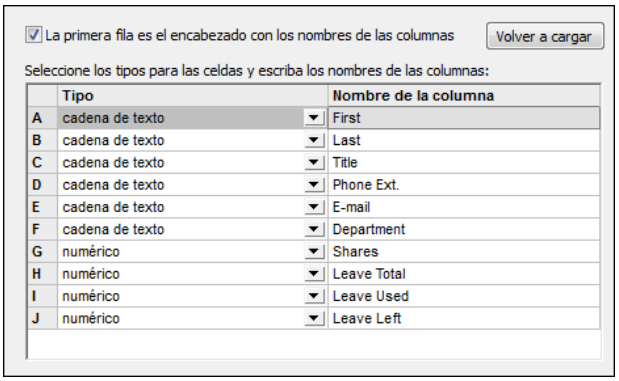
Cuadro de diálogo "Seleccionar un rango de celdas"

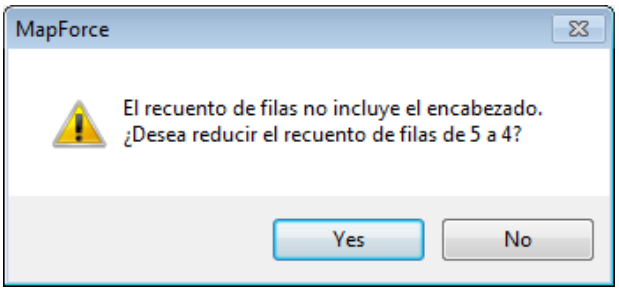
Estas son las opciones de configuración que ofrece este cuadro de diálogo:

Cargar el rango desde el archivo Excel de entrada	**Si se leen datos de un libro de Excel...** Use esta opción para seleccionar un rango de una hoja de cálculo, un rango con nombre o una tabla.

	 Si la opción *Mostrar las hojas de cálculo por su nombre* está deshabilitada (*véase* Agregar y eliminar hojas de cálculo), en esta lista podrá ver los datos de todas las hojas de cálculo. Con el botón **Actualizar** puede actualizar los rangos de celdas desde el archivo Excel de entrada. Recuerde que solamente son compatibles rangos rectangulares. **Si se escriben datos en un libro de Excel...** Esta opción estará deshabilitada.
Fila inicial	La opción Fila sirve para definir la primera fila de datos para este rango concreto. Por ejemplo, si introduce "5" como fila inicial, MapForce leerá (o escribirá) a partir de la quinta fila del libro de Excel. La opción *Rango anterior con desplazamiento de* solamente es relevante si en la misma hoja de cálculo ya hay otro rango. Esta opción indica a MapForce que debe desplazar el rango actual *N* filas hacia abajo desde el rango anterior. El valor de desplazamiento mínimo es 1.
Recuento de filas	**Si se leen datos de un libro de Excel...** **Recuento** define el número exacto de filas desde la que se deben leer los datos, empezando por la posición definida en la opción *Fila inicial*. Este valor se rellena automáticamente si seleccionó un rango con nombre o una tabla. **Dinámico** indica a MapForce que debe leer todas las filas que encuentre en los datos de origen a partir de la *Fila inicial*. Use esta opción si el rango es el último rango definido de la hoja de cálculo. De lo contrario los rangos siguientes no seleccionarán datos del archivo Excel de entrada. **Si se escriben datos en un libro de Excel...** **Recuento** define el número exacto de filas donde se deben escribir datos, empezando por la posición definida en la opción *Fila inicial*. Recuerde que si la instancia de entrada contiene más filas de las permitidas por **Recuento**, MapForce escribe solamente el número de filas definidas por **Recuento** y omite el resto de datos sin emitir advertencia alguna.

© 2018 Altova GmbH

	La opción Dinámico ordena a MapForce que debe escribir todas las filas que encuentre en la instancia de entrada, empezando por *Fila inicial*. Si definió una fila de encabezado con la opción *La primera fila es el encabezado con los nombres de las columnas*, la opción **Recuento** no tiene en cuenta la fila de encabezado (véase el archivo de ejemplo **Excel_Company_to_XML.mfd**). 
Mostrar un solo elemento "Celdas" para todas las columnas	Agrupa todos los elementos de tipo celda en un solo elemento llamado `Celdas` (*imagen siguiente*).  **Si se leen datos de un libro de Excel...** Use esta opción si desea leer todas las celdas de una fila concreta. Puede ver un ejemplo en los archivos **ExcelColumnsToRecords.mfd** y **ExcelWith2Dimensions.mfd** del proyecto **MapForceExamples** (carpeta **OOXML Excel 2007+**). **Si se escriben datos en un libro de Excel...** Use esta opción para escribir datos en celdas de la misma fila. Para ver un ejemplo consulte el archivo **Altova_Hierarchical_Excel.mfd**.
Mostrar los elementos por separado para las columnas	Esta opción sirve para acceder por separado a las columnas del rango de celdas dado. Si seleccionó un rango de hoja de cálculo, un rango con nombre o una tabla, los nombres de columna se rellenan automáticamente. De lo contrario, podrá seleccionar

	nombres de columna concretos con solo escribir su letra en los campos *de* y *a*. Si los rangos del archivo Excel de entrada han cambiado, haga clic en **Actualizar** para que los cambios se reflejen en el componente. Marque la casilla *La primera fila es el encabezado con los nombres de las columnas* si quiere que MapForce tenga en cuenta que la primera fila del rango es el título de la columna. Si está activada la opción **Recuento** y después activa/ desactiva la casilla *La primera fila es el encabezado con los nombres de las columnas*, MapForce preguntará si desea ajustar el valor de recuento. Esto evita que el recuento sea demasiado alto o bajo.  Recuerde que la opción **Recuento** no tiene en cuenta la fila de encabezado.

7.7.6 Insertar columnas entre columnas actuales

Cuando asigne datos a una hoja de cálculo Excel, podrá agregar columnas nuevas entre las columnas que ya existen, sin necesidad de rehacer las conexiones actuales. Basta con hacer clic en el botón derecho en una celda del componente Excel y seleccionar **Agregar celda delante** en el menú contextual.

Nota: el comando **Agregar celda delante** está disponible si la celda pertenece a un rango de la fila (es decir, debe ser secundario de un elemento `Fila` o `Filas` del componente Excel). Además, la opción *Mostrar los elementos por separado para las columnas* debe estar seleccionada en el cuadro de diálogo "Seleccionar un rango de celdas" (*véase* *Seleccionar rangos de celdas*).

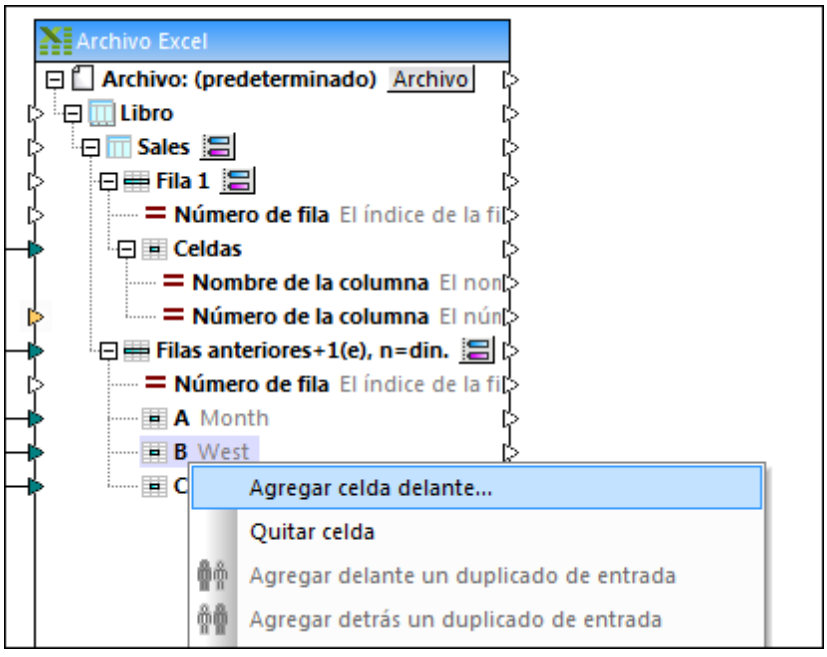
Por ejemplo, imaginemos que creó una asignación que lee datos de un archivo XML y los pasa a una hoja de cálculo Excel (como la asignación **Sales_to_Excel.mfd** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**). Esta asignación genera una hoja de cálculo Excel formada por tres columnas (imagen siguiente): las columnas *Month*, *West* y *East*.

	A	B	C	D
2	Month	West	East	
3	Jan	110.4	75.3	
4	Feb	114.3	65.2	
5	Mar	134.2	86.1	
6	Apr	107.3	112.1	
7	May	114.4	93.8	
8	Jun	113.9	72.4	
9	Jul	89.4	67.4	
10	Aug	95.3	84.9	
11	Sep	107.2	99.5	
12	Oct	129.7	82.5	
13	Nov	137.1	101.9	
14	Dec	152.6	120.6	
15				

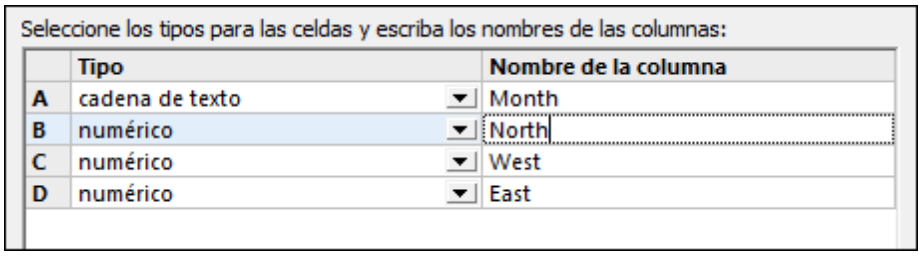
Output of mapping Sales_to_Excel.mfd

Imaginemos que necesita agregar una columna nueva llamada "North" justo antes de la columna "West". Ahora puede hacerlo sin necesidad de rehacer las conexiones actuales:

1. Haga clic con el botón secundario en la celda B de la asignación y seleccione **Agregar celda delante** en el menú contextual.

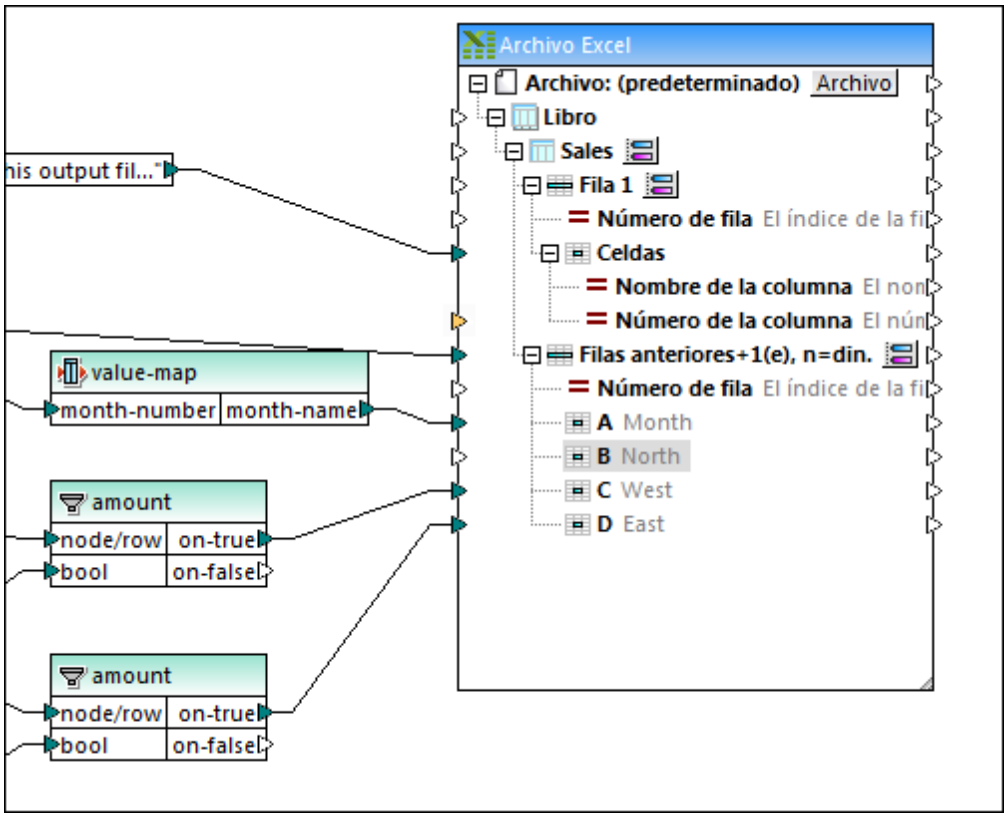


2. Si quiere puede introducir un nombre de columna (p. ej. "North").



3. Haga clic en **Aceptar** para terminar.

La columna nueva ("North") se añade a la asignación y ahora puede crear conexiones con esta columna. Las demás conexiones que ya existían no se ven afectadas por el cambio.



Sales_to_Excel.mfd mapping (después de los cambios)

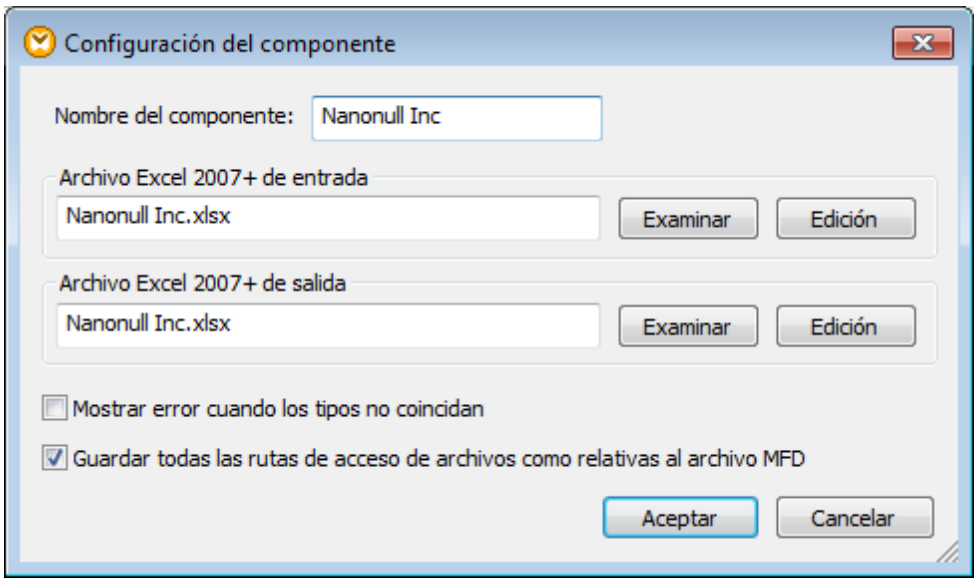
Puede usar el mismo método para eliminar columnas actuales sin modificar las conexiones que ya existen: haga clic con el botón derecho en una celda de la asignación y después elija **Quitar celda** en el menú contextual.

7.7.7 Configuración de componentes Excel 2007+

Tras añadir un componente Excel 2007+ al área de asignación, podrá configurarlo desde el cuadro de diálogo "Configuración del componente". Este cuadro de diálogo se puede abrir de varias maneras:

© 2018 Altova GmbH

- Seleccionando el comando de **Componente | Propiedades** (este menú solamente se habilita si hay un componente seleccionado).
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y seleccionando **Propiedades** en el menú contextual.



Cuadro de diálogo "Configuración del componente"

Estas son las opciones que ofrece el cuadro de diálogo:

Nombre del componente	Este nombre se genera automáticamente cuando se crea el componente en el área de asignación, pero puede cambiarse en este campo. Si el nombre del componente se generó automáticamente y después selecciona un archivo de instancia, MapForce le preguntará si desea actualizar también el nombre del componente. El nombre del componente puede incluir espacios (p. ej. "Archivo de entrada") y puntos (p. ej. "Orders.xlsx"). El nombre del componente no puede incluir barras diagonales, barras diagonales inversas, dos puntos, comillas dobles, espacios iniciales ni espacios finales. Cuando cambie el nombre del componente, tenga en cuenta que: • Si tiene pensado implementar la asignación en FlowForce Server, el nombre del componente debe ser un nombre único. • Se recomienda utilizar caracteres que sean

	compatibles con la línea de comandos. Recuerde que los caracteres regionales pueden tener codificaciones distintas en Windows y en la línea de comandos.
Archivo Excel 2007+ de entrada	Este es el archivo de Microsoft Excel donde MapForce debe leer datos. Este campo es relevante para los componentes de origen y MapForce utiliza este archivo para leer los nombres y las columnas de las hojas de cálculo de Excel. Para cambiar la ubicación del archivo de entrada haga clic en el botón **Examinar** y seleccione el archivo nuevo. Para editar el archivo en Microsoft Excel haga clic en el botón **Editar**.
Archivo Excel 2007+ de salida	Este es el nombre o la ruta de acceso del archivo de Microsoft Excel donde MapForce debe escribir datos. Este campo es relevante para el componente de destino. Para cambiar la ubicación del archivo de entrada haga clic en el botón **Examinar** y seleccione el archivo nuevo. Para editar el archivo en Microsoft Excel haga clic en el botón **Editar**. Si no se especifica un archivo de salida, MapForce genera resultados en un archivo llamado **xlsx-mapforce.xlsx**. Este archivo se genera por defecto en la misma carpeta del archivo .mfd (a no ser que se configure una ruta de acceso distinta). Los datos del archivo Excel de salida se sobrescribirán cuando se ejecute la transformación de la asignación de datos.
Mostrar error cuando los tipos no coincidan	Habilita los mensajes de error en la ventana Mensajes cuando un tipo de datos declarado en el componente y los datos del archivo .xlsx de entrada no coincidan. Por ejemplo, imagine que esta casilla está marcada y que en el cuadro de diálogo "Seleccionar un rango de celdas" declaró una columna como numérica. Si el archivo Excel contiene datos de texto en esta columna, MapForce emitirá un mensaje de error en la ventana Mensajes cuando conecte esta columna con el componente de destino y genere la vista previa de resultados.
Guardar todas las rutas de acceso de archivos como relativas al archivo MFD	Si marca esta casilla, MapForce guarda todas las rutas de acceso del cuadro de diálogo "Configuración del componente" como relativas a la ubicación del archivo de diseño de MapForce (.mfd). Consulte el apartado Usar rutas de acceso relativas en un componente para obtener más información.

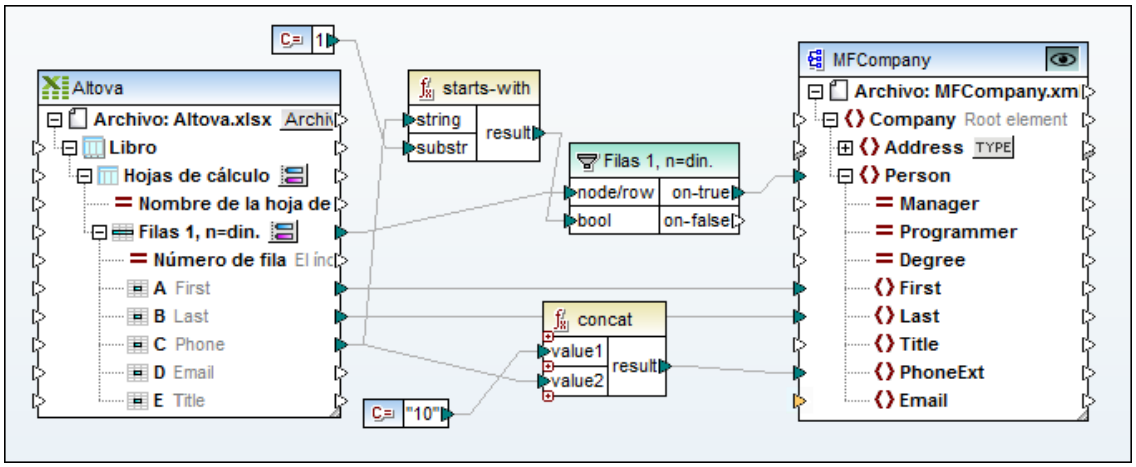
© 2018 Altova GmbH

7.7.8 Ejemplo: asignación de datos Excel 2007+ a XML

El archivo de asignación de este ejemplo está disponible en esta ruta de acceso: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Excel-mapping.mfd**. Cuando abra el archivo de ejemplo, observará que contiene tres transformaciones de asignación de datos distintas. En este apartado nos ocupamos de las dos primeras.

El objetivo de la primera asignación de datos es:

- Seleccionar en el libro de Excel de origen las personas cuya extensión telefónica (columna C del libro) empiece por "1".
- Agregar el prefijo "10" a la extensión telefónica original y escribir la nueva extensión en un archivo XML de destino, junto con el nombre y el apellido de las personas.



Excel-mapping.mfd (ejemplo nº1)

La asignación de datos está configurada de la siguiente manera:

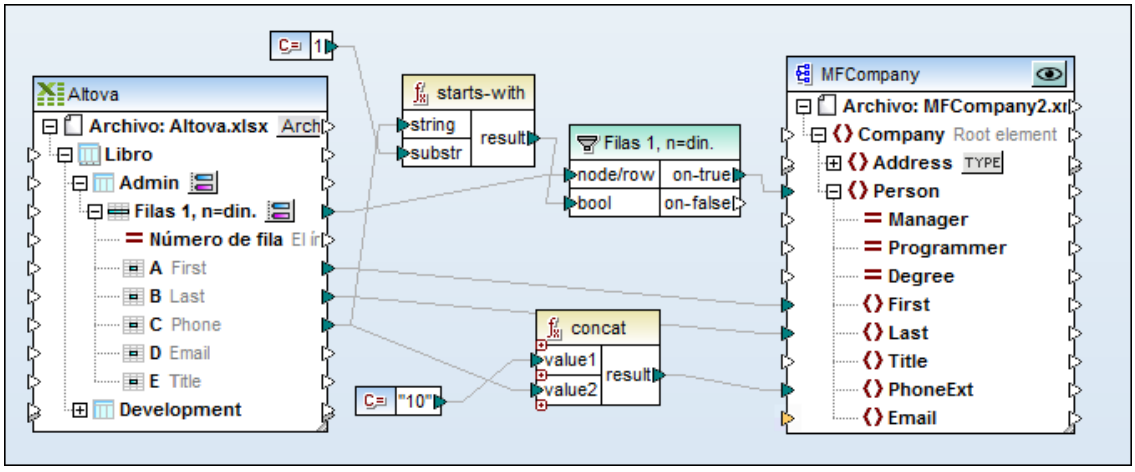
- **Altova.xlsx** es el libro de Excel 2007+ de origen. Las columnas A y B aportan el nombre y el apellido de las personas respectivamente. La columna C aporta el número de extensión telefónica.
- Las dos hojas de cálculo del libro aparecen como un solo nodo en el componente (es decir, se desactivó la casilla *Mostrar las hojas de cálculo por su nombre*). Se trata del nodo **`Hojas de cálculo`** situado bajo el nodo **`Libro`**.
- La función **`starts-with`** comprueba si la extensión telefónica (columna C) empieza por "1" y, si el resultado es `true`, entonces los registros se envían al componente de destino por medio de un componente de filtrado.
- El filtro **concat** añade el prefijo "10" a todas las extensiones telefónicas filtradas y escriba la nueva extensión telefónica en el nodo `PhoneExt`.
- MFCompany.xsd es el componente de destino y cuando se generen los resultados de la transformación incluirá los datos de las personas filtradas.

El resultado de la asignación de datos es un archivo XML que contiene los datos de cuatro personas.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xsi:schemaLocation="http://my-company.com/namespace
3    <Person>
4      <First>Steve</First>
5      <Last>Meier</Last>
6      <PhoneExt>10114</PhoneExt>
7    </Person>
8    <Person>
9      <First>Max</First>
10     <Last>Nafta</Last>
11     <PhoneExt>10122</PhoneExt>
12   </Person>
13   <Person>
14     <First>Carl</First>
15     <Last>Franken</Last>
16     <PhoneExt>10147</PhoneExt>
17   </Person>
18   <Person>
19     <First>Mark</First>
20     <Last>Redgreen</Last>
21     <PhoneExt>10152</PhoneExt>
22   </Person>
23 </Company>
```

La segunda asignación del archivo **Excel-mapping.mfd** es idéntica a la primera, con la única diferencia que las hojas de cálculo se habilitaron por separado con la opción *Mostrar las hojas de cálculo por su nombre*. Esta asignación se configuró de la siguiente manera:

- Las hojas de cálculo **`Admin`** y **`Development`** pueden verse bajo el nodo **`Libro`**.
- Solamente se definieron conectores entre la hoja de cálculo **`Admin`** y el componente de destino.



Excel-mapping.mfd (ejemplo nº2)

El resultado de la asignación de datos es un archivo XML que contiene los datos de dos personas solamente.

© 2018 Altova GmbH

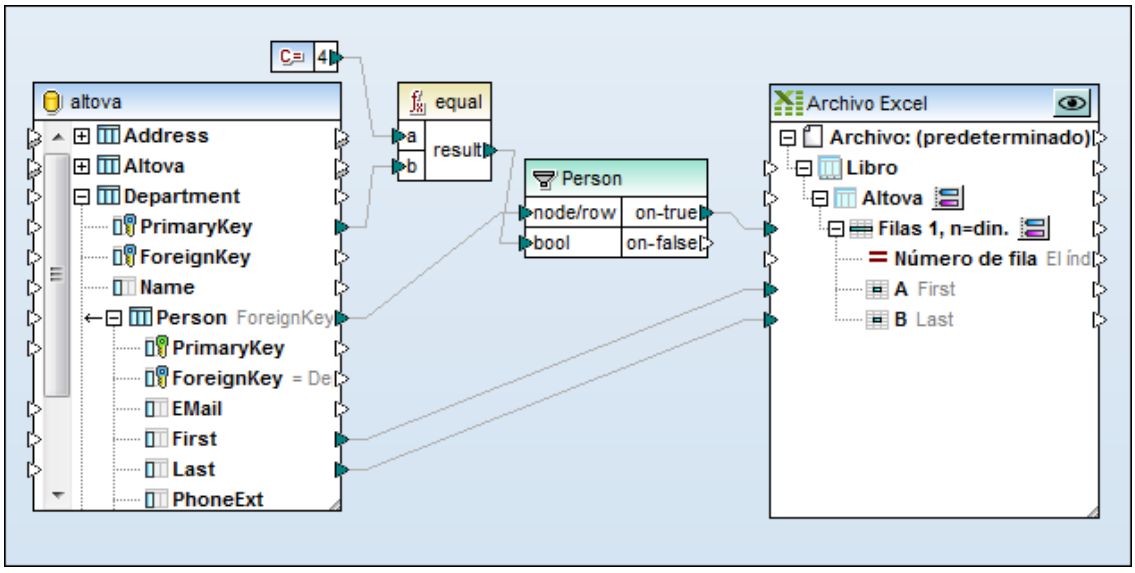
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xsi:schemaLocation="http://my-company.com/namespace
3    <Person>
4      <First>Steve</First>
5      <Last>Meier</Last>
6      <PhoneExt>10114</PhoneExt>
7    </Person>
8    <Person>
9      <First>Max</First>
10     <Last>Nafta</Last>
11     <PhoneExt>10122</PhoneExt>
12   </Person>
13 </Company>
```

7.7.9 Ejemplo: asignación de datos de BD a Excel 2007+

El archivo de asignación de este ejemplo está disponible en esta ruta de acceso: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Excel-mapping.mfd**. Cuando abra el archivo de ejemplo, observará que contiene tres transformaciones de asignación de datos distintas. En este apartado nos ocupamos de la tercera y última.

El objetivo de esta asignación de datos es:

- Extraer de la base de datos **altova** las personas cuyo departamento tiene una clave principal igual a 4 (es decir, las personas que trabajan en el departamento informático).
- Escribir los registros extraídos de la base de datos en un componente Excel 2007+ predeterminado.



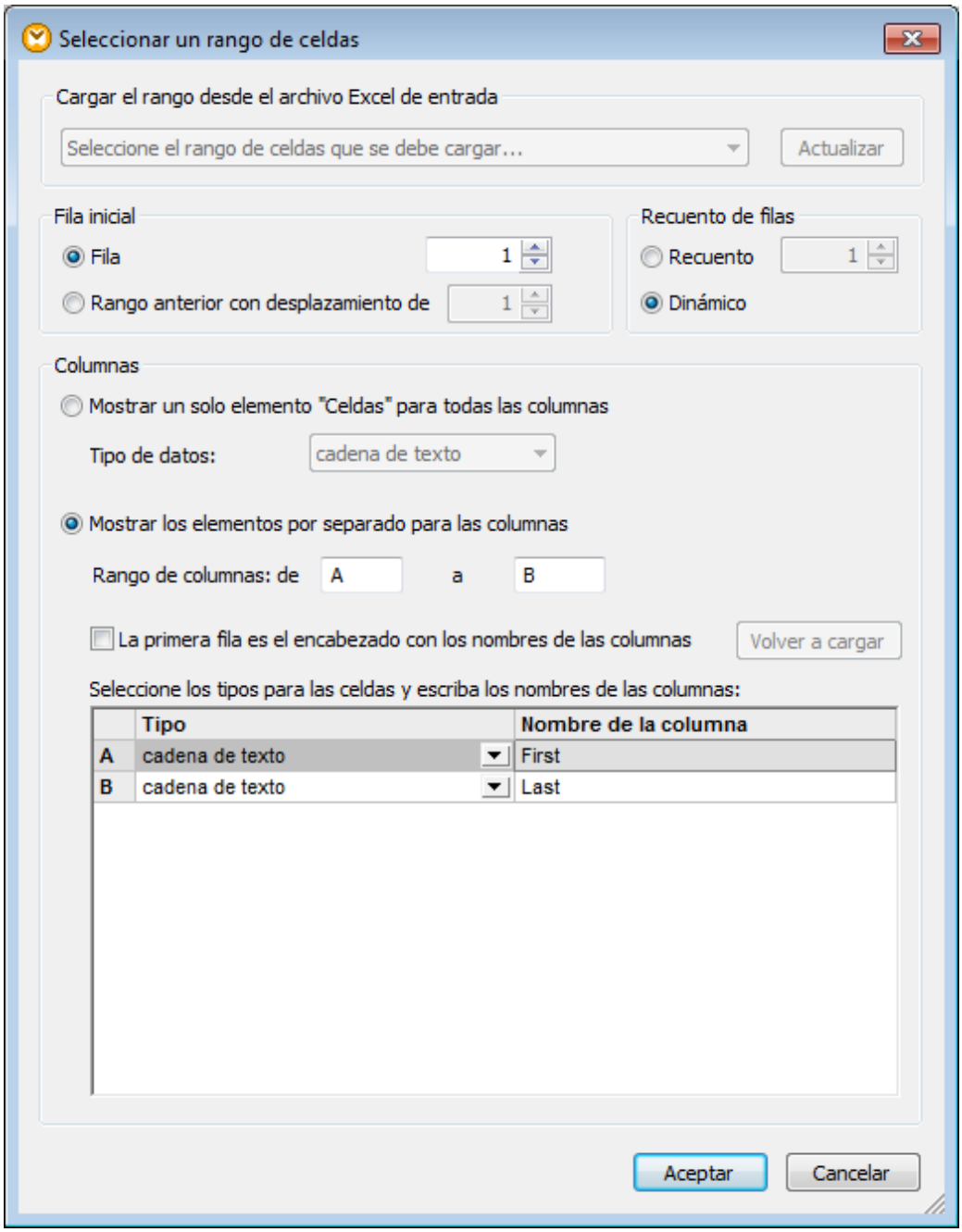
Excel-mapping.mfd (ejemplo nº3)

Esta asignación de datos se configuró de la siguiente manera:

1. La base de datos **altova** se añadió al área de asignación desde la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial** con ayuda del

comando de menú **Insertar | Base de datos** y del asistente para la conexión de bases de datos Microsoft Access.

2. El componente Excel 2007+ predeterminado se añadió al área de asignación con el comando de menú **Insertar | Archivo Excel 2007+** y pasando por alto la opción de especificar un archivo de muestra.
3. Se cambió el nombre de la primera hoja de cálculo (**`Hoja1`**) con ayuda del botón . El nuevo nombre de la hoja de cálculo es **`Altova`**.
4. Se configuró el rango **`Filas 1, n=din`** con ayuda del botón . Las opciones de configuración de este rango de celdas son:



5. Además se definieron estas opciones:

 © 2018 Altova GmbH

- Se usa la función **`equal`** para comparar el valor de la clave principal **PrimaryKey** con el valor "4", que viene dado por un componente de constante.
- El componente de filtrado pasa los campos **First** y **Last** al componente de destino si la condición booleana es `true` (es decir, si la clave principal del departamento es "4").
- El nodo **on-true** está conectado al nodo **`Filas 1, n=din`** del archivo Excel.

El resultado de la asignación de datos es un libro de Excel que contiene cuatro personas que trabajan en el departamento informático.



	A	B	C	D
1	Alex	Martin		
2	George	Hammer		
3	Jessica	Bander		
4	Lui	King		
5				
6				

Altova

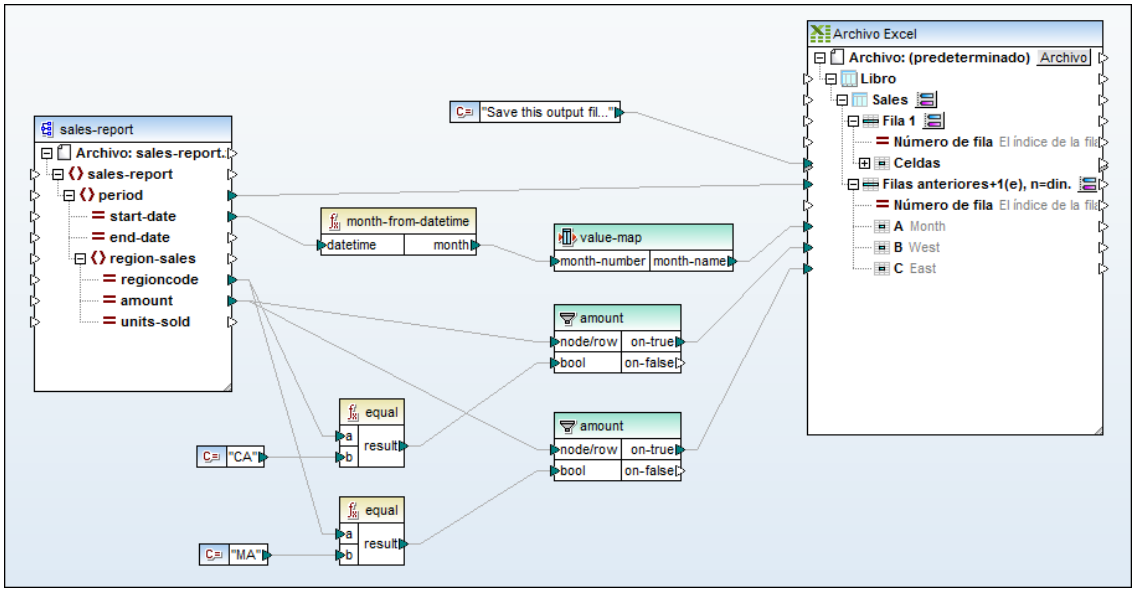
7.7.10 Ejemplo: suministrar datos a hojas Excel con formato previo

Las hojas de cálculo que genera MapForce no contienen formato, solamente datos. Si necesita aplicar formato a los datos Excel generados por MapForce, puede hacerlo fuera de MapForce con ayuda de funciones estándar de Excel:

1. Genere la hoja Excel con MapForce y guárdela en el disco (p. ej. con el nombre **xlsx-mapforce.xlsx**).
2. Cree una hoja Excel nueva con Excel (p. ej. con el nombre **Sales-presentation.xlsx**).
3. Desde **Sales-presentation.xlsx** cree referencias externas (enlaces) a los rangos de celdas de **xlsx-mapforce.xlsx**. Esta operación solo tiene que realizarse una vez (consulte las instrucciones https://support.office.com/es-es/article/Crear-una-referencia-externa-enlace-a-un-rango-de-celdas-de-otro-libro-c98d1803-dd75-4668-ac6a-d7cca2a9b95f?ui=es-ES&rs=es-ES&ad=ES).
4. Aplique el formato necesario (fuentes, colores, ,etc.) en **Sales-presentation.xlsx**.

A partir de ahora podrá generar datos tantas veces como sea necesario en la hoja **xlsx-mapforce.xslx**. Para actualizar **Sales-presentation.xlsx** con los datos más recientes generados por MapForce haga clic en el botón **Actualizar todo** de la pestaña *Datos*. El formato se mantendrá intacto.

En el ejemplo que aparece a continuación primero se generan algunos datos Excel con MapForce y se guardan en un archivo. Para ver estos datos con un formato atractivo se necesita otra hoja Excel (preconfigurada para tener referencias externas a esta primera hoja). Para continuar abra el diseño de asignación **<Documentos>\Altova\MapForce2018\MapForceExamples\Sales_to_Excel.mfd**.



Sales_to_Excel.mfd

Ahora haga abra el panel *Resultados* para generar el resultado de la asignación.

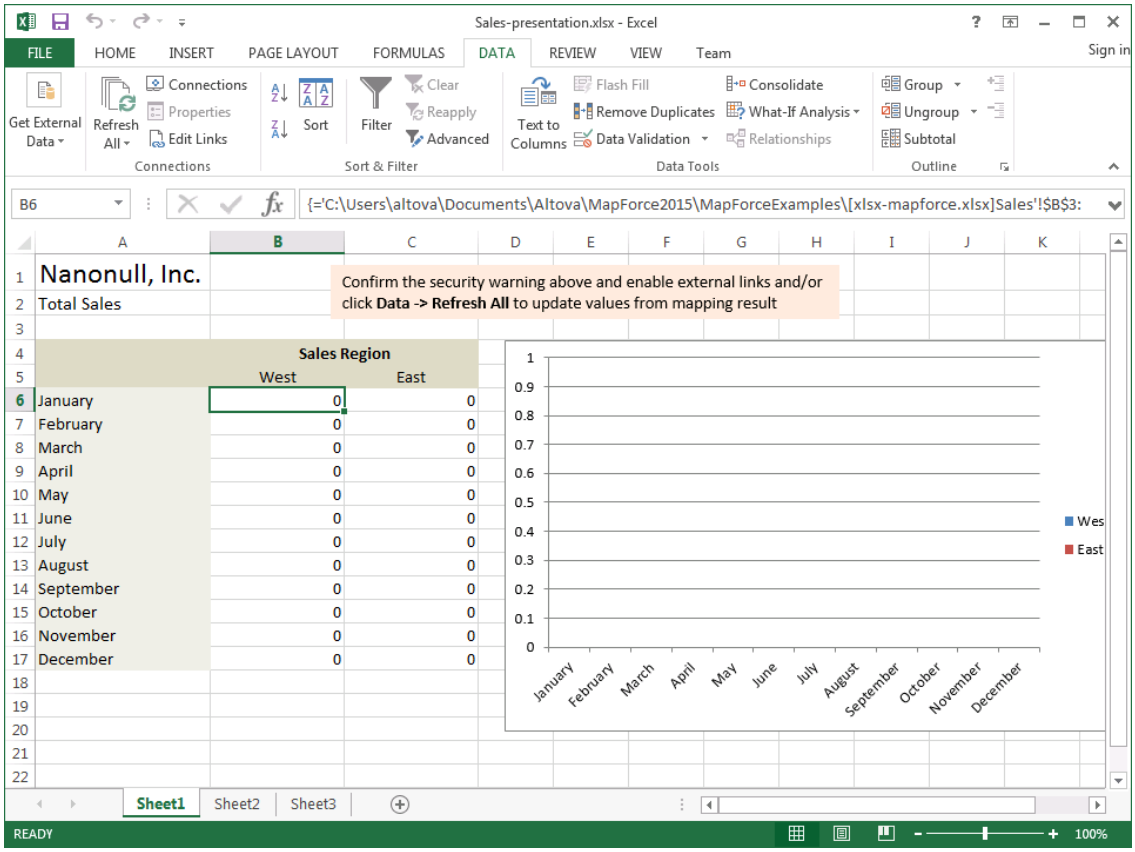
Después haga clic en el botón **Guardar resultado generado** de la barra de herramientas y guarde los resultados como "xlsx-mapforce.xlsx" en la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples**.

	A	B	C	D	E	F
1	Save this output file and open Sales-presentation.xlsx in Excel for a presentation view.					
2	Month	West	East			
3	Jan	110.4	75.3			
4	Feb	114.3	65.2			
5	Mar	134.2	86.1			
6	Apr	107.3	112.1			
7	May	114.4	93.8			
8	Jun	113.9	72.4			
9	Jul	89.4	67.4			
10	Aug	95.3	84.9			
11	Sep	107.2	99.5			
12	Oct	129.7	82.5			
13	Nov	137.1	101.9			
14	Dec	152.6	120.6			
15						

Ahora puede ver los mismos datos con formato en otra hoja de cálculo que está configurada para leer datos del archivo **xlsx-mapforce.xlsx**. Abra el archivo **Sales-presentation.xlsx** en Excel (desde la misma carpeta). Aparece una advertencia avisándole de que los enlaces externos están deshabilitados. Estos enlaces se tienen que habilitar para crear el enlace con los datos de

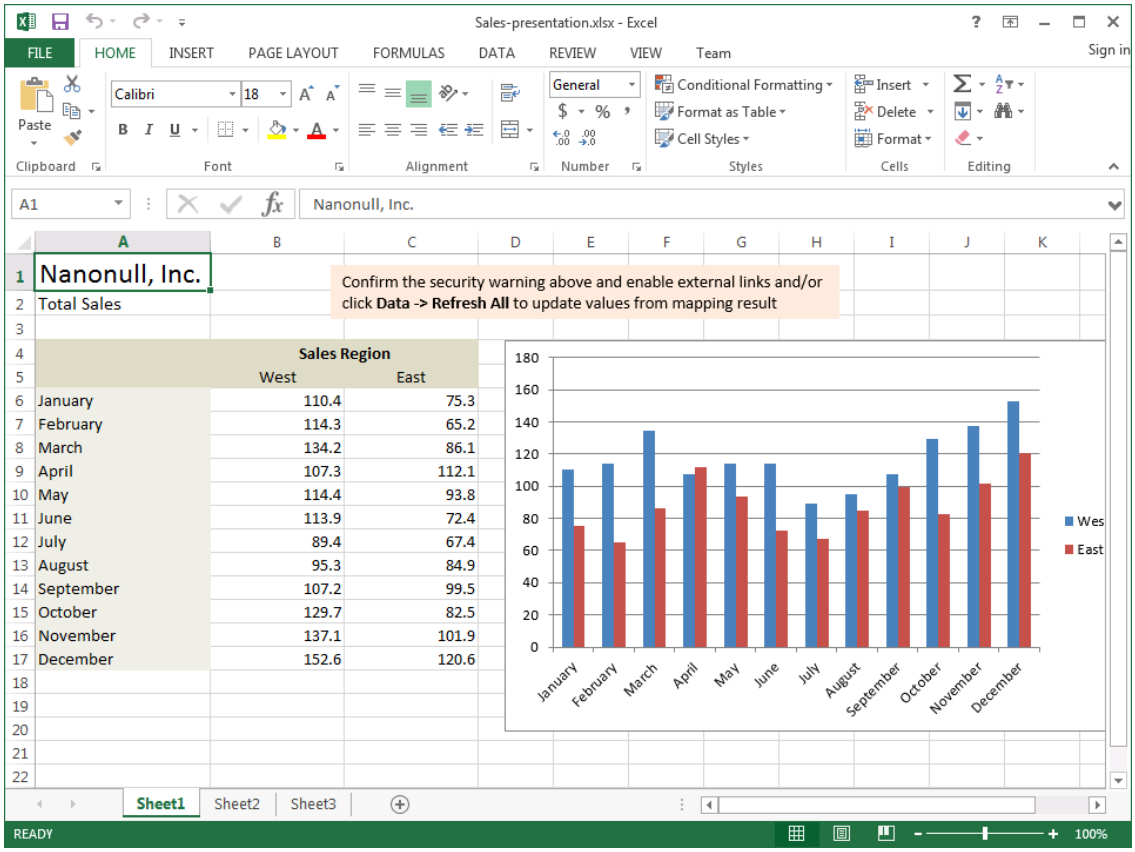
 © 2018 Altova GmbH

origen. La posición actual del cursor (B6) muestra que el rango de datos de origen es **Sales'!$B$3:$C$14** del archivo **xlsx-mapforce.xlsx**.



Sales-presentation.xlsx

Haga clic en el botón Habilitar (o Habilitar contenido en Excel 2016). Los enlaces externos se habilitan y los datos de origen subyacentes aparecen en forma de tabla y de gráfico.



Sales-presentation.xlsx (with updated references)

También puede actualizar a mano los enlaces que vinculan las dos hojas de cálculo:

- En la pestaña *Datos* haga clic en **Actualizar todo**.

© 2018 Altova GmbH

7.8 XBRL

El lenguaje ampliado para informes comerciales XBRL (Extensible Business Reporting Language) es un estándar abierto internacional que permite el intercambio de información comercial. XBRL es utilizado por gobiernos, instituciones financieras y empresas para la creación de informes financieros y para comunicar información reglamentaria. XBRL está gestionado por un consorcio global sin ánimo de lucro llamado XBRL International (https://www.xbrl.org).

Los requisitos de cada empresa u organización a la hora de trabajar con XBRL pueden ser muy distintos (p. ej. validación de datos XBRL, creación de datos XBRL, creación de taxonomías XBRL, exportación de datos XBRL a otros formatos, etc.). Altova ofrece varios productos diseñados para trabajar con XBRL:

- **Altova XMLSpy** edita o crea taxonomías nuevas y genera informes XBRL basados en archivos de transformación XSLT (estos archivos de transformación se pueden crear con Altova StyleVision).
- **Altova StyleVision** crea hojas de estilos/plantillas de taxonomías para generar informes XBRL.
- **Altova MapForce** crea asignaciones de datos entre archivos XBRL y otros formatos compatibles con MapForce (como Microsoft Excel 2007+, bases de datos o archivos CSV) y permite usar taxonomías XBRL a la hora de diseñar la estructura de la asignación de datos. Por ejemplo, puede filtrar datos concretos de documentos de instancia XBRL o crear instancias XBRL a partir de datos extraídos de una base de datos.
- **RaptorXML+XBRL Server** es un procesador XBRL multiplataforma para validar instancias y taxonomías XBRL.
- **Altova Work in Process (WIP) XBRL add-in for Excel** está destinado a empresas que deben presentar informes XBRL US Work In Process y permite ver y preparar datos de informes XBRL en Microsoft Excel directamente.

Para poder asignar datos de archivos de instancia XBRL en MapForce se necesita la taxonomía XBRL subyacente. La taxonomía está compuesta por un archivo de ficheros (por lo general un archivo ZIP) que aporta la organización o entidad a la que se deben presentar los datos XBRL. Por ejemplo, a continuación aparecen tres taxonomías que se pueden usar en asignaciones de datos de MapForce:

- US-GAAP (Principios contables generalmente aceptados), publicada por el consejo de normas de contabilidad financiera de EE UU (Financial Accounting Standards Board, http://www.fasb.org)
- NIIF (Norma Internacional de Información Financiera), publicada por la fundación IFRS (Norma Internacional de Información Financiera) Foundation (http://www.ifrs.org)
- COREP / FINREP (Taxonomías de intercambio de información financiera publicadas por la Autoridad Bancaria Europea), publicadas por la Autoridad Bancaria Europea (http://www.eba.europa.eu)

Siempre y cuando tenga los archivos de taxonomía del proveedor, podrá crear asignaciones de datos sin más requisitos. Sin embargo, debe tener en cuenta que MapForce solo viene con las versiones más recientes de la taxonomía US-GAAP y que las demás taxonomías tardarán un tiempo en cargarse y en validarse a no ser que se instalen localmente. Para instalar localmente más taxonomías (o versiones anteriores de las taxonomías) descargue y ejecute el programa de instalación *XBRL Taxonomies* de la página de descarga de componentes de Altova (http://www.altova.com/es/download_components.html).

MapForce es compatible con estos estándares XBRL:

- XBRL 2.1
- XBRL Dimensions 1.0
- XBRL Table Linkbase 1.0 (*véase Trabajar con tablas XBRL*)
- XBRL Taxonomy Packages 1.0 (*véase Paquetes de taxonomías XBRL*)
- Inline XBRL 1.0 y 1.1, incluido el estándar Transformation Registry 3 (*véase Leer datos de Inline XBRL*)

Normativa

En esta documentación se utiliza la terminología XBRL definida en estas especificaciones.

Especificación	Dirección URL
XBRL Specification 2.1	http://www.xbrl.org/Specification/XBRL-2.1/REC-2003-12-31/XBRL-2.1-REC-2003-12-31+corrected-errata-2013-02-20.html
Dimensions Specification 1.0	http://www.xbrl.org/specification/dimensions/rec-2012-01-25/dimensions-rec-2006-09-18+corrected-errata-2012-01-25-clean.html
Table Linkbase Specification 1.0	http://www.xbrl.org/Specification/table-linkbase/REC-2014-03-18/table-linkbase-REC-2014-03-18.html
XBRL Taxonomy Packages 1.0	https://www.xbrl.org/Specification/taxonomy-package/REC-2016-04-19/taxonomy-package-REC-2016-04-19.html
Inline XBRL	https://specifications.xbrl.org/spec-group-index-inline-xbrl.html

Además, cuando se hace referencia a un apartado o sección de una especificación, esta documentación utiliza el símbolo §. Por ejemplo, para referirnos a la sección 5.1 de la especificación XBRL Table Linkbase 1.0 usaremos:

Las tablas (especificación Table Linkbase 1.0, §5.1) se representan con el icono  *en el componente XBRL.*

7.8.1 Agregar archivos XBRL como componentes de la asignación

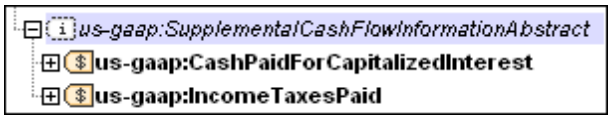
Para poder crear asignaciones de datos entre los datos de la asignación y un documento XBRL, debe insertar el componente XBRL en la asignación. Esto se puede hacer de dos maneras:

- Insertando el documento como documento XML plano
- Insertando el documento como documento XBRL con una estructura jerárquica

La primera opción inserta el archivo XBRL como un archivo XML plano, sin estructura jerárquica XBRL ni conceptos de lógica de negocio. Esta suele ser la mejor opción si se trata de una asignación muy simple en la que no se controlan dimensiones ni exige la generación automática de contextos XBRL.

La segunda opción inserta la taxonomía XBRL y muestra su contenido de forma jerárquica. En este caso, MapForce resuelve las referencias a la DTS (colección de taxonomías por descubrir) y genera automáticamente la estructura XBRL derivada. Esto significa que el componente XBRL de

la asignación incluirá componentes XBRL (que pueden ser nodos del archivo XBRL o nodos *abstractos* derivados de la DTS). Por ejemplo, la estructura jerárquica que aparece a continuación se deriva de los arcos de presentación `xlink:from` y `xlink:to` de la base de enlaces de presentación. (El término base de enlaces se usa aquí con el significado que tiene en la especificación Xlink). Más concretamente, representa una colección de enlaces (o una base de datos de vínculo). Las bases de enlaces Xlink se amplían con XBRL y ofrecen información adicional sobre los conceptos definidos en la taxonomía.



Este ejemplo solamente pretende ilustrar brevemente cómo resuelve MapForce la taxonomía para poder presentar los datos de una forma conceptual más adecuada para las operaciones de la asignación. MapForce también puede presentar estructuras derivadas de otras bases de enlaces, como bases de enlaces de tabla, por ejemplo (*véase Trabajar con tablas XBRL*).

Para más información sobre qué elementos pueden aparecer en un componente XBRL y ver su representación gráfica, consulte el apartado Información sobre componentes XBRL.

Para abrir documentos XBRL como XML plano

- Haga clic en **Insertar | Archivo o esquema XML**.

Para abrir documentos XBRL con control automático de contextos:

1. Haga clic en **Insertar | Documento XBRL**.
2. Seleccione el archivo de taxonomía (`*.xsd`) o archivo de instancia XBRL (`*.xbrl` o `*.xml`). Si selecciona una taxonomía, aparece otro diálogo donde debe seleccionar un archivo de instancia válido.
3. Haga clic en **Examinar...** si quiere usar este componente XBRL como instancia de origen y seleccione el archivo XBRL de instancia.

Al abrirse una taxonomía XBRL, aparece un aviso solicitando que seleccione las vistas de estructuras que deben aparecer en el componente XBRL. Si no tiene claro qué vista debe seleccionar, deje la opción predeterminada tal y como está: MapForce selecciona automáticamente una de las opciones analizando los datos de la DTS que está cargada. Las vistas de estructuras se pueden cambiar en cualquier momento (véase *Seleccionar vistas de estructuras*).

7.8.2 Leer datos de Inline XBRL

Algunas autoridades reguladoras que trabajan con XBRL ofrecen (o aceptan) información en formato Inline XBRL en lugar de o además de XBRL plano. Inline XBRL o iXBRL se usa para designar contenido XBRL electrónico que está incrustado en documentos HTML, lo cual permite presentar datos XBRL en formato legible en un explorador web. Consulte este enlace para ver una lista de especificaciones relacionadas con iXBRL: https://specifications.xbrl.org/spec-group-index-inline-xbrl.html.

En MapForce los archivos que contienen datos iXBRL (como archivos .htm, .html o .xhtml) pueden usarse como origen de datos en las asignaciones de datos. Es decir, MapForce puede leer datos de archivos de instancia Inline XBRL y asignar sus datos a cualquier formato de destino compatible. Recuerde que MapForce no puede escribir datos en instancias Inline XBRL de destino.

Requisitos

- El lenguaje de transformación de la asignación debe ser el motor integrado BUILT-IN (*véase* Seleccionar un lenguaje de transformación). Los demás lenguajes no son compatibles con iXBRL. Cuando se selecciona el motor de ejecución integrado BUILT-IN el resultado de las asignaciones de datos se puede consultar en MapForce directamente o el usuario puede compilar la asignación en un archivo de ejecución y ejecutar este archivo con MapForce Server (*véase* Compilar asignaciones en archivos de ejecución de MapForce Server).
- La taxonomía del archivo Inline XBRL puede ser una dirección URL o una referencia a un archivo local. Si se trata de una referencia a un archivo local y la asignación se ejecuta con MapForce Server, debe tener en cuenta estos requisitos adicionales:
  - El archivo de taxonomía debe estar en el mismo directorio que el archivo .mfx (el archivo de ejecución de la asignación de datos).
  - El archivo Inline XBRL debe apuntar al archivo de taxonomía por medio de una ruta de acceso relativa (p. ej. `<link:schemaRef xlink:type="simple" xlink:href="taxonomía.xsd" />`.
  - El cuadro de diálogo "Configuración del componente" (*véase* Configuración de componentes XBRL) también debe hacer referencia al archivo de taxonomía con una ruta de acceso relativa (p. ej. "taxonomía.xsd").
- El archivo Inline XBRL de origen debe contener una sola instancia XBRL. No se admiten varias instancias.

Agregar archivos de instancia Inline XBRL a la asignación

1. En el menú **Insertar** haga clic en **Documento XBRL**.
2. Ahora tiene dos opciones:
   a. si cuenta con la taxonomías XBRL, navegue hasta el archivo de taxonomía (que suele tener la extensión .xsd). Debe ser una taxonomía válida para el archivo de instancia Inline XBRL. También puede especificar una URL con el botón **Cambiar a URL** del cuadro de diálogo "Examinar" (*véase* Agregar componentes desde una URL).
   b. si no cuenta con el archivo de taxonomía XBRL ni con su URL, navegue hasta el archivo de instancia Inline XBRL. MapForce tratará de determinar y cargar la taxonomía XBRL correspondiente de forma automática.
3. Cuando la aplicación solicite un archivo de instancia, navegue hasta el archivo HTML que contiene los datos iXBRL (que suele tener la extensión .html, .htm o .xhtml).
4. Cuando la aplicación le pida que seleccione una vista estructurada, deje la opción predeterminada. Esto puede cambiarse más adelante (*véase* Seleccionar vistas de estructuras).
5. Haga clic en **Aceptar**.

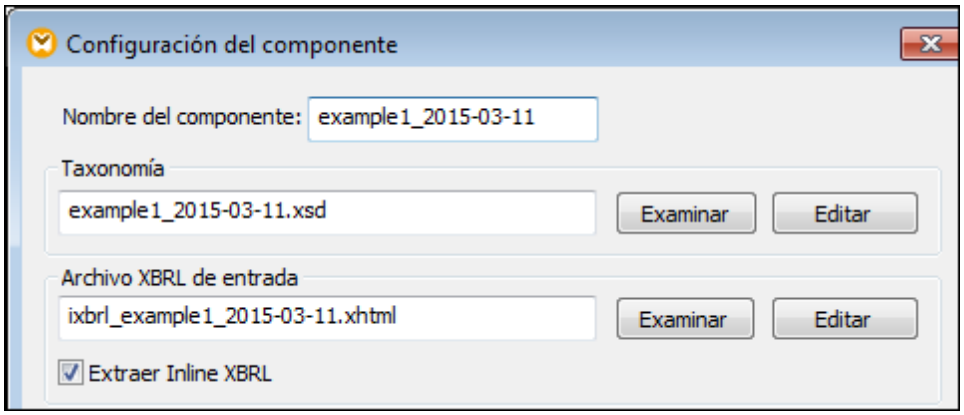
Llegados a este punto la estructura del contenido Inline XBRL aparece en el componente. Si se detectaran errores de validación, estos aparecerán en la ventana Mensajes. De lo contrario podrá

© 2018 Altova GmbH

dibujar conexiones entre el componente Inline XBRL y la estructura de destino (ya sea XML, texto o base de datos) igual que si se tratara de un componente XBRL estándar.

Casilla *Extraer Inline XBRL*

Cuando se activa la casilla *Extraer Inline XBRL* del cuadro de diálogo "Configuración del componente", el archivo .html o .xhtml especificado como entrada se tratará como archivo Inline XBRL. MapForce extraerá el contenido Inline XBRL del archivo de entrada y lo validará cuando se cierre el cuadro de diálogo. Esta casilla suele estar seleccionada automáticamente cuando se añade una instancia Inline XBRL a la asignación. Para comprobar si está activada haga doble clic en el título del componente XBRL (o haga clic con el botón derecho en el componente y seleccione **Propiedades**).



7.8.3 Paquetes de taxonomías XBRL

Un paquete de taxonomías XBRL es un archivo ZIP que contiene una copia desconectada de una taxonomía XBRL. La ventaja de usar un paquete de taxonomías XBRL (siempre y cuando el autor de la taxonomía lo ofrezca) es que los recursos XBRL, como los esquemas, por ejemplo, se resuelven desde archivos de taxonomía locales (desconectados) en lugar de recuperarse de Internet. La estructura y el formato de los paquetes de taxonomías XBRL se dispone en la recomendación Taxonomy Packages 1.0 (https://www.xbrl.org/Specification/taxonomy-package/REC-2016-04-19/taxonomy-package-REC-2016-04-19.html).

Si el autor de la taxonomía XBRL permite descargar paquetes de taxonomías XBRL que se ajusten a la recomendación recién descrita, podrá usar esos paquetes con MapForce, que se usarán en las funciones de validación y procesamiento XBRL (determinadas direcciones URL, como las referencias de esquema, se redirigirán a los archivos locales que incluyen los paquetes de taxonomías). Además, los paquetes de taxonomías permiten ver todos los puntos de entrada de la taxonomía XBRL en MapForce y mejorar el rendimiento de la validación XBRL.

Nota: las herramientas de escritorio de Altova compatibles con paquetes de taxonomías XBRL son XMLSpy, MapForce y StyleVision, que comparten la lista de paquetes de taxonomías XBRL. Es decir, si edita la lista en una aplicación la lista se modificará automáticamente en las demás aplicaciones.

Funcionamiento

Cada paquete de taxonomías XBRL contiene un archivo de catálogo interno (**catalog.xml**), que define los identificadores URI que se deben redirigir (reasignar) a recursos locales. Los archivos de catálogo cumplen con lo dispuesto por un subconjunto restringido de la especificación XML Catalog (https://www.oasis-open.org/committees/download.php/14809/xml-catalogs.html). En MapForce existe un mecanismo similar para reasignar identificadores URI a recursos locales (*véase Archivos de catálogo*).

En cuanto se añaden en MapForce (instrucciones más abajo), los paquetes de taxonomías XBRL empiezan a influir en el mecanismo de MapForce para resolver recursos para la validación XBRL. Por ejemplo, si añade una instancia XBRL a la asignación, su esquema se cargará desde el paquete de taxonomías local, en lugar de recuperarse de Internet (si es que el paquete de taxonomías está activo y su archivo **catalog.xml** reasigna el URI del esquema a un recurso local). En otras palabras, esto afectará a todos los procesos de validación y procesamiento XBRL posteriores.

Advertencia: los catálogos de los paquetes pueden redirigir a recursos incompatibles
Los recursos a los que apunte el archivo de catálogo de un paquete activo se usarán para todas las operaciones de MapForce que requieran dicho recurso. Por ejemplo, el recurso puede ser un esquema XML que se usa para la validación XML y para la validación XBRL. Si el recurso desconectado incluido en el archivo de catálogo del paquete es incompatible con el entorno actual, podrían darse errores. Cuando esto ocurra, desactive el paquete de taxonomías y envíe la información del error al autor del paquete.

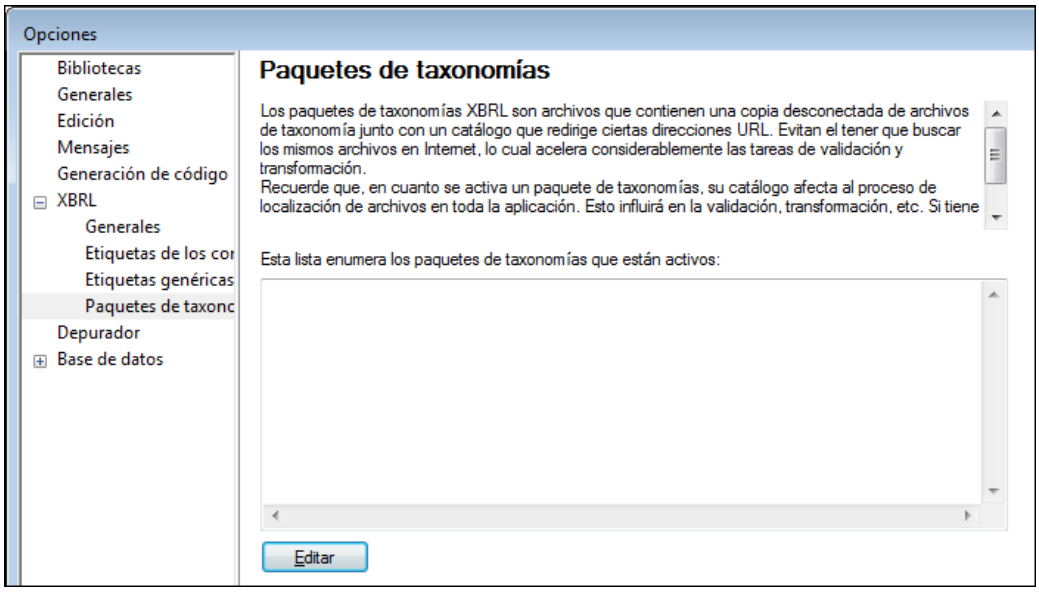
Agregar un paquete de taxonomías

1. Primero descargue el paquete de taxonomías XBRL (archivo .zip) y guárdelo en un directorio local que no suela cambiar mucho. No descomprima el archivo.

Nota: el archivo .zip debe cumplir con lo dispuesto por la recomendación Taxonomy Packages 1.0 (https://www.xbrl.org/Specification/taxonomy-package/REC-2016-04-19/taxonomy-package-REC-2016-04-19.html).

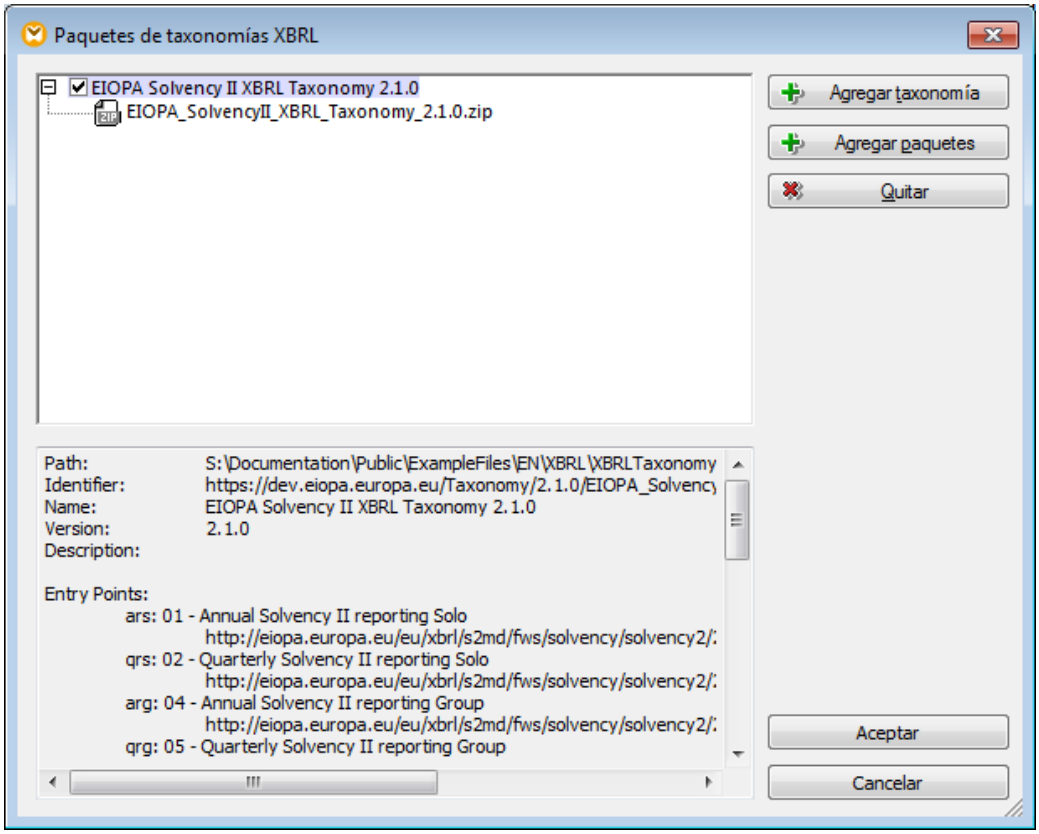
2. En el menú **Herramientas** haga clic en **Opciones | XBRL | Paquetes de taxonomías**.

© 2018 Altova GmbH



3. Haga clic en el botón **Editar**.
4. Ahora haga clic en **Agregar taxonomía** y navegue hasta el paquete de taxonomías en formato .zip. Para seleccionar varios paquetes mantenga pulsada la tecla **Mayús** mientras selecciona los paquetes.

La lista de paquetes de taxonomías XBRL recién añadidos aparece en el cuadro de diálogo en forma de estructura jerárquica compuesta por dos niveles. El primer nivel representa la taxonomía y el segundo los paquetes que la componen. Cuando se selecciona un paquete de taxonomías en el cuadro de diálogo, sus datos aparecen en el panel inferior (incluso su ubicación desconectada).



La casilla situada a la izquierda de una entrada de taxonomía indica si la taxonomía está activa o no. Las taxonomías recién añadidas se activan por defecto. Para desactivar una taxonomía basta con desactivar su casilla (p. ej. a veces es necesario desactivar una taxonomía si el paquete da problemas o si necesita alternar entre dos versiones de una misma taxonomías).

Cuando quiera agregar un paquete más a una taxonomía actual, seleccione la entrada de la taxonomía y después haga clic en **Agregar paquetes**. El paquete aparecerá en el segundo nivel de la taxonomía. Para eliminar un paquete selecciónelo y haga clic en **Eliminar** (esto no elimina los archivos del disco)

7.8.4 Información sobre componentes XBRL

Esta tabla muestra los elementos (nodos) que suelen formar parte de los componentes XBRL en MapForce. Tenga en cuenta que algunos nodos solamente están disponibles si selecciona determinado tipo de vista (véase Seleccionar vistas de estructuras). Por ejemplo, los nodos propios de la base de enlaces de tabla no se pueden ver si elige presentar solamente los datos XBRL de las bases de enlaces de presentación y de definición.

Elemento	Icono	Descripción
Elemento abstracto		Los elementos abstractos se usan para organizar *hechos* relacionados. Los elementos abstractos se

 © 2018 Altova GmbH

Elemento	Icono	Descripción
		definen dentro de la base de enlaces de presentación o de las redes de miembro de dominio de las bases de enlaces de definición. Estos elementos no existen realmente en el archivo XBRL de instancia, pero sirven para agrupar hechos de forma intuitiva.
Desglose		Ver la especificación Table Linkbase 1.0, §5.4.
Contexto		Un nodo de contexto es un contenedor para todos los *hechos* comerciales relacionados. Los nodos de contexto situados dentro de un hipercubo se ocupan automáticamente de sus elementos de contexto y de sus dimensiones. El nodo **`xbrli:context`** es un secundario del elemento raíz XBRL y se utiliza para el control manual de las dimensiones.
Dimensión		Las dimensiones XBRL (especificación XBRL Dimensions 1.0, §2.5) se usan para estructurar la información contextual para *hechos* comerciales. En la taxonomía las dimensiones se definen dentro de un hipercubo.
Miembro explícito		Un miembro de una dimensión que está definido por una enumeración de valores `QName` (*véase* *Mostrar dimensiones en un componente*).
Valor de miembro explícito		El valor de un miembro explícito.
Hecho		Los hechos (especificación XBRL 2.1, §1.4) son los valores de los elementos XBRL. Pueden ser de varios tipos: • monetarios () • de cadena () • numéricos () • generales ( ) • de tipo shares ()
Nota al pie		Las notas al pie permiten asignar información adicional a los hechos.
Hipercubo		Los hipercubos (especificación XBRL Dimensions 1.0, §2.2) usan información de las bases de enlaces de definición y de presentación para dar una estructura jerárquica a las dimensiones, a los contextos y a otros conceptos XBRL relacionados. Consulte Trabajar con hipercubos XBRL para obtener más información.
Elemento raíz XBRL		El elemento **`xbrli:xbrl`** es el elemento raíz de instancia de todos los componentes XBRL.
Nodo de regla		Ver la especificación Table Linkbase 1.0, §6.6.

Elemento	Icono	Descripción
Nodo estructural		Ver la especificación Table Linkbase 1.0, §5.5.
Tabla		Ver la especificación Table Linkbase 1.0, §5.1. MapForce muestra la estructura de las tablas en una vista jerárquica (véase *Trabajar con tablas XBRL*).
Conjunto de tablas		Un conjunto de tablas (especificación Table Linkbase 1.0, §5.2) es un grupo de varias tablas.
Tupla		Las tuplas (especificación XBRL 2.1, §1.4) son elementos complejos cuyos miembros son hechos y otras tuplas.
Miembro con tipo		Miembros definidos por un elemento de esquema XML (véase http://www.xbrl.org/specification/dimensions/rec-2012-01-25/dimensions-rec-2006-09-18+corrected-errata-2012-01-25-clean.html#sec-typed-dimensions).
Unidad		El elemento xbrli:unit contiene las unidades a las que se refieren los elementos XBRL. Cuando se realizan asignaciones de datos es obligatorio definir un valor para el elemento **xbrl:unit**. Por ejemplo, si se trata de dólares `UnitID = usd` y `Measure = iso4217:USD`. Consulte Trabajar con valores XBRL predeterminados para obtener más información.
Vista		Las vistas representan funciones de vínculo extendido de las bases de enlaces de presentación y de definición de una taxonomía XBRL.

Recuerde que el icono es genérico y se usa con varios tipos de nodos diferentes, incluido el elemento XBRL raíz.

En los componentes XBRL también pueden aparecer estos otros elementos gráficos si así lo requiere el contexto:

Elemento gráfico	Icono	Descripción
Información adicional		Este icono aparece junto a los elementos que tienen información asociada (p. ej. un mensaje de error). Haga clic en el icono para ver la información adicional en la ventana Mensajes.
Error		Este icono indica que hay un mensaje de error.
Mostrar menú contextual		Este icono aparece junto a los elementos que disponen de un menú contextual con opciones propias.
Se desconoce		Este icono aparece junto a los elementos que MapForce no puede resolver debido a referencias de espacio de nombres no válidas o cuando el elemento al que se hace referencia no existe en la taxonomía.

© 2018 Altova GmbH

7.8.5 Seleccionar vistas de estructuras

MapForce le ofrece la opción de seleccionar qué vistas de estructuras XBRL aparecen en el componente. Esto se puede hacer cuando se añade el documento XBRL en el área de asignación (véase Agregar archivos XBRL como componentes de la asignación) o más tarde. Estas son las vistas de estructuras que puede elegir:

Vista de estructura	Descripción
Tablas de la base de enlaces de tabla	Muestra las tablas definidas en la base de enlaces de tabla de la taxonomía. Esta opción se deshabilita si la taxonomía no contiene definiciones de tabla. Esta vista es condicional. Para continuar seleccione esta casilla o la casilla *Vistas de las bases de enlaces de presentación y definición*.
Vistas de las bases de enlaces de presentación y definición	Muestra los datos de las bases de enlaces de presentación y definición. La base de enlaces de presentación incluye jerarquías y la de definición incluye funciones de vínculo extendido con hipercubos y dimensiones. Esta vista es condicional. Para continuar seleccione esta casilla o la casilla *Tablas de la base de enlaces de tabla*.
Todos los conceptos (con gestión de contexto)	Muestra el nodo `Todos los conceptos` en la estructura jerárquica. Este nodo contiene el hipercubo `Sin dimensión` que permite crear asignaciones entre todos los conceptos de la taxonomía, independientemente de si se incluyen dentro de los hipercubos por medio de las dos dimensiones predeterminadas **`identifier`** y **`period`**. El control automático de contextos viene dado por el nodo **`context`**, cuyos elementos secundarios son todos los conceptos XBRL de la taxonomía. Los elementos abstractos no aparecen en esta vista. Esta vista es opcional.
Todos los conceptos (sin procesar)	Muestra el nodo `Todos los conceptos (sin procesar)` que ofrece acceso a todos los hechos de la instancia XBRL sin control automático de contextos ni control de dimensiones. Presenta la estructura XML sin procesar de los conceptos del archivo de instancia. El atributo `contextRef` debe asignarse a mano. Esta vista es opcional.

Notes

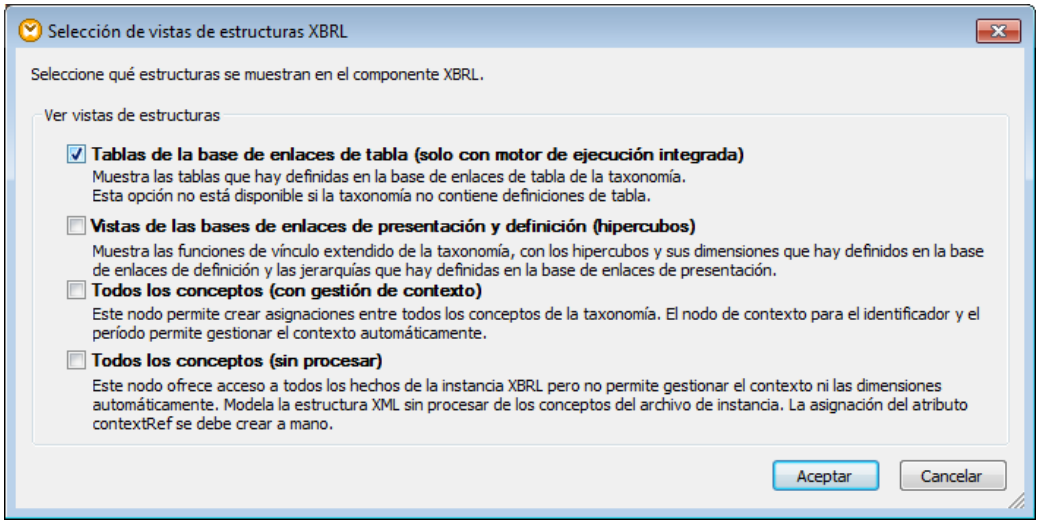
- A la hora de leer datos de un componente XBRL (es decir, si el componente XBRL es el componente de origen), puede elegir varias vistas de estructuras.
- A la hora de escribir datos en un componente XBRL (es decir, si el componente

XBRL es el componente de destino, la vista **Tablas de la base de enlaces de tabla** no se puede usar junto con ninguna otra vista.

- La vista **Tablas de la base de enlaces de tabla** exige usar el motor de transformación integrado BUILT-IN (véase Seleccionar un lenguaje de transformación).

Para seleccionar las vistas de estructuras que deben aparecer en el componente XBRL

1. En el elemento raíz del componente XBRL haga clic en el botón **Mostrar menú contextual** y seleccione el comando **Seleccionar vistas de estructuras...** en el menú contextual.



2. Ahora seleccione qué vistas deben aparecer en el componente XBRL y haga clic en **Aceptar**.

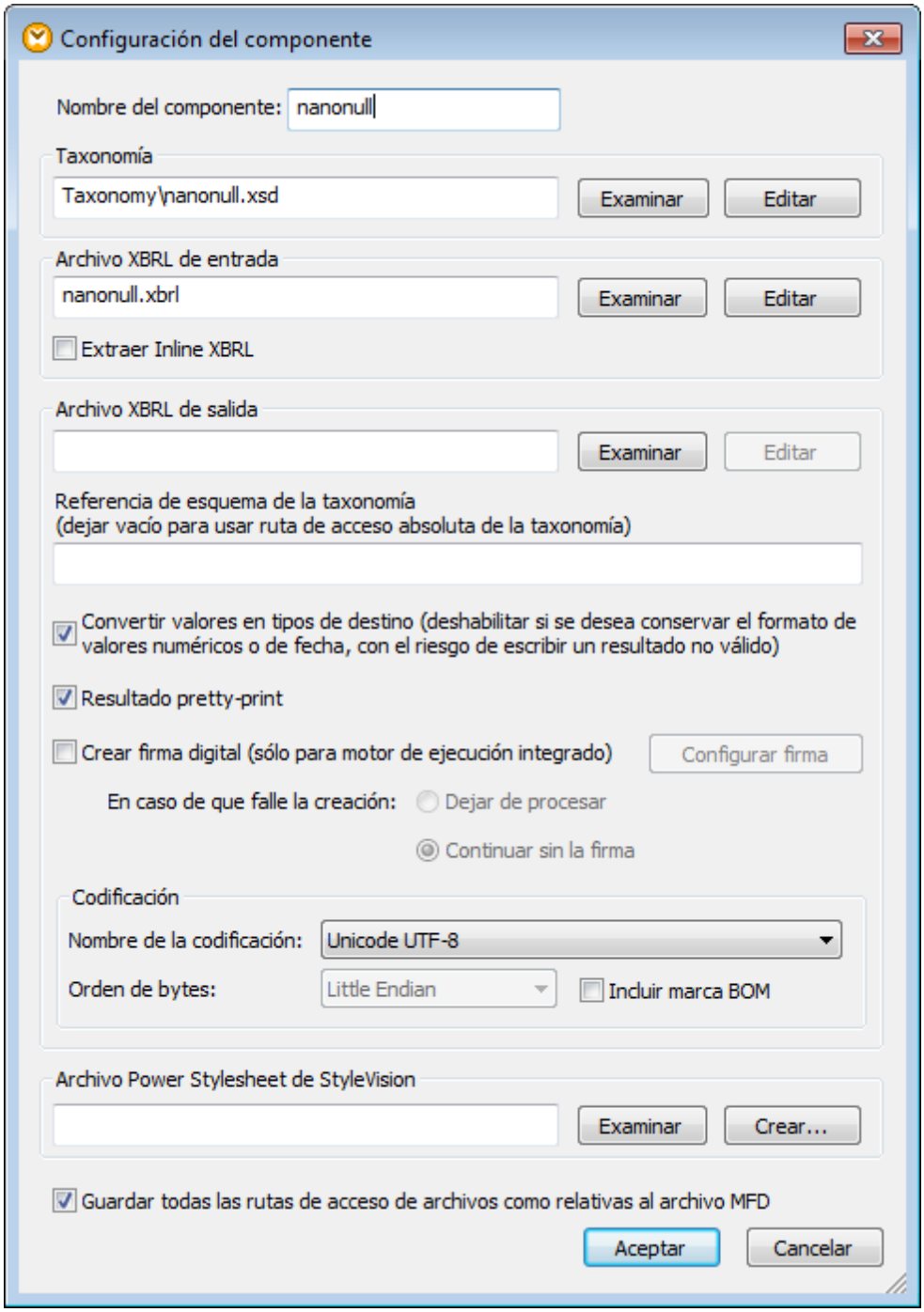
7.8.6 Configuración de componentes XBRL

Tras añadir el componente XBRL al área de asignación (véase Agregar archivos XBRL como componentes de la asignación), podrá elegir la configuración del componente en el cuadro de diálogo "Configuración del componente".

Para abrir el cuadro de diálogo "Configuración del componente"

- Haga doble clic en la barra de título del componente XBRL.

© 2018 Altova GmbH



Estas son las opciones que ofrece este cuadro de diálogo:

Taxonomía	Indica el nombre de archivo y la ruta de acceso del archivo de taxonomía principal. Haga clic en **Examinar** para seleccionar un archivo. Haga clic en **Editar** para editar el archivo en un editor XBRL

	(p. ej. XMLSpy).
Archivo XBRL de entrada	Indica el nombre de archivo de la instancia XBRL de entrada del componente XBRL seleccionado. Este campo se rellena automáticamente cuando se inserta el componente XBRL y se le asigna un archivo XBRL de instancia. Haga clic en **Examinar** para seleccionar otro archivo. Haga clic en **Editar** para editar el archivo en un editor XBRL (p. ej. XMLSpy).
Archivo XBRL de salida	Indica el nombre de archivo y la ruta de acceso de la instancia XBRL de destino si el componente se utiliza como destino de la asignación. El contenido del campo *Archivo XBRL de entrada* se copia automáticamente en este campo cuando se asigna el archivo XBRL de instancia. Si no asigna ninguna instancia XBRL de entrada al componente, entonces este campo contiene el nombre y la ruta de acceso de la taxonomía (con la extensión `.xml`). Haga clic en **Examinar** para seleccionar otro archivo. Haga clic en **Editar** para editar el archivo en un editor XBRL (p. ej. XMLSpy).
Extraer Inline XBRL	Si marca esta casilla, el contenido Inline XBRL se extraerá del archivo .html, .htm o .xhtml especificado como archivo de entrada.
Referencia de esquema de la taxonomía	La ruta de acceso del esquema asociado y correspondiente al archivo MFD. Use este campo si quiere especificar una taxonomía diferente para la validación. La referencia a la taxonomía se escribe en el atributo `href` del elemento `link:schemaRef`. Por ejemplo: `<link:schemaRef xlink:type="simple" xlink:href="..\..\nanonull.xsd"/>`
Convertir valores en tipos de destino	Indica si los tipos del esquema XML de destino se deben usar en la asignación y si todos los datos asignados al componente de destino se deben tratar como valores de tipo `string`. Esta opción está activada por defecto.
Resultado pretty-print	Adapta el formato del documento XBRL de salida para darle un aspecto jerárquico. Cada nodo secundario tiene un desplazamiento de una tabulación con respecto al nodo primario.

© 2018 Altova GmbH

Crear firma digital	Sirve para añadir una firma digital al archivo de instancia XBRL de salida. Puede agregar una firma digital si está seleccionado el lenguaje de transformación integrado BUILT-IN (*véase Firmas digitales*).
Codificación	Sirve para indicar la codificación del archivo de instancia de salida: • Nombre de la codificación • Orden de bytes • Si se debe incluir la marca BOM o no
Archivo Power Stylesheet de StyleVision	Sirve para seleccionar o crear una hoja de estilos de Altova StyleVision. Estos archivos sirven para presentar los datos del archivo XBRL de instancia en varios formatos (HTML, PDF, etc.).
Guardar todas las rutas de acceso de archivos como relativas al archivo MFD	Sirve para guardar las rutas de acceso de los archivos que aparecen en el cuadro de diálogo como relativas a la ubicación del archivo de diseño de MapForce (`.mfd`). Esta opción afecta a estos archivos: • El archivo de taxonomía XBRL • El archivo XBRL de entrada • El archivo XBRL de salida • La hoja de estilos de StyleVision Consulte el apartado Usar rutas de acceso relativas en un componente para obtener más información.

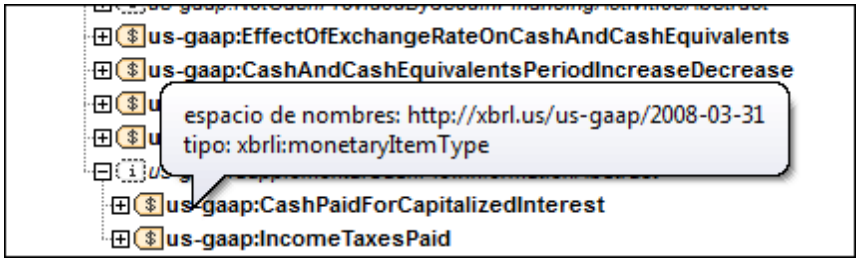
7.8.7 Establecer preferencias para XBRL

En MapForce puede definir varias preferencias a la hora de trabajar con XBRL:

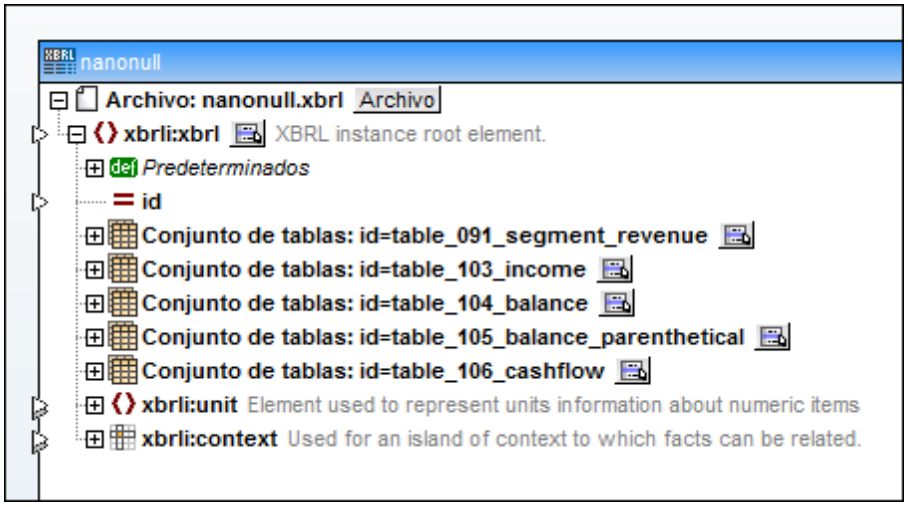
- Puede ver información adicional sobre los elementos de los componentes (*véase Habilitar información rápida y anotaciones*)
- Puede configurar opciones XBRL generales para toda la aplicación (*véase Opciones XBRL generales*).

7.8.7.1 Habilitar información rápida y anotaciones

Si habilita la información rápida, al pasar el puntero por los elementos del componente aparece información adicional sobre el elemento (*imagen siguiente*).



Además puede mostrar u ocultar el texto de anotación (si existe). Si las habilita, las anotaciones aparecen en la parte derecha del componente. Por ejemplo, en la imagen siguiente la anotación del elemento raíz `xbrli:xbrl` es "*XBRL instance root element*".



Para habilitar o deshabilitar la información rápida tiene dos opciones:

- hacer clic en **Vista | Mostrar información rápida** o
- hacer clic en el botón **Mostrar información rápida** de la barra de herramientas.

Para habilitar o deshabilitar las anotaciones también tiene dos opciones:

- hacer clic en **Vista | Mostrar anotaciones** o
- hacer clic en el botón **Mostrar anotaciones** de la barra de herramientas.

7.8.7.2 Opciones XBRL generales

En MapForce puede configurar estas opciones XBRL generales para toda la aplicación:

- El idioma de etiqueta de los elementos XBRL y de sus anotaciones.
- Las funciones de etiqueta preferidas para el nombre de los elementos XBRL.
- El tipo concreto de funciones de etiqueta para las anotaciones de los elementos XBRL.
- Paquetes de taxonomías XBRL personales (*véase* *Paquetes de taxonomías XBRL*).

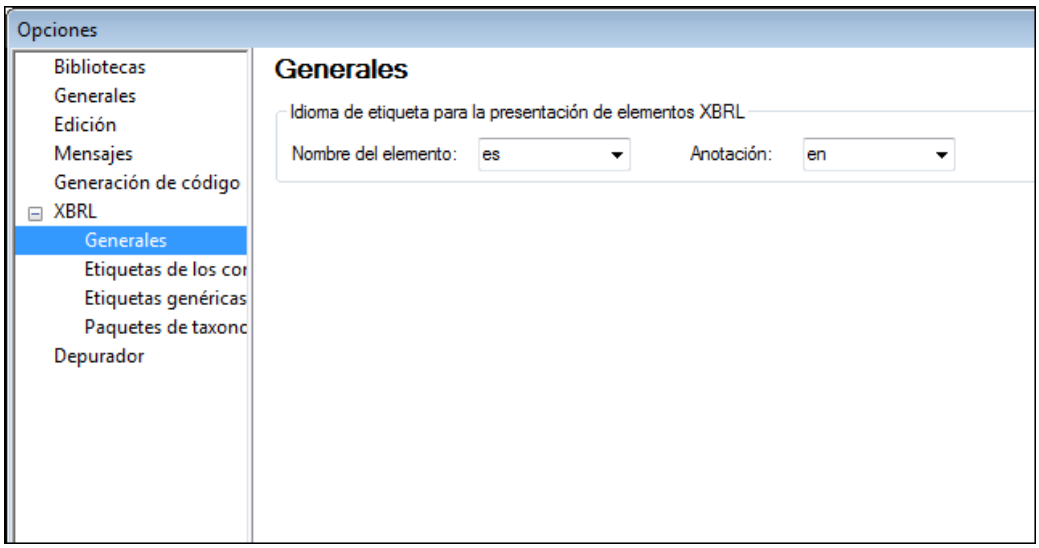
Todas estas opciones se pueden configurar en el cuadro de diálogo "Opciones", en la pestaña *XBRL*.

Para cambiar las opciones XBRL generales

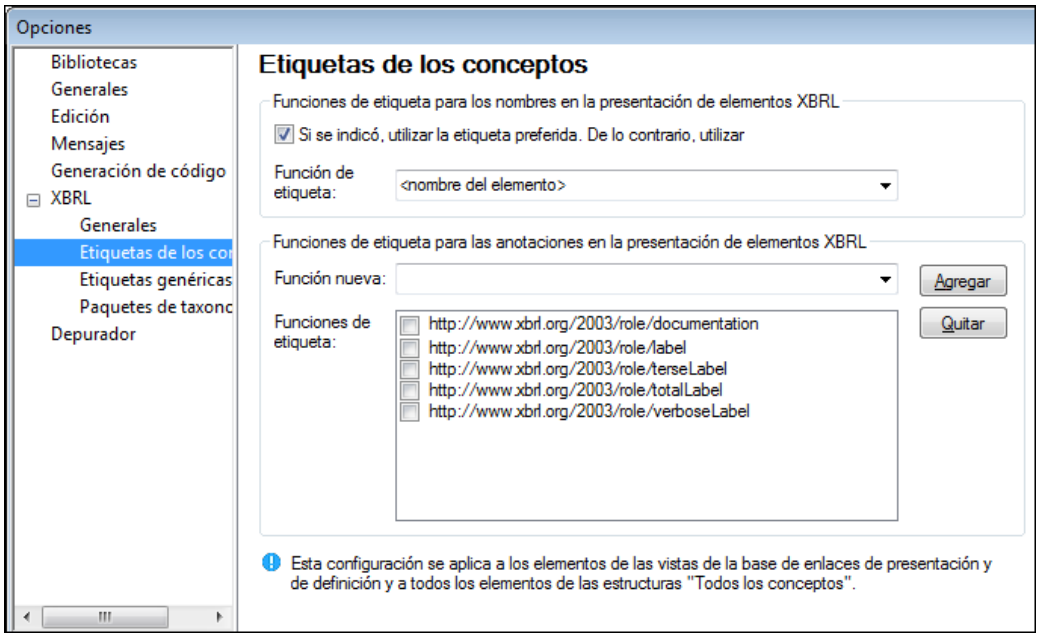
1. Seleccione el comando **Vista | Opciones de visualización XBRL**.
2. Ahora tiene varias opciones:

© 2018 Altova GmbH

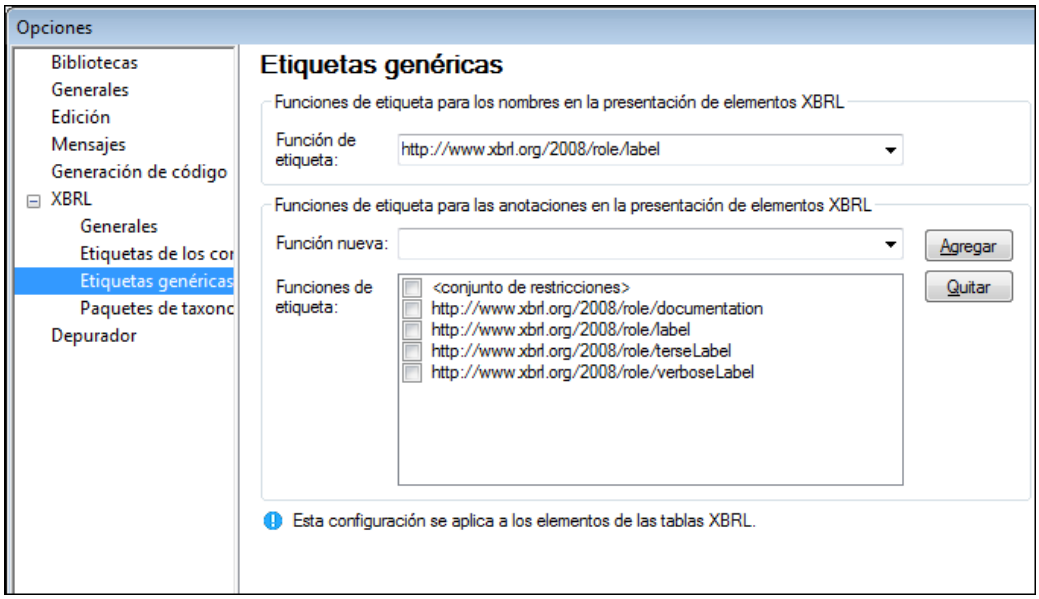
- Para cambiar el idioma de etiqueta de los elementos XBRL y de sus anotaciones haga clic en **XBRL |Generales**:



- Para cambiar la función de etiqueta preferida y las funciones de etiqueta para las anotaciones de los elementos de estas vistas haga clic en **XBRL | Etiquetas de los conceptos**:
  - *Vistas de las bases de enlaces de presentación y definición*
  - *Todos los conceptos (con gestión del contexto)*
  - *Todos los conceptos (sin procesar)*



- Para cambiar la función de etiqueta preferida y las funciones de etiqueta para las anotaciones de los elementos de la vista *Tablas de la base de enlaces de tabla* haga clic en **XBRL | Etiquetas genéricas**.

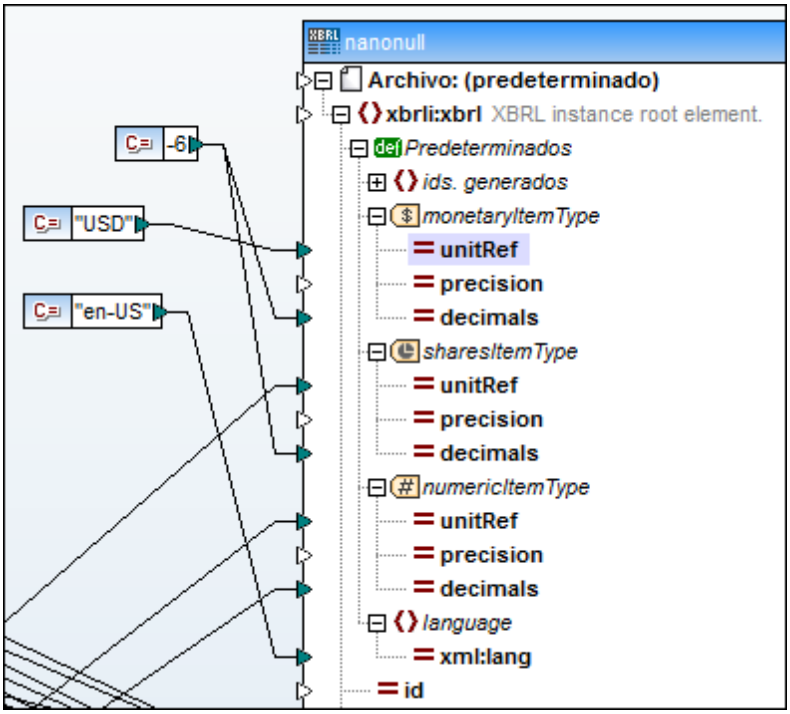


- Para agregar o cambiar paquetes de taxonomías XBRL haga clic en **XBRL | Paquetes de taxonomías** (véase Paquetes de taxonomías XBRL).

7.8.8 Trabajar con valores XBRL predeterminados

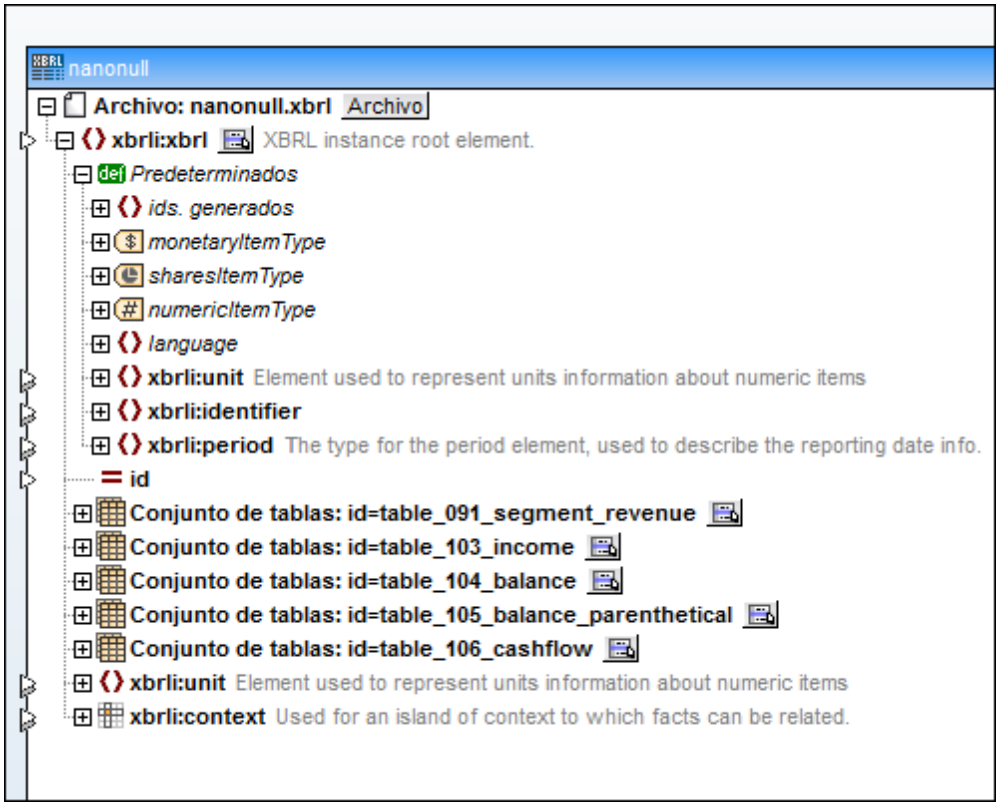
En MapForce el concepto de *unidades/valores predeterminados* permiten asignar valores a los atributos del componente XBRL sin necesidad de realizar conexiones explícitas entre todos ellos. Por ejemplo, si asigna un identificador de unidad constante al atributo `unitRef` del elemento `monetaryItemType` de la estructura *`Predeterminados`* (*imagen siguiente*), se asignará el valor predeterminado a todos los identificadores de unidad de tipo monetario, excepto cuando su entrada esté asignada explícitamente a otro valor. En la imagen también puede observar el uso del valor predeterminado del atributo `xml:lang`, que define el idioma de una nota al pie.

© 2018 Altova GmbH



El nodo *Predeterminados* aparece por defecto cuando se crea un componente XBRL nuevo. Sin embargo, puede ocultarlo si no necesita asignar valores a los nodos de *Predeterminados*. El nodo *Predeterminados* puede habilitarse en el nivel raíz (para todo el documento XBRL) o en los diferentes nodos de cualquier nivel jerárquico del componente XBRL. Recuerde que como se pueden definir valores predeterminados en cualquier nivel de la estructura XBRL, cada subestructura puede tener valores predeterminados distintos.

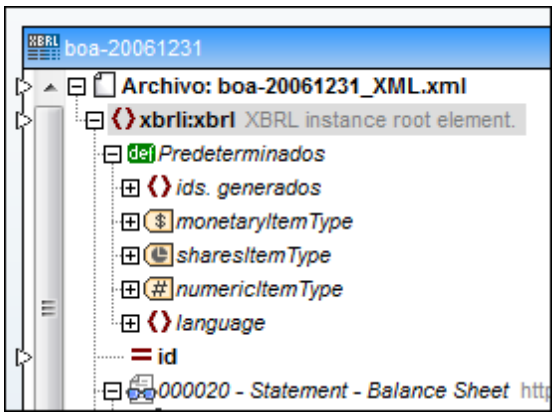
También puede asignar valores a los predeterminados del valor aspecto de la base de enlaces de tabla si el componente XBRL utiliza la vista de base de enlaces de table en lugar de las vistas de base de enlaces de presentación/definición. En la imagen siguiente, por ejemplo, los elementos `xbrli:unit`, `xbrli:identifier` y `xbrli:period` son predeterminados del valor aspecto que puede usar para crear asignaciones de datos en tablas XBRL.



Para habilitar el nodo Predeterminados en un elemento concreto:

Haga clic con el botón derecho en el elemento donde quiere ver las unidades predeterminadas y seleccione **XBRL | Ver predeterminados**. (Otra opción es hacer clic en el botón interactivo del elemento y elegir **Ver predeterminados** en el menú contextual que aparece.)

Como resultado se inserta el nodo *Predeterminados* que permite conectar valores predeterminados personales para los diferentes elementos.



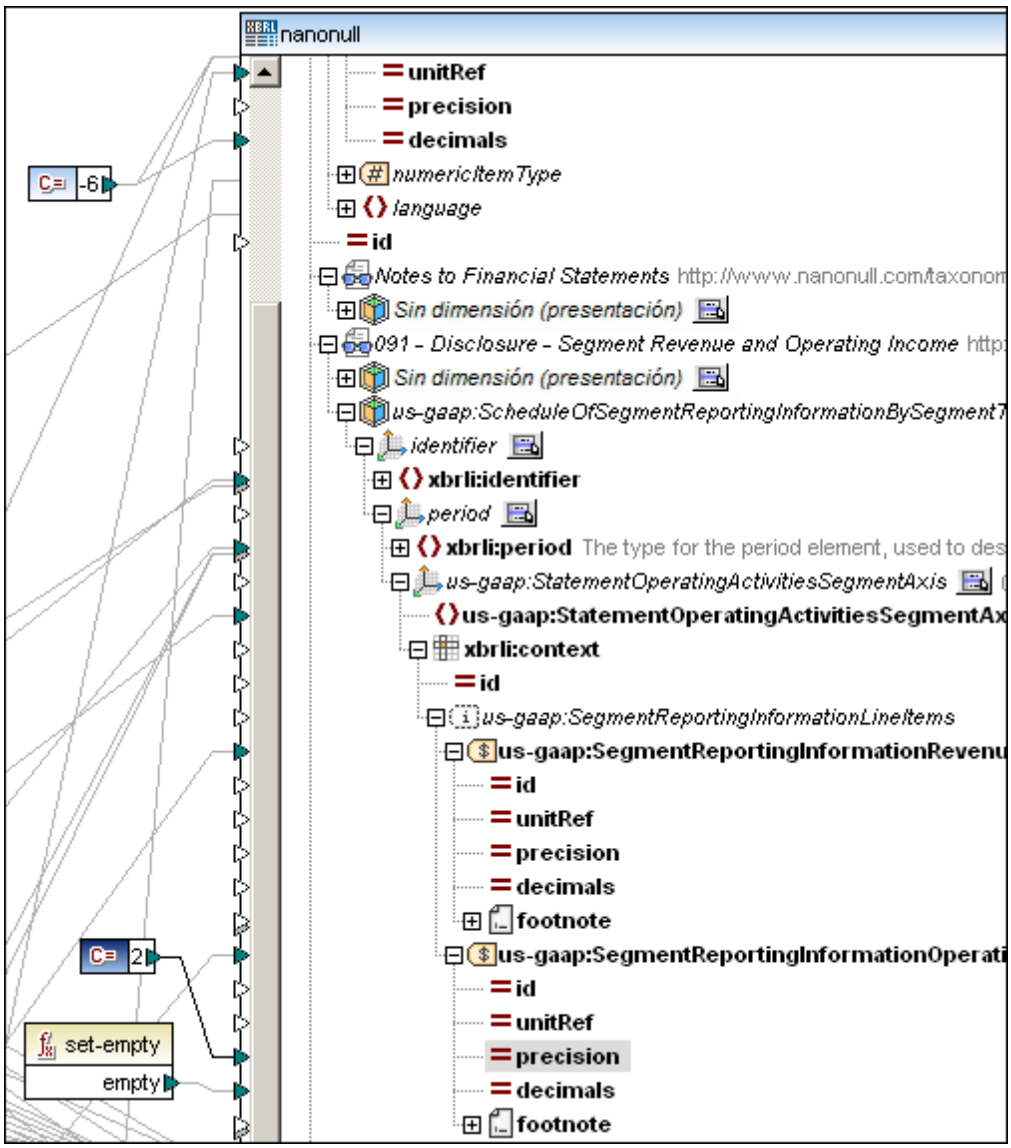
Reemplazar o desactivar un valor predeterminado

Si se definió un valor predeterminado para algún atributo del concepto (p. ej. para el atributo

decimals), se puede quitar esta opción de forma local en cada concepto, por medio de la función **set-empty**.

En la imagen siguiente la función **set-empty** desactiva el valor predeterminado -6 del elemento de tipo monetario **Segment Reporting Information, Revenue**.

Este elemento se incluirá en la contabilidad con una precisión 2 como valor del atributo **precision**, mientras que el elemento **Segment Reporting Information, Operating Income** aparecerá en la contabilidad con un atributo **decimals** cuyo valor es la unidad predeterminada -6.



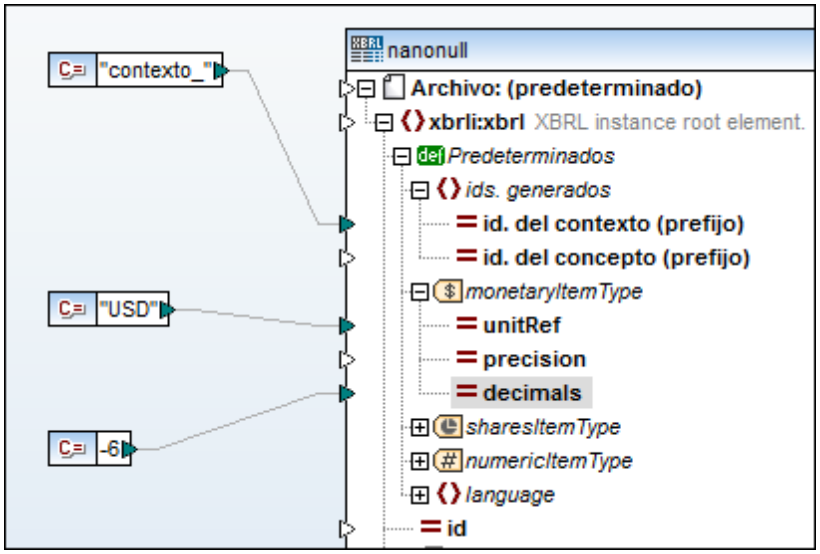
Control de contextos

La estructura jerárquica de los componentes XBRL permite a MapForce generar automáticamente contextos **xbrli:context** en el archivo XBRL de salida cuando se contabilizan conceptos relacionados.

El valor del atributo **id** de un contexto de un archivo XBRL es el valor del atributo **contextRef** de cada concepto XBRL. MapForce numera todos los contextos creados en un archivo de instancia de salida.

Puede personalizar el contexto asignándole un texto al nodo **id. del contexto (prefijo)** situado bajo el nodo **ids. generados**.

Por ejemplo, si asignamos la constante `contexto_` como prefijo predeterminado, se crean ids. de contexto consecutivos en el archivo de salida, es decir: `contexto_1`, `contexto_2`, `contexto_3` y así sucesivamente. **Nota:** si no define un valor predeterminado, MapForce usa el prefijo `ctx_` para todos los ids. de contexto.



Si el archivo XBRL de salida contiene **notas al pie** (elementos `footnote`), los conceptos relacionados deben tener identificadores de concepto para poder crear enlaces a las notas al pie generadas automáticamente (elementos `footnoteLink`). Estos atributos identificadores se generan automáticamente y puede asignar un texto al nodo **id. de concepto (prefijo)**.

Si no se asigna el prefijo, MapForce usa el prefijo `fact_` para todos los ids. de concepto.

7.8.9 Trabajar con hipercubos XBRL

Los hipercubos habilitan la gestión automática de contextos y pueden ser de varios tipos:

- definidos por la taxonomía (p. ej *Statement [Table]*)
- generados por MapForce para simplificar las dimensiones predeterminadas **period** e **identifier**, derivadas de la base de enlaces de presentación (p. ej. *Sin dimensión (presentación)*)
- generados por MapForce dentro del nodo **Todos los conceptos** *Sin dimensión*

Todos los hipercubos contienen dos dimensiones predeterminadas (**identifier** y **period**) que permiten leer/escribir con facilidad estos elementos en cada contexto. Además las dimensiones definidas en la taxonomía se relacionan automáticamente a los elementos **xbrli:segment** y

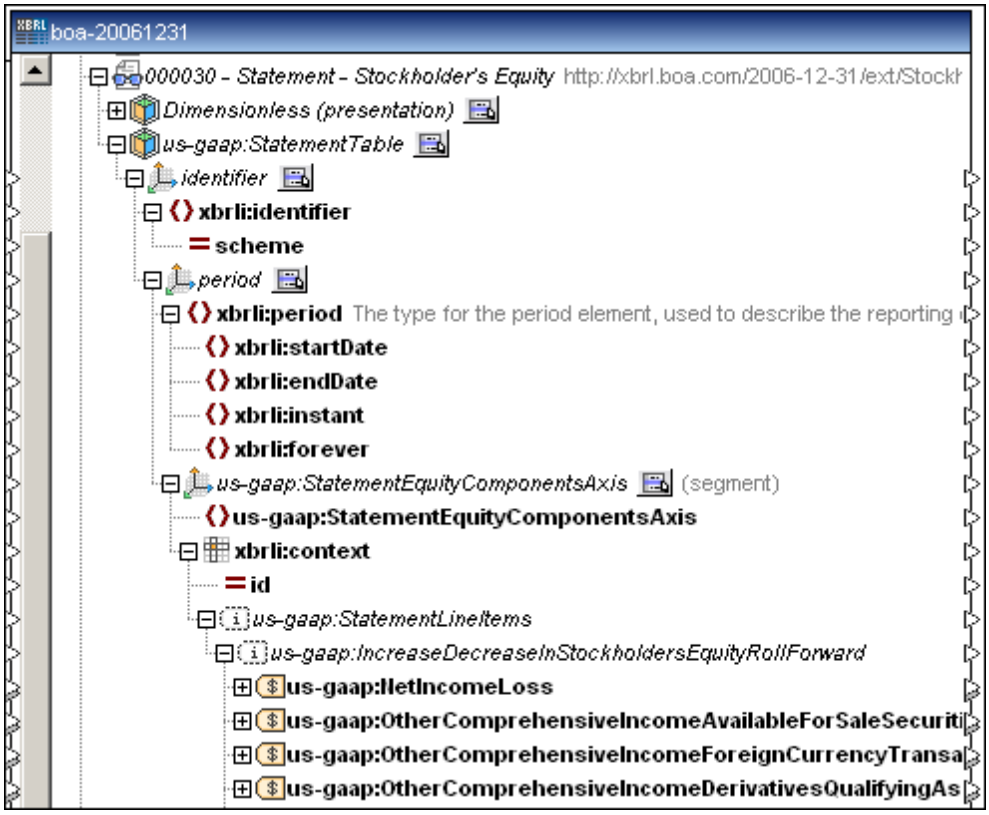
xbrli:scenario del contexto.

Los hipercubos denominados **Sin dimensión (presentación)** usan ambas dimensiones predeterminadas. El orden jerárquico de los conceptos que aparecen dentro de su nodo de contexto se toma de la base de enlaces de presentación.

Los elementos de los hipercubos denominados **Sin dimensión** también usan las dos dimensiones predeterminadas, pero no tienen un orden jerárquico y muestran una lista sin procesar de todos los conceptos definidos en la taxonomía.

Los demás hipercubos se definen dentro de la taxonomía y llevan el nombre definido en la base de enlaces de etiqueta de la taxonomía.

Los hipercubos, al igual que sus dimensiones o (ejes) tienen un botón interactivo que abre un menú contextual con opciones para definir la presentación de cada dimensión del componente. Por ejemplo, en la imagen siguiente puede ver un ejemplo de archivo de taxonomía que contiene tanto los hipercubos generados como los hipercubos definidos por la taxonomía.



Nota: MapForce muestra todos los hipercubos que tienen conceptos que se pueden contabilizar. Si una de las dimensiones del hipercubo no tiene dominio, no se representa en el componente XBRL.

7.8.9.1 Ver las dimensiones de un componente

Las dimensiones de XBRL hacen referencia a los valores de dimensión **explícitos** o **con tipo** de la instancia. La anotación de cada dimensión (que aparece entre paréntesis) indica si la

dimensión se contabiliza en el elemento **xbrli:segment** o **xbrli:scenario** del contexto.

Las dimensiones **con tipo** muestran los elementos de su tipo XML Schema como secundarios. Sus valores se pueden asignar a otros componentes directamente.

Las dimensiones **explícitas** de una taxonomía XBRL tienen un valor de tipo `xs:QName` de cierto dominio. Esto incluye los valores de miembro de dominio XBRL y el valor del elemento de dominio XBRL propiamente dicho. Las dimensiones explícitas pueden representarse de dos formas diferentes, dependiendo de los requisitos de la asignación y del componente al que se asignen los datos XBRL.

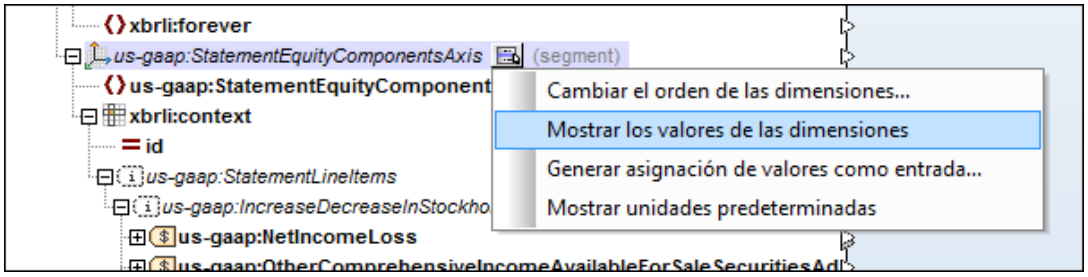
En un principio la dimensión explícita se presenta con **un solo nodo secundario** y se puede asignar directamente usando el secundario (p. ej. *`Statement, Equity Components [Axis]`*).

Esto es útil (para un componente XBRL de destino) cuando los valores de la dimensión se pueden derivar de un campo de los datos de origen (p. ej. de un campo de BD o de una columna de una tabla Excel). Ya que los datos de origen no suelen contener el tipo de datos QName necesario, MapForce puede crearlo automáticamente usando una función de asignación de valores (véase Generar asignaciones de valores para dimensiones de hipercubos).

Si desea usar varias asignaciones diferentes para los hechos, puede ver cada valor del dominio de la dimensión en un nodo diferente.

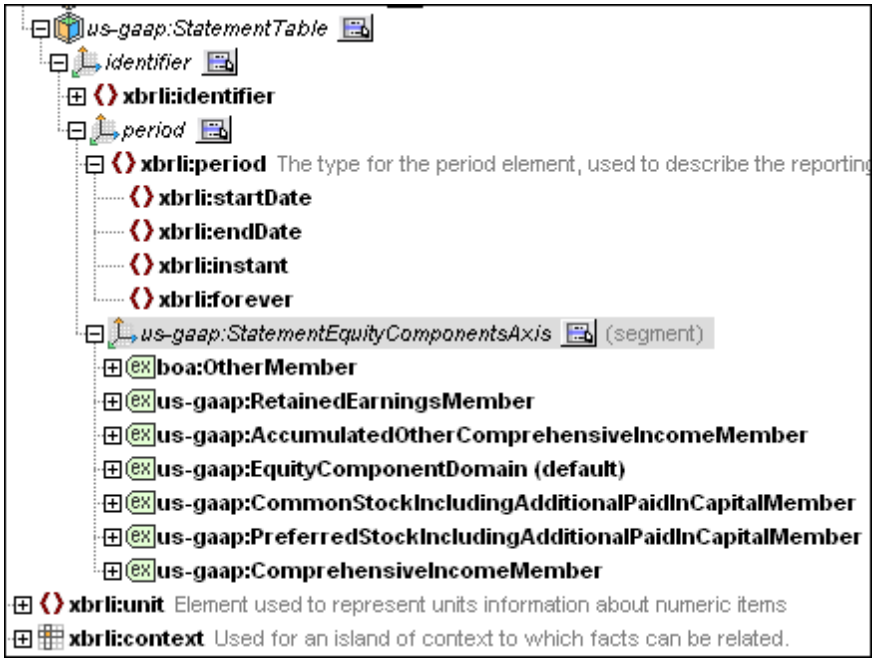
Para ver los valores de las dimensiones del componente:

1. Haga clic en el botón interactivo de la dimensión cuyos valores desea ver y seleccione la opción **Mostrar los valores de las dimensiones**.

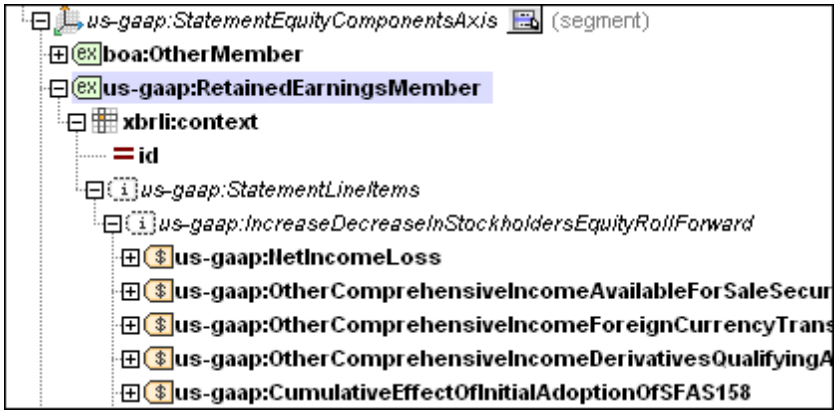


Bajo el nombre de la dimensión aparecen los elementos de dominio y miembro (**`domain/member`**) de la dimensión, cada uno de ellos con un icono verde claro. Se trata de los miembros explícitos de un dominio y llevan el prefijo `ex` en el icono.

© 2018 Altova GmbH



Cada miembro explícito contiene la misma subestructura, lo cual permite crear varias asignaciones diferentes.



Cuando se asigna el resultado de un concepto, solamente se utilizan los valores para los que el elemento de contexto relacionado tiene el valor de dimensión adecuado (p. ej. el valor de *Net Income (Loss)* del archivo de instancia se asigna solamente para los contextos que contienen el valor *Comprehensive Income [Member]* para la dimensión *Statement, Equity Components [Axis]*. No hace falta crear ningún filtro.)

A la hora de generar instancias XBRL de salida, MapForce también puede generar automáticamente valores de dimensión adecuados dentro del contexto. Por ejemplo, por cada elemento de tipo monetario *Net Income (Loss)* contabilizado, el nodo de contexto **`xbrli:context`** adquiere, dentro de su elemento **`xbrli:segment`**, un elemento para la dimensión explícita *Statement, Equity Components [Axis]* que contiene el valor *Comprehensive Income [Member]*.

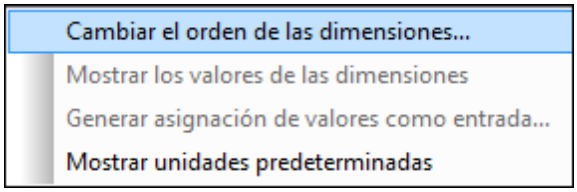
7.8.9.2 Cambiar el orden de las dimensiones

En un principio MapForce muestra todas las dimensiones del hipercubo como nodos secundarios anidados, creando automáticamente una jerarquía. El orden jerárquico de las dimensiones del hipercubo se puede cambiar para ajustarlo al orden del otro lado (no XBRL) de la asignación.

Además, cuando sea necesario establecer valores de dimensión para ciertos conceptos, MapForce puede mostrar una dimensión sin jerarquía y mostrarla como elemento secundario del nodo de contexto.

Para cambiar el orden de las dimensiones:

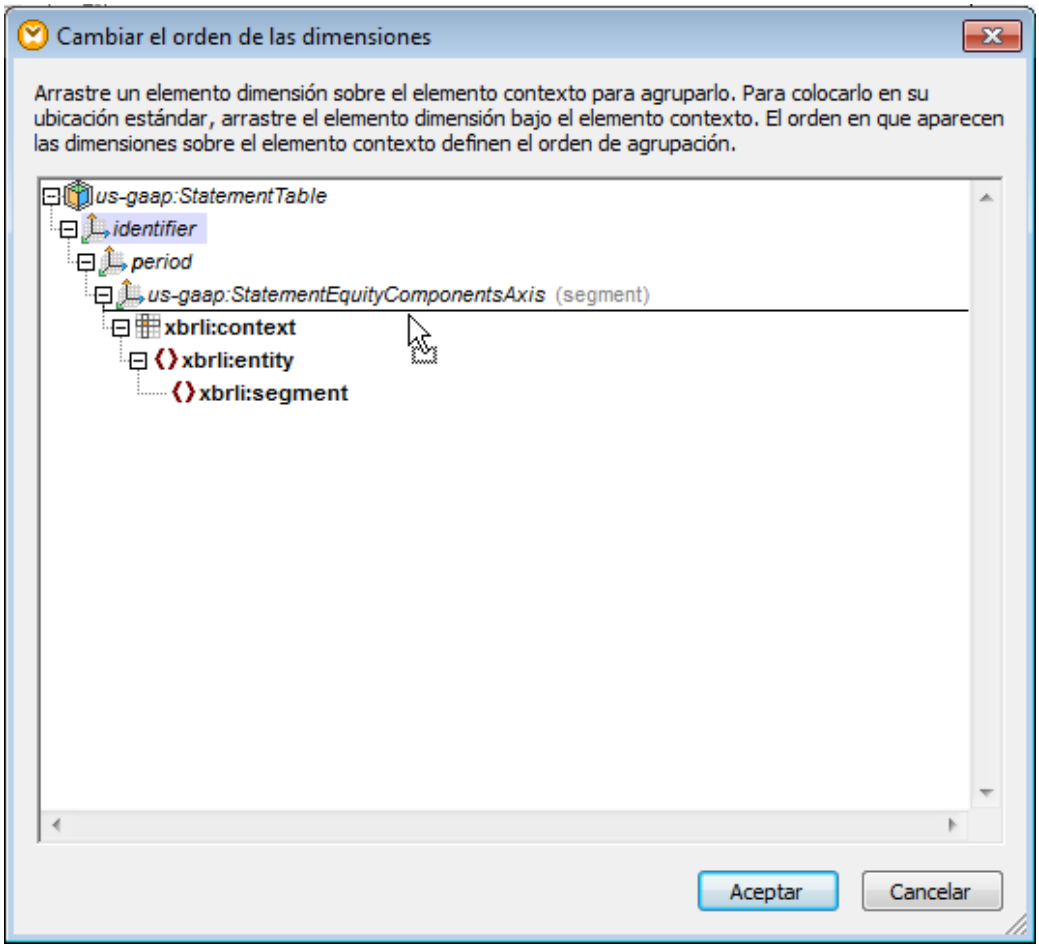
1. Haga clic en el botón interactivo de cualquier dimensión del hipercubo y seleccione la opción **Cambiar el orden de las dimensiones**.



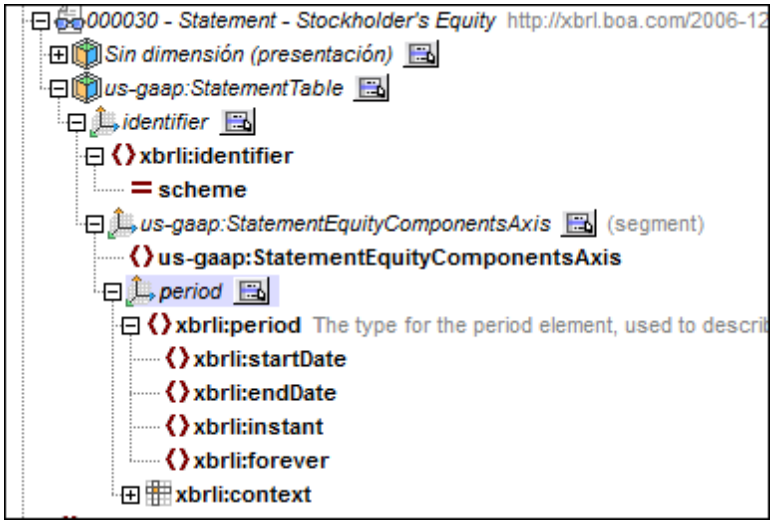
   A continuación se abre un cuadro de diálogo donde puede cambiar la posición de las dimensiones del hipercubo. Recuerde que los hipercubos tienen dos dimensiones predeterminadas (**`identifier`** y **`period`**) cuya posición se puede cambiar también.

2. Haga clic en la dimensión pertinente y arrástrela por el cuadro de diálogo hasta su nueva posición.

© 2018 Altova GmbH



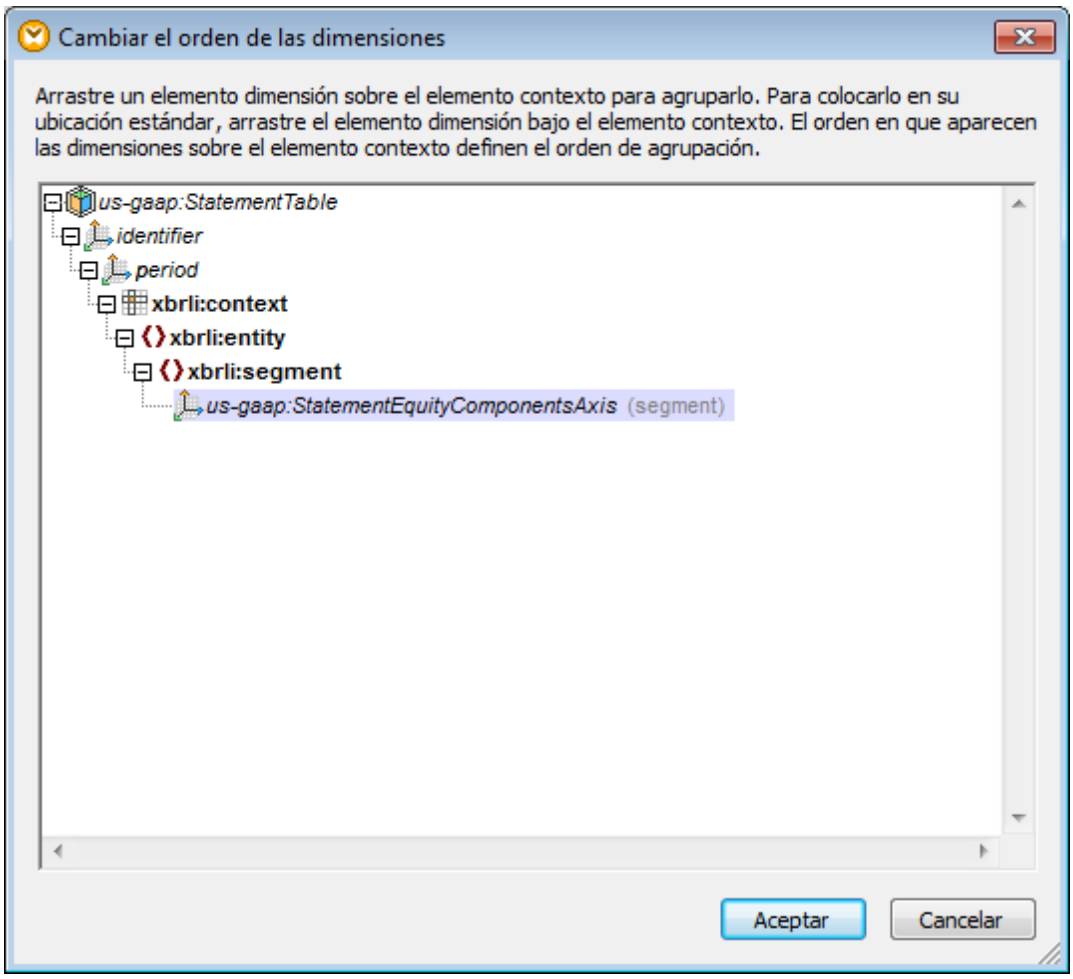
Observe que aparece una línea horizontal cuando se pasa el puntero por una posición donde se puede colocar la dimensión.



3. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. La dimensión se coloca en su nueva posición dentro del componente.

Para excluir una dimensión de la jerarquía:

Arrastre la dimensión y colóquela debajo del elemento **xbrli:context**.



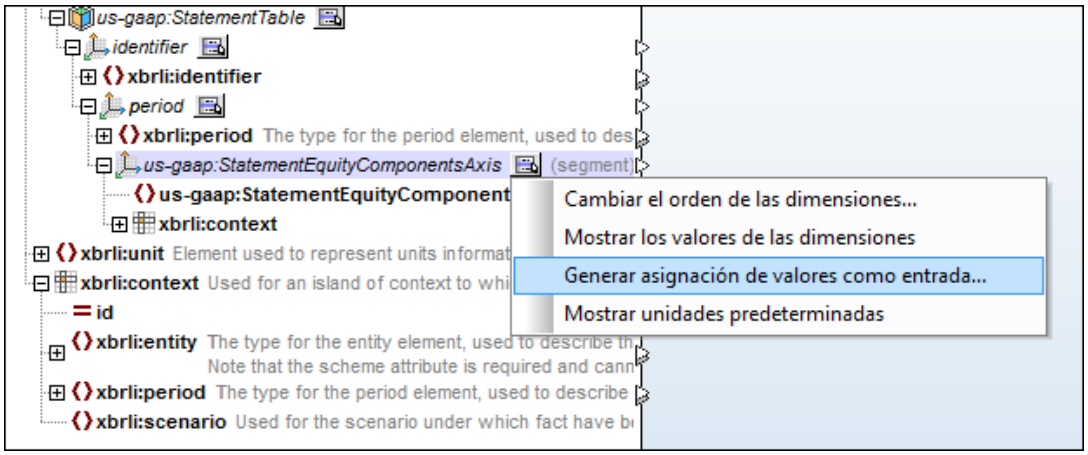
7.8.9.3 Generar asignaciones de valores para dimensiones explícitas de hipercubo

Esta opción transforma datos de entrada de cualquier tipo en un QName válido en el componente de destino. En otras palabras, la cadena de entrada se convierte en el nombre con prefijo (QName).

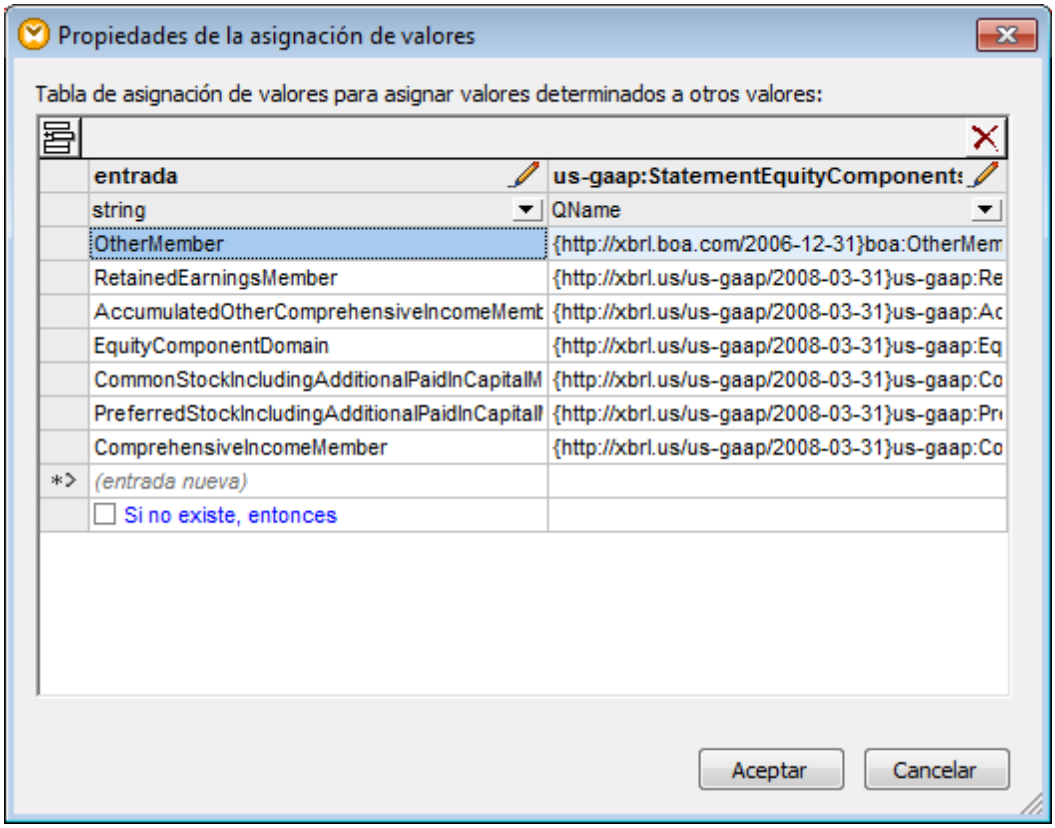
Para generar una tabla de búsqueda (asignación de valores) para la dimensión de hipercubo seleccionada:

1. Compruebe que los valores de la dimensión no están visibles en la dimensión elegida.
2. Haga clic en el botón interactivo de la dimensión y seleccione **Generar asignación de valores como entrada...** en el menú contextual.

© 2018 Altova GmbH



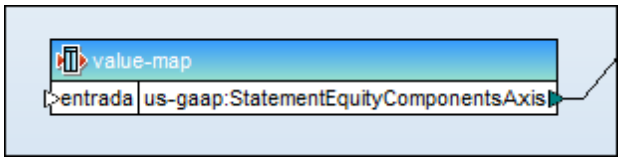
Esto abre el cuadro de diálogo "Propiedades de la asignación de valores", que contiene valores de entrada y salida generados automáticamente a partir de los miembros de dominio y del dominio predeterminado de la dimensión, tal y como se definieron en la taxonomía.



3. Edite los valores de entrada si hace falta y haga clic en **Aceptar** para insertar el componente de asignación de valores. Recuerde que, si quiere, puede editar el texto del encabezado de columna (haciendo doble clic en él).

   Como resultado se inserta la función de asignación de valores **`value-map`** y en ella

aparece el nombre de los parámetros de entrada y salida. El parámetro de salida se conecta automáticamente al elemento `domain` del componente de destino.



4. Conecte el elemento de entrada que contiene los datos de entrada que desea transformar al parámetro **`entrada`** de la función.

Para ver un ejemplo de uso general de la función de asignación de valores, consulte la sección Usar asignaciones de valores.

7.8.10 Trabajar con tablas XBRL

Si su taxonomía XBRL hace referencia a la base de enlaces de tabla (consulte la especificación Table Linkbase 1.0), puede configurar MapForce para ver los tipos de nodo correspondientes a la base de enlaces de tabla (p. ej. las tablas y los desgloses). De este modo podrá realizar asignaciones de datos entre los hechos representados en las tablas de la vista. Para más información sobre cómo configurar MapForce consulte el apartado *Seleccionar vistas de estructuras*.)

Para ver la información de la base de enlaces de tabla, MapForce suele utilizar el modelo estructural, excepto en estos casos:

- Los nodos de regla combinados son visibles.
- Los nodos acumulados que no tienen elementos del mismo nivel no forman un nivel jerárquico distinto.

Este modelo es uno de los tres modelos de datos definidos en la especificación Table Linkbase 1.0, §4. Además, para identificar las partes del modelo estructural al que hace referencia la asignación de datos, MapForce también utiliza información del modelo de definición.

La estructura de los componentes de MapForce solo tiene una dimensión, mientras que las tablas pueden tener hasta tres dimensiones (x, y, z). Por eso, para representar las dimensiones de una tabla, MapForce anida los desgloses unos dentro de otros en este orden: z, y, x. Es decir, en MapForce los nodos de desglose tienen la función de indicar dónde comienza cada desglose.

En esta sección explicamos cómo trabajar con entidades de la base de enlaces de tabla:

- Mostrar y ocultar desgloses
- Cambiar el orden de los desgloses
- Trabajar con parámetros

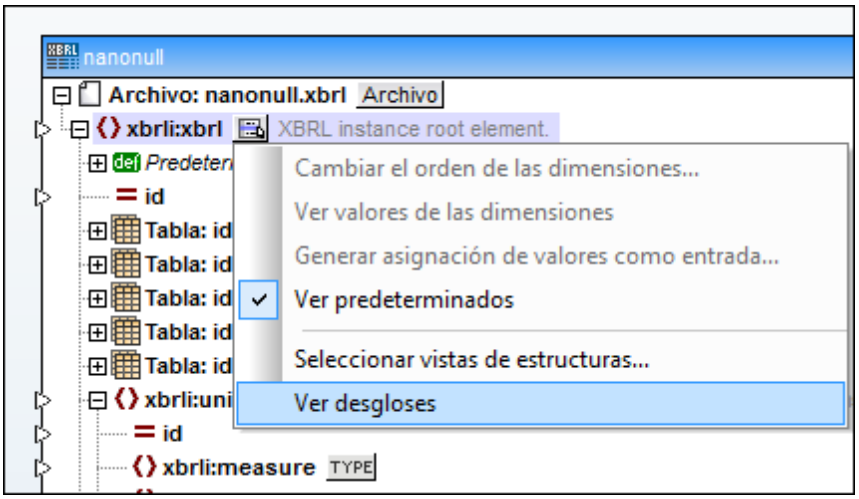
© 2018 Altova GmbH

7.8.10.1 Mostrar y ocultar desgloses

Si el documento XBRL hace referencia a la base de enlaces de tabla, podrá elegir entre mostrar u ocultar los desgloses en el componente XBRL.

Para mostrar u ocultar los desgloses:

1. Busque un nodo del componente XBRL que contenga el botón **Mostrar menú contextual** (p. ej. el nodo raíz).



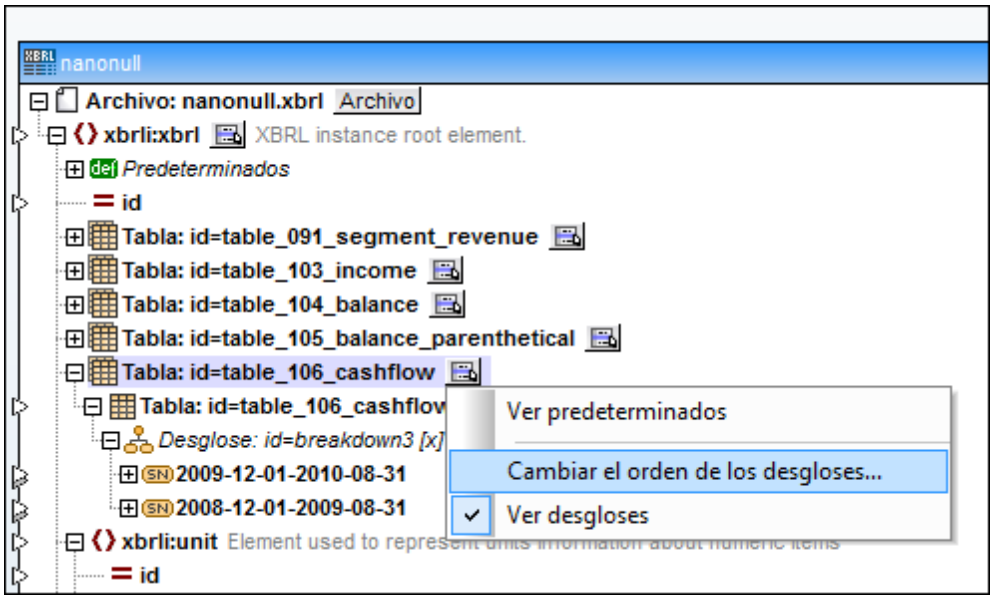
2. Haga clic en ese botón y después seleccione **Ver desgloses** en el menú contextual.

7.8.10.2 Cambiar el orden de los desgloses

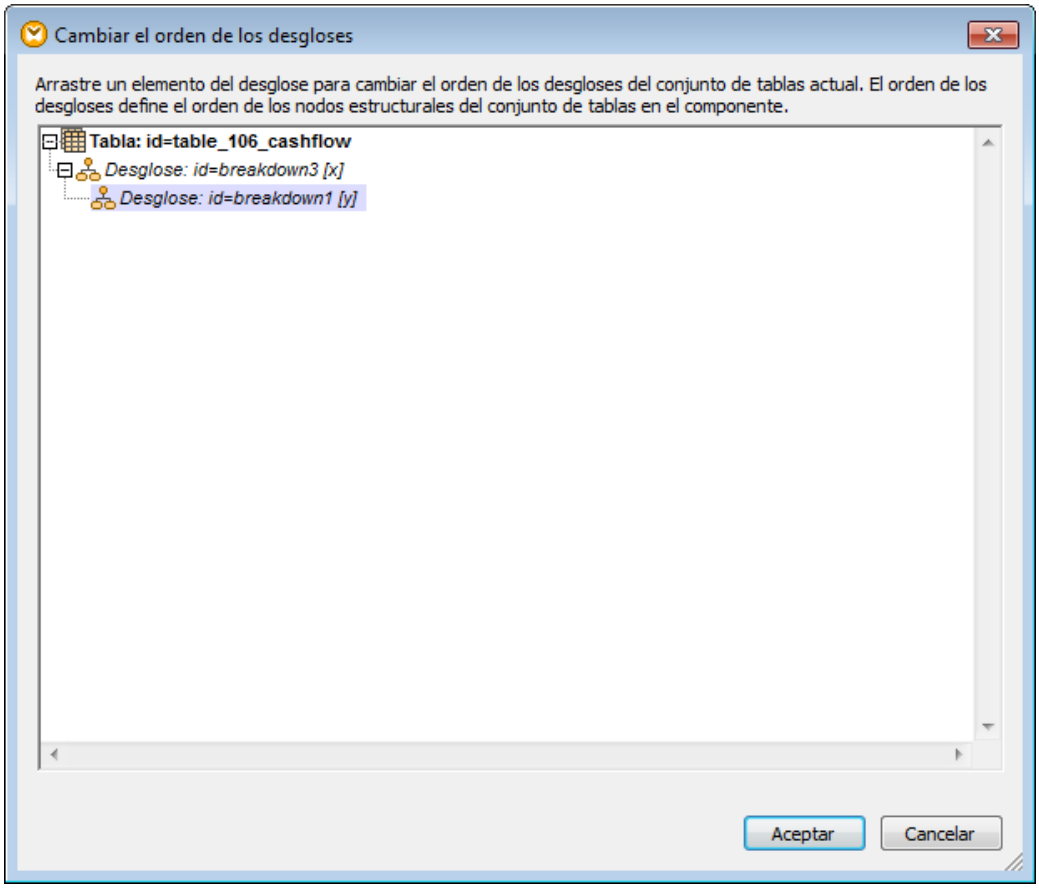
Si el documento XBRL hace referencia a la base de enlaces de tabla, podrá cambiar el orden de los desgloses en el componente XBRL.

Para cambiar el orden de los desgloses:

1. Busque el nodo del componente XBRL donde desea cambiar el orden de los desgloses.
2. Haga clic en el botón interactivo situado a la derecha del nodo y seleccione **Cambiar el orden de los desgloses** en el menú contextual. Si este comando no está disponible, significa que la operación no procede en el nodo seleccionado.



3. En el cuadro de diálogo que aparece arrastre los desgloses hasta la ubicación deseada (p. ej. puede arrastrar el desglose ***y*** encima del desglose ***x***).



4. Para terminar haga clic en **Aceptar**.

 © 2018 Altova GmbH

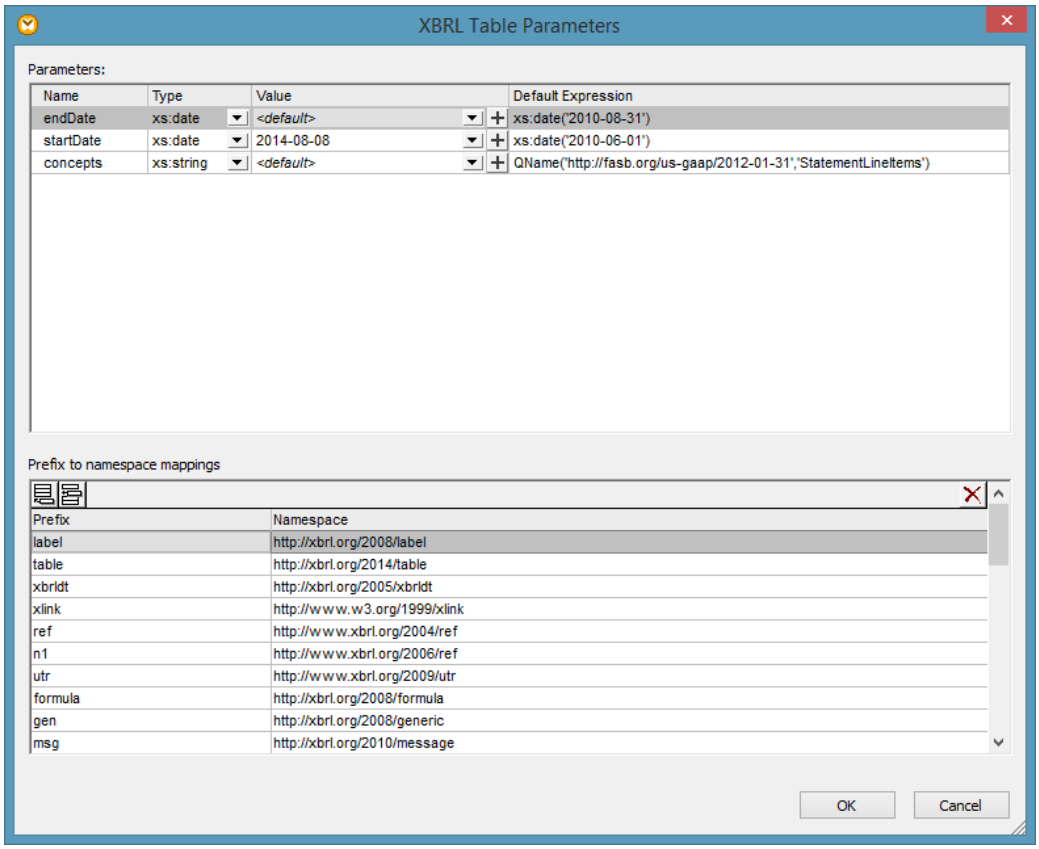
7.8.10.3 Trabajar con parámetros

Si el documento XBRL hace referencia a la base de enlaces de tabla y contiene parámetros (especificación Table Linkbase 1.0, §5.3), podrá modificar los valores de los parámetros. Si el contexto lo permite, también podrá anexar valores de parámetro nuevos a los parámetros disponibles.

Los valores de parámetro se pueden cambiar en el cuadro de diálogo "Parámetros de tabla XBRL" (instrucciones más abajo). Este cuadro de diálogo muestra los parámetros que se definieron tanto a nivel global (es decir, en cualquier parte de la DTS) como en las diferentes tablas. Cuando se cambia un valor de parámetro y se cierra el cuadro de diálogo, MapForce intenta regenerar todas las tablas del componente y combinar los nodos de la estructura jerárquica. Si faltan parámetros y esto impide regenerar la estructura, el nodo correspondiente muestra un nodo de error. Para más información sobre el error consulte la información de la ventana Mensajes.

Para cambiar los valores de los parámetros:

1. Haga clic en el botón interactivo del nodo raíz y seleccione **Definir parámetros...** en el menú contextual. (Si este comando no está disponible, significa que la base de enlaces de tabla no contiene parámetros.)



2. Seleccione el parámetro correspondiente y escriba su nuevo valor en la columna *Valor*.

Para anexar valores de parámetro nuevos a los parámetros disponibles:

1. Seleccione el parámetro y después haga clic en el botón **Agregar valor de parámetro** de la columna *Valor*.
2. Ahora escriba el valor de parámetro nuevo.

Para eliminar un valor anexado:

Haga clic en el botón **Eliminar valor** situado junto al valor que quiere eliminar.

7.8.11 Ejemplos de asignaciones de datos XBRL

Este apartado incluye estos ejemplos de asignaciones XBRL:

- Asignación de datos Microsoft Access a XBRL: explica cómo asignar datos de una BD de Microsoft Access a una taxonomía XBRL para generar un archivo XBRL de instancia válido.
- Asignación de datos Microsoft Excel a XBRL: explica cómo asignar datos de una hoja de cálculo de Microsoft Excel a una taxonomía XBRL para generar un archivo XBRL de instancia válido.

7.8.11.1 Asignar datos de Microsoft Access a XBRL

Este ejemplo es el archivo `DB_to_XBRL.mfd` de la carpeta `...\MapForceExamples` y utiliza varios filtros y funciones para extraer datos de una BD.

La taxonomía `nanonull.xsd` se deriva de la taxonomía US:GAAP. La asignación crea una instancia XBRL de salida que contiene todos los contextos, conceptos y notas al pie para una declaración y tres extractos.

El informe contable **091 - Disclosure - Segment Revenue and Operating Income** muestra cómo asignar valores de dimensiones en MapForce. El hipercubo **`us-gaap:ScheduleOfSegmentReportingInformationBySegmentTable`** contiene la dimensión explícita: `us-gaap:StatementOperatingActivitiesSegmentAxis`.

Su dominio se amplía en la taxonomía con tres valores de la dimensión: **`nanonull:USA`**, **`nanonull:Europe`** y **`nanonull:Asia`**. En la asignación puede ver que la función de asignación de valores **`value-map`** asigna los valores de la columna **`Name`** de la tabla **`Region`** de la BD a los valores necesarios de tipo QName de la dimensión.

El informe contable **106 - Statement - Nanonull and Consolidated Statement of Cash Flows** muestra cómo escribir hechos en la instancia XBRL de salida relacionados tanto con períodos de tipo intervalo como de tipo instantáneo.

La contabilización adecuada de hechos como **Cash and cash equivalents at beggining (end) of period** se puede conseguir duplicando el elemento **`period`** en la estructura de la jerarquía.

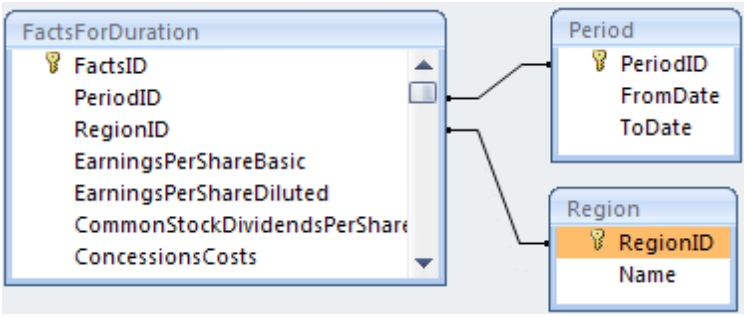
La asignación define dos unidades en la instancia de salida: **`USD`** y **`perShareItemType`**. Para ello hace falta duplicar el elemento XBRL **`xbrli:unit`**. Los elementos de medida relacionados se

© 2018 Altova GmbH

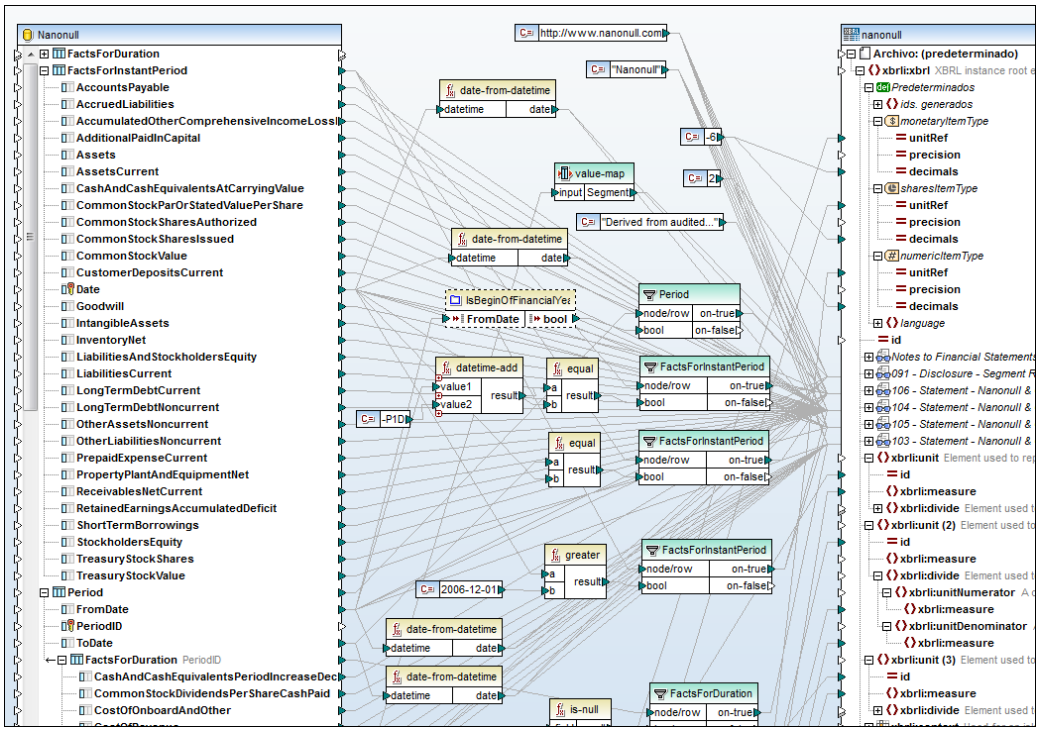
crean usando las funciones **xbrl-measure-currency** y **xbrl-measure-shares** de la biblioteca XBRL.

Los hechos de las tablas de la BD se dividen dependiendo de si se refieren a un período de tipo instantáneo o a uno de tipo intervalo.

- La tabla **FactsForInstantPeriod** es una tabla plana de valores.
- La tabla **FactsForDuration** es jerárquica y cada uno de los hechos que contiene se refiere a un **PeriodID** concreto así como a una **RegionID**.



La tabla **Period** usa los campos **FromDate** y **ToDate** para definir las fechas de inicio y de fin de período, mientras que la tabla **Region** se refiere a los hechos de una región específica, es decir, Asia, Europa o EE UU.



Elementos XBRL obligatorios necesarios en un archivo XBRL de instancia:

- **unitRef** y conceptos monetarios **decimals** o **precision**
- **xbrli:identifier** y **scheme** de la dimensión **identifier**

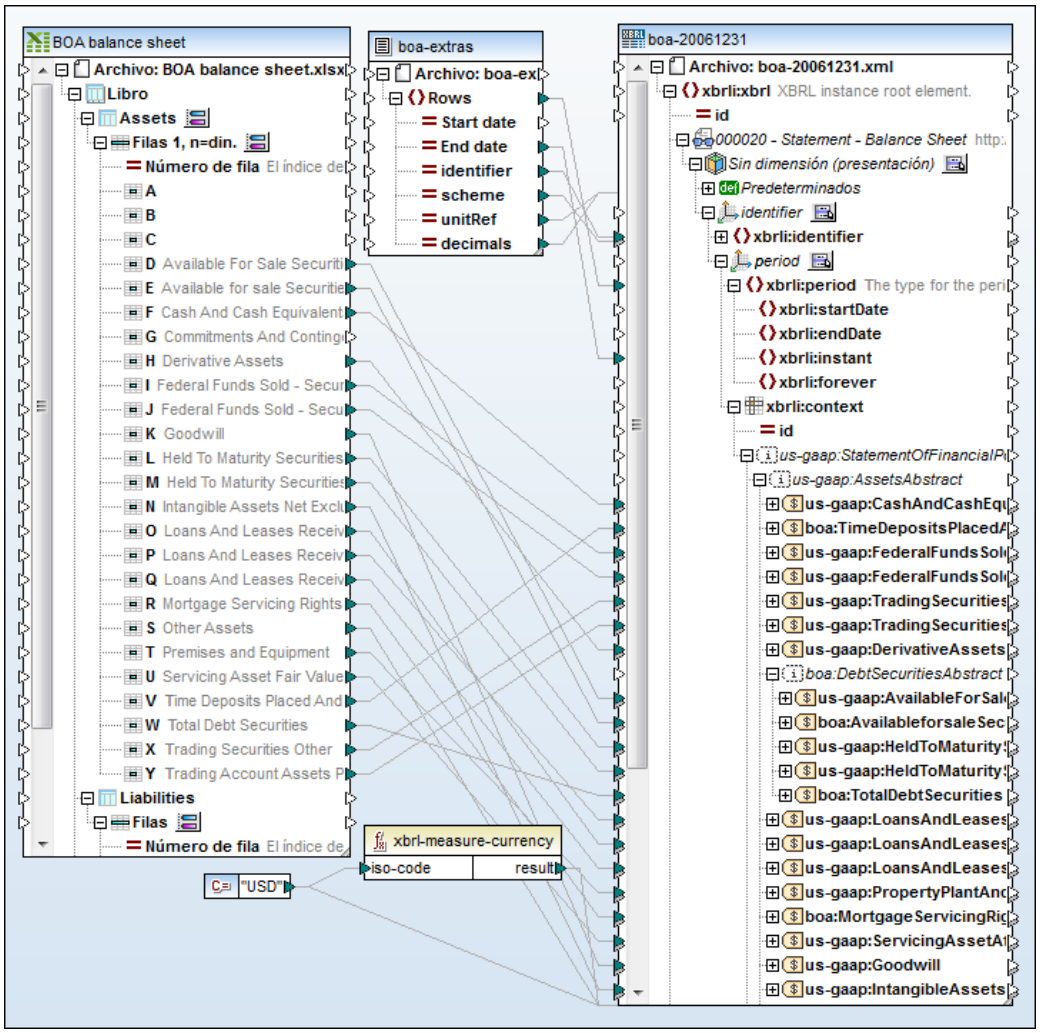
- **xbrli:period** y los elementos **xbrli:instant** o **xbrli:startDate/xbrli:endDate**
- **xbrli:id** y **xbrli:measure** en el elemento **xbrli:unit**

Consulte también el apartado Asignar datos de Microsoft Excel a XBRL.

7.8.11.2 Asignar datos de Microsoft Excel a XBRL

El archivo de asignación de datos utilizado en este ejemplo es `boa-balance-sheet.mfd` , disponible en la carpeta `...\MapForceExamples\Tutorial\`. Los archivos de taxonomía utilizados en el ejemplo están en la carpeta `..\Tutorial`.

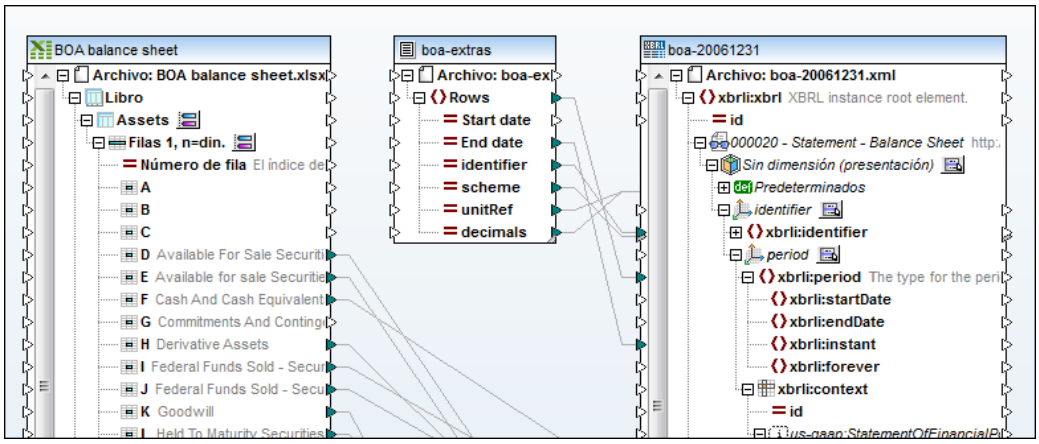
En este archivo de ejemplo puede ver cómo se asignan datos de una hoja Excel a una taxonomía XBRL para generar un archivo XBRL de instancia. Para no complicar demasiado las cosas, en el ejemplo solamente se crearon asignaciones con la hoja de cálculo **Assets**. El resultado de esta asignación es un documento XBRL válido que contiene los datos de la hoja **Assets** relacionados con un instante concreto, el día 31 de diciembre de 2006.



En este ejemplo las columnas Excel, que están en orden alfabético, se asignan directamente a los elementos/hechos XBRL de la taxonomía (p. ej. **Cash and Cash Equivalents**) .

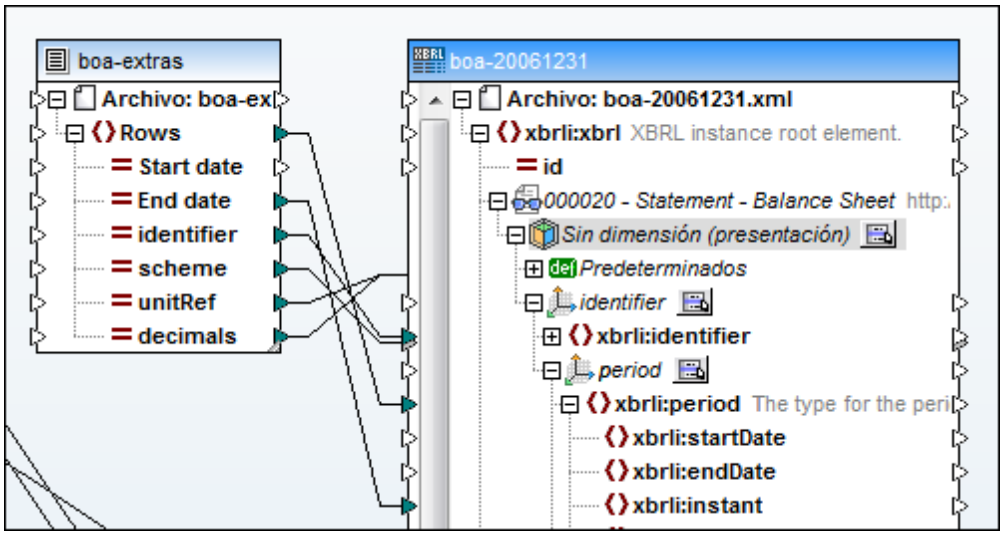
 © 2018 Altova GmbH

Tenga en cuenta que el nombre de los elementos de los componentes de origen y destino no son idénticos ni es necesario que lo sean. El nombre del elemento de la taxonomía de destino determina el nombre que se utiliza en el documento XBRL de instancia resultante. Sin embargo, si el nombre de los elementos es el mismo, tendrá la ventaja de poder usar la opción **Conectar automáticamente los secundarios equivalentes** (*véase* *Conectar los secundarios equivalentes*).



Hay varios elementos que deben asignarse para poder generar un archivo XBRL de instancia válido. Algunos elementos obligatorios vienen dados por un archivo de texto (**boa-extras**) cuyos campos se asignan a elementos secundarios del elemento *`Predeterminados`* del hipercubo *`Sin dimensión (presentación)`*.

Para insertar el elemento *`Predeterminados`* haga clic con el botón derecho en una dimensión de un hipercubo y seleccione la opción **XBRL | Ver predeterminados** o haga clic en el botón interactivo y seleccione la misma opción en el menú contextual.

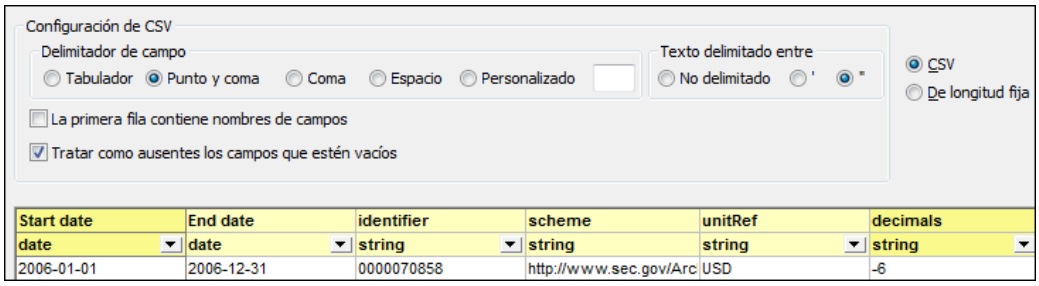


Elementos XBRL obligatorios necesarios en un archivo XBRL de instancia:

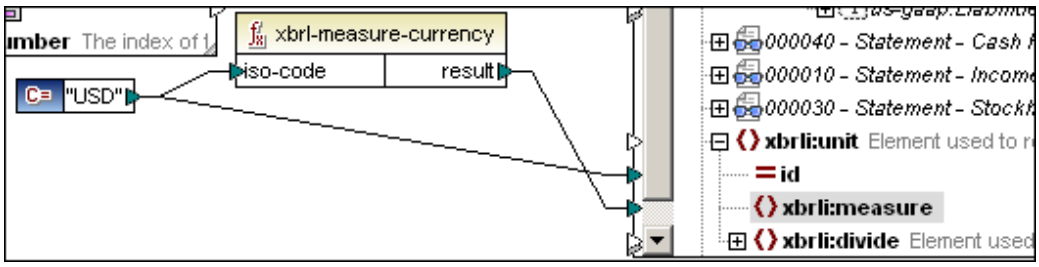
- **`unitRef`** y conceptos monetarios **`decimals`** o **`precision`**
- **`xbrli:identifier`** y **`scheme`** de la dimensión **`identifier`**

- **`xbrli:period`** y los elementos **`xbrli:instant`** o **`xbrli:startDate/xbrli:endDate`**
- **`xbrli:id`** y **`xbrli:measure`** en el elemento **`xbrli:unit`**

El archivo de texto **boa-extras** aporta los datos para algunos de los elementos obligatorios (*imagen siguiente*).



Los datos para los elementos **`xbrli:id`** y **`xbrli:measure`**, situados al final del componente de taxonomía, vienen dados por una constante.



Haga clic en el panel *Resultados* para ver el archivo XBRL de instancia resultante.

- Haga clic en el botón **Validar archivo de salida** de la barra de herramientas Resultados para comprobar la validez del archivo XBRL. En la ventana Mensajes aparecen los mensajes de error y advertencia.
- Haga clic en el botón **Configurar la vista Texto** para configurar la vista de los resultados.

© 2018 Altova GmbH

```
<?xml version="1.0" encoding="UTF-8"?>
<xbrl xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:xl="http://www.xbrl.o
  <link:schemaRef xlink:type="simple" xlink:href="A:/AA-tutorial-xbrl-example/boa-20061231.xsd" />
  <context id="ctx1">
    <entity>
      <identifier scheme="http://www.sec.gov/Archives/edgar/data/70858/000119312507042036/d10k.htm">0000070858</iden
    </entity>
    <period>
      <instant>2006-12-31</instant>
    </period>
  </context>
  <us-gaap:CashAndCashEquivalentsAtCarryingValue contextRef="ctx1" unitRef="USD" decimals="-6">36429</us-gaap:CashAn
  <boa:TimeDepositsPlacedAndOtherShorttermInvestments contextRef="ctx1" unitRef="USD" decimals="-6">13952</boa:TimeDep
  <us-gaap:FederalFundsSoldAndSecuritiesPurchasedUnderAgreementsToResell contextRef="ctx1" unitRef="USD" decimals="-6
  <us-gaap:FederalFundsSoldAndSecuritiesPurchasedUnderAgreementsToResellPledgedAsCollateral contextRef="ctx1" unitRef=
  <us-gaap:TradingSecuritiesOther contextRef="ctx1" unitRef="USD" decimals="-6">153052</us-gaap:TradingSecuritiesOther>
  <us-gaap:TradingSecuritiesPledgedAsCollateral contextRef="ctx1" unitRef="USD" decimals="-6">922274</us-gaap:TradingSecu
  <us-gaap:DerivativeAssets contextRef="ctx1" unitRef="USD" decimals="-6">23439</us-gaap:DerivativeAssets>
  <us-gaap:AvailableForSaleSecuritiesDebtSecurities contextRef="ctx1" unitRef="USD" decimals="-6">192806</us-gaap:Availabl
  <boa:AvailableforsaleSecuritiesDebtSecuritiesCollateral contextRef="ctx1" unitRef="USD" decimals="-6">83875</boa:Availablef
  <us-gaap:HeldToMaturitySecurities contextRef="ctx1" unitRef="USD" decimals="-6">40</us-gaap:HeldToMaturitySecurities>
  <us-gaap:HeldToMaturitySecuritiesFairValue contextRef="ctx1" unitRef="USD" decimals="-6">40</us-gaap:HeldToMaturitySecu
  <boa:TotalDebtSecurities contextRef="ctx1" unitRef="USD" decimals="-6">192846</boa:TotalDebtSecurities>
  <us-gaap:LoansAndLeasesReceivableGrossCarryingAmount contextRef="ctx1" unitRef="USD" decimals="-6">706490</us-gaa
  <us-gaap:LoansAndLeasesReceivableAllowance contextRef="ctx1" unitRef="USD" decimals="-6">9016</us-gaap:LoansAndLe
  <us-gaap:LoansAndLeasesReceivableNetReportedAmount contextRef="ctx1" unitRef="USD" decimals="-6">697474</us-gaap:L
  <us-gaap:PropertyPlantAndEquipmentNet contextRef="ctx1" unitRef="USD" decimals="-6">9255</us-gaap:PropertyPlantAndEqu
  <boa:MortgageServicingRights contextRef="ctx1" unitRef="USD" decimals="-6">3045</boa:MortgageServicingRights>
  <us-gaap:ServicingAssetAtFairValueAmount contextRef="ctx1" unitRef="USD" decimals="-6">20</us-gaap:ServicingAssetAtF
  <us-gaap:Goodwill contextRef="ctx1" unitRef="USD" decimals="-6">65662</us-gaap:Goodwill>
  <us-gaap:IntangibleAssetsNetExcludingGoodwill contextRef="ctx1" unitRef="USD" decimals="-6">9422</us-gaap:IntangibleAss
  <us-gaap:OtherAssets contextRef="ctx1" unitRef="USD" decimals="-6">119683</us-gaap:OtherAssets>
  <unit id="USD">
    <measure xmlns:iso4217="http://www.xbrl.org/2003/iso4217">iso4217:USD</measure>
  </unit>
</xbrl>
```

7.9 HL7 versión 3

MapForce 2018 es automáticamente compatible con HL7 versión **3.x** porque este estándar está basado en XML.

En la página Componentes de MapForce del sitio web de Altova encontrará un instalador para los esquemas XML y los archivos de configuración HL7 V2.2 - V2.5.1. Durante la instalación seleccione la opción *Instalación personalizada* para instalar los componentes y esquemas XML HL7 V3 solamente.

Ubicación de los esquemas XML HL7 tras la instalación:

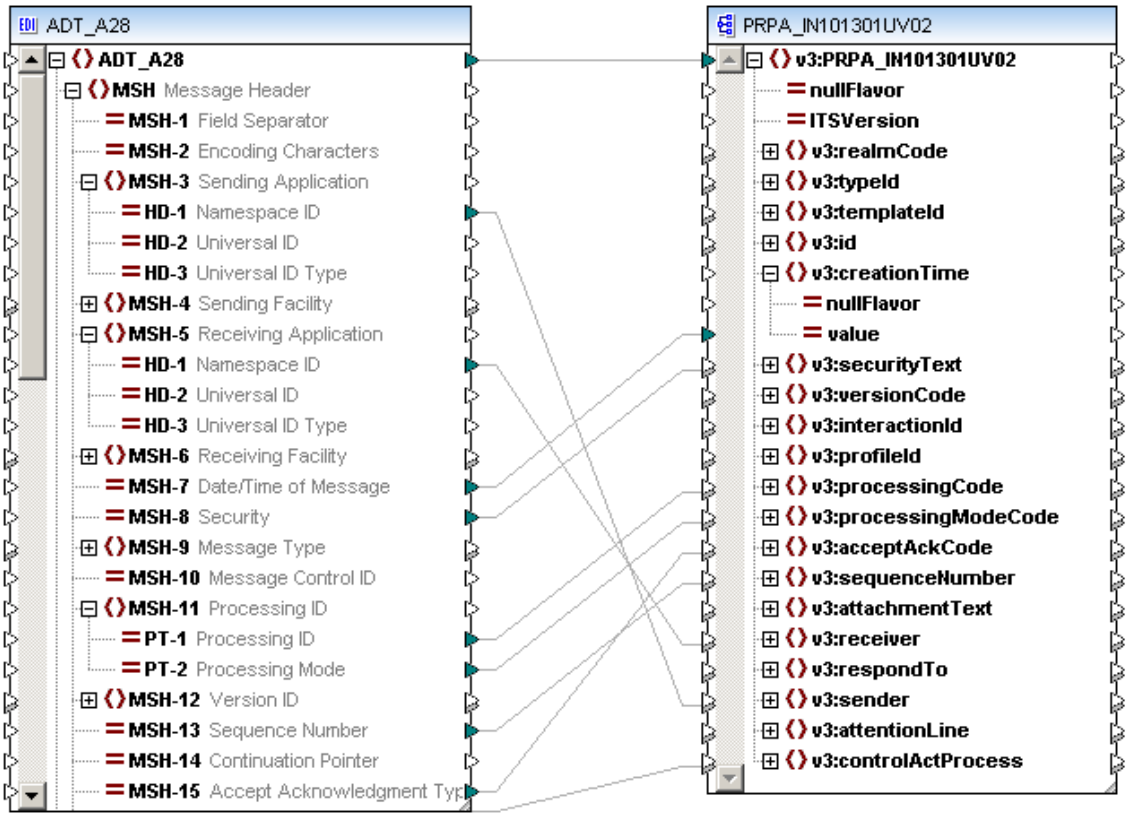
• MapForce de 32 bits en sistemas operativos de 32 bits • MapForce de 64 bits en sistemas operativos de 64 bits	`C:\Archivos de programa\Altova\Common2018\Schemas\hl7v3`
• MapForce de 32 bits en sistemas operativos de 64 bits	`C:\Archivos de programa(x86)\Altova\Common2018\Schemas\hl7v3`

En MapForce puede usar documentos HL7 como componentes de origen y destino. Estos datos también se pueden asignar a componentes de esquema XML, base de datos , EDI , etc..

El ejemplo que mostramos a continuación es el archivo **HL7V260_To_HL7V3.mfd**, disponible en la carpeta `...\MapForceExamples` y en él se crean asignaciones de datos entre un documento HL7 V2.6 y un documento HL7 V3.

- El archivo EDI `ADT_A28` está en la carpeta `...\MapForceExamples`.
- El esquema de destino `PRPA_IN101301UV02.xsd` está en la carpeta `...\MapForceExamples\HL7V3_Example_Schemas`.

© 2018 Altova GmbH



El archivo XML resultante aparece en la siguiente imagen.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <PRPA_IN101301UV02 xsi:schemaLocation="urn:hl7-org:v3
    C:/DOCUME~1/MY/MYDOCU~1/Altova/MapForce2009/MapForceExamples/HL7V3_Example_Schemas/
    http://www.w3.org/2001/XMLSchema" xmlns="urn:hl7-org:v3" xmlns:xsi="http://www.w3.org/2001/
3     <creationTime value="20070802153810000000000" />
4     <securityText>
5       <reference />
6     </securityText>
7     <processingCode code="P" />
8     <processingModeCode />
9     <acceptAckCode code="AL" />
10    <sequenceNumber />
11    <receiver>
12      <typeId assigningAuthorityName="Therapies" />
13    </receiver>
14    <sender>
15      <typeId assigningAuthorityName="RHAPSODY" />
16    </sender>
17    <controlActProcess>
18      <effectiveTime>
19        <high value="20070802153810000000000" />
20      </effectiveTime>
21      <authorOrPerformer>
22        <time>
23          <low />
24          <high value="20070802153810000000000" />
25        </time>
26      </authorOrPerformer>
```


© 2018 Altova GmbH

Altova MapForce 2018 Enterprise Edition

Funciones

8 Funciones

Las funciones suponen un potente mecanismo para transformar datos. En esta sección encontrará instrucciones para trabajar con funciones (tanto sin son funciones integradas de MapForce, como si son funciones definidas por el usuario o importadas de fuentes externas). A continuación puede ver un resumen de los temas que incluye esta sección:

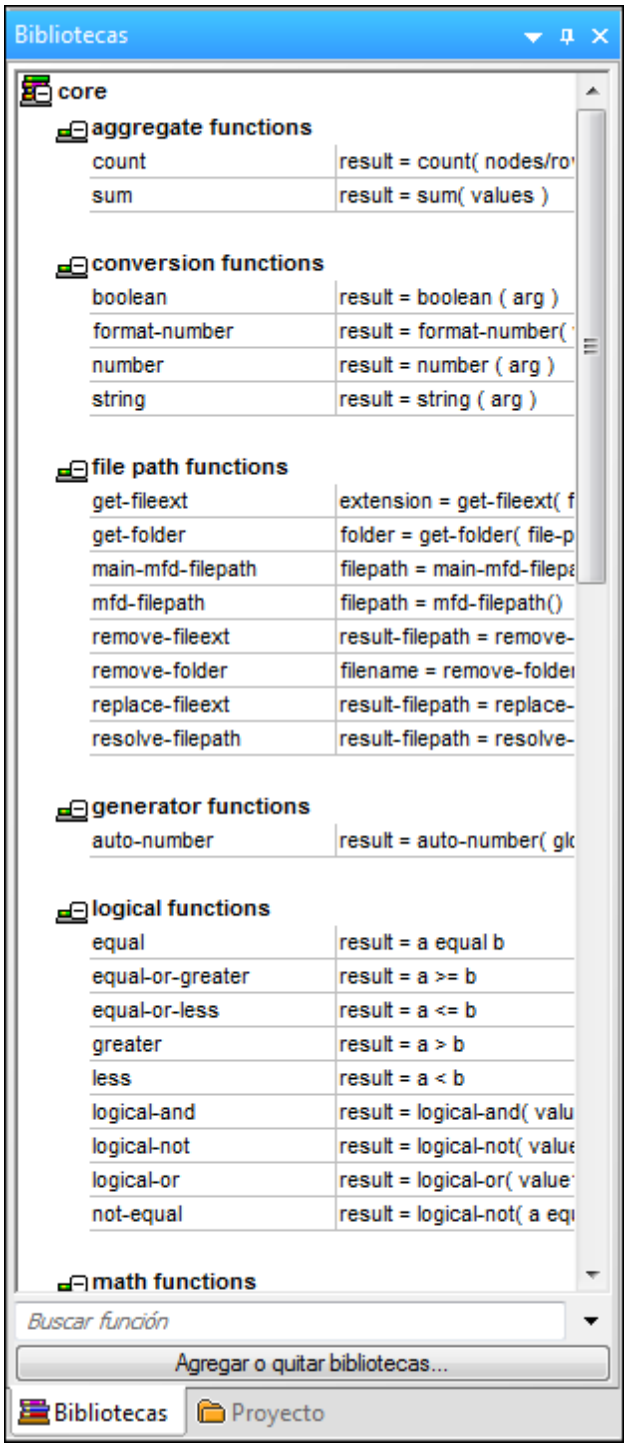
Si le interesa...	Consulte este tema...
Aprender a trabajar con funciones en MapForce.	Trabajar con funciones
Crear sus propias funciones en MapForce.	Funciones definidas por el usuario
Agregar funciones XSLT personales a MapForce.	Importar funciones XSLT 1.0 o 2.0 personales
Agregar bibliotecas .NET DLL y Java .class personales a MapForce.	• Importar bibliotecas Java y .NET personales • Referencias manuales a bibliotecas Java, C# y C++ personales
Escribir su propia biblioteca Java para utilizarla con MapForce.	Crear una biblioteca Java
Escribir su propia biblioteca C# para utilizarla con MapForce.	Crear una biblioteca C#
Escribir su propia biblioteca C++ para utilizarla con MapForce.	Crear una biblioteca C++
Ver todas las funciones integradas de MapForce o consultar la descripción de una función.	Referencia de la biblioteca de funciones

© 2018 Altova GmbH

8.1 Trabajar con funciones

Para usar una función en una asignación de datos:

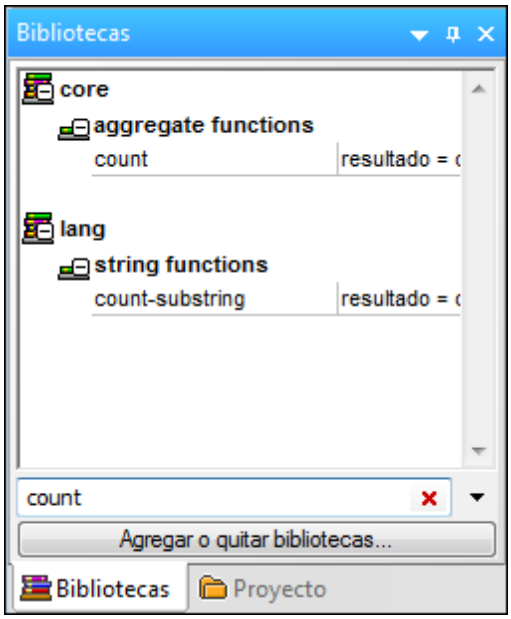
1. Seleccione el lenguaje de transformación (*véase* *Seleccionar el lenguaje de transformación*).
2. Haga clic en la función pertinente en la ventana Bibliotecas y arrástrela hasta el área de asignación.



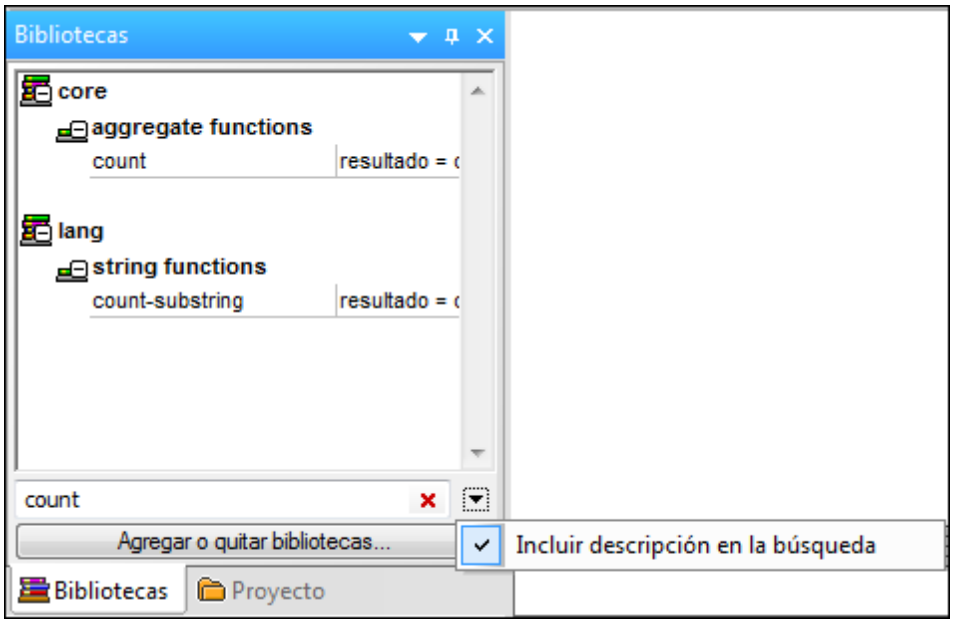
Para buscar una función en la ventana Bibliotecas:

1. Empiece a teclear el nombre de la función en el cuadro de texto situado en la parte inferior de la ventana Bibliotecas.

© 2018 Altova GmbH



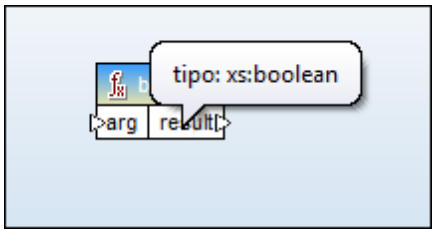
MapForce busca en el nombre de las funciones y en su descripción. Si desea excluir la descripción de las funciones de la búsqueda, haga clic en el icono en forma de flecha y deshabilite la opción **Incluir descripción en la búsqueda**.



Para cancelar la búsqueda pulse la tecla **Esc** o haga clic en el icono ×.

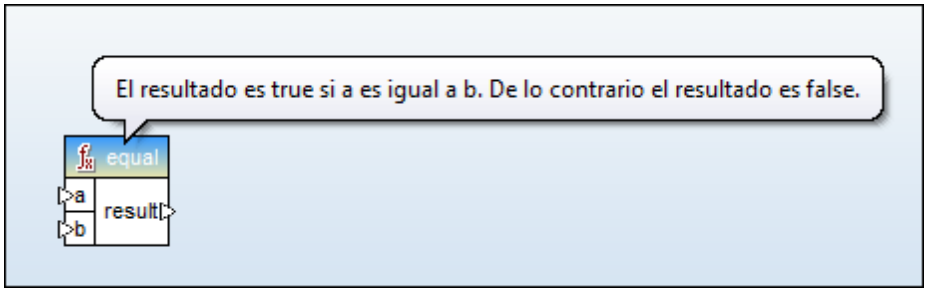
Para ver el tipo de datos de un argumento de entrada/salida de una función:

1. Compruebe que está activado el botón **Mostrar información rápida** de la barra de herramientas.
2. Pase el puntero por encima del argumento de la función.



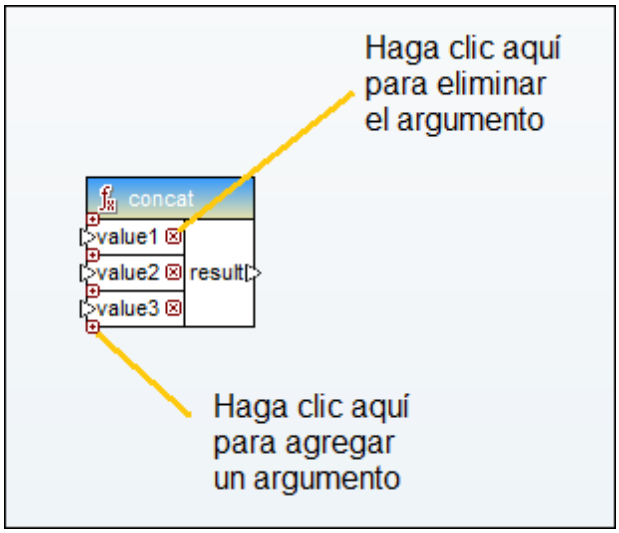
Para ver la descripción de una función:

1. Compruebe que está activado el botón **Mostrar información rápida** de la barra de herramientas.
2. Pase el puntero por encima de la función (este mecanismo funciona tanto en la ventana Bibliotecas como en el área de asignación)



Para agregar o eliminar argumentos de las funciones (si procede):

- Haga clic en el icono **Agregar parámetro** (⊡) o **Eliminar parámetro** (⊠) del parámetro que desea agregar/eliminar respectivamente.



Si arrastra y coloca una conexión en el icono ⊡, MapForce añade automáticamente el

© 2018 Altova GmbH

parámetro y lo conecta.

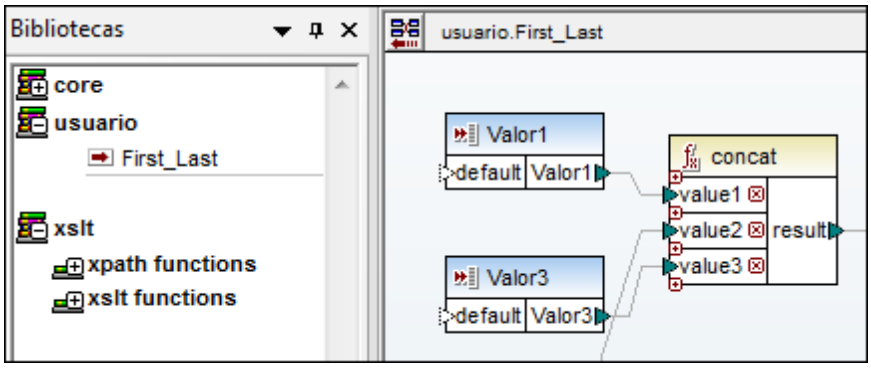
Para buscar todas las instancias de una función dentro de la asignación de datos activa:

- Haga clic con el botón derecho en el nombre de la función en la ventana Bibliotecas y elija **Buscar todas las llamadas** en el menú contextual. El resultado de la búsqueda aparece en la ventana Mensajes.

8.2 Funciones definidas por el usuario

En MapForce puede crear funciones definidas por el usuario de forma visual, con las mismas operaciones que se usan en el panel *Asignación*.

Una vez definidas, las funciones están disponibles en la ventana Bibliotecas (*ver la función* ***First_Last*** *de la imagen*) y se utilizan igual que las demás funciones de MapForce. Las funciones definidas por el usuario permiten organizar la asignación en varios bloques que se pueden volver a utilizar en la misma asignación o en otras asignaciones.



Las funciones definidas por el usuario se almacenan en el archivo ***.mfd**, junto con la asignación principal.

Una función definida por el usuario utiliza **componentes** de **entrada** y **salida** para pasar la información desde la asignación principal (o desde otra función definida por el usuario) hasta la función definida por el usuario y otra vez a la asignación principal.

Las funciones definidas por el usuario pueden tener componentes de origen locales (es decir, situados dentro de la función propiamente dicha), como por ejemplo esquemas XML o bases de datos, lo cual es práctico a la hora de implementar funciones de búsqueda.

Las funciones definidas por el usuario pueden tener un número ilimitado de entradas y salidas. Éstas pueden ser valores simples, nodos XML, bases de datos, archivos EDI o estructuras de FlexText.

Las funciones definidas por el usuario permiten:

- reunir varias funciones de procesamiento en un solo componente (p. ej. para dar formato a un campo concreto o para buscar un valor)
- reutilizar estos componentes un número ilimitado de veces
- importar funciones definidas por el usuario en otras asignaciones (cargando el archivo de asignación como biblioteca)
- usar funciones inline para descomponer una asignación compleja en varias partes más pequeñas para editarlas por separado
- asignar esquemas recursivos creando funciones definidas por el usuario recursivas

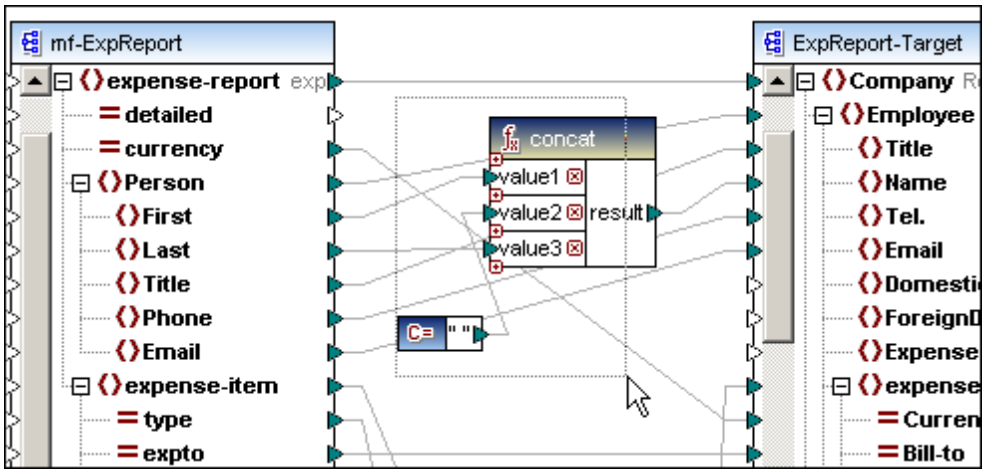
Las funciones definidas por el usuario se pueden crear desde cero o a partir de otras funciones ya disponibles en el panel *Asignación*.

El ejemplo que aparece a continuación se basa en el archivo **Tut-ExpReport.mfd** de la carpeta ...\MapForceExamples\Tutorial\.

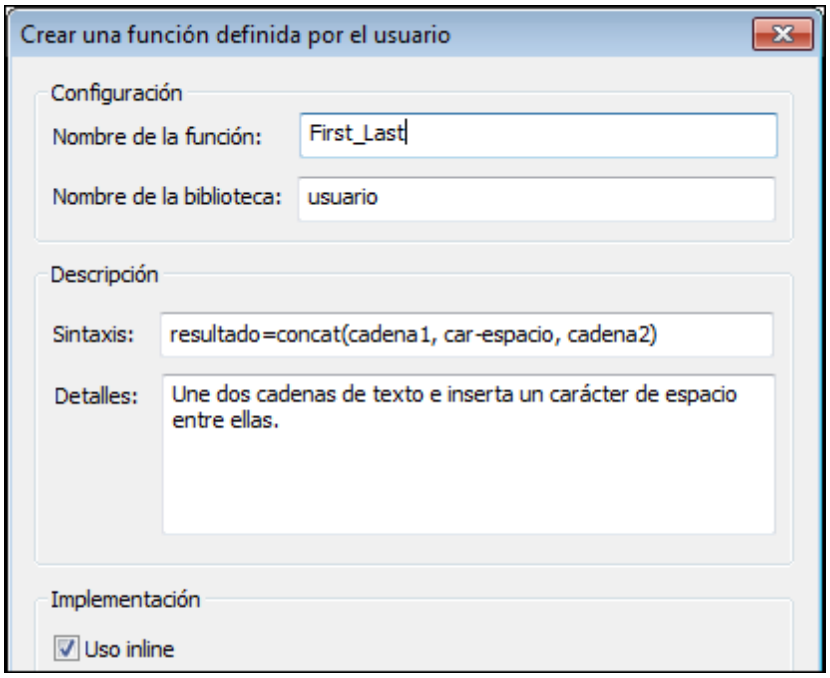
© 2018 Altova GmbH

Crear una función definida por el usuario a partir de componentes de la asignación

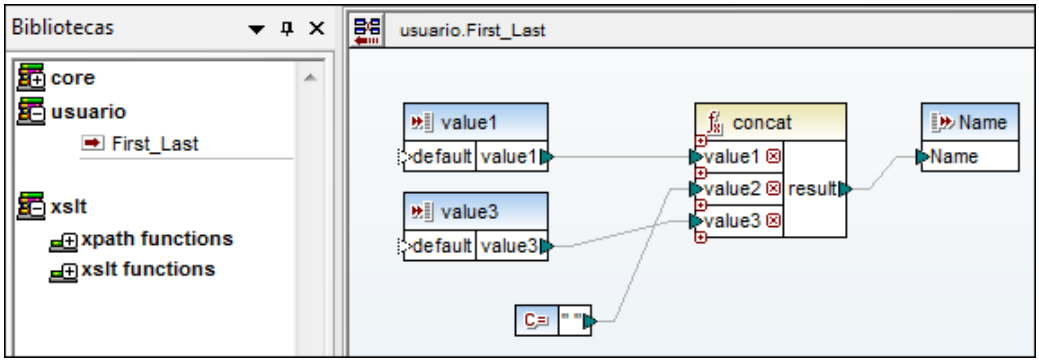
1. Arrastre el puntero por el área de asignación para seleccionar la constante y la función **concat** (también puede hacer clic en estos componentes mientras pulsa la tecla **Ctrl**).



2. Seleccione la opción de menú **Función | Crear una función definida por el usuario a partir de la selección**.
3. Aparece un cuadro de diálogo donde puede escribir el nombre de la nueva función definida por el usuario (es decir, **`First_Last`**).
   Nota: los caracteres válidos para el nombre de la función son los caracteres alfanuméricos (a-z, A-Z, 0-9), el carácter de subrayado (_), el guión (-) y los dos puntos (:).
4. En los campos *Sintaxis* y *Detalles* puede incluir más información sobre la nueva función. Cuando termine haga clic en **Aceptar**. El texto introducido en el campo *Detalles* aparece al pasar el puntero encima de la función. El nombre de biblioteca predeterminado para las funciones de usuario nuevas es **user**, pero puede crear un nombre de biblioteca nuevo en el campo *Nombre de la biblioteca*.

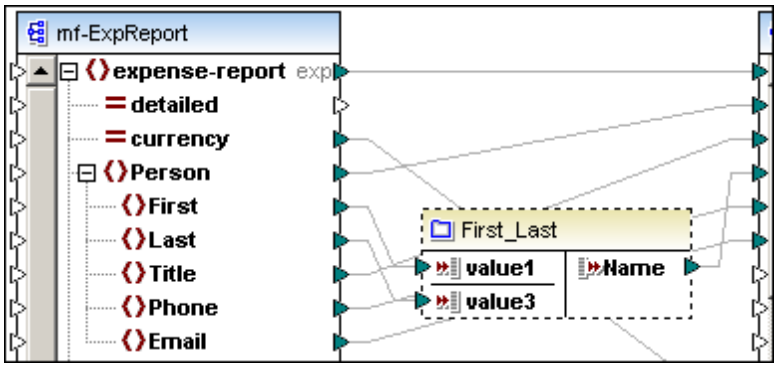


Los elementos que componen la función aparecen en un panel nuevo cuyo título es el nombre de la función. La biblioteca **usuario** aparece en el la ventana Bibliotecas y debajo el nombre de la función recién creada **`First_Last`**.



Haga clic en el botón **Volver**  para volver a la ventana de asignación principal. Observe cómo los componentes se reunieron en una sola función llamada **`First_Last`** y sus parámetros de entrada y salida se conectaron automáticamente.

© 2018 Altova GmbH



Observe que las funciones definidas por el usuario tienen un contorno de guiones. Para más información consulte el apartado Funciones definidas por el usuario estándar e inline.

Para usar la función en la asignación actual, haga clic en el nombre de la función en la ventana Bibliotecas y arrástrela hasta el panel *Asignación*. Para aprender a usarla en otras asignaciones consulte las el apartado instrucciones que aparecen más abajo

Abrir funciones definidas por el usuario

Las funciones definidas por el usuario se pueden abrir de dos maneras distintas:

- haciendo doble clic en la barra de título de la función definida por el usuario en el panel *Asignación* o
- haciendo doble clic en la función definida por el usuario en la ventana Bibliotecas.

Acto seguido se abre un panel en el que aparecen los componentes que forman la función. Para volver a la asignación principal haga clic en el botón **Volver**  . Al hacer doble clic en una función definida por el usuario de un archivo *.mfd diferente (en el panel principal de asignación) se abre ese archivo *.mfd en otra pestaña.

Navegar por las funciones definidas por el usuario

Mientras se navega por varias pestañas (o paneles de funciones definidas por el usuario), MapForce genera automáticamente un historial para que pueda recorrer las pestañas/los paneles, haciendo clic en los iconos **Adelante/Atrás**. El historial es válido para toda la sesión, es decir puede cambiar de un archivo *.mfd a otro mientras retrocede/avanza por las pestañas.

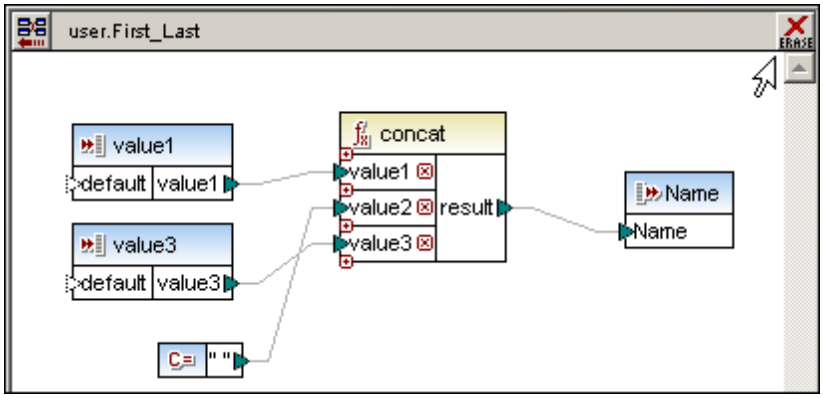
 Pulse el botón **Volver** para volver a la pestaña de asignación principal.

 Pulse el botón **Atrás** para retroceder en el historial.

 Pulse el botón **Adelante** para avanzar en el historial.

Eliminar funciones definidas por el usuario en la biblioteca

1. Haga doble clic en la función definida por el usuario en la ventana Bibliotecas.
2. Haga clic en el botón **Borrar** situado en la esquina superior derecha del panel.



Reutilizar (importar) funciones definidas por el usuario

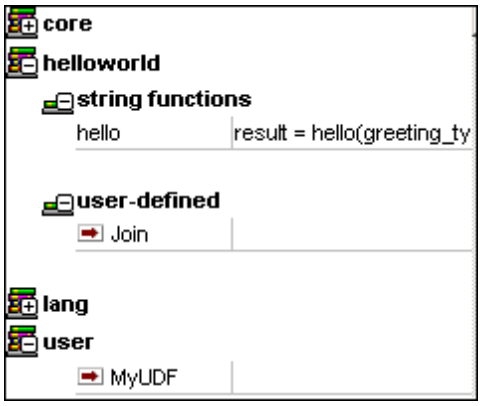
Las funciones definidas por el usuario en una asignación se pueden importar en otra asignación diferente:

1. Haga clic en el botón **Agregar o quitar bibliotecas** en la parte inferior de la ventana Bibliotecas.
2. En el cuadro de diálogo que aparece haga clic en el botón **Agregar** y seleccione el archivo *.mfd que contiene las funciones definidas por el usuario que desea importar. La función definida por el usuario aparece ahora en la ventana Bibliotecas. Si creó la función definida por el usuario con el nombre de biblioteca predeterminado, la biblioteca importada se llamará **user**. Si no se llama así, busque el nombre de biblioteca que especificó cuando creó la función definida por el usuario.
2. Arrastre la función que acaba de importar desde la ventana Bibliotecas hasta el área de asignación.

> Si se usa el mismo nombre de biblioteca en varios archivos *.mfd o en varias bibliotecas personalizadas (véase Importar bibliotecas Java y .NET personalizadas), todas las funciones de todas las fuentes disponibles aparecerán bajo el mismo nombre de biblioteca (en la ventana Bibliotecas). Sin embargo, solamente se pueden editar con doble clic las funciones del documento que esté activo en ese momento.

Aquí puede ver un ejemplo:

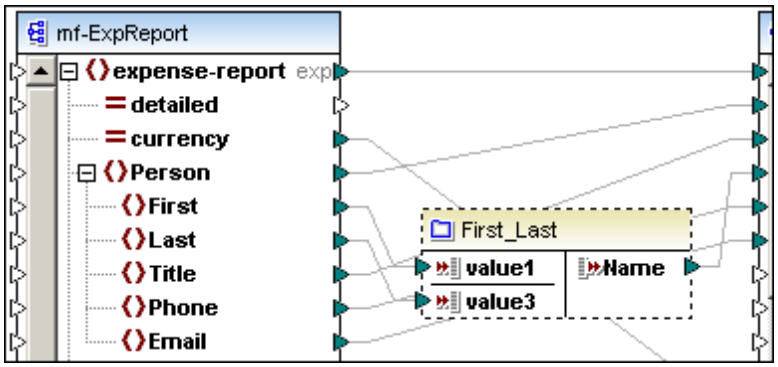
© 2018 Altova GmbH



- la función **`hello`** de la biblioteca **`helloworld`** se importó de una biblioteca personal
- la función **`Join`** de la biblioteca **`helloworld`** es una función definida por el usuario presente en el archivo *.mfd actual
  - la función **`MyUDF`** de la biblioteca **`user`** también es una función definida por el usuario presente en el archivo *.mfd actual

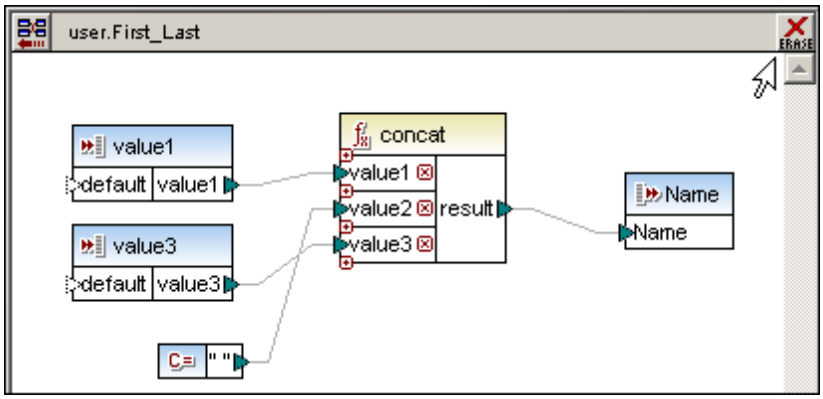
No olvide que, si realiza cambios en una función importada, cuando guarde el archivo ***.mfd** los cambios se aplicarán también a las asignaciones desde las que se importó la función.

Orden de los parámetros en funciones definidas por el usuario



El orden de los parámetros de las funciones definidas por el usuario depende de varios aspectos:

- los parámetros de entrada y salida se ordenan según su posición, de arriba a abajo (empezando por la esquina superior izquierda).
- si dos parámetros tienen la misma posición vertical, tiene prioridad el que esté situado más a la izquierda.
- en el improbable caso que dos parámetros tengan exactamente la misma posición, el identificador interno del componente se usa automáticamente.



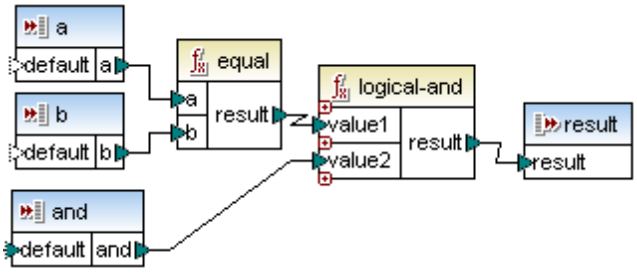
Notas:

- la posición del componente y las acciones de ajuste de tamaño no se pueden deshacer.
- los componentes de entrada y salida recién añadidos se crean bajo la entrada y salida creada inmediatamente antes.
- puede mezclar parámetros complejos y simples. El orden de los parámetros se deduce de la posición del componente.

8.2.1 Parámetros de las funciones

Los **parámetros** de las funciones se representan dentro de las funciones definidas por el usuario en forma de **componentes** de **entrada** y **componentes** de **salida**.

- Componentes/parámetros de entrada: **a**, **b**, **and**
- Componentes/parámetros de salida: **result**

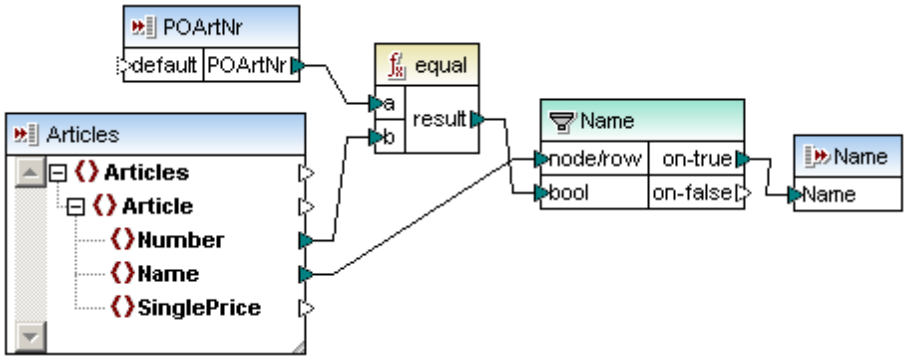


Los parámetros de entrada se utilizan para pasar datos de la asignación principal a la función definida por el usuario. Los parámetros de salida se utilizan para devolver los datos a la asignación principal. Recuerde que a las funciones definidas por el usuario se les puede llamar desde otras funciones definidas por el usuario.

Parámetros simples y complejos
Los parámetros de entrada y salida de las funciones definidas por el usuario pueden ser de varios tipos:

- Valores simples (p. ej. `string` o `integer`)
- Árboles de nodos complejos (p. ej. un elemento XML con atributos y nodos secundarios)

 © 2018 Altova GmbH



El parámetro de entrada **`POArtNr`** es un valor **simple** de tipo `string`

El parámetro de entrada **`Articles`** es un árbol de nodos XML **complejo** (un documento XML)

El parámetro de salida **`Name`** es un valor simple de tipo `string`

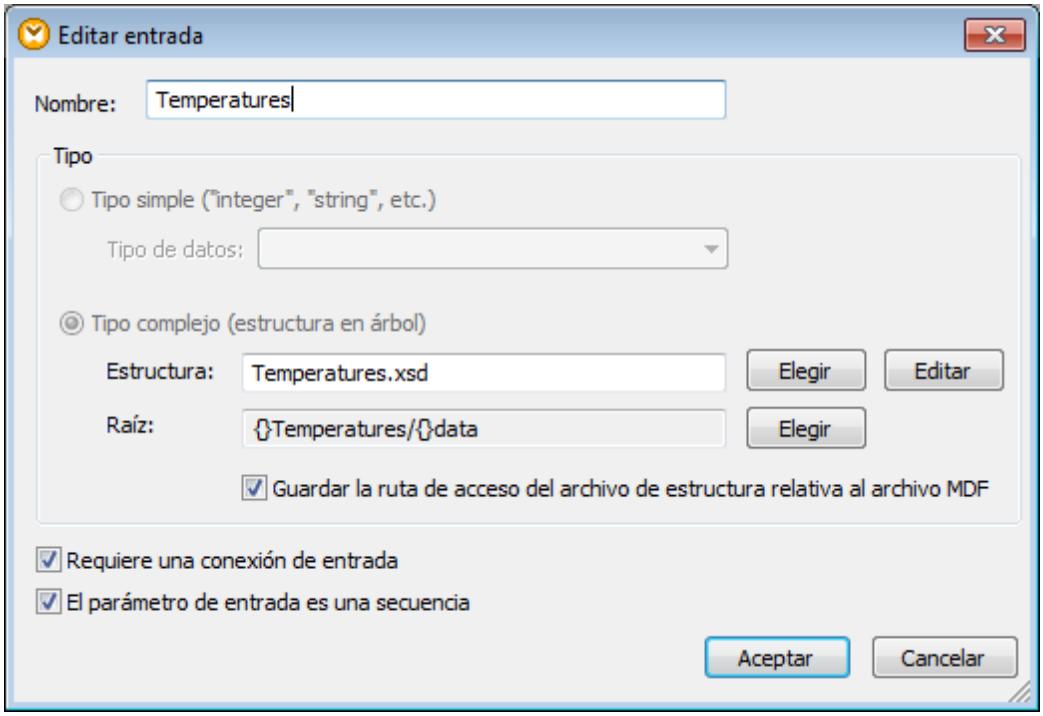
Nota: las funciones definidas por el usuario de las dos imágenes anteriores están disponibles en el archivo de ejemplo **`PersonListByBranchOffice.mfd`** , guardado en la carpeta `. . . \MapForceExamples`.

Secuencias

Las secuencias son datos compuestos por un rango (o secuencia) de valores. Los **parámetros** simples y complejos de las funciones definidas por el usuario (sus entradas y salidas) se pueden definir como secuencias en el cuadro de diálogo de propiedades del componente.

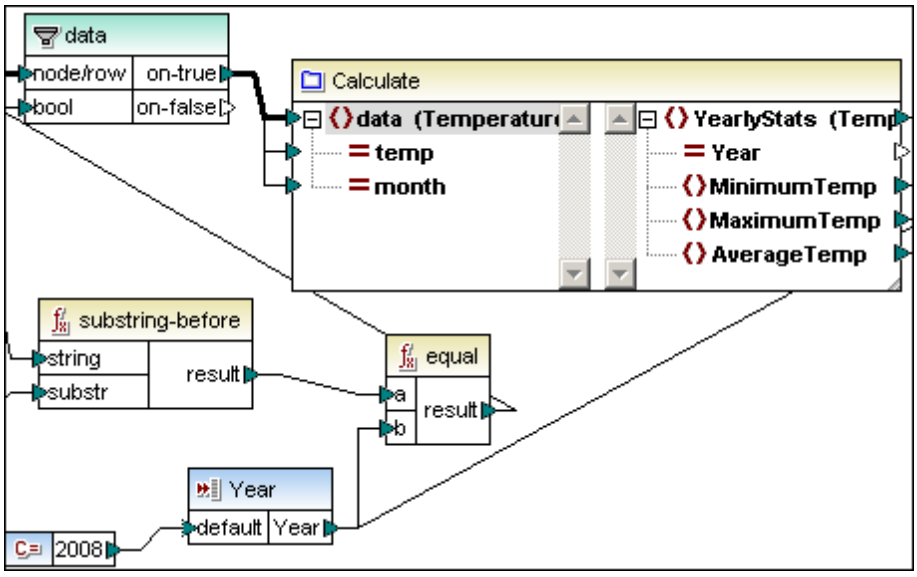
En las funciones de agregado (p. ej. **`min`**, **`max`**, **`avg`**, etc.) puede usar este tipo de entradas para suministrar un valor determinado de la secuencia de entrada.

Si activa la casilla *El parámetro de entrada es una secuencia,* el componente trata la entrada como si fuera una secuencia de valores. Si desactiva la casilla, la entrada se trata como si fuera un valor único.



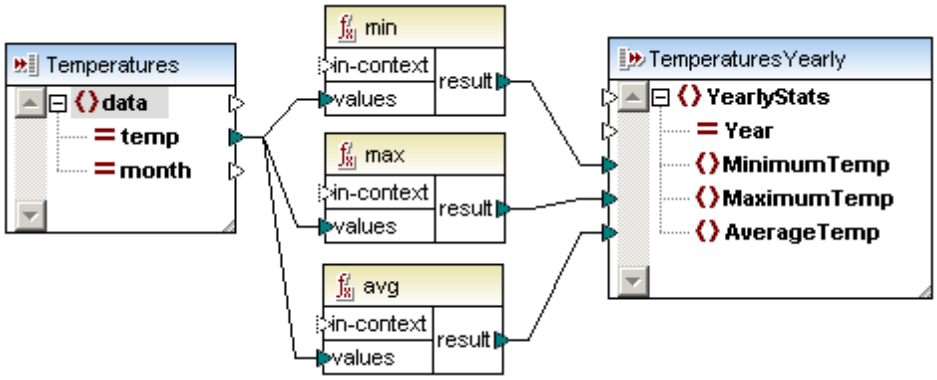
Este tipo de datos de entrada (de secuencia o no secuencia) determina **con qué frecuencia** se llama a la función.

- Cuando la función definida por el usuario está conectada a un parámetro **secuencial**, se le llama **una sola vez** y toda la secuencia se pasa a la función definida por el usuario.



La imagen anterior muestra la función definida por el usuario **`Calculate`**, del archivo de asignación `InputIsSequence.mfd` (disponible en la carpeta `...\MapForceExamples`). El componente de entrada **`Temperatures`** (*imagen siguiente*) se definió como secuencia.

 © 2018 Altova GmbH



- Cuando la función definida por el usuario está conectada a un parámetro **no secuencial**, se le llama **una vez por cada elemento** de la secuencia.

Nota: la configuración secuencial de los parámetros de entrada/salida se ignora cuando la función definida por el usuario es de tipo inline.

Es importante tener en cuenta que, si conecta una secuencia vacía a un parámetro no secuencial, a la función no se le llama en absoluto.

Esto puede ocurrir si la estructura de origen tiene elementos **opcionales** o cuando una condición de un filtro no devuelve elementos. Para evitar que esto ocurra utilice la función substitute-missing antes de la entrada de la función (evitando que la secuencia esté vacía) o defina la entrada como secuencia y defina el procesamiento de secuencias vacías dentro de la función.

Cuando una función pasa una secuencia de varios valores a su componente de salida y el componete de salida no se definió como secuencia, solamente se usa el primer resultado cuando se llama a la función.

8.2.2 Funciones definidas por el usuario estándar e inline

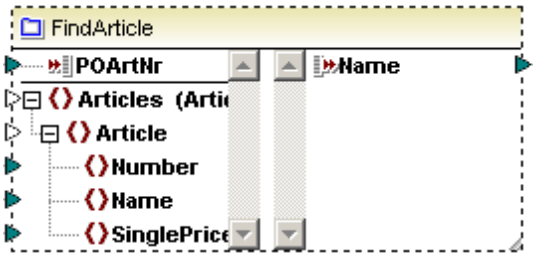
Las funciones inline son diferentes de las demás funciones por la manera en que se implementan cuando se genera código.

- El código para las funciones de tipo **inline** se inserta en **todas las posiciones** donde se llama a las funciones definidas por el usuario o donde se utilizan
- El código de una función **normal** se implementa como una llamada a función

Por tanto, es como si las funciones inline fueran sustituidas por su implementación. Por este motivo son ideales para descomponer asignaciones complejas en varias partes más pequeñas.

Nota: el uso de funciones **inline** incrementa significativamente la cantidad de código de programa generado porque el código de la función definida por el usuario se inserta en todas las posiciones donde se llama a la función o donde se utiliza.

Las **funciones definidas por el usuario** de tipo **INLINE** tienen un contorno de guiones:



Las funciones definidas por el usuario inline **admiten:**

- **varios componentes de salida**

Las funciones definidas por el usuario inline **no admiten:**

- parámetros configurados como contexto prioritario
- llamadas recursivas a una función definida por el usuario inline

Las **funciones definidas por el usuario** de tipo **ESTÁNDAR** (es decir, no inline) tienen un contorno sólido:



Las funciones definidas por el usuario de tipo estándar (no inline) **admiten:**

- un solo componente de salida
- **llamadas recursivas** (aquellas en las que se debe dar la condición de salida, p. ej. usar una condición If-Else en la que una rama o valor sale de la recursión)
- parámetros configurados como contexto prioritario

Nota: aunque las funciones estándar no admiten varios componentes de salida, estos componentes se pueden crear en este tipo de función. Sin embargo, cuando intente generar código o ver una vista previa del resultado de la asignación, aparecerá un mensaje de error en la ventana Mensajes.

Si no usa una recursión en su función, puede cambiar el tipo de función a inline.

Las funciones definidas por el usuario de tipo estándar (no inline) **no admiten:**

- la conexión directa de filtros a componentes de **entrada** simples no secuenciales
- el uso de funciones de secuencia o agregado en componentes de entrada simples (como **`exists`**, **`substitute-missing`**, **`sum`**, **`group-by`**, etc.)

Nota sobre generación de código
La implementación de una función definida por el usuario de tipo estándar se genera una sola vez como función o plantilla XSLT a la que se puede llamar. Cada componente de la función definida por el usuario genera código para una **llamada a función**, en la que las entradas se pasan como parámetros y la salida es el valor devuelto de la función (componente).

En tiempo de ejecución se evalúan primero todos los valores del parámetro de entrada y después se llama a la función cada vez que aparecen los datos de entrada. Para más información consulte el apartado Parámetros de las funciones.

© 2018 Altova GmbH

Para cambiar de tipo de función definida por el usuario:

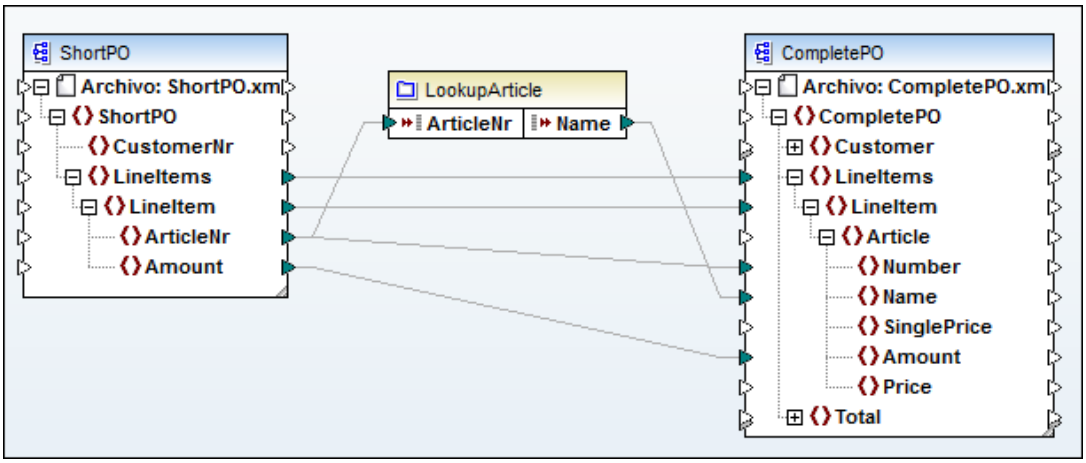
1. Haga doble clic en la función definida por el usuario para ver sus componentes.
2. Seleccione la opción de menú **Función | Configuración de la función** y active/desactive la casilla *Uso inline*.

Funciones definidas por el usuario y conexiones de copia total
Cuando cree conexiones de copia total entre un esquema y un parámetro complejo de una función definida por el usuario, ambos componentes deben estar basados en el mismo esquema. Sin embargo, no es necesario que tengan el mismo elemento raíz. Para ver un ejemplo consulte el apartado Definir componentes de salida complejos.

8.2.3 Crear una función de búsqueda simple

Este apartado utiliza el archivo de ejemplo `lookup-standard.mfd`, disponible en la carpeta **...\MapForceExamples**.

Objetivo:
Crear una función de búsqueda genérica que suministre los datos de **`Articles/Number`** del archivo XML **Articles** para compararlos con los datos de **`Article`** de otro archivo XML, el archivo **ShortPO**.



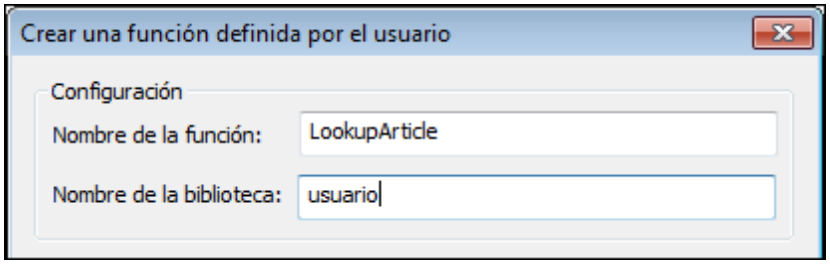
Instrucciones generales:

- Inserte el esquema `ShortPO.xsd` y asigne el archivo `ShortPO.xml` como archivo XML de entrada.
- Inserte el esquema `CompletePO.xsd` y seleccione el elemento raíz **`CompletePO`**.
- Inserte una función definida por el usuario nueva siguiendo los pasos descritos a continuación.

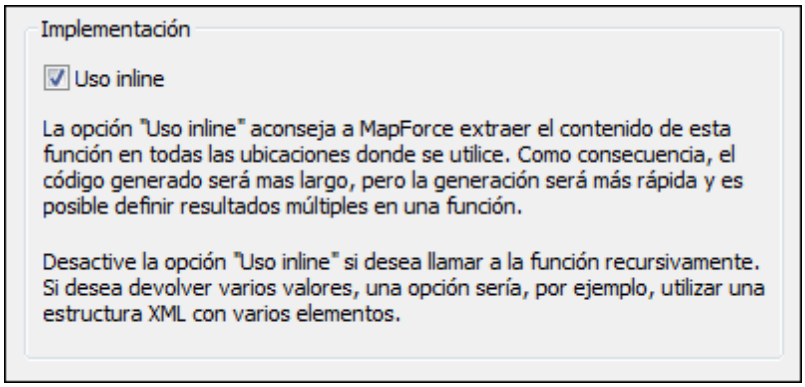
Para crear una función definida por el usuario:

1. Seleccione la opción de menú **Función | Crear función definida por el usuario**.
2. En el cuadro de diálogo que aparece escriba el nombre de la función, p. ej.

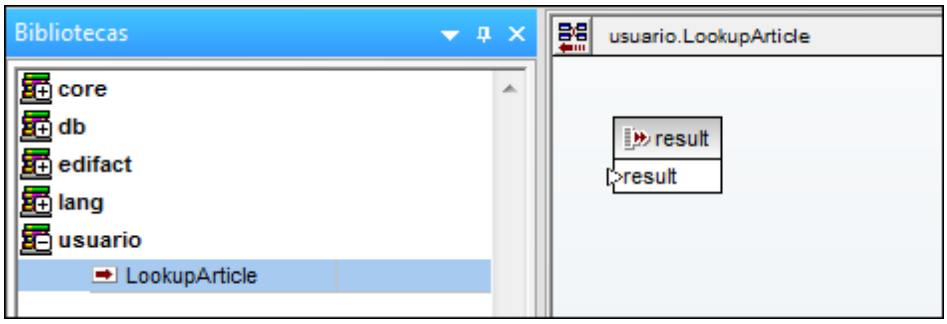
LookupArticle, y la biblioteca en la que desea almacenarla (p. ej. **usuario**).



3. Desactive la casilla *Uso inline* y haga clic en **Aceptar** para confirmar.



Aparece un panel con un solo elemento: un parámetro de salida llamado **result**.

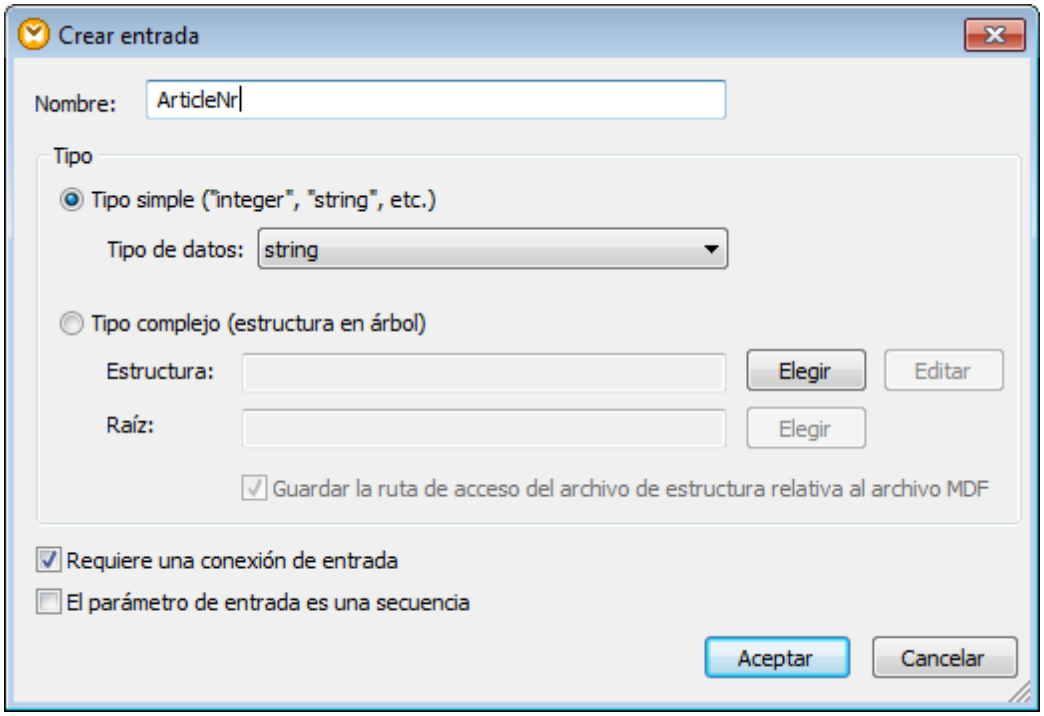


Este panel es el área de trabajo donde podrá definir la función definida por el usuario.

En la ventana **Bibliotecas** se creó una biblioteca nueva llamada **usuario** y bajo ella aparece la función **LookupArticle**.

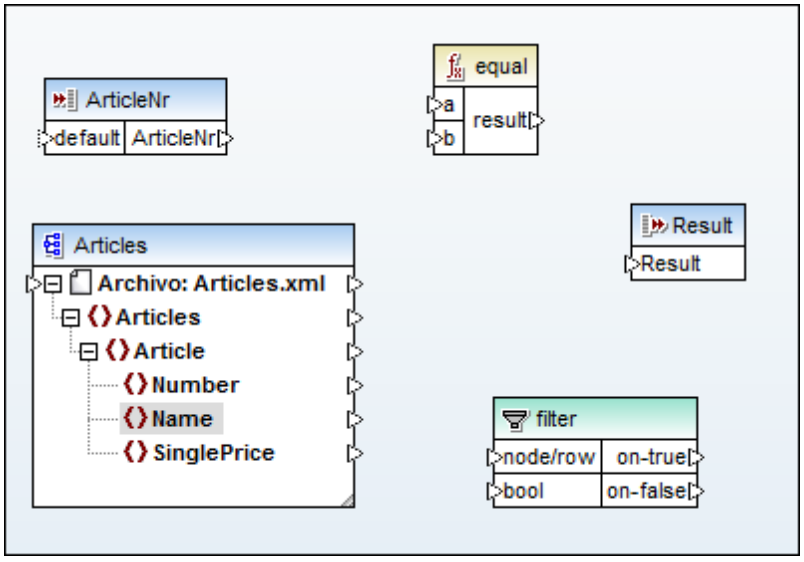
4. Haga clic en el icono **Insertar esquema o archivo XML** para insertar el esquema **Articles** y seleccionar el archivo XML **Articles** como origen de datos.
5. Haga clic en el icono **Insertar componente de entrada** para insertar un componente de entrada.
6. Escriba el nombre del parámetro de entrada (**ArticleNr**) y haga clic en **Aceptar**.

© 2018 Altova GmbH



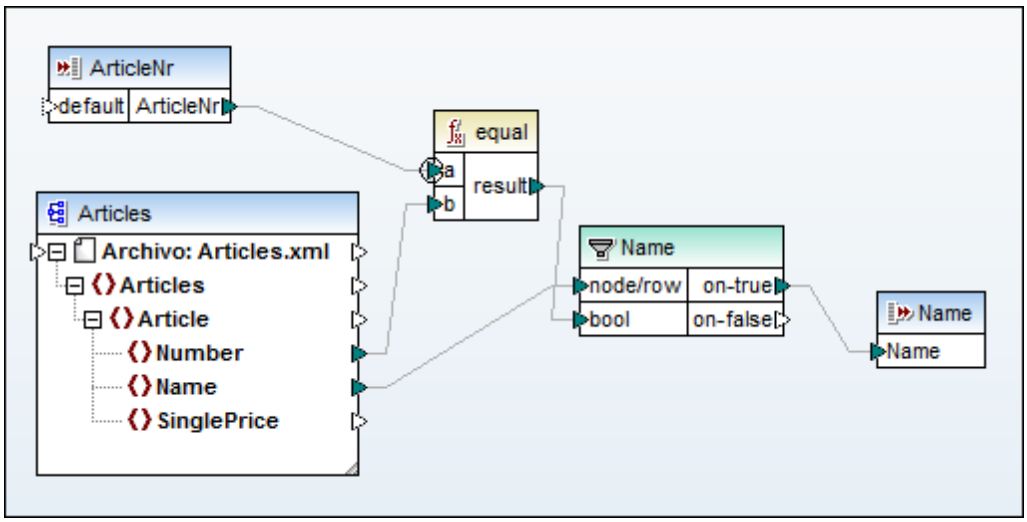
Este componente funciona como entrada de datos para la función definida por el usuario y constituye el icono de entrada de la función definida por el usuario.

7. Inserte la función **equal** de la biblioteca **core | logical**.
8. Inserte un componente de filtrado haciendo clic en el icono **Insertar filtro** de la barra de herramientas.



Ahora cree las asignaciones que aparecen en la imagen siguiente entre los componentes de la función definida por el usuario.

9. Tras crear las asignaciones haga clic con el botón derecho en el parámetro **a** de la función **equal** y seleccione **Contexto prioritario** en el menú contextual.
10. Haga **doble clic** en la salida de la función y escriba el nombre **Name** para el parámetro de salida.



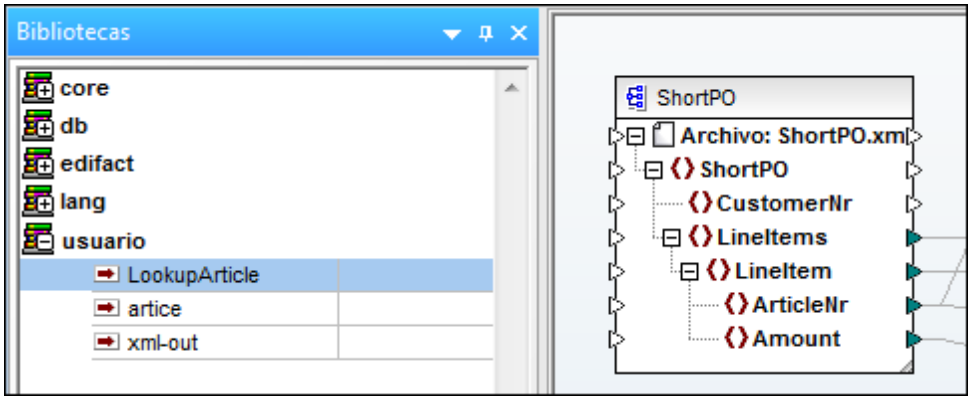
La función ya está definida.

Nota: al hacer doble clic en las entradas y salidas de la función aparece un cuadro de diálogo donde puede: (i) cambiar el nombre y el tipo de datos del parámetro, (ii) definir si la función debe tener un icono de entrada (*Requiere una conexión de entrada*) y (iii) definir el parámetro como secuencia.

La función definida por el usuario recién creada:

- tiene una **entrada** (**`ArticleNr`**) que recibirá los datos del archivo XML **ShortPO** en la asignación principal.
- **compara** los datos de **`ArticleNr`** del archivo XML **ShortPO** con los datos de **`Article/Number`** del archivo XML **Articles** (que se insertó en la función con dicho objetivo).
- utiliza un **filtro** para enviar los registros de **`Article/Name`** al componente de salida si el resultado de la comparación es **`true`**.
- tiene una **salida** (**`Name`**) que enviará los registros de **`Article/Name`** al archivo XML de destino **CompletePO**.

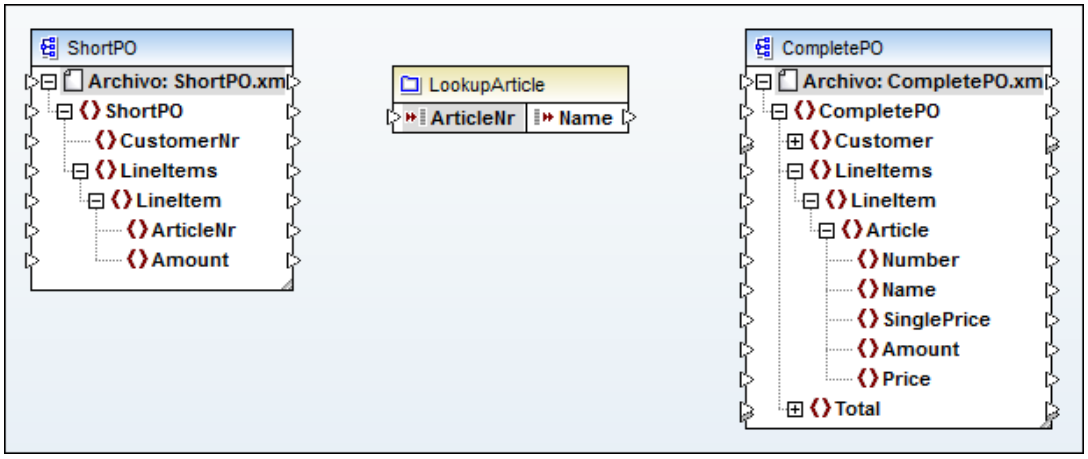
11. Haga clic en el icono **Volver** para volver a la asignación principal.
La función definida por el usuario **`LookupArticle`** ahora está disponible en la biblioteca **usuario**.



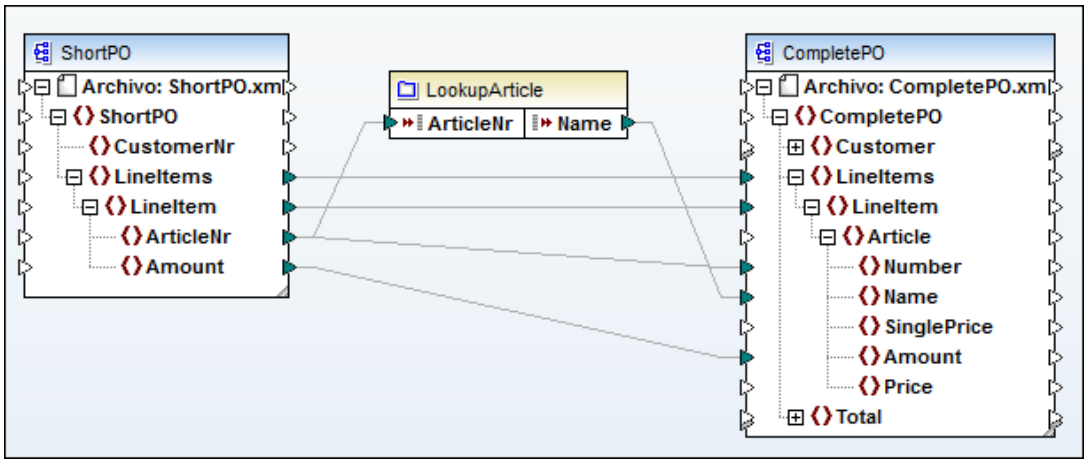
 © 2018 Altova GmbH

12. Arrastre la función **`LookupArticle`** hasta el panel **Asignación**.

    La función definida por el usuario aparece:
    - con el nombre **`LookupArticle`** en la barra de título
    - con los iconos de entrada y salida **`ArticleNr`** y **`Name`** respectivamente.



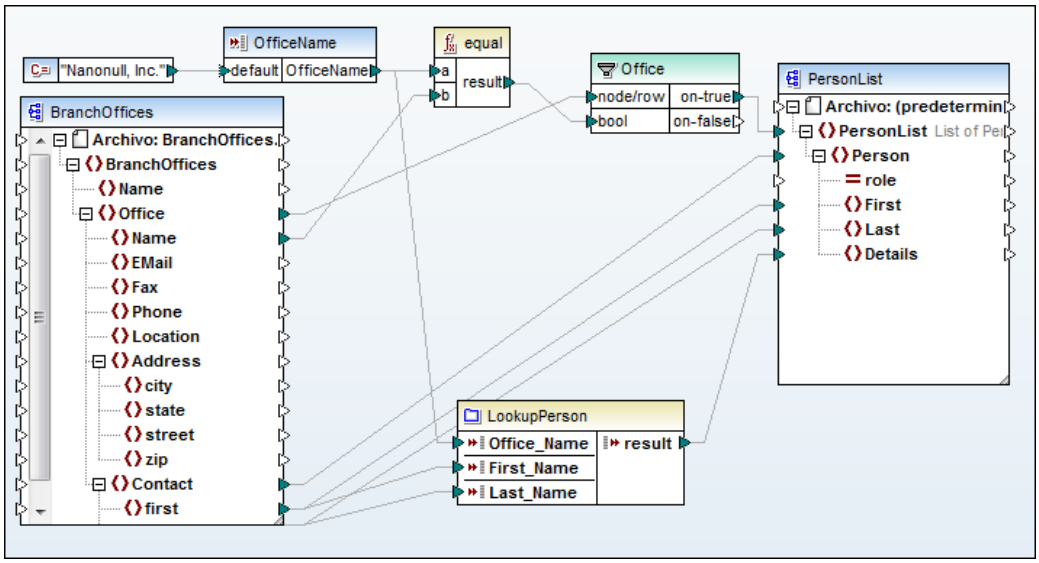
13. Cree las conexiones que aparecen en la imagen siguiente y abra el panel *Resultados* para ver el resultado de la asignación.



8.2.4 Ejemplo de función definida por el usuario

El archivo **PersonListByBranchOffice.mfd** de la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples** incluye estas características:

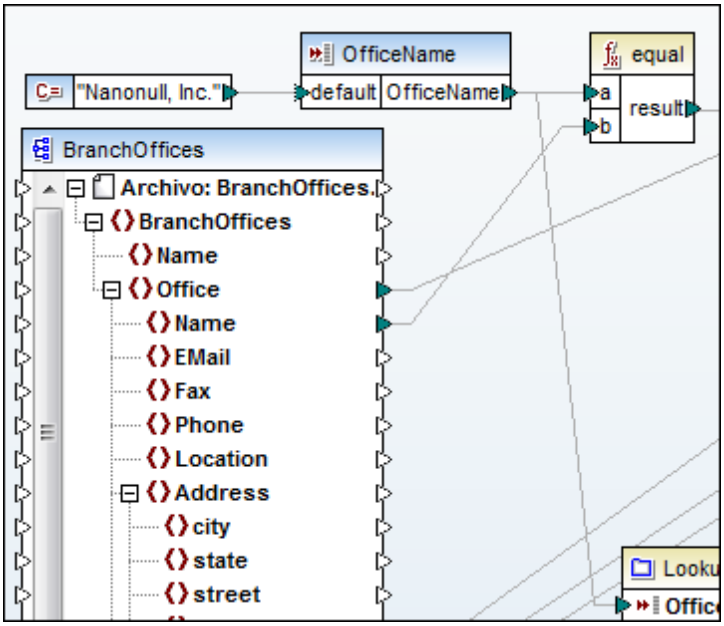
- Funciones definidas por el usuario anidadas (p. ej. **LookupPerson**).
- Funciones de búsqueda que generan un resultado de cadena (p. ej. **LookupPerson**).
- Parámetros de entrada **opcionales** que también pueden dar un valor **predeterminado** (p. ej. el componente **EqualAnd** que está en el componente LookupPerson).
- Parámetros de entrada **configurables** que también pueden hacer de parámetros de la línea de comandos cuando se ejecute el código de asignación generado.



Parámetros de entrada configurables

El componente de entrada (OfficeName) recibe datos cuando se ejecuta la asignación. Esto es posible de dos maneras:

- como parámetro de la **línea de comandos** cuando se ejecuta el código generado (p. ej. Mapping.exe /OfficeName "Nanonull Partners, Inc.")
- como valor de **vista previa** cuando se usa el motor de ejecución integrado para obtener una vista previa de los resultados en el panel *Resultados*.



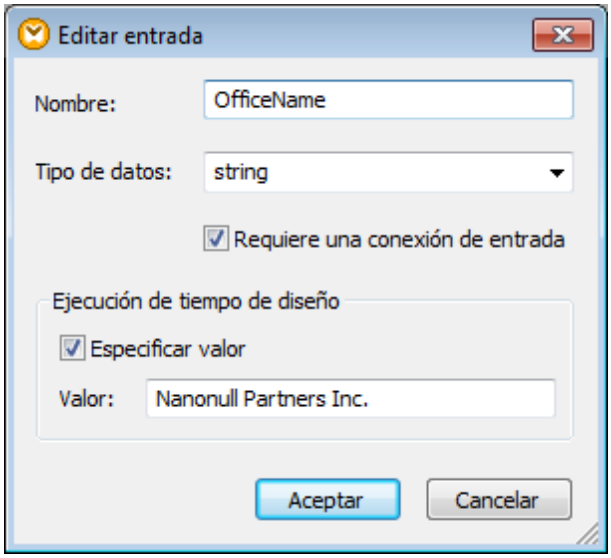
 © 2018 Altova GmbH

Para definir el valor de entrada:

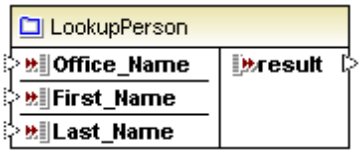
1. Haga doble clic en el componente de entrada e introduzca un valor distinto en el cuadro de texto *Valor* del grupo *Ejecución de tiempo de diseño* (p. ej. "Nanonull Partners, Inc.". Haga clic en **Aceptar** para confirmar.
2. Haga clic en el panel *Resultados* para ver el efecto.
   Ahora aparece un conjunto de personas distinto.

   Recuerde que los datos que introduzca en este cuadro de diálogo solamente se usan en el modo de vista previa (es decir, cuando se hace clic en el panel *Resultados*). Si no introduce ningún valor o si desactiva la casilla *Especificar valor*, entonces se usarán los datos que estén asignados al icono de entrada predeterminado.

   Consulte el apartado Componentes de entrada para obtener más información.



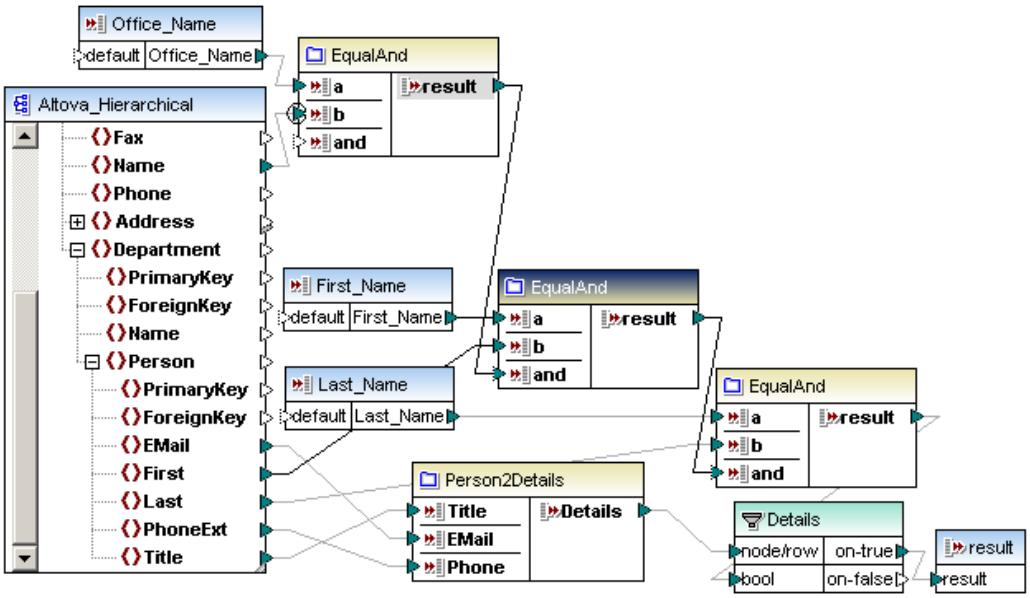
Componente LookupPerson



Si hace doble clic en este componente, podrá ver los componentes que lo constituyen (*imagen siguiente*). Este componente se encarga de:

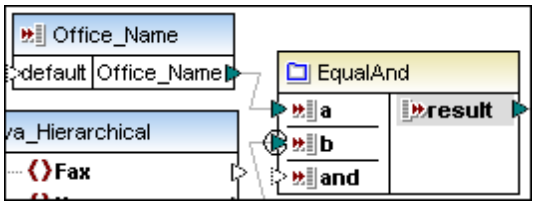
- **Comparar** los elementos Office, First y Last Name de BranchOffices.xml con los mismos campos del archivo Altova_Hierarchical.xml file (para la comparación usa los componentes de **entrada** y la función definida por el usuario **EqualAnd**).
- **Combinar** los nodos Email, PhoneExt y Title con ayuda de la función definida por el usuario **Person2Details**.
- **Pasar** los datos combinados al componente de **salida** si todas las comparaciones EqualAnd previas dan como resultado true (es decir, si pasan el valor true al componente de filtrado).

Las funciones definidas por el usuario siempre tienen un valor de salida, que incluso puede ser una cadena vacía. Este sería el caso si el valor binario del componente de filtrado fuera false. En ese caso el resultado de la función sería una cadena vacía, en lugar de los datos que suministra el componente Person2Details.



- Los tres componentes de **entrada**, Office_Name, First_Name, Last_Name, reciben datos del archivo BranchOffices.xml.
- El componente **EqualAnd** compara dos valores y suministra un valor de comparación **opcional**, así como un valor predeterminado.
- Person2Details combina tres campos personales y pasa el resultado al componente de filtrado.

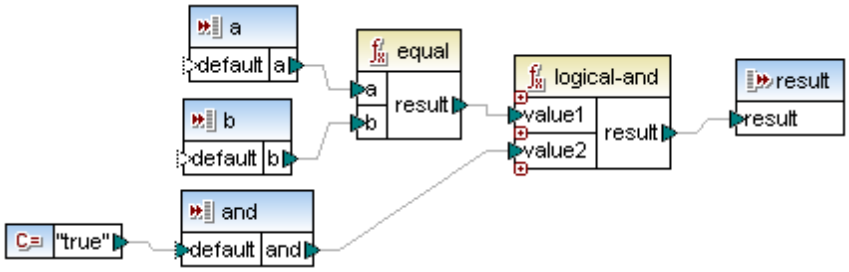
Componente EqualAnd



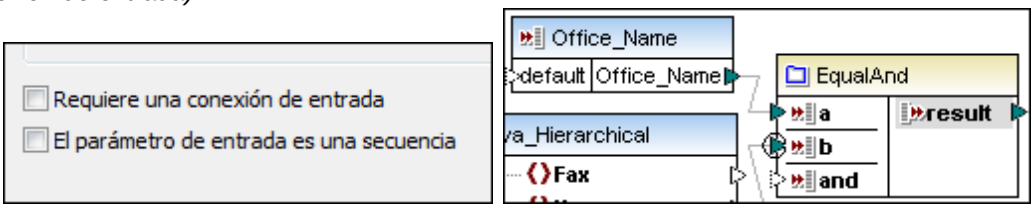
Si hace doble clic en este componente, podrá ver los componentes que lo constituyen (*imagen siguiente*). Este componente se encarga de:

- Comparar dos parámetros de entrada (**a** y **b**) y pasar el resultado al componente logical-and. Observe que el parámetro **b** se definió como **contexto prioritario**. Esto garantiza que se procesen primero los datos personales de la oficina que vienen dados por el parámetro de entrada **a** se procesen.
- Combinar en **Logical-and** el resultado de la primera comparación con un parámetro de entrada **opcional** ("and").
- Pasar el valor binario de esta comparación al parámetro de salida.

 © 2018 Altova GmbH

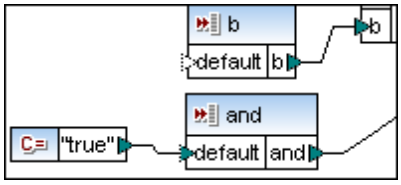


Parámetros opcionales
Si hace doble clic en el parámetro "and" de la función definida por el usuario **EqualAnd** del ejemplo anterior, podrá decidir si el parámetro es opcional o no (con la casilla *Requiere una conexión de entrada)*.



Si la casilla *Requiere una conexión de entrada* está desactivada:

- No es necesario que el icono de entrada de esta función definida por el usuario esté conectado (p. ej. el parámetro **and** de la primera función EqualAnd no tiene un conector de entrada). El icono de entrada tiene un contorno de guiones que indica que es opcional.
- Puede suministrarse un valor **predeterminado** con solo conectar un componente dentro de la función definida por el usuario (p. ej. usando un componente de constante que contenga el valor "true").



- La asignación entre otro elemento y la entrada opcional tiene prioridad sobre el valor predeterminado. Por ejemplo, el parámetro "and" de la segunda función EqualAnd recibe datos de entrada del parámetro "result" de la primera función EqualAnd definida por el usuario.

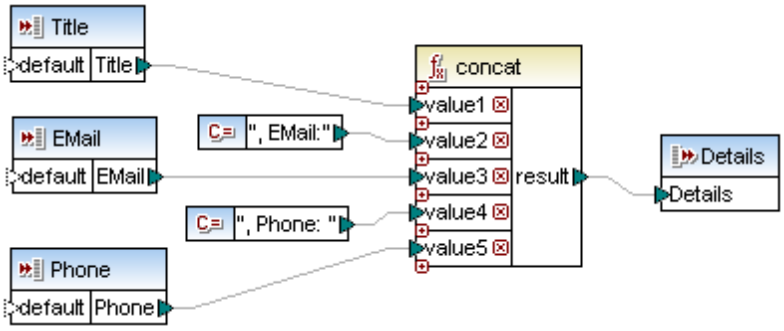
Componente Person2Details



Si hace doble clic en este componente, podrá ver los componentes que lo constituyen (*imagen*

siguiente). Este componente se encarga de:

- Encadenar tres entradas y pasar la cadena resultante al parámetro de salida.
- Puede cambiar el nombre del parámetro y seleccionar el tipo de datos con solo hacer doble clic en el parámetro de salida.



8.2.5 Función definida por el usuario compleja: nodo XML como entrada

Este apartado utiliza el archivo de ejemplo `lookup-udf-in.mfd`, disponible en la carpeta [...\MapForceExamples](). A continuación explicamos cómo se define una función definida por el usuario de tipo inline con un componente de entrada complejo.

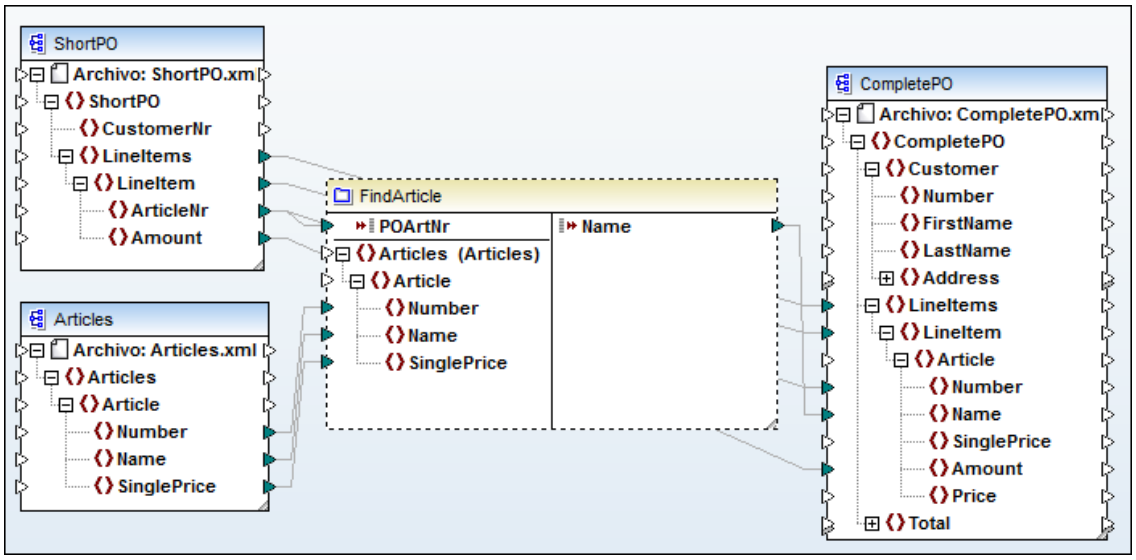
Tenga en cuenta que la función definida por el usuario **`FindArticle`** tiene dos partes.

La parte de la izquierda contiene los parámetros de entrada:

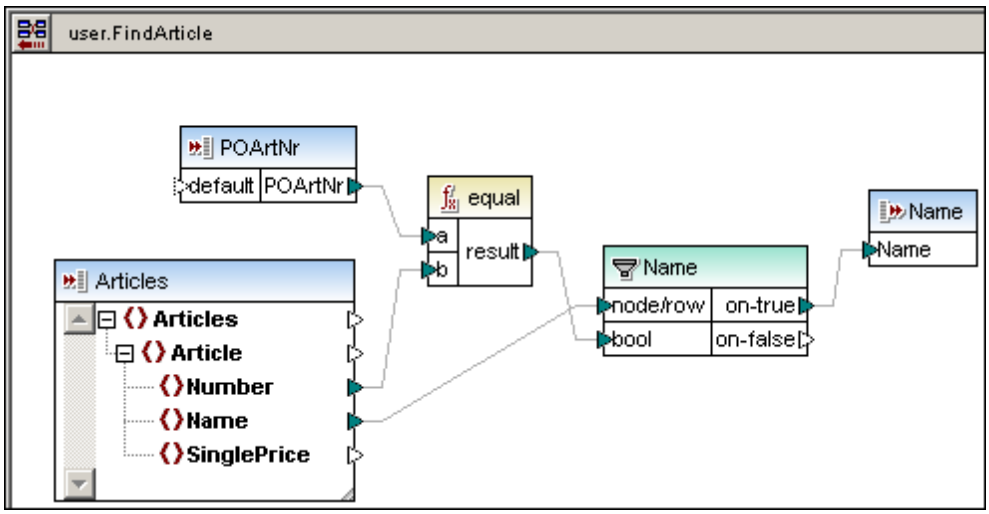
- un parámetro de entrada simple llamado **`POArtNr`**
- un parámetro de entrada complejo llamado **`Articles`**, cuyos nodos XML secundarios tienen asignaciones directas a otros componentes de la asignación

La parte de la derecha contiene:

- un parámetro de salida simple llamado **`Name`**.



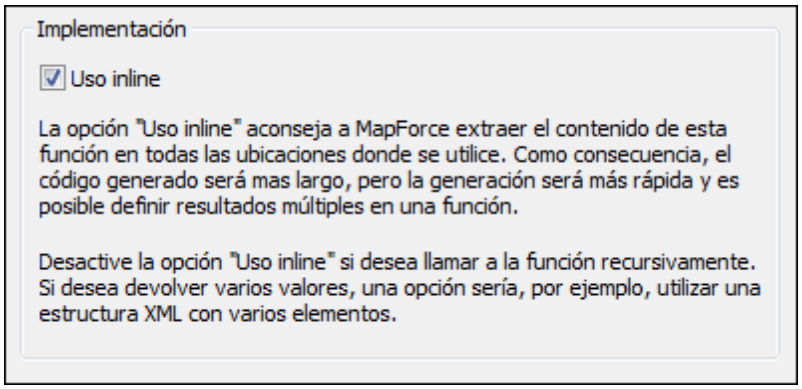
A continuación puede ver los componentes que constituyen la función definida por el usuario, con los dos componentes de entrada a la izquierda y el de salida a la derecha.



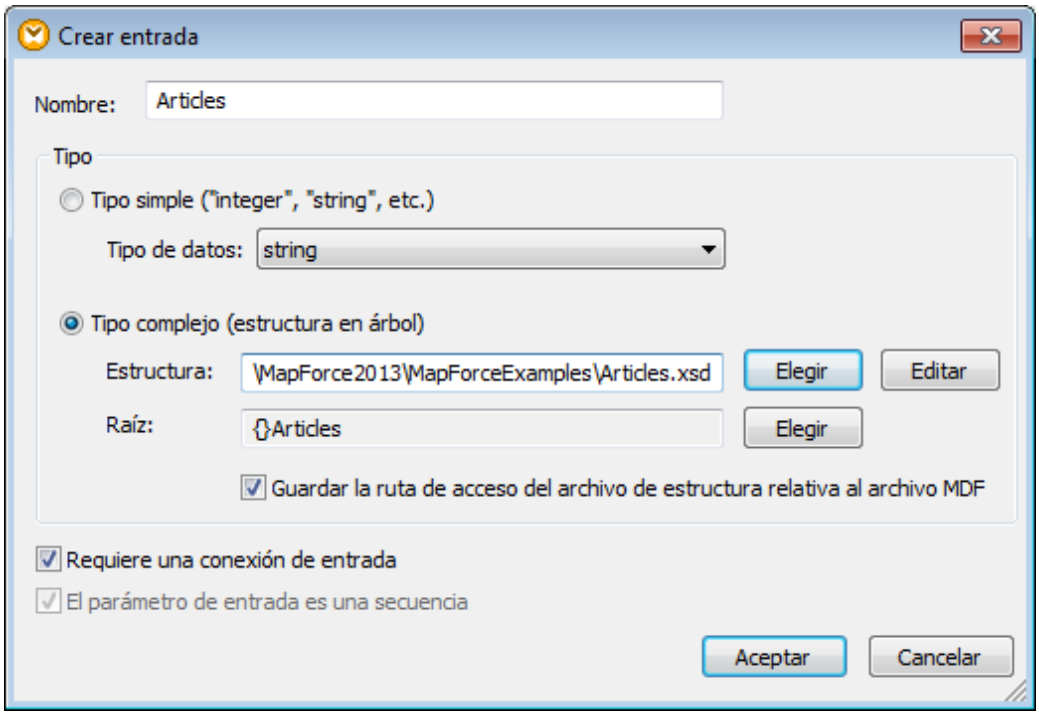
8.2.5.1 Definir componentes de entrada complejos

Siga estas instrucciones para crear una función que tome una estructura XML como parámetro de entrada:

1. Cree una función definida por el usuario (**Función | Crear una función definida por el usuario**). Recuerde que la opción *Uso inline* está activada por defecto.



2. Haga clic en el icono **Insertar componente de entrada** de la barra de herramientas.
3. Escriba el nombre del componente de entrada en el campo *Nombre*.

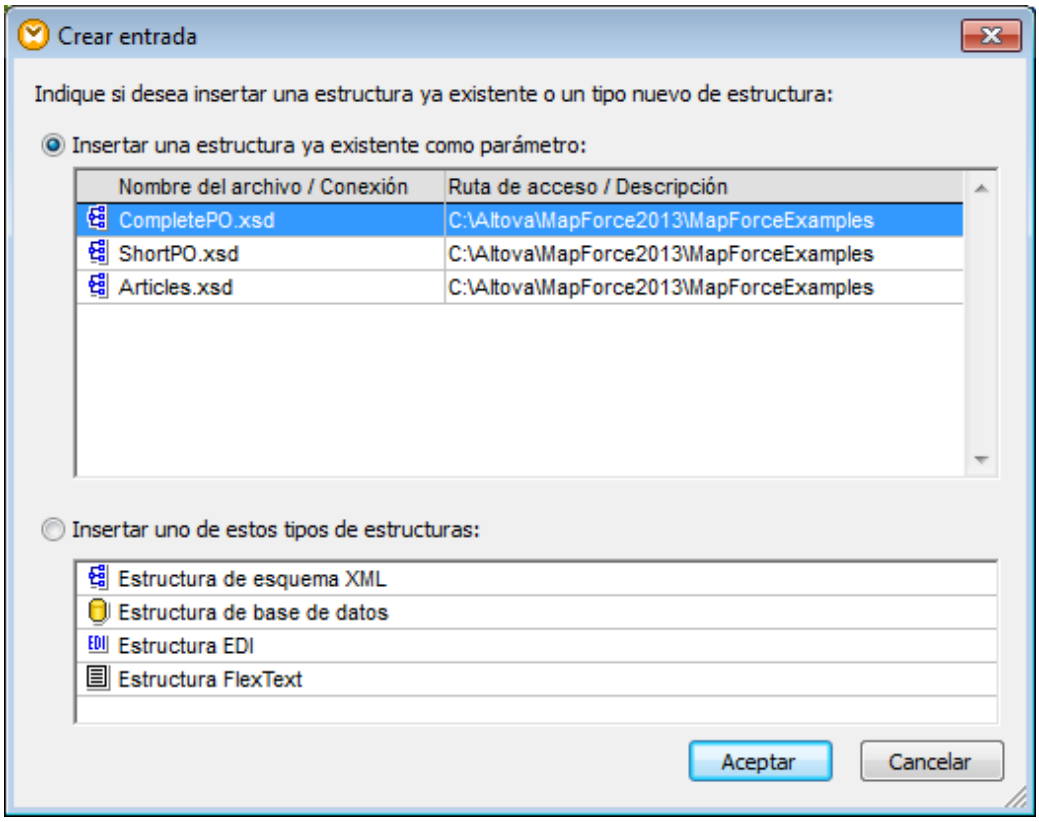


4. Seleccione el botón de opción *Tipo complejo (estructura en árbol)* y después haga clic en el botón **Elegir** del campo *Estructura*. Se abre otro cuadro de diálogo con dos cuadros de lista.

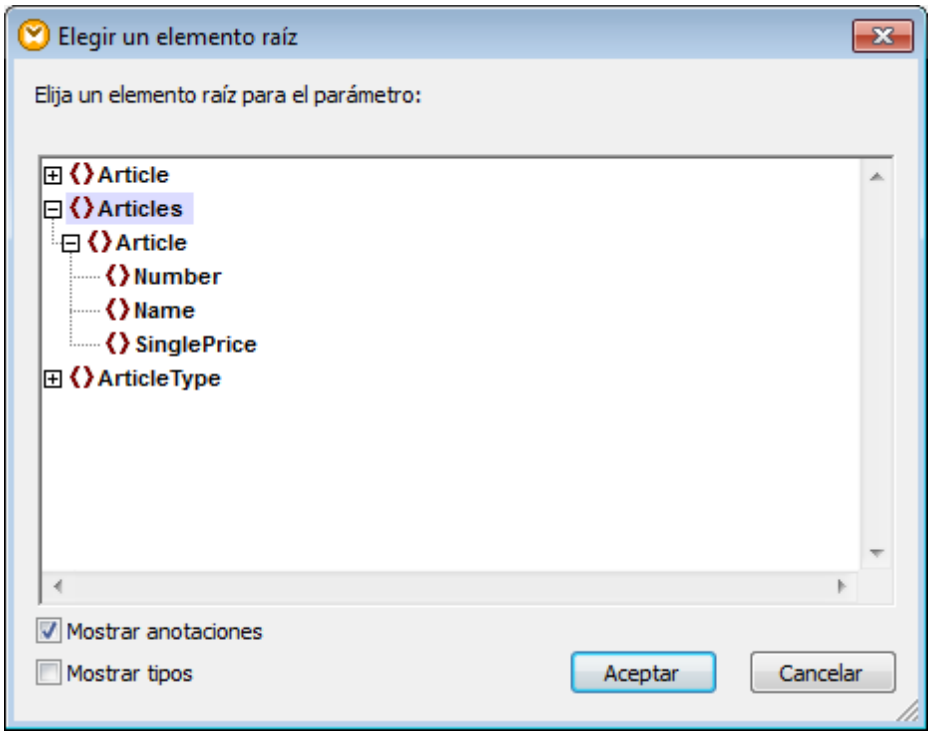
   El primer cuadro de lista enumera los componentes ya **existentes** en la asignación (en este caso, tres esquemas). Es decir, la lista enumera todos los componentes que ya se insertaron en la asignación activa (p. ej. un esquema XML, una BD, un archivo EDI o una estructura FlexText).

   En el cuadro de lista inferior puede seleccionar una estructura de datos compleja **nueva** (es decir, un esquema XML, una BD, un archivo EDI o una estructura FlexText).

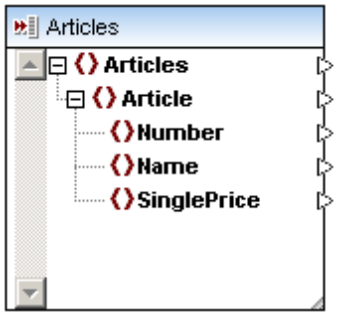
© 2018 Altova GmbH



5. Seleccione el botón de opción *Insertar una estructura nueva*, después la opción **Estructura de esquema XML** y haga clic en **Aceptar**.
6. Seleccione **`Articles.xsd`** en el cuadro de diálogo Abrir.
7. Seleccione el elemento que desea usar como elemento raíz del componente (p. ej. **`Articles`**) y haga clic en **Aceptar**. Haga otra vez clic en **Aceptar** para cerrar el otro cuadro de diálogo.

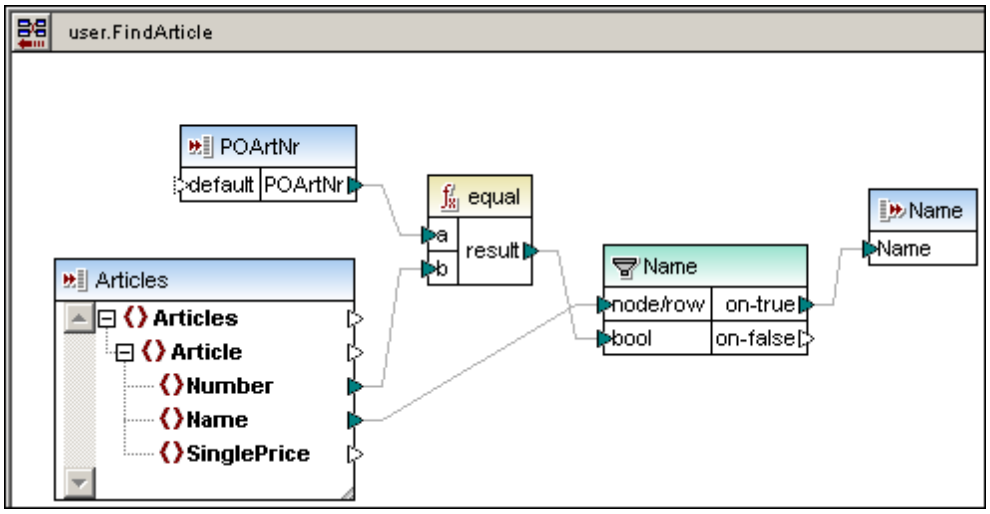


El componente **`Articles`** se inserta en la función definida por el usuario. Observe que a la izquierda del nombre del componente aparece el icono de entrada . Esto indica que el componente se usa como componente de entrada complejo.



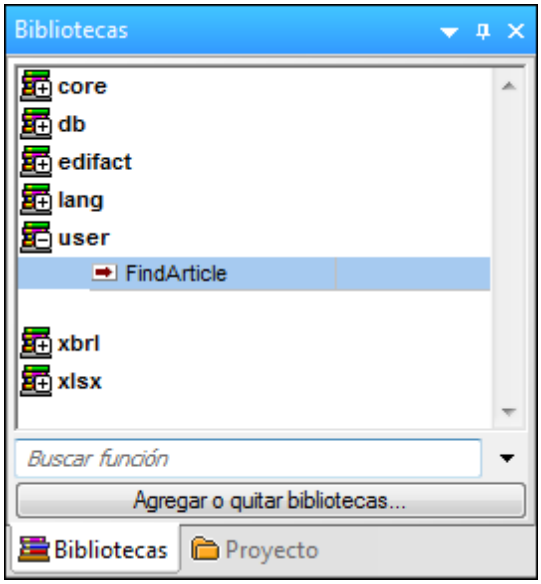
8. Inserte el resto de componentes que aparecen en la imagen siguiente, es decir, otro componente de entrada simple (**`POArtNr`**), un filtro, la función **`equal`** y un componente de salida (**`Name`**). Después conéctelos tal y como muestra la imagen.

 © 2018 Altova GmbH

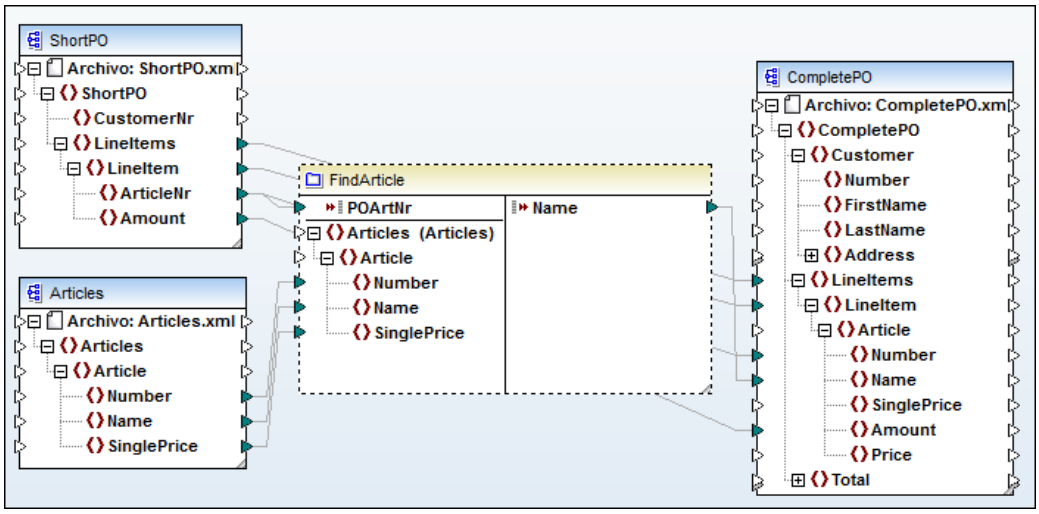


- el componente de entrada **`Articles`** recibe sus datos de **fuera** de la función. Los iconos de entrada que permiten la asignación de datos a este componente están ahí.
- a los componentes de entrada complejos no se les puede asignar un archivo XML de instancia para suministrar datos desde la función definida por el usuario.
- el componente de entrada **`POArtNr`** aporta los datos de número de artículo de **`ShortPO`** con los que se deben comparar los datos de **`Article | Number`**.
- el filtro filtra los registros cuyos números son idénticos y los envía al componente de salida.

9. Haga clic en el icono **Volver** para volver a la asignación principal.
10. Inserte la función definida por el usuario nueva (arrastrándola desde la ventana Bibliotecas hasta el panel *Asignación*).



11. Cree las conexiones que aparecen en esta imagen.



La parte de la izquierda contiene los parámetros de entrada a los que se asignan elementos de los dos archivos XML:

- **ShortPO** aporta los datos para el parámetro de entrada **`POArtNr`**.
- **Articles** aporta los datos para el parámetro de entrada complejo. El archivo de instancia `Articles.xml` se asignó al archivo de esquema **Articles** cuando se insertó el componente.
- El parámetro de entrada complejo **`Articles`** con sus nodos XML secundarios, a los que se asignaron datos desde el componente **Articles**.

La parte de la derecha contiene:

- un parámetro de entrada simple llamado **`Name`**, que pasa los elementos filtrados que tienen el mismo número de **`Article`** al elemento **`Name`** del componente de destino **CompletePO**.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <CompletePO xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3    <LineItems>
4      <LineItem>
5        <Article>
6          <Number>3</Number>
7          <Name>Pants</Name>
8          <Amount>5</Amount>
9        </Article>
10     </LineItem>
11     <LineItem>
12       <Article>
13         <Number>1</Number>
14         <Name>T-Shirt</Name>
15         <Amount>17</Amount>
16       </Article>
17     </LineItem>
18   </LineItems>
19 </CompletePO>
```

Nota: cuando cree conexiones de **copia total** entre un esquema y un parámetro de una función definida por el usuario, ambos componentes deben estar basados en el mismo esquema. Sin embargo, no es necesario que ambos tengan el mismo elemento raíz.

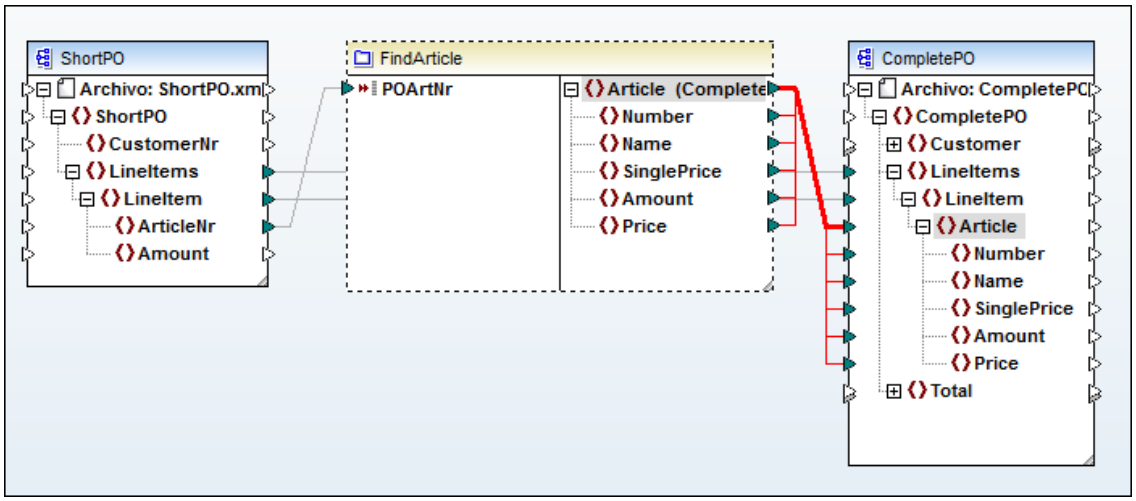
 © 2018 Altova GmbH

8.2.6 Función definida por el usuario compleja: nodo XML como salida

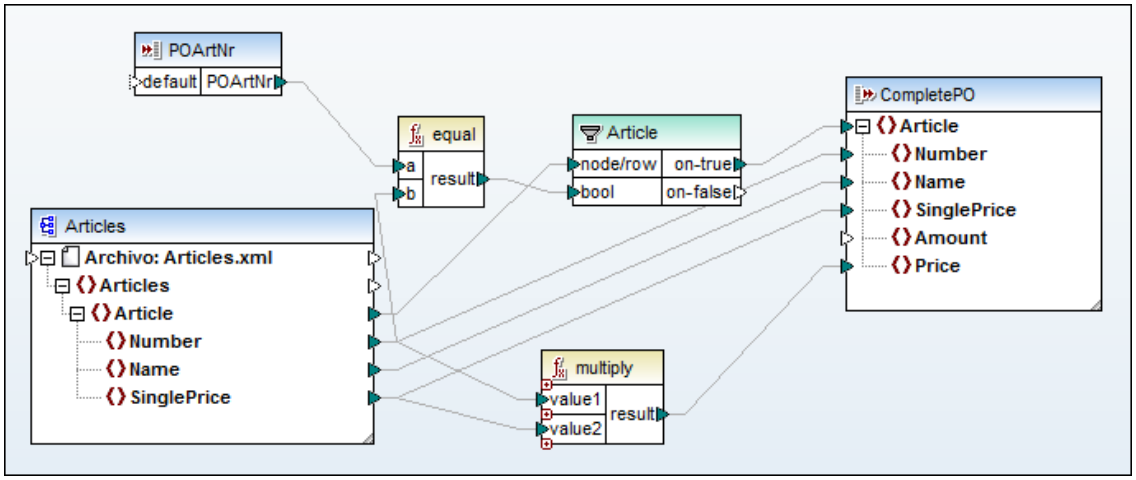
Este apartado utiliza el archivo de ejemplo `lookup-udf-out.mfd`, disponible en la carpeta **...\MapForceExamples** A continuación explicamos cómo se define una función definida por el usuario de tipo inline con un componente de salida complejo.

Tenga en cuenta que la función definida por el usuario FindArticle tiene dos partes:

- La parte de la izquierda contiene un parámetro de entrada simple llamado **`POArtNr`**
- La parte de la derecha contiene un parámetro de salida complejo llamado **`Article (CompletePO)`**, cuyos nodos XML secundarios están asignados al componente de destino **CompletePO**.



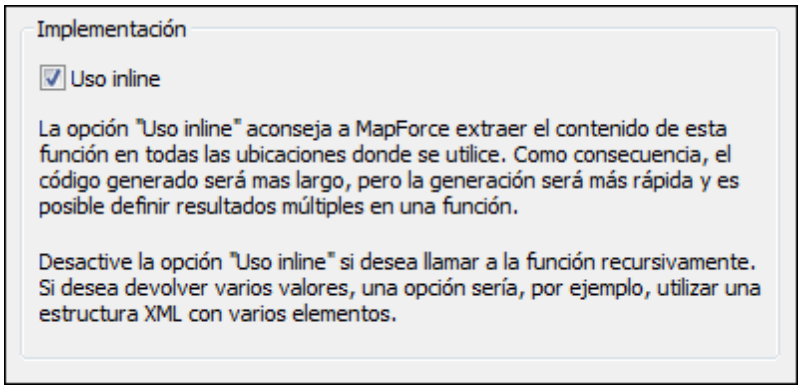
A continuación puede ver los componentes que constituyen la función definida por el usuario, con los dos componentes de entrada a la izquierda y el de salida a la derecha.



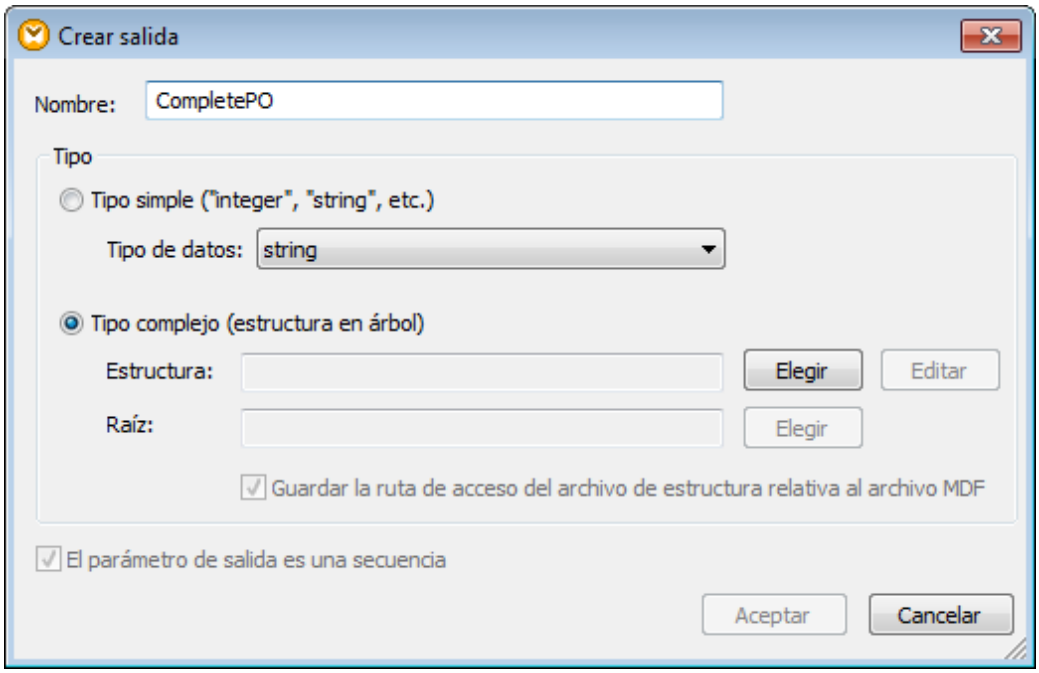
8.2.6.1 *Definir componentes de salida complejos*

Siga estas instrucciones para crear una función que devuelva una estructura XML como parámetro de salida:

1. Cree una función definida por el usuario (**Función | Crear una función definida por el usuario**). Recuerde que la opción *Uso inline* está activada por defecto.



2. Haga clic en el icono **Insertar componente de salida**  de la barra de herramientas. En el cuadro de diálogo escriba el nombre del componente de salida en el campo *Nombre* (p. ej. CompletePO).



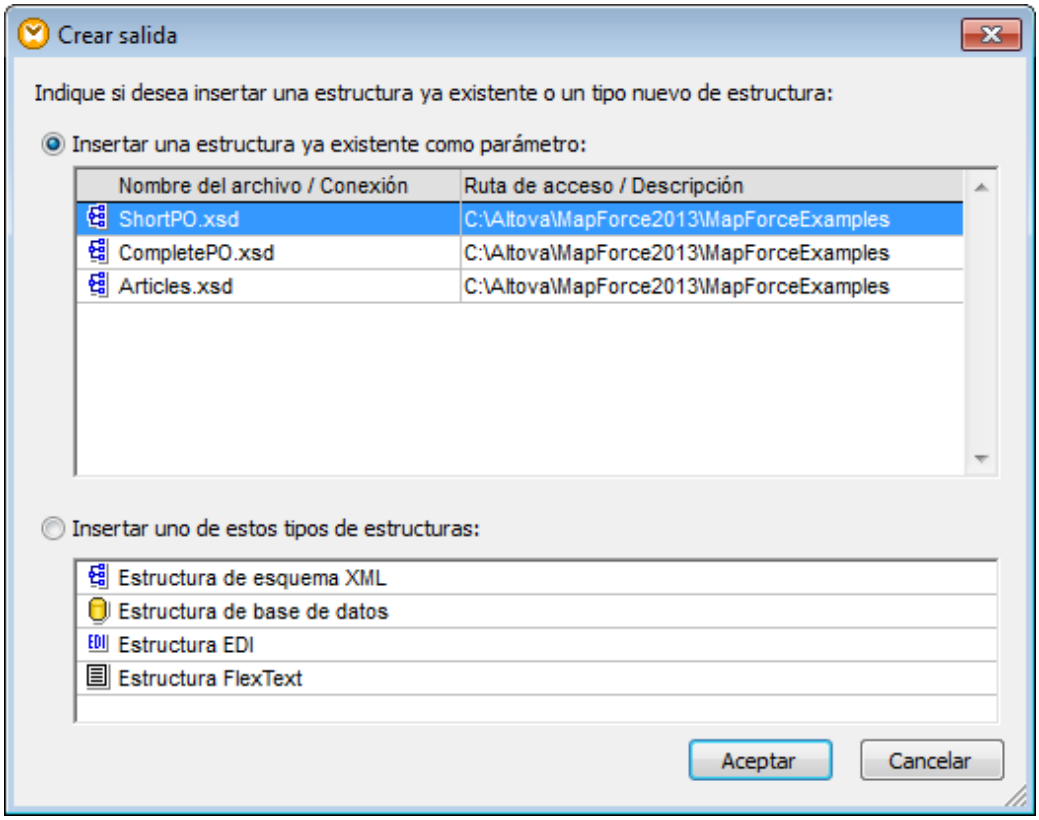
3. Seleccione el botón de opción *Tipo complejo (estructura en árbol)* y después haga clic en el botón **Elegir** del campo *Estructura*. Se abre otro cuadro de diálogo con dos cuadros de listas.

   El primer cuadro de lista enumera los componentes ya **existentes** en la asignación (en

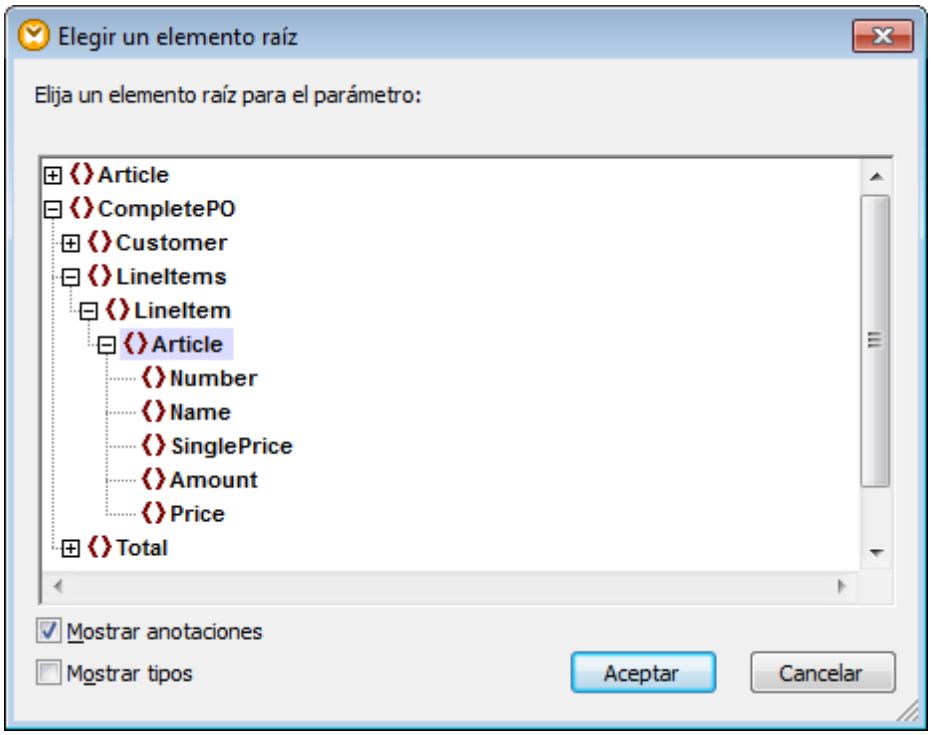
© 2018 Altova GmbH

este caso, tres esquemas). Es decir, la lista enumera todos los componentes que ya se insertaron en la asignación activa (p. ej. un esquema XML, una BD, un archivo EDI o una estructura FlexText).

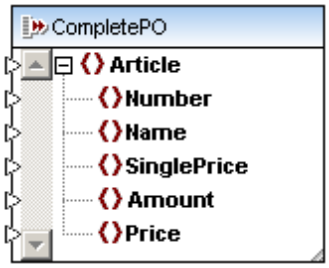
En el cuadro de lista inferior puede seleccionar una estructura de datos compleja **nueva** (es decir, un esquema XML, una BD, un archivo EDI o una estructura FlexText).



4. Seleccione el botón de opción *Insertar una estructura nueva*, seleccione la opción **Estructura de esquema XML** y haga clic en **Aceptar**.
5. Seleccione el esquema `CompletePO.xsd` en el cuadro de diálogo Abrir.
6. Haga clic en el elemento que desea usar como elemento raíz del componente (p. ej. **`Article`**) y haga clic en **Aceptar**. Haga otra vez clic en **Aceptar** para cerrar el otro cuadro de diálogo.

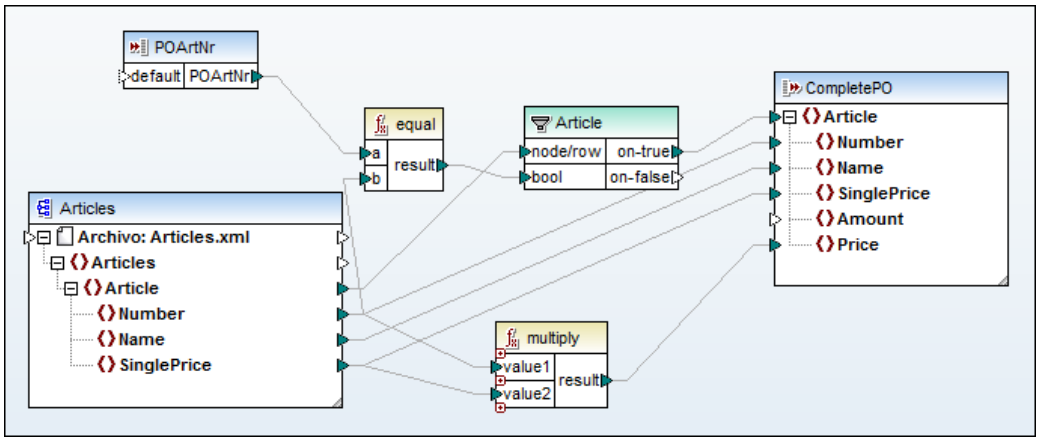


El componente **CompletePO** se inserta en la función definida por el usuario. Observe que a la izquierda del nombre del componente aparece el icono de salida . Esto indica que el componente se utiliza como parámetro/componente de salida complejo.



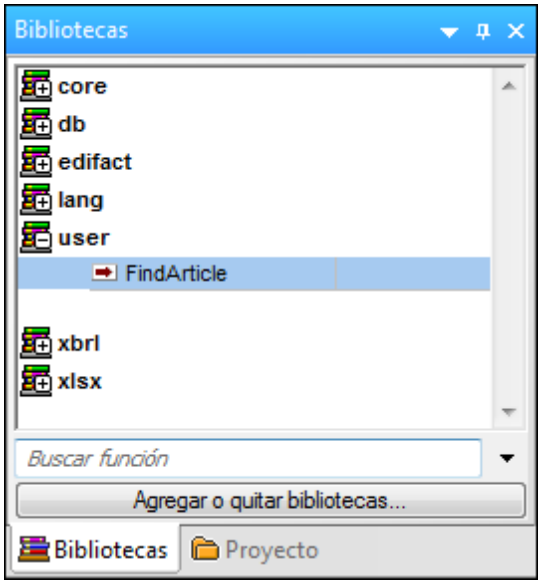
7. Inserte el esquema XML **Articles** en la función definida por el usuario y asigne el archivo `Articles.xml` como XML de instancia.
8. Inserte el resto de componentes que aparecen en la imagen siguiente, es decir: el componente de entrada simple **`POArtNr`**, un filtro, una función **`equal`** y una función **`multiply`**. Después conéctelas como muestra la imagen.

 © 2018 Altova GmbH

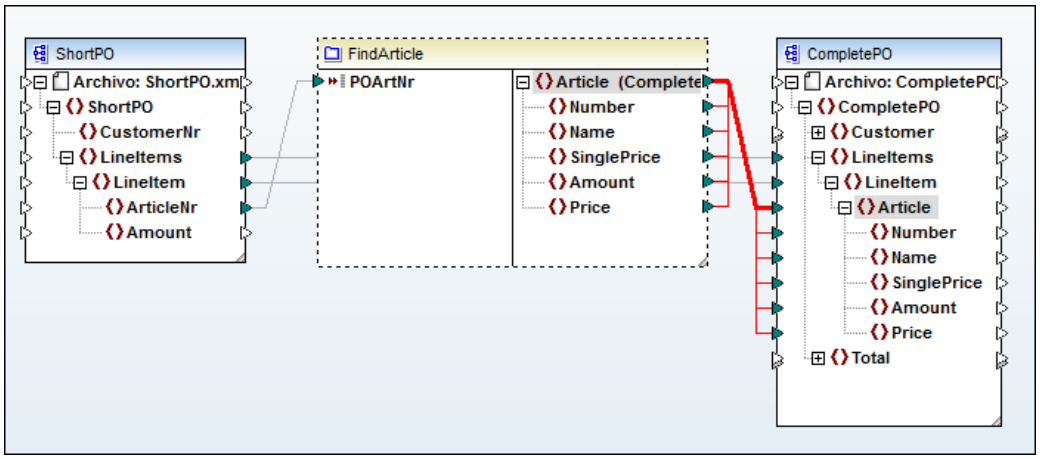


- el componente **Articles** recibe datos del archivo de instancia `Articles.xml`, situado dentro de la función definida por el usuario.
- los componentes de entrada aportan los datos de **`POArtNr`** y **`Amount`**, con los que se comparan los datos de **`Articles | Number & Price`**.
- el filtro filtra los registros que tienen números idénticos y los pasa al componente de salida **CompletePO**.

9. Haga clic en el icono **Volver**  para volver a la asignación principal.
10. Inserte la función definida por el usuario nueva (arrastrándola desde la ventana Bibliotecas hasta el panel *Asignación*).



11. Cree las conexiones que aparecen en esta imagen. Tras crear el conector entre el parámetro de salida **`Article (CompletePO)`** y el componente de destino, haga clic con el botón derecho en el conector y seleccione **Copia total** en el menú contextual. Los demás conectores se generan automáticamente y se resaltan en rojo (*imagen siguiente*).



Nota: cuando cree conexiones de **copia total** entre un esquema y un parámetro de una función definida por el usuario, ambos componentes deben estar basados en el mismo esquema. Sin embargo, no es necesario que ambos tengan el mismo elemento raíz.

La parte izquierda de la función contiene el parámetro de entrada, al que se asigna un solo elemento:

- **ShortPO** aporta el número de artículo al parámetro de entrada **`POArtNr`**.

La parte derecha de la función contiene:

- un parámetro de salida complejo llamado **`Article (CompletePO)`** con nodos XML secundarios, que asigna los elementos filtrados (los que tienen el mismo número de artículo) al componente de destino **CompletePO**.

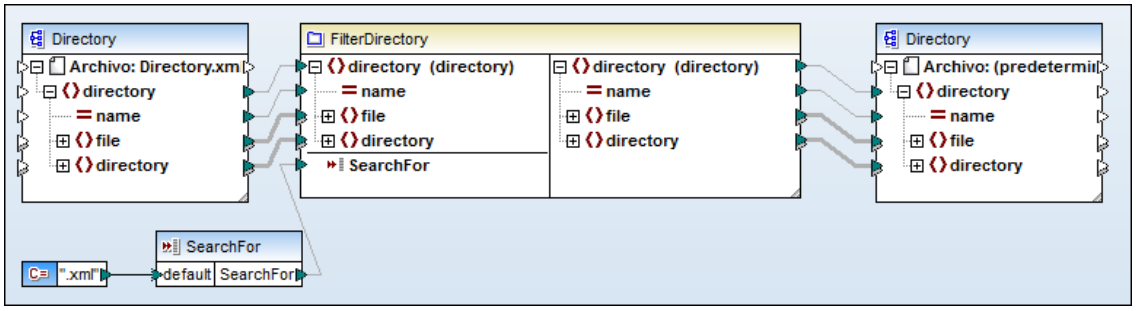
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <CompletePO xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespace
3    <LineItems>
4      <LineItem>
5        <Article>
6          <Number>3</Number>
7          <Name>Pants</Name>
8          <SinglePrice>34</SinglePrice>
9          <Price>102</Price>
10       </Article>
11     </LineItem>
12     <LineItem>
13       <Article>
14         <Number>1</Number>
15         <Name>T-Shirt</Name>
16         <SinglePrice>25</SinglePrice>
17         <Price>25</Price>
18       </Article>
```

8.2.7 Asignación recursiva definida por el usuario

En este apartado explicamos cómo se creó la asignación **RecursiveDirectoryFilter.mfd** de la carpeta ...**\MapForceExamples** y cómo diseñar asignaciones de datos recursivas. La carpeta de proyecto MapForceExamples contiene algunos ejemplos más de asignaciones de datos

recursivas.

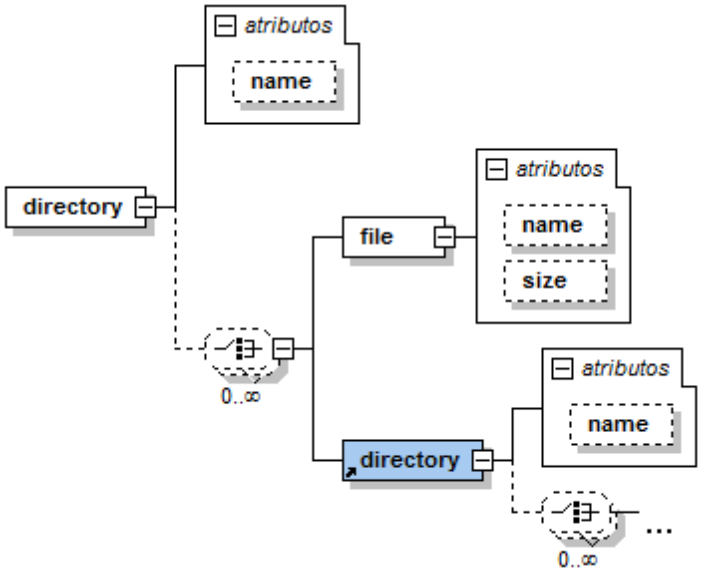
En la imagen siguiente puede ver la asignación final que contiene la función recursiva definida por el usuario **FilterDirectory**. El objetivo de esta función es filtrar una lista de archivos .xml del archivo de origen.



El **archivo de origen** que contiene los datos sobre los archivos y directorios para esta asignación es Directory.xml. Este archivo XML incluye los datos sobre directorios y archivos de forma jerárquica:

```
24      <directory name="output">
25        <file name="examplesite1.css" size="3174"/>
26        <directory name="images">
27          <file name="blank.gif" size="88"/>
28          <file name="block_file.gif" size="13179"/>
29          <file name="block_schema.gif" size="9211"/>
30          <file name="nav_file.gif" size="60868"/>
31          <file name="nav_schema.gif" size="6002"/>
32        </directory>
33      </directory>
34    </directory>
35    <directory name="Import">
36      <file name="altova.mdb" size="266240"/>
37      <file name="Data_shape.mdb" size="225280"/>
38    </directory>
39    <directory name="IndustryStandards">
40      <directory name="News">
41        <file name="high-tide.jpg" size="10793"/>
42        <file name="Newsml-example.xml" size="5004"/>
43        <file name="nitf-example.xml" size="9327"/>
44      </directory>
```

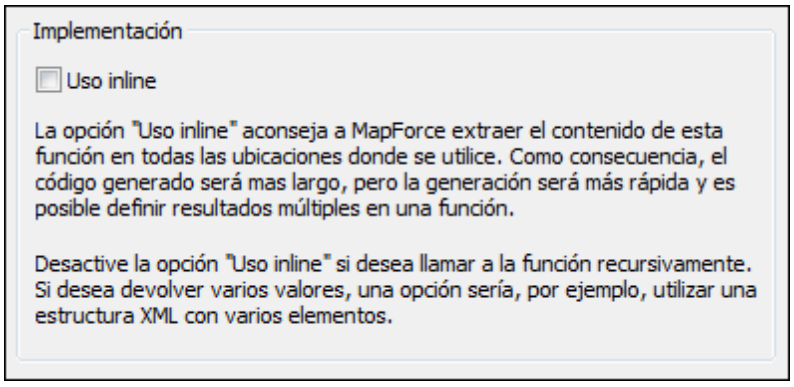
El archivo de esquema XML al que hace referencia el archivo Directory.xml tiene un elemento recursivo llamado "directory" que permite un número indeterminado de subdirectorios y de archivos debajo del elemento directory.



8.2.7.1 Crear una función recursiva definida por el usuario

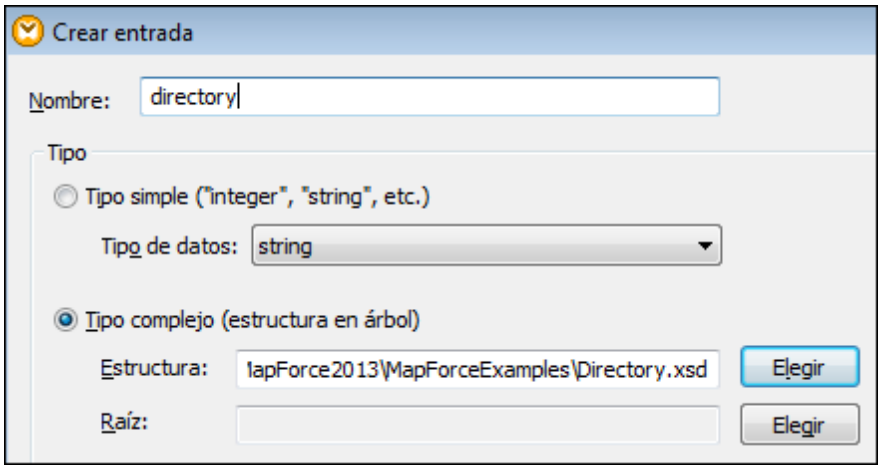
Siga estas instrucciones para crear una función recursiva definida por el usuario:

1. Cree una función definida por el usuario (**Función | Crear una función definida por el usuario**) llamada FilterDirectory.
2. Asegúrese de desactivar la casilla *Uso inline* del grupo de opciones *Implementación* para que la función sea recursiva. Haga clic en **Aceptar**.

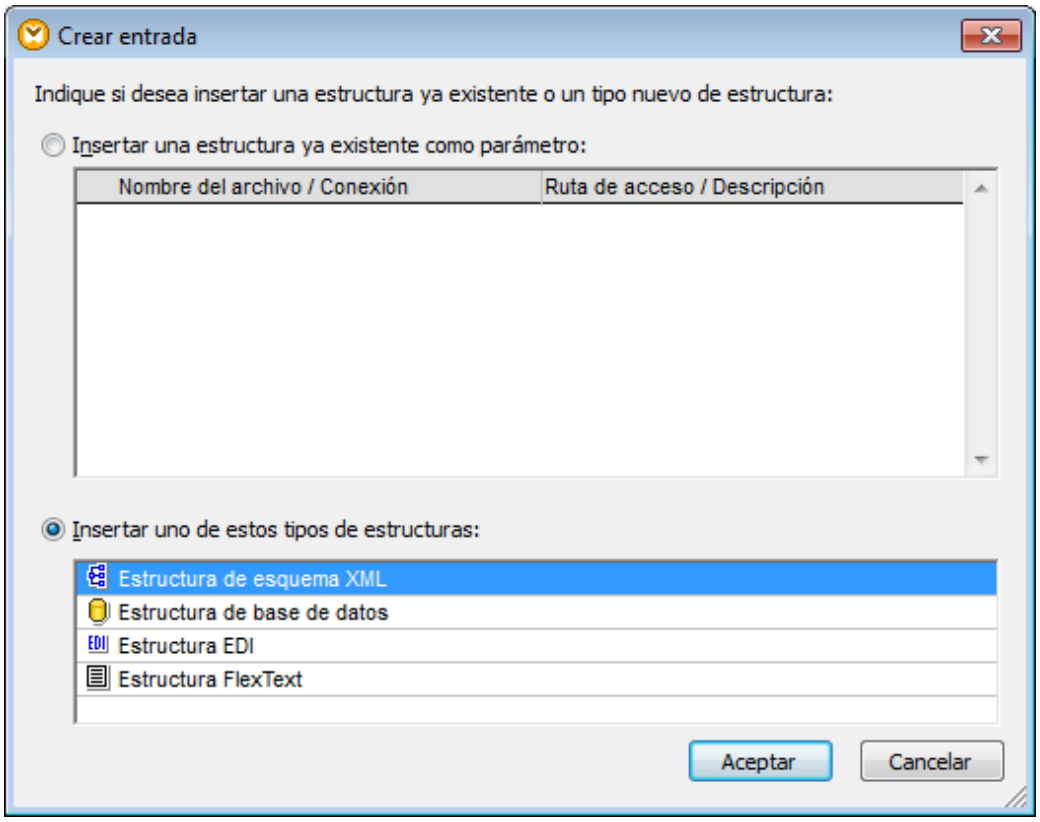


   Ahora estamos en la ventana **FilterDirectory, donde podemos crear la función definida por el usuario.**
3. Seleccione **Función | Insertar componente de entrada**.
4. Defina el nombre de componente "directory" y haga clic en el botón de opción *Tipo complejo (estructura en árbol)*.

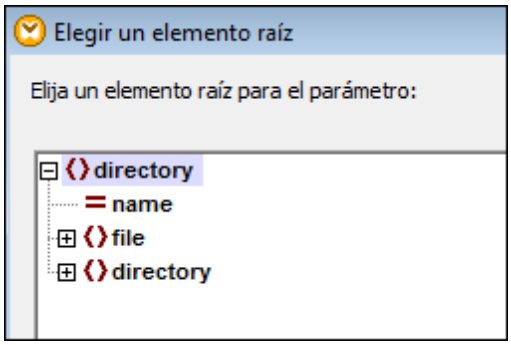
 © 2018 Altova GmbH



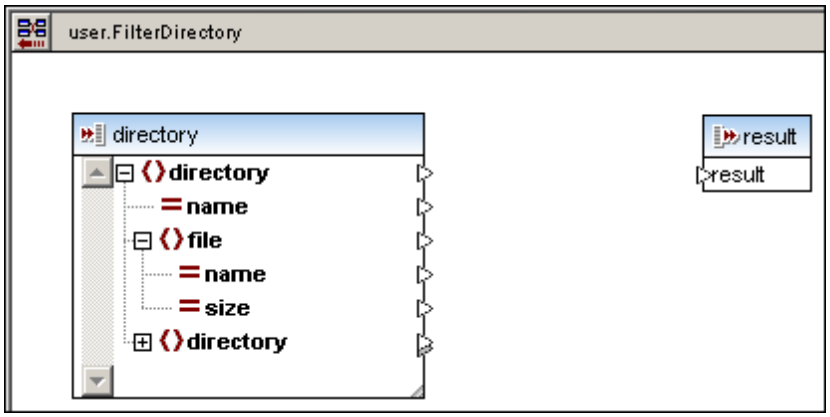
5. Haga clic en el botón **Elegir** y después en la entrada **`Estructura de esquema XML`** del panel inferior. Después haga clic en **Aceptar**.



6. Seleccione el archivo **Directory.xsd** de la carpeta ...\MapForceExamples y haga clic en el botón **Abrir**.
7. Cuando la aplicación solicite el elemento raíz haga clic en **Aceptar** (el elemento raíz debería ser `directory`).

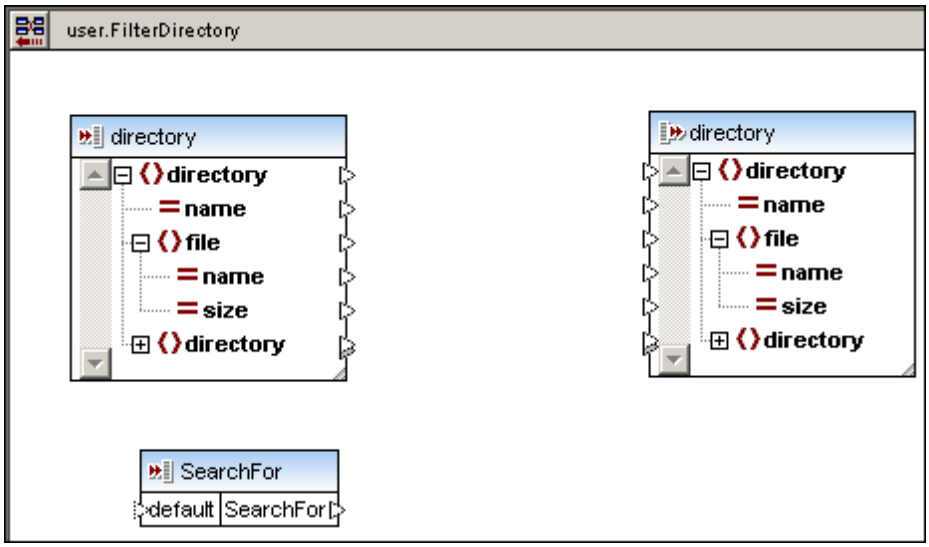


8. Haga clic en **Aceptar** para insertar el parámetro de entrada complejo. Esta es la función definida por el usuario que acabamos de crear:



9. Elimine el componente de salida simple **result** (porque aquí debemos insertar un componente de salida complejo).
10. Seleccione **Función | Insertar componente de salida...** para insertar un componente de salida y use el método de los pasos anteriores para que el componente de salida "directory" sea de tipo complejo. Ahora tenemos dos componentes complejos: uno de entrada y otro de salida.
11. Seleccione **Función | Insertar componente de entrada...** para insertar un componente de entrada de tipo simple llamado "SearchFor". Desactive la casilla *Requiere una conexión de entrada*.

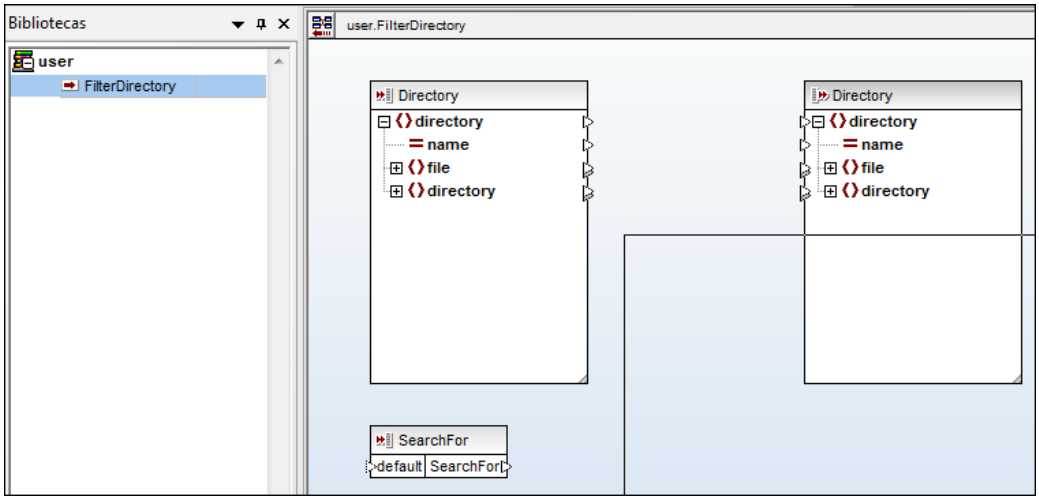
© 2018 Altova GmbH



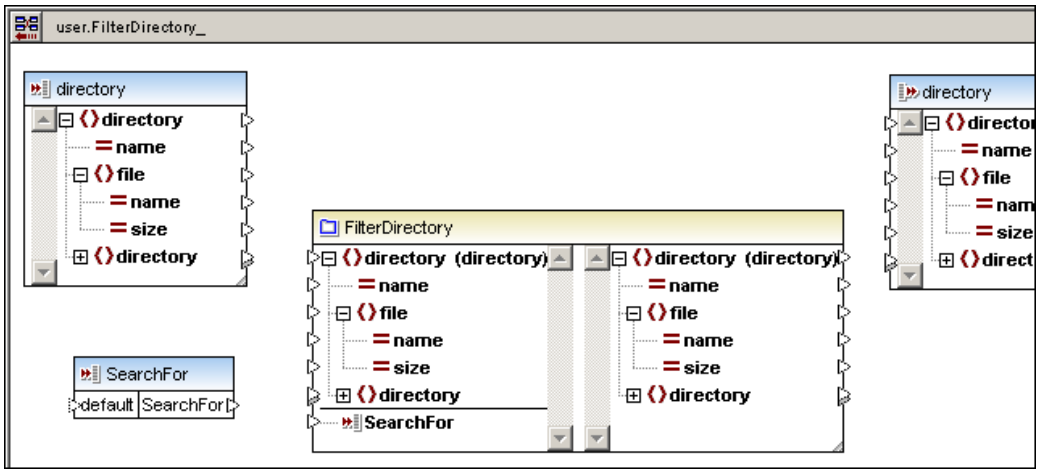
Insertar la función recursiva definida por el usuario

Llegados a este punto todos los componentes de entrada y salida necesarios están definidos para la función definida por el usuario. Ahora falta insertar la función en la ventana actual de la función definida por el usuario.

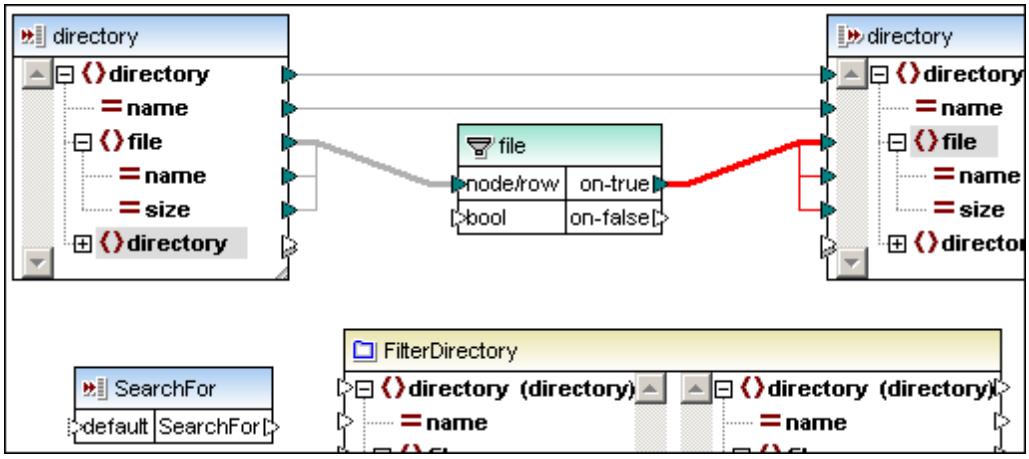
1. Busque la función "FilterDirectory" en la sección **user** de la ventana Bibliotecas.
2. Haga clic en la función FilterDirectory y arrástrela hasta la ventana de FilterDirectory en la que ha estado trabajando hasta ahora.



La función definida por el usuario aparece en la ventana de la función definida por el usuario en forma de componente recursivo.

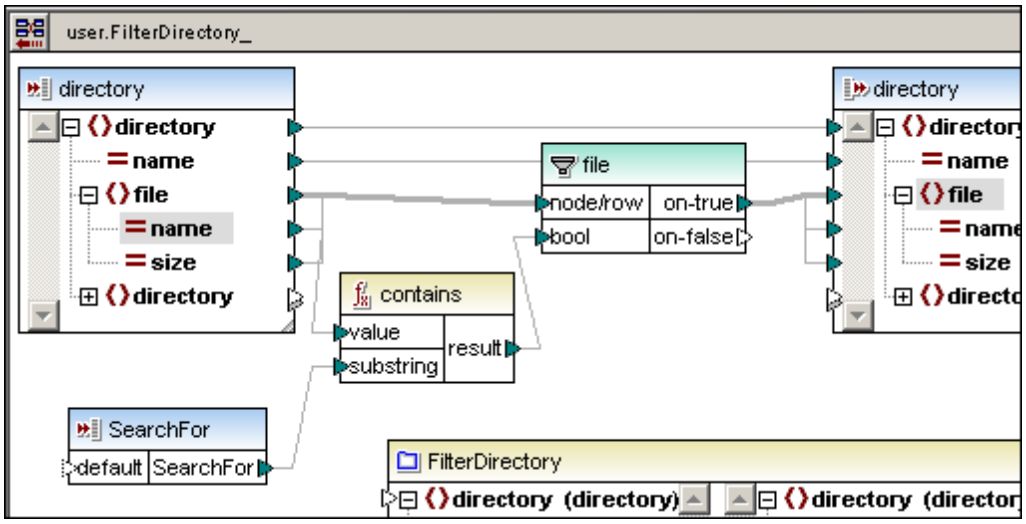


3. Conecte los nodos **directory, name** y **file** del componente de entrada con los nodos del mismo nombre del componente de salida.
4. Haga clic con el botón derecho en el conector que une los nodos **file** y seleccione **Insertar filtro** en el menú contextual para introducir un componente de filtrado.
5. Haga clic con el botón derecho en el conector **on-true** y seleccione **Copia total** en el menú contextual.
   Ahora los conectores aparecen como conectores de copia total.



6. Inserte una función **contains** de la biblioteca **Core | String functions**.
7. Conecte **name** con **value** y el parámetro **SearchFor** con **substring**. Después conecte el resultado con el nodo **bool** del filtro.

© 2018 Altova GmbH

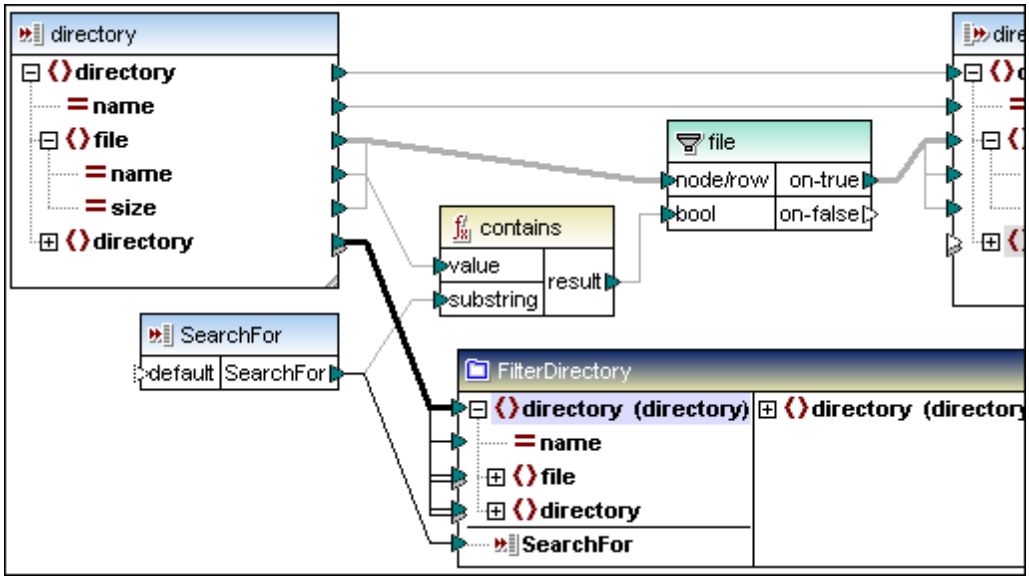


8. Para terminar conecte el nodo **SearchFor** del componente de entrada con el nodo **SearchFor** de la función definida por el usuario.

Definir la recursión

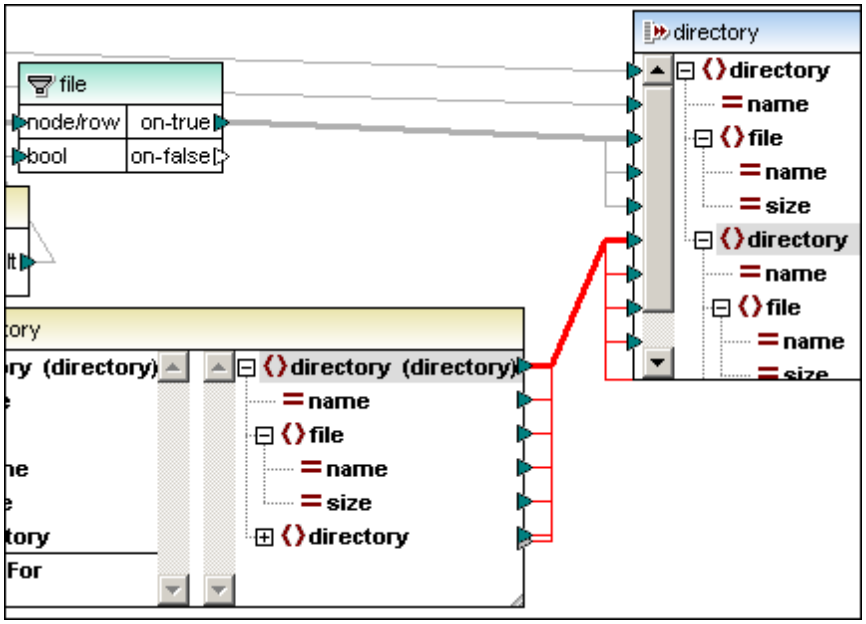
Llegados a este punto la asignación de un nivel de recursión en un solo directorio está terminada. Ahora falta definir el procesamiento del subdirectorio.

1. Para empezar asegúrese de que el icono **Alternar la conexión automática de secundarios** está activado en la barra de herramientas.
2. Conecte el nodo inferior **directory** del componente de entrada con el nodo superior **directory** de la función recursiva definida por el usuario.



3. Conecte el nodo superior de salida **directory** de la función definida por el usuario con el nodo inferior **directory** del componente de salida.

4. Haga clic con el botón derecho en el conector superior, seleccione **Copia total** en el menú contextual y haga clic en **Aceptar** si desea crear una conexión de copia total.

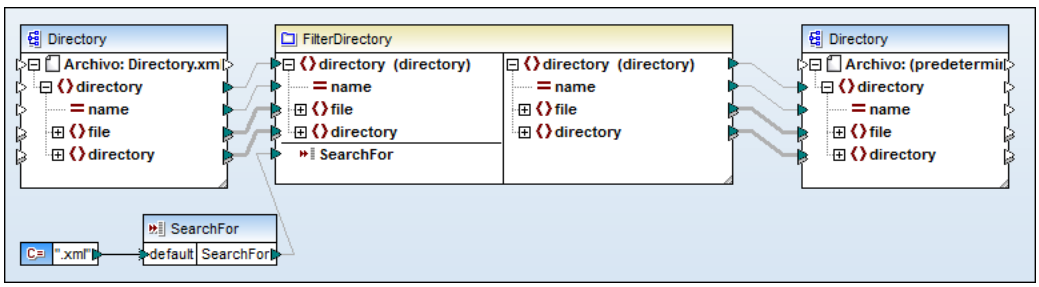


La función definida por el usuario ya está terminada.

Ahora haga clic en **Volver** para seguir diseñando la asignación en la ventana principal.

Ventana principal de asignación

1. Arrastre la función **FilterDirectory** de la sección **user** de la ventana Bibliotecas hasta el área de asignación principal.
2. Seleccione **Insertar | Archivo de esquema XML** para insertar el esquema Directory.xsd y después seleccione el archivo de instancia Directory.xml.
3. Use el mismo método para insertar un componente de salida basado en el esquema Directory.xsd y haga clic en **Omitir** cuando la aplicación solicite un archivo de instancia.
4. Inserte un componente de constante y después un componente de entrada (p. ej. SearchFor).
5. Cree las conexiones que aparecen en la imagen siguiente.
6. Cuando conecte los conectores de nivel superior, de **Directory** a **Directory**, a ambos lados del componente definido por el usuario, haga clic con el botón derecho en el conector y seleccione **Copia total** en el menú contextual.



7. Haga clic en el panel *Resultados* para ver el resultado de la asignación.

 © 2018 Altova GmbH

Nota: si hace doble clic en el nodo inferior directory del componente **Directory**, se abre un nuevo nivel de recursión (es decir, verá un nuevo nivel **directory | file | directory**). Con el conector de copia total se usan automáticamente todos los niveles de recursión actuales en la instancia XML y no será necesario expandir los niveles de recursión manualmente.

8.3 Importar funciones XSLT 1.0/2.0 personales

Puede ampliar las bibliotecas de funciones XSLT 1.0 y 2.0 de MapForce con sus propias funciones personales, siempre y cuando estas devuelvan tipos simples.

> Solamente se admiten funciones personales que devuelvan tipos simples (p. ej. cadenas).

Para importar funciones de un archivo XSLT:

1. En el menú **Herramientas** haga clic en **Opciones** (o haga clic en **Agregar o quitar bibliotecas** en la parte inferior de la ventana Bibliotecas).
2. Haga clic en el botón **Agregar** y navegue hasta el archivo .xsl o .xslt.

Los archivos XSLT importados aparecen como bibliotecas en la ventana Bibliotecas y muestran todas las plantillas con nombres como si fueran funciones. Si no puede ver la biblioteca importada, compruebe que el lenguaje de transformación seleccionado es XSLT (*véase Seleccionar un lenguaje de transformación*).

Notas:

- Para poder importarlas en MapForce las funciones deben declararse en el archivo XSLT como plantillas con nombre según lo estipulado por la especificación XSLT. También puede importar funciones que aparezcan en un documento XSLT 2.0 con el formato `<xsl:function name="MiFunción">`. Si el archivo XSLT importado importa o incluye otros archivos XSLT, también se importarán dichos archivos y sus funciones.
- Los conectores de entrada asignables de las funciones importadas personales dependen del número de parámetros utilizado en la llamada a plantilla. También se admiten parámetros opcionales.
- No se admiten espacios de nombres.
- Si realiza cambios en archivos XSLT que ya están importados en MapForce, los cambios se detectan automáticamente y puede elegir si los archivos se vuelven a cargar.
- Cuando escriba plantillas con nombre, asegúrese de que las instrucciones XPath utilizadas en la plantilla están enlazadas al espacio de nombres correcto. Para ver los enlaces de espacio de nombres de la asignación genere una vista previa del código XSLT de salida.

Tipos de datos en XPath 2.0

Si el documento XML hace referencia a un esquema XML y es válido según este esquema, deberá convertir o construir explícitamente tipos de datos que no se conviertan implícitamente al tipo de datos necesario por medio de una operación.

En el modelo de datos de XPath 2.0 que utiliza el motor XSLT 2.0 de Altova todos los valores de nodos **atomizados** del documento XML tienen asignado el tipo de datos `xs:untypedAtomic`. El tipo `xs:untypedAtomic` es adecuado para las conversiones de tipo implícitas.

Por ejemplo:

- la expresión `xs:untypedAtomic("1") + 1` da como resultado un valor de 2 porque el valor `xdt:untypedAtomic` se convierte **implícitamente** en `xs:double` por medio del

© 2018 Altova GmbH

operador de suma.
- los operadores aritméticos convierten los operandos implícitamente en `xs:double`.
- los operadores de comparación de valores convierten los operandos en `xs:string` antes de compararlos.

Temas relacionados:

Ejemplo: agregar funciones XSLT 1.0 personales
Ejemplo: sumar valores de nodos
Implementación del motor XSLT 1.0
Implementación del motor XSLT 2.0

8.3.1 Ejemplo: agregar funciones XSLT personales

Este ejemplo explica cómo importar funciones XSLT 1.0 personales en MapForce. Los archivos necesarios para seguir este ejemplo están en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples**.

- **Name-splitter.xslt:** este archivo XSLT define una plantilla con nombre llamada "**tokenize**" que tiene un solo parámetro llamado "string". La plantilla recorre una cadena de entrada y separa todas las letras mayúsculas de la cadena con un espacio.

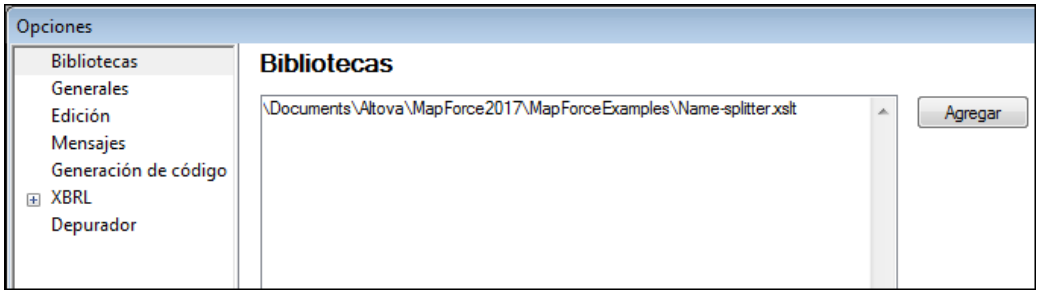
```
2  <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform
3    <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>
4
5    <xsl:template match="*">
6      <xsl:for-each select=".">
7      <xsl:call-template name="tokenize">
8      <xsl:with-param name="string" select="."/>
9      </xsl:call-template>
10     </xsl:for-each>
11   </xsl:template>
12
13   <xsl:template name="tokenize">
14     <xsl:param name="string" select="."/>
15     <xsl:variable name="caps" select="translate($string, '-abcdefghijklmnopqrstuv
16     <xsl:variable name="capscount" select="string-length($caps)"/>
17     <xsl:variable name="token">
```

- **Name-splitter.xml:** el archivo de instancia XML de origen que se debe procesar.
- **Customers.xsd:** el esquema XML de origen.
- **CompletePO.xsd:** el esquema XML de destino.

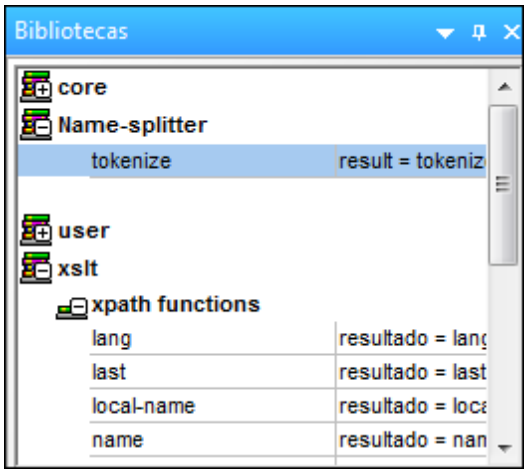
Para agregar una función XSLT personal:

1. Seleccione el lenguaje de transformación XSLT (*véase Seleccionar un lenguaje de transformación*).
2. Haga clic en el botón **Agregar o quitar bibliotecas** situado en la parte inferior de la ventana Bibliotecas (también puede hacer clic en **Herramientas | Opciones** y seleccionar **Bibliotecas**).
3. Ahora haga clic en el botón **Agregar** y navegue hasta el archivo XSL o XSLT que

contiene la plantilla con nombre que desea usar como función (en este caso **Name-splitter.xslt**).

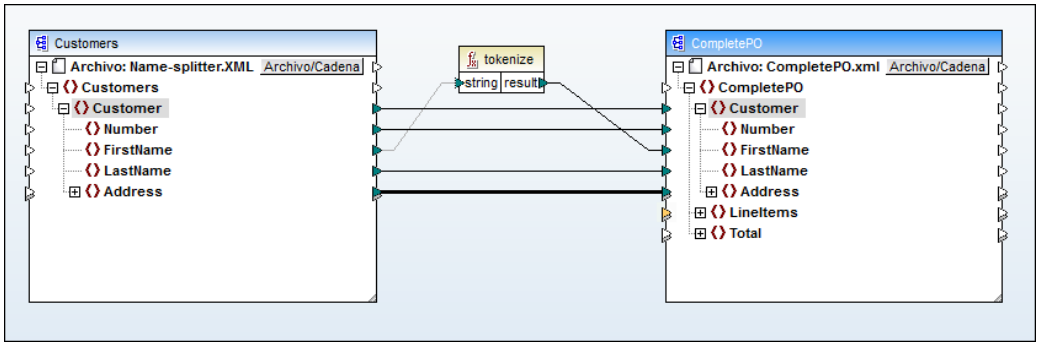


4. Haga clic en **Aceptar**. El nombre del archivo XSLT aparece en la ventana Bibliotecas junto con las funciones definidas como plantillas con nombre (en este ejemplo es **Name-splitter** con la función **`tokenize`**).



Para usar la función XSLT en la asignación de datos:

1. Arrastre la función **`tokenize`** hasta el área de asignación y cree las asignaciones de datos que aparecen en esta imagen:



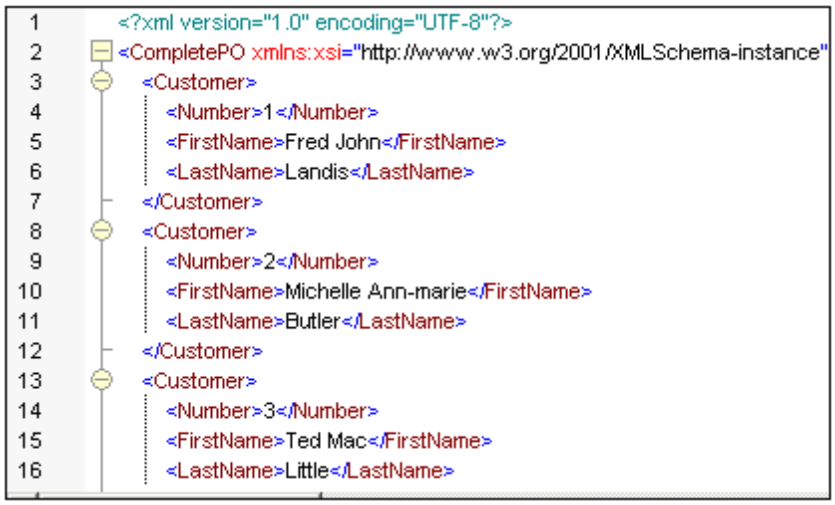
2. Haga clic en el panel *XSLT* para ver el código XSLT de salida.

 © 2018 Altova GmbH



Nota: en cuanto una plantilla con nombre se utiliza en una asignación de datos, el archivo XSLT que contiene la plantilla con nombre **se incluye** en el código XSLT de salida (**xsl:include href**...) y **se le llama** con el comando **xsl:call-template**.

3. Haga clic en el panel *Resultados* para ver el resultado de la asignación de datos.



Para quitar bibliotecas XSLT personales de MapForce:

1. Haga clic en el botón **Agregar o quitar bibliotecas** situado en la parte inferior de la ventana Bibliotecas.
2. Haga clic en la biblioteca XSLT que desea eliminar y después haga clic en **Eliminar**.

8.3.2 Ejemplo: sumar valores de nodos

Este ejemplo explica cómo procesar varios nodos de un documento XML y asignar el resultado en forma de un solo valor a un documento XML de destino. Concretamente el objetivo de esta asignación de datos es calcular el precio de todos los productos de un archivo XML de origen y escribirlo como un solo valor en un archivo XML de salida. Los archivos utilizados en este ejemplo están en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**:

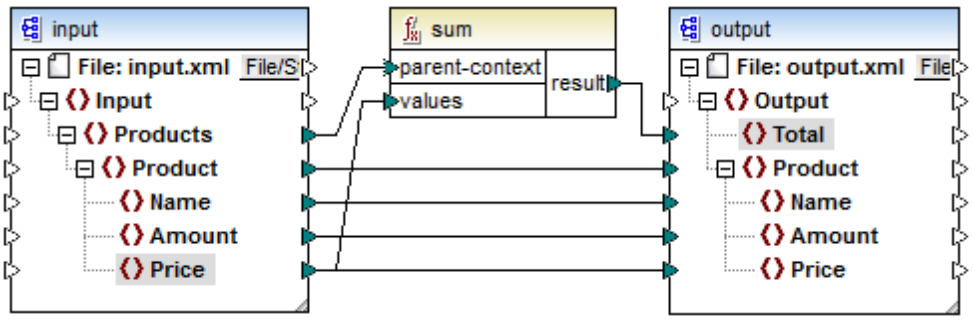
- **Summing-nodes.mfd:** el archivo de asignación de datos
- **input.xml:** el archivo XML de origen
- **input.xsd:** el esquema XML de origen
- **output.xsd:** el esquema XML de destino
- **Summing-nodes.xslt:** una hoja de estilos XSLT personal que contiene una plantilla con nombre que sumará los nodos.

Hay dos maneras de conseguir nuestro objetivo:

- con ayuda de la función de agregado `sum` de la biblioteca **core**.
- importando una hoja de estilos XSLT personal en MapForce.

Solución nº1: usar la función de agregado "sum"

Para usar la función de agregado `sum` a la asignación basta con arrastrarla desde la ventana Bibliotecas hasta el panel de asignación. Recuerde que las funciones que aparecen en la ventana Bibliotecas dependen del lenguaje de transformación que esté seleccionado. Tras colocar la función en el área de asignación deberá crear las conexiones de asignación que aparecen en esta imagen:



Para más información sobre las funciones de agregado de la biblioteca **core** consulte el apartado core | aggregate functions (agregado).

Solución nº2: usar una hoja de estilos XSLT personal

El objetivo del ejemplo es sumar los campos `Price` de los productos del archivo XML de origen (en este caso los productos A y B).

```
<?xml version="1.0" encoding="UTF-8"?>
```


© 2018 Altova GmbH

```
<Input xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="input.xsd">
   <Products>
      <Product>
         <Name>ProductA</Name>
         <Amount>10</Amount>
         <Price>5</Price>
      </Product>
      <Product>
         <Name>ProductB</Name>
         <Amount>5</Amount>
         <Price>20</Price>
      </Product>
   </Products>
</Input>
```

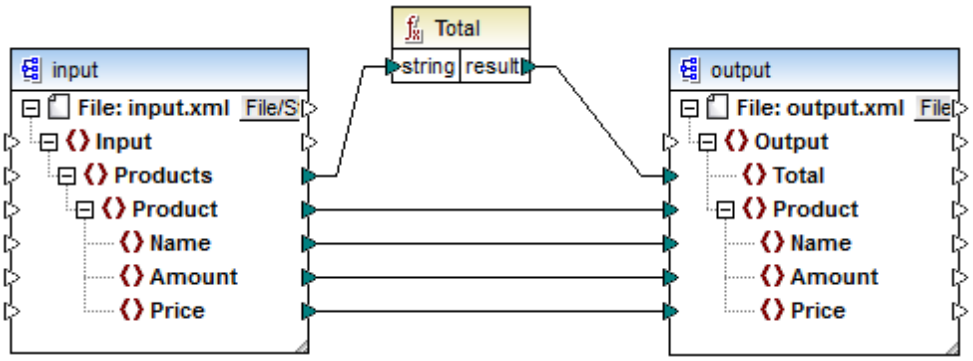
A continuación puede ver una hoja de estilos XSLT personal que usa la plantilla con nombre "Total" y un solo parámetro `string`. La plantilla recorre el archivo XML de entrada y suma todos los valores que obtiene la expresión `/Product/Price`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/
Transform">
   <xsl:output method="xml" version="1.0" encoding="UTF-8" indent="yes"/>

   <xsl:template match="*">
      <xsl:for-each select=".">
      <xsl:call-template name="Total">
         <xsl:with-param name="string" select="."/>
      </xsl:call-template>
      </xsl:for-each>
   </xsl:template>

   <xsl:template name="Total">
   <xsl:param name="string"/>
      <xsl:value-of select="sum($string/Product/Price)"/>
   </xsl:template>
</xsl:stylesheet>
```

Nota: para sumar los nodos en XSLT 2.0 debe cambiar la declaración de hoja de estilos por `version="2.0"`.

Para importar la hoja de estilos XSLT en MapForce:

1. Seleccione el lenguaje de transformación XSLT.
2. En la ventana Bibliotecas haga clic en el botón **Agregar o quitar bibliotecas**.
3. En el cuadro de diálogo "Opciones" haga clic en el botón **Agregar**.
4. Navegue hasta el archivo **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Summing-nodes.xslt**.
5. Arrastre la función **`Total`** de la biblioteca recién añadida (**Summing-nodes**) hasta el área de asignación y cree las conexiones que aparecen en esta imagen:



Haga clic en el panel *Resultados* para obtener una vista previa del resultado de la asignación. La suma de los dos campos `Price` aparecen ahora en el campo `Total`.

```
<?xml version="1.0" encoding="UTF-8"?>
<Output xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="output.xsd">
   <Total>25</Total>
   <Product>
      <Name>ProductA</Name>
      <Amount>10</Amount>
      <Price>5</Price>
   </Product>
   <Product>
      <Name>ProductB</Name>
      <Amount>5</Amount>
      <Price>20</Price>
   </Product>
</Output>
```


© 2018 Altova GmbH

8.4 Importar funciones XQuery 1.0 personales

Cuando el lenguaje de transformación seleccionado es XQuery, MapForce muestra las bibliotecas de funciones XQuery en la ventana Bibliotecas. Si quiere, puede ampliar esta biblioteca con sus propias funciones XQuery personales con solo importar sus módulos de biblioteca XQuery 1.0 personales en MapForce.

Para poder importarse en MapForce, el archivo XQuery debe cumplir estos requisitos:

- Debe ser un módulo de biblioteca válido según lo estipulado por la especificación XQuery. Es decir, debe empezar con una declaración de módulo como `module namespace <prefix>="<namespace name"`
- Todas las funciones declaradas en el módulo de biblioteca importado deben devolver tipos de datos atómicos (p. ej. `xs:string`, `xs:boolean`, `xs:integer`, etc.). Los parámetros de las funciones también deben tener tipos atómicos.

Para importar un módulo de biblioteca XQuery:

1. En el menú **Herramientas** haga clic en **Opciones** (o haga clic en el botón **Agregar o quitar bibliotecas** situado en la parte inferior de la ventana Bibliotecas).
2. Haga clic en el botón **Agregar** y navegue hasta el archivo de biblioteca **.xq** o **.xquery**.

Si el módulo de biblioteca que desea importar no es compatible, MapForce emitirá un aviso al respecto. Si el módulo es compatible, aparecerá en la ventana Bibliotecas.

Si no puede ver el módulo de biblioteca XQuery importado, asegúrese de que el lenguaje de transformación que está seleccionado es XQuery (*véase* *Seleccionar un lenguaje de transformación*).

Temas relacionados:

Implementación del motor XQuery

8.5 Importar bibliotecas Java y .NET personales

En MapForce puede importar archivos de clases Java compilados y archivos de ensamblados .NET (incluidos los ensamblados .NET 4.0). Si las bibliotecas importadas contienen funciones que usan tipos de datos básicos como parámetros y que devuelven tipos simples, dichas funciones aparecerán en la ventana Biblioteca y se podrán usar en los diseños de asignación de datos como cualquier otra función de MapForce. El resultado de las asignaciones de datos que incluyen funciones Java y .NET importadas se puede ver en el panel *Resultados* y las funciones estarán disponibles en el código generado.

Notas:

- para importar funciones Java o .NET personales necesitará los archivos de clases Java compilados (.class) o los archivos de ensamblados .NET. No se pueden importar archivos Java .jar ni archivos .dll que no sean ensamblados .NET.
- los archivos de clases Java compilados (.class) se pueden importar si el lenguaje de transformación seleccionado es Java. Es necesario tener instalado Java Runtime Environment 7 o superior. Solamente son compatibles ciertos tipos y miembros (*véase Funciones Java compatibles*).
- los archivos de ensamblados .NET se pueden importar si el lenguaje de transformación seleccionado es C#. Los ensamblados .NET también pueden estar escritos en lenguajes .NET como C++.NET o VB.NET, siempre y cuando solamente usen como parámetros tipos de datos básicos de System Assembly y tipos devueltos (*véase Funciones .NET compatibles*).
- si la asignación usa archivos de clases Java o ensamblados .NET importados, no podrá seleccionar el lenguaje de transformación C++.
- la importación de funciones de ensamblados C++ nativos tiene algunas limitaciones y requiere un procedimiento especial (*véase Referencias manuales a bibliotecas Java, C# y C++ personales*).
- si la asignación usa archivos de clases Java o ensamblados .NET importados, no podrá seleccionar el lenguaje de transformación XSLT (deberá escribir una función XSLT personal que haga de adaptador).
- todas las funciones a las que llame una asignación de MapForce deben ser idempotente (esto significa que deben devolver el mismo valor cada vez que se llame a la función con los mismos parámetros de entrada). El orden exacto y el número de veces que MapForce llama a una función es indefinido.
- en el caso de los archivos de clases Java importados y de sus paquetes, no es necesario que se añadan a la variable CLASSPATH porque el motor de ejecución integrado y el código Java de salida agregarán automáticamente los paquetes importados a la variable classpath del motor Java o a Ant. Sin embargo, las dependencias de los archivos de clases y paquetes importados no se resolverán automáticamente. Por tanto, si los archivos de clases o paquetes Java dependen de otros archivos de clases, deberá agregar los directorios primarios de todos los paquetes dependientes a la variable de entorno CLASSPATH.

Funciones Java compatibles

Clases de nivel superior, clases de miembro estáticas y clases de miembro no estáticas:

- `new <nombreClase>(<arg1>, <arg2>, ...)`
- `<objeto>.new <clase-miembro>(<arg1>, <arg2>, ...)`

© 2018 Altova GmbH

Funciones de miembro y estáticas:

- `<función>(<arg1>, <arg2>, ...)`
- `<objeto>.<método>(<arg1>, ...)`

Conexiones compatibles entre tipos XML Schema y tipos Java:

Tipo XML Schema	Tipo Java
xs:string	String
xs:byte	byte
xs:short	short
xs:int	int
xs:long	long
xs:boolean	boolean
xs:float	float
xs:double	double
xs:decimal	java.math.BigDecimal
xs:integer	java.math.BigInteger

Las conexiones pueden hacerse en ambas direcciones. Los demás tipos Java (incluidos los tipos matriz) no son compatibles. Los métodos que usen dichos parámetros o valores devueltos se omitirán.

Se pueden usar tipos de objeto llamando a su constructor o como valor devuelto de un método. Se pueden asignar a otros métodos Java, pero no se puede manipular el objeto en MapForce.

Funciones .NET compatibles

Clases de nivel superior y clases de miembro:

- `new <nombreClase>(<arg1>, <arg2>, ...)`

Funciones de miembro y funciones estáticas:

- `<función>(<arg1>, <arg2>, ...)`
- `<objeto>.<método>(<arg1>, ...)`

Conexiones compatibles entre tipos XML Schema y tipos .NET/C#:

Tipos XML Schema	Tipos .NET	Tipos C#
xs:string	System.String	string
xs:byte	System.SByte	sbyte
xs:short	System.Int16	short

Tipos XML Schema	Tipos .NET	Tipos C#
xs:int	System.Int32	int
xs:long	System.Int64	long
xs:unsignedByte	System.Byte	byte
xs:unsignedShort	System.UInt16	ushort
xs:unsignedInt	System.UInt32	uint
xs:unsignedLong	System.UInt64	ulong
xs:boolean	System.Boolean	bool
xs:float	System.Single	float
xs:double	System.Double	double
xs:decimal	System.Decimal	decimal

Las conexiones pueden hacerse en ambas direcciones. Los demás tipos .NET/C# (incluidos los tipos matriz) no son compatibles. Los métodos que usen dichos parámetros o valores devueltos se omitirán.

Se pueden usar tipos de objeto llamando a su constructor o como valor devuelto de un método. Se pueden asignar a otros métodos .NET, pero no se puede manipular el objeto en MapForce.

8.5.1 Ejemplo: importar clase Java personales

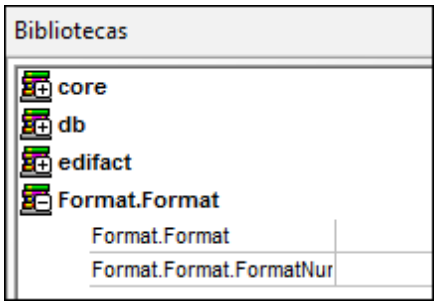
Este ejemplo explica cómo importar un archivo de clases Java personal en MapForce.

Nota: para este ejemplo es necesario tener Java SE 7 Runtime Environment o superior.

Para agregar el archivo .class de Java como biblioteca de MapForce:

1. Elija el lenguaje de transformación Java.
2. Haga clic en el botón **Agregar o quitar bibliotecas** situado en la parte inferior de la ventana Bibliotecas.
3. Ahora haga clic en el botón **Agregar** y seleccione el archivo **Format.class** de la carpeta **...\MapForceExamples\Java\Format**. Aparece un mensaje confirmando que se añadió una biblioteca nueva. La biblioteca importada aparece en la ventana Bibliotecas.

© 2018 Altova GmbH



Para obtener una vista previa del resultado de la asignación en MapForce:

1. Abra el archivo **FormatNumber.mfd** situado en la carpeta **...\MapForceExamples\Java**.
2. Abra el panel *Resultados* para ver el resultado de la asignación.

```
 1   Start date,End date,Region,Amount
 2   2008-01-01,2008-01-31,CA,"110.400,00"
 3   2008-01-01,2008-01-31,MA,"75.300,00"
 4   2008-02-01,2008-02-29,CA,"114.300,00"
 5   2008-02-01,2008-02-29,MA,"65.200,00"
 6   2008-03-01,2008-03-31,CA,"134.200,00"
 7   2008-03-01,2008-03-31,MA,"86.100,00"
 8   2008-04-01,2008-04-30,CA,"107.300,00"
 9   2008-04-01,2008-04-30,MA,"112.100,00"
10   2008-05-01,2008-05-31,CA,"114.400,00"
11   2008-05-01,2008-05-31,MA,"93.800,00"
```

Para ejecutar la asignación en Java:

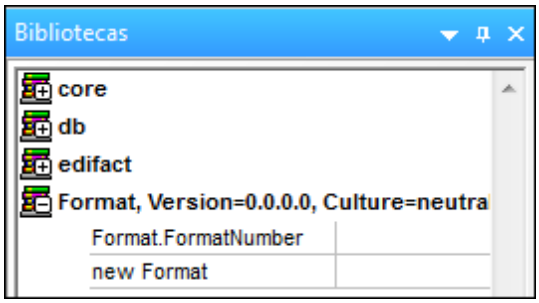
1. En el menú **Archivo** haga clic en **Generar código en | Java**.
2. Seleccione el directorio de destino donde se debe generar el código y haga clic en **Aceptar**.
3. Importe las bibliotecas generadas a su proyecto de Java y genere la aplicación Java (*véase* *Ejemplo: generar una aplicación Java con Eclipse y Ant*).

8.5.2 Ejemplo: importar ensamblado .NET DLL personal

Este ejemplo explica cómo importar un archivo de ensamblados .NET DLL creado en C#.

Para agregar el archivo de ensamblados .NET:

1. Elija el lenguaje de transformación C#.
2. Haga clic en el botón **Agregar o quitar bibliotecas** situado en la parte inferior de la ventana Bibliotecas.
3. Ahora haga clic en el botón **Agregar** y seleccione el archivo **Format.dll** de la carpeta **...\MapForceExamples\C#\Format\bin\Debug**. Aparece un mensaje confirmando que se añadió una biblioteca nueva. La biblioteca importada aparece en la ventana Bibliotecas.



Para obtener una vista previa del resultado de la asignación en MapForce:

1. Abra el archivo **FormatNumber.mfd** situado en la carpeta **...\MapForceExamples\C#**.
2. Abra el panel *Resultados* para ver el resultado de la asignación.

```
 1   Start date,End date,Region,Amount
 2   2008-01-01,2008-01-31,CA,"110.400,00"
 3   2008-01-01,2008-01-31,MA,"75.300,00"
 4   2008-02-01,2008-02-29,CA,"114.300,00"
 5   2008-02-01,2008-02-29,MA,"65.200,00"
 6   2008-03-01,2008-03-31,CA,"134.200,00"
 7   2008-03-01,2008-03-31,MA,"86.100,00"
 8   2008-04-01,2008-04-30,CA,"107.300,00"
 9   2008-04-01,2008-04-30,MA,"112.100,00"
10   2008-05-01,2008-05-31,CA,"114.400,00"
11   2008-05-01,2008-05-31,MA,"93.800,00"
```

Para ejecutar la asignación desde una aplicación C# personal:

1. En el menú **Archivo** haga clic en **Generar código en | C#**.
2. Seleccione el directorio de destino donde se debe generar el código y haga clic en **Aceptar**.
3. Genere la aplicación con Visual Studio y ejecute la aplicación de consola generada (*véase* *Generar código C#*).

© 2018 Altova GmbH

8.6 Referencias manuales a bibliotecas Java, C# y C++ personales

Además de importar bibliotecas personales en MapForce, puede crear referencias a dichas bibliotecas con ayuda de un archivo de funciones MapForce personal (archivo .mff). El archivo de bibliotecas .mff es un archivo XML donde se definen a mano las correspondencias entre definiciones de clase del código personal y de MapForce. Tras crear el archivo .mff personal podrá importarlo en MapForce como si se tratase de un archivo de clases Java o de ensamblados .NET.

Notas:

- Para poder aparecer en la ventana Bibliotecas de MapForce las funciones importadas deben tener tipos devueltos y parámetros de tipo simple. Puede consultar listas de tipos de datos para cada lenguaje en el apartado Correspondencias entre tipos de datos.
- Cuando importe bibliotecas de funciones de archivos .mff personales, la vista previa de resultados tendrá algunas limitaciones. Para las bibliotecas escritas en C++ la vista previa no estará disponible. En el caso de Java y C#, la vista previa estará disponible cuando la biblioteca use tipos de lenguaje nativos, pero no lo estará si la biblioteca importa las clases generadas de Altova. Sin embargo, recuerde que puede generar código en el lenguaje al que hace referencia su biblioteca. Las funciones personales estarán disponibles en el código generado. Es decir, podrá ejecutar la asignación a partir del código generado.
- El orden exacto en el que la asignación llama a las funciones es indefinido. MapForce puede almacenar en caché los resultados calculados o evaluar las expresiones en cualquier orden. Por tanto, se recomienda utilizar funciones personales que no tengan efectos secundarios.
- Es importante distinguir entre funciones definidas por el usuario y bibliotecas de funciones personales. Las funciones definidas por el usuario se crean de forma gráfica dentro de una asignación de datos y no pueden ni deben guardarse en un archivo *.mff porque se guardan en el archivo de asignación .mfd donde se crearon. Para más información consulte la sección Funciones definidas por el usuario.
- Si su versión es anterior a la versión 2010, puede que necesite actualizar los tipos de datos utilizados en sus funciones personales (*véase Asignaciones de tipos de datos*).

Para aprender a crear y configurar un archivo .mff personal consulte el apartado Configurar el archivo .mff.

Para ver algunos ejemplos consulte estos apartados:

- Ejemplo: crear una biblioteca C# personal
- Ejemplo: crear una biblioteca C++ personal
- Ejemplo: crear una biblioteca Java personal

8.6.1 Configurar el archivo .mff

El archivo de funciones de MapForce (.mff) es un archivo de configuración en formato XML que permite adaptar funciones de bibliotecas Java, C# y C++ personales a MapForce para que aparezcan en la ventana Bibliotecas. Un archivo .mff hace de intermediario entre las bibliotecas personales del usuario y MapForce y debe configurarse para especificar:

- las interfaces con las funciones personales y
- dónde se puede encontrar la implementación en el código generado.

En este apartado explicamos cómo conseguir estos dos objetivos.

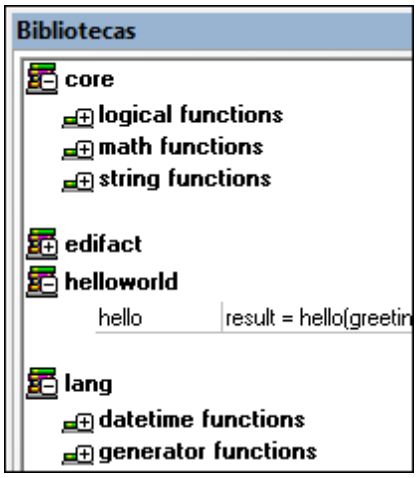
Nota: los archivos de biblioteca ***.mff** deben ser válidos según el archivo de esquema **mff.xsd** de la carpeta **Altova\MapForce2018\MapForceExamples**, relativa a la carpeta **(Mis) Documentos**. El esquema **mff.xsd** defina la configuración de biblioteca personal y es para uso interno. Altova GmbH se reserva el derecho de cambiar el formato de este archivo en futuras versiones del software.

Este fragmento de código corresponde a un archivo .mff para C#:

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xsi:noNamespaceSchemaLocation="mff.xsd" version="8" library="helloworld">
   <implementations>
      <implementation language="cs">
         <setting name="namespace" value="HelloWorldLibrary"/>
         <setting name="class" value="Greetings"/>
         <setting name="reference" value="C:\HelloWorldLibrary
\HelloWorldLibrary.dll"/>
      </implementation>
   </implementations>
   <group name="string functions">
      <component name="hello">
         <sources>
            <datapoint name="greeting_type" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="cs">
               <function name="HelloFunction"/>
            </implementation>
         </implementations>
         <description>
            <short>result = hello(greeting_type)</short>
            <long>Returns a greeting sentence according to the given
greeting_type.</long>
         </description>
      </component>
   </group>
</mapping>
```

La imagen que aparece a continuación muestra el resultado de importar un archivo .mff personal en MapForce. Observe que la biblioteca personal **helloworld** aparece en la ventana Bibliotecas y que contiene la función de cadena "hello".

© 2018 Altova GmbH



Más abajo encontrará instrucciones para adaptar el archivo .mff a los requisitos de su proyecto.

Configurar el nombre de la biblioteca

El nombre de la biblioteca está en la línea del archivo .mff que aparece a continuación. El nombre de la biblioteca se escribe en minúsculas.

```
<mapping xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xsi:noNamespaceSchemaLocation="mff.xsd" version="8" library="helloworld">
```

En este ejemplo, la biblioteca aparecerá en la ventana Bibliotecas de MapForce con el nombre "helloworld".

Configurar las implementaciones del lenguaje

El elemento `<implementations>` es un elemento obligatorio que especifica con qué lenguajes debe ser compatible la biblioteca y debe ser un secundario de `<mapping>`. Veamos un ejemplo:

```
...
<mapping xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xsi:noNamespaceSchemaLocation="mff.xsd" version="8" library="helloworld">
   <implementations>
      <implementation language="cs">
         <setting name="namespace" value="HelloWorldLibrary"/>
         <setting name="class" value="Greetings"/>
         <setting name="reference" value="C:\HelloWorldLibrary
\HelloWorldLibrary.dll"/>
      </implementation>
   </implementations>
...
```

Las opciones de cada elemento `<implementation>` permiten al código generado llamar a las funciones definidas en Java, C++ o C#.

Puede configurar el archivo .mff para que apunte a más de un lenguaje de programación. En este caso, cada lenguaje debe contener un elemento `<implementation>`. Más abajo encontrará información sobre las opciones disponibles para cada lenguaje de programación.

Java

```
...
<implementation language="java">
   <setting name="package" value="com.hello.functions"/>
   <setting name="class" value="Hello"/>
</implementation>
...
```

Es importante que el código generado pueda encontrar su archivo **Hello.class**. Por tanto, asegúrese de que su clase está en la variable classpath de Java. La variable classpath de Java predeterminada se encuentra en las variables de entorno del sistema.

Recuerde que, cuando se trabaja con bibliotecas Java personales, solamente se puede tener una clase por cada archivo *.mff.

C#

```
...
   <implementation language="cs">
      <setting name="namespace" value="HelloWorldLibrary"/>
      <setting name="class" value="Hello"/>
      <setting name="reference" value=" C:\HelloWorldLibrary
\HelloWorldLibrary.dll"/>
   </implementation>
...
```

Es importante que el espacio de nombres del código se corresponda con el espacio de nombres definido en el archivo .mff (en el fragmento de código anterior el espacio de nombres es `HelloWorldLIbrary`). Lo mismo ocurre con el nombre de la clase (en el fragmento de código anterior el nombre de clase es `Hello`). La tercera opción, `reference`, aporta la ruta de acceso del .dll que debe vincularse al código generado.

C++

```
...
   <implementation language="cpp">
         <setting name="namespace" value="helloworld"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\HelloWorldLibrary"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
   </implementation>
...
```


© 2018 Altova GmbH

En el fragmento de código C++ anterior podemos ver que:

- `namespace` es el espacio de nombres donde se definirá la clase `Greetings`. Debe ser igual que el nombre de la biblioteca.
- `path` es la ruta de acceso donde se encuentran los archivos de inclusión y de origen.
- cuando se genera código para una asignación de datos, los archivos de inclusión y de origen se copian en el directorio **targetdir/libraryname** (definido cuando se selecciona el comando de menú **Archivo | Generar código xxx** y se elije el directorio de destino) y también se incluyen en el archivo de proyecto.

Todos los archivos de inclusión se incluirán en el algoritmo que se genera.

Agregar un componente

En la ventana Bibliotecas de MapForce las funciones aparecen anidadas dentro de un grupo de funciones (p. ej. "string functions"). En el archivo .mff una función corresponde a un elemento `<component>`. Igualmente, cada `<component>` debe estar anidado dentro de un elemento `<group>`. Por ejemplo:

```
...
<group name="string functions">
   <component name="hello">
      …
   </component>
</group>
...
```

El código que aparece a continuación define un ejemplo de función (componente) llamada `hello`.

```
...
<component name="hello">
   <sources>
      <datapoint name="greeting_type" type="xs:boolean"/>
   </sources>
   <targets>
      <datapoint name="result" type="xs:string"/>
   </targets>
   <implementations>
   …
   </implementations>
   <description>
      <short>result = hello(greeting_type)</short>
      <long>Returns a greeting sentence according to the given
greeting_type.</long>
   </description>
</component>
...
```

Este sería el aspecto del componente anterior en MapForce:



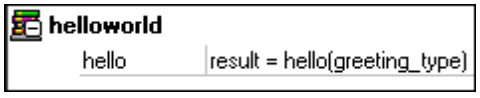
En el fragmento de código, se definió `<datapoint>` como parámetro de entrada o salida de una función (o conector de entrada o salida). El argumento `type` de `<datapoint>` especifica el tipo de datos del parámetro (o el tipo de datos del valor devuelto).

Por cada función se permite un punto de datos de destino como máximo, pero no hay límite en el número de puntos de datos de origen que se pueden definir.

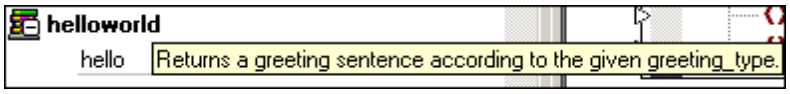
El tipo de datos de cada punto de datos debe ser un tipo XML Schema (p. ej. `xs:string`, `xs:integer`, etc.). Estos tipos de datos deben corresponder a los tipos de datos de los parámetros de la función que definió en la biblioteca Java, C++ o C#. Para ver la correspondencia entre tipos de datos XML Schema y de otros lenguajes consulte el apartado Correspondencias entre tipos de datos.

En la ventana Bibliotecas de MapForce las funciones van acompañadas de una descripción breve y una más larga. La descripción breve aparece siempre a la derecha del nombre de la función. La descripción más larga aparece cuando se pasa el puntero por encima de la descripción breve.

Descripción breve:



Descripción larga:



Definir implementaciones de lenguaje

Llegados a este punto debemos crear una conexión entre la función de la ventana Bibliotecas y la función de las clases Java, C# o C++ personales. Esto se consigue con el elemento `<implementation>`.

Tal y como se explica más a arriba, las funciones pueden tener varios elementos `<implementation>`, uno por cada lenguaje de programación compatible. Una función se puede llamar "helloFunction" en Java o "HelloFunctionResponse" en C++. Por eso es necesario especificar un nombre de función distinto por cada lenguaje de programación. A continuación puede ver una función con tres implementaciones distintas:

```
...
<component name="hello">
...
   <implementations>
      <implementation language="cs">
         <function name="HelloFunction"/>
```

 © 2018 Altova GmbH

```
      </implementation>
      <implementation language="java">
         <function name="helloFunction"/>
      </implementation>
      <implementation language="cpp">
         <function name="HelloFunctionResponse"/>
      </implementation>
   </implementations>
...
</component>
...
```

El valor que especifique como nombre de función debe ser idéntico al nombre del método de la clase Java, C# o C++.

8.6.2 Importar el archivo .mff a MapForce

Tras crear el archivo .mff personal (*véase* *Configurar el archivo .mff*), podrá importarlo en MapForce siguiendo estas instrucciones:

1. En el menú **Herramientas** haga clic en **Opciones** (o haga clic en el botón **Agregar o quitar bibliotecas** situado en la parte inferior de la ventana Bibliotecas).
2. Ahora haga clic en el botón **Agregar** y seleccione el archivo .mff.

La biblioteca importada aparece en la ventana Bibliotecas si el lenguaje de transformación seleccionado es el lenguaje al que apunta la biblioteca personal recién importada.

Si guarda el archivo *.mff en la carpeta **...\Altova\MapForce2018\MapForceLibraries** relativa a la carpeta **Archivos de programa** (o **Archivos de programa (x86)**), la biblioteca se cargará automáticamente en la ventana Bibliotecas cuando inicie MapForce. Las bibliotecas y sus funciones se pueden habilitar o deshabilitar con solo agregar o eliminar el correspondiente archivo de biblioteca (*.mff).

8.6.3 Correspondencias entre tipos de datos

En esta tabla puede ver los tipos de datos que se admiten como tipos devueltos de función y tipos de parámetro de función cuando se crean archivos .mff personales que adaptarán sus bibliotecas Java, C# y C++ personales a MapForce. La tabla incluye tanto tipos de datos nativos como no nativos. Recuerde que si necesita usar tipos de datos no nativos como los tipos date, time y duration de Altova, sus bibliotecas Java y C# personales deberán incluir una referencia a las bibliotecas de Altova. En el caso de C++, deberá importar las bibliotecas de Altova. Para más información sobre cómo generar las bibliotecas de Altova consulte la sección Generador de código.

Tipo XML Schema	Tipo Java	Tipo C#	Tipo C++
`anyAtomicType`	`String`	`string`	`string_type`

Tipo XML Schema	Tipo Java	Tipo C#	Tipo C++
anySimpleType	String	string	string_type
anyURI	String	string	string_type
base64Binary	byte[]	byte[]	altova::mapforce::blob
boolean	boolean	bool	bool
byte	int	int	int
date	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
dateTime	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
dayTimeDuration	com.altova.types.Duration	Altova.Types.Duration	altova::Duration
decimal	java.math.BigDecimal	decimal	double
double	double	double	double
duration	com.altova.types.Duration	Altova.Types.Duration	altova::Duration
ENTITIES	String	string	string_type
ENTITY	String	string	string_type
float	double	double	double
gDay	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gMonth	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gMonthDay	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gYear	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
gYearMonth	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
hexBinary	byte[]	byte[]	altova::mapforce::blob
ID	String	string	string_type
IDREF	String	string	string_type
IDREFS	String	string	string_type
int	int	int	int

© 2018 Altova GmbH

Tipo XML Schema	Tipo Java	Tipo C#	Tipo C++
integer	java.math.BigInteger	decimal	__int64
language	String	string	string_type
long	long	long	__int64
Name	String	string	string_type
NCName	String	string	string_type
negativeInteger	java.math.BigInteger	decimal	__int64
NMTOKEN	String	string	string_type
NMTOKENS	String	string	string_type
nonNegativeInteger	java.math.BigInteger	decimal	unsigned __int64
nonPositiveInteger	java.math.BigInteger	decimal	__int64
normalizedString	String	string	string_type
NOTATION	String	string	string_type
positiveInteger	java.math.BigInteger	decimal	unsigned __int64
QName	javax.xml.namespace.QName	Altova.Types.QName	altova::QName
short	int	int	int
string	String	string	string_type
time	com.altova.types.DateTime	Altova.Types.DateTime	altova::DateTime
token	String	string	string_type
unsignedByte	long	ulong	unsigned __int64
unsignedInt	long	ulong	unsigned __int64
unsignedLong	java.math.BigInteger	ulong	unsigned __int64
unsignedShort	long	ulong	unsigned __int64
untypedAtomic	String	string	string_type
yearMonthDuration	com.altova.types.Duration	Altova.Types.Duration	altova::Duration

8.6.4 Ejemplo: crear una biblioteca C# personal

Este apartado explica cómo se puede crear una biblioteca C# y configurar un archivo .mff para que esa biblioteca aparezca en la ventana Bibliotecas de MapForce:

1. Cree un proyecto de bibliotecas de clases nuevo en Visual Studio. Observe que la función se definió como `public static`.

```
using System;

namespace HelloWorldLibrary
{
    public class Greetings
    {
        public static string HelloFunction(bool GreetingType)
        {
            if (GreetingType)
                return "Hello World!";
            return "Hello User!";
        }
    }
}
```

2. Si necesita tipos XML Schema especiales (como date y duration), deberá agregar una referencia desde el proyecto de Visual Studio a la biblioteca **Altova.dll**. Para obtener esta biblioteca debe generar código C# a partir de una asignación de datos de MapForce con funciones personales. El archivo **Altova.dll** se guarda en el directorio **..\Altova\bin\debug** relativo al directorio donde se generó el código. Para agregar una referencia a **Altova.dll** en Visual Studio haga clic en **Proyecto | Agregar referencia** y navegue hasta el archivo **Altova.dll**. Además, debe añadir esta línea a su código: `using Altova.Types;`. Para más información sobre las correspondencias entre tipos XML Schema y tipos C# consulte el apartado Correspondencias entre tipos de datos.
3. Genere su proyecto de Visual Studio. El archivo **HelloWorldLibrary.dll** se genera en el directorio de salida del proyecto.
4. En un editor XML cree un archivo .mff nuevo y valídelo con el esquema **..\Archivos de programa\MapForceLibraries\mff.xsd**. Asegúrese de que el texto que aparece resaltado más abajo apunta al archivo **HelloWorldLibrary.dll**. Para más información consulte el apartado Configurar el archivo .mff.

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="helloworld" xmlns:xs="http://
www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cs">
         <setting name="namespace" value="HelloWorldLibrary"/>
         <setting name="class" value="Greetings"/>
         <setting name="reference" value="C:\Projects
\HelloWorldLibrary.dll"/>
      </implementation>
```


© 2018 Altova GmbH

```
   </implementations>
   <group name="Greetings">
      <component name="HelloFunction">
         <sources>
            <datapoint name="greeting_type" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string" />
         </targets>
         <implementations>
            <implementation language="cs">
               <function name="HelloFunction"/>
            </implementation>
         </implementations>
         <description>
            <short>result = hello(greeting_type)</short>
            <long>Returns a greeting according to the given greeting
type.</long>
         </description>
      </component>
   </group>
</mapping>
```

Ya ha terminado de crear una biblioteca personal y el archivo .mff que adapta esta biblioteca a MapForce. El archivo .mff personal se puede usar en MapForce (*véase* *Importar el archivo .mff en MapForce*).

8.6.5 Ejemplo: crear una biblioteca C++ personal

Este apartado explica cómo se puede crear una biblioteca C++ y configurar un archivo .mff para que esa biblioteca aparezca en la ventana Bibliotecas de MapForce:

1. Cree un archivo de encabezado (.h) para su biblioteca de clases. Por ejemplo, a continuación puede ver un ejemplo de archivo de encabezado llamado **Greetings.h**.

```
#ifndef HELLOWORLDLIBRARY_GREETINGS_H_INCLUDED
#define HELLOWORLDLIBRARY_GREETINGS_H_INCLUDED

#if _MSC_VER > 1000
   #pragma once
#endif // _MSC_VER > 1000

using namespace altova;

namespace helloworld {

class ALTOVA_DECLSPECIFIER Greetings
{
public:
   static string_type   HelloFunctionResponse(bool greetingType);
};
```

```
} // namespace HelloWorldLibrary

#endif // HELLOWORLDLIBRARY_GREETINGS_H_INCLUDED
```

Observe que la función se declaró como estática y que el espacio de nombres `altova` se importa. Recuerde que debe escribir `ALTOVA_DECLSPECIFIER` delante del nombre de clase para que las clases se compilen correctamente (tanto si usa vínculos dinámicos o estáticos en el código que genere más adelante).

2. Cree un archivo .cpp que se llame igual que el archivo de encabezado. El archivo .cpp debe estar en el mismo directorio que el archivo .h. Por ejemplo, a continuación puede ver un ejemplo de archivo .cpp llamado **Greetings.cpp** que incluye el archivo **Greetings.h** que creamos en el primer paso.

```
#include "StdAfx.h"
#include "../Altova/Altova.h"
#include "../Altova/AltovaException.h"
#include "../Altova/SchemaTypes.h"

#include "Greetings.h"

namespace helloworld {

   string_type   Greetings::HelloFunctionResponse(bool greetingType)
   {
      if( greetingType )
         return _T("Hello World!");
      return _T("Hello User!");
   }

}
```

Fíjese en las líneas que importan el archivo **StdAfx.h** y varias bibliotecas de Altova. Estas líneas no se deben modificar. Las rutas de acceso de las bibliotecas de Altova son correctas. En el código que se generará estas rutas de acceso dirigirán a los archivos pertinentes.

A diferencia de Java y C#, en C++ no es necesario compilar los archivos de origen C ++. Se copiarán en el código generado y se compilarán con el resto del código de asignación.

3. En un editor XML cree un archivo .mff nuevo y valídelo con el esquema **..\Archivos de programa\MapForceLibraries\mff.xsd**. Asegúrese de que el texto que aparece resaltado más abajo apunta al directorio de los archivos de encabezado y cpp que creamos en los pasos anteriores. Recuerde que el espacio de nombres y los nombres de función y los tipos de datos aquí definidos deben corresponderse con los del código C++ tal y como se explica en el apartado Configurar el archivo .mff. Para más información sobre los tipos de datos consulte Correspondencias entre tipos de datos.

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="helloworld" xmlns:xs="http://
www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="cpp">
         <setting name="namespace" value="helloworld"/>
         <setting name="class" value="Greetings"/>
         <setting name="path" value="C:\Projects\HelloWorld"/>
         <setting name="include" value="Greetings.h"/>
         <setting name="source" value="Greetings.cpp"/>
      </implementation>
   </implementations>
   <group name="Greetings">
      <component name="HelloFunctionResponse">
         <sources>
            <datapoint name="greeting_type" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="cpp">
               <function name="HelloFunctionResponse"/>
            </implementation>
         </implementations>
         <description>
            <short>result = hello(greeting_type)</short>
            <long>Returns a greeting according to the given greeting
type.</long>
         </description>
      </component>
   </group>
</mapping>
```

Ya ha terminado de crear una biblioteca personal y el archivo .mff que adapta esta biblioteca a MapForce. El archivo .mff personal se puede usar en MapForce (*véase* *Importar el archivo .mff en MapForce*). Recuerde que, para poder ejecutar asignaciones que usan bibliotecas C++ nativas, deberá generar código C++ y ejecutar la asignación desde el código o aplicación C++.

Resolver errores de compilación C++

Si recibe un error de compilador en la línea que aparece a continuación, modifique las propiedades del proyecto para que incluyan una referencia al archivo **msado15.dll**.

```
#import "msado15.dll" rename("EOF", "EndOfFile")
```

En Visual Studio 2008:

1. Haga clic en **Herramientas | Opciones**.
2. Expanda **Proyectos y soluciones > Directorios VC++**.

3. Bajo "Mostrar directorios para", seleccione **Incluir archivos** y añada una entrada nueva que apunte al directorio donde está el archivo **msado15.dll** (por lo general se trata del directorio **C:\Archivos de programa\Common Files\System\ADO**).
4. Genere el proyecto.

8.6.6 Ejemplo: crear una biblioteca Java personal

Este apartado explica cómo se puede crear una biblioteca Java y configurar un archivo .mff para que esa biblioteca aparezca en la ventana Bibliotecas de MapForce:

1. Cree un proyecto de Java nuevo en su entorno de desarrollo (p. ej. Eclipse).
2. Añada al proyecto un paquete nuevo llamado `com.hello.functions` compuesto por una clase llamada `Hello`. A continuación puede ver un fragmento de código donde puede ver que la función `HelloFunction` se definió como `public static`.

```
package com.hello.functions;

public class Hello {
   public static String HelloFunction ( boolean greetingType ) {
      if( greetingType )
         return "Hello World!";
      return "Hello User!";
   }
}
```

3. También tiene la opción de importar el paquete `com.altova.types` si su proyecto necesita ser compatible con tipos especiales como date, time o duration. Para obtener este paquete debe generar código Java a partir de una asignación de datos que no incluya funciones personales.

```
import com.altova.types.*;
```

4. Compile la biblioteca personal en un archivo de clases y añádalo a la variable classpath de Java.
5. En un editor XML cree un archivo .mff nuevo y valídelo con el esquema **..\Archivos de programa\MapForceLibraries\mff.xsd**. Asegúrese de que el texto que aparece resaltado más abajo apunta al espacio de nombres y a la clase que definió en el código Java. Para más información consulte el apartado Configurar el archivo .mff.

```
<?xml version="1.0" encoding="UTF-8"?>
<mapping version="9" library="helloworld" xmlns:xs="http://
www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="mff.xsd">
   <implementations>
      <implementation language="java">
         <setting name="namespace" value="com.hello.functions"/>
         <setting name="class" value="Hello"/>
      </implementation>
   </implementations>
   <group name="Greetings">
```


© 2018 Altova GmbH

```
      <component name="HelloFunction">
         <sources>
            <datapoint name="greeting_type" type="xs:boolean"/>
         </sources>
         <targets>
            <datapoint name="result" type="xs:string"/>
         </targets>
         <implementations>
            <implementation language="java">
               <function name="HelloFunction"/>
            </implementation>
         </implementations>
         <description>
            <short>result = hello(greeting_type)</short>
            <long>Returns a greeting according to the given greeting
type.</long>
         </description>
      </component>
   </group>
</mapping>
```

Ya ha terminado de crear una biblioteca personal y el archivo .mff que adapta esta biblioteca a MapForce. El archivo .mff personal se puede usar en MapForce (*véase* *Importar el archivo .mff en MapForce*).

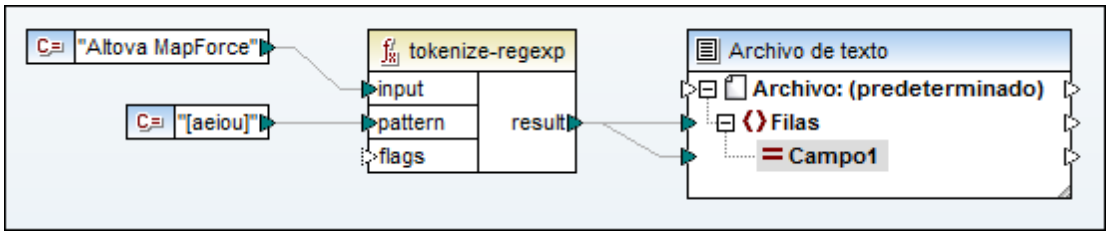
8.7 Expresiones regulares

MapForce admite el uso de expresiones regulares en el parámetro **pattern** de las funciones **match-pattern** y **tokenize-regexp** para buscar cadenas concretas.

La sintaxis y la semántica de las expresiones regulares para XLST y XQuery es la definida en http://www.w3.org/TR/xmlschema-2/. Tenga en cuenta que la sintaxis de las expresiones regulares varía de un lenguaje de programación a otro.

Terminología:

input	la cadena en la que trabaja la expresión regular
pattern	la expresión regular
flags	parámetro opcional para definir cómo se debe interpretar la expresión regular
result	el resultado de la función



La función **tokenize-regexp** devuelve una secuencia de cadenas. La conexión con elemento **Filas** crea una fila por cada elemento de la secuencia.

Sintaxis de las expresiones regulares

Literales (un solo carácter):
por ejemplo, la letra "a" es la expresión regular más básica. Busca la primera aparición del carácter "a" de la cadena.

Clases de caracteres []

Conjunto de caracteres entre corchetes.

Solamente se busca **uno** de los caracteres que aparecen entre corchetes.

pattern **[aeiou]**
Busca una vocal en minúsculas.

pattern **[mj]ust**
Busca *must* o *just.*

No olvide que el parámetro **pattern** distingue entre mayúsculas y minúsculas. Por ejemplo, **a** no encontrará **A**.

Intervalos de caracteres **[a-z]**
Crea el intervalo comprendido entre dos caracteres. Solamente se encontrará uno de los

© 2018 Altova GmbH

caracteres en cada búsqueda.

pattern **[a-z]**
Busca cualquier carácter de la **a** a la **z** que esté en minúsculas.

Clases negadas **[^]**
Si usa el acento circunflejo como primer carácter tras el corchete de apertura, se niega la clase de caracteres.

pattern **[^a-z]**
Busca caracteres que no estén en la clase de caracteres, incluidas las líneas nuevas.

Metacaracteres "."

Este metacarácter busca un solo carácter, sea cual sea (excepto líneas nuevas)

pattern **.**
Busca un solo carácter, sea cual sea.

Cuantificadores ? + * {}

Los cuantificadores definen cuántas veces debe aparecer un componente de la expresión regular dentro de la cadena **`input`** para que la búsqueda obtenga resultados.

?
cero o una — busca cero o una coincidencia de la cadena precedente (la cadena es opcional)

+
una o más — busca una o más coincidencias de la cadena precedente

*****
cero o más — busca cero o más coincidencias de la cadena precendente

{}
mínimo/máximo de repeticiones — número de repeticiones de la cadena o trozo

p. ej. mo{1,3} encuentra `mo`, `moo`, `mooo`.

()
subpatrones
los paréntesis se usan para agrupar partes de la expresión regular.

|
Alternancia/OR
permite probar de izquierda a derecha las subexpresiones
(horse|make) sense encuentra `horse sense` o `make sense`

Parámetro flags

Estos parámetros opcionales definen cómo se debe interpretar la expresión regular. Para establecer las opciones se usan letras, que pueden estar en cualquier orden y se pueden repetir.

s
Si está presente, el proceso de búsqueda opera en el modo *dot-all*.

El metacarácter "**.**" encuentra un carácter, sea el que sea. Si la cadena de entrada **`input`** contiene `hello` y `world` en dos líneas diferentes, la expresión `hello*world` solamente encontrará resultados si se establece la marca **`flag s`**.

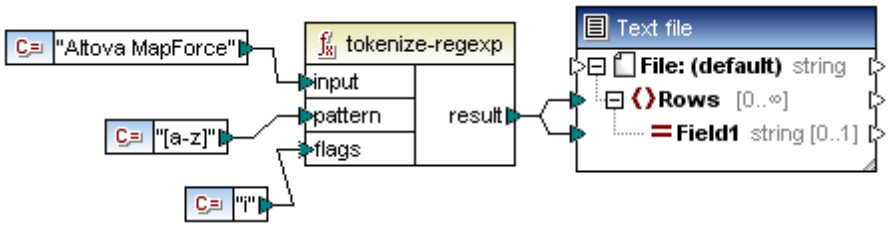
m
Si está presente, el proceso de búsqueda opera en el modo multilínea.

En el modo multilínea el acento circunflejo **^** busca el principio de una línea, **sea cual sea**. Es decir, el inicio de una cadena entera **y** el primer carácter que aparece después del carácter de línea nueva.

El carácter de dólar **$** busca el fin de una línea, **sea cual sea**. Es decir, el final de una cadena entera y el primer carácter que aparece antes del carácter de línea nueva.

El carácter de línea nueva es **`#x0A`**.

i
Si está presente, el proceso de búsqueda no distingue entre mayúsculas y minúsculas.
La expresión regular `[a-z]` más la marca **i** encontrará todas las letras de la a-z (en minúsculas) y entre A-Z (en mayúsculas).



x
Si está presente, los caracteres de espacio en blanco se quitan de la expresión regular antes de iniciar el proceso de búsqueda. Los caracteres de espacio en blanco son `#x09`, `#x0A`, `#x0D` y `#x20`.

Excepción:
Los caracteres de espacio en blanco sin expresiones de clases de caracteres no se eliminan (por ejemplo, `[#x20]`).

Nota: cuando se genera código, las características avanzadas de la sintaxis regex pueden variar de unos lenguajes a otros. Para más información consulte la documentación de su lenguaje.

© 2018 Altova GmbH

8.8 Referencia de la biblioteca de funciones

En esta sección se describen todas las funciones integradas de MapForce que aparecen en la ventana Bibliotecas.

Dependiendo del lenguaje de transformación que esté seleccionado, la ventana Bibliotecas mostrará unas funciones u otras (*véase Seleccionar un lenguaje de transformación*). La biblioteca **core** es una colección de funciones disponibles en los lenguajes C++, C#, Java y como mínimo uno de estos lenguajes: XQuery, XPath o XSLT. La biblioteca **lang** incluye funciones disponibles en los lenguajes C++, C# y Java. Las demás bibliotecas incluyen funciones asociadas con cada tipo de datos de salida.

Limitaciones de las funciones XPath 2.0: varias funciones XPath 2.0 relacionadas con secuencias no están disponibles.

8.8.1 core | aggregate functions (agregado)

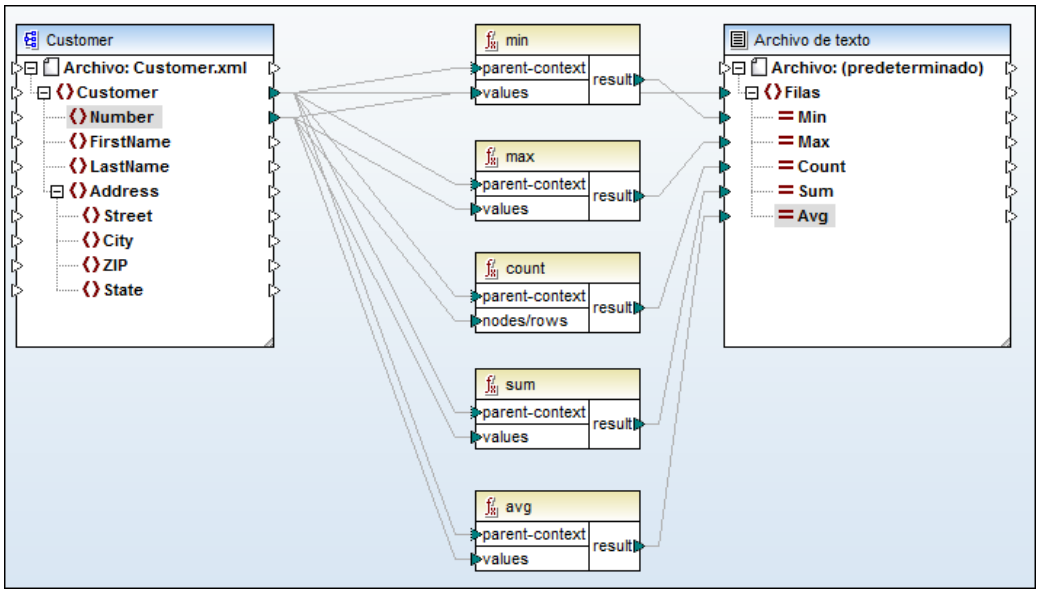
Las funciones de agregado de la biblioteca **core | aggregate** realizan operaciones en un conjunto o secuencia de valores de entrada. Los datos de entrada para las funciones **`min`**, **`max`**, **`sum`** y **`avg`** se convierten al tipo de datos `decimal` para poder procesarlos.

- Los valores de entrada deben conectarse al parámetro **`values`** de la función.
- Puede conectar un nodo de contexto (elemento) al parámetro **`parent-context`** para reemplazar el contexto predeterminado desde el que se toma la secuencia de entrada. El parámetro **`parent-context`** es un parámetro opcional.
- El parámetro **`result`** de la función debe conectarse al elemento de destino.

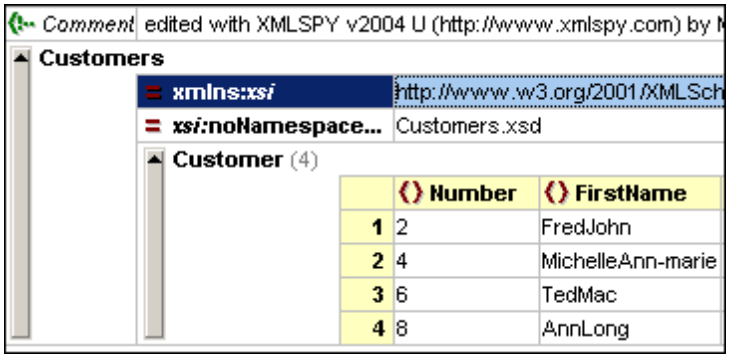
La asignación que aparece a continuación es la del archivo `Aggregates.mfd`, guardado en la carpeta `...\Tutorial` y explica cómo utilizar este tipo de funciones.

Las funciones de agregado tienen dos entradas:

- la entrada **`values`** (**`nodes/rows`**) se conecta al elemento del componente de origen que suministra los datos, en este caso el elemento **`Number`**.
- la entrada **`parent-context`** se conecta al elemento en el que se desea iterar (es decir, el contexto. En este caso se quiere iterar en todos los elementos **`Customer`**). Este parámetro es opcional.

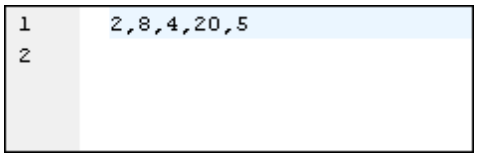


En este caso la instancia de entrada es un archivo XML que contiene estos datos:



- Los datos de origen suministrados a la entrada **values** es la secuencia de números 2,4,6,8.
- El componente de salida en este caso es un simple archivo de texto.

  La asignación anterior genera estos resultados:

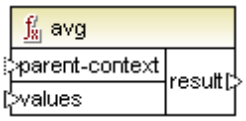


min=2, max=8, count=4, sum=20 y avg=5.

8.8.1.1 avg

Devuelve el valor promedio de todos los valores de la secuencia de entrada. El promedio de un conjunto de valores vacío es un conjunto vacío. Esta función no está disponible en XSLT1.

 © 2018 Altova GmbH

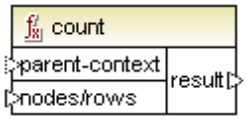


Argumento	Descripción
parent-context	Argumento opcional. Suministra el contexto primario (*véase Reemplazar el contexto de asignación*).
values	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. Recuerde que el valor de argumento dado debe ser numérico.

Para ver un ejemplo de uso consulte el diseño de asignación de datos **GroupTemperaturesByYear.mfd** del directorio **<Documentos>\Altova\MapForce2018\MapForceExamples**.

8.8.1.2 count

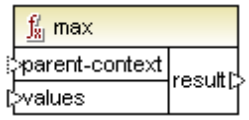
Devuelve el número de elementos que componen la secuencia de entrada. El recuento de un conjunto vacío es 0. Esta función no es totalmente compatible con XSLT1.



Argumento	Descripción
parent-context	Argumento opcional. Suministra el contexto primario (*véase Reemplazar el contexto de asignación*).
nodes/rows	Este argumento debe estar conectado al elemento de origen que se debe contar.

8.8.1.3 max

Devuelve el valor máximo de todos los valores numéricos de la secuencia de entrada. Tenga en cuenta que esta función devuelve un conjunto vacío si el argumento **strings** es un conjunto vacío. Esta función no está disponible en XSLT1.

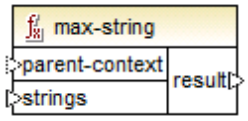


Argumento	Descripción
parent-context	Argumento opcional. Suministra el contexto primario (*véase Reemplazar el contexto de asignación*).
values	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. Recuerde que el valor de argumento dado debe ser numérico. Para obtener el máximo de una secuencia de cadenas utilice la función **`max-string`**.

Para ver un ejemplo de uso consulte el diseño de asignación de datos **GroupTemperaturesByYear.mfd** del directorio **<Documentos>\Altova\MapForce2018\MapForceExamples**.

8.8.1.4 max-string

Devuelve el valor máximo de todos los valores de cadena de la secuencia de entrada. Por ejemplo: `max-string("a", "b", "c")` devuelve "c". Esta función no está disponible en XSLT1.

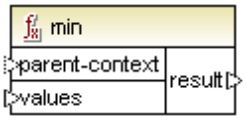


Argumento	Descripción
parent-context	Argumento opcional. Suministra el contexto primario (*véase Reemplazar el contexto de asignación*).
strings	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. El valor de argumento dado debe ser una secuencia (cero o varios) de `xs:string`.

Tenga en cuenta que la función devuelve un conjunto vacío si el argumento **strings** es un conjunto vacío.

8.8.1.5 min

Devuelve el valor mínimo de todos los valores numéricos de la secuencia de entrada. El mínimo de un conjunto vacío es un conjunto vacío. Esta función no está disponible en XSLT1.



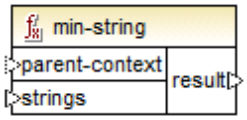
Argumento	Descripción
parent-context	Argumento opcional. Suministra el contexto primario (*véase Reemplazar el*

Argumento	Descripción
	contexto de asignación).
values	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. Recuerde que el valor de argumento dado debe ser numérico. Para obtener el mínimo de una secuencia de cadenas utilice la función **`min-string`**.

Para ver un ejemplo de uso consulte el diseño de asignación de datos **GroupTemperaturesByYear.mfd** del directorio **<Documentos>\Altova\MapForce2018\MapForceExamples**.

8.8.1.6 min-string

Devuelve el valor mínimo de todos los valores de cadena de la secuencia de entrada. Por ejemplo: `min-string("a", "b", "c")` devuelve "a". Esta función no está disponible en XSLT1.

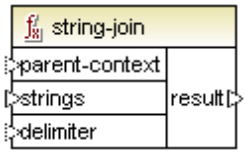


Argumento	Descripción
parent-context	Argumento opcional. Suministra el contexto primario (*véase Reemplazar el contexto de asignación*).
strings	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. El valor de argumento dado debe ser una secuencia (cero o varios) de `xs:string`.

Tenga en cuenta que la función devuelve un conjunto vacío si el argumento **strings** es un conjunto vacío.

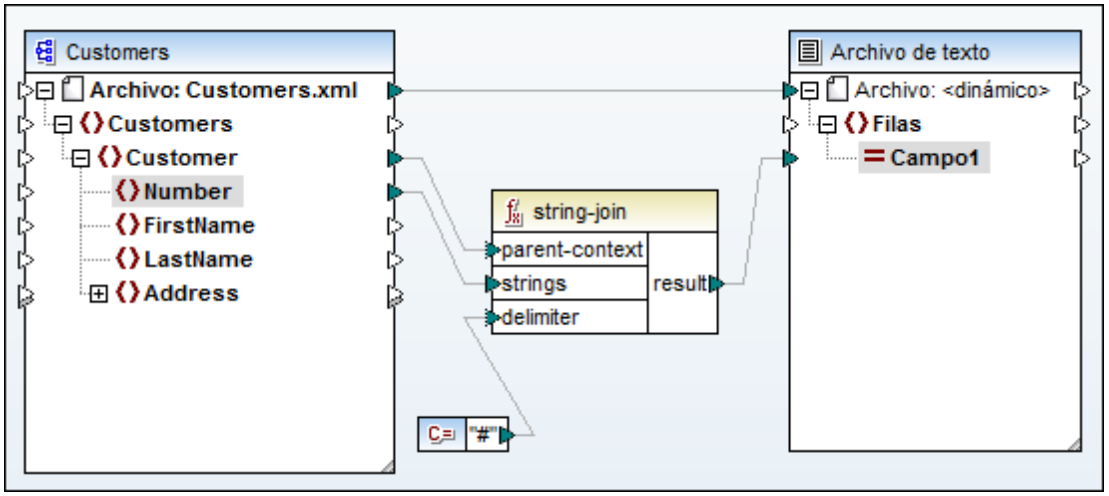
8.8.1.7 string-join

Enlaza todos los valores de la secuencia de entrada y forma una cadena delimitada por la cadena suministrada como delimitador al parámetro **`delimiter`**. Cuando se aplica a un conjunto vacío el resultado es una cadena vacía. Esta función no está disponible en XSLT1.



El ejemplo de la siguiente imagen muestra cuatro números: 2, 4, 6 y 8. La constante ofrece el carácter # como delimitador.

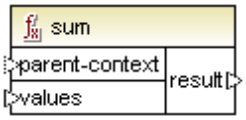
Resultado = 2#4#6#8



Si no especifica un delimitador (**delimiter**), el valor de argumento predeterminado es una cadena vacía (es decir, ningún delimitador). Resultado = 2468.

8.8.1.8 sum

Devuelve la suma aritmética de todos los valores de la secuencia de entrada. La suma de un conjunto vacío es 0.



Argumento	Descripción
parent-context	Argumento opcional. Suministra el contexto primario (*véase Reemplazar el contexto de asignación*).
values	Este argumento debe estar conectado a un elemento de entrada que suministra los datos propiamente dichos. Recuerde que el valor de argumento dado debe ser numérico.

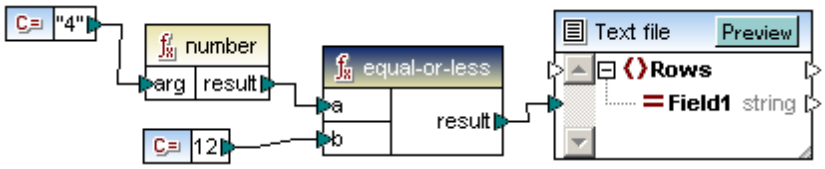
También puede consultar el apartado Ejemplo: sumar valores de nodo.

8.8.2 core | conversion functions (conversión)

MapForce ofrece varias funciones de conversión de tipos en la biblioteca **conversion** para realizar conversiones explícitas de tipos de datos. Tenga en cuenta que, en la mayoría de los casos, MapForce crea conversiones automáticamente y que solamente necesitará usar estas funciones en casos muy concretos.

Si los nodos de entrada son de tipos diferentes (p. ej. entero y cadena), puede usar las funciones de conversión para forzar una comparación de cadenas o numérica.

© 2018 Altova GmbH



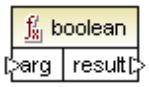
En el ejemplo de la imagen la primera constante es de tipo string y contiene la cadena "4". La segunda constante contiene la constante numérica 12. Para poder comparar estos dos valores de forma explícita los tipos deben coincidir.

Si añadimos una función **number** a la primera constante, la constante de cadena se convierte en el valor numérico 4. El resultado de la comparación será "true".

Si no se usara la función **number** (es decir, si 4 estuviera conectado al parámetro **a** directamente), se produciría una comparación de cadenas y su resultado sería "false".

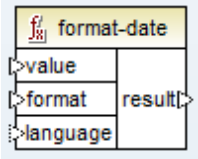
8.8.2.1 *boolean*

Convierte un valor numérico de entrada en un valor booleano (o un valor de cadena en un valor numérico) P. ej. `0` se convierte en `false` o `1` se convierte en `true`, valores que después se pueden usar en funciones lógicas (como **`equal`**, **`greater`**, etc.), en filtros o en condiciones **`if-Else`**.



8.8.2.2 *format-date*

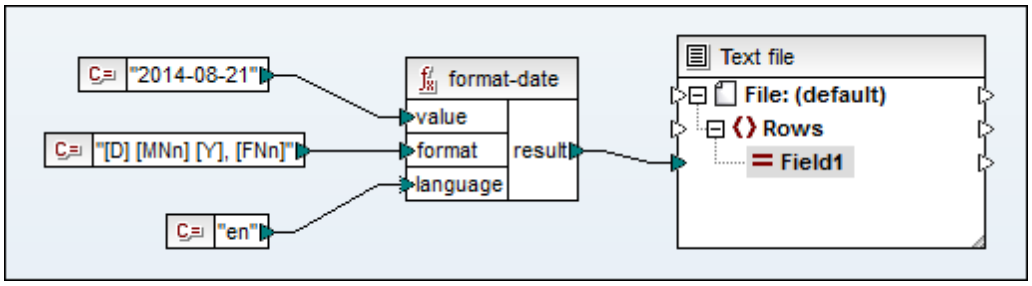
Convierte un valor de entrada `xs:date` en un valor de cadena y le aplica el formato indicado por las opciones especificadas.



Argumento	Descripción
value	Fecha a la que se debe aplicar formato.
format	Cadena de formato que identifica el formato que se debe aplicar a la fecha. Este argumento se usa igual que el argumento `format` de la función **`format-dateTime`**.
language	Argumento opcional que devuelve el nombre del mes y el día de la semana en el idioma seleccionado. Valores válidos:

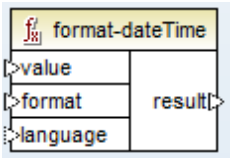
Argumento	Descripción
	en (predet.) inglés **es** español **de** alemán **ja** japonés

En la imagen siguiente, por ejemplo, el resultado es "21 August 2014, Thursday". Para traducir este valor al idioma español, defina el valor `es` para el argumento `language`.



8.8.2.3 *format-dateTime*

Convierte un valor de fecha y hora (`xs:dateTime`) en una cadena de texto. La representación de cadena de la fecha y hora sigue el formato que dicte el valor del argumento `format`.



Argumento	Descripción
value	El valor `dateTime` al que se debe aplicar formato.
format	Cadena de formato que identifica el formato que se debe aplicar a **value**.
language	Argumento opcional que devuelve el nombre del mes y el día de la semana en el idioma seleccionado. Valores válidos: **en (predet.)** inglés **es** español **de** alemán **ja** japonés

Nota: si el resultado de la función (**`result`**) se conecta a un nodo de un tipo que no sea `string`, el formato se puede perder porque el valor se convierte al tipo de destino. Esta conversión

 © 2018 Altova GmbH

automática se puede deshabilitar desactivando la casilla *Convertir valores en tipos de destino* en el cuadro de diálogo "Configuración" del componente de destino (*véase Cambiar la configuración de los componentes*).

El argumento **format** está compuesto por una cadena que lleva los llamados marcadores de variable entre corchetes. Los caracteres situados fuera de los corchetes son caracteres literales que se deben copiar en el resultado. Si necesita usar corchetes como caracteres literales en el resultado, escríbalos dos veces.

Cada marcador de variable está compuesto por (i) un especificador de componente que identifica qué componente de fecha u hora se debe mostrar, (ii) un modificador de formato opcional, (iii)otro modificador de presentación opcional y (iv) un modificador de ancho opcional precedido de una coma, si existe.

```
format := (literal | argument)*
argument := [component(format)?(presentation)?(width)?]
width := , min-width ("-" max-width)?
```

Estos son los componentes:

Especificador	Descripción	Presentación predeterminada
Y	año (valor absoluto)	cuatro dígitos (2010)
M	mes del año	1-12
D	día del mes	1-31
d	día del año	1-366
F	día de la semana	nombre del día (dependiendo del idioma)
W	semana del año	1-53
w	semana del mes	1-5
H	hora (24 horas)	0-23
h	hora (12 horas)	1-12
P	A.M. o P.M.	alfabética (dependiendo del idioma)
m	minutos de una hora	00-59
s	segundos de un minuto	00-59
f	segundos fraccionarios	numérica, con un decimal
Z	uso horario como diferencia horaria de UTC	+08:00
z	uso horario como diferencia horaria usando GMT	GMT+n

El modificador de formato puede ser uno de estos:

Carácter	Descripción	Ejemplo
1	formato decimal numérico sin ceros iniciales: 1, 2, 3, ...	1, 2, 3
01	formato decimal, con dos dígitos: 01, 02, 03, ...	01, 02, 03
N	nombre del componente, todo en mayúsculas	LUNES, MARTES [1)]
n	nombre del componente, todo en minúsculas	lunes, martes [1)]
Nn	nombre del componente, primera letra en mayúsculas	Lunes, Martes [1)]

Note: los modificadores N, n y Nn solamente son compatibles con estos componentes: M, d, D.

El modificador del ancho, si existe, se introduce con una coma y toma este formato:

```
, min-width ("-" max-width)?
```

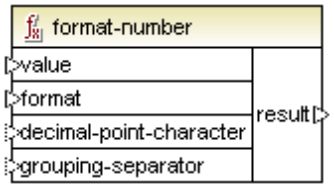
La tabla siguiente muestra algunos ejemplos de formato de valores `xs:dateTime`, obtenido con ayda de la función **`format-dateTime`**. La columna *Valor* de la tabla especifica el valor dado al argumento **`value`**. La columna *Formato* de la tabla especifica el valor del argumento **`format`**. La columna *Resultado* muestra el valor que devuelve la función.

Valor	Formato	Resultado
2003-11-03T00:00:00	`[D]/[M]/[Y]`	3/11/2003
2003-11-03T00:00:00	`[Y]-[M,2]-[D,2]`	2003-11-03
2003-11-03T00:00:00	`[Y]-[M,2]-[D,2] [H,2]:[m]:[s]`	2003-11-03 00:00:00
2010-06-02T08:02	`[Y] [MNn] [D01] [F,3-3] [d] [H]:[m]:[s].[f]`	2010 June 02 Wed 153 8:02:12.054
2010-06-02T08:02	`[Y] [MNn] [D01] [F,3-3] [d] [H]:[m]:[s].[f] [z]`	2010 June 02 Wed 153 8:02:12.054 GMT+02:00
2010-06-02T08:02	`[Y] [MNn] [D1] [F] [H]:[m]:[s].[f] [Z]`	2010 June 2 Wednesday 8:02:12.054 +02:00
2010-06-02T08:02	`[Y] [MNn] [D] [F,3-3] [H01]:[m]:[s]`	2010 June 2 Wed 08:02:12

© 2018 Altova GmbH

8.8.2.4 format-number

Convierte un número en una cadena. La función está disponible para XSLT 1.0, XSLT 2.0, Java, C#, C++ y el motor de ejecución integrado.



Argumento	Descripción
value	Argumento obligatorio. Suministra el número al que se debe aplicar formato.
format	Argumento obligatorio. Suministra una cadena de formato que indica de qué forma se debe dar formato al número. Este argumento se usa igual que el argumento `format` de la función **`format-dateTime`**.
decimal-point-format	Argumento opcional. Suministra el carácter que se debe usar como carácter de punto decimal. El valor predeterminado es el carácter de punto (`.`).
grouping-separator	Argumento opcional. Suministra el carácter que se debe usar para separa grupos de números. El valor predeterminado es el carácter de coma (`,`).

Nota: si el resultado de la función (es decir, **`result`**) se conecta a un nodo de un tipo que no sea string, el formato se puede perder porque el valor se convierte al tipo de destino. Esta conversión automática se puede deshabilitar desactivando la casilla *Convertir valores en tipos de destino* en el cuadro de diálogo "Configuración" del componente de destino.

Format:

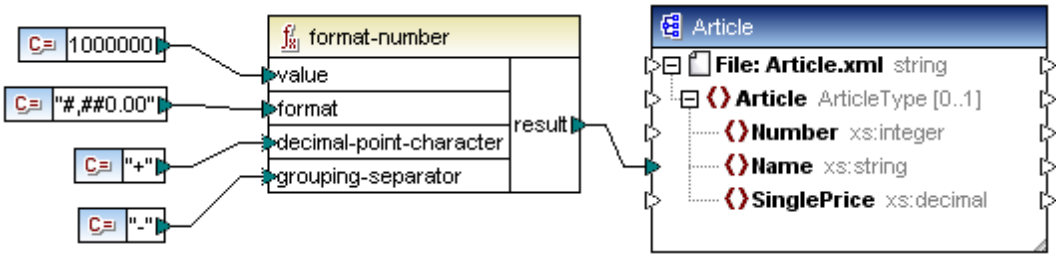
```
format := subformat (;subformat)?
  subformat := (prefix)? integer (.fraction)? (suffix)?
  prefix := any characters except special characters
  suffix := any characters except special characters
  integer := (#)* (0)* ( allowing ',' to appear)
  fraction := (0)* (#)* (allowing ',' to appear)
```

El primer *subformat* se utiliza para dar formato a los números positivos y el segundo para los números negativos. Si solamente se da un *subformat*, se utiliza el mismo para los números negativos, pero con un signo - antes del *prefix*.

Carácter especial	Predeterminado	Descripción
dígito cero	0	en este punto del resultado siempre

Carácter especial	Predeterminado	Descripción
		aparecerá un dígito
dígito	#	en este punto del resultado siempre aparecerá un dígito, excepto si es un cero inicial o final no significativo
punto decimal	.	separa el entero y la parte de fracción del número
separador de grupos	,	separa grupos de dígitos
signo de porcentaje	%	multiplica el número por 100 y lo muestra como porcentaje
por mil	‰	multiplica el número por 1000 y lo muestra como por mil

Los caracteres utilizados por el carácter de punto decimal y de separador de grupos siempre son "." y "," respectivamente. No obstante, se pueden cambiar en el resultado formateado asignando constantes a estos nodos.



El resultado de la función **`format-number`** anterior es:

- El carácter de punto decimal se cambió por +.
- El separador de grupos se cambió por -.

```
<Article xsi:noNamespaceSchemaLocation='
  <Name>1-000-000+00</Name>
</Article>
```

Redondeo

El método de redondeo utilizado para esta función es al alza (p. ej. redondea al alza si la fracción es mayor o igual a 0.5 y redondea a la baja si la fracción es menor que 0.5). Este método de redondeo solamente afecta al código generado y si la opción seleccionada es el motor de ejecución integrado.

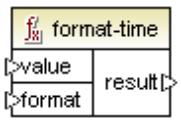
El XSLT 1.0 el modo de redondeo es indeterminado. En XSLT 2.0 el modo de redondeo es *round-half-to-even*, es decir se redondea al número par más próximo.

© 2018 Altova GmbH

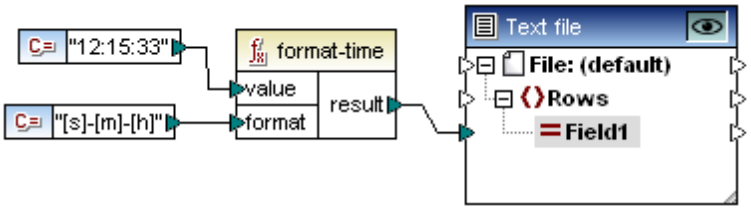
Número	Cadena de formato	Resultado
1234.5	#,##0.00	1,234.50
123.456	#,##0.00	123.46
1000000	#,##0.00	1,000,000.00
-59	#,##0.00	-59.00
1234	###0.0###	1234.0
1234.5	###0.0###	1234.5
.00025	###0.0###	0.0003
.00035	###0.0###	0.0004
0.25	#00%	25%
0.736	#00%	74%
1	#00%	100%
-42	#00%	-4200%
-3.12	#.00;(#.00)	(3.12)
-3.12	#.00;#.00CR	3.12CR

8.8.2.5 *format-time*

Convierte un valor de entrada `xs:time` en un valor de cadena. El argumento **format** se usa igual que el argumento `format` de la función **`format-dateTime`**.



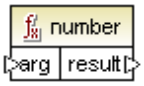
Ejemplo:



Resultado: 33-15-12

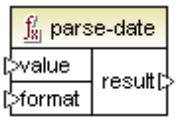
8.8.2.6 *number*

Convierte una cadena de entrada en un número. También convierte una entrada booleana en un número.



8.8.2.7 *parse-date*

Disponible en Java, C#, C++ y el motor de ejecución integrado.

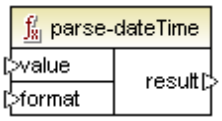


Convierte una cadena en una fecha y omite el componente de hora. Esta función usa como base la función parse-dateTime, pero omite el componente de hora. El resultado es de tipo `xs:date`.

Para ver un ejemplo de uso consulte el apartado Ejemplo: asignar datos de un canal RSS.

8.8.2.8 *parse-dateTime*

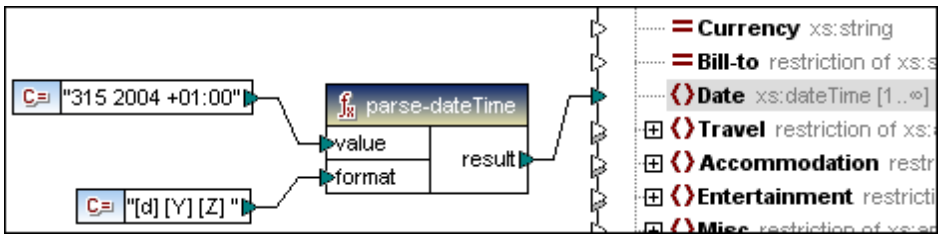
Disponible para Java, C#, C++ y el motor de ejecución integrado.



Convierte un valor de fecha/hora expresado en forma de cadena en un valor de tipo `dateTime`. Esta función toma estos dos argumentos:

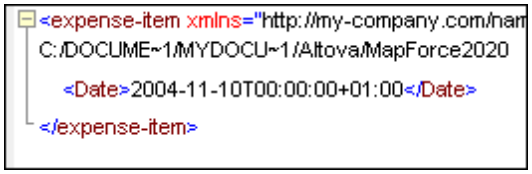
Argumento	Descripción
`value`	El valor de cadena que se debe convertir.
`format`	La máscara de formato que se debe aplicar al argumento **`value`**.

Por ejemplo, en la asignación que aparece a continuación, el valor de cadena **`315 2004 +01:00`** especifica el día nº 315 del año 2004 en la zona horaria GMT+01:00. Este valor se convierte en su `dateTime` equivalente aplicando la máscara de formato **`[d] [Y] [Z]`**.



 © 2018 Altova GmbH

El resultado sería:



Una máscara de formato puede estar compuesta por estos componentes:

Componente	Descripción	Presentación predeterminada
Y	año (valor absoluto)	cuatro dígitos (2010)
M	mes del año	1-12
D	día del mes	1-31
d	día del año	1-366
H	hora (24 horas)	0-23
h	hora (12 horas)	1-12
P	A.M. o P.M.	alfabética (dependiendo del idioma)
m	minutos de una hora	00-59
s	segundos de un minuto	00-59
f	segundos fraccionarios	numérica, con un decimal
Z	uso horario como diferencia horaria de UTC	+08:00
z	uso horario como diferencia horaria usando GMT	GMT+n

Algunos de estos componentes toman modificadores (p. ej. se pueden usar para interpretar una fecha como fecha de un solo dígito o de dos dígitos):

Modificador	Descripción	Ejemplo
1	formato decimal numérico sin ceros iniciales: 1, 2, 3, ...	1, 2, 3
01	formato decimal, con dos dígitos: 01, 02, 03, ...	01, 02, 03
N	nombre del componente, todo en mayúsculas	FEBRERO, MARZO
n	nombre del componente, todo en minúsculas	febrero, marzo

Modificador	Descripción	Ejemplo
Nn	nombre del componente, primera letra en mayúsculas	Febrero, Marzo

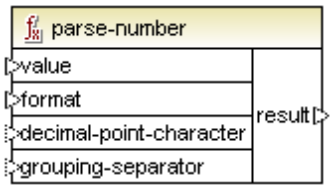
Nota: los modificadores *N*, *n* y *Nn* solo admiten el componente *M* (mes).

A continuación puede ver algunos ejemplos más:

Valor	Formato	Resultado
`21-03-2002 16:21:12.492 GMT +02:00`	`[D]-[M]-[Y] [H]:[m]:[s].[f] [z]`	`2002-03-21T16:21:12.492 +02:00`
`315 2004 +01:00`	`[d] [Y] [Z]`	`2004-11-10T00:00:00 +01:00`
`1.December.10 03:2:39 p.m. +01:00`	`[D].[MNn].[Y,2-2] [h]:[m]:[s] [P] [Z]`	`2010-12-01T15:02:39 +01:00`
`20110620`	`[Y,4-4][M,2-2][D,2-2]`	`2011-06-20T00:00:00`

8.8.2.9 parse-number

Disponible para Java, C#, C++ y el motor de ejecución integrado.



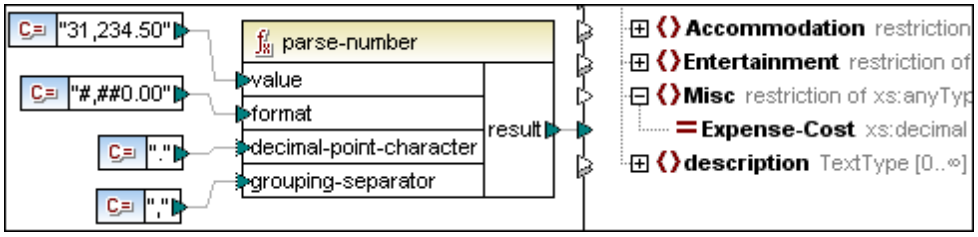
Convierte una cadena de entrada en un número decimal.

Argumento	Descripción
value	La cadena que se debe analizar/convertir en número
format	Una cadena de formato que identifica el formato actual del número (opcional). El valor predeterminado es #,##0.#
decimal-point-character	El carácter que se debe utilizar como carácter de punto decimal. El carácter predeterminado es "." (opcional)
grouping-separator	El separador/delimitador utilizado para separar grupos de números. El carácter predeterminado es "," (opcional)

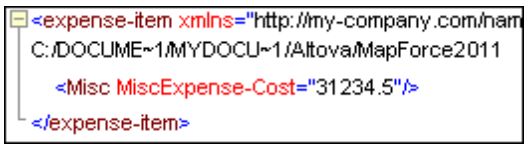
La cadena **`format`** utilizada en la función **`parse-number`** es la misma que se utiliza en la función format-number.

© 2018 Altova GmbH

Ejemplo en MapForce:

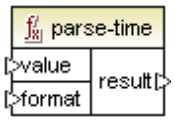


Resultado:



8.8.2.10 parse-time

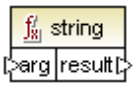
Disponible en Java, C#, C++ y el motor de ejecución integrado.



Convierte una cadena en una hora, pero ignora el componente fecha. Esta función se basa en la función parse-dateTime pero ignora el componente fecha. El resultado es de tipo `xs:time`.

8.8.2.11 string

Convierte un valor de entrada en una cadena. La función también se puede usar para recuperar el contenido de texto de un nodo.



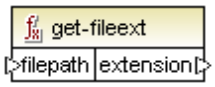
Si el nodo de entrada es un tipo complejo XML, todos los descendientes también se generan como una sola cadena.

8.8.3 core | file path functions (ruta de archivos)

Las funciones de ruta de acceso de la biblioteca **core | file path** sirven para acceder a datos de una ruta de acceso (como carpetas, nombres de archivos y extensiones) y manipularlos para después procesarlos en la asignación. Estas funciones están disponibles para todos los lenguajes compatibles con MapForce.

8.8.3.1 get-fileext

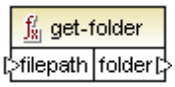
Devuelve la extensión de la ruta de archivo, incluido el carácter "."



P. ej. `c:\data\Sample.mfd` devuelve `".mfd"`

8.8.3.2 get-folder

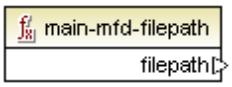
Devuelve el nombre de la carpeta de la ruta de archivo, incluida la barra diagonal final o la barra diagonal inversa final.



P. ej. `c:/data/Sample.mfd` devuelve `c:/data/`

8.8.3.3 main-mfd-filepath

Devuelve la ruta completa del archivo `mfd` que contiene la asignación principal. Si el archivo `mfd` está sin guardar, la función devuelve una cadena vacía.



8.8.3.4 mfd-filepath

Si se le llama desde la asignación principal, la función devuelve el mismo resultado que la función **`main-mfd-filepath`**, es decir, la ruta completa del archivo `mfd` que contiene la asignación principal. Si el archivo `mfd` está sin guardar, la función devuelve una cadena vacía.

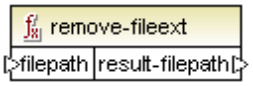


Si se le llama desde dentro de una **función definida por el usuario** que es **importada** por un archivo `mfd`, la función **`mfd-filepath`** devuelve la ruta completa del archivo `mfd` importado que contiene la **definición** de la función definida por el usuario.

© 2018 Altova GmbH

8.8.3.5 remove-fileext

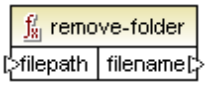
Borra la extensión de la ruta de archivo, incluido el carácter "."



P. ej. `c:/data/Sample.mfd` devuelve `c:/data/Sample`.

8.8.3.6 remove-folder

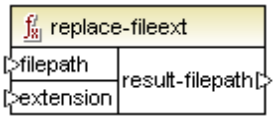
Borra el directorio de la ruta de acceso, incluida la barra diagonal final o la barra diagonal inversa final.



P. ej. `c:/data/Sample.mfd` devuelve `Sample.mfd`.

8.8.3.7 replace-fileext

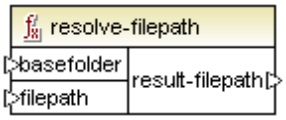
Reemplaza la extensión de la ruta de acceso dada por el parámetro **`filepath`** con la extensión dada por el parámetro **`extension`**.



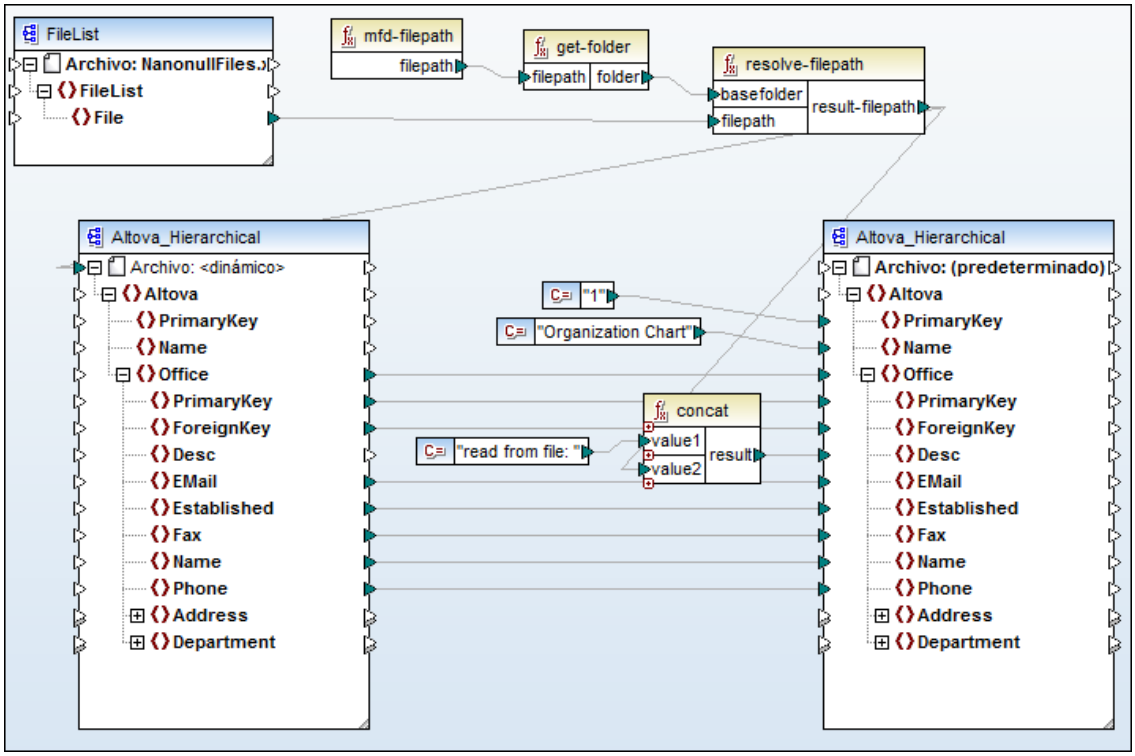
P. ej. si el parámetro **`filepath`** es la ruta de acceso `c:/data/Sample.mfd` y `.mfp` es la extensión del parámetro **`extension`**, la función devuelve `c:/data/Sample.mfp`.

8.8.3.8 resolve-filepath

Convierte una ruta de acceso relativa en una carpeta base relativa o absoluta. La función admite "." (directorio actual) y ".." (directorio primario).



Para ver un ejemplo abra el archivo de asignación **MergeMultipleFiles_List.mfd** , disponible en la carpeta ...\MapForceExamples.

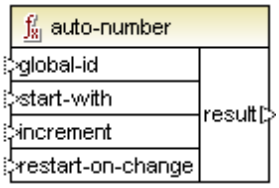


8.8.4 core | generator functions (generador)

La biblioteca de funciones **core / generator** incluye funciones que generan valores.

8.8.4.1 auto-number

La función **auto-number** genera enteros en los nodos de destino de un componente, en base a los parámetros definidos. El resultado es un valor que empieza en el parámetro **start_with** y se incrementa con el valor de **increment**. Los valores predeterminados son **start-with=1** e **increase=1**. Ambos parámetros pueden ser negativos.



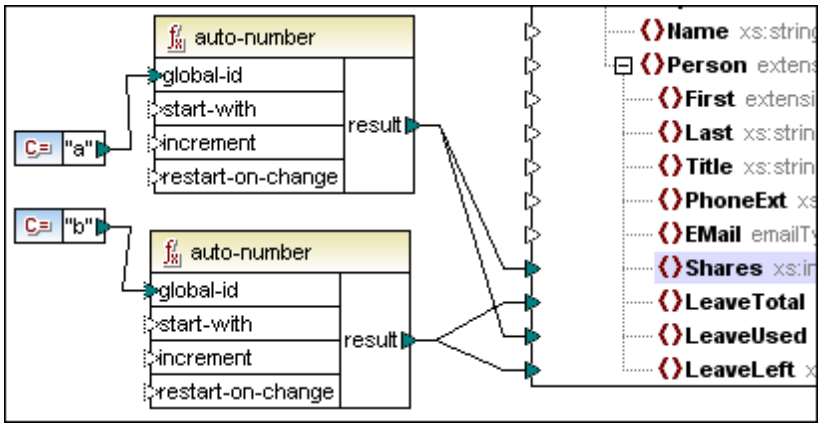
Compruebe que el conector del parámetro **result** (de la función **auto-number**) está conectado directamente al nodo de destino. Recuerde que el código generado no llama a las funciones en ningún orden en particular. MapForce puede copiar en caché los resultados calculados para volver a utilizaros o evaluar las expresiones en cualquier orden. Por tanto, recomendamos encarecidamente que utilice con cuidado la función **auto-number**.

Parámetro global-id

 © 2018 Altova GmbH

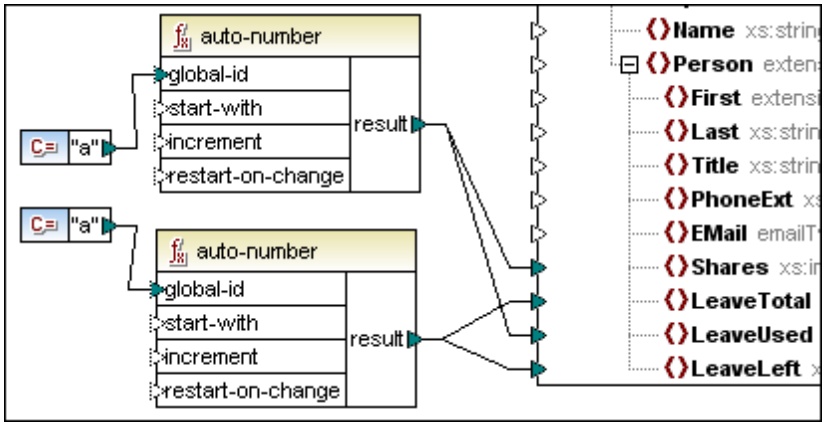
Sirve para sincronizar la secuencia numérica resultante de dos funciones **`auto-number`** distintas conectadas al mismo componente de destino. Si las dos funciones **`auto-number`** no tienen el mismo **`global-id`**, cada función incrementa los elementos de destino por separado. En el ejemplo siguiente, las funciones tienen parámetros **`global-id`** diferentes (**`a`** y **`b`**).

El resultado de la asignación es `1,1,2,2`. La primera función (con **`global-id a`**) aporta el primer 1 y la segunda (con **`global-id b`**) aporta el segundo 1.



Si ambas funciones tienen **el mismo `global-id`**, digamos **`a`**, entonces ambas funciones conocen el estado actual (o valor real) de la otra y ambos números están sincronizados (*imagen siguiente*).

En este caso el resultado de la asignación es `1, 2, 3, 4`. La primera función aporta el primer 1 y la segunda función ahora aporta un 2.



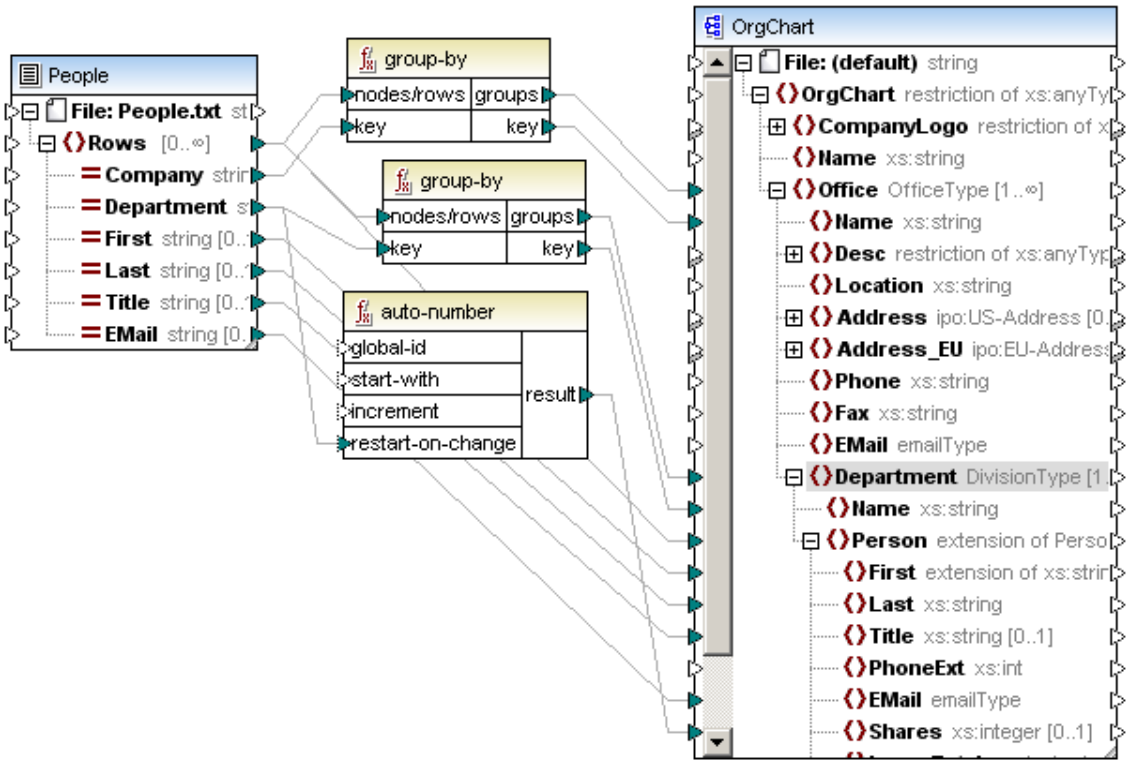
Parámetro start-with
El valor inicial utilizado para comenzar la secuencia de numeración automática. Valor predeterminado = 1.

Parámetro increment
El incremento que se debe aplicar a la secuencia de numeración automática. Valor predeterminado = 1.

Parámetro restart-on-change
Restablece el contador de numeración automática al parámetro **`start-with`** cuando cambia el

contenido de los elementos conectados.

En el ejemplo siguiente, **`start-with`** e **`increment`** tienen el valor predeterminado 1. En cuanto cambia el **contenido** de **`Department`** (es decir, cambia el nombre del departamento), el contador se restablece y la secuencia de cada departamento nuevo empieza por 1.



8.8.5 core | logical functions (lógica)

Las funciones lógicas de la biblioteca **core | logical** se usan por lo general para comparar datos de entrada y su resultado son los valores booleanos `true` o `false`. Estas funciones se suelen utilizar para probar datos antes de pasarlos, por medio de un filtro, a un subconjunto del componente de destino.

Parámetros de entrada = **`a | b`** o **`value1 | value2`**
Parámetro de salida = **`result`**

El resultado de la evaluación de dos nodos de entrada depende de los valores de entrada así como de los tipos de datos utilizados para la comparación.

Por ejemplo, la comparación "less than" de los valores enteros 4 y 12 da como resultado el valor booleano "true" porque 4 es menor que 12. Si las dos cadenas de entrada contienen "4" y "12", entonces el análisis léxico da como resultado el valor de salida "false" porque "4" es alfabéticamente mayor que el primer carácter "1" del segundo operando (12).

Si todos los tipos de datos de entrada son del mismo tipo (p. ej. todos los nodos de entrada son de tipo numérico o son cadenas), entonces la comparación se hace para el tipo común.

© 2018 Altova GmbH

Si los nodos de entrada son de tipos distintos (p. ej. un entero y una cadena o una cadena y una fecha), entonces el tipo de datos utilizado para la comparación es **el tipo de datos de entrada más general y menos limitado** de los dos.

Antes de iniciarse la comparación de los dos valores, todos los valores de entrada se convierten en el tipo de datos común. Por ejemplo, el tipo de datos "string" es menos limitado que "integer". La comparación del valor entero 4 con la cadena "12" convierte el valor entero 4 en la cadena "4", que después se compara con la cadena "12".

Nota: las funciones lógicas no se pueden usar para comprobar la existencia de valores null. Si se da un valor null como argumento de una función lógica, la función devuelve un valor null. Para más información consulte el apartado Valores Nil y Nillable.

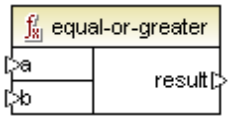
8.8.5.1 *equal*

El resultado es **`true`** si **`a=b`**. De lo contrario, el resultado es **`false`**.



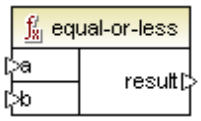
8.8.5.2 *equal-or-greater*

El resultado es **`true`** si **`a`** es igual/mayor que **`b`**. De lo contrario, el resultado es **`false`**.



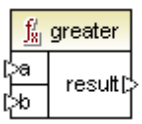
8.8.5.3 *equal-or-less*

El resultado es **`true`** si **`a`** es igual o menor que **`b`**. De lo contrario, el resultado es **`false`**.



8.8.5.4 *greater*

El resultado es **`true`** si **`a`** es mayor que **`b`**. De lo contrario, el resultado es **`false`**.



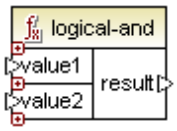
8.8.5.5 less

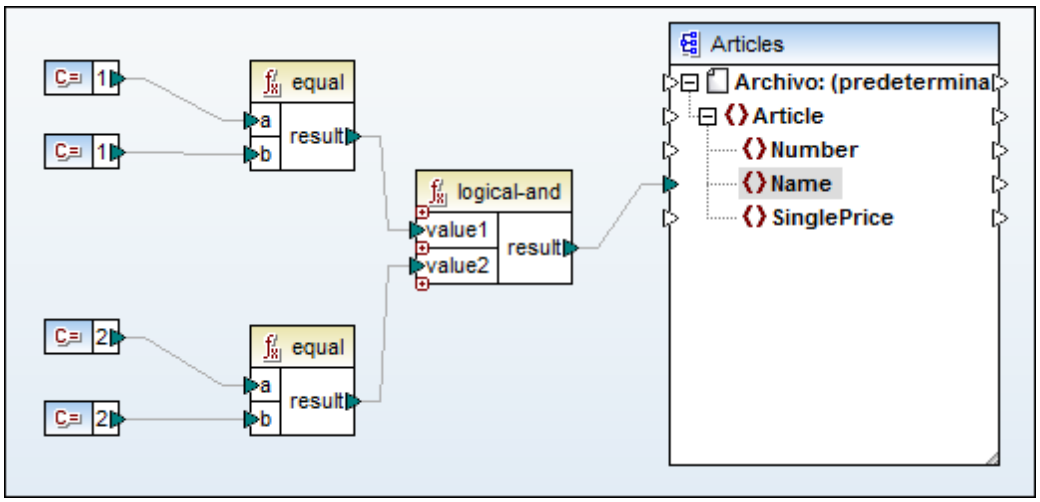
El resultado es **true** si **a** es menor que **b**. De lo contrario, el resultado es **false**.



8.8.5.6 logical-and

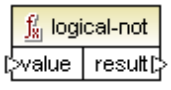
Si tanto **value1** como **value2** son **true**, el resultado es **true**. Si son diferentes, el resultado es **false**.





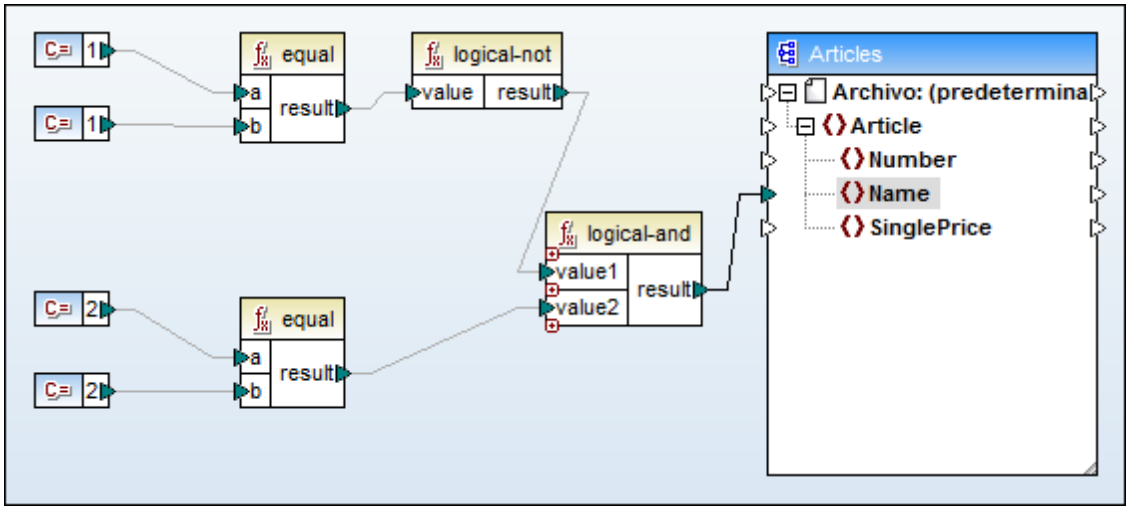
8.8.5.7 logical-not

Invierte o le da la vuelta al estado lógico/resultado. Si el valor de entrada es **true**, el resultado de la función es **false**. Si el valor de entrada es **false**, el resultado de la función es **true**.



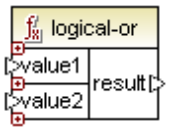
La función **logical-not** que aparece a continuación invierte el resultado de la función **equal**. La función **logical-and** ahora solamente devuelve el valor **true** si los valores booleanos de **value1** y **value2** son diferentes (es decir, **true-false** o **false-true**).

 © 2018 Altova GmbH



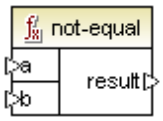
8.8.5.8 *logical-or*

Exige que ambos valores de entrada sean booleanos. Si uno de los dos valores de entrada (ya sea **value1** o **value2**) de la función **logical-or** es **true**, el resultado de la función es **true**. Si ambos son **false**, el resultado es **false**.



8.8.5.9 *not-equal*

El resultado es **true** si **a** no es igual a **b**.

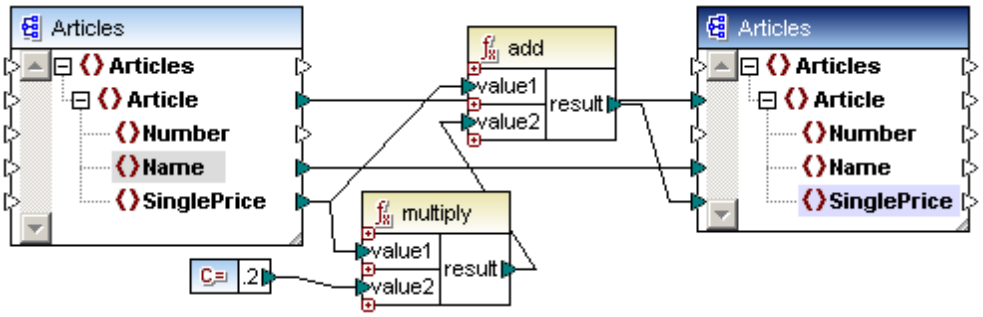


8.8.6 core | math functions (matemáticas)

Las funciones matemáticas de la biblioteca **core | math** sirven para realizar operaciones matemáticas básicas con los datos. Tenga en cuenta que no se pueden usar para cálculos con datos `duration` ni `datetime`.

Parámetros de entrada = **value1 | value2**
Parámetro de salida = **result**

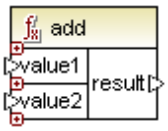
MapForce convierte automáticamente los valores de entrada en decimales para poder procesarlos.



En la asignación de esta imagen, por ejemplo, se añade un 20% de impuestos sobre las ventas en los artículos asignados al componente de destino.

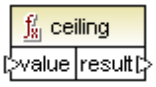
8.8.6.1 add

El resultado es el valor decimal resultante de sumar **value1** a **value2**.



8.8.6.2 ceiling

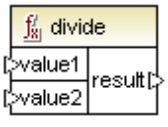
El resultado es el entero más pequeño que es mayor o igual que **value**, es decir, el valor entero mayor más próximo al valor decimal de entrada **value**.



P. ej. si el resultado de una función de división es `11.2` y a este resultado le aplicamos la función **ceiling**, el resultado se convierte en `12` (es decir, el número entero mayor más cercano).

8.8.6.3 divide

El resultado es el valor decimal resultante de dividir **value1** por **value2**. La precisión del resultado depende del lenguaje de destino. Utilice la función round-precision para definir la precisión del resultado.



© 2018 Altova GmbH

8.8.6.4 floor

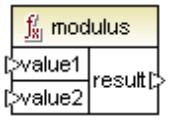
El resultado es el entero mayor que sea menor o igual que **`value`**, es decir, el valor entero menor más próximo al parámetro decimal **`value`** de entrada.



P. ej. si el resultado de una función de división es `11.2` y a este resultado le aplicamos la función **`floor`**, el resultado se convierte en `11` (es decir, el número entero menor más cercano).

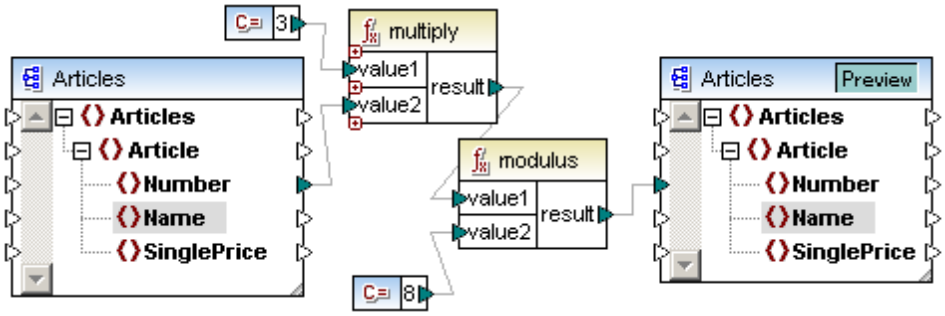
8.8.6.5 modulus

El resultado es el entero restante después de dividir **`value1`** por **`value2`**.



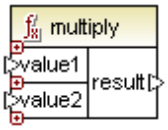
En la asignación que aparece a continuación, los números se multiplicaron por `3` y se pasaron al parámetro **`value1`** de la función **`modulus`**. Los valores de entrada ahora son 3, 6, 9 y 12.

Cuando se aplica/utiliza la función **`modulus`** con **`value2`** = 8, los enteros restantes son 3, 6, 1 y 4.



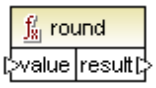
8.8.6.6 multiply

El resultado es el valor decimal resultante de multiplicar **`value1`** por **`value2`**.



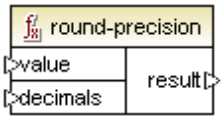
8.8.6.7 round

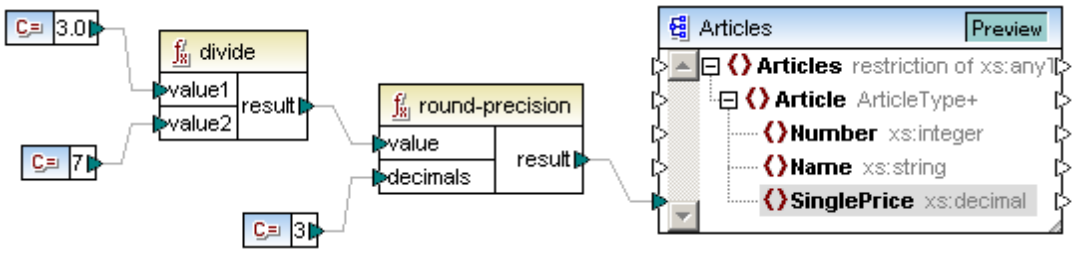
Devuelve el valor redondeado al entero más cercano. Cuando el valor esté justo entre dos enteros, se utilizará el algoritmo *Desempate la mitad alejándose del cero.* Por ejemplo, el valor `10.5` se redondearía hasta `11` y el valor `-10.5` se redondearía hasta `-10`.



8.8.6.8 round-precision

El resultado es el valor decimal del número redondeado a las cifras decimales definidas por el parámetro **`decimals`**.

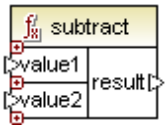




Por ejemplo, en la asignación anterior, el resultado es `0.429`. Si quiere que este resultado aparezca correctamente en el archivo XML, debe estar asignado a un elemento de tipo `xs:decimal`.

8.8.6.9 subtract

El resultado es el valor decimal resultante de restar **`value2`** a **`value1`**.



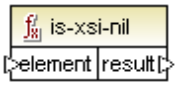
8.8.7 core | node functions (nodo)

Las funciones de la biblioteca **core | node** sirven para obtener acceso a nodos o procesarlos.

© 2018 Altova GmbH

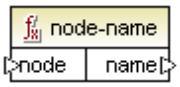
8.8.7.1 *is-xsi-nil*

Devuelve **`true`** (<OrderID>true</OrderID>) si el nodo elemento del componente de origen tiene un atributo `xsi:nil` con el valor `true`.

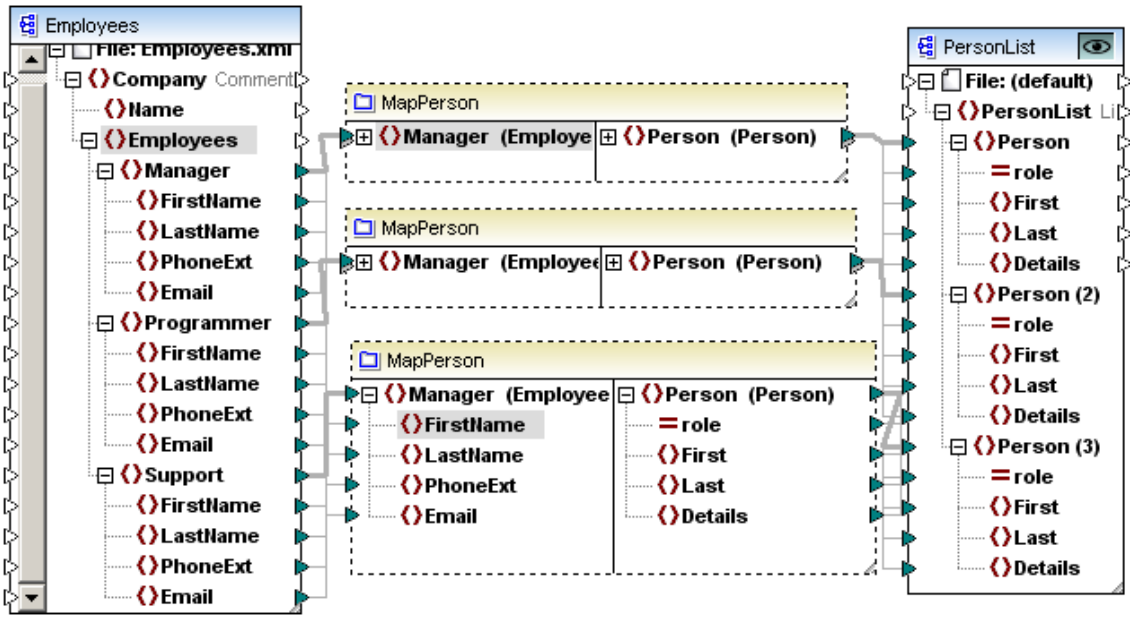


8.8.7.2 *node-name*

Devuelve el QName del nodo conectado, a no ser que sea un nodo XML `text()`. Si es así, devuelve un QName vacío. Esta función solamente funciona con los nodos que tienen nombre. Si XSLT es el lenguaje de destino (que llama a la función fn:node-name), la función devuelve una secuencia vacía para los nodos que no tienen nombre.

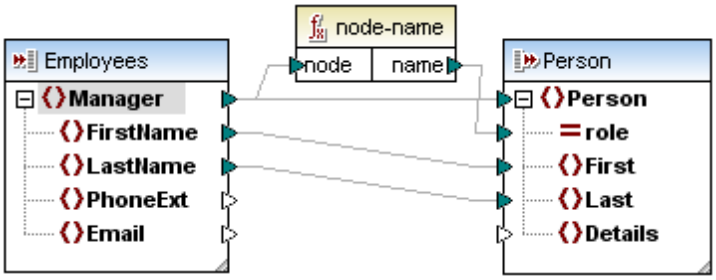


- No puede obtener un nombre de campos/tablas de BD.
- Esta función no es compatible con XBRL ni Excel.
- No se puede obtener un nombre del nodo de entrada **`Archivo`**.
- Los nodos de servicio web se comportan como nodos XML, excepto que:
  - node-name no es compatible con `part`.
  - node-name no es compatible con el nodo raíz (de entrada o salida).



La función definida por el usuario **`MapPerson`** usa la función **`node-name`** para recuperar el nombre del nodo de entrada y colocarlo en el atributo `role`. El nodo raíz del esquema `Employees.xsd`, en

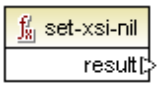
la función definida por el usuario, es **Manager**.



Manager recibe sus datos de **fuera** de la función definida por el usuario y su contenido puede ser `Manager`, `Programmer` o `Support`. Estos son los datos que se pasan al atributo `role` del componente de destino **PersonList**.

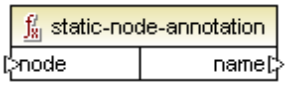
8.8.7.3 *set-xsi-nil*

Establece el nodo de destino en xsi:nil.



8.8.7.4 *static-node-annotation*

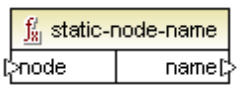
Devuelve una cadena con la anotación del nodo conectado. El parámetro de entrada **node** debe ser (i) un nodo del componente de origen o (ii) una función inline directamente conectada a un parámetro, directamente conectado a su vez a un nodo de la asignación que llama a la función.



La conexión debe ser directa. No puede pasar por un filtro ni por una función definida por el usuario no inline. Se trata de una pseudo función, que en tiempo de generación se reemplaza con un texto adquirido del nodo conectado y, por tanto, está disponible en todos los lenguajes.

8.8.7.5 *static-node-name*

Devuelve una cadena que contiene el nombre del nodo conectado. El parámetro de entrada **node** debe ser (i) un nodo del componente de origen o (ii) una función inline directamente conectada a un parámetro, directamente conectado a su vez a un nodo de la asignación que llama a la función.



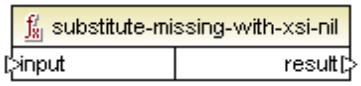
La conexión debe ser directa. No puede pasar por un filtro ni por una función definida por el

 © 2018 Altova GmbH

usuario no inline. Se trata de una pseudo función, que en tiempo de generación se reemplaza con un texto adquirido del nodo conectado y, por tanto, está disponible en todos los lenguajes.

8.8.7.6 substitute-missing-with-xsi-nil

Para los nodos de contenido simple esta función reemplaza los valores que faltan o los valores nulos del componente de origen con el atributo `xsi:nil` en el nodo de destino.

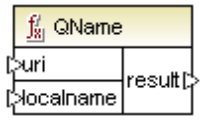


8.8.8 core | QName functions (QName)

Las funciones QName permiten manipular los nombres completos (QName) en documentos XML.

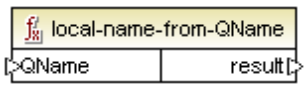
8.8.8.1 QName

Construye un QName a partir del URI de espacio de nombres y una parte local. Use esta función para crear un QName en un componente de destino. Los parámetros **uri** y **localname** pueden suministrarse por medio de una función de constante.

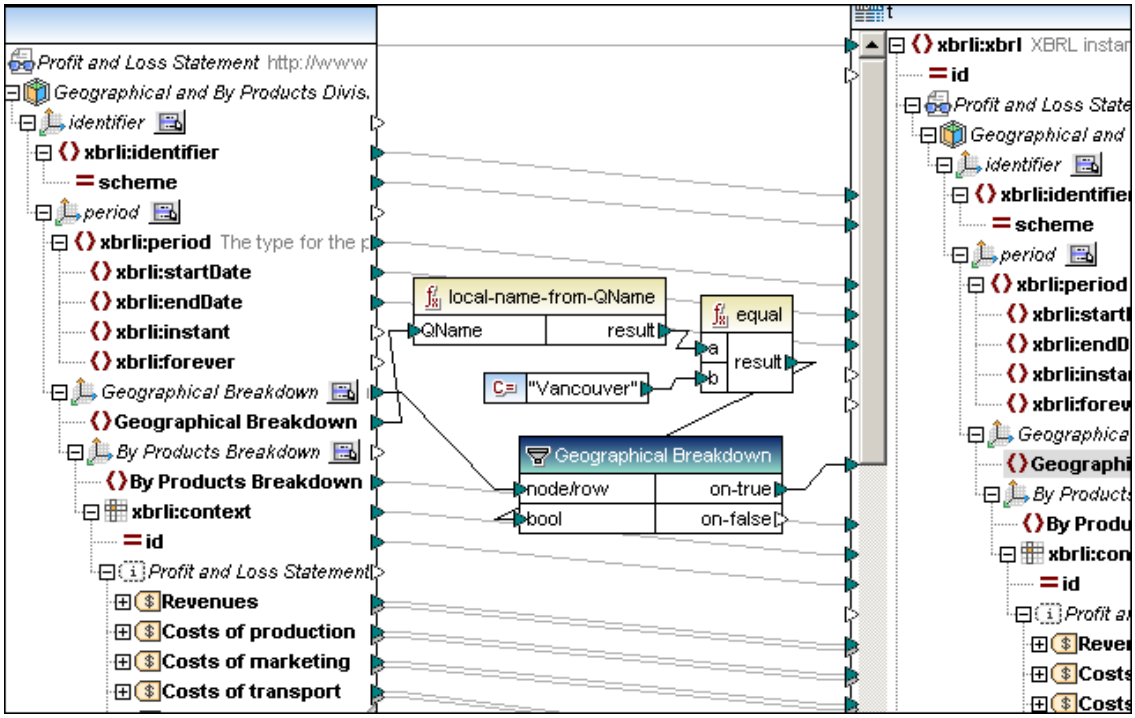


8.8.8.2 local-name-from-QName

Devuelve la parte de nombre local del QName.



Esta función puede ser muy práctica a la hora de trabajar con documentos de instancia XBRL que contienen hipercubos.



En la asignación de la imagen se filtran los hechos cuyo **nombre local** del **contenido** del miembro explícito (d-g:Vancouver) es igual a "Vancouver". Observe que el contenido del miembro es de hecho un QName.

```
<xbrli:context id="Year2007_Vancouver_BeveragesTotal">
  <xbrli:entity>
    <xbrli:identifier scheme="http://www.stockexchange/ticker">ACME</xbrli:identifier>
    <xbrli:segment>
      <xbrldi:explicitMember dimension="d-g:GeographicalBreakdown">d-g:Vancouver</xbrl
      <xbrldi:explicitMember dimension="d-b:ByProductsBreakdown">d-b:BeveragesTotal</x
    </xbrli:segment>
```

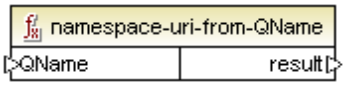
En la asignación se filtran todos los hechos que pertenecen a la dimensión GeographicalBreakdown y se envían al componente de destino.

```
<context id="Year2007_Vancouver_BeveragesTotal">
  <entity>
    <identifier scheme="http://www.stockexchange/ticker">ACME</identifier>
    <segment>
      <explicitMember dimension="d-b:ByProductsBreakdown" xmlns="http://xbrl.org/2006/xbrldi">d-b:Beverages
      <explicitMember dimension="d-g:GeographicalBreakdown" xmlns="http://xbrl.org/2006/xbrldi" />
    </segment>
  </entity>
  <period>
    <startDate>2007-01-01</startDate>
    <endDate>2007-12-31</endDate>
  </period>
</context>
<p:Revenues contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">4</p:Revenues>
<p:CostsOfProduction contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">1</p:Cos
<p:CostsOfMarketing contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">1</p:Cost
<p:ProfitLoss contextRef="Year2007_Vancouver_BeveragesTotal" unitRef="USD" decimals="0">2</p:ProfitLoss>
```


© 2018 Altova GmbH

8.8.8.3 namespace-uri-from-QName

Devuelve la parte URI de espacio de nombres del QName.



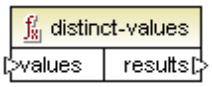
8.8.9 core | sequence functions (secuencia)

Las funciones de secuencia de la biblioteca **core | sequence** sirven para procesar secuencias de entrada y agrupar su contenido. El valor/contenido del parámetro de entrada **key**, asignado a nodos/filas, se utiliza para agrupar la secuencia.

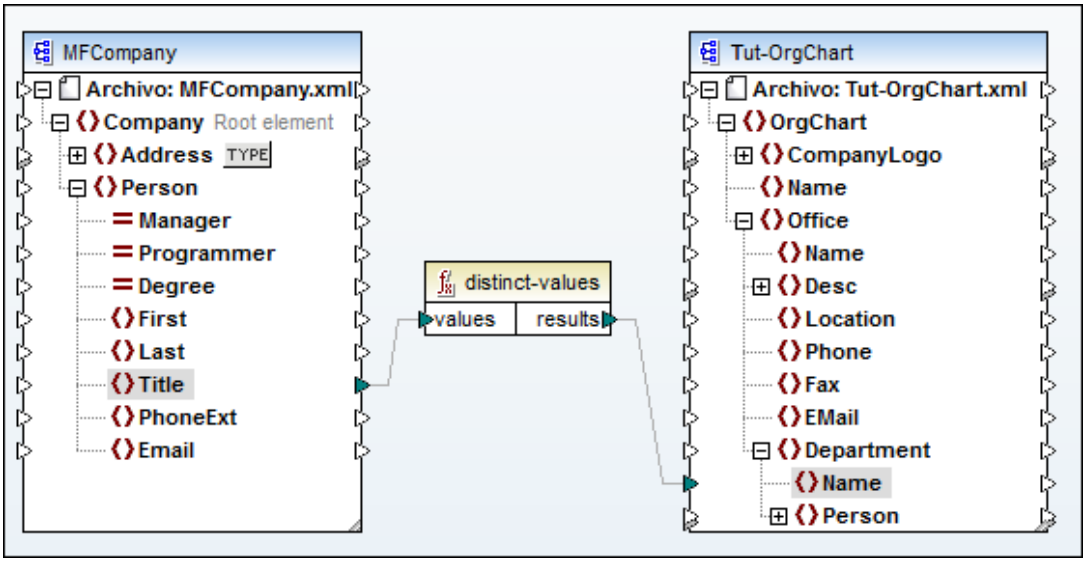
- El parámetro de entrada **key** tiene un tipo de datos arbitrario que se puede convertir en `string` en las funciones de secuencia **group-adjacent** y **group-by**.
- El parámetro de entrada **bool** tiene el tipo de datos **Boolean** en las funciones de secuencia **group-starting-with** y **group-ending-with**.
- El parámetro de salida **key** es la clave del grupo actual.

8.8.9.1 distinct-values

Permite quitar los valores duplicados de una secuencia y asignar los elementos únicos al componente de destino.



En el ejemplo que aparece a continuación se analiza el contenido de los elementos del componente de origen "Title" y cada título que sea único se asigna al elemento Department / Name del componente de destino.

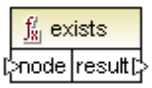


Observe que la secuencia de los elementos Title del componente de origen se conserva cuando se asignan al componente de destino.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <OrgChart xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart C:/DOCU
3    <Office>
4      <Department>
5        <Name>Office Manager</Name>
6        <Name>Accounts Receivable</Name>
7        <Name>Accounting Manager</Name>
8        <Name>Marketing Manager Europe</Name>
9        <Name>Art Director</Name>
10       <Name>Program Manager</Name>
11       <Name>Software Engineer</Name>
12       <Name>Technical Writer</Name>
13       <Name>IT Manager</Name>
14       <Name>Web Developer</Name>
15       <Name>Support Engineer</Name>
16       <Name>PR &amp;amp; Marketing Manager US</Name>
17     </Department>
18   </Office>
19 </OrgChart>
```

8.8.9.2 exists

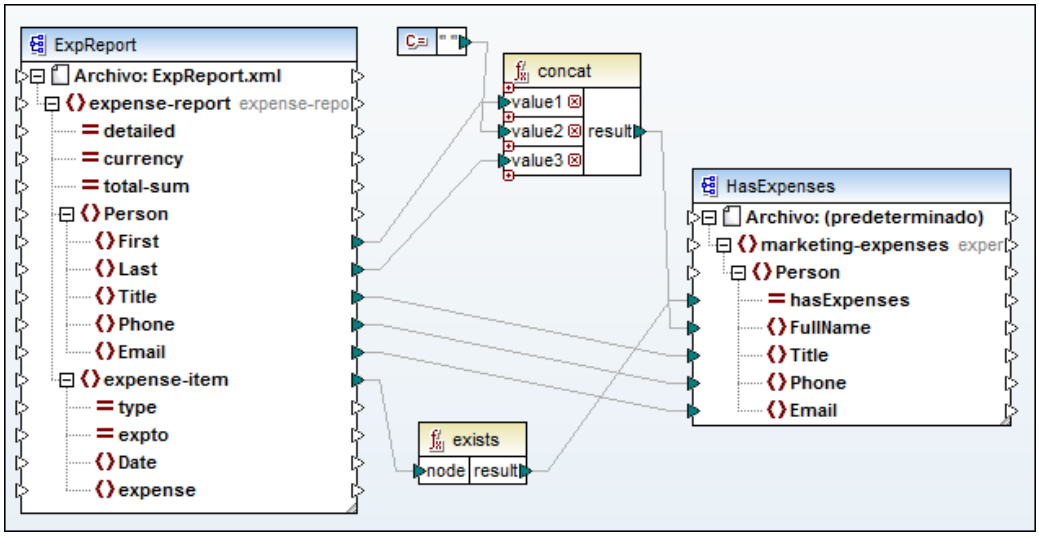
Devuelve `true` si el nodo existe. De lo contrario devuelve `false`.



El diseño de asignación de datos **HasMarketingExpenses.mfd** de la carpeta ...\MapForceExamples contiene el ejemplo que aparece en la imagen siguiente.

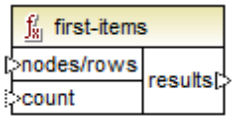
Si existe un elemento expense-item en el XML de origen, entonces el atributo "hasExpenses" recibe el valor `true` en el esquema o archivo XML de destino.

© 2018 Altova GmbH



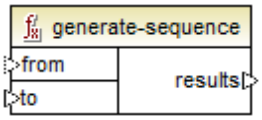
8.8.9.3 *first-items*

Devuelve los X primeros elementos de la secuencia de entrada, siendo X el número que suministra el parámetro **count**. Por ejemplo, si se asigna el valor **3** al parámetro **count** y un nodo primario al parámetro **nodes/rows**, entonces el resultado de la función serán los tres primeros elementos del nodo primario.



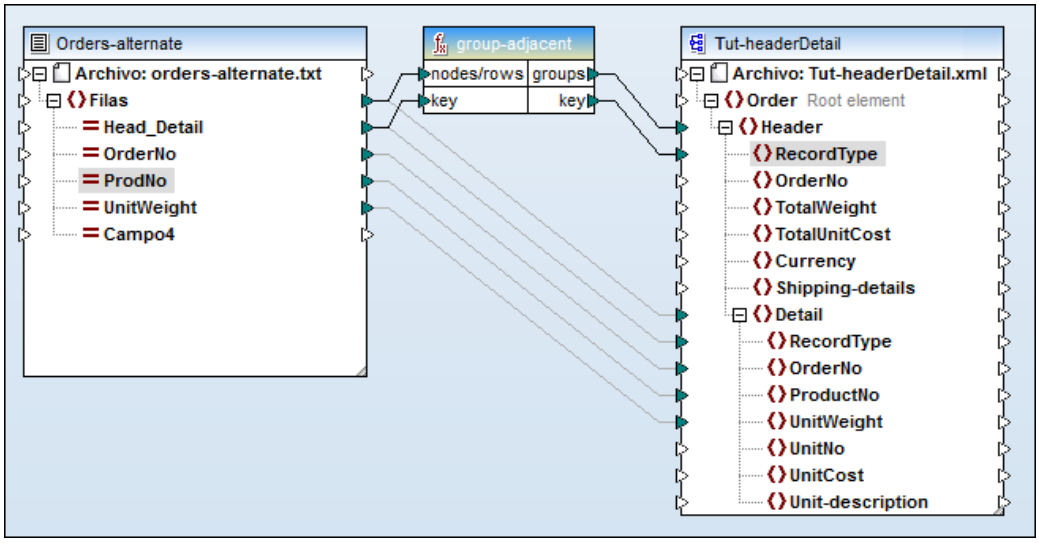
8.8.9.4 *generate-sequence*

Crea una secuencia de enteros usando como límite los parámetros **from** y **to**.



8.8.9.5 *group-adjacent*

Agrupa la secuencia del parámetro de entrada **nodes/rows** en grupos de elementos adyancentes que comparten la misma clave **key**. Observe que la función **group-adjacent** utiliza el contenido del nodo/elemento como clave de agrupación.

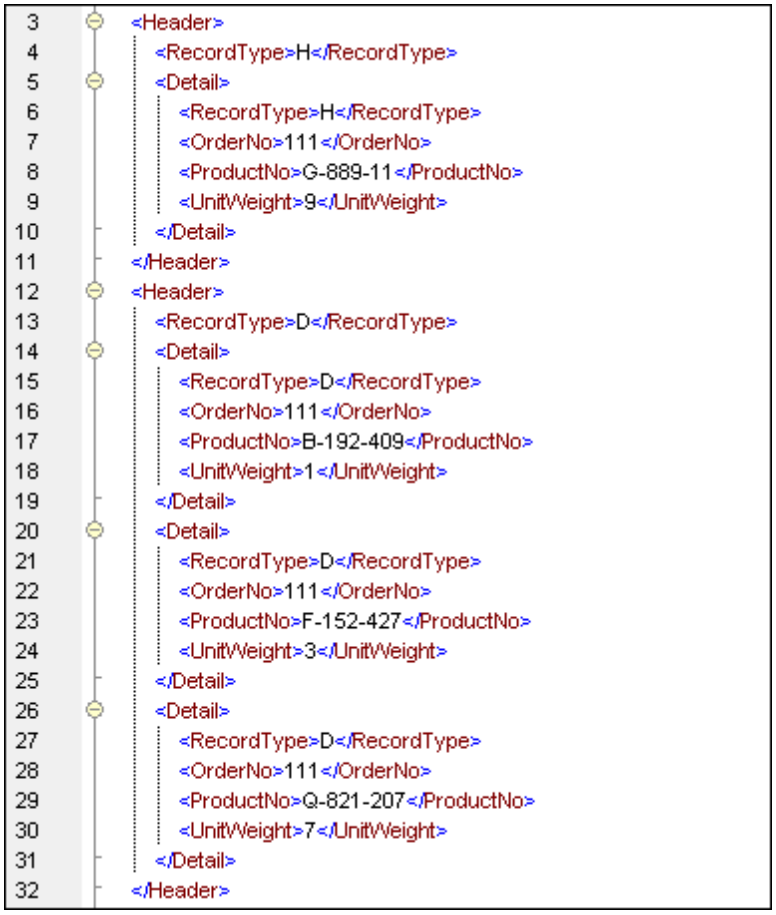


Por ejemplo, imagine que está trabajando con el archivo CSV que aparece a continuación y que necesita agrupar todos los registros Header y Detail.

Head/Detail	OrderNo	ProdNo	Unitweight	Field5	Field6	Field7	Field8	Fi
string	string	string	string	string	string	string	string	st
H	111	G-889-11	9		Container ship			
D	111	B-192-409	1	2	232	Barley		
D	111	F-152-427	3	1	456	Corn		
D	111	Q-821-207	7	5	52	Coconut		
H	222	A-978-4	22		Air freight			
D	222	M-623-111	8	8	78	Oil		
D	222	L-524-201	2	3	669	Miscellaneous		

- Se empieza un grupo nuevo con el primer elemento (en este caso H).
- Como el siguiente elemento (o clave) de la secuencia es diferente (es decir, D), se empieza otro grupo llamado D.
- Ahora los dos elementos D siguientes se añaden al mismo grupo D (porque son del mismo tipo).
- Se empieza un grupo H nuevo con el elemento H.
- A este le sigue un grupo D nuevo que contiene dos elementos D.

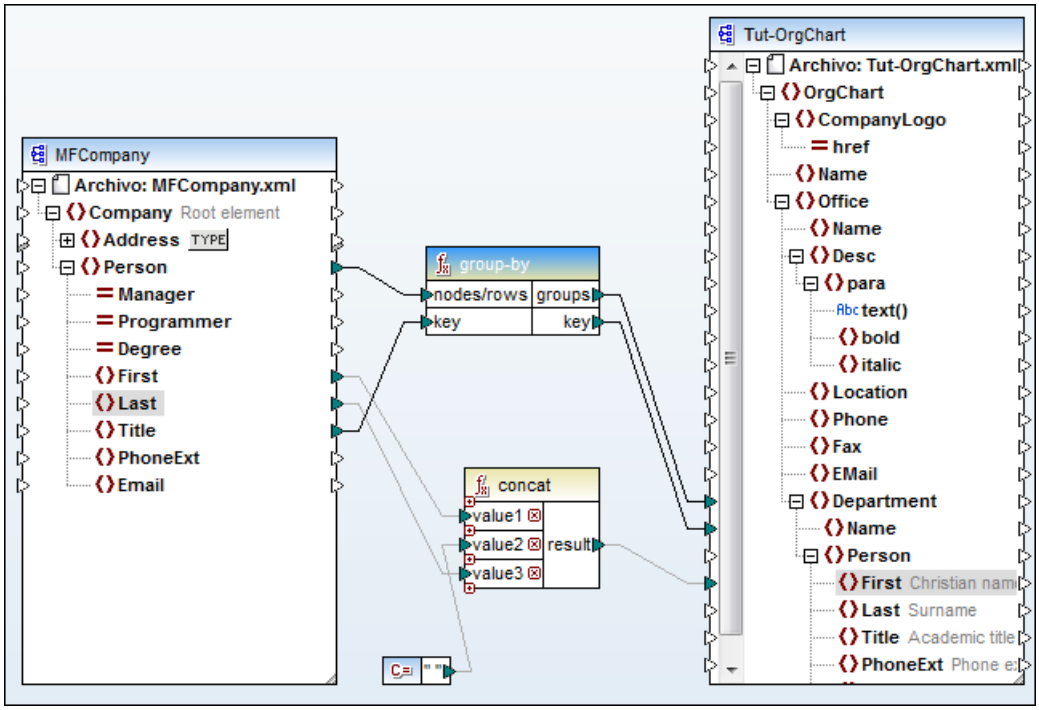
© 2018 Altova GmbH



8.8.9.6 group-by

Agrupa la secuencia del parámetro de entrada **nodes/rows** en grupos de elementos no necesariamente adyacentes que comparten la misma clave **key**. Los grupos se generan en el orden de aparición de la clave en la secuencia de entrada. A continuación puede ver un ejemplo que lo explica mejor:

- La clave que define los grupos del componente de origen es el elemento Title. Esta clave sirve para agrupar las personas de la compañía.
- El nombre del grupo se coloca en el elemento Department/Name del componente de destino, mientras que el nombre y el apellido (concatenados) de las personas se colocan en el elemento secundario Person/First.



Observe que group-by utiliza el **contenido** del nodo/elemento como clave de agrupación. El contenido del campo Title se utiliza para agrupar las personas y se asigna al elemento Department/Name del destino.

Además, en el ejemplo anterior existe un **filtro implícito** entre las filas del documento de origen y del documento de destino. En el documento de destino los elementos Department solamente tienen los elementos Person que **coinciden con la clave** de agrupación porque el componente group-by crea la jerarquía necesaria instantáneamente.

Si tiene una jerarquía plana (CSV, FLF, etc.) con un nombre de archivo de salida dinámico, creada parcialmente a partir del valor de clave, el filtro implícito todavía existe. Esto significa que no hará falta conectar la salida "groups" a ningún elemento del componente de destino.

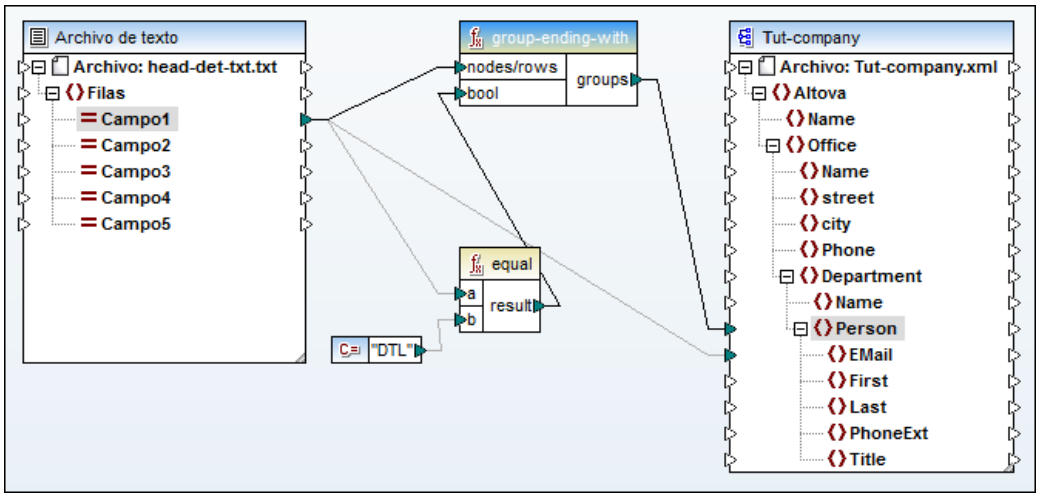
Al hacer clic en el panel *Resultados* puede verse el resultado del proceso de agrupación.

© 2018 Altova GmbH

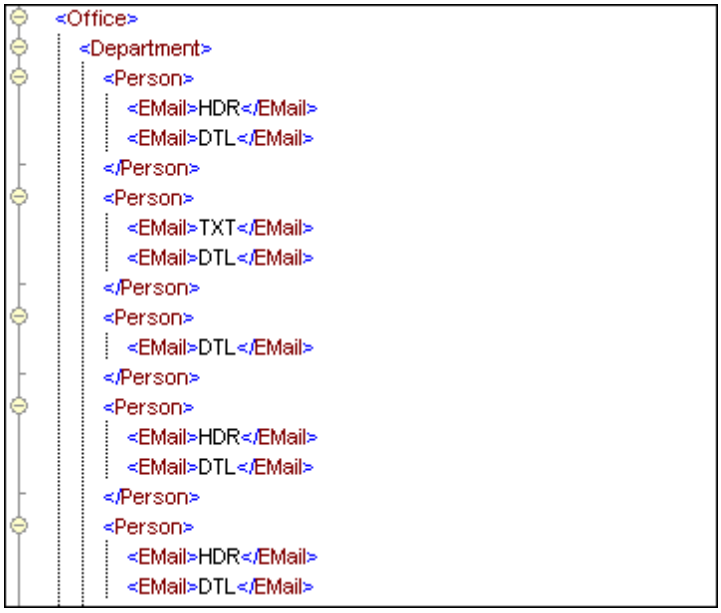
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <OrgChart xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart C:/DOCU
3    <Office>
4      <Department>
5        <Name>Office Manager</Name>
6        <Person>
7          <First>Vernon Callaby</First>
8          <First>Steve Meier</First>
9        </Person>
10     </Department>
11     <Department>
12       <Name>Accounts Receivable</Name>
13       <Person>
14         <First>Frank Further</First>
15         <First>Theo Bone</First>
16       </Person>
17     </Department>
```

8.8.9.7 *group-ending-with*

Agrupa la secuencia del parámetro de entrada **`nodes/rows`** en grupos, terminando con un grupo nuevo si el parámetro **`bool`** es `true`. Este ejemplo muestra el resultado que se obtiene cuando se usa "DTL" como elemento group-ending-with.



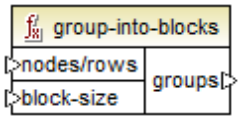
En este caso el **valor** de los elementos/nodos no tienen por qué ser idénticos ni siquiera existir. El nodo "**pattern**" (es decir, los nombres de nodo/elemento) debe ser idéntico para que pueda darse la agrupación.



En el resultado se puede ver que empieza un grupo nuevo cuando "DTL" puede ser el último elemento.

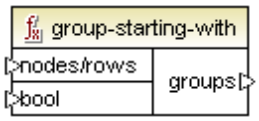
8.8.9.8 group-into-blocks

Agrupa la secuencia de entrada **nodes/rows** en bloques del mismo tamaño. El tamaño de los bloques viene dado por el parámetro **block-size**.



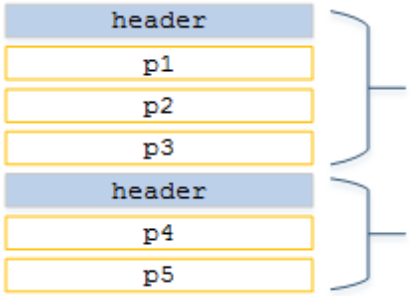
8.8.9.9 group-starting-with

Agrupa la secuencia del parámetro de entrada **nodes/rows** en grupos, empezando con un grupo nuevo si el parámetro **bool** es **true**.

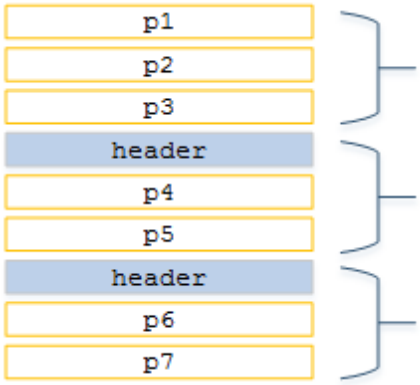


El ejemplo que aparece aquí muestra una secuencia de nodos donde **bool** devuelve `true` cuando se encuentra el nodo "header". Al aplicar la función **group-starting-with** a esta secuencia de nodos se obtienen dos grupos como resultado.

 © 2018 Altova GmbH

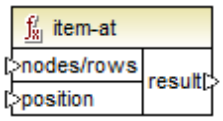


Observe que el primer nodo de la secuencia empieza un grupo nuevo independientemente del valor que tenga **bool**. Es decir, una secuencia como la que aparece a continuación daría lugar a tres grupos.



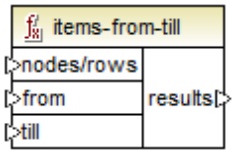
8.8.9.10 item-at

Devuelve los nodos de la secuencia **`nodes/rows`** situados en la posición dada por el parámetro **`position`**. El primer elemento está en la posición **`1`**.



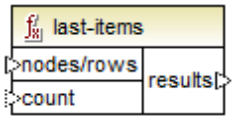
8.8.9.11 items-from-till

Devuelve una secuencia de **`nodes/rows`** usando los parámetros **`from`** y **`till`** como límite de la secuencia. El primer elemento está en la posición **`1`**.



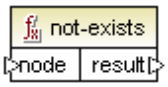
8.8.9.12 last-items

Devuelve los **X** últimos nodos de la secuencia **nodes/rows**, siendo X el número dado por el parámetro **count**. El primer elemento está en la posición **1**.



8.8.9.13 not-exists

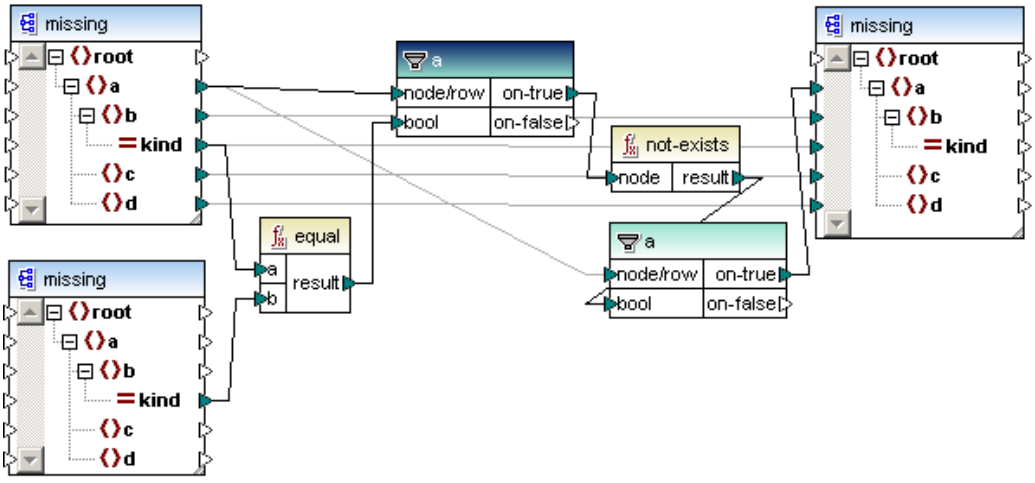
Si el nodo existe, la función devuelve **false**. De lo contrario, devuelve **true**.



En el ejemplo que aparece a continuación se usa la función not-exists para asignar nodos que no existen en uno de los dos archivos de origen.

¿Cómo funciona la asignación del ejemplo?

- Compara los nodos de los dos archivos XML de origen.
- Filtra los nodos del primer archivo XML de origen que no existen en el segundo archivo XML de origen.
- Asigna solamente los nodos que faltan y su contenido al archivo de destino.



Los dos archivos de instancia XML aparecen a continuación. Estas son las diferencias que hay entre los dos archivos:

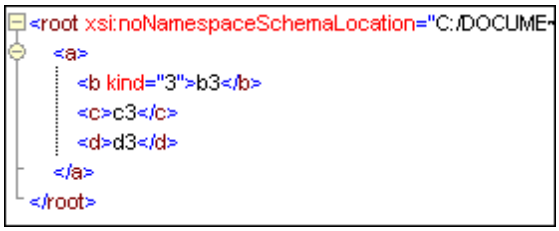
- **a.xml** (izda) contiene el nodo <b **kind="3"**>, que no está en el archivo b.xml.
- **b.xml** (dcha) contiene el nodo <b **kind="4">**, que no está en el archivo a.xml.

© 2018 Altova GmbH



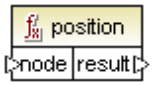
- La función equal compara el tipo de atributo en ambos archivos XML y pasa el resultado al filtro.
- La función not-exists se coloca después del filtro inicial para seleccionar los nodos que faltan en cada uno de los archivos de origen.
- Se usa otro filtro más para pasar solamente el nodo que falta y los demás datos del archivo a.xml al destino.

En el resultado de la asignación el nodo que falta en el archivo b.xml (<b kind="3">) se pasa al componente de destino.



8.8.9.14 position

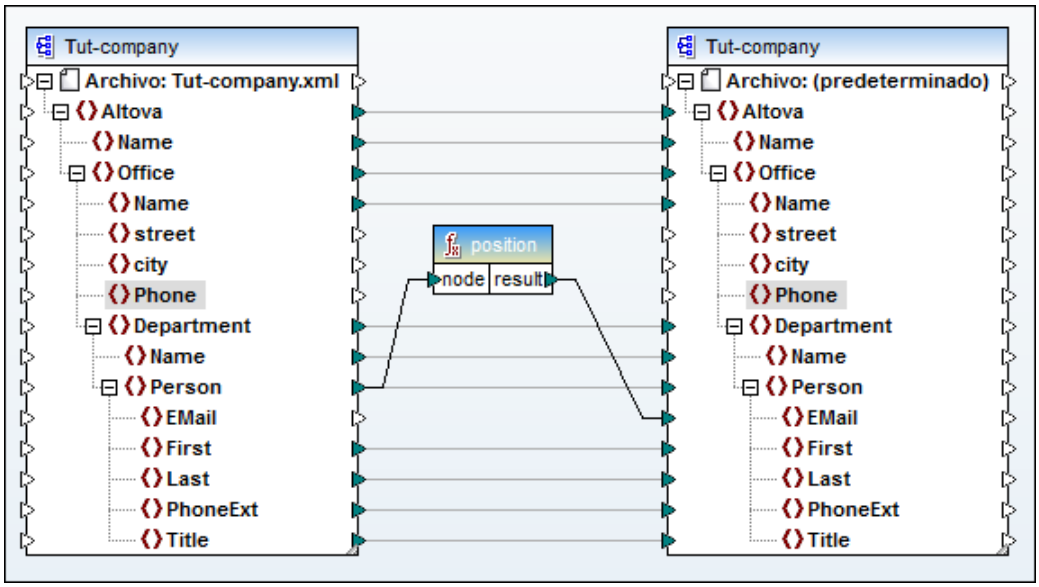
Devuelve la posición de un nodo dentro de la secuencia de la que forma parte.



La función position permite determinar la posición de un nodo de una secuencia o usar una posición concreta para filtrar elementos dependiendo de su posición.

El elemento de contexto viene definido por el elemento que está conectado al parámetro "node" de la función position (Person).

En la asignación que aparece a continuación se añade un número de posición a cada elemento Person de cada elemento Department.



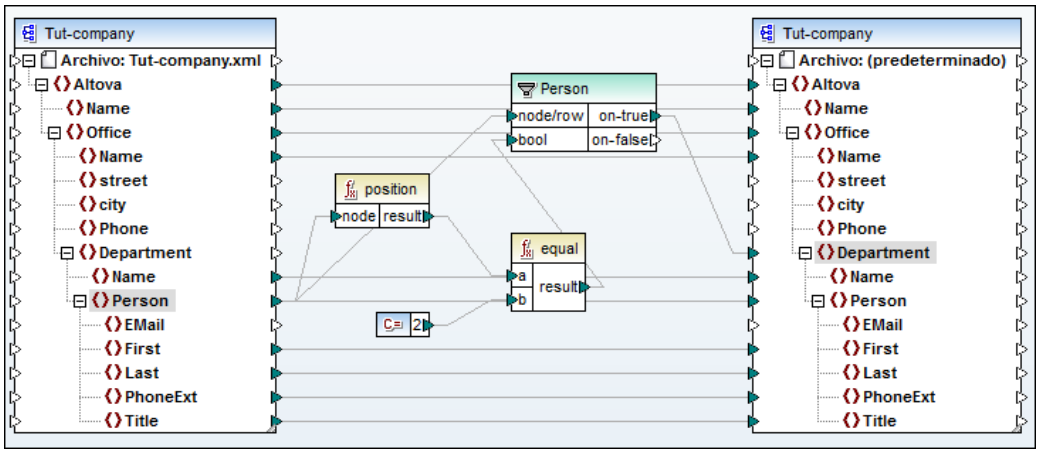
El número de posición se restaura para cada elemento Department del elemento Office.

```
<Office>
  <Name>Microtech, Inc.</Name>
  <Department>
    <Name>Admin</Name>
    <Person>
      <EMail>1</EMail>
      <First>Albert</First>
      <Last>Aldrich</Last>
      <PhoneExt>582</PhoneExt>
      <Title>Manager</Title>
    </Person>
    <Person>
      <EMail>2</EMail>
      <First>Bert</First>
      <Last>Bander</Last>
      <PhoneExt>471</PhoneExt>
      <Title>Accounts Receivable</Title>
    </Person>
```

Si usa la función position junto con un filtro podrá asignar solamente determinados nodos que tengan cierta posición en el componente de origen.

El parámetro de filtro "node/row" y el parámetro "node" de la función position deben estar conectados al mismo elemento del componente de origen para poder filtrar la posición concreta de la secuencia.

© 2018 Altova GmbH



El resultado que genera esta asignación es:

- El segundo elemento Person de cada elemento Department de cada elemento Office

```
<Office>
  <Name>Microtech, Inc.</Name>
  <Department>
    <Name>Admin</Name>
    <Person>
      <EMail>b.bander@microtech.com</EMail>
      <First>Bert</First>
      <Last>Bander</Last>
      <Title>Accounts Receivable</Title>
    </Person>
  </Department>
  <Department>
    <Name>Sales and Marketing</Name>
    <Person>
      <EMail>e.ellas@microtech.com</EMail>
      <First>Eve</First>
      <Last>Ellas</Last>
      <Title>Art Director</Title>
    </Person>
  </Department>
  <Department>
    <Name>Manufacturing</Name>
    <Person>
      <EMail>g.gundall@microtech.com</EMail>
```

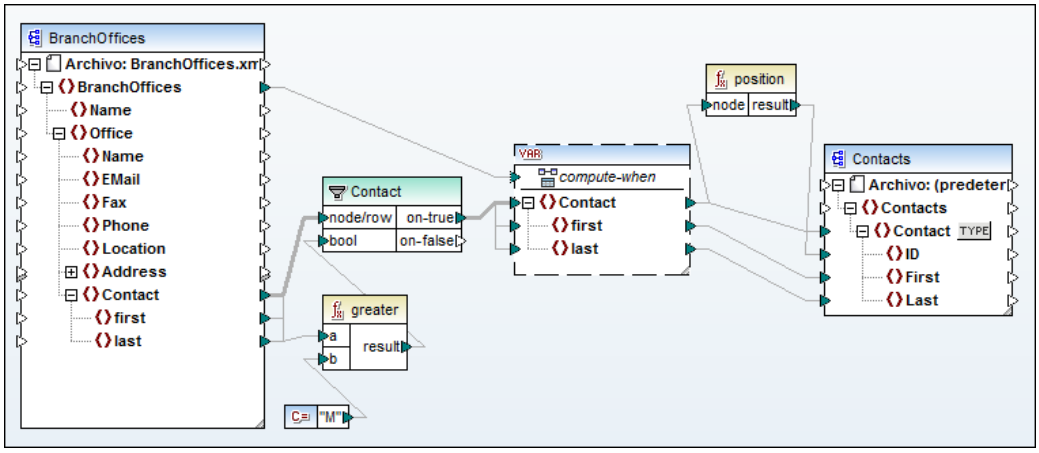
Buscar la posición de los elementos en una secuencia filtrada:

Como el componente de filtrado no es una función de secuencia, no se puede usar directamente con la función position para buscar la posición de los elementos filtrados. En este caso debe utilizarse el componente de variable.

Los resultados de los componentes de variable siempre son secuencias (es decir, listas delimitadas de valores), que se pueden usar para crear secuencias.

- El componente de variable sirve para recopilar los contactos filtrados cuyo apellido empiece con una letra superior a "M".

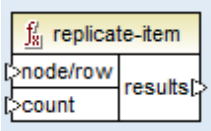
- Después se pasan los contactos (desde la variable) al componente de destino.
- Después la función posicion numera estos contactos de forma secuencial.



8.8.9.15 replicate-item

Repeats every item in the input sequence the number of times specified in the `count` argument. If you connect a single item to the `node/row` argument, the function returns N items, where N is the value of the `count` argument. If you connect a sequence of items to the `node/row` argument, the function repeats each individual item in the sequence `count` times, processing one item at a time. For example, if `count` is 2, then the sequence `(1,2,3)` produces `(1,1,2,2,3,3)`.

Repite cada elemento de la secuencia de entrada tantas veces como se indique en el argumento **`count`**. Si conecta un solo elemento al argumento `node/row`, la función devuelve N elementos, siendo N el valor del argumento `count`. Si conecta una secuencia de elementos al argumento `node/row`, la función repite cada uno de los elementos de la secuencia tantas veces como indique `count` y los procesa uno por uno. Por ejemplo, si `count` es 2, entonces la secuencia `(1,2,3)` produce `(1,1,2,2,3,3)`.



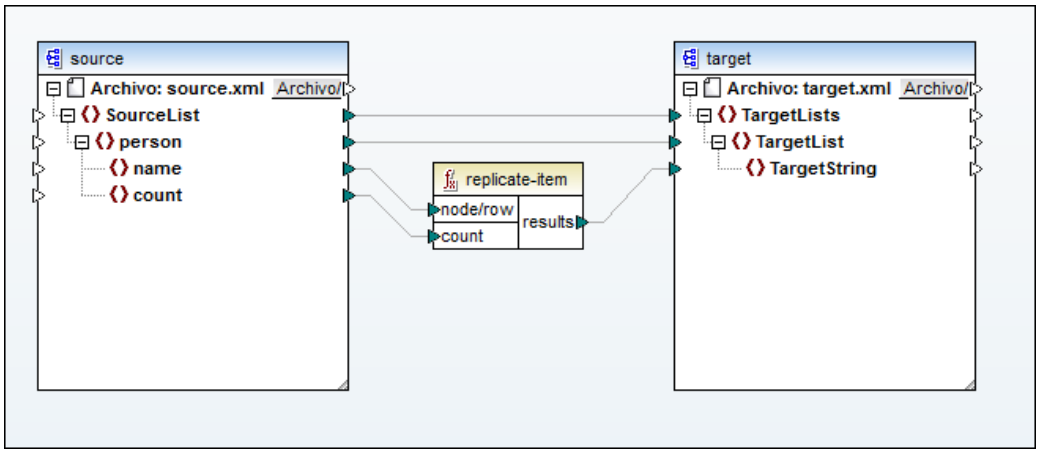
Además, recuerde que puede dar un valor `count` distinto para cada elemento. Por ejemplo, imagine que tiene un archivo XML de origen con esta estructura:

```
<?xml version="1.0" encoding="UTF-8"?>
<SourceList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="source.xsd">
   <person>
      <name>Michelle</name>
      <count>2</count>
```


© 2018 Altova GmbH

```
   </person>
   <person>
      <name>Ted</name>
      <count>4</count>
   </person>
   <person>
      <name>Ann</name>
      <count>3</count>
   </person>
</SourceList>
```

Con ayuda de la función **replicate-item** puede repetir cada nombre de persona las veces que quiera en el componente de destino. Para conseguirlo conecte el nodo <count> de cada persona a la entrada count de la función **replicate-item**:



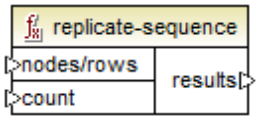
Este sería el resultado:

```
<?xml version="1.0" encoding="UTF-8"?>
<TargetLists xsi:noNamespaceSchemaLocation="target.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <TargetList>
      <TargetString>Michelle</TargetString>
      <TargetString>Michelle</TargetString>
   </TargetList>
   <TargetList>
      <TargetString>Ted</TargetString>
      <TargetString>Ted</TargetString>
      <TargetString>Ted</TargetString>
      <TargetString>Ted</TargetString>
   </TargetList>
   <TargetList>
      <TargetString>Ann</TargetString>
      <TargetString>Ann</TargetString>
      <TargetString>Ann</TargetString>
   </TargetList>
```

```
</TargetLists>
```

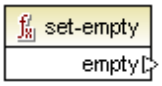
8.8.9.16 replicate-sequence

Repite todos los elementos de la secuencia de entrada tantas veces como indique el argumento `count`. Por ejemplo, si `count` es 2, entonces la secuencia `(1,2,3)` produce `(1,2,3,1,2,3)`.



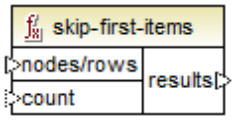
8.8.9.17 set-empty

Devuelve una secuencia vacía. Por ejemplo, puede usar esta función para cancelar los valores predeterminados de un documento XBRL definidos al principio de la taxonomía/del componente XBRL.



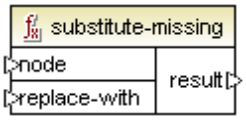
8.8.9.18 skip-first-items

Pasa por alto los X primeros nodos/elementos de la secuencia de entrada (siendo X el número dado por el parámetro **count**) y devuelve el resto de la secuencia.



8.8.9.19 substitute-missing

Esta función es una mezcla de la función **exists** y la condición **if-else**. Con ella puede asignar el contenido del campo actual si el nodo existe en el archivo XML de origen. De lo contrario utiliza el elemento asignado al parámetro **replace-with**.



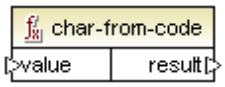
© 2018 Altova GmbH

8.8.10 core | string functions (cadena)

La biblioteca **core | string** ofrece las funciones de cadena más comunes para manipular diferentes tipos de datos de origen y extraer porciones, comprobar si existen subcadenas o recuperar información de las cadenas.

8.8.10.1 char-from-code

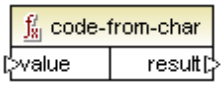
El resultado es el carácter que represente el valor Unicode decimal del parámetro **`value`**.



Para ver un ejemplo consulte el apartado Reemplazar caracteres especiales.

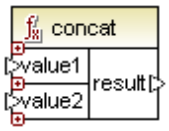
8.8.10.2 code-from-char

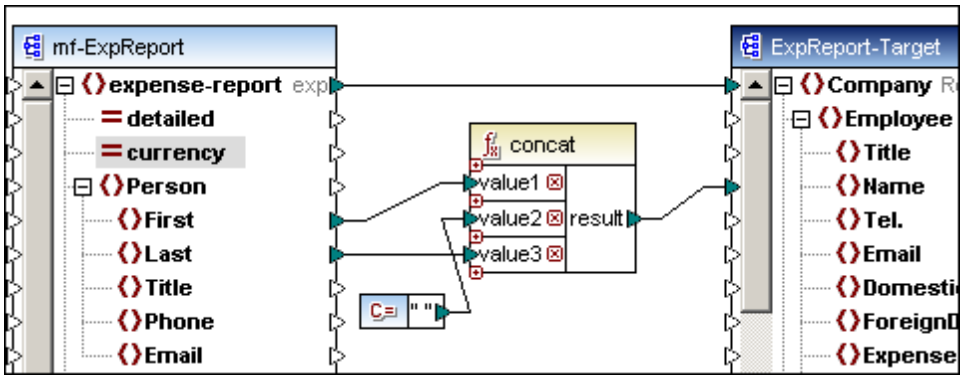
El resultado es el valor Unicode decimal del primer carácter del parámetro **`value`**.



8.8.10.3 concat

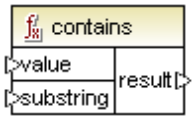
Concatena (anexa) dos o más valores dando lugar a una sola cadena. Todos los valores de entrada se convierten automáticamente en valores de tipo `string`.





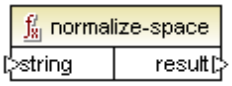
8.8.10.4 contains

El resultado es **`true`** si los datos asignados al parámetro de entrada **`value`** contienen la cadena asignada al parámetro de entrada **`substring`**.



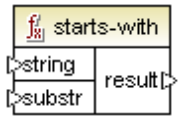
8.8.10.5 normalize-space

El resultado es la cadena de entrada normalizada (es decir, se eliminan los espacios iniciales y finales y cada secuencia de espacios en blancos consecutivos se reemplaza con un solo espacio en blanco). El carácter Unicode para el espacio es (U+0020).



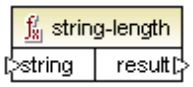
8.8.10.6 starts-with

El resultado es **`true`** si la cadena de entrada **`string`** empieza por la cadena de **`substr`**. De lo contrario, el resultado es **`false`**.



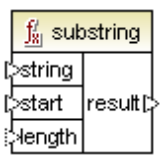
8.8.10.7 string-length

El resultado es el número de caracteres asignados al parámetro de entrada **`string`**.



8.8.10.8 substring

El resultado es la subcadena (fragmento de cadena) del parámetro de entrada **`string`**, siendo **`start`** la posición del carácter de inicio y **`length`** la longitud de la subcadena.



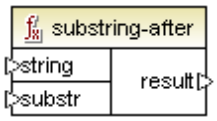
© 2018 Altova GmbH

Si no se indica el parámetro **length**, el resultado es un fragmento que empieza en la posición de inicio y termina en la posición final de la cadena. Los índices cuentan desde 1.

P. ej. la función **substring("56789",2,3)** da como resultado 678.

8.8.10.9 substring-after

El resultado es el fragmento de la cadena **string** a partir de la primera aparición del parámetro **substr**. Si la cadena del parámetro **substr** no aparece en el parámetro **string**, el resultado de la función es una cadena vacía.

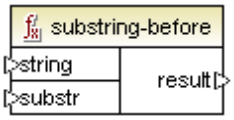


P. ej. la función **substring-after("2009/01/04","/")** da como resultado la subcadena 01/04.

Para ver un ejemplo de uso consulte el apartado Ejemplo: asignar datos de un canal RSS.

8.8.10.10 substring-before

El resultado es el fragmento de la cadena del parámetro **string** hasta la primera aparición de los caracteres del parámetro **substr**. Si los caracteres de **substr** no aparecen en **string**, el resultado es una cadena vacía.



P. ej. la función **substring-before ("2009/01/04","/")** da como resultado la subcadena 2009.

8.8.10.11 tokenize

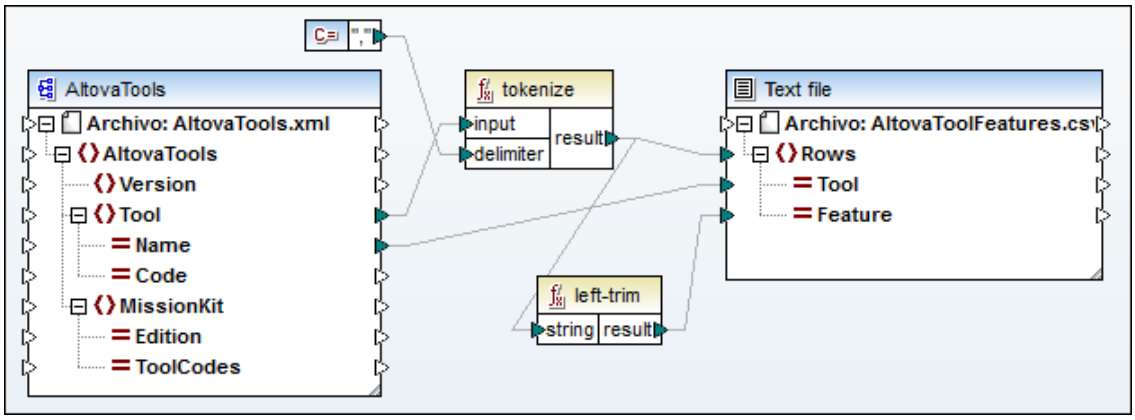
El resultado es la cadena **input** dividida en una secuencia formada por caracteres delimitados por el parámetro **delimiter**. El resultado se puede seguir procesando.



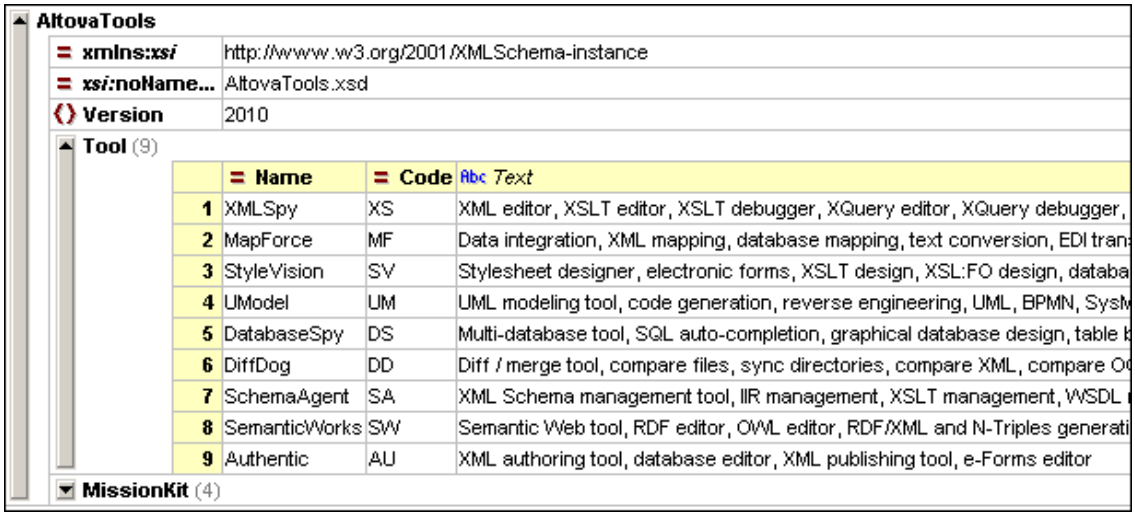
P. ej. la cadena **input** es A,B,C y el delimitador es ",". El resultado será A B C.

Ejemplo

En el diseño de asignación de datos **tokenizeString1.mfd** de la carpeta ...\MapForceExamples puede ver un ejemplo de uso de la función tokenize.



A continuación puede ver el archivo XML de origen. El elemento **Tool** tiene dos atributos: Name y Code. Y los datos del elemento **Tool** es texto delimitado por comas.



AltovaTools

xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance	
xsi:noName...	AltovaTools.xsd	
Version	2010	

Tool (9)

	Name	Code	Text
1	XMLSpy	XS	XML editor, XSLT editor, XSLT debugger, XQuery editor, XQuery debugger,
2	MapForce	MF	Data integration, XML mapping, database mapping, text conversion, EDI tran
3	StyleVision	SV	Stylesheet designer, electronic forms, XSLT design, XSL:FO design, databa
4	UModel	UM	UML modeling tool, code generation, reverse engineering, UML, BPMN, Sysl
5	DatabaseSpy	DS	Multi-database tool, SQL auto-completion, graphical database design, table b
6	DiffDog	DD	Diff / merge tool, compare files, sync directories, compare XML, compare O
7	SchemaAgent	SA	XML Schema management tool, IIR management, XSLT management, WSDL
8	SemanticWorks	SW	Semantic Web tool, RDF editor, OWL editor, RDF/XML and N-Triples generati
9	Authentic	AU	XML authoring tool, database editor, XML publishing tool, e-Forms editor

MissionKit (4)

Características de esta asignación de datos

- La función tokenize recibe datos del elemento **Tool** y utiliza el delimitador "**,**" para dividir los datos en secciones (p. ej. la primera sección es "XML editor").
- Como el parámetro result está asignado al elemento **Rows** del componente de destino, se genera una **fila** por cada sección.
- El parámetro result también está asignado a la función **left-trim**, que elimina los espacios en blanco iniciales de cada sección.
- El resultado del parámetro left-trim se asigna al elemento **Feature** del componente de destino.
- El archivo de salida del componente de destino se definió como archivo CSV (AltovaToolFeatures.csv) con el delimitador de campos punto y coma (haga doble clic en el componente para ver su configuración).

Resultado de la asignación de datos

- Por cada elemento Tool del archivo de origen se asigna Name al elemento Tool del componente de destino.

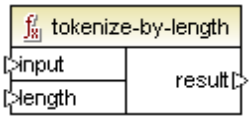
 © 2018 Altova GmbH

- Cada sección del contenido dividido de Tool se anexa al elemento Feature.
- Por ejemplo, la primera herramienta (XMLSpy) obtiene la primera sección de Feature (XML editor).
- Esto se repite para cada sección del elemento Tool actual y después para todos los elementos Tool.
- A continuación puede ver el resultado en el panel *Resultados*.

```
1   Tool;Feature
2   XMLSpy;XML editor
3   XMLSpy;XSLT editor
4   XMLSpy;XSLT debugger
5   XMLSpy;XQuery editor
6   XMLSpy;XQuery debugger
7   XMLSpy;XML Schema / DTD editor
8   XMLSpy;WSDL editor
9   XMLSpy;SOAP debugger
10  MapForce;Data integration
11  MapForce;XML mapping
12  MapForce;database mapping
```

8.8.10.12 tokenize-by-length

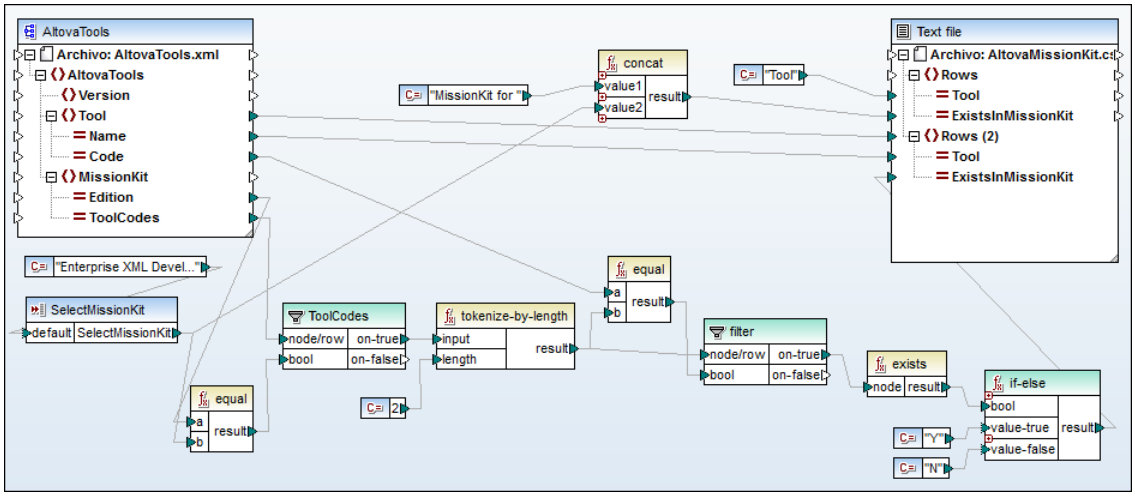
El resultado es la cadena **`input`** dividida en una secuencia formada por tantos caracteres como defina el parámetro **`length`**. El resultado se puede seguir procesando.



P. ej. la cadena **`input`** es ABCDEF y el parámetro **`length`** es 2 . El resultado es AB CD EF.

Ejemplo

En el diseño de asignación de datos **tokenizeString2.mfd** de la carpeta ...\MapForceExamples puede ver un ejemplo de uso de la función **tokenize-by-length**.



El archivo XML de origen se puede ver más abajo y es el mismo que el del ejemplo anterior.

El elemento **MissionKit** también tiene dos atributos (**Edition** y **ToolCodes**) pero no tiene contenido.

Tool (9)

	= Name	= Code	Abc Text
1	XMLSpy	XS	XML editor, XSLT editor, XSLT debugger, XQuery editor, XQuery debugger, XML S
2	MapForce	MF	Data integration, XML mapping, database mapping, text conversion, EDI translator,
3	StyleVision	SV	Stylesheet designer, electronic forms, XSLT design, XSL:FO design, database rep
4	UModel	UM	UML modeling tool, code generation, reverse engineering, UML, BPMN, SysML, pro
5	DatabaseSpy	DS	Multi-database tool, SQL auto-completion, graphical database design, table brows
6	DiffDog	DD	Diff / merge tool, compare files, sync directories, compare XML, compare OOXML,
7	SchemaAgent	SA	XML Schema management tool, IIR management, XSLT management, WSDL manag
8	SemanticWorks	SW	Semantic Web tool, RDF editor, OWL editor, RDF/XML and N-Triples generation and
9	Authentic	AU	XML authoring tool, database editor, XML publishing tool, e-Forms editor

MissionKit (4)

	= Edition	= ToolCodes
1	Enterprise Software Architects	XSMFSVUMDSDDSASW
2	Professional Software Architects	XSMFSVUMDS
3	Enterprise XML Developers	XSMFSVDDSASW
4	Professional XML Developers	XSMFSV

Objetivo de la asignación

Generar una lista que muestre qué herramientas de Altova forman parte de cada edición de MissionKit.

Características de esta asignación de datos

- El componente de entrada **SelectMissionKit** recibe su entrada predeterminada de una constante ("Enterprise XML Developers").
- La función **equal** compara el valor de entrada con el valor "**Edition**" y pasa el resultado al parámetro **bool** del filtro ToolCodes.
- La entrada **node/row** del filtro ToolCodes viene dada por el elemento **ToolCodes** del archivo de origen. El valor para la edición Enterprise XML Developers es: XSMFSVDDSASW.
- El valor XSMFSVDDSASW se envía al parámetro **on-true** y después al parámetro **input** de la función **tokenize-by-length**.

¿Qué hace la función tokenize-by-length?

- El valor de entrada ToolCodes es XSMFSVDDSASW y se divide en varias secciones de dos caracteres cada una (definidas por el parámetro **length** que es 2, lo cual da lugar a 6 secciones).
- Cada sección (situada en el parámetro b) de la función equal se compara con el valor **Code** de dos caracteres del archivo de origen (del cual existen 9 entradas en total).
- El resultado de la comparación (true/false) se envía al parámetro **bool** del filtro.
- **Todas** las secciones de la función tokenize-by-length se envían al parámetro **node/row** del filtro.
- La función **exists** busca los nodos que existen o que faltan que recibió del parámetro **on-true** del componente de filtrado.

  Los nodos que existen son aquellos en los que la sección **ToolCodes** y el valor Code **coinciden**.

© 2018 Altova GmbH

Los nodos que faltan son aquellos donde **ninguna sección ToolCodes coincide** con ningún valor Code.

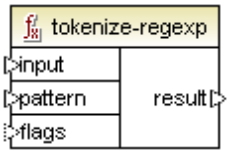
- Los resultados bool de la función **exists** se pasan a la función **if-else** que pasa una Y al destino si el nodo existe y una N si no existe.

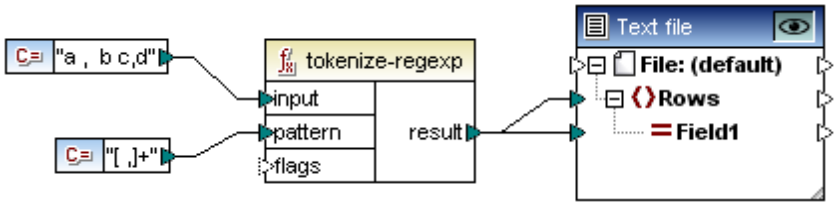
Resultado de la asignación

```
1   Tool;MissionKit for Enterprise XML Developers
2   XMLSpy;Y
3   MapForce;Y
4   StyleVision;Y
5   UModel;N
6   DatabaseSpy;N
7   DiffDog;Y
8   SchemaAgent;Y
9   SemanticWorks;Y
10  Authentic;N
11
```

8.8.10.13 tokenize-regexp

El resultado es la cadena de entrada dividida en una secuencia de cadenas, en la que la expresión regular **`pattern`** define el separador. El parámetro **`result`** no genera los separadores. También puede usar marcas opcionales.





En el ejemplo anterior la cadena **`input`** es una serie de caracteres separados por espacios y/o comas (es decir, `a , b c,d`).

La expresión regular **`pattern`** define una clase de caracteres **[**"espacio""coma"**]** de los cuales se usará un solo carácter (es decir, o bien el espacio, o bien la coma=.

El cuantificador + especifica "una o más" apariciones de la clase de caracteres/cadena.

La cadena resultante es:

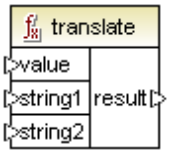
1	a
2	b
3	c
4	d
5	

Recuerde que la sintaxis de las expresiones regulares varía ligeramente de un lenguaje a otro. La función **tokenize-regexp** en C++ solamente está disponible en Visual Studio 2008 SP1 (o superior).

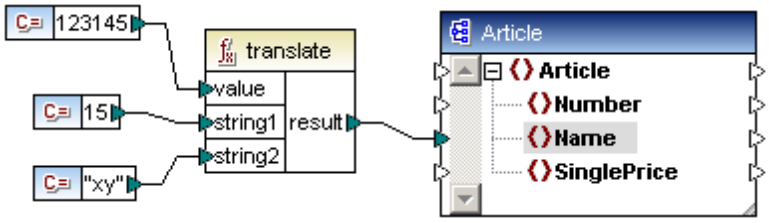
Consulte el apartado Expresiones regulares para más información.

8.8.10.14 translate

Los caracteres de la cadena **string1** (*cadena de búsqueda*) se reemplazan con los caracteres de la cadena **string2** (*cadena de reemplazo*) en la cadena de entrada **value**.



Si en la cadena **string2** no hay caracteres que correspondan, entonces se elimina el carácter.



Por ejemplo:
-la cadena de entrada **value** es 123145
-la cadena de búsqueda **string1** es 15
-la cadena de reemplazo **string2** es xy

Como resultado:
-cada carácter **1** se reemplaza con **x** en la cadena de entrada **value**
-cada carácter **5** se reemplaza con **y** en la cadena de entrada **value**

La cadena resultante es **x23x4y**

Si el parámetro **string2** está vacío, el carácter de **string1** se elimina.

Por ejemplo:
- la cadena de entrada **value** es aabaacbca
- la cadena de búsqueda **string1** es "**a**"
- la cadena de reemplazo **string2** es "" (una cadena vacía)

 © 2018 Altova GmbH

La cadena resultante es **bcbc**

Otro ejemplo:
-la cadena de entrada **value** es aabaacbca
-la cadena de búsqueda **string1** es "**ac**"
-la cadena de reemplazo **string2** es "**ca**"

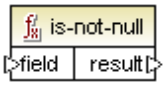
La cadena resultante es **ccbccabac**

8.8.11 db (base de datos)

La biblioteca **db** contiene funciones que sirven para definir los resultados de la asignación cuando se encuentran campos NULL en bases de datos.

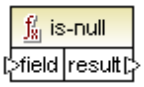
8.8.11.1 is-not-null

Si el campo es NULL, la función devuelve **false**. De lo contrario devuelve **true**.



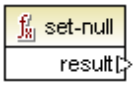
8.8.11.2 is-null

Si el campo es NULL, la función devuelve **true**. De lo contrario devuelve **false**.



8.8.11.3 set-null

Se utiliza para configurar una columna de la BD como NULL. Esta función también sobrescribe el valor predeterminado de la columna con NULL. Si no se conecta a un campo de BD sino a otro componente, funciona como una secuencia vacía.



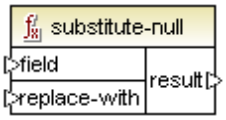
Nota:

- Si conecta **set-null** a otra función, no se llama a la otra función en absoluto. Si conecta **set-null** a una función de secuencia (p. ej. **count**), se llama a la función con una secuencia vacía.
- Si conecta **set-null** a funciones especiales, los filtros y las condiciones **IF-Else** funcionan normalmente y los campos se configuran como NULL. En los filtros esto corresponde a la entrada **node/row**.
- Si usa **set-null** como entrada para un elemento **simpleType**, no se creará ese

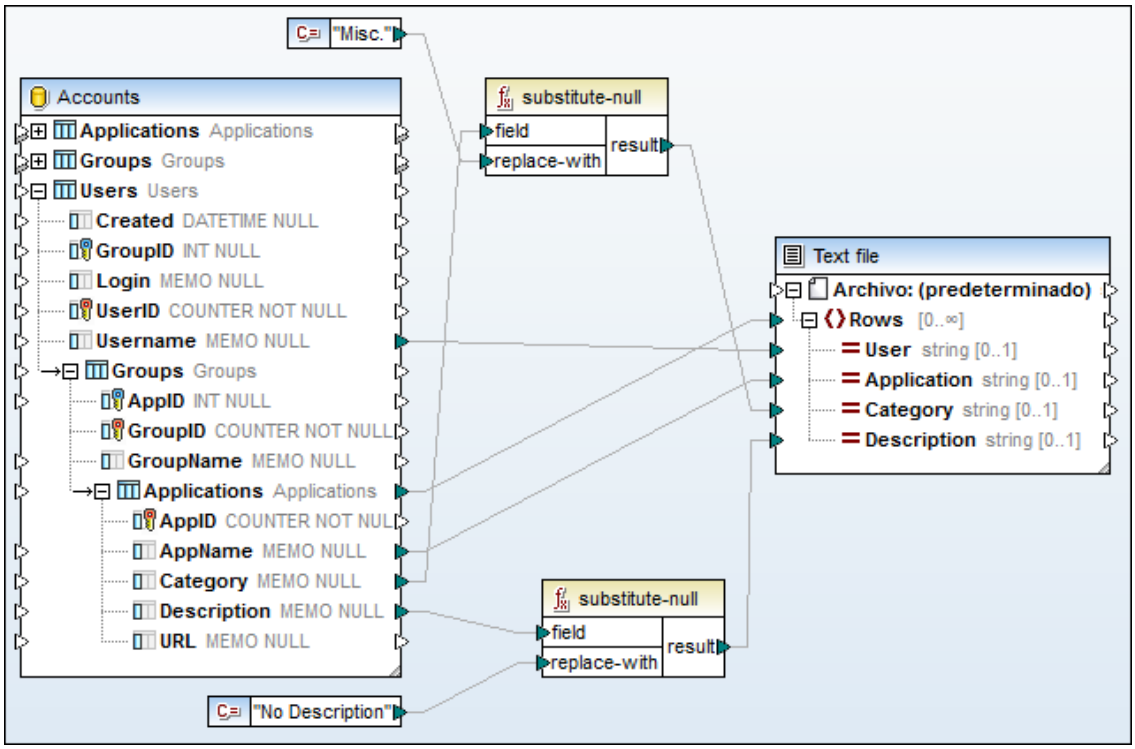
elemento en el componente de destino.

8.8.11.4 substitute-null

Sirve para asignar el contenido del campo actual si es que existe. De lo contrario se utiliza el elemento asignado al parámetro **`replace-with`**.



La imagen siguiente muestra una asignación en la que se usa la función **`substitute-null`**. Se trata del archivo de asignación **DB-ApplicationList**, disponible en la carpeta **... \MapForceExamples**.



La primera función **`substitute-null`** (la que aparece en la parte superior) comprueba si existe la entrada **`Category`** en la tabla **`Applications`**. Puesto que **`Category`** no existe en la aplicación **`Notepad`**, se asigna la cadena de texto `Misc.` al elemento **`Category`** del archivo de texto.

La segunda función **`substitute-null`** (la que aparece en la parte inferior) comprueba si existe la entrada **`Description`** y, si no existe, asigna la cadena de texto `No description`.

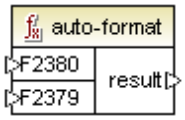
© 2018 Altova GmbH

8.8.12 edifact

La biblioteca **edifact** ofrece funciones para convertir formatos `date, time` y `period` de EDI en datos `xs:date`.

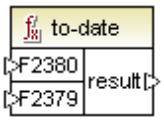
8.8.12.1 auto-format

El resultado es el valor `date`, `time` o `datetime` extraído de la fuente codificada (F2380) dado el código de formato (F2379).



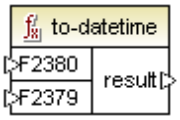
8.8.12.2 to-date

El resultado es el valor `date` extraído de la fuente codificada (F2380) dado el código de formato (F2379).



8.8.12.3 to-datetime

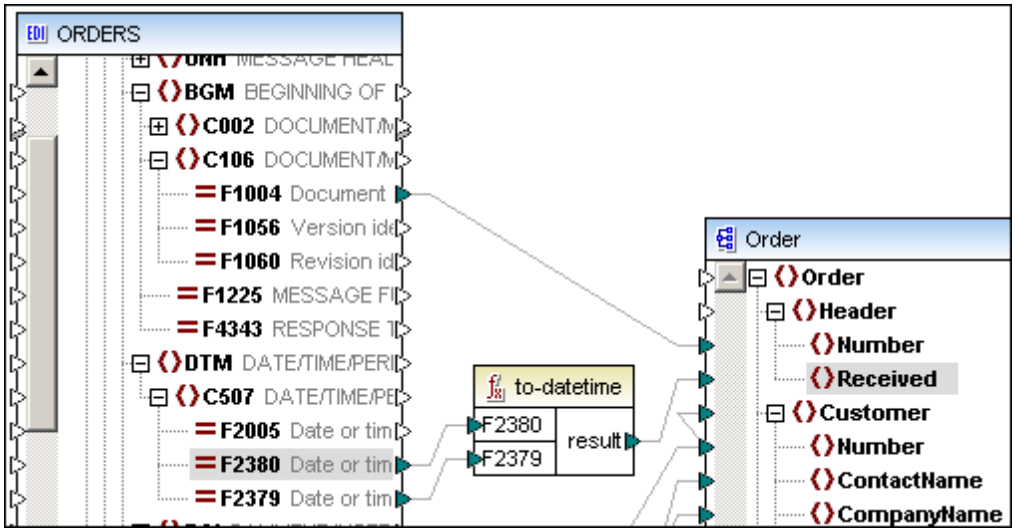
El resultado es el valor `datetime` extraído de la fuente codificada (F2380) dado el código de formato (F2379).



Por ejemplo, en el archivo EDI siguiente, el valor `Date/Time/Period` aparece en el campo `DTM`.

```
UNB+UNOB:1+003897733:01:MFGB+PARTNER ID:ZZ:ROUTING
ADDR+970101:1230+00000000000001++ORDERS++++1'
UNH+0001+ORDERS:D:08A:UN'
BGM+221+ABC123456XYZ+9'
DTM+4:200404301742PDT:303'
FTX+PUR+3++Pizza purchase order'
RFF+CT:123-456'
RFF+CR:1122'
NAD+SE+999::92++24h Pizza+Long Way+San-Francisco+C
CTA+SR+:Ted Little'
```

Este valor se convierte en `xs:date` usando la función **`to-datetime`**.

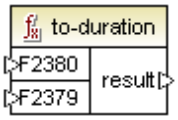


El resultado aparece a continuación. Este ejemplo está en el archivo **EDI-Order.mfd** de la carpeta ...\MapForceExamples.

```
<?xml version="1.0" encoding="UTF-8"?>
<Order xsi:noNamespaceSchemaLocation="C:/DOCUME~1
http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://w
  <Header>
    <Number>ABC123456XYZ</Number>
    <Received>2004-04-30T17:42:00-09:00</Received>
  </Header>
  <Customer>
    <Number>123</Number>
```

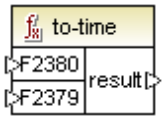
8.8.12.4 *to-duration*

El resultado es el valor `Duration` extraído de la fuente codificada (F2380) dado el código de formato (F2379).



8.8.12.5 *to-time*

El resultado es el valor `Time` extraído de la fuente codificada (F2380) dado el código de formato (F2379).



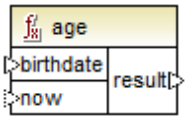
© 2018 Altova GmbH

8.8.13 lang | datetime functions (fechaHora)

La biblioteca **lang** contiene funciones para trabajar con los lenguajes Java, C# o C++.

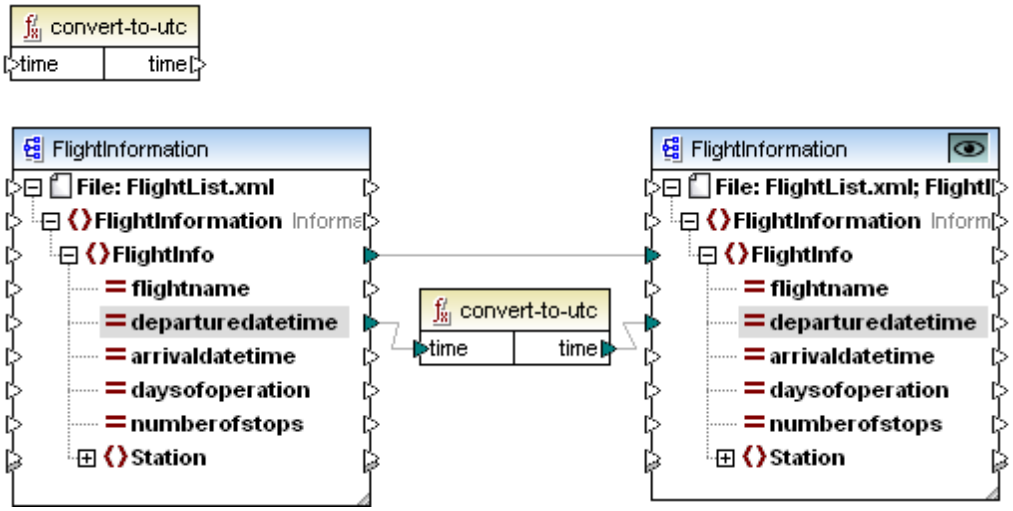
8.8.13.1 age

age
El resultado es los años de edad de la persona. El argumento `now` es opcional y el valor predeterminado es la fecha actual del sistema. El resultado es la cantidad total de años comprendidos entre la fecha de nacimiento y la fecha actual. Si hay un valor asignado al argumento `now`, el resultado es el número de años comprendido entre ambas fechas.



8.8.13.2 convert-to-utc

Convierte el parámetro de entrada local **`time`** en la hora universal coordinada o GMT. (La función tiene en cuenta el componente de uso horario, p. ej. `+5:00`).



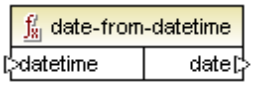
Por ejemplo, el valor `datetime` del documento de instancia:
`departuredatetime="2001-12-17T09:30:02+05:00"`

Resultado:
`departuredatetime="2001-12-17T04:30:02"`

Nota: si el valor `dateTime` de origen tiene el formato `departuredatetime="2001-12-17T09:30:02Z"` no se convierten los datos porque la **Z final** define que es Hora Z, es decir, UTC. El resultado será `departuredatetime="2001-12-17T09:30:02".`

8.8.13.3 date-from-datetime

El resultado es la parte `date` del argumento de entrada **`datetime`**. La parte `date` del **`dateTime`**, empezando por la T del documento de instancia, se fija en 0. Observe que el incremento de uso horario no se cambia.

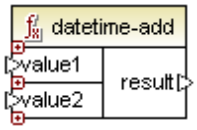


Por ejemplo, el valor `datetime` del documento de instancia:
`departuredatetime="2001-12-17T09:30:02+05:00"`

Resultado:
`departuredatetime="2001-12-17T00:00:00+05:00"`

8.8.13.4 datetime-add

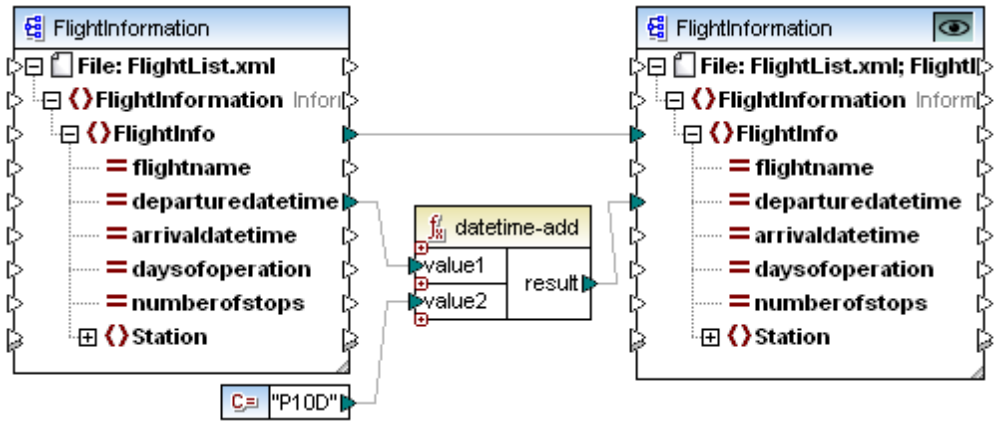
El resultado es el valor `datetime` que se obtiene al añadir un valor `duration` (segundo argumento) a un valor `datetime` (primer argumento).



Los valores `duration` se deben introducir en este formato: P1Y2M3DT04H05M. Los valores de período pueden convertirse en negativos añadiendo el carácter - antes del designador P (por ejemplo, -P1D).

P es el designador de período y es obligatorio. El resto sería:
1 año (Y), 2 meses (M), 3 días (D), designador de hora T, 04 horas (H), 05 minutos (M).

Por ejemplo, en la asignación siguiente se añaden diez días al elemento `departuredatetime` (P10D).



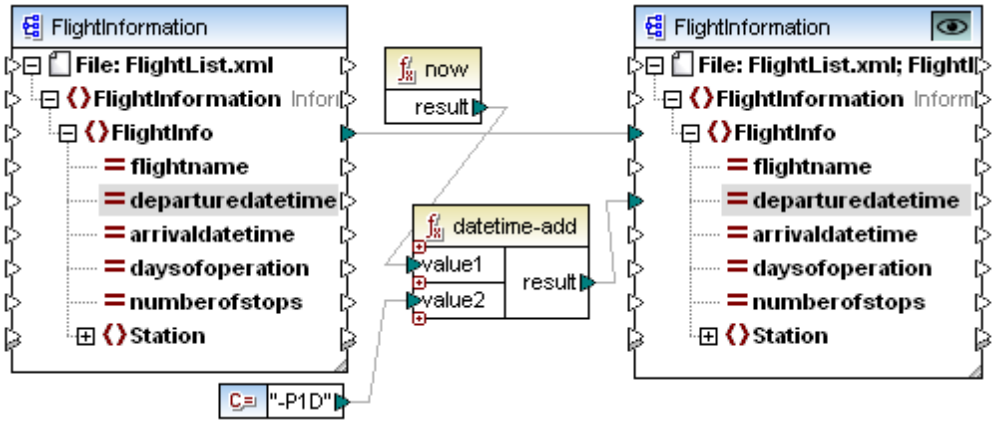
Por ejemplo, valor `datetime` del documento de instancia:

© 2018 Altova GmbH

departuredatetime="2001-12-**17**T09:30:02+05:00"

Resultado:
departuredatetime="2001-12-**27**T09:30:02+05:00"

Para extraer la fecha de ayer de la entrada dateTime:
Use la función **now** para introducir la fecha/hora actual incluido el uso horario. El período se puede convertir en negativo usando el carácter - antes del designador P (por ejemplo -P1D, menos 1 día).

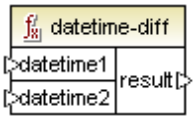


Por ejemplo, el valor datetime actual es 28 de febrero 2012, 17:19:54.748(miliseg)+01uso horario.
now="2012-02-**28**T17:19:54.748+01:00"

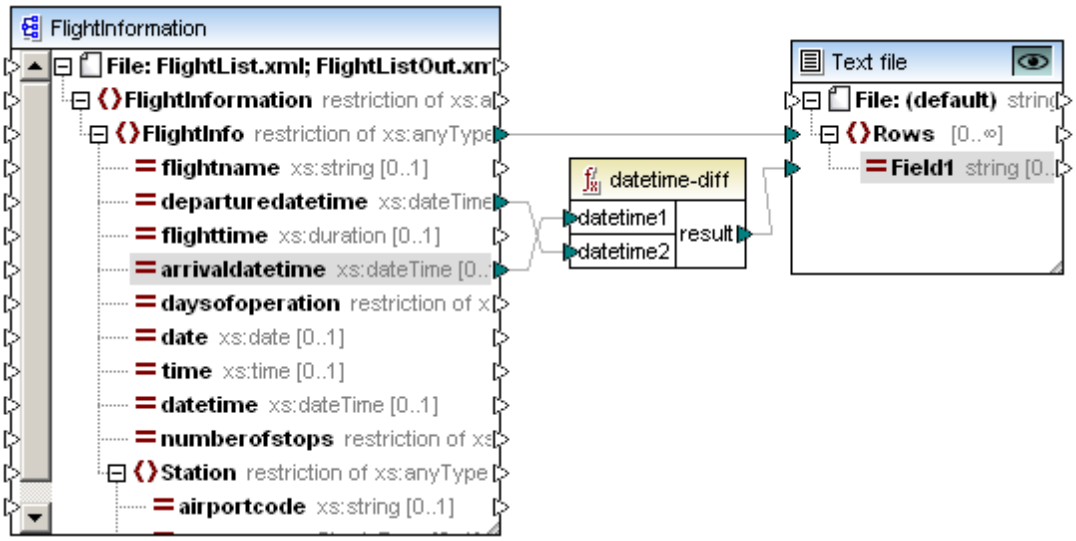
Resultado:
departuredatetime="2012-02-**27**T17:19:54.748+01:00"
es decir, 27 de febrero 2012, 17:19:54.748(miliseg)+01uso horario

8.8.13.5 datetime-diff

El resultado es la duración que se obtiene al sustraer **datetime2** (segundo argumento) a **datetime1** (primer argumento). El resultado se puede asignar a un valor string, duration o datetype.



Observe que el elemento **arrivaldatetime** se conectó al argumento **datetime1** y **departuredatetime** se conectó al argumento **datetime2**.



Por ejemplo, si queremos encontrar la diferencia en forma de duración entre las horas de salida (`departure`) y de llegada (`arrival`).

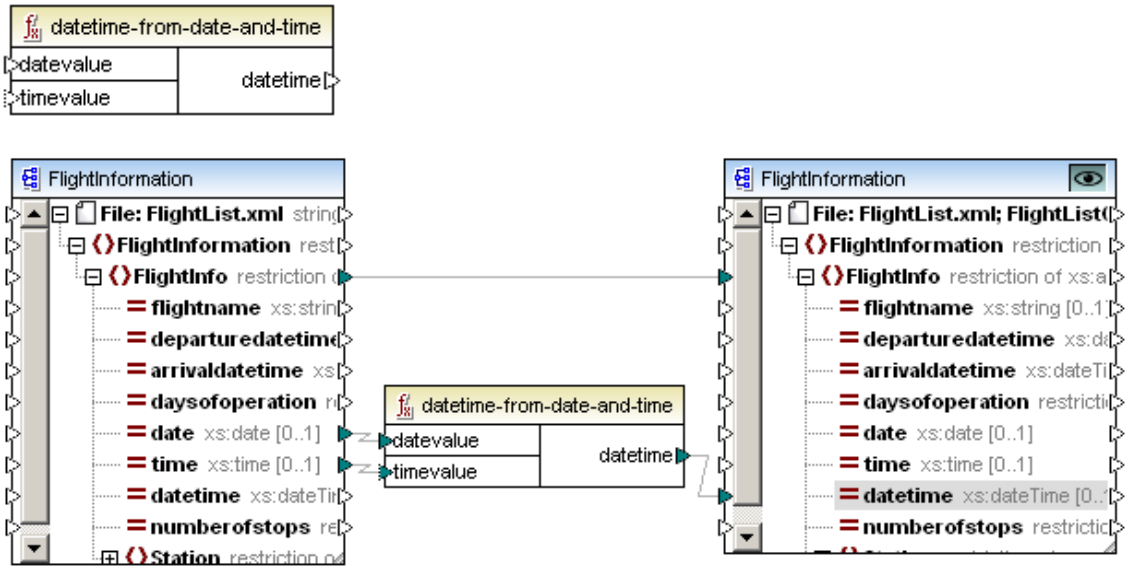
```
datetime1 arrivaldatetime="2001-12-17T19:30:02+05:00"
datetime2 departuredatetime="2001-12-17T09:30:02+05:00"
```

Resultado: la diferencia entre las dos es 10 horas.

```
result= PT10H
```

8.8.13.6 datetime-from-date-and-time

El resultado es un valor `datetime` creado a partir del argumento **`datevalue`** y el argumento **`timevalue`**. El primer argumento debe ser de tipo `xs:date` y el segundo de tipo `xs:time`. El resultado se puede asignar a un tipo de datos `string` o `dateTime`.



Ejemplo:

 © 2018 Altova GmbH

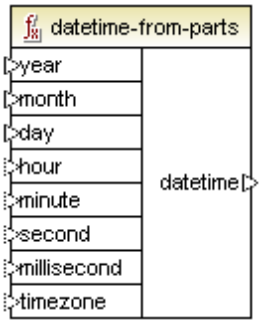
```
date="2012-06-29"
time="11:59:55"
```

Resultado:

```
dateTime="2012-06-29T11:59:55"
```

8.8.13.7 datetime-from-parts

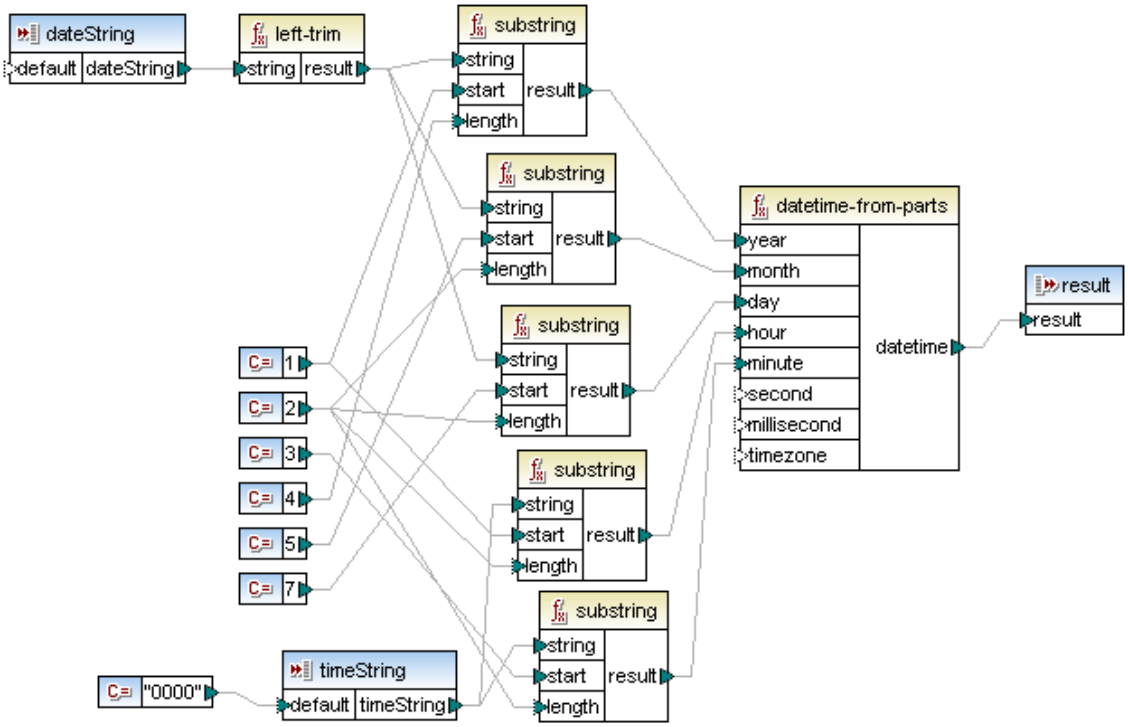
El resultado es un valor `datetime` creado uniendo cualquiera de estos argumentos: **`year`**, **`month`**, **`day`**, **`hour`**, **`minute`**, **`second`**, **`millisecond`** y **`timezone`**. Esta función normaliza automáticamente los parámetros suministrados. Por ejemplo, 32 de enero se convierte automáticamente en 1 de febrero.



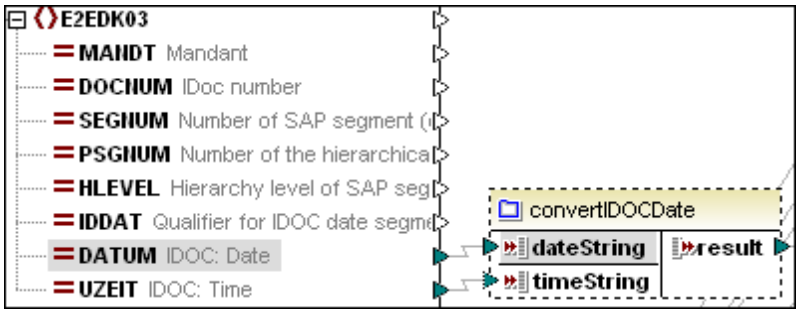
Todos los argumentos son de tipo `xs:int`, excepto el argumento **`millisecond`**, que es de tipo `xs:decimal`. El parámetro **`result`** es un valor de tipo `xs:dateTime`.

En la imagen siguiente puede ver la función definida por el usuario **`convertIDOCDate`**, disponible en la asignación de muestra `IDoc_Order.mfd` de la carpeta `...\MapForceExamples`. Esta función crea el valor **`datetime`** a partir de una cadena de entrada usando las funciones **`left-trim`** y **`substring`**.

Observe que **`year`**, **`month`** y **`day`** son parámetros obligatorios y el resto son opcionales.



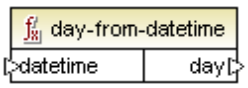
Los campos de fecha y hora vienen dados por el archivo de instancia IDOC:



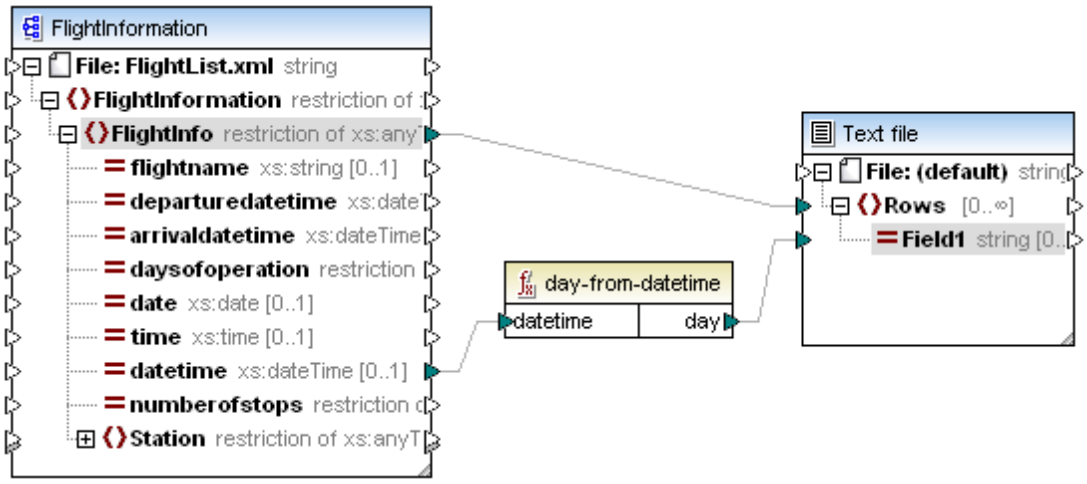

```
IDOC:Date   19990621
ICOC:Time   0930
Resultado      1999-06-21T09:30:00
```

8.8.13.8 day-from-datetime

El resultado es el día del argumento `datetime`.



 © 2018 Altova GmbH

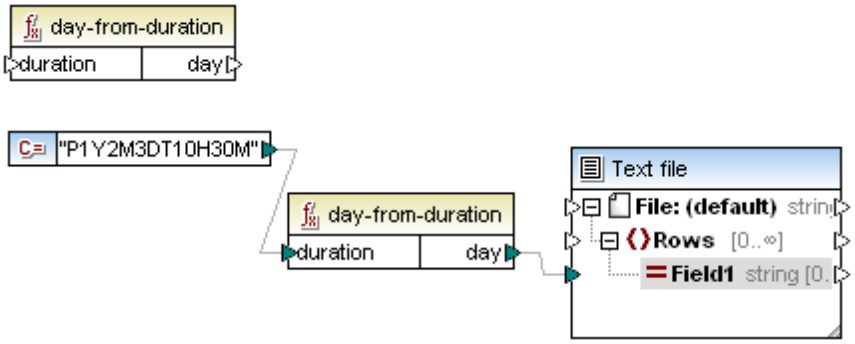


Ejemplo:
`datetime="2001-12-17T10:30:03+01:00"`

Resultado: `17`

8.8.13.9 day-from-duration

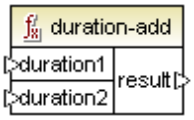
El resultado es el día del argumento `duration`.



Ejemplo:
`duration="P1Y2M3DT10H30M"`
Resultado: `3`

8.8.13.10 duration-add

El resultado es la duración obtenida al sumar la duración **`duration1`** y la duración **`duration2`**.



Ejemplo:
`duration1="P0Y0M3DT03H0M" (3 días y 3 horas)`
`duration2="P0Y0M3DT01H0M" (3 días y 1 hora)`

```
result: P6DT4H (6 días y 4 horas)
```

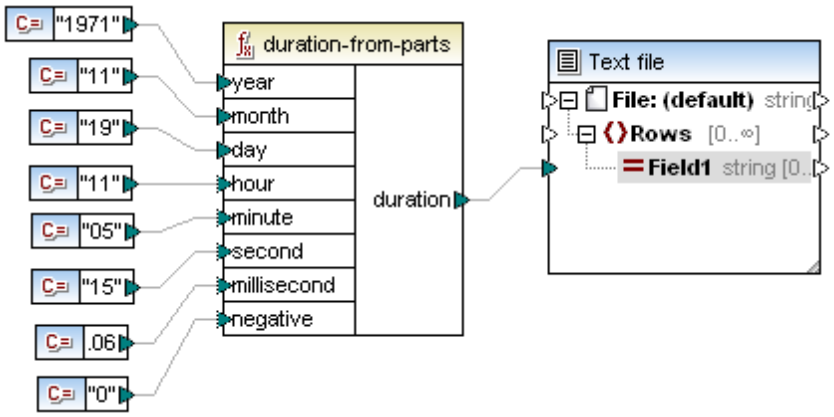
8.8.13.11 duration-from-parts

El resultado es una duración calculada al combinar estos argumentos: **year, month, day, hour, minute, second, millisecond, negative.**



Las duraciones tienen este formato `P1Y2M3DT04H05M06.07S` es decir, P de período, 1 año (Y) 2 meses (M) 3 días (D) T de designador de hora, 04 horas (H), 05 minutos (M) 06.07 segundos.milisegundos.

Todos los argumentos son de tipo `xs:int` excepto el argumento **`millisecond`**, que es de tipo `xs:decimal` y el argumento **`negative`**, que es de tipo `xs:boolean` (es decir, 1 para `true` y 0 para `false`). El parámetro **`duration`** es de tipo `xs:duration`.



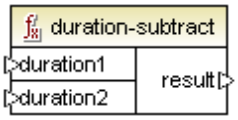
Partes: 1971 year, 11 month, 19 day, 11 hour, 05 minutes, 15.06 seconds, negative period "false".

Resultado:
`duration="P1971Y11M19DT11H5M15.00006S"`

8.8.13.12 duration-subtract

El resultado es la duración que se obtiene al sustraer **`duration2`** a **`duration1`**.

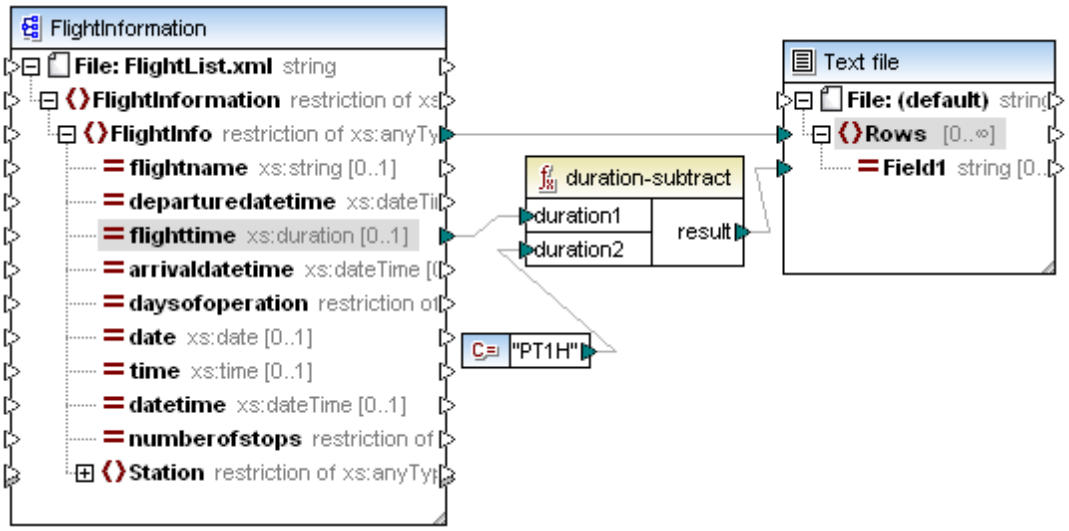
© 2018 Altova GmbH



Los argumentos **duration** deben introducirse con este formato: `P1Y2M3DT04H05M`. El período se puede convertir en negativo usando el carácter - antes del designador P (por ejemplo -P1D, menos 1 día).

P es el designador de período y es obligatorio. El resto sería:
1 año (Y), 2 meses (M), 3 días (D), designador de hora T, 04 horas (H), 05 minutos (M).

En el ejemplo de la imagen siguiente se sustrae 1 hora a la hora del vuelo (**flighttime**), es decir, `PT1H`.

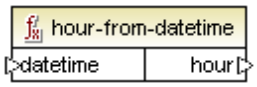


Ejemplo:
`duration1="P0Y0M0DT05H07M"`
`duration2="PT1H"`

Resultado:
`PT4H7M`

8.8.13.13 hour-from-datetime

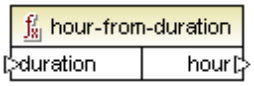
El resultado es el componente **hour** del argumento **datetime**.



Ejemplo:
`datetime="2001-12-17T09:30:02+05:00"`
`hour= 9`

8.8.13.14 hour-from-duration

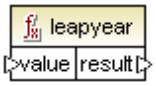
El resultado es el componente **hour** del argumento **duration**.

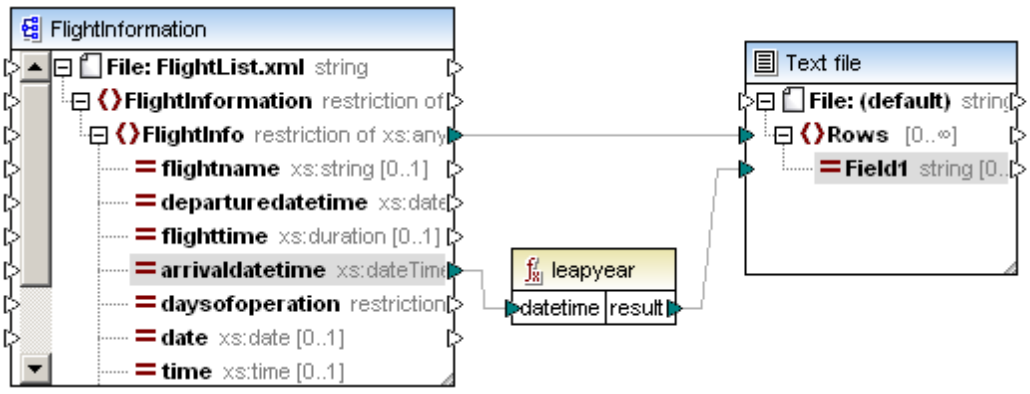


Ejemplo:
```
duration="P0Y0M0DT05H07M"
hour= 5
```

8.8.13.15 leapyear

El resultado es **true** o **false**, dependiendo de si el año del valor **dateTime** es bisiesto.

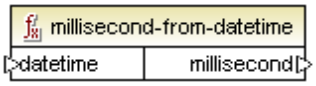




Ejemplo:
```
arrivaldatetime="2001-12-17T19:30:02+05:00"
result="false"
```

8.8.13.16 millisecond-from-datetime

El resultado es el componente **millisecond** del argumento **datetime**.

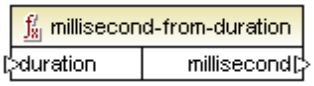


Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
millisecond= 544
```


© 2018 Altova GmbH

8.8.13.17 millisecond-from-duration

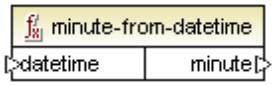
El resultado es el componente **millisecond** del argumento **duration**.



Ejemplo:
```
duration="P0Y0M0DT05H07M02.227S"
millisecond= 227
```

8.8.13.18 minute-from-datetime

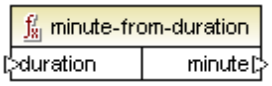
El resultado es el componente **minute** del argumento **datetime**.



Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
minute= 30
```

8.8.13.19 minute-from-duration

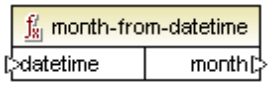
El resultado es el componente **minute** del argumento **duration**.



Ejemplo:
```
duration="P0Y0M0DT05H07M02.227S"
minute= 7
```

8.8.13.20 month-from-datetime

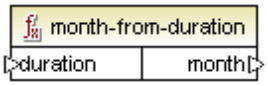
El resultado es el componente **month** del argumento **dateTime**.



Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
month= 12
```

8.8.13.21 month-from-duration

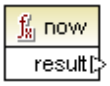
El resultado es el componente **month** del argumento **duration**.



Ejemplo:
duration="P0Y**04**M0DT05H07M02.227S"
month= 4

8.8.13.22 now

El resultado es la fechaHora actual (incluida la zona horaria).

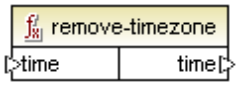


Ejemplo:
result= 2012-03-06T14:44:57.567+01:00

Para aprender a extraer la fecha del día de ayer consulte el apartado dedicado a la función core | lang | datetime-add.

8.8.13.23 remove-timezone

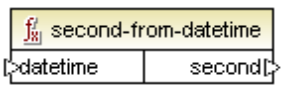
Elimina el componente de uso horario (p. ej. +5:00) del parámetro de entrada **time**.



Ejemplo:
departuredatetime="2001-12-17T09:30:02**+05:00**"
time: 2001-12-17T09:30:02

8.8.13.24 second-from-datetime

El resultado es el componente **second** del argumento **dateTime**.

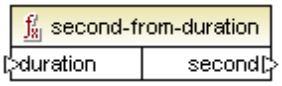


Ejemplo:
datetime="2001-12-17T09:30:**02**.544+05:00"
second= 2

© 2018 Altova GmbH

8.8.13.25 second-from-duration

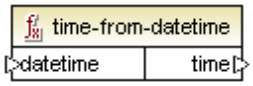
El resultado es el componente **second** del argumento **duration**.



Ejemplo:
```
duration="P0Y04M0DT05H07M02.227S"
second= 2
```

8.8.13.26 time-from-datetime

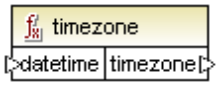
Devuelve el componente **time** del argumento **dateTime**.



Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
time= 09:31:02+05:00
```

8.8.13.27 timezone

Devuelve el uso horario (p. ej. +05:00) en relación al UTC del valor **dateTime**. El uso horario se indica en minutos.



Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
timezone= 300
```

8.8.13.28 weekday

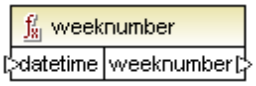
Devuelve el día de la semana indicado por el valor **dateTime**, siendo `Lunes=1` y así sucesivamente hasta `Domingo=7`.



Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
weekday= 1
```

8.8.13.29 weeknumber

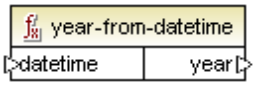
Devuelve el número de la semana del año indicado por el valor **dateTime**.



Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
weeknumber= 51
```

8.8.13.30 year-from-datetime

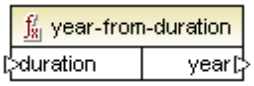
El resultado es el componente **year** del argumento **dateTime**.



Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
year= 2001
```

8.8.13.31 year-from-duration

El resultado es el componente **year** del argumento **duration**.



Ejemplo:
```
duration="P01Y04M0DT05H07M02.227S"
year= 1
```

8.8.14 lang | generator functions (generador)

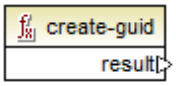
La biblioteca **lang** incluye funciones para trabajar con los lenguajes Java, C# y C++.

Las funciones de generador de **lang | generator** sirven para generar valores para campos de BD que no reciben datos de entrada del componente de origen (ya sea un componente EDI, de esquema o de BD).

© 2018 Altova GmbH

8.8.14.1 create-guid

El resultado es un identificador único global (como cadena con codificación hexadecimal) para un campo concreto.

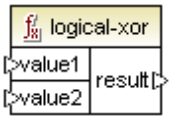


8.8.15 lang | logical functions (lógica)

La biblioteca **lang** incluye funciones para trabajar con los lenguajes Java, C# y C++.

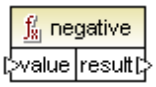
8.8.15.1 logical-xor

El resultado es `true` si **`value1`** es distinto de **`value2`**. De lo contrario, el resultado es `false`.



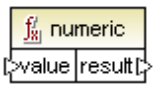
8.8.15.2 negative

El resultado es `true` si **`value`** es negativo (es decir, inferior a cero). De lo contrario, el resultado es `false`.



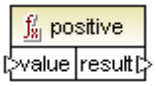
8.8.15.3 numeric

El resultado es `true` si **`value`** es un número. De lo contrario, el resultado es `false`. La entrada suele ser una cadena.



8.8.15.4 positive

El resultado es `true` si **`value`** es positivo (es decir, igual o mayor que cero). De lo contrario, el resultado es `false`.

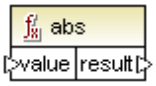


8.8.16 lang | math functions (matemáticas)

La biblioteca **lang** incluye funciones para trabajar con los lenguajes Java, C# y C++.

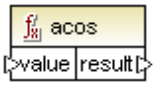
8.8.16.1 abs

El resultado es el valor absoluto del parámetro de entrada **value**.



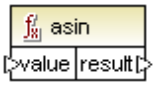
8.8.16.2 acos

El resultado es el arco coseno del parámetro de entrada **value**.



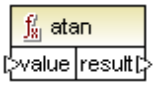
8.8.16.3 asin

El resultado es el arco seno del parámetro de entrada **value**.



8.8.16.4 atan

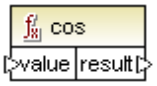
El resultado es el arco tangente del parámetro de entrada **value**.



© 2018 Altova GmbH

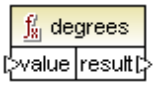
8.8.16.5 cos

El resultado es el coseno del parámetro de entrada **value**.



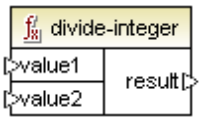
8.8.16.6 degrees

El resultado de convertir en grados el valor de **value** dado en radianes.



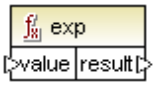
8.8.16.7 divide-integer

El resultado es el entero resultante de dividir **value1** por **value2**. Por ejemplo, **15 divide-integer 2**, el entero resultante es 7.



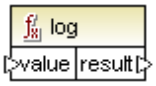
8.8.16.8 exp

El resultado es **e** (el logaritmo natural base) elevado a la potencia **value**.



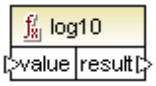
8.8.16.9 log

El resultado es el logaritmo natural de **value**.



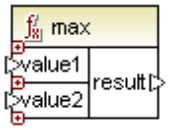
8.8.16.10 log10

El resultado es el logaritmo (base 10) de **`value`**.



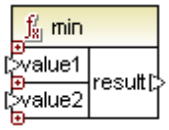
8.8.16.11 max

El resultado es el valor numéricamente mayor de **`value1`** comparado con **`value2`**.



8.8.16.12 min

El resultado es el valor numéricamente menor de **`value1`** comparado con **`value2`**.



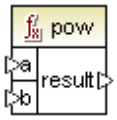
8.8.16.13 pi

El resultado es el valor de π.



8.8.16.14 pow

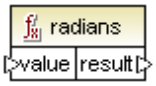
El resultado es el valor de **a** elevado a la potencia **b**.



© 2018 Altova GmbH

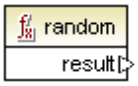
8.8.16.15 radians

El resultado de convertir en radianes el valor de **value** dado en grados.



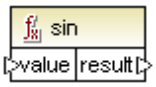
8.8.16.16 random

El resultado es un valor pseudodaleatorio comprendido entre 0.0 y 1.0.



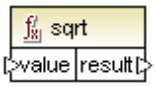
8.8.16.17 sin

El resultado es el seno de **value**.



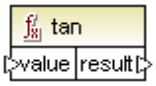
8.8.16.18 sqrt

El resultado es la raíz cuadrada de **value**.



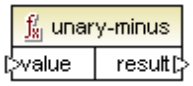
8.8.16.19 tan

El resultado es la tangente de **value**.



8.8.16.20 unary-minus

El resultado es la negación del valor con signo de **value**. Por ejemplo, +3 da -3 como resultado. -3 da 3 como resultado.

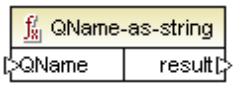


8.8.17 lang | QName functions (QName)

La biblioteca **lang** incluye funciones para trabajar con los lenguajes Java, C# y C++.

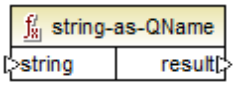
8.8.17.1 QName-as-string

El resultado es la representación de cadena única del QName.



8.8.17.2 string-as-QName

Convierte la representación de cadena de un QName en un QName.

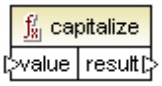


8.8.18 lang | string functions (cadena)

La biblioteca **lang** incluye funciones para trabajar con los lenguajes Java, C# y C++.

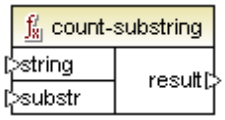
8.8.18.1 capitalize

El resultado es la cadena de entrada **`value`** después de poner en mayúsculas la primera letra de cada palabra.



8.8.18.2 count-substring

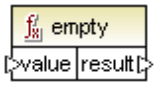
El resultado es el número de veces que la cadena **`substr`** aparece en el parámetro **`string`**.



 © 2018 Altova GmbH

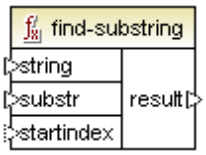
8.8.18.3 empty

El resultado es **`true`** si la cadena de entrada **`value`** está vacía. De lo contrario, el resultado es **`false`**.



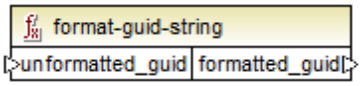
8.8.18.4 find-substring

Devuelve la posición de la primera aparición de la cadena **`substr`** dentro de la cadena **`string`**, a partir de la posición **`startindex`**. El primer carácter tiene la posición 1. Si no se encuentra la cadena **`dubstr`**, el resultado es 0.



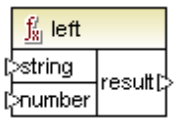
8.8.18.5 format-guid-string

El resultado es una cadena GUID de formato correcto **`formatted_guid`**, usando como cadena de entrada **`unformatted_guid`**, que se puede usar en campos de BD. Consulte también la función create-guid de la biblioteca de funciones **lang | generator functions**.



8.8.18.6 left

El resultado es una cadena que contiene los primeros **number** caracteres de **`string`**.

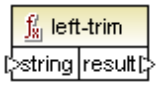


Ejemplo:

```
string="Esto es una frase"
number=4
result=Esto
```

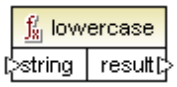
8.8.18.7 left-trim

El resultado es la cadena de entrada **`string`** después de quitarle todos los espacios en blanco iniciales.



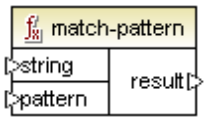
8.8.18.8 lowercase

El resultado es la cadena de entrada **`string`** en minúsculas. Para los caracteres Unicode se usan los caracteres correspondientes en minúsculas.



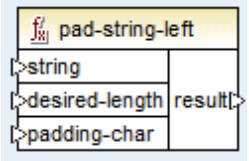
8.8.18.9 match-pattern

El resultado es **`true`** si la cadena de entrada **`string`** coincide con la expresión regular definida por el parámetro **`pattern`**. De lo contrario, el resultado es **`false`**. La sintaxis de la expresión regular depende del lenguaje de destino seleccionado (*véase* *Expresiones regulares*).



8.8.18.10 pad-string-left

Devuelve una cadena que se rellena a la izquierda con determinado carácter y hasta determinada longitud. La longitud que debe tener la cadena y el carácter de relleno se dan en forma de argumento.

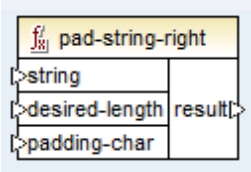


string	Especifica la cadena de entrada.
desired-length	Define la longitud que debe tener la cadena después de aplicar el relleno.
padding-char	Define el carácter que se debe usar como carácter de relleno.

 © 2018 Altova GmbH

8.8.18.11 pad-string-right

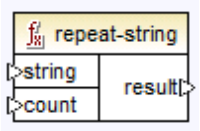
Devuelve una cadena que se rellena a la izquierda con determinado carácter y hasta determinada longitud. La longitud que debe tener la cadena y el carácter de relleno se dan en forma de argumento.



string	Especifica la cadena de entrada.
desired-length	Define la longitud que debe tener la cadena después de aplicar el relleno.
padding-char	Define el carácter que se debe usar como carácter de relleno.

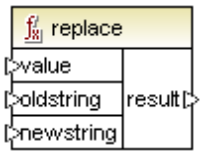
8.8.18.12 repeat-string

Repite la cadena dada tantas veces como indique el argumento **count**.



8.8.18.13 replace

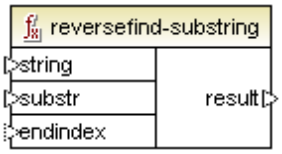
El resultado es una cadena nueva formada por la cadena de entrada **`value`**, después de sustituir cada instancia de **`oldstring`** por **`newstring`**.



Puede ver un ejemplo de uso en el apartado Reemplazar caracteres especiales.

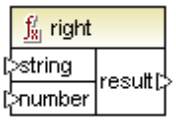
8.8.18.14 reversefind-substring

Devuelve la posición de la primera aparición de **`substr`** dentro de **`string`**, a partir de la posición **`endindex`**, es decir, de derecha a izquierda. El primer carácter tiene la posición 1. Si no se encuentra la cadena **`substr`**, el resultado es 0.



8.8.18.15 right

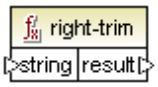
El resultado es una cadena que contiene los últimos **n** caracteres de la cadena **string**.



Ejemplo:
```
string="Esto es una frase"
number=5
result=frase
```

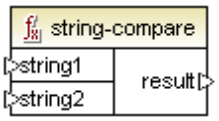
8.8.18.16 right-trim

El resultado es la cadena de entrada **string** después de eliminar todos los caracteres de espacio en blanco finales.



8.8.18.17 string-compare

Devuelve el resultado de comparar la cadena **string1** con la cadena **string2** y distinguiendo entre mayúsculas y minúsculas. Si **string1=string2**, el resultado es 0.



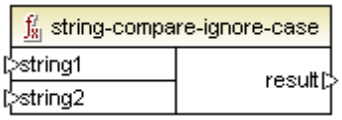
Si **string1** es menor que **string2**, entonces el resultado es < 0.

Si **string1** es mayor que **string2**, entonces el resultado es > 0.

8.8.18.18 string-compare-ignore-case

Devuelve el resultado de comparar la cadena **string1** con la cadena **string2** sin distinguir entre mayúsculas y minúsculas. Si **string1=string2**, el resultado es 0.

 © 2018 Altova GmbH

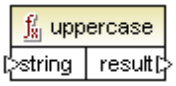


Si **`string1`** es menor que **`string2`**, entonces el resultado es < 0.

Si **`string1`** es mayor que **`string2`**, entonces el resultado es > 0.

8.8.18.19 uppercase

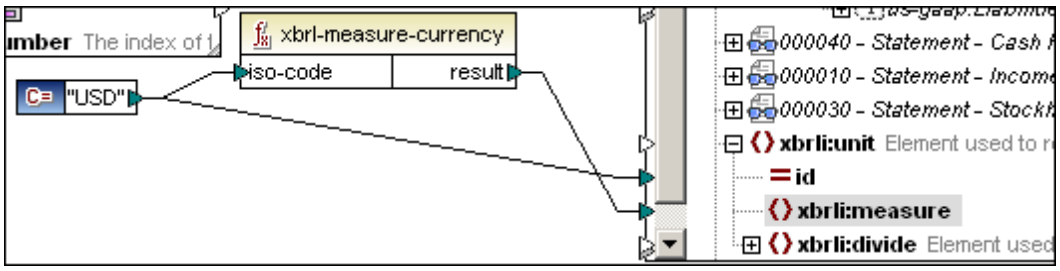
El resultado es la cadena de entrada **`string`** después de pasarla toda a mayúsculas. Para los caracteres Unicode se usan los caracteres correspondientes en mayúsculas.



8.8.19 xbrl

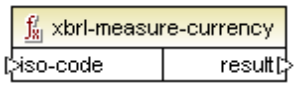
La biblioteca **xbrl** contiene funciones que convierten datos en el formato QName necesario para los archivos de instancia/taxonomía XBRL.

Los elementos secundarios `xbrl:ID` y `xbrl:measure` del elemento `xbrl:unit` son obligatorios en los archivos XBRL de instancia y, por tanto, deben tener asignaciones de datos. De lo contrario, la asignación no es válida.



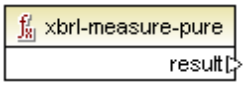
8.8.19.1 xbrl-measure-currency

El resultado es el QName para elementos de tipo `monetaryItemType` del código de divisas ISO 4217.



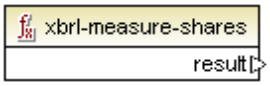
8.8.19.2 xbrl-measure-pure

El resultado es el QName para elementos de tasas, porcentajes y relaciones.



8.8.19.3 xbrl-measure-shares

El resultado es el QName para elementos de tipo `sharesItemType`.

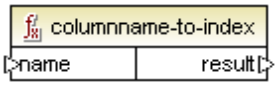


8.8.20 xlsx

La biblioteca **xlsx** ofrece funciones para convertir datos de fecha y hora Excel.

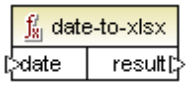
8.8.20.1 columnname-to-index

Devuelve el índice de la columna cuyo nombre coincide con el valor del parámetro **`name`** (la columna 1 se llama "A").



8.8.20.2 date-to-xlsx

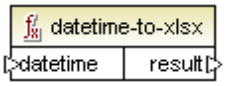
Devuelve la representación Excel del valor de *fecha, hora, fecha/hora* que se extrae de la entrada **`date`**.



© 2018 Altova GmbH

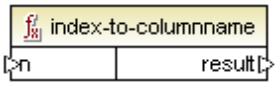
8.8.20.3 datetime-to-xslx

Devuelve la representación Excel del valor de *fecha/hora* que se extrae de la entrada **`datetime`**.



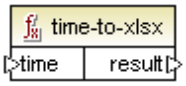
8.8.20.4 index-to-columnname

Devuelve el nombre de la columna número **`n`** (la columna A es la columna número 1).



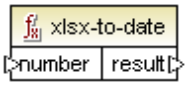
8.8.20.5 time-to-xlsx

Devuelve la representación Excel del valor *hora* que se extrae de la entrada **`time`**.



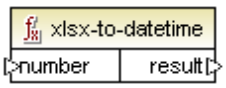
8.8.20.6 xlsx-to-date

Devuelve el valor *fecha* que se extrae de la representación Excel.



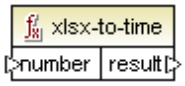
8.8.20.7 xlsx-to-datetime

Devuelve el valor *fecha/hora* que se extrae de la representación Excel.



8.8.20.8 xlsx-to-time

Devuelve el valor *hora* que se extrae de la representación Excel.



8.8.21 xpath2 | accessors (descriptores de acceso)

La biblioteca **xpath2** contiene funciones para trabajar con los lenguajes XSLT2 y XQuery.

8.8.21.1 base-uri

La función **base-uri** toma un argumento de nodo como entrada y devuelve el URI del recurso XML que contiene el nodo. El resultado es de tipo `xs:string`. MapForce devuelve un error si no se suministra un nodo de entrada.

8.8.21.2 node-name

La función node-name toma un nodo como argumento de entrada y devuelve su QName. Cuando el QName se representa como una cadena de texto, tiene el formato `prefijo:nombrelocal` (si el nodo tiene un prefijo) o el formato `nombrelocal` (si el nodo no tiene prefijo). Para obtener el URI de espacio de nombres de un nodo, utilice la función **namespace-URI-from-QName** (de la biblioteca de funciones **xpath2 | qname-related**).

8.8.21.3 string

La función **string** funciona igual que el constructor `xs:string`: convierte su argumento en `xs:string`.

Cuando el argumento de entrada es un valor de un tipo atómico (por ejemplo `xs:decimal`), este valor atómico se convierte en un valor de tipo `xs:string`. Si el argumento de entrada es un nodo, se extrae el valor de cadena del nodo. (El valor de cadena de un nodo es una concatenación de los valores de los descendientes del nodo.)

8.8.22 xpath2 | anyURI functions

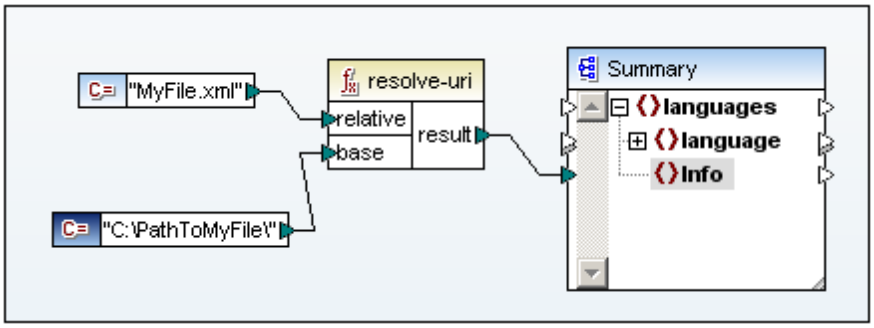
La biblioteca **xpath2** contiene funciones para trabajar con los lenguajes XSLT2 y XQuery.

8.8.22.1 resolve-uri

La función **resolve-uri** toma un URI como primer argumento (tipo de datos `xs:string`) y lo resuelve a partir del URI del segundo argumento (tipo de datos `xs:string`).

 © 2018 Altova GmbH

El resultado (tipo de datos `xs:string`) es un URI combinado. De este modo se puede convertir un URI relativo (primer argumento) en un URI absoluto, resolviéndolo a partir de un URI base.



En el ejemplo de esta imagen el primer argumento aporta el URI relativo y el segundo aporta el URI base. El URI resuelto será una concatenación del URI base y el URI relativo. Es decir `C:\PathtoMyFile\MyFile.xml`.

Nota: ambos argumentos son de tipo de datos `xs:string` y para el proceso de combinación ambos datos de entrada se tratan como cadenas de texto. Por tanto, no se puede comprobar si los recursos identificados por estos URI existen o no. MapForce devuelve un error si falta el segundo argumento.

8.8.23 xpath2 | boolean functions (booleanas)

La biblioteca **xpath2** contiene funciones para trabajar con los lenguajes XSLT2 y XQuery.

Las funciones booleanas de la biblioteca **xpath2 | boolean** son las funciones **`true`** y **`false`**. Estas funciones no toman ningún argumento y devuelven los valores booleanos de constante `true` y `false` respectivamente. Estas funciones se pueden utilizar cuando se necesite un valor booleano de constante.

8.8.23.1 false

Devuelve el valor booleano `false`.

8.8.23.2 true

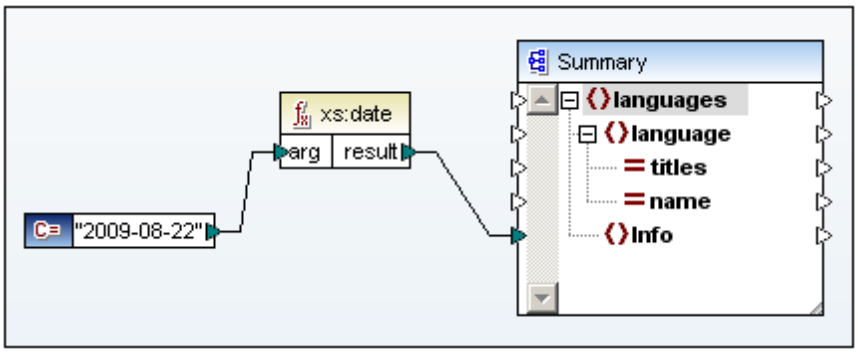
Devuelve el valor booleano `true`.

8.8.24 xpath2 | constructors (constructores)

La biblioteca **xpath2** contiene funciones para trabajar con los lenguajes XSLT2 y XQuery.

Las funciones constructores de la biblioteca de funciones **xpath2** construyen ciertos tipos de datos a partir del texto de entrada. Por lo general, el formato léxico del texto de entrada debe ser el formato esperado del tipo de datos que se debe construir. De lo contrario, la transformación no se puede realizar.

Por ejemplo, si desea construir un tipo de datos `xs:date`, utilice la función constructor `xs:date`. El texto de entrada debe tener el formato léxico del tipo de datos `xs:date`, es decir: `AAAA-MM-DD` (*imagen siguiente*).



En la imagen anterior, observe que se utilizó la constante `2009-08-22` como argumento de entrada de la función. La entrada también podría obtenerse de un nodo de un documento de origen.

La función `xs:date` devuelve el texto de entrada (`2009-08-22`), que es de tipo de datos `xs:string` (especificado en la constante), como tipo de datos `xs:date` de salida.

Al pasar el puntero del ratón encima del argumento de entrada de la función aparece el tipo de datos esperado para el argumento.

8.8.25 xpath2 | context functions (contexto)

La biblioteca **xpath2** contiene funciones para trabajar con los lenguajes XSLT2 y XQuery.

Las funciones de contexto de la biblioteca **xpath2 | context** ofrecen la hora y la fecha, la intercalación predeterminada utilizada por el procesador y el tamaño de la secuencia actual y la posición del nodo actual.

8.8.25.1 current-date

Devuelve la fecha actual (`xs:date`) del reloj del sistema.

8.8.25.2 current-dateTime

Devuelve la fecha y la hora actuales (`xs:dateTime`) del reloj del sistema.

8.8.25.3 current-time

Devuelve la hora actual (`xs:time`) del reloj del sistema.

© 2018 Altova GmbH

8.8.25.4 default-collation

La función **`default-collation`** no tiene argumentos y devuelve la intercalación predeterminada, es decir, la intercalación utilizada cuando no se especifica una intercalación para una función que la necesita.

El motor XSLT 2.0 de Altova solamente es compatible con la intercalación de punto de código Unicode. Las comparaciones (incluidas las funciones **`fn:max`** y **`fn:min`**) se basan en esta intercalación.

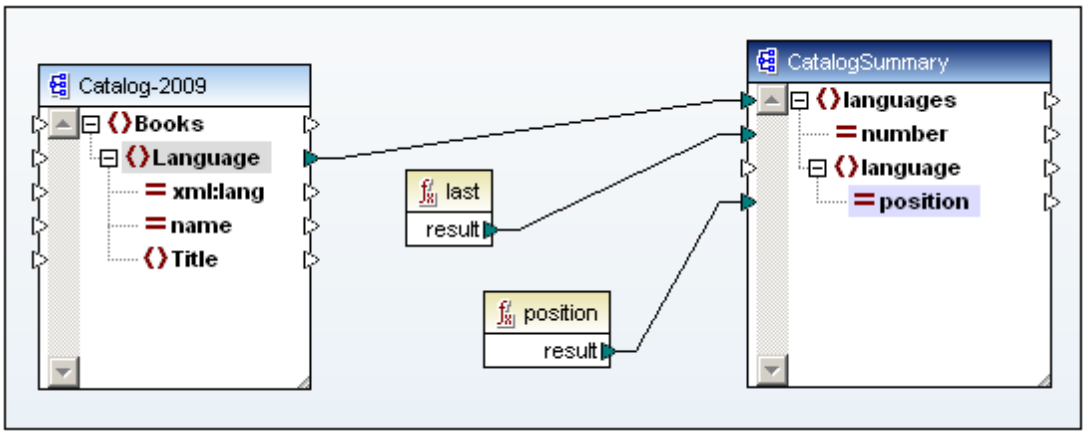
8.8.25.5 implicit-timezone

Devuelve el valor de la propiedad de zona horaria implícita "*implicit timezone*" a partir del contexto de evaluación.

8.8.25.6 last

Las funciones **`last`** y **`position`** tampoco tienen argumentos. La función **`last`** devuelve la posición del último nodo del conjunto de nodos de contexto. La función **`position`** devuelve la posición del nodo actual del conjunto de nodos que se está procesando.

El conjunto de nodos de contexto de los nodos a los que se dirigen las funciones es el conjunto de nodos al que afectarán las funciones. Por ejemplo, en la imagen siguiente, el conjunto de nodos de los elementos `Language` es el conjunto de nodos de contexto para las funciones **`last`** y **`position`**.



En el ejemplo, la función **`last`** devuelve la posición del último nodo del conjunto de nodos de contexto (el conjunto de nodos de los elementos `Language`) como valor del atributo `number`. Este valor también es el tamaño del conjunto de nodos, puesto que indica el número de nodos que contiene el conjunto.

La función **`position`** devuelve la posición del nodo `Language` que se está procesando en ese momento. La posición de cada nodo `Language` dentro del conjunto de nodos de los elementos `Language` se escribe en el atributo `language/@position`.

Recomendamos utilizar las funciones **position** y **count** de la biblioteca de funciones **core**.

8.8.26 xpath2 | durations, date, time functions (duración, fecha y hora)

La biblioteca **xpath2** contiene funciones para trabajar con los lenguajes XSLT2 y XQuery.

Las funciones de duración, fecha y hora de la biblioteca **xpath2** permiten ajustar las fechas y horas al uso horario, extraer componentes de los datos fecha/hora y sustraer una unidad fecha/hora a otra.

Funciones Adjust-to-Timezone

Cada una de estas funciones toma un valor `date`, `time` o `dateTime` como primer argumento y ajusta los datos de entrada añadiendo, eliminando o modificando el componente de uso horario en función del valor del segundo argumento.

Cuando el primer argumento no contiene datos de uso horario (por ejemplo, la fecha `2009-01` o la hora `14:00:00`), hay tres posibilidades:

- Si está presente el argumento `timezone` (el segundo argumento de la función), el resultado contendrá el uso horario especificado en el segundo argumento.
- Si falta el argumento `timezone` (el segundo argumento de la función), el resultado contendrá el uso horario implícito, es decir, el del sistema.
- Si el argumento `timezone` (el segundo argumento de la función) está vacío, el resultado no contendrá el uso horario.

Cuando el primer argumento contiene datos de uso horario (por ejemplo, la fecha `2009-01-01 +01:00` o la hora `14:00:00+01:00`), hay tres posibilidades:

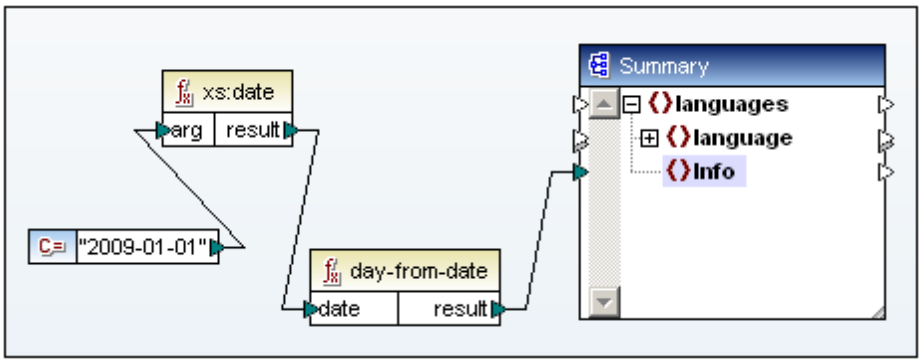
- Si está presente el argumento `timezone` (el segundo argumento de la función), el resultado contendrá el uso horario especificado en el segundo argumento. El uso horario original se sustituye con el uso horario del segundo argumento.
- Si falta el argumento `timezone` (el segundo argumento de la función), el resultado contendrá el uso horario implícito, es decir, el del sistema. El uso horario original se sustituye con el uso horario del sistema.
- Si el argumento `timezone` (el segundo argumento de la función) está vacío, el resultado no contendrá el uso horario.

Funciones From

Cada función **From** extrae un componente de los (i) datos de fecha u hora o de los (ii) datos de duración. Los resultados son de tipo de datos `xs:decimal`.

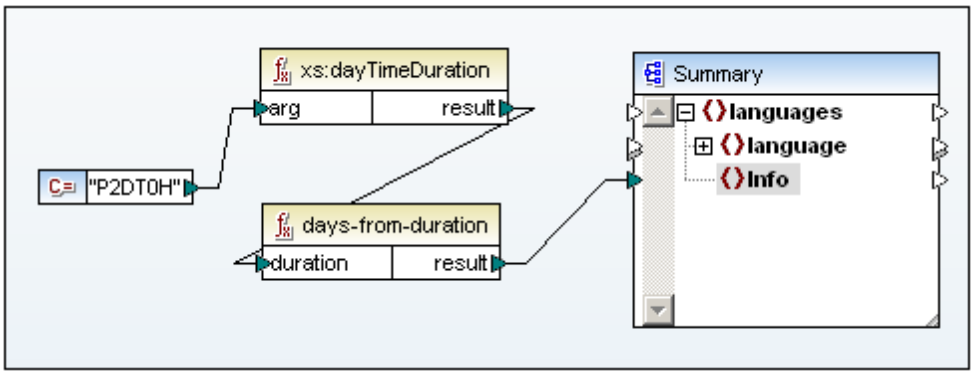
Por ejemplo, si queremos extraer un componente de los datos de fecha u hora, podemos usar la función **day-from-date** (*imagen siguiente*).

© 2018 Altova GmbH



El argumento de entrada es una fecha (`2009-01-01`) de tipo `xs:date`. La función **`day-from-date`** extrae el componente día de la fecha (`1`) con tipo de datos `xs:decimal`.

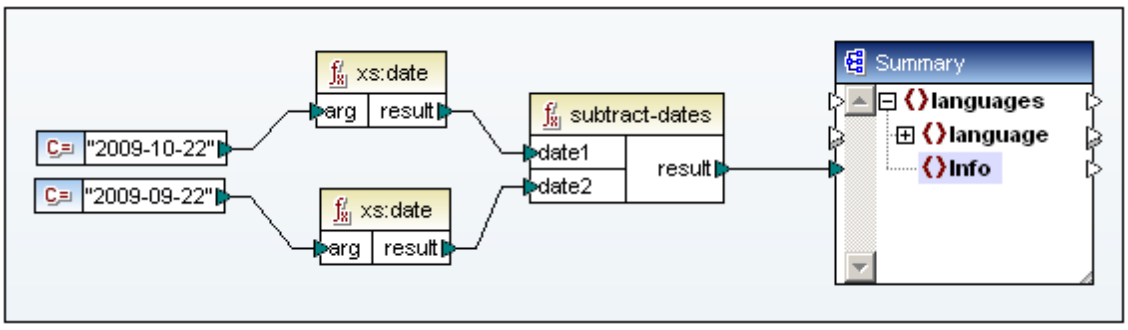
Para extraer los componentes hora de las duraciones es necesario indicar la duración como `xs:yearMonthDuration` (para extraer años y meses) o como `xs:dayTimeDuration` (para extraer días, horas, minutos y segundos). El resultado será de tipo `xs:decimal`. En el ejemplo siguiente, los datos de entrada de la función **`days-from-duration`** es el tipo de datos `dayTimeDuration` de `P2DT0H`. El resultado es el `2`, de tipo de datos `xs:decimal`.



Funciones Subtract

Hay tres funciones de sustracción con las que puede sustraer un valor de hora a otro y obtener un valor de duración como resultado. Se trata de las funciones: **`subtract-dates`**, **`subtract-times`** y **`subtract-dateTimes`**.

En el ejemplo de la imagen se usa la función **`subtract-dates`** para sustraer una fecha a otra (`2009-10-22` menos `2009-09-22`). El resultado es `P30D`, de tipo de datos `dayTimeDuration`.

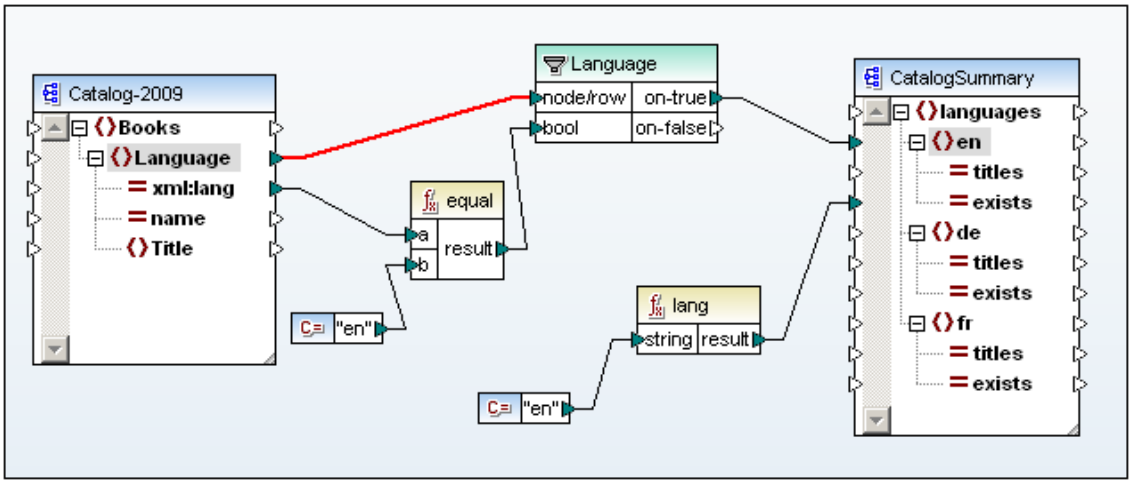


8.8.27 xpath2 | node functions (nodo)

La biblioteca **xpath2 | node** incluye estas funciones de nodo:

lang

Esta función toma como argumento una cadena que identifica un código de idioma (como `en`). La función devuelve `true` o `false` dependiendo de si el nodo de contexto tiene un atributo `xml:lang` con el valor especificado por el argumento de la función.



Por ejemplo, en la imagen anterior:

1. En el esquema de origen el elemento `Language` tiene un atributo `xml:lang`.
2. Los nodos `Language` se filtran para que se procesen solamente los nodos `Language` cuyo atributo `xml:lang` tenga el valor `en` (la prueba de filtro se especifica en la función **`equal`**).
3. El nodo `Language` es el nodo de contexto donde se crea el elemento `en` en el documento de salida.
4. El resultado de la función **`lang`** (`true` o `false`) se envía al atributo `en/@exists` de salida. El argumento de la función viene dado por la constante `en`. La función **`lang`** comprueba si el nodo de contexto en este punto (el elemento `Language`) tiene un atributo `xml:lang` con valor `en` (el argumento de la función). Si lo tiene, la función devuelve `true`. Si no, devuelve `false`.

local-name, name, namespace-uri

Las funciones **`local-name`**, **`name`** y **`namespace-uri`** devuelven el nombre local, el nombre y el URI de espacio de nombres respectivamente del nodo de entrada. Por ejemplo, en el caso del nodo `altova:Products`, el nombre local es `Products`, el nombre es `altova:Products` y el URI de espacio de nombres es el URI del espacio de nombres al que está enlazado el prefijo `altova:` (por ejemplo `https://www.altova.com/mapforce`).

Cada una de estas tres funciones tiene dos variantes:

- Sin argumento: la función se aplica al nodo de contexto (*imagen anterior*).
- Con argumento (que debe ser un nodo): la función se aplica al nodo especificado.

© 2018 Altova GmbH

El resultado de estas tres funciones (en todas sus variantes) es una cadena de texto.

number

Esta función convierte la cadena de entrada en un número. También convierte valores de entrada booleanos en un número.

La función **number** toma un nodo como entrada, atomiza el nodo (es decir, extrae su contenido), convierte el valor en un decimal y devuelve el valor convertido. Los tipos que se pueden convertir en números son los tipos `boolean`, `string` y otros tipos numéricos. Los valores de entrada no numéricos (como cadenas no numéricas) dan como resultado `NaN` (no es un número).

Esta función tiene dos variantes:

- Sin argumento: la función se aplica al nodo de contexto.
- Con argumento (que debe ser un nodo): la función se aplica al nodo especificado.

8.8.28 xpath2 | numeric functions (numéricas)

La biblioteca de funciones **xpath2 | numeric** ofrece estas funciones numéricas:

abs

Esta función toma un valor numérico como entrada y devuelve su valor absoluto como `decimal`. Por ejemplo, si el argumento de entrada es `-2` o `+2`, la función devuelve `2`.

round-half-to-even

Esta función redondea el número dado (primer argumento) al grado de precisión (número de decimales) dado en el segundo argumento, que es opcional. Por ejemplo, si el primer argumento es `2.141567` y el segundo argumento es `3`, el primer argumento (el número) se redondea con tres decimales y el resultado sería `2.141`. Si no se especifica la precisión (si falta el segundo argumento), el número se redondea con cero decimales, es decir, se convierte en un número entero.

La parte "even" del nombre de la función hace referencia a que se redondeo hasta un número par cuando un dígito del número dado está entre dos valores. Por ejemplo `round-half-to-even(3.475, 2)` devolvería el valor `3.48`.

8.8.29 xpath2 | string functions (cadena)

La biblioteca **xpath2 | string** ofrece estas funciones de cadena:

compare

Esta función toma dos cadenas como argumentos y las compara alfabéticamente. Si la cadena `String-1` es alfabéticamente menor que la cadena *String-2* (por ejemplo: `A y B`), el resultado de la función es -1. Si las dos cadenas son iguales (por ejemplo, `A` y `A`), la función devuelve `0`. Si la cadena `String-1` es mayor que `String-2` (por ejemplo, `B y A`), la función devuelve +1.

Esta función tiene una variante que sirve para elegir la intercalación que se debe utilizar para comparar las cadenas. Cuando no se indica la intercalación, se utiliza la intercalación predeterminada (la intercalación de punto de código Unicode).

ends-with

Esta función comprueba si la cadena `String-1` termina con la cadena `String-2`. Si es así, devuelve el valor `true`. Si no, devuelve `false`.

Esta función tiene una variante que sirve para elegir la intercalación que se debe utilizar para comparar las cadenas. Cuando no se indica la intercalación, se utiliza la intercalación predeterminada (la intercalación de punto de código Unicode). Los motores de Altova solamente son compatibles con la intercalación de punto de código Unicode.

escape-uri

Esta función toma un URI como entrada para el primer argumento de cadena y aplica a la cadena las normas de escape de URI RFC 2396. El segundo argumento booleano (`escape-reserved`) debe ser `true()` si los caracteres con significado reservado en los URI se deben escapar (por ejemplo "+" o "/").

Ejemplos:

`escape-uri("My A+B.doc", true())` daría como resultado `My%20A%2B.doc`
`escape-uri("My A+B.doc", false())`daría como resultado `My%20A+B.doc`

lower-case

Esta función toma una cadena como argumento y pone en minúsculas todos los caracteres que estén en mayúsculas.

matches

Esta función comprueba si la cadena de entrada (primer argumento) coincide con una expresión regular (segundo argumento). La sintaxis de las expresiones regulares debe ser la sintaxis definida para la faceta `pattern` de XML Schema. La función devuelve `true` si la cadena coincide con la expresión regular. De lo contrario, el resultado es `false`.

La función toma un argumento opcional llamado `flags`. Este argumento `flags` puede tener cuatro valores (`i`, `m`, `s`, `x`) y se pueden usar varios a la vez (p. ej. `imx`) Si se deja vacío, se usan los valores predeterminados.

El significado de estos argumentos es:

- `i` Utilizar el modo de no distinción entre mayúsculas y minúsculas. El modo predeterminado es el modo de distinción entre mayúsculas y minúsculas.
- `m` Usar el modo multilínea, que entiende que la cadena de entrada tiene varias líneas, cada una de ellas separada por un carácter de línea nueva (`x0a`). Los metacaracteres `^` y `$` indican el comienzo y el final de cada línea. El modo predeterminado es el modo de cadena, en el que la cadena comienza y termina con los metacaracteres `^` y `$`.
- `s` Usar el modo *dot-all*. El modo predeterminado es el modo *not-dot-all*, en el que el metacarácter "." coincide con todos los caracteres excepto el carácter de línea nueva

 © 2018 Altova GmbH

(`x0a`). En el modo *dot-all*, el punto **"."** también coincide con el carácter de línea nueva.

`x` Omitir espacios en blanco. Los caracteres de espacio en blanco no se ignoran por defecto.

normalize-unicode

Esta función normaliza la cadena de entrada (primer argumento) según las reglas de normalización especificadas (segundo argumento). Son compatibles los formatos de normalización NFC, NFD, NFKC y NFKD.

replace

Esta función toma la cadena del primer argumento como entrada, busca en ella repeticiones de la expresión regular (segundo argumento) y reemplaza los resultados encontrados con la cadena del tercer argumento.

Las reglas de coincidencia son las indicadas para el atributo `matches`. La función también toma el argumento opcional `flags`. Los valores de este argumento son los mismos que los de la función **`matches`** (*ver más arriba*).

starts-with

Esta función comprueba si la cadena `String-1` empieza con la cadena `String-2`. Si es así, la función devuelve `true`. De lo contrario devuelve `false`.

Esta función tiene una variante que sirve para elegir la intercalación que se debe utilizar para comparar las cadenas. Cuando no se indica la intercalación, se utiliza la intercalación predeterminada (la intercalación de punto de código Unicode). Los motores de Altova solamente son compatibles con la intercalación de punto de código Unicode.

substring-after

Esta función devuelve la parte de la cadena `String-1` (primer argumento) que aparece después de la cadena de prueba `String-2` (segundo argumento). La función toma un tercer argumento opcional que especifica la intercalación que se debe usar para la comparación de las cadenas. Cuando no se indica la intercalación, se utiliza la intercalación predeterminada (la intercalación de punto de código Unicode). Los motores de Altova solamente son compatibles con la intercalación de punto de código Unicode.

substring-before

Esta función devuelve la parte de la cadena `String-1` (primer argumento) que aparece antes de la cadena de prueba `String-2` (segundo argumento). La función toma un tercer argumento opcional que especifica la intercalación que se debe usar para la comparación de las cadenas. Cuando no se indica la intercalación, se utiliza la intercalación predeterminada (la intercalación de punto de código Unicode). Los motores de Altova solamente son compatibles con la intercalación de punto de código Unicode.

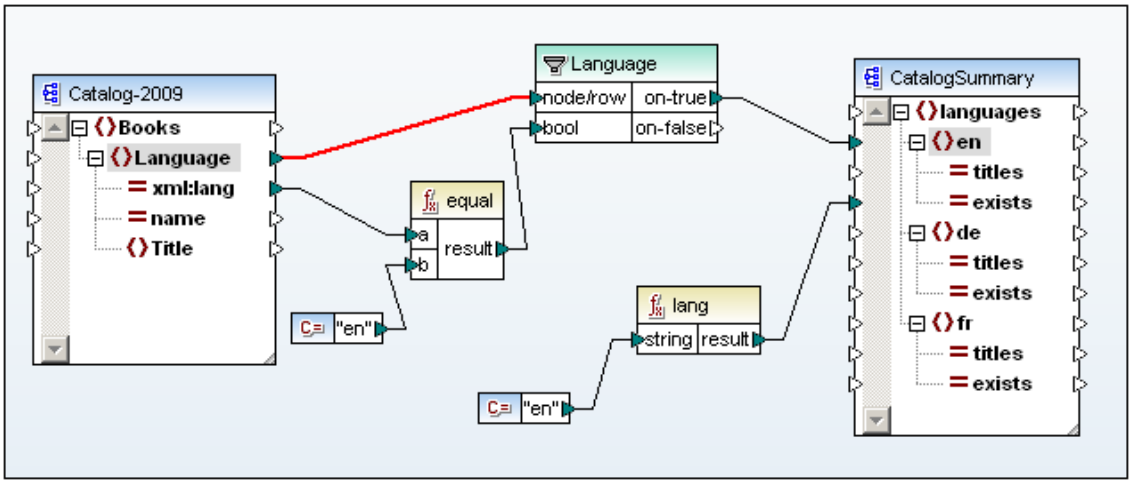
upper-case

Esta función toma una cadena y pone en mayúsculas todos los caracteres que aparezcan en minúsculas.

8.8.30 xslt | xpath functions

Las funciones de la biblioteca **xslt | xpath** son funciones de conjunto de nodos XPath 1.0. Estas funciones toman un nodo o conjunto de nodos como contexto y devuelve información sobre ellos. Estas funciones suelen tener:

- un nodo de contexto (por ejemplo, el nodo de contexto para la función **lang** de la imagen siguiente es el elemento `Language` del esquema de origen).
- un argumento de entrada (por ejemplo, el argumento de entrada para la función **lang** de la imagen siguiente es la constante `en`). Las funciones `last` y `position` no toman argumentos.



lang

Esta función toma una cadena de entrada que identifica el código de idioma (p. ej. `en`). La función devuelve `true` o `false` dependiendo de si el nodo de contexto tiene o no un atributo `xml:lang` cuyo valor coincide con el argumento de la función. En la imagen anterior, por ejemplo:

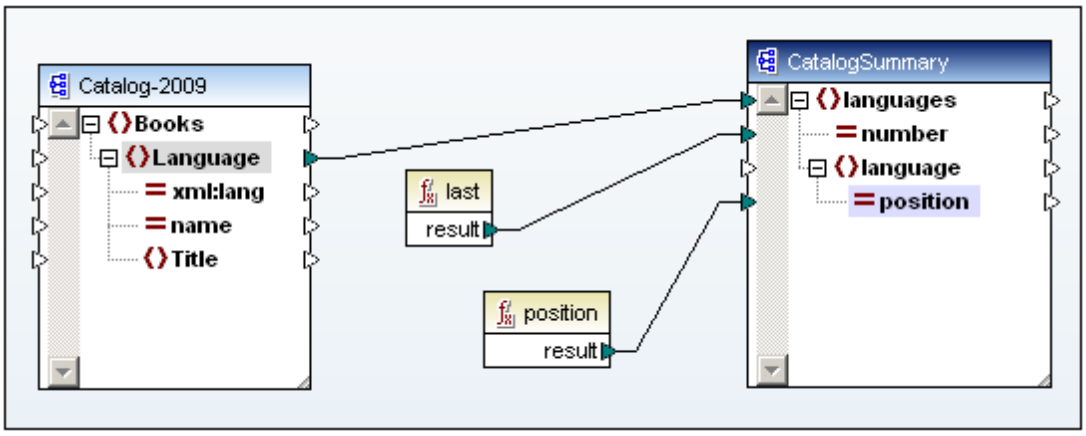
1. El elemento `Language` del esquema de origen tiene un atributo `xml:lang`.
2. Los nodos `Language` se filtran para que solamente se procesen aquellos cuyo atributo `xml:lang` tenga el valor `en` (la prueba de filtro se especifica en la función **equal**).
3. El nodo `Language` es el nodo de contexto en el punto donde se crea el elemento `en` en el documento de destino.
4. El resultado de la función **lang** (`true` o `false`) se envía al atributo `en/@exists` del documento de destino. El argumento de la función viene dado por la constante `en`. La función **lang** comprueba si el nodo de contexto en este punto (el elemento `Language`) tiene o no un atributo `xml:lang` con valor `en` (el argumento de la función). Si es así, la función devuelve `true`. Si no, devuelve `false`.

last, position

Las funciones **last** y **position** tampoco tienen argumentos. La función **last** devuelve la posición del último nodo del conjunto de nodos de contexto. La función **position** devuelve la posición del nodo actual del conjunto de nodos que se está procesando.

© 2018 Altova GmbH

El conjunto de nodos de contexto de los nodos a los que se dirigen las funciones es el conjunto de nodos al que afectarán las funciones. Por ejemplo, en la imagen siguiente, el conjunto de nodos de los elementos `Language` es el conjunto de nodos de contexto para las funciones **`last`** y **`position`**.



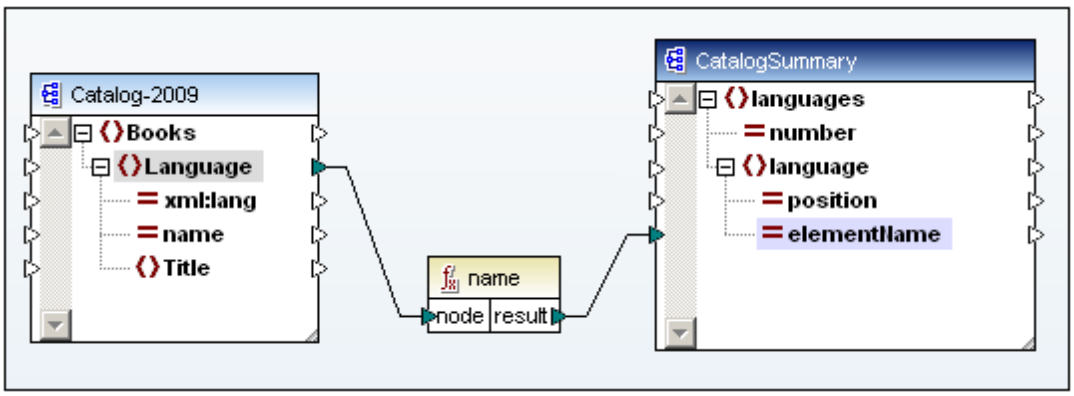
En el ejemplo, la función **`last`** devuelve la posición del último nodo del conjunto de nodos de contexto (el conjunto de nodos de los elementos `Language`) como valor del atributo `number`. Este valor también es el tamaño del conjunto de nodos, puesto que indica el número de nodos que contiene el conjunto.

La función **`position`** devuelve la posición del nodo `Language` que se está procesando en ese momento. La posición de cada nodo `Language` dentro del conjunto de nodos de los elementos `Language` se escribe en el atributo `language/@position`. Recomendamos utilizar las funciones **`position`** y **`count`** de la biblioteca de funciones **`core`**.

name, local-name, namespace-uri

Estas funciones devuelven respectivamente el nombre, el nombre local y el URI de espacio de nombres del nodo de entrada. En la imagen siguiente puede ver cómo se usan (observe que no se especifica el nodo de contexto).

La función **`name`** devuelve el nombre del nodo `Language` y lo envía al atributo `language/@elementname`. Si el argumento de la función es un conjunto de nodos en lugar de un solo nodo, devuelve el nombre (o nombre local o URI de espacio de nombres) del primer nodo del conjunto de nodos.



La función **name** devuelve el QName del nodo. La función **local-name** devuelve la parte de nombre local del QName del nodo. Por ejemplo, si el QName de un nodo es `altova:MiNodo`, el nombre local es `MiNodo`.

El URI de espacio de nombres es el URI del espacio de nombres al que pertenece el nodo. Por ejemplo, se puede declarar el prefijo `altova:` para crear una asignación a un URI de espacio de nombres de esta manera: `xmlns:altova="https://www.altova.com/namespaces"`.

Nota: en la biblioteca de funciones **core** puede encontrar más funciones XPath 1.0.

8.8.31 xslt | xslt functions

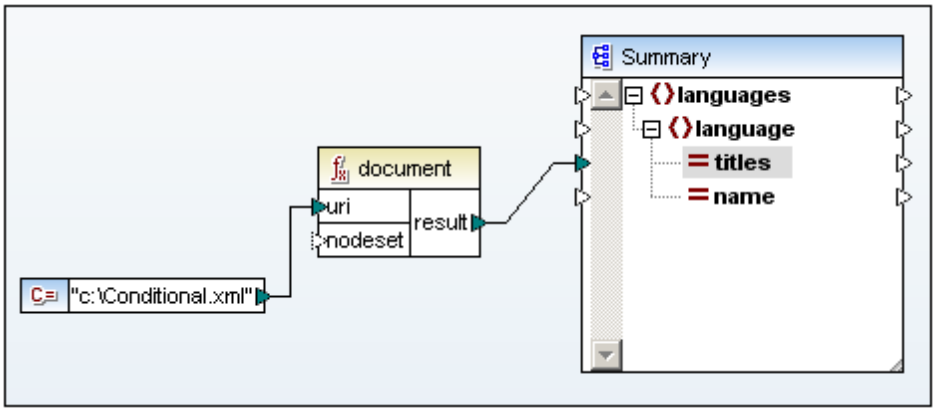
Las funciones de la biblioteca **xslt | xslt** son funciones XSLT 1.0.

8.8.31.1 currrent

Esta función no toma argumentos y devuelve el nodo actual.

8.8.31.2 document

Esta función se aplica a un documento XML externo (con el argumento **uri**). El argumento opcional **nodeset** especifica un nodo, cuyo URI base se utiliza para resolver el URI dado en el primer argumento si este URI es relativo. El resultado de la función se envía a un nodo del documento de destino.



No olvide que el argumento **uri** es una cadena y debe ser la ruta de acceso absoluta de un archivo.

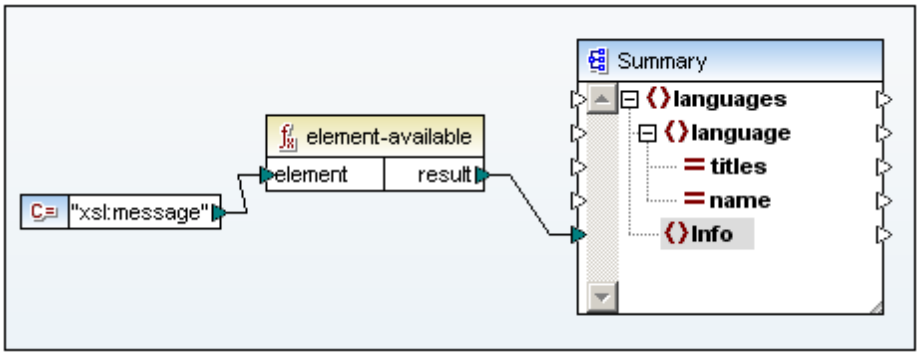
8.8.31.3 element-available

Esta función comprueba si un elemento es compatible con el procesador XSLT (el elemento se da como único argumento de cadena de la función).

La cadena del argumento se evalúa como QName. Por tanto, los elementos XSLT deben tener un

© 2018 Altova GmbH

prefijo `xsl:` y los elementos XML Schema deben tener un prefijo `xs:` porque estos los prefijos declarados para estos espacios de nombres en el XSLT que se genera para la asignación.



La función devuelve un valor booleano.

8.8.31.4 function-available

Esta función es similar a la función **`element-available`** y comprueba si el nombre de la función dado como argumento es compatible con el procesador XSLT.

La cadena de entrada se evalúa como QName. La función devuelve un valor booleano.

8.8.31.5 generate-id

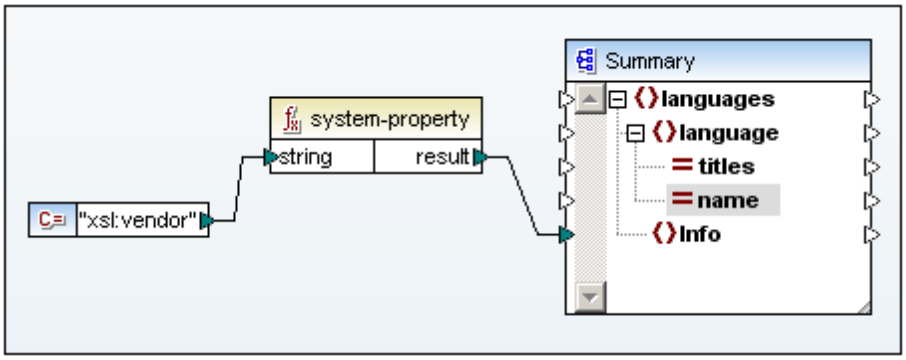
Esta función genera una cadena única que identifica el primer nodo del conjunto de nodos identificado por el argumento de entrada opcional.

Si no tiene ningún argumento, la función genera el identificador en el nodo de contexto. El resultado se puede dirigir después a cualquier nodo del documento de salida.

8.8.31.6 system-property

Esta función devuelve las propiedades del procesador XSLT (el sistema). Para los procesadores XSLT hay tres propiedades de sistema obligatorias, todas situadas en el espacio de nombres XSLT. Se trata de `xsl:version`, `xsl:vendor` y `xsl:vendor-url`.

La cadena de entrada se evalúa como QName y debe tener el prefijo `xsl:prefix`, porque este es el prefijo asociado con el espacio de nombres XSLT de la hoja de estilos XSLT subyacente.



8.8.31.7 unparsed-enity-uri

Si usa una DTD puede declarar en ella una entidad sin analizar. Esta entidad sin analizar (por ejemplo, una imagen) tendrá un URI que ubica la entidad sin analizar.

La cadena de entrada de la función debe coincidir con el nombre de la entidad sin analizar que se declaró en la DTD. La función devuelve el URI de la entidad sin analizar, que puede dirigirse a cualquier nodo del documento de destino (al nodo `href` por ejemplo).

© 2018 Altova GmbH

Altova MapForce 2018 Enterprise Edition

Implementar servicios web SOAP

9 Implementar servicios web SOAP

With MapForce Enterprise edition you can generate Java or C# program code that implements SOAP Web services, from existing Web Services Description Language (WSDL) files. You can map data to WSDL operations as follows:

- From the input of the WSDL operation to any data sources supported by MapForce, including flat files, XML, XBRL, EDI, Microsoft Excel, and databases.
- From data sources supported by MapForce to the output of the WSDL operation.

MapForce supports WSDL 1.1 and WSDL 2.0 (for additional support information and limitations, see SOAP/WSDL Support Notes and Java SOAP Web Services Specifics).

Note: As an alternative to developing SOAP Web services from existing WSDL files, you can also deploy MapForce mappings to MapForce Server, where you can run them through an API or command line call. Additionally, with FlowForce Server, you can further expose the mapping as a generic (not SOAP) Web service on the machine where FlowForce Server runs. Any mapping designed for the BUILT-IN transformation language qualifies for deployment to MapForce Server or FlowForce Server. FlowForce Server is a dedicated server solution that is capable not only to convert MapForce mappings into generic Web services, but also run them as scheduled or on demand jobs. For more information, see Deploying Mappings to FlowForce Server.

Prerequisites

To create a Web service with MapForce, the WSDL file of the Web service is required. Note that you can design WSDL files and test SOAP requests with XMLSpy, for example. Additionally, you need platform-specific software required to build, deploy, and run a SOAP Web service, for example:

Java

- Java Development Kit 1.7 or later
- Apache Tomcat: http://tomcat.apache.org
- Apache Axis2: http://ws.apache.org/axis2/, a SOAP framework running within Tomcat
- Apache Ant: http://ant.apache.org/

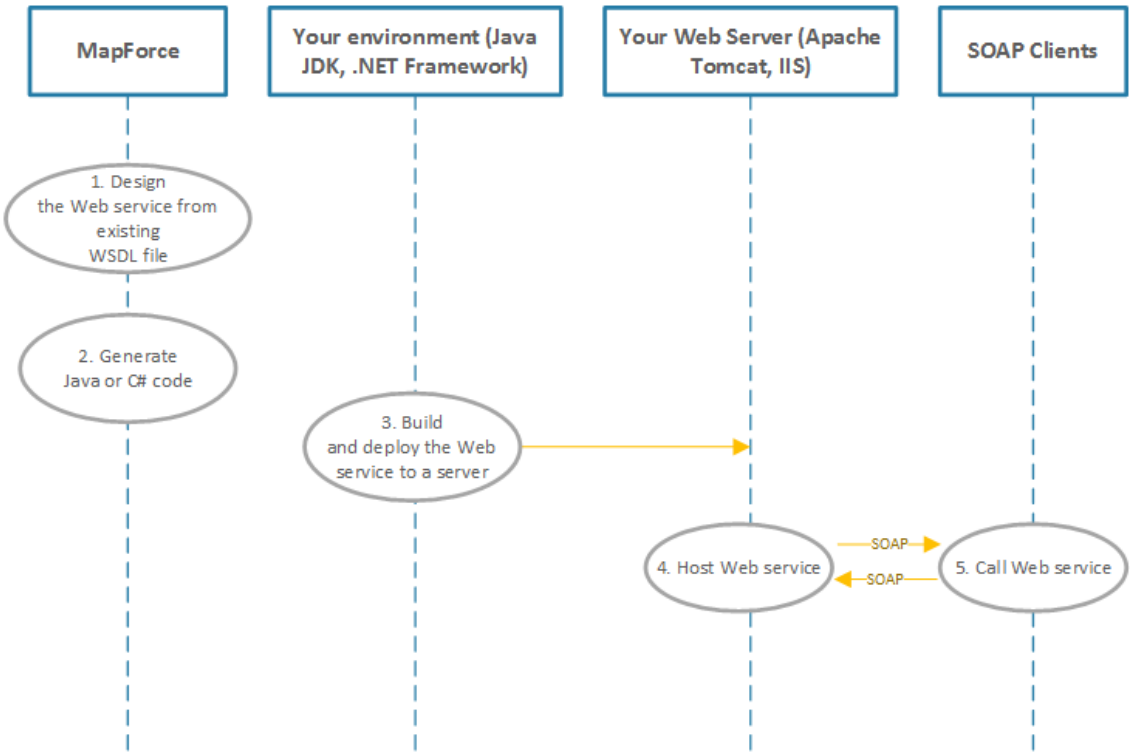
C#

- Microsoft Visual Studio 2008, 2010, 2013, 2015, 2017
- Microsoft Internet Information Services (IIS) version 5.0 or later.

How it works

The following diagram illustrates the entire process, starting from implementing a Web service from an existing WSDL file up to the stage when it can be called by SOAP client applications. Note that only the first two steps take place in MapForce. Steps 3 and 4 depend on the specifics of the third-party tools and environments used by your organization.

© 2018 Altova GmbH



The process illustrated above works as follows:

Step #	Description
1. Design the Web Service from an existing WSDL file	Once you have the WSDL file, you can start a new MapForce Web Service Project. A MapForce Web Service Project has a predefined structure that enables you to quickly access a particular WSDL operation or Web service in it. When you start a MapForce Web Service Project, you can optionally instruct MapForce to generate mapping files automatically for all WSDL operations found in the WSDL file. As an alternative, you can select individually the WSDL Services and WSDL Endpoints to be included in the MapForce Web Service Project. In the graphical user interface of MapForce, WSDL operations correspond to mappings, and the operation's input and output parameters appear as components in the mapping. While working with mappings that include input or output of WSDL operations, you can preview the execution result as you would do for any other mapping. To do this, you need a sample SOAP request file (XML format) that conforms to the WSDL grammar of the Web service (such instance files can be generated with XMLSpy).

Step #	Description
2. Generate Java or C# Code	When your Web Service Project is ready in MapForce, you can generate the C# or Java code either for individual mappings (for testing purposes), or for the entire project. In the latter case, MapForce creates a complete Web service server in the language of choice (C# or Java).
3. Build and deploy the Web service to a server	In this step, you compile the generated C# or Java code (outside MapForce) and deploy it to your custom Web server. Your Web server must be capable to host SOAP services and respond to SOAP client calls, see the "Prerequisites" section above.
4. Host Web service	The compiled Web service can be hosted by the server of your choice, for example: • Apache Tomcat • Internet Information Services (IIS)
5. Call Web service	The clients of the Web service include any application that is capable of generating SOAP calls and processing the response, including XMLSpy or MapForce itself.

For more information about step 1, see Creating SOAP Web Service Projects. For all other steps, see Example: Generating SOAP Web Services (C#) and Example: Generating SOAP Web Services (Java)

© 2018 Altova GmbH

9.1 Compatibilidad con SOAP y WSDL

Esta tabla resume qué protocolos, estándares, elementos, etc. son compatibles con MapForce.

WSDL	Versión 1.1, nota del W3C http://www.w3.org/TR/wsdl Versión 2.0, recomendación del W3C http://www.w3.org/TR/wsdl20/
Sistema de tipos WSDL	XML Schema 2001
SOAP	Versión 1.1: http://www.w3.org/TR/2000/NOTE-SOAP-20000508/ Versión 1.2: http://www.w3.org/TR/soap12-part0/
Protocolos	SOAP por HTTP (los protocolos HTTP POST, HTTP GET no son compatibles).
C#	En cada operación de C# SOAPAction debe ser distinto.
Enlaces	No se permite usar varias operaciones con el mismo nombre (sección 2.5 de la especificación WSDL 1.1).
style/use	• Document/literal: compatible • RPC/literal: compatible en C# • RPC/encoded: compatibilidad limitada • Se admite un style/use por cada servicio web (Java) o por cada operación (C#)
Encabezados SOAP	Depende de la plataforma subyacente.
SOAP encodingStyle	Si use="encoded", se asume el estilo de codificación **"http://schemas.xmlsoap.org/soap/encoding/"** para `soap:Body`. No se admiten otros estilos de codificación. El atributo `encodingStyle` se ignora en los mensajes (sección 4.1.1 de la especificación SOAP 1.1).
Referencias	• No se admiten referencias a recursos externos (sección 5.4.1 de la especificación SOAP 1.1). • Se admiten referencias a elementos independientes.
SOAP-ENC:Array	Se admite el acceso lineal. No se admiten matrices parciales ni matrices ralas.
Mejoras SOAP personales	Incompatibles
Valores fijos o predeterminados en esquemas	Incompatibles
Validación de mensajes no SOAP	No se validan, se pasan al marco subyacente.

Espacios de nombres	En WSDL no son válidas las entradas que no sean espacios de nombres y por tanto no son compatibles (WSDL y XML 1.0).

WSDL 1.1

portType

Un elemento <portType> define una interfaz de servicio web, es decir:

- las **operaciones** que se pueden llevar a cabo y
- los **mensajes** que participan como entrada y salida de cada operación.

types

El elemento <types> define los tipos de datos que utiliza el servicio web. MapForce admite el uso de esquemas XML en archivos WSDL porque es el sistema de tipos más frecuente para archivos WSDL. MapForce muestra estos elementos (tipos de datos) como elementos del componente (mensaje), para que pueda crear asignaciones entre ellos y otros elementos o construcciones directamente.

message

El elemento <message> define las **partes** de cada mensaje y los **elementos de datos** de los parámetros de entrada y salida de una operación. Estos son los mensajes que intercambian el cliente y el servidor. Hay tres tipos de mensajes: de entrada, salida y error. En MapForce cada **mensaje** es un **componente** desde el que se pueden crear asignaciones de datos. Los mensajes pueden tener una o más partes.

Cuando se use la combinación `document / literal` en MapForce, es necesario que el **mensaje** o la **parte de mensaje** remita a un elemento global en lugar de a un **tipo**. Por ejemplo, en el fragmento de código siguiente, el atributo `element` remite a un elemento global definido en un esquema (`ns2:Vendor`):

```
<message name="processRequest">
      <part name="inputData" element="ns2:Vendor"/>
</message>
```

Por el contrario, el siguiente ejemplo remite a un tipo del esquema:

```
<message name="processRequest">
      <part name="inputData" type="ns2:VendorType"/>
</message>
```

operation

Las operaciones usan mensajes como parámetros de entrada y salida. Una operación puede tener:

- un mensaje de entrada
- cero o más mensajes de salida
- cero o más mensajes de error

Los mensajes de entrada solamente se pueden usar como componentes de origen. Los mensajes de salida y de error solamente se pueden usar como componentes de destino.

© 2018 Altova GmbH

WSDL 2.0

WSDL 2.0 es bastante diferente de la especificación WSDL 1.1. Estas son las principales diferencias:

- Los elementos `portType` se llaman interfaces en WSDL 2.0.
- Los mensajes y las partes se definen usando el sistema de tipos de XML Schema en el elemento `types` de WSDL 2.0.
- Los puertos se llaman extremos en WSDL 2.0.
- Las entradas y salidas de las operaciones de WSDL 20. se definen en el esquema XML.

En MapForce el cuadro de diálogo "Configuración del componente" de un componente WSDL muestra el campo de entrada *Extremo*, tanto para los puertos WSDL 1.1 como para los extremos WSDL 2.0.

9.2 Crear proyectos de servicio web a partir de archivos WSDL

This example shows you how to create a SOAP Web service in MapForce. The mapping project created in this example is also available at the following path: **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial\Query Person database.mfp**. The Web service in this example has the following goals:

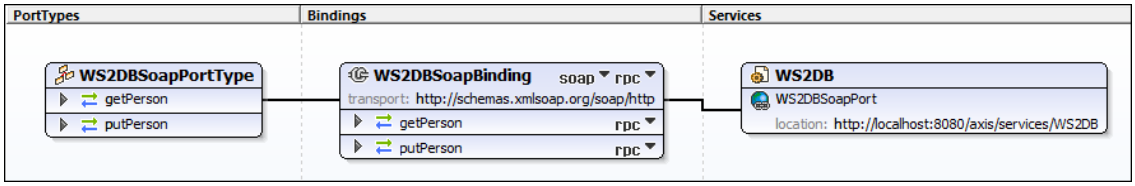
- Retrieve from a Microsoft Access database on a server a set of person records that satisfy some search criteria, through a SOAP Request. (The query is entered at runtime on the client, and is then sent to the server. The server then returns an XML response with the search results).
- Insert new records into the Microsoft Access database, also through SOAP requests. (The person details are provided in the SOAP Request. On success, the server returns a confirmation that the person details have been successfully added to the database.)

Note the following:

- The process of creating the Web service in MapForce does not depend on the target programming language; it is identical when generating Java, or C# Web services. The differences arise only when you compile and deploy the Web service on the web server, as further described in this documentation.
- In order to generate sample SOAP requests required to preview the mapping output, this example uses XMLSpy Enterprise Edition. The source WSDL file of this MapForce project was also created with XMLSpy (and can be validated with it). Designing Web services in MapForce is also possible without XMLSpy; however, in this case you will need a SOAP client capable to generate SOAP requests from WSDL files.

Creating a Web service project

As stated before, a WSDL file is required to create a SOAP Web service with MapForce. In this example, we will use a WSDL file that was created with XMLSpy Enterprise edition. The image below illustrates the WSDL structure as it is displayed in the WSDL view of XMLSpy. The important thing to notice is that it defines two operations (`getPerson` and `putPerson`) that correspond to the goals outlined above.

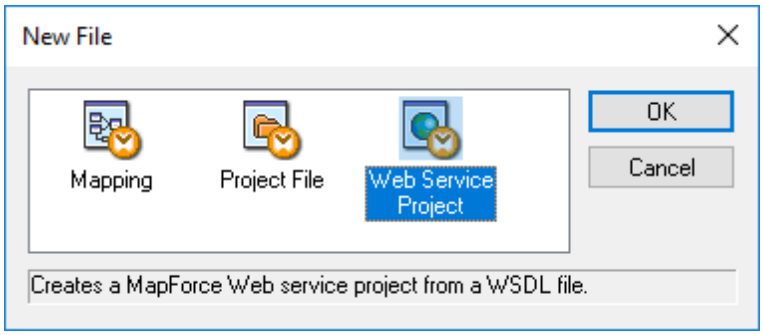


Note: The WSDL file must be valid before you can create a MapForce Web Service project from it.

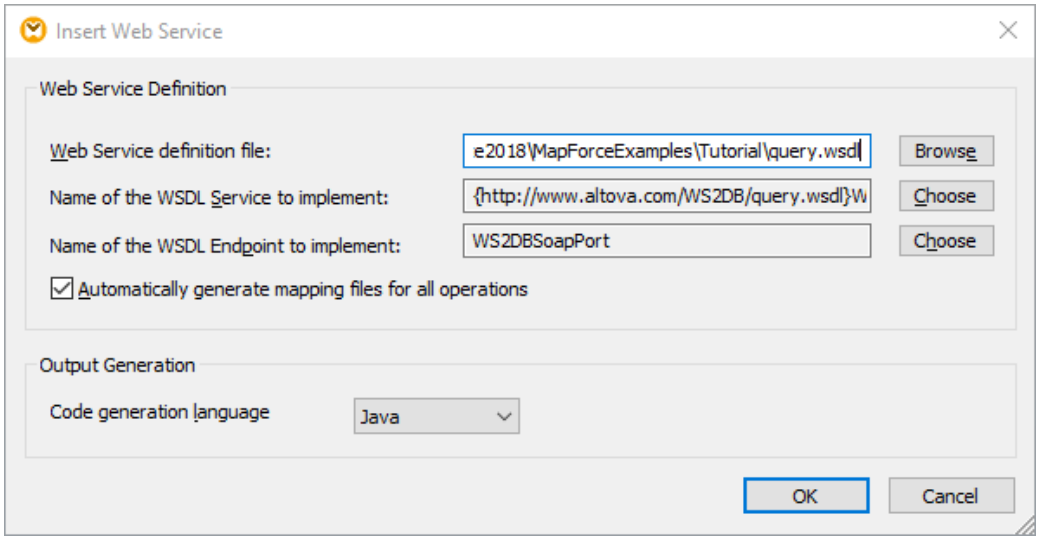
To create a Web service project from this web service, follow the steps below:

1. On the **File** menu, click **New**, and select "Web Service Project".

 © 2018 Altova GmbH

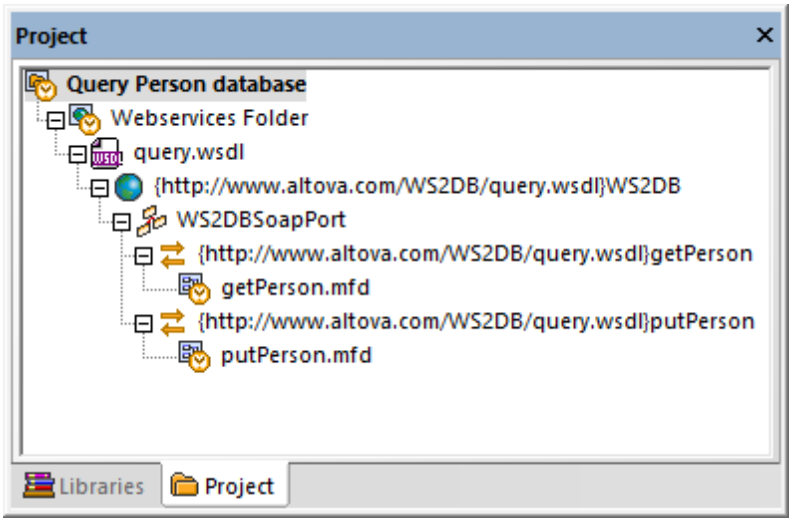


2. Browse for the **query.wsdl** file available in the folder **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial** . After you select the WSDL file, MapForce automatically fills in the remaining fields.



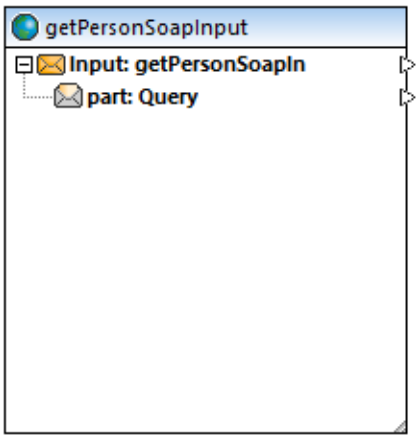
3. Click OK.
4. When prompted, enter the name of the new WSDL project, and click **Save**.

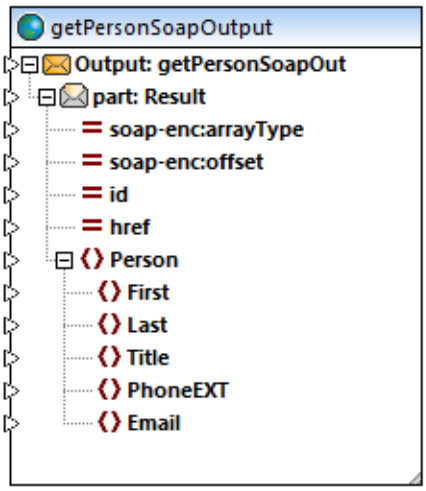
The Project window shows the project and WSDL name, as well as each of the operations defined in the WSDL file. The two operations are **getPerson** and **putPerson**.



Designing the "getPerson" operation

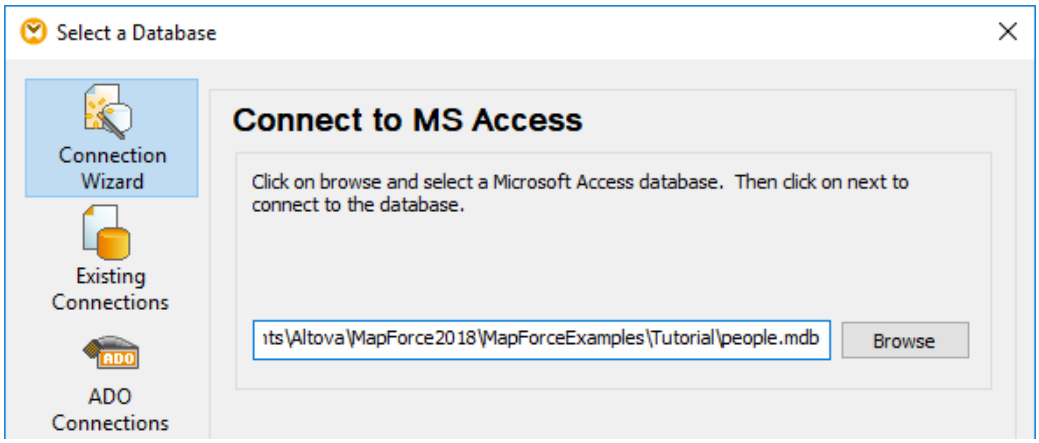
Double-click the **getPerson.mfd** file in the Project window to load its contents in the main pane. The **getPersonSoapIn** component contains the query (item) which will be used to query the database through the Web service. The **getPersonSoapOut** component contains the `Person` structure defined in the WSDL file.



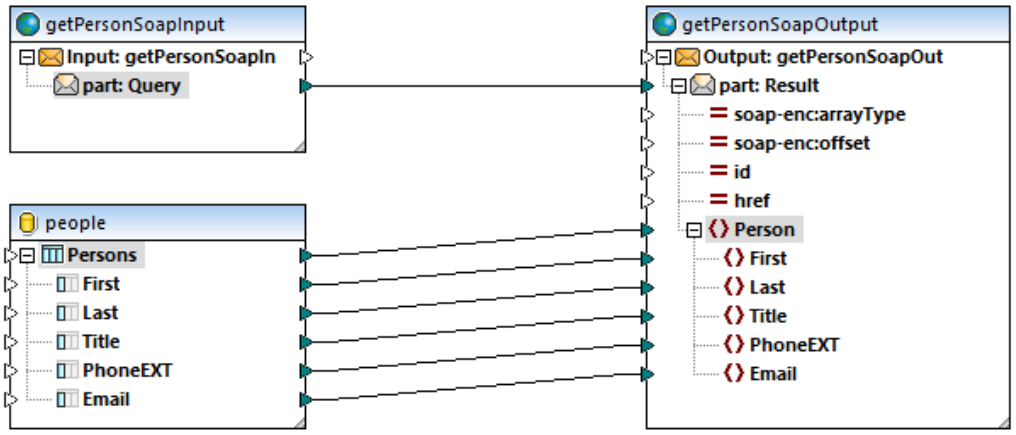


On the **Insert** menu, click **Database**, and add to the mapping area the following Access database: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\people.mdb**. The database contains only one table, `Persons`, whose fields match those in the `getPersonSoapOut` component. For more information about adding databases as mapping components, see Databases and MapForce.

 © 2018 Altova GmbH



Draw connections between components as shown below:



The mapping design above is not yet complete but clearly illustrates what will be happening at runtime. The connection between **part:Query** and **part:Result** represents the Web service call. The **part:Query** item of the `getPersonSoapIn` component is the query placeholder (it is where the query string is entered in the SOAP client). The connections between the database table and the **Person** item of the **getPersonSoapOutput** component extract values from the corresponding database fields and pass them to the corresponding target items. You can already quickly preview the output as follows:

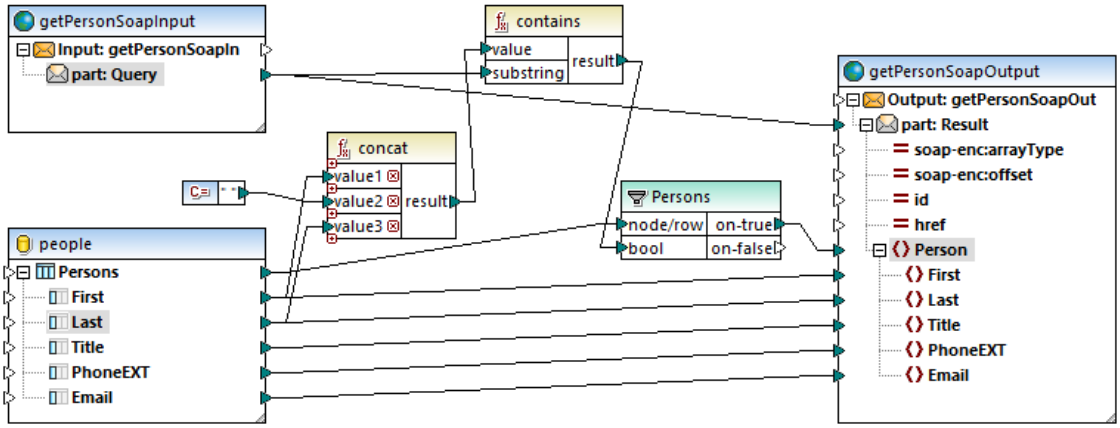
1. Right-click the **getPersonSoapInput** component.
2. Click **Browse** and select the following example request: **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial\getPersonRequest.xml**.
3. Click **OK**.
4. Click the **Output** tab.

At this stage, the mapping extracts all the records from the database table, which is not very useful. Therefore, we need to add a filter and extract only items that match the request string.

1. Add the **`concat`** function, by dragging it from the Libraries window. This function will be used to concatenate the `First` and `Last` names of each person.
2. On the **Insert** menu, click **Constant**, and add a constant which contains one space character. The constant will supply the space character between the first and the last

name extracted from the database.

3. Add the **contains** function, by dragging it from the Libraries window.
4. Right-click the connection between the source **Persons** table and **Person** target node, and select **Insert Filter: Nodes/Rows** from the context menu.
5. Draw connections between components as shown below.

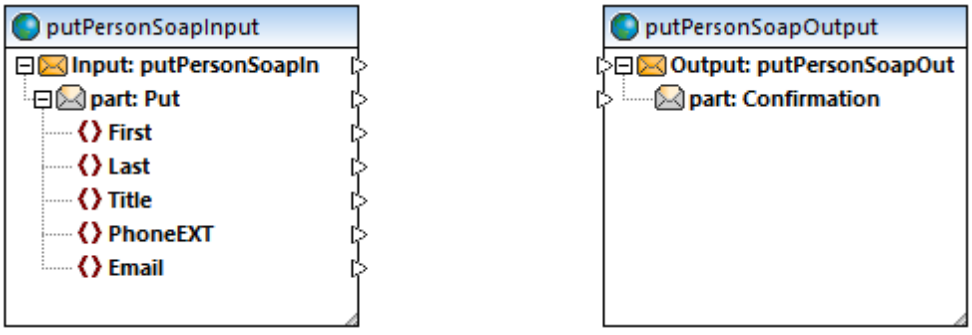


"getPerson" mapping

As illustrated above, the **contains** function returns **true** when the query string matches the full or partial name of any person in the database (where "name" is provided by the **concat** function, and it consists of the first name, followed by a space, followed by the last name). When there is a match, the details of the matching person are included in the response message. Therefore, in this example, the search value "Ro" would return all the following records: "Martin **Ro**pe", "**Ro**nald Superstring", "**Ro**bert Darkmatter", and "**Ro**ger Gravity".

Designing the "putPerson" operation

Double-click the **putPerson.mfd** file in the Project window to load its contents in the main pane. The **putPersonSoapInput** component supplies the request structure (items that will be inserted into the database through the Web service). The **putPersonSoapOutput** component represents the result of the Web service call. It contains a confirmation part to display the outcome of the call to the Web service caller.



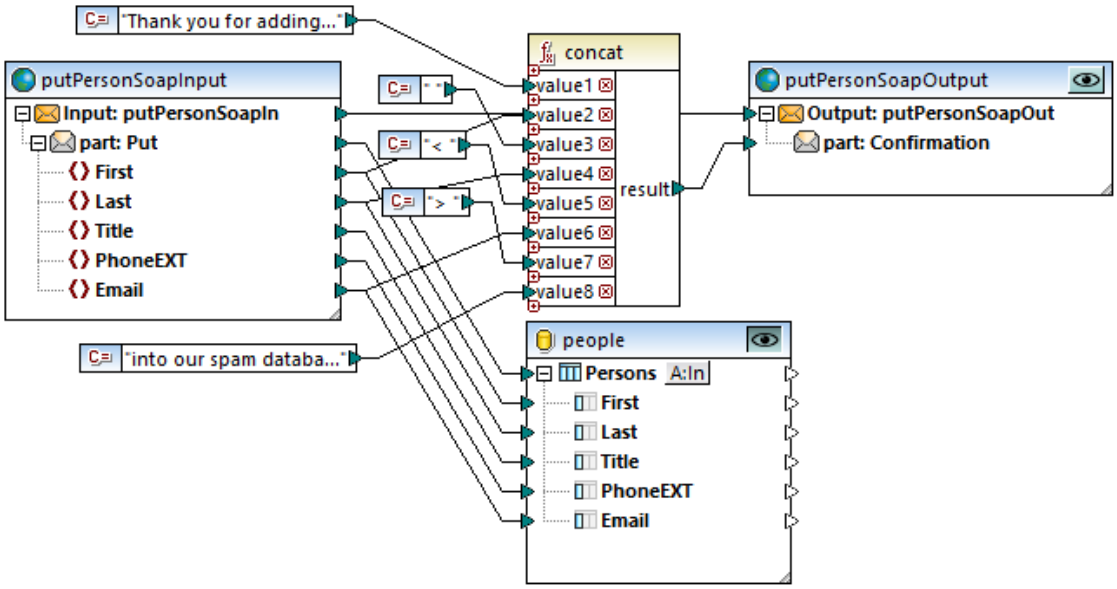
Follow the same instructions as above to add the following to the mapping:

- The **people.mdb** Access database (this is where data supplied by the Web service

 © 2018 Altova GmbH

caller will be inserted)
- The **concat** function and five constants (these will be used to build the confirmation string shown to the caller of the Web service)

Draw the connections as shown below:



"putPerson" mapping

In the mapping above, the connection between **part: Put** and the database will cause the database to be updated at runtime. On the target side, the **part: Confirmation** response is obtained by concatenating various fields from the request, as well as some text supplied by constants. For more information about mappings which update databases, see Mapping Data to Databases.

To preview the output of the mapping:

1. Right-click the **getPersonSoapInput** component.
2. Click **Browse** and select the following example request: **<Documentos>\Altova \MapForce2018\MapForceExamples\Tutorial\putPersonRequest.xml**.
3. Click the **Output** tab.

When you preview the mapping, MapForce will generate the INSERT query required to update the database and display it in the Output window, but will not perform the actual update. To run the query against the database, select the menu command **Output | Run SQL-Script**.

```
/*
The following SQL statements are only for preview and may not be executed in another SQL query tool!
To execute these statements use function "Run SQL-script" from menu "Output".
Connect to database using the following connection-string:
Data Source=C:\Users\altova\Documents\Altova\MapForce2018\MapForceExamples\Tutorial\people.mdb;Provider=Microsoft.Jet.OLEDB.4.0
*/

INSERT INTO [Persons] ([First], [Last], [Title], [PhoneEXT], [Email]) VALUES ('John', 'Doe', 'Mr', '123', 'john.doe@example.org')

```

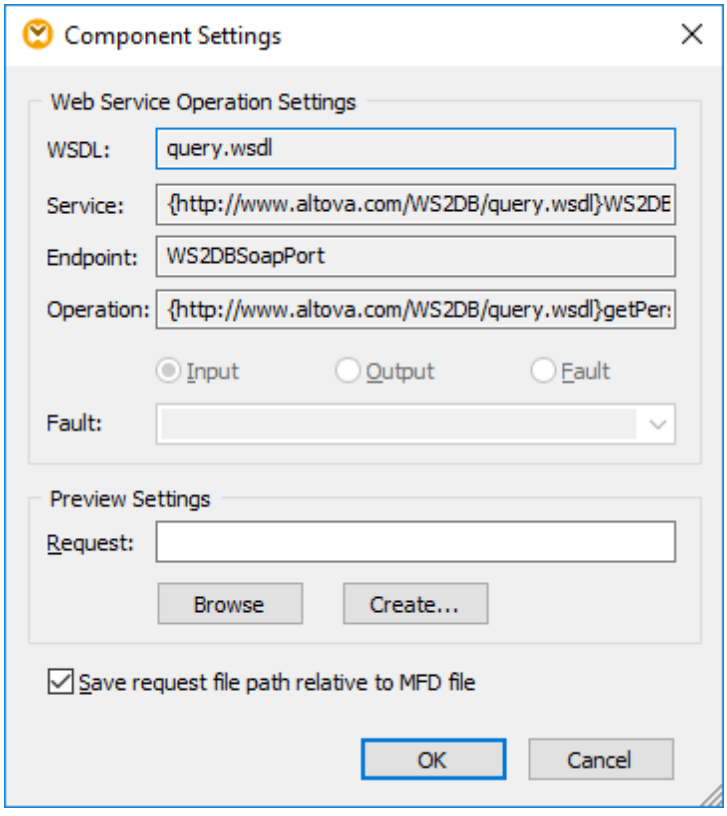
By contrast, when the compiled Web service is deployed to a server and called by a client, the

database will be updated with immediate effect. See also Executing Mappings Which Modify Databases.

Previewing the mapping output

Mappings created from WSDL files can be previewed in MapForce before they are deployed to an actual Web server. To preview such mappings, a sample SOAP request (XML file) is necessary which conforms to the WSDL file. You can create a sample request for each operation (mapping) inside your Web service project either with XMLSpy or another SOAP client of your choice. If you use XMLSpy, run the menu command **SOAP | Create New SOAP Request** to generate the request. If XMLSpy is installed alongside MapForce, you can generate a sample request directly from MapForce, for example:

1. Right-click the **getPersonSoapInput** component on the mapping, and select **Properties** from the context menu.



2. Click **Create**.
3. When prompted, save the XML file to a location on the disk.

The request file is generated with some dummy input values (for example, "String") instead of input parameters. Before using the request file, make sure to replace the dummy values with the actual request values that you want to use. For this project, two sample requests are available:

1. **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial \getPersonRequest.xml.** Use this file to preview the output of the **getPerson** mapping. By default, this request file supplies the value "Ro" as input to the Web service. This will

 © 2018 Altova GmbH

retrieve from the database all people records where the last or first name contains the text "Ro".

2. **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial \putPersonRequest.xml.** Use this file to preview the output of the **putPerson** mapping. By default, this request file supplies the details of a fictitious person as input to the Web service.

To obtain a different output while previewing the two mappings, therefore, edit the files above so that they contain the input values you wish to test.

Next steps

You are now ready to generate code that implements the Web service (either Java or C#), compile it, and deploy it to a server, as further described in the following examples:

- Example: Generating SOAP Web Services (C#)
- Example: Generating SOAP Web Services (Java)

9.3 Definir errores de servicio web SOAP

A WSDL file can contain a `fault` element for an operation and a message attribute that contains the fault message. Accordingly, when designing a SOAP Web service in MapForce, you can add a "Fault" component to your mapping. This will cause the Web service to throw an exception if a certain condition is satisfied.

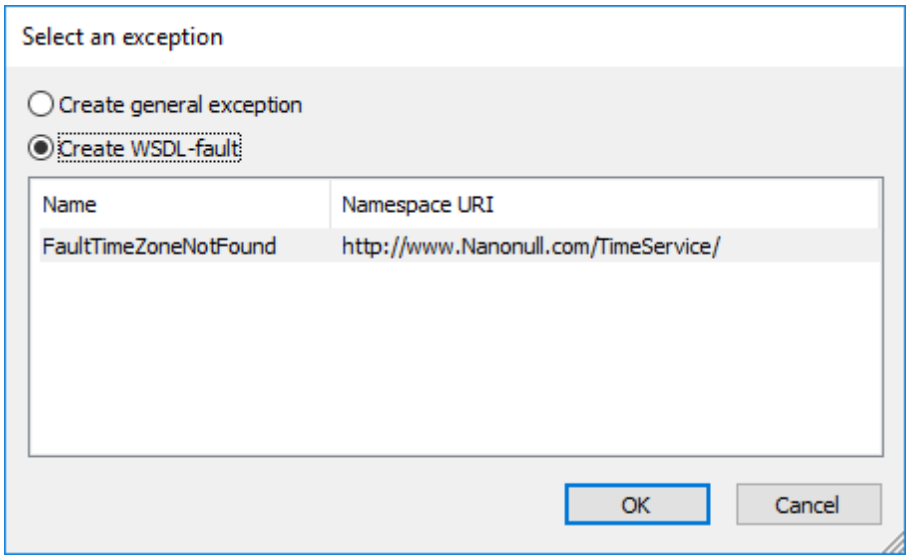
> To insert a Fault component in a mapping, a `fault` element must be present in the WSDL file.

You can define the failure condition inside the mapping, and you can also set the text of the error message. When the condition is satisfied, the Web service will fail as follows:

- In a live environment (at execution time, after the Web service is deployed to a server), the Web service will fail and return the custom-defined message to the caller.
- In MapForce (at design time), the mapping execution stops, and you can view the message in the **Message** window.

To add a fault component to a MapForce mapping:

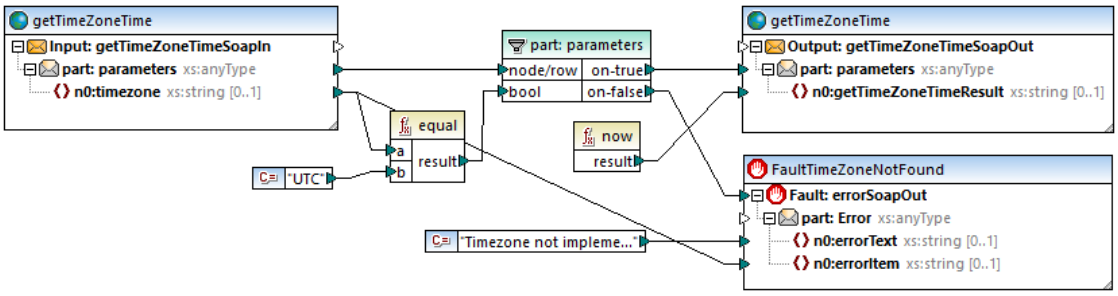
1. On the **Insert** menu, click **Exception**. (Alternatively, click the **Exception** toolbar button.)
2. Select **Create WSDL-fault**.



Note: If the **Create WSDL-fault** option is disabled and you see the message "No faults for this operation", this means that no faults exist in the WSDL file, and, consequently, none can be added in MapForce.

For an example of a mapping which contains a Web service fault, open the following sample MapForce project: **<Documentos>\Altova\MapForce2018\MapForceExamples\TimeService\TimeService.mfp**. Double-click the mapping (operation) **getTimeZoneTime.mfd**.

© 2018 Altova GmbH



In the mapping above, the exception should be triggered when **n0:timezone** is not equal to UTC.

- The **equal** function checks to see if timezone equals UTC, with the bool result being passed on to the filter.
- If the condition is false, i.e. something other than UTC, the **on-false** parameter of the filter activates the **Fault:errorSoapOut** exception and the mapping process is halted. (Note that you can also connect the exception to the on-true parameter, if that is what you need.)
- Two sets of error text are supplied by the SoapFault message.

It is very important to note the filter placement in the example:

- Both parameters of the filter component, **on-true** and **on-false**, must be mapped. One of them needs to be mapped to the fault component, and the other, to the target component that receives the filtered source data. Otherwise, the fault component will never be triggered.
- The exception and target components must be directly connected to the filter component. Functions or other components may not be placed between the filter and either the exception, or target components.

9.4 Información sobre servicios web SOAP Java

MapForce generates all necessary code and scripts needed to create a SOAP Web service. However, for the Web service to be available to consumers, the generated code must be built and deployed to the Apache Tomcat server with Axis2 installed. Additionally, Apache Ant is required to build the Java code.

Generating and building Java code

1. Open the Web service project from which you want to generate code. See the previous section, Creating Web Service Projects from WSDL files, for an example of how to create such a project.
2. On the **Project** menu, click **Generate code in | Java**, and select the target directory. When code generation completes, several folders and files are created in the target directory, including a **com** directory which contains Altova generic classes, as well as the actual classes of the Web service project.
3. Build the generated Java code (by supplying to Apache Ant the **build.xml** file generated by MapForce). As a result, an Axis Archive File (*.aar) file is created, which you can then deploy to Axis2.

Note for Eclipse users: when you open the generated project with Eclipse, you may see an error like "The import org.apache cannot be resolved". In this case, make sure that the Axis2 libraries are added to the Java build path. To add the Axis2 libraries to the Java build path in Eclipse 4.4.2, do the following:

1. Right-click the project in the Package Explorer, and select **Properties**.
2. Click **Java Build Path**.
3. On the **Libraries** tab, click **Add External JARs**, and add the Axis2 libraries from the `<AXIS2_HOME>\lib` folder.

Deploying the Web service

To deploy the Web service, do one of the following:

- Open the "Upload Services" Web administration page of Axis2 and upload the .aar file created in the previous step
- Do a manual upload. For example, if your Tomcat server was installed to the folder `<TOMCAT_HOME>`, you can manually copy the .aar file to `<TOMCAT_HOME>\webapps\axis2\WEB-INF\services`.

For a step-by-step example, see Example: Generating SOAP Web Services (Java).

Undeployment

Delete the *.aar file from the `<TOMCAT_HOME>\webapps\axis2\WEB-INF\services` folder.

© 2018 Altova GmbH

Axis2 limitations

Axis2 support for **RPC/encoded** is limited. MapForce can, however, generate RPC/encoded Web services (both SOAP 1.1 and SOAP 1.2). The limitation is that the original WSDL is not retrieved from the Web server.

This means that, for example, http://127.0.0.1/axis2/services/WS2DB?wsdl would **not** return a usable .wsdl file.

For **document/literal** Web services, the URL above will provide a usable and correct .wsdl file. It will differ from the original, however: comments will be stripped out, and namespaces will be changed. It will, however, still have the same semantics as the original .wsdl file with which the service was generated.

Although Axis2 does not support **RPC/encoded**, it is able to generate WSDL from deployed Java code (compiled code), and thus MapForce-generated code can process **RPC/encoded** messages; Axis2 is just used for transport.

Known issue: namespaces in the SOAP response message

The code generated by MapForce instantiates the **`javax.xml.transform.TransformerFactory`** class. When the class implementation is loaded, it might be read from the **`javax.xml.transform.TransformerFactory`** system property (for details, refer to the Java documentation of this class).

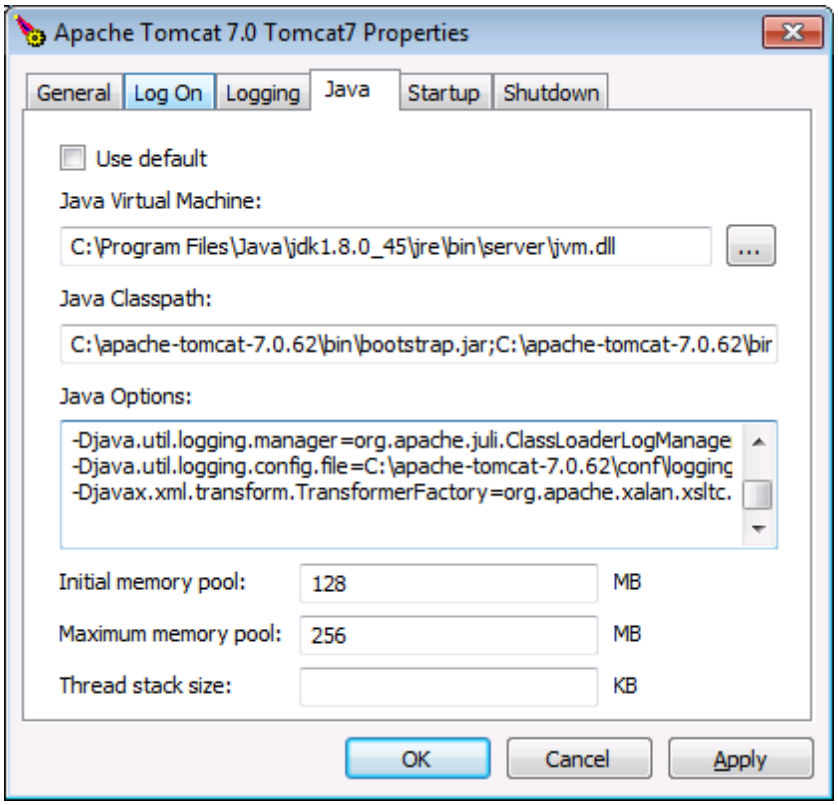
If the system property **`javax.xml.transform.TransformerFactory`** in your Java environment defines the implementation **`org.apache.xalan.transformer.TransformerIdentityImpl`**, the SOAP response message may be generated with incorrect namespaces.

To prevent this issue, it is recommended to use the **`org.apache.xalan.xsltc.trax.TransformerFactoryImpl`** implementation. You can do this by adding the following parameter to the Java Virtual Machine (JVM):

```
-
Djavax.xml.transform.TransformerFactory=org.apache.xalan.xsltc.trax.Transforme
rFactoryImpl
```

The instructions for adding the parameter to the JVM depend on the operating system and server you are using. The following instructions are applicable to Apache Tomcat 7.0 configured to run as a service application on Windows 7:

1. Run **Tomcat7w.exe** (this file is located in the **\bin** subfolder of the Tomcat distribution root folder, `CATALINA_HOME`).
2. Click the **Java** tab.
3. At the end of "Java Options" box, enter: `-Djavax.xml.transform.TransformerFactory=org.apache.xalan.xsltc.trax.TransformerFactoryImpl`



 © 2018 Altova GmbH

9.5 Ejemplo: generar servicios web SOAP (C#)

This example illustrates how to generate a SOAP Web service with MapForce. You will generate C# program code from a sample MapForce project available at the following path: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\\Query Person database.mfp**. This project implements a SOAP Web service that exposes two operations: `getPerson` and `putPerson`. This Web service communicates in the background with a Microsoft Access database to retrieve or add person details, respectively, through the operations above. For an example of how to create such projects in MapForce, see Creating Web Service Projects from WSDL Files.

This example uses Visual Studio 2015 to compile the generated C# code. The solution will be deployed to Internet Information Services (IIS) 10 with ASP.NET enabled.

Tenga en cuenta que, para simplificar, en el ejemplo se emplea una configuración de servidor web mínima que no debe entenderse como preceptiva. En un entorno de producción es muy probable que el servidor web, las directivas de seguridad de la organización y otros factores exijan un enfoque distinto.

Para llamar al servicio web SOAP en este ejemplo se utiliza XMLSpy Enterprise Edition. Si no tiene XMLSpy, necesitará un cliente SOAP para probar el servicio web o, si lo prefiere, deberá escribir código de programa que pueda llamar al servicio web.

Note: If you are using Visual Studio 2008, and IIS 7.x, you may first need to install the Windows feature "IIS Metabase and IIS 6 configuration compatibility".

Step 1: Prepare the query.wsdl file

This example Web service originates from the following WSDL file: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\query.wsdl**. By default, the **query.wsdl** file defines the service details for both C# and Java in two separate sections, one of which is commented out. Therefore, before generating C# code, locate the following section in the **query.wsdl** file and make sure that it is not commented out:

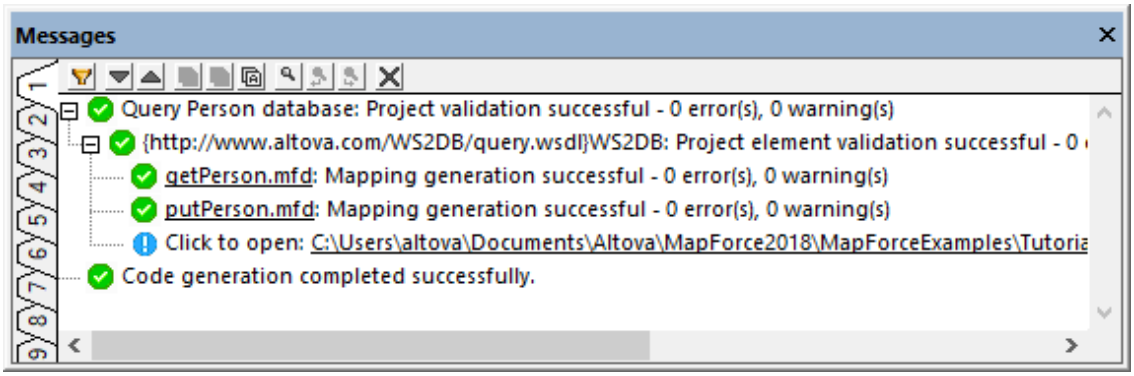
```
<service name="WS2DB">
   <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
      <soap:address location="http://localhost/WS2DB/WS2DB.asmx"/>
   </port>
</service>
```

Also, make sure that the following section (only applicable to Java) is commented out:

```
<!--<service name="WS2DB">
   <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
      <soap:address location="http://localhost:8080/axis/services/WS2DB"/>
   </port>
</service>-->
```

Step 2: Generate C# code and build it

Run MapForce and open the "Query Person database.mfp" project from the folder **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**. Right-click the project in the Project window, and select **Generate code in | C#**. MapForce generates program code and displays the outcome in the Messages window:



By default, code will be generated in a subdirectory called "output", relative to the project's directory. For the scope of this tutorial, we will leave the default settings as is. However, if you want to change the output directory, see Setting the Code Generation Settings .

Run Visual Studio and open the generated solution. Be patient while all projects in the solution finished loading.

> You may need to run Visual Studio as administrator. Also, this example solution must be compiled as a 32-bit application, as shown below.

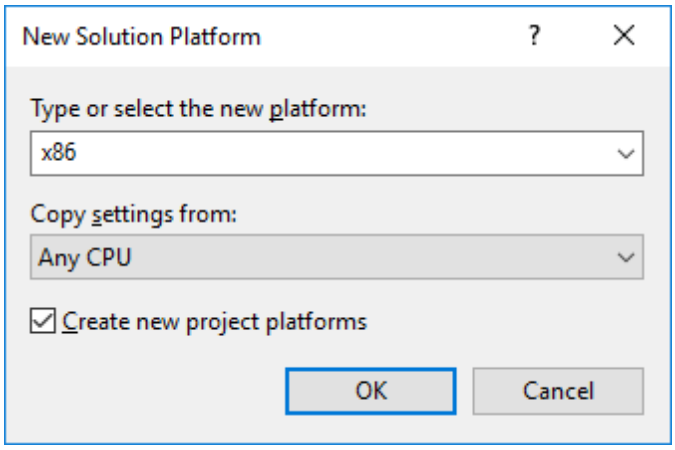
You can build the solution as follows:

- On the **Build** menu, click **Build Solution (Ctrl+Shift+B)**.

Note that this example queries a Microsoft Access database using the ADO provider for Access, which only works with 32-bit applications. Therefore, the solution must be compiled as a 32-bit application, as follows:

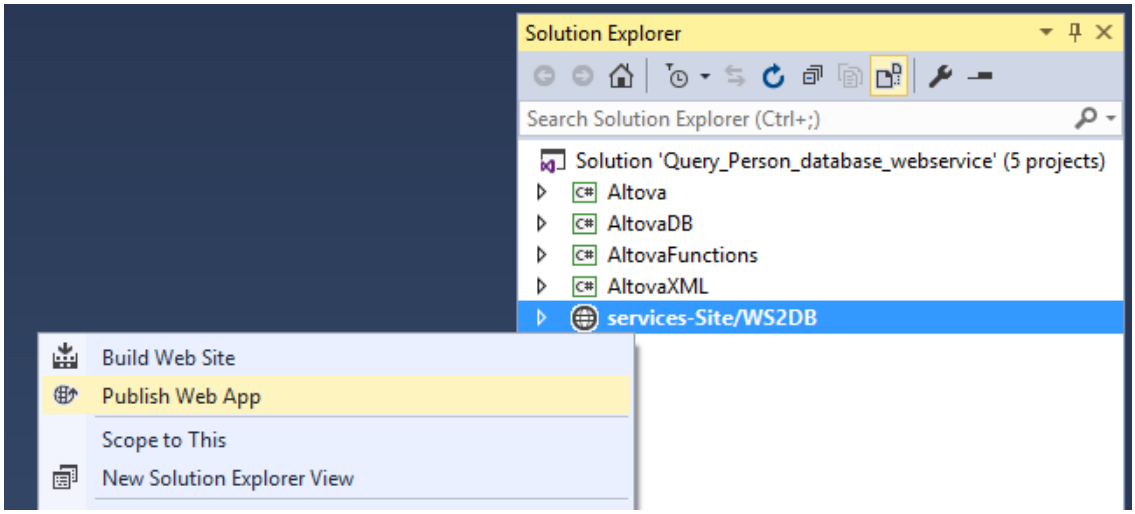
- On the **Build** menu, click **Configuration Manager**, create a new solution platform for x86, and build again.

© 2018 Altova GmbH

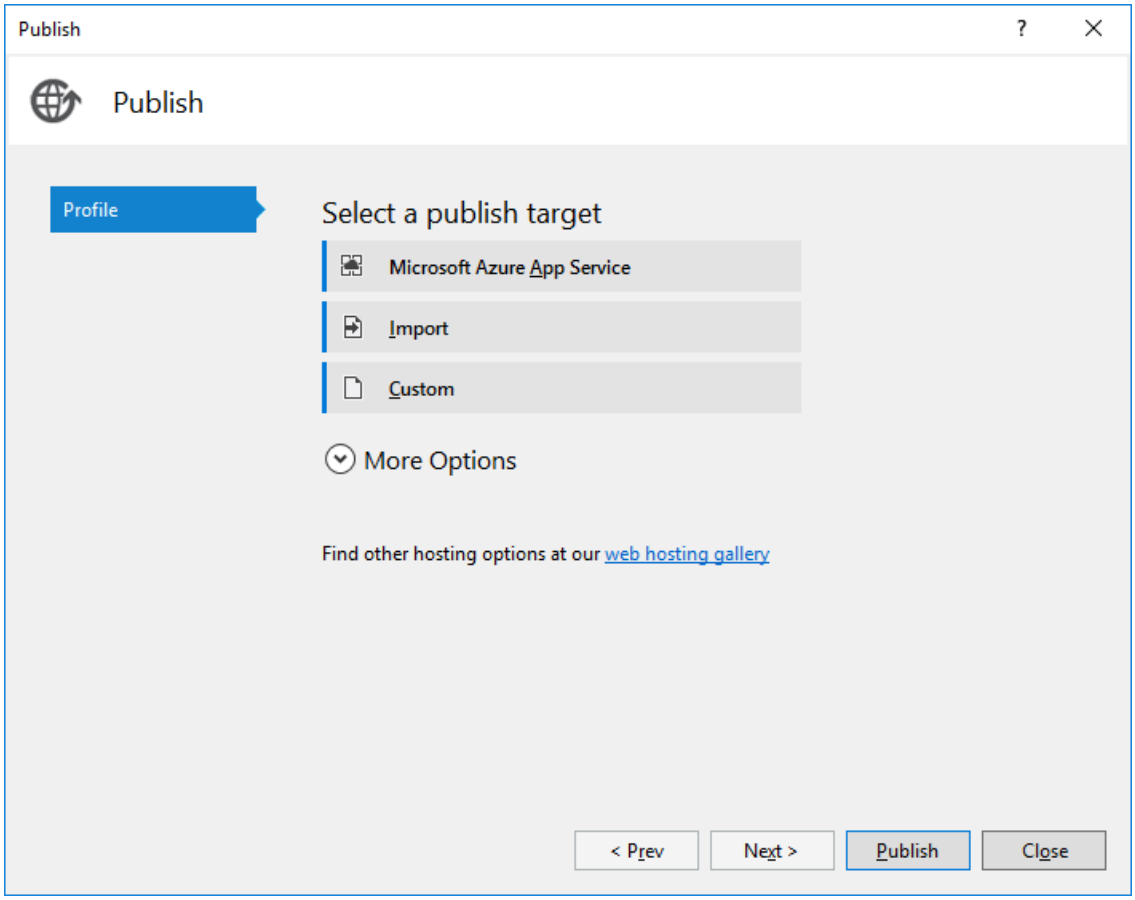


Step 3: Deploy the Web service to IIS

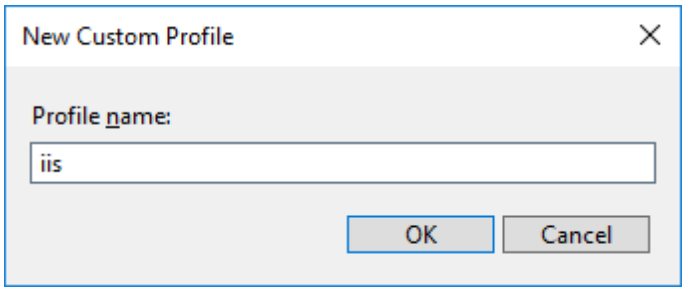
In the Solution Explorer window of Visual Studio, right-click the project **services-Site/WS2DB** and select "Publish Web App" from the content menu.



When prompted to select a publish target, select "Custom".

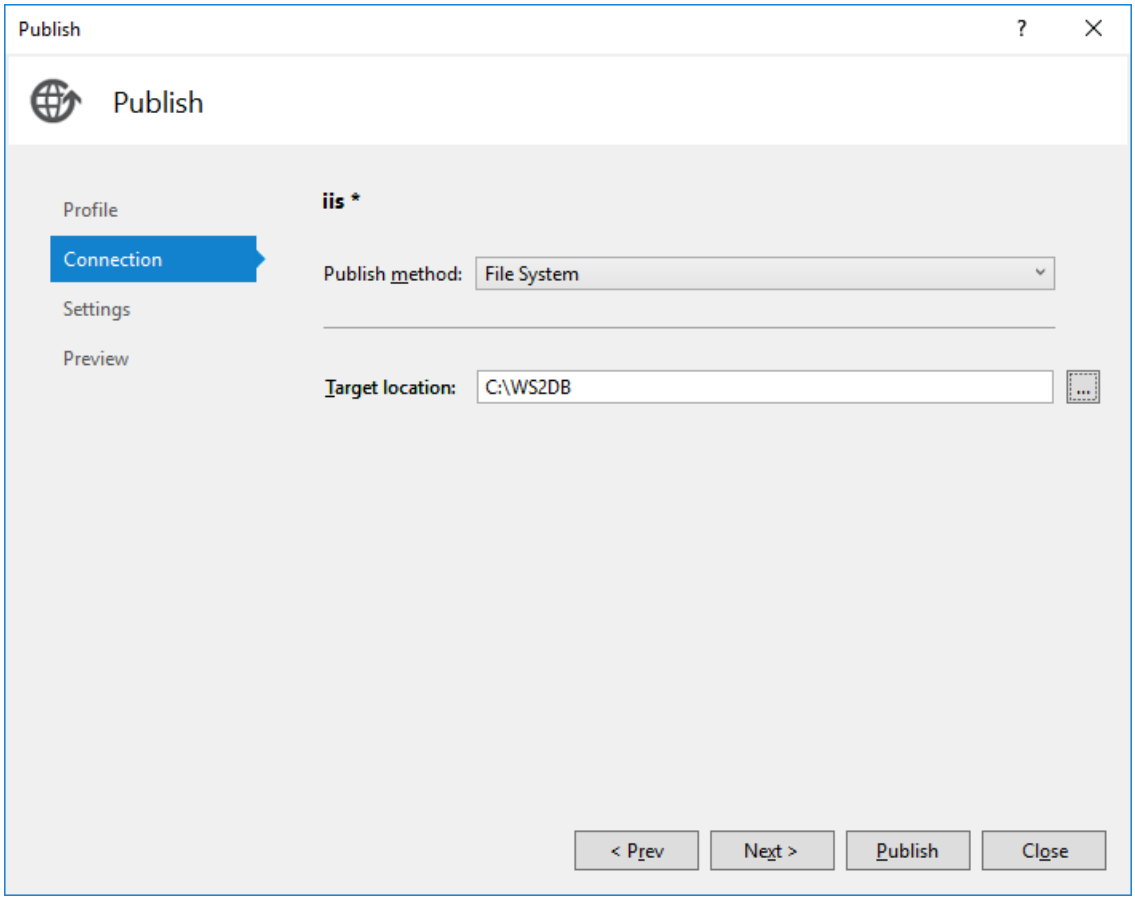


When prompted to enter a profile name, enter a value that will be easy to recognize later, for example "iis".

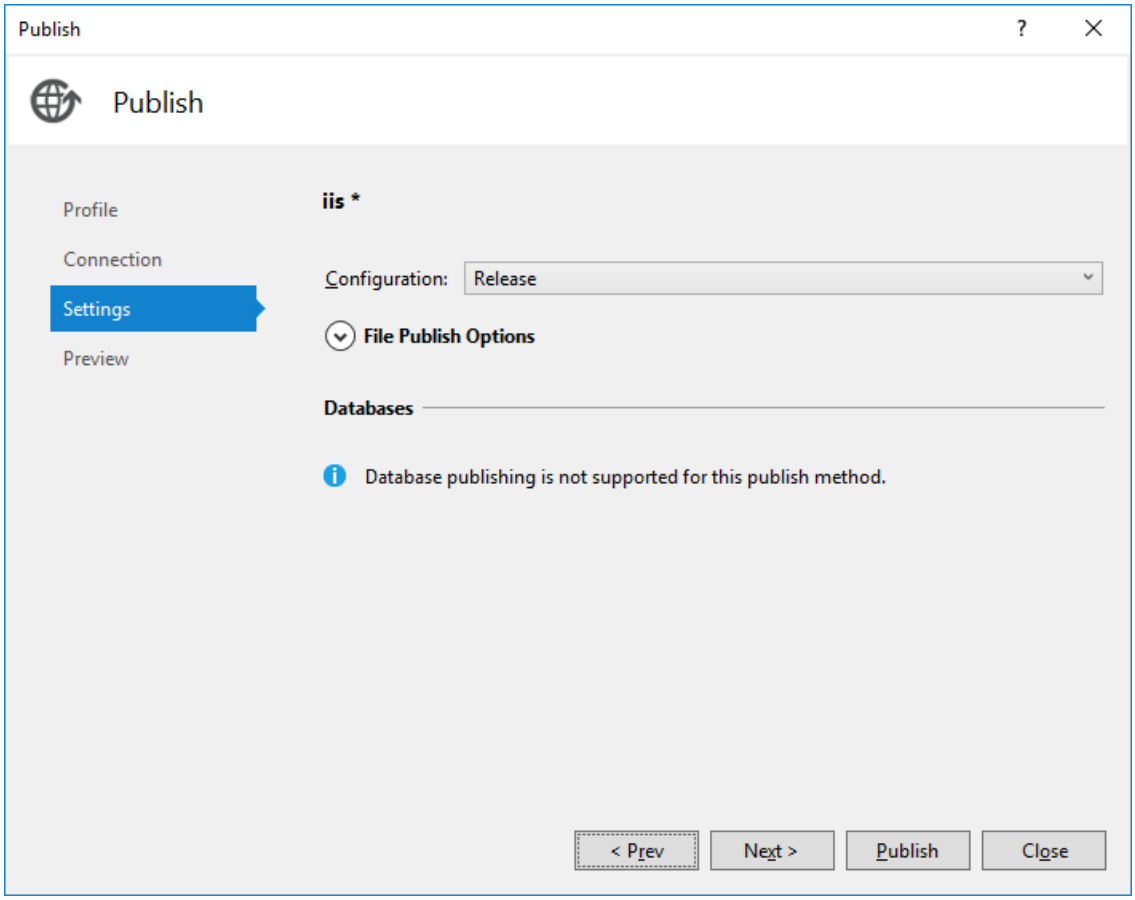


Select "File System" as publish method and choose a destination folder (in this example, "C:\WS2DB").

 © 2018 Altova GmbH

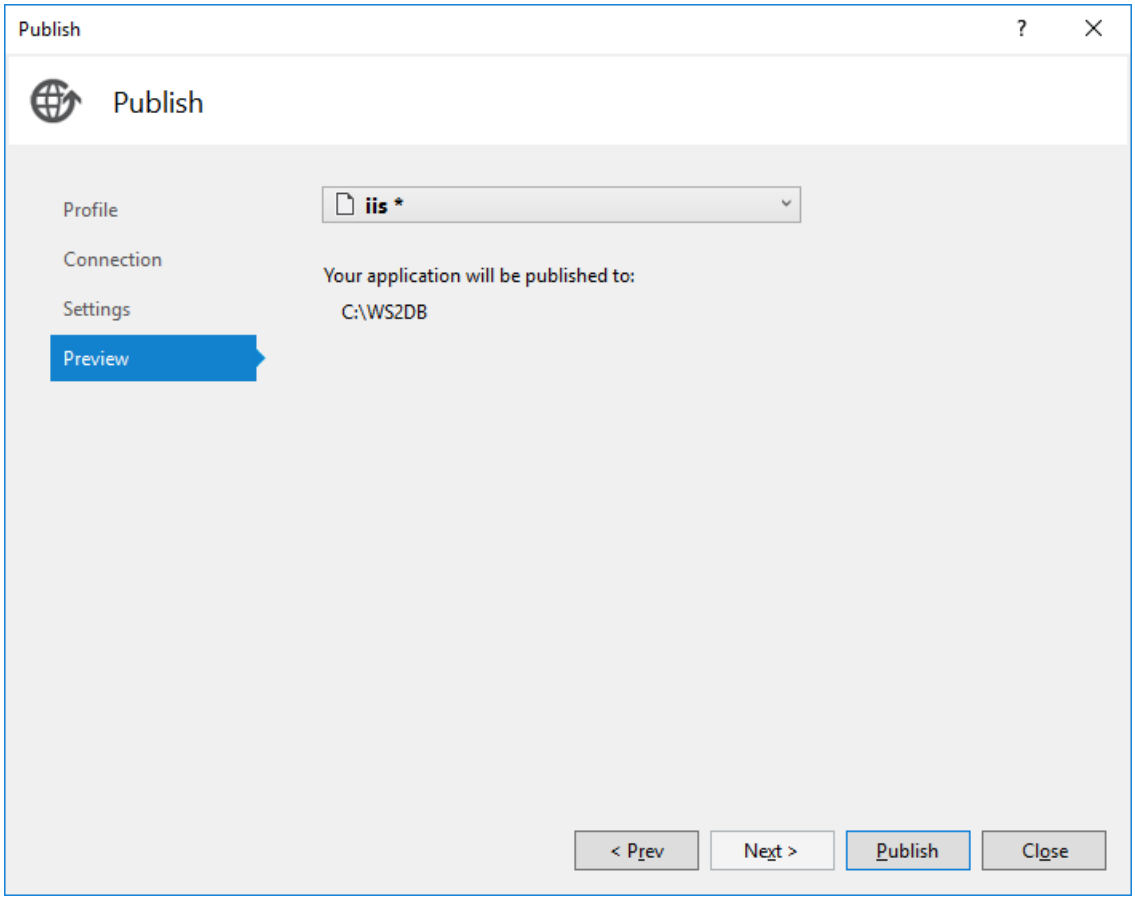


Click **Next**. Leave the default settings as is.

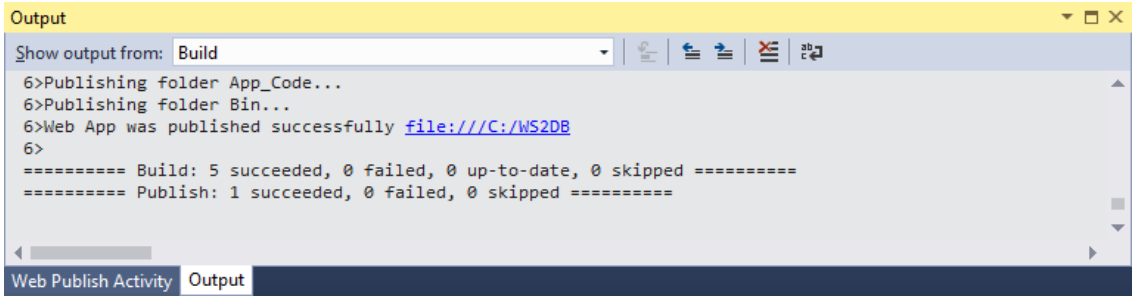


Click **Next**, and then click **Publish**.

© 2018 Altova GmbH



The Output window displays the result.

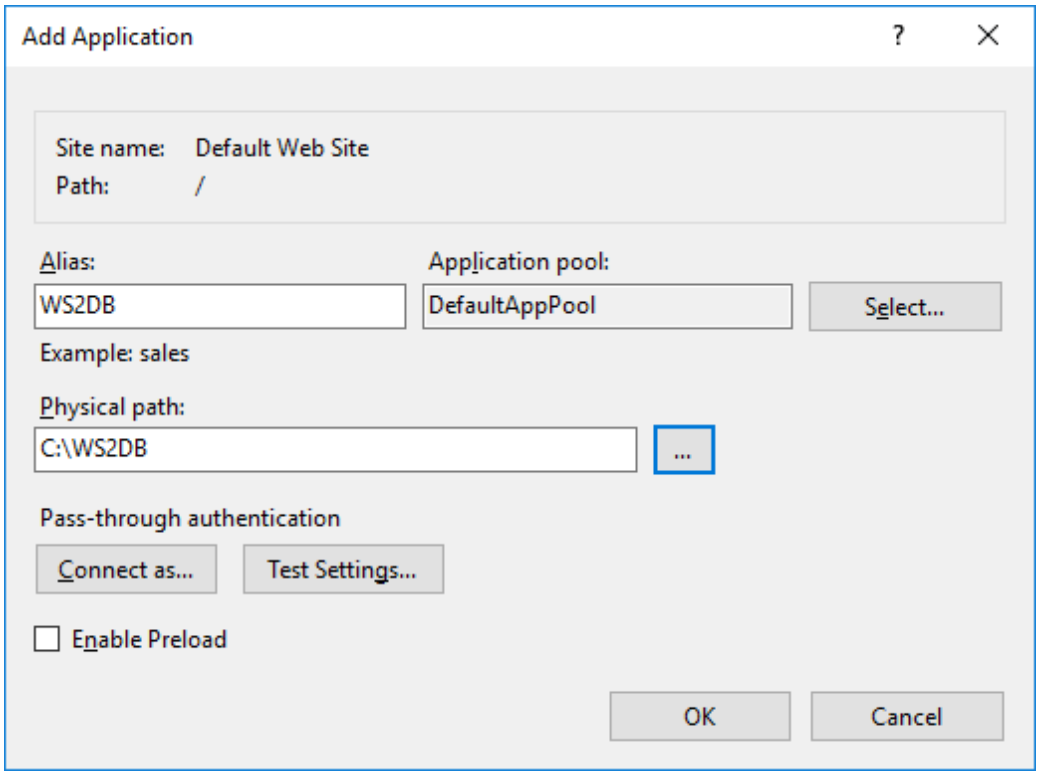


Step 4: Configure the application in IIS Manager

You can add the new application to IIS as follows:

1. In IIS, right-click **Sites | Default Web Site**, and select **Add Application** from the context menu.
2. Under "Alias name", enter a name to identify your web application ("services", in this example).
3. Under "Physical path", enter the path where the application was previously published ("C:

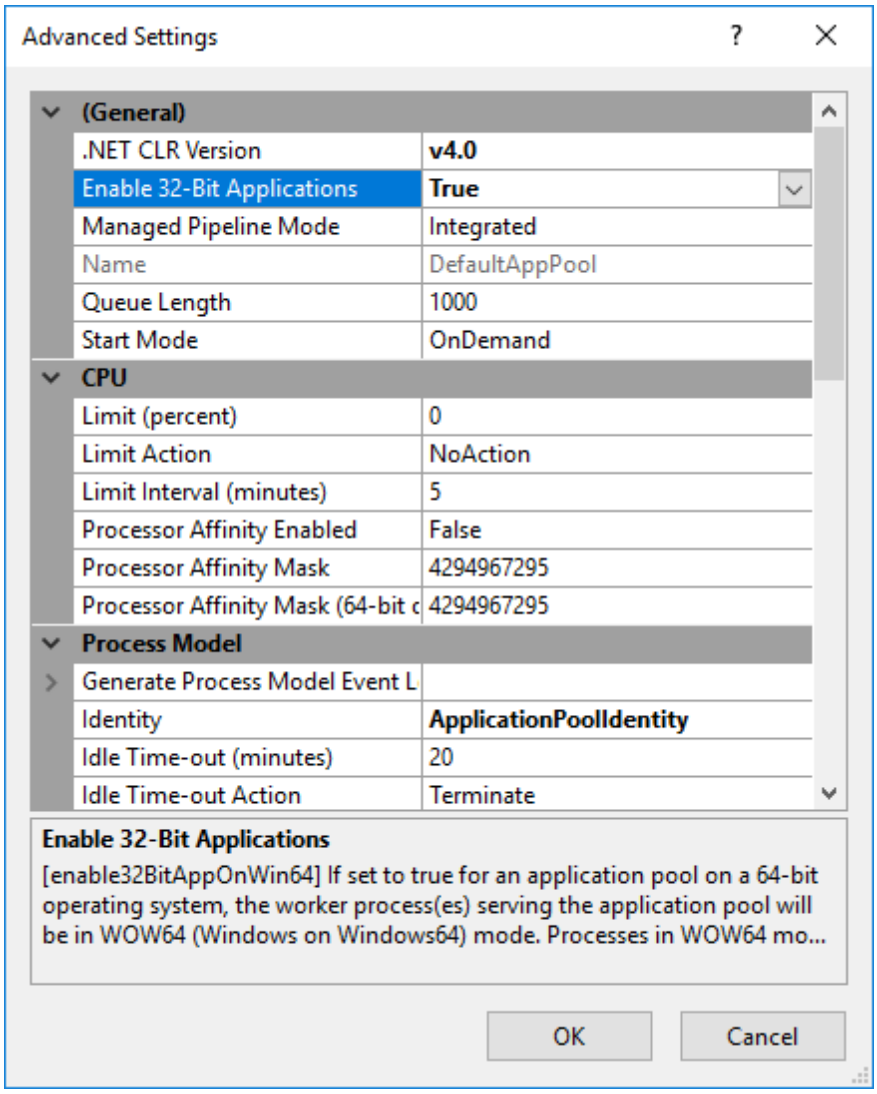
\WS2DB", in this example).



Additionally, for this particular example, it is necessary that 32-bit applications are enabled for the pool where the app is running.

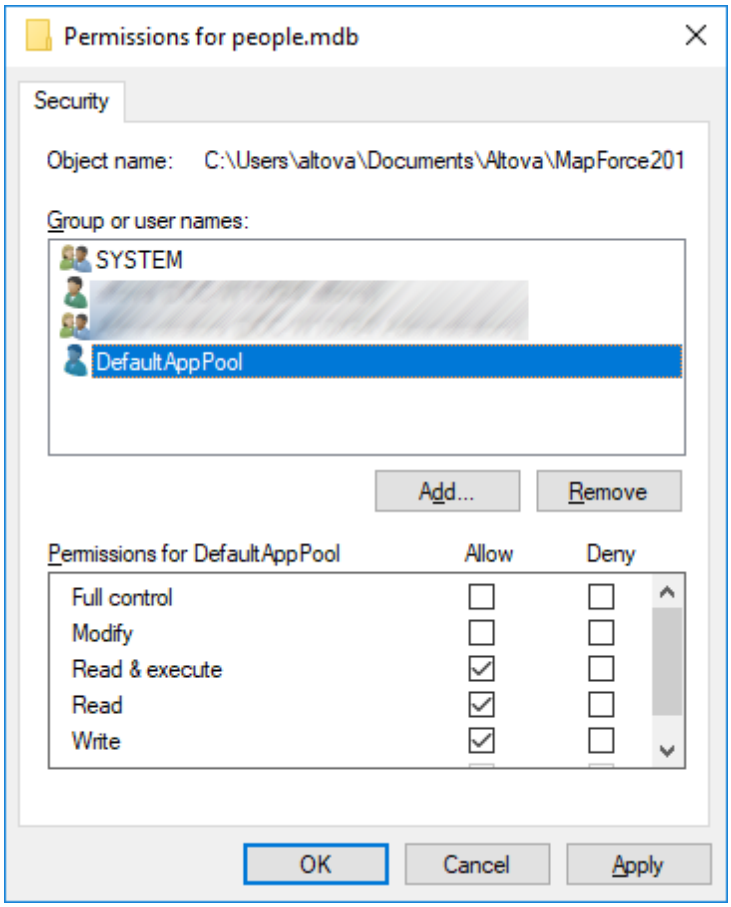
1. In IIS, right-click the application pool where the app was deployed, and select **Advanced Settings** from the context menu.
2. Under **Enable 32-bit Applications**, select the option **True**.

© 2018 Altova GmbH

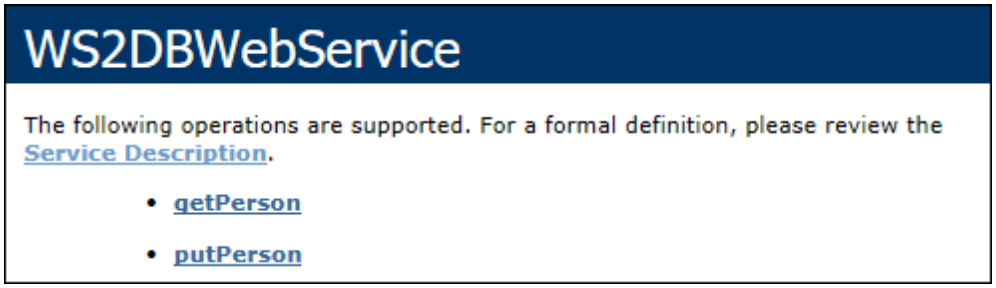


Finally, since this example reads data from a Microsoft Access database, the IIS application pool must have permissions to access the database file. Unless you modified the MapForce project, the database is available at the following path: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\people.mdb**.

1. In Windows Explorer, right-click the **people.mdb** file, and select **Properties**.
2. On the **Security** tab, click **Edit**.
3. Click **Add**, and enter the object name as follows: **IIS AppPool\DefaultAppPool**. Change "DefaultAppPool" to a different value if you deployed the app to a pool other than the default one.
4. Grant the required permissions, and click **Apply**. (In this example, permission to read and write the database file is required.)



To test if the Web service is up and running, access the following URL: `http://localhost/WS2DB`.



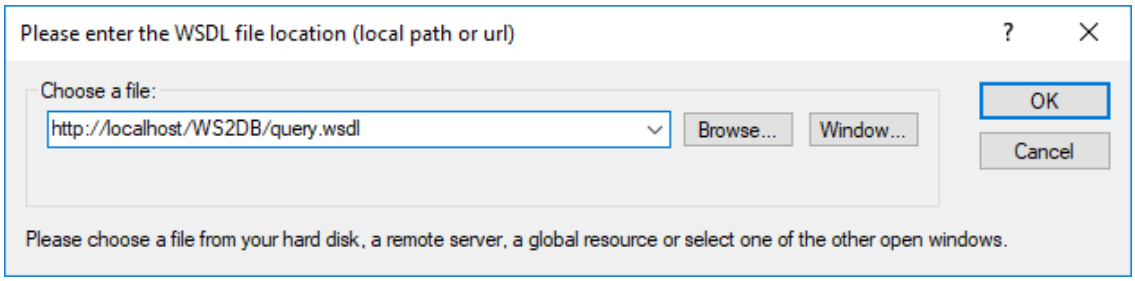
If the Web service details load in the browser, you can proceed to calling the Web service from a client application, as shown below.

Calling the Web service

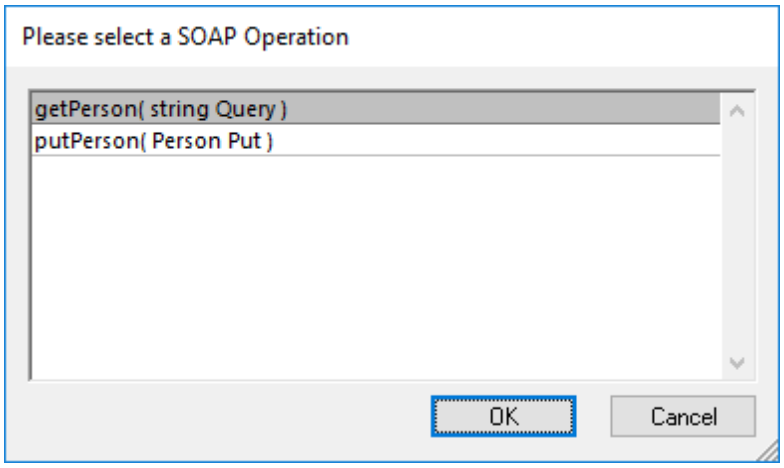
If you followed the steps above literally, then the WSDL of the Web service is available at `http://localhost/WS2DB/query.wsdl`. To call the Web service we just deployed, we will use XMLSpy, although you can use, of course, other SOAP clients as well.

 © 2018 Altova GmbH

Run XMLSpy (Enterprise Edition). On the **SOAP** menu, click **Create New SOAP Request**. When prompted, enter the WSDL of the Web service (in this example, `http://localhost/WS2DB/query.wsdl`).



Click OK. When prompted to select a SOAP Operation, select **getPerson**.



XMLSpy generates a sample request like the one below.

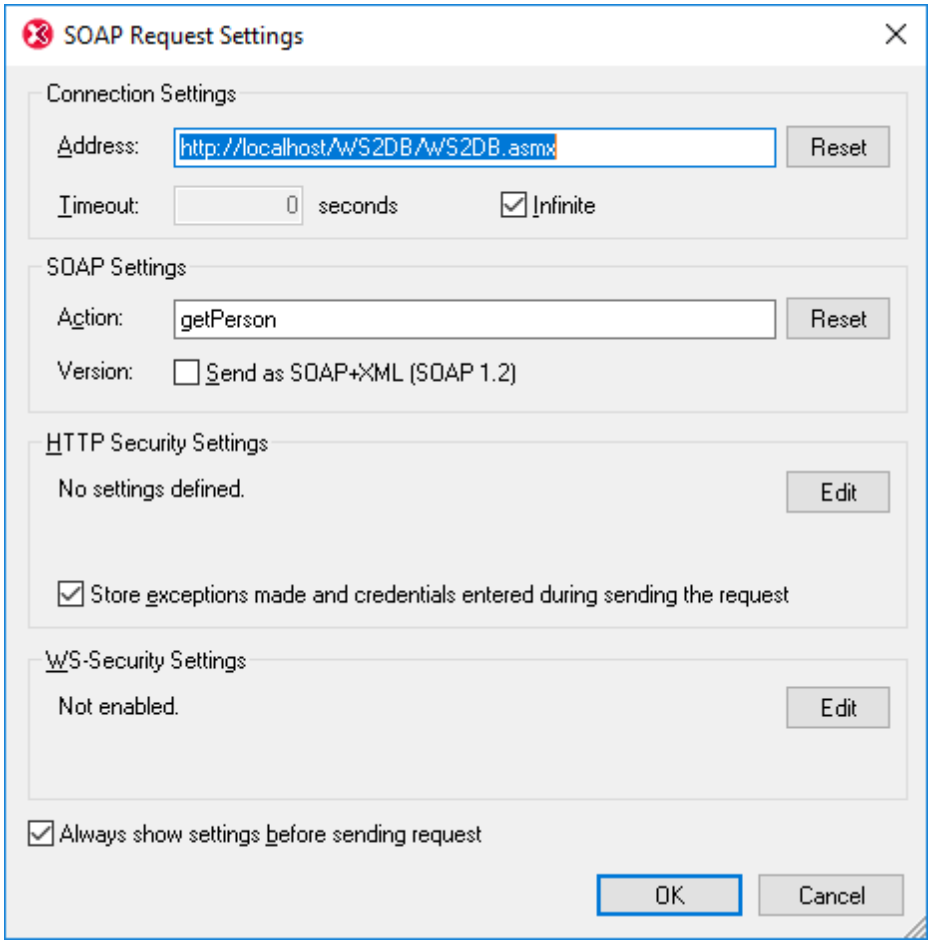
```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/
XMLSchema">
   <SOAP-ENV:Body>
      <getPerson SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/
encoding/">
         <Query xsi:type="xsd:string">String</Query>
      </getPerson>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Locate the <Query> element of the request, and replace "String" with the value you want to supply as input to the Web service. As stated before, this Web service retrieves people details from an Access database. For the scope of this example, in order for the call to actually retrieve some data, "String" must be replaced with "Ro" (in other words, only people whose first or last

name contains "Ro" would be retrieved). The new request now looks as follows:

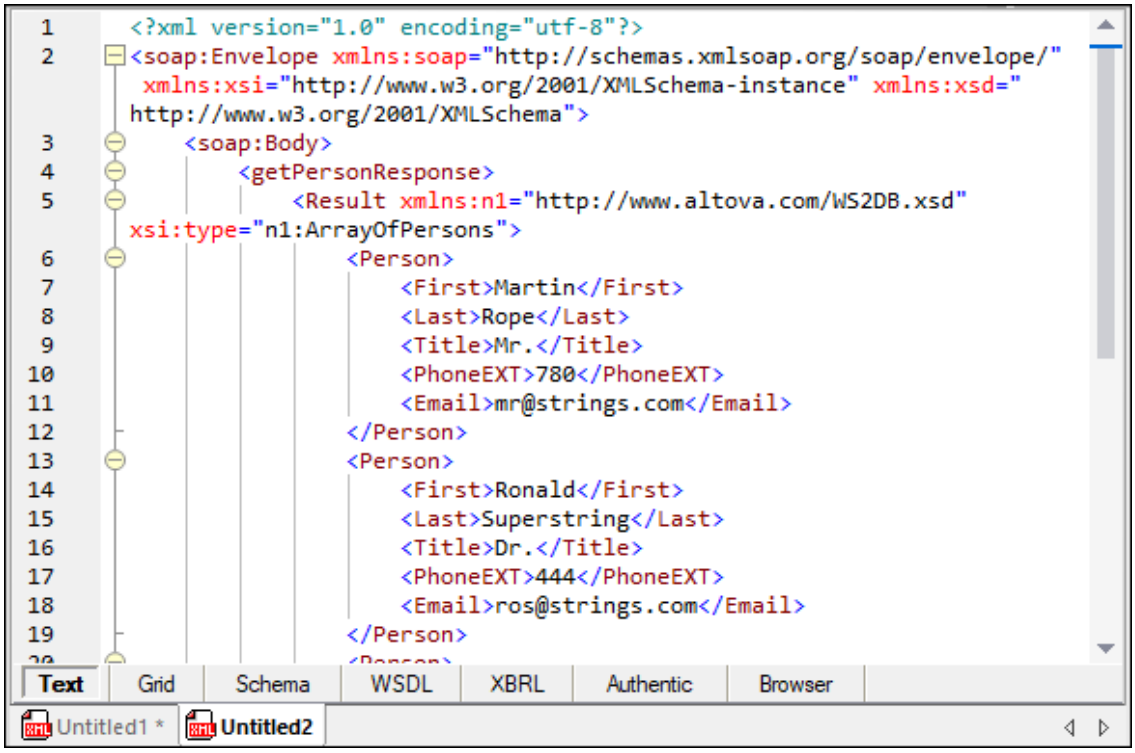
```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/
XMLSchema">
   <SOAP-ENV:Body>
      <getPerson SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/
encoding/">
         <Query xsi:type="xsd:string">Ro</Query>
      </getPerson>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

You can now send the request above to the server. On the **SOAP** menu, click **Send Request to Server**. The SOAP Request Settings dialog box opens.



Click **OK**. XMLSpy initiates the call, and returns the response in the editor. The image below illustrates a successful response.

© 2018 Altova GmbH



Using the same approach as described above, you can also call the **putPerson** operation exposed by this Web service. For step-by-step instructions, see "Calling the Web Service" section in the Java tutorial. Only the WSDL URL is different in the Java tutorial; all other instructions for calling the Web service are identical.

Troubleshooting

The following table lists common problems that you might encounter when building or invoking the Web service, and their solution.

Problem	Solution
Calling the Web service returns the following error: `The 'Microsoft.Jet.OLEDB.4.0' provider is not registered on the local machine."`	1. Build the C# solution in Visual Studio as a 32-bit application. 2. Deploy to IIS. 3. Right-click the IIS application pool where you deployed the application, and select **Advanced Settings**. 4. Under **Enable 32-bit applications**, select **True**.
Calling the Web service returns the following error: `The Microsoft Jet database engine`	• Grant to the IIS application pool permission to read the Access database file, see "Step 4: Configure the application in IIS Manager" above.

Problem	Solution
`cannot open the file 'C:\Usuarios \<usuario>\Documentos\Altova \MapForce2018\MapForceExamples \Tutorial\people.mdb'. It is already opened exclusively by another user, or you need permission to view its data.`	
Calling the Web service returns the following error: `System.Data.OleDb.OleDbException: Operation must use an updateable query.`	• Grant to the IIS application pool permission to write the Access database file, see "Step 4: Configure the application in IIS Manager" above.

© 2018 Altova GmbH

9.6 Ejemplo: generar servicios web SOAP (Java)

This example illustrates how to generate a SOAP Web service with MapForce. You will generate Java program code from a sample MapForce project available at the following path: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Query Person database.mfp**. This project implements a SOAP Web service that exposes two operations: `getPerson` and `putPerson`. This Web service communicates in the background with a Microsoft Access database to retrieve or add person details, respectively, through the operations above. For an example of how to create such projects in MapForce, see Creating Web Service Projects from WSDL Files.

This example requires Java 7 and Apache Ant to compile the generated Java code. The .aar file will be deployed to a running Apache Tomcat server with Axis2 installed.

Tenga en cuenta que, para simplificar, en el ejemplo se emplea una configuración de servidor web mínima que no debe entenderse como preceptiva. En un entorno de producción es muy probable que el servidor web, las directivas de seguridad de la organización y otros factores exijan un enfoque distinto.

Para llamar al servicio web SOAP en este ejemplo se utiliza XMLSpy Enterprise Edition. Si no tiene XMLSpy, necesitará un cliente SOAP para probar el servicio web o, si lo prefiere, deberá escribir código de programa que pueda llamar al servicio web.

Note: As stated before, this Web service example accesses a Microsoft Access database. With Java 7, connectivity to Access databases is possible through the ODBC-JDBC bridge included in Java. However, the ODBC-JDBC bridge has been removed in Java 8, and Oracle recommends using JDBC drivers provided by the vendor, see http://docs.oracle.com/javase/7/docs/technotes/guides/jdbc/bridge.html. Therefore, to run this example in a Java 8 or newer environment, you will need to find and install third-party JDBC drivers that can connect to a Microsoft Access database.

Step 1: Prepare the query.wsdl file

This example Web service originates from the following WSDL file: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\query.wsdl**. By default, the **query.wsdl** file defines the service details for C# and Java in two separate sections, one of which is commented out. Therefore, before generating Java code, locate the following section in the **query.wsdl** file and make sure that it is commented out:

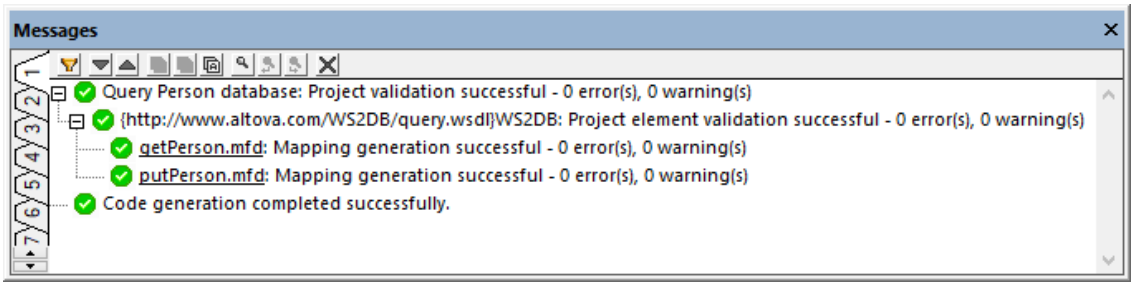
```
<!--service name="WS2DB">
   <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
      <soap:address location="http://localhost/WS2DB/WS2DB.asmx"/>
   </port>
</service-->
```

Also, make sure that the following section (only applicable to Java) is NOT commented out:

```
<service name="WS2DB">
   <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
      <soap:address location="http://localhost:8080/axis/services/WS2DB"/>
   </port>
</service>
```

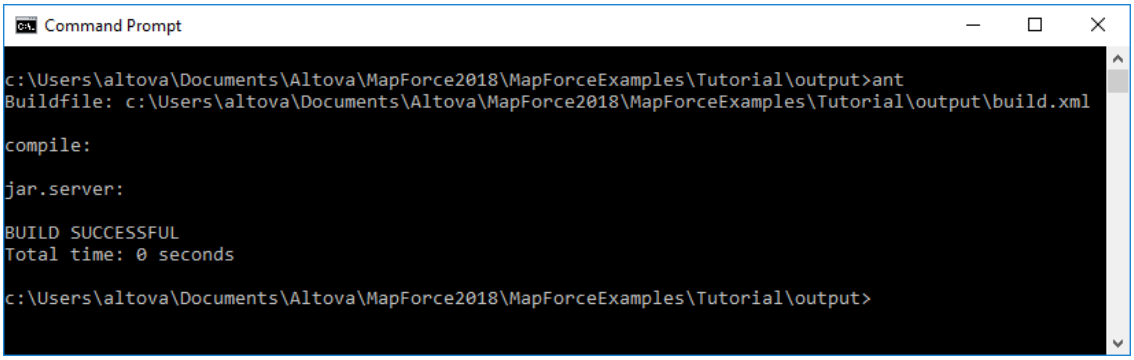
Step 2: Generate Java code and build it

Run MapForce and open the "Query Person database.mfp" project from the folder **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial**. Right-click the project in the Project window, and select **Generate code in | Java**. MapForce generates program code and displays the outcome in the Messages window:



By default, code will be generated in a subdirectory called "output", relative to the project's directory. For the scope of this tutorial, we will leave the default settings as is. However, if you want to change the output directory, see Setting the Code Generation Settings.

To build the Java code, run Ant at the command line in the same directory as the **build.xml** file.



As a result, an Axis Archive (.aar) file called **Query_Person_database.aar** is generated in the output directory.

© 2018 Altova GmbH

Before you can call Ant from any directory as illustrated above, it must be installed and configured correctly. Here is a quick summary (for more details, check the Ant manual from http://ant.apache.org):

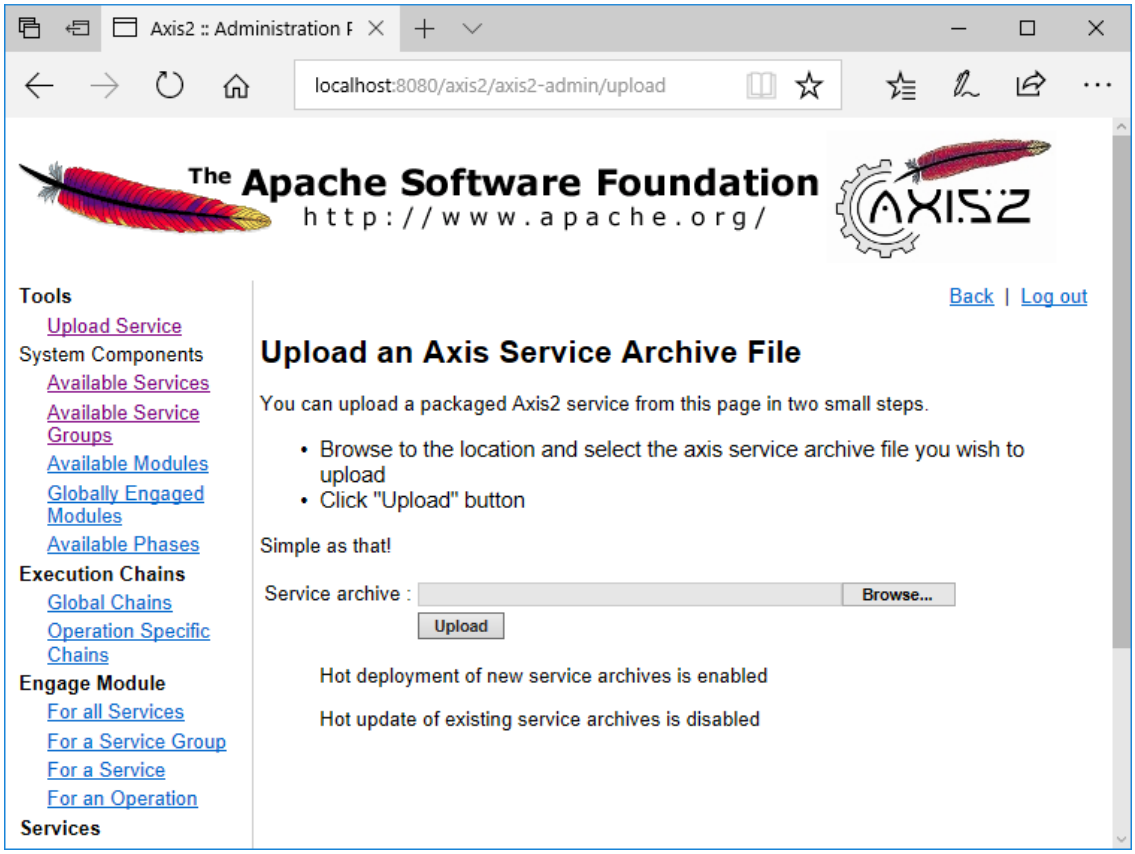
- The `PATH` environment variable must include the path to the "bin" subdirectory of Ant root directory.
- The `ANT_HOME` environment variable must be set to the directory where Ant is installed.
- The `JAVA_HOME` environment variable must be set to the directory where Java JDK is installed.
- Check the Ant is running by typing "ant" at the command line. The message "Buildfile: build.xml does not exist!" means that Ant is configured and is looking for a build file.

Step 3: Deploy the Web service to Axis 2

In this example, Axis2 v1.7.6 runs on Apache Tomcat 8.5.0 installed on a local machine. In brief, Axis2 can be installed and deployed to a running Tomcat server as follows (for more details, refer to Axis2 documentation):

1. Download the Axis2 WAR (Web Archive) distribution from http://axis.apache.org/axis2/java/core/download.html.
2. Unpack the downloaded package to a local directory.
3. Copy the **axis2.war** file to Tomcat's directory **%CATALINA_HOME%\webapps**.
4. Check that Axis2 is running, by opening: `http://localhost:8080/axis2`

Open the Axis2 Web administration page and upload the .aar file generated in the previous step.



A new Web service called "WS2DB" should now be available and active in the Axis2 administration page.

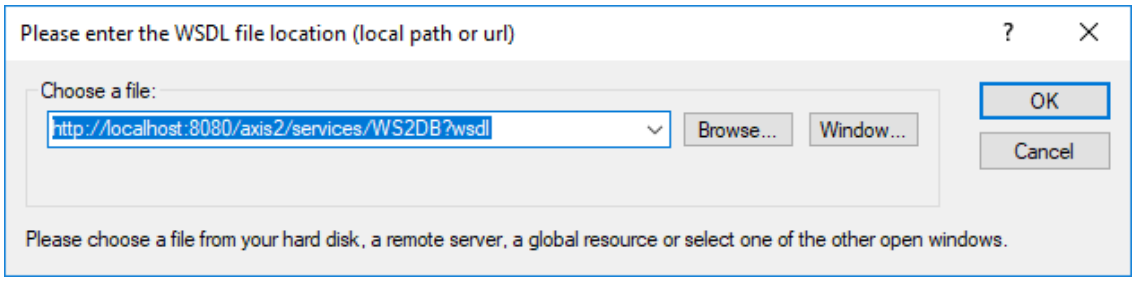


 © 2018 Altova GmbH

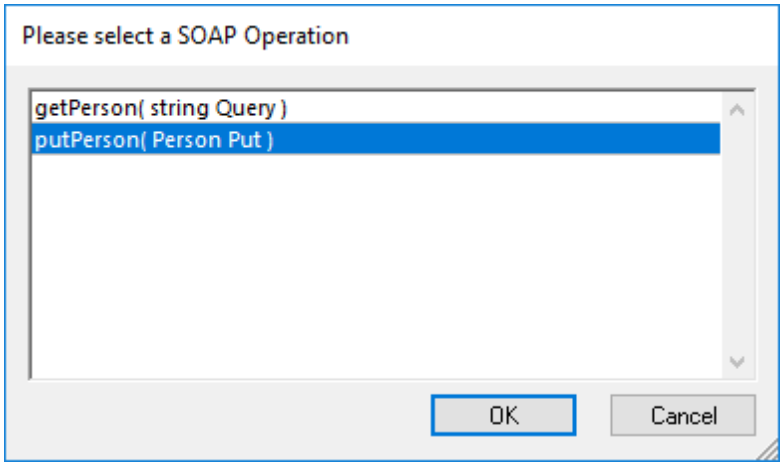
Calling the Web service

If you followed the steps above literally, then the WSDL of the Web service is available at `http://localhost:8080/axis2/services/WS2DB?wsdl`. To call the Web service we just deployed, we will use XMLSpy, although you can use, of course, other SOAP clients as well.

Run XMLSpy (Enterprise Edition). On the **SOAP** menu, click **Create New SOAP Request**. When prompted, enter the WSDL of the Web service (in this example, `http://localhost:8080/axis2/services/WS2DB?wsdl`).



Click OK. When prompted to select a SOAP Operation, select **putPerson**.



XMLSpy generates a sample request like the one below.

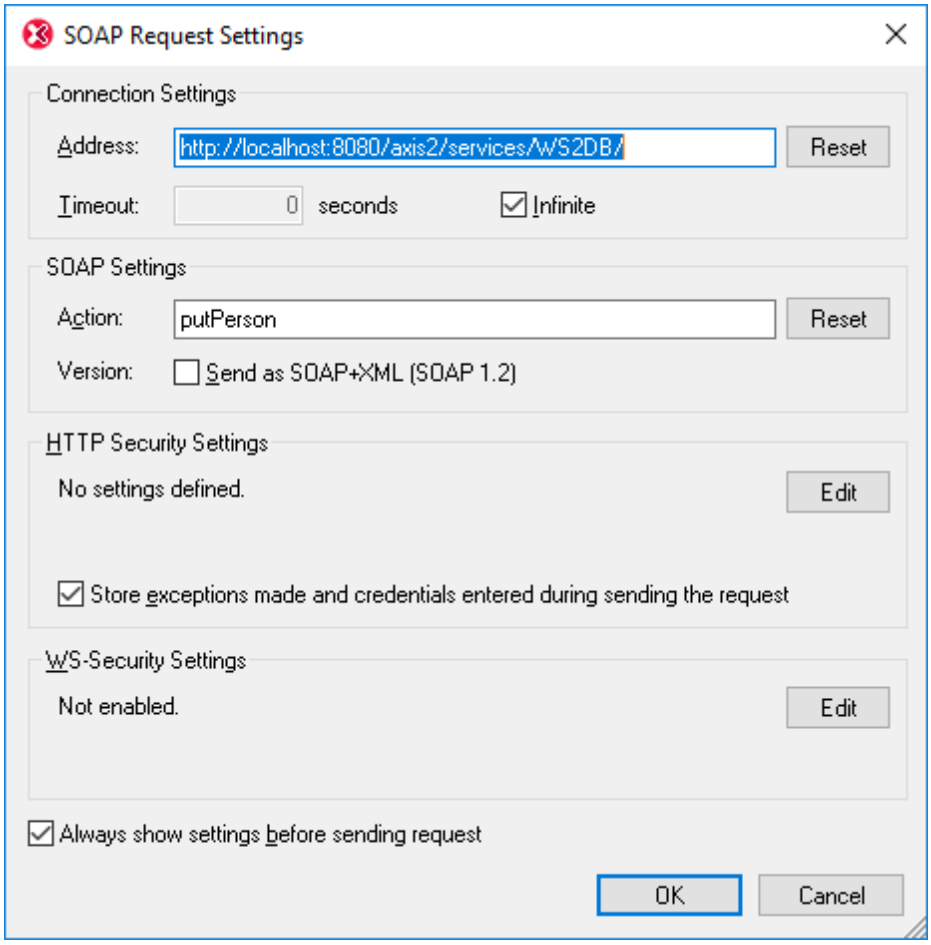
```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/
XMLSchema" xmlns:m0="http://www.altova.com/WS2DB.xsd">
   <SOAP-ENV:Body>
      <putPerson SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/
encoding/">
         <Put xsi:type="m0:Person">
            <First xsi:type="xsd:string">String</First>
            <Last xsi:type="xsd:string">String</Last>
            <Title xsi:type="xsd:string">String</Title>
            <PhoneEXT xsi:type="xsd:string">String</PhoneEXT>
            <Email xsi:type="xsd:string">String</Email>
         </Put>
      </putPerson>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Replace "String" with the person details you want to supply as input to the Web service. For the scope of this example, you could use a request like the one below:

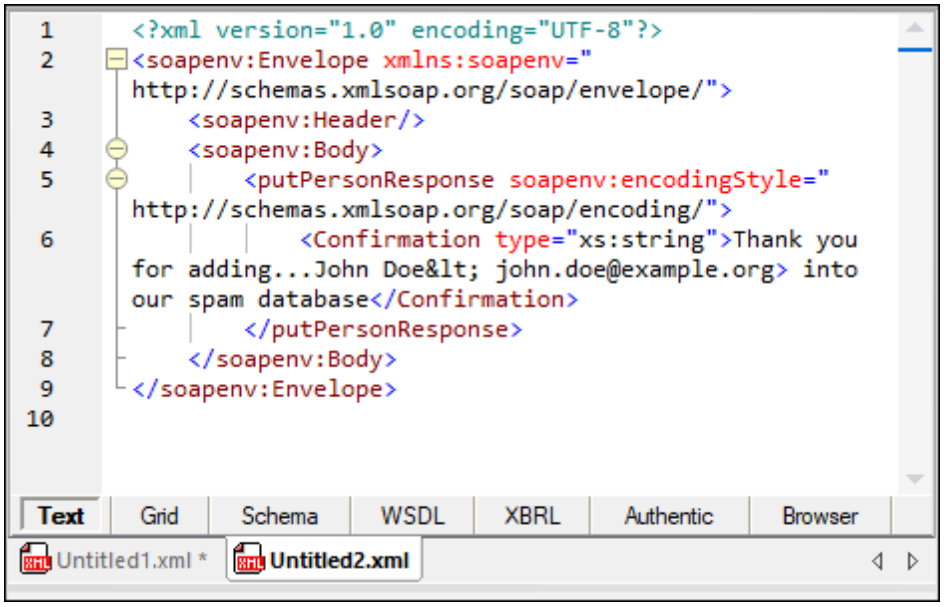
```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsi="http://
www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/
XMLSchema" xmlns:m0="http://www.altova.com/WS2DB.xsd">
   <SOAP-ENV:Body>
      <putPerson SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/
encoding/">
         <Put xsi:type="m0:Person">
            <First xsi:type="xsd:string">John</First>
            <Last xsi:type="xsd:string">Doe</Last>
            <Title xsi:type="xsd:string">Mr</Title>
            <PhoneEXT xsi:type="xsd:string">123</PhoneEXT>
            <Email xsi:type="xsd:string">john.doe@example.org</Email>
         </Put>
      </putPerson>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

You can now send the request above to the server. On the **SOAP** menu, click **Send Request to Server**. The SOAP Request Settings dialog box opens.

© 2018 Altova GmbH



Click **OK**. XMLSpy initiates the call, and returns the response in the editor. The image below illustrates a successful response.



Using the same approach as described above, you can also call the **getPerson** operation exposed by this Web service. For step-by-step instructions, see "Calling the Web Service" section in the C# tutorial. Only the WSDL URL is different in the C# tutorial; all other instructions for calling the Web service are identical.

 © 2018 Altova GmbH

Altova MapForce 2018 Enterprise Edition

Llamar a servicios web

10 Llamar a servicios web

MapForce permite llamar a servicios web desde las asignaciones de datos directamente. Esto significa que puede insertar una llamada a servicio web (o, como se denomina en MapForce, una *función de servicio web*) en una asignación de datos, conectar componentes de entrada y salida a la llamada al servicio web y consumir el resultado según proceda (por ejemplo, puede consultar la vista previa en la ventana de resultados de MapForce, pasar el resultado a otro componente o escribirlo en un archivo). Esta característica convierte hace de MapForce un potente cliente de servicios web que se puede configurar fácilmente desde la interfaz gráfica del usuario.

MapForce permite llamar a servicios web de tipo WSDL y REST. Por tanto, cuando añada un servicio web a la asignación de datos podrá hacerlo de dos maneras distintas:

- Si se trata de un servicio web de tipo WSDL, deberá aportar a MapForce el archivo WSDL (lenguaje de definición de servicio web) del servicio web al que se debe llamar. MapForce utiliza el archivo WSDL para comunicarse con el servidor web. Si el archivo WSDL implementa varios servicios, extremos y operaciones, podrá seleccionarlos y actualizarlos en MapForce directamente.
- Si se trata de un servicio web que no sea WSDL, deberá introducir a mano los detalles del servicio web. Es decir, deberá indicar la URL, el método de solicitud (p. ej. GET, POST o PUT), la estructura de solicitud y de respuesta (como esquema XML o JSON) y los parámetros. Además, si tiene el archivo WADL (lenguaje de definición de aplicación web) del servicio web al que se debe llamar, podrá importar la definición de este archivo. Y si tiene un ejemplo de URL del servicio web, puede configurar MapForce para que analice la URL y extraiga automáticamente consultas, plantillas y parámetros de matriz para que no tengan que definirse a mano.

Para llamar a un servicio web el lenguaje (motor de ejecución) de la asignación debe ser BUILT-IN (véase Seleccionar un lenguaje de transformación). Las llamadas a servicios web creadas con MapForce se pueden ejecutar en MapForce directamente o en otro equipo o plataforma con ayuda de MapForce Server, desde la interfaz de la línea de comandos o desde la API (http://www.altova.com/es/mapforce/mapforce-server.html).

En el caso de los servicios web de tipo WSDL también podrá elegir el lenguaje de transformación C# o Java. Por tanto, las llamadas a servicios web de tipo WSDL también se pueden ejecutar desde el código C# o Java generado con MapForce.

Nota: en C# y Java se admite la opción de autenticación HTTPS predeterminada (es decir, MapForce revisa el certificado de servidor). Pero en estos lenguajes no podrá configurar más opciones HTTPS (es decir, certificados de cliente).

Cuando seleccione el lenguaje de transformación BUILT-IN podrá llamar a servicios web no seguros y a servicios web que exijan autenticación HTTP básica o autenticación HTTPS (TLS) con certificados digitales. MapForce también ofrece opciones para comprobar si el certificado digital del servidor web al que se llama es válido y de confianza.

Algunos servicios web exigen autenticación HTTP preventiva (esta opción también se puede configurar en MapForce directamente). Además, para los servicios de tipo WSDL se ofrece compatibilidad básica con WS-Security. Es decir, en MapForce puede configurar el contenido de los tokens de seguridad SOAP **UsernameToken** y **Timestamp**.

© 2018 Altova GmbH

Si el servidor web no responde, puede configurar en MapForce el intervalo tras el que la llamada debe expirar. En MapForce también puede controlar otras respuestas de error del servidor (p. ej. URL incorrectas).

Cuando se trate de servicios web de tipo WSDL podrá usar estos protocolos en MapForce:

SOAP 1.1, SOAP 1.2	Son compatibles los estilos *RPC/encoded* y *document/literal*. Si el servidor web devuelve un error WSDL, se detiene la ejecución de la asignación de datos. Para este tipo de casos puede insertar un componente de excepción en el área de asignación que se ocupará del error. Si el servidor web devuelve un error no WSDL, se detiene la ejecución de la asignación y se devuelve un mensaje de error (o el mensaje aparece en pantalla si está consultando la vista previa de resultados).
HTTP GET	Es compatible el estilo *url-encoded*.
HTTP POST	Son compatibles los estilos *url-encoded* y *text/xml*.

Cuando se trate de servicios web genéricos (de tipo REST) se admiten llamadas realizadas a través del protocolo HTTP usando verbos HTTP (p. ej. GET, POST, PUT). Recuerde que MapForce admite contenido XML y JSON en el cuerpo de la solicitud y de la respuesta de los servicios web. Por ahora no se admite ningún otro tipo de contenido.

10.1 Agregar llamadas a servicios web (de tipo REST)

En este apartado explicamos cómo llamar a un servicio web genérico (no WSDL) desde una asignación de datos. Esto incluye una amplia categoría de servicios web que siguen parcial o totalmente una estructura denominada *REST* (y que suelen denominarse servicios web *RESTful* o de tipo REST).

Los servicios web HTTP genéricos suelen llevar estructuras de solicitud o respuesta personalizadas en la parte del cuerpo del mensaje. MapForce admite datos JSON o XML en el cuerpo de la solicitud y de la respuesta. Por tanto, desde el punto de vista de MapForce, puede llamar a prácticamente cualquier tipo de servicio web HTTP que requiera o devuelva estructuras XML o JSON siempre y cuando pueda aportar la estructura a MapForce en forma de esquema JSON, XML o DTD. MapForce también aceptará como estructura un archivo XML con una referencia de esquema válida.

La estructura del servicio web puede estar publicada por el proveedor de servicio como esquema XML o JSON o en un lenguaje formal como WADL (o incluso en una especificación de lectura humana). Si ya tiene el esquema XML o JSON de la solicitud/respuesta, crear la llamada al servicio web será muy sencillo. Si tiene la definición de servicio web como archivo WADL, podrá importarla y realizar los ajustes que sean necesarios. Por último, si tiene un archivo de instancia XML o JSON de ejemplo pero no tiene un archivo de esquema, podrá crear o generar el esquema con XMLSpy (https://www.altova.com/es/xmlspy.html). Si lo necesita, puede convertir la instancia XML en instancia JSON y viceversa desde Altova XMLSpy.

Nota: WADL no ofrece ningún método estándar para definir estructuras JSON, solo estructuras XML.

Agregar una llamada a un servicio web genérico

1. Haga clic en el comando de menú **Insertar | Función de servicio web** (también puede hacer clic en el botón **Insertar función de servicio web**  de la barra de herramientas).
2. En el grupo de opciones *Definición del servicio* elija la opción *Manual*.
3. Si tiene un archivo WADL que describa el servicio, haga clic en **Importar desde archivo WADL...** y seleccione el archivo (*véase* *Importar información de servicio web desde un archivo WADL*).
4. Seleccione el método de solicitud HTTP que se debe utilizar para llamar al servicio web. Puede seleccionar un valor de la lista o escribir el nombre del método que desea usar. Debe tener en cuenta el uso de mayúsculas y minúsculas en los nombres de métodos HTTP.
5. Introduzca la URL del servicio web. Si la URL del servicio web usa parámetros, debe tener en cuenta que:
   a. si llama a un servicio web con los parámetros de estilo "Plantilla" o "Matriz", ponga los parámetros entre llaves. Por ejemplo: `http://ejemplo.org/api/products/{id}`. Después defina la configuración de cada parámetro en la tabla *Parámetros*. En tiempo de ejecución MapForce procesa los nombres de parámetro que están entre llaves y produce la URL final que incluye valores reales.
   b. si llama a un servicio web con parámetros URL "Consulta" (por ejemplo `http://example.org/api/products?sort=asc&category=1&page=1`), no introduzca la parte de la consulta en el cuadro de texto *URL*. Debe definir los parámetros en la

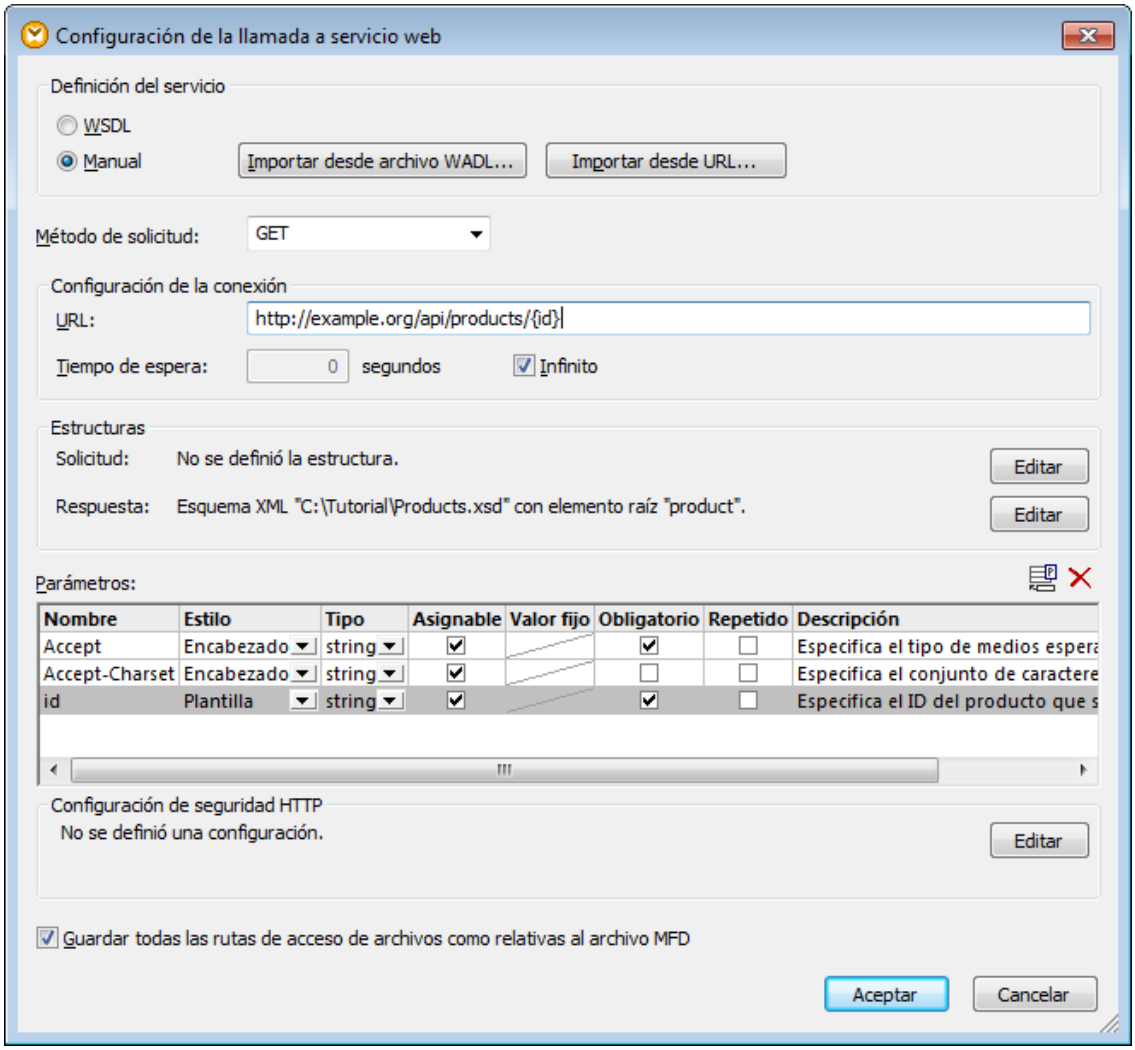
tabla *Parámetros* solamente.
Consulte el apartado Definir parámetros de servicios web para ver un ejemplo.

6. En el grupo de opciones *Tiempo de espera* tiene la opción de introducir el plazo de tiempo en segundos tras el cual la conexión expirará si el servidor no responde.
7. Si el método HTTP requiere o devuelve la parte del cuerpo de la solicitud (como estructura XML o JSON), haga clic en el botón **Editar** situado en el grupo de opciones *Estructura* y busque el esquema JSON o XML de la parte del cuerpo de la solicitud (*véase Definir la estructura de respuesta y solicitud del servicio web*).
8. En el grupo de opciones *Parámetros* defina los parámetros del servicio web. También puede hacer clic en **Importar desde URL** para importar los parámetros desde una URL de ejemplo del servicio web y rellenar la tabla *Parámetros* automáticamente (*véase Importar parámetros de servicio web desde una URL*). Tras importar los parámetros desde la URL, podrá modificar el contenido de la tabla *Parámetros* si es necesario. Para más información consulte el apartado Definir los parámetros del servicio web.

Nota: para especificar encabezados de solicitud personales, añada un parámetro con el estilo Encabezado y asegúrese de que el nombre del parámetro coincida con el nombre del encabezado y que el valor del parámetro coincida con el valor del encabezado. Además, si necesita pasar el valor del encabezado de la solicitud desde la asignación propiamente dicha, entonces utilice el tipo de parámetro "Asignable".

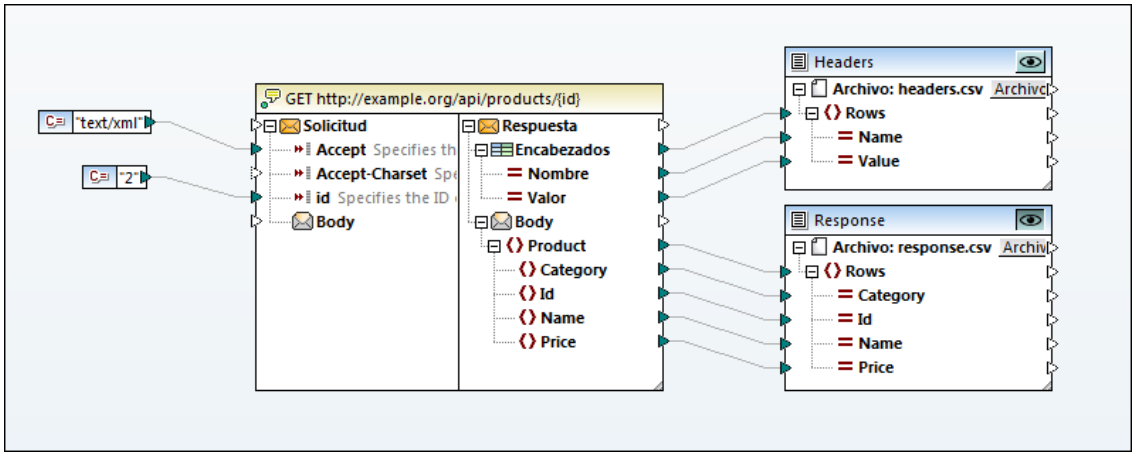
9. Si el servicio web utiliza autenticación HTTP o seguridad basada en certificado, haga clic en **Editar** (en el grupo de opciones *Configuración de seguridad HTTP*) y rellene los campos necesarios (*véase Configurar seguridad HTTP*).

Cuando haga clic en **Aceptar** se añadirá un componente de servicio web nuevo al área de asignación.



La asignación que aparece a continuación llama a un servicio web para recuperar un producto en función de su ID con ayuda de una solicitud GET. En este ejemplo concreto, el ID que se da en la solicitud HTTP tiene el valor de constante 2. Sin embargo, también puede ser un parámetro para la asignación (*véase* *Pasar parámetros a la asignación*) o puede venir dado por cualquier componente que sea compatible con MapForce. Además del parámetro `id`, la solicitud contiene el encabezado `Accept: text/xml`. Los encabezados que se deben usar en la solicitud y sus valores se pueden configurar con total flexibilidad (*véase* *Definir los parámetros del servicio web*).

 © 2018 Altova GmbH



Esta asignación se configuró para tener dos resultados:

1) Los encabezados como valores separados por comas.
2) El cuerpo de la respuesta como valores separados por comas.

Para seleccionar cuál de estos dos resultados aparece en la vista previa active el botón situado en la esquina superior derecha de cada componente de destino.

Observe que el componente de servicio web tiene dos partes: la solicitud y la respuesta. La parte **Solicitud** permite aportar datos de la asignación al servicio web, mientras que la parte **Respuesta** permite acceder a los datos que devuelve el servicio web y asignarlos a otros formatos. La estructura de la solicitud y de la respuesta depende de los parámetros (así como de la estructura XML o JSON) que se definiera en el cuadro de diálogo "Configuración de la llamada a servicio web" (véase Definir la estructura de la respuesta y solicitud del servicio web).

Para llamar al servicio web con determinados parámetros de solicitud, dibuje conexiones entre cualquier tipo de componente compatible con MapForce (p. ej. un archivo XML o JSON) y la parte **Solicitud**. Igualmente, para asignar datos devueltos por el servicio web a otro formato, dibuje conexiones entre la parte **Respuesta** y cualquier tipo de componente compatible. Si acaba de empezar a utilizar MapForce y necesita información sobre cómo dibujar conexiones de asignación consulte la sección Trabajar con conexiones.

Los encabezados de respuesta devueltos por el servicio web también se pueden asignar si son encabezados adicionales (los que no empiezan por "Content"). Los valores de encabezado están disponibles en el componente de servicio web y pueden verse en el elemento **Encabezados**, que contiene dos elementos secundarios: **Nombre** y **Valor**. Esta estructura hace de secuencia y permite asignar datos de los encabezados que devuelve el servicio web. Para asignar datos desde los encabezados de la respuesta a otros formatos compatibles con MapForce, conecte el nodo de estructura **Encabezados** con una secuencia de destino en el área de asignación. Por ejemplo, si conecta la secuencia **Encabezados** (y sus secundarios) con una secuencia **Filas** (y sus secundarios) de un componente CSV, un encabezado correspondería a una fila del archivo CSV.

Las asignaciones de datos que contengan llamadas a servicios web HTTP genéricas se pueden ejecutar igual que la mayoría de asignaciones de datos:

- A mano, desde MapForce, al hacer clic en el panel *Resultados*. En este caso el resultado de la llamada puede verse inmediatamente en el panel *Resultados*.
- Desde la línea de comandos o con llamadas a la API con MapForce Server (http://www.altova.com/es/mapforce/mapforce-server.html). Antes será necesario compilar la asignación en un archivo de ejecución (*véase Compilar asignaciones en archivos de ejecución de MapForce Server*).
- Desde un trabajo de FlowForce Server que ejecute MapForce Server (http://www.altova.com/es/flowforce.html). Consulte el apartado Implementar asignaciones en FlowForce Server para obtener más información.

La ejecución de la asignación fallará y enviará un mensaje de error cuando:

- La llamada al servicio web devuelva un código de estado HTTP superior a 299.
- La respuesta XML o JSON no se pueda analizar.
- No sea posible llamar al servicio web por un error de conexión o problemas de resolución del DNS.

Para aprender a crear llamadas a servicios web genéricos y ver instrucciones paso a paso consulte estos apartados:

- Ejemplo: llamar a un servicio web de tipo REST
- Ejemplo: asignar datos de un canal RSS

Importar información del servicio web desde un archivo WADL

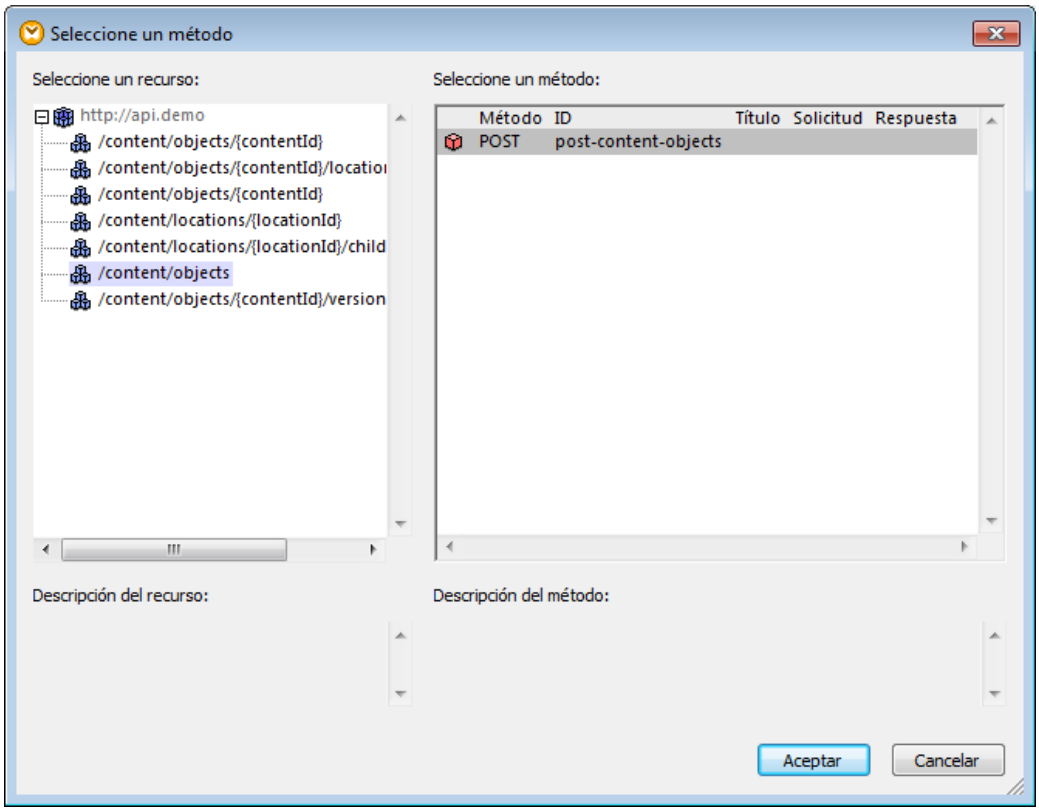
El lenguaje de descripción de aplicaciones web (WADL) sirve para representar el contenido de las respuestas y solicitudes utilizadas en una aplicación web, incluidos los servicios web RESTful. MapForce permite importar información de servicios web a partir de la versión 2009 de WADL (http://www.w3.org/Submission/2009/SUBM-wadl-20090831/).

Si el servicio web tiene un archivo WADL, puede importar la definición de servicio web desde este archivo en lugar de introducirla a mano. Cuando importe información de un archivo WADL, los parámetros y la estructura del cuerpo del mensaje XML se rellenarán automáticamente en el cuadro de diálogo "Configuración de la llamada a servicio web". Una vez finalizada la importación, podrá modificar los parámetros y la estructura de la respuesta/solicitud a mano. Estos cambios no afectarán al archivo WADL subyacente.

Para importar información del servicio web desde un archivo WADL:

1. Abra el cuadro de diálogo "Configuración de la llamada a servicio web" (*véase Configurar llamadas a servicios web*).
2. En el grupo de opciones *Definición del servicio* seleccione la opción *Manual*.
3. Haga clic en **Importar desde archivo WADL** y navegue hasta el archivo WADL.

© 2018 Altova GmbH



4. En el cuadro de diálogo seleccione el recurso y el método y después haga clic en **Aceptar**.

Importar parámetros de servicio web desde una URL

Si la URL del servicio web utiliza parámetros, podrá configurar MapForce para que analice la URL e importa todos los parámetros automáticamente. Cuando utilice esta opción, tenga en cuenta que todos los parámetros de plantilla, matriz y consulta que se extraigan de la URL aparecerán en la tabla *Parámetros* del cuadro de diálogo "Configuración de la llamada a servicio web", donde podrá manipularlos y editarlos.

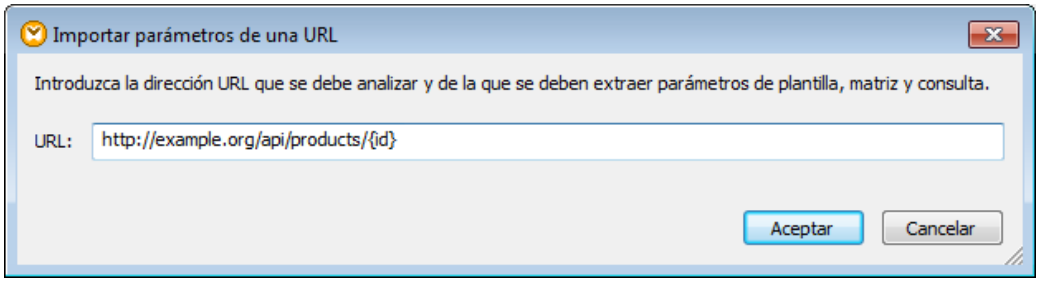
Ejemplo de dirección URL con parámetros de plantilla: `http://example.org/api/products/{id}`
Ejemplo de dirección URL con parámetros de matriz: `http://example.org/api/products;sort=asc;category=1;page=1`
Ejemplo de dirección URL con parámetros de consulta: `http://example.org/api/products?sort=asc&category=1&page=1`

Nota: las direcciones URL deben empezar con "http://" o "https://". De lo contrario no se pueden analizar.

Para importar parámetros de servicio web desde una URL:

1. Abra el cuadro de diálogo "Configuración de la llamada a servicio web" (*véase Configurar llamadas a servicios web*).
2. En el grupo de opciones *Definición del servicio* seleccione la opción *Manual*.

3. Haga clic en **Importar desde una URL**.



4. Introduzca o pegue la URL del servicio web en el cuadro de texto y haga clic en **Aceptar**.

Definir los parámetros del servicio web

Cuando necesite llamar a un servicio web con parámetros de URL, estos parámetros deben estar explícitamente definidos en la tabla Parámetros del cuadro de diálogo "Configuración de la llamada a servicio web". La tabla *Parámetros* también sirve para definir encabezados personales en la solicitud de servicio web, tal y como puede ver en los ejemplos que aparecen más abajo.

Puede introducir parámetros a mano o, si lo prefiere, puede importarlos desde un archivo WADL o desde una URL (*véase Importar información de servicio web desde un archivo WADL e Importar parámetros de servicio web desde una URL respectivamente*). Los parámetros importados aparecerán en la tabla *Parámetros,* donde podrá manipularlos y editarlos.

Para agregar o quitar parámetros de servicio web a mano:

1. Abra el cuadro de diálogo "Configuración de la llamada a servicio web" (*véase Configurar llamadas a servicios web*).
2. En el grupo de opciones *Definición del servicio* seleccione la opción *Manual*.
3. Utilice los botones **Agregar parámetro** ( ) y **Eliminar parámetro** ( ) según corresponda.

A continuación describimos las columnas que componen la tabla *Parámetros*:

Nombre	Nombre del parámetro de URL. El nombre de parámetro debe ser único y debe estar formado por letras, dígitos, puntos, guiones y guiones bajos. Los nombres de parámetro no pueden incluir espacios.
Estilo	Sintaxis (estilo) del parámetro de URL. Utilice el estilo *Encabezado* para agregar un parámetro al encabezado HTTP cuando se llame al servicio web. Por ejemplo, si añade un parámetro llamado `Accept` con el valor `text/xml` será igual que especificar el encabezado `Accept: text/xml`, que informa al servidor web de que MapForce espera que la respuesta esté en formato XML. Para más información sobre los encabezados HTTP consulte http://www.iana.org/assignments/message-headers/message-headers.xhtml. Utilice el estilo *Consulta* para parámetros URL que definan pares clave-valor en el formato **`?clave=valor&clave=valor`**

 © 2018 Altova GmbH

	Por ejemplo: `http://example.org/api/products?sort=asc&category=1&page=1` Utilice el estilo *Plantilla* para parámetros de URL que vayan entre llaves. Por ejemplo: `http://example.org/api/products/{id}` . En estos casos, MapForce añade caracteres de escape a los valores siguiendo las normas RFC 6570 (http://tools.ietf.org/html/rfc6570). Utilice el estilo *Matriz* para parámetros URL que definan pares clave-valor usando en el formato **`;clave=valor;clave=valor;`** Por ejemplo: `http://example.org/api/products;sort=asc;category=1;page=1;` Para usar parámetros de matriz booleanos, elija el estilo *Matriz* y después el tipo *Booleano* (consulte también la opción siguiente).
Tipo	Especifica el tipo de datos del parámetro (cadena, entero, fecha, etc.). Puede ser cualquier tipo XML Schema, pero recuerde que, excepto las cadenas, todos los valores se pueden convertir en cadena cuando tiene lugar la llamada al servicio web. No obstante, es importante establecer un tipo de datos si desea que MapForce muestre mensajes de error de conversión cuando intente llamar a un servicio web con valores incorrectos.
Asignable	Marque esta casilla si quiere pasar valores a este parámetro desde la asignación. Esta opción y la opción *Valor fijo* se excluyen mutuamente.
Valor fijo	Especifica el valor del parámetro. Solamente es relevante si el parámetro tiene un valor constante. No es relevante si el parámetro es asignable (ver opción anterior).
Obligatorio	Marque esta casilla si el servicio web requiere este parámetro. En el caso de parámetros obligatorios y asignables, MapForce fuerza la validación (es decir, si el parámetro no tiene un valor MapForce muestra un mensaje de error).

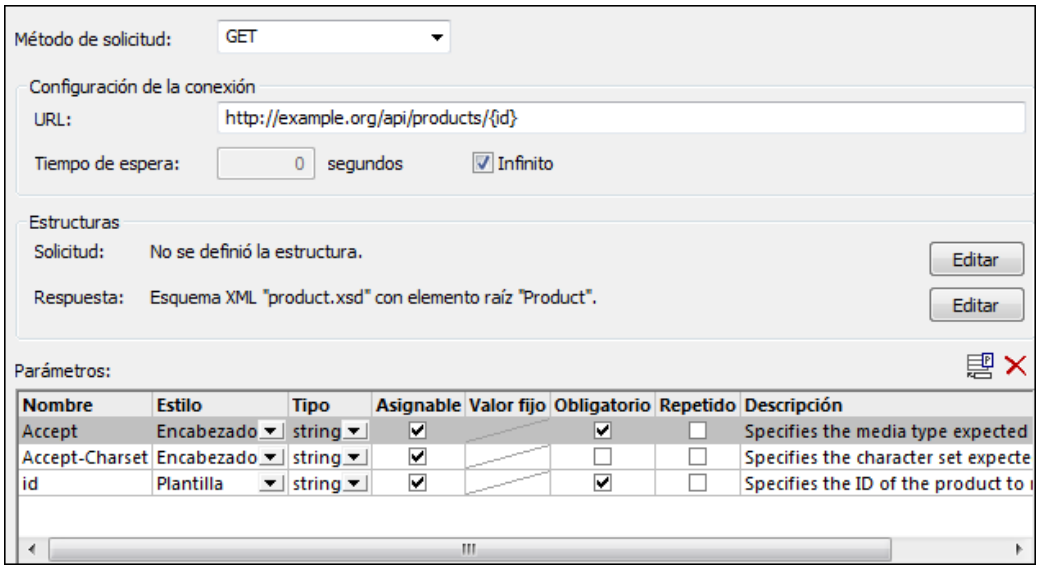
Repetido	Especifica si el parámetro tiene un solo valor o si tiene varios. Esta opción solamente es relevante para los parámetros asignables y sirve para pasar varios valores en la misma llamada al servicio web (por medio de un solo parámetro). Si marca esta casilla, podrá conectar una secuencia de valores al nodo de estructura del parámetro en la asignación (en lugar de un solo valor). MapForce tratará la secuencia de valores en función del estilo del parámetro: • Si se trata de parámetros de plantilla, los valores se envían al servicio web separados por comas. Por ejemplo: `http://example.org/api/products/1,2,3` • Si se trata de parámetros de consulta, se repite el nombre del parámetro tantas veces como valores haya. Por ejemplo: `http://example.org/api/products?color=red&color=green&color=blue` • Si el estilo es *Matriz*, los valores se separan con comas. Por ejemplo: `http://example.org/api/products;color=red;color=green;color=blue;size=big;size=small;` • Si el estilo es *Encabezado*, se repite el encabezado HTTP tantas veces como valores haya.
Descripción	Descripción opcional del parámetro. Si el parámetro es asignable, la descripción que se introduzca aquí aparecerá en el componente de asignación como texto de anotación del elemento de asignación.

Ejemplo nº1

El servicio web de este ejemplo recupera un producto en base a su identificador (`id`) por medio de un método HTTP GET. La URL del servicio web especifica el parámetro `id` entre llaves. Observe que el parámetro `id` también existe en la tabla *Parámetros* y que tiene el estilo *Plantilla*. Además, está configurado como parámetro *asignable*, es decir, que el parámetro aparecerá en la asignación como nodo de estructura que se puede conectar al valor real de `id` (que se podría tomar de una base de datos, de un archivo o de una constante, por ejemplo). Cuando se ejecute la asignación, este parámetro se reemplazará con el valor real. Por tanto, si el valor es "1", la URL será `http://example.org/api/products/1`.

Recuerde que, si quiere aportar un valor constante para `id`, también puede desactivar la casilla *Asignable* e introducir el valor en la columna *Valor fijo*.

© 2018 Altova GmbH

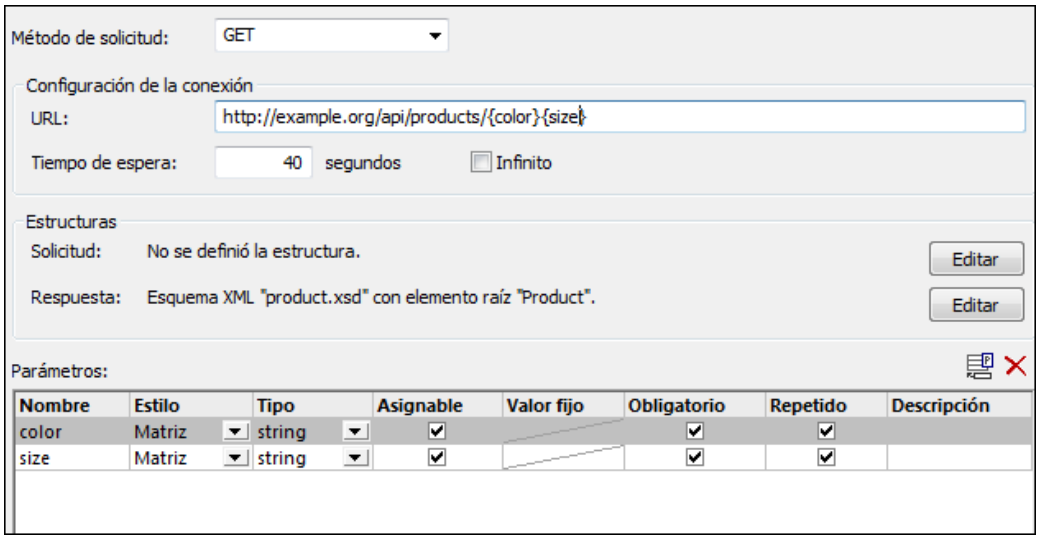


Los parámetros `Accept` y `Accept-Charset` tienen el estilo *Encabezado*. Estos parámetros se usan para llamar al servicio web con encabezados de solicitud personales. Hay dos maneras de aportar un valor al encabezado:

- dejando marcada la opción *Asignable* y dando el valor de encabezado personal desde la asignación o
- desactivando la asilla *Asignable* e introduciendo el valor en la columna *Valor fijo* directamente.

Ejemplo nº2

El servicio web de este ejemplo recupera una lista de productos que coinciden con el color y el tamaño dados como argumentos. El estilo de los parámetros es *Matriz,* es decir, se pueden definir como marcadores de posición tanto dentro de la URL como en la tabla de asignación. Observe que los parámetros están marcados como asignables y que la opción *Repetido* está marcada. Esto significa que su valor se leerá desde una secuencia de valores de la asignación (por ejemplo, una lista de filas dentro de un archivo de texto, un nodo XML o una columna de base de datos) y se enviará al servicio web en tiempo de ejecución. Por tanto, una dirección URL como esta podrá convertirse en la URL `http://example.org/api/products/;color=red;color=blue;size=big;size=small` siempre y cuando la asignación aporte los valores `red` y `blue` para el color y `big` y `small` para el tamaño.



Si necesita una URL como `http://example.org/api/products;color=red,blue;size=big,small`, entonces deberá:

1. Introducir la URL como `http://example.org/api/products;color={color};size={size}`.
2. En la tabla *Parámetros* cambiar el estilo del parámetro por *Plantilla*.

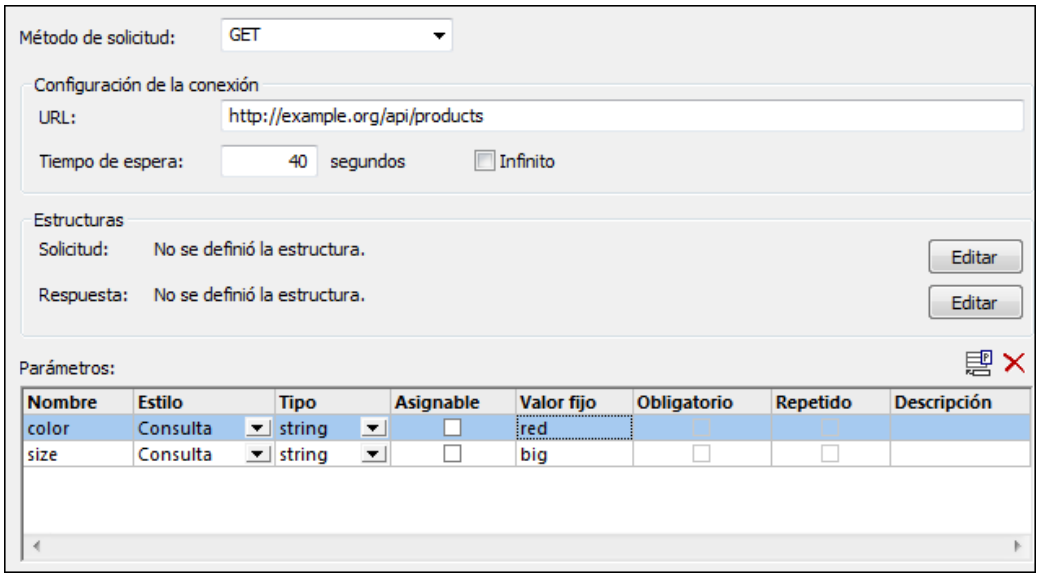
Si necesita una URL como `http://example.org/api/products?color=red,blue&size=big,small`, entonces deberá:

1. Introducir la URL como `http://example.org/api/products?color={color}&size={size}`.
2. En la tabla *Parámetros* cambiar el estilo del parámetro por *Plantilla*.

Ejemplo nº3

El servicio web de este ejemplo también recupera una lista de productos que coincidan con el color y el tamaño dados como argumentos, pero esta vez usando el estilo *Consulta*. Para este estilo no es necesario definir los parámetros como marcadores de posición en la URL. Solamente se definen en la tabla *Parámetros*. Los valores de parámetro se introducen en la tabla *Parámetros* directamente, en el campo *Valor fijo* (así que deberá desactivar la casilla *Asignable*). Por tanto, cuando se ejecute la asignación, la URL pasará a ser `http://example.org/api/products?color=red&size=big` .

© 2018 Altova GmbH



Definir la estructura de la respuesta y solicitud del servicio web

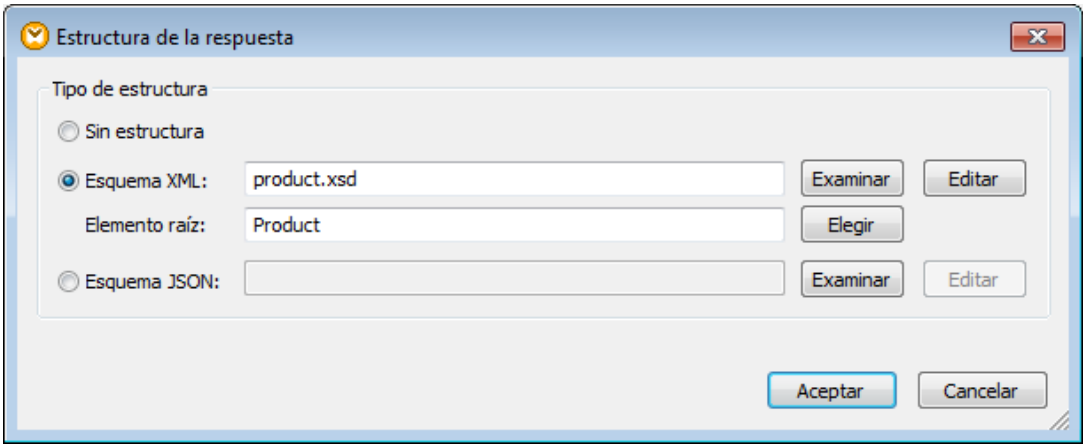
Cuando se trate de un servicio web que use solicitudes HTTP como POST o PUT, podrá incluir el contenido XML o JSON en el cuerpo del mensaje de solicitud. Igualmente, el servidor web puede devolver contenido XML o JSON en la respuesta a la llamada de la API. En casos como este, deberá indicar a MapForce cuál es el esquema del cuerpo del mensaje HTTP. Así los elementos definidos por la estructura aparecerán como elementos asignables cuando termine de añadir el servicio web a la asignación.

Para definir el esquema XML o JSON de una estructura de solicitud o respuesta:

1. Obtenga el esquema XML, JSON o DTD de la estructura de solicitud o respuesta del proveedor del servicio web. MapForce también acepta como estructura un archivo XML con una referencia de esquema válida.

Nota: si tiene un archivo de solicitud o respuesta de ejemplo pero no tiene un archivo de esquema, puede usar Altova XMLSpy para generar el archivo de esquema. Con XMLSpy también puede convertir un archivo de instancia XML en JSON y viceversa.

2. Abra el cuadro de diálogo "Configuración de la llamada a servicio web" (*véase* *Configurar llamadas a servicios web*).
3. En el grupo de opciones *Definición del servicio* seleccione la opción *Manual*.
4. En el grupo de opciones *Estructuras* haga clic en el botón **Editar** de la solicitud (o de la respuesta, según corresponda).
5. En el cuadro de diálogo haga clic en **Examinar** e introduzca la ruta de acceso del archivo de esquema. Si selecciona un archivo de instancia XML, deberá tener una referencia de esquema válida.



Algunos esquemas XML definen elementos con declaración global (es decir, elementos cuyo elemento primario es el elemento `schema`). Cuando trabaje con este tipo de esquemas, podrá elegir qué elemento del esquema hará de elemento raíz en la estructura de la asignación en MapForce. Esto se consigue haciendo clic con el botón **Elegir** y seleccionando el elemento raíz en el siguiente cuadro de diálogo.

© 2018 Altova GmbH

10.2 Agregar llamadas a servicios web (de tipo WSDL)

Este apartado explica cómo añadir a una asignación una llamada a un servicio web de tipo WSDL. Para aprender a añadir servicios web genéricos a la asignación consulte el apartado anterior Agregar llamadas a servicios web (de tipo REST).

Antes de añadir la llamada a servicio web de tipo WSDL al área de asignación, asegúrese de que tiene el archivo WSDL que describe el servicio web al que intentará conectarse.

Para agregar una llamada a un servicio web de tipo WSDL:

1. En el menú **Insertar** seleccione el comando **Función de servicio web** (o haga clic en el botón **Insertar función de servicio web**  de la barra de herramientas).
2. En el grupo de opciones *Definición del servicio* seleccione en la opción *WSDL*.
3. En el grupo de opciones *Configuración WSDL* haga clic en **Examinar** y seleccione el archivo WSDL. En la carpeta de ejemplos de MapForce encontrará dos archivos WSDL de ejemplo (uno para la versión 1.0 y otro para la versión 2.0). Estas son las rutas de acceso de estos archivos de ejemplo: **<Documentos>\Altova\MapForce2018\MapForceExamples\TimeService\TimeService.wsdl** and **<Documentos>\Altova\MapForce2018\MapForceExamples\TimeServiceWsdl2\TimeService20.wsdl**.
4. Si el archivo WSDL define varios servicios, operaciones y extremos, haga clic en **Elegir** para seleccionar las opciones necesarias.
5. Defina las demás opciones de configuración (*véase Configurar llamadas a servicios web*).
6. Haga clic en **Aceptar**.

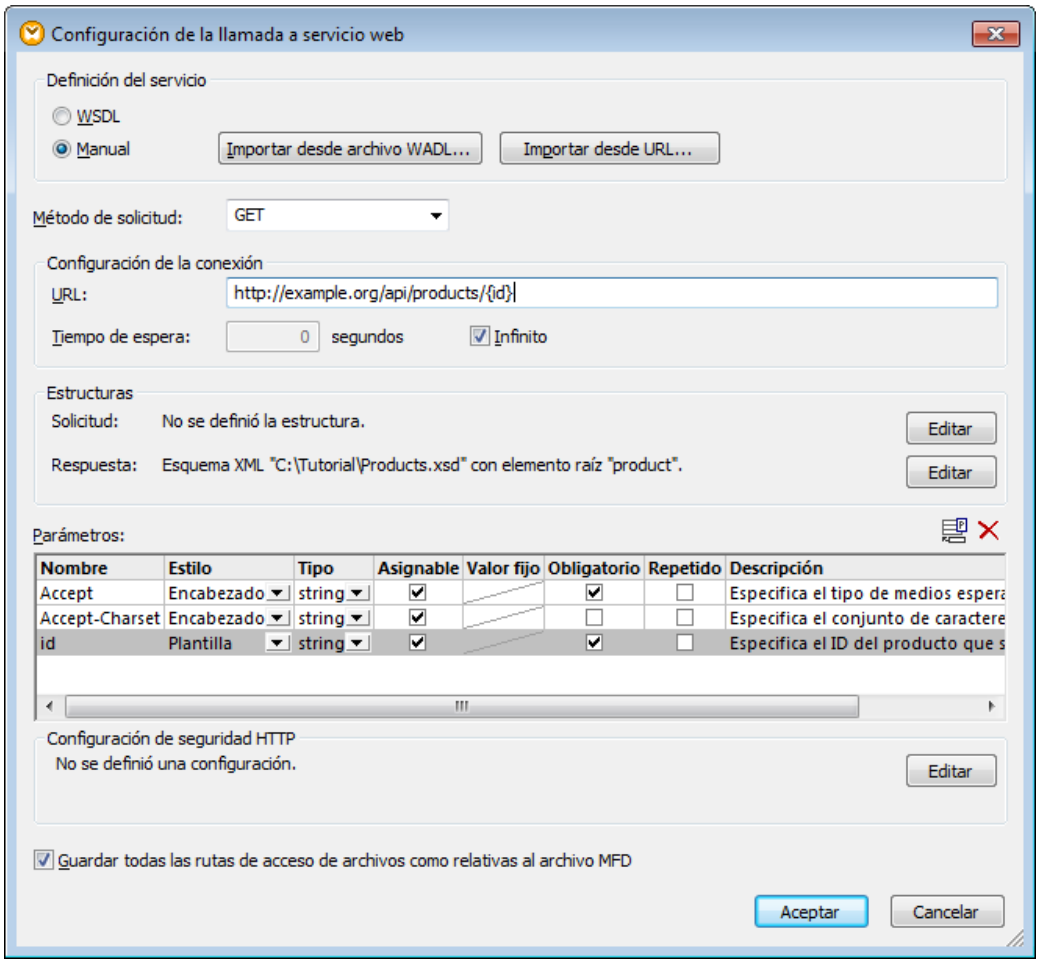
Consulte también el apartado Ejemplo: llamar a un servicio web SOAP

10.3 Configurar llamadas a servicios web

Puede cambiar la configuración de los servicios web desde el cuadro de diálogo "Configuración de la llamada a servicio web". Hay varias maneras de abrir este cuadro de diálogo:

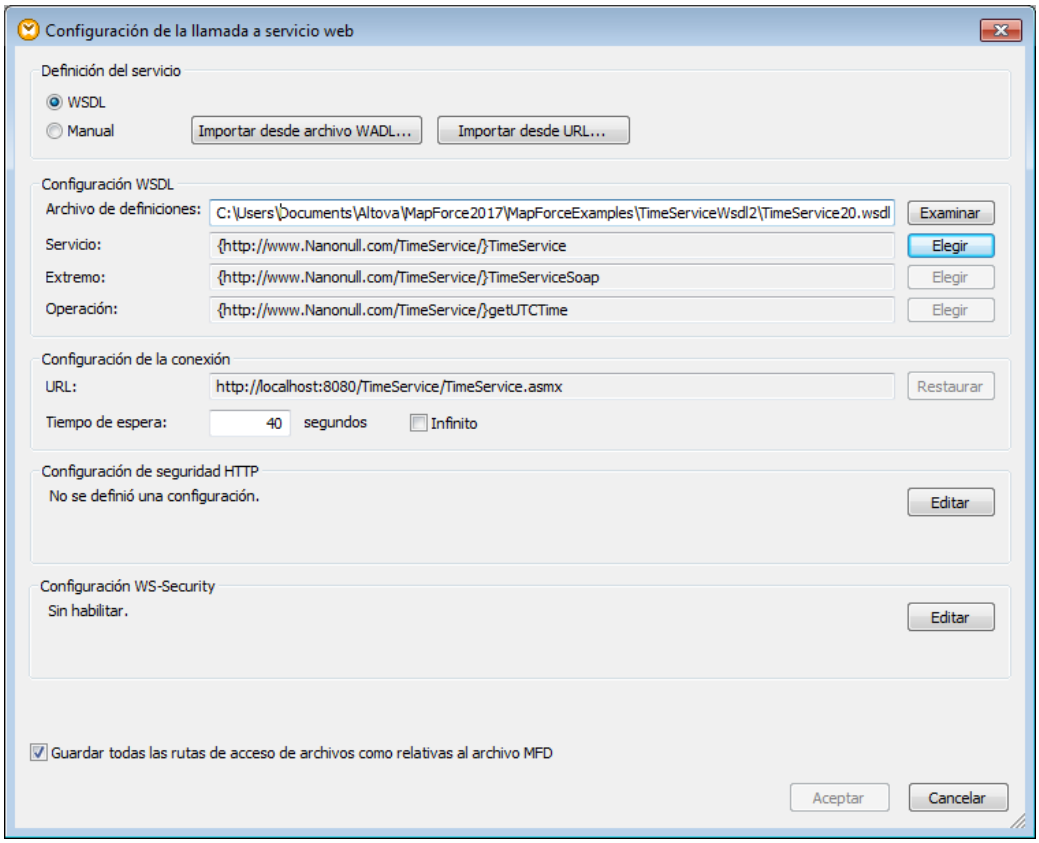
- Haciendo clic en el componente de servicio web en la asignación y seleccionando el comando de menú **Componente | Propiedades**.
- Haciendo doble clic en el título del componente.
- Haciendo clic con el botón derecho en el título del componente y seleccionando **Propiedades** en el menú contextual.

La lista de opciones disponibles dependerá de si la opción de definición de servicio elegida es *WSDL* o *Manual*. A continuación puede ver las opciones de configuración disponibles para un servicio HTTP genérico:



En la imagen siguiente puede ver las opciones de configuración disponibles para un servicio web de estilo WSDL:

© 2018 Altova GmbH



A continuación describimos en detalle todas las opciones de configuración disponibles.

Definición del servicio

Si necesita configurar un servicio web de estilo WSDL, elija la opción *WSDL*. De lo contrario seleccione *Manual*.

Si selecciona *Manual*, deberá introducir a mano las opciones de configuración del servicio web en el cuadro de diálogo. Y si tiene el archivo WADL del servicio web, podrá importar la configuración haciendo clic en **Importar desde archivo WADL** (*véase Importar información del servicio web desde WADL*). Además, si desea extraer los parámetros del servicio web de una URL, haga clic en el botón **Importar desde URL** (*véase Importar parámetros del servicio web desde una URL*).

Configuración WSDL

Este grupo de opciones solamente es relevante para servicios web de estilo WSDL.

Definiciones WSDL	Especifica el archivo WSDL del servicio web al que se debe llamar.

Servicio	Especifica el nombre del servicio web al que se debe llamar. Si el archivo WSDL define varios servicios, haga clic en **Elegir** para seleccionar el necesario.
Extremo	Especifica el extremo (o puerto) del servicio web al que se debe llamar. Si el servicio web seleccionado define varios extremos de servicio web, haga clic en **Elegir** para seleccionar el necesario.
Operación	Especifica la operación del servicio web al que se debe llamar. Si el extremo seleccionado define varias operaciones de servicio web, haga clic en **Elegir** para seleccionar la necesaria.

Método de solicitud

Esta configuración solamente es relevante si se trata de un servicio web genérico (no WSDL) y define el método HTTP que utiliza el servicio (p. ej. GET, POST, PUT, etc.).

Configuración de la conexión

URL especifica la dirección (URL) del servicio web. **Tiempo de espera** define el plazo de tiempo pasado el cual la llamada al servicio web expirará si no se obtiene respuesta del servidor. Seleccione *Infinito* si la llamada debe esperar una respuesta durante un plazo indefinido de tiempo.

Parámetro

Este grupo de opciones solamente es relevante si se trata de un servicio web genérico (no WSDL). La tabla *Parámetros* especifica los parámetros de URL que MapForce usará para llamar al servicio web (*véase* *Definir los parámetros del servicio web*).

Estructuras

Este grupo de opciones solamente es relevante si se trata de un servicio web genérico (no WSDL) (*véase* *Definir la estructura de la respuesta y solicitud del servicio web*).

Configuración de seguridad HTTP

Si el servicio web exige una autenticación previa (por HTTP o HTTPS), haga clic en **Editar** para especificar la configuración de autenticación (*véase* *Configurar seguridad HTTP*).

Configuración WS-Security

Estas opciones solamente son relevantes si se trata de un servicio web de tipo WSDL. Si el servicio utiliza seguridad SOAP, haga clic en **Editar** para especificar más opciones de configuración (*véase* *Configurar seguridad WS-Security*).

© 2018 Altova GmbH

Guardar todas las rutas de acceso de archivos como relativas al archivo MFD

Si habilita esta opción, MapForce guarda todas las rutas de acceso que aparecen en el cuadro de diálogo "Configuración del componente" como rutas relativas a la ubicación del archivo de diseño de MapForce (.mfd). Para más información consulte el apartado Usar rutas de acceso absolutas y relativas.

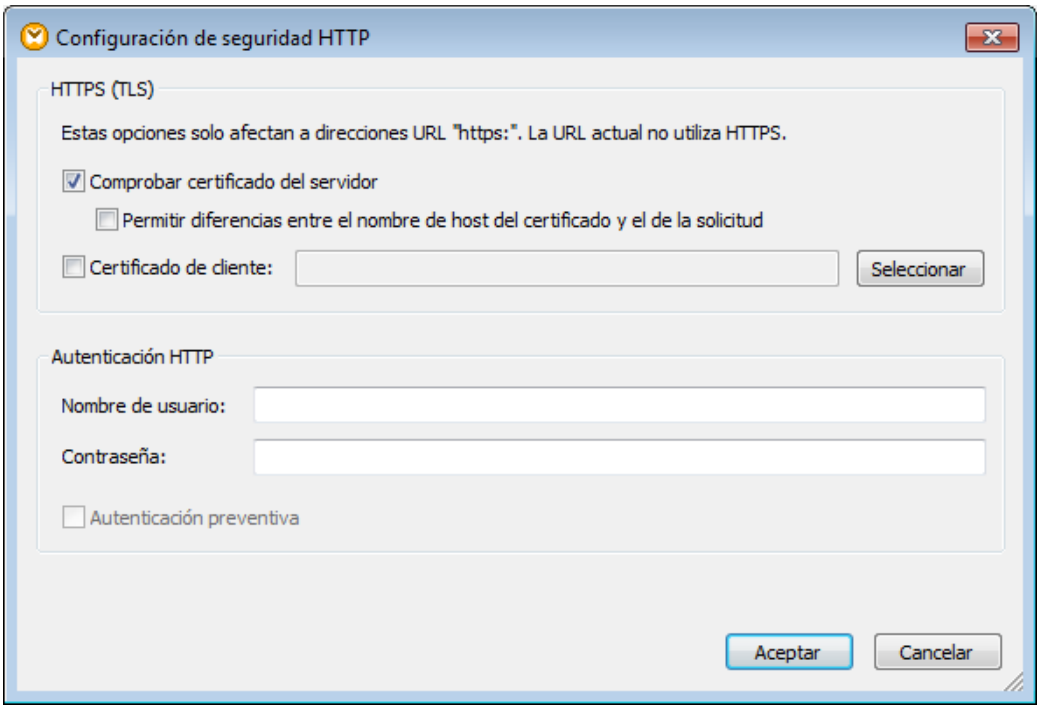
10.4 Configurar seguridad HTTP

Debe definir opciones de configuración de seguridad HTTP en estos casos:

- Cuando llame a un servicio web por HTTPS y el servicio web exija un certificado de cliente.
- Cuando el servicio use un certificado de servidor incorrecto.
- Cuando el servicio web exija autenticación HTTP.

Para configurar la seguridad HTTP:

1. Abra el cuadro de diálogo "Configuración de la llamada a servicio web" (*véase Configurar llamadas a servicios web*).
2. Haga clic en el botón **Editar** del grupo de opciones *Configuración de seguridad HTTP.*



Cuadro de diálogo "Configuración de seguridad HTTP"

El grupo de opciones *HTTPS (TLS)* es relevante para llamadas a servicios web realizadas por HTTPS.

Comprobar certificado del servidor	Esta casilla se activa por defecto. Por tanto, MapForce está configurado por defecto para comprobar el certificado del servidor antes de continuar con la solicitud. La solicitud del servicio web (y la asignación de datos) darán lugar a errores si el servidor no es de confianza o si el sistema operativo no está configurado para confiar en el servidor web.

© 2018 Altova GmbH

	No se recomienda desactivar esta casilla a no ser que haya un motivo concreto para ello. Consulte la sección Gestión de certificados digitales para obtener más información.
Permitir diferencias entre el nombre de host del certificado y el de la solicitud	A veces el certificado de servidor emitido para determinado nombre de host (p. ej. *www.example.com*) se instala en un nombre de host distinto (p. ej. *example.com*). Marque esta casilla para continuar con la autenticación aunque el nombre de host del certificado no coincida con el nombre de host al que llama el servicio web.
Certificado de cliente	Haga clic en **Seleccionar** para elegir un certificado de cliente del almacén de certificados *Usuario actual/Personal*. Se entiende que el certificado de cliente ya está en el almacén de certificados *Usuario actual/Personal*. Si no es así, puede importarlo con el complemento Certificados (*véase Acceso a los almacenes de certificados en Windows*). Si tiene pensado implementar la asignación de datos para ejecutarla en otro sistema operativo, deberá instalar el mismo certificado en el sistema operativo de destino. Consulte la sección Gestión de certificados digitales para obtener más información.

El grupo de opciones *Autenticación HTTP* es relevante para servicios web que exijan autenticación HTTP.

Nombre de usuario	Especifica el nombre de usuario necesario para acceder al servicio web.
Contraseña	Especifica la contraseña necesaria para acceder al servicio web.
Autenticación preventiva	Marque esta casilla si el servicio web está configurado para exigir datos de autenticación en la primera llamada. Si no se marca esta casilla, MapForce intentará acceder al servicio web sin nombre de usuario ni contraseña y los utilizará si el servidor requiere autorización (estado HTTP 401).

10.5 Configurar seguridad WS-Security

Debe definir opciones de configuración WS-Security si el servicio web está protegido con el protocolo WS-Security y exige un token de seguridad **TokenUsuario**.

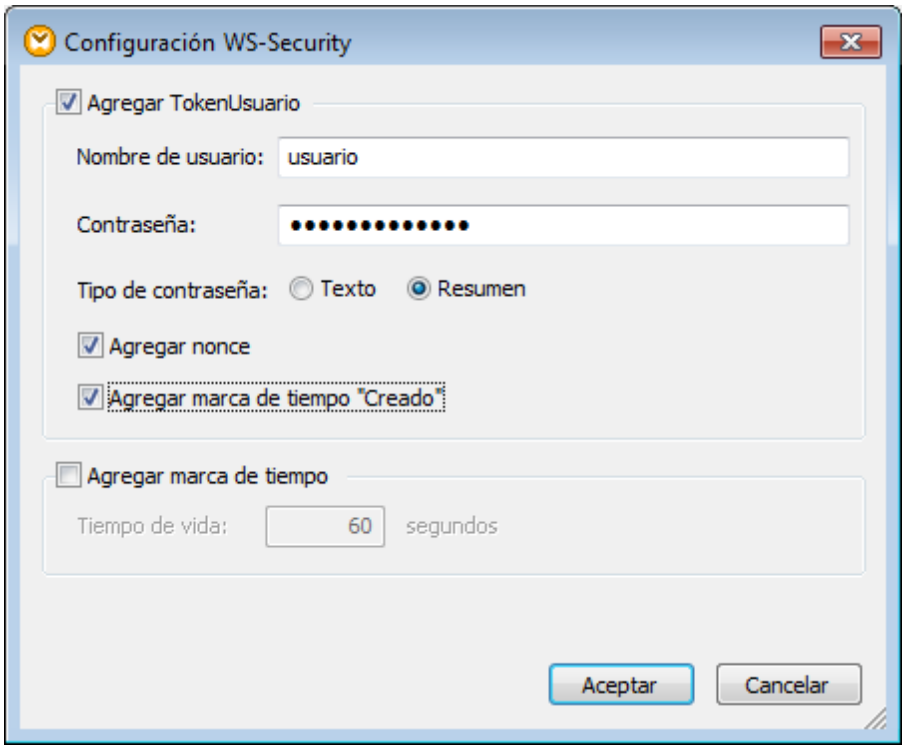
Convenciones

En este apartado usamos varias abreviaturas para los espacios de nombres de los servicios web:

Prefijo	Espacio de nombres
wsse	`http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd`
wsu	`http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd`

Para configurar la seguridad WS-Security:

1. Abra el cuadro de diálogo "Configuración de la llamada a servicio web" (*véase Configurar llamadas a servicios web*).
2. Haga clic en el botón **Editar** del grupo de opciones *Configuración WS-Security.*



Cuadro de diálogo "Configuración WS-Security"

© 2018 Altova GmbH

Agregar TokenUsuario	Un TokenUsuario es un elemento opcional de WS-security A UsernameToken is an optional WS-security element present in the header of the SOAP message. The UsernameToken is used by the Web server to authenticate the caller of the Web service. In the **wsse** specification (véase Conventions), this field corresponds to: `/wsse:UsernameToken`
Username	Enter the username included in the UsernameToken. In the **wsse** specification (véase Conventions), this field corresponds to: `/wsse:UsernameToken/wsse:Username`
Password	Enter the text of the password included in the UsernameToken. In the **wsse** specification (véase Conventions), this field corresponds to: `/wsse:UsernameToken/wsse:Password`
Password type	Select the type of password included in the UsernameToken. Select **Digest** if the Web server expects the password in this mode; otherwise select **Text**. In the **wsse** specification (véase Conventions), this field corresponds to: `/wsse:UsernameToken/wsse:Password/@Type`
Add nonce	Select this check box if you want to add a nonce to the Username token. A nonce is a random value which uniquely identifies each UsenameToken to provide additional security. If you enable this option, it is recommended to enable the **Add "Created" timestamp** option as well. In the **wsse** specification (véase Conventions), this field corresponds to: `/wsse:UsernameToken/wsse:Nonce`
Add "Created" timestamp	Select this option to add a timestamp to each nonce. In the **wsse** specification (véase Conventions), this field corresponds to: `/wsse:UsernameToken/wsu:Created`
Add Timestamp	Select this check box if you want to enable the time-to-live

	(TTL) value for the SOAP message (véase the next option). In the **wsu** specification (véase Conventions), this field corresponds to: `/wsu:Timestamp`
Time to live	Enter the time-to-live (TTL) for the SOAP message to diminish the chance of someone intercepting the message and replaying it.

© 2018 Altova GmbH

10.6 Ejemplo: llamar a un servicio web de tipo REST

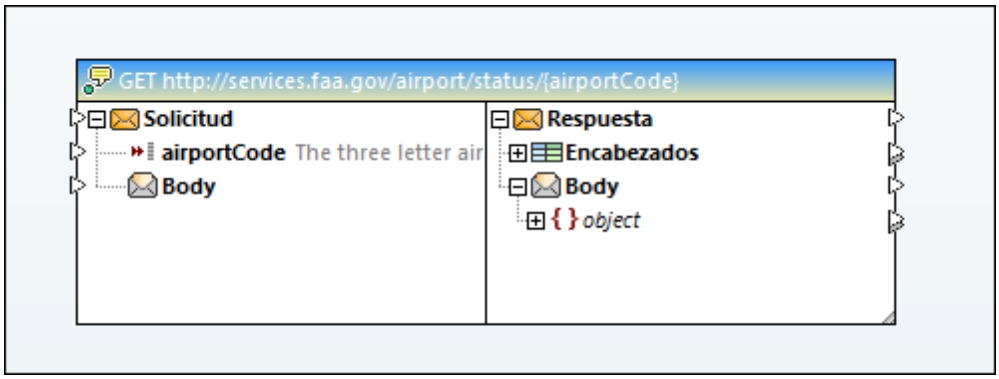
Este ejemplo sirve para explicar cómo se llama a un servicio web HTTP genérico desde MapForce. La descripción del servicio web al que se llama en este ejemplo puede encontrarse en http://services.faa.gov/docs/services/airport/#airportStatus. Este servicio web devuelve el estado actual de cualquier aeropuerto de EE UU en forma de estructura XML o JSON y acepta el código de tres letras del aeropuerto como argumento (p. ej. "SFO", "IAD", "ABE", "DFW", etc.). El ejemplo viene acompañado de un archivo de diseño de asignación de datos que está en esta carpeta: **<Documentos>\Altova\MapForce2018\MapForceExamples\AirportStatus.mfd** .

En este ejemplo, la respuesta del servicio web está en formato JSON y se asigna a un archivo JSON. Por tanto, necesitaremos el esquema JSON de la estructura de respuesta del servicio web. En este caso, hemos generado el esquema JSON con XMLSpy, a partir de un ejemplo de respuesta que ofrece el proveedor del servicio web. Encontrará este esquema en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\AirportStatus.schema.json**.

Para la parte de la solicitud vamos a llamar al servicio web con un parámetro que aportará el valor "SFO" en la solicitud. Para la parte de la respuesta vamos a asignar los datos de la respuesta a un archivo JSON. Por tanto, la asignación de datos estará compuesta por tres componentes principales: la llamada al servicio web, el parámetro de entrada y el archivo JSON de salida.

Paso nº1: agregar el componente de servicio web

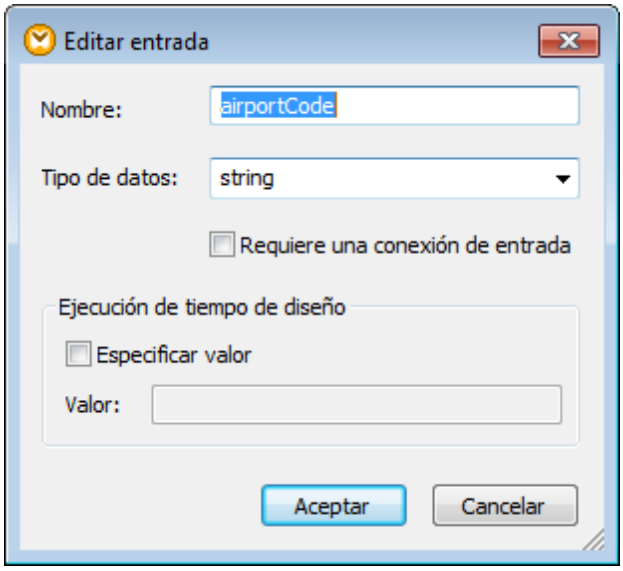
1. Asegúrese de que el lenguaje de transformación de la asignación de datos es BUILT-IN (*véase Seleccionar un lenguaje de transformación*).
2. En el menú **Insertar** haga clic en el comando **Función de servicio web**.
3. En el grupo de opciones *Definición del servicio* elija la opción *Manual*.
4. Elija el método de solicitud **GET** y defina la URL `http://services.faa.gov/airport/status/{airportCode}`. El valor que aparece entre llaves es un parámetro plantilla que se reemplazará con el código del aeropuerto en tiempo de ejecución (*véase Agregar llamadas a servicios web (de tipo REST)*).
5. Haga clic en el botón **Agregar parámetro** y añada un parámetro nuevo a la tabla *Parámetros*. Observe que el nombre del parámetro debe ser idéntico al del parámetro que se especifica entre llaves en la URL. Establezca el estilo *Plantilla*, elija el tipo *Cadena* y marque las casillas *Asignable* y *Obligatorio*. El campo *Descripción* puede dejarse vacío porque es opcional.
6. Ahora añada un encabezado para decirle al servidor web que el cliente (es decir, MapForce) espera JSON en la respuesta. Para añadir el encabezado haga clic otra vez en el botón **Agregar parámetro**, añada un parámetro llamado *Accept*, defina el estilo *Encabezado* e introduzca el valor fijo **application/json**.
7. En el grupo de opciones *Respuesta* haga clic en el botón Editar y navegue hasta el esquema de la respuesta del servicio web. El esquema está en la ruta de acceso **<Documentos>\Altova\MapForce2018\MapForceExamples\AirportStatus.schema.json**.
8. Haga clic en **Aceptar** para terminar.



Llegados a este punto el área de asignación incluirá un servicio web y ya podemos asignar datos desde la respuesta hasta la solicitud.

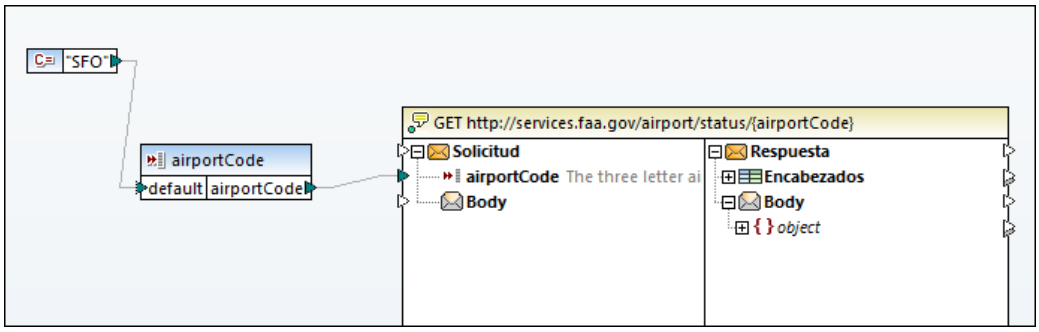
Paso nº2: agregar el parámetro de entrada

1. En el menú **Insertar** haga clic en el comando **Insertar componente de entrada**.
2. Introduzca el nombre de parámetro *airportCode* y desactive la casilla *Requiere una conexión de entrada* (porque la entrada vendrá dada por una constante, como se explica en el paso siguiente).



3. Añada una constante (con el comando de menú **Insertar | Constante**) con el valor *SFO* y conéctela al parámetro de entrada.
4. Conecte la salida del parámetro *airportCode* a la entrada de la solicitud, como muestra la siguiente imagen.

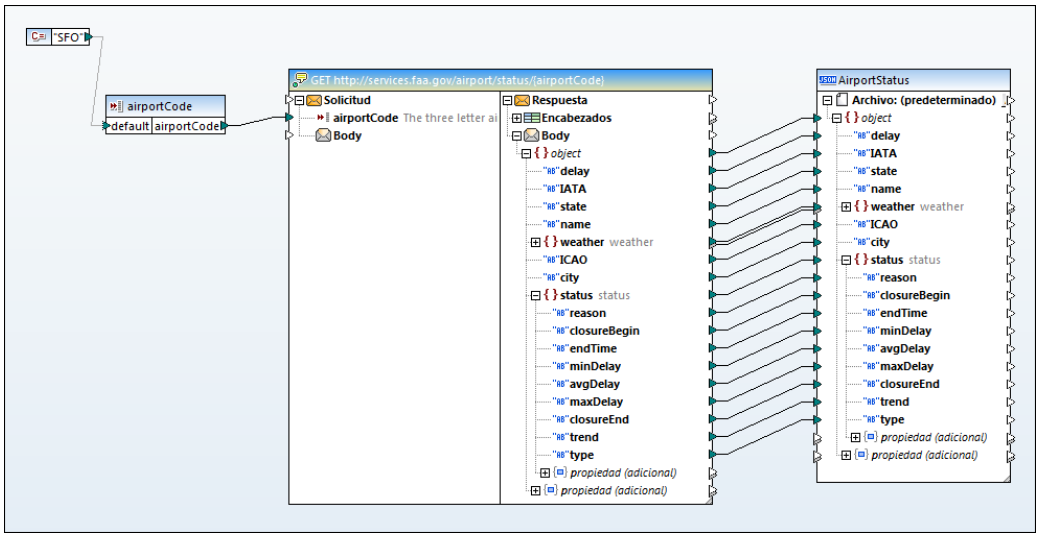
 © 2018 Altova GmbH



Ya está lista la parte de solicitud del servicio web. Cuando se ejecute la asignación, se llamará al servicio web con el valor que viene dado por la constante (en este caso *SFO*).

Paso nº3: agregar la salida JSON

1. En el menú **Insertar** haga clic en el comando **Archivo o esquema JSON**.
2. Navegue hasta el esquema **<Documentos>\Altova\MapForce2018 \MapForceExamples\AirportStatus.schema.json** y haga clic en **Abrir**.
3. Cuando la aplicación solicite un archivo JSON de muestra, haga clic en **Omitir** (el archivo JSON de muestra no es necesario porque la aplicación generará uno).
4. En el menú Conexión compruebe que está activado el comando **Conectar automáticamente los secundarios equivalentes**.
5. Conecte el nodo `object` del cuerpo de la respuesta del servicio web al nodo `object` del componente JSON, tal y como muestra la siguiente imagen. Como el comando **Conectar automáticamente los secundarios equivalentes** está activado, todos los nodos descendientes se conectan automáticamente.



Paso nº4: ejecutar la asignación

Ahora ya puede llamar al servicio web. Haga clic en el panel *Resultados* para ejecutar la asignación y consultar la vista previa del resultado que se genera. Si la llamada al servicio web se ejecuta correctamente, en el panel *Resultados* aparecerá la estructura JSON que se devuelve. Si la llamada no se ejecuta correctamente, MapForce devolverá un mensaje de error. Tal y como se

indicó en la descripción del servicio web, si el servidor no está disponible, se devuelve el error 502 (puerta de enlace incorrecta). También puede recibir el mensaje de error HTTP 404 (no se encontró) si usa un código de aeropuerto personal incompatible con el servicio web.

Si tiene MapForce Server (http://www.altova.com/es/mapforce/mapforce-server.html), también puede compilar la asignación en un archivo de ejecución (*.mfx) y ejecutar este archivo desde la línea de comandos o desde la API de MapForce Server en el equipo servidor donde se ejecuta MapForce Server (véase Compilar asignaciones en archivos de ejecución de MapForce Server).

En resumen, ya sabe crear una llamada a servicio web HTTP genérico que utiliza un método GET para recuperar datos de estado de los aeropuertos en tiempo real. Para más información consulte el apartado Agregar llamadas a servicios web (de tipo REST).

© 2018 Altova GmbH

10.7 Ejemplo: asignar datos de un canal RSS

Este ejemplo explica cómo puede llamar a un servicio web HTTP genérico desde MapForce para poder asignar datos de un canal RSS a un documento de Microsoft Excel. El ejemplo viene acompañado de un archivo de diseño que está en esta ruta de acceso: **<Documentos>\Altova \MapForce2018\MapForceExamples\RssReader.mfd** .

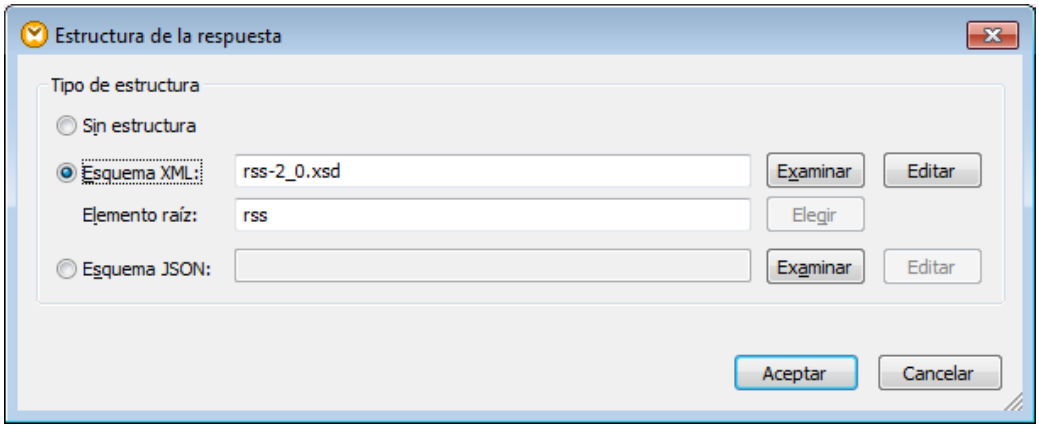
En este ejemplo la respuesta del servicio web se asigna desde el canal RSS del blog de Altova blog (https://blog.altova.com/feed). MapForce necesitará el esquema de la estructura de respuesta del servicio web para poder crear la estructura de los datos que devuelve el canal RSS. Este esquema está ubicado en esta ruta de acceso:**<Documentos>\Altova\MapForce2018 \MapForceExamples\rss-2-0.xsd**.

Para llamar a un canal RSS no hace faltan parámetros de solicitud, de modo que la parte de solicitud del servicio web estará vacía. La respuesta, por su parte, se asignará a un archivo de Microsoft Excel. Por último, para poder facilitar la lectura de la fecha de publicación de cada entrada del canal RSS, en el archivo Excel las fechas tendrán el formato AAAA-MM-DD (para lo cual se usarán funciones de procesamiento de datos).

Por tanto, el diseño de asignación estará compuesto por tres componentes: la llamada al servicio web, el componente de destino Excel y las funciones de procesamiento de datos.

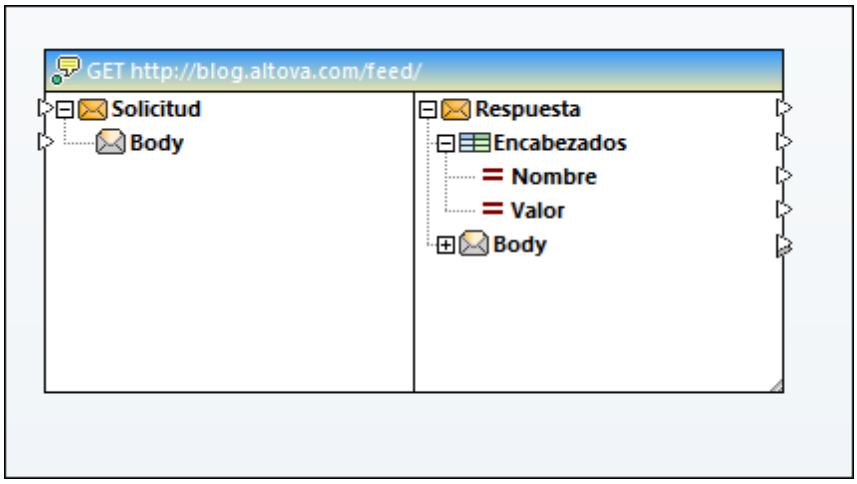
Paso nº1: agregar el componente de servicio web

1. Asegúrese de tener seleccionado el motor de transformación integrado (BUILT-IN) (*véase Seleccionar un lenguaje de transformación*).
2. Seleccione el comando de menú **Insertar | Función de servicio web**.
3. En el grupo de opciones *Definición del servicio* elija la opción *Manual*.
4. Elija el método de solicitud GET y la URL `http://blog.altova.com/feed`.
5. En el grupo de opciones *Respuesta* haga clic en el botón **Editar** y busque el esquema de la respuesta del servicio web. El esquema está en esta ruta de acceso: **<Documentos> \Altova\MapForce2018\MapForceExamples\rss-2-0.xsd**.



6. Para terminar haga clic en **Aceptar**.

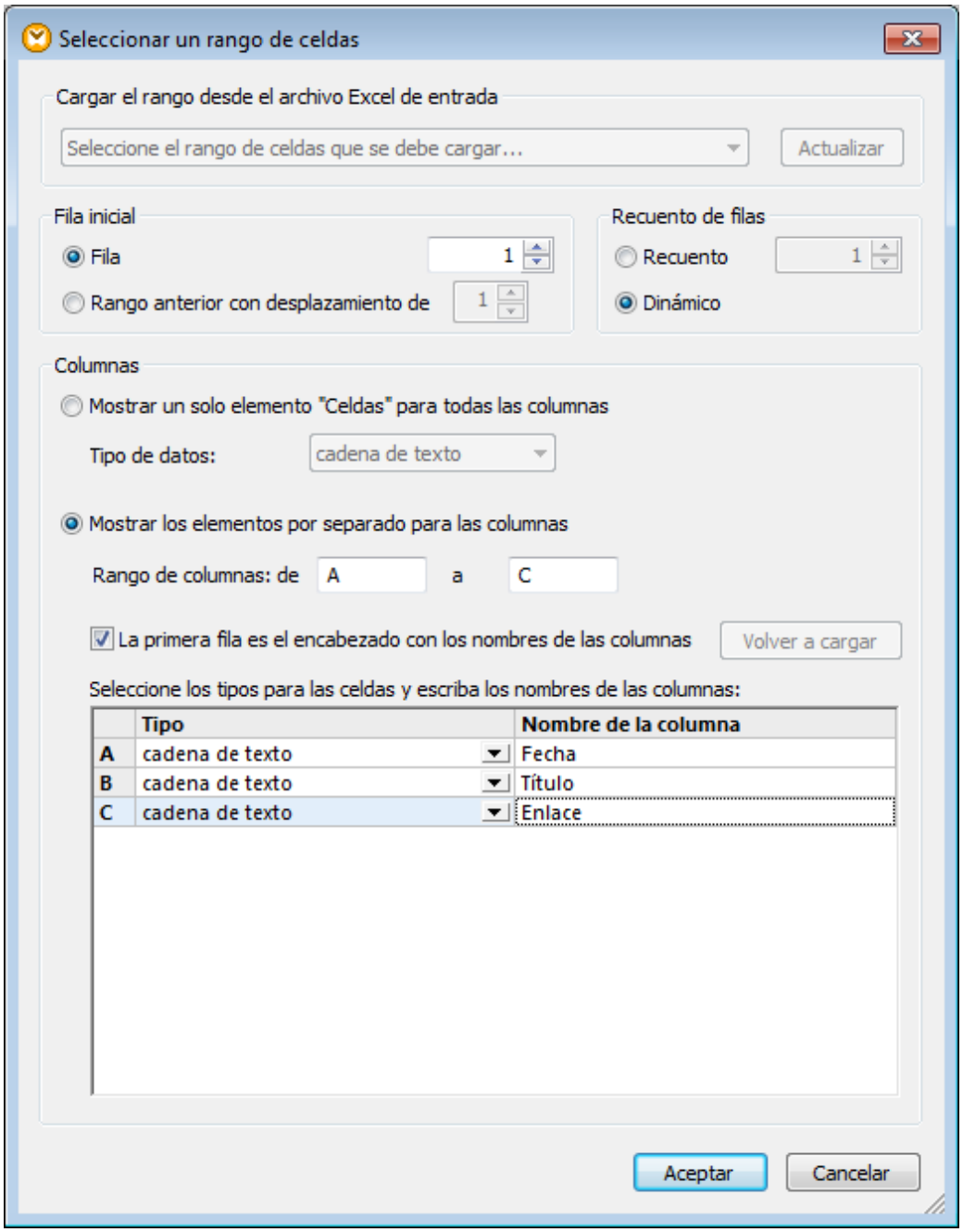
Llegados a este punto, el servicio web aparece en el área de asignación y podemos empezar a asignar datos desde la parte de respuesta.



Paso nº2: agregar el componente Excel de destino

1. Seleccione el comando de menú **Insertar | Archivo Excel 2007+**.
2. Cuando la aplicación solicite el archivo de muestra, haga clic en **Omitir**. El componente Excel aparece en el área de asignación.
3. Haga clic en el botón de **Hoja 1** y desactive la casilla *Mostrar las hojas de cálculo por su nombre*.
4. Haga clic en el botón de **Fila 1** y defina las opciones que aparecen en la imagen siguiente.

© 2018 Altova GmbH



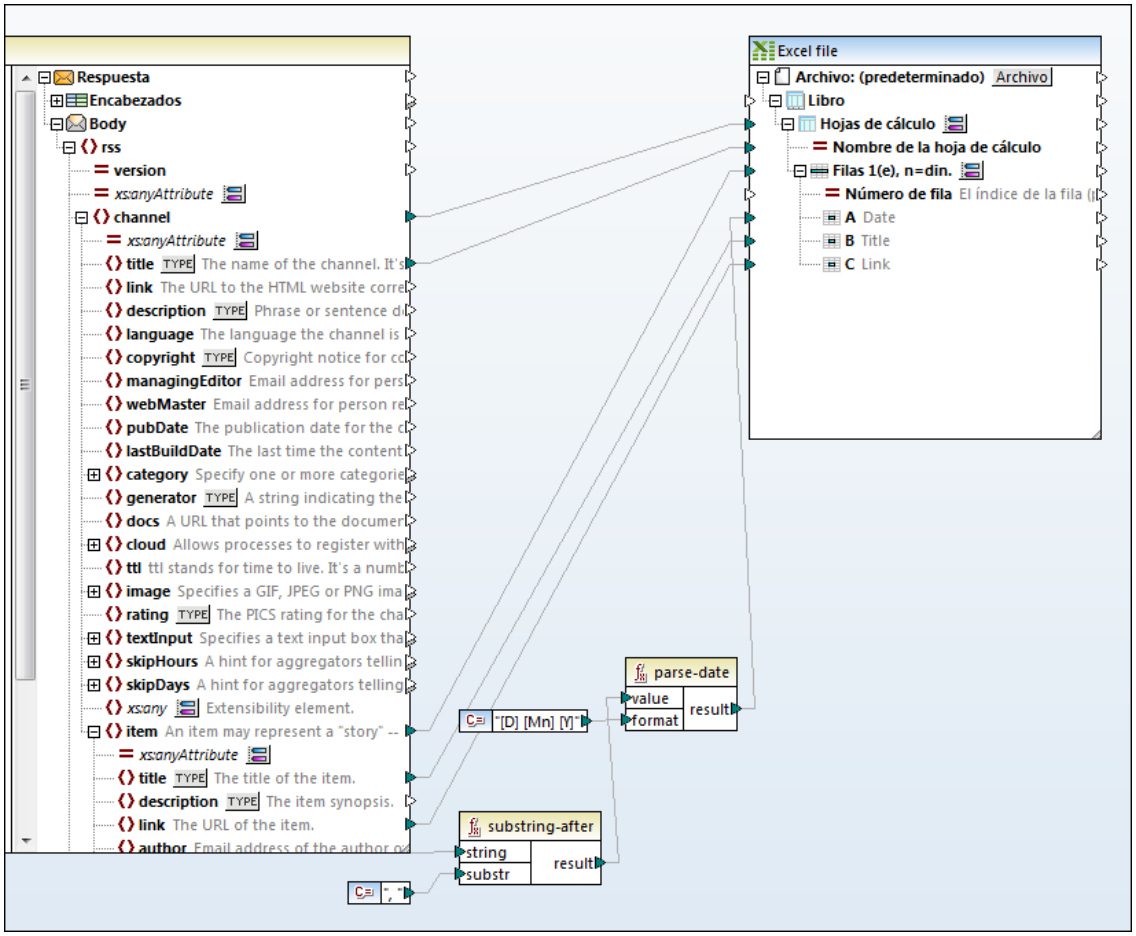
5. Cree conexiones entre estos elementos del cuerpo de la respuesta y estos elementos del componente Excel:

Cuerpo de la respuesta	Archivo Excel
`channel`	`Worksheets`
`title`	`Worksheet Name`
`item`	`Rows`
`item/title`	`Row B (Title)`
`item/link`	`Row C (Link)`

Llegados a este punto todavía falta un valor de entrada para **Fila A (Fecha)**. En el siguiente paso lo añadiremos.

Paso nº3: agregar las funciones de procesamiento de datos

La fecha de la respuesta del canal RSS aparece en formato RFC-822 (p. ej. `Mié, 20 enero 2016 14:49:35 +0000`). El objetivo, sin embargo, es presentar la fecha en el archivo Excel en formato `AAAA-MM-DD`. Para conseguirlo debemos añadir las funciones **`substring-after`** y **`parse-date`** de la biblioteca de funciones **core** de MapForce (encontrará instrucciones al respecto en el apartado Trabajar con funciones).



 © 2018 Altova GmbH

Como puede ver en la imagen anterior, la función **substring-after** toma el valor del nodo **pubDate** y devuelve solamente el texto que aparece después de `, `. El valor resultante se pasa a la función **parse-date**, que la analiza con la máscara `[D] [Mn] [Y]` y devuelve el valor como `xs:date`. La máscara de formato significa "día numérico del mes, seguido de un espacio, seguido de un nombre de mes en mayúsculas, seguido de un espacio, seguido del año con cuatro dígitos" (consulte la descripción de la función **format-dateTime** para obtener más información).

Por último, el valor analizado se pasa a la fila **Fecha** del componente Excel.

Paso nº4: ejecutar la asignación

Ya podemos llamar al servicio web. Haga clic en el panel *Resultados* para ejecutar la asignación y consultar una vista previa de los resultados generados. Si la llamada al servicio web se ejecuta correctamente, en el panel *Resultados* aparece la estructura de datos devuelta. Si no se ejecuta correctamente, MapForce devuelve un mensaje de error.

Si tiene MapForce Server (http://www.altova.com/es/mapforce/mapforce-server.html), también puede compilar la asignación en un archivo de ejecución de asignaciones (*.mfx) y ejecutarlo desde la línea de comandos o desde la API de MapForce Server en el equipo servidor donde se ejecute MapForce Server (véase Compilar asignaciones en archivos de ejecución de MapForce Server).

Ya hemos terminado de crear una llamada a servicio web HTTP genérico que usa un método GET para leer datos de un canal RSS. Consulte el apartado Agregar una llamada a servicio web (de tipo REST) para obtener más información sobre los servicios web genéricos.

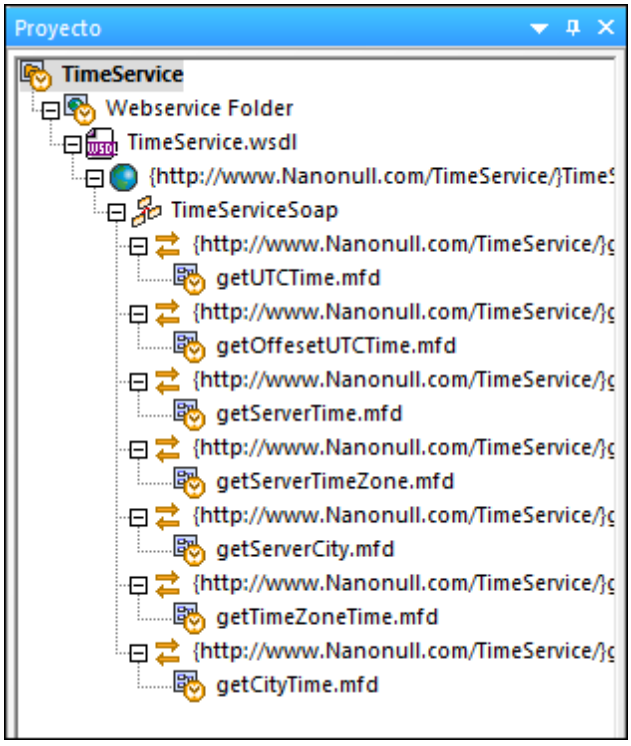
10.8 Ejemplo: llamar a un servicio web SOAP

En este ejemplo aprenderá a consultar un servicio web de hora usando como entrada una constante. El servicio web propiamente dicho se implementó con ayuda de MapForce, pero puede implementarlo con cualquier otra tecnología que admite un protocolo compatible.

La asignación de datos de este ejemplo forma parte del proyecto de asignación **TimeService.mfp**, disponible en la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\TimeService**. El archivo de proyecto **TimeService2.mfp** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\TimeserviceWsdl2** es compatible con WSDL 2.0.

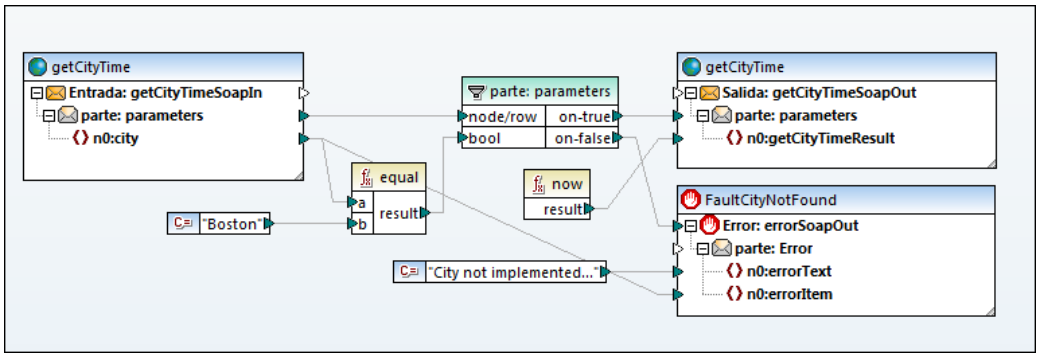
Para ver cómo se implementó el servicio web:

1. Seleccione el comando de menú **Archivo | Abrir** y seleccione el archivo **TimeService.mfp** de la carpeta **<Documentos>\Altova\MapForce2018\MapForceExamples\Timeservice**. Los archivos asociados con el proyecto se cargan en la ventana Proyecto.



2. Haga doble clic en la entrada **getCityTime.mfd** de la ventana Proyecto.

© 2018 Altova GmbH

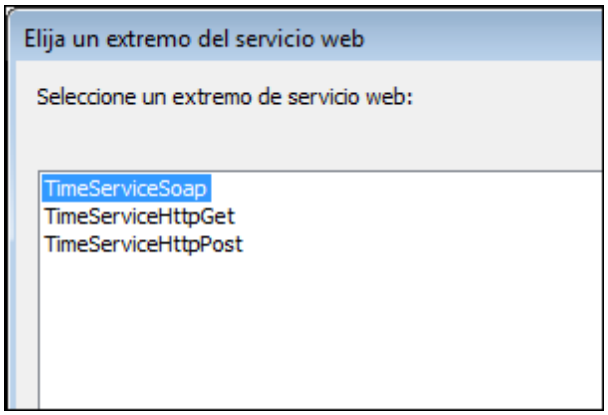


La asignación **getCityTime.mfd** acepta como entrada un nombre de ciudad y devuelve la hora actual en la salida. Si la ciudad no es Boston, se devuelve un error WSDL. La asignación toma los datos de entrada del archivo **getCityTimeRequest.xml** de la carpeta **<Documentos>\Altova \MapForce2018\MapForceExamples\TimeService**. Si hace doble clic en el componente **getCityTimeSoapIn**, verá que este archivo está configurado como origen de datos.

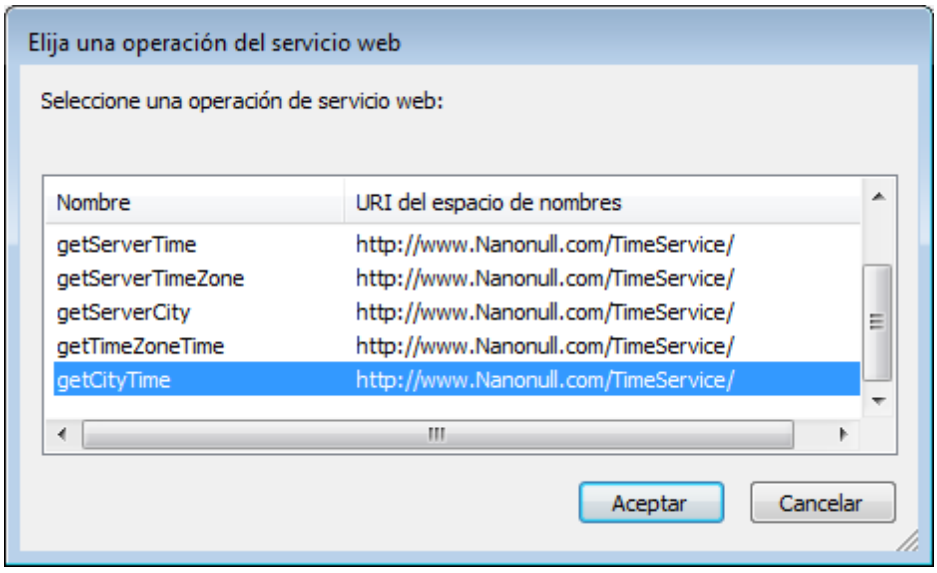
Para este ejemplo imaginaremos que este proyecto de servicio web ya está compilado e implementado en un servidor web real, al que llamaremos en los siguientes pasos del ejemplo.

Para llamar a la función de servicio web "getCityTime" desde una asignación de datos:

1. Seleccione el comando de menú **Archivo | Nuevo**, haga clic en el icono **Asignación** y confirme con el botón **Aceptar**.
2. Haga clic en el comando de menú **Insertar | Función de servicio web...** o en el botón  de la barra de herramientas.
3. Haga clic en **Examinar** para seleccionar el archivo de definición WSDL. Seleccione el archivo **TimeService.wsdl** del directorio **TimeService** y haga clic en el botón **Abrir**.
4. Cuando la aplicación solicite el extremo del servicio web, elija el extremo **TimeServiceSoap**.

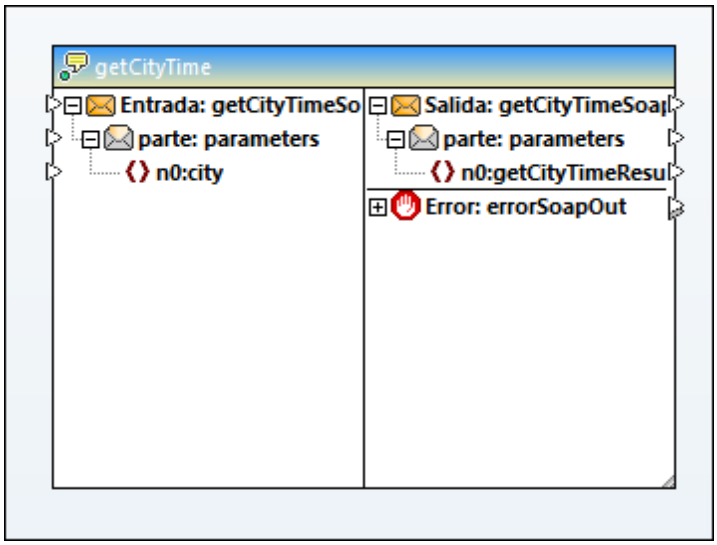


5. Cuando la aplicación solicite una operación de servicio web, elija la operación **getCityTime**.



6. Cuando la aplicación solicite la configuración de la llamada WSDL, deje las opciones de configuración predeterminadas como están y haga clic en **Aceptar** (*véase Configurar llamadas a servicios web*).

   La función de servicio web **getCityTime** se inserta en forma de componente, pero recuerde que realmente representa los ocho componentes que forman el archivo **getCityTime.mfd** que está guardado en el proyecto WSDL.



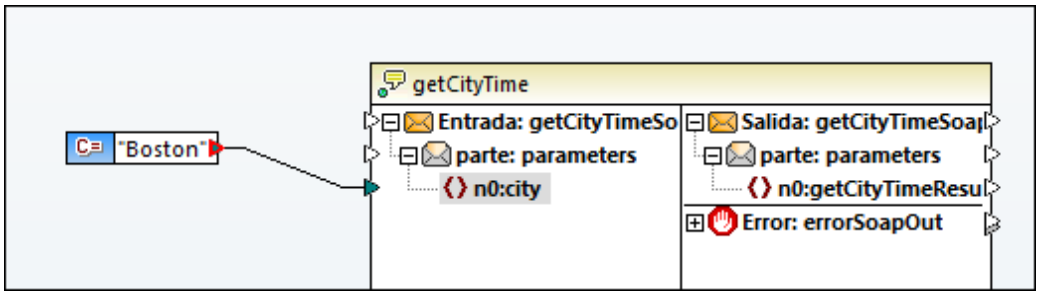
   La sección izquierda del componente define la entrada de datos (SoapIn), mientras que la derecha define la salida de datos (SoapOut), que también puede incluir una sección de error si así se definió en el archivo .wsdl.

Para definir los componentes de entrada y salida para la función de servicio web:

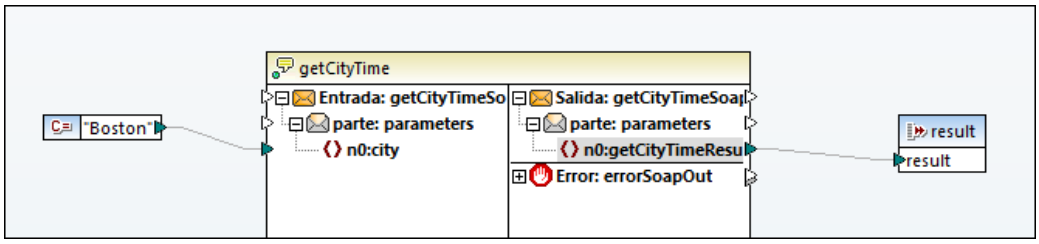
1. Inserte el componente que aportará los datos de entrada (p. ej. un componente de constante, texto o esquema). Para este ejemplo insertaremos un componente de

 © 2018 Altova GmbH

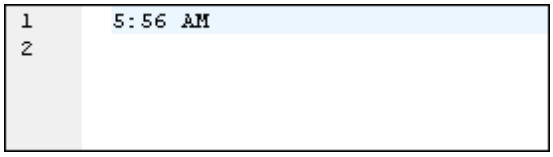
constante con el valor de cadena *Boston*.



2. Conecte la constante al elemento **n0:city**.
3. Inserte un componente de salida simple (con el comando de menú **Función | Insertar componente de salida**).
4. Conecte el elemento **n0:getCityTimeResult** al componente de salida.



5. Ahora haga clic en el panel *Resultados* para ver el resultado de la asignación.

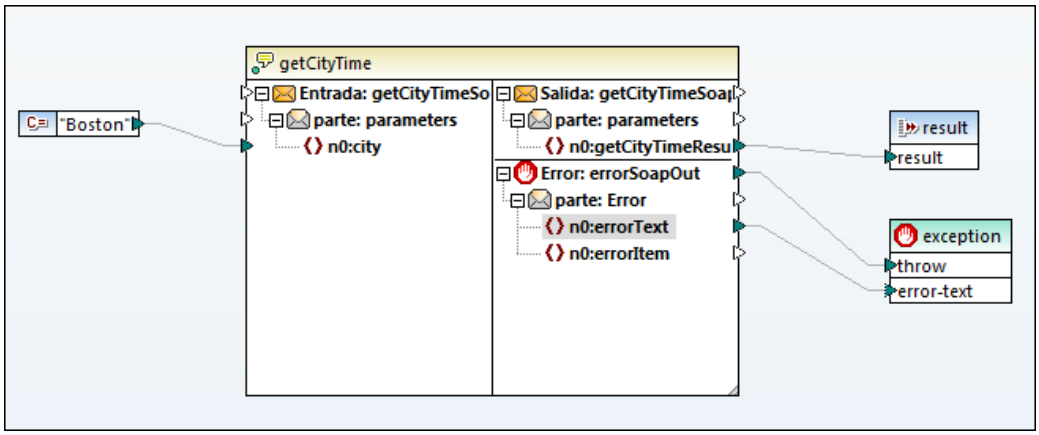


La hora actual en Boston aparece en el panel *Resultados*.

Nota: el valor de entrada de la función de servicio web tiene prioridad sobre el origen de datos de la asignación original. Por ejemplo, el valor *Boston* del componente de constante tiene prioridad sobre el archivo de origen **getCityTimeRequest.xml** de la asignación de datos original.

Para asignar errores de servicio web:

1. Seleccione el comando de menú **Insertar | Excepción** o haga clic en el botón **Excepción** de la barra de herramientas.
2. Asigne el elemento **Error:** al elemento **throw** del componente de excepción.
3. Asigne el elemento **n0:errorText** al elemento **error-text** del componente de excepción.



© 2018 Altova GmbH

10.9 Gestión de certificados digitales

La gestión de certificados digitales es un aspecto crucial del intercambio seguro de datos entre equipos clientes y servidores web. Como las asignaciones no solo se pueden ejecutar en Windows con Altova MapForce, sino que también se pueden ejecutar en servidores Windows, Linux y OS X / macOS con MapForce Server (por separado o como parte de FlowForce Server), en esta sección explicamos cómo puede gestionar certificados HTTPS en las distintas plataformas.

En el contexto del protocolo HTTPS es importante distinguir entre certificados de cliente y de servidor.

Certificados de servidor

Un certificado de servidor identifica un servidor como entidad de confianza para una aplicación cliente como MapForce. El certificado de servidor puede estar firmado digitalmente por una entidad de certificación comercial o por la organización del usuario. En ambos casos, mientras se diseña la asignación en MapForce tiene la opción de especificar:

- Si se debe comprobar el certificado de servidor.
- Si la solicitud debe continuar incluso si se detectan diferencias entre el nombre del certificado y el nombre de host.

Esta configuración puede editarse en el cuadro de diálogo "Configuración de seguridad HTTPS" de MapForce (*véase* *Configuración de seguridad HTTPS*). Si habilita la opción para comprobar el certificado de servidor, debe tener en cuenta que:

- Si llama a un servidor web cuyo certificado está firmado por una entidad de certificación de confianza, es probable que el sistema operativo ya esté configurado para confiar en el certificado de servidor, sin necesidad de realizar ajustes en el sistema.
- Si llama a un servidor web cuyo certificado está autofirmado (p. ej. un servidor de red local de su organización), deberá configurar el sistema operativo para que confíe en el certificado.

En la mayoría de los casos, puede comprobar el nivel de confianza que hay entre el sistema operativo y el servidor web con solo introducir la URL del servicio web en la barra de direcciones del explorador. Si el servidor no es de confianza o el sistema operativo no está configurado para confiar en el servidor, el explorador mostrará un mensaje parecido a "Esta conexión no es de confianza" o "Hay un problema con el certificado de este sitio web". Sin embargo, si el explorador utiliza una base de datos de certificados que no es la del sistema operativo (p. ej. Firefox 35.0.1 en Ubuntu 14.04) entonces no podrá comprobar el nivel de confianza con el explorador web.

En Windows puede establecer confianza con el servidor siguiendo las instrucciones del explorador e importando/instalando los certificados necesarios en el almacén *Entidades de certificación raíz de confianza* (*véase* *Confiar en certificados servidor en Windows*). En OS X / macOS puede realizar una operación parecida en Safari (*véase* *Confiar en certificados servidor en OS X / macOS*). Para Linux consulte el apartado Confiar en certificados servidor en Linux.

Certificados de cliente

Mientras que los certificados de servidor sirven para identificar un servidor como entidad de

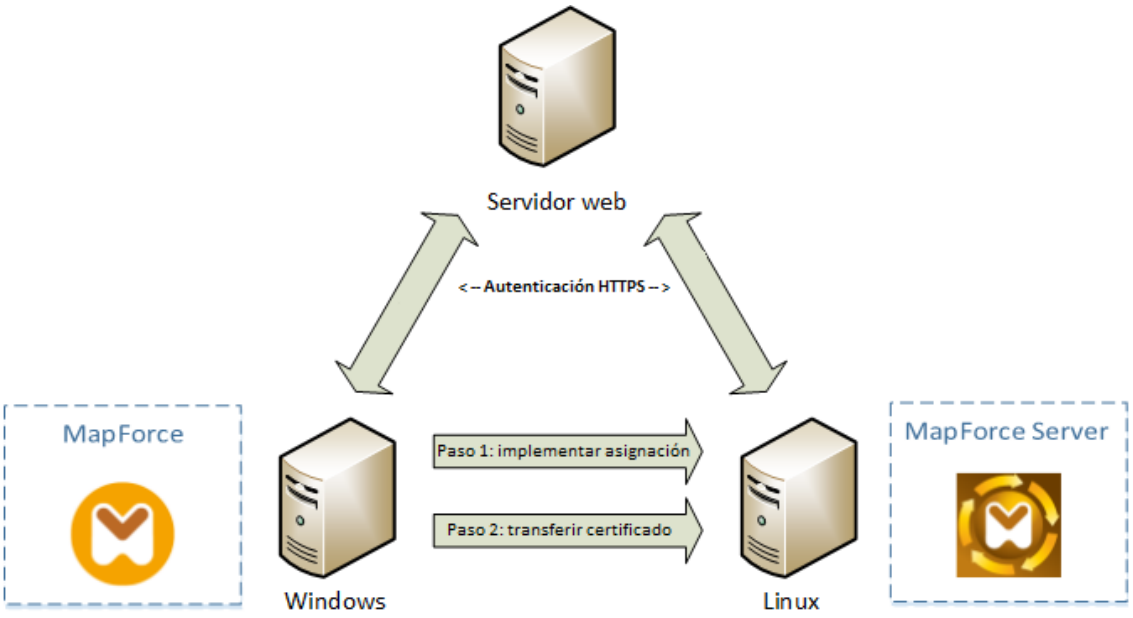
confianza, los certificados de cliente sirven principalmente para autenticar al autor de la llamada con el servidor web. Si tiene pensado llamar a un servidor web que requiere certificados de cliente, quizás sea recomendable ponerse en contacto con el administrador del servidor web y solicitar instrucciones para configurar el cliente correctamente. Por ejemplo, en el caso de IIS, el servidor web puede estar configurado para ocuparse de conexiones HTTPS y certificados de cliente de varias maneras:

- Puede exigir HTTPS y omitir el certificado de cliente
- Puede exigir HTTPS y aceptar el certificado de cliente
- Puede exigir HTTPS y también el certificado de cliente

El hecho de que la solicitud al servicio web funcione o no dependerá tanto de la configuración del servidor web como de la aplicación cliente. Por ejemplo, si el servidor web está configurado para exigir un certificado de cliente, para que funcione la llamada la aplicación que efectúa la llamada debe presentar un certificado de cliente válido.

Desde el punto de vista del sistema MapForce, lo mismo ocurre cuando las asignaciones incluyen llamadas a servicios web por HTTPS. Para que estas asignaciones se ejecuten correctamente, es necesario que el servidor web esté configurado para aceptar o exigir el certificado de cliente y que el sistema operativo donde se ejecuta la asignación suministre el certificado de cliente adecuado al servidor web.

En el diagrama que aparece a continuación puede ver un ejemplo donde se transfiere un certificado de cliente utilizado por MapForce a un servidor Linux que ejecuta MapForce Server. Una vez transferido al sistema operativo, el certificado se podrá usar en MapForce Server para autenticarse en el servidor web y ejecutar la asignación correctamente.



Implementación de asignaciones con certificados de cliente en otros equipos

MapForce es capaz de usar cifrado TLS (*Seguridad de la capa de transporte*) además de HTTP, que sustituye al protocolo SSL (*Capa de sockets seguros*). Tenga en cuenta que si la implementación cliente o el servidor no son compatibles con TSL es posible que se recurra a SSL.

 © 2018 Altova GmbH

Para garantizar la compatibilidad con llamadas web con autenticación de certificado de cliente en varias plataformas, MapForce y MapForce Server requieren que la gestión de certificados esté implementada en cada plataforma. Esto garantiza que la gestión de certificados siempre esté dentro del ámbito del sistema operativo. Cada sistema operativo ofrece diferentes funciones de gestión de certificados, tal y como muestra la tabla.

Platforma	Gestión de certificados e implementación
Windows	Puede gestionar certificados con el complemento *Certificados* (*véase Acceso a los almacenes de certificados en Windows*). La biblioteca *SChannel* ofrece compatibilidad con TLS.
Linux	Puede gestionar certificados con la herramienta y biblioteca OpenSSL de la línea de comandos. Si el equipo Linux que ejecuta MapForce Server todavía no es compatible con OpenSSL, deberá descargarlo e instalarlo para poder gestionar certificados. La biblioteca OpenSSL ofrece compatibilidad con TLS (https://www.openssl.org/).
OS X / macOS	Puede gestionar certificados con el administrador *Keychain Access Manager*, disponible en **Finder > Applications > Utilities**. La biblioteca nativa Secure Transport ofrece compatibilidad con TLS.

Si ejecuta la asignación en un sistema operativo Windows donde ya puede consumir correctamente el mismo servicio web al que tiene pensado llamar desde MapForce, lo normal es que no necesite realizar cambios en la configuración de certificados (*véase Certificados de cliente en Windows*). Sin embargo, si diseña asignaciones con MapForce en un equipo Windows y después las implementa en otro equipo (con otro sistema operativo), el certificado de cliente no se almacena ni se copia junto con el paquete implementado. Para que la llamada a servicio web y la asignación se ejecuten correctamente, el certificado de cliente debe existir también en el sistema operativo de destino.

Para transferir un certificado del sistema Windows a otro equipo Windows, exporte el certificado (con clave privada) del sistema de origen (*véase Exportar certificados de Windows*). Después debe importar el mismo certificado al almacén Usuario actual/Personal del sistema operativo de destino (*véase Certificados de cliente en Windows*).

Consulte los apartados Certificados de cliente en Linux y Certificados de cliente en OS X / macOS para obtener más información sobre cómo transferir certificados de cliente a plataformas Linux y OS X / macOS.

10.9.1 Confiar en certificados servidor en Linux

Para establecer confianza con un servidor web en Linux es necesario obtener el archivo de certificado del servidor web, copiarlo al almacén de certificados del sistema y actualizarlo (*ver instrucciones más abajo*). Una manera de obtener el certificado servidor es con el explorador, como puede ver en el siguiente ejemplo.

Siga estas instrucciones solamente si está seguro de la autenticidad del certificado del servidor web.

Debian, Ubuntu

1. Copie el archivo de certificado del servidor web en este directorio.

```
sudo cp /home/downloads/server_cert.crt /usr/local/share/ca-certificates/
```

2. Actualice el almacén de certificados de la siguiente manera:

```
sudo update-ca-certificates
```

Cent OS

1. Instale el paquete `ca-certificates`:

```
yum install ca-certificates
```

2. Habilite la función de configuración dinámica de entidades de certificación:

```
update-ca-trust enable
```

3. Copie el certificado de servidor en este directorio:

```
cp server_cert.crt /etc/pki/ca-trust/source/anchors/
```

4. Utilice este comando:

```
update-ca-trust extract
```

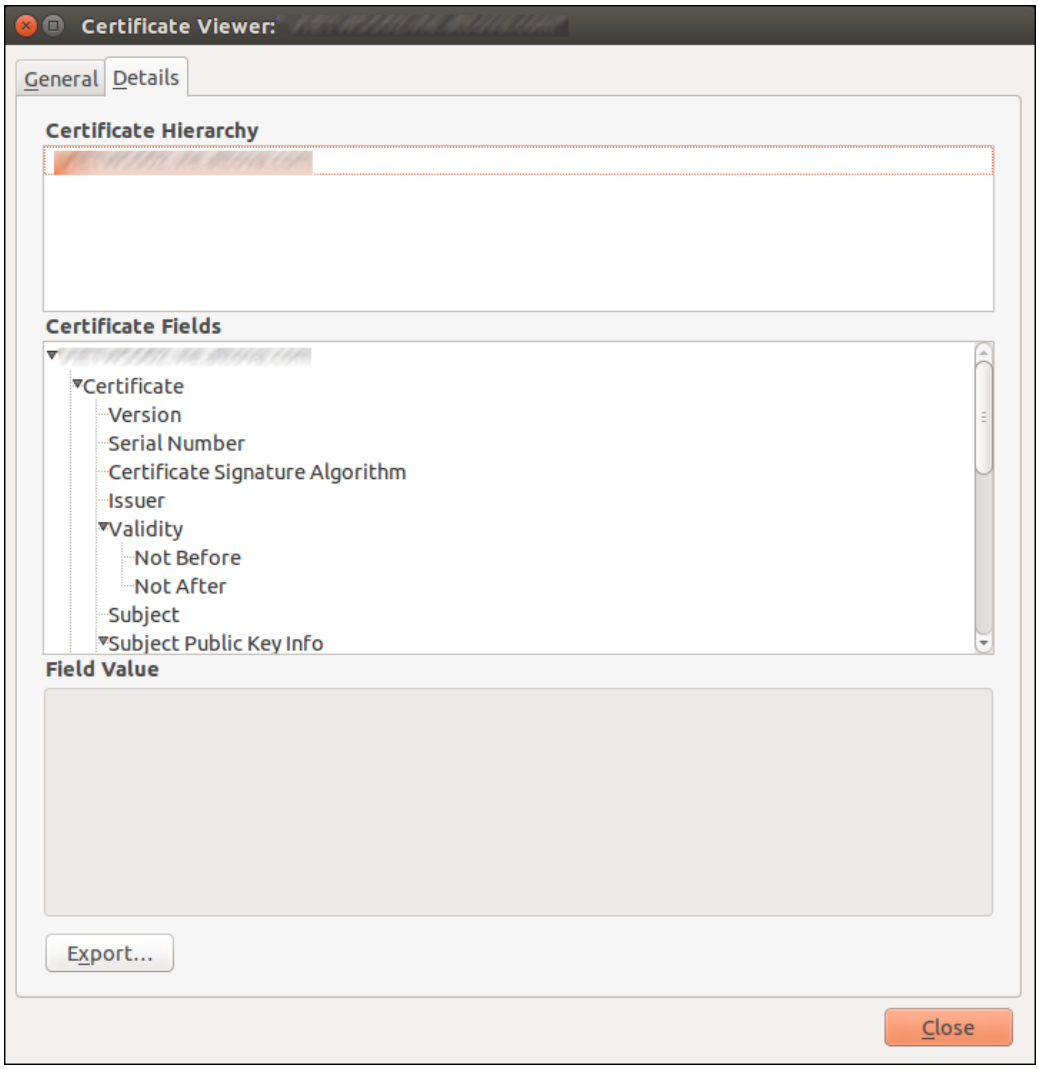
Ejemplo: exportar el certificado del servidor web con Firefox en Ubuntu

1. Inicie Firefox y acceda a la URL del servidor web.
2. Cuando aparezca el mensaje "This connection is untrusted" (*Esta conexión no es de confianza)* haga clic en **Add Security Exception** para añadir una excepción de seguridad.

© 2018 Altova GmbH



3. Haga clic en el botón **View**.
4. Haga clic en la pestaña **Details**.



5. Haga clic en el botón **Export** para exportar y guardar el archivo del certificado en un directorio local.

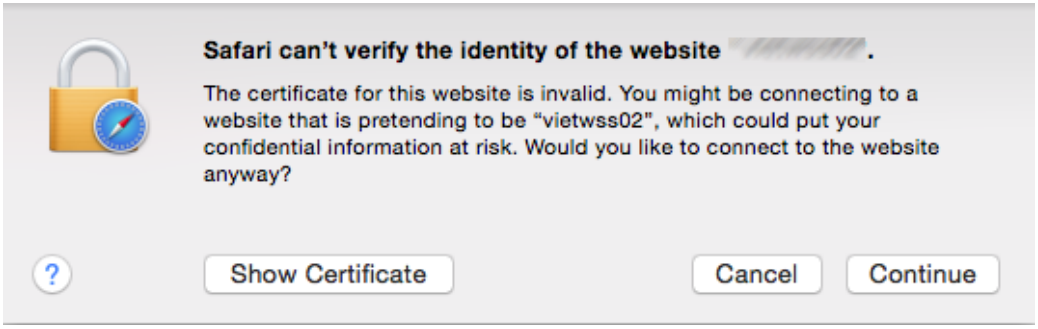
10.9.2 Confiar en certificados servidor en OS X / macOS

Para establecer confianza con un servidor web en Safari:

> Siga estas instrucciones solamente si está seguro de la autenticidad del certificado del servidor web.

1. En la barra de direcciones del explorador escriba la dirección HTTPS del servidor web.
2. Cuando se le solicite conectarse al sitio web haga clic en el botón **Show Certificate** para ver el certificado.

© 2018 Altova GmbH



3. Seleccione la opción **Always trust {certificate} when connecting to {website}** para confiar siempre en ese certificado cuando se conecte a ese sitio web.
4. Haga clic en el botón **Continue** para continuar e introducir su contraseña.

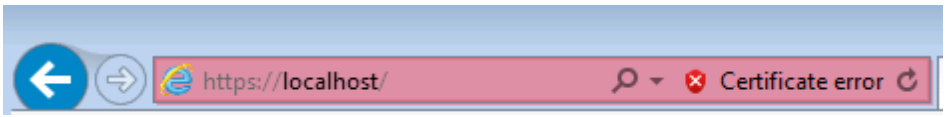
10.9.3 Confiar en certificados servidor en Windows

Para establecer confianza con un servidor web deberá obtener el archivo de certificado del servidor web e importarlo al almacén de certificados del sistema (véase https://technet.microsoft.com/en-us/library/cc754489(v=ws.11).aspx. Una manera de obtener el certificado servidor es con el explorador, como puede ver en el siguiente ejemplo.

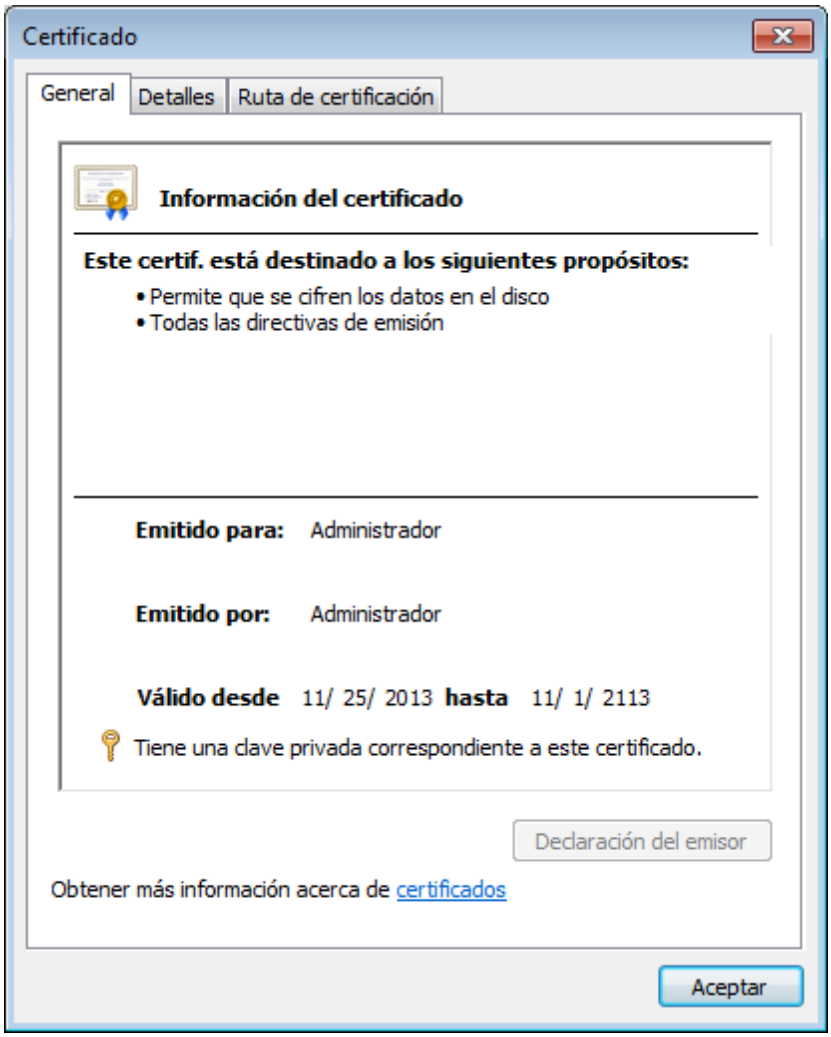
Para establecer confianza con un servidor web en Internet Explorer 11:

> Siga estas instrucciones solamente si está seguro de la autenticidad del certificado del servidor web.

1. En la barra de dirección del explorador introduzca la dirección HTTPS del servidor web.
2. Haga clic en **Pasar a este sitio web (no recomendado)**.
3. Haga clic en el área *Error de certificado* y después en **Ver certificados**.

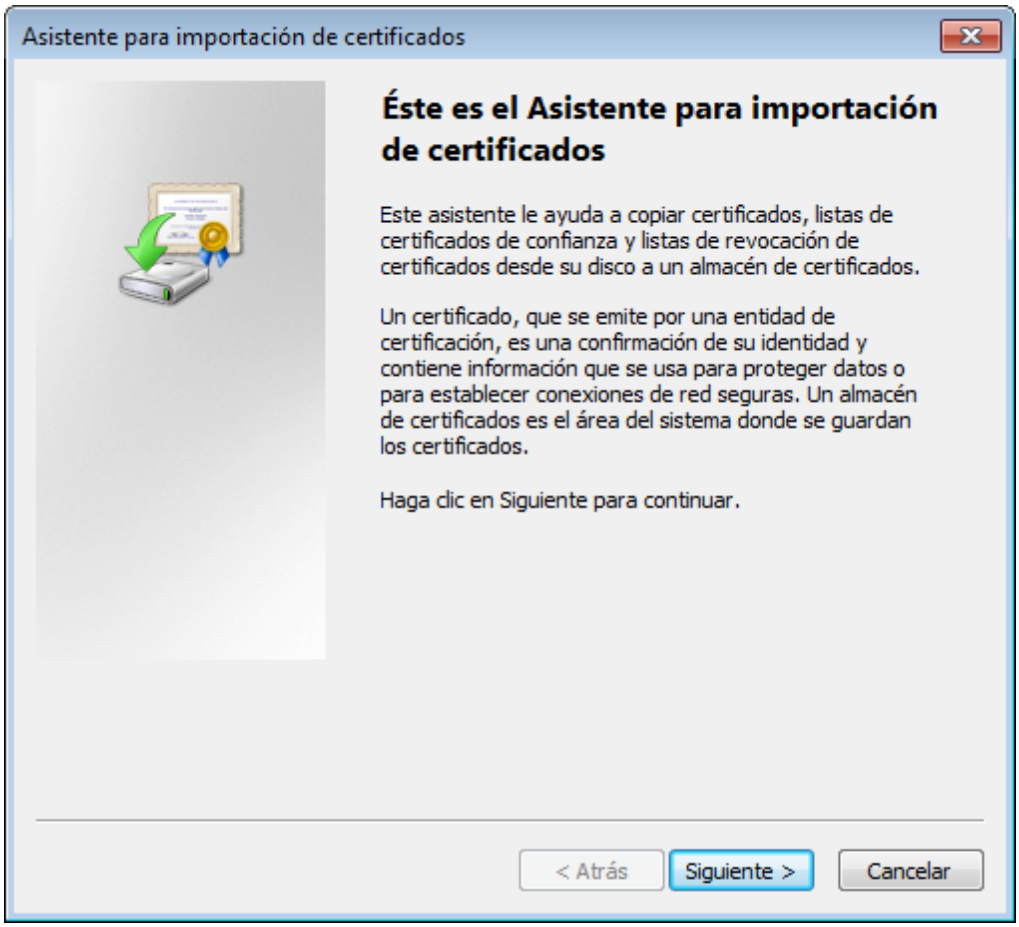


4. Haga clic en **Instalar certificado**.

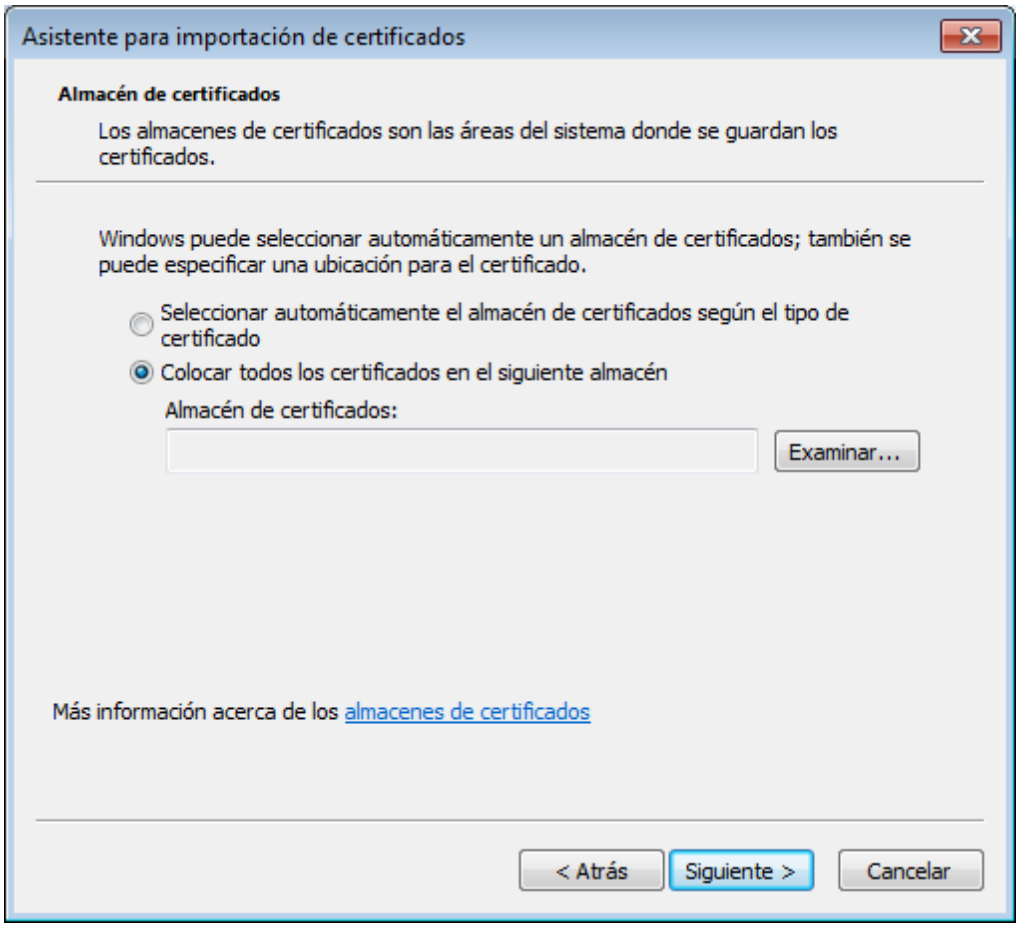


5. Haga clic en **Siguiente**.

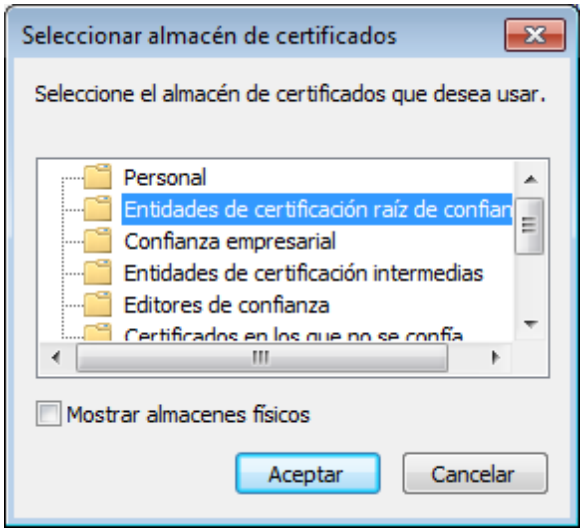
© 2018 Altova GmbH



6. Elija la opción que permite seleccionar el almacén a mano.



7. Busque las entidades de certificación raíz de confianza y haga clic en **Aceptar**.



8. Haga clic en **Finalizar** para confirmar y terminar.

© 2018 Altova GmbH

10.9.4 Acceso a los almacenes de certificados en Windows

En Windows puede gestionar certificados desde el complemento *Certificados* de Microsoft Management Console (MMC) o desde Internet Explorer.

Para abrir el complemento *Certificados* de MMC para el usuario actual de Windows:

- Ejecute `certmgr.msc` en la línea de comandos.

Para abrir el cuadro de diálogo de gestión de certificados en Internet Explorer:

1. En el menú **Herramientas** haga clic en **Opciones de Internet**.
2. Haga clic en la pestaña *Contenido* y después en *Certificados*.

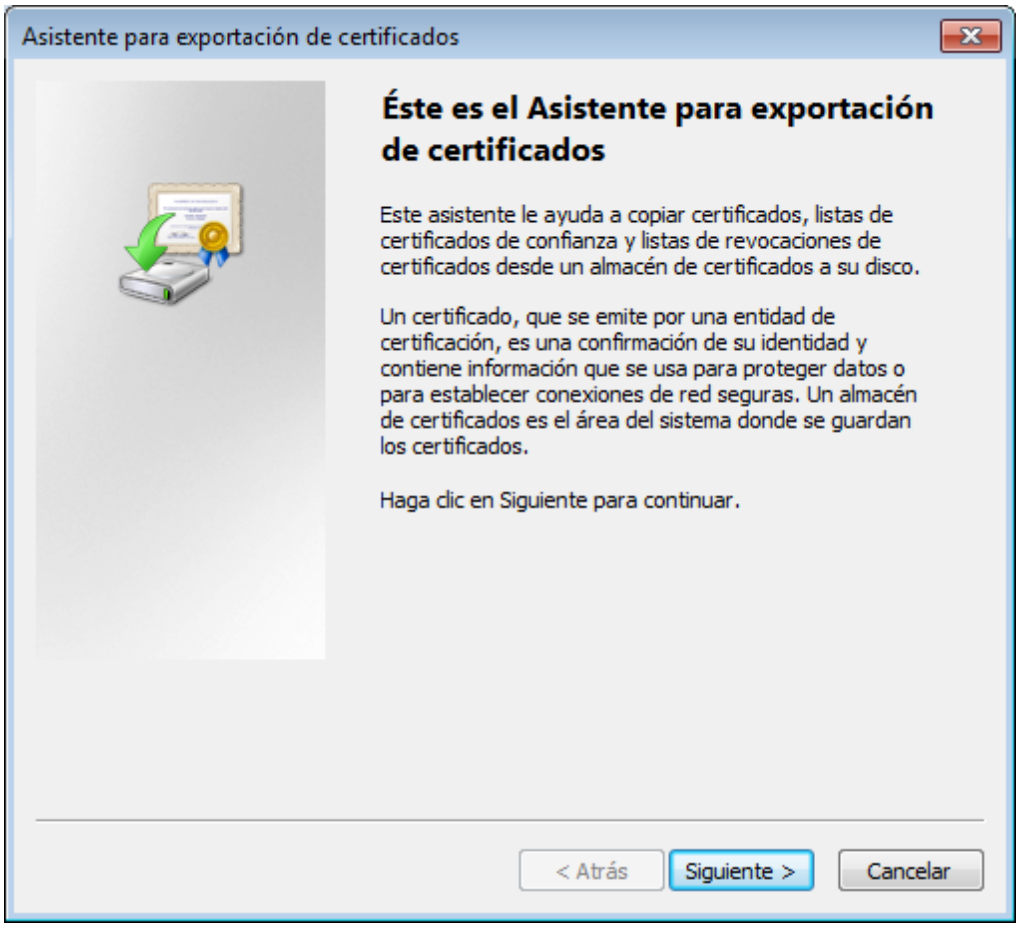
10.9.5 Exportar certificados de Windows

En el caso de asignaciones que llaman a servicios web por HTTPS e implementadas en servidores OS X / macOS o Linux que ejecutan MapForce Server o FlowForce Server, es necesario que el equipo no Windows tenga acceso al mismo certificado de cliente que se utilizó en Windows para diseñar y probar la asignación.

Para ejecutar estas asignaciones en sistemas operativos no Windows con MapForce Server, basta con exportar el certificado necesario con clave privada desde Windows e importarlo al sistema operativo de destino.

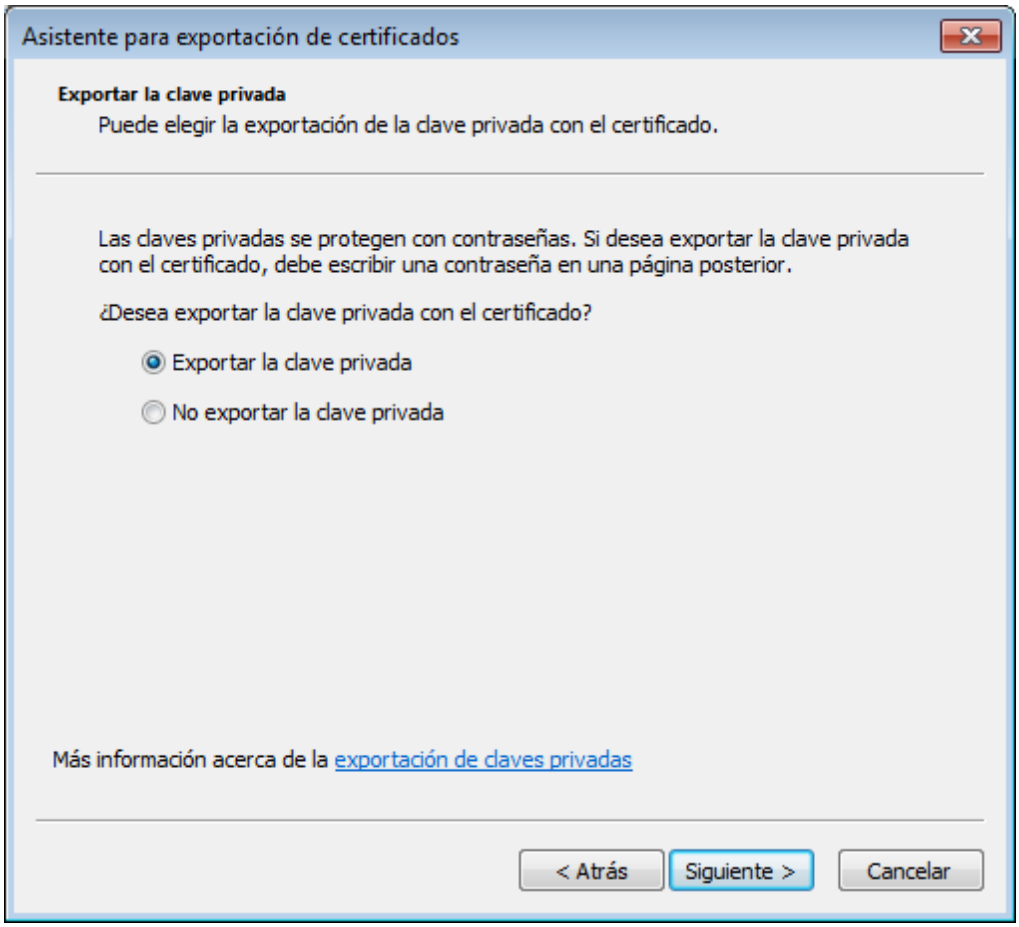
Para exportar un certificado con clave privada desde Windows:

1. En Windows abra el complemento *Certificados* (*véase* *Acceso a los almacenes de certificados en Windows*).
2. Haga clic con el botón derecho en el certificado que desea exportar, apunte a **Todas las tareas** y después haga clic en **Exportar**.
3. Ahora haga clic en **Siguiente**.

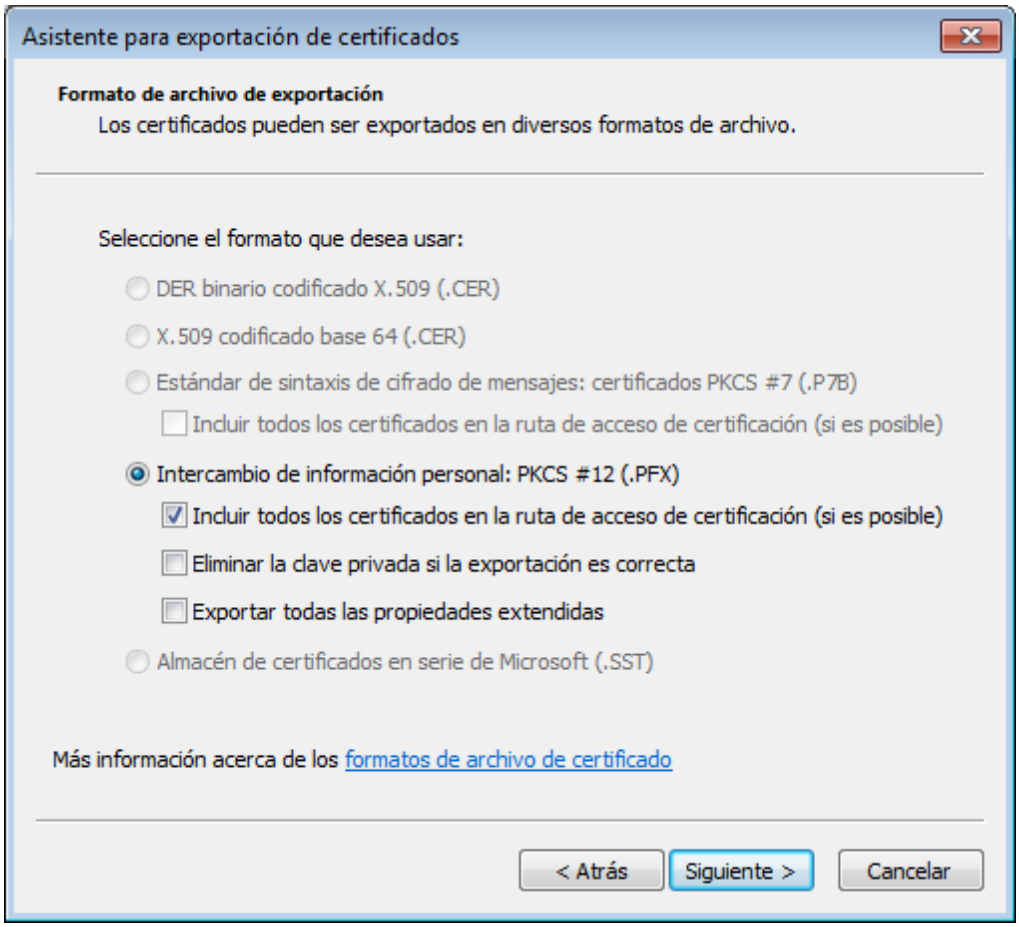


4. Elija la opción que permite exportar el certificado junto con su clave privada y después haga clic en **Siguiente**.

© 2018 Altova GmbH



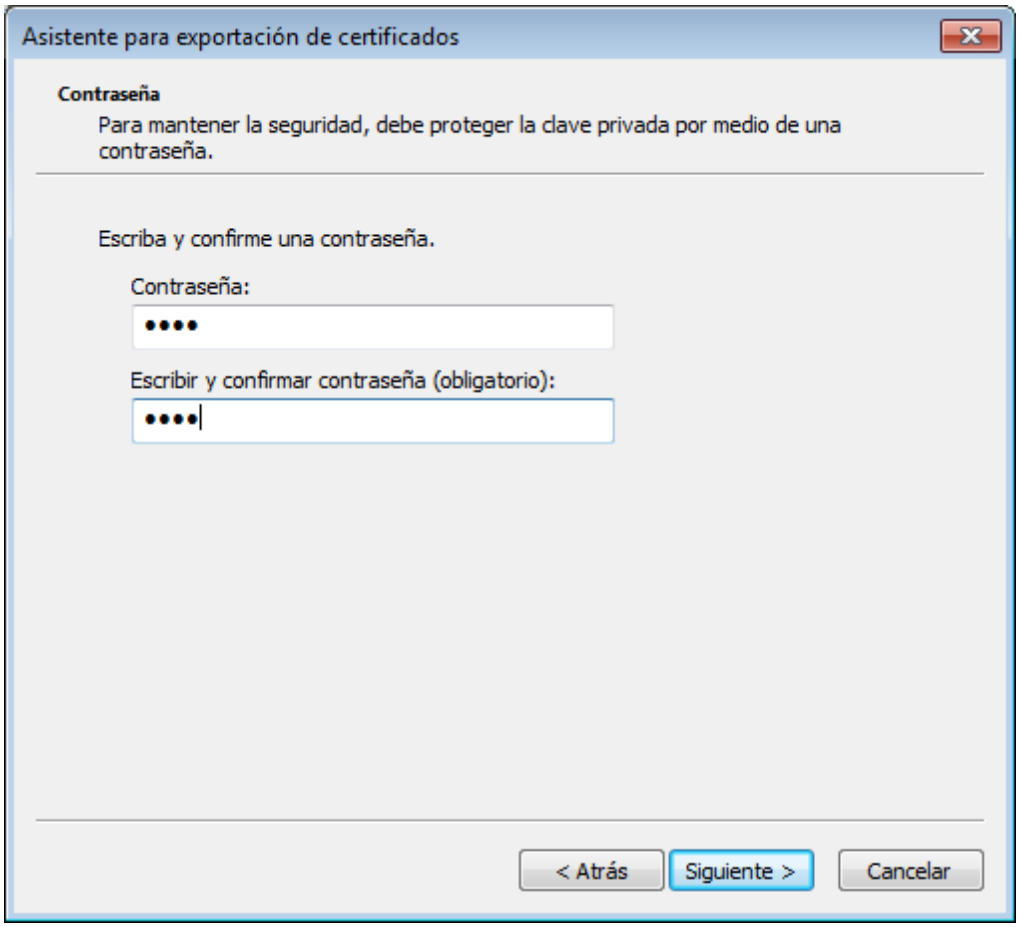
5. Elija el formato de archivo *Personal Information Exchange - PKCS #12 (.pfx)* y después haga clic en **Siguiente**.



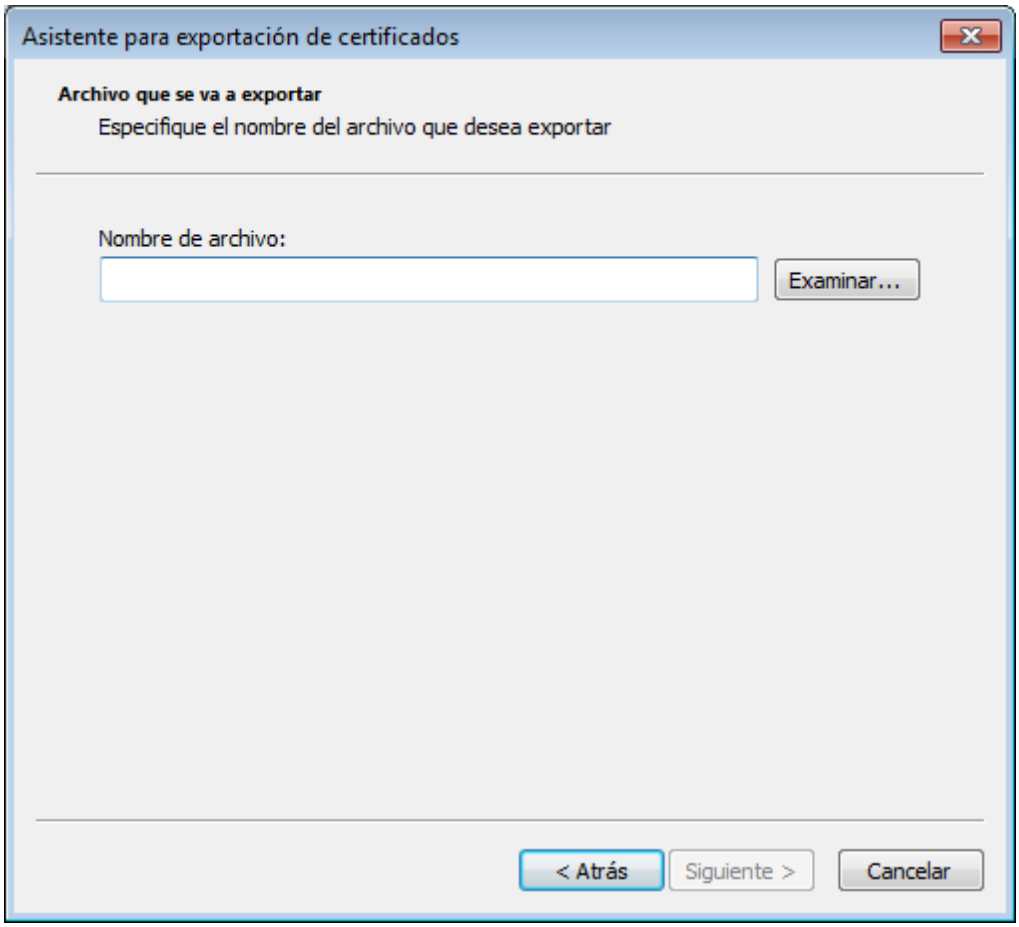
Nota: compruebe que la opción *Eliminar la clave privada si la exportación es correcta* **no está marcada**. De lo contrario no podrá volver a usar el certificado una vez exportado.

6. Introduzca la contraseña y haga clic en **Siguiente**. Esta es la contraseña que necesitará después de copiar el certificado al sistema operativo de destino.

© 2018 Altova GmbH

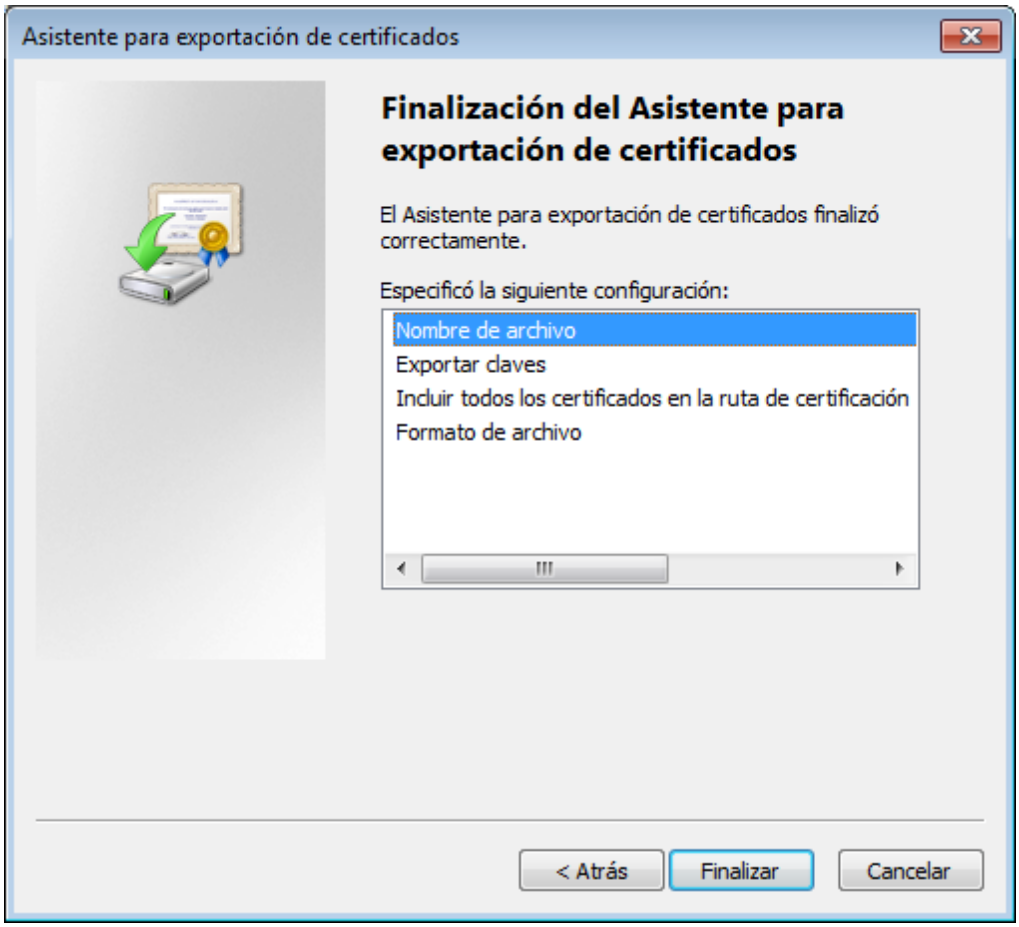


7. Busque el archivo que desea exportar y haga clic en **Siguiente**.



8. Para terminar haga clic en **Finalizar**.

© 2018 Altova GmbH



10.9.6 Certificados de cliente en Linux

Si sus asignaciones incluyen autenticación de servicios web por HTTPS mediante certificados de cliente, siga estos pasos para implementar dichas asignaciones en el equipo Linux que ejecuta MapForce Server:

1. Abra la asignación que llama al servicio web.
2. Haga doble clic en el título del componente de servicio web para abrir el cuadro de diálogo "Configuración del componente".
3. Haga clic en el botón **Editar** situado junto a *Configuración de seguridad HTTP*.
4. En el cuadro de diálogo "Configuración de seguridad HTTP" marque la casilla *Certificado de cliente* y después seleccione el certificado necesario en el almacén **Usuario actual/ Personal** de Windows (*véase* *Configuración de seguridad HTTP*).
5. Guarde y compile la asignación en un archivo de ejecución o impleméntela en FlowForce Server (*véase* *Compilar asignaciones en archivos de ejecución* e *Implementar asignaciones en FlowForce Server*).
6. Transfiera el certificado de cliente que necesita la llamada al servicio web al sistema operativo de destino. Compruebe que el certificado tiene una clave privada y que la propiedad *Uso mejorado de claves* del certificado incluye *Autenticación de cliente*.

Para transferir el certificado de cliente a Linux:

1. Exporte el certificado de cliente con clave privada desde Windows en formato de archivo *Personal Information Exchange - PKCS #12 (.pfx)* (*véase Exportar certificados de Windows*).
2. Copie este archivo al equipo Linux.
3. Convierta el archivo .pfx al formato .pem con este comando:

```
openssl pkcs12 -in cert.pfx -out "John Doe.pem" -nodes
```

Este comando analiza el archivo .pfx y genera un archivo .pem sin cifrar la clave privada. Los certificados con clave privada cifrada piden una contraseña y no son compatibles con ejecuciones en el servidor.

Ejecutar la asignación

Para instruir a MapForce Server que utilice el archivo .pem como certificado cliente, configure el parámetro `--certificatespath` cuando ejecute la asignación. El parámetro `--certificatespath` define la ruta de acceso del directorio donde se almacenan los certificados que necesita la asignación actual. Por ejemplo, si la ruta de acceso del certificado es `/home/John/John Doe.pem`, entonces `--certificatespath` debe configurarse con el valor `/home/John`.

Si no se indica el parámetro `--certificatespath`, el comportamiento predeterminado de MapForce Server es buscar certificados en el directorio `$HOME/.config/altova/certificates` del usuario actual.

> Para que la asignación se ejecute correctamente el archivo del certificado debe tener la extensión .pem y el nombre de archivo debe coincidir con el nombre común (CN) del certificado, incluidos los espacios (p. ej. **John Doe.pem**). Si el nombre común contiene una barra diagonal (/), sustitúyala por un carácter de subrayado (_).

Si tiene pensado ejecutar la asignación como trabajo de FlowForce Server, copie el archivo del certificado en el directorio `$HOME/.config/altova/certificates`. Cuando ejecute el trabajo, FlowForce Server utilizará este directorio para buscar los archivos de certificados que necesite la asignación.

> Por motivos de seguridad, compruebe que los archivos de certificado no pueden ser leídos por otros usuarios porque contienen información confidencial.

10.9.7 Certificados de cliente en OS X / macOS

Si sus asignaciones de datos incluyen autenticación de servicios web a través de certificados de cliente HTTPS, siga estos pasos para implementar estas asignaciones en un OS X que ejecute MapForce Server:

© 2018 Altova GmbH

1. Abra la asignación que llama al servicio web.
2. Haga doble clic en el título del componente de servicio web para abrir el cuadro de diálogo "Configuración del componente".
3. Haga clic en el botón **Editar** situado junto a *Configuración de seguridad HTTP*.
4. En el cuadro de diálogo "Configuración de seguridad HTTP" marque la casilla *Certificado de cliente* y después seleccione el certificado necesario (*véase* *Configuración de seguridad HTTP*).
5. Si el nombre del certificado no coincide totalmente con el nombre de host del servidor, marque la casilla *Permitir diferencias entre el nombre de host del certificado y el de la solicitud*.
6. Guarde e implemente la asignación en el sistema operativo de destino (*véase* *Implementar una asignación de MapForce*).
7. Transfiera el certificado de cliente que necesita la llamada al servicio web al sistema operativo de destino. Compruebe que el certificado tiene una clave privada y que la propiedad *Uso mejorado de claves* del certificado incluye *Autenticación de cliente*.

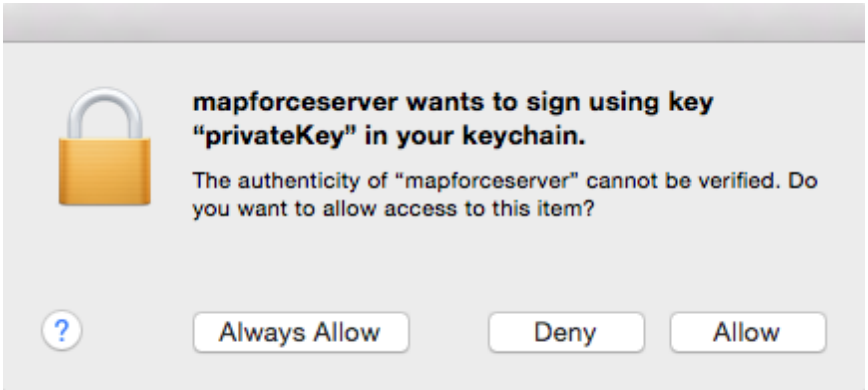
Para transferir el certificado de cliente a OS X / macOS:

1. Exporte el certificado de cliente con clave privada desde Windows en formato de archivo *Personal Information Exchange - PKCS #12 (.pfx)* (*véase* *Exportar certificados de Windows*) y copie este archivo a OS X / macOS.
2. Si todavía no lo ha hecho, compruebe que el sistema operativo confía en el certificado de servidor (*véase* Confiar en certificados servidor en OS X / macOS).
3. Ejecute Keychain Access desde **Finder > Applications > Utilities**.
4. En el menú **File** haga clic en **Import Items**.
5. Busque el certificado de cliente que exportó de Windows en el paso nº1 y seleccione las llaves de destino.
6. Haga clic en **Open** para abrir e introducir la contraseña con la que se cifró el certificado.

Ejecutar la asignación

Ahora ya puede ejecutar la asignación con el comando `run` de MapForce Server, pero tenga en cuenta estos aspectos:

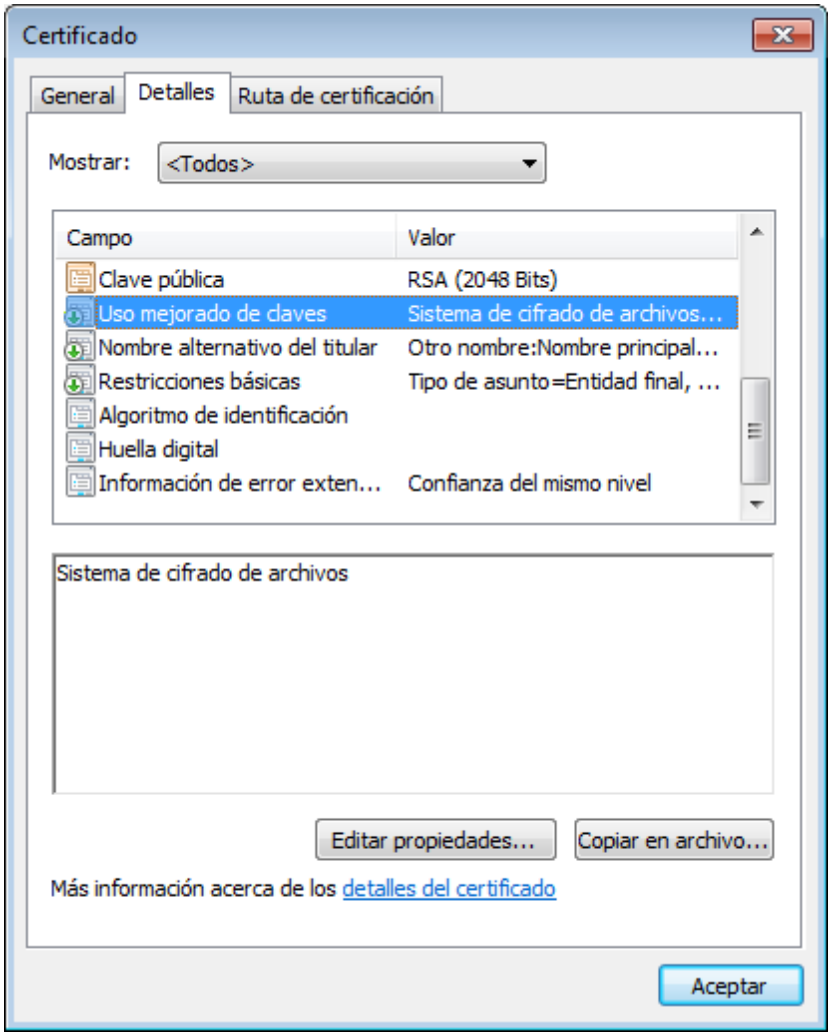
- Si ejecuta la asignación de forma remota por SSH, primero debe desbloquear las llaves con el comando `security unlock-keychain` command.
- Si ejecuta la asignación desde la interfaz del usuario gráfica de OS X / macOS, haga clic en **Allow** para permitir a MapForce Server acceder a las llaves.



10.9.8 Certificados de cliente en Windows

Si ejecuta en Windows una asignación que requiere certificados de cliente, lo requisitos para ejecutar correctamente la asignación son:

- El certificado de cliente debe existir en el almacén de certificados Usuario actual/ Personal. Para que el certificado exista en este almacén, debe ser importado con el asistente de importación de certificados (más información en https://technet.microsoft.com/es-es/library/cc754489(v=ws.11).aspx).
- El certificado debe tener una clave privada.
- La propiedad *Uso mejorado de claves* debe incluir *Autenticación de cliente*.



La biblioteca que utiliza la versión actual de Altova MapForce tiene algunas limitaciones. Por ese motivo Windows seleccionará automáticamente el certificado necesario del almacén de certificados cuando ejecute la asignación. La asignación se ejecutará correctamente si, tras filtrar el almacén de certificados **Usuario actual/Personal**, el servidor encuentra un certificado adecuado. Recuerde que de la autenticación HTTPS (y de la operación de selección del

© 2018 Altova GmbH

certificado) se ocupa Windows y no Altova MapForce ni MapForce Server. En algunos casos, si existen varios certificados en el almacén de certificados **Usuario actual/Personal**, puede que el sistema operativo seleccione automáticamente un certificado inadecuado. Esto puede dar lugar a un error en la ejecución de la asignación. Esto puede evitarse limitando el número de certificados disponibles en el almacén de certificados **Usuario actual/Personal**.

Altova MapForce 2018 Enterprise Edition

Automatizar asignaciones de datos

11 Automatizar asignaciones de datos

Las asignaciones diseñadas en MapForce se pueden ejecutar en un entorno servidor (incluidos servidores Linux y macOS) y en estos motores de transformación de Altova que ofrecen un rendimiento nivel servidor (y cuyas licencias deben comprarse por separado):

- *RaptorXML Server:* la ejecución de asignaciones en este motor es ideal si el lenguaje de transformación de la asignación es XSLT 1.0, XSLT 2.0 o XQuery (*véase Automatización con RaptorXML Server*).
- *MapForce Server (o MapForce Server Advanced Edition):* este motor de transformación es adecuado para las asignaciones cuyo lenguaje de transformación sea el motor integrado BUILT-IN*. El lenguaje BUILT-IN es compatible con la mayoría de las características de MapForce, mientras que MapForce Server (y en particular MapForce Server Advanced Edition) ofrece un rendimiento óptimo a la hora de ejecutar asignaciones de datos(*véase Automatización con MapForce Server*).

** el lenguaje de transformación BUILT-IN está disponible en las ediciones Professional y Enterprise de MapForce.*

Además, MapForce permite automatizar la generación de código XSLT, XQuery, C#, C++ y Java desde la interfaz de la línea de comandos. Esto incluye la compilación de archivos de ejecución (.mfx) para la ejecución con MapForce Server. Consulte la sección Interfaz de la línea de comandos de MapForce para obtener más información.

© 2018 Altova GmbH

11.1 Automatización con RaptorXML Server

RaptorXML Server (en adelante RaptorXML) es el procesador XML y XBRL de alta velocidad y de tercera generación de Altova. Está optimizado para los estándares más recientes y para entornos de computación en paralelo. RaptorXML aprovecha la actual omnipresencia de equipos multinúcleo para ofrecer rapidísimas funciones de procesamiento de datos XML y XBRL.

Altova ofrece dos ediciones distintas de RaptorXML, que se puede descargar e instalar desde la página de descargas de Altova (https://www.altova.com/es/download-trial-server.html)::

- RaptorXML Server es un motor de procesamiento XML ultrarápido compatible con XML, XML Schema, XSLT, XPath, XQuery y muchos otros formatos. Esta edición viene con el paquete de instalación de FlowForce Server.
- RaptorXML+XBRL Server ofrece todas las características de RaptorXML Server además de funciones de procesamiento y validación compatibles con toda la gama de estándares XBRL.

Si genera código en XSLT 1.0 o 2.0 o en XQuery, MapForce crea un archivo de procesamiento por lotes llamado `.bat` que se guarda en la carpeta de salida que usted elija. Cuando se ejecuta, este archivo de procesamiento por lotes llama a RaptorXML Server y ejecuta la transformación XSLT (o XQuery) en el servidor.

Si tiene pensado ejecutar o automatizar asignaciones de datos de MapForce para otro formato de salida consulte el apartado Automatización con MapForce Server.

Nota: también puede obtener una vista previa del código XSLT y del código XQuery con el motor integrado.

11.2 Automatización con MapForce Server

MapForce Server (https://www.altova.com/es/mapforce/mapforce-server.html) es un producto servidor para empresas compatible con sistemas Windows, Linux y OS X / macOS. MapForce Server se encarga de ejecutar asignaciones en un entorno servidor (incluso en plataformas no Windows) con un alto rendimiento. Puede ejecutar en el servidor cualquier asignación de MapForce que use el motor integrado como lenguaje de ejecución de destino (véase Seleccionar un lenguaje de transformación). MapForce Server puede funcionar en modo independiente (invocado desde la línea de comandos o desde la API) o como módulo de FlowForce Server.

Si decide utilizar MapForce Server en modo independiente, deberá compilar la asignación de datos de MapForce y copiarla en el equipo donde se ejecuta MapForce Server. La asignación se ejecutará con el comando `run` de la línea de comandos de MapForce Server. También puede ejecutar la asignación invocando el método `run` de la API de MapForce Server. Para más información consulte el apartado *Compilar asignaciones en archivos de ejecución de MapForce Server*.

Si decide ejecutar MapForce Server bajo la dirección de FlowForce Server, podrá implementar la asignación de datos en un equipo de destino a través de una conexión HTTP (o SSL/HTTPS) desde MapForce directamente. En el servidor podrá ejecutar la asignación en forma de trabajo programado o mediante una llamada a servicio web definida desde la interfaz de administración de FlowForce Server. Para más información consulte el apartado *Implementar asignaciones en FlowForce Server*.

Altova ofrece dos ediciones distintas de MapForce Server:

- MapForce Server
- MapForce Server Advanced Edition (esta edición forma parte del paquete de instalación de FlowForce Server)

La edición MapForce Server Advanced Edition ofrece las mismas características que MapForce Server pero está optimizada para determinado tipo de asignaciones, como aquellas que combinan o filtran gran cantidad de datos. El objetivo es aumentar la velocidad de ejecución.

Limitaciones:

- Incompatible con firmas XML digitales.
- Las conexiones de base de datos ADO, ADO.NET y ODBC solo son compatibles con Windows (*véase Conexiones de base de datos en Linux y Mac*).

© 2018 Altova GmbH

11.3 Preparar asignaciones para ejecutarlas en servidores

Una asignación de datos diseñada y probada con MapForce puede hacer referencia a recursos (p. ej. bases de datos) situados fuera del equipo actual y en otro sistema operativo. Además, en MapForce, las rutas de acceso de las asignaciones de datos siguen por defecto las normas de Windows. Y, por último, el equipo donde se ejecuta MapForce Server puede ser compatible o no con las conexiones de base de datos con las que era compatible el equipo donde se diseñó la asignación de datos. Por todos estos motivos, será necesario preparar las asignaciones para ejecutaras en el entorno del servidor, sobre todo si el equipo de destino no es el mismo que el equipo de origen.

Nota: el término "equipo de origen" se usa para designar al equipo donde está instalado MapForce, mientras que "equipo de destino" se refiere al equipo donde está instalado MapForce Server o FlowForce Server. En los casos más sencillos, se trata del mismo equipo, pero en casos más complejos sería más posible ejecutar MapForce en un equipo Windows y ejecutar MapForce Server o FlowForce Server en un equipo Linux o OS X / macOS.

Por todo ello, lo más recomendable es comprobar que la asignación es válida (desde MapForce) antes de implementarla en FlowForce Server o compilarla en un archivo de ejecución de MapForce Server (*véase* *Validar asignaciones*).

Si MapForce Server se ejecuta de forma independiente (y no junto con FlowForce Server), estas son las licencias que necesitará:

- En el equipo de origen, necesitará la edición MapForce Enterprise o Professional para diseñar la asignación y compilarla en un archivo de ejecución de MapForce Server (.mfx), como se explica en el apartado Compilar asignaciones en archivos de ejecución de MapForce Server.
- En el equipo de destino, necesitará la edición MapForce Server o MapForce Server Advanced Edition para ejecutar la asignación de datos.

Si MapForce Server se gestiona desde FlowForce Server, entonces necesitará:

- En el equipo de origen, necesitará la edición MapForce Enterprise o Professional para diseñar la asignación e implementarla en el equipo de destino, como se explica en el apartado Implementar asignaciones en FlowForce Server.
- Tanto MapForce Server como FlowForce Server deben tener asignada una licencia en el equipo de destino. MapForce Server se encarga de ejecutar la asignación de datos, mientras que FlowForce Server sirve la asignación en forma de trabajo, lo cual permite acceder a numerosas características (como la ejecución bajo demanda o programada, la ejecución como servicio web, control de errores, procesamiento condicional, notificaciones por correo electrónico, etc.).
- FlowForce Server debe estar en funcionamiento en la dirección de red y puerto configurados. Es decir, el servicio "FlowForce Web Server" debe estar iniciado y configurado para aceptar conexiones de clientes HTTP (o HTTPS si corresponde) y no debe estar bloqueado por el servidor de seguridad. El servicio "FlowForce Server" también debe estar iniciado y en ejecución en la dirección y puerto designados.
- Debe tener una cuenta de usuario de FlowForce Server con permiso para acceder a uno de los contenedores (al contenedor **/public** pueden acceder todos los usuarios que inicien sesión).

Si el equipo de origen no es el mismo que el de destino, deberá tener en cuenta que:

- Si tiene pensado ejecutar la asignación en un equipo de destino con una versión independiente de MapForce Server, deberá copiar todos los archivos y esquemas de entrada y salida a los que haga referencia la asignación al equipo de destino. Si por el contrario MapForce Server se gestiona desde FlowForce Server, entonces no es necesario copiar estos archivos a mano porque los archivos de instancia y de esquema se incluyen en el paquete que se implementa en el equipo de destino, como se explica en el apartado Implementar asignaciones en FlowForce Server.
- Si la asignación incluye componentes de base de datos que necesitan controladores de base de datos específicos, dichos controladores deberán estar instalados en el equipo de destino también. Por ejemplo, si la asignación lee datos de una base de datos de Microsoft Access, en el equipo de destino también deberá tener instalado Microsoft Access o Microsoft Access Runtime (https://www.microsoft.com/es-es/download/details.aspx?id=50040).
- Cuando se implementa una asignación en plataformas que no son Windows, las conexiones de base de datos ADO, ADO.NET y ODBC se cambian automáticamente por JDBC (ver siguiente punto de este apartado).
- Si la asignación contiene llamadas a funciones personalizadas (por ejemplo, a archivos .dll o .class), dichas dependencias no se implementan junto con la asignación porque no se conocen antes del tiempo de ejecución. En dichos casos, puede copiarlas a mano en el equipo de destino.
- Algunas asignaciones leen varios archivos de entrada con ayuda de una ruta de acceso comodín (véase Procesar varios archivos de entrada o salida de forma dinámica). En este caso el nombre de los archivos de entrada no se conocen antes del tiempo de ejecución y, por tanto, no se implementan. Para poder ejecutar la asignación correctamente los archivos de entrada deben existir en el equipo de destino.
- Si la asignación llama a un servicio web que requiere autenticación HTTPS con un certificado de cliente, el certificado debe transferirse al equipo de destino (*véase Gestión de certificados digitales*).
- Si la asignación se conecta a bases de datos basadas en archivos (como Microsoft Access y SQLite), deberá transferir a mano el archivo de base de datos al equipo de destino o guardarlo en un directorio compartido al que pueda acceder tanto el equipo de origen como el de destino (*ver más abajo*).

Notas sobre sistemas Linux y OS X / macOS

Si tiene pensado ejecutar la asignación en un servidor Linux o OS X / macOS, asegúrese de que la asignación sigue la convención sobre rutas de acceso correspondiente y que utiliza una conexión de base de datos compatible.

Para transferir rutas de acceso a sistemas operativos que no sean Windows utilice rutas de acceso relativas cuando diseñe la asignación en MapForce(*véase Usar rutas de acceso relativas y absolutas*). Por ejemplo, puede copiar todos los archivos de entrada y salida que necesita la asignación en el mismo directorio de la asignación y después hacerles referencia usando el nombre de archivo. Es importante saber que tanto MapForce Server como FlowForce Server funcionan con un *directorio de trabajo* con el que se resuelven todas las rutas de acceso relativas (*véase Rutas de acceso en diferentes entornos de ejecución*). El directorio de trabajo se especifica en tiempo de ejecución:

- En FlowForce Server se especifica editando el parámetro `Directorio de trabajo` de un

 © 2018 Altova GmbH

trabajo.
- En la API de MapForce Server API se especifica con la propiedad `WorkingDirectory` de la API de COM y .NET API o con el método `setWorkingDirectory` de la API de Java.
- En la línea de comandos de MapForce Server el directorio de trabajo es el directorio actual de la shell de comandos.

En cuanto a las conexiones de base de datos, tenga en cuenta que los equipos Linux y OS X / macOS no son compatibles con conexiones ADO, ADO.NET y ODBC. Por tanto, si el equipo de destino es Linux o OS X / macOS, estas conexiones se convertirán en conexiones JDBC cuando la asignación se implemente en FlowForce o cuando se compile en un archivo de ejecución de MapForce Server. Cuando esto ocurra, antes de implementar o compilar la asignación podrá elegir entre dos opciones:

- crear la conexión JDBC con la base de datos (*véase Configurar una conexión JDBC*) o
- rellenar los datos de conexión de base de datos JDBC en la sección *Configuración JDBC* del componente de base de datos (*véase Configurar componentes de base de datos*).

Si la asignación utiliza una conexión nativa a una base de datos PostgreSQL o SQLite, la conexión nativa se conserva y la conversión en JDBC no tiene lugar (*véase Asignaciones de datos de BD en varios entornos de ejecución*). Si la asignación se conecta a una base de datos basada en archivos (como Microsoft Access y SQLite) deberá configurar algunas opciones más (ver punto siguiente).

Bases de datos basadas en archivos

Las bases de datos basadas en archivos (como Microsoft Access y SQLite) no se incluyen en el paquete que se implementa en FlowForce Server ni en el archivo de ejecución que se compila para MapForce Server. Por tanto, si el equipo de origen y de destino no son el mismo, deberá seguir estas instrucciones:

1. En MapForce haga clic con el botón derecho en la asignación y en el cuadro de diálogo "Configurar asignación" desactive la casilla *Convertir las rutas de acceso en absolutas en el código generado* (*véase Cambiar configuración de la asignación*).
2. Haga clic con el botón derecho en el componente de base de datos de la asignación y añada una conexión al archivo de base de datos usando una ruta de acceso relativa (*véase Configurar la ruta de acceso de BD basadas en archivos*). La manera más sencilla de evitar problemas relacionados con la ruta de acceso es guardar el diseño de asignación (archivo .mfd) en el mismo directorio que el archivo de base de datos y hacer referencia a este último desde la asignación usando el nombre de archivo (es decir, una ruta de acceso relativa).
3. Copie el archivo de base de datos en un directorio del equipo de destino (que llamaremos *directorio de trabajo*). Recuerde el nombre de este directorio porque lo necesitará para ejecutar la asignación en el servidor (ver más abajo).

Hay varias maneras de ejecutar este tipo de asignaciones en el servidor:

- Si tiene pensado ejecutar la asignación con MapForce Server bajo control de FlowForce Server, configure el trabajo de FlowForce Server para que apunte al directorio de trabajo creado previamente. El archivo de base de datos debe residir en este directorio de trabajo (puede ver un ejemplo en el apartado Exponer un trabajo como servicio web de la documentación de FlowForce Server).
- Si tiene pensado ejecutar la asignación con MapForce Server de forma independiente y

desde la línea de comandos, cambie del directorio actual al directorio de trabajo (p. ej. `cd ruta\de\directorio\de\trabajo`) antes de llamar al comando `run` de MapForce Server.

- Si tiene pensado ejecutar la asignación con la API de MapForce Server, especifique el directorio de trabajo mediante programación antes de ejecutar la asignación. Para esto sirve la propiedad `WorkingDirectory` del objeto MapForce Server en la API de COM y .NET. En la API de Java se utiliza el método `setWorkingDirectory`.

Si tanto el equipo de origen como el de destino es un equipo Windows que se ejecuta en una red local, también puede configurar la asignación para que lea el archivo de base de datos de un directorio compartido común:

1. Guarde el archivo de base de datos en un directorio compartido común al que pueda acceder tanto el equipo de origen como el de destino.
2. En la asignación haga clic con el botón derecho en el componente de base de datos y añada una conexión al archivo de base de datos usando una ruta de acceso absoluta (*véase* *Configurar la ruta de acceso de BD basadas en archivos*).

Recursos globales

Si la asignación incluye referencias a recursos globales en lugar de rutas de acceso directas o conexiones de base de datos, dichas referencias se conservan cuando se compila la asignación en un archivo de ejecución de MapForce Server (.mfx) y cuando se implementa la asignación en FlowForce Server (*véase* *Recursos globales en distintos entornos de ejecución*).

Nota: FlowForce Server no compatible con la característica de recursos globales de Altova. Si tiene pensado ejecutar la asignación con MapForce Server bajo control de FlowForce Server, evite utilizar recursos globales.

Paquetes de taxonomías XBRL

Cuando implemente una asignación que remita a paquetes de taxonomías XBRL (*véase* *Paquetes de taxonomías XBRL*) en FlowForce Server, MapForce recopila todas las referencias externas que existen en la asignación y las resuelve usando la configuración actual y los paquetes de taxonomías que están instalados en ese momento. Si hay referencias externas sin resolver que apunten a un paquete de taxonomías, dicho paquete se implementará junto con la asignación. FlowForce Server utilizará ese paquete para ejecutar la asignación. Para actualizar el paquete de taxonomías que utiliza FlowForce Server deberá cambiarlo primero en MapForce y volver a implementar la asignación.

Recuerde que el catálogo raíz de MapForce Server influye en el modo en que se resuelven las taxonomías en el equipo de destino. El catálogo raíz se encuentra en esta ruta de acceso relativa al directorio de instalación de MapForce Server: **etc/RootCatalog.xml**.

Los paquetes de taxonomías que se implementaron con la asignación se usarán si el catálogo raíz de MapForce Server no incluye ya dichos paquetes o no contiene un paquete que se definiera para el mismo prefijo URL. El catálogo raíz de MapForce Server tiene prioridad sobre la taxonomía que se implementó.

Si ejecuta MapForce Server de forma independiente (y no bajo control de FlowForce Server), tiene la opción de especificar qué catálogo raíz debe utilizar la asignación:

© 2018 Altova GmbH

- En la línea de comandos se hace agregando la opción `-catalog` al comando `run`.
- En la API de MapForce Server se hace llamando al método `SetOption` y dando la cadena `"catalog"` como primer argumento y la ruta de acceso del catálogo raíz como segundo argumento.

Si la asignación utiliza componentes XBRL con bases de enlaces de tabla, entonces deberá aportar el paquete de taxonomías o el archivo de configuración del paquete de taxonomías a la asignación en tiempo de ejecución:

- En la línea de comandos de MapForce Server esto se hace agregando la opción `--taxonomy-package` o `--taxonomy-packages-config-file` al comando `run`.
- En la API de MapForce Server esto se hace llamando al método `SetOption`. El primer argumento debe ser `"taxonomy-package"` o `"taxonomy-packages-config-file"`. El segundo argumento debe ser la ruta de acceso real del paquete de taxonomías (o del archivo de configuración del paquete de taxonomías.

11.4 Compilar asignaciones en archivos de ejecución de MapForce Server

Si en la asignación de datos de MapForce eligió el motor de transformación integrado, la asignación se podrá ejecutar con MapForce y con MapForce Server (ver apartado anterior). Hay dos maneras de ejecutar asignaciones de datos con MapForce Server:

- Si MapForce Server se ejecuta en modo independiente (es decir, si no tiene FlowForce Server instalado), la asignación debe compilarse en un archivo de ejecución **.mfx** (ver más abajo). El archivo .mfx se puede ejecutar en la línea de comandos con el comando `run`. También puede ejecutarlo invocando el método `run` de la API de MapForce Server. Consulte la documentación de MapForce Server (https://manual.altova.com/es/MapForceServer/) para obtener más información.

- Si MapForce Server se ejecuta como módulo de FlowForce Server, la asignación se puede implementar en un equipo donde se ejecuten FlowForce Server y MapForce Server (*véase Implementar asignaciones en FlowForce Server*).

Requisitos

Consulte el apartado anterior (*Preparar asignaciones para ejecutarlas en servidores*).

Para compilar una asignación en un archivo de ejecución de MapForce Server (.mfx):

1. Abra una asignación de datos en MapForce (por ejemplo, **miAsignación.mfd**).
2. Haga clic en el comando de menú **Archivo | Compilar en archivo de ejecución de MapForce Server**.
3. Seleccione la carpeta donde desea guardar el archivo .mfx y cambie el nombre del archivo si lo necesita.
4. Haga clic en **Guardar**. El archivo de ejecución de MapForce Server **miAsignación.mfx** se genera en la carpeta seleccionada.

Para compilar una asignación en un archivo de ejecución de MapForce Server (.mfx) desde la línea de comandos:

- Ejecute MapForce desde la línea de comandos, especifique el archivo de asignación y la opción `/COMPILE` de la línea de comandos.

Por ejemplo, con el comando que aparece a continuación puede compilar la asignación **C:\Usuarios\altova\Documentos\Altova\MapForce2018\MapForceExamples\SimpleTotal.mfd** en un archivo de ejecución de MapForce Server que se creará en el directorio de salida **C:\Usuarios\altova\Desktop**.

```
"C:\Archivos de programa (x86)\Altova\MapForce2018\MapForce.exe" "C:\Usuarios
\altova\Documentos\Altova\MapForce2018\MapForceExamples\SimpleTotal.mfd" /
COMPILE "C:\Usuarios\altova\Desktop"
```

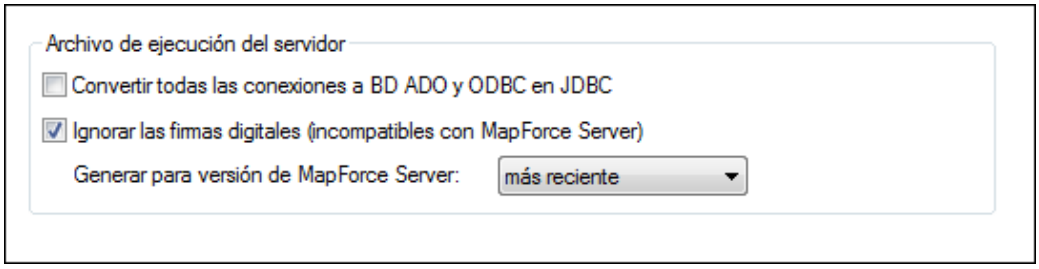

© 2018 Altova GmbH

Consulte la documentación de la interfaz de la línea de comandos de MapForce para obtener más información.

Compilar asignaciones para una versión concreta de MapForce Server

Si su versión de MapForce Server es anterior a su versión de MapForce, puede que MapForce Server no pueda ejecutar correctamente los archivos creados con la versión más reciente de MapForce. Si este es su caso, puede compilar el archivo .mfx para una versión concreta de MapForce Server:

1. Haga clic en el comando de menú **Herramientas | Opciones** y después elija **Generación de código**.
2. En el grupo de opciones *Archivo de ejecución para servidor,* junto al campo *Generar para versión de MapForce Server*, seleccione la versión de MapForce Server que necesita en la lista desplegable.



Cuando instale una versión más reciente de MapForce Server, recuerde que deberá cambiar esta opción según corresponda. También puede seleccionar la versión **más reciente** en la lista desplegable (opción predeterminada) y se compilará el archivo .mfx para la versión más reciente de MapForce Server (lo cual le permitirá beneficiarse de las mejoras y características más recientes del producto).

Para especificar una versión concreta de MapForce Server desde la línea de comandos debe ejecutar el comando `/COMPILE` con el conmutador `/MFXVERSION`.
Por ejemplo:

```
"C:\Archivos de programa (x86)\Altova\MapForce2018\MapForce.exe" /COMPILE /
MFXVERSION:2018
```

Consulte la documentación de la interfaz de la línea de comandos de MapForce para obtener más información.

Otras opciones

La compilación de archivos de ejecución de MapForce Server también puede verse afectada por estas opciones de la aplicación:

Convertir todas las conexiones a BD ADO y ODBC en JDBC	Si esta opción está habilitada, las conexiones de base de datos ADO, ADO.NET y ODBC se transforman en JDBC (usando el controlador JDBC y la URL de la base de datos definida en el cuadro de diálogo "Configuración del componente". *Véase Configuración de*

	componentes de base de datos). The JDBC connection will be used implicitly if the target machine is a Linux or macOS server (see Database Connections on Linux and Mac). La conexión JDBC se usará de forma implícita si el equipo de destino es un servidor Linux o macOS (*véase Conexiones de base de datos en Linux y Mac*).
Ignorar las firmas digitales (incompatibles con MapForce Server)	Esta opción solamente es relevante en MapForce Enterprise Edition y está habilitada por defecto. Si utiliza firmas XML digitales, la asignación omitirá la información sobre firmas digitales porque MapForce Server no es compatible con esta característica

Para ver o cambiar estas opciones:

- Haga clic en el comando de menú **Herramientas | Opciones** y después elija **Generación de código**.

También puede acceder a estas opciones desde la interfaz de la línea de comandos.

© 2018 Altova GmbH

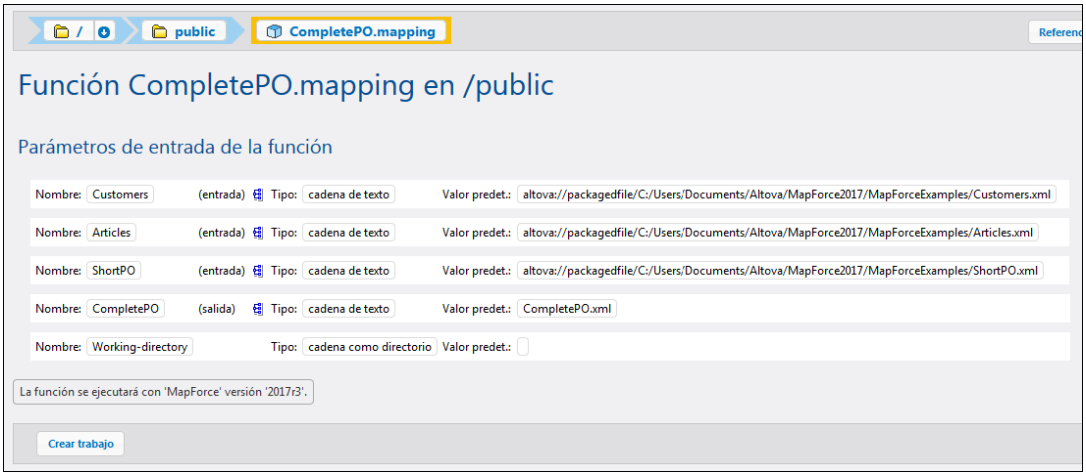
11.5 Implementar asignaciones en FlowForce Server

Cuando se implementa una asignación de datos en FlowForce Server, MapForce puede organizar los recursos que utiliza la asignación en un solo objeto y lo pasa por HTTP al equipo donde se ejecuta FlowForce Server. La implementación de asignaciones de MapForce en FlowForce Server permite automatizar su ejecución a través de trabajos. Tras implementar la asignación, podrá crear un trabajo de FlowForce Server basado en la asignación implementada y sacar partido a todas las características de los trabajos de FlowForce (p. ej. definir condiciones desencadenantes, exponer el trabajo como servicio web, etc.).

Nota: el término *equipo de origen* hace referencia al equipo donde está instalado MapForce, mientras que *equipo de destino* se utiliza para denominar el equipo donde está instalado FlowForce Server. Estos dos equipos pueden ser el mismo o, cuando se trata de entornos de trabajo más sofisticados, MapForce puede ejecutarse en un equipo Windows, por ejemplo, y FlowForce Server en un equipo Linux o OS X / macOS, por ejemplo.

El paquete que se implementa en FlowForce incluye:

- La asignación propiamente dicha. Tras la implementación, la asignación de datos estará disponible en la interfaz de administración de FlowForce Server en forma de función de asignación (.mapping) y en la ruta de acceso que se especificara. Los componentes de origen se convierten en argumentos de entrada y los componentes de destino se convierten en argumentos de salida de dicha función.



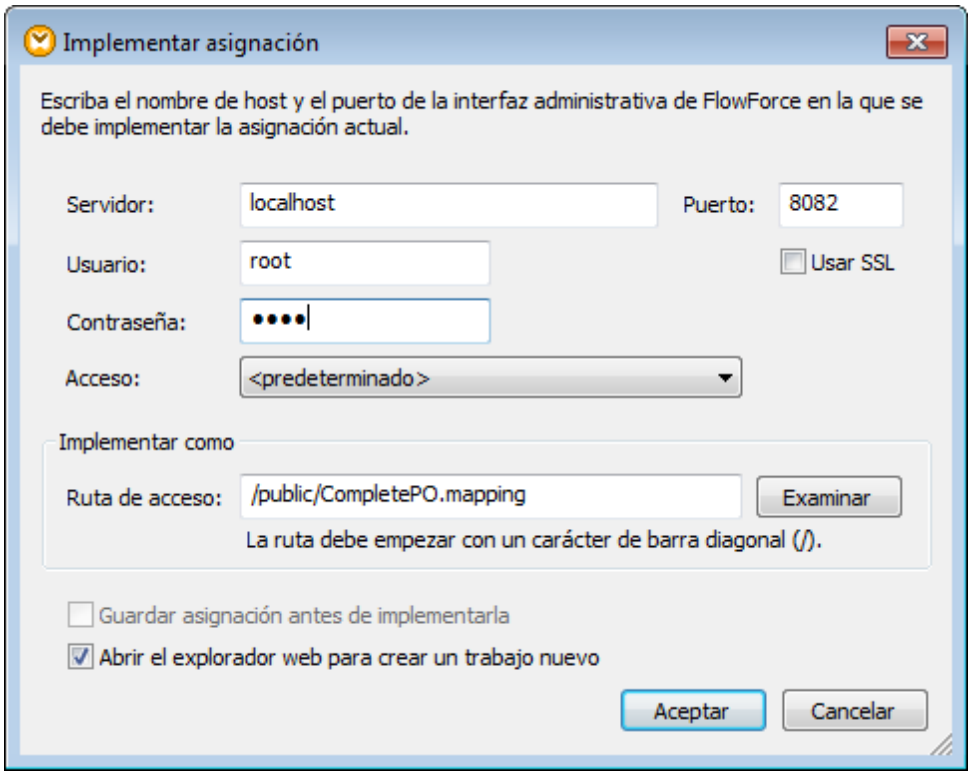
- Todo tipo de archivos de instancia de entrada (XML, CSV, Texto, EDI, Excel, JSON, XBRL) que son utilizados por la asignación de datos.
- Si la asignación utiliza configuraciones EDI y FlexText, éstas se implementan tal y como se encuentran en el equipo de origen en el momento de la implementación. Si tras la implementación modifica las configuraciones EDI o FlexText, deberá volver a implementar la asignación en FlowForce Server.
- Si la asignación utiliza taxonomías XBRL o paquetes de taxonomías XBRL, éstas se implementan tal y como se encuentran en el equipo de origen en el momento de la implementación (ver más abajo).

Requisitos

Consulte el apartado *Preparar asignaciones para ejecutarlas en servidores*.

Implementar la asignación en FlowForce Server

1. Compruebe que el lenguaje de transformación es BUILT-IN (*véase Seleccionar un lenguaje de transformación*).
2. Haga clic en el comando **Archivo | Implementar en FlowForce Server**. Aparece el cuadro de diálogo "Implementar asignación".



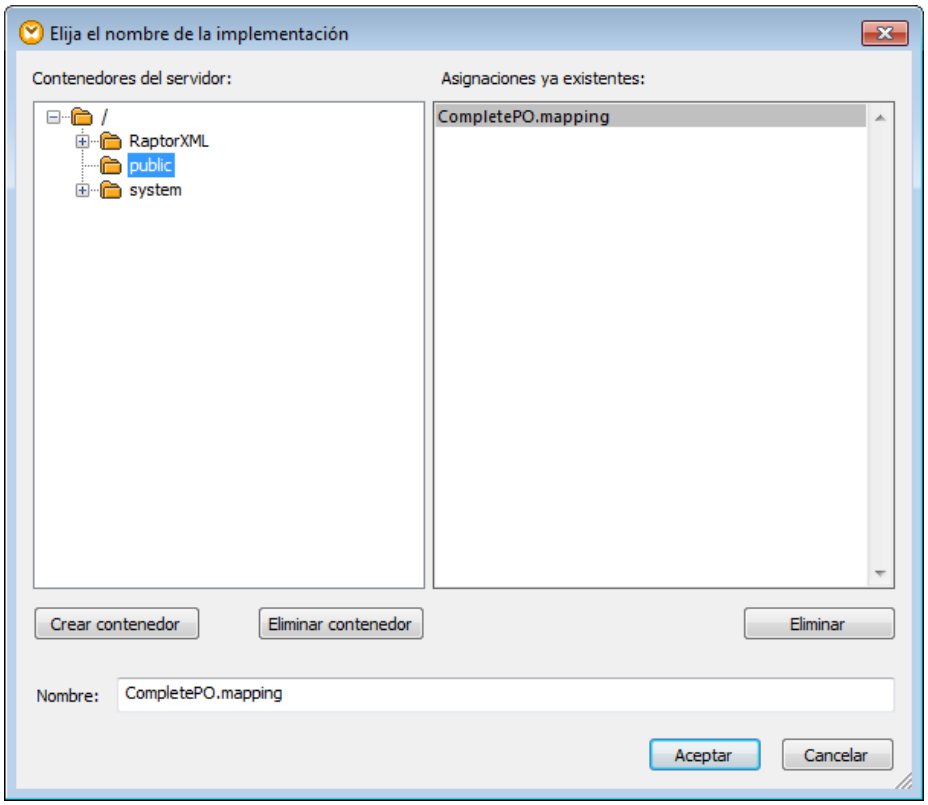
3. Seleccione las opciones de implementación (ver más abajo) y haga clic en **Aceptar**. Si marcó la casilla *Abrir el explorador web para crear un trabajo nuevo*, la interfaz de administración de FlowForce Server se abre en el explorador, donde podrá empezar a crear el trabajo de FlowForce.

A continuación describimos las opciones de implementación del cuadro de diálogo.

Opción	Descripción
Servidor y puerto	Introduzca el nombre de host (o dirección IP) y el puerto del servidor de FlowForce Server. Si FlowForce Server se ejecuta en el mismo equipo y en el puerto predeterminado, la dirección será **localhost** y el puerto **8082**. Si no sabe qué dirección y puerto usar, inicie sesión en la interfaz de administración web de FlowForce Server y consulte qué dirección IP y puerto aparece en la barra

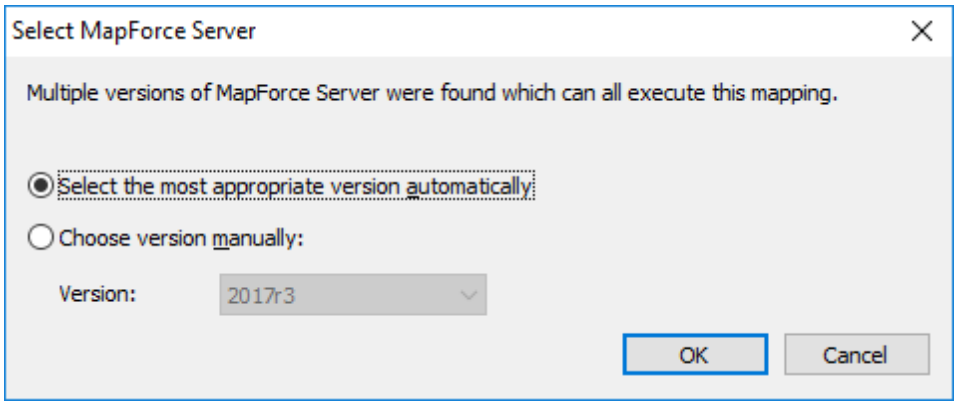
© 2018 Altova GmbH

	de dirección del explorador web. Si tiene problemas de conectividad, compruebe que el equipo donde se ejecuta FlowForce Server está configurado para admitir conexiones entrantes en el puerto y en la dirección designados. Para implementar la asignación a través de una conexión con cifrado SSL debe marcar la casilla *Usar SSL*. Al hacerlo, la aplicación entiende que FlowForce ya está configurado para aceptar conexiones SSL. Para más información consulte la documentación de FlowForce Server.
Usuario y contraseña	El usuario y la contraseña que debe introducir depende del valor de la opción *Acceso* (ver siguiente opción). Si en esta opción se selecciona **Predeterminado** o **Directo**, entonces debe introducir su nombre de usuario y contraseña de FlowForce Server. De lo contrario, introduzca su nombre de usuario y contraseña de Windows y seleccione el nombre de dominio de Windows en la opción *Acceso.*
Acceso	Si habilitó la integración con Windows Active Directorio en FlowForce Server, seleccione el nombre de dominio de Windows en la lista desplegable e introduzca sus credenciales de Windows en los campos *Usuario* y *Contraseña* (ver opción anterior).
Ruta de acceso	Haga clic en Examinar y seleccione la ruta de acceso donde se debe guardar la función de asignación dentro de la jerarquía de contenedores de FlowForce Server. La ruta de acceso predeterminada es el contenedor **/public** de FlowForce Server. En el cuadro de diálogo "Elija el nombre de implementación" también puede crear y eliminar contenedores y asignaciones de datos (siempre y cuando disponga de los permisos correspondientes).

	 *Cuadro de diálogo "Elija el nombre de implementación"*
Guardar asignación antes de implementarla	Esta opción está habilitada si la asignación que se intenta implementar está sin guardar. Marque esta casilla para guardarla antes de implementarla.
Abrir el explorador web para crear un trabajo nuevo	Si marca esta casilla, la interfaz de administración web de FlowForce Server se abrirá en el explorador una vez terminada la implementación y podrá empezar a crear un trabajo de FlowForce Server inmediatamente.

Si el servidor donde implementa la asignación ejecuta varias versiones de MapForce Server bajo la dirección de FlowForce Server (solo si se trata de servidores Windows), entonces la aplicación solicitará que especifique la versión de MapForce Server en la que tiene pensado ejecutar la asignación:

© 2018 Altova GmbH



Nota: este cuadro de diálogo aparece cuando el directorio de instalación de FlowForce Server contiene archivos .tool distintos para cada versión de MapForce Server que esté bajo control de FlowForce Server. Cuando se instala MapForce Server junto con FlowForce Server, se añade un archivo .tool a este directorio automáticamente. La ruta de acceso donde se almacenan los archivos .tool es: **C:\Archivos de programa\Altova \FlowForceServer2018\tools**. Si tiene otras versiones de MapForce Server y desea ejecutarlas bajo la dirección de FlowForce Server, deberá copiar a mano sus archivos .tool en el directorio mencionado. El archivo .tool de MapForce Server está en este directorio: **C:\Archivos de programa\Altova\MapForceServer2018\etc**.

11.6 Integración con AS2

AS2 (del inglés *Applicability Statement 2* o declaración de aplicabilidad) es una especificación que permite el intercambio seguro de archivos por Internet. AS2 se utiliza principalmente para el intercambio de archivos EDIINT (EDI a través de Internet) y de archivos XML por HTTP o HTTPS.

FlowForce Server Advanced Edition está diseñada para permitir el envío y la recepción de mensajes AS2 a y de socios comerciales. Gracias a sus funciones integradas FlowForce Server también permite procesar datos AS2 y almacenarlos de forma local. Además, si necesita preparar datos AS2 a partir de algún origen de datos (una BD, por ejemplo), convertirlos en otros formatos o enviarlos a un servicio web, también puede incluir MapForce y MapForce Server en el proceso AS2.

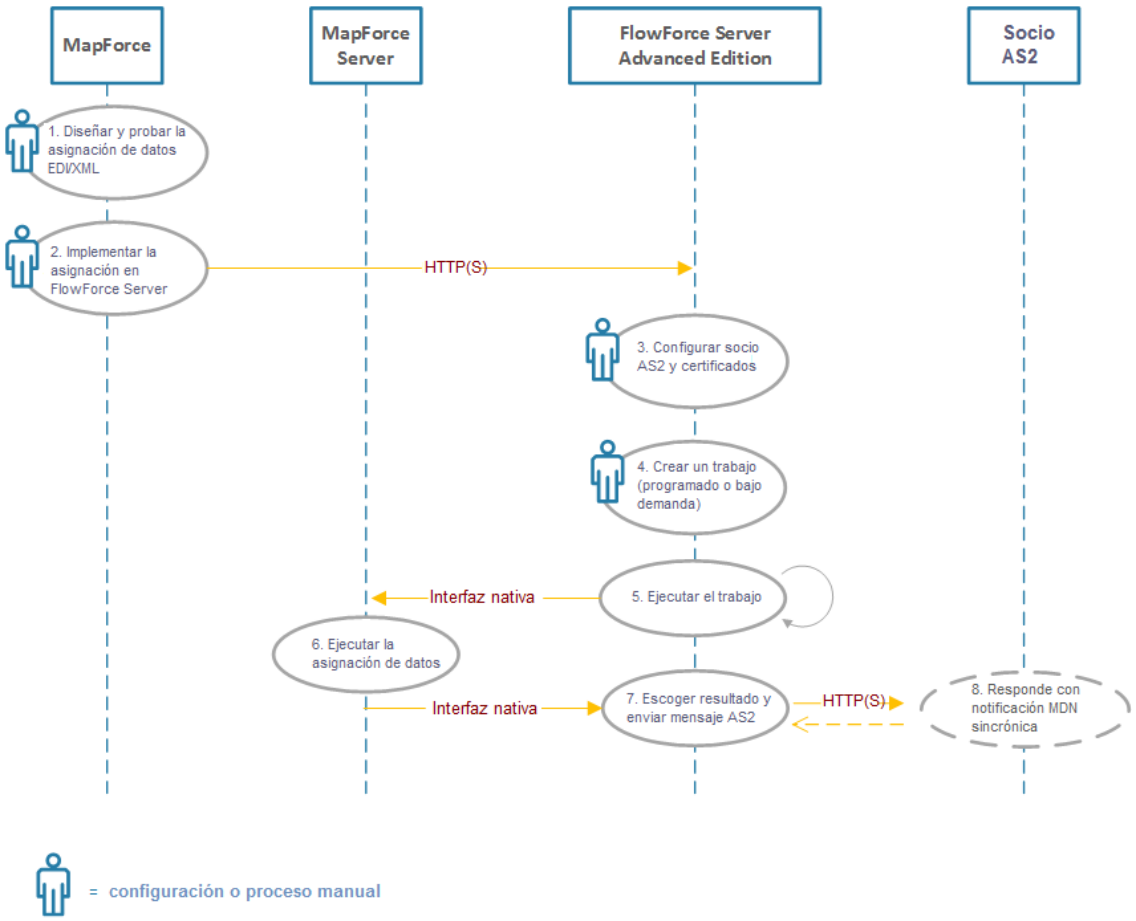
MapForce y MapForce Server son necesarias en los siguientes casos:

1. Para generar o realizar asignaciones de datos en cualquier formato compatible con MapForce (XML, XBRL, Excel, BD, servicios web) antes de enviárselos a socios AS2.
2. Para transformar los datos recibidos de socios AS2 de distintas maneras (por ejemplo convertirlos en Excel, en un esquema XML diferente, almacenarlos en una BD, enviarlos a un servicio web, etc.).

© 2018 Altova GmbH

Generar y enviar datos AS2

Si necesita preparar o generar datos AS2 con MapForce antes de enviárselos a socios, a grandes rasgos el proceso es el siguiente:



Generar y enviar datos AS

En el esquema tanto MapForce Server como FlowForce Server están instalados en el mismo equipo, que puede ser Windows, Linux o OS X. MapForce puede ejecutarse en el mismo equipo que MapForce Server y FlowForce Server (siempre y cuando se trate de un equipo Windows) o en otro equipo distinto que se pueda conectar a FlowForce vía HTTP o HTTPS. El socio AS2 es un servidor remoto con el que FlowForce Server se comunica por HTTP(S).

A continuación explicamos paso a paso el proceso AS2 del esquema anterior:

Paso	Descripción
1. Diseñar y probar la asignación de datos EDI/XML	Con MapForce puede diseñar una transformación de asignación de datos que tome datos de entrada en varios formatos (texto plano, CSV, JSON, XML, EDI, bases de datos, servicios web, etc.) y que genere archivos en un formato de destino (p. ej. UN/EDIFACT). Las asignaciones

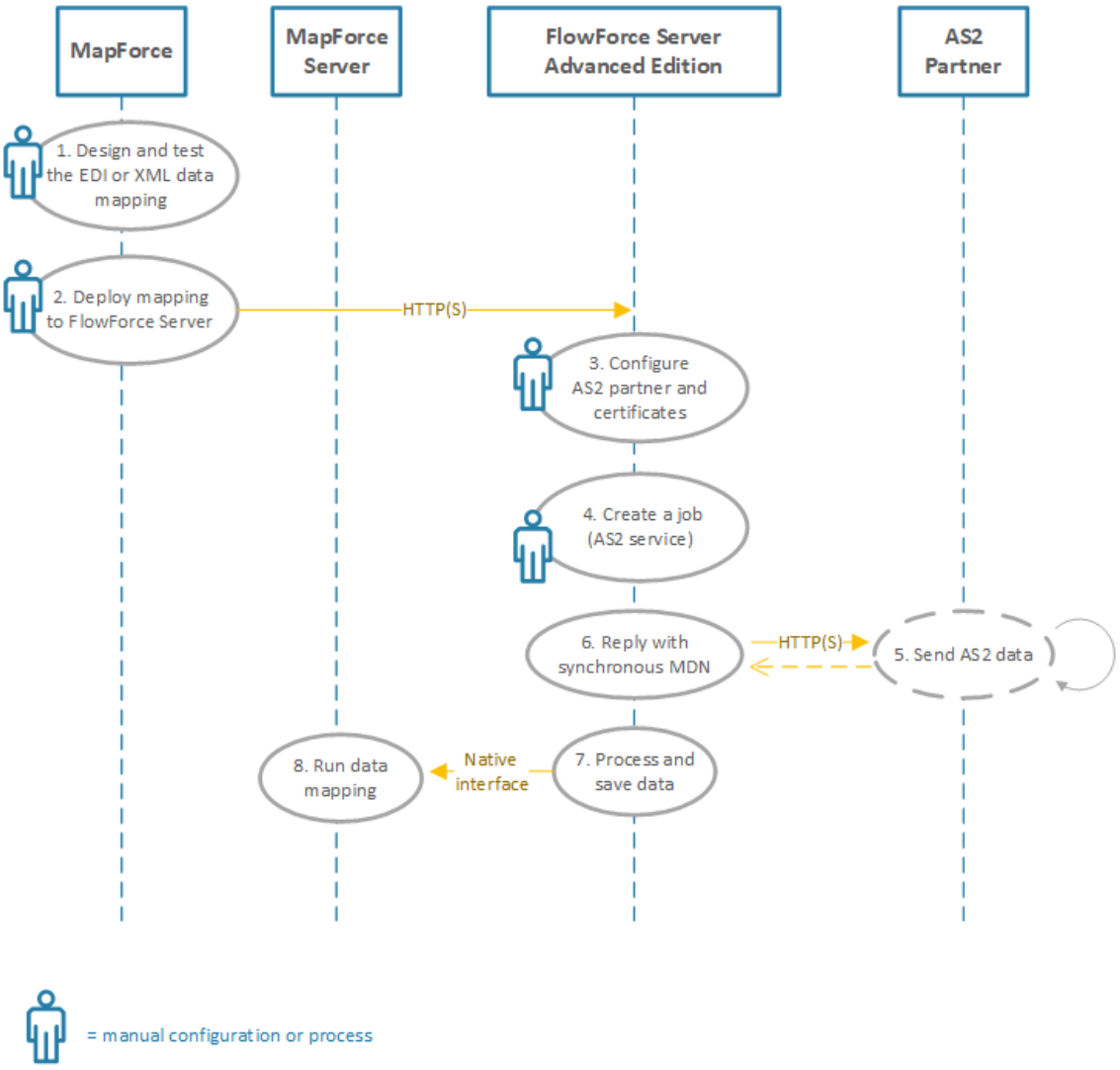
Paso	Descripción
	de datos destinadas a trabajar con datos EDI se diseñan igual que cualquier otra asignación de datos y en la documentación de MapForce encontrará numerosos ejemplos al respecto (*véase la sección EDI*). Mientras diseña la asignación podrá validar y consultar una vista previa del resultado en MapForce directamente (en el panel *Resultados*). Si quiere ejecutar la asignación en un entorno servidor, deberá diseñarla y probarla para el lenguaje de transformación integrado (*véase Seleccionar un lenguaje de transformación*).
2. Implementar la asignación en FlowForce Server	FlowForce Server automatiza varias tareas por medio de trabajos programados o bajo demanda que se pueden definir desde una interfaz web. FlowForce Server también puede automatizar la ejecución de asignaciones de datos diseñadas con MapForce, siempre y cuando el producto MapForce Server se ejecute bajo la dirección de FlowForce Server. Una vez conseguido el resultado correspondiente con la asignación de MapForce, podrá automatizar su ejecución implementándola en FlowForce Server (*véase Implementar asignaciones en FlowForce Server*).
3. Configurar socio AS2 y certificados	Para establecer la comunicación con socios AS2 deberá conocer sus datos de conectividad AS2 (como el URI y el nombre AS2) y deberá obtener certificados de intercambio. Los certificados y los datos del socio deben importarse e introducirse en FlowForce Server.
4. Crear un trabajo	Es necesario crear un trabajo de FlowForce para (i) ejecutar la asignación y producir el resultado necesario y (ii) enviar el mensaje AS2 (ver paso nº7). Estas dos acciones pueden ser pasos de ejecución de un mismo trabajo o dos trabajos distintos.
5. Ejecutar el trabajo	La ejecución del trabajo de FlowForce Server creado en el paso anterior puede configurarse de varias maneras. Por ejemplo, puede ejecutarse como llamada a servicio web o cada vez que un archivo se modifique en el sistema de archivos. O se puede programar para que se ejecute a una hora y fecha determinadas. Este paso está completamente automatizado.
6. Ejecutar la asignación de datos	Este paso también es automático y lo ejecuta MapForce Server. Si el trabajo está configurado para ejecutar una asignación de datos (de forma programada o bajo demanda), se produce una llamada interna a MapForce Server. Como resultado MapForce Server ejecuta la asignación y devuelve el resultado a FlowForce Server.
7. Escoger resultado y enviar mensaje AS2	Para poder enviar el mensaje AS2 el trabajo (o el paso de ejecución dentro del trabajo) debe llamar a la función integrada de FlowForce Server **`/system/as2/send`**. Esta

© 2018 Altova GmbH

Paso	Descripción
	función toma un número de parámetros necesarios para enviar el mensaje AS2, incluido el objeto socio AS2 configurado en el paso nº3, el URI del socio y el contenido del mensaje AS2 que desea enviar. También puede ser necesario que el trabajo llame a otras funciones de expresión AS2 de FlowForce Server para poder convertir el resultado de la asignación al formato necesario (p. ej. de archivo a secuenciad de datos).
8. El socio responde con notificación MDN sincrónica	Cuando cree el objeto socio AS2 en el paso nº3, tiene la opción de solicitar que el socio envíe una notificación MDN (notificación de disposición de mensaje) como respuesta al mensaje AS2 que envía FlowForce Server. El socio debe enviar la notificación MDN en la misma sesión que la llamada HTTP saliente de FlowForce Server (es decir, debe configurarse como notificación sincrónica).

Recibir y procesar datos AS2

Si su organización recibe datos AS2 de sus socios comerciales, puede configurar un flujo de trabajo adicional para la recepción de datos. Con él su organización no solo es capaz de recibir y almacenar datos AS2, sino también de transformarlos en otros formatos, guardarlos en una base de datos o enviarlos a otros servicios web. Por ejemplo, podría recibir archivos en formato EDI o XML de socios comerciales AS2 y usarlos comodatos de entrada para una asignación que se ejecute de forma recurrente como trabajo de FlowForce. En este caso, un proceso AS2 de ejemplo sería el siguiente:



Recibir y procesar datos AS2

El proceso AS2 de ejemplo anterior funciona como sigue:

Paso nº	Descripción
1, 2, 3	Se trata de los mismos pasos que en la tabla anterior, con la diferencia de que esta vez la asignación debe tomar como entrada un archivo que su organización espera recibir de un socio comercial AS2 (por ejemplo, un archivo EDI o XML).
4. Crear un trabajo (servicio AS2)	En este paso debe crear un trabajo de FlowForce que exponga el servicio AS2. El servicio AS2 atiende solicitudes de sus socios comerciales AS2 desde una dirección HTTP(S) y un puerto previamente configurados.

© 2018 Altova GmbH

Paso nº	Descripción
5. Enviar datos AS2	En este paso un socio comercial AS2 envía un mensaje AS2 al servicio AS2. Para que la comunicación tenga éxito, el nombre del socio AS2 y los certificados se deben haber definido previamente en FlowForce Server.
6. Contestar con MDN sincrónica	FlowForce Server contesta al socio comercial AS2 con una MDN sincrónica que indica el resultado de la operación (éxito o error).
7. Procesar y almacenar datos	Al recibir un mensaje entrante, un trabajo de FlowForce convierte los datos recibidos en una cadena o un archivo que luego guarda en un directorio o pasa a otro trabajo como argumento. La lógica exacta de procesamiento se puede configurar con ayuda de las funciones de expresión integradas de FlowForce Server.
8. Ejecutar asignación de datos	El trabajo de FlowForce Server que recibe los datos AS2 tiene la opción de invocar el trabajo de asignación de datos que se creó en el paso 1. Este trabajo toma como datos entrada los datos AS2 recibidos del socio y luego los procesa de forma compatible con MapForce: por ejemplo, transformándolos a otro formato, guardándolos en una BD, enviándolos a otro servicio web, etc.

Para obtener información específica sobre la configuración AS2 en FlowForce Server consulte el manual de FlowForce Server Advanced Edition (https://www.altova.com/es/documentation).

11.7 Interfaz de la línea de comandos de MapForce

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

The general syntax of a MapForce command at the command line is:

```
MapForce.exe <filename> [/{target} [[<outputdir>] [/options]]]
```

Legend

The following notation is used to indicate command line syntax:

Notation	Description
Text without brackets or braces	Items you must type as shown
<Text inside angle brackets>	Placeholder for which you must supply a value
[Text inside square brackets]	Optional items
{Text inside braces}	Set of required items; choose one
Vertical bar (|)	Separator for mutually exclusive items; choose one
Ellipsis (...)	Items that can be repeated

<filename>

The mapping design (.mfd) or mapping project (.mfp) file from which code is to be generated. To generate code for the whole project, set the target `/GENERATE` (see description below) and enter the project path as `<filename>`, for example, **MapForceExamples.mfp**.

/{target}

Specifies the target language or environment for which code is to be generated. The following code generation targets are supported.

Target	Description
`/COMPILE[:compileoptions]`	Compiles a mapping to a MapForce Server execution file (.mfx). Optionally, the following options can be supplied, delimited by a comma:

© 2018 Altova GmbH

Target	Description
	`JDBC` Transforms all database connections to JDBC using the JDBC driver and the database URL defined in the Database Component Settings dialog box, see also Database Component Settings. `NOXMLSIGNATURES` Suppresses the generation of digital signatures in the MapForce Server Execution file (note that digital signatures are not supported by MapForce Server).
`/GENERATE`	Generates project code for all mappings in the project file using the current folder settings, see Managing Project Folders. If you select this target, make sure to supply a MapForce project (.mfp file) as `<filename>`.
`/XSLT`	Generates XSLT 1.0 code.
`/XSLT2`	Generates XSLT 2.0 code.
`/XQuery`	Generates XQuery code.
`/JAVA`	Generates Java code.
`/CS`	Generates C# code. This command also optionally allows setting specific options for code generation, namely: `/CS[:{VS2008|VS2010|VS2013|VS2015|VS2017}]` `VS2008` Visual Studio 2008 `VS2010` Visual Studio 2010 `VS2013` Visual Studio 2013 `VS2015` Visual Studio 2015 `VS2017` Visual Studio 2017 If no Visual Studio version is specified, code will be generated using the Visual Studio version defined in the code generation options, see Code Generator Options.
`/CPP`	Generates C++ code. This command also optionally allows setting specific code generation options, namely: `/CPP[:{VS2008|VS2010|VS2013|VS2015|VS2017},{MSXML|XERCES3},{LIB|DLL},{MFC|NoMFC}]`

Target	Description
	The first option group set the target Visual Studio version. Valid values: `VS2008` Visual Studio 2008 `VS2010` Visual Studio 2010 `VS2013` Visual Studio 2013 `VS2015` Visual Studio 2015 `VS2017` Visual Studio 2017 The second option group specifies the XML library targeted by the generated code. Valid values: `MSXML` Generate code for MSXML 6.0 `XERCES3` Generate code for Xerces 3 The third option group specifies whether static as opposed to dynamic libraries should be generated. Valid values: `LIB` Generate static LIB libraries `DLL` Generate DLL libraries The fourth option group specifies whether code should be generated with or without MFC support. Valid values: `MFC` Enable MFC support `NoMFC` Disable MFC support If the options above are not specified, code will be generated using the Visual Studio version defined in the code generation options, see Code Generator Options.

<outputdir>

Optional parameter which specifies the output directory. If an output path is not supplied, the current working directory will be used. Note that any relative file paths are relative to the current working directory.

When target is `/GENERATE` and the `<outputdir>` parameter is not set, the code generation language, as well as the output path of each mapping, are supplied by the settings defined for each folder inside the project, see Managing Project Folders.

© 2018 Altova GmbH

When target is `/GENERATE` and the `<outputdir>` parameter is set, the `<outputdir>` value supplied at the command line takes precedence over the output directory defined at the root project level. It does not take precedence, however, over the code generation settings defined at each folder inside the project.

/options

The `/options` are not mutually exclusive. One or more of the following options can be set.

Option	Description
`[/MFXVERSION[:<version>]`	This option is applicable if the target is `/COMPILE`. It compiles the MapForce Server Execution (.mfx) file for a particular version of MapForce Server. You can supply as value any version of MapForce Server, starting with **2013r2** onwards, up to the current MapForce version. See also Compiling mappings for a specific MapForce Server version.
`/GLOBALRESOURCEFILE <filename>`	This option is applicable if the mapping uses Global Resources to resolve input or output file or folder paths, or databases. For more information, see Altova Global Resources. The option `/GLOBALRESOURCEFILE` specifies the path to a Global Resource .xml file. Note that, if `/GLOBALRESOURCEFILE` is set, then `/GLOBALRESOURCECONFIG` must also be set.
`/GLOBALRESOURCECONFIG <config>`	This option specifies the name of the Global Resource configuration (see also the previous option). Note that, if `/GLOBALRESOURCEFILE` is set, then `/GLOBALRESOURCECONFIG` must also be set.
`/LIBRARY <libname> (...)`	Use together with a code generation target language to specify additional function libraries. This option can be specified more than once to load multiple libraries. These libraries are temporarily (for this one run) added to the libraries from **Tools | Options | Libraries**.
`/LOG <logfilename>`	Generates a log file at the specified path. `<logfilename>` can be a full path name, for example, it can include both a directory and a file name. However, if a full path is supplied, the directory must exist for the log file to be generated. If you only specify the file name, then the file will be placed in the `<outputdir>` directory.

Remarks

- Relative paths are relative to the working directory, which is the current directory of the application calling MapForce. This applies to the path of the .mfd filename, .mfp filename, output directory, log filename, and global resource filename.
- Do not use the end backslash and closing quote at the command line (for example, "C:

\My directory\"). These two characters are interpreted by the command line parser as a literal double quotation mark. Use the double backslash \\ if spaces occur in the command line and you need the quotes ("c:\My Directory\\"), or try to avoid using spaces and therefore quotes at all.

Examples

1) To start MapForce and open the mapping `<filename>.mfd`, use:

```
MapForce.exe <filename>.mfd
```

2) To generate XSLT 2.0 code and also create a log file with the name `<logfilename>`, use:

```
MapForce.exe <filename>.mfd /XSLT2 <outputdir> /LOG <logfilename>
```

3) To generate XSLT 2.0 code taking into account the global resource configuration `<grconfigname>` from the global resource file `<grfilename>`, use:

```
Mapforce.exe <filename>.mfd /XSLT2 <outputdir> /GLOBALRESOURCEFILE
<grfilename> /GLOBALRESOURCECONFIG <grconfigname>
```

4) To generate a C# application for Visual Studio 2015 and output a log file, use:

```
MapForce.exe <filename>.mfd /CS:VS2015 <outputdir> /LOG <logfilename>
```

5) To generate a C++ application using the code generation settings defined in **Tools | Options**, and output a log file, use:

```
MapForce.exe <filename>.mfd /CPP <outputdir> /LOG <logfilename>
```

6) To generate a C++ application for Visual Studio 2015, MSXML, with static libraries, MFC support, and no log file, use:

```
MapForce.exe <filename>.mfd /CPP:VS2015,MSXML,LIB,MFC
```

7) To generate a C++ application for Visual Studio 2015, Xerces, with dynamic libraries, no MFC support, and a log file, use:

```
MapForce.exe <filename>.mfd /CPP:VS2015,XERCES,DLL,NoMFC <outputdir> /LOG
<logfilename>
```

8) To generate a Java application and also output a log file, use:

```
MapForce.exe <filename>.mfd /JAVA <outputdir> /LOG <logfilename>
```

9) To generate code for all mappings in the project, using the language and output directory

 © 2018 Altova GmbH

defined in the folder settings (of each folder inside the project), use:

```
MapForce.exe <filename>.mfp /GENERATE /LOG <logfilename>
```

10) To generate Java code for all mappings in the project file, use:

```
MapForce.exe <filename>.mfp /JAVA /LOG <logfilename>
```

Note that the code generation language defined in the folder settings are ignored, and Java is used for all mappings.

11) To supply input and output files at the command line for a previously compiled Java mapping, use:

```
java -jar <mappingfile>.jar /InputFileName <inputfilename> /OutputFileName
<outputfilename>
```

The `/InputFileName` and `/OutputFileName` parameters are the names of special input components in the MapForce mapping that allow you to use parameters in command line execution (see Supplying Parameters to the Mapping).

12) To compile a mapping to a MapForce Server execution file, for MapForce Server version 2018, and suppress XML signatures:

```
MapForce.exe <filename>.mfd /COMPILE:NOXMLSIGNATURES <outputdir> /
MFXVERSION:2018 /LOG <logfilename>
```

Altova MapForce 2018 Enterprise Edition

Personalizar MapForce

12 Personalizar MapForce

Esta sección del manual explica cómo trabajar con recursos globales de Altova y con archivos de catálogo, cómo personalizar el resultado de la asignación de datos y cómo generar y personalizar la documentación de la asignación.

© 2018 Altova GmbH

12.1 Cambiar opciones de MapForce

Las opciones generales de configuración de MapForce se pueden cambiar con el comando **Opciones** del menú **Herramientas**.

El cuadro de diálogo "Opciones" incluye varias pestañas:

Bibliotecas

- Desde aquí puede agregar o eliminar bibliotecas de funciones personales (*véase Importar funciones XSLT 1.0 o 2.0 personales*, *Importar funciones XQuery 1.0 personales* e *Importar bibliotecas Java y .NET personales*).

Generales

- Puede especificar si el logotipo aparece o se oculta cuando se inicia MapForce.
- Puede habilitar y deshabilitar el fondo degradado del panel *Asignación*.
- Puede activar la opción *Limitar la presentación de anotaciones...* que afecta a los componentes que admiten anotaciones (p. ej. esquemas XML, archivos EDI, etc.). Si el texto de la anotación tiene varias líneas y habilita esta opción, solamente se presentarán las primeras X líneas del componente. Esta opción también afecta a las instrucciones SELECT que estén visibles en los componentes.
- Puede definir la codificación de caracteres predeterminada para componentes nuevos. Esta configuración se puede modificar en cada componente por separado (*véase Cambiar la configuración del componente*).
- Puede definir el tiempo de espera de ejecución para la vista previa de resultados en el panel *Resultados*.
- Puede especificar si el resultado de la asignación se escribe en archivos temporales (opción predeterminada) o en los archivos finales directamente cuando se genera la vista previa de resultados en el panel *Resultados*.

  Advertencia: si habilita la opción "Escribir directamente en archivos de salida finales" sobrescribirá los archivos de salida sin solicitar la confirmación previa del usuario.

- Puede especificar el tamaño máximo del texto que aparece en el panel Resultado cuando se genera la vista previa de resultados de asignaciones que generan archivos XML y de texto de gran tamaño (*véase Vista previa de resultados*).

Edición

- Puede especificar si los componentes y funciones deben alinearse con los demás componentes cuando se arrastran con el ratón (*véase Alinear componentes*).
- Si habilita la opción *Eliminación inteligente de componentes*, la aplicación recordará las conexiones que pertenecían a componentes eliminados (*véase Conservar conexiones tras eliminación de componentes*).

Mensajes

- Aquí puede volver a habilitar notificaciones de mensajes deshabilitados previamente al activar la casilla *No volver a mostrar este mensaje*.

Generación de código

- Aquí puede configurar opciones relacionadas con la generación de código de programa y de archivos de ejecución de MapForce Server. Para más información consulte los apartados Opciones del generador de código y Compilar asignaciones de datos en archivos de ejecución de MapForce Server respectivamente.

XBRL

- Aquí puede definir opciones de configuración relacionadas con componentes XBRL (*véase Opciones XBRL generales*).

Depurador

- Aquí puede configurar opciones relacionadas con la depuración de asignaciones (*véase Configurar el depurador*).

Base de datos

- Aquí puede definir opciones de configuración relacionadas con la consulta de bases de datos en el panel *Consulta de BD* (*véase Examinar y consultar bases de datos*).

© 2018 Altova GmbH

12.2 Recursos globales de Altova

Los recursos globales de Altova es una característica que permite hacer referencia a archivos, carpetas y bases de datos y poder reutilizar este tipo de recursos, configurarlos y ponerlos a disposición de otras aplicaciones de Altova.

Por ejemplo, imagine que varias asignaciones de MapForce leen datos del mismo archivo XML de forma rutinaria. Si por algún motivo el nombre del archivo cambiara, se producirían errores de tipo "archivo no encontrado" en varios contextos y el flujo de trabajo quedaría dañado. Para evitar este tipo de problemas podemos crear *alias de archivo* (en otras palabras, un recurso global) y cambiar las asignaciones para que hagan referencia a este recurso global en lugar de remitir al archivo en el disco. De este modo, si el nombre de archivo llegara a cambiar, solo haría falta actualizar el alias de archivo.

Puede definir recursos globales y compartirlos en todas estas aplicaciones de escritorio de Altova: Authentic, MobileTogether Designer, MapForce, DatabaseSpy y XMLSpy. En el lado servidor, estas aplicaciones servidor de Altova pueden consumir recursos globales: MapForce Server, MapForce Server Advanced Edition, RaptorXML Server, RaptorXML+XBRL Server.

Los recursos globales (tanto si se trata de archivos, de carpetas o de bases de datos) se pueden usar en MapForce en numerosos casos:

- Para suministrar rutas de acceso configurables como entrada de la asignación (*véase* *Ejemplo: ejecutar asignación con archivos de entrada variables*).
- Para redirigir el resultado de la asignación a una ruta de acceso configurable (*véase* *Ejemplo: generar resultados en carpetas variables*).
- Para suministrar una ruta de acceso configurable a una hoja de estilos SPS de StyleVision (archivo .sps), si es que la asignación utiliza un archivo SPS. Por ejemplo, en lugar de hacer referencia a un archivo SPS de StyleVision desde la configuración de un componente de MapForce, se puede hacer referencia a un archivo SPS definido previamente como recurso global y que tiene dos configuraciones posibles (p. ej. "Website.sps" y "Print.sps"). Consulte Aplicar estilos a resultados de la asignación con StyleVision para obtener más información.
- Para reutilizar una conexión de base de datos. Si tiene una base de datos definida como recurso global (en cualquier aplicación de Altova), podrá conectarse a ella sin necesidad de seguir todos los pasos del proceso de conexión (*véase* *Usar una conexión a través de recursos globales*).
- Para cambiar con facilidad la base de datos de la que lee datos la asignación o la base de datos en la que escribe datos la asignación (siempre y cuando las bases de datos tengan estructuras idénticas pero datos distintos). Consulte Ejemplo: cambiar de base de datos para obtener más información.

Nota

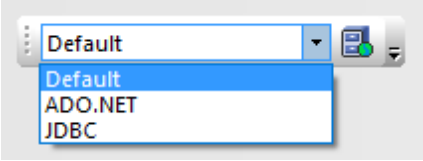
- FlowForce Server no admite el uso de recursos globales. MapForce Server puede consumir recursos globales en la línea de comandos o a nivel de API.
- MapForce Basic Edition no puede consumir conexiones de base de datos definidas como recursos globales.

12.2.1 Crear recursos globales

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

A Global Resource alias is a reusable reference which represents a file or folder path, or a database connection. Aliases are defined only once and can be reused as many times as necessary in contexts which support them, including across multiple Altova applications. Taking databases as example, if you frequently work with a specific database in more than one Altova application, then it is a good idea to add the database connection as a Global Resource. This way, you wouldn't need to go through all the Database Connection Wizard steps each time when you need to connect to the same database from another Altova application.

File, folder, and database aliases are configurable in their turn, by means of so-called "configurations". Configurations make it possible to easily switch between files, folders and databases that are consumed or produced by Altova applications, which is particularly useful for testing scenarios. For example, you could create a database alias that consists of three separate connections to the same database, each with a different driver kind: (a) ODBC, the default connection kind, (b) JDBC, and (c) ADO.NET. This way, to connect to the database with a specific driver, you would just select the corresponding configuration from the Global Resources drop-down list before running the mapping.



Global Resources drop-down list

Configurations can also help you generate mapping output to variable folders, with a click of a button. For example, you could create a folder alias with two configurations: (a) "Testing", which points to directory **C:\Testing** and (b) "Production", which points to directory **C:\Production**. It is then possible to configure a mapping to generate output to either **C:\Testing** or **C:\Production** folders, just by selecting the required configuration from the Global Resources drop-down list before running the mapping. This example is discussed in more detail in Example: Generate Output to Variable Folders.

How to create a Global Resource alias

1. On the **Tools** menu, click **Global Resources**. (Alternatively, click the **Global Resource** toolbar button.)
2. Click **Add** and select the resource type you wish to create (file, folder, database).
3. Enter a descriptive name for this alias in the **Resource alias** text box (for example, "MappingInputFile", "MappingOutputFolder", "DatabaseConnection").

 © 2018 Altova GmbH

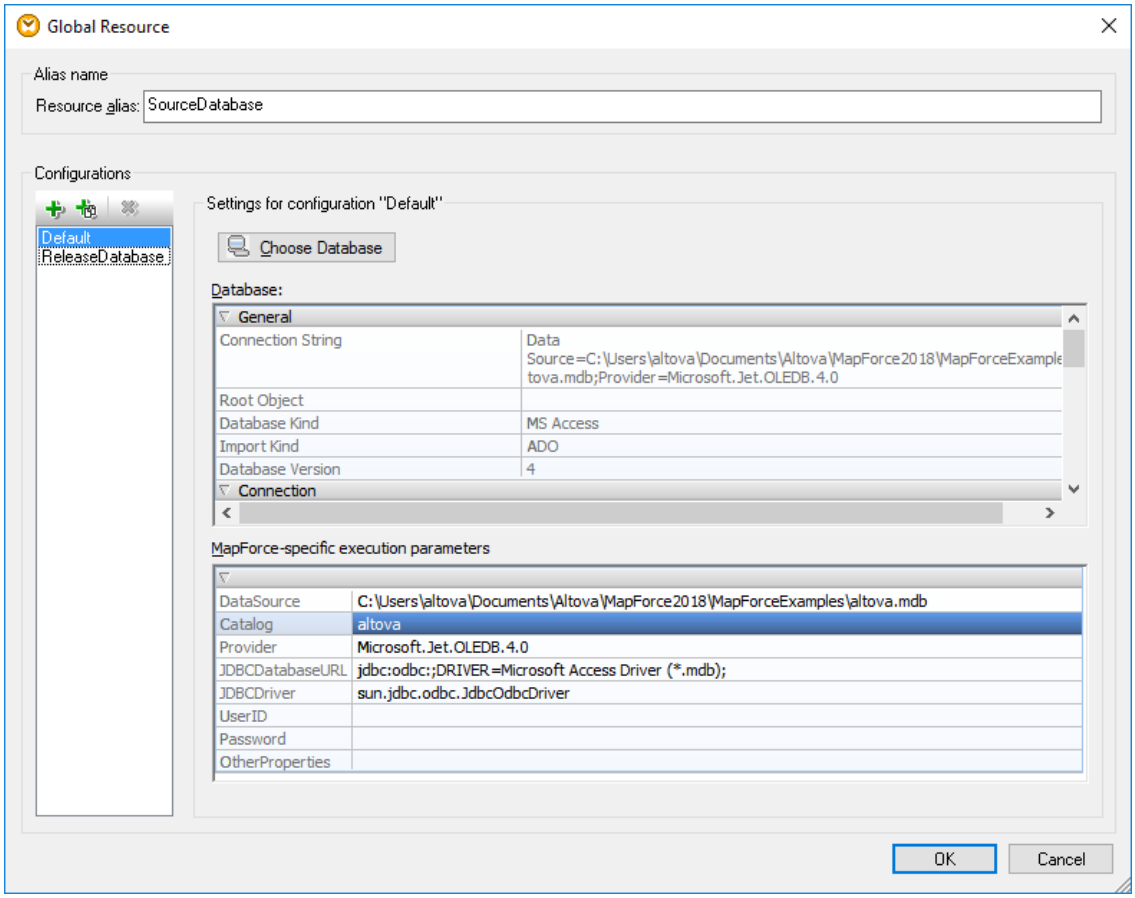
4. Set up the "Default" configuration:
   a) If it's a file or folder, browse for the file or folder to which this resource should point by default.
   b) If it's a database connection, click **Choose Database** and follow the Database Connection Wizard to connect to the database (see Connecting to a Database). This database connection will be used by default when the mapping runs (unless a different configuration is explicitly selected from the Global Resources drop-down list or supplied as a command line parameter in server execution).
5. Optionally, if the resource should have an additional configuration (for example, a driver kind in case of databases, or an alternative path in case of files or folders), click the **Add configuration** button, enter a descriptive name (for example "ProductionFolder" or "JDBC_Alternative"), and set it up as follows:
   a) If it's a file or folder, browse for the file or folder to which this resource should point as an alternative to the default configuration defined in previous step.
   b) If it's a database connection, follow the Database Connection Wizard to connect to the database. This database connection will be used as an alternative to the default one.

   In some cases, it might be more convenient to create a configuration as a copy of the default configuration, and then edit it. In this case, click the **Add configuration as a copy of the currently selected configuration** button.
6. Repeat the previous step for each additional configuration required.

12.2.2 Bases de datos como recursos globales

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

When you add a database connection as a Global Resource, the database connection parameters are automatically populated on the Global Resource dialog box.



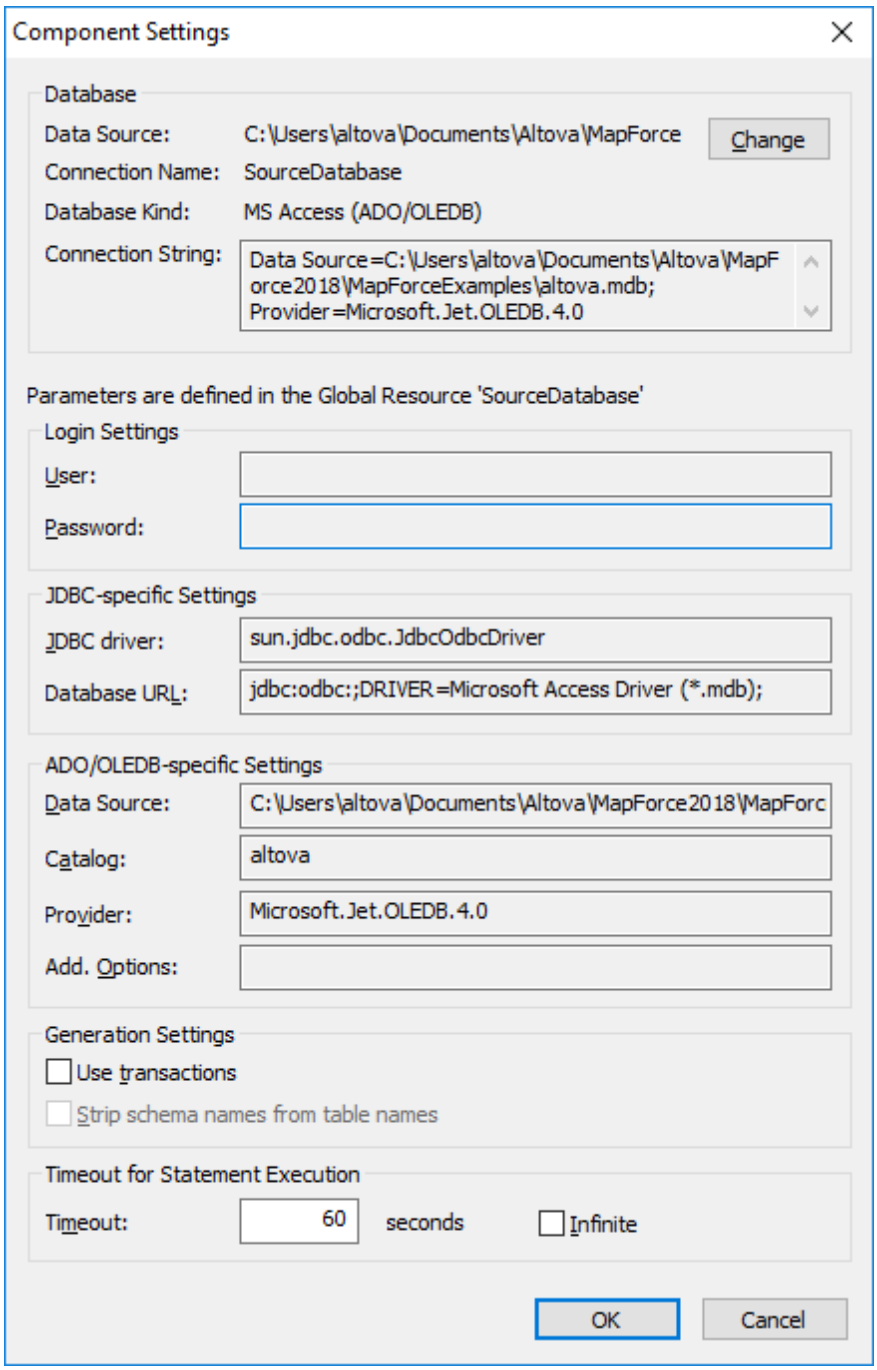
Global Resource dialog box

On the Global Resource dialog box, it is possible to edit some of the database connection parameters. As illustrated above, the parameters are grouped into two categories:

Database	These parameters are shared between Altova applications. In MapForce, they are used at design time, that is, when the mapping is loaded, or when you click the **Output** tab in MapForce to preview the mapping.
MapForce-specific execution parameters	These parameters are applicable when you generate program code or compile a mapping to MapForce Server execution file (.mfx). They are used at mapping runtime, as follows: • In generated C++, C#, or Java program code. • If you compiled the mapping to a MapForce Server execution file, and automatic JDBC conversion took place. For more information about automatic JDBC conversion, see Database mappings in various execution environments.

 © 2018 Altova GmbH

If a mapping uses a Global Resource to connect to a database, then the database connection details visible in the Global Resource dialog box take precedence over those defined on the mapping. In the Component Settings dialog box, illustrated below, notice that the database connection settings become grayed out. The dialog box also informs you that the connectivity parameters are defined as a Global Resource.



Component Settings dialog box

To change the database component to connect to the database directly (without using Global Resources), click **Change**, and follow the wizard steps to reconnect to the database.

12.2.3 Resultados de transformaciones de MapForce y StyleVision como recursos globales

También se pueden crear recursos globales que, en lugar de apuntar a un archivo estático, lean determinado archivo producido por una asignación de MapForce o por una transformación de StyleVision. En este caso, la aplicación de Altova que consuma el recurso global primero llamará a MapForce o a StyleVision, después ejecutará la asignación o transformación correspondiente y, por último, buscará el archivo resultante. Esto permite definir flujos de trabajo en los que participan varias aplicaciones de Altova (p. ej. pasar el resultado de una asignación de MapForce o de una transformación de StyleVision como entrada de otra asignación o transformación).

Para ver un ejemplo en el que XMLSpy consume el resultado (la salida) de una asignación de MapForce con ayuda de recursos globales consulte el apartado Ejemplo: crear un flujo de trabajo de aplicación.

Nota

- Para poder usar el resultado de una asignación como recurso global, el lenguaje de transformación de la asignación debe ser BUILT-IN o la asignación solo puede contener componentes que sean compatibles con el lenguaje BUILT-IN (p. ej. algunas funciones XSLT no son compatibles con el lenguaje BUILT-IN).
- MapForce Basic Edition no permite usar resultados de transformaciones ni asignaciones como recursos globales.

12.2.4 Archivo XML de recursos globales

Todos los recursos globales de Altova, independientemente de la aplicación en la que se crearan, se almacenan por defecto en esta ruta de acceso: **C:\Usuarios\Documentos\Altova \GlobalResources.xml**. Esto garantiza la transparencia del proceso, permite crear copias de seguridad con facilidad y facilita su transporte a otros equipos donde estén instalados los productos de Altova. También puede cambiar el nombre del archivo **GlobalResources.xml** o copiarlo para crear varios archivos de recursos globales. No obstante, cada aplicación de Altova puede tener como máximo un archivo de recursos globales activo.

Para configurar el archivo de recursos globales activo:

1. En el menú **Herramientas** haga clic en **Recursos globales** (o en el botón **Recurso global** de la barra de herramientas.)
2. Haga clic en **Examinar** y seleccione el archivo XML de recursos globales necesario.

© 2018 Altova GmbH

Si usa varios archivos de recursos globales, asegúrese de que el archivo que está activo contiene todos los recursos globales que necesita para ejecutar la asignación. Por ejemplo, si la asignación se configuró para leer datos de una ruta de acceso con ayuda de un recurso global, el archivo activo de recursos globales debe contener dicho recurso global. De lo contrario, recibirá mensajes de error de tipo "Error al resolver el recurso global" en la ventana Mensajes.

12.2.5 Recursos globales en distintos entornos de ejecución

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

A mapping using Global Resources will behave differently in each environment where it is run, as shown below.

Global Resources in XSLT, XSLT2, XQuery

When you generate XSLT or XSLT2 code and the mapping uses Global Resources, this does not affect the generated XSLT stylesheet in any way. With or without Global Resources, the input and output files are not a permanent assignment and can be specified flexibly anyway when you run the XSLT stylesheet in your XSLT processor. The same applies for generated XQuery code.

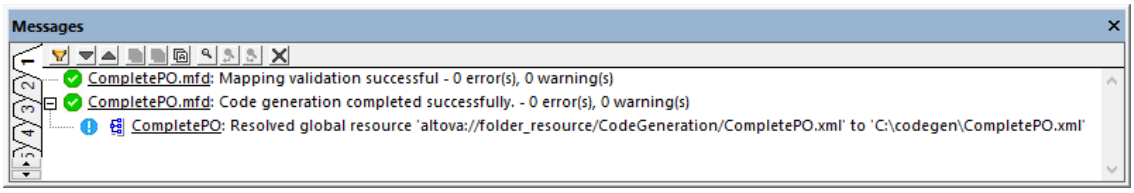
An exception to this is the **DoTransform.bat** file generated for RaptorXML execution. Any Global Resources used by the mapping will be resolved to actual paths in **DoTransform.bat**, taking into account the value (configuration) which is currently selected from the Global Resource drop-down list.



For information about supplying Global Resources to RaptorXML, refer to the RaptorXML documentation (see https://www.altova.com/documentation.html).

Global Resources in C#, C++, Java

When you generate C#, C++, or Java program code, any Global Resources used by the mapping will be resolved. For example, a file or folder alias defined as Global Resource will be converted to the actual file or folder path. If a particular Global Resource configuration is selected from the Global Resources drop-down list, then the code will be generated for the selected configuration. The **Messages** window provides information as to how exactly a Global Resource was resolved, for example:



To generate code for a particular Global Resource configuration, select it from the Global Resource drop-down list before generating code. Alternatively, if you generate code from the command line, supply the **GLOBALRESOURCEFILE** and **GLOBALRESOURCECONFIG** parameters at the command line (see also MapForce Command Line Interface).

It is not possible to switch or refer to Global Resources from generated code (instead, you can modify the code to change the input or output file path).

Note: In C# or Java, you can change not only the path but also the data type of input or output, see Changing the data type of the mapping input/output (C#, Java).

Global Resources in MapForce Server

When you compile a mapping to a MapForce Server execution file (.mfx), any Global Resources references used by the mapping are preserved as such. In MapForce Server, the following is required to run an .mfx file compiled from a mapping which uses Global Resources:

1. The path to the Global Resource XML file (that is, the file where Global Resources are defined, see The Global Resources XML File).
2. The Global Resource configuration name. The name of the default configuration is "Default". If you created additional configurations, as explained in Creating Global Resources, then the desired configuration must be called by its corresponding name.

The Global Resource file path and the name of the configuration can be specified as follows:

- If you run the mapping through the command line interface, set the options `--globalresourceconfig` and `--globalresourcefile` after the `run` command, for example:

```
C:\Program Files (x86)\Altova\MapForceServer2018\bin\MapForceServer.exe
run SomeMapping.mfx --globalresourcefile="C:\Usuarios\me\Documentos
\Altova\GlobalResources.xml" --globalresourceconfig="Default"
```

- If you run the mapping through the MapForce Server API, call the method **`SetOptions`** two times before calling the **`Run`** method. The first call is required to supply the Global Resource XML file path as option, and the second one is required to supply the Global Resource configuration name.

For more information, refer to the MapForce Server documentation (see https://www.altova.com/documentation.html).

© 2018 Altova GmbH

12.2.6 Ejemplo: ejecutar asignación con archivos de entrada variables

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

Let's assume that, as part of your job duties, you frequently run a mapping that takes as input an XML file. Under normal circumstances, whenever you want to change the input XML of the mapping, you can open the properties of the source XML component and browse for the new input file, see Changing the Component Settings. This is easy to accomplish if it's a one time task. However, what if you need to change the input XML file of the mapping multiple times per day, or even per hour? For example, every morning you need to run the mapping and generate a report by using one XML file as mapping input, and every evening the same report must be generated from another XML file. This is where Global Resources can help you: instead of editing the mapping multiple times per day (or keeping multiple copies of it), you could configure the mapping to read from a file defined as a global resource (a so-called "file alias"). To address the requirement laid out in this example, the file alias could be configured to have two configurations:

1. "Default" - This configuration would supply a "morning" XML file as mapping input
2. "EveningReports" - This configuration would supply an "evening" XML file as mapping input.

Having these configurations in place would make it possible to run the mapping with either input file. Once the file alias is set up as shown below, you will be able to select the desired configuration from a drop-down list, before running the mapping.

Step 1: Create the Global Resource

The file alias can be created as follows:

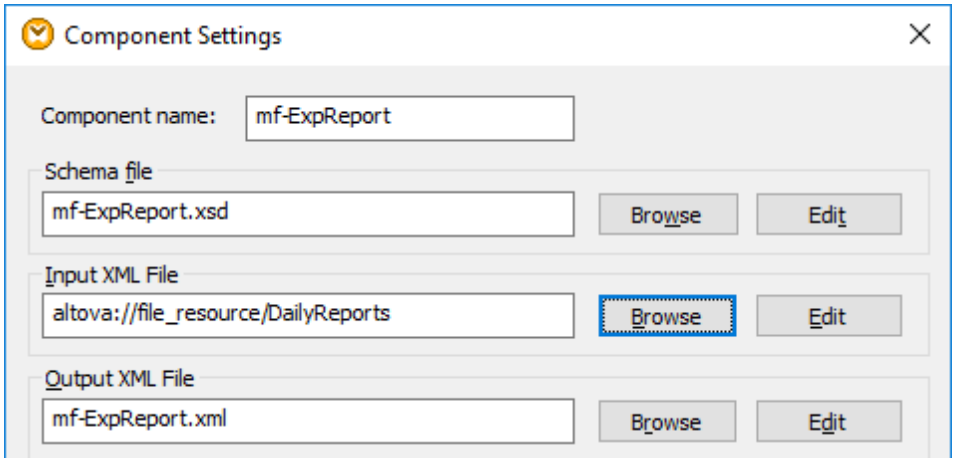
1. On the **Tools** menu, click **Global Resources**. (Alternatively, click the **Global Resource** toolbar button.)
2. Click **Add | File**.
3. Enter a name in the **Resource alias** text box (in this example, "DailyReports" would be an appropriate name).
4. Click **Browse** and select the following file: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\mf-ExpReport.xml**.
5. Click **Add Configuration** and name it "EveningReports".
6. Click **Browse** and this time select the following file: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\mf-ExpReport2.xml**.

Step 2: Use the Global Resource in the mapping

The required Global Resource has now been created; however, the mapping is not using it yet. To change the mapping so that it reads from the previously defined file alias (Global Resource), do

the following:

1. Open the following mapping **<Documentos>\Altova\MapForce2018 \MapForceExamples\Tutorial\Tut-ExpReport.mfd**.
2. Right-click the header of the source component on the mapping, and select **Properties** from the context menu.
3. Next to **Input XML file**, click **Browse**.
4. Click **Switch to Global Resources** and select the file alias "DailyReports" defined previously.
5. Click **Open**. The input XML file path has now become **altova://file_resource/ DailyReports**, which indicates that the path uses a Global Resource.



Step 3: Run the mapping with the desired configuration

You can now easily switch the input XML file before running the mapping, as follows:

- On the **Tools** menu, click **Active Configuration | Default**, to use the file **mf-ExpReport.xml** as input.
- On the **Tools** menu, click **Active Configuration | EveningReports**, to use the file **mf-ExpReport2.xml** as input.

Alternatively, select the required configuration from the **Global Resources** drop-down list.



To preview the mapping result with either configuration, click the **Output** tab and observe differences in the generated output.

12.2.7 Ejemplo: generar resultados en carpetas variables

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

© 2018 Altova GmbH

This example illustrates how mapping output can be redirected to different folders by means of Global Resources.

Let's suppose that sometimes you need to generate the mapping output to one directory (for example, **C:\Testing**), while in certain cases output must be generated to another directory (for example, **C:\Production**). With Global Resources, this is possible by creating a folder alias with two configurations:

1. "Default" configuration - Generates output to **C:\Testing**
2. "Production" configuration - Generates output to **C:\Production**.

The steps below illustrate how to achieve this goal.

Step 1: Create the Global Resource

The folder alias can be created as follows:

1. On the **Tools** menu, click **Global Resources**. (Alternatively, click the **Global Resource** toolbar button.)
2. Click **Add | Folder**.
3. Enter a name in the **Resource alias** text box (in this example, "OutputDirectory" could be an appropriate name).
4. Click **Browse** and select the following folder: **C:\Testing**. (Make sure that this folder already exists on your operating system.)
5. Click **Add Configuration** and enter a name for the new configuration (in this example, "ProductionDirectory").
6. Click **Browse** and this time select the following folder: **C:\Production**. (Make sure that this folder already exists on your operating system.)

Step 2: Use the Global Resource in the mapping

The required Global Resource has now been created; however, the mapping is not using it yet. To change the mapping so that it uses from the previously defined folder alias (Global Resource), do the following:

1. Open the following mapping **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Tut-ExpReport.mfd**.
2. Right-click the target component on the mapping, and select **Properties** from the context menu.
3. Next to **Output XML file**, click **Browse**.
4. Click **Switch to Global Resources**, and then click **Save**.
5. When prompted to save the output XML file, enter **output.xml** (or another descriptive file name that you wish to give to the output file). The output XML file path has now become **altova://folder_resource/OutputDirectory/output.xml**, which indicates that the path is defined as a Global Resource.

Step 3: Run the mapping with the desired configuration

You can now easily switch to the desired mapping output folder file before running the mapping, as follows:

- On the **Tools** menu, click **Active Configuration | Default**, and then click the **Output** tab to preview the mapping result. The mapping output (either a temporary or a permanent file, as explained below) will be generated in the **C:\Testing** directory.
- On the **Tools** menu, click **Active Configuration | ProductionDirectory**, and then click the **Output** tab. The mapping output (either a temporary or a permanent file, as explained below) will be generated in the **C:\Production** directory.

Note: The mapping output is written by default as a temporary file, unless you explicitly configured MapForce to write output to permanent files.

To configure MapForce to generate permanent files instead of temporary, do the following:

1. On the **Tools** menu, click **Options**.
2. In the **General** section, select the option **Write directly to final output files**.

12.2.8 Ejemplo: cambiar de base de datos

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

When a mapping reads or writes data from a database, it is possible to switch the database connection immediately before mapping runtime (for example, from a release to a production database, and vice versa). This example illustrates how to accomplish this by means of Global Resources. Switching databases this way implies that both databases have the same structure but different data. For the purpose of this example, we will be working with the following Microsoft Access databases:

- **altova.mdb**, from the directory: **<Documentos>\Altova\MapForce2018 \MapForceExamples**. This database plays the role of the default development database.
- **altova.mdb**, from the directory: **<Documentos>\Altova\MapForce2018 \MapForceExamples\Tutorial**. This database plays the role of the production database.

The requirement is to easily supply to the mapping either of the two databases immediately before runtime, without editing the mapping. To achieve this requirement, we will create a database Global Resource (database alias) with two configurations:

1. *Default*. This configuration will point to the default development database.
2. *Release*. This configuration will point to the release database.

© 2018 Altova GmbH

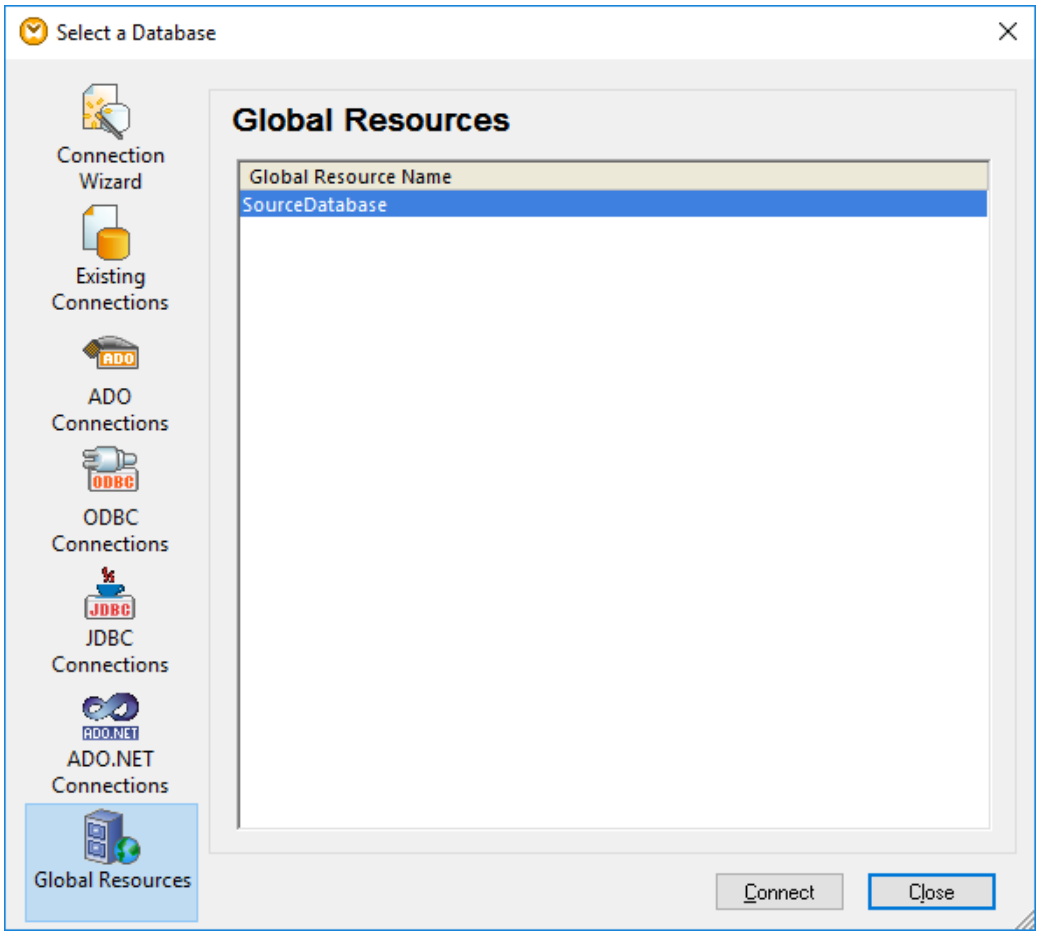
Step 1: Create the Global Resource (database alias)

1. On the **Tools** menu, click **Global Resources**. (Alternatively, click the **Global Resource** toolbar button.)
2. Click **Add | Database**.
3. Enter a descriptive name in the **Resource alias** text box (for example, "SourceDatabase").
4. Click **Choose Database**, select **Microsoft Access (ADO)**, and browse for the development database (**<Documentos>\Altova\MapForce2018\MapForceExamples\altova.mdb**).
5. Click **Add Configuration** and name it "ReleaseDatabase".
6. Click **Choose Database**, select **Microsoft Access (ADO)**, and this time browse for the production database (**<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\altova.mdb**).

Step 2: Use the Global Resource in the mapping

Now that the database alias has been created, the mapping must be modified to use it.

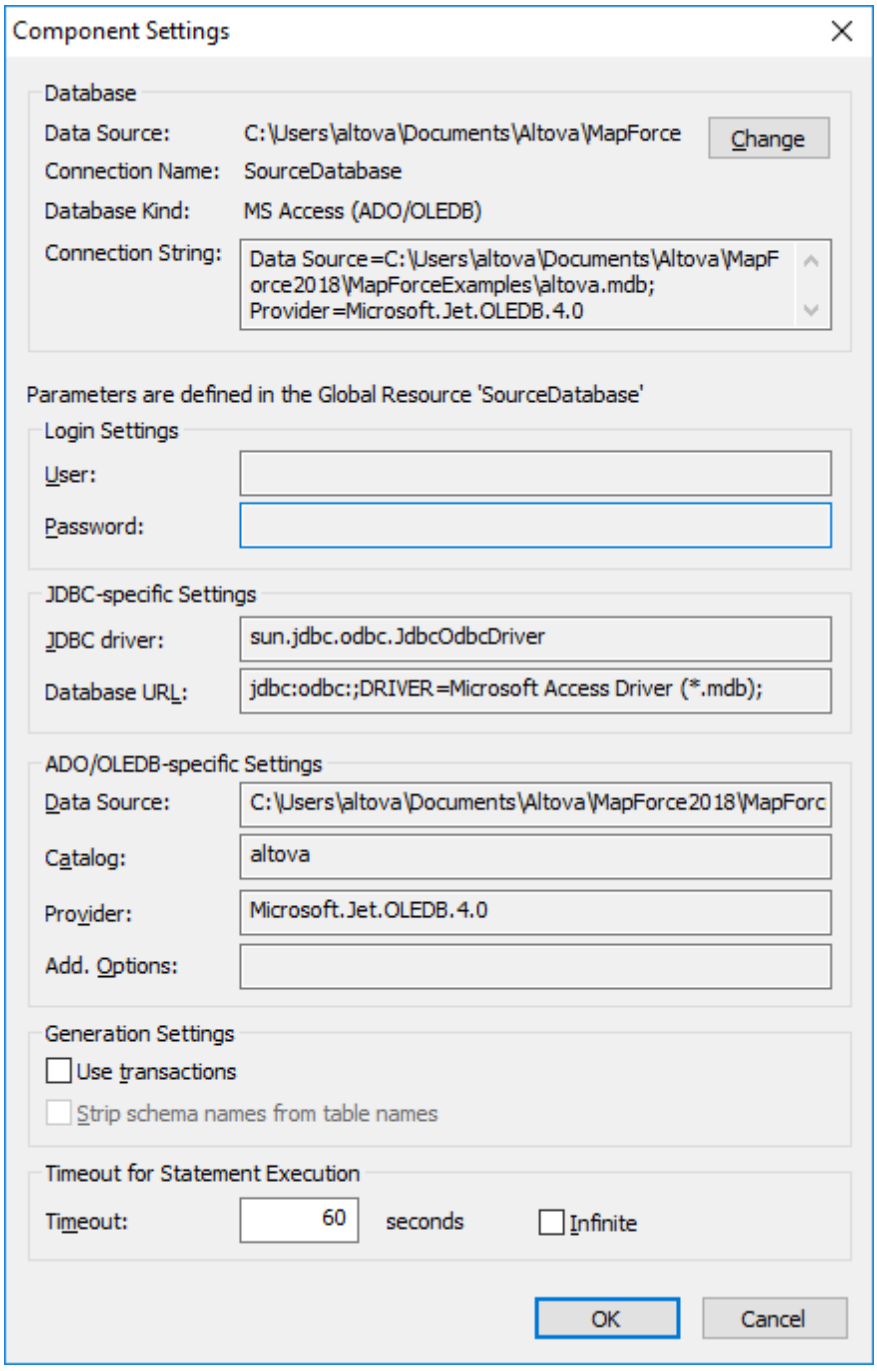
1. Open the following mapping: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\PersonDB.mfd**.
2. Right-click the database component, and select **Properties** from the context menu.
3. Click **Change**, and select the "SourceDatabase" alias created previously.



4. Click **Connect**.
5. When prompted to select the database objects, leave the default selection as is, and click OK.

Note: When a database connection is defined as a Global Resource, the settings on the Component Settings dialog are grayed out, as illustrated below. As mentioned by the text on the dialog box, the connectivity parameters can be changed from the Global Resources dialog box (which can be opened by clicking the **Global Resource** toolbar button). See also Databases as Global Resources.

© 2018 Altova GmbH



Step 3: Run the mapping with the desired configuration

You can now easily switch to the desired database before running the mapping, as follows:

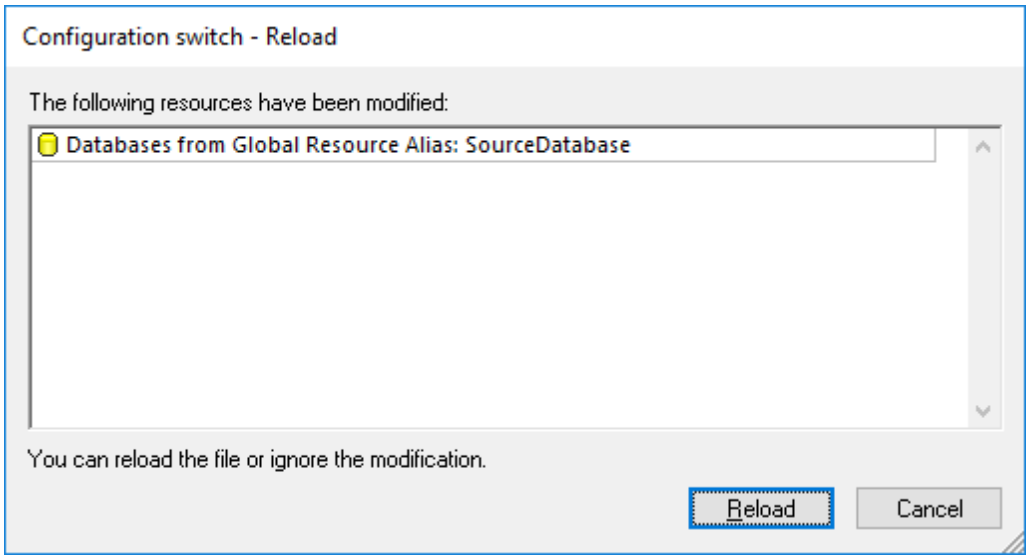
- On the **Tools** menu, click **Active Configuration | Default**, to read data from the development database.
- On the **Tools** menu, click **Active Configuration | ReleaseDatabase**, to read data from

the production database.

Alternatively, select the required configuration from the **Global Resources** drop-down list.



When you switch configurations, a dialog box prompts you that the source database is about to be reloaded:



Note: Both databases used in this example contain similar data, so there are no differences in the generated output after you run the mapping.

12.2.9 Ejemplo: crear un flujo de trabajo de aplicación

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

This example illustrates how to create a simple workflow between Altova MapForce and Altova XMLSpy, using Global Resources. Specifically, it shows you how to trigger the execution of a MapForce mapping directly from XMLSpy, and open in XMLSpy the mapping output generated by MapForce. To make this possible, we will create a Global Resource of type "Result of MapForce Transformation", as illustrated below.

Step 1: Create the Global Resource

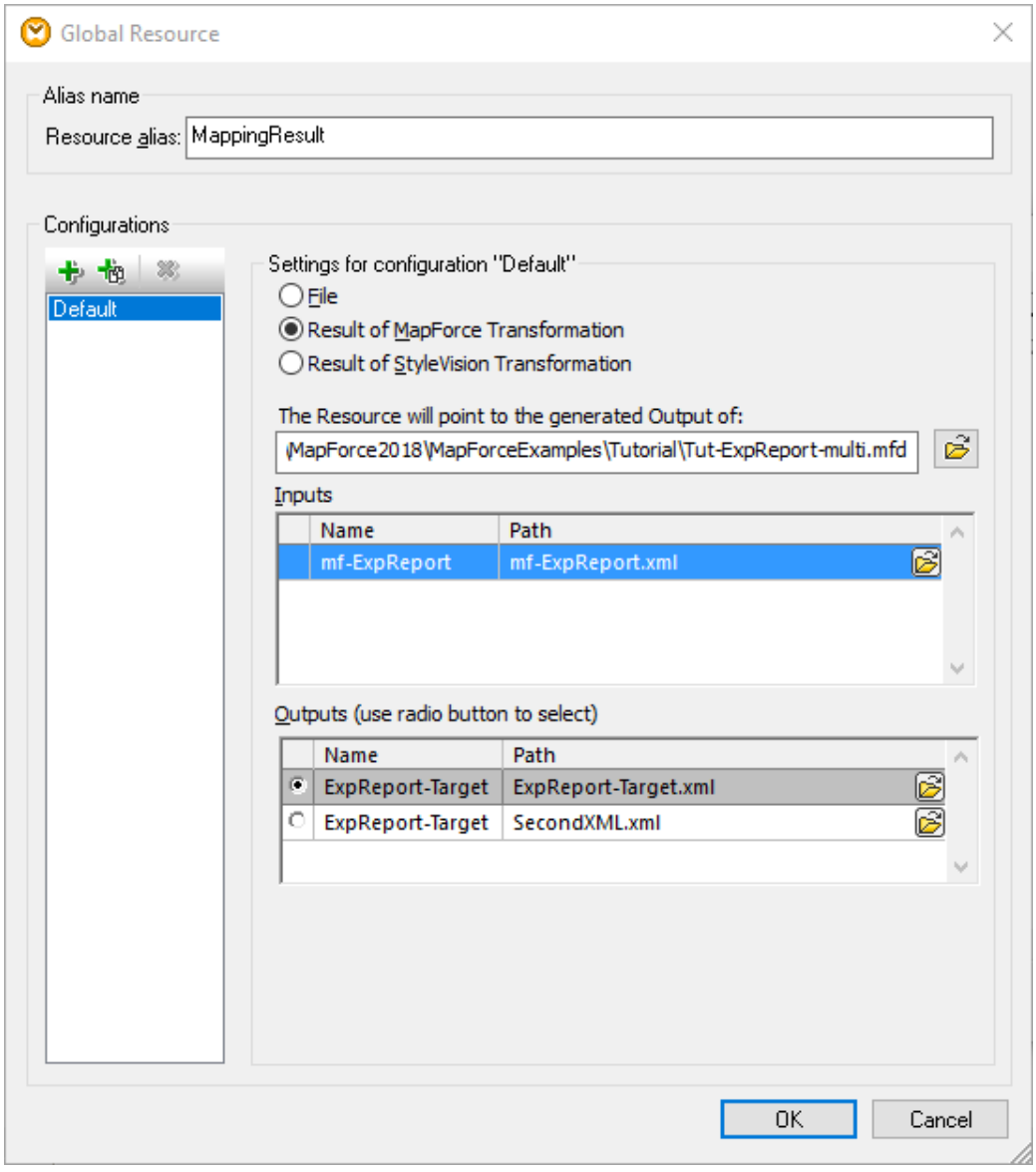
This step can be performed from both MapForce and XMLSpy.

1. On the **Tools** menu, click **Global Resources**. (Alternatively, click the **Global Resource**

 © 2018 Altova GmbH

toolbar button.)

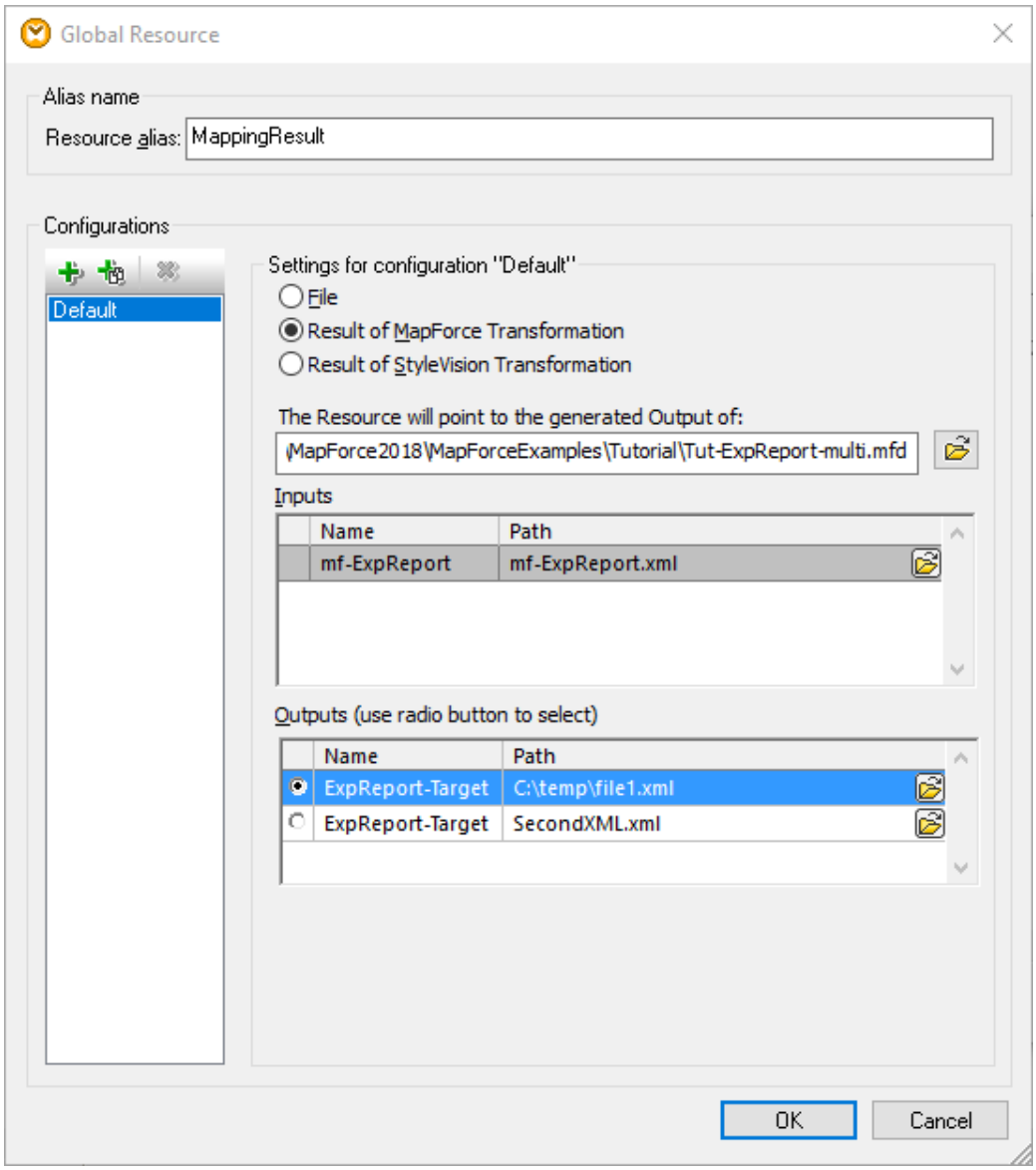
2. Click **Add | File**.
3. Enter a descriptive name in the **Resource alias** text box (in this example, "MappingResult").
4. Select the option **Result of MapForce Transformation**.
5. Click **Browse** and select the mapping **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\Tut-ExpReport-multi.mfd**. Be patient while the "Inputs" and "Outputs" sections on the dialog box are populated. As shown below, this mapping has one input and two outputs.



For the scope of this example, we would like to generate each of the two output files to the folder **C:\temp**, and change the default file name. To achieve this, we will create a configuration for each desired output, as follows:

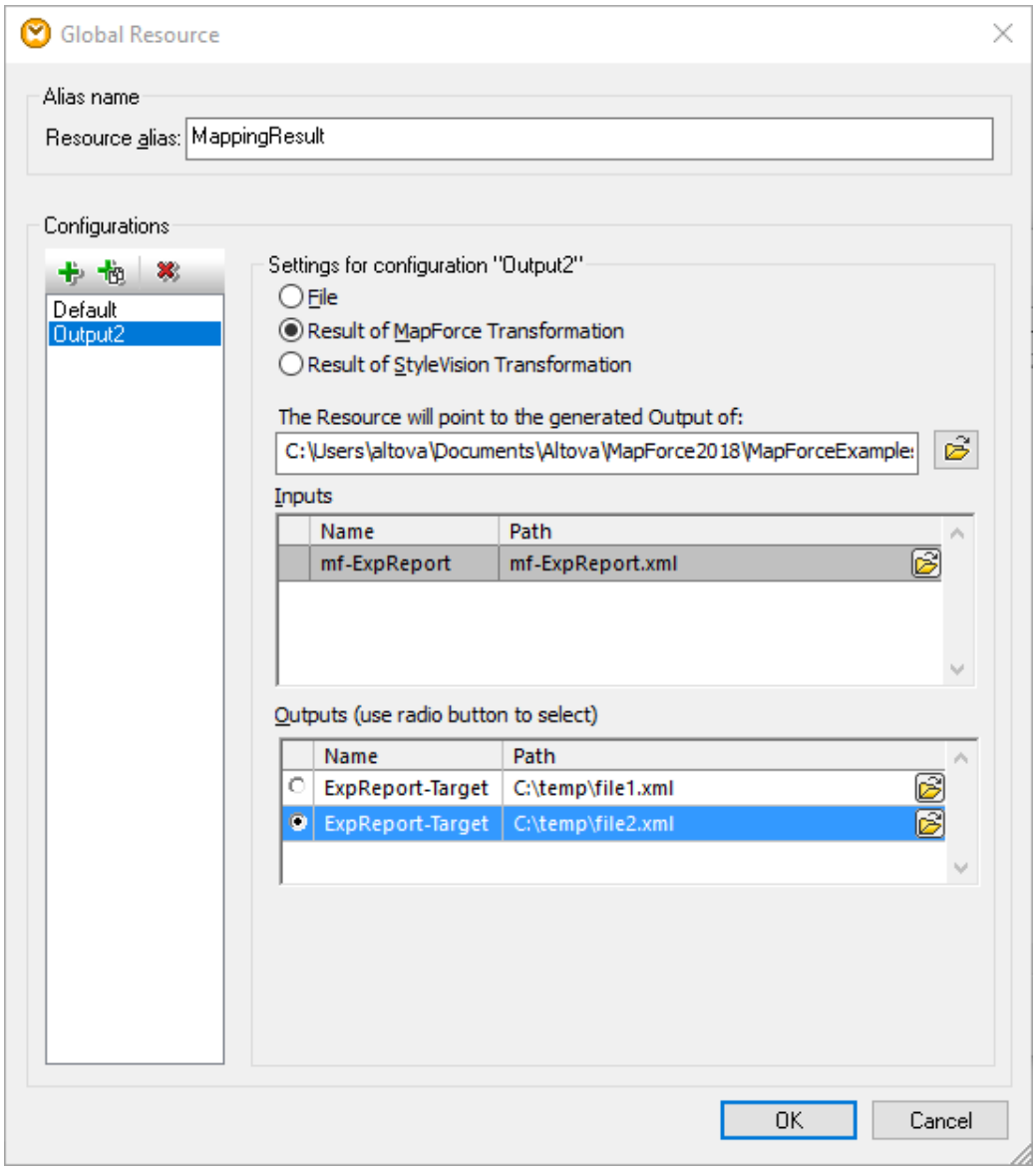
1. Under "Outputs", click **Browse** next to the first output and enter **C:\temp\file1.xml** as destination file name. This is the default configuration which will produce the first output

file when triggered.



2. Click **Add Configuration as a copy...** and enter a name for the new configuration (in this example, "Output2").
3. Under "Outputs", click **Browse** next to the second output and enter **C:\temp\file2.xml** as destination file name. This is the alternative configuration which produces the second output file.

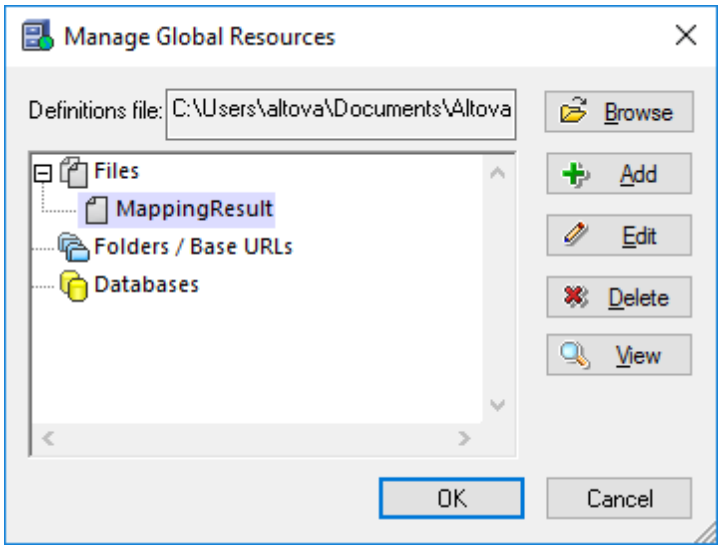
© 2018 Altova GmbH



Step 2: Trigger the workflow

The Global Resource created in the previous step can now be consumed from XMLSpy, as follows.

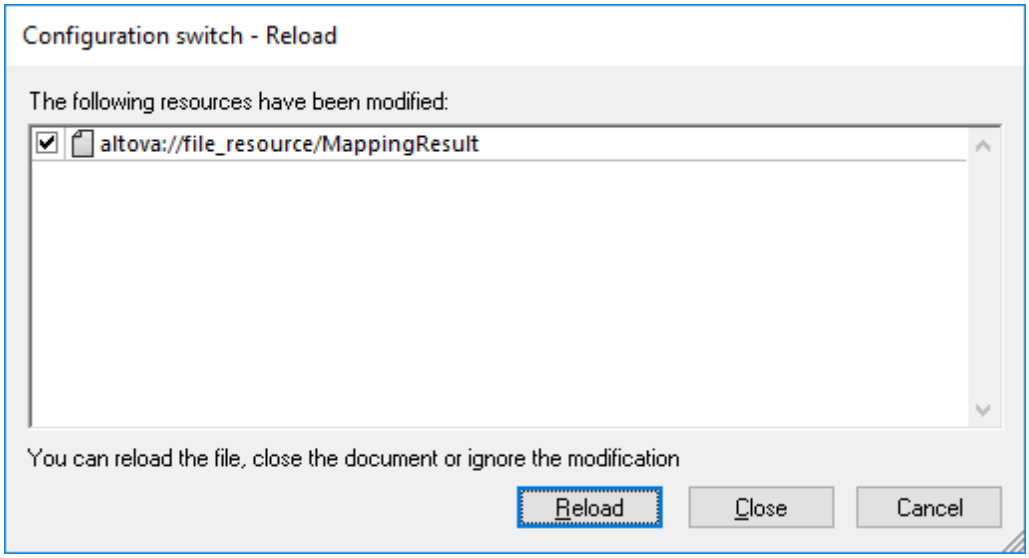
1. Run XMLSpy.
2. On the **Tools** menu, click **Global Resources**.
3. In the "Files" group, click the "MappingResult" Global Resource created previously, and then click **View**.



This executes the mapping, produces the default output (**file1.xml**) and loads it into the main pane of XMLSpy. The file is also saved as **C:\temp\file1.xml**.

To trigger the mapping execution with the alternative configuration, do the following:

1. On the **Tools** menu, click **Active Configuration | Output2**.



2. Click **Reload** when prompted.

As a result, the alternative output file is loaded into the main pane of XMLSpy. The file is also saved as: **C:\temp\file2.xml**.

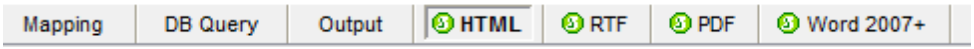
 © 2018 Altova GmbH

12.3 Aplicar estilos a resultados de la asignación con StyleVision

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

In mappings where the target component is XML or XBRL, it is possible to preview and save the mapping output as HTML, RTF, PDF and Word 2007+ documents, provided that Altova StyleVision is installed on your computer. If you are using the Enterprise edition of StyleVision, then charts will also be rendered in these previews.

When a mapping supports preview in any of these formats, additional tabs become available next to the **Output** tab, for example:



StyleVision preview tabs (MapForce Enterprise Edition)

Note the following:

- When StyleVision Professional is installed, it is possible to preview **HTML** and **RTF** output. With StyleVision Enterprise, it is possible to preview **HTML**, **RTF**, **PDF**, and **Word 2007+** output.
- Previewing mapping output as PDF requires Java, Acrobat Reader, and FOP (Formatting Objects Processor) version 0.93 or 1.0. FOP is installed together with StyleVision, unless you opted not to install it when installing StyleVision.
- In the 64-bit edition of MapForce, the Word 2007+ and RTF previews are opened as a non-embedded application.
- If your mapping contains components that act both as source and target (pass-through components), the StyleVision preview will only be possible for those components where the **Preview** button of the component has been set as active. For more information about such mappings, see Chained Mappings.

In order to preview data from a mapping in this way, the following is required:

- Altova StyleVision must be installed on your computer, either as a standalone installation, or as part of Altova MissionKit.
- The target component must have a StyleVision Power Stylesheet (SPS) file associated to it. The stylesheet file can be created or edited with StyleVision. You cannot edit or change the stylesheet in MapForce directly, but you can open it via MapForce in StyleVision. Once the stylesheet is ready, you can assign it to a target MapForce component, as shown below.

Assigning a StyleVision Power Stylesheet to a target component

1. In StyleVision, create the required stylesheet file. Make sure to use as source the same XML schema as that of the MapForce component.
2. In MapForce, right-click the target XML or XBRL component, and select **Properties**.
3. On the Component Settings dialog box, next to **StyleVision Power Stylesheet file**, browse for the stylesheet file created previously.



Note: The path to the StyleVision Power Stylesheet file can be absolute or relative, see Using Relative and Absolute Paths.

Saving the StyleVision-generated output

You can save the StyleVision-generated HTML, PDF, RTF, or Word 2007+ output to a file in a similar way as saving the result of any other mapping. Namely, after previewing the mapping, do one of the following:

- Click the **Save generated output** ( ) toolbar button.
- On the **Output** menu, click **Save Output File**.

Automating generation of HTML, PDF, RTF, Word 2007+ files with Altova product suite

If your mapping should generate HTML, PDF, RTF, and Word 2007+ files automatically (either on the same or on a different computer or even platform), this is possible with MapForce Server and StyleVision Server (these are separately licensed server products that extend the functionality of MapForce and StyleVision, respectively). In this scenario, each application plays the following distinct role:

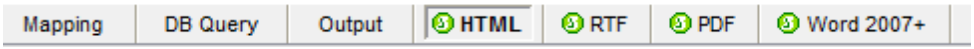
- MapForce - enables you to design the mapping (.mfd file) which defines the data transformation inputs and outputs (for example, database to XML)
- MapForceServer - runs the executable mapping (.mfx file) at the command-line or from an API (either on the same or a different operating system)
- StyleVision - enables you to design the stylesheet (.sps file) required to transform mapping output to HTML, PDF, RTF, Word 2007+
- StyleVision Server - runs the .sps stylesheet which transforms the mapping output to a target desired format. This happens at the command line or from an API (either on the same or a different operating system).
- Both StyleVision Server and MapForce Server can optionally run under the management of FlowForce Server (licensed separately). In this scenario, MapForce mappings and StyleVision transformations can run as scheduled, triggered, or on-demand jobs, and thus be fully automated.

 © 2018 Altova GmbH

12.3.1 Ejemplos de asignaciones con hojas de estilos StyleVision

En este momento no podemos ofrecer este contenido en su idioma, pero aquí tiene la versión en inglés.

Many of the mappings included in the MapForce examples folder (**<Documentos>\Altova \MapForce2018\MapForceExamples**) have StyleVision Power Stylesheets (.sps files) assigned to their target components. When that is the case, the mapping contains the additional StyleVision preview tabs.

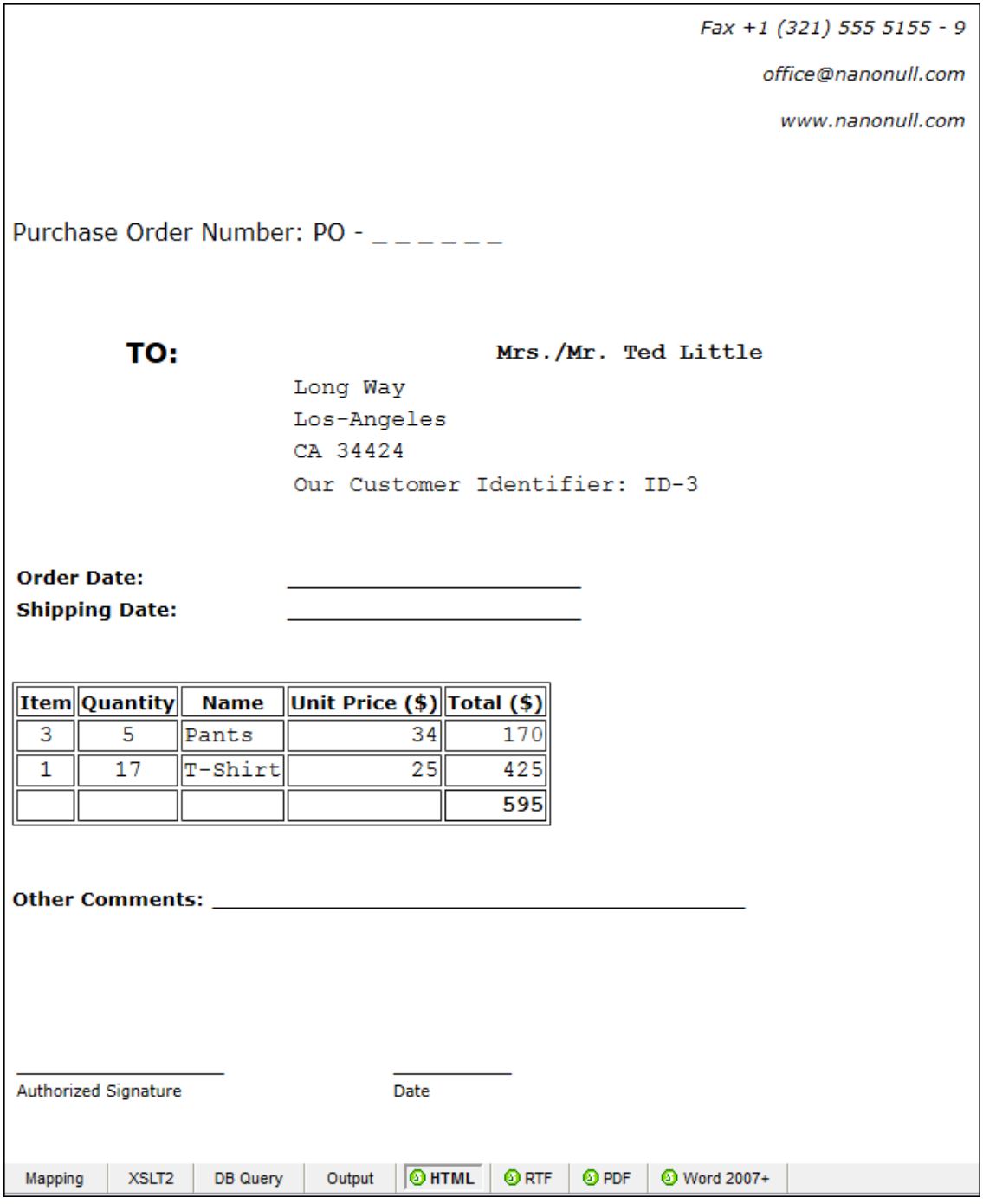


StyleVision preview tabs (MapForce Enterprise Edition)

One such example is **CompletePO.mfd** available at the following path: **<Documentos>\Altova \MapForce2018\MapForceExamples\CompletePO.mfd**. This mapping produces a purchase order in XML format. Right-click the target component, select **Properties**, and notice that it has an .sps file assigned to it.



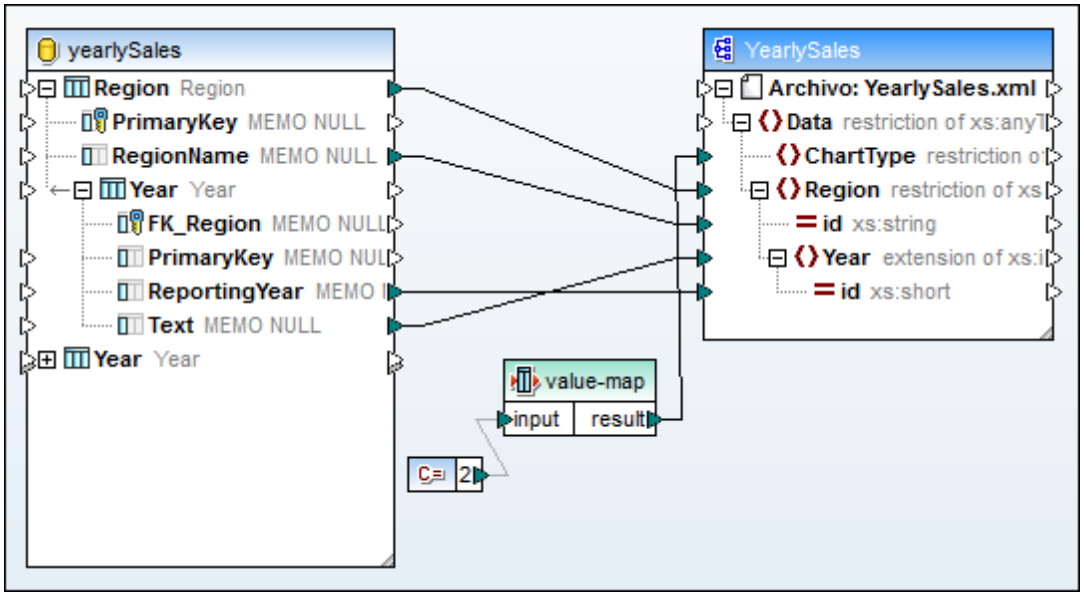
Click the **Output** tab to view the output data in HTML format.



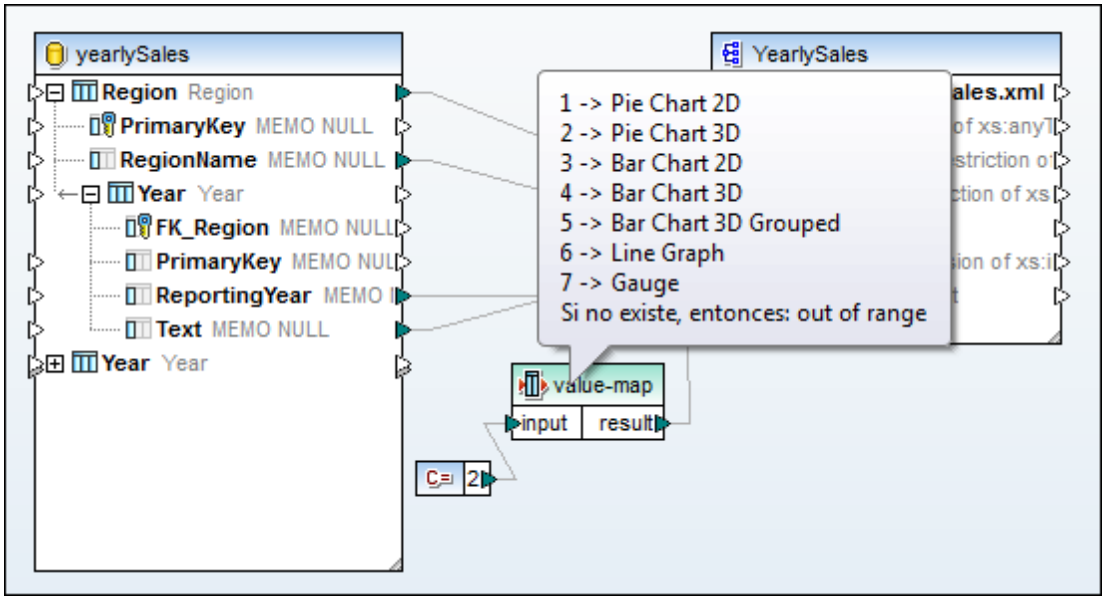
HTML preview of CompletePO.mfd (MapForce Enterprise edition)

Another example is **YearlySales.mfd**. This mapping is available at the following path: **<Documentos>\Altova\MapForce2018\MapForceExamples\Tutorial\YearlySales.mfd**.

 © 2018 Altova GmbH



The stylesheet assigned to this mapping was designed (in StyleVision) in such a way that it is possible to control the type of the chart by changing the value of the `ChartType` element. This makes it possible to change the chart type directly from the mapping. Namely, you can change the default value of the constant to any value from 1 through 7. Notice that, if you place the mouse cursor over the **value-map** component, the possible values are displayed.



The default value of the constant is "2", which generates a 3D Pie chart in the output. To display other chart types, change this value to any other allowed value, click the **Output** tab, and observe the result.

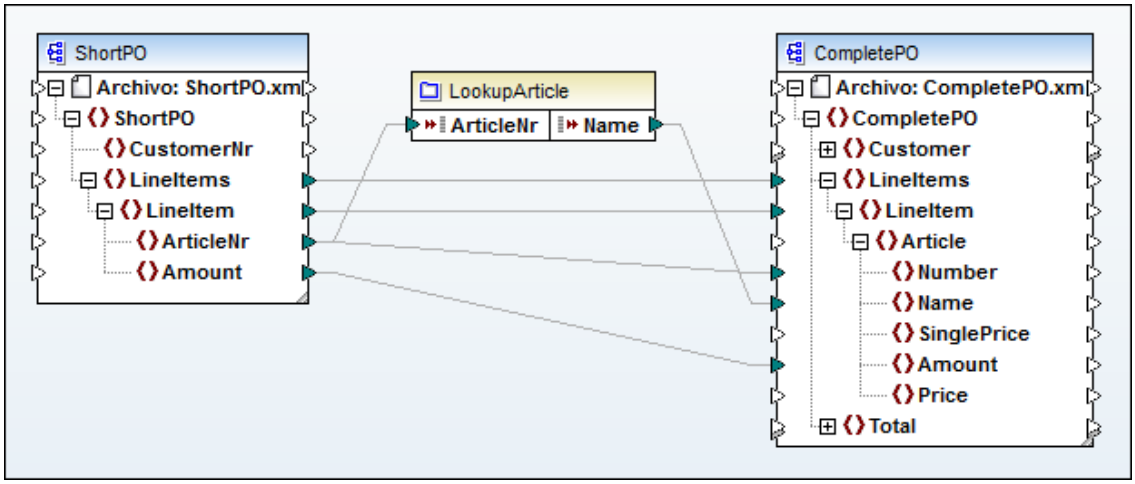
12.4 Generar y personalizar la documentación de la asignación

Con el comando **Archivo | Generar documentación** puede generar documentación detallada sobre su asignación en formato HTML, MS Word, RTF y PDF. La documentación generada se puede modificar y utilizar sin el permiso de Altova.

Al hacer clic en el comando aparece el cuadro de diálogo "Generar documentación", donde puede seleccionar qué componentes se incluyen en la documentación. Además puede aplicar el diseño fijo predeterminado de MapForce o usar una hoja de estilos SPS de StyleVision para dar formato personalizado a la documentación (*véase* *Diseño personalizado*).

Nota: para generar documentación basada en una hoja de estilos SPS es necesario tener StyleVision instalado en el equipo. Los elementos relacionados entre sí están enlazados por medio de hipervínculos para que pueda navegar de componente en componente con facilidad. Recuerde que para generar la documentación en formato MS Word es necesario tener MS Word (2000 o superior) instalado en el equipo.

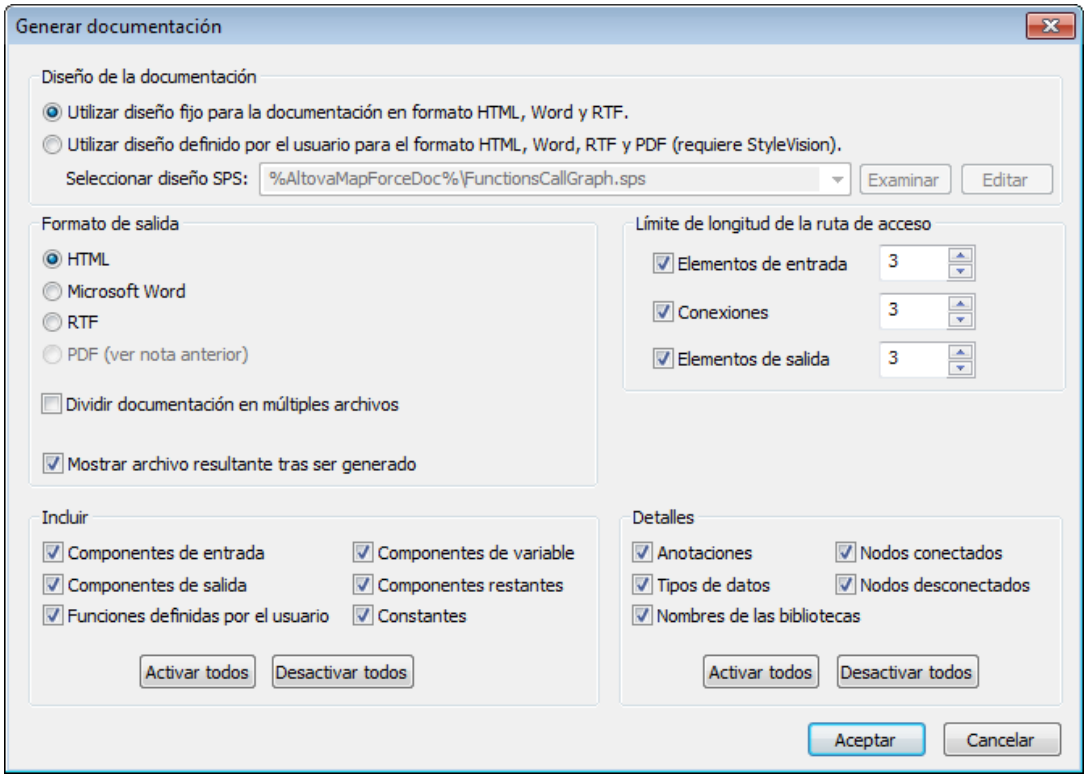
La imagen que aparee a continuación muestra un fragmento de la asignación `Lookup-standard.mfd`, disponible en la carpeta `...\MapForceExamples`.



Para generar la documentación de esta asignación:

- Seleccione la opción de menú **Archivo | Generar documentación**.
  Aparece este cuadro de diálogo:

© 2018 Altova GmbH



Diseño de la documentación

- *Utilizar diseño fijo....:* seleccione esta opción para usar la hoja de estilos integrada para dar formato a la documentación.
- *Utilizar diseño definido por el usuario...:* seleccione esta opción para usar una hoja de estilos SPS creada en StyleVision. MapForce viene con varios archivos SPS que puede usar para dar formato a la documentación y están disponibles en la carpeta `...\Documentos\Altova\MapForce2018\Documentation\MapForce\`.
- Botón **Examinar**: haga clic en este botón para buscar el archivo SPS predefinido que desea usar.
- Botón **Editar**: haga clic en este botón para editar el archivo SPS seleccionado en StyleVision.

Estas son las hojas de estilos SPS predefinidas disponibles en la carpeta `...MapForce2018\Documentation\MapForce`:

- **FunctionCallGraph.sps:** muestra el gráfico de llamadas de la asignación principal y, si las hay, de las funciones definidas por el usuario.
- **FunctionsUsedBy.sps:** muestra qué funciones se usan de forma directa o indirecta en la asignación.
- **ImpactAnalysis.sps:** enumera todos los nodos de origen y destino y la ruta de conexión a través de funciones hasta el nodo de destino.
- **OverallDocumentation.sps:** muestra todos los nodos de origen, conexiones, funciones y nodos de destino. Se trata de la documentación más detallada y es idéntica a la del diseño fijo (opción *Utilizar diseño fijo...*).

Formato de salida

- En este grupo de opciones puede elegir el formato de salida de la documentación: *HTML*, *Microsoft Word*, *RTF* o *PDF*.

  Los documentos de Microsoft Word se crean con la extensión de archivo `.doc` si utiliza el diseño fijo y con la extensión `.docx` si usa un archivo SPS de StyleVision.

  El formato de salida PDF está disponible solamente si usa un archivo SPS de StyleVision para generar la documentación.

- *Dividir documentación en múltiples archivos*: seleccione esta opción si desea documentar los componentes de entrada, salida, constantes y funciones definidas por el usuario en archivos separados. Si utiliza el diseño fijo, se crean hipervínculos entre los archivos de forma automática.

- *Mostrar archivo resultante tras ser generado*: esta opción está disponible para todos los formatos de salida. Si marca esta casilla, los archivos de documentación resultantes se abren en el explorador predeterminado (si eligió el formato HTML), en MS Word (formato MS Word) y en la aplicación predeterminada para archivos RTF (formato RTF).

Límite de longitud de la ruta de acceso

En este grupo de opciones puede definir la longitud máxima de la ruta de acceso que tienen los elementos en la documentación. P. ej. `.../ShortPO/LineItems/LineItem`, corresponde a una longitud máxima de 3 (3 es la configuración predeterminada).

Incluir

En este grupo de opciones puede definir qué componentes aparecen en la documentación.

Detalles

En este grupo de opciones puede elegir qué información se incluye en la documentación.

- Si marca la casilla *Nombres de las bibliotecas* en los nombres de las funciones aparece, por ejemplo, el prefijo `core`.
- Puede documentar tanto nodos conectados como nodos desconectados.

Nota: con los botones **Activar todos** / **Desactivar todos** puede seleccionar rápidamente las casillas de cada grupo.

Cuando termine de configurar la documentación, haga clic en **Aceptar**. Aparece un cuadro de diálogo donde debe introducir el nombre del archivo de salida y la ubicación donde desea guardarlo. A continuación puede ver un fragmento de un archivo de documentación (una documentación de asignación formada por un solo archivo).

Esta tabla enumera las conexiones salientes **desde** el componente de salida hasta el componente de destino.

© 2018 Altova GmbH

lookup-standard.mfd (C:\Altova\MapForce2013\MapForceExamples\lookup-standard.mfd)

Entrada **ShortPO** (ShortPO.xsd)

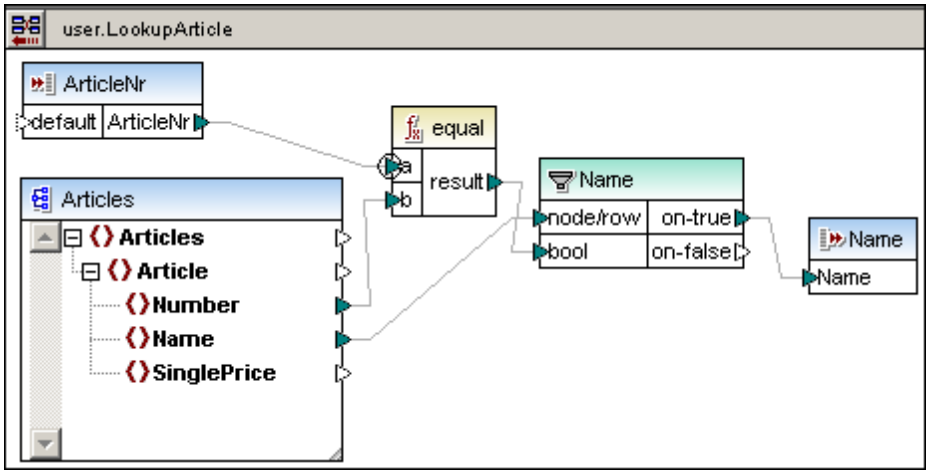
Nodos	Conexiones	
Archivo: ShortPO.xml Tipo: string		
ShortPO Tipo: restriction of xs:anyType [0..1]		
ShortPO/CustomerNr Tipo: xs:integer		
ShortPO/LineItems Tipo: restriction of xs:anyType	*directa*	CompletePO/LineItems Tipo: restriction of xs:anyType
ShortPO/LineItems/LineItem Tipo: restriction of xs:anyType [1..∞]	*directa*	CompletePO/LineItems/LineItem Tipo: restriction of xs:anyType [1..∞]
.../LineItems/LineItem/ArticleNr Tipo: xs:integer	*directa*	.../LineItem/Article/Number Tipo: xs:integer
	user.LookupArticle => ArticleNr | Name =>	.../LineItem/Article/Name Tipo: xs:string
.../LineItems/LineItem/Amount Tipo: xs:integer	*directa*	.../LineItem/Article/Amount Tipo: xs:integer

Los componentes aparecen documentados en este orden: componentes de entrada, salidas, constantes, funciones definidas por el usuario y, por último, funciones de bibliotecas.

P. ej. en la imagen anterior podemos ver la documentación del **componente de entrada ShortPO**:

- Los dos primeros elementos (**`ShortPO`** y **`ShortPO/CustomerNr`**) no están conectados con ningún elemento del componente de salida. Por tanto, las celdas de la columna **Conexiones** están vacías.
- El elemento **`ShortPO/LineItems`** está conectado directamente con el elemento **`CompletePO/LineItems`** del componente de destino.
- El elemento **`/LineItems/LineItem/ArticleNr`** tiene dos conexiones:
  - una conexión directa con el elemento **`LineItem/Article/Number`** del componente de destino
  - una conexión con la función definida por el usuario **`LookupArticle`**, donde **`ArticleNr`** es el parámetro de entrada y **`Name`** es el parámetro de salida.

  En la imagen siguiente puede ver el contenido de esta función definida por el usuario:



En la imagen siguiente podemos ver la documentación del componente de salida (destino). La tabla enumera las conexiones **entrantes** al componente de destino, desde el componente de origen.

Salida **CompletePO** (CompletePO.xsd)

Conexiones		Nodos
		Archivo: CompletePO.xml Tipo: string
		CompletePO Tipo: restriction of xs:anyType [0..1]
		CompletePO/Customer Tipo: restriction of xs:anyType
ShortPO/LineItems Tipo: restriction of xs:anyType	*directa*	CompletePO/LineItems Tipo: restriction of xs:anyType
ShortPO/LineItems/LineItem Tipo: restriction of xs:anyType [1..∞]	*directa*	CompletePO/LineItems/LineItem Tipo: restriction of xs:anyType [1..∞]
		.../LineItems/LineItem/Article Tipo: extension of ArticleType
.../LineItems/LineItem/ArticleNr Tipo: xs:integer	*directa*	.../LineItem/Article/Number Tipo: xs:integer
.../LineItems/LineItem/ArticleNr Tipo: xs:integer	**user.LookupArticle** => ArticleNr | Name =>	.../LineItem/Article/Name Tipo: xs:string
		.../LineItem/Article/SinglePrice Tipo: xs:decimal
.../LineItems/LineItem/Amount Tipo: xs:integer	*directa*	.../LineItem/Article/Amount Tipo: xs:integer
		.../LineItem/Article/Price Tipo: xs:decimal
		CompletePO/Total Tipo: restriction of xs:anyType [0..1]

- Los dos primeros elementos (**`CompletePO`** y **`CompletePO/Customer`**) no están

 © 2018 Altova GmbH

conectados con ningún elemento del componente de origen. Por tanto, las celdas de la columna **Conexiones** están vacías.

- El elemento **CompletePO/LineItems** está conectado directamente con el elemento **ShortPO/LineItems** del componente de origen.
- El elemento **LineItem/Article/Name** está conectado, a través de la función definida por el usuario **LookupArticle**, con el elemento de origen **LineItems/LineItem/ArticleNr**.

La función definida por el usuario define

user.LookupArticle

Entrada (necesario) **core.input ArticleNr**

Nodos	Conexiones	
ArticleNr Tipo: string	**core.equal** => a | result => **core.filter** => bool | on-true =>	**core.output** => Name Tipo: string

Entrada **Articles** (Articles.xsd)

Nodos	Conexiones	
Archivo: Articles.xml Tipo: string		
Articles Tipo: restriction of xs:anyType [0..1]		
Articles/Article Tipo: ArticleType [1..∞]		
Articles/Article/Number Tipo: xs:integer	**core.equal** => b | result => **core.filter** => bool | on-true =>	**core.output** => Name Tipo: string
Articles/Article/Name Tipo: xs:string	**core.filter** => node/row | on-true =>	**core.output** => Name Tipo: string
Articles/Article/SinglePrice Tipo: xs:decimal		

Salida **core.output Name**

Conexiones		Nodos
core.input | ArticleNr => Tipo: string	**core.equal** => a | result => **core.filter** => bool | on-true =>	Name Tipo: string
Articles/Article/Number Tipo: xs:integer	**core.equal** => b | result => **core.filter** => bool | on-true =>	
Articles/Article/Name Tipo: xs:string	**core.filter** => node/row | on-true =>	

12.4.1 Hojas de estilos predefinidas de StyleVision

Documentación de la asignación PersonListByBranchOffice.mfd creada con el archivo SPS de gráfico de llamadas FunctionsCallGraph.sps

This report shows call graphs of the main mapping and all user-defined functions.

Show all functions Collapse all functions

Main mapping

|---core.equal
|---core.filter
|---user.LookupPerson
| |---core.filter
| |---user.EqualAnd
| | |---core.equal
| | |---core.logical-and
| |---user.Person2Details
| | |---core.concat

user.LookupPerson

|---core.filter
|---user.EqualAnd
| |---core.equal
| |---core.logical-and
|---user.Person2Details
| |---core.concat

user.EqualAnd

|---core.equal
|---core.logical-and

user.Person2Details

|---core.concat

Documentación de la asignación PersonListByBranchOffice.mfd creada con el archivo SPS FunctionsUsedBy.sps para ver funciones utilizadas

© 2018 Altova GmbH

Library **core**

Function	Directly used by	Indirectly used by
core.equal	Main mapping user.EqualAnd	user.LookupPerson
core.filter	Main mapping user.LookupPerson	
core.logical-and	user.EqualAnd	Main mapping user.LookupPerson
core.concat	user.Person2Details	Main mapping user.LookupPerson

Library **user**

Function	Directly used by	Indirectly used by
user.LookupPerson	Main mapping	
user.EqualAnd	user.LookupPerson	Main mapping
user.Person2Details	user.LookupPerson	Main mapping

Documentación de la asignación PersonListByBranchOffice.mfd creada con el archivo SPS de análisis de impacto ImpactAnalysis.sps

This report lists every input and output node connection independently and is perfect for further impact analysis with modelling tools.

Input Node	Functions	Output Node
OfficeName	core.equal, core.filter	PersonList
OfficeName	user.LookupPerson	PersonList/Person/Details
BranchOffices/Office	core.filter	PersonList
BranchOffices/Office/Name	core.equal, core.filter	PersonList
BranchOffices/Office/Contact		PersonList/Person
.../Office/Contact/first		PersonList/Person/First
.../Office/Contact/first	user.LookupPerson	PersonList/Person/Details
.../Office/Contact/last		PersonList/Person/Last
.../Office/Contact/last	user.LookupPerson	PersonList/Person/Details

Documentación general de la asignación PersonListByBranchOffice.mfd

creada con el archivo SPS OverallDocumentation.sps

Mapping **PersonListByBranchOffice.mfd**

Input **core.OfficeName**

Nodes	Connections	
OfficeName Type: string	**core.equal => a | result =>** **core.filter => bool | on-true =>**	PersonList Type: restriction of xs:anyType [0..1] Annotation: List of Persons
	user.LookupPerson => **Office_Name | result =>**	PersonList/Person/Details Type: xs:string [0..1]

Input **BranchOffices** (BranchOffices.xsd)

Nodes	Connections	
File: BranchOffices.xml Type: string		
BranchOffices Type: restriction of xs:anyType [0..1]		
BranchOffices/Name Type: restriction of xs:string		
BranchOffices/Office Type: restriction of xs:anyType [0..∞]	**core.filter => node/row | on-true =>**	PersonList Type: restriction of xs:anyType [0..1] Annotation: List of Persons
BranchOffices/Office/Name Type: restriction of xs:string	**core.equal => b | result =>** **core.filter => bool | on-true =>**	PersonList Type: restriction of xs:anyType [0..1] Annotation: List of Persons

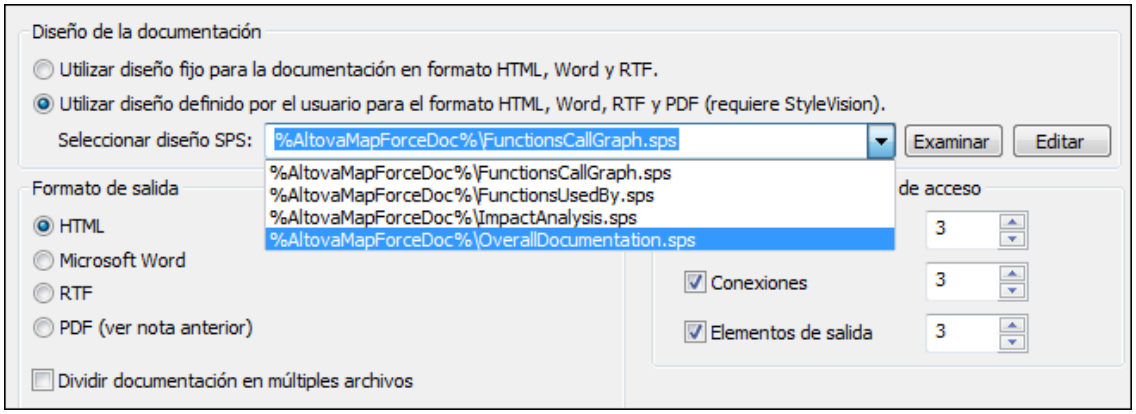
12.4.2 Diseño personalizado

Si lo prefiere, en lugar de usar el diseño de documentación fijo, puede crear un diseño personalizado para generar su documentación de MapForce. El diseño personalizado se crea en un archivo SPS de StyleVision. Recuerde que MapForce viene con cuatro archivos SPS predefinidos que puede usar para la documentación. Para más información consulte el apartado anterior.

Seleccionar el archivo SPS para la documentación de MapForce

En el cuadro de diálogo "Generar documentación" (**Archivo | Generar documentación**) puede seleccionar el archivo SPS que desea usar para generar la documentación. Seleccione el botón de opción *Utilizar diseño definido por el usuario...* y haga clic en la flecha desplegable del cuadro combinado y seleccione un archivo. El archivo SPS seleccionado por defecto es `OverallDocumentation.sps`.

© 2018 Altova GmbH



Estos archivos SPS predefinidos están en la carpeta `...MapForce2018\Documentation\MapForce`.

Nota: para generar la documentación con un archivo SPS es necesario tener StyleVision instalado en el equipo.

Crear el archivo SPS

Los archivos SPS (StyleVision Power Stylesheet) se crean con Altova StyleVision. El archivo SPS para la documentación de MapForce debe estar basado en el esquema XML que define la estructura del documento XML que contiene la documentación de MapForce.

Este esquema se llama **MapForceDocumentation.xsd** y viene en el paquete de instalación de MapForce. Puede encontrarlo en el carpeta `...\Documentos\Altova\MapForce2018\Documentation\MapForce`.

Para crear un diseño SPS en StyleVision, arrastre los nodos del esquema `MapForceDocumentation.xsd` al panel de diseño y asígneles estilos y propiedades. Tenga en cuenta que el esquema `MapForceDocumentation.xsd` **incluye** el archivo de esquema `Documentation.xsd`, que se encuentra en la carpeta primaria.

También puede añadir componentes adicionales como hipervínculos e imágenes al diseño SPS. Para más información, consulte el *Manual del usuario de StyleVision*.

La ventaja de usar un archivo SPS para generar la documentación de la asignación es un mayor control del diseño de la documentación. Además, si utiliza un archivo SPS también puede generar la documentación en formato PDF.

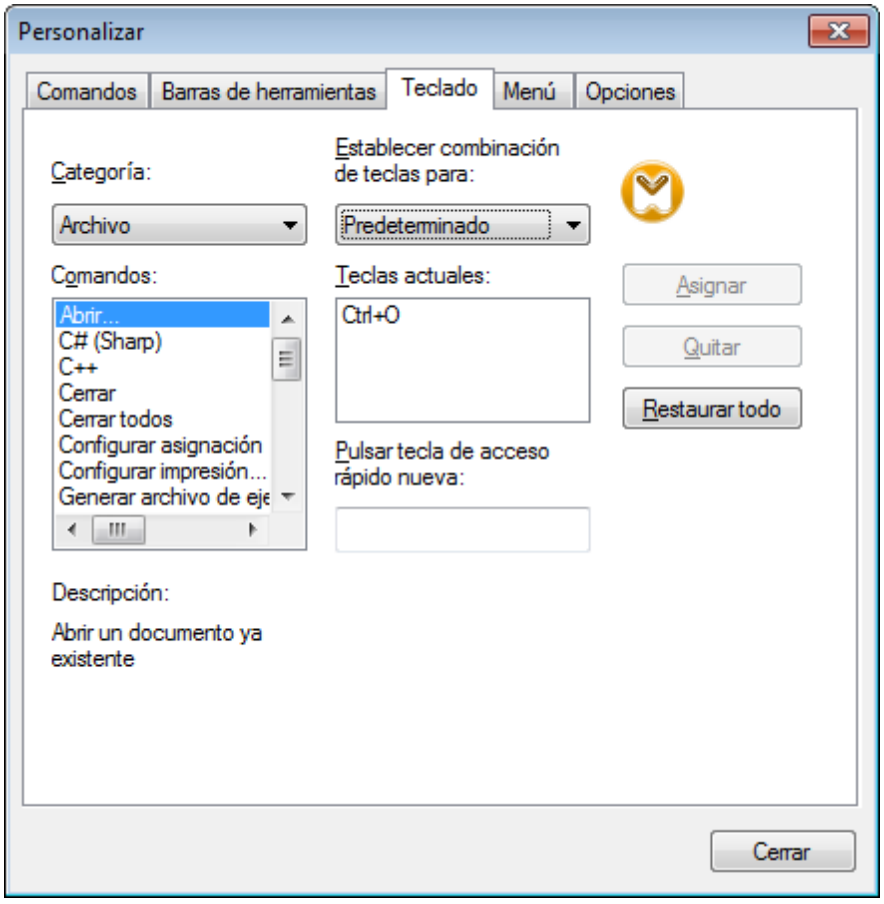
12.5 Personalizar teclas de acceso rápido

En MapForce las teclas de acceso rápido se definen y modifican así:

1. Haga clic en el menú **Herramientas** y después en **Personalizar**.
2. Haga clic en la pestaña *Teclado*.

Para asignar una tecla de acceso rápido a un comando:

1. Seleccione **Herramientas | Personalizar** y haga clic en la pestaña *Teclado* del cuadro de diálogo "Personalizar".
2. Haga clic en el cuadro combinado *Categoría* y seleccione el menú de MapForce donde está el comando.
3. En el cuadro de lista *Comandos* seleccione el comando al que desea asignar una tecla de acceso rápido.
4. Haga clic en el cuadro de texto *Pulsar tecla de acceso rápido nueva* y pulse las teclas que deben activar el comando.



La tecla de acceso rápido aparece inmediatamente en el cuadro de texto. Si la tecla de acceso ya está asignada a algún comando, esto se indica bajo el cuadro de texto.

5. Haga clic en el botón **Asignar** para asignar la tecla de acceso rápido al comando. La tecla de acceso rápido aparece ahora en el cuadro de lista *Teclas actuales*. (Para borrar

© 2018 Altova GmbH

el texto del cuadro de texto *Pulsar tecla de acceso rápido,* pulse cualquier tecla de control (**Ctrl**, **Alt** o **Mayús**).)

Para quitar la asignación de una tecla de acceso rápido o eliminarla:

1. En el cuadro de lista *Teclas actuales* haga clic en la tecla de acceso rápido que desea eliminar.
2. Haga clic en el botón **Quitar**.
3. Haga clic en **Cerrar** para confirmar.

Nota: el cuadro combinado *Establecer combinación de teclas para:* no tiene por ahora ninguna función.

Estas son las teclas de acceso rápido asignadas por defecto:

Teclas de acceso rápido	
F1	Menú **Ayuda**
F2	Marcador siguiente (panel *Resultados*)
F3	Buscar Siguiente
F10	Activar barra de menú
+ del teclado numérico	Expandir el nodo actual
- del teclado numérico	Contraer el nodo actual
* del teclado numérico	Expandir todo a partir del nodo actual
CTRL + TAB	Cambia de una asignación a otra
CTRL + F6	Recorre todas las ventanas abiertas
CTRL + F4	Cierra la asignación activa
Alt + F4	Cierra MapForce
Alt + F, F, 1	Abre el último archivo
Alt + F, T, 1	Abre el último proyecto
CTRL + N	Archivo nuevo
CTRL + O	Abrir archivo
CTRL + S	Guardar archivo
CTRL + P	Imprimir archivo
CTRL + A	Seleccionar todo
CTRL + X	Cortar
CTRL + C	Copiar
CTRL + V	Pegar
CTRL + Z	Deshacer
CTRL + Y	Rehacer
Supr	Eliminar componente (con previo aviso)
Mayús + Supr	Eliminar componente (sin previo aviso)
CTRL + F	Buscar
F3	Buscar siguiente
Mayús + F3	Buscar anterior
Teclas de dirección	
(arriba / abajo)	Seleccionar el siguiente / anterior elemento del

	componente
Esc	Abandonar los cambios / cerrar el cuadro de diálogo
Entrar	Confirma una selección

Teclas de acceso rápido del panel Resultados

CTRL + F2	Insertar o quitar marcador
F2	Siguiente marcador
Mayús + F2	Marcador anterior
CTRL + Mayús + F2	Quitar todos los marcadores

Teclas de acceso rápido de zoom

CTRL + rueda del ratón hacia adelante	Acercarse
CTRL + rueda del ratón hacia detrás	Alejarse
CTRL + 0 (cero)	Restablecer el nivel de zoom

© 2018 Altova GmbH

12.6 Archivos de catálogo

MapForce es compatible con un subconjunto del mecanismo de catalogación XML OASIS. El mecanismo de catalogación permite a MapForce recuperar de carpetas locales del usuario los esquemas (y hojas de estilos y otros archivos) usados con frecuencia. Esto incrementa la velocidad global de procesamiento, permite al usuario trabajar sin conexión (es decir, sin estar conectado a una red) y mejora la portabilidad de los documentos (porque los identificadores URI se tienen que cambiar sólo en los archivos de catálogo).

A continuación describimos cómo funciona el mecanismo de catalogación en MapForce.

RootCatalog.xml

Al iniciarse, MapForce carga un archivo llamado `RootCatalog.xml` (*cuya estructura aparece a continuación*), que contiene una lista de los archivos de catálogo que se buscarán. El usuario puede modificar esta lista y añadir tantos archivos de catálogo como desee, escribiendo cada archivo en un elemento `nextCatalog`. MapForce busca cada uno de estos archivos de catálogo y sus URI se resuelven de acuerdo con las asignaciones especificadas en ellos.

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog"
     xmlns:spy="http://www.altova.com/catalog_ext"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="urn:oasis:names:tc:entity:xmlns:xml:catalog
Catalog.xsd">
  <nextCatalog catalog="%PersonalFolder%/Altova/%AppAndVersionName%/
CustomCatalog.xml"/>
  <nextCatalog catalog="CoreCatalog.xml"/>
  <!-- Include all catalogs under common schemas folder on the first directory
level -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/Schemas"
catalog="catalog.xml" spy:depth="1"/>
  <!-- Include all catalogs under common XBRL folder on the first directory
level -->
  <nextCatalog spy:recurseFrom="%AltovaCommonFolder%/XBRL"
catalog="catalog.xml" spy:depth="1"/>
</catalog>
```

En el fragmento de código anterior, observe que en las carpetas `Schemas` y `XBRL` de la carpeta identificada con la variable `%AltovaCommonFolder%` están los archivo de catálogo llamados `catalog.xml`. (El valor de la variable `%AltovaCommonFolder%` se indica en la tabla que aparece más abajo.)

Los archivos de catálogo de la carpeta de archivos comunes de Altova realizan asignaciones entre los identificadores de sistema e identificadores públicos predefinidos de las taxonomías XBRL y los esquemas (como SVG y WSDL) y los identificadores URI que apuntan a copias locales de los respectivos esquemas. Estos esquemas se instalan en la carpeta de archivos comunes de Altova cuando se instala MapForce. Rogamos no cree asignaciones duplicadas en estos archivos porque se pueden producir errores.

CoreCatalog.xml, CustomCatalog.xml y Catalog.xml

En el ejemplo de código anterior del archivo `RootCatalog.xml` se indica que se busquen los archivos `CoreCatalog.xml` y `CustomCatalog.xml`:

- `CoreCatalog.xml` contiene ciertas asignaciones propias de Altova que sirven para localizar esquemas en la carpeta de archivos comunes de Altova.
- `CustomCatalog.xml` es un archivo base en el que el usuario puede crear asignaciones propias. En el archivo `CustomCatalog.xml` puede crear asignaciones para cualquier esquema, siempre y cuando el esquema no esté controlado por los archivos de catálogo de la carpeta de archivos comunes de Altova. Para crear asignaciones use los elementos compatibles con el mecanismo de catalogación OASIS (*véase más adelante*).
- En la carpeta de archivos comunes de Altova hay varios archivos `Catalog.xml`. Cada archivo está dentro de la carpeta de un esquema o de una taxonomía XBRL concretos de la carpeta de archivos comunes de Altova. Además, cada archivo `Catalog.xml` realiza asignaciones entre los identificadores de sistema e identificadores públicos y los identificadores URI que apuntan a las copias locales de los respectivos esquemas.

Ubicación de los archivos de catálogo y los esquemas

Los archivos `RootCatalog.xml` y `CoreCatalog.xml` se instalan en la carpeta de aplicación de MapForce. El archivo `CustomCatalog.xml` está ubicado en su carpeta `MisDocumentos\Altova\MapForce`. Cada archivo `catalog.xml` está en una carpeta de esquema y estas carpetas están dentro de las carpetas: `%AltovaCommonFolder%\Schemas` y `%AltovaCommonFolder%\XBRL`.

Variables de entorno Shell y variables de Altova

En el elemento `nextCatalog` puede utilizar algunas variables de entorno Shell para indicar la ruta de acceso a las ubicaciones del sistema (*ver el fragmento anterior del archivo RootCatalog.xml*). Estas son las variables de entorno Shell compatibles:

`%AltovaCommonFolder%`	C:\Archivos de programa\Altova\Common2018
`%DesktopFolder%`	Ruta de acceso completa de la carpeta Escritorio del usuario actual.
`%ProgramMenuFolder%`	Ruta de acceso completa de la carpeta del menú Programas del usuario actual.
`%StartMenuFolder%`	Ruta de acceso completa de la carpeta del menú Inicio del usuario actual.
`%StartUpFolder%`	Ruta de acceso completa de la carpeta Inicio del usuario actual.
`%TemplateFolder%`	Ruta de acceso completa de la carpeta de plantillas del usuario actual.
`%AdminToolsFolder%`	Ruta de acceso completa del directorio del sistema de archivos que almacena las herramientas administrativas del usuario actual.
`%AppDataFolder`	Ruta de acceso completa de la carpeta Datos de programa del usuario

© 2018 Altova GmbH

%	actual.
%CommonAppDataFolder%	Ruta de acceso completa del directorio de archivos que contiene datos del programa de todos los usuarios.
%FavoritesFolder%	Ruta de acceso completa de la carpeta Favoritos del usuario actual.
%PersonalFolder%	Ruta de acceso completa de la carpeta personal del usuario actual.
%SendToFolder%	Ruta de acceso completa de la carpeta SendTo del usuario actual.
%FontsFolder%	Ruta de acceso completa de la carpeta Fuentes del sistema.
%ProgramFilesFolder%	Ruta de acceso completa de la carpeta Archivos de programa del usuario actual.
%CommonFilesFolder%	Ruta de acceso completa de la carpeta Common files del usuario actual.
%WindowsFolder%	Ruta de acceso completa de la carpeta Windows del usuario actual.
%SystemFolder%	Ruta de acceso completa de la carpeta System del usuario actual.
%LocalAppDataFolder%	Ruta de acceso completa al directorio del sistema de archivos que sirve como repositorio de datos para aplicaciones locales (no roaming).
%MyPicturesFolder%	Ruta de acceso completa a la carpeta Mis imágenes.

Cómo funcionan los catálogos: documentos DTD

Los catálogos se suelen usar para redireccionar una llamada a una DTD hasta un URI local. Para ello es necesario realizar asignaciones, en el archivo de catálogo, entre los identificadores de sistema o públicos y el URI local pertinente. De este modo, cuando se lee la declaración DOCTYPE en un archivo XML, el identificador de sistema o público localiza el recurso local necesario con ayuda de la asignación del archivo de catálogo.

Para los esquemas más utilizados el identificador `PUBLIC` suele estar predefinido y, por tanto, sólo hace falta que el URI del archivo de catálogo apunte a la copia local correcta. Cuando se analiza el documento XML, se lee el identificador `PUBLIC` del documento. Si se encuentra este identificador en un archivo de catálogo, se buscará la URL correspondiente del archivo de catálogo y se leerá el esquema desde esta ubicación. Por ejemplo, imaginemos que abrimos este archivo SVG en MapForce:

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg width="20" height="20" xml:space="preserve">
  <g style="fill:red; stroke:#000000">
     <rect x="0" y="0" width="15" height="15"/>
     <rect x="5" y="5" width="15" height="15"/>
  </g>
</svg>
```

En este caso se leería el documento y se buscaría el identificador `PUBLIC` en el catálogo. Imaginemos que el archivo de catálogo contiene esta entrada:

```
<catalog>
  ...
    <public publicId="-//W3C//DTD SVG 1.1//EN" uri="schemas/svg/svg11.dtd"/>
  ...
</catalog>
```

En este caso, se encuentra un identificador `PUBLIC`, de modo que la búsqueda de la DTD del SVG se redirecciona al URI `schemas/svg/svg11.dtd` (esta ruta es relativa al archivo de catálogo) y este archivo local se usará como DTD. Si en el catálogo no hay una asignación para el identificador `Public`, entonces se usa la URL del documento XML (en el ejemplo anterior: `http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd`).

El subconjunto de catálogos compatible con MapForce

Cuando cree entradas en el archivo `CustomCatalog.xml` (o en cualquier otro archivo de catálogo que sea leído por MapForce), utilice únicamente los elementos que aparecen a continuación de la especificación de catálogos OASIS. En la lista que aparece más adelante explicamos los valores de los atributos de cada elemento. Si desea consultar una descripción más detallada, visite la página de la especificación XML Catalogs. Recuerde que todos los elementos puede tomar el atributo `xml:base`, que se usa para especificar el URI base del elemento.

- `<public publicId="PublicID of Resource" uri="URL of local file"/>`
- `<system systemId="SystemID of Resource" uri="URL of local file"/>`
- `<uri name="filename" uri="URL of file identified by filename"/>`
- `<rewriteURI uriStartString="StartString of URI to rewrite" rewritePrefix="String to replace StartString"/>`
- `<rewriteSystem systemIdStartString="StartString of SystemID" rewritePrefix="Replacement string to locate resource locally"/>`

Cuando no exista un identificador público, como es el caso de casi todas las hojas de estilos, el identificador de sistema se puede asignar directamente a una URL con el elemento `system`. Además, un URI se puede asignar a otro URI con el elemento `uri`. Los elementos `rewriteURI` y `rewriteSystem` sirven para volver a escribir la parte inicial de un URI o identificador de sistema respectivamente. Gracias a ello se puede sustituir el principio de la ruta de acceso de un archivo y, por consiguiente, se puede apuntar a otro directorio. Para más información sobre estos elementos, consulte la especificación XML Catalogs.

© 2018 Altova GmbH

Extensiones de archivo y edición inteligente basada en un esquema

Mediante los archivos de catálogo también puede especificar a qué tipo de documentos (según su extensión de archivo) se les puede aplicar las funciones de edición inteligente de MapForce de acuerdo con las reglas del esquema que usted indique. Por ejemplo, si crea la extensión de archivo personalizada `.myhtml` para archivos (HTML) que deben ser válidos con respecto a la DTD HTML, entonces puede habilitar la función de edición inteligente para los archivos que tengan esta extensión. Para ello, añada el elemento que aparece a continuación al archivo de catálogo `CustomCatalog.xml` y como elemento secundario del elemento `<catalog>`:

```
<spy:fileExtHelper ext="myhtml" uri="schemas/xhtml/xhtml1-transitional.dtd"/>
```

Así se habilita la función de edición inteligente (finalización automática, ayudantes de entrada, etc.) para archivos `.myhtml` en MapForce en base a la DTD transicional XHTML 1.0. Consulte el archivo `catalog.xml` de la carpeta `%AltovaCommonFolder%\Schemas\xhtml`, que contiene entradas similares a este ejemplo.

Especificaciones de XML Schema

La información de la especificación XML Schema está integrada en MapForce y esta información interna se usa para validar los documentos de esquema XML (`.xsd`). Por tanto, en los documentos de esquema XML no se deberían hacer referencia a ningún esquema que defina la especificación XML Schema.

El archivo `catalog.xml` de la carpeta `%AltovaCommonFolder%\Schemas\schema` contiene referencias a documentos DTD que implementan especificaciones antiguas de XML Schema. Rogamos no valide sus documentos de esquema XML con estos esquemas. Los archivos referenciados se incluyen con el único objetivo de aportar información a MapForce para sus ayudantes de entrada, en caso de que el usuario quiera crear documentos basados en estas recomendaciones.

Más información

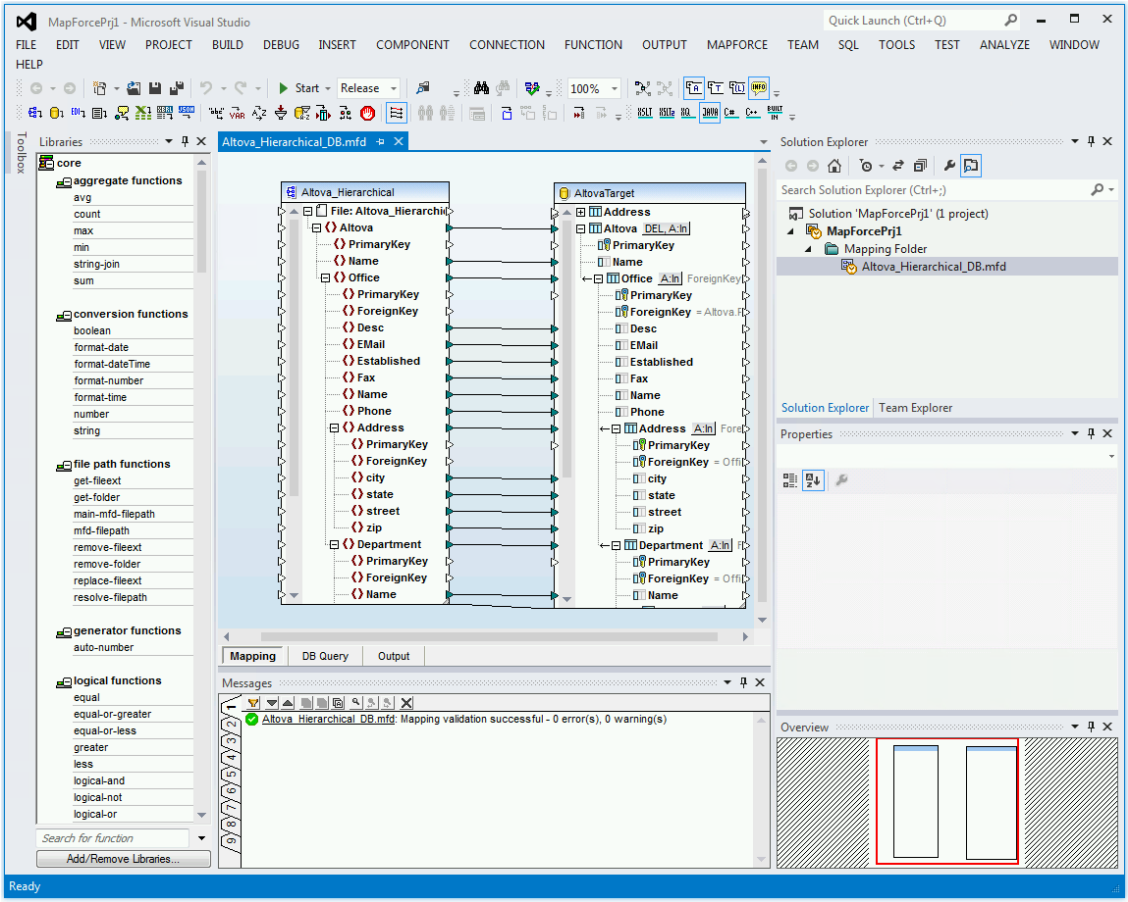
Para más información, consulte la especificación XML Catalogs.

Altova MapForce 2018 Enterprise Edition

Complemento de MapForce para Visual Studio

13 Complemento de MapForce para Visual Studio

Puede integrar MapForce 2018 en Microsoft Visual Studio 2008/2010/2012/2013/2015/2017. Esto permite combinar las funciones de asignación de datos de MapForce con el entorno de desarrollo de Visual Studio. Cuando tenga el complemento de MapForce habilitado, podrá crear proyectos y archivos de asignación de datos desde Visual Studio directamente. También podrá personalizar las opciones de MapForce (como los menús y las barras de herramientas).



Complemento MapForce Enterprise Edition (en Visual Studio 2012)

Esta sección incluye varios apartados:

- Habilitar el complemento
- Trabajar con asignaciones y proyectos
- Acceso a menús y funciones frecuentes

© 2018 Altova GmbH

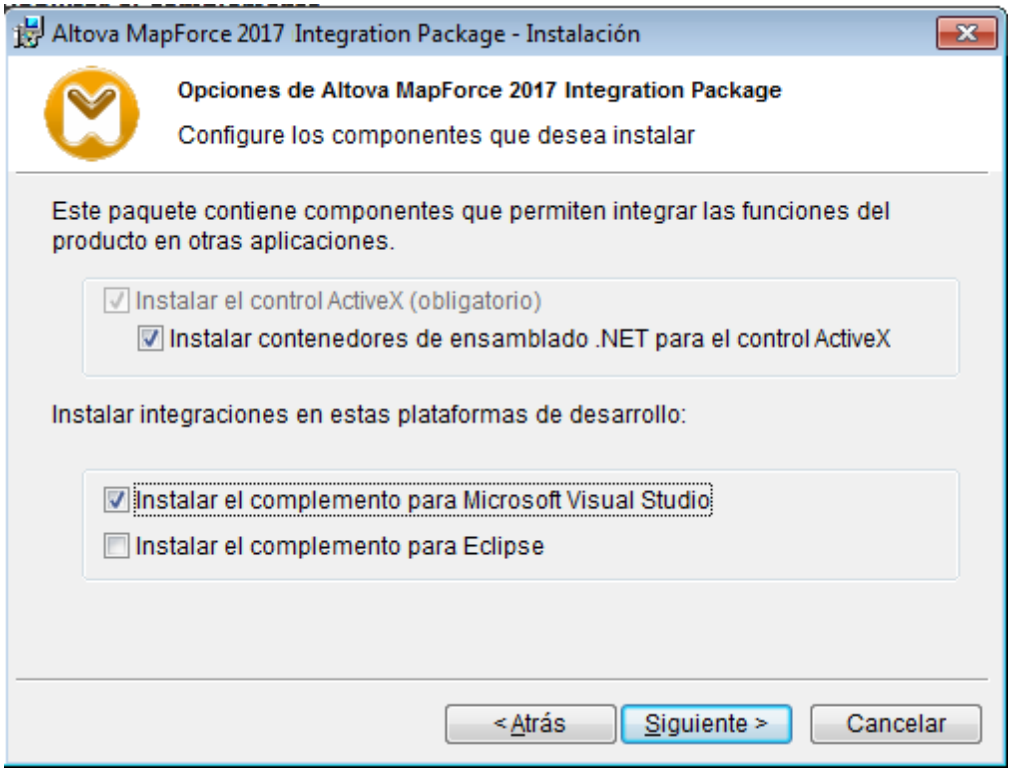
13.1 Habilitar el complemento

Requisitos:

- Microsoft Visual Studio 2008/2010/2012/2013/2015/2017
- MapForce (edición Enterprise Edition o Professional Edition)
- Paquete de integración de MapForce (se puede descargar desde https://www.altova.com/es/download/mapforce.html.

Nota: para poder usar MapForce como complemento de Visual Studio instale la versión de 32 bits de MapForce y del paquete de integración de MapForce. Esto es necesario porque Visual Studio no admite complementos de 64 bits.

Para habilitar el complemento de MapForce para Visual Studio lo primero que debe hacer es descargar y ejecutar el paquete de integración de MapForce. Durante el proceso de instalación siga las instrucciones que aparecen en pantalla y asegúrese de seleccionar la opción *Instalar el complemento para Visual Studio*.



Después deberá seleccionar la versión de Visual Studio donde se debe habilitar el complemento de MapForce. Por ejemplo:



Nota: solamente podrá seleccionar versiones de Visual Studio que estén instaladas en el equipo.

Una vez instalado el paquete de integración, todas las funciones y características de MapForce estarán disponibles en Visual Studio.

Habilitar el complemento de MapForce a mano

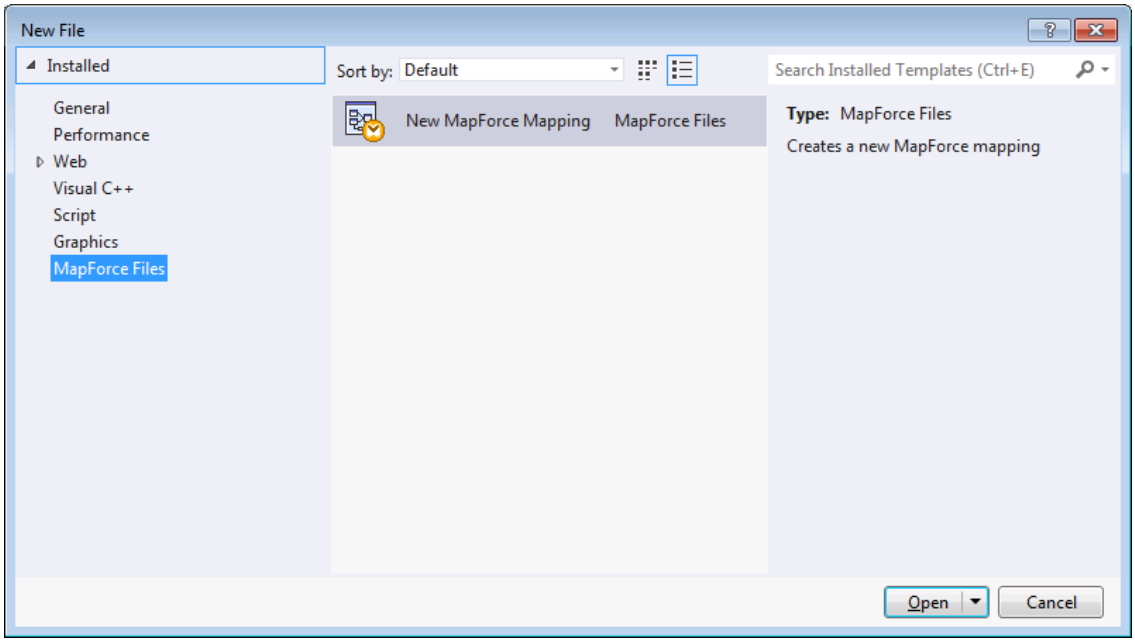
Si el complemento no se habilita automáticamente durante el proceso de instalación:

1. Navegue hasta el directorio donde está instalado el ejecutable de Visual Studio (p. ej. `c:\Archivos de programa\Microsoft Visual Studio 8\Common7\IDE`).
2. Ejecute el símbolo del sistema como administrador e introduzca `devenv.exe /setup`.
3. Espere a que termine el proceso antes de empezar a usar la aplicación en Visual Studio.

© 2018 Altova GmbH

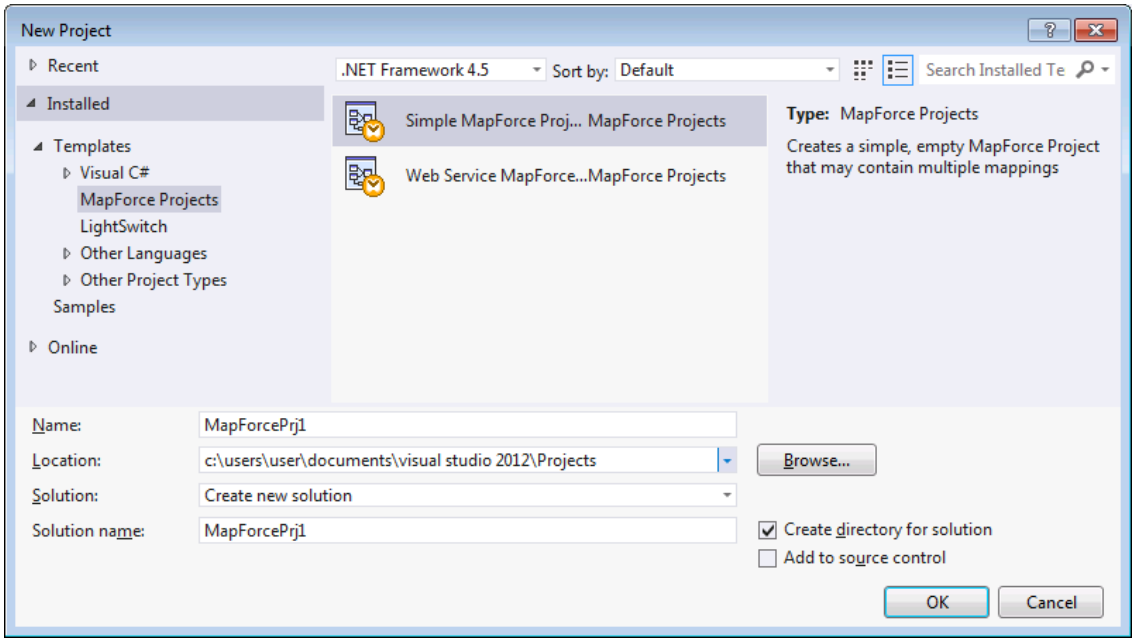
13.2 Trabajar con asignaciones y proyectos

Cuando está habilitado el complemento de MapForce para Visual Studio, se pueden crear y abrir asignaciones y proyectos de asignación en el entorno Visual Studio y no solo en la interfaz gráfico de MapForce. Por ejemplo, cuando cree un archivo nuevo en Visual Studio (con el comando **Archivo | Nuevo**) o cuando añada un elemento nuevo al proyecto (con el comando de menú **Proyecto | Agregar elemento nuevo**), podrá seleccionar el tipo de archivo **Archivos de MapForce**.



Cuadro de diálogo "Archivo nuevo" (Visual Studio 2012 con el complemento de MapForce Enterprise Edition)

Del mismo modo, cuando cree un proyecto nuevo de Visual Studio, podrá seleccionar **Proyectos de MapForce** como plantilla de proyecto. En la imagen que aparece a continuación puede ver un cuadro de diálogo "Proyecto nuevo" de Visual Studio 2012 con el complemento MapForce Enterprise Edition habilitado.



Cuadro de diálogo "Proyecto nuevo" (Visual Studio 2012 con el complemento de MapForce Enterprise Edition)

Desde la interfaz nativa de Visual Studio también podrá abrir archivos y proyectos de MapForce (p. ej. desde el menú **Archivo | Abrir | Archivos** o **Archivo | Abrir | Proyecto/Soluciones**).

© 2018 Altova GmbH

13.3 Acceso a menús y funciones frecuentes

Cuando está habilitado el complemento de MapForce para Visual Studio, tendrá acceso a los menús y a las funciones que aparecen a continuación. Se trata de la configuración predeterminada, pero si quiere puede cambiar la posición de los menús y de las barras de herramientas desde el menú **Herramientas | Opciones** de Visual Studio.

Recursos globales

La característica de recursos globales de MapForce está disponible en el menú **MapForce | Recursos globales** de Visual Studio.

Depuración de asignaciones de datos

Los comandos de depuración de asignaciones de datos están en el menú **MapForce | Depuración** y en la barra de herramientas **Depuración**. También puede iniciar la depuración con las teclas de acceso rápido (*véase Comandos del depurador*).

Opciones de MapForce

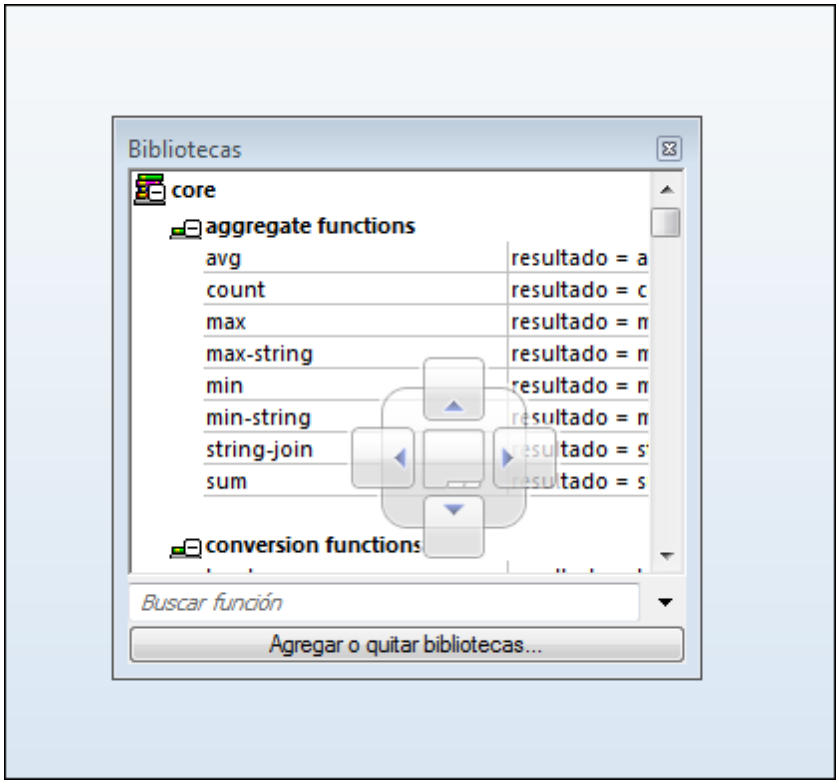
Las opciones de configuración de MapForce están disponibles en el menú **Herramientas | Opciones de MapForce** de Visual Studio.

Personalizar el panel de asignación

Cuando en el panel principal de Visual Studio hay una asignación de MapForce abierta, se habilita el menú **Vista | MapForce**. Este menú incluye las mismas opciones que la versión independiente de MapForce.

Ventana Bibliotecas

La ventana Bibliotecas de MapForce no está habilitada por defecto en Visual Studio. Si suele trabajar con esta ventana, puede habilitarla desde el menú **Vista | MapForce | Ventana Bibliotecas** (este menú se habilita en Visual Studio cuando en la ventana principal de Visual Studio hay una asignación de datos abierta). Una vez habilitada la ventana Bibliotecas, podrá acoplarla en la interfaz al igual que cualquier otro componente de Visual Studio.

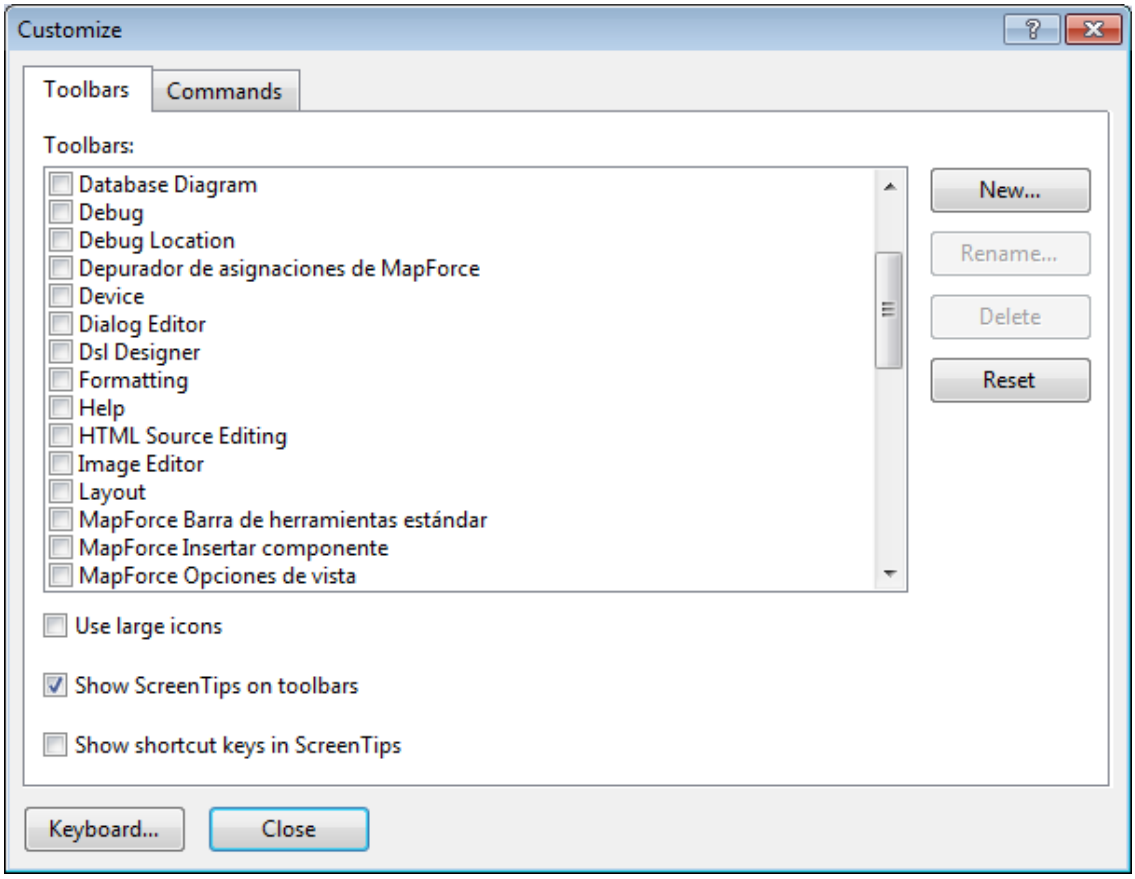


Ventana Bibliotecas (Visual Studio 2008 con complemento MapForce Enterprise Edition)

Personalizar barras de herramientas y comandos

Puede personalizar los menús y las barras de herramientas de MapForce desde el menú **Herramientas | Opciones** de Visual Studio.

© 2018 Altova GmbH



Cuadro de diálogo "Personalizar" (Visual Studio 2008 con complemento MapForce Enterprise Edition)

Ayuda y soporte técnico

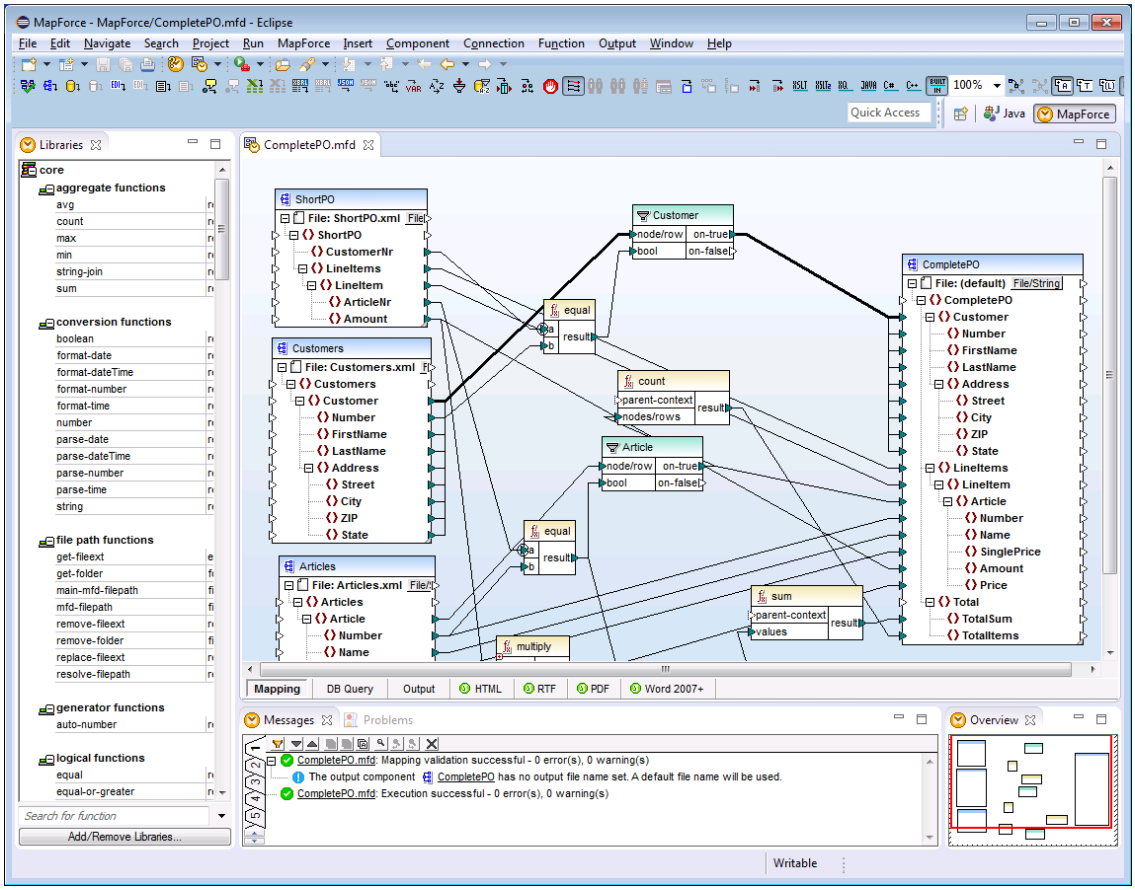
Los comandos de ayuda, soporte técnico y actualización del software están en el menú **Ayuda | Ayuda de MapForce** de Visual Studio.

Altova MapForce 2018 Enterprise Edition

Complemento de MapForce para Eclipse

14 Complemento de MapForce para Eclipse

Eclipse es un marco de trabajo de código abierto que incluye varios tipos de aplicaciones en forma de complementos. MapForce Enterprise y Professional se pueden integrar en Eclipse 4.5 / 4.6 / 4.7 y podrá acceder a todas las funciones de MapForce desde Eclipse directamente.



Complemento de MapForce Enterprise Edition para Eclipse

Los apartados de esta sección ofrecen instrucciones para instalar y usar el complemento de MapForce en Eclipse.

- Instalación
- Perspectiva MapForce
- Acceso a menús y funciones frecuentes
- Trabajar con asignaciones y proyectos
- Ampliar el complemento de MapForce para Eclipse

© 2018 Altova GmbH

14.1 Instalación

Requisitos:

- MapForce Enterprise o Professional Edition.
- Java Runtime Environment (JRE) 6.0 o superior (véase http://www.oracle.com/technetwork/java/javase/downloads/index.html). Instale la versión de 32 o 64 bits dependiendo de la versión de MapForce que tenga instalada.
- Eclipse Platform 4.5 / 4.6 / 4.7 (véase http://www.eclipse.org). Instale la versión de 32 o 64 bits dependiendo de la versión de MapForce que tenga instalada.

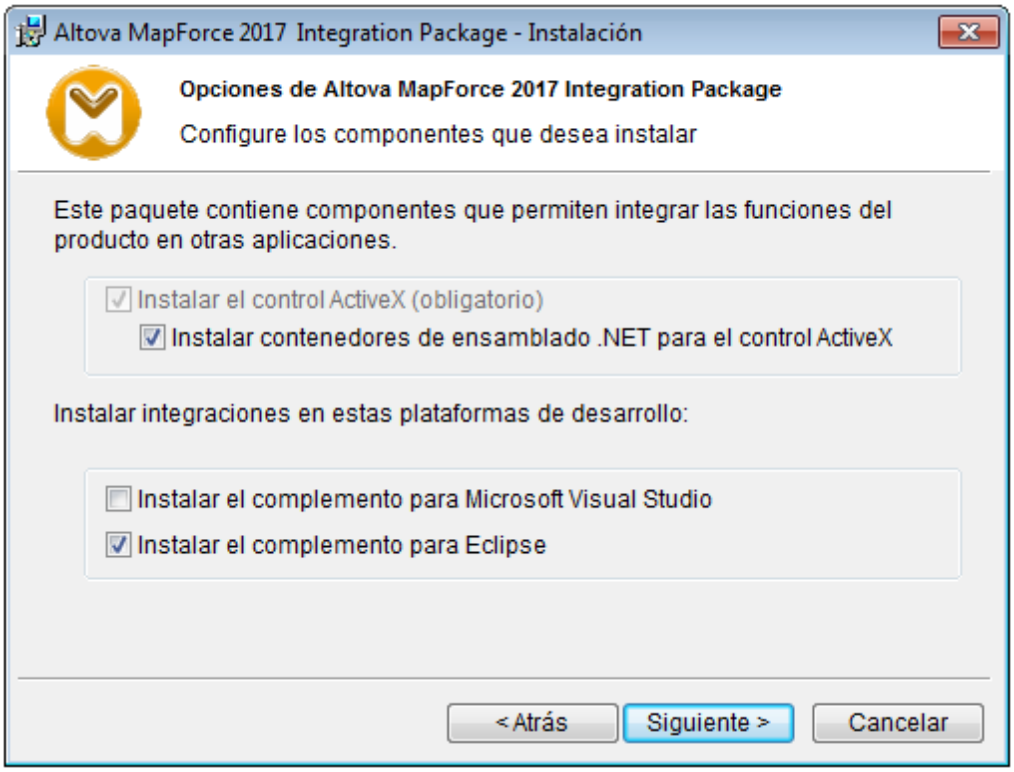
> Si ya ha instalado Eclipse 4.5 usando el instalador de Eclipse, no es posible ejecutar en el mismo equipo las versiones de 32 y 64 bits del complemento MapForce. Esta limitación tiene su origen en el instalador de Eclipse y no se aplica si instala manualmente tanto las versiones de Eclipse (de 32 y 64 bits).

Instalar el complemento de MapForce para Eclipse

1. Descargue el paquete de integración de MapForce desde la página de descargas de Altova (http://www.altova.com/es/download_components.html).
2. Compruebe que Eclipse no está en ejecución y ejecute el paquete de integración que acaba de descargar.

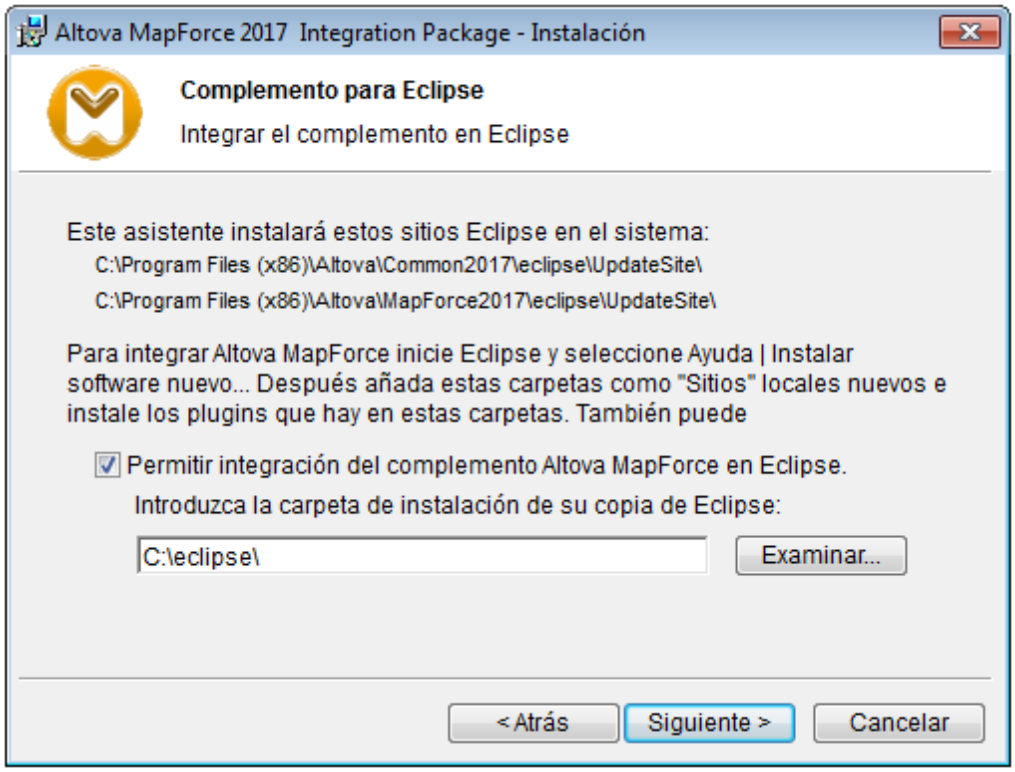
> Eclipse debe estar cerrado mientras se instala o desinstala el paquete de integración de MapForce.

3. Llegado el momento, seleccione la opción *Instalar el complemento para Eclipse* y después haga clic en **Siguiente**.



4. Cuando deba elegir cómo se integra el complemento de MapForce en Eclipse, siga estas instrucciones:
   a. Para completar la instalación a mano (opción recomendada), seleccione *Permitir integración del complemento Altova MapForce en Eclipse* y busque el directorio donde está ubicado el ejecutable de Eclipse (`eclipse.exe`).
   b. Para completar la instalación del complemento por separado y desde Eclipse directamente, desactive la casilla *Permitir integración...* (ver instrucciones más abajo).

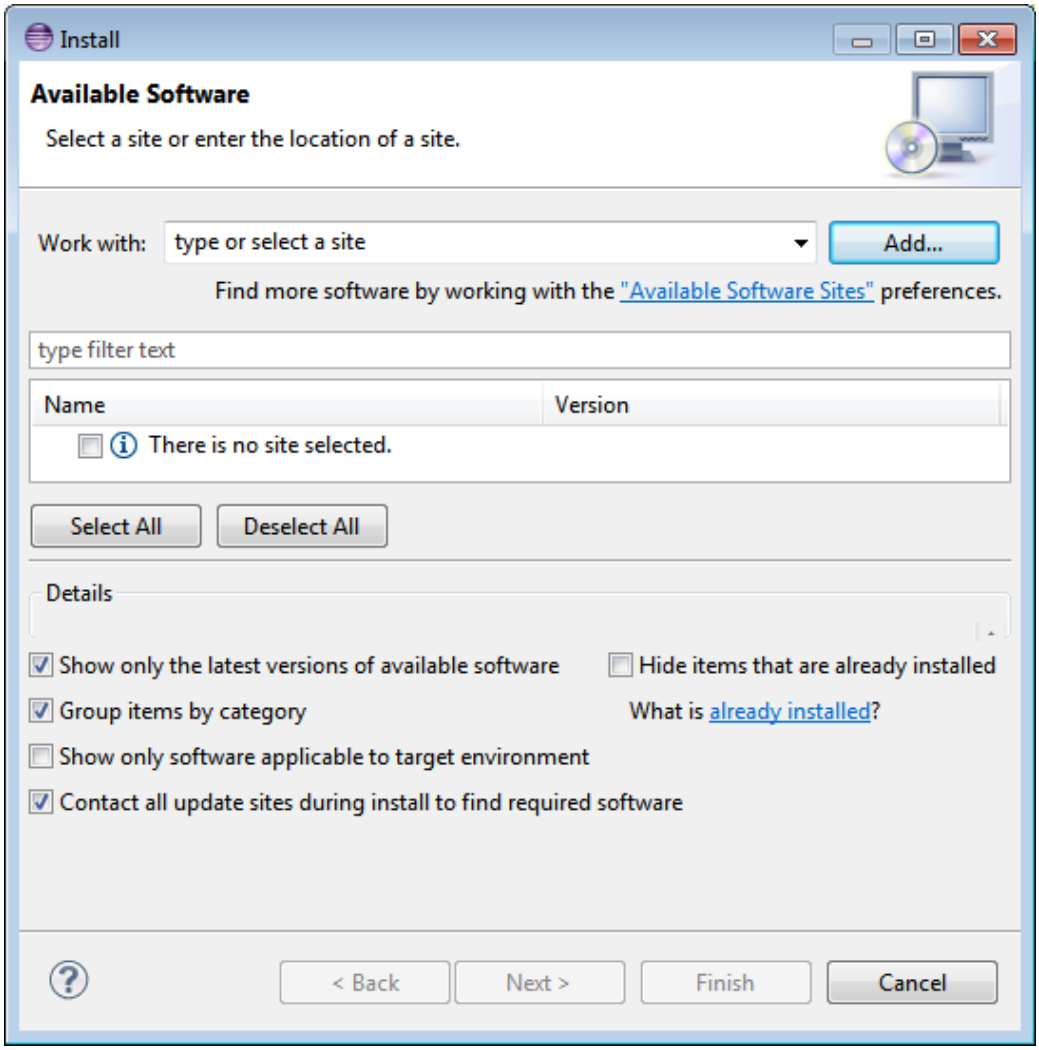
© 2018 Altova GmbH



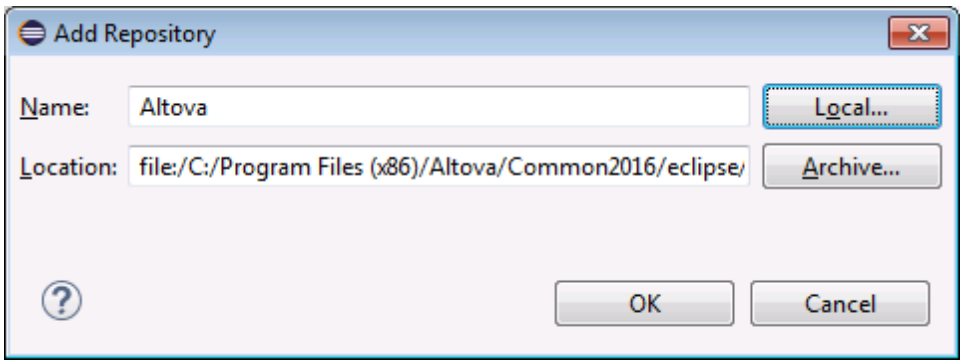
2. Haga clic en **Siguiente** y termine la instalación. Si eligió la integración automática, la perspectiva MapForce y sus menús se habilitarán en cuanto inicie Eclipse.

Integrar el complemento de MapForce para Eclipse a mano

1. En Eclipse haga clic en el comando **Help | Install New Software** (Ayuda | Instalar software nuevo).
2. En el cuadro de diálogo que aparece (*imagen siguiente*) haga clic en el botón **Add** (Agregar).



3. En el cuadro de diálogo que aparece, llamado *Add Repository* (Agregar repositorio, *imagen siguiente*) haga clic en el botón **Local**.
4. Busque la carpeta `c:\Archivos de programa\Altova\Common2018\eclipse\UpdateSite` y selecciónela. Escriba un nombre para el sitio (p. ej. Altova) y haga clic en **OK** para aceptar.

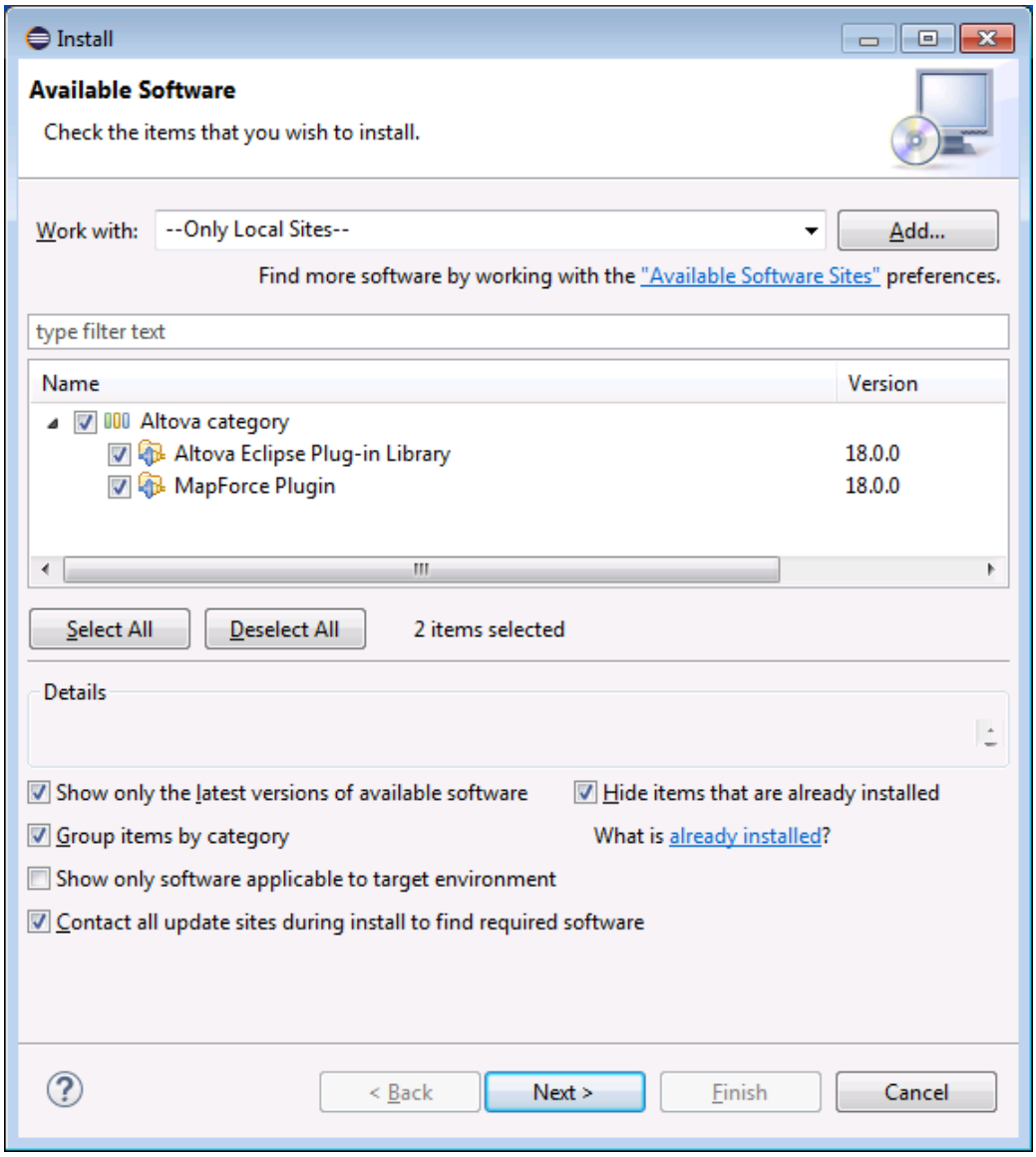


5. Repita los pasos 2, 3 y 4 pero esta vez seleccione la carpeta `c:\Archivos de programa\Altova\MapForce2018\eclipse\UpdateSite` y escriba un nombre como

 © 2018 Altova GmbH

"Altova MapForce".

6. En el cuadro combinado *Work With* (Trabajar con) del cuadro de diálogo "*Install*" seleccione la opción *All Available Sites* (Todos los sitios disponibles, *imagen siguiente*). Como resultado aparecen todos los complementos disponibles en el panel inferior. Marque la casilla de nivel superior de la carpeta *Altova category* (*imagen siguiente*). Después haga clic en ***Next*** para continuar.

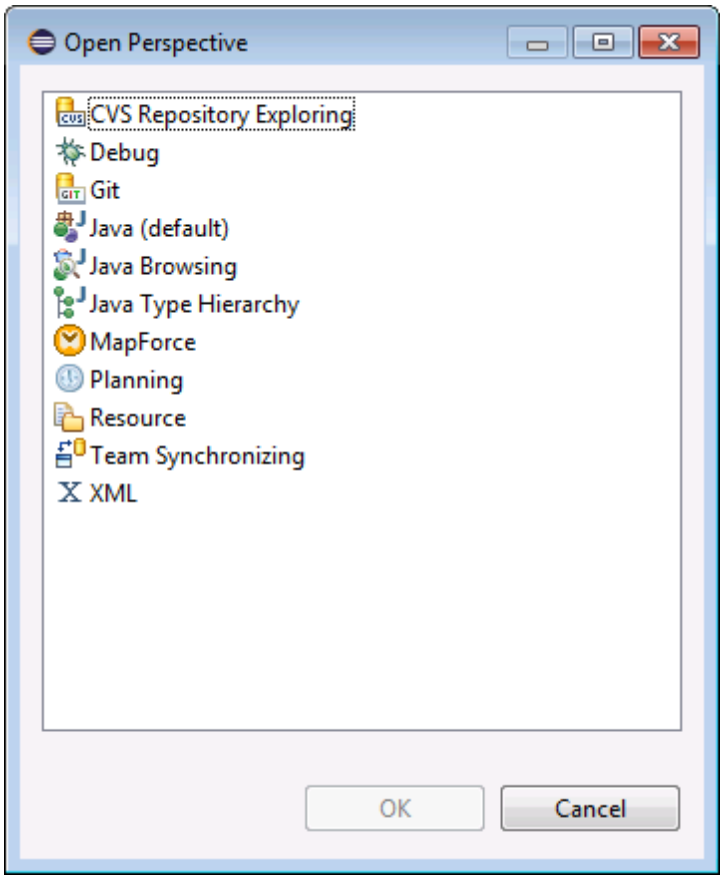


7. Aparece una pantalla llamada "*Install Details*" (detalles de la instalación) en la que puede repasar los elementos que se van a instalar. Haga clic en **Next** para continuar.
8. Aparece la pantalla "*Review Licenses*" (Repasar licencias). Seleccione la opción *I accept the terms of the license agreement* (Acepto el contrato de licencia). Para el complemento de MapForce no es necesario otro contrato de licencia (basta con la clave de licencia de MapForce Enterprise o Professional Edition). Ahora haga clic en el botón **Finish** para finalizar.

Nota: si tiene problemas con el complemento (por ejemplo, si faltan iconos), inicie Eclipse desde la línea de comandos con la marca `-clean`.

14.2 Perspectiva MapForce

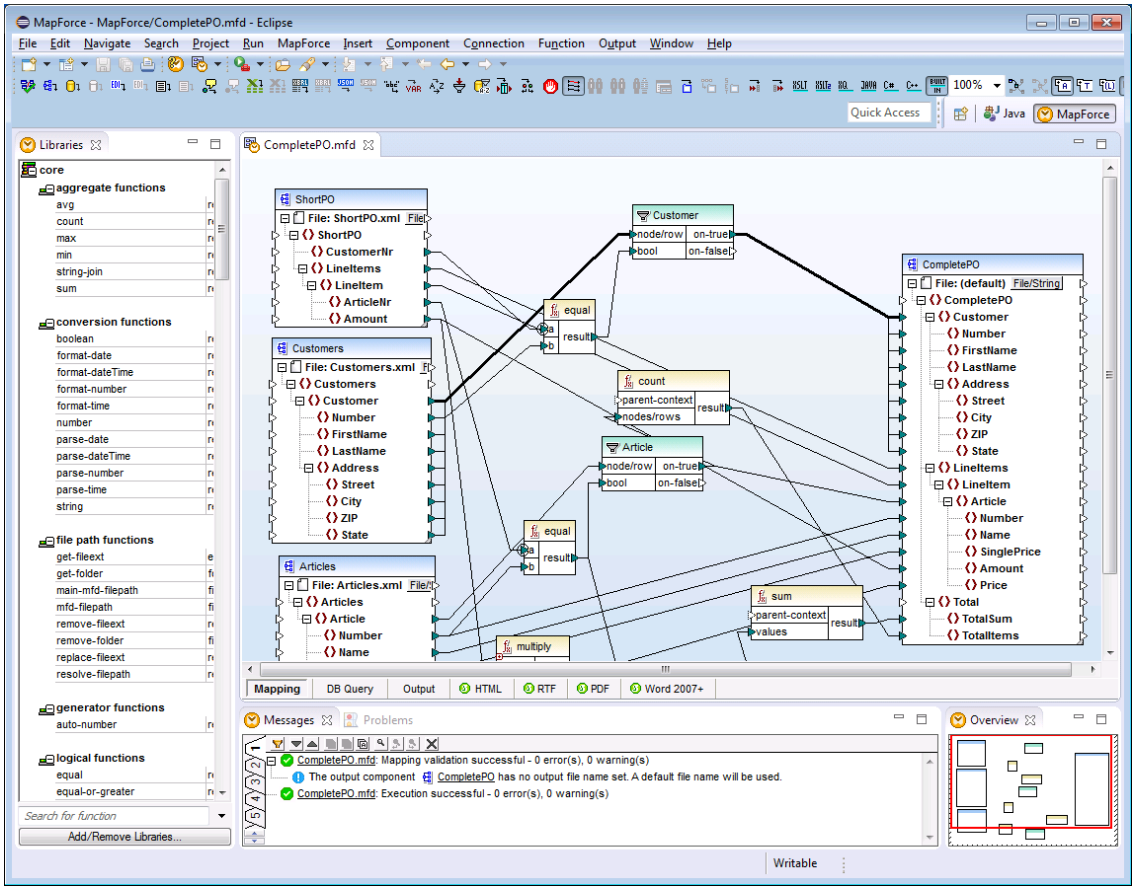
Una vez instalado el complemento de MapForce, la perspectiva MapForce estará disponible en Eclipse. El diseño de esta perspectiva es muy parecido al de la interfaz del usuario de MapForce. Para cambiar a la perspectiva MapForce haga clic en **Window | Open Perspective | Other** y elija **MapForce** en la lista.



Selección de la perspectiva MapForce en Eclipse

La perspectiva MapForce se puede usar como cualquier otra perspectiva de Eclipse. Se puede cambiar a esta perspectiva siempre que se quiera (**Window | Navigation | Next Perspective**). También puede personalizar las vistas que contiene y muchas otras opciones desde las preferencias de Eclipse. Para personalizar la perspectiva MapForce en Eclipse 4.4, cambie a la perspectiva MapForce y seleccione el comando de menú **Window | Customize Perspective**. Para más información consulte la documentación de Eclipse. En la siguiente imagen puede ver la perspectiva MapForce en el entorno Eclipse.

© 2018 Altova GmbH



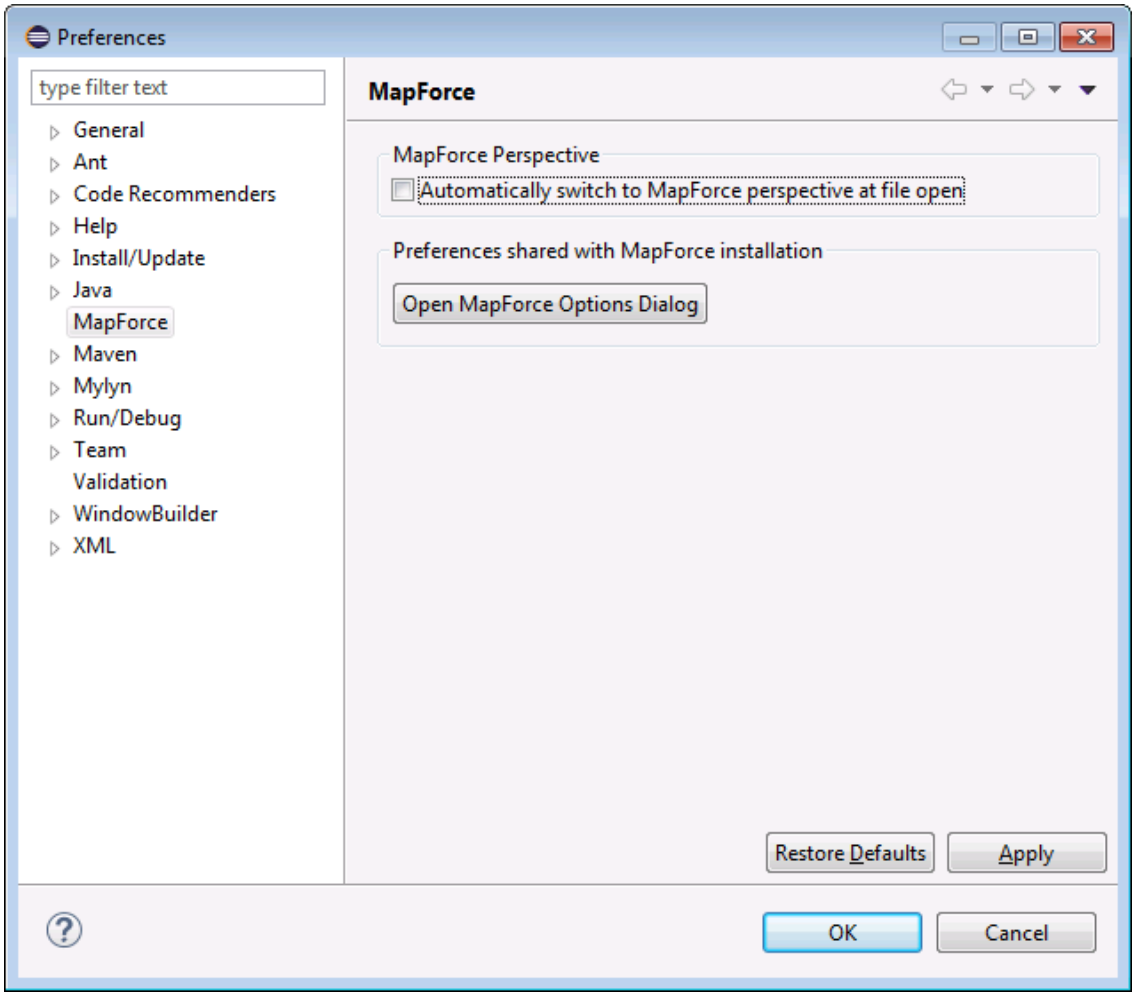
Perspectiva MapForce (complemento MapForce Enterprise Edition para Eclipse)

La perspectiva MapForce está formada por varios componentes predeterminados:

- La ventana de diseño de asignaciones, disponible en forma de editor de Eclipse. Tiene las mismas pestañas y características que la versión independiente de MapForce.
- La ventana Bibliotecas, disponible en forma de vista de Eclipse y situada a la izquierda del editor de asignaciones principal. Si no está visible, cambie a la perspectiva MapForce y seleccione el comando de menú **Window | Show View | Bibliotecas**. Esta vista permite trabajar con funciones predefinidas o personales y con las bibliotecas de funciones.
- El panel Mensajes, disponible en forma de vista de Eclipse y situado bajo el editor de asignaciones principal. Si no está visible, cambie a la perspectiva MapForce y seleccione el comando de menú **Window | Show View | Mensajes**. Esta vista muestra mensajes de validación, errores y advertencias.
- El panel Vista general, disponible en forma de vista de Eclipse. Si no está visible, cambie a la perspectiva MapForce y seleccione el comando de menú **Window | Show View | Vista general**. Esta vista permite navegar rápidamente hasta cualquier sección del diseño de asignación.

También puede configurar Eclipse para que cambie a la perspectiva MapForce automáticamente cuando se abra una asignación de MapForce: haga clic el comando de menú **Window | Preferences**, seleccione **MapForce** y después marque la casilla *Cambiar automáticamente a la*

perspectiva MapForce cuando se abran archivos.



Cuadro de diálogo de preferencias de Eclipse

© 2018 Altova GmbH

14.3 Acceso a menús y funciones frecuentes

En Eclipse puede acceder a la mayoría de funciones de MapForce desde los mismos menús que ofrece la versión independiente. Sin embargo, algunos menús y comandos son distintos en el complemento para Eclipse (ver más abajo). Sin embargo, recuerde que puede personalizar la perspectiva MapForce desde Eclipse (véase Perspectiva MapForce).

Nota: en Eclipse algunos grupos de menú o comandos de MapForce están deshabilitados (o no están disponibles) si el contexto no es relevante. Por ejemplo, el menú **Insertar** solamente está disponible si el archivo activo es un diseño de asignación (.mfd).

Para más información sobre los menús estándar de MapForce consulte la sección Referencia de menús.

Comandos generales

En la versión independiente de MapForce los comandos relacionados con los archivos de diseño de asignaciones (como **Validar**, **Implementar en FlowForce Server**, **Generar código**, etc.) están en el menú **Archivo**. En Eclipse, sin embargo, estos comandos están en el menú **MapForce** y en la barra de herramientas MapForce. Recuerde que los comandos para abrir y guardar archivos (incluidos los archivos de proyecto de MapForce) están en el menú **File** (Archivo) de Eclipse.



Barra de herramientas MapForce de Eclipse

El botón  de la barra de herramientas abre el archivo de ayuda de MapForce.

El botón  de la barra de herramientas incluye comandos relacionados con archivos de MapForce. Si expande este botón, los comandos de la lista dependerán del tipo de archivo que esté activo en el editor de Eclipse. Por ejemplo, los comandos relacionados con los archivos de diseño de asignaciones (.mfd) solo están disponibles cuando dicho tipo de archivo está activo en el editor de Eclipse.

Depuración de asignaciones

Los comandos de depuración de asignaciones están en el menú **MapForce | Depuración** y en la barra de herramientas Depuración. Tenga en cuenta que las teclas de acceso rápido para los comandos de depuración de asignaciones (como **F5**, **F10**, etc.) ya están reservados para otros comandos de Eclipse y no están disponibles.

Recursos globales

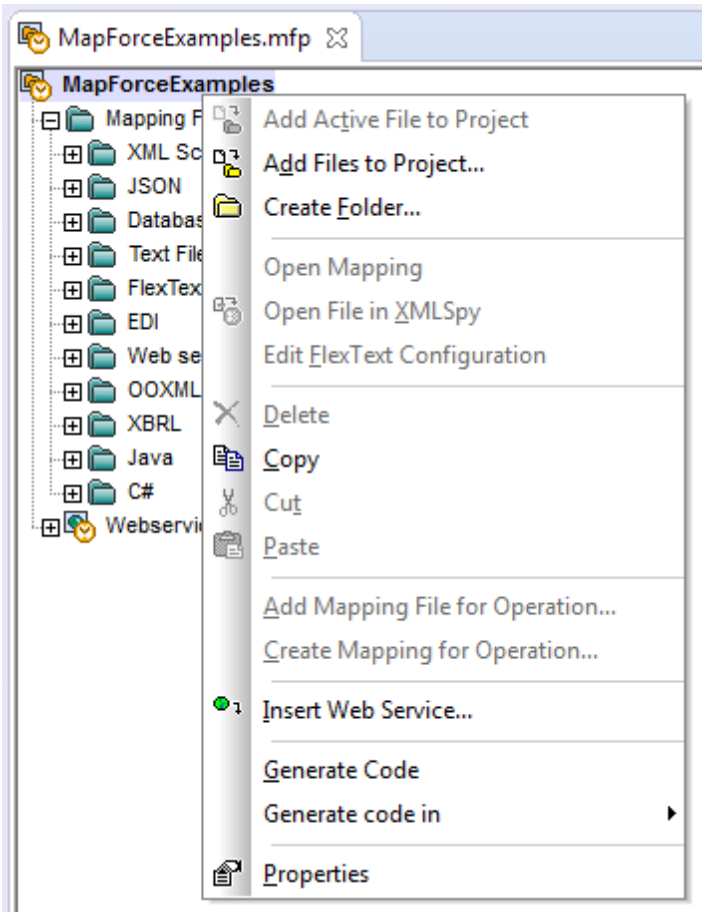
Hay dos maneras de acceder a los recursos globales y gestionarlos:

- Haga clic en el botón  de la barra de herramientas y elija **Recursos globales**.
- En el menú **MapForce** haga clic en **Recursos globales**.

Proyectos de MapForce

En la versión independiente de MapForce el menú **Proyecto** incluye varios comandos para trabajar con archivos de proyecto de asignación (.mfp). En Eclipse estos comandos están disponibles en otras posiciones:

- Los comandos para abrir o guardar proyectos están en el menú **File** (Archivo) de Eclipse.
- El resto de comandos relacionados con los proyectos están disponibles en menús contextuales. Para ver estos comandos primero debe crear o abrir un proyecto de MapForce (.mfp) en Eclipse y después hacer clic con el botón derecho en el proyecto.

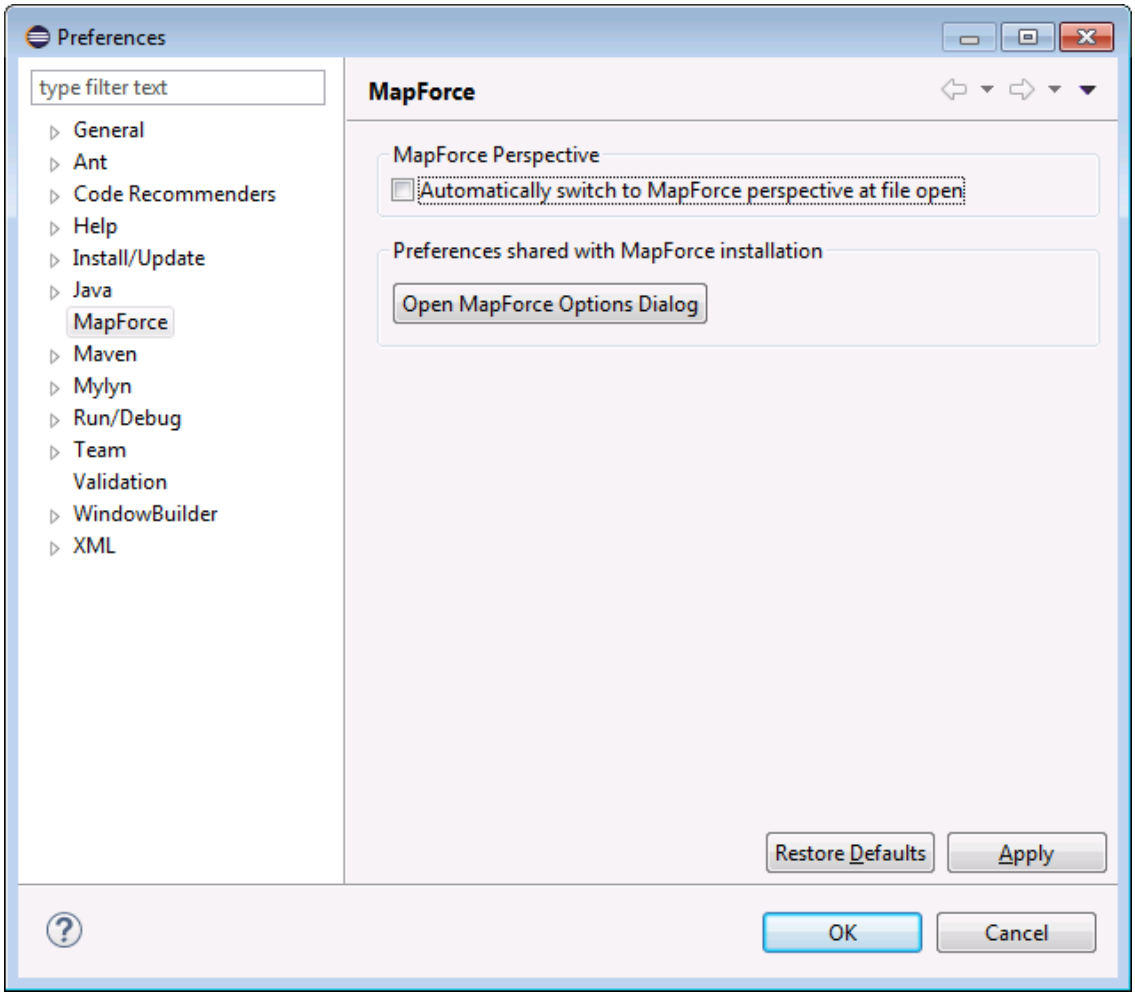


Tenga en cuenta que, además de proyectos estándar de MapForce (.mfp), en Eclipse también puede crear proyectos de tipo MapForce/Eclipse. Este tipo de proyectos son de carácter mixto y pueden configurarse para la generación automática de código de MapForce. Para más información consulte el apartado Trabajar con asignaciones y proyectos.

Opciones de MapForce

Las opciones de MapForce están disponibles en el menú **Window | Preferences** de Eclipse. En el cuadro de diálogo de preferencias ("*Preferences*") seleccione **MapForce** y después haga clic en **Abrir cuadro de diálogo "Opciones" de MapForce**.

© 2018 Altova GmbH



Cuadro de diálogo de preferencias

Ventana Bibliotecas

En Eclipse la ventana Bibliotecas de MapForce está disponible en forma de vista y está situada por defecto a la izquierda de la ventana principal del editor. Todas las vistas de MapForce están visibles en la interfaz de Eclipse cuando está activa la perspectiva MapForce.

Versión del complemento de MapForce

Para saber qué versión del complemento de MapForce tiene instalada haga clic en el comando de menú **Help | About Eclipse** de Eclipse y después seleccione el icono de MapForce.

Ayuda y soporte técnico

Los menús **Ayuda**, **Soporte técnico**, **Buscar actualizaciones** y **Acerca de** están en el menú **Help | Ayuda de MapForce** de Eclipse.

14.4 Trabajar con asignaciones y proyectos

Cuando está instalado el complemento de MapForce para Eclipse, podrá crear asignaciones y proyectos de asignación de datos desde un proyecto de Eclipse igual que en la versión independiente de MapForce. Para diseñar, probar, compilar e implementar asignaciones y para generar código de asignación de datos puede crear un proyecto de Eclipse nuevo o usar uno que ya exista (p. ej. un proyecto de Java al que desee añadir asignaciones de MapForce).

Además, podrá trabajar con todas sus asignaciones de datos dentro de un tipo de proyecto especial que está disponible cuando está instalado el complemento de MapForce: el proyecto MapForce/Eclipse. A no ser que prefiera personalizarlo, el proyecto MapForce/Eclipse se asigna por defecto a un generador Java Builder y a un generador de código MapForce. Además tiene dos naturalezas Eclipse: la naturaleza MapForce y la naturaleza JDT (Java Development Tools). Por tanto, se puede decir que, cuando se guardan o modifican sus recursos (p. ej. un archivo de asignación de datos), el proyecto MapForce/Eclipse se comporta de la siguiente manera:

- Si está habilitada la opción **Project > Build automatically**, el código de asignación de datos se genera automáticamente. Cuando en el proyecto MapForce/Eclipse existen archivos de proyecto de MapForce, el lenguaje de generación de código y las carpetas de destino de los resultados vienen determinados por la configuración de cada archivo de proyecto. De lo contrario, Eclipse solicita al usuario que elija una ubicación.
- Los errores y mensajes aparecen en las vistas Mensajes y Problemas.

Esta sección incluye varios apartados:

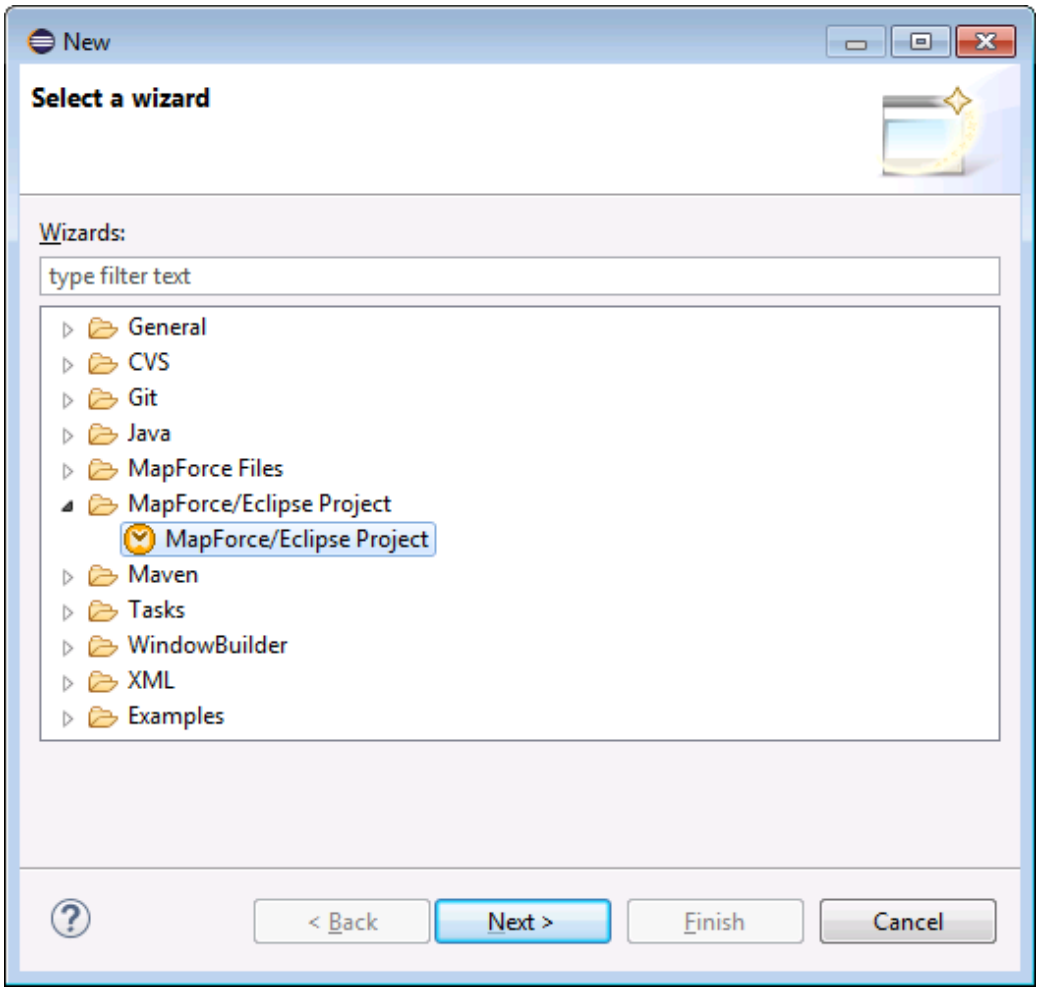
- Crear un proyecto MapForce/Eclipse
- Crear asignaciones nuevas
- Importar asignaciones a un proyecto de Eclipse
- Configurar la generación automática de código MapForce

14.4.1 Crear un proyecto MapForce/Eclipse

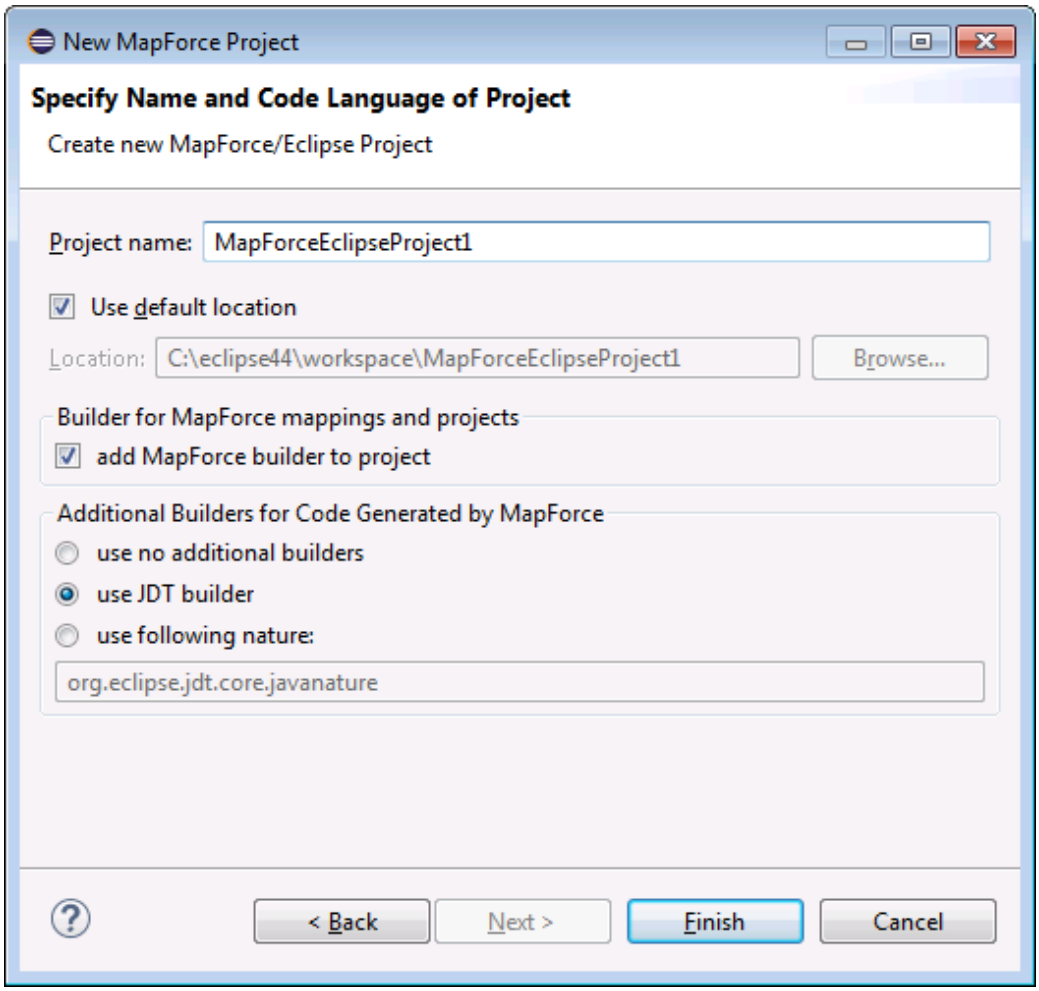
Para crear un proyecto MapForce/Eclipse:

1. En el menú **File** haga clic en **New | Other**.
2. Seleccione la categoría **MapForce/Eclipse Project**.

© 2018 Altova GmbH



3. Haga clic en **Next** para continuar.



4. Introduzca el nombre del proyecto y elija la ubicación donde se debe guardar. Deje las opciones *add MapForce builder to project* y *use JDT builder* como están.
5. Haga clic en **Finish** para finalizar.

14.4.2 Crear asignaciones nuevas

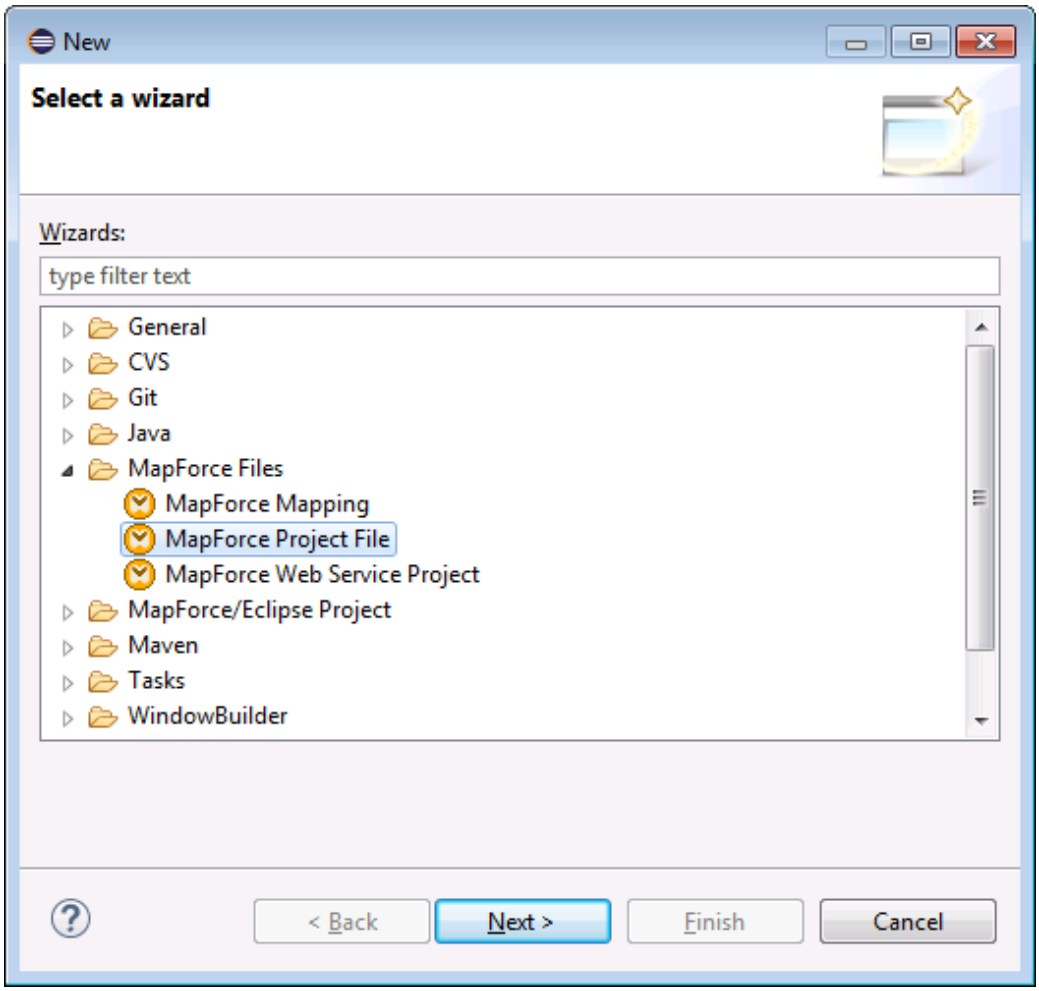
Dentro de un proyecto de Eclipse puede crear estos tipos de archivos de MapForce:

- Asignaciones de datos de MapForce
- Archivos de proyecto de MapForce
- Proyectos de servicio web de MapForce (solo en la edición MapForce Enterprise)

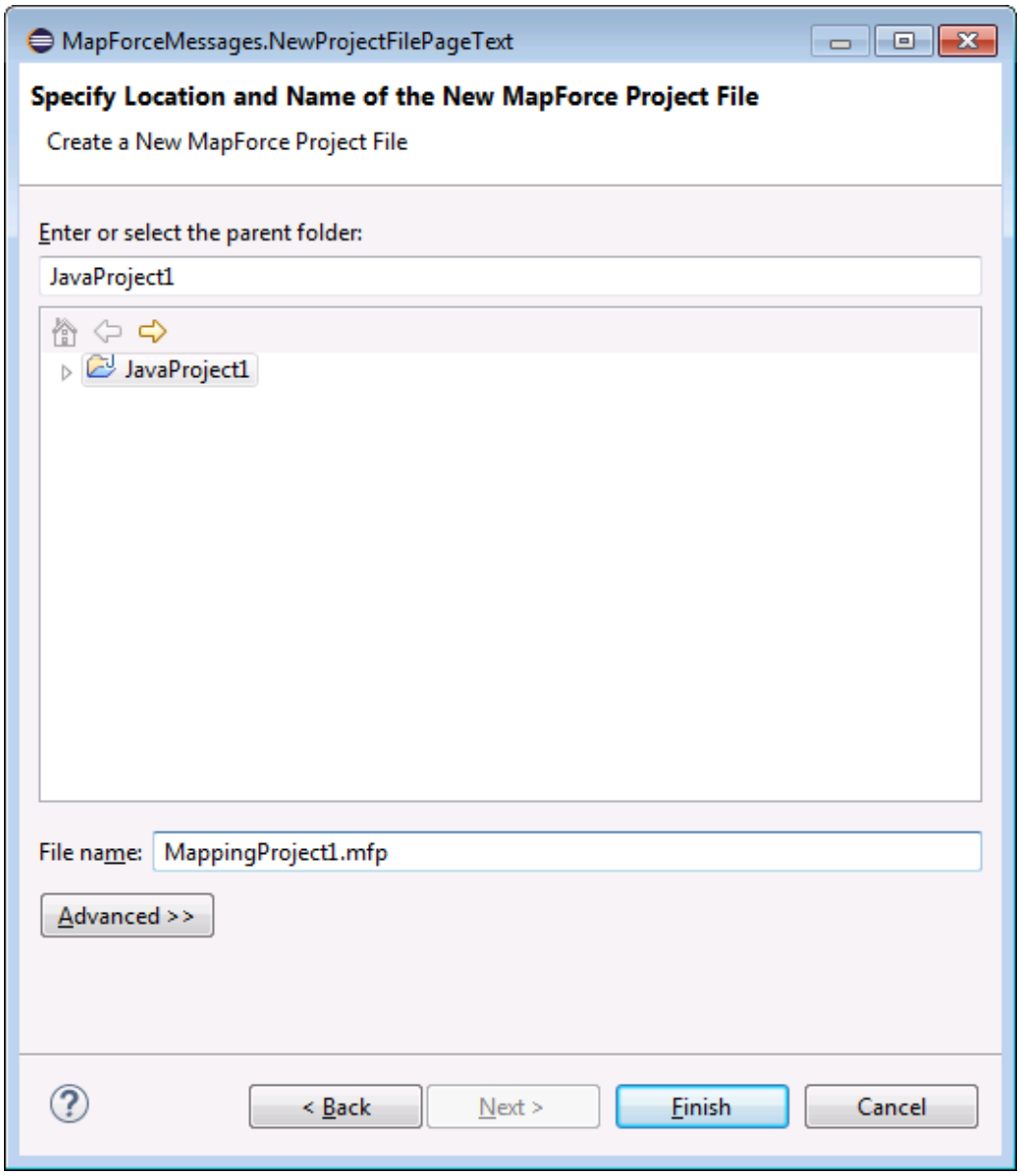
Para crear este tipo de archivos dentro de un proyecto de Eclipse:

1. Cree un proyecto de Eclipse nuevo o abra un proyecto actual.
2. En el menú **File** haga clic en **New | Other**.

© 2018 Altova GmbH



3. Seleccione el tipo de archivo que desea crear en el cuadro de diálogo y haga clic en **Next** para continuar.



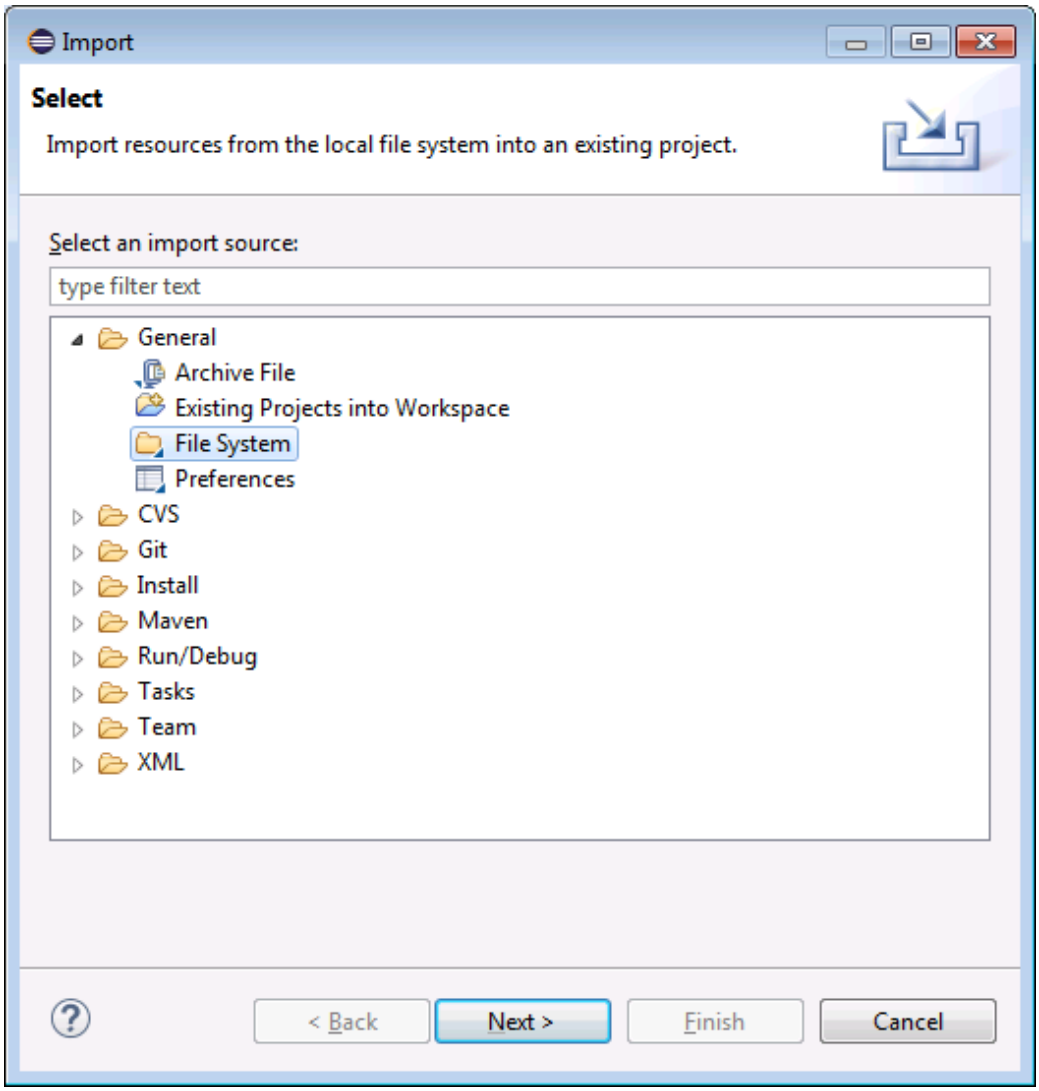
4. Seleccione una carpeta raíz del proyecto actual y haga clic en **Finish** para finalizar.

14.4.3 Importar asignaciones a un proyecto de Eclipse

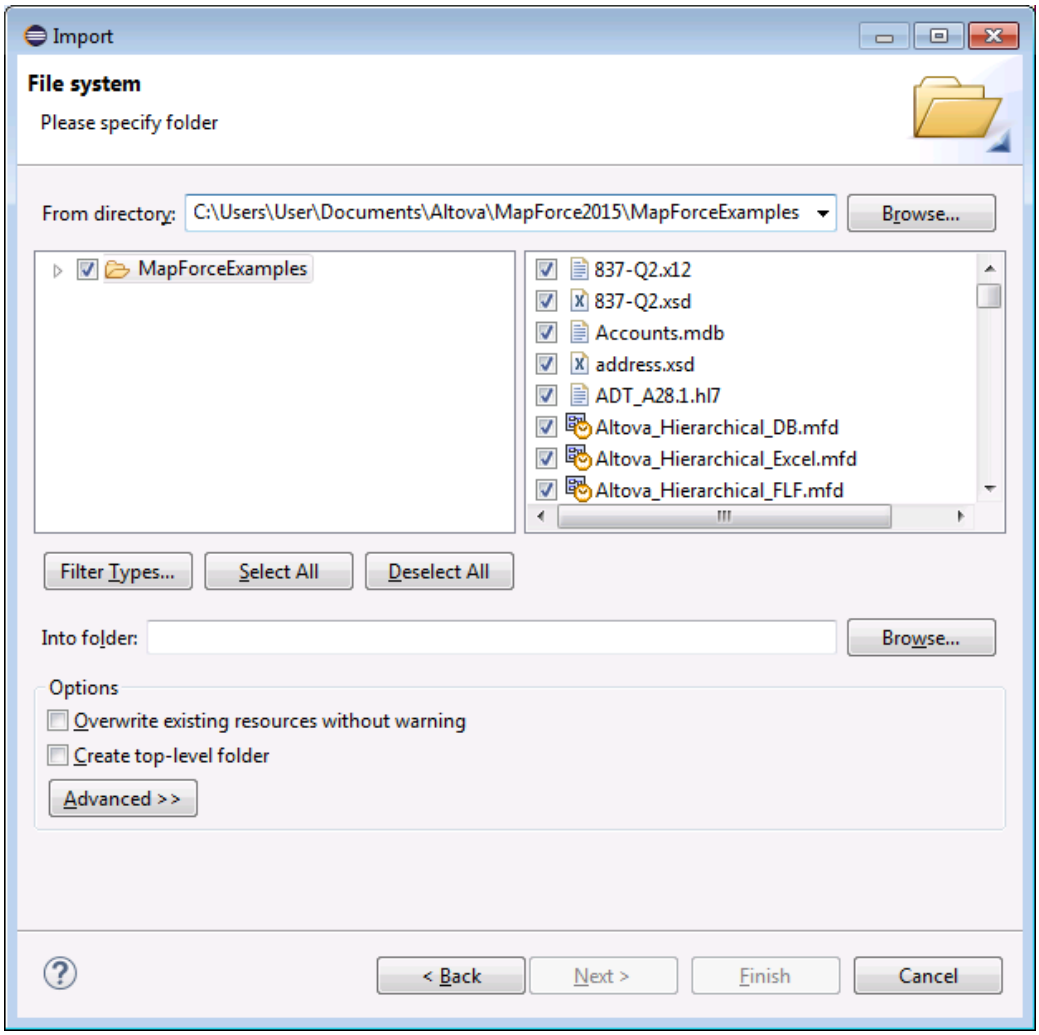
Para importar asignaciones de datos de MapForce y sus archivos dependientes en un proyecto de Eclipse:

1. Abra el proyecto en el que desea importar archivos.
2. En el menú **File** haga clic en **Import**.

© 2018 Altova GmbH

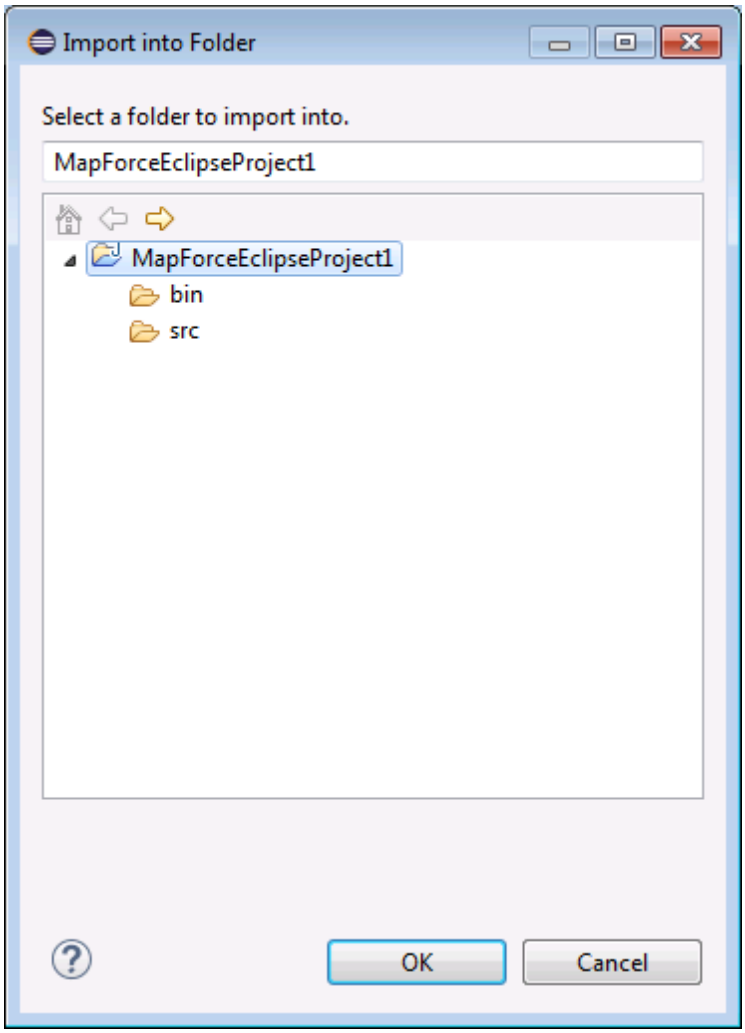


3. Seleccione **File System** y después haga clic en **Next** para continuar.



4. En el campo **From directory** navegue hasta la ubicación de los archivos que desea importar y seleccione los archivos necesarios.
5. En el campo **Into folder** haga clic en el botón **Browse** y seleccione el proyecto donde desea agregar los archivos (p. ej. *MapForceEclipseProyecto1*).

© 2018 Altova GmbH



6. Haga clic en **OK** para aceptar y después en **Finish** para finalizar.

14.4.4 Configurar la generación automática de código MapForce

La generación automática de código de MapForce está habilitada por defecto en todos los proyectos MapForce/Eclipse (*véase* *Crear un proyecto MapForce/Eclipse*). Si desea habilitar la generación automática de código de MapForce en un proyecto actual que no sea de tipo *MapForce/Eclipse*, deberá hacerlo de forma manual, añadiendo el proyecto al generador de código *Generación de código de MapForce MapForce* y a la naturaleza *MapForce*.

Para agregar el generador *Generación de código de MapForce* a un proyecto:

- Añada al archivo **.project** de Eclipse las líneas de código que aparecen resaltadas en amarillo a continuación:

```
<buildSpec>
    <buildCommand>
```

```
        <name>org.eclipse.jdt.core.javabuilder</name>
        <arguments>
        </arguments>
      </buildCommand>
      <buildCommand>
        <name>com.altova.mapforceeclipseplugin.MapForceBuilder</name>
        <arguments>
        </arguments>
      </buildCommand>
   </buildSpec>
```

Para agregar la naturaleza *MapForce* a un proyecto:

- Añada al archivo **.project** de Eclipse las líneas de código que aparecen resaltadas en amarillo a continuación:

```
   <natures>
      <nature>org.eclipse.jdt.core.javanature</nature>
      <nature>com.altova.mapforceeclipseplugin.MapForceNature</nature>

   </natures>
```

Consejo: puede abrir el archivo **.project** rápidamente desde la vista de navegación *Navigator* de Eclipse (para habilitar esta vista haga clic en el comando de menú **Window | Show View | Navigator**).

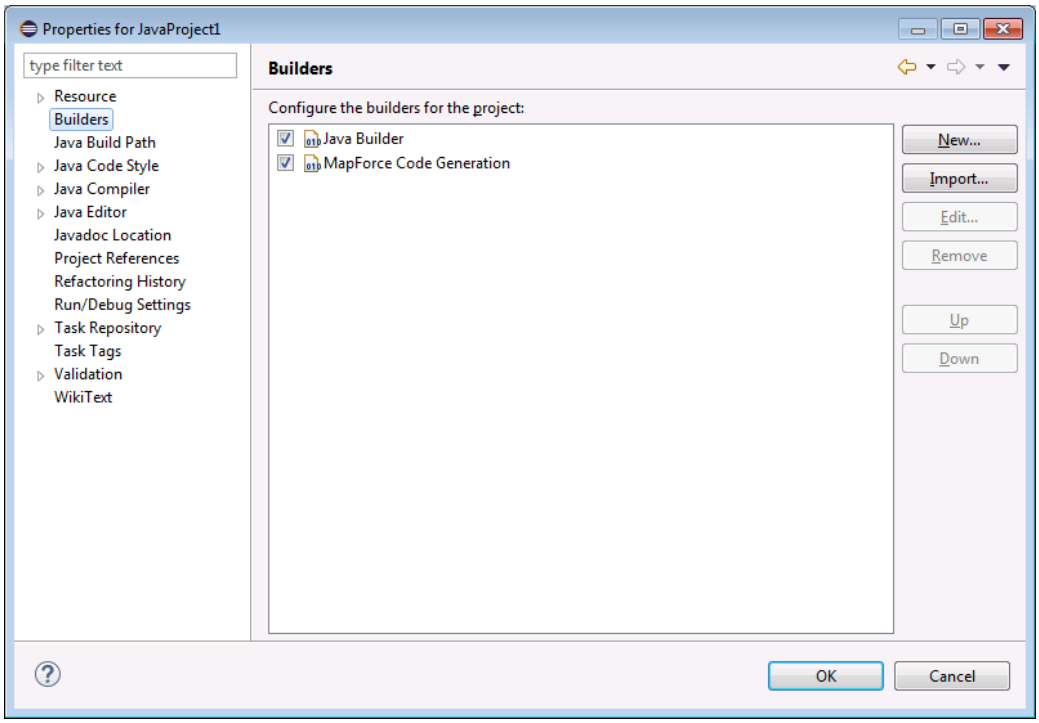
Para activar / desactivar la generación automática de código de MapForce:

- En el menú **Project** haga clic en el comando **Build automatically**.

Para deshabilitar el generador de código de MapForce:

1. En el menú **Project** haga clic en **Properties**.
2. Después haga clic en **Builders**.

© 2018 Altova GmbH



3. Desactive la casilla *Generación de código de MapForce*.

14.5 Ampliar el complemento de MapForce para Eclipse

El complemento de MapForce para Eclipse ofrece un punto de extensión Eclipse cuyo identificador es **com.altova.mapforceeclipseplugin.MapForceAPI**. Puede usar este punto de extensión para adaptar o ampliar las funciones del complemento de MapForce. El punto de extensión ofrece acceso a la interfaz de COM del control MapForce y de la API de MapForce.

El paquete de instalación de MapForce para Eclipse contiene un ejemplo de complemento que utiliza este punto de extensión y busca eventos de archivo abierto en todas las asignaciones nuevas de MapForce. Además establece el nivel de zoom de la vista en un 70%.

La documentación de JavaDoc para el punto de extensión está en el directorio de instalación del complemento de MapForce (que suele ser **C:\Archivos de programa\Altova\MapForce2018 \eclipse\docs**).

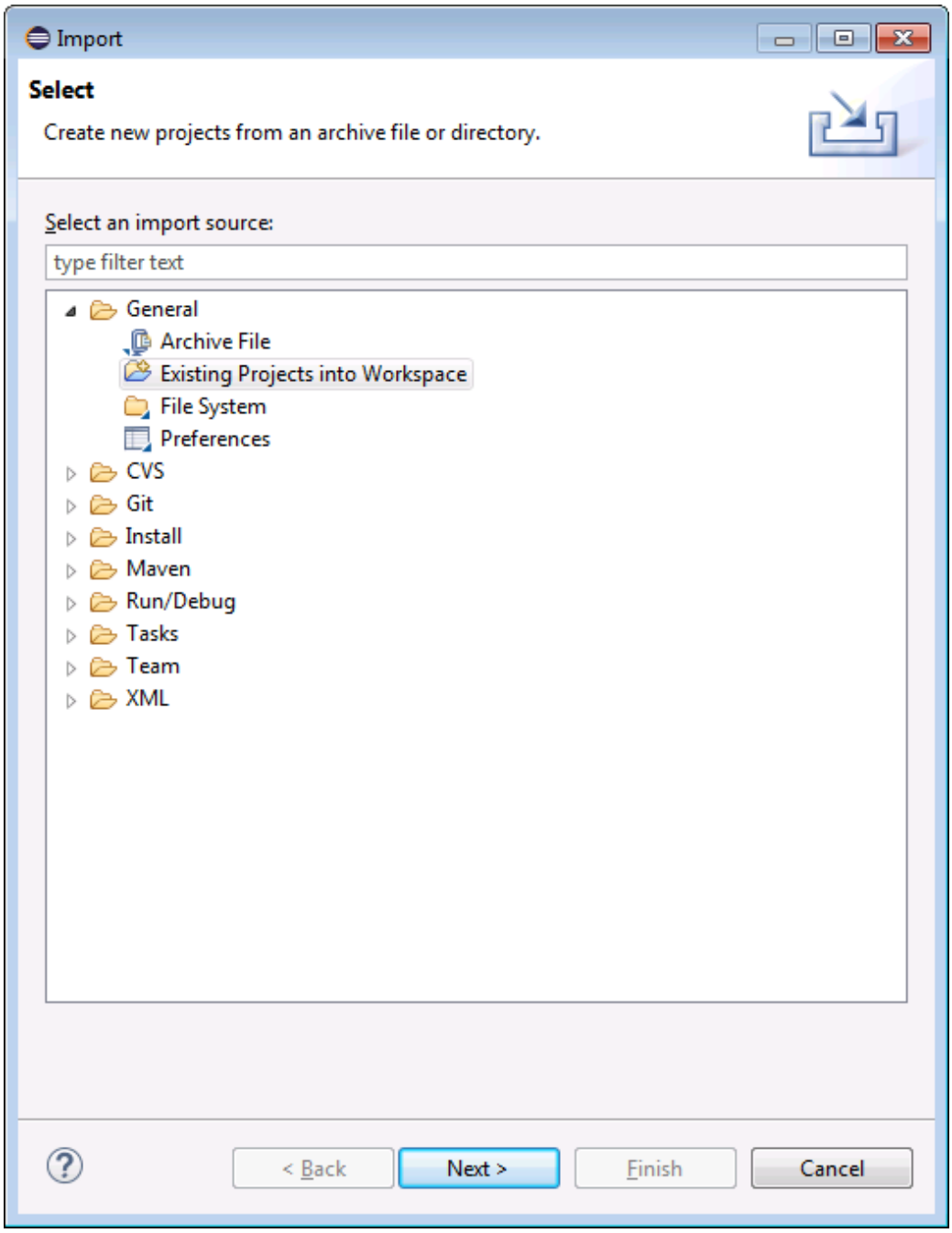
Antes de instalar y ejecutar el complemento de MapForce de ejemplo, compruebe que cumple con estos requisitos:

- Utiliza Java (32 bits), Eclipse (32 bits), MapForce (32 bits) y el paquete de integración de MapForce (32 bits).
- Tiene instalado el complemento JDT (Java Development Tools).
- Tiene instalado el entorno de desarrollo de complementos PDE de Eclipse.

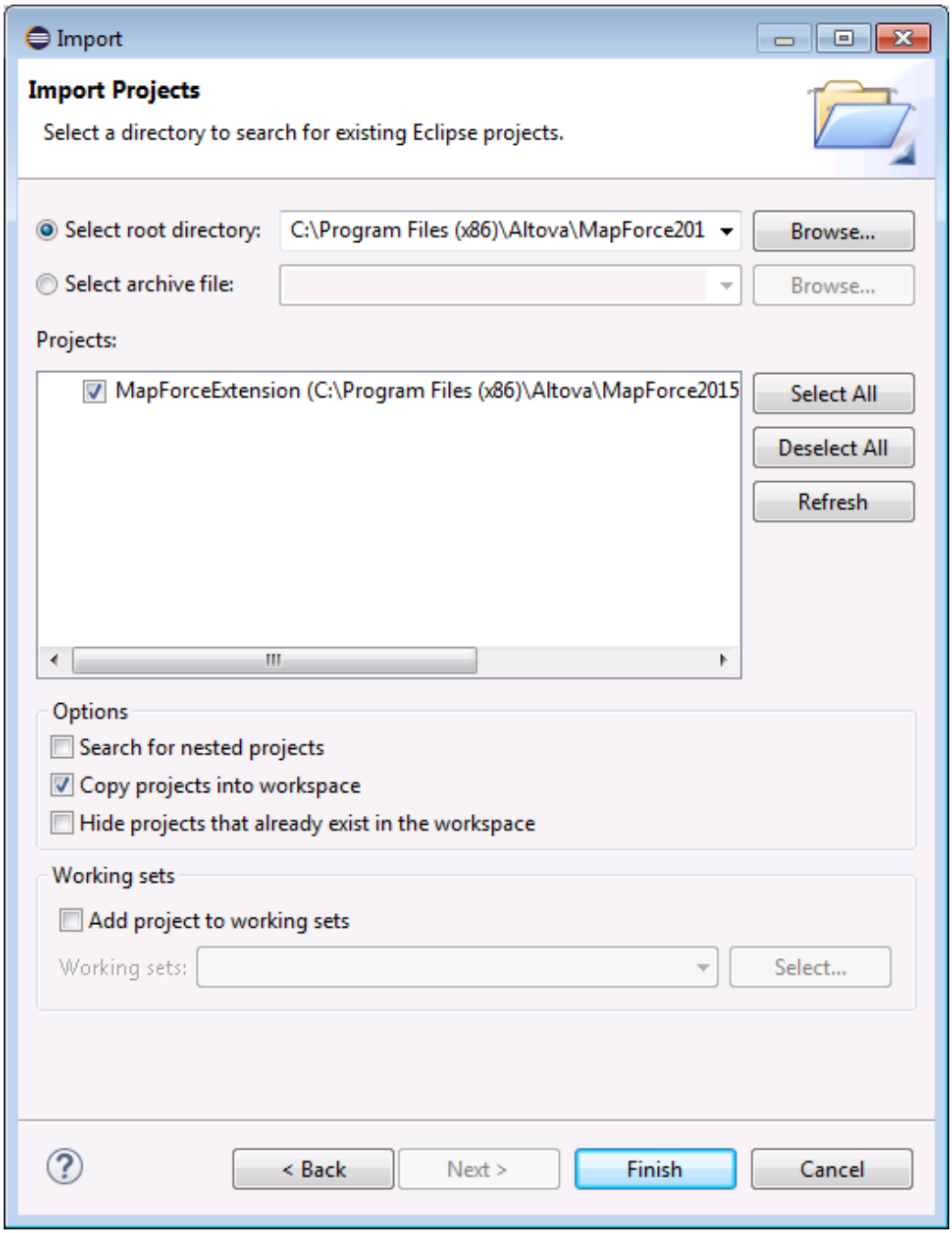
Para importar el proyecto de complemento MapForce de ejemplo al espacio de trabajo:

1. Inicie Eclipse.
2. En el menú **File** haga clic en **Import**.
3. Seleccione la opción **General | Existing projects into Workspace** y haga clic en **Next** para continuar.

© 2018 Altova GmbH



4. Haga clic en el botón **Browse...** situado junto al campo de selección del directorio raíz (*Select root directory*) y elija el directorio del proyecto de ejemplo (p. ej. **C:\Archivos de programa\Altova\MapForce2018\eclipse\workspace\MapForceExtension**).



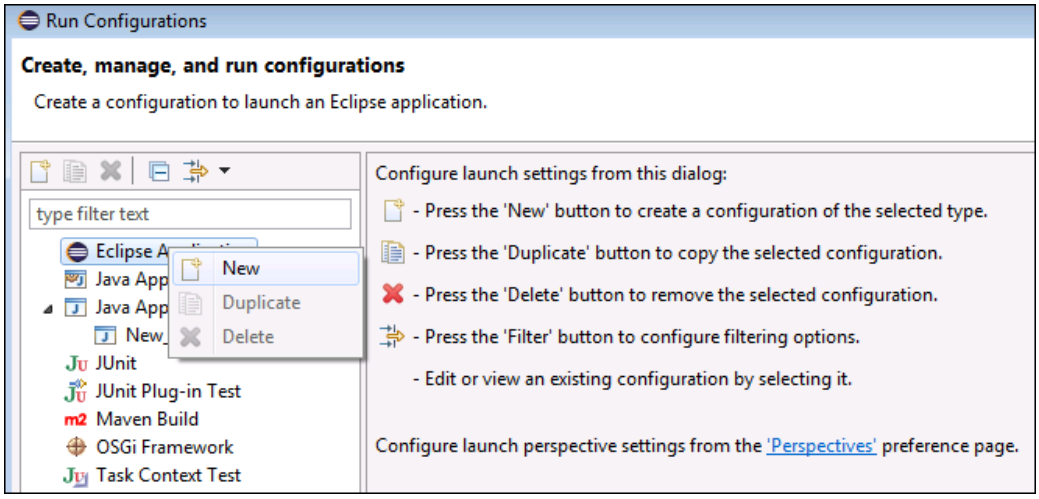
5. Ahora marque la casilla *Copy projects into workspace* (copiar proyectos en el espacio de trabajo) y haga clic en **Finish** para finalizar. En el espacio de trabajo se crea un proyecto nuevo llamado "MapForceExtension".

Para ejecutar el complemento de extensión de ejemplo:

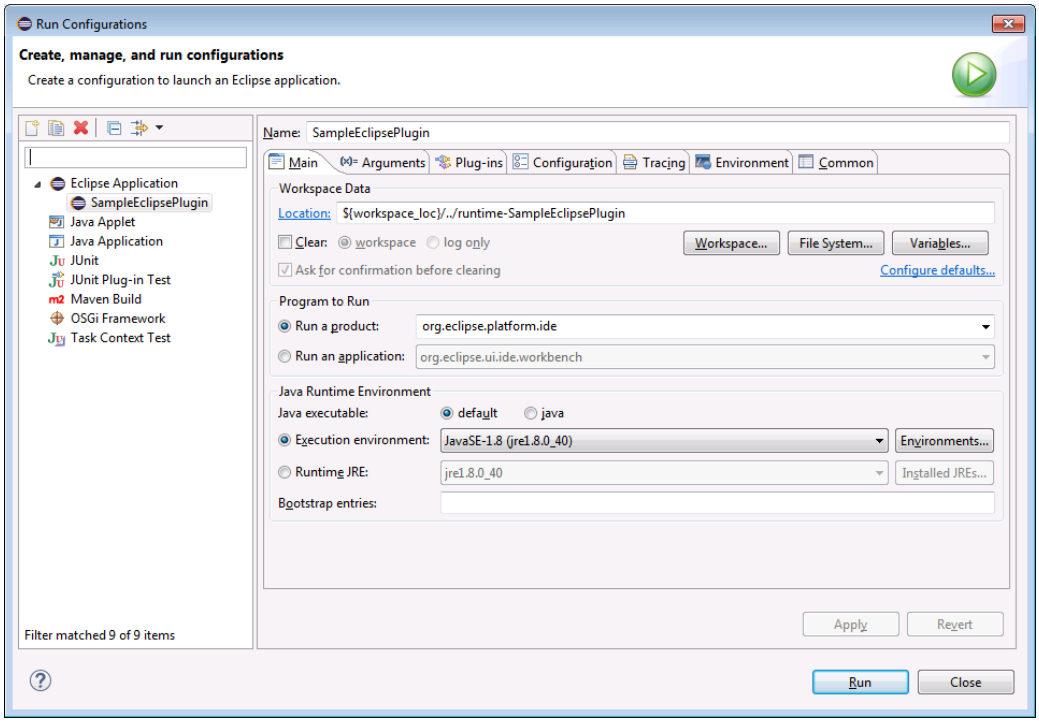
1. Cambie a la perspectiva Java.
2. En el menú de ejecución **Run** haga clic en el comando **Run Configurations**.
3. Haga clic con el botón derecho en **Eclipse Application** y seleccione **New**. (Si no puede

© 2018 Altova GmbH

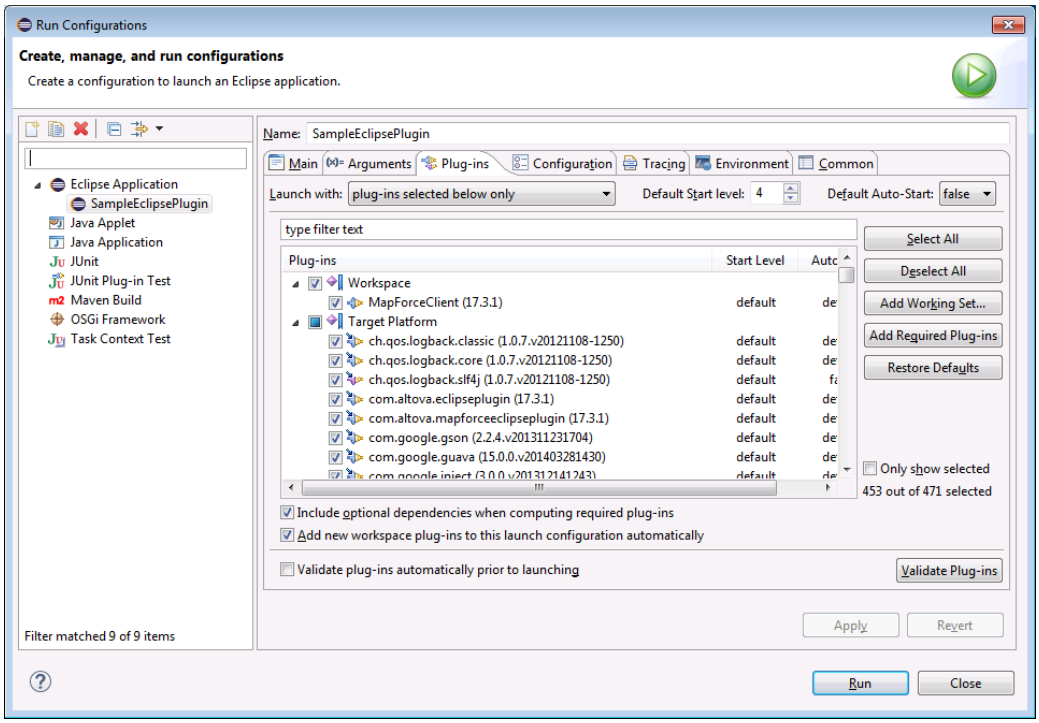
ver la opción **Eclipse Application** en la lista, significa que el entorno de desarrollo de complementos PDE de Eclipse no está instalado. Para instalar PDE haga clic en el comando **Install New Software** del menú de ayuda **Help** e instale **Eclipse Plugin Development Tools** desde el sitio de descargas "*The Eclipse Project Updates*".)



4. Introduzca un nombre nuevo para la nueva configuración (p. ej. *SampleMapForcePlugin*) y haga clic en **Apply** para aplicar la configuración.



5. Compruebe que el complemento del espacio de trabajo **MapForceClient** está seleccionado en la pestaña de complementos *Plug-ins*.



6. Haga clic en el botón **Run** para comenzar la ejecución. Esto abre un área de trabajo nuevo en Eclipse.
7. Abra cualquier asignación de MapForce en el nuevo área de trabajo y observe que se abre con un nivel de zoom del 70%.

 © 2018 Altova GmbH

Altova MapForce 2018 Enterprise Edition

Referencia de menús

15 Referencia de menús

Esta sección describe los comandos de menú de MapForce.

© 2018 Altova GmbH

15.1 Archivo

Nuevo
Crea un documento de asignación nuevo o un proyecto de asignación de datos (`*.mfp`).

Abrir
Abre archivos de asignación (*.mfd) o proyectos de asignación de datos (`*.mfp`). Tenga en cuenta que no se pueden abrir asignaciones con características de una edición superior de MapForce.

Guardar
Guarda la asignación activa usando el nombre de archivo activo en ese momento.

Guardar como
Guarda la asignación activa con otro nombre o permite al usuario indicar un nombre nuevo si es la primera vez que se guarda el archivo.

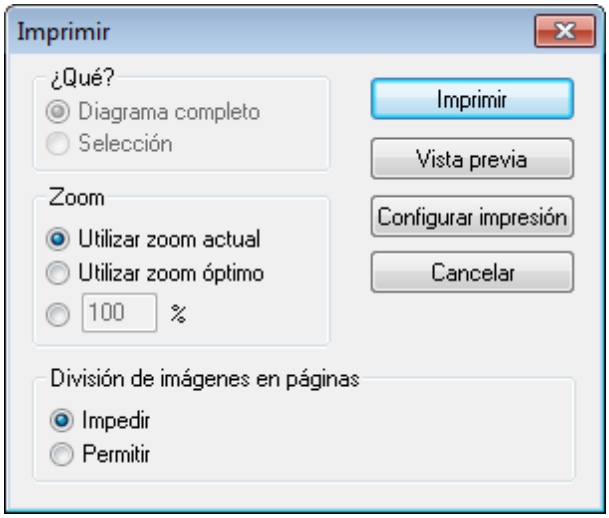
Guardar todos
Guarda todos los archivos que están abiertos.

Volver a cargar
Vuelve a cargar el archivo de asignación activo. Aparece un aviso preguntando si desea descartar los cambios realizados o guardarlos.

Cerrar
Cierra el archivo de asignación activo. Aparece un aviso preguntando si desea descartar los cambios realizados o guardarlos.

Cerrar todos
Cierra todos los archivos de asignación activos. Aparece un aviso preguntando si desea guardar los archivos que están sin guardar.

Imprimir
Abre el cuadro de diálogo "Imprimir".



Cuadro de diálogo "Imprimir"

La opción *Utilizar zoom actual* conserva el factor de zoom actual. La opción *Utilizar zoom óptimo* ajusta el tamaño de la asignación al tamaño de la página. También puede especificar el factor de zoom en porcentaje. Las barras de desplazamiento de los componentes de la asignación no se imprimen. También puede *permitir* o *impedir* la división de las imágenes en varias páginas.

Vista previa de impresión
Abre el cuadro de diálogo "Imprimir" (ver comando **Imprimir**).

Configurar impresión
Abre el cuadro de diálogo "Configurar impresión" donde puede definir la impresora y la configuración del papel.

Validar asignación
Comprueba si todas las asignaciones (conectores) son válidas y emite advertencias y errores de validación (*véase* *Validar asignaciones*).

Configurar asignación
Abre el cuadro de diálogo donde puede definir las opciones de configuración del documento activo (*véase* *Cambiar configuración de la asignación*).

Generar código en el lenguaje seleccionado
Genera código en el lenguaje seleccionado para la asignación activa. El lenguaje que está seleccionado está visible en la barra de herramientas: XSLT, XSLT 2, XQuery, Java, C# o C++.

Generar código en | XSLT (XSLT2)
Genera los archivos XSLT necesarios para la transformación de datos a partir de los archivos de origen. Al hacer clic en este comando se abre el cuadro de diálogo "Buscar carpeta" donde puede seleccionar la ubicación de los archivos XSLT. El nombre de los archivos XSLT que se generan se define en el cuadro de diálogo "Configurar asignación" (*véase* *Cambiar configuración de la asignación*).

© 2018 Altova GmbH

Generar código en | XQuery
Genera los archivos XQuery necesarios para la transformación de datos a partir de los archivos de origen. Al hacer clic en este comando se abre el cuadro de diálogo "Buscar carpeta" donde puede seleccionar la ubicación de los archivos XQuery. El nombre de los archivos XSLT que se generan se define en el cuadro de diálogo "Configurar asignación" (*véase Cambiar configuración de la asignación*).

Generar código en | Java | C# | C++
Estos comandos generan el código fuente necesario para un programa de aplicación completo para la transformación de datos. Al hacer clic en este comando se abre el cuadro de diálogo "Buscar carpeta" donde puede seleccionar la ubicación de los archivos generados. El nombre de los archivos de aplicación que se generan (así como el del archivo de proyecto C# `*.csproj`, del archivo de solución `*.sln` o del archivo de proyecto C++ `*.vcproj`) se define el cuadro de diálogo "Configurar asignación" (*véase Cambiar configuración de la asignación*).

El nombre de archivo creado por el código ejecutado es el que aparece en el campo *Archivo XML de salida* del cuadro de diálogo Configuración del componente si el destino es un archivo o esquema XML.

Compilar en archivo de ejecución de MapForce Server
Genera un archivo que se puede ejecutar directamente en MapForce Server para ejecutar la transformación de la asignación de datos (*véase Compilar una asignación de MapForce*).

Implementar en FlowForce Server
Implementa la asignación activa en FlowForce Server (*véase Implementar una asignación de MapForce*).

Generar documentación
Genera documentación para los proyectos de asignación en varios formatos de salida (*véase Generar y personalizar la documentación de la asignación*).

Archivos recientes
Una lista con archivos abiertos recientemente.

Salir
Cierra la aplicación. Antes aparece un aviso preguntando si desea guardar los archivos que estén sin guardar.

15.2 Edición

La mayoría de los comandos del menú **Edición** se habilitan cuando se abre el panel *Resultados* o cuando se consulta la vista previa de código XSLT en la pestaña *XSLT*.

Deshacer 

MapForce ofrece un número ilimitado de operaciones deshacer, que puede usar para retroceder en todos los cambios realizados.

Rehacer 

Este comando permite rehacer cambios anulados anteriormente con el comando **Deshacer**. Puede avanzar y retroceder por el historial de cambios usando estos dos comandos.

Buscar 

Permite buscar una cadena de texto en las pestañas *XSLT*, *XSLT2*, *XQuery* o *Resultados*.

Buscar siguiente (F3) 

Busca la siguiente instancia del término de búsqueda.

Buscar anterior (Mayús F3) 

Busca la instancia anterior del término de búsqueda.

Cortar / Copiar / Pegar / Eliminar

Estos comandos de edición estándar de Windows permiten cortar, copiar, etc. componentes y funciones de la ventana de asignación.

Seleccionar todo

Selecciona todos los componentes del panel *Asignación* o todo el texto de los paneles *XSLT*, *XSLT2*, *XQuery* o *Resultados*.

© 2018 Altova GmbH

15.3 Insertar

Archivo o esquema XML 

Inserta un archivo de esquema XML o de instancia XML a la asignación. Si selecciona un archivo XML que haga referencia a un esquema, la asignación no necesita más información. Si selecciona un archivo XML sin referencia a un esquema, deberá generar automáticamente el esquema XML correspondiente (*véase Generar un esquema XML*). Si selecciona un archivo de esquema XML, aparece un aviso preguntando si desea incluir un archivo XML de instancia que aporte los datos para la vista previa.

Base de datos 

Inserta una base de datos como componente de origen o destino (*véase Bases de datos en MapForce*).

EDI 

Inserta un documento EDI como componente de origen o destino (*véase la sección EDI*).

Archivo de texto 

Inserta un archivo plano (es decir, un archivo de texto CSV o de longitud fija) en la asignación de datos. Ambos tipos de archivo pueden usarse como componente de origen o de destino. Además, si desea procesar archivos de texto que tengan una estructura que no sea CSV o FLF, puede usar la aplicación MapForce FlexText.

Función de servicio web 

Inserta en la asignación una llamada a un servicio web (*véase Llamar a servicios web*).

Archivo Excel 2007+ 

Inserta un archivo Microsoft Excel 2007+ (Office Open XML) (`*.xlsx`) como componente de origen o destino (*véase Microsoft OOXML Excel 2007+*). Si no tiene Excel 2007+ instalado, esto no le impide crear asignaciones de datos entre archivos Excel 2007+. El único impedimento es que no podrá consultar la vista previa de resultados en el panel *Resultados.* Sí podrá guardar los resultados con el comando **Guardar el archivo de salida** del menú **Resultados**.

Documento XBRL 

Inserta un documento de instancia XBRL o una taxonomía XBRL en la asignación (*véase Agregar archivos XBRL como componentes de la asignación*).

Archivo o esquema JSON 

Inserta un archivo o un esquema JSON en la asignación (*véase Agregar archivos JSON como componentes de la asignación*).

Insertar componente de entrada 

Si en el área de asignación está visible una asignación de datos, este comando inserta un componente de entrada (*véase Pasar parámetros a la asignación*). Si en el área de asignación está visible una función definida por el usuario, este comando agrega un componente de entrada

en la función (*véase* *Definir componentes de entrada complejos*).

Insertar componente de salida 

Si en el área de asignación está visible una asignación de datos, este comando inserta un componente de salida (*véase* *Obtener valores de cadena de una asignación*). Si en el área de asignación está visible una función definida por el usuario, este comando agrega un componente de salida en la función (*véase* *Definir componentes de salida complejos*).

Constante 

Inserta una constante que aporta datos fijos a un conector de entrada. Los datos se introducen en un cuadro de diálogo que aparece cuando se empieza a crear la constante. Puede elegir entre estos tipos de datos: cadena, número, demás opciones.

Variable 

Inserta una variable intermedia que equivale a una función definida por el usuario no inline. Las variables son componentes estructurales, sin archivos de instancia, que sirven para simplificar el proceso de asignación (*véase* *Variables intermedias*).

Componente de ordenación: nodos/filas

Inserta un componente que permite ordenar nodos (*véase* *Ordenar datos*).

Filtro: nodos/filas 

Inserta un componente que usa dos parámetros de entrada (**`node/row`** y **`bool`**) y dos de salida (**`on-true`** y **`on-false`**). Si el valor booleano es `true`, el valor del parámetro **`node/row`** se pasa al parámetro **`on-true`**. Si el booleano es `false`, el valor del parámetro **`node/row`** se pasa al parámetro **`on-false`**. Consulte la sección Filtros y condiciones para obtener más información.

WHERE/ORDER de SQL 

Inserta un componente que filtrar datos de la base de datos de forma condicional (*véase* *Componente WHERE/ORDER de SQL*).

Asignación de valores 

Inserta un componente que transforma un valor de entrada en un valor de salida usando una tabla de búsqueda. Es un componente muy práctico si necesita asignar un conjunto de valores a otro conjunto de valores (p. ej. números de meses en nombres de meses). Consulte el apartado Usar asignaciones de valores para obtener más información.

Condición IF-Else 

Inserta un componente de tipo If-Else (véase Filtros y condiciones).

Excepción 

Este componente permite interrumpir el proceso de asignación cuando se cumple determinada condición o definir mensajes de error (fault) en proyectos de asignación WSDL. Consulte Agregar excepciones y Elementos de error fault en servicios web para obtener más información.

© 2018 Altova GmbH

15.4 Proyecto

MapForce integra la interfaz de múltiples documentos y permite al usuario agrupar las asignaciones de datos en proyectos de asignación (*véase* *Trabajar con proyectos de asignación*).

Volver a cargar el proyecto
Vuelve a cargar el proyecto activo y cambia a la ventana Proyecto.

Cerrar el proyecto
Cierra el proyecto activo.

Guardar el proyecto
Guarda el proyecto activo.

Agregar archivos al proyecto
Permite agregar asignaciones al proyecto actual desde el cuadro de diálogo "Abrir".

Agregar archivo activo al proyecto
Agrega el archivo activo al proyecto que está abierto.

Crear carpeta
Agrega una carpeta nueva a la estructura del proyecto activo. Este comando solamente está disponible cuando esta operación es posible (*véase* *Gestionar las carpetas de proyecto*).

Abrir asignación
Abre la asignación seleccionada/resaltada en la ventana Proyecto.

Crear asignación para la operación
Crea un archivo de asignación para la operación seleccionada del proyecto WSDL. El nombre de la operación definido en el archivo WSDL aparece en el cuadro de diálogo "Guardar como", que se abre automáticamente.

Agregar archivo de asignación para la operación...
Permite agregar un archivo de asignación ya existente a la operación WSDL activa. Seleccione el archivo de asignación en el cuadro de diálogo "Abrir".

Insertar servicio web...
Permite insertar un servicio web basado en un archivo WSDL ya existente.

Abrir archivo en XMLSpy
Abre el archivo WSDL seleccionado (el archivo que está resaltado en la ventana Proyecto) en Altova XMLSpy.

Generar código para todo el proyecto
Genera código de proyecto para todo el proyecto activo en la ventana Proyecto. El código se genera en el lenguaje seleccionado para todos los archivos de asignación `*.mfd` en cada una de las carpetas.

Generar código en...
Genera código de proyecto en el lenguaje seleccionado en el menú emergente.

Propiedades

Abre un cuadro de diálogo donde puede definir la configuración global del proyecto (*véase Configurar la generación de código*).

Proyectos recientes: 1. 2. etc.

Una lista con proyectos abiertos recientemente.

© 2018 Altova GmbH

15.5 Componente

Cambiar de elemento raíz
Permite cambiar el elemento raíz del documento de instancia XML.

Editar la definición del esquema en XMLSpy
Seleccione esta opción (después de hacer clic en el documento/esquema XML) para abrir el archivo de esquema XML en la vista Esquema de XMLSpy.

Editar la configuración de FlexText
Abre el módulo FlexText donde puede editar un archivo de FlexText ya existente.

Agregar, quitar o editar objetos de la base de datos
Permite agregar, eliminar o cambiar los objetos de la base de datos dentro del componente de base de datos.

Crear asignación a EDI X12 997
La confirmación funcional X12 997 informa del estado del intercambio EDI. Todos los errores encontrados durante el procesamiento del documento aparecen en esta confirmación. MapForce puede generar documentos X12 997 automáticamente en el panel *Asignación*. Para más información sobre los segmentos de error y lo que significan consulte la documentación de la confirmación X12 997.

Crear asignación a EDI X12 999
El conjunto de transacciones de la confirmación de implementación X12 999 notifica el incumplimiento de las normas de implementación HIPAA y errores de aplicación. A cada transacción EDI enviada a una organización debe contestarse enviando una transacción 999. Para más información consulte el apartado Generar una confirmación de implementación X12 999.

Actualizar
Vuelve a cargar la estructura del componente de BD activo desde la base de datos.

Agregar delante un duplicado de entrada
Inserta una copia (un clon) del elemento seleccionado antes del elemento seleccionado. Los elementos duplicados no tienen iconos de salida, es decir, no se pueden usar como origen de datos. Para ver un ejemplo consulte la sección Asignar varios orígenes de datos a un destino del Tutorial. Haga clic con el botón derecho en un duplicado para moverlo a una posición nueva (opciones **Subir/Bajar** del menú contextual que aparece).

Agregar detrás un duplicado de entrada
Inserta una copia (un clon) del elemento seleccionado después del elemento seleccionado. Los elementos duplicados no tienen iconos de salida, es decir, no se pueden usar como origen de datos. Para ver un ejemplo consulte la sección Asignar varios orígenes de datos a un destino del Tutorial. Haga clic con el botón derecho en un duplicado para moverlo a una posición nueva (opciones **Subir/Bajar** del menú contextual que aparece).

Quitar el duplicado
Quita el elemento duplicado seleccionado. Para ver un ejemplo consulte la sección Asignar varios orígenes de datos a un destino del Tutorial.

Acciones de tabla de la base de datos
Permite definir qué acciones de tabla se deben llevar a cabo con los datos asignados en cada tabla de la BD de destino. Consulte la sección Configurar acciones de tabla de la base de datos para más información.

Consultar la base de datos
Crea una instrucción SELECT basada en la tabla o en el campo activos en el componente de base de datos. Este comando se habilita al hacer clic una vez en la tabla o el campo y la instrucción SELECT se introduce automáticamente en la ventana SELECT.

Alinear el árbol a la izquierda
Coloca todos los elementos de un componente en el lado izquierdo de la ventana.

Alinear el árbol a la derecha
Coloca todos los elementos de un componente en el lado derecho de la ventana. Este diseño es práctico para crear asignaciones de datos cuyo destino es un esquema.

Propiedades
Abre un cuadro de diálogo con las opciones de configuración actuales del componente (*véase Cambiar configuración de los componentes*).

© 2018 Altova GmbH

15.6 Conexión

Conectar automáticamente los secundarios equivalentes
Activa o desactiva la opción **Conexión automática de secundarios** así como el icono de la barra de herramientas.

Configurar la conexión de secundarios equivalentes
Abre el cuadro de diálogo "Configurar la conexión de secundarios equivalentes" donde puede definir las opciones de conexión (*véase* *Conectar elementos secundarios equivalentes*).

Conectar los secundarios equivalentes
Este comando permite crear varios conectores para los elementos que tienen el **mismo nombre** en el esquema de origen y en el de destino. Aparece un cuadro de diálogo donde puede definir la conexión. Las opciones definidas en este cuadro de diálogo se conservan y se aplican cada vez que se conecten dos elementos siempre y cuando esté activo el icono **Conexión automática de secundarios**  de la barra de herramientas (*véase* *Conectar elementos secundarios equivalentes*).

Basada en el destino (estándar)
Cambia el tipo de conector a una asignación estándar (*véase* *Conexiones basadas en el destino*).

Copia total (copia los elementos secundarios)
Crea conectores para todos los elementos secundarios equivalentes, donde cada uno de los conectores secundarios aparece como una rama del conector primario (*véase* *Conexiones de copia total*).

Basada en origen (contenido mixto)
Cambia el tipo de conector a una asignación basada en origen / de contenido mixto (*véase* *Conexiones basadas en el origen*).

Propiedades
Abre un cuadro de diálogo donde puede definir la configuración específica (contenido mixto) del conector actual. Las opciones que no estén disponibles en ese momento aparecen atenuadas. No olvide que estas opciones de configuración también afectan a los elementos `complexType`, que no tienen nodos de texto (*véase* *Configuración de las conexiones*).

15.7 Función

Crear una función definida por el usuario
Crea una función nueva definida por el usuario (*véase* *Funciones definidas por el usuario*).

Crear función definida por el usuario a partir de la selección
Crea una función nueva definida por el usuario basada en los elementos que están seleccionados en el área de asignación.

Configuración de la función
Abre el cuadro de diálogo de configuración de la función definida por el usuario que está activa para poder cambiar su configuración.

Quitar función
Elimina la función definida por el usuario que está activa (siempre y cuando el contexto lo permita).

Insertar componente de entrada
Si en el área de asignación está visible una asignación de datos, este comando inserta un componente de entrada (*véase* *Entradas simples*). Si en el área de asignación está visible una función definida por el usuario, este comando agrega un componente de entrada en la función (*véase* *Definir componentes de entrada complejos*).

Insertar componente de salida
Si en el área de asignación está visible una asignación de datos, este comando inserta un componente de salida (*véase* *Salidas simples*). Si en el área de asignación está visible una función definida por el usuario, este comando agrega un componente de salida en la función (*véase* *Definir componentes de salida complejos*).

© 2018 Altova GmbH

15.8 Resultados

XSLT 1.0, XSLT 2.0, XQuery, Java, C#, C++, Motor de ejecución integrado
Establece el lenguaje de transformación en el que se debe ejecutar la asignación de datos (*véase Seleccionar un lenguaje de transformación*).

Validar archivo de salida
Valida el archivo XML de salida con el esquema al que se hace referencia (*véase Validar el resultado de la asignación*).

Guardar el archivo de salida
Guarda en un archivo los datos visibles en el panel *Resultados*.

Guardar todos los archivos de salida
Guarda todos los archivos de salida generados de las asignaciones dinámicas (*véase Procesar varios archivos de entrada o salida de forma dinámica*).

Volver a generar archivo de salida
Vuelve a generar los datos visibles en el panel *Resultados*.

Ejecutar script SQL
Si hay un script SQL en el panel *Resultados*, el script ejecuta la asignación en la base de datos de destino, teniendo en cuenta las acciones de tabla definidas.

Insertar o quitar marcador
Inserta un marcador en la posición del cursor en el panel *Resultados*.

Marcador siguiente
Navega hasta el siguiente marcador en el panel *Resultados*.

Marcador anterior
Navega hasta el marcador anterior en el panel *Resultados*.

Quitar todos los marcadores
Quita todos los marcadores que están definidos en el panel *Resultados*.

Texto XML pretty-print
Ajusta el formato del documento XML en el panel *Resultados* para presentarlo de forma estructurada. A cada nodo secundario se le aplica una sangría (1 tabulación) con respecto a su primario. Para controlar el tamaño de la sangría utilice la opción *Tamaño de la tabulación* del grupo de opciones *Tabulaciones*.

Configurar la vista Texto
Abre el cuadro de diálogo "Configurar la vista Texto", donde puede personalizar la configuración de esta vista en los paneles *Resultados*, *XSLT* y *XQuery*. También muestra las teclas de acceso rápido que se pueden usar en la ventana. Para más información consulte el apartado Características de la vista Texto.

15.9 Depuración

Iniciar depuración (F11)

Inicia o continúa la depuración hasta llegar a un punto de interrupción o hasta que finaliza la asignación.

Detener depuración (Mayús + F5)

Detiene la depuración. Esto comando cierra el modo depurador y rehabilita el modo estándar de MapForce.

Paso a paso por instrucciones (F11)

Ejecuta la asignación hasta que se termina un solo paso en cualquier posición de la asignación. En el depurador el término ***paso*** se usa para designar a un grupo lógico de cálculos dependientes que normalmente producen un solo elemento de una secuencia.

Dependiendo del contexto de la asignación, este comando puede leerse como "ir a la izquierda / ir al secundario de destino / ir al primario de origen".

Paso a paso por procedimientos (F10)

Continúa con la ejecución hasta que finaliza el paso actual (o se termina otra vez para otro elemento de la secuencia) o hasta que finaliza un paso no relacionado. Este comando se salta los cálculos que son entradas del paso actual.

Paso a paso para salir (Mayús + F11)

Continúa con la ejecución hasta que el resultado del paso actual se consume o hasta que se ejecuta un paso que no es una entrada ni un secundario del consumo. Este comando sale del cálculo actual.

Dependiendo del contexto de la asignación, este comando puede leerse como "ir a la derecha / ir al primario de destino / ir al secundario de origen".

Paso mínimo (Ctrl + F11)

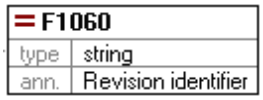
Continúa con la ejecución hasta que se produce o consume un valor. Este comando subdivide un paso y se detiene dos veces por cada conexión: cuando su origen produce un valor y cuando su destino lo consume. MapForce no siempre calcula los valores en el orden que sugiere la asignación, así que los eventos de producción y consumo no siempre van de la mano.

© 2018 Altova GmbH

15.10 Vista

Mostrar anotaciones

Muestra las anotaciones del esquema XML (así como información EDI) en la ventana del componente.
Si también se activa el icono **Mostrar tipos**, el tipo y la anotación aparecen en una tabla.



Mostrar tipos

Muestra los tipos de datos del esquema para cada elemento o atributo.
Si también se activa el icono **Mostrar anotaciones**, el tipo y la anotación aparecen en una tabla.

Mostrar biblioteca en el título de la función

Muestra el nombre de la biblioteca en paréntesis en el título de la función.

Mostrar información rápida

Muestra información rápida con un texto explicativo al pasar el puntero del ratón por encima de la función.

Opciones de visualización XBRL

Véase el apartado Opciones XBRL generales.

Mostrar los conectores del componente seleccionado

Permite alternar entre una vista con todos los conectores de la asignación y una vista con los conectores relacionados con los componentes que están seleccionados.

Mostrar los conectores desde su origen a su destino

Permite alternar entre:

- una vista con todos los conectores **directamente unidos** al componente que está seleccionado y
- una vista con los conectores unidos al componente seleccionado, que empiezan en el componente de origen y terminan en el de destino

Zoom

Abre el cuadro de diálogo de zoom, donde puede indicar el factor de zoom en formato numérico o con un control deslizante.

Atrás

Retrocede en las asignaciones que están abiertas en el panel **Asignación**.

Adelante

Avanza por las asignaciones que están abiertas en el panel **Asignación**.

Barra de estado

Muestra/oculta la barra de estado que está visible bajo la ventana Mensajes.

Ventana Bibliotecas
Muestra/oculta la ventana Biblioteca, que contiene todas las funciones.

Mensajes
Muestra/oculta la ventana Mensajes, donde aparecen los resultados de la validación, etc. La ventana Mensajes se activa automáticamente cuando se genera código para mostrar el resultado de la validación.

Vista general
Muestra/oculta la ventana Vista general. Arrastre el rectángulo rojo por la ventana de vista general para navegar por el panel **Asignación**.

Ventana Proyecto
Muestra/oculta la ventana Proyecto.

© 2018 Altova GmbH

15.11 Herramientas

Recursos globales
Abre el cuadro de diálogo "Administrar recursos globales" donde puede agregar, editar o eliminar recursos globales en el archivo XML de recursos globales (*véase* *Recursos globales de Altova*).

Configuración activa
Permite seleccionar la configuración de recurso global que está activa en una lista que enumera todas las configuraciones definidas previamente como recursos globales.

Crear asignación invertida
Crea una asignación *invertida* a partir de la asignación activa que será la base de una asignación nueva. Tenga en cuenta que el resultado no es una asignación completa porque en la asignación invertida solamente se conservan las conexiones directas entre los componentes de la asignación original. Es muy posible que la asignación resultante no sea válida o que no se pueda ejecutar (al abrir el panel **Resultados**) sin antes realizar ciertos cambios.

Cuando se invierte una asignación de datos, el componente de origen pasa a ser el componente de destino y viceversa. Si se asignó un archivo de instancia XML de entrada o salida a un componente, estos archivos también se permutarán.

Estos son los datos que se conservan:

- conexiones directas entre los componentes
- conexiones directas entre los componentes de una asignación en cadena
- el tipo de conexión estándar, de contenido mixto y de copia total
- la configuración de componentes de paso a través
- los componentes de base de datos

Estos son los datos que no se conservan:

- conexiones a través de funciones, filtros, etc. así como las funciones, los filtros, etc.
- funciones definidas por el usuario
- componentes de servicio web

Restaurar barras de herramientas y ventanas
Restaura las barras de herramientas, los ayudantes de entrada, las ventanas acopladas, etc. a su estado predeterminado. Es necesario reiniciar MapForce para que los cambios surtan efecto.

Personalizar...
Abre un cuadro de diálogo donde puede personalizar la interfaz gráfica del usuario de MapForce, incluidas las barras de herramientas que aparecen en la interfaz. También permite editar los menús contextuales y las teclas de acceso rápido (*véase* *Personalizar teclas de acceso rápido*).

Opciones
Abre un cuadro de diálogo donde puede cambiar la configuración predeterminada de MapForce

(*véase* *Cambiar opciones de MapForce*).

© 2018 Altova GmbH

15.12 Ventanas

En cascada

Este comando reorganiza todas las ventanas abiertas **en forma de cascada** (es decir, escalonadas).

En mosaico horizontal

Este comando reorganiza todas las ventanas abiertas **en forma de mosaico horizontal**, mostrando todas las ventanas a la vez.

En mosaico vertical

Este comando reorganiza todas las ventanas abiertas **en forma de mosaico vertical**, mostrando todas las ventanas a la vez.

1
2

Esta lista enumera todas las ventanas que están abiertas en cada momento y permite cambiar de una ventana a otra.
También puede usar las teclas **Ctrl+Tabulador** o **Ctrl+F6** para pasar de una ventana a otra.

15.13 Ayuda

Contenido

Descripción

Abre la ayuda en pantalla de MapForce por la tabla de contenido, que aparece en el panel izquierdo de la ventana de ayuda. Esta tabla de contenido ofrece un resumen de la documentación. Haga clic en una entrada de la tabla para abrir la sección correspondiente de la documentación.

Índice

Descripción

Abre la ayuda en pantalla de MapForce por el índice de palabras clave, que aparece en el panel izquierdo de la ventana de ayuda. Este índice enumera todas las palabras clave de la ayuda y permite navegar a un tema con solo hacer doble clic en la palabra clave correspondiente. Si una palabra clave está asociada a varios temas, la ventana de ayuda muestra todos estos temas en pantalla.

Buscar

Descripción

Abre la ayuda en pantalla de MapForce por la función de búsqueda, que aparece en el panel izquierdo de la ventana de ayuda. Para buscar un término introduzca el término de búsqueda en el campo de consulta y pulse **Entrar**. El sistema de ayuda realiza una búsqueda en toda la documentación y devuelve una lista de resultados. Haga doble clic en una entrada para abrir la sección correspondiente de la documentación.

Activación del software

Descripción

Tras descargar el producto de software de Altova puede registrarlo o activarlo con una clave de evaluación gratuita o con una clave de licencia permanente.

- ***Clave de evaluación gratuita:*** cuando inicie el software por primera vez aparecerá el cuadro de diálogo "Activación del software". Este cuadro de diálogo incluye un botón para solicitar un código clave de evaluación gratuita. Introduzca su nombre, el nombre de su compañía y su dirección de correo electrónico y haga clic en **Enviar solicitud**. Altova le enviará la clave de evaluación al correo electrónico proporcionado. Ahora introduzca el código clave en el cuadro de diálogo "Activación del software" y haga clic en **Aceptar**

© 2018 Altova GmbH

para empezar a trabajar con el software, que permanecerá desbloqueado durante 30 días.

- ***Clave de licencia permanente:*** el cuadro de diálogo "Activación del software" también incluye un botón para comprar una clave de licencia permanente. Este botón conduce a la tienda en línea de Altova, donde podrá adquirir una clave de licencia permanente para el producto. Altova ofrece licencias para un solo usuario y licencias para varios usuarios (ambos tipos estarán en el correo electrónico que reciba de Altova). Las licencias para un solo usuario contienen los datos de la licencia, su nombre, el nombre de su compañía, su correo electrónico y un código clave. Las licencias para varios usuarios contienen los datos de la licencia, el nombre de su compañía y un código clave. Recuerde que el contrato de licencia prohíbe instalar más copias de las permitidas por la licencia adquirida. Recuerde introducir los datos que solicita el cuadro de diálogo de registro tal y como aparecen en el correo electrónico que recibió con las licencias.

Nota: recuerde que los datos de la licencia en el cuadro de diálogo "Activación del software" deben ser idénticos a los que aparecen en el correo electrónico que recibió con las licencias. En el caso de licencias para varios usuarios, cada usuario debe introducir su nombre en el campo *Nombre*.

Claves por correo electrónico y las distintas formas de activar las licencias de los productos de Altova

El correo electrónico que recibirá de Altova contiene:

- Los detalles de su licencia (nombre, compañía, correo electrónico, código clave)
- En un adjunto, un archivo de licencia con la extensión **`.altova_licenses`**

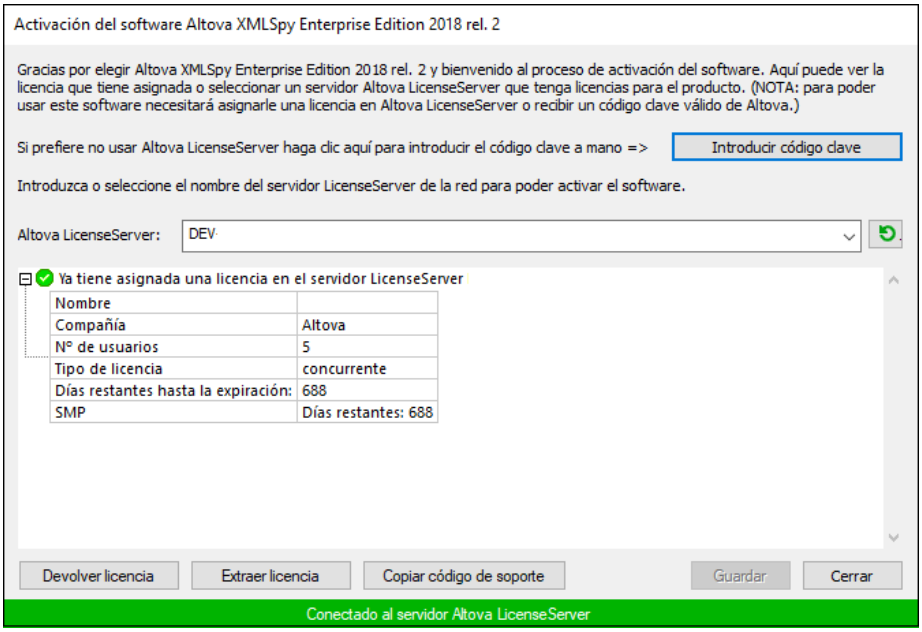
Para activar su producto de Altova, puede optar por una de las siguientes opciones:

- Introducir los detalles de la licencia recibidos con el correo electrónico en el cuadro de diálogo de activación de software de su producto de Altova y hacer clic en **Activar**.
- Guardar el archivo de licencia (**`.altova_licenses`**) en su equipo, hacer doble clic en el archivo de licencia, introducir los detalles necesarios en el cuadro de diálogo que aparece y finalmente hacer clic en **Aplicar claves**.
- Guardar el archivo de licencia (**`.altova_licenses`**) en su equipo y cargarlo desde esa ubicación a su Altova LicenseServer. Puede: (i) adquirir la licencia de su producto Altova con el cuadro de diálogo de activación de software del producto o (ii) asignar la licencia al producto de Altova LicenseServer. Para obtener más información sobre la gestión de licencias con el LicenseServer, lea el resto de esta sección.

El cuadro de diálogo "Activación del software" (*imagen siguiente*) se abre con el comando **Ayuda | Activación del software**.

Hay dos maneras de activar el software:

- Introduciendo la información de la licencia (con el botón **INTRODUCIR código clave...**) o
- Adquiriendo una licencia a través de un servidor Altova LicenseServer de la red (haciendo clic en el botón **Usar Altova LicenseServer**, situado al final del cuadro de diálogo). Para ello es necesario que el servidor LicenseServer tenga una licencia para su producto en el repositorio de licencias. Si así es, el cuadro de diálogo "Activación del software" emite un mensaje a tal efecto (*imagen siguiente*) y basta con hacer clic en el botón **Guardar** para adquirir la licencia.



Una vez se ha adquirido una licencia para un equipo específico (es decir, "instalada") del servidor LicenseServer, no se puede devolver al mismo hasta 7 días después. Transcurridos estos 7 días podrá devolver la licencia de ese equipo (con el botón **Devolver licencia**) para que pueda ser adquirida por otro cliente. No obstante, el administrador de LicenseServer puede anular asignaciones de licencias desde la interfaz web del servidor LicenseServer en cualquier momento. Observe que únicamente se pueden devolver las licencias instaladas en equipos específicos, no las licencias concurrentes.

Extracción de licencias

Puede extraer una licencia del repertorio durante un período máximo de 30 días de modo que la licencia se almacene en el equipo donde se ejecuta el producto. Esto le permitirá trabajar sin conexión a Internet, lo cual puede ser útil si desea trabajar en un entorno que no dispone de acceso a su servidor Altova LicenseServer (p. ej. cuando el producto servidor de Altova está instalado en un equipo portátil y el usuario se encuentra de viaje). Mientras la licencia esté extraída, LicenseServer indicará que la licencia está en uso y no podrá ser utilizada por ningún otro equipo. La licencia vuelve automáticamente a su estado insertado cuando finaliza el período de extracción de la licencia. La

 © 2018 Altova GmbH

licencia extraída también se puede insertar en el servidor en cualquier momento con el botón **Insertar** del cuadro de diálogo "Activación del software".

Siga estas instrucciones para extraer una licencia:

1. En el cuadro de diálogo "Activación del software" haga clic en el botón **Extraer licencia** (*imagen anterior*).
2. Aparece el cuadro de diálogo "Extracción de licencias". Seleccione el periodo de extracción y haga clic en **Extraer**.
3. La licencia se extrae y el cuadro de diálogo "Activación del software" muestra información sobre la extracción de la licencia, incluida la fecha y la hora en la que expira el plazo de extracción. Ahora, en lugar del botón **Extraer licencia**, aparece el botón **Insertar licencia**. Para insertar la licencia basta con hacer clic en este botón. Como la licencia vuelve automáticamente a su estado de inserción cuando finaliza el plazo de extracción, compruebe que el plazo seleccionado coincide con el período de tiempo que tiene pensado trabajar sin conexión a Internet.

Nota: para poder extraer licencias esta característica debe estar habilitada en el servidor LicenseServer. Si esta característica no está habilitada, recibirá un mensaje de error a tal efecto. Cuando esto ocurra, póngase en contacto con el administrador de su servidor LicenseServer.

Copiar código de soporte
Haga clic en **Copiar código de soporte** para copiar los detalles de la licencia en el portapapeles. Esta es la información que deberá introducir al ponerse en contacto con el equipo de soporte técnico a través del formulario de soporte técnico.

Altova LicenseServer es una práctica herramienta para administrar en tiempo real todas las licencias de Altova de la red y ofrece información detallada sobre cada licencia, asignaciones a clientes y uso de las licencias. La ventaja de usar este producto está en sus características administrativas. Altova LicenseServer puede descargarse gratis del sitio web de Altova. Para más información consulte la documentación de Altova LicenseServer.

Formulario de pedido

Descripción

Hay dos maneras de comprar licencias para los productos de Altova: con el botón **Comprar código clave...** del cuadro de diálogo "Activación del software" (ver apartado anterior) o con el comando **Ayuda | Formulario de pedido,** que le lleva directamente a la tienda en línea de Altova.

Registro del software

Descripción

Este comando abre la página de registro de productos de Altova en una pestaña del explorador. Si registra el software, recibirá información sobre actualizaciones y versiones nuevas del producto.

Buscar actualizaciones

Descripción

Comprueba si existe una versión más reciente del producto en el servidor de Altova y emite un mensaje a tal efecto.

Centro de soporte técnico

Descripción

Es un enlace al centro de soporte técnico del sitio web de Altova. El centro de soporte técnico incluye preguntas frecuentes, foros de debate y un formulario para ponerse en contacto con el equipo de soporte técnico de Altova.

Preguntas más frecuentes

Descripción

Es un enlace a la página de preguntas frecuentes del sitio web de Altova. Esta página se actualiza constantemente con las preguntas que recibimos de nuestros clientes.

Descargar herramientas gratis y componentes

Descripción

Es un enlace al centro de descargas de componentes del sitio web de Altova. Aquí puede descargar software adicional para usarlo con los productos de Altova, como procesadores XSLT y XSL-FO y paquetes de integración. Estos componentes suelen ser totalmente gratis.

© 2018 Altova GmbH

MapForce en Internet

Descripción

Es un enlace al sitio web de Altova, donde encontrará más información sobre MapForce, otros productos de Altova y tecnologías relacionadas.

Cursos de MapForce

Descripción

Es un enlace a la página de cursos del sitio web de Altova. Aquí puede seguir todos los cursos sobre productos y tecnologías relacionados con la línea de software de Altova.

Acerca de MapForce

Descripción

Abre la pantalla de presentación de la aplicación, que incluye el número de versión del producto e información sobre copyright.

Altova MapForce 2018 Enterprise Edition

Generador de código

16 Generador de código

El generador de código es una característica integrada de MapForce que sirve para generar código Java, C++ o C# a partir de archivos de asignación de datos diseñados con MapForce. Puede generar código a partir de asignaciones de datos con un solo origen y destino, pero también de asignaciones con varios orígenes de datos y varios destinos. Como resultado obtendrá una aplicación entera y totalmente funcional que lleva a cabo la operación de asignación de datos. Una vez generado el código, la asignación se podrá ejecutar con solo iniciar la aplicación recién generada. También podrá importar el código generado en otras aplicaciones o ampliarlo con más funciones.

Si la asignación de datos usa esquemas XML o archivos DTD, también tiene la opción de generar bibliotecas contenedoras del esquema (*véase* *Generar código a partir de esquemas XML o DTD*). Las bibliotecas contenedoras permiten trabajar con los datos XML de forma abstracta, sin preocuparse por la API XML subyacente (como MSXML, Apache Xerces, Microsoft System.Xml o JAXP).

© 2018 Altova GmbH

16.1 Introducción al generador de código

El objetivo principal del código generado es ejecutar una asignación de datos de MapForce. Además, puede generar bibliotecas contenedoras del esquema para los esquemas XML que utiliza la asignación de datos. Estas bibliotecas le servirán para escribir o leer datos en las instancias XML.

El código generado se expresa en los lenguajes de programación C++, Java o C#.

Lenguaje de destino	**C++**	**C#**	**Java**
Entornos de desarrollo	Microsoft Visual Studio 2008, 2010, 2013, 2015, 2017	Microsoft Visual Studio 2008, 2010, 2013, 2015, 2017	Java 1.7 o superior Eclipse 4.4 o superior Apache Ant (archivo build.xml)
Implementaciones XML DOM	MSXML 6.0 Apache Xerces 3	System.Xml	JAXP
API de BD	ADO	ADO.NET	JDBC

C++
Puede configurar si el código C++ generado debe usar MSXML 6.0 o Apache Xerces 3 o superior. MapForce genera archivos de proyecto (.vcproj) y de solución (.sln) completos para todas las versiones compatibles de Visual Studio (ver tabla más arriba). El código generado también puede ser compatible con MFC.

Nota: cuando cree código C++ para Visual Studio y utilice la biblioteca Xerces compilada previamente para Visual C++, antes deberá cambiar una opción de configuración del compilador en todos los proyectos de la solución:

1. Seleccione todos los proyectos en el Explorador de soluciones.
2. En el menú **Proyecto** haga clic en **Propiedades**.
3. Ahora haga clic en **Propiedades de configuración | C/C++ | Lenguaje**.
4. En la lista de configuraciones seleccione *Todas las configuraciones*.
5. Cambie **Treat wchar_t as Built-in Type** por **No (/Zc:wchar_t-)**

C#
El código C# generado utiliza las clases XML de .NET (System.Xml) y se puede usar desde cualquier lenguaje de programación compatible con .NET (como VB.NET, Managed C++ o J#). Se pueden generar archivos de proyecto para todas las versiones compatibles de Visual Studio (ver tabla más arriba).

Java
El código Java generado se escribe de acuerdo con la API de Java estándar para XML Processing (JAXP) e incluye un archivo de compilación Ant y archivos de proyecto para todas las versiones compatibles de Java y Eclipse (ver tabla más arriba).

Código de salida generado

La carpeta de destino designada incluirá todas las bibliotecas y los archivos que se necesitan para ejecutar la asignación, es decir:

- un número variable de bibliotecas de Altova que la asignación necesita (p. ej. las bibliotecas de funciones de Altova, las bibliotecas de base de datos, bibliotecas EDI, etc.)
- una aplicación de asignación completa. Cuando se compila y ejecuta, la aplicación lleva a cabo la transformación de la asignación de datos.

Plantillas del generador de código

El código de salida se puede personalizar todo lo que se quiera mediante un sencillo pero potente idioma de plantilla que le brinda un total control a la hora de asignar tipos de datos integrados de XML Schema a tipos de datos primitivos de un lenguaje de programación determinado.

Con este idioma de plantilla puede reemplazar el motor de análisis y validación subyacente, personalizar el código y adaptarlo a los requisitos de su organización o usar bibliotecas base diferentes como Microsoft Foundation Classes (MFC) y Standard Template Library (STL).

© 2018 Altova GmbH

16.2 Novedades

Versión 2018

- Compatibilidad con Microsoft Visual Studio 2013, 2015, 2017
- Fin de compatibilidad con Visual Studio 2005 y Xerces 2.x

Versión 2014

- Eliminación de la opción de modo de compatibilidad para la generación de código.

Versión 2011

Corrección de fallos y mejoras.

Versión 2010 R3

- Compatibilidad con Microsoft Visual Studio 2010.
- Compatibilidad con MSXML 6.0 en el código C++ generado.
- Compatibilidad con componentes de destino de 64 bits en proyectos C++ y C#.

Versión 2010 R2

Versión 2010

- Las facetas "enumeration" de XML Schema ya están disponibles como constantes simbólicas en las clases generadas (usando plantillas de 2007r3).

Versión 2009 sp1

- Compatibilidad con Apache Xerces 3.x (las versiones Xerces 2.6.x y posteriores siguen siendo compatibles).

Versión 2009

- Se ha rediseñado la implementación de la asignación generada para que sea compatible con secuencias y agrupaciones. No se han realizado cambios en la API.

Versión 2008 R2

- Generación de archivos de proyectos de Visual Studio 2008 para C# y C++.
- El código de asignación C# y Java generado con MapForce puede usar lectores y sistemas de escritura, secuencias, cadenas o documentos DOM como componentes de origen y destino.

Versión 2008

Las plantillas SPL 2007 R3 se han mejorado:

- Ahora puede eliminar elementos uno por uno
- Ahora puede acceder a metadatos del esquema (p. ej. nombres de elementos, facetas, enumeraciones, apariciones, etc.)
- Los tipos complejos derivados por extensión ahora se generan como clases derivadas

Versión 2007 R3

El generador de código se ha rediseñado para la versión 2007 Release 3 con el objetivo de simplificar el uso del código generado, reducir el volumen del código y mejorar el rendimiento.

- Gestión de documentos y nodos XML con propiedad explícita para evitar pérdida de memoria y habilitar subprocesos múltiples
- Sintaxis nueva para evitar colisiones de nombres
- Tipos de datos nuevos para un uso más sencillo y un mayor rendimiento (uso de tipos nativos cuando sea posible, control de valores null...)
- Los atributos ya no se generan como colecciones
- El contenido de elementos simples ahora también se considera un atributo especial
- Un modelo de objetos interno nuevo (importante para las plantillas SPL personalizadas)
- Modo de compatibilidad para generar código en el estilo de versiones anteriores
- Clases contenedoras ahora sólo se generan a petición para código más pequeño

© 2018 Altova GmbH

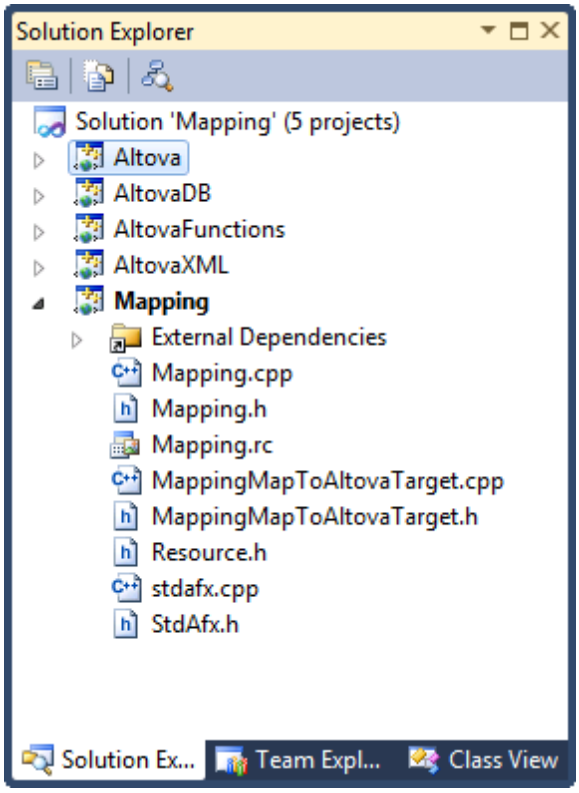
16.3 Generar código C++

Puede generar código C++ para Visual Studio 2008, 2010, 2013, 2015, 2017. El código generado incluye archivos **.sln** y .**vcproj** para Visual Studio. A la hora de generar código debe tener en cuenta que:

- puede generar código a partir de un diseño de asignación (.mfd) o de un proyecto de asignación de datos (.mfp). Si genera código a partir de una asignación de datos, la aplicación resultante ejecuta la transformación de asignación de datos correspondiente. Por el contrario, si genera código a partir de un proyecto de MapForce (.mfp) que incluya varias asignaciones de datos, la aplicación resultante ejecuta en bloque todas las asignaciones que incluye el proyecto.
- las opciones generales de generación de código se pueden cambiar haciendo clic en el comando de menú **Herramientas | Opciones** y después en la pestaña *Generación*.
- puede cambiar el nombre de la aplicación que se genera y otras opciones de configuración desde el comando de menú **Archivo | Configurar asignación**. El nombre predeterminado de la aplicación es **Mapping**.
- si la asignación de datos contiene componentes de base de datos, puede ver determinadas opciones de la base de datos haciendo clic en el componente y seleccionando el comando de menú **Componente | Propiedades**.

Las soluciones C++ generadas en MapForce suelen incluir:

- varias bibliotecas firmadas por Altova que la asignación de datos necesita (todas empiezan con el prefijo **Altova**) y
- el proyecto de asignación principal (en nuestro ejemplo: **Mapping**), que incluye la aplicación de asignación de datos y los archivos independientes.



Solución C++ de muestra generada con MapForce

Esta sección incluye varios apartados:

- Generar código a partir de una asignación de datos
- Generar código a partir de un proyecto de asignación de datos
- Compilar el proyecto
- Ejecutar la aplicación

16.3.1 Generar código a partir de una asignación de datos

Para generar código C++ a partir de un archivo de diseño de asignación de datos (.mfd):

1. Revise y seleccione las opciones de generación de código adecuadas (*véase Opciones del generador de código*).
2. En el menú **Archivo** haga clic en el comando **Generar código en | C++**.
3. Seleccione el directorio de destino para los archivos que se deben generar y haga clic en **Aceptar** para confirmar. El resultado de la generación de código (mensaje de error o éxito) aparece en la ventana Mensajes.

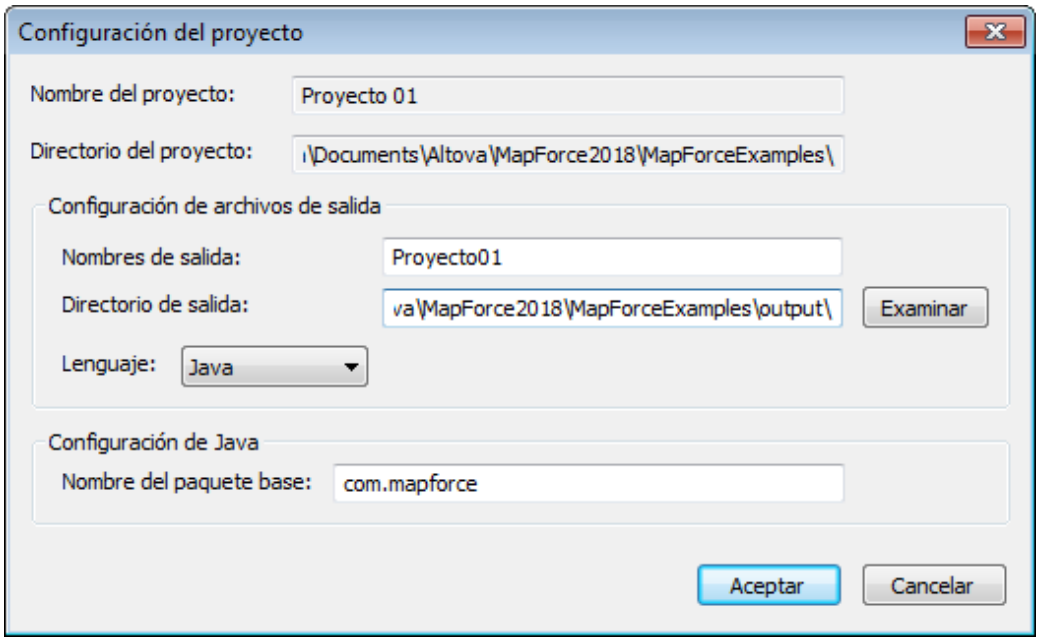
El nombre predeterminado de la aplicación generada es **Mapping**. Si lo prefiere, puede modificar el nombre predeterminado así como otras opciones de configuración de la aplicación (*véase Cambiar configuración de la asignación*).

© 2018 Altova GmbH

16.3.2 Generar código a partir de un proyecto de asignación

Para generar código a partir de un proyecto de asignación de datos (.mfp):

1. Si todavía no lo ha hecho, abra el proyecto de asignación de datos en MapForce.
2. Haga clic con el botón derecho en la ventana Proyecto y después elija el comando **Propiedades**.



3. Revise y modifique la configuración del proyecto según proceda (sobre todo debe comprobar que el lenguaje de destino y el directorio de salida están configurados correctamente). Después haga clic en **Aceptar**.
4. En el menú **Proyecto** haga clic en el comando **Generar código para todo el proyecto**.

El progreso y el resultado del proceso de generación de código (con mensajes de error o éxito) aparece reflejado en la ventana Mensajes.

La aplicación generada recibe el mismo nombre que el nombre del proyecto, pero si el nombre del proyecto contiene espacios, en el nombre de la aplicación los espacios se convierten en caracteres de subrayado. El código se genera por defecto en el directorio donde está ubicado el proyecto de MapForce, en el subdirectorio **output**.

Para cambiar el directorio de salida y el nombre del proyecto haga clic en el proyecto en la ventana Proyecto y después seleccione el comando de menú **Proyecto | Propiedades**. Si el proyecto de MapForce tiene carpetas, podrá cambiar las opciones de generación de código para cada una de las carpetas por separado (clic con el botón derecho en una carpeta y después **Propiedades** en el menú contextual). Todas las carpetas del proyecto heredan por defecto las opciones de generación de código del proyecto de MapForce.

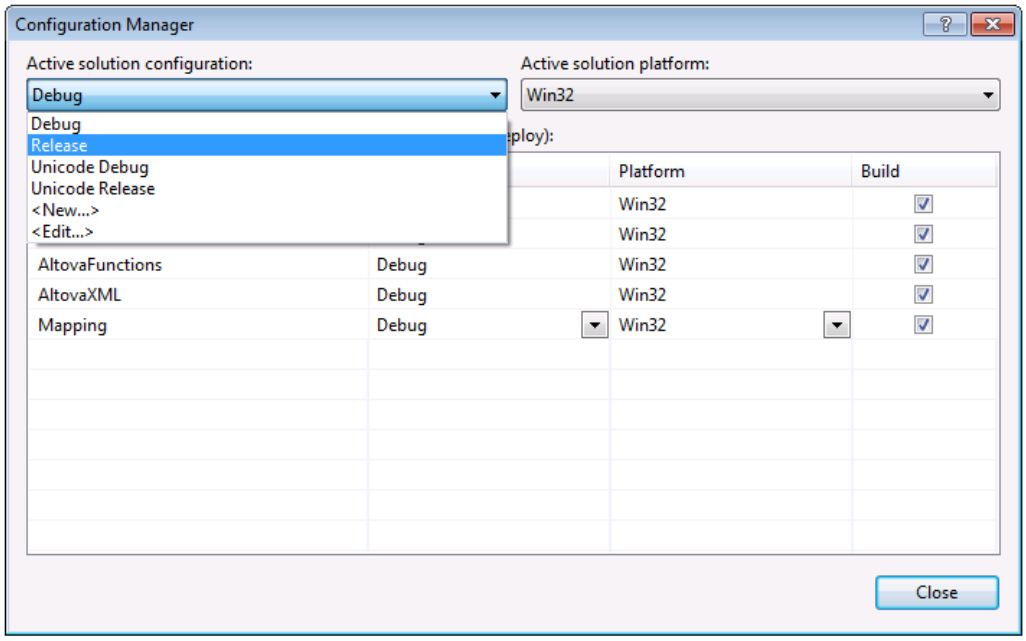
16.3.3 Compilar el proyecto

Una vez generado el código C++, el siguiente paso consiste en compilarlo en Visual Studio:

1. Abra el archivo de solución generado (.sln) en Visual Studio.

El nombre predeterminado del archivo de solución es **Mapping.sln** y su ubicación es el subdirectorio **Mapping** relativo al directorio donde guardó el código generado. Si cambió el nombre de la aplicación desde la configuración de la asignación de datos, entonces el nombre del archivo .sln también se modificó de la forma correspondiente. Por ejemplo, si cambió el nombre de la aplicación por **MiAplicación**, entonces el archivo de solución se llamará **MiAplicación.sln** y estará ubicado en el subdirectorio **MiAplicación**.

2. En el menú **Compilar** haga clic en **Administrador de configuración**.



3. Seleccione la configuración de compilación correspondiente (**Depuración**, **Versión**, **Depuración Unicode**, **Versión Unicode**). Recuerde que solamente las opciones Unicode admiten el conjunto de caracteres Unicode completo en XML y en otros archivos. Las opciones no Unicode trabajan con la página de código local de su instalación de Windows.
4. En el menú **Compilar** haga clic en **Compilar solución**.

16.3.4 Ejecutar la aplicación

Cuando termine de compilar el proyecto de Visual Studio, se genera una aplicación de la línea de comandos llamada **Mapping.exe**. (Recuerde que si cambió el nombre de la aplicación en la configuración de la asignación de datos, el nombre del ejecutable también se modificó consecuentemente.)

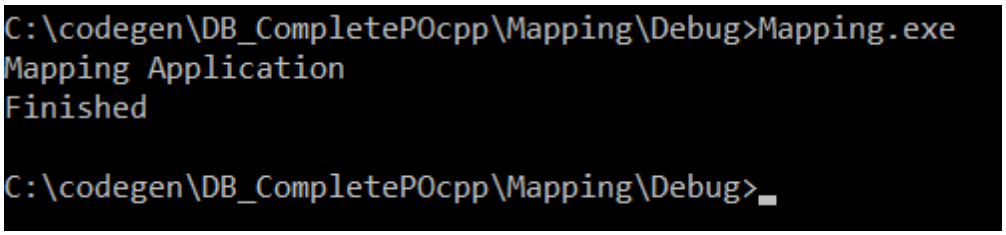
Podrá encontrar la aplicación de asignación de datos en uno de estos subdirectorios relativos al

 © 2018 Altova GmbH

archivo .sln (dependiendo de la opción de compilación elegida):

- Debug
- Release
- Unicode Debug
- Unicode Release

Para ejecutar la aplicación abra un símbolo del sistema, cambie el directorio actual por la ruta de acceso del ejecutable y ejecútelo. Por ejemplo:



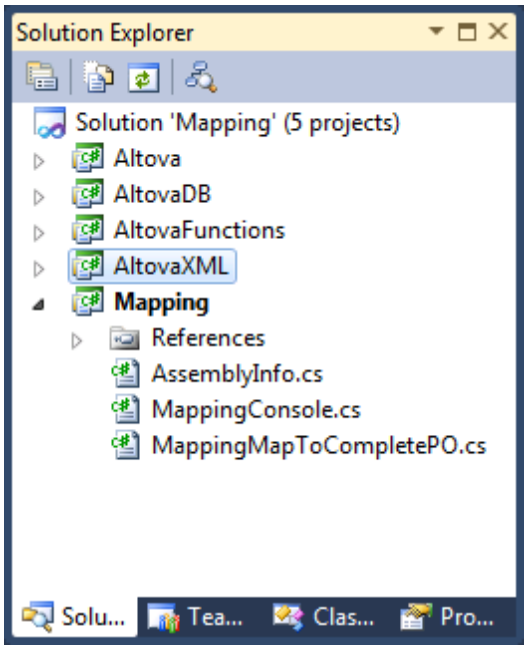
16.4 Generar código C#

Puede generar código C# para Visual Studio 2008, 2010, 2013, 2015, 2017. El código generado incluye archivos **.sln** y **.csproj** para Visual Studio. Los usuarios de Mono pueden usar archivos de solución de Visual Studio con xbuild. A la hora de generar código debe tener en cuenta que:

- puede generar código a partir de un diseño de asignación (.mfd) o de un proyecto de asignación de datos (.mfp). Si genera código a partir de una asignación de datos, la aplicación resultante ejecuta la transformación de asignación de datos correspondiente. Por el contrario, si genera código a partir de un proyecto de MapForce (.mfp) que incluya varias asignaciones de datos, la aplicación resultante ejecuta en bloque todas las asignaciones que incluye el proyecto.
- las opciones generales de generación de código se pueden cambiar haciendo clic en el comando de menú **Herramientas | Opciones** y después en la pestaña *Generación*.
- puede cambiar el nombre de la aplicación que se genera y otras opciones de configuración desde el comando de menú **Archivo | Configurar asignación**. El nombre predeterminado de la aplicación es **Mapping**.
- si la asignación de datos contiene componentes de base de datos, puede ver determinadas opciones de la base de datos haciendo clic en el componente y seleccionando el comando de menú **Componente | Propiedades**.

Las soluciones C# generadas en MapForce suelen incluir:

- varias bibliotecas firmadas por Altova que la asignación de datos necesita (todas empiezan con el prefijo **Altova**) y
- el proyecto de asignación principal (en nuestro ejemplo: **Mapping**), que incluye la aplicación de asignación de datos y los archivos independientes.



Solución C# de muestra generada con MapForce

© 2018 Altova GmbH

Esta sección incluye varios apartados:

- Generar código a partir de una asignación de datos
- Generar código a partir de un proyecto de asignación de datos
- Compilar el proyecto
- Ejecutar la aplicación

16.4.1 Generar código a partir de una asignación de datos

Para generar código C# a partir de un archivo de diseño de asignación de datos (.mfd):

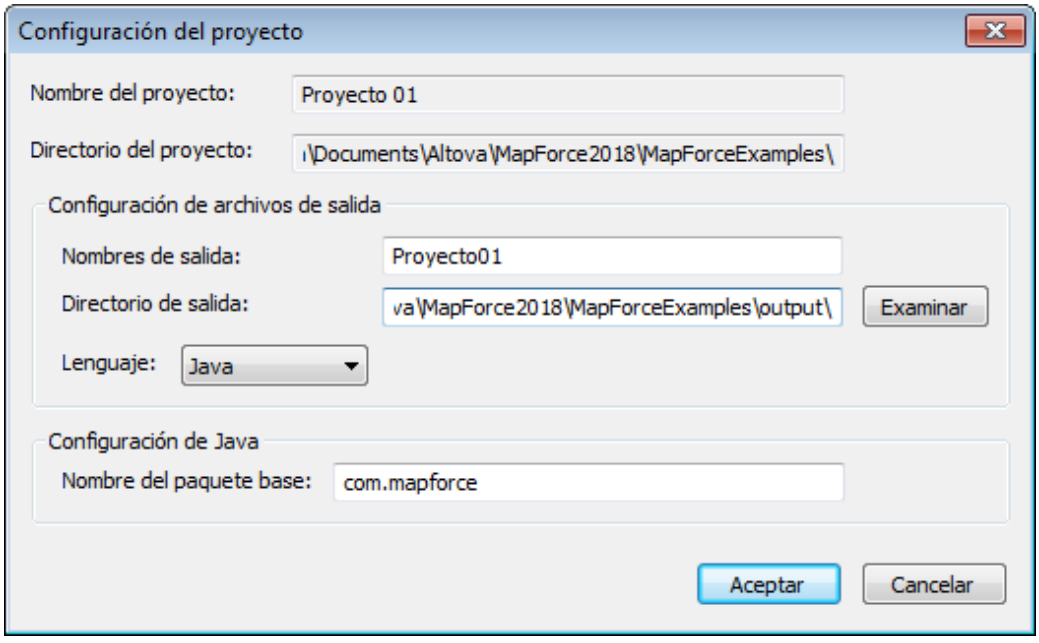
1. Revise y seleccione las opciones de generación de código adecuadas (*véase Opciones del generador de código*).
2. En el menú **Archivo** haga clic en el comando **Generar código en | C#**.
3. Seleccione el directorio de destino para los archivos que se deben generar y haga clic en **Aceptar** para confirmar. El resultado de la generación de código (mensaje de error o éxito) aparece en la ventana Mensajes.

El nombre predeterminado de la aplicación generada es **Mapping**. Si lo prefiere, puede modificar el nombre predeterminado así como otras opciones de configuración de la aplicación (*véase Cambiar configuración de la asignación*).

16.4.2 Generar código a partir de un proyecto de asignación

Para generar código a partir de un proyecto de asignación de datos (.mfp):

1. Si todavía no lo ha hecho, abra el proyecto de asignación de datos en MapForce.
2. Haga clic con el botón derecho en la ventana Proyecto y después elija el comando **Propiedades**.



3. Revise y modifique la configuración del proyecto según proceda (sobre todo debe comprobar que el lenguaje de destino y el directorio de salida están configurados correctamente). Después haga clic en **Aceptar**.
4. En el menú **Proyecto** haga clic en el comando **Generar código para todo el proyecto**.

El progreso y el resultado del proceso de generación de código (con mensajes de error o éxito) aparece reflejado en la ventana Mensajes.

La aplicación generada recibe el mismo nombre que el nombre del proyecto, pero si el nombre del proyecto contiene espacios, en el nombre de la aplicación los espacios se convierten en caracteres de subrayado. El código se genera por defecto en el directorio donde está ubicado el proyecto de MapForce, en el subdirectorio **output**.

Para cambiar el directorio de salida y el nombre del proyecto haga clic en el proyecto en la ventana Proyecto y después seleccione el comando de menú **Proyecto | Propiedades**. Si el proyecto de MapForce tiene carpetas, podrá cambiar las opciones de generación de código para cada una de las carpetas por separado (clic con el botón derecho en una carpeta y después **Propiedades** en el menú contextual). Todas las carpetas del proyecto heredan por defecto las opciones de generación de código del proyecto de MapForce.

16.4.3 Compilar el proyecto

Una vez generado el código C#, el siguiente paso consiste en compilarlo en Visual Studio:

1. Abra el archivo de solución generado (.sln) en Visual Studio.

El nombre predeterminado del archivo de solución es **Mapping.sln** y su ubicación es el subdirectorio **Mapping** relativo al directorio donde guardó el código generado. Si cambió el nombre de la aplicación desde la configuración de la asignación de datos, entonces el nombre del archivo .sln también se modificó de la forma correspondiente. Por ejemplo, si cambió el nombre de la aplicación por **MiAplicación**, entonces el archivo de solución se llamará **MiAplicación.sln** y estará ubicado en el subdirectorio **MiAplicación**.

2. En el menú **Compilar** haga clic en **Administrador de configuración**.
3. Seleccione la configuración de compilación correspondiente (**Depuración**, **Versión**).
4. En el menú **Compilar** haga clic en **Compilar solución**.

16.4.4 Ejecutar la aplicación

Cuando termine de compilar el proyecto de Visual Studio, se genera una aplicación de la línea de comandos llamada **Mapping.exe**. (Recuerde que si cambió el nombre de la aplicación en la configuración de la asignación de datos, el nombre del ejecutable también se modificó consecuentemente.)

Podrá encontrar la aplicación de asignación de datos en uno de estos subdirectorios relativos al archivo .sln (dependiendo de la opción de compilación elegida):

- bin\Debug
- bin\Release

 © 2018 Altova GmbH

Para ejecutar la aplicación abra un símbolo del sistema, cambie el directorio actual por la ruta de acceso del ejecutable y ejecútelo. Por ejemplo:

```
C:\codegen\DB_CompletePOcs\Mapping\bin\Release>Mapping.exe
Mapping Application
Connecting to CustomersAndArticles database...

Finished

C:\codegen\DB_CompletePOcs\Mapping\bin\Release>
```

16.5 Generar código Java

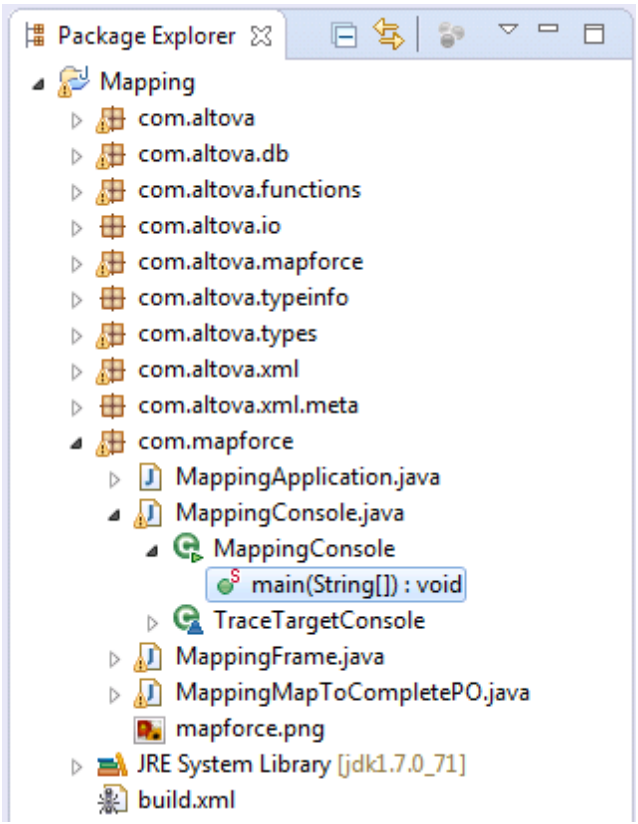
Puede generar código de programa para Java 1.7 o superior. A la hora de generar código debe tener en cuenta que:

- puede generar código a partir de un diseño de asignación (.mfd) o de un proyecto de asignación de datos (.mfp). Si genera código a partir de una asignación de datos, la aplicación resultante ejecuta la transformación de asignación de datos correspondiente. Por el contrario, si genera código a partir de un proyecto de MapForce (.mfp) que incluya varias asignaciones de datos, la aplicación resultante ejecuta en bloque todas las asignaciones que incluye el proyecto.
- las opciones generales de generación de código se pueden cambiar haciendo clic en el comando de menú **Herramientas | Opciones** y después en la pestaña *Generación*.
- puede cambiar el nombre de la aplicación que se genera y otras opciones de configuración desde el comando de menú **Archivo | Configurar asignación**. El nombre predeterminado de la aplicación es **Mapping**.
- si la asignación de datos contiene componentes de base de datos, puede ver determinadas opciones de la base de datos haciendo clic en el componente y seleccionando el comando de menú **Componente | Propiedades**.

Los proyectos Java generados en MapForce suelen incluir:

- Varios paquetes Java firmados por Altova que la asignación de datos necesita (todos empiezan con el prefijo **com.altova**).
- El paquete **com.mapforce**, que incluye la aplicación de asignación de datos y sus archivos independientes como puede ver más abajo el nombre de este paquete se puede cambiar). Los dos archivos más importantes de este paquete son:
  - la aplicación de asignación de datos Java como aplicación de diálogo (**MappingApplication.java**) y
  - la aplicación de asignación de datos Java como aplicación de consola (**MappingConsole.java**).
- Un archivo **build.xml** que se puede ejecutar con Apache Ant para compilar el proyecto y generar archivos JAR.

© 2018 Altova GmbH



Aplicación Java de muestra generada con MapForce (en entorno Eclipse)

Esta sección incluye varios apartados:

- Generar código a partir de una asignación de datos
- Generar código a partir de un proyecto de asignación de datos
- Control de referencias JDBC
- Compilar el proyecto con Ant
- Ejemplo: ejecutar y compilar código Java con Eclipse y Ant

16.5.1 Generar código a partir de una asignación de datos

To generate Java code from a mapping design file (.mfd):

1. Review and select the desired code generation options (see Code generator options).
2. On the **File** menu, click **Generate code in | Java**.
3. Select a destination directory for the generated files, and then click OK to confirm. The result of code generation (error or success message) is displayed in the Messages window.

The default name of the generated application is **Mapping**, and the default name of the base package is **com.mapforce**. If required, you can change these from the Mapping Settings dialog box (see Changing the mapping settings).

Para generar código Java a partir de un archivo de diseño de asignación de datos (.mfd):

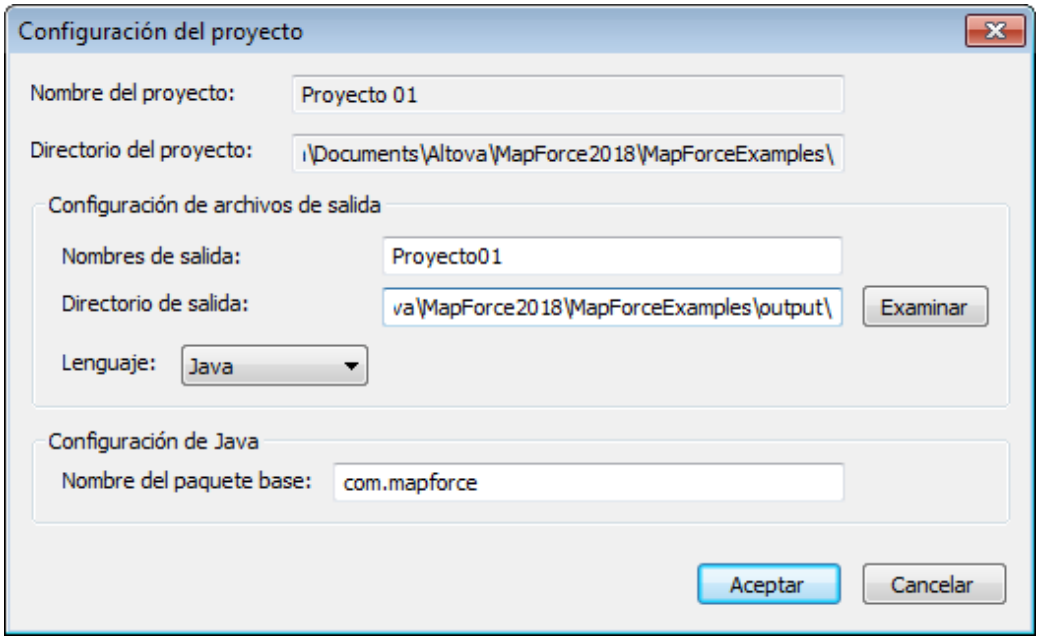
1. Revise y seleccione las opciones de generación de código adecuadas (*véase Opciones del generador de código*).
2. En el menú **Archivo** haga clic en el comando **Generar código en | Java**.
3. Seleccione el directorio de destino para los archivos que se deben generar y haga clic en **Aceptar** para confirmar. El resultado de la generación de código (mensaje de error o éxito) aparece en la ventana Mensajes.

El nombre predeterminado de la aplicación generada es **Mapping** y el nombre predeterminado del paquete base es **com.mapforce**. Si lo prefiere, puede modificar estos nombres predeterminados en el cuadro de diálogo "Configurar asignación" (*véase Cambiar configuración de la asignación*).

16.5.2 Generar código a partir de un proyecto de asignación

Para generar código a partir de un proyecto de asignación de datos (.mfp):

1. Si todavía no lo ha hecho, abra el proyecto de asignación de datos en MapForce.
2. Haga clic con el botón derecho en la ventana Proyecto y después elija el comando **Propiedades**.



3. Revise y modifique la configuración del proyecto según proceda (sobre todo debe comprobar que el lenguaje de destino y el directorio de salida están configurados correctamente). Después haga clic en **Aceptar**.
4. En el menú **Proyecto** haga clic en el comando **Generar código para todo el proyecto**.

El progreso y el resultado del proceso de generación de código (con mensajes de error o éxito) aparece reflejado en la ventana Mensajes.

© 2018 Altova GmbH

La aplicación generada recibe el mismo nombre que el nombre del proyecto, pero si el nombre del proyecto contiene espacios, en el nombre de la aplicación los espacios se convierten en caracteres de subrayado. El código se genera por defecto en el directorio donde está ubicado el proyecto de MapForce, en el subdirectorio **output**.

Para cambiar el directorio de salida y el nombre del proyecto haga clic en el proyecto en la ventana Proyecto y después seleccione el comando de menú **Proyecto | Propiedades**. Si el proyecto de MapForce tiene carpetas, podrá cambiar las opciones de generación de código para cada una de las carpetas por separado (clic con el botón derecho en una carpeta y después **Propiedades** en el menú contextual). Todas las carpetas del proyecto heredan por defecto las opciones de generación de código del proyecto de MapForce.

16.5.3 Control de referencias JDBC

Si la asignación se conecta a una base de datos por JDBC, compruebe que los controladores JDBC que utiliza la asignación están instalados en el sistema (*véase* *Crear una conexión JDBC*). Para ver la configuración JDBC actual de cualquier componente de base de datos haga clic en el componente y después seleccione el comando de menú **Componente | Propiedades**.

Además, si tiene pensado compilar archivos JAR a partir del código Java generado, debe agregar el atributo "Class-Path" para su controlador de base de datos en el archivo **build.xml**. Esto garantiza que la referencia al controlador de base de datos esté disponible en el archivo de manifiesto (**MANIFEST.MF**) después de la compilación del proyecto.

Para agregar el atributo "Class-Path":

1. En el archivo **build.xml** añada una referencia al archivo JAR del controlador de base de datos (un nuevo atributo "Class-Path"). Por ejemplo, para MySQL 5.1.16, el valor del atributo "Class-Path" sería:

   ```
   <attribute name="Class-Path" value="mysql-connector-java-5.1.16-
   bin.jar"/>
   ```

   El elemento **manifest** del archivo **build.xml** ahora tendría más o menos este aspecto:

   ```
   <manifest file="U:\_TTP_Verification__related\42730__.__SAP_iDoc\CODE_Jav
     <attribute name="Created-By" value="MapForce 2014"/>
     <attribute name="Main-Class" value="com.mapforce.Paccar866DBDELFOR_
     <attribute name="Class-Path" value="mysql-connector-java-5.1.16-bin.jar"/>
   </manifest>
   ```

2. Copie el archivo JAR del controlador JDBC a la carpeta que contiene el archivo JAR de la aplicación generada.

16.5.4 Compilar el proyecto con Ant

Apache Ant es una popular biblioteca Java (y herramienta de la línea de comandos) que automatiza la compilación de proyectos Java (*véase* *http://ant.apache.org/manual/*). Ant trabaja con los archivos de compilación, que definen el origen y el destino de la compilación de código y

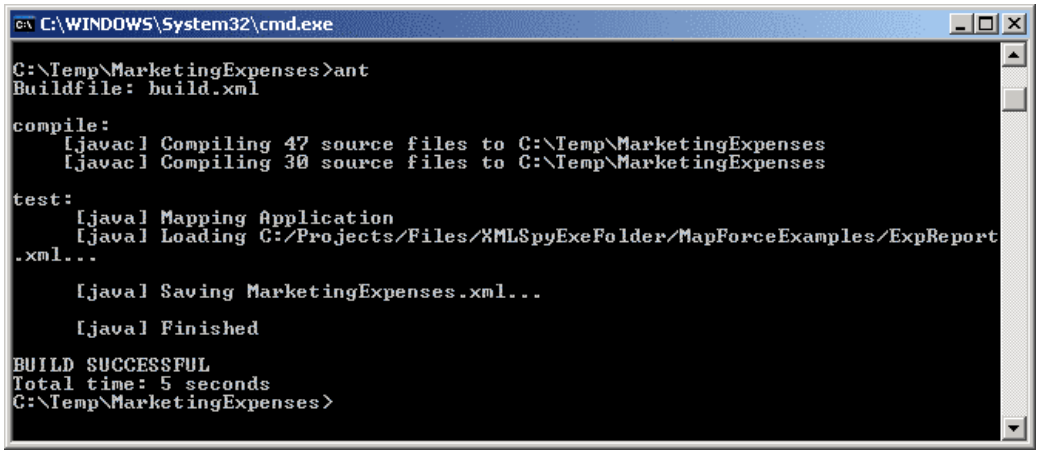
opciones de compilación concretas. Los proyectos generados con MapForce incluyen un archivo llamado **build.xml** que Ant puede reconocer, de modo que puede usar Ant para compilar proyectos generados con MapForce fácilmente.

Ant puede estar disponible en su sistema o estar integrado con Eclipse (o con otros entornos IDE Java). El sitio web http://ant.apache.org/manual/ incluye instrucciones para instalar Ant en el sistema. Para aprender a usar Ant en Eclipse puede consultar nuestro tutorial de Eclipse o la documentación de Eclipse.

Para saber si en su sistema existe una versión independiente de Ant (y no la versión integrada en Eclipse) abra un símbolo del sistema y teclee `ant` en la línea de comandos. Si Ant ya está disponible, el mensaje será parecido a este: `Buildfile: buildxml does not exist!` Este mensaje indica que Ant está instalado y está intentando compilar un archivo **build.xml** pero dicho archivo no existe en el directorio actual. Si ejecuta Ant desde un directorio que contenga un archivo **build.xml**, Ant ejecutará el archivo **build.xml** con las opciones de compilación definidas en él.

Para compilar un proyecto Java generado con MapForce con Ant:

1. Abra un símbolo del sistema y navegue hasta el directorio donde se generó el proyecto Java (recuerde que el directorio debe contener el archivo **build.xml**).
2. En el símbolo del sistema introduzca `ant`. Este comando compilará y ejecutará el código Java de acuerdo con las opciones de compilación definidas en el archivo **build.xml**.



Para generar un archivo JAR con Ant:

- En el símbolo del sistema introduzca `ant jar`.

Si necesita ayuda con la sintaxis y las opciones de los comandos de Ant, introduzca `ant -help` en la línea de comandos.

16.5.5 Ejemplo: crear una aplicación Java con Eclipse y Ant

Este ejemplo explica paso a paso cómo se puede generar una aplicación Java con MapForce y cómo se puede compilar con el entorno Eclipse y Apache Ant. Cuando termine de seguir los pasos del ejemplo, obtendrá una aplicación Java completa compilada que ejecuta una de las asignaciones de muestra que vienen con MapForce.

© 2018 Altova GmbH

Si ya sabe compilar aplicaciones Java con Eclipse y Ant, debe saber que para este ejemplo no hay ningún requisito especial aparte de estos:

- debe tener instalado Java Development Kit (JDK), Eclipse (https://www.eclipse.org/) y Ant (http://ant.apache.org/). Eclipse suele ofrecer funciones para la compilación de proyectos con Ant (*véase Compilar el proyecto con Ant*).
- la variable de entorno Windows PATH debe incluir la ruta de acceso del directorio **bin** de JDK (p. ej. **C:\Java\jdk1.7.0\bin**). Se trata de un requisito básico para el desarrollo de aplicaciones para la plataforma Java. Para más información consulte http://docs.oracle.com/javase/tutorial/essential/environment/paths.html.

Para este ejemplo se usó esta configuración:

- JDK 1.7.
- Eclipse IDE for Java Developers (Luna Service Release 4.4.1).
- Ant 1.9.2, que ya está integrado en la edición de Eclipse mencionada en el punto anterior (es decir, no se instaló ni configuró por separado).

Estos son los pasos o tareas de nuestro ejemplo:

- Paso nº1: generar código Java
- Paso nº2: importar el proyecto en Eclipse
- Paso nº3: ejecutar el proyecto como aplicación de interfaz gráfica
- Paso nº4: ejecutar el proyecto como aplicación de consola
- Paos nº5: compilar el archivo JAR con Ant

16.5.5.1 Paso nº1: generar código Java

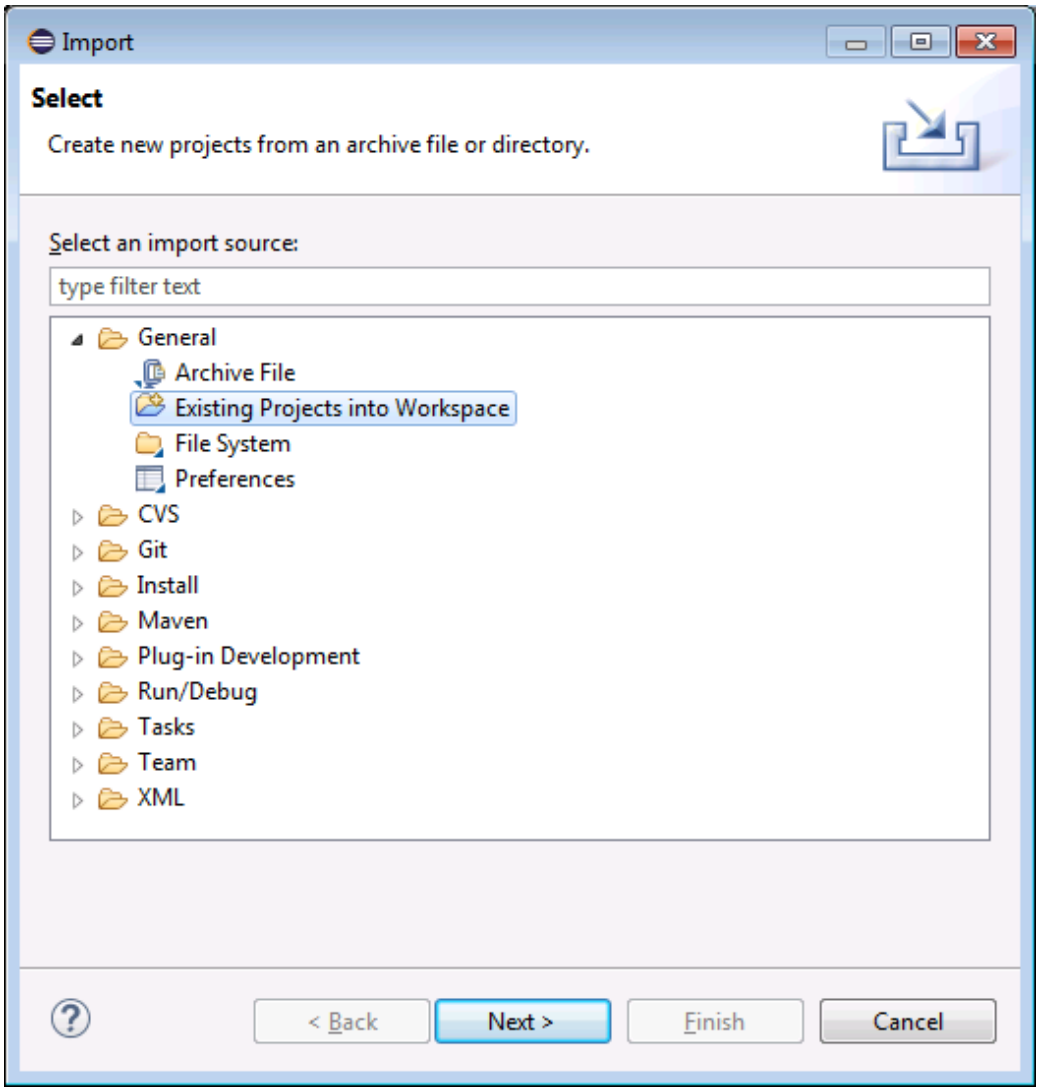
Para generar el código Java en MapForce:

1. En el menú **Archivo** haga clic en el comando **Abrir** y navegue hasta la asignación de datos **CompletePO.mfd** del directorio <Documentos>\Altova\MapForce2018\MapForceExamples\.
2. En el menú **Resultados** haga clic en el comando Java. Esto activa el lenguaje de transformación Java.
3. En el menú **Archivo** haga clic en el comando **Generar código en | Java**. Cuando la aplicación lo solicite, navegue hasta el directorio donde se debe guardar el proyecto Java (p. ej en el espacio de trabajo predeterminado de Eclipse). Para nuestro ejemplo usaremos el directorio de trabajo **C:\workspace\CompletePO**.

16.5.5.2 Paso nº2: importar el proyecto en Eclipse

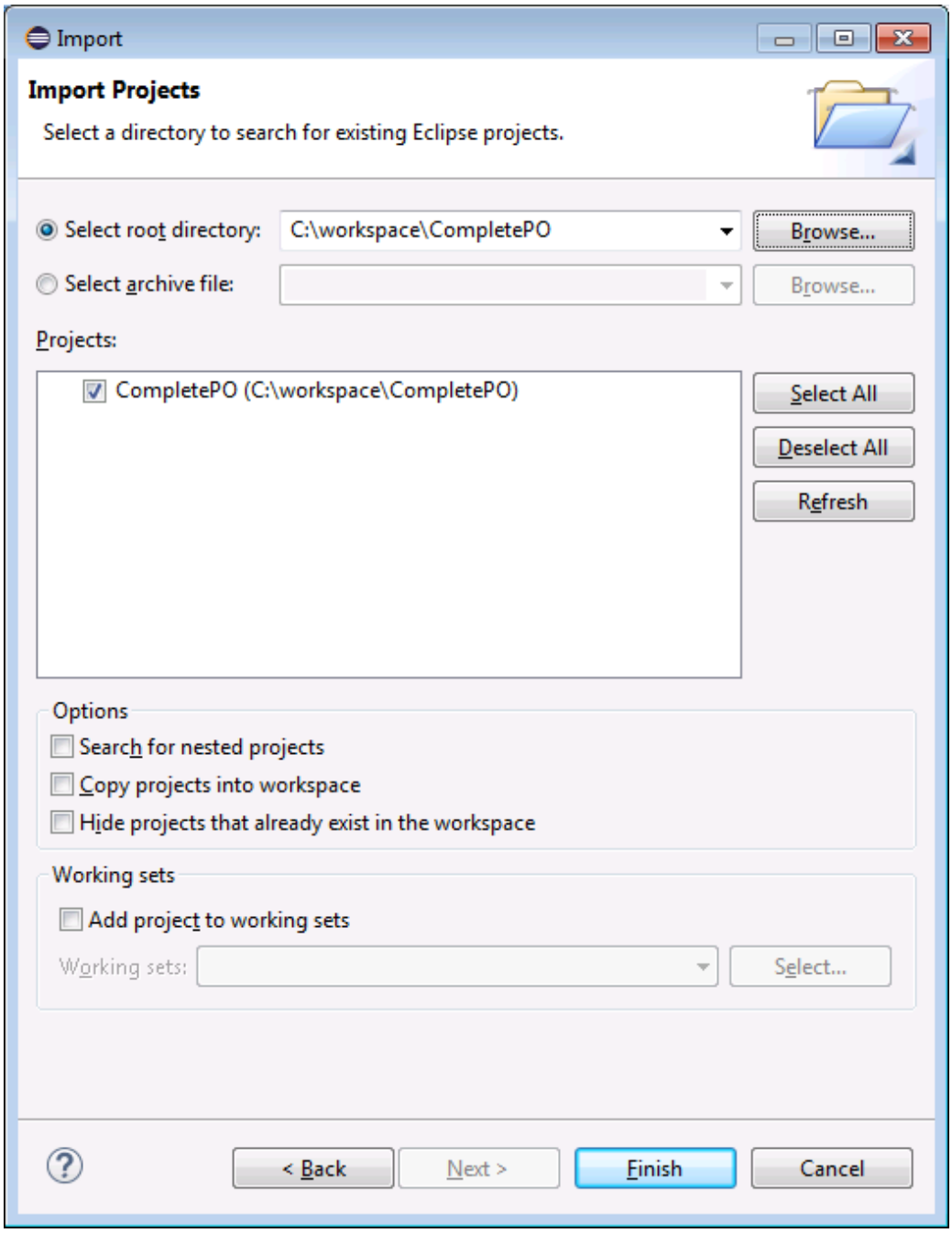
Para importar el proyecto en Eclipse:

1. Si todavía no lo ha hecho, ejecute Eclipse y cambie a la perspectiva Java predeterminada (con el comando de menú **Window | Open Perspective**).
2. En el menú **File** haga clic en el comando **Import** y después seleccione **Existing Projects into Workspace** (proyectos actuales al espacio de trabajo).

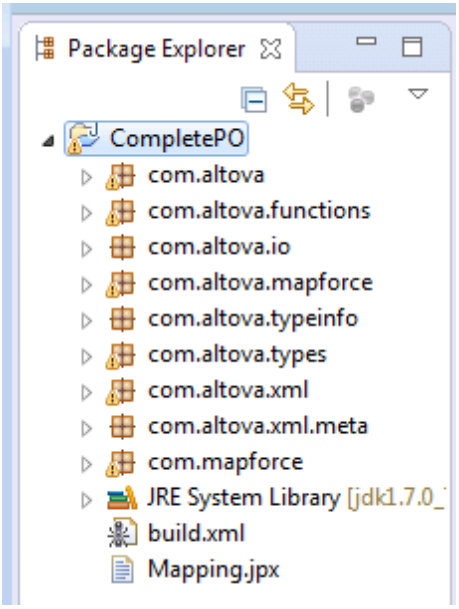


3. Haga clic en **Next** para continuar.

© 2018 Altova GmbH



4. Navegue hasta la carpeta donde guardó el código generado y después haga clic en **Finish** para finalizar. El proyecto Java creado con MapForce aparece ya en la vista del explorador de paquetes *Package Explorer*. Si no puede ver la vista *Package Explorer,* puede activarla con el comando de menú **Window | Show View | Package Explorer**.

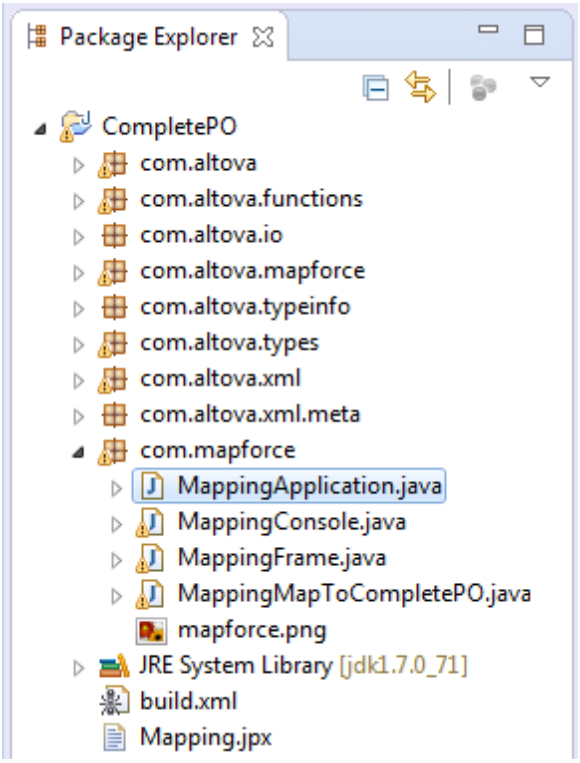


16.5.5.3 Paso nº3: ejecutar el proyecto como aplicación de interfaz gráfica

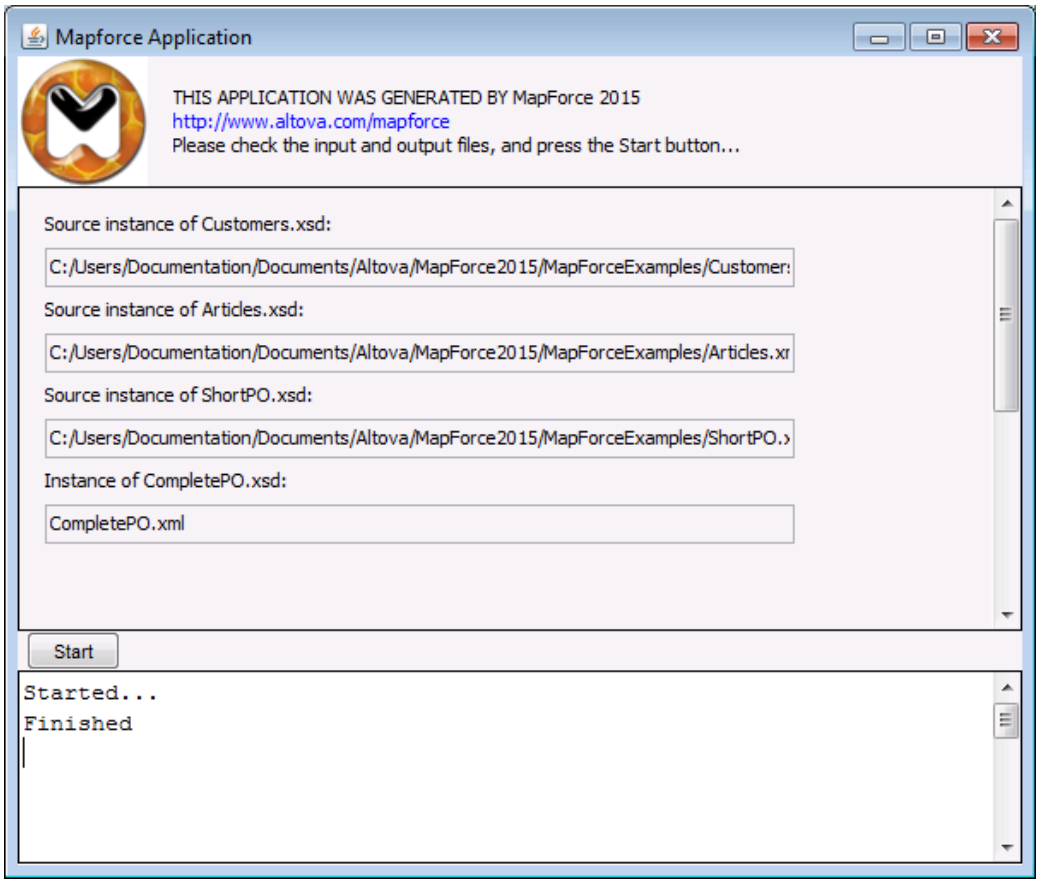
Para ejecutar el proyecto Java como aplicación de interfaz gráfica:

1. En la vista *Package Explorer* de Eclipse seleccione el archivo **MappingApplication.java** situado en el paquete **com.mapforce**.

© 2018 Altova GmbH



2. En el menú de ejecución **Run** haga clic en el comando **Run As | Java application** para ejecutar el archivo como aplicación Java.
3. En la ventana de la aplicación MapForce que aparece haga clic en **Start** para iniciar la ejecución de la asignación de datos.



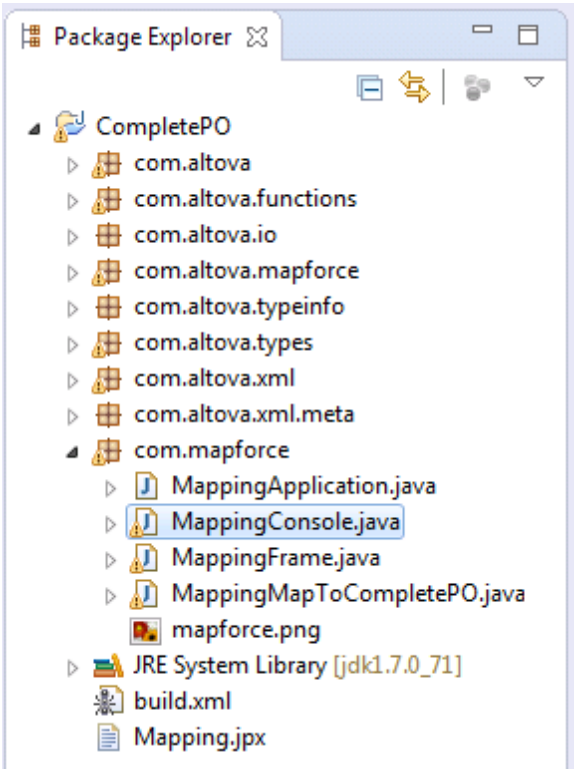
Si Eclipse detecta errores de configuración del sistema o de tiempo de ejecución, recibirá un mensaje de Eclipse. Si no se encuentran errores, la aplicación Java ejecuta la transformación de asignación de datos y genera el archivo **CompletePO.xml** en la ruta de acceso de salida (en nuestro ejemplo: **C:\workspace\CompletePO\CompletePO.xml**).

16.5.5.4 Paso nº4: ejecutar el proyecto como aplicación de consola

Para ejecutar el proyecto Java como aplicación de consola:

1. En la vista del explorador de paquetes *Package Explorer* de Eclipse seleccione el archivo **MappingConsole.java** situado en el paquete **com.mapforce**.

© 2018 Altova GmbH



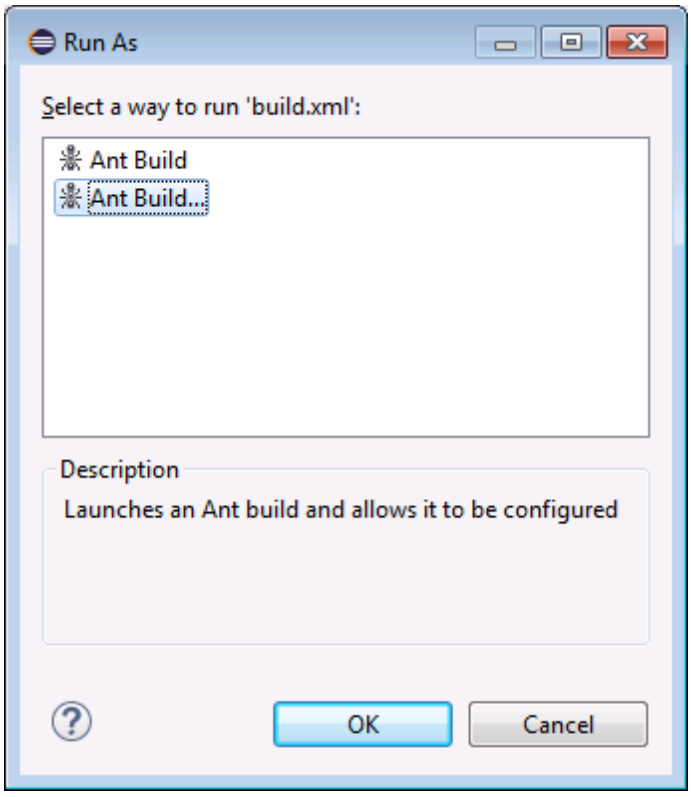
2. En el menú de ejecución **Run** haga clic en el comando **Run As | Java application** para ejecutar el archivo como aplicación Java.

Si Eclipse detecta errores de configuración del sistema o de tiempo de ejecución, recibirá un mensaje de Eclipse. Si no se encuentran errores, la aplicación Java ejecuta la transformación de asignación de datos y genera el archivo **CompletePO.xml** en la ruta de acceso de salida (en nuestro ejemplo: **C:\workspace\CompletePO\CompletePO.xml**).

16.5.5.5 Paso nº5: compilar el archivo JAR con Ant

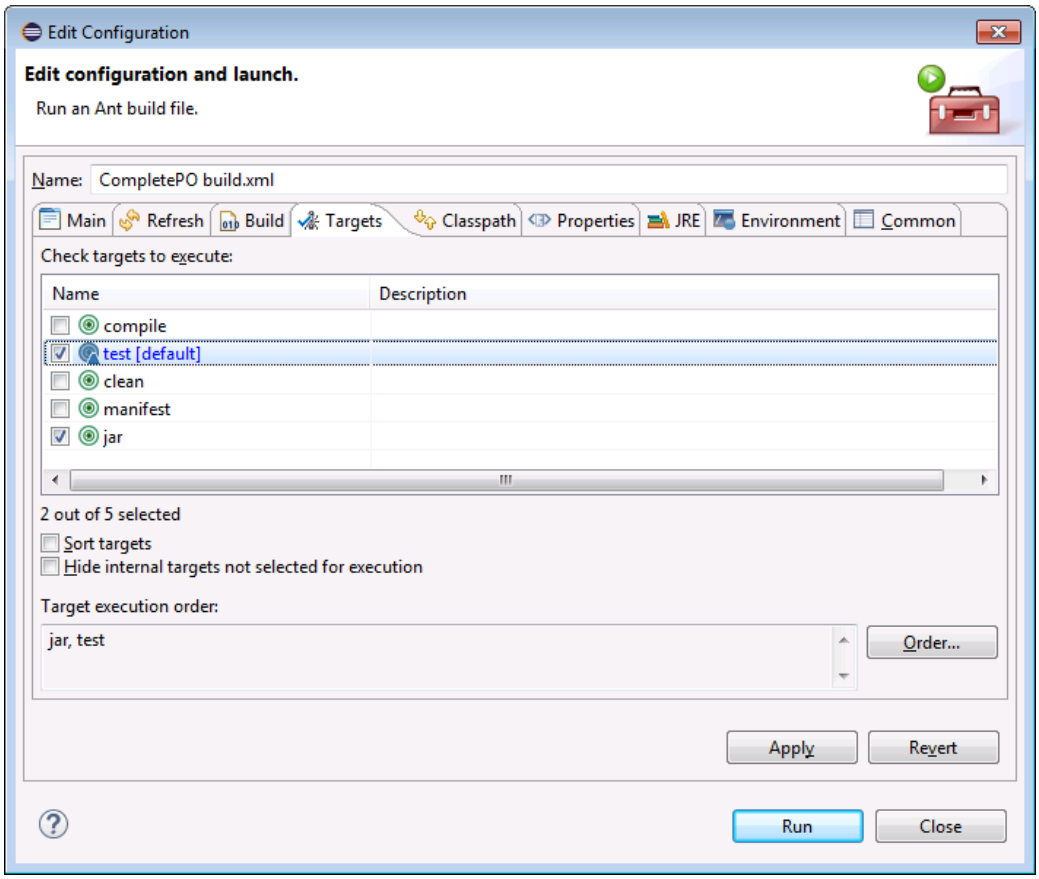
Para compilar el archivo .jar con Ant:

1. En la vista del explorador de paquetes *Package Explorer* de Eclipse haga clic en el archivo **build.xml** situado en la raíz del proyecto.
2. Haga clic en el comando **Run** del menú **Run**.



3. Aparece un cuadro de diálogo que ofrece dos opciones. Si elije la primera opción, Eclipse inicia la compilación Ant con la configuración predeterminada. Si elije la segunda opción, podrá cambiar la configuración de la compilación Ant antes de iniciarla. Por eso recomendamos usar la segunda opción.
4. En el panel *Targets* seleccione los destinos que desea incluir en la compilación Ant. En el ejemplo que nos ocupa, seleccionamos los destinos **test** y **jar**.

© 2018 Altova GmbH



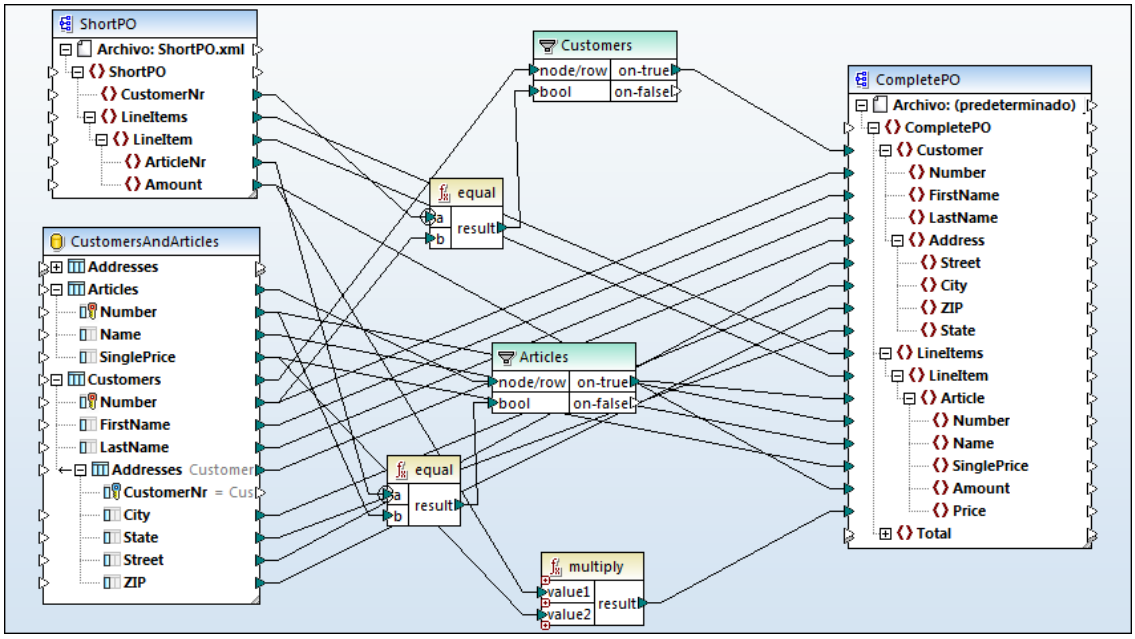
5. Haga clic en el botón **Run** para ejecutar el archivo. Eclipse ejecuta el archivo de compilación Ant y muestra en el resultado en la vista de la consola *Console*.

16.6 Integrar código generado con MapForce

El código generado con MapForce se puede integrar o se puede adaptar a otras aplicaciones, aunque el resultado de la generación de código sea una aplicación completa y totalmente funcional. Algunas situaciones en las que puede ser necesario modificar el código generado:

- Si necesita definir archivos de origen y destino personalizados para la aplicación de asignación de datos.
- Si necesita añadir códigos de control de errores personalizados.
- Si en el código C# o Java desea cambiar el tipo de datos de la entrada de la asignación mediante programación (p. ej. cambiar tipos string por tipos stream).

En esta sección explicamos cómo conseguir estos objetivos y para ello se usa la asignación de muestra **DB_CompletePO.mfd** disponible en el directorio <Documentos>\Altova\MapForce2018\MapForceExamples\.



Asignación de muestra DB_CompletePO.mfd de MapForce

Como puede ver en la imagen, la asignación de datos está compuesta por dos orígenes y un destino:

- **ShortPO.xml** es un archivo XML de origen.
- **CustomersAndArticles.mdb** es una base de datos de origen.
- **CompletePO.xml** es el archivo XML de destino.

En el código generado estos orígenes y destinos se convertirán en dos parámetros de entrada y uno de salida que se aportan al método **run** para ejecutar la asignación de datos (ver apartados de la sección). Por ahora basta con tener en cuenta estos aspectos básicos de la generación de código en MapForce:

© 2018 Altova GmbH

- El número de orígenes y destinos del diseño de asignación corresponde al número de parámetros de asignación del método **run** en el código generado.
- Si modifica el número de orígenes y destinos de la asignación, entonces deberá volver a generar el código.
- Si realiza cambios en el código generado y vuelve a generar el código en la misma ubicación, todos los cambios se sobrescribirán.

Si una asignación de datos incluye componentes de base de datos, el método **run** generado incluye el objeto de conexión de base de datos en la ubicación correspondiente. Por ejemplo, si la asignación de datos utiliza tres orígenes (contenido de texto, contenido XML y una base de datos) para asignar datos a un solo archivo de salida, MapForce generará este método **run**:

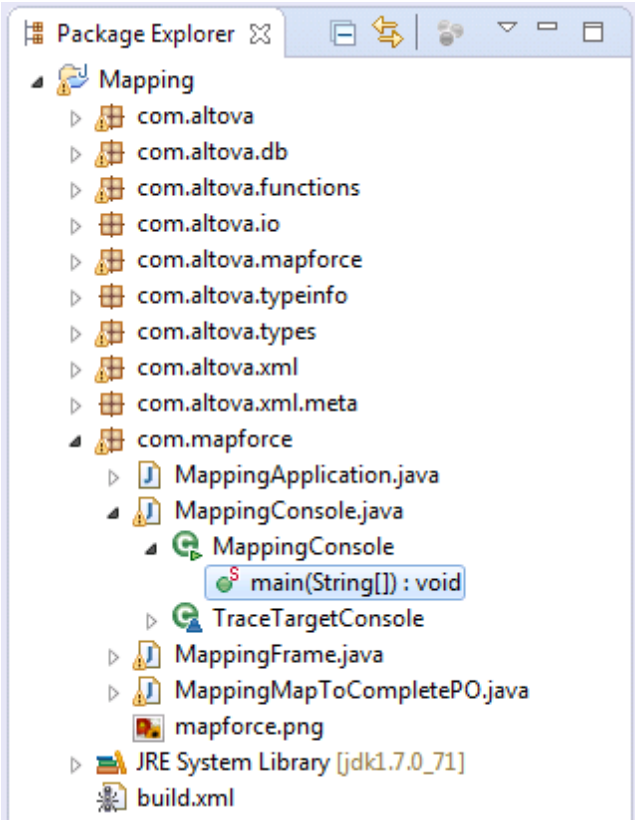
Java

```
void run(Input in1, Input in2, java.sql.Connection dbConn, Output out1);
```

El orden de los argumentos es importante. Y, como verá en los ejemplos de esta sección, cuando integre el código generado, puede modificar los parámetros **dbConn** o usar los parámetros predeterminados generados por MapForce.

16.6.1 Ejemplo: Java

El entorno IDE Java utilizado para este ejemplo es Eclipse. Para empezar debe generar código Java a partir de la asignación de ejemplo **DB_CompletePO** (situada en el directorio <Documentos>\Altova\MapForce2018\MapForceExamples\) y después importe el proyecto en Eclipse.



Aplicación Java de muestra generada con MapForce (en entorno Eclipse)

Para editar la aplicación de consola Java generada debe encontrar el método `main(String[] args)` de la aplicación generada (*imagen anterior*). Si no modificó el nombre predeterminado del paquete base antes de generar el código, este método estará en la clase `MappingConsole` del paquete `com.mapforce`. Si, por el contrario, modificó el nombre predeterminado del paquete base, entonces estará en la clase `MappingConsole` del paquete.

Para editar la aplicación de interfaz gráfica Java generada debe encontrar la posición desde la que se invoca al método `run` en la aplicación generada. Si no modificó el nombre predeterminado del paquete base antes de generar el código, el método **`run`** se invoca desde la clase `MappingFrame.java` del paquete `com.mapforce`.

La muestra de código que aparece a continuación es un extracto del método `main` de la aplicación de consola Java generada. Los orígenes y destinos de la asignación de datos aparecen resaltados en amarillo y están definidos como parámetros del método **`run`**. Como esta asignación utiliza una conexión de base de datos, el parámetro correspondiente a la conexión tiene una estructura especial. Concretamente está compuesta por la cadena de conexión (`jdbc:odbc:;DRIVER=Microsoft Access Driver (*.mdb);DBQ=CustomersAndArticles.mdb`) y por dos argumentos vacíos reservados para el **nombre de usuario** y para la **contraseña** (en texto no cifrado) si se trata de una base de datos que requiere esta información.

Recuerde que las rutas de acceso del código que aparece a continuación son relativas y no absolutas.

 © 2018 Altova GmbH

```
com.altova.io.Input ShortPO2Source =
com.altova.io.StreamInput.createInput("ShortPO.xml");
com.altova.io.Output CompletePO2Target = new
com.altova.io.FileOutput("CompletePO.xml");

MappingMapToCompletePOObject.run(
                  com.altova.db.Dbs.newConnection(
                     "jdbc:odbc:;DRIVER=Microsoft Access Driver
(*.mdb);DBQ=CustomersAndArticles.mdb",
                     "",
                     ""),
                  ShortPO2Source,
                  CompletePO2Target);
```

Para definir archivos de origen o destino de la asignación personales:

- Busque los parámetros que se pasan al método `run` y edítelos según corresponda. En el código de ejemplo anterior **`com.altova.db.Dbs.newConnection`** y **`ShortPO2Source`** constituyen la entrada de la asignación y **`CompletePO2Target`** es la salida.

Para agregar un código de control de errores adicional:

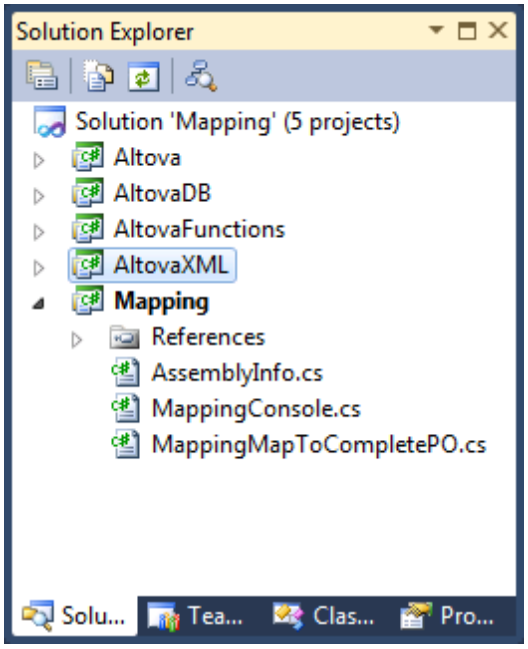
- Si se trata de una aplicación de consola Java, edite el código que aparece debajo del código **`catch`** `(Exception e)`.
- Si se trata de una aplicación de interfaz gráfica Java, edite el código que aparece debajo del código **`catch`** `(Exception ex)`.

Si necesita instrucciones para modificar el tipo de datos de los parámetros que se dan como entrada/salida de la asignación consulte el apartado Cambiar el tipo de datos de la entrada/salida de la asignación (C#, Java).

16.6.2 Ejemplo: C#

El entorno IDE utilizado para este ejemplo es Visual Studio 2010. Para empezar debe generar código C# a partir de la asignación de ejemplo **DB_CompletePO** (situada en el directorio <Documentos>\Altova\MapForce2018\MapForceExamples\) y después abra la solución en Visual Studio.

El nombre predeterminado del archivo de solución es **Mapping.sln** y su ubicación es el subdirectorio **Mapping** relativo al directorio donde guardó el código generado. Si cambió el nombre de la aplicación desde la configuración de la asignación de datos, entonces el nombre del archivo .sln también se modificó de la forma correspondiente. Por ejemplo, si cambió el nombre de la aplicación por **MiAplicación**, entonces el archivo de solución se llamará **MiAplicación.sln** y estará ubicado en el subdirectorio **MiAplicación**.



Solución C# de muestra generada con MapForce

Abra el archivo **MappingConsole.cs** y busque el método `main(String[] args)`. La muestra de código que aparece a continuación es un extracto de este método. Los orígenes y destinos de la asignación de datos aparecen resaltados en amarillo y están definidos como parámetros del método **Run**. Como esta asignación lee datos de una base de datos, también hay un parámetro de entrada que es una cadena de conexión a una base de datos. Si lo necesita, puede modificar la cadena de conexión de la base de datos.

Recuerde que las rutas de acceso del código que aparece a continuación son relativas y no absolutas.

```
Altova.IO.Input ShortPO2Source =
Altova.IO.StreamInput.createInput("ShortPO.xml");
Altova.IO.Output CompletePO2Target = new
Altova.IO.FileOutput("CompletePO.xml");

MappingMapToCompletePOObject.Run(
                                  "Provider=Microsoft.Jet.OLEDB.4.0; Data
Source=CustomersAndArticles.mdb; ",
                                  ShortPO2Source,
                                  CompletePO2Target);
```

Para definir archivos de origen o destino de la asignación personales:

- Busque los parámetros que se pasan al método **Run** y edítelos según corresponda. En el código de ejemplo anterior la entrada de la asignación es una cadena de conexión a la base de datos **CustomersAndArticles.mdb** y también `ShortPO2Source`. La salida de la asignación es `CompletePO2Target` .

 © 2018 Altova GmbH

Para agregar un código de control de errores adicional:

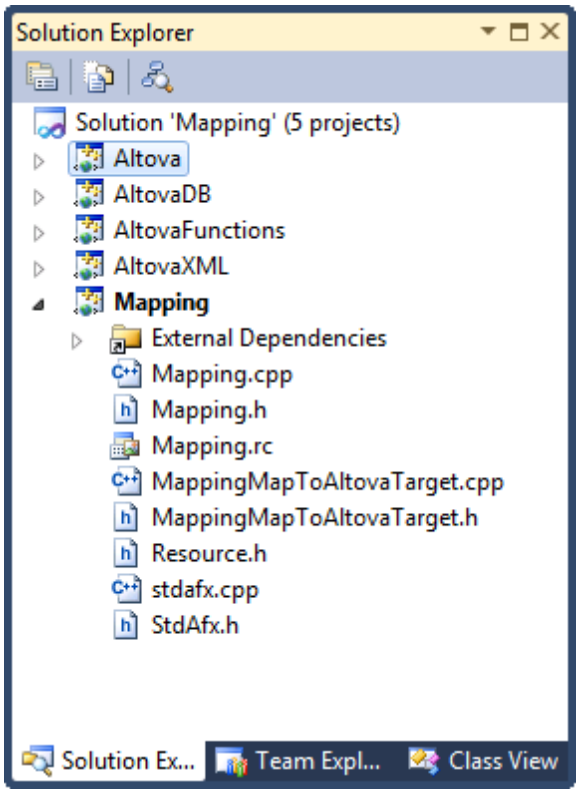
- Edite el código que aparece debajo del código **`catch`** `(Exception e)`.

Si necesita instrucciones para modificar el tipo de datos de los parámetros que se dan como entrada/salida de la asignación consulte el apartado Cambiar el tipo de datos de la entrada/salida de la asignación (C#, Java).

16.6.3 Ejemplo: C++

El entorno IDE utilizado para este ejemplo es Visual Studio 2010. Para empezar debe generar código C++ a partir de la asignación de ejemplo **DB_CompletePO** (situada en el directorio <Documentos>\Altova\MapForce2018\MapForceExamples\) y después abra la solución en Visual Studio.

El nombre predeterminado del archivo de solución es **Mapping.sln** y su ubicación es el subdirectorio **Mapping** relativo al directorio donde guardó el código generado. Si cambió el nombre de la aplicación desde la configuración de la asignación de datos, entonces el nombre del archivo .sln también se modificó de la forma correspondiente. Por ejemplo, si cambió el nombre de la aplicación por **MiAplicación**, entonces el archivo de solución se llamará **MiAplicación.sln** y estará ubicado en el subdirectorio **MiAplicación**.



Solución C++ de muestra generada con MapForce

Abra el archivo **MappingConsole.cpp** y busque el método `_tmain`. La muestra de código que aparece a continuación es un extracto de este método. Los orígenes y destinos de la asignación están definidos como parámetros del método **Run**. Como esta asignación lee datos de una base de datos, también hay un parámetro de entrada que es una cadena de conexión a una base de datos. Si lo necesita, puede modificar la cadena de conexión de la base de datos.

Recuerde que las rutas de acceso del código que aparece a continuación son relativas y no absolutas.

```
MappingMapToCompletePO MappingMapToCompletePOObject;
                    MappingMapToCompletePOObject.Run(
                            _T("Provider=Microsoft.Jet.OLEDB.4.0; Data
Source=CustomersAndArticles.mdb; "),
                            _T("ShortPO.xml"),
                            _T("CompletePO.xml"));
```

Para definir archivos de origen o destino de la asignación personales:

- Busque los parámetros que se pasan al método **Run** y edítelos según corresponda. En el código de ejemplo anterior la entrada de la asignación es una cadena de conexión a la base de datos **CustomersAndArticles.mdb** y también **_T("ShortPO.xml")**. La salida de la asignación es **_T("CompletePO.xml")** .

Para agregar un código de control de errores adicional:

- Edite el código que aparece debajo del código **catch** `(CAltovaException& e)`.

16.6.4 Cambiar el tipo de datos de la entrada/salida de la asignación (

Este apartado ofrece información detallada sobre los tipos de objetos que puede usar mediante programación si tiene pensado ejecutar asignaciones de MapForce desde aplicaciones Java o C# personalizadas.

Puede usar varios objetos de entrada y salida (como archivos, cadenas, documentos DOM, etc.) como parámetros del método **run**. El método **run** es la función más importante de las clases de asignación que se generan. Tiene un parámetro por cada componente de origen o destino estático de la asignación y un parámetro final para el componente de salida. Los componentes que procesan varios archivos no aparecen como parámetros del método **run** porque en este caso los nombres de archivo se procesan de forma dinámica dentro de la asignación misma.

Los objetos que pueden utilizarse como parámetros del método **run** están disponibles en el paquete **com.altova.io** (Java) y en el espacio de nombres Altova.IO (C#). Estas son las clases base de los objetos de entrada y salida que se generan:

C#

```
Altova.IO.Input
Altova.IO.Output
```


© 2018 Altova GmbH

Java

```
com.altova.io.Input
com.altova.io.Output
```

En la siguiente tabla puede ver los tipos de objeto que se pueden usar como parámetros de entrada/salida del método **run** (incluidos sus formatos de archivo de entrada/salida correspondientes).

Tipo de objeto	XML	Microsoft Excel*	EDI* **	FlexText*	CSV	Archivos de longitud fija
Archivos	✓	✓	✓	✓	✓	✓
Objetos de secuencia binaria	✓	✓	✓	✓	✓	✓
Cadenas	✓	–	✓	✓	✓	✓
Escritor/lector de entrada/salida (objetos de secuencia de caracteres)	✓	–	✓	✓	✓	✓
Documentos DOM	✓	–	–	–	–	–

***** Formatos compatibles con la edición MapForce Enterprise Edition solamente.
****** Incluye X12 y HL7.

Archivos

Los objetos de tipo archivo (identificados en el código por su nombre de archivo) tienen esta definición:

C#

```
Altova.IO.FileInput(string nombreArchivo)
Altova.IO.FileOutput(string nombreArchivo)
```

Java

```
com.altova.io.FileInput(String nombreArchivo)
com.altova.io.FileOutput(String nombreArchivo)
```

Objetos de secuencia binaria

Los objetos de secuencia binaria del código generado son una alternativa a los archivos físicos. El uso de objetos de secuencia binaria no ofrece ninguna ventaja en cuanto al uso de memoria. Esta es su definición:

C#

```
Altova.IO.StreamInput(System.IO.Stream secuencia)
Altova.IO.StreamOutput(System.IO.Stream secuencia)
```

Java

```
com.altova.io.StreamInput(java.io.InputStream secuencia)
com.altova.io.StreamOutput(java.io.OutputStream secuencia)
```

Notas:

- Se supone que los objetos de secuencia binaria se abren y están listos antes de llamar al método **run**.
- Por defecto el método **run** cierra la secuencia cuando se termina. Para evitar este comportamiento basta con insertar este código antes de llamar al método **run**:

Java

```
MappingMapToSomething.setCloseObjectsAfterRun(false);  // Java
```

C#

```
MappingMapToSomething.CloseObjectsAfterRun = false;   // C#
```

Cadenas

Los objetos de tipo cadena tienen esta definición:

C#

```
Altova.IO.StringInput(string contenido)
Altova.IO.StringOutput(StringBuilder contenido)
```

Java

```
com.altova.io.StringInput(String contenidoXML)
com.altova.io.StringOutput()
```

En Java **StringOutput** no toma ningún argumento. Puede acceder al contenido con:

```
// asignación de Cadena a (otra) Cadena
String MyText = "<here>is some XML text</here>";

Input input = new StringInput(MyText);
Output output = new StringOutput();

MappingMapToMyText.run(input, output);

String myTargetData = output.getString().toString();
```

El método **getString()** devuelve **StringBuffer**, de ahí que **toString()** sea necesario.

En C# **StringOutput** toma un argumento (**StringBuilder**) que debe aportarse de antemano. Puede que el argumento **StringBuilder** ya contenga datos, de modo que el resultado de la asignación se le anexa.

© 2018 Altova GmbH

No se pueden crear asignaciones entre orígenes/destinos Excel y cadenas de texto.

Escritor/lector de entrada/salida (objetos de secuencia de caracteres)

Los objetos de secuencia de caracteres tienen esta definición:

C#

```
Altova.IO.ReaderInput(System.IO.TextReader lector)
Altova.IO.WriterOutput(System.IO.TextWriter escritor)
```

Java

```
com.altova.io.ReaderInput(java.io.Reader lector)
com.altova.io.WriterOutput(java.io.Writer escritor)
```

Notas:

- Se supone que los objetos de secuencia de caracteres se abren y están listos antes de llamar al método **run**.
- Los orígenes/destinos Excel no se pueden leer desde secuencias de caracteres ni escribirse en secuencias de caracteres.
- Por defecto el método **run** cierra la secuencia cuando se termina. Para evitar este comportamiento basta con insertar este código antes de llamar al método **run**:

Java

```
MappingMapToSomething.setCloseObjectsAfterRun(false);  // Java
```

C#

```
MappingMapToSomething.CloseObjectsAfterRun = false;   // C#
```

Documentos DOM

Los documentos DOM tienen esta definición:

C#

```
Altova.IO.DocumentInput(System.Xml.XmlDocument documento)
Altova.IO.DocumentOutput(System.Xml.XmlDocument documento)
```

Java

```
com.altova.io.DocumentInput(org.w3c.dom.Document documento)
com.altova.io.DocumentOutput(org.w3c.dom.Document documento)
```

Notas:

- El documento que se pasa al constructor **DocumentOutput** como destino debe estar vacío.
- Después de llamar a **run** el documento DOM que genera el constructor de

`DocumentOutput` ya contiene datos asignados, por lo que no es necesario "guardar en documento". Una vez creada la asignación, puede manipular el documento según corresponda.

- A los documentos DOM solamente se puede asignar contenido XML.

Ejemplo

Imaginemos que desea integrar el código generado con MapForce en su aplicación Java. La asignación de MapForce está compuesta por dos archivos XML de origen y un archivo de texto de destino. Cuando genere el código de MapForce, la función `run` tendrá este aspecto:

```
void run(Input in1, Input in2, Output out1);
```

Ahora imaginemos que su aplicación exige asignar datos de un archivo local y de una secuencia binaria a una secuencia de caracteres. Como los datos los aportan otros orígenes, la aplicación deberá declarar los orígenes y los destinos:

```
String filename; // Declarar el origen de la primera entrada
Java.io.InputStream stream; // Declarar el origen de la segunda entrada
Java.io.Writer writer; // Declarar la salida como secuencia de caracteres
```

Además deben construirse estos contenedores para la función `run` generada por MapForce:

```
// se asume que com.altova.io ya está importado:
Input input1 = new FileInput(filename);
Input input2 = new StreamInput(stream);
Output output1 = new WriterOutput(writer);
```

Ahora puede llamar a la función `run` generada por MapForce:

```
MappingMapToSomething.run(input1, input2, output1);
```

El comportamiento en **C#** es prácticamente idéntico, con la diferencia de que `run` se llama `Run` y de que la secuencia .NET y las clases lectoras/escritoras se llaman de otra manera.

Con la misma técnica puede usar otros tipos de datos de entrada y salida, como cadenas o documentos DOM.

© 2018 Altova GmbH

16.7 Generar código fuente a partir de esquemas o DTD

Cuando genera código a partir de una asignación de datos, MapForce genera una aplicación completa que ejecuta todos los pasos de la asignación de datos automáticamente. Si lo desea, también puede generar bibliotecas para todos los esquemas XML utilizados en la asignación. Estos esquemas permitirán al código crear o leer con facilidad las instancias XML utilizadas o creadas por el código de la asignación.

Para generar bibliotecas para todos los esquemas XML utilizados en la asignación de datos marque la casilla *Generar clases contenedoras* en el cuadro de diálogo "Opciones" (*véase Opciones del generador de código*). La próxima vez que genere código MapForce no sólo creará la aplicación de asignación de datos sino que además generará clases contenedoras para todos los esquemas que utiliza la asignación:

C++ o C#	Java	Objetivo
Altova	`com.altova`	Biblioteca base que contiene compatibilidad con Common Language Runtime, idéntica para todos los esquemas.
AltovaXML	`com.altova.xml`	Biblioteca base que contiene compatibilidad con Common Language Runtime para XML, idéntica para todos los esquemas.
MiEsquema	*`com.MiEsquema`*	Biblioteca que contiene declaraciones generadas a partir del esquema de entrada, llamado archivo de esquema o DTD. Esta biblioteca es un contenedor DOM (W3C Document Object Model) que permite leer, modificar y crear documentos XML de forma sencilla y segura. Todos los datos están dentro del DOM y hay métodos para extraer y actualizar datos y crear datos nuevos en el DOM. El código C++ generado es compatible con Microsoft MSXML o Apache Xerces 3. La sintaxis para usar el código generado es idéntica en ambas implementaciones. El código C# generado utiliza la biblioteca System.XML estándar de .NET como implementación DOM subyacente. El código Java generado utiliza JAXP como interfaz DOM subyacente.

Cuando cree prototipos de aplicaciones a partir de esquemas XML que cambien con frecuencia, a veces será necesario generar código una y otra vez en el mismo directorio para que los cambios en el esquema se reflejen inmediatamente en el código. Recuerde que la aplicación de prueba que se genera y las bibliotecas de Altova se sobrescribirán cada vez

que genere código en el mismo directorio de destino. Por tanto, recuerde que no debe añadir código a la aplicación de prueba que se genera, sino que debe integrar las bibliotecas de Altova en el proyecto (ver *Integrar bibliotecas contenedoras de esquemas*).

Además de las bibliotecas base mencionadas en la tabla, también se generan algunas bibliotecas de ayuda. Estas bibliotecas son utilizadas por las bibliotecas base de Altova y no deben ser utilizadas en integraciones personales porque pueden sufrir cambios en versiones futuras de la aplicación.

Espacios de nombres y generación de nombres

MapForce genera clases que se corresponden con todos los elementos o tipos complejos declarados que redefinen todos los tipos complejos del esquema XML, conservando la derivación de clases tal y como definan las extensiones de tipos complejos del esquema. Cuando se trate de esquemas complejos que importen componentes de esquemas desde varios espacios de nombres distintos, MapForce conserva esta información generando los espacios de nombres C#/C++ o los paquetes Java correspondientes.

Por lo general, el generador de código intenta conservar los nombres del esquema original para los espacios de nombres, clases y miembros generados. En el lenguaje de destino los caracteres no válidos que se encuentren en los identificadores se reemplazan con "_". A los nombres que estén en conflicto con otros nombres o palabras reservadas se les anexa un número para que sean nombres únicos. Si cambia la configuración predeterminada de la plantilla SPL, puede que la generación de nombres se vea afectada.

Los espacios de nombres del esquema XML se convierten en paquetes en Java o en espacios de nombres en C#/C++, usando siempre el prefijo de espacios de nombres del esquema como espacio de nombres del código. La biblioteca completa se adjunta en un paquete o espacio de nombres derivado del nombre del archivo de esquema. Así podrá usar varias bibliotecas generadas en un programa sin que haya conflictos entre nombres.

Tipos de datos

XML Schema tiene un modelo de tipos de datos más complejo que Java, C# o C++. El generador de código convierte los tipos XML Schema integrados en tipos primitivos propios de cada lenguaje de programación o en clases derivadas con la biblioteca de Altova. Los tipos complejos y derivados definidos en el esquema se convierten en clases en la biblioteca que se genera. Las facetas de enumeración de tipos simples se convierten en constantes simbólicas.

La asignación entre tipos simples se puede configurar en la plantilla SPL.

Si los archivos de instancia XML utilizan tipos de esquema relacionados con horas y duración, estos se convierten en clases nativas de Altova en el código generado. Para más información consulte estos apartados:

- Referencia de las clases generadas (C++)
- Referencia de las clases generadas (C#)
- Referencia de las clases generadas (Java)

© 2018 Altova GmbH

Para más información sobre la conversión de tipos y otros detalles propios de cada lenguaje consulte estos apartados:

- Bibliotecas contenedoras de esquemas (C++)
- Bibliotecas contenedoras de esquemas (C#)
- Bibliotecas contenedoras de esquemas (Java)

Gestión de memoria

Una estructura DOM está compuesta por nodos que siempre son propiedad de un documento DOM concreto, aunque el nodo no forme parte del contenido del documento en ese momento. Todas las clases generadas son referencias a los nodos DOM que representan, no valores. Esto significa que si se asigna una instancia de una clase generada, el valor no se copia, sino que se crea una referencia adicional a los mismos datos.

Compatibilidad con XML Schema

Estas construcciones de XML Schema se traducen en el código:

a) Espacios de nombres XML

b) Tipos simples:

- Tipos de XML Schema integrados
- Tipos simples derivados por extensión
- Tipos simples derivados por restricción
- Facetas
- Enumeraciones
- Patrones

c) Tipos complejos

- Nodo anyType integrado
- Tipos complejos definidos por el usuario
- Derivados por extensión: se asignan a clases derivadas
- Derivados por restricción
- Contenido complejo
- Contenido simple
- Contenido mixto

Estas características de XML Schema avanzadas no son compatibles con las clases contenedoras generadas:

- Comodines: `xs:any` y `xs:anyAttribute`
- Modelos de contenido (sequence, choice, all). El compositor de nivel superior está disponible en el lenguaje SPL, pero las clases generadas no lo exigen.
- Valores predeterminados y fijos para atribuots. Están disponibles en el lenguaje SPL pero las clases generadas no los exigen ni establecen.
- Los atributos `xsi:type`, tipos abstractos. Cuando necesite escribir el atributo `xsi:type`, utilice el método `SetXsiType()` de las clases generadas.

- Tipos de unión: no todas las combinaciones son compatibles.
- Los grupos de sustitución son compatibles en parte solamente (se resuelven como "choice").
- Atributos `nillable="true"` y `xsi:nil`
- Restricciones de unicidad
- Restricciones de identidad (`key` y `keyref`)

16.7.1 Bibliotecas contenedoras de esquemas (C++)

Tipos de caracteres

El código C++ generado se puede compilar con o sin compatibilidad con Unicode. Dependiendo de la configuración elegida, los tipos `string_type` y `tstring` se definirán como `std::string` o `std::wstring` y estarán formados por caracteres estrechos o anchos. Para usar caracteres Unicode en su archivo XML que no se puedan representar con el conjunto de caracteres de 8 bits actual deberá habilitar la compatibilidad con Unicode. Deberá prestar especial atención a las macros `_T()` . Esta macro garantizan el correcto almacenamiento de las constantes de cadena (tanto si se compila para programas Unicode como para programas no Unicode).

Tipos de datos

Esta es la correspondencia predeterminada de tipos XML Schema con tipos de datos C++:

XML Schema	C++	Observaciones
`xs:string`	`string_type`	`string_type` se define como `std::string` o `std:wstring`
`xs:boolean`	`bool`	
`xs:decimal`	`double`	C++ no tiene un tipo decimal por lo que se debe usar `double`.
`xs:float, xs:double`	`double`	
`xs:integer`	`__int64`	`xs:integer` tiene un rango ilimitado que se corresponde con `__int64` para mayor eficacia.
`xs:nonNegativeInteger`	`unsigned __int64`	ver anterior
`xs:int`	`int`	
`xs:unsignedInt`	`unsigned int`	
`xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth`	`altova::DateTime`	
`xs:duration`	`altova::Duration`	
`xs:hexBinary and xs:base64Binary`	`std::vector<unsigned char>`	La codificación y decodificación de datos binarios se realiza

© 2018 Altova GmbH

XML Schema	C++	Observaciones
		automáticamente.
`xs:anySimpleType`	`string_type`	

Los tipos XML Schema que faltan en la lista son tipos derivados y corresponde al mismo tipo C++ que su correspondiente tipo base.

Clases generadas

Por cada tipo del esquema se genera una clase que contiene un miembro por cada atributo y elemento del tipo. Los miembros se llaman igual que los atributos o elementos del esquema original (en caso de conflicto se anexa un número). Para los tipos simples se generan operadores de asignación y conversión. Para los tipos simples con facetas enumeration, se pueden usar los métodos `GetEnumerationValue()` y `SetEnumerationValue(int)` junto con las constantes generadas para cada valor de enumeración. Además, el método `StaticInfo()` permite acceder a la información del esquema como uno de estos tipos:

`altova::meta::SimpleType`
`altova::meta::ComplexType`

Las clases generadas a partir de tipos complejos incluyen el método `SetXsiType()`, que permite configurar el atributo `xsi:type` del tipo. Este método puede ser muy útil a la hora de crear elementos de instancia XML de un tipo derivado.

Además de las clases para los tipos declarados en el esquema XML, se genera también una clase de documento ("CDoc"). Esta clase contiene todos los elementos raíz posibles como miembros, así como varios métodos más. Para más información sobre esta clase consulte `[SuEsquema]::[CDoc]` .

Nota: el nombre real de la clase depende del nombre del esquema .xsd.

Por cada atributo o elemento miembro de un tipo de esquema se genera una clase nueva. Para más información sobre dichas clases consulte:

`[SuEsquema]::AtributoMiembro`
`[SuEsquema]::ElementoMiembro`

Nota: el nombre real de las clases depende del nombre del atributo o elemento del esquema.

Consulte también el apartado Ejemplo: usar las bibliotecas contenedoras del esquema.

Control de errores

Los errores se notifican por medio de excepciones. Estas son las clases de excepción que están definidas en el espacio de nombres `altova`:

Clase	Clase base	Descripción

`Error`	`std::logic_error`	Error de lógica de programa interna (independiente de los datos de entrada).
`Exception`	`std::runtime_error`	Clase base para errores de tiempo de ejecución.
`InvalidArgumentsException`	`Exception`	Se llamó a un método con valores de argumento no válidos.
`ConversionException`	`Exception`	Excepción generada cuando falla una conversión de tipo.
`StringParseException`	`ConversionException`	Un valor del espacio léxico no se puede convertir en un espacio de valor.
`ValueNotRepresentableException`	`ConversionException`	Un valor del espacio de valor no se puede convertir en espacio léxico.
`OutOfRangeException`	`ConversionException`	Un valor de origen no se puede representar en el dominio de destino.
`InvalidOperationException`	`Exception`	Se intentó una operación que no es válida en el contexto dado.
`DataSourceUnavailableException`	`Exception`	Se encontró un problema al intentar cargar una instancia XML.
`DataTargetUnavailableException`	`Exception`	Se encontró un problema al intentar guardar una instancia XML.

Todas las clases de excepción contienen un texto de mensaje y un puntero que apunta a una posible excepción interna.

Método	Finalidad
`string_type message()`	Devuelve una descripción textual de la excepción.
`std::exception inner()`	Devuelve la excepción que causó esta excepción (si procede) o devuelve NULL.

Acceso a la información del esquema

La biblioteca generada permite acceder a información estática del esquema por medio de estas clases. Todos los métodos se declaran como `const`.

`altova::meta::Attribute`
`altova::meta::ComplexType`
`altova::meta::Element`
`altova::meta::SimpleType`

© 2018 Altova GmbH

Los métodos que devuelven una de las clases de metadatos devuelven un objeto NULL si no existe la propiedad correspondiente.

16.7.2 Bibliotecas contenedoras de esquemas (C#)

Esta es la correspondencia predeterminada de tipos XML Schema con tipos de datos C#:

XML Schema	C#	Observaciones
`xs:string`	`string`	
`xs:boolean`	`bool`	
`xs:decimal`	`decimal`	`xs:decimal` tiene un rango y una precisión ilimitados y se corresponde `decimal` para una mayor eficacia.
`xs:float, xs:double`	`double`	
`xs:long`	`long`	
`xs:unsignedLong`	`ulong`	
`xs:int`	`int`	
`xs:unsignedInt`	`uint`	
`xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth`	`Altova.Types.DateTime`	
`xs:duration`	`Altova.Types.Duration`	
`xs:hexBinary and xs:base64Binary`	`byte[]`	La codificación y decodificación de datos binarios se realiza automáticamente.
`xs:anySimpleType`	`string`	

Los tipos XML Schema que faltan en la lista son tipos derivados y corresponde al mismo tipo C# que su correspondiente tipo base.

Clases generadas

Por cada tipo del esquema se genera una clase que contiene un miembro por cada atributo y elemento del tipo. Los miembros se llaman igual que los atributos o elementos del esquema original (en caso de conflicto se anexa un número). Para los tipos simples se generan operadores de asignación y conversión. Para los tipos simples con facetas enumeration, se pueden usar los métodos `GetEnumerationValue()` y `SetEnumerationValue(int)` junto con las constantes

generadas para cada valor de enumeración. Además, el método `StaticInfo()` permite acceder a la información del esquema como uno de estos tipos:

`Altova.Xml.Meta.SimpleType`
`Altova.Xml.Meta.ComplexType`

Las clases generadas a partir de tipos complejos incluyen el método `SetXsiType()`, que permite configurar el atributo `xsi:type` del tipo. Este método puede ser muy útil a la hora de crear elementos de instancia XML de un tipo derivado.

Además de las clases para los tipos declarados en el esquema XML, se genera también una clase de documento ("Doc"). Esta clase contiene todos los elementos raíz posibles como miembros, así como varios métodos más. Para más información sobre esta clase consulte `[SuEsquema].[Doc]` .

Nota: el nombre real de la clase depende del nombre del esquema .xsd.

Por cada atributo o elemento miembro de un tipo de esquema se genera una clase nueva. Para más información sobre dichas clases consulte:

`[SuTipoEsquema].AtributoMiembro`
`[SuTipoEsquema].ElementoMiembro`

Nota: el nombre real de las clases depende del nombre del atributo o elemento del esquema.

Control de errores

Los errores se notifican por medio de excepciones. Estas son las clases de excepción que están definidas en el espacio de nombres `Altova`:

Clase	Clase base	Descripción
`ConversionException`	`Exception`	Excepción generada cuando falla una conversión de tipo.
`StringParseException`	`ConversionException`	Un valor del espacio léxico no se puede convertir en espacio de valor.
`DataSourceUnavailableException`	`System.Exception`	Se encontró un problema al intentar cargar una instancia XML.
`DataTargetUnavailableException`	`System.Exception`	Se encontró un problema al intentar guardar una instancia XML.

Además se suelen usar estas excepciones .NET:

Clase	Descripción
`System.Exception`	Clase base para errores de tiempo de ejecución.
`System.ArgumentException`	Se llamó a un método con valores de argumento no

© 2018 Altova GmbH

Clase	Descripción
	válidos o falló una conversión de tipo.
`System.FormatException`	Un valor del espacio léxico no se puede convertir en espacio de valor.
`System.InvalidCastException`	No se puede convertir un valor en otro tipo.
`System.OverflowException`	No se puede representar un valor de origen en el dominio de destino.

Acceso a la información del esquema

La biblioteca generada permite acceder a información estática del esquema por medio de estas clases.

`Altova.Xml.Meta.Attribute`
`Altova.Xml.Meta.ComplexType`
`Altova.Xml.Meta.Element`
`Altova.Xml.Meta.SimpleType`

Las propiedades que devuelven una de las clases de metadatos devuelven un objeto NULL si no existe la propiedad correspondiente.

16.7.3 Bibliotecas contenedoras de esquemas (Java)

Esta es la correspondencia predeterminada de tipos XML Schema con tipos de datos Java:

XML Schema	Java	Observaciones
`xs:string`	`String`	
`xs:boolean`	`boolean`	
`xs:decimal`	`java.math.BigDecimal`	
`xs:float, xs:double`	`double`	
`xs:integer`	`java.math.BigInteger`	
`xs:long`	`long`	
`xs:unsignedLong`	`java.math.BigInteger`	Java no tiene tipos sin firmar.
`xs:int`	`int`	
`xs:unsignedInt`	`long`	Java no tiene tipos sin firmar.
`xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth`	`com.altova.types.DateTime`	

XML Schema	Java	Observaciones
xs:duration	com.altova.types.Duration	
xs:hexBinary and xs:base64Binary	byte[]	La codificación y decodificación de datos binarios se realiza automáticamente.
xs:anySimpleType	string	

Los tipos XML Schema que faltan en la lista son tipos derivados y corresponde al mismo tipo Java que su correspondiente tipo base.

Clases generadas

Por cada tipo del esquema se genera una clase que contiene un miembro por cada atributo y elemento del tipo. Los miembros se llaman igual que los atributos o elementos del esquema original (en caso de conflicto se anexa un número). Para los tipos simples se generan operadores de asignación y conversión. Para los tipos simples con facetas enumeration, se pueden usar los métodos `GetEnumerationValue()` y `SetEnumerationValue(int)` junto con las constantes generadas para cada valor de enumeración. Además, el método `StaticInfo()` permite acceder a la información del esquema como uno de estos tipos:

`com.altova.xml.meta.SimpleType`
`com.altova.xml.meta.ComplexType`

Las clases generadas a partir de tipos complejos incluyen el método `SetXsiType()`, que permite configurar el atributo `xsi:type` del tipo. Este método puede ser muy útil a la hora de crear elementos de instancia XML de un tipo derivado.

Además de las clases para los tipos declarados en el esquema XML, se genera también una clase de documento ("Doc"). Esta clase contiene todos los elementos raíz posibles como miembros, así como varios métodos más. Para más información sobre esta clase consulte `com.[SuEsquema].[Doc]`.

Nota: el nombre real de la clase depende del nombre del esquema .xsd.

Por cada atributo o elemento miembro de un tipo de esquema se genera una clase nueva. Para más información sobre dichas clases consulte:

`com.[SuEsquema].[SuTipoEsquema].AtributoMiembro`
`com.[SuEsquema].[SuTipoEsquema].ElementoMiembro`

Nota: el nombre real de las clases depende del nombre del atributo o elemento del esquema.

Control de errores

Los errores se notifican por medio de excepciones. Estas son las clases de excepción que están definidas en el espacio de nombres `com.altova`:

© 2018 Altova GmbH

Clase	Clase base	Descripción
`SourceInstanceUnvailableException`	`Exception`	Se encontró un problema al intentar cargar una instancia XML.
`TargetInstanceUnavailableException`	`Exception`	Se encontró un problema al intentar guardar una instancia XML.

Además se suelen usar estas excepciones Java:

Clase	Descripción
`java.lang.Error`	Error de lógica de programa interna (independiente de los datos de entrada).
`java.lang.Exception`	Clase base para errores de tiempo de ejecución.
`java.lang.IllegalArgumentsException`	Se llamó a un método con valores de argumento no válidos o falló una conversión de tipo.
`java.lang.ArithmeticException`	Excepción generada cuando falla una conversión de tipo numérico.

Acceso a la información del esquema

La biblioteca generada permite acceder a información estática del esquema por medio de estas clases.

`com.altova.xml.meta.Attribute`
`com.altova.xml.meta.ComplexType`
`com.altova.xml.meta.Element`
`com.altova.xml.meta.SimpleType`

Las propiedades que devuelven una de las clases de metadatos devuelven un objeto NULL si no existe la propiedad correspondiente.

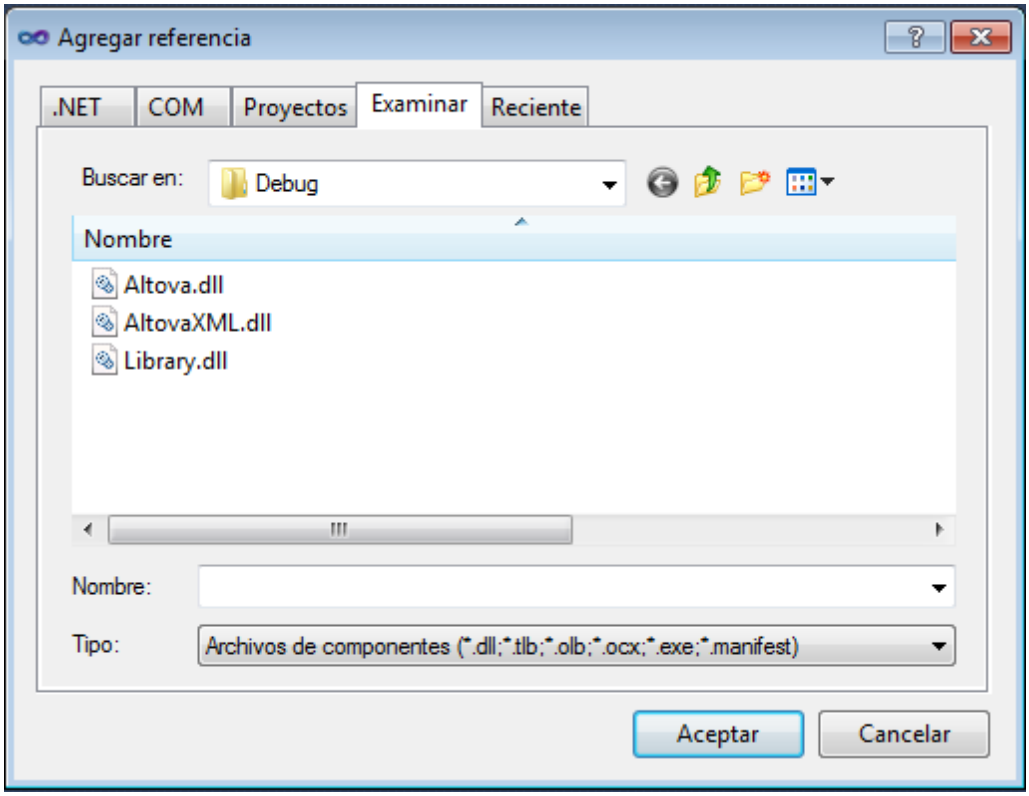
16.7.4 Integrar las bibliotecas contenedoras de esquemas

Para usar las bibliotecas de Altova en su proyecto basta con hacerles referencia o incluirlas en el proyecto (ver más abajo).

C#

Para integrar las bibliotecas de Altova en un proyecto C# actual:

1. Después de generar código a partir de un esquema con MapForce (p. ej. a partir de **YourSchema.xsd**), compile la solución **YourSchema.sln** que acaba de generar en Visual Studio. Esta solución está en una carpeta de proyecto que se llama igual que el esquema.
2. Haga clic con el botón derecho en el proyecto en Visual Studio y seleccione **Agregar referencia**.
3. En la pestaña *Examinar* busque las bibliotecas **Altova.dll**, **AltovaXML.dll** y **YourSchema.dll**, ubicadas en el directorio de salida de los proyectos generados (p. ej. **bin\Debug**).



C++

La forma más sencilla de integrar bibliotecas en un proyecto C++ es agregar a la solución los archivos de proyecto generados. Por ejemplo, imagine que generó código a partir de un esquema llamado **Library.xsd** y que seleccionó el directorio de destino **c:\codegen\cpp\library**. Las bibliotecas generadas en este caso están en estas rutas de acceso:

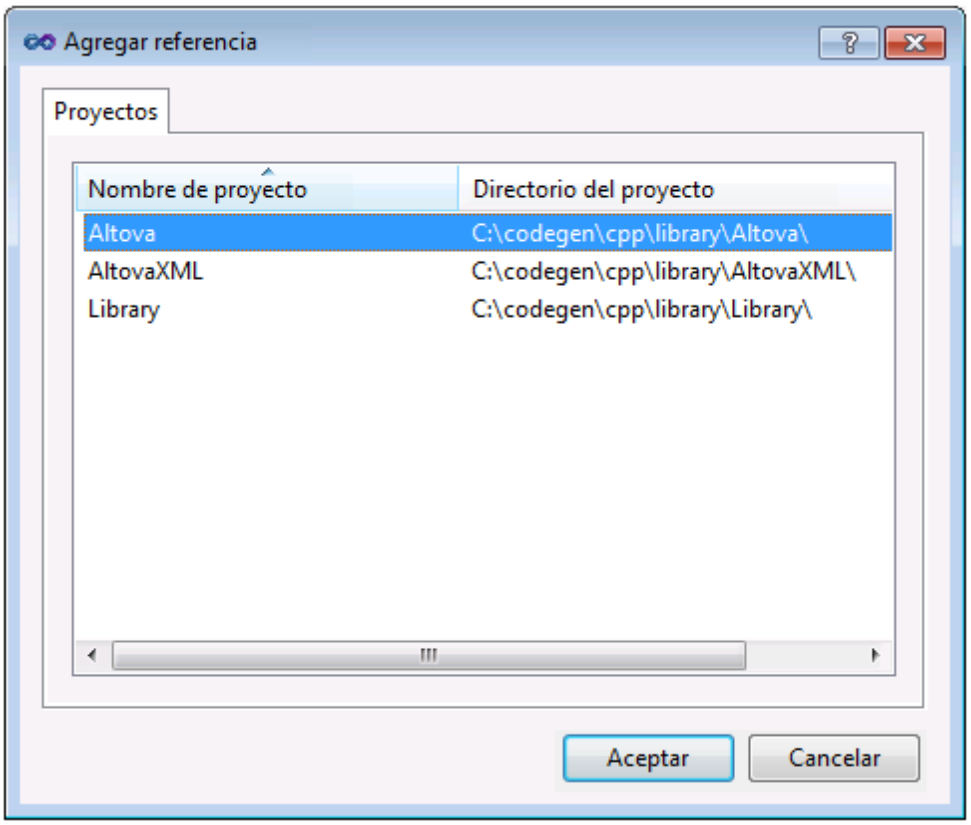
- c:\codegen\cpp\library\Altova.vcxproj
- c:\codegen\cpp\library\AltovaXML\AltovaXML.vcxproj
- c:\codegen\cpp\library\Library.vcxproj

Primero abra la solución **c:\codegen\cpp\library\Library.sln** y compílela en Visual Studio.

Después abra su solución de Visual Studio (en Visual Studio 2010, por ejemplo). Haga clic con el botón derecho en la solución y seleccione **Agregar | Proyecto existente**. Ahora añada uno a uno los archivos de proyecto enumerados más arriba. Tenga paciencia mientras Visual Studio

© 2018 Altova GmbH

analiza los archivos. Después haga clic con el botón derecho en el proyecto y seleccione **Propiedades**. En el cuadro de diálogo "Páginas de propiedades" seleccione **Propiedades comunes | Framework y referencias** y después haga clic en **Agregar referencia nueva**. Añada estos proyectos: *Altova*, *AltovaXML* y *Library*.



Si necesita ayuda a la hora de usar las funciones de una biblioteca personal consulte la documentación de MSDN:

- Si tiene pensado generar bibliotecas estáticas, consulte https://msdn.microsoft.com/es-es/library/ms235627(v=vs.100).aspx
- Si tiene pensado generar bibliotecas dinámicas, consulte https://msdn.microsoft.com/es-es/library/ms235636(v=vs.100).aspx

La opción de generar bibliotecas estáticas o dinámicas se encuentra entre las opciones de generación de código (*véase Opciones del generador de código*).

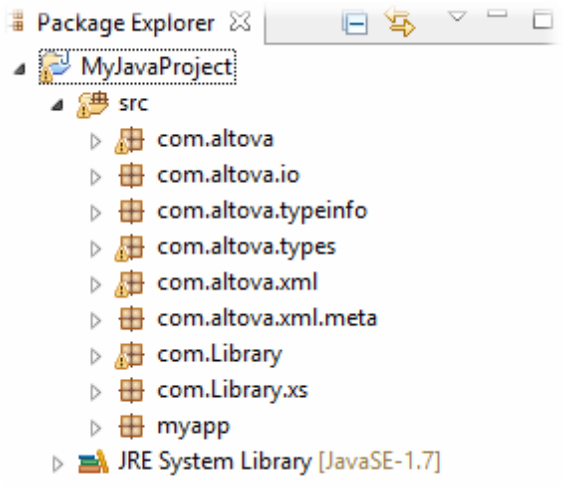
Java

Puede integrar sus paquetes de Altova en su proyecto Java copiando el directorio **com** del código generado en el directorio que almacena los paquetes fuente de su proyecto Java (p. ej. **C:\Workspace\MyJavaProject\src**). Por ejemplo, imagine que generó código en el directorio **c:\codegen\java\library**. Las clases de Altova generadas en este caso están en **c:\codegen\java\library\com**.

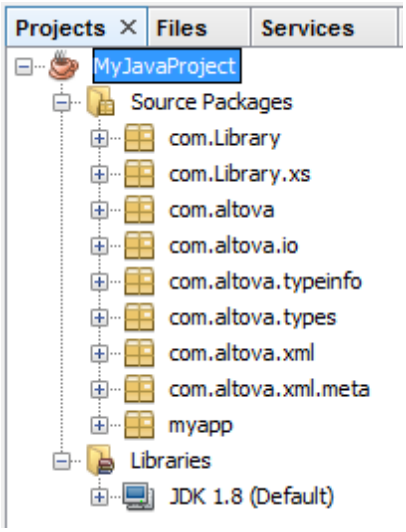
Tras copiar las bibliotecas actualice el proyecto (en Eclipse seleccione el proyecto en el

explorador de paquetes y pulse **F5**. En NetBeans IDE 8.0 seleccione el comando de menú **Source | Scan for External Changes**).

Una vez terminada la operación de copia, los paquetes de Altova aparecerán en el explorador de paquetes (en Eclipse) o bajo *Source Packages* en el panel *Proyects* (en NetBeans IDE).



Paquetes de Altova en Eclipse 4.4



Paquetes de Altova en NetBeans IDE 8.0.2

16.7.5 Ejemplo: usar las bibliotecas contenedoras de esquemas

Este ejemplo explica cómo utilizar las bibliotecas contenedoras de esquemas que se generan para escribir o leer mediante programación documentos XML conformes con el esquema. Antes de usar el código de muestra, recomendamos dedicar un tiempo a comprender la estructura del

esquema de ejemplo. Este esquema es necesario para generar las bibliotecas de código que se utilizan en este ejemplo y si conoce y comprende su estructura podrá trabajar más fácilmente con el código que se genera cuando necesite adaptarlo a sus requisitos personales.

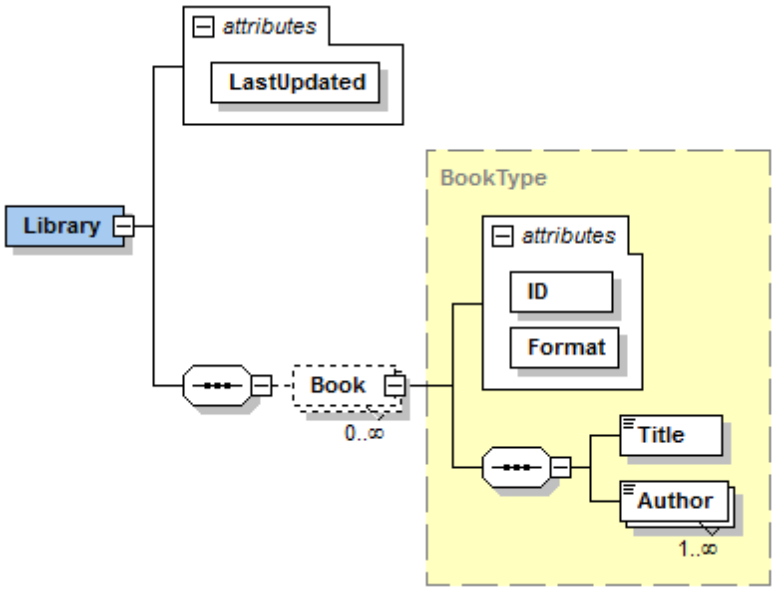
16.7.5.1 Ejemplo de esquema

El esquema que se utiliza en este ejemplo describe una biblioteca de libros. Su definición completa aparece más abajo. Si quiere obtener los mismos resultados que se obtienen en esta sección, guarde este fragmento de código como **Library.xsd**. Este esquema se necesita para generar las bibliotecas de código utilizadas más adelante en esta sección.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns="http://www.nanonull.com/LibrarySample" xmlns:xs="http://
www.w3.org/2001/XMLSchema" targetNamespace="http://www.nanonull.com/
LibrarySample" elementFormDefault="qualified"
attributeFormDefault="unqualified">
   <xs:element name="Library">
      <xs:complexType>
         <xs:sequence>
            <xs:element name="Book" type="BookType" minOccurs="0"
maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:attribute name="LastUpdated" type="xs:dateTime"/>
      </xs:complexType>
   </xs:element>
   <xs:complexType name="BookType">
      <xs:sequence>
         <xs:element name="Title" type="xs:string"/>
         <xs:element name="Author" type="xs:string" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="ID" type="xs:integer" use="required"/>
      <xs:attribute name="Format" type="BookFormatType" use="required"/>
   </xs:complexType>
   <xs:complexType name="DictionaryType">
      <xs:complexContent>
         <xs:extension base="BookType">
            <xs:sequence>
               <xs:element name="FromLang" type="xs:string"/>
               <xs:element name="ToLang" type="xs:string"/>
            </xs:sequence>
         </xs:extension>
      </xs:complexContent>
   </xs:complexType>
   <xs:simpleType name="BookFormatType">
      <xs:restriction base="xs:string">
         <xs:enumeration value="Hardcover"/>
         <xs:enumeration value="Paperback"/>
         <xs:enumeration value="Audiobook"/>
         <xs:enumeration value="E-book"/>
      </xs:restriction>
   </xs:simpleType>
```

```
</xs:schema>
```

Library es un elemento raíz de un `complexType` cuya representación gráfica aparece a continuación (tomada de la vista Esquema de XMLSpy):

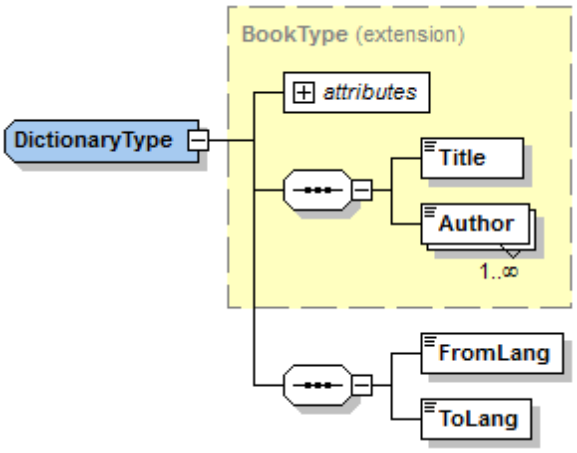


Como puede ver, la biblioteca tiene un atributo **LastUpdated** (definido comos `xs:dateTime`) y almacena una secuencia de libros. Cada libro es un `xs:complexType` y tiene dos atributos: **ID** (definido como `xs:integer`) y **Format**. El formato de los libros puede ser tapa dura (`hardcover`), bolsillo (`paperback`), audiolibro (`audiobook`) o libro electrónico (`e-book`). En el esquema el atributo **Format** está definido como `xs:simpleType` que usa una enumeración de los mencionados valores.

Además cada libro tiene un elemento **Title** (definido como `xs:string`) y como mínimo un elemento **Author** (definido como `xs:string`).

La biblioteca también puede incluir libros que son diccionarios. Estos tienen el tipo `DictionaryType`, que se deriva por extensión del tipo `BookType`. Es decir, un diccionario hereda todos los atributos y elementos de un libro y además toma dos elementos más: **FromLang** y **ToLang** (*imagen siguiente*).

© 2018 Altova GmbH



Los elementos **FromLang** y **ToLang** almacenan el idioma de partida y de destino del diccionario.

Veamos a continuación una instancia XML válida de acuerdo con el esquema anterior (siempre y cuando esté en el mismo directorio que el esquema):

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://
www.nanonull.com/LibrarySample" xsi:schemaLocation="http://www.nanonull.com/
LibrarySample Library.xsd" LastUpdated="2016-02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/
LibrarySample" xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

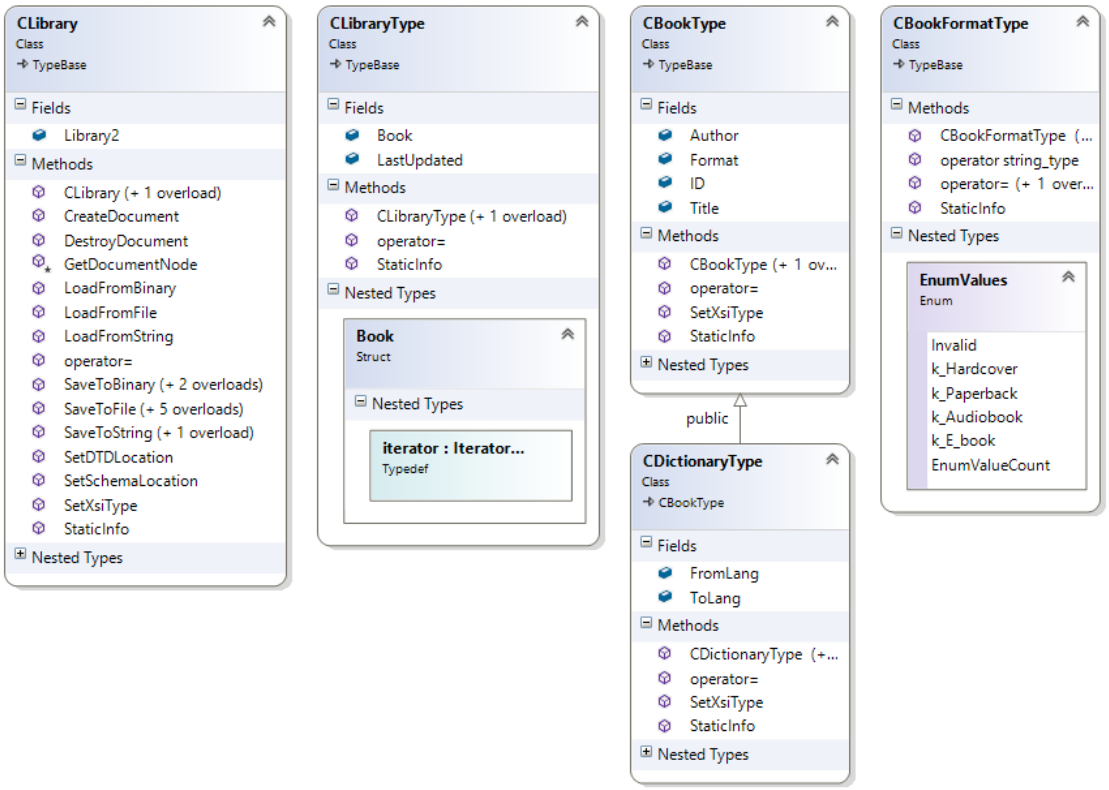
En el apartado siguiente explicamos cómo leer o escribir en dicho archivo mediante programación. Para empezar es necesario generar el código contenedor del esquema (siguiendo las instrucciones del apartado Generar código a partir de esquemas XML o documentos DTD).

16.7.5.2 Leer y escribir documentos XML (C++)

Tras generarse el código a partir del esquema de **Library** (*véase Ejemplo de esquema*), se crea una aplicación de prueba C++ junto con varias bibliotecas Altova secundarias.

Información sobre las bibliotecas C++ generadas

En este diagrama puede ver algunas de las clases más importantes del código generado.



La clase central del código generado es la clase `CLibrary`, que representa el documento XML. Dicha clase se genera por cada esquema y su nombre depende del nombre del archivo de esquema (en nuestro ejemplo es **Library.xsd**). Como puede verse en el diagrama, esta clase aporta métodos para cargar documentos desde archivos, secuencias binarias o cadenas de texto (o para guardar documentos en archivos, secuencias y cadenas). Para ver una lista de los miembros que expone esta clase consulte la referencia de la clase (`[SuEsquema]::[CDoc]`).

El campo `Library2` de la clase `CLibrary` representa la raíz real del documento. El número que aparece al final sirve para evitar conflictos entre los nombres de clase. **Library** es un elemento del archivo XML, de modo que en el código C++ tiene una clase plantilla como tipo (`MemberElement`). La clase plantilla expone métodos y propiedades para interactuar con el elemento Library. En general, cada atributo y cada elemento de un tipo del esquema recibe un tipo en el código generado con las clases plantilla `MemberAttribute` y `MemberElement` respectivamente. Para más información consulte la referencia de las clases `[SuEsquema]::AtributoMiembro` y `[SuEsquema]::ElementoMiembro`.

La clase `CLibraryType` se genera a partir del tipo complejo del esquema con el mismo nombre (*véase Bibliotecas contenedoras de esquemas (C++)*). Observe que la clase `CLibraryType` contiene los campos `Book` y `LastUpdated`. Según la lógica mencionada previamente, estos campos corresponden al elemento **Book** y al atributo **LastUpdated** del esquema y permiten manipular elementos y atributos del documento XML de instancia mediante programación (operaciones anexar, eliminar, etc.).

© 2018 Altova GmbH

Como **DictionaryType** es un tipo complejo derivado de **BookType** esta relación también se refleja en las clases generadas. Como puede verse en el diagrama, la clase `CDictionaryType` hereda la clase `CBookType`.

Si su esquema XML define tipos simples como enumeraciones, los valores enumerados están disponibles como valores `Enum` en el código generado. En el esquema utilizado en este ejemplo, el formato de los libros puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de un `Enum` que es miembro de la clase `CBookFormatType`.

Escribir un documento XML

1. En Visual Studio abra la solución **LibraryTest.sln** que se generó a partir del esquema **Library** mencionado anteriormente.

Cuando cree prototipos de aplicaciones a partir de esquemas XML que cambien con frecuencia, a veces será necesario generar código una y otra vez en el mismo directorio para que los cambios en el esquema se reflejen inmediatamente en el código. Recuerde que la aplicación de prueba que se genera y las bibliotecas de Altova se sobrescribirán cada vez que genere código en el mismo directorio de destino. Por tanto, recuerde que no debe añadir código a la aplicación de prueba que se genera, sino que debe integrar las bibliotecas de Altova en el proyecto (ver *Integrar bibliotecas contenedoras de esquemas*).

2. En el explorador de soluciones abra el archivo **LibraryTest.cpp** y edite el método `Example()` como se indica a continuación:

```
#include <ctime> // obligatorio para obtener hora actual
using namespace Library; // obligatorio para trabajar con bibliotecas
de Altova

void Example()
{
    // Crear un documento XML nuevo vacío
    CLibrary libDoc = CLibrary::CreateDocument();

    // Crear el elemento raíz <Library> y agregarlo al documento
    CLibraryType lib = libDoc.Library2.append();

    // Obtener la hora actual y establecer el atributo "LastUpdated"
usando clases de Altova
    time_t t = time(NULL);
    struct tm * now = localtime( & t );
    altova::DateTime dt = altova::DateTime(now->tm_year + 1900, now-
>tm_mon + 1, now->tm_mday, now->tm_hour, now->tm_min, now->tm_sec);
    lib.LastUpdated = dt;

    // Crear un nuevo <Book> y agregarlo a la biblioteca
    CBookType book = lib.Book.append();

    // Establecer el atributo "ID" del libro
    book.ID = 1;
```

```
    // Establecer el atributo "Format" del <Book> usando una constante
de enumeración
    book.Format.SetEnumerationValue( CBookFormatType::k_Paperback );

    // Agregar los elementos <Title> y <Author> y establecer valores
    book.Title.append() = _T("The XML Spy Handbook");
    book.Author.append() = _T("Altova");

    // Anexar un diccionario (libro de tipo derivado) y rellenar sus
atributos y elementos
    CDictionaryType dictionary =
CDictionaryType(lib.Book.append().GetNode());
    dictionary.ID = 2;
    dictionary.Format.SetEnumerationValue( CBookFormatType::k_E_book);
    dictionary.Title.append() = _T("English-German Dictionary");
    dictionary.Author.append() = _T("John Doe");
    dictionary.FromLang.append() = _T("English");
    dictionary.ToLang.append() = _T("German");

    // Como diccionario es un tipo derivado, establecer el atributo
xsi:type del elemento book
    dictionary.SetXsiType();

    // De forma opcional establecer la ubicación del esquema
    libDoc.SetSchemaLocation(_T("Library.xsd"));

    // Guardar el documento XML en un archivo con la codificación
predeterminada(UTF-8)
    // "true" hace que el archivo tenga formato pretty-print
    libDoc.SaveToFile(_T("GeneratedLibrary.xml"), true);

    // Destruir el documento
    libDoc.DestroyDocument();
}
```

3. Pulse **F5** para iniciar la depuración. Si el código se ejecuta correctamente, el archivo **GeneratedLibrary.xml** se crea en el directorio de salida de la solución.

Leer un documento XML

1. Abra la solución **LibraryTest.sln** en Visual Studio.
2. Guarde el código que aparece a continuación como **Library1.xml** en un directorio que el código de programa pueda leer (p. ej. en el mismo directorio que **LibraryTest.sln**).

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://
www.nanonull.com/LibrarySample" xsi:schemaLocation="http://www.nanonull.com/
LibrarySample Library.xsd" LastUpdated="2016-02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
```


© 2018 Altova GmbH

```
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/
LibrarySample" xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. En el explorador de soluciones abra el archivo **LibraryTest.cpp** y edite el método `Example()` como se indica a continuación.

```
using namespace Library;
void Example()
{
	// Cargar documento XML
	CLibrary libDoc = CLibrary::LoadFromFile(_T("Library1.xml"));

	// Obtener el primer y único elemento raíz <Library>
	CLibraryType lib = libDoc.Library2.first();

	// Compruebe si existe un elemento:
	if (!lib.Book.exists())
	{
		tcout << "This library is empty." << std::endl;
		return;
	}

	// iteración: por cada <Book>...
	for (Iterator<CBookType> itBook = lib.Book.all(); itBook; ++itBook)
	{
		// valores de salida del atributo ISBN y del primer y único
elemento title
		tcout << "ID: " << itBook->ID << std::endl;
		tcout << "Title: " << tstring(itBook->Title.first()) <<
std::endl;

		// leer y compara un valor de enumeración
		if (itBook->Format.GetEnumerationValue() ==
CBookFormatType::k_Paperback)
			tcout << "This is a paperback book." << std::endl;

		// por cada <Author>...
		for (CBookType::Author::iterator itAuthor = itBook->Author.all();
itAuthor; ++itAuthor)
			tcout << "Author: " << tstring(itAuthor) << std::endl;

		// alternativa: usar recuento e índice
		for (unsigned int j = 0; j < itBook->Author.count(); ++j)
			tcout << "Author: " << tstring(itBook->Author[j]) <<
std::endl;
	}

	// Destruir el documento
```

```
        libDoc.DestroyDocument();
}
```

4. Pulse **F5** para iniciar la depuración.

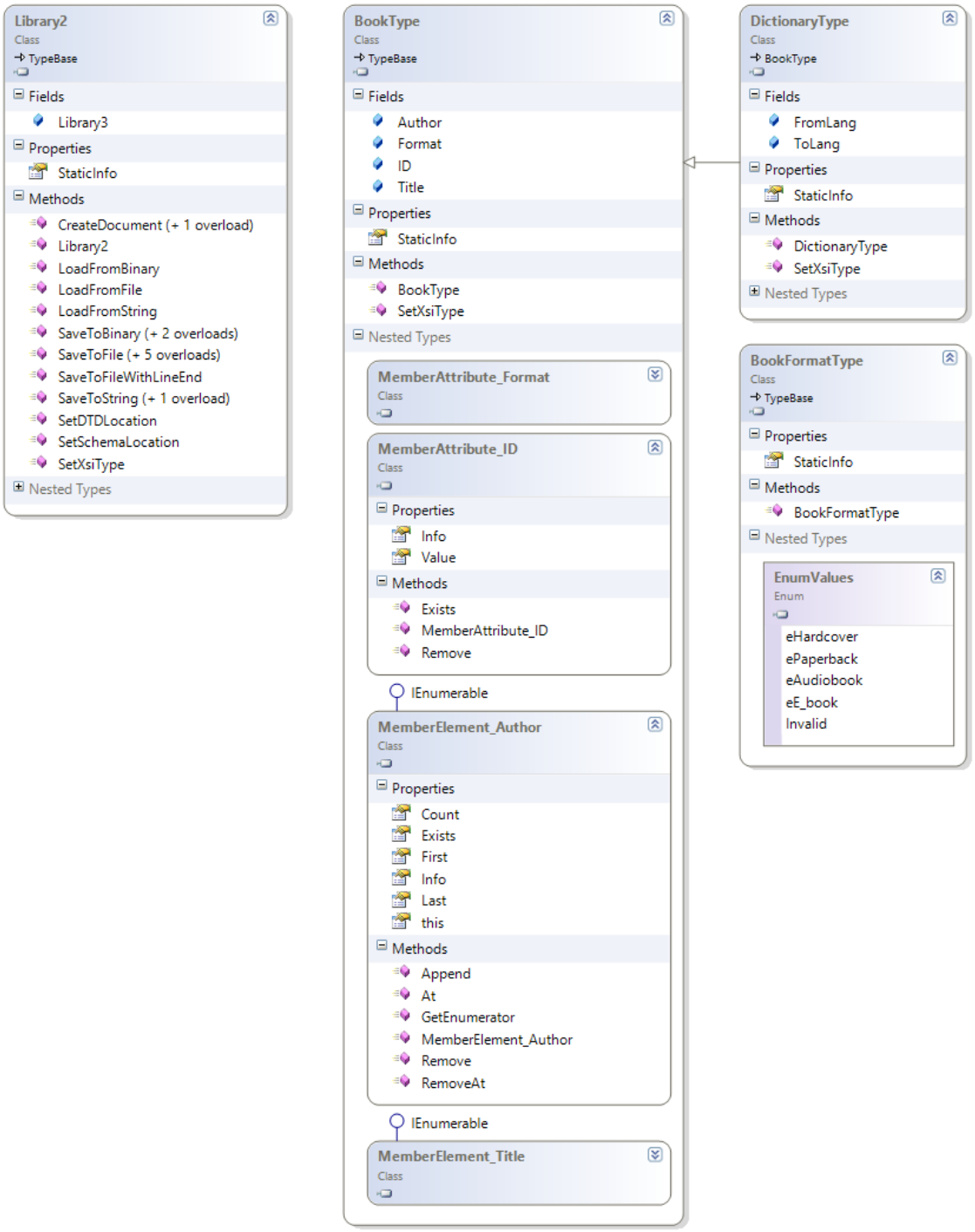
16.7.5.3 Leer y escribir documentos XML (C#)

Tras generarse el código a partir del esquema de **Library** (*véase Ejemplo de esquema*), se crea una aplicación de prueba C# junto con varias bibliotecas Altova secundarias.

Información sobre las bibliotecas C# generadas

En este diagrama puede ver algunas de las clases más importantes del código generado.

© 2018 Altova GmbH



La clase central del código generado es la clase `Library2`, que representa el documento XML. Dicha clase se genera por cada esquema y su nombre depende del nombre del archivo de esquema (en nuestro ejemplo es **Library.xsd**). Observe que esta clase se llama `Library2` para evitar conflictos entre los nombres de espacio de nombres. Como puede verse en el diagrama, esta clase aporta métodos para cargar documentos desde archivos, secuencias binarias o cadenas de texto (o para guardar documentos en archivos, secuencias y cadenas). Para ver una descripción de esta clase consulte la referencia de la clase (`[SuEsquema].[Doc]`).

El miembro `Library3` de la clase `Library2` representa la raíz real del documento. De nuevo podemos observar que el número que aparece al final sirve para evitar conflictos entre los nombres de clase.

De acuerdo con las reglas de generación de código mencionadas en el apartado Bibliotecas contenedoras de esquemas (C#), se generan clases miembro por cada atributo y por cada elemento de un tipo. En el código generado el nombre de dichas clases miembros va precedido por el prefijo `MemberAttribute_` y `MemberElement_` respectivamente. En el diagrama anterior puede ver ejemplos de dichas clases: `MemberAttribute_ID` y `MemberElement_Author`, que se generan a partir del elemento **Author** y del atributo **ID** de un libro respectivamente. Dichas clases permiten manipular los correspondientes elementos y atributos del documento XML de instancia mediante programación (operaciones anexar, eliminar, establecer valor, etc.). Para más información consulte la referencia de las clases `[SuTipoEsquema].MemberAttribute` y `[SuTipoEsquema].MemberElement`.

Como en el esquema **DictionaryType** es un tipo complejo derivado de **BookType**, esta relación también se refleja en las clases generadas. Como puede verse en el diagrama, la clase `DictionaryType` hereda la clase `BookType`.

Si su esquema XML define tipos simples como enumeraciones, los valores enumerados están disponibles como valores `Enum` en el código generado. En el esquema utilizado en este ejemplo, el formato de los libros puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de un `Enum` que es miembro de la clase `BookFormatType`.

Escribir un documento XML

1. En Visual Studio abra la solución **LibraryTest.sln** que se generó a partir del esquema Library mencionado anteriormente.

Cuando cree prototipos de aplicaciones a partir de esquemas XML que cambien con frecuencia, a veces será necesario generar código una y otra vez en el mismo directorio para que los cambios en el esquema se reflejen inmediatamente en el código. Recuerde que la aplicación de prueba que se genera y las bibliotecas de Altova se sobrescribirán cada vez que genere código en el mismo directorio de destino. Por tanto, recuerde que no debe añadir código a la aplicación de prueba que se genera, sino que debe integrar las bibliotecas de Altova en el proyecto (ver *Integrar bibliotecas contenedoras de esquemas*).

2. En el explorador de soluciones abra el archivo **LibraryTest.cs** y edite el método `Example()` como se indica a continuación:

```
protected static void Example()
{
    // Crear una biblioteca XML nueva
    Library2 doc = Library2.CreateDocument();
    // Anexar el elemento raíz
    LibraryType root = doc.Library3.Append();
```


© 2018 Altova GmbH

```
            // Crear la fecha de generación de biblioteca con ayuda de la
clase DateTime de Altova
            Altova.Types.DateTime dt = new
Altova.Types.DateTime(System.DateTime.Now);
            // Anexar la fecha a la raíz
            root.LastUpdated.Value = dt;

            // Agregar un libro nuevo
            BookType book = root.Book.Append();
            // Establecer el valor del atributo ID
            book.ID.Value = 1;
            // Establecer el formato del libro (enumeración)
            book.Format.EnumerationValue =
BookFormatType.EnumValues.eHardcover;
            // Establecer los elementos Title y Author
            book.Title.Append().Value = "The XMLSpy Handbook";
            book.Author.Append().Value = "Altova";

            // Anexar un diccionario (libro de tipo derivado) y rellenar
sus atributos y elementos
            DictionaryType dictionary = new
DictionaryType(root.Book.Append().Node);
            dictionary.ID.Value = 2;
            dictionary.Title.Append().Value = "English-German Dictionary";
            dictionary.Format.EnumerationValue =
BookFormatType.EnumValues.eE_book;
            dictionary.Author.Append().Value = "John Doe";
            dictionary.FromLang.Append().Value = "English";
            dictionary.ToLang.Append().Value = "German";
            // Como es un tipo derivado, establecer el atributo xsi:type
del elemento book
            dictionary.SetXsiType();

            // De forma opcional, establecer la ubicación del esquema
(ajustar la ruta de acceso
            // si el esquema no está en la misma carpeta que el archivo de
instancia que se generó)
            doc.SetSchemaLocation("Library.xsd");

            // Guardar el documento XML con la opción pretty print
habilitada
            doc.SaveToFile("GeneratedLibrary.xml", true);
        }
```

3. Pulse **F5** para iniciar la depuración. Si el código se ejecuta correctamente, el archivo **GeneratedLibrary.xml** se crea en el directorio de salida de la solución (que suele ser **bin/Debug**).

Leer un documento XML

1. Abra la solución **LibraryTest.sln** en Visual Studio.
2. Guarde el código que aparece a continuación en un archivo llamado **Library.xml** en el directorio de salida del proyecto (predeterminado: **bin/Debug**). Este es el archivo que leerá el código de programa.

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://
www.nanonull.com/LibrarySample" xsi:schemaLocation="http://www.nanonull.com/
LibrarySample Library.xsd" LastUpdated="2016-02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/
LibrarySample" xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

3. En el explorador de soluciones abra el archivo **LibraryTest.cs** y edite el método `Example()` como se indica a continuación:

```
       protected static void Example()
        {
            // Cargar el archivo XML en una instancia nueva de Library
            Library2 doc = Library2.LoadFromFile("Library.xml");
            // Obtener el elemento raíz
            LibraryType root = doc.Library3.First;

            // Leer la fecha de generación de Library
            Altova.Types.DateTime dt = root.LastUpdated.Value;
            string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
            Console.WriteLine("The library generation date is: " +
dt_as_string);

            // Iteración: por cada <Book>...
            foreach (BookType book in root.Book)
            {
                // Valores de salida del atributo ID y del (primer y único)
elemento title
                Console.WriteLine("ID:    " + book.ID.Value);
                Console.WriteLine("Title: " + book.Title.First.Value);

                // Leer y comparar un valor de enumeración
                if (book.Format.EnumerationValue ==
BookFormatType.EnumValues.ePaperback)
                    Console.WriteLine("This is a paperback book.");

                // Iteración: por cada <Author>
                foreach (xs.stringType author in book.Author)
                    Console.WriteLine("Author: " + author.Value);

                // Determinar si se trata de un libro de tipo derivado
                if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
```


© 2018 Altova GmbH

```
                {
                    // Buscar el valor del atributo xsi:type
                    string xsiTypeValue =
book.Node.Attributes.GetNamedItem("xsi:type").Value;
                    // Obtener el URI de espacio de nombres
                    // y el prefijo de consulta de este espacio de nombres
                    string namespaceUri = book.Node.NamespaceURI;
                    string prefix =
book.Node.GetPrefixOfNamespace(namespaceUri);

                    // si este libro tiene DictionaryType
                    if (namespaceUri == "http://www.nanonull.com/
LibrarySample" && xsiTypeValue.Equals(prefix + ":DictionaryType"))
                    {
                        // generar campos adicionales
                        DictionaryType dictionary = new
DictionaryType(book.Node);
                        Console.WriteLine("Language from: " +
dictionary.FromLang.First.Value);
                        Console.WriteLine("Language to: " +
dictionary.ToLang.First.Value);
                    }
                    else
                    {
                        throw new Exception("Unexpected book type");
                    }
                }
            }

            Console.ReadLine();
        }
```

4. Pulse **F5** para iniciar la depuración. Si el código se ejecuta correctamente, el código de programa leerá el archivo **GeneratedLibrary.xml** y su contenido aparecerá en el resultado de la consola.

Leer y escribir elementos y atributos

El acceso a los atributos y elementos se consigue con la propiedad `Value` de la clase de elemento o atributo miembro que se genera. Por ejemplo:

```
// Valores de salida del atributo ID y del (primer y único) elemento title
Console.WriteLine("ID:    " + book.ID.Value);
Console.WriteLine("Title: " + book.Title.First.Value);
```

Para obtener el valor del elemento **Title** en este ejemplo concreto también se utilizó el método `First()`. Esto se debe a que este es el primer (y único) elemento **Title** de un libro. Cuando necesite seleccionar un elemento concreto de una lista por medio del índice, utilice el método `At()`.

La clase que se genera por cada elemento miembro de un tipo implementa la interfaz estándar `System.Collections.IEnumerable`. Esto permite recorrer varios elementos del mismo tipo. En

este ejemplo concreto puede recorrer todos los libros de un objeto Library como se puede ver a continuación:

```
// Iteración: por cada <Book>...
foreach (BookType book in root.Book)
{
    // introduzca aquí su código...
}
```

Para agregar un elemento nuevo utilice el método `Append()`. Por ejemplo, a continuación se anexa el elemento raíz al documento:

```
// Anexar el elemento raíz a la biblioteca
LibraryType root = doc.Library3.Append();
```

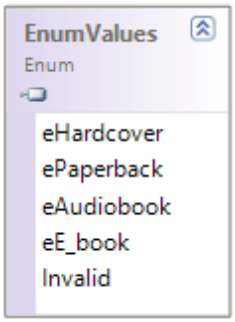
Puede establecer el valor de un atributo (como ID) de la siguiente manera:

```
// Establecer el valor del atributo ID
book.ID.Value = 1;
```

Para más información consulte la referencia de las clases `[SuTipoEsquema].MemberAttribute` y `[SuTipoEsquema].MemberElement`.

Leer y escribir valores de enumeración

Si el esquema XML define tipos simples como enumeraciones, los valores enumerados estarán disponibles como valores `Enum` en el código generado. En el esquema utilizado en este ejemplo el formato de un libro puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de una `Enum`:



Para asignar valores de enumeración a un objeto utilice código como este:

```
// Establecer el formato del libro (enumeración)
book.Format.EnumerationValue = BookFormatType.EnumValues.eHardcover;
```

Puede leer dichos valores de enumeración desde documento XML de instancia de la siguiente manera:

© 2018 Altova GmbH

```
// Leer y comparar un valor de enumeración
if (book.Format.EnumerationValue == BookFormatType.EnumValues.ePaperback)
Console.WriteLine("This is a paperback book.");
```

Cuando la condición IF no sea suficiente, cree un modificador para determinar cada valor de enumeración y procesarlos según corresponda en cada caso.

Trabajar con tipos xs:dateTime y xs:duration

Si el esquema desde el que se genera código utiliza tipos de hora y duración como `xs:dateTime` o `xs:duration`, estos tipos se convierten en clases nativas de Altova en el código generado. Por tanto, para escribir un valor de fecha o duración en el documento XML debe seguir estas instrucciones:

1. Construya un objeto `Altova.Types.DateTime` o `Altova.Types.Duration` (bien desde `System.DateTime` o utilizando partes como horas y minutos como en `Altova.Types.DateTime` y `Altova.Types.Duration`).
2. Establezca el objeto como valor del elemento o atributo que necesita. Por ejemplo:

```
// Crear la fecha de generación de la biblioteca usando la clase DateTime de
Altova
Altova.Types.DateTime dt = new Altova.Types.DateTime(System.DateTime.Now);
// Anexar la fecha a la raíz
root.LastUpdated.Value = dt;
```

Para leer una fecha o duración de un documento XML:

1. Declare el valor de elemento (o atributo) como objeto `Altova.Types.DateTime` o `Altova.Types.Duration`.
2. Aplique formato al elemento o atributo que necesita. Por ejemplo:

```
// Leer la fecha de generación de la biblioteca
Altova.Types.DateTime dt = root.LastUpdated.Value;
string dt_as_string = dt.ToString(DateTimeFormat.W3_dateTime);
Console.WriteLine("The library generation date is: " + dt_as_string);
```

Para más información consulte la referencia de las clases `Altova.Types.DateTime` y `Altova.Types.Duration`.

Trabajar con tipos derivados

Si su esquema XML define tipos derivados puede conservar la derivación de tipos en los documentos XML que cree o cargue mediante programación. Tomando el esquema utilizado en este ejemplo, el fragmento de código que aparece a continuación explica cómo se crea un libro nuevo de tipo derivado `DictionaryType`:

```
// Anexar un diccionario (libro de tipo derivado) y rellenar sus atributos y
elementos
DictionaryType dictionary = new DictionaryType(root.Book.Append().Node);
```

```
dictionary.ID.Value = 2;
dictionary.Title.Append().Value = "English-German Dictionary";
dictionary.Author.Append().Value = "John Doe";
dictionary.FromLanguage.Append().Value = "English";
dictionary.ToLanguage.Append().Value = "German";

// Como es un tipo derivado, asegúrese de establecer el atributo xsi:type del
elemento book
dictionary.SetXsiType();
```

Recuerde que es importante establecer el atributo `xsi:type` del libro recién creado. Esto garantiza que el esquema interprete correctamente el tipo de libro a la hora de validar el documento XML.

El fragmento de código que aparece a continuación identifica un libro de tipo derivado `DictionaryType` en la instancia XML que se carga. Primero se busca el valor del atributo `xsi:type` del nodo book. Si el URI de espacio de nombres de este nodo es `http://www.nanonull.com/LibrarySample` y el prefijo de consulta URI y el tipo coinciden con el valor del atributo `xsi:type`, entonces sabemos que se trata de un diccionario.

```
      // Determinar si este libro es de tipo derivado
      if (book.Node.Attributes.GetNamedItem("xsi:type") != null)
      {
          // Buscar el valor del atributo xsi:type
          string xsiTypeValue =
book.Node.Attributes.GetNamedItem("xsi:type").Value;
          // Obtener el URI de espacio de nombres y el prefijo de consulta de
este espacio de nombres
          string namespaceUri = book.Node.NamespaceURI;
          string prefix = book.Node.GetPrefixOfNamespace(namespaceUri);

          // Si este libro tiene DictionaryType
          if (namespaceUri == "http://www.nanonull.com/LibrarySample" &&
xsiTypeValue.Equals(prefix + ":DictionaryType"))
          {
             // generar campos adicionales
             DictionaryType dictionary = new DictionaryType(book.Node);
             Console.WriteLine("Language from: " +
dictionary.FromLang.First.Value);
             Console.WriteLine("Language to: " +
dictionary.ToLang.First.Value);
          }
          else
          {
              throw new Exception("Unexpected book type");
          }
      }
```

16.7.5.4 Leer y escribir documentos XML (Java)

Tras generarse el código a partir del esquema de **Library** (*véase* *Ejemplo de esquema*), se crea un proyecto de prueba Java junto con varias bibliotecas Altova secundarias.

© 2018 Altova GmbH

Información sobre las bibliotecas Java generadas

En este diagrama puede ver algunas de las clases más importantes del código generado.



La clase central del código generado es la clase `Library2`, que representa el documento XML. Dicha clase se genera por cada esquema y su nombre depende del nombre del archivo de esquema (en nuestro ejemplo es **Library.xsd**). Observe que esta clase se llama `Library2` para evitar conflictos entre los nombres de espacio de nombres. Como puede verse en el diagrama,

 © 2018 Altova GmbH

esta clase aporta métodos para cargar documentos desde archivos, secuencias binarias o cadenas de texto (o para guardar documentos en archivos, secuencias y cadenas). Para ver una descripción de esta clase consulte la referencia de la clase (`com.[SuEsquema].[Doc]`).

El miembro `Library3` de la clase `Library2` representa la raíz real del documento. De nuevo podemos observar que el número que aparece al final sirve para evitar conflictos entre los nombres de clase.

De acuerdo con las reglas de generación de código mencionadas en el apartado Bibliotecas contenedoras de esquemas (Java), se generan clases miembro por cada atributo y por cada elemento de un tipo. En el código generado el nombre de dichas clases miembros va precedido por el prefijo `MemberAttribute_` y `MemberElement_` respectivamente. En el diagrama anterior puede ver ejemplos de dichas clases: `MemberAttribute_ID` y `MemberElement_Author`, que se generan a partir del elemento **Author** y del atributo **ID** de un libro respectivamente. Dichas clases permiten manipular los correspondientes elementos y atributos del documento XML de instancia mediante programación (operaciones anexar, eliminar, establecer valor, etc.). Para más información consulte la referencia de las clases `com.[SuEsquema].[SuTipoEsquema].MemberAttribute` y `com.[SuEsquema].[SuTipoEsquema].MemberElement`.

Como en el esquema **DictionaryType** es un tipo complejo derivado de **BookType**, esta relación también se refleja en las clases generadas. Como puede verse en el diagrama, la clase `DictionaryType` hereda la clase `BookType`.

Si su esquema XML define tipos simples como enumeraciones, los valores enumerados están disponibles como valores `Enum` en el código generado. En el esquema utilizado en este ejemplo, el formato de los libros puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de un `Enum` que es miembro de la clase `BookFormatType`.

Escribir un documento XML

1. En el menú **File** de Eclipse haga clic en el comando **Import** y seleccione **Existing Projects into Workspace** (Proyectos en el espacio de trabajo). Después haga clic en **Next** para continuar.
2. Junto a **Select root directory** (Seleccionar directorio raíz) haga clic en el botón **Browse**. Después seleccione el directorio donde generó el código Java y haga clic en **Finish** para finalizar.
3. En el explorador de paquetes de Eclipse expanda el paquete **com.LibraryTest** y abra el archivo **LibraryTest.java**.

> Cuando cree prototipos de aplicaciones a partir de esquemas XML que cambien con frecuencia, a veces será necesario generar código una y otra vez en el mismo directorio para que los cambios en el esquema se reflejen inmediatamente en el código. Recuerde que la aplicación de prueba que se genera y las bibliotecas de Altova se sobrescribirán cada vez que genere código en el mismo directorio de destino. Por tanto, recuerde que no debe añadir código a la aplicación de prueba que se genera, sino que debe integrar las bibliotecas de Altova en el proyecto (ver *Integrar bibliotecas contenedoras de esquemas*).

4. Edite el método `Example()` como se indica a continuación:

```
protected static void example() throws Exception {
      // crear un documento XML nuevo vacío
      Library2 libDoc = Library2.createDocument();

      // crear el elemento raíz <Library> y agregarlo al documento
      LibraryType lib = libDoc.Library3.append();

      // establecer el atributo "LastUpdated"
      com.altova.types.DateTime dt = new
com.altova.types.DateTime(DateTime.now());
      lib.LastUpdated.setValue(dt);

      // crear un nuevo <Book> y rellenar sus elementos y atributos
      BookType book = lib.Book.append();
      book.ID.setValue(java.math.BigInteger.valueOf(1));
      book.Format.setEnumerationValue( BookFormatType.EPAPERBACK );
      book.Title.append().setValue("The XML Spy Handbook");
      book.Author.append().setValue("Altova");

      // crear un diccionario (libro de tipo derivado) y rellenar sus
elementos y atributos
      DictionaryType dict = new
DictionaryType(lib.Book.append().getNode());
      dict.ID.setValue(java.math.BigInteger.valueOf(2));
      dict.Title.append().setValue("English-German Dictionary");
      dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
      dict.Author.append().setValue("John Doe");
      dict.FromLang.append().setValue("English");
      dict.ToLang.append().setValue("German");
      dict.setXsiType();

      // establecer la ubicación del esquema (opcional)
      libDoc.setSchemaLocation("Library.xsd");

      // guardar el documento XML en un archivo con la codificación
predeterminada (UTF-8). el valor "true" aplica formato pretty-print al
archivo.
      libDoc.saveToFile("Library1.xml", true);
   }
```

5. Compile y ejecute el proyecto Java. Si el código se ejecuta correctamente, se crea el archivo **Library1.xml** en el directorio del proyecto.

Leer un documento XML

1. En el menú **File** de Eclipse haga clic en el comando **Import** y seleccione **Existing Projects into Workspace** (Proyectos en el espacio de trabajo). Después haga clic en **Next** para continuar.
2. Junto a **Select root directory** (Seleccionar directorio raíz) haga clic en el botón **Browse**. Después seleccione el directorio donde generó el código Java y haga clic en **Finish** para finalizar.
3. Guarde el código que aparece a continuación en un archivo llamado **Library1.xml** en un directorio local (deberá referirse a la ruta de acceso del archivo **Library1.xml** desde el

© 2018 Altova GmbH

código).

```
<?xml version="1.0" encoding="utf-8"?>
<Library xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://
www.nanonull.com/LibrarySample" xsi:schemaLocation="http://www.nanonull.com/
LibrarySample Library.xsd" LastUpdated="2016-02-03T17:10:08.4977404">
   <Book ID="1" Format="E-book">
      <Title>The XMLSpy Handbook</Title>
      <Author>Altova</Author>
   </Book>
   <Book ID="2" Format="Paperback" xmlns:n1="http://www.nanonull.com/
LibrarySample" xsi:type="n1:DictionaryType">
      <Title>English-German Dictionary</Title>
      <Author>John Doe</Author>
      <FromLang>English</FromLang>
      <ToLang>German</ToLang>
   </Book>
</Library>
```

4. En el explorador de paquetes de Eclipse expanda el paquete **com.LibraryTest** y abra el archivo **LibraryTest.java**.
5. Edite el método `Example()` como se indica a continuación:

```
   protected static void example() throws Exception {
      // cargar documento XML desde una ruta de acceso (ajustar la ruta si
procede)
      Library2 libDoc = Library2.loadFromFile("Library1.xml");

      // obtener el primer (y único) elemento raíz <Library>
      LibraryType lib = libDoc.Library3.first();

      // comprobar si existe un elemento:
      if (!lib.Book.exists()) {
         System.out.println("This library is empty.");
         return;
      }

      // leer el tipo DateTime del esquema
      com.altova.types.DateTime dt = lib.LastUpdated.getValue();
      System.out.println("The library was last updated on: " +
dt.toDateString());

      // iteración: por cada <Book>...
      for (java.util.Iterator itBook = lib.Book.iterator(); itBook.hasNext();)
 {
         BookType book = (BookType) itBook.next();
         // valores de salida del atributo ID y del (primer y único) elemento
title
         System.out.println("ID: " + book.ID.getValue());
         System.out.println("Title: " + book.Title.first().getValue());

         // leer y comparar un valor de enumeración
```

```
         if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
            System.out.println("This is a paperback book.");

         // por cada <Author>...
         for (java.util.Iterator itAuthor = book.Author.iterator(); itAuthor
               .hasNext();)
            System.out.println("Author: " + ((com.Library.xs.stringType)
itAuthor.next()).getValue());

         // buscar el tipo derivado de este libro
         // usando el valor del atributo xsi:type y usando DOM
         org.w3c.dom.Node bookNode = book.getNode();
         if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
            // obtener el valor del atributo xsi:type
            String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

            // obtener el URI de espacio de nombres y prefijo de consulta de
este espacio de nombres
            String namespaceUri = bookNode.getNamespaceURI();
            String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

            // si xsi:type coincide con el URI de espacio de nombres y con el
tipo del nodo book
            if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                  && (  xsiTypeValue.equals(lookupPrefix +
":DictionaryType" )))    {
               // ...entonces se trata de un libro de tipo derivado
(diccionario)
               DictionaryType dictionary = new
DictionaryType(   book.getNode());
               // generar el valor de los elementos "FromLang" y "ToLang"
elements
               System.out.println("From language: " +
dictionary.FromLang.first().getValue());
               System.out.println("To language: " +
dictionary.ToLang.first().getValue());
            }
            else
            {
               // emitir un error
               throw new java.lang.Error("This book has an unknown type.");
            }
         }
      }
   }
```

6. Compile y ejecute el proyecto Java. Si el código se ejecuta correctamente, el código de programa leerá el archivo **Library1.xml** y su contenido aparecerá en la vista de consola *Console*.

© 2018 Altova GmbH

Leer y escribir elementos y atributos

El acceso a los atributos y elementos se consigue con el método `getValue()` de la clase de elemento o atributo miembro que se genera. Por ejemplo:

```
// valores de salida del atributo ID y del (primer y único) elemento title
System.out.println("ID: " + book.ID.getValue());
System.out.println("Title: " + book.Title.first().getValue());
```

Para obtener el valor del elemento **Title** en este ejemplo concreto también se utilizó el método `first()`. Esto se debe a que este es el primer (y único) elemento **Title** de un libro. Cuando necesite seleccionar un elemento concreto de una lista por medio del índice, utilice el método `at()`.

Para recorrer varios elementos utilice la iteración basada en índice o `java.util.Iterator`. Por ejemplo, puede recorrer los libros de la biblioteca de la siguiente manera:

```
// iteración basada en índice
for (int j = 0; j < lib.Book.count(); ++j ) {
   // introduzca aquí su código
}

// iteración alternativa usando java.util.Iterator
for (java.util.Iterator itBook = lib.Book.iterator(); itBook.hasNext();) {

      // introduzca aquí su código
   }
```

Para agregar un elemento nuevo utilice el método `append()`. Por ejemplo, este código anexa un elemento raíz **Library** vacío al documento:

```
// crear el elemento raíz <Library> y agregarlo al documento
LibraryType lib = libDoc.Library3.append();
```

Tras anexar el elemento, puede establecer el valor de cualquiera de sus elementos o atributos con ayuda del método `setValue()`.

```
// establecer el valor del elemento Title
book.Title.append().setValue("The XML Spy Handbook");
// establecer el valor del atributo ID
book.ID.setValue(java.math.BigInteger.valueOf(1));
```

Para más información consulte la referencia de las clases `com.[SuEsquema].[SuTipoEsquema].MemberAttribute` y `com.[SuEsquema].[SuTipoEsquema].MemberElement`.

Leer y escribir valores de enumeración

Si el esquema XML define tipos simples como enumeraciones, los valores enumerados estarán disponibles como valores `Enum` en el código generado. En el esquema utilizado en este ejemplo el

formato de un libro puede ser tapa dura, bolsillo, libro electrónico y audiolibro. Por tanto, en el código generado estos valores estarán disponibles a través de una `Enum` (véase el diagrama de la clase `BookFormatType` que aparece más arriba). Para asignar valores de enumeración a un objeto utilice código parecido a este:

```
// establecer un valor de enumeración
book.Format.setEnumerationValue( BookFormatType.EPAPERBACK );
```

Dichos valores de enumeración se pueden leer desde documentos XML de instancia de la siguiente manera:

```
// leer un valor de enumeración
if (book.Format.getEnumerationValue() == BookFormatType.EPAPERBACK)
        System.out.println("This is a paperback book."
```

Cuando la condición IF no sea suficiente, cree un modificador para determinar cada valor de enumeración y procesarlo según corresponda en cada caso.

Trabajar con tipos xs:dateTime y xs:duration

Si el esquema desde el que se genera código utiliza tipos de hora y duración como `xs:dateTime` o `xs:duration`, estos tipos se convierten en clases nativas de Altova en el código generado. Por tanto, para escribir un valor de fecha o duración en el documento XML debe seguir estas instrucciones:

1. Construya un objeto `com.altova.types.DateTime` o `com.altova.types.Duration`.
2. Establezca el objeto como valor del elemento o atributo que necesita. Por ejemplo:

```
// establecer el valor de un atributo de tipo DateTime
com.altova.types.DateTime dt = new com.altova.types.DateTime(DateTime.now());
lib.LastUpdated.setValue(dt);
```

Para leer una fecha o duración de un documento XML:

1. Declare el valor de elemento (o atributo) como objeto `com.altova.types.DateTime` o `com.altova.types.Duration`.
2. Aplique formato al elemento o atributo que necesita. Por ejemplo:

```
// leer un tipo DateTime
com.altova.types.DateTime dt = lib.LastUpdated.getValue();
   System.out.println("The library was last updated on: " +
dt.toDateString());
```

Para más información consulte la referencia de las clases `com.altova.types.DateTime` y `com.altova.types.Duration`.

© 2018 Altova GmbH

Trabajar con tipos derivados

Si su esquema XML define tipos derivados puede conservar la derivación de tipos en los documentos XML que cree o cargue mediante programación. Tomando el esquema utilizado en este ejemplo, el fragmento de código que aparece a continuación explica cómo se crea un libro nuevo de tipo derivado `DictionaryType`:

```
// crear un diccionario (libro de tipo derivado) y rellenar sus elementos y
atributos
DictionaryType dict = new DictionaryType(lib.Book.append().getNode());
dict.ID.setValue(java.math.BigInteger.valueOf(2));
dict.Title.append().setValue("English-German Dictionary");
dict.Format.setEnumerationValue(BookFormatType.EE_BOOK);
dict.Author.append().setValue("John Doe");
dict.FromLang.append().setValue("English");
dict.ToLang.append().setValue("German");
dict.setXsiType();
```

Recuerde que es importante establecer el atributo `xsi:type` del libro recién creado. Esto garantiza que el esquema interprete correctamente el tipo de libro a la hora de validar el documento XML.

El fragmento de código que aparece a continuación identifica un libro de tipo derivado `DictionaryType` en la instancia XML que se carga. Primero se busca el valor del atributo `xsi:type` del nodo book. Si el URI de espacio de nombres de este nodo es `http://www.nanonull.com/LibrarySample` y el prefijo de consulta URI y el tipo coinciden con el valor del atributo `xsi:type`, entonces sabemos que se trata de un diccionario.

```
        // buscar el tipo derivado de este libro
        // usando el valor del atributo xsi:type y usando DOM
        org.w3c.dom.Node bookNode = book.getNode();
        if (bookNode.getAttributes().getNamedItem("xsi:type") != null) {
           // obtener el valor del atributo xsi:type
           String xsiTypeValue =
bookNode.getAttributes().getNamedItem("xsi:type").getNodeValue();

           // obtener el URI de espacio de nombres y el prefijo de consulta
del nodo book
           String namespaceUri = bookNode.getNamespaceURI();
           String lookupPrefix = bookNode.lookupPrefix(namespaceUri);

           // si xsi:type coincide con el URI de espacio de nombres y el tipo
del nodo book
           if (namespaceUri == "http://www.nanonull.com/LibrarySample"
                 && (  xsiTypeValue.equals(lookupPrefix +
":DictionaryType" )))    {
              // ...entonces se trata de un libro de tipo derivado
(diccionario)
              DictionaryType dictionary = new
DictionaryType(   book.getNode());
              // generar el valor de los elementos "FromLang" y "ToLang"
elements
```

```
               System.out.println("From language: " +
dictionary.FromLang.first().getValue());
               System.out.println("To language: " +
dictionary.ToLang.first().getValue());
            }
            else
            {
               // emitir un error
               throw new java.lang.Error("This book has an unknown type.");
            }
         }
```


© 2018 Altova GmbH

16.8 Referencia de las clases generadas (C++)

Esta sección describe las clases C++ que se generan con MapForce a partir de una DTD o esquema XML (*véase* *Generar código a partir de esquemas XML o DTD*). Puede integrar estas clases en su código para leer, modificar y escribir documentos XML.

Nota: el código generado puede incluir otras clases compatibles que no se enumeran en este apartado y que pueden cambiar en el futuro.

16.8.1 altova::DateTime

Esta clase permite procesar atributos o elementos XML de tipo hora y fecha (p. ej. `xs:dateTime`).

Constructores

Nombre	Descripción
`DateTime()`	Inicializa una instancia nueva de la clase `DateTime` en la medianoche 12:00:00 del 1 de enero de 0001.
`DateTime(__int64 value, short timezone)`	Inicializa una instancia nueva de la clase `DateTime`. El parámetro `value` representa el número de ciclos (intervalos de 100 nanosegundos) que han pasado desde la medianoche 12:00:00 del 1 de enero de 0001.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto y segundo dados como argumento.
`DateTime(int year, unsigned char month, unsigned char day, unsigned char hour, unsigned char minute, double second, short timezone)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto, segundo y zona horaria dados como argumento. La zona horaria se expresa en minutos y puede ser positiva o negativa. Por ejemplo, la zona horaria "UTC-01:00" se expresa como "-60".

Métodos

Nombre	Descripción
`unsigned char Day() const`	Devuelve el día del mes del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 1 y 31.

Nombre	Descripción
`int DayOfYear() const`	Devuelve el día del año del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 1 y 366.
`bool HasTimezone() const`	Devuelve el valor binario **true** si el objeto `DateTime` actual tiene definida una zona horaria. De lo contrario devuelve **false**.
`unsigned char Hour() const`	Devuelve la hora del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 0 y 23.
`static bool IsLeapYear(int year)`	Devuelve el valor binario **true** si el año de la clase `DateTime` es un año bisiesto. De lo contrario devuelve **false**.
`unsigned char Minute() const`	Devuelve el minuto del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 0 y 59.
`unsigned char Month() const`	Devuelve el mes del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 1 y 12.
`__int64 NormalizedValue() const`	Devuelve el valor del objeto `DateTime` expresado como UTC.
`double Second() const`	Devuelve el segundo del objeto `DateTime` actual. Los valores devueltos están comprendidos entre 0 y 59.
`void SetTimezone(short tz)`	Establece la zona horaria del objeto `DateTime` actual en el valor de zona horaria dado como argumento. El argumento **tz** se expresa en minutos y puede ser positivo o negativo.
`short Timezone() const`	Devuelve la zona horaria en minutos del objeto `DateTime` actual. Antes de usar este método asegúrese de que el objeto realmente tiene una zona horaria (llamando al método `HasTimezone()`).
`__int64 Value() const`	Devuelve el valor del objeto `DateTime` expresado en el número de ciclos (intervalos de 100 nanosegundos) que han pasado desde la medianoche 12:00:00 del 1 de enero de 0001.
`int Weekday() const`	Devuelve el día de la semana del objeto `DateTime` actual como entero. Los valores están comprendidos entre 0 y 6, siendo 0 el lunes (ISO-8601).
`int Weeknumber() const`	Devuelve el número de la semana del año del objeto `DateTime` actual. Los valores devueltos son conformes a ISO-8601.
`int WeekOfMonth() const`	Devuelve el número de la semana del mes del objeto `DateTime` actual. Los valores devueltos son conformes a ISO-8601.

© 2018 Altova GmbH

Nombre	Descripción
`int Year() const`	Devuelve el año del objeto `DateTime` actual.

Ejemplo

```
void Example()
{
	// inicializar una nueva instancia de DateTime en la medianoche
12:00:00 del 1 de enero de 0001
	altova::DateTime dt1 = altova::DateTime();

	// inicializar una nueva instancia de DateTime usando el año, mes, día,
hora, minuto y segundo
	altova::DateTime dt2 = altova::DateTime(2015, 11, 10, 9, 8, 7);

	// inicializar una nueva instancia de DateTime usando el año, mes, día,
hora, minuto, segundo y la zona horaria UTC +01:00
	 altova::DateTime dt = altova::DateTime(2015, 11, 22, 13, 53, 7, 60);

	// Obtener el valor de este objeto DateTime
	std::cout << "El número de ciclos del objeto DateTime es: " <<
dt.Value() << std::endl;

	// Obtener el año
	cout << "El año es: " << dt.Year() << endl;
	// Obtener el mes
	cout << "El mes es: " << (int)dt.Month() << endl;
	// Obtener el día del mes
	cout << "El día del mes es: " << (int) dt.Day() << endl;
	// Obtener el día del año
	cout << "El día del año es: " << dt.DayOfYear() << endl;
	// Obtener la hora
	cout << "La hora es: " << (int) dt.Hour() << endl;
	// Obtener el minuto
	cout << "El minuto es: " << (int) dt.Minute() << endl;
	// Obtener el segundo
	cout << "El segundo es: " << dt.Second() << endl;
	// Obtener el día de la semana
	cout << "El día de la semana es: " << dt.Weekday() << endl;
	// Obtener el número de la semana
	cout << "La semana del año es: " << dt.Weeknumber() << endl;
	// Obtener la semana del mes
	cout << "La semana del mes es: " << dt.WeekOfMonth() << endl;

	// Comprobar si una instancia de DateTime tiene una zona horaria
	if (dt.HasTimezone() == TRUE)
	{
		// generar el valor de la zona horaria
		cout <<  "La zona horaria es: " << dt.Timezone() << endl;
	}
	else
	{
		cout <<  "No se definió la zona horaria." << endl;
```

```
        }

        // Construir un objeto DateTime con una zona horaria UTC+01:00 (Viena)
        altova::DateTime vienna_dt = DateTime(2015, 11, 23, 14, 30, 59, +60);
        // Generar el resultado en formato legible
        cout << "La hora de Viena: "
                << (int) vienna_dt.Month()
                << "-" << (int) vienna_dt.Day()
                << " " << (int) vienna_dt.Hour()
                << ":" << (int) vienna_dt.Minute()
                << ":" << (int) vienna_dt.Second()
                << endl;

        // Convertir el valor en hora UTC
        DateTime utc_dt = DateTime(vienna_dt.NormalizedValue());
        // Generar el resultado en formato legible
        cout << "La hora UTC:    "
               << (int) utc_dt.Month()
               << "-" << (int) utc_dt.Day()
               << " " << (int) utc_dt.Hour()
               << ":" << (int) utc_dt.Minute()
               << ":" << (int) utc_dt.Second()
               << endl;

        // Comprobar si el año es bisiest
        int year = 2016;
        if( altova::DateTime::IsLeapYear(year) )
        { cout << year << " es año bisiesto" << endl; }
        else
        { cout << year << " no es año bisiesto" << endl; }
}
```

16.8.2 altova::Duration

Esta clase permite procesar atributos o elementos XML de tipo `xs:duration`.

Constructores

Nombre	Descripción
`Duration()`	Inicializa una instancia nueva de la clase `Duration` en un valor vacío.
`Duration(const DayTimeDuration& dt)`	Inicializa una instancia nueva de la clase `Duration` en una duración definida por el argumento `dt` (*véase altova::DayTimeDuration*).
`Duration(const YearMonthDuration& ym)`	Inicializa una instancia nueva de la clase `Duration` en la duración definida por el argumento `ym` (véase *altova::YearMonthDuration*).
`Duration(const`	Inicializa una instancia nueva de la clase `Duration` en

© 2018 Altova GmbH

Nombre	Descripción
`YearMonthDuration& ym, const DayTimeDuration& dt)`	la duración definida por los argumentos `dt` y `ym` (*véase altova::YearMonthDuration y altova::DayTimeDuration*).

Métodos

Nombre	Descripción
`int Days() const`	Devuelve el número de días de la instancia de `Duration` actual.
`DayTimeDuration DayTime() const`	Devuelve el día y la duración en horas de la instancia de `Duration` actual expresada como objeto `DayTimeDuration` *(véase altova::DayTimeDuration)*.
`int Hours() const`	Devuelve el número de horas de la instancia de `Duration` actual.
`bool IsNegative() const`	Devuelve el valor binario **true** si la instancia de `Duration` actual es negativa.
`bool IsPositive() const`	Devuelve el valor binario **true** si la instancia de `Duration` actual es positiva.
`int Minutes() const`	Devuelve el número de minutos de la instancia de `Duration` actual.
`int Months() const`	Devuelve el número de meses de la instancia de `Duration` actual.
`double Seconds() const`	Devuelve el número de segundos de la instancia de `Duration` actual.
`YearMonthDuration YearMonth() const`	Devuelve el año y la duración en meses de la instancia de `Duration` actual expresada como objeto `YearMonthDuration` (*véase altova::YearMonthDuration*).
`int Years() const`	Devuelve el número de años de la instancia de `Duration` actual.

Ejemplo

En este ejemplo se crea un objeto `Duration` nuevo y se leen sus valores.

```
void ExampleDuration()
{
	// Crear un objeto Duration vacío
	altova::Duration empty_duration = altova::Duration();
```

```
	// Crear un objeto Duration usando un valor de duración actual
	altova::Duration duration1 = altova::Duration(empty_duration);

	// Crear una duración YearMonth de seis años y cinco meses
	altova::YearMonthDuration yrduration = altova::YearMonthDuration(6,
5);

	// Crear una duración DayTime de cuatro días, tres horas, dos minutos y
un segundo
	altova::DayTimeDuration dtduration = altova::DayTimeDuration(4, 3, 2,
1);

	// Crear un objeto Duration combinando las dos duraciones creadas
previamente
	altova::Duration duration = altova::Duration(yrduration, dtduration);

	// Obtener el número de años de esta instancia de Duration
	cout << "Años:  " << duration.Years() << endl;

	// Obtener el número de meses de esta instancia de Duration
	cout << "Meses: " << duration.Months() << endl;

	// Obtener el número de días de esta instancia de Duration
	cout << "Días:   " << duration.Days() << endl;

	// Obtener el número de horas de esta instancia de Duration
	cout << "Horas:  " << duration.Hours() << endl;

	// Obtener el número de minutos de esta instancia de Duration
	cout << "Minutos: " << duration.Minutes() << endl;

	// Obtener el número de segundos de esta instancia de Duration
	cout << "Segundos: " << duration.Seconds() << endl;
}
```

16.8.3 altova::DayTimeDuration

Esta clase permite procesar tipos XML Schema de duración compuestos por un día y una hora.

Constructores

Nombre	Descripción
`DayTimeDuration()`	Inicializa una instancia nueva de la clase `DayTimeDuration` en un valor vacío.
`DayTimeDuration(int days, int hours, int minutes, double seconds)`	Inicializa una instancia nueva de la clase `DayTimeDuration` en el número de días, horas, minutos y segundos dados como argumento.
`explicit`	Inicializa una instancia nueva de la clase

© 2018 Altova GmbH

Nombre	Descripción
`DayTimeDuration(__int64 value)`	`DayTimeDuration` en una duración compuesta por tantos ciclos (intervalos de 100 nanosegundos) como aporta el argumento **value**.

Métodos

Nombre	Descripción
`int Days() const`	Devuelve el número de días de la instancia de `DayTimeDuration` actual.
`int Hours() const`	Devuelve el número de horas de la instancia de `DayTimeDuration` actual.
`bool IsNegative() const`	Devuelve el valor binario **true** si la instancia de `DayTimeDuration` actual es negativa.
`bool IsPositive() const`	Devuelve el valor binario **true** si la instancia de `DayTimeDuration` actual es positiva.
`int Minutes() const`	Devuelve el número de minutos de la instancia de `DayTimeDuration` actual.
`double Seconds() const`	Devuelve el número de segundos de la instancia de `DayTimeDuration` actual.
`__int64 Value() const`	Devuelve el valor (en ciclos) de la instancia de `DayTimeDuration` actual.

16.8.4 altova::YearMonthDuration

Esta clase permite procesar tipos XML Schema de duración compuestos por un año y un mes.

Constructores

Nombre	Descripción
`YearMonthDuration()`	Inicializa una instancia nueva de la clase `YearMonthDuration` en un valor vacío.
`YearMonthDuration(int years, int months)`	Inicializa una instancia nueva de la clase `YearMonthDuration` en el número de años y meses dados por los argumentos **years** y **months**.
`explicit YearMonthDuration(int value)`	Inicializa una instancia nueva de la clase `YearMonthDuration` en una duración compuesta por tantos ciclos (intervalos de 100 nanosegundos) como

Nombre	Descripción
	da el argumento **value**.

Métodos

Nombre	Descripción
`bool IsNegative() const`	Devuelve el valor binario **true** si la instancia de `YearMonthDuration` actual es negativa.
`bool IsPositive() const`	Devuelve el valor binario **true** si la instancia de `YearMonthDuration` actual es positiva.
`int Months() const`	Devuelve el número de meses de la instancia de `YearMonthDuration` actual.
`int Value() const`	Devuelve el valor (en ciclos) de la instancia de `YearMonthDuration` actual.
`int Years()`	Devuelve el número de años de la instancia de `YearMonthDuration` actual.

16.8.5 altova::meta::Attribute

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de atributos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un atributo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier atributo que esté definido en el esquema XML.

Métodos

Nombre	Descripción
`SimpleType GetDataType()`	Devuelve el tipo del contenido del atributo.
`string_type GetLocalName()`	Devuelve el nombre local del atributo.
`string_type GetNamespaceURI()`	Devuelve el URI de espacio de nombres del atributo.
`bool IsRequired()`	Devuelve true si el atributo es obligatorio.

Operadores

© 2018 Altova GmbH

Nombre	Descripción
`bool operator()`	Devuelve true si este no es el atributo NULL.
`bool operator!()`	Devuelve true si este es el atributo NULL.

16.8.6 altova::meta::ComplexType

Esta clase permite acceder a información del esquema sobre clases generadas a partir de tipos complejos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo complejo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo complejo que esté definido en el esquema XML.

Métodos

Nombre	Descripción
`Attribute FindAttribute(const char_type* localName, const char_type* namespaceURI)`	Busca el atributo que tiene el nombre local y el URI de espacio de nombres especificados.
`Element FindElement(const char_type* localName, const char_type* namespaceURI)`	Busca el elemento que tiene el nombre local y el URI de espacio de nombres especificados.
`std::vector<Attribute> GetAttributes()`	Devuelve una lista de todos los atributos.
`ComplexType GetBaseType()`	Devuelve el tipo base de este tipo.
`SimpleType GetContentType()`	Devuelve el tipo simple del contenido.
`std::vector<Element> GetElements()`	Devuelve una lista de todos los elementos.
`string_type GetLocalName()`	Devuelve el nombre local del tipo.
`string_type GetNamespaceURI()`	Devuelve el URI de espacio de nombres del tipo.

Operadores

Nombre	Descripción
`bool operator()`	Devuelve true si este no es el tipo complejo NULL.
`bool operator!()`	Devuelve true si este es el tipo complejo NULL.

16.8.7 altova::meta::Element

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de elementos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un elemento de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier elemento que esté definido en el esquema XML.

Métodos

Nombre	Descripción
`ComplexType GetDataType()`	Devuelve el tipo del elemento. Recuerde que este siempre es un tipo complejo incluso si se declaró como tipo simple en el esquema original. Use GetContentType() del objeto devuelto para obtener el tipo de contenido simple.
`string_type GetLocalName()`	Devuelve el nombre local del elemento.
`unsigned int GetMaxOccurs()`	Devuelve el valor maxOccurs que está definido en el esquema.
`unsigned int GetMinOccurs()`	Devuelve el valor minOccurs que está definido en el esquema.
`string_type GetNamespaceURI()`	Devuelve el URI de espacio de nombres del elemento.

Operadores

Nombre	Descripción
`bool operator()`	Devuelve false si este no es el elemento NULL.
`bool operator!()`	Devuelve true si este es el elemento NULL.

© 2018 Altova GmbH

16.8.8 altova::meta::SimpleType

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de tipos simples. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo simple de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo simple que esté definido en el esquema XML.

Métodos

Nombre	Descripción
`SimpleType GetBaseType()`	Devuelve el tipo base de este tipo.
`std::vector<string_type> GetEnumerations()`	Devuelve una lista de todas las facetas de enumeración.
`unsigned int GetFractionDigits()`	Devuelve el valor de esta faceta.
`unsigned int GetLength()`	Devuelve el valor de esta faceta.
`string_type GetLocalName()`	Devuelve el nombre local de este tipo.
`string_type GetMaxExclusive()`	Devuelve el valor de esta faceta.
`string_type GetMaxInclusive()`	Devuelve el valor de esta faceta.
`unsigned int GetMaxLength()`	Devuelve el valor de esta faceta.
`string_type GetMinExclusive()`	Devuelve el valor de esta faceta.
`string_type GetMinInclusive()`	Devuelve el valor de esta faceta.
`unsigned int GetMinLength()`	Devuelve el valor de esta faceta.
`string_type GetNamespaceURI()`	Devuelve el URI de espacio de nombres del tipo.
`std::vector<string_type> GetPatterns()`	Devuelve una lista de todas las facetas de patrón.
`unsigned int GetTotalDigits()`	Devuelve el valor de esta faceta.
`WhitespaceType GetWhitespace()`	Devuelve el valor de la faceta de espacio en blanco, que es uno de estos: • Whitespace_Unknown

Nombre	Descripción
	• Whitespace_Preserve • Whitespace_Replace • Whitespace_Collapse

Operadores

Nombre	Descripción
`bool operator()`	Devuelve true si este no es el SimpleType NULL.
`bool operator!()`	Devuelve true si este es el SimpleType NULL.

16.8.9 [SuEsquema]::[CDoc]

Cuando se genera código a partir de un esquema XML, el código generado ofrece una clase de documento que se llama igual que el esquema. Esta clase tiene como miembros todos los elementos raíz posibles, así como los métodos que aparecen a continuación. Observe que en los nombres de método, `CDoc` corresponde al nombre de la clase de documento generada.

Métodos

Nombre	Descripción
`static CDoc CreateDocument()`	Crea un documento XML nuevo vacío. Debe publicarse con DestroyDocument().
`void DestroyDocument()`	Destruye un documento. Todas las referencias al documento y a sus nodos se invalidarán. Esta llamada debe hacerse cuando se termina de trabajar con un documento.
`static CDoc LoadFromBinary(const std:vector<unsigned char>& xml)`	Carga un documento XML desde una matriz de bytes.
`static CDoc LoadFromFile(const string_type& fileName)`	Carga un documento XML desde un archivo.
`static CDoc LoadFromString(const string_type& xml)`	Carga un documento XML desde una cadena de texto.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint)`	Guarda un documento XML en una matriz de bytes. Si su valor es true, el argumento `prettyPrint` ajusta el formato del documento XML para facilitar su lectura.
`std::vector<unsigned char>`	Guarda un documento XML en una matriz

© 2018 Altova GmbH

Nombre	Descripción
`SaveToBinary(bool prettyPrint, const string_type & encoding)`	de bytes (con formato pretty-print opcional) con la codificación especificada.
`std::vector<unsigned char> SaveToBinary(bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
`void SaveToFile(const string_type & fileName, bool prettyPrint)`	Guarda un documento XML en un archivo, con formato pretty-print opcional.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl)`	Guarda un documento XML en un archivo. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
`void SaveToFile(const string_type & fileName, bool omitXmlDecl, const string_type & encoding)`	Guarda un documento XML en un archivo. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM)`	Guarda un documento XML en un archivo (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
`void SaveToFile(const string_type & fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, bool bBigEndian, bool bBOM, const string_type & lineend)`	Guarda un documento XML en un archivo(con formato pretty-print opcional) con la codificación y el final de línea especificados. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode. Este método solo está disponible si generó código para la biblioteca XML Xerces3 (*véase* *Opciones del generador de código*).
`void SaveToFile(const string_type& fileName, bool prettyPrint, bool omitXmlDecl, const string_type & encoding, const string_type & lineend)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación y el final de línea especificados. Este método solo está disponible si generó código para la biblioteca XML Xerces3 (*véase* *Opciones del generador de código*).
`void SaveToFile(const string_type & fileName, bool prettyPrint, const string_type & encoding)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación especificada.

Nombre	Descripción
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, bool bBigEndian, bool bBOM, const string_type & lineend)`	Guarda un documento XML en un archivo con la codificación y el final de línea especificados. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode. Este método solo está disponible si generó código para la biblioteca XML Xerces3 (*véase Opciones del generador de código*).
`void SaveToFile(const string_type& fileName, bool prettyPrint, const string_type & encoding, const string_type & lineend)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación y el final de línea especificados. Este método solo está disponible si generó código para la biblioteca XML Xerces3 (véase Opciones del generador de código).
`string_type SaveToString(bool prettyPrint)`	Guarda un documento XML en una cadena de texto, con formato pretty-print opcional.
`string_type SaveToString(bool prettyPrint, bool omitXmlDecl)`	Guarda un documento XML en una cadena de texto, con formato pretty-print opcional. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
`void SetDTDLocation(const string_type & dtdLocation)`	Agrega una declaración DOCTYPE con el ID de sistema especificado. Es obligatorio que ya exista un elemento raíz. Este método es incompatible con MSXML porque no se puede agregar una declaración DOCTYPE a un documento que está en memoria.
`void SetSchemaLocation(const string_type & schemaLocation)`	Agrega un atributo `xsi:schemaLocation` o `xsi:noNamespaceSchemaLocation` al elemento raíz. Es obligatorio que ya exista un elemento raíz.

© 2018 Altova GmbH

16.8.10 [SuEsquema]::MemberAttribute

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada atributo miembro de un tipo.

Métodos

Nombre	Descripción
`bool exists()`	Devuelve true si el atributo existe.
`int GetEnumerationValue()`	Solo se genera para tipos de enumeración. Devuelve una de las constantes generadas para los posibles valores o "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
`altova::meta::Attribute info()`	Devuelve un objeto para consultar información del esquema (*véase* *altova::meta::Attribute*).
`void remove()`	Quita el atributo de su elemento primario.
`void SetEnumerationValue(int)`	Solo se genera para tipos de enumeración. Pase una de las constantes generadas para los posibles valores a este método para establecer el valor.

16.8.11 [SuEsquema]::MemberElement

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada elemento miembro de un tipo. En la tabla que aparece a continuación `MemberType` corresponde al nombre del elemento miembro propiamente dicho.

Métodos

Nombre	Descripción
`Iterator<MemberType> all()`	Devuelve un objeto para recorrer instancias del elemento miembro.
`MemberType append()`	Crea un elemento nuevo y lo anexa a su primario.
`unsigned int count()`	Devuelve el recuento de elementos.

Nombre	Descripción
`int GetEnumerationValue()`	Solo se genera para tipos de enumeración. Devuelve una de las constantes generadas para los posibles valores o "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
`bool exists()`	Devuelve true si existe al menos un elemento.
`MemberType first()`	Devuelve la primera instancia del elemento miembro.
`MemberType operator[](unsigned int index)`	Devuelve el elemento miembro especificado por el índice.
`altova::meta::Element info()`	Devuelve un objeto para consultar información del esquema (*véase* *`altova::meta::Element`*).
`MemberType last()`	Devuelve la última instancia del elemento miembro.
`void remove()`	Elimina todas las instancias del elemento en su primario.
`void remove(unsigned int index)`	Elimina la instancia del elemento especificada por el índice.
`void SetEnumerationValue(int)`	Solo se genera para tipos de enumeración. Pase una de las constantes generadas para los posibles valores a este método para establecer el valor.

© 2018 Altova GmbH

16.9 Referencia de las clases generadas (C#)

Esta sección describe las clases C# que se generan con MapForce a partir de una DTD o esquema XML (*véase* *Generar código a partir de esquemas XML o DTD*). Puede integrar estas clases en su código para leer, modificar y escribir documentos XML.

Nota: el código generado puede incluir otras clases compatibles que no se enumeran en este apartado y que pueden cambiar en el futuro.

16.9.1 Altova.Types.DateTime

Esta clase permite procesar atributos o elementos XML que tienen tipos de fecha y hora (p. ej. `xs:dateTime`).

Constructores

	Nombre	Descripción
	`DateTime(DateTime obj)`	Inicializa una instancia nueva de la clase `DateTime` en el objeto `DateTime` dado como argumento.
	`DateTime(System.DateTime newvalue)`	Inicializa una instancia nueva de la clase `DateTime` en el objeto `System.DateTime` dado como argumento.
	`DateTime(int year, int month, int day, int hour, int minute, double second, int offsetTZ)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto, segundo y diferencia de zona horaria dados como argumentos.
	`DateTime(int year, int month, int day, int hour, int minute, double second)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto y segundo dados como argumentos.
	`DateTime(int year, int month, int day)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes y día dados como argumentos.

Propiedades

	Nombre	Descripción
	`bool HasTimezone`	Obtiene un valor booleano que indica si `DateTime` tiene una zona horaria.
	`static DateTime Now`	Obtiene un objeto `DateTime` que está definido con la fecha y hora actuales de este equipo.

	Nombre	Descripción
	`short TimezoneOffset`	Obtiene o establece la diferencia de zona horaria, en minutos, del objeto `DateTime`.
	`System.DateTime Value`	Obtiene o establece el valor del objeto `DateTime` como valor de un `System.DateTime`.

Métodos

	Nombre	Descripción
	`int CompareTo(object obj)`	La clase `DateTime` implementa la interfaz `IComparable`. Este método compara la instancia actual de `DateTime` con otro objeto y devuelve un entero que indica si la posición de la instancia actual es anterior, posterior o igual que la del otro objeto en el criterio de ordenación. Consulte también https://msdn.microsoft.com/es-es/library/system.icomparable.compareto(v=vs.110).aspx
	`override bool Equals(object obj)`	Devuelve **true** si el objeto especificado es igual que el objeto actual. De lo contrario devuelve **false**.
	`System.DateTime GetDateTime(bool correctTZ)`	Devuelve un objeto `System.DateTime` de la instancia `Altova.Types.DateTime` actual. El argumento booleano `correctTZ` especifica si la hora del objeto devuelto debe ajustarse a la zona horaria de la instancia `Altova.Types.DateTime` actual.
	`override int GetHashCode()`	Devuelve el código hash de la instancia actual.
	`int GetWeekOfMonth()`	Devuelve el número de la semana del mes como entero.

	`static DateTime Parse( string s )`	Crea un objeto `DateTime` a partir de la cadena dada como argumento. Por ejemplo, estos valores de cadena de muestra se convertirían correctamente en un objeto `DateTime`:
		
		`2015-01-01T23:23:23`
		`2015-01-01`
		`2015-11`
		`23:23:23`
		
		Si la cadena no se puede convertir en un objeto `DateTime`, entonces se genera una excepción.
		
		Tenga en cuenta que este método es estático y solo se le puede llamar en la clase `Altova.Types.DateTime` propiamente dicha y no

© 2018 Altova GmbH

	Nombre	Descripción
		en una instancia de la clase.
	`static DateTime Parse(string s, DateTimeFormat format)`	Crea un objeto `DateTime` a partir de una cadena usando el formato dado como argumento. Para ver una lista de formatos posibles consulte Altova.Types.DateTimeFormat. Si la cadena no se puede convertir en un objeto `DateTime`, entonces se genera una excepción. Tenga en cuenta que este método es estático y solo se le puede llamar en la clase `Altova.Types.DateTime` propiamente dicha y no en una instancia de la clase.
	`override string ToString()`	Convierte el objeto `DateTime` en una cadena.
	`string ToString(DateTimeFormat format)`	Convierte el objeto `DateTime` en una cadena usando el formato dado como argumento. Para ver una lista de formatos posibles consulte Altova.Types.DateTimeFormat.

Operadores

Nombre	Descripción
`!=`	Determina si `DateTime` a no es igual que `DateTime` b.
`<`	Determina si `DateTime` a es menor que `DateTime` b.
`<=`	Determina si `DateTime` a es menor o igual que `DateTime` b.
`==`	Determina si `DateTime` a es igual que `DateTime` b.
`>`	Determina si `DateTime` a es mayor que `DateTime` b.
`>=`	Determina si `DateTime` a es mayor o igual que `DateTime` b.

Ejemplos

Antes de usar estos ejemplos en su programa debe importar los tipos de Altova:

```
using Altova.Types;
```

Este ejemplo muestra cómo se pueden crear objetos `DateTime`:

```
protected static void DateTimeExample1()
{
	// Crear un objeto DateTime a partir de la hora actual del sistema
	Altova.Types.DateTime dt = new
Altova.Types.DateTime(System.DateTime.Now);
	Console.WriteLine("La hora actual es: " + dt.ToString());

	// Crear un objeto Altova DateTime a partir de las partes (sin zona
horaria)
	Altova.Types.DateTime dt1 = new Altova.Types.DateTime(2015, 10, 12, 10,
50, 33);
	Console.WriteLine("Mi hora es : " + dt1.ToString());

	// Crear un objeto Altova DateTime a partir de las partes (con zona
horaria UTC+60 minutos)
	Altova.Types.DateTime dt2 = new Altova.Types.DateTime(2015, 10, 12, 10,
50, 33, 60);
	Console.WriteLine("Mi hora con zona horaria es: " + dt2.ToString());

	// Crear un objeto Altova DateTime analizando una cadena
	Altova.Types.DateTime dt3 = Altova.Types.DateTime.Parse("2015-01-
01T23:23:23");
	Console.WriteLine("La hora se creó a partir de la cadena: " +
dt3.ToString());

	// Crear un objeto Altova DateTime analizando una cadena con formato de
fecha de esquema
	Altova.Types.DateTime dt4 = Altova.Types.DateTime.Parse("2015-01-01",
DateTimeFormat.W3_date);
	Console.WriteLine("La hora se creó a partir de una cadena con formato
de fecha de esquema: " + dt4.ToString());
}
```

Este ejemplo muestra cómo se puede dar formato a objetos `DateTime`:

```
protected static void DateTimeExample2()
{
	// Crear un objeto DateTime a partir de la hora actual del sistema
	Altova.Types.DateTime dt = new
Altova.Types.DateTime(System.DateTime.Now);

	// Generar la fechaHora sin formato
	Console.WriteLine("Hora sin formato: " + dt.ToString());

	// Generar este DateTime con formato usando lo siguiente
	Console.WriteLine("S_DateTime:       " +
dt.ToString(DateTimeFormat.S_DateTime));
	Console.WriteLine("S_Days:           " +
dt.ToString(DateTimeFormat.S_Days));
	Console.WriteLine("S_Seconds:        " +
dt.ToString(DateTimeFormat.S_Seconds));
	Console.WriteLine("W3_date:          " +
dt.ToString(DateTimeFormat.W3_date));
	Console.WriteLine("W3_dateTime:      " +
dt.ToString(DateTimeFormat.W3_dateTime));
```


© 2018 Altova GmbH

```
      Console.WriteLine("W3_gDay:          " +
dt.ToString(DateTimeFormat.W3_gDay));
      Console.WriteLine("W3_gMonth:        " +
dt.ToString(DateTimeFormat.W3_gMonth));
      Console.WriteLine("W3_gMonthDay:     " +
dt.ToString(DateTimeFormat.W3_gMonthDay));
      Console.WriteLine("W3_gYear:         " +
dt.ToString(DateTimeFormat.W3_gYear));
      Console.WriteLine("W3_gYearMonth:    " +
dt.ToString(DateTimeFormat.W3_gYearMonth));
      Console.WriteLine("W3_time:          " +
dt.ToString(DateTimeFormat.W3_time));
}
```

16.9.2 Altova.Types.DateTimeFormat

El tipo de enumeración `DateTimeFormat` tiene estos valores de constante:

Valor	Descripción	Ejemplo
S_DateTime	Da al valor el formato de un dateTime estándar, con una precisión de una décimo millonésima de segundo, incluida la zona horaria.	`2015-11-12 12:19:03.9019132 +01:00`
S_Days	Da al valor el formato del número de días que han pasado desde la época UNIX.	`735913.631897345108796296 2963`
S_Seconds	Da al valor el formato del número de segundos que han pasado desde la épica UNIX, con una precisión de una décimo millonésima de segundo.	`63582937678.0769062`
W3_date	Da al valor el formato de fecha de esquema.	`2015-11-12`
W3_dateTime	Da al valor el formato de fechaHora de esquema.	`2015-11-12T15:12:14.5194251`
W3_gDay	Da al valor el formato de gDay de esquema.	`---12` (si la fecha es el día 12 del mes)
W3_gMonth	Da al valor el formato de gMonth de esquema.	`--11` (si el mes es noviembre)
W3_gMonthDay	Da al valor el formato de gMonthDay de esquema.	`--11-12` (si la fecha es el día 12 de noviembre)

Valor	Descripción	Ejemplo
W3_gYear	Da al valor el formato de gYear de esquema.	`2015` (si el año es 2015)
W3_gYearMonth	Da al valor el formato de gYearMonth de esquema.	`2015-11` (si el años es 2015 y el mes es noviembre)
W3_time	Da al valor el formato de hora de esquema, con una precisión de una décimo millonésima de segundo.	`15:19:07.5582719`

16.9.3 Altova.Types.Duration

Esta clase permite procesar atributos o elementos XML de tipo `xs:duration`.

Constructores

	Nombre	Descripción
	`Duration(Duration obj)`	Inicializa una instancia nueva de la clase `Duration` en el objeto `Duration` dado como argumento.
	`Duration(System.TimeSpan newvalue)`	Inicializa una instancia nueva de la clase `Duration` en el objeto `System.TimeSpan` dado como argumento.
	`Duration(long ticks)`	Inicializa una instancia nueva de la clase `Duration` en el número de ciclos dado como argumento.
	`Duration(int newyears, int newmonths, int days, int hours, int minutes, int seconds, double partseconds, bool bnegative)`	Inicializa una instancia nueva de la clase `Duration` en una duración compuesta por las partes dadas como argumentos.

Propiedades

	Nombre	Descripción
	`int Months`	Obtiene o establece el número de meses de la instancia actual de `Duration`.
	`System.TimeSpan Value`	Obtiene o establece el valor (como `System.TimeSpan`) de la instancia actual de `Duration`.

© 2018 Altova GmbH

	Nombre	Descripción
	`int Years`	Obtiene o establece el número de años de la instancia actual de `Duration`.

Métodos

	Nombre	Descripción
	`override bool Equals(object other)`	Devuelve **true** si el objeto especificado es igual al objeto actual. De lo contrario devuelve **false**.
	`override int GetHashCode()`	Devuelve el código hash de la instancia actual.
	`bool IsNegative()`	Devuelve **true** si la instancia actual de `Duration` representa una duración negativa.
	`static Duration Parse( string s, ParseType pt )`	Devuelve un objeto `Altova.Types.Duration` analizado desde una cadena dada como argumento, usando el tipo de análisis dado como argumento. Son valores válidos de tipo de análisis: **DURATION** Analizar duración asumiendo que existen las partes año, mes, día y duración. **YEARMONTH** Analizar duración asumiendo que solo existen las partes año y mes. **DAYTIME** Analizar duración asumiendo que solo existen las partes día y hora. Recuerde que este método es estático solo se le puede llamar en la clase propiamente dicha y no desde una instancia de la clase.
	`override string ToString()`	Convierte la instancia `Duration` actual en una cadena. Por ejemplo, un intervalo de tiempo de 3 horas, 4 minutos y 5 segundos se convertiría en la cadena "`PT3H4M5S`".
	`string ToYearMonthString( )`	Convierte la instancia `Duration` actual en una cadena usando el tipo de análisis YEARMONTH.

Operadores

Nombre	Descripción
`!=`	Determina si `Duration` a no es igual que `Duration` b.

Nombre	Descripción
==	Determina si `Duration` a es igual que `Duration` b.

Ejemplos

Antes de usar estos ejemplos en su programa debe importar los tipos de Altova:

```
using Altova.Types;
```

Este ejemplo muestra cómo se pueden crear objetos `Duration`:

```
protected static void DurationExample1()
{
	// Crear un intervalo de tiempo nuevo de 3 horas, 4 minutos y 5
segundos
	System.TimeSpan ts = new TimeSpan(3, 4, 5);
	// Crear un Duration a partir del intervalo de tiempo
	Duration dr = new Duration(ts);
	// El resultado es: PT3H4M5S
	Console.WriteLine("Duración creada a partir de intervaloTiempo: " +
dr.ToString());

	// Crear un Altova.Types.Duration negativo a partir de 6 años, 5 meses,
4 días, 3 horas, 2 minutos, 1 segundo y .33 de un segundo
	Duration dr1 = new Duration(6, 5, 4, 3, 2, 1, .33, true);
	// The output is: -P6Y5M4DT3H2M1.33S
	Console.WriteLine("Duración creada a partir de partes: " +
dr1.ToString());

	// Crear un Duration a partir de una cadena usando el tipo de análisis
DAYTIME
	Duration dr2 = Altova.Types.Duration.Parse("-P4DT3H2M1S",
Duration.ParseType.DAYTIME);
	// The output is -P4DT3H2M1S
	Console.WriteLine("Duración creada a partir de cadena: " +
dr2.ToString());

	// Crear una duración a partir de ciclos
	Duration dr3 = new Duration(System.DateTime.UtcNow.Ticks);
	// Generar el resultado
	Console.WriteLine("Duración creada a partir de ciclos: " +
dr3.ToString());
}
```

Este ejemplo muestra cómo se pueden obtener valores de los objetos `Duration`:

```
protected static void DurationExample2()
{
	// Crear un Altova.Types.Duration negativo a partir de 6 años, 5 meses,
4 días, 3 horas, 2 minutos, 1 segundo y .33 de un segundo
```


© 2018 Altova GmbH

```
        Duration dr = new Duration(6, 5, 4, 3, 2, 1, .33, true);
        // El resultado es: -P6Y5M4DT3H2M1.33S
        Console.WriteLine("La duración completa es: " + dr.ToString());

        // Obtener solo la parte año y mes como cadena
        string dr1 = dr.ToYearMonthString();
        Console.WriteLine("La parte YEARMONTH es: " + dr1);

        // Obtener el número de años de la duración
        Console.WriteLine("Años: " + dr.Years);

        // Obtener el número de meses de la duración
        Console.WriteLine("Meses: " + dr.Months);
}
```

16.9.4 Altova.Xml.Meta.Attribute

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de atributos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un atributo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier atributo que esté definido en el esquema XML.

Propiedades

	Nombre	Descripción
	`SimpleType DataType`	Devuelve el tipo del contenido del atributo.
	`string LocalName`	Devuelve el nombre local del atributo.
	`string NamespaceURI`	Devuelve el URI de espacio de nombres del atributo.
	`XmlQualifiedName QualifiedName`	Devuelve el nombre completo del atributo.
	`bool Required()`	Devuelve true si el atributo es obligatorio.

16.9.5 Altova.Xml.Meta.ComplexType

Esta clase permite acceder a información del esquema sobre clases generadas a partir de tipos complejos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo complejo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo complejo que esté definido en el esquema XML.

Propiedades

	Nombre	Descripción
	`Attribute[] Attributes`	Devuelve una lista de todos los atributos.
	`ComplexType BaseType`	Devuelve el tipo base de este tipo o null si no existe el tipo base.
	`SimpleType ContentType`	Devuelve el tipo simple del contenido.
	`Element[] Elements`	Devuelve una lista de todos los elementos.
	`string LocalName`	Devuelve el nombre local de este tipo.
	`string NamespaceURI`	Devuelve el URI de espacio de nombres del tipo.
	`XmlQualifiedName QualifiedName`	Devuelve el nombre completo de este tipo.

Métodos

	Nombre	Descripción
	`ComplexType BaseType`	Devuelve el tipo base de este tipo.
	`bool Equals(obj)`	Comprueba si dos objetos de información remiten al mismo tipo, basándose en la comparación de los nombres completos. Devuelve true si el tipo tiene el mismo nombre completo.
	`Attribute FindAttribute(string localName, string namespaceURI)`	Busca el atributo con el nombre local y el URI de espacio de nombres especificados.
	`Element FindElement(string localName, string namespaceURI)`	Busca el elemento con el nombre local y el URI de espacio de nombres especificados.

16.9.6 Altova.Xml.Meta.Element

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de elementos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un elemento de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier elemento que esté definido en el esquema XML.

© 2018 Altova GmbH

Propiedades

	Nombre	Descripción
	`ComplexType DataType`	Devuelve el tipo del elemento. Recuerde que este siempre es un tipo complejo incluso si se declaró como simple en el esquema original. Use `ContentType` del objeto devuelto para obtener el tipo de contenido simple.
	`string LocalName`	Devuelve el nombre local del elemento.
	`int MaxOccurs`	Devuelve el valor `maxOccurs` definido en el esquema.
	`int MinOccurs`	Devuelve el valor `minOccurs` definido en el esquema.
	`string NamespaceURI`	Devuelve el URI de espacio de nombres del elemento.
	`XmlQualifiedName QualifiedName`	Devuelve el nombre completo del elemento.

16.9.7 Altova.Xml.Meta.SimpleType

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de tipos simples. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo simple de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo simple que esté definido en el esquema XML.

Propiedades

	Nombre	Descripción
	`SimpleType BaseType`	Devuelve el tipo base de este tipo.
	`string[] Enumerations`	Devuelve una lista de todas las facetas de enumeración.
	`int FractionDigits`	Devuelve el valor de esta faceta.
	`int Length`	Devuelve el valor de esta faceta.
	`string LocalName`	Devuelve el nombre local de este tipo.
	`string MaxExclusive`	Devuelve el valor de esta faceta.
	`string MaxInclusive`	Devuelve el valor de esta faceta.

	Nombre	Descripción
	`int MaxLength`	Devuelve el valor de esta faceta.
	`string MinExclusive`	Devuelve el valor de esta faceta.
	`string MinInclusive`	Devuelve el valor de esta faceta.
	`int MinLength`	Devuelve el valor de esta faceta.
	`string NamespaceURI`	Devuelve el URI de espacio de nombres del tipo.
	`string[] Patterns`	Devuelve las facetas de patrón o null si no hay patrones especificados.
	`XmlQualifiedName QualifiedName`	Devuelve el nombre completo de este tipo.
	`int TotalDigits`	Devuelve el valor de esta faceta.
	`WhitespaceType Whitespace`	Devuelve la faceta de normalización de espacios en blanco.

16.9.8 [SuEsquema].[Doc]

Cuando se genera código a partir de un esquema XML, el código generado ofrece una clase de documento que se llama igual que el esquema. Esta clase tiene como miembros todos los elementos raíz posibles, así como los métodos que aparecen a continuación. Observe que en los nombres de método, `CDoc` corresponde al nombre de la clase de documento generada.

Métodos

	Nombre	Descripción
	`static Doc CreateDocument()`	Crea un documento XML nuevo vacío.
	`static Doc CreateDocument(string encoding)`	Crea un documento XML nuevo vacío, con codificación de tipo "encoding"
	`static Doc LoadFromBinary(byte[] binary)`	Carga un documento XML desde una matriz de bytes.
	`static Doc LoadFromFile(string filename)`	Carga un documento XML desde un archivo.
	`static Doc LoadFromString(string xmlstring)`	Carga un documento XML desde una cadena de texto.
	`byte[]`	Guarda un documento XML en una matriz de bytes,

© 2018 Altova GmbH

	Nombre	Descripción
	`SaveToBinary(bool prettyPrint)`	con formato pretty-print opcional.
	`byte[] SaveToBinary(bool prettyPrint, string encoding)`	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada.
	`byte[] SaveToBinary(bool prettyPrint, string encoding, bool bBigEndian, bool bBOM)`	Guarda un documento XML en una matriz de bytes, (con formato pretty-print opcional) con la codificación, el orden de bytes y la marca BOM especificados.
	`void SaveToFile(string fileName, bool prettyPrint)`	Guarda un documento XML en un archivo, con formato pretty-print opcional.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl)`	Guarda un documento XML en un archivo, con formato pretty-print opcional. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación especificada. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void SaveToFile(string fileName, bool prettyPrint, string encoding, string lineend)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación y los caracteres de final de línea especificados.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, string lineend)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación y los caracteres de final de línea especificados. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void SaveToFile(string fileName, bool prettyPrint, bool omitXmlDecl, string encoding, bool bBigEndian, bool bBOM, string lineend)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación, el orden de bytes, la marca BOM y los caracteres de final de línea especificados. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void SaveToFileWithLineEnd(string fileName, bool`	Guarda un documento XML en un archivo, con formato pretty-print opcional y con caracteres de final de línea. Si el argumento `omitXmlDecl` tiene el

	Nombre	Descripción
	`prettyPrint, bool omitXmlDecl, string lineend)`	valor true, la declaración XML no se escribirá.
	`string SaveToString(bool prettyPrint)`	Guarda un documento XML en un archivo, con formato pretty-print opcional.
	`string SaveToString(bool prettyPrint, bool omitXmlDecl)`	Guarda un documento XML en un archivo, con formato pretty-print opcional. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void SetDTDLocation(string dtdLocation)`	Agrega una declaración DOCTYPE con el ID de sistema especificado. Es obligatorio que ya exista un elemento raíz.
	`void SetSchemaLocation(string schemaLocation)`	Agrega un atributo xsi:schemaLocation o xsi:noNamespaceSchemaLocation al elemento raíz. Es obligatorio que ya exista un elemento raíz.

16.9.9 [SuTipoEsquema].MemberAttribute

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada atributo miembro de un tipo. En las tablas que aparecen a continuación `AttributeType` corresponde al tipo del atributo miembro propiamente dicho.

Métodos

	Nombre	Descripción
	`bool Exists()`	Devuelve true si el atributo existe.
	`void Remove()`	Quita el atributo de su elemento primario.

Propiedades

	Nombre	Descripción
	`int EnumerationValue`	Solo se genera para tipos de enumeración. Establece u obtiene el valor del atributo con ayuda de una de las constantes generadas para los posibles valores. Devuelve "no válido"

© 2018 Altova GmbH

	Nombre	Descripción
		si el valor no coincide con ninguno de los valores enumerados en el esquema.
	`Altova.Xml.Meta.Attribute Info`	Devuelve un objeto para consultar información del esquema (*véase* *`Altova.Xml.Meta.Attribute`*).
	`AttributeType Value`	Establece u obtiene el valor del atributo.

16.9.10 [SuTipoEsquema].MemberElement

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada elemento miembro de un tipo. La clase implementa la interfaz estándar `System.Collections.IEnumerable` para que pueda utilizarse con la instrucción `foreach`.

En la tabla que aparece a continuación `MemberType` corresponde al tipo del elemento miembro propiamente dicho.

Métodos

	Nombre	Descripción
	`MemberType Append()`	Crea un elemento nuevo y lo anexa a su primario.
	`MemberType At(int index)`	Devuelve el elemento miembro especificado por el índice.
	`System.Collections.IEnumerator GetEnumerator()`	Devuelve un objeto para recorrer instancias del elemento miembro.
	`void Remove()`	Elimina todas las instancias del elemento en su primario.
	`void RemoveAt(int index)`	Elimina la instancia del elemento especificada por el índice.

Propiedades

	Nombre	Descripción
	`int Count`	Devuelve el recuento de elementos.

	Nombre	Descripción
	`int EnumerationValue`	Solo se genera para tipos de enumeración. Establece u obtiene el valor del elemento con ayuda de una de las constantes generadas para los posibles valores. Devuelve "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
	`bool Exists`	Devuelve true si existe al menos un elemento.
	`MemberType First`	Devuelve la primera instancia del elemento miembro.
	`Altova.Xml.Meta.Element Info`	Devuelve un objeto para consultar información del esquema (véase `Altova.Xml.Meta.Element`).
	`MemberType Last`	Devuelve la última instancia del elemento miembro.
	`MemberType this[int index]`	Devuelve el elemento miembro especificado por el índice.
	`MemberType Value`	Establece u obtiene el contenido del elemento (solo se genera si el elemento puede tener contenido mixto o simple).

© 2018 Altova GmbH

16.10 Referencia de las clases generadas (Java)

Esta sección describe las clases Java que se generan con MapForce a partir de una DTD o esquema XML (*véase* *Generar código a partir de esquemas XML o DTD*). Puede integrar estas clases en su código para leer, modificar y escribir documentos XML.

Nota: el código generado puede incluir otras clases compatibles que no se enumeran en este apartado y que pueden cambiar en el futuro.

16.10.1 com.altova.types.DateTime

Esta clase permite procesar atributos o elementos XML que tienen tipos de fecha y hora (p. ej. `xs:dateTime`).

Constructores

	Nombre	Descripción
c	`public DateTime()`	Inicializa una instancia nueva de la clase `DateTime` en un valor vacío.
c	`public DateTime(DateTime newvalue)`	Inicializa una instancia nueva de la clase `DateTime` en el valor `DateTime` dado como argumento.
c	`public DateTime(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond, int newoffsetTZ)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto, segundo, la parte fraccionaria del segundo y la zona horaria dados como argumentos. La parte fraccionaria del segundo `newpartsecond` debe estar comprendida entre 0 y 1. La diferencia de zona horaria `newoffsetTZ` puede ser negativa o positivo y se expresa en minutos.
c	`public DateTime(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes, día, hora, minuto, segundo y la parte fraccionaria de un segundo dados como argumentos.
c	`public DateTime(int newyear, int newmonth, int newday)`	Inicializa una instancia nueva de la clase `DateTime` en el año, mes y día dados como argumentos.
c	`public DateTime(Calendar newvalue)`	Inicializa una instancia nueva de la clase `DateTime` en el valor `java.util.Calendar` dado como argumento.

Métodos

	Nombre	Descripción
S	`static DateTime now()`	Devuelve la hora actual como un objeto `DateTime`.

S	`static DateTime parse(String s)`	Devuelve un objeto `DateTime` analizado desde el valor de cadena dado como argumento. Por ejemplo, estos valores de cadena de muestra se convertirían correctamente en un objeto `DateTime`:
		
		`2015-11-24T12:54:47.969+01:00`
		`2015-11-24T12:54:47`
		`2015-11-24`

	`int getDay()`	Devuelve el día de la instancia `DateTime` actual.
	`int getHour()`	Devuelve la hora de la instancia `DateTime` actual.
	`int getMillisecond()`	Devuelve el milisegundo de la instancia `DateTime` actual, como valor entero.
	`int getMinute()`	Devuelve el minuto de la instancia `DateTime` actual.
	`int getMonth()`	Devuelve el mes de la instancia `DateTime` actual.
	`double getPartSecond()`	Devuelve la parte fraccionaria del segundo de la instancia `DateTime` actual, como valor `double`. El valor devuelto es mayor que cero y menor que uno. Por ejemplo: `0.313`
	`int getSecond()`	Devuelve el segundo de la instancia `DateTime` actual.
	`int getTimezoneOffset()`	Devuelve la diferencia de zona horaria, en minutos, de la instancia `DateTime` actual. Por ejemplo, la zona horaria "UTC-01:00" se devolvería como: `-60`
	`Calendar getValue()`	Devuelve la instancia `DateTime` actual como un valor `java.util.Calendar`.
	`int getWeekday()`	Devuelve el día de la semana de la instancia `DateTime` actual. Los valores están comprendidos entre el 0 y el 6, siendo 0 el lunes (ISO-8601).
	`int getYear()`	Devuelve el año de la instancia `DateTime` actual.
	`int hasTimezone()`	Devuelve información sobre la zona horaria de la instancia `DateTime` actual. Los valores devueltos posibles son: `CalendarBase.TZ_MI` No está definida la

© 2018 Altova GmbH

	Nombre	Descripción
		`SSING` diferencia de zona horaria. `CalendarBase.TZ_UTC` La zona horaria es UTC. `CalendarBase.TZ_OFFSET` Está definida una diferencia de zona horaria.
●	`void setDay( int nDay )`	Establece el día de la instancia `DateTime` actual en el valor dado como argumento.
●	`void setHasTimezone( int nHasTZ )`	Establece la información de zona horaria de la instancia `DateTime` actual en el valor dado como argumento. Este método puede utilizarse para eliminar la información de zona horaria o establecer la zona horaria en UTC (hora universal coordinada). Los valores válidos para el argumento `nHasTZ` son: `CalendarBase.TZ_MISSING` Establecer la diferencia de zona horaria como indefinida. `CalendarBase.TZ_UTC` Establecer la zona horaria en UTC. `CalendarBase.TZ_OFFSET` Si el objeto actual tiene una diferencia de zona horaria, dejarla sin cambiar.
●	`void setHour( int nHour )`	Establece la hora de la instancia `DateTime` actual en el valor dado como argumento.
●	`void setMinute( int nMinute )`	Establece el minuto de la instancia `DateTime` actual en el valor dado como argumento.
●	`void setMonth( int nMonth )`	Establece el mes de la instancia `DateTime` actual en el valor dado como argumento.
●	`void setPartSecond( double nPartSecond )`	Establece la parte fraccionaria del segundo de la instancia `DateTime` actual en el valor dado como argumento.
●	`void setSecond( int nSecond )`	Establece el segundo de la instancia `DateTime` actual en el valor dado como argumento.
●	`void setTimezoneOffset( int nOffsetTZ )`	Establece la diferencia de zona horaria de la instancia `DateTime` actual en el valor dado como argumento. El valor `nOffsetTZ` debe ser un entero (negativo o positivo) y debe expresarse en minutos.

	Nombre	Descripción
	`void setYear( int nYear )`	Establece el año de la instancia `DateTime` actual en el valor dado como argumento.
	`String toString()`	Devuelve la representación de cadena de la instancia `DateTime` actual. Por ejemplo: `2015-11-24T15:50:56.968+01:00`

Ejemplos

Antes de usar este ejemplo en su programa debe importar los tipos de Altova:

```
import com.altova.types.*;
```

Este ejemplo muestra cómo se pueden crear objetos `DateTime`:

```
protected static void DateTimeExample1()
{
	// Inicializar una instancia nueva de la clase DateTime en la hora
actual
	DateTime dt = new DateTime(DateTime.now());
	System.out.println("DateTime creado a partir de la fecha y hora
actuales: " + dt.toString());

	// Inicializar una instancia nueva de la clase DateTime aportando las
partes
	DateTime dt1 = new DateTime(2015, 11, 23, 14, 30, 24, .459);
	System.out.println("DateTime a partir de las partes (sin zona horaria):
" + dt1.toString());

	// Inicializar una instancia nueva de la clase DateTime aportando las
partes
	DateTime dt2 = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
	System.out.println("DateTime a partir de las partes (con zona horaria
negativa): " + dt2.toString());

	// Inicializar una instancia nueva de la clase DateTime analizando un
valor de cadena
	DateTime dt3 = DateTime.parse("2015-11-24T12:54:47.969+01:00");

	System.out.println("DateTime analizado a partir de la cadena: " +
dt3.toString());
}
```

Este ejemplo muestra cómo se pueden obtener valores de objetos `DateTime`:

```
protected static void DateTimeExample2()
   {
```


© 2018 Altova GmbH

```
      // Inicializar una instancia nueva de la clase DateTime en la hora
actual
      DateTime dt = new DateTime(DateTime.now());

      // Generar el año, mes y día de esta instancia DateTime con formato
      String str1 = String.format("Año: %d; Mes: %d; Día: %d;", dt.getYear(),
dt.getMonth(), dt.getDay());
      System.out.println(str1);

      // Generar la hora, el minuto y el segundo de esta instancia DateTime
con formato
      String str2 = String.format("Hora: %d; Minuto: %d; Segundo: %d;",
dt.getHour(), dt.getMinute(), dt.getSecond());
      System.out.println(str2);

      // Devolver la zona horaria (en minutos) de esta instancia DateTime
      System.out.println("Zona horaria: " + dt.getTimezoneOffset());

      // Obtener DateTime como valor java.util.Calendar
      java.util.Calendar dt_java = dt.getValue();
      System.out.println("" + dt_java.toString());

      // Devolver el día de la semana de esta instancia DateTime
      System.out.println("Día de la semana: " + dt.getWeekday());

      // Comprobar si la instancia DateTime tiene definida una zona horaria
      switch(dt.hasTimezone())
      {
         case CalendarBase.TZ_MISSING:
            System.out.println("Sin zona horaria.");
            break;
         case CalendarBase.TZ_UTC:
            System.out.println("La zona horaria es UTC.");
            break;
         case CalendarBase.TZ_OFFSET:
            System.out.println("Este objeto tiene una zona horaria.");
            break;
         default:
            System.out.println("Imposible determinar si se definió una zona
horaria.");
            break;
      }
   }
```

Este ejemplo muestra cómo se puede cambiar la diferencia horaria de un objeto `DateTime`:

```
protected static void DateTimeExample3()
{
       // Crear un objeto DateTime nuevo con zona horaria -0100 UTC
       DateTime dt = new DateTime(2015, 11, 24, 14, 30, 24, .459, -60);
       // Generar el valor antes del cambio
       System.out.println("Antes: " + dt.toString());
       // Cambiar la diferencia a +0100 UTC
```

```
        dt.setTimezoneOffset(60);
        // Generar el valor después del cambio
        System.out.println("Después:  " + dt.toString());
}
```

16.10.2 com.altova.types.Duration

Esta clase permite procesar atributos o elementos XML de tipo `xs:duration`.

Constructores

	Nombre	Descripción
c	`Duration(Duration newvalue)`	Inicializa una instancia nueva de la clase `Duration` en el objeto `Duration` dado como argumento.
c	`Duration(int newyear, int newmonth, int newday, int newhour, int newminute, int newsecond, double newpartsecond, boolean newisnegative)`	Inicializa una instancia nueva de la clase `Duration` en una duración construida con las partes dadas como argumentos.

Métodos

	Nombre	Descripción
s	`static Duration getFromDayTime( int newday, int newhour, int newminute, int newsecond, double newpartsecond )`	Devuelve un objeto `Duration` creado a partir del número de días, horas, minutos, segundos y partes fraccionarias de segundo dados como argumentos.
s	`static Duration getFromYearMonth( int newyear, int newmonth )`	Devuelve un objeto `Duration` creado a partir del número de años y meses dados como argumentos.
s	`static Duration parse( String s )`	Devuelve un objeto `Duration` creado a partir de la cadena dada como argumento. Por ejemplo, la cadena `-P1Y1M1DT1H1M1.333S` puede utilizarse para crear una duración negativa de un año, un mes, un día, una hora, un minuto, un segundo y 0.333 partes fraccionarias de un segundo. Para crear una duración negativa basta con anexar el signo menos (-) a la cadena.

© 2018 Altova GmbH

	Nombre	Descripción
S	`static Duration parse( String s, ParseType pt )`	Devuelve un objeto `Duration` creado a partir de la cadena dada como argumento, usando un formato de análisis concreto. El formato de análisis puede ser cualquiera de estos: **ParseType.DAYTIME** Debe utilizarse cuando la cadena **s** está compuesta por cualquiera de estos: días, horas, minutos, segundos, partes fraccionarias de segundo. Por ejemplo `-P4DT4H4M4.774S`. **ParseType.DURATION** Debe utilizarse cuando la cadena **s** está compuesta por cualquiera de estos: años, meses, días, horas, minutos, segundos, partes fraccionarias de segundo. Por ejemplo `P1Y1M1DT1H1M1.333S`. **ParseType.YEARMONTH** Debe utilizarse cuando la cadena **s** está compuesta por cualquiera de estos: años, meses. Por ejemplo: `P3Y2M`.
	`int getDay()`	Devuelve el número de días de la instancia `Duration` actual.
	`long getDayTimeValue()`	Devuelve el valor de día y hora (en milisegundos) de la instancia `Duration` actual. Años y meses se omiten.
	`int getHour()`	Devuelve el número de horas de la instancia `Duration` actual.
	`int getMillisecond()`	Devuelve el número de milisegundos de la instancia `Duration` actual.
	`int getMinute()`	Devuelve el número de minutos de la instancia `Duration` actual.
	`int getMonth()`	Devuelve el número de meses de la instancia `Duration` actual.
	`double getPartSecond()`	Devuelve el partes fraccionarias de segundo de la instancia `Duration` actual.
	`int getSecond()`	Devuelve el número de segundos de la instancia `Duration` actual.
	`int getYear()`	Devuelve el número de años de la instancia

	Nombre	Descripción
		`Duration` actual.
●	`int getYearMonthValue()`	Devuelve el valor de año y mes (en meses) de la instancia `Duration` actual. Días, horas, segundos y milisegundos se omiten.
●	`boolean isNegative()`	Devuelve el valor booleano **true** si la instancia `Duration` actual es positiva.
●	`void setDayTimeValue(long l)`	Establece la duración en el número de milisegundos dados como argumento, afectando solo a la parte día y hora de la duración.
●	`void setNegative( boolean isnegative )`	Convierte la instancia `Duration` actual en una duración negativa.
●	`void setYearMonthValue(int l)`	Establece la duración en el número de meses dados como argumento. Esto solo afecta la parte años y meses de la duración.
●	`String toString()`	Devuelve la representación de cadena de la instancia `Duration`. Por ejemplo: `-P4DT4H4M4.774S`
●	`String toYearMonthString()`	Devuelve la representación de cadena de la parte YearMonth de la instancia `Duration` actual. Por ejemplo: `P1Y2M`

Ejemplos

Antes de usar este ejemplo en su programa debe importar los tipos de Altova:

```
import com.altova.types.*;
import com.altova.types.Duration.ParseType;
```

Este ejemplo muestra cómo se pueden crear objetos `Duration`:

```
protected static void ExampleDuration()
{
	// Crear una duración negativa de 1 año, 1 mes, 1 día, 1 hora, 1
minuto, 1 segundo y 0.333 partes fraccionarias de segundo
	Duration dr = new Duration(1, 1, 1, 1, 1, 1, .333, true);

	// Crear una duración a partir de un objeto Duration que ya existe
	Duration dr1 = new Duration(dr);
```


© 2018 Altova GmbH

```
        // Crear una duración de 4 días, 4 horas, 4 minutos, 4 segundos, .774
partes fraccionarias de segundo
        Duration dr2 = Duration.getFromDayTime(4, 4, 4, 4, .774);

        // Crear una duración de 3 años y 2 meses
        Duration dr3 = Duration.getFromYearMonth(3, 2);

        // Crear una duración a partir de una cadena
        Duration dr4 = Duration.parse("-P4DT4H4M4.774S");

        // Crear una duración a partir de una cadena, usando formatos de
análisis concretos
        Duration dr5 = Duration.parse("-P1Y1M1DT1H1M1.333S",
ParseType.DURATION);
        Duration dr6 = Duration.parse("P3Y2M", ParseType.YEARMONTH);
        Duration dr7 = Duration.parse("-P4DT4H4M4.774S", ParseType.DAYTIME);
}
```

Este ejemplo muestra cómo se puede obtener y establecer el valor de objetos `Duration`:

```
protected static void DurationExample2()
{
        // Crear una duración de 1 año, 2 meses, 3 días, 4 horas, 5 minutos, 6
segundos y 333 milisegundos
        Duration dr = new Duration(1, 2, 3, 4, 5, 6, .333, false);
        // Generar el número de días de esta duración
        System.out.println(dr.getDay());

        // Crear una duración positiva de un año y 333 milisegundos
        Duration dr1 = new Duration(1, 0, 0, 0, 0, 0, .333, false);
        // Generar el valor de día y hora en milisegundos
        System.out.println(dr1.getDayTimeValue());

        // Crear una duración positiva de 1 año, 1 mes, 1 día, 1 hora, 1
minuto, 1 segundo y 333 milisegundos
        Duration dr2 = new Duration(1, 1, 1, 1, 1, 1, .333, false);
        // Generar el valor de año y mes en meses
        System.out.println(dr2.getYearMonthValue());

        // Crear una duración positiva de 1 año y 1 un mes
        Duration dr3 = new Duration(1, 1, 0, 0, 0, 0, 0, false);
        // Generar el valor
        System.out.println("The duration is now: " + dr3.toString());
        // Establecer la parte DayTime de la duración en 1000 milisegundos
        dr3.setDayTimeValue(1000);
        // Generar el valor
        System.out.println("The duration is now: " + dr3.toString());
        // Establecer la parte YearMonth de la duración en 1 mes
        dr3.setYearMonthValue(1);
        // Generar el valor
        System.out.println("The duration is now: " + dr3.toString());
        // Generar la parte de mes y año de la duración
        System.out.println("The YearMonth part of the duration is: " +
```

```
dr3.toYearMonthString());
}
```

16.10.3 com.altova.xml.meta.Attribute

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de atributos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un atributo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier atributo que esté definido en el esquema XML.

Métodos

	Nombre	Descripción
	`SimpleType getDataType()`	Devuelve el tipo del contenido del atributo.
	`String getLocalName()`	Devuelve el nombre local del atributo.
	`String getNamespaceURI()`	Devuelve el URI de espacio de nombres del atributo.
	`boolean isRequired()`	Devuelve true si el atributo es obligatorio.

16.10.4 com.altova.xml.meta.ComplexType

Esta clase permite acceder a información del esquema sobre clases generadas a partir de tipos complejos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo complejo de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier tipo complejo que esté definido en el esquema XML.

Métodos

	Nombre	Descripción
	`Attribute findAttribute(String localName, String namespaceURI)`	Busca el atributo con el nombre local y el URI de espacio de nombres especificados.
	`Element findElement(String localName, String namespaceURI)`	Busca el elemento con el nombre local y el URI de espacio de nombres especificados.
	`Attribute[]`	Devuelve una lista de todos los atributos.

© 2018 Altova GmbH

	Nombre	Descripción
	`GetAttributes()`	
●	`ComplexType getBaseType()`	Devuelve el tipo base de este tipo.
●	`SimpleType getContentType()`	Devuelve el tipo simple del contenido.
●	`Element[] GetElements()`	Devuelve una lista de todos los elementos.
●	`String getLocalName()`	Devuelve el nombre local de este tipo.
●	`String getNamespaceURI()`	Devuelve el URI de espacio de nombres del tipo.

16.10.5 com.altova.xml.meta.Element

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de elementos. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un elemento de un documento XML. Por el contrario, permite obtener información mediante programación sobre cualquier elemento que esté definido en el esquema XML.

Métodos

	Nombre	Descripción
●	`ComplexType getDataType()`	Devuelve el tipo del elemento. Recuerde que este siempre es un tipo complejo incluso si se declaró como simple en el esquema original. Use `getContentType()` del objeto devuelto para obtener el tipo de contenido simple.
●	`String getLocalName()`	Devuelve el nombre local del elemento.
●	`int getMaxOccurs()`	Devuelve el valor maxOccurs definido en el esquema.
●	`int getMinOccurs()`	Devuelve el valor minOccurs definido en el esquema.
●	`String getNamespaceURI()`	Devuelve el URI de espacio de nombres del elemento.

16.10.6 com.altova.xml.meta.SimpleType

Esta clase permite acceder a la información del esquema sobre clases generadas a partir de tipos simples. Recuerde que esta clase no tiene como objetivo aportar información dinámica sobre las distintas instancias de un tipo simple de un documento XML. Por el contrario, permite

obtener información mediante programación sobre cualquier tipo simple que esté definido en el esquema XML.

Métodos

	Nombre	Descripción
	`SimpleType getBaseType()`	Devuelve el tipo base de este tipo.
	`String[] getEnumerations()`	Devuelve una matriz de todas las facetas de enumeración.
	`int getFractionDigits()`	Devuelve el valor de esta faceta.
	`int getLength()`	Devuelve el valor de esta faceta.
	`String getLocalName()`	Devuelve el nombre local del tipo.
	`String getMaxExclusive()`	Devuelve el valor de esta faceta.
	`String getMaxInclusive()`	Devuelve el valor de esta faceta.
	`int getMaxLength()`	Devuelve el valor de esta faceta.
	`String getMinExclusive()`	Devuelve el valor de esta faceta.
	`String getMinInclusive()`	Devuelve el valor de esta faceta.
	`int getMinLength()`	Devuelve el valor de esta faceta.
	`String getNamespaceURI()`	Devuelve el URI de espacio de nombres del tipo.
	`String[] getPatterns()`	Devuelve una matriz de todas las facetas de patrón.
	`int getTotalDigits()`	Devuelve el valor de esta faceta.
	`int getWhitespace()`	Devuelve el valor de la faceta de espacio en blanco, que es uno de estos: `com.altova.typeinfo.WhitespaceType.Whitespace_Unknown` `com.altova.typeinfo.WhitespaceType.Whitespace_Preserve` `com.altova.typeinfo.WhitespaceType.Whitespace_Replace` `com.altova.typeinfo.WhitespaceType.Whitespace_Collapse`

© 2018 Altova GmbH

16.10.7 com.[SuEsquema].[Doc]

Cuando se genera código a partir de un esquema XML, el código generado ofrece una clase de documento que se llama igual que el esquema. Esta clase tiene como miembros todos los elementos raíz posibles, así como los miembros que aparecen a continuación. Observe que en los nombres de método, `CDoc` corresponde al nombre de la clase de documento generada.

Métodos

	Nombre	Descripción
S	`static Doc createDocument()`	Crea un documento XML nuevo vacío.
S	`static Doc loadFromBinary(byte[] xml)`	Carga un documento XML desde una matriz de bytes.
S	`static Doc loadFromFile(String fileName)`	Carga un documento XML desde un archivo.
S	`static Doc loadFromString(String xml)`	Carga un documento XML desde una cadena de texto.
	`byte[] saveToBinary(boolean prettyPrint)`	Guarda un documento XML en una matriz de bytes, con formato pretty-print opcional.
	`byte[] saveToBinary(boolean prettyPrint, String encoding)`	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada.
	`byte[] saveToBinary(boolean prettyPrint, String encoding, boolean bigEndian, boolean writeBOM)`	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
	`void saveToFile(String fileName, boolean prettyPrint)`	Guarda un documento XML en un archivo, con formato pretty-print opcional.
	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl)`	Guarda un documento XML en una matriz de bytes, con formato pretty-print opcional, con codificación UTF-8. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void saveToFile(String fileName, boolean`	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la

	Nombre	Descripción
	`prettyPrint, boolean omitXmlDecl, String encoding)`	codificación especificada. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void saveToFile(String fileName, boolean prettyPrint, boolean omitXmlDecl, String encoding, boolean bBigEndian, boolean bBOM)`	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
	`void saveToFile(String fileName, boolean prettyPrint, String encoding)`	Guarda un documento XML en un archivo, (con formato pretty-print opcional) con la codificación especificada.
	`void saveToFile(String fileName, boolean prettyPrint, String encoding, boolean bBigEndian, boolean bBOM)`	Guarda un documento XML en una matriz de bytes (con formato pretty-print opcional) con la codificación especificada. Para codificaciones Unicode se puede especificar el orden de bytes y la marca BOM Unicode.
	`String saveToString(boolean prettyPrint)`	Guarda un documento XML en una cadena de texto, con formato pretty-print opcional.
	`String saveToString(boolean prettyPrint, boolean omitXmlDecl)`	Guarda un documento XML en una cadena de texto, con formato pretty-print opcional. Si el argumento `omitXmlDecl` tiene el valor true, la declaración XML no se escribirá.
	`void setSchemaLocation(String schemaLocation)`	Agrega un atributo `xsi:schemaLocation` o `xsi:noNamespaceSchemaLocation` al elemento raíz. Es obligatorio que ya exista un elemento raíz.

16.10.8 com.[SuEsquema].[SuTipoEsquema].MemberAttribute

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada atributo miembro de un tipo. En la tabla que aparece a continuación `AttributeType` corresponde al tipo del atributo miembro propiamente dicho.

Métodos

	Nombre	Descripción
	`boolean exists()`	Devuelve true si el atributo existe.

© 2018 Altova GmbH

	Nombre	Descripción
	`int getEnumerationValue()`	Solo se genera para tipos de enumeración. Devuelve una de las constantes generadas para los posibles valores o "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
	`com.altova.xml.meta.Attribute getInfo()`	Devuelve un objeto para consultar información del esquema (véase `com.altova.xml.meta.Attribute`).
	`AttributeType getValue()`	Obtiene el valor del atributo.
	`void remove()`	Quita el atributo de su elemento primario.
	`void setEnumerationValue(int)`	Solo se genera para tipos de enumeración. Pase una de las constantes generadas para los posibles valores a este método para establecer el valor.
	`void setValue(AttributeType value)`	Establece el valor del atributo.

16.10.9 com.[SuEsquema].[SuTipoEsquema].MemberElement

Cuando se genera código a partir de un esquema XML, se crea una clase como esta por cada elemento miembro de un tipo. En la tabla que aparece a continuación `MemberType` corresponde al tipo del elemento miembro propiamente dicho.

Métodos

	Nombre	Descripción
	`MemberType append()`	Crea un elemento nuevo y lo anexa a su primario.
	`MemberType at(int index)`	Devuelve la instancia del elemento miembro en el índice especificado.
	`int count()`	Devuelve el recuento de elementos.
	`boolean exists()`	Devuelve true si existe al menos un elemento.
	`MemberType first()`	Devuelve la primera instancia del

	Nombre	Descripción
		elemento miembro.
	`int getEnumerationValue()`	Solo se genera para tipos de enumeración. Devuelve una de las constantes generadas para los posibles valores o "no válido" si el valor no coincide con ninguno de los valores enumerados en el esquema.
	`com.altova.xml.meta.Element getInfo()`	Devuelve un objeto para consultar información del esquema (véase `com.altova.xml.meta.Element`).
	`MemberType getValue()`	Obtiene el contexto del elemento (solo se genera si el elemento puede tener contenido simple o mixto).
	`java.util.Iterator iterator()`	Devuelve un objeto para recorrer instancias del elemento miembro.
	`MemberType last()`	Devuelve la última instancia del elemento miembro.
	`void remove()`	Elimina todas las instancias del elemento en su primario.
	`void removeAt(int index)`	Elimina la instancia del elemento especificada por el índice.
	`void setEnumerationValue(int index)`	Solo se genera para tipos de enumeración. Pase una de las constantes generadas para los posibles valores a este método para establecer el valor.
	`void setValue(MemberType value)`	Establece el contexto del elemento (solo se genera si el elemento puede tener contenido simple o mixto).

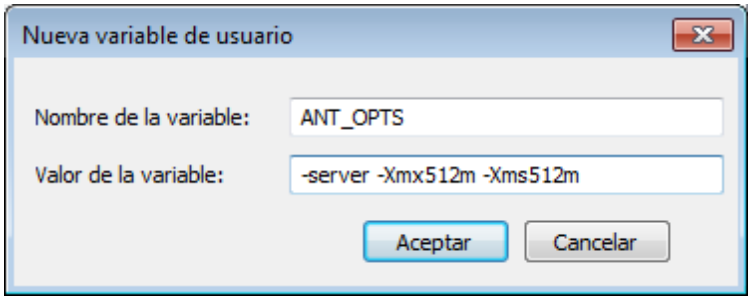
© 2018 Altova GmbH

16.11 Consejos para la generación de código

Solución para las excepciones de memoria insuficiente durante la compilación de Java

Las asignaciones de datos complejas que incluyen esquemas de gran tamaño pueden producir una gran cantidad de código, lo cual puede dar lugar a una excepción java.lang.OutofMemory durante la compilación con Ant. Para resolver esto:

- Añada la variable de entorno **ANT_OPTS**, que asigna al compilador ciertas opciones de Ant, como la opción de memoria. Configure su valor como se muestra a continuación.



- Para asegurarse de que el compilador y el código generado se ejecutan en el mismo proceso que Ant, cambie el valor del atributo **fork**, de build.xml, por **false**.

Quizás sea necesario personalizar los valores, dependiendo de la cantidad de memoria disponible en su equipo y del tamaño del proyecto en el que esté trabajando. Para más información, consulte su documentación de Java VM.

Es posible que, al ejecutar el comando `ant jar`, reciba un mensaje de error parecido a este: "`[...] el archivo contiene más de 65535 entidades`". Para evitar este error, recomendamos utilizar Ant 1.9 o superior y, además, añadir `zip64mode="as-needed"` al elemento `<jar>` en el archivo **build.xml**.

Reservar nombres de métodos

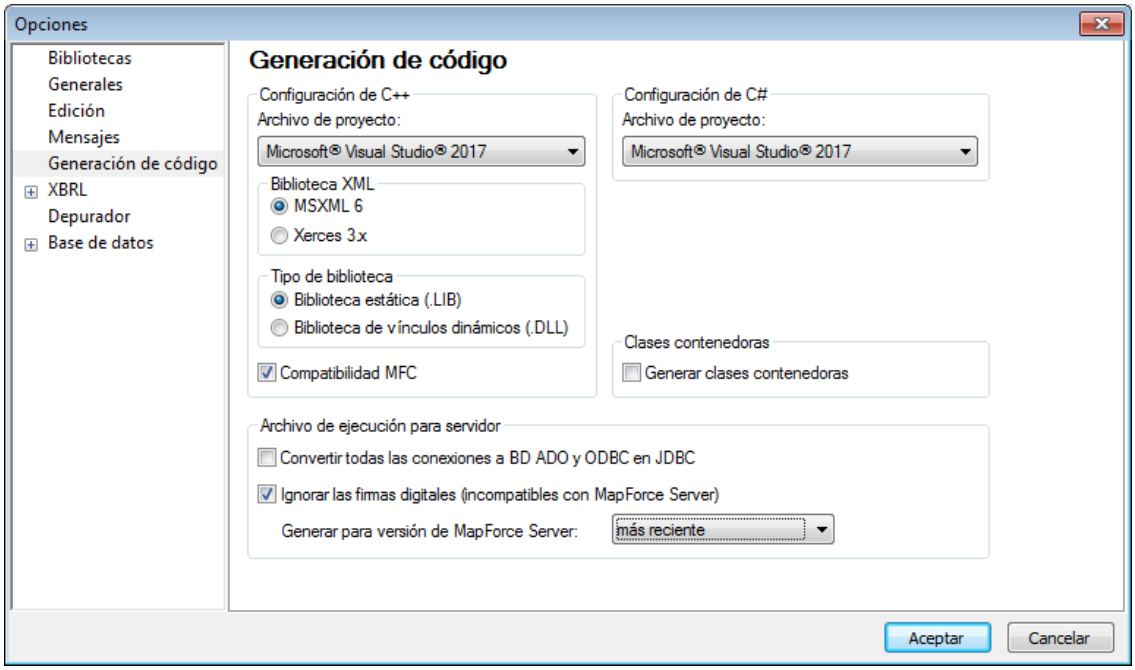
Cuando personalice la generación de código usando los archivos spl que vienen con la aplicación, quizás sea necesario reservar nombres para evitar conflictos con otros símbolos. Para hacerlo:

1. Navegue hasta la subcarpeta del lenguaje de programación de la subcarpeta **spl** del directorio de instalación del programa, p. ej `C:\Archivos de programa\Altova\MapForce2018\spl\java\`.
2. Abra el archivo **settings.spl** e inserte una línea nueva en la sección de reserva, p. ej. reserve "miPalabraReservada".
3. Vuelva a generar el código de programa.

16.12 Opciones del generador de código

Para ver o cambiar la configuración de MapForce para la generación de código:

- En el menú **Herramientas** haga clic en **Opciones** y después en **Generación de código**.



Estas son las opciones de configuración del generador de código:

Configuración de C++	Define la configuración del compilador para el entorno C++: • La edición de Visual Studio (2008, 2010, 2013, 2015, 2017) • La biblioteca XML (MSXML, Xerces 3.x) • Si deben generarse bibliotecas estáticas o dinámicas. • Si debe generarse código compatible con MFC o no.
Configuración de C#	Define la configuración del compilador para el entorno C#, es decir, la edición de Visual Studio (2008, 2010, 2013, 2015, 2017).
Clases contenedoras	Permite generar clases contenedoras para esquemas XML (*véase* *Generar código a partir de esquemas XML o DTD*). Estas clases contenedoras se pueden usar en otro código que incluya el código generado con MapForce.
Archivo de ejecución del servidor	Estas opciones son relevantes cuando se compilan asignaciones en archivos de ejecución de MapForce Server. No afectan a la generación de código C#, C++ ni Java. Para más información consulte el apartado Compilar asignaciones en archivos de

© 2018 Altova GmbH

	ejecución de MapForce Server.

16.13 SPL: el lenguaje de programación Spy

Esta sección ofrece una introducción al lenguaje de programación Spy (en adelante, *SPL*), que es el lenguaje de plantillas del generador de código.

En esta sección asumimos que el usuario tiene cierta experiencia en programación y ciertos conocimientos sobre operadores, variables, funciones y clases, así como sobre conceptos básicos de programación orientada a objetos, que se usa en gran medida en SPL.

Las plantillas utilizadas por MapForce están en la carpeta **...\MapForce2018\spl**. Puede usar estos archivos para orientarse a la hora de crear sus propias plantillas.

¿Cómo funciona el generador de código?

Las entradas del generador de código son los archivos de plantilla (`.spl`) y el modelo de objetos que viene con XMLSpy. Los archivos de plantilla contienen instrucciones SPL (para crear archivos, leer información del modelo de objetos y realizar cálculos) intercaladas con fragmentos de código literal en el lenguaje de programación de destino.

El archivo de plantilla lo interpreta el generador de código y produce archivos de código fuente **`.cpp`**, **`.java`** y **`.cs`** , archivos de proyecto o cualquier otro tipo de proyecto, dependiendo de la plantilla. El código fuente se puede compilar después en un archivo ejecutable que accede a los datos XML descritos por el archivo de esquema.

Los archivos SPL tienen acceso a una gran variedad de información que se recopila de los esquemas de origen. Recuerde que un archivo SPL no está unido a ningún esquema concreto, sino que permite acceder a todos los esquemas. Asegúrese de que escribe sus archivos SPL de forma genérica, evitando estructuras, etc. que pertenezcan a esquemas concretos.

Ejemplo: crear un archivo nuevo en SPL

```
[create "test.cpp"]
#include "stdafx.h"
[close]
```

Se trata de un archivo SPL muy básico. Crea un archivo llamado **test.cpp** y coloca la instrucción `include` dentro del archivo. El comando de cierre (`close`) termina la plantilla.

16.13.1 Estructura básica de SPL

Un archivo SPL contiene texto literal, que se debe reproducir, intercalado con instrucciones del generador de código.

Las instrucciones del generador de código van entre corchetes: "**[**" y "**]**".
Entre cada par de corchetes se pueden introducir varias instrucciones. Las instrucciones se separan con una línea nueva o con dos puntos "**:**".

Estos dos ejemplos son igual de válidos:

Ejemplo nº1

```
[$x = 42
```


© 2018 Altova GmbH

```
$x = $x + 1]
```

Ejemplo nº2

```
[$x = 42: $x = $x + 1]
```

Agregar texto a los archivos

El texto que no aparece entre corchetes **[ ]** se escribe directamente en el archivo de salida actual. Si no hay un archivo de salida actual, el texto se pasa por alto (consulte el apartado Crear archivos de salida). Para reproducir corchetes literales, introduzca el carácter de escape **** de la siguiente manera: **\[** texto **\].** Para reproducir el carácter ****, use ****.

Comentarios

Los comentarios introducidos en un bloque de instrucciones empiezan con el carácter **'** y terminan en la línea siguiente o con un carácter de cierre de bloque: **]**.

16.13.2 Declaraciones

Estas instrucciones se evalúan mientras se analiza el archivo de plantilla SPL. A estas instrucciones **no** les afectan las instrucciones de control de flujo como condiciones, bucles o subrutinas y siempre se evalúan exactamente una vez.

Estas palabras clave, como todas las palabras clave de SPL, no distinguen entre mayúsculas y minúsculas.

Recuerde que todas estas declaraciones deben estar dentro de un bloque delimitado por corchetes.

map *clave nombreasignación* **to** *valor* [, *clave* **to** *valor*]...

Esta instrucción añade información a una asignación. Consulte el resto del apartado para ver usos concretos de esta instrucción.

map *schemanativetype tipoesquema* **to** *tipoindicado*

El tipo integrado de XML Schema indicado se asignará al tipo o clase nativos indicados, usando el formateador indicado. Esta configuración es propia de la versión 2007r3 o posterior. *Tipoindicado* es el nombre de un tipo o clase nativos, seguido de una coma y de la instancia de clase del formateador.

Ejemplo:

```
map schemanativetype "double" to "double,Altova::DoubleFormatter"
```

map type *tipoesquema* **to** *nombreclase*

El tipo integrado de XML Schema indicado se asignará a la clase indicada. Esta configuración es propia de la generación de código para la versión 2007 o anterior.

Ejemplo:

```
map type "float" to "CSchemaFloat"
```

default *configuración* **is** *valor*

Esta instrucción permite determinar cómo se derivan los nombres de clase y miembro desde el esquema XML.
Recuerde que los nombres de configuración sí distinguen entre mayúsculas y minúsculas.

Ejemplo:
```
default "InvalidCharReplacement" is "_"
```

Nombre de configuración	Explicación
`ValidFirstCharSet`	Caracteres permitidos para iniciar un identificador
`ValidCharSet`	Caracteres permitidos para otros caracteres en un identificador
`InvalidCharReplacement`	El carácter que reemplazará todos los caracteres de los nombres que no están en la configuración ValidCharSet
`AnonTypePrefix`	Prefijo para nombres de tipos anónimos*
`AnonTypeSuffix`	Sufijo para nombres de tipos anónimos*
`ClassNamePrefix`	Prefijo para los nombres de las clases generadas
`ClassNameSuffix`	Sufijo para los nombres de las clases generadas
`EnumerationPrefix`	Prefijo para constantes simbólicas declaradas para valores de enumeración
`EnumerationUpperCase`	"on" para poner los nombres de constante de enumeración en mayúsculas
`FallbackName`	Si un nombre tiene sólo caracteres que no están en ValidCharSet, use este

* Los nombres de tipos anónimos se construyen con AnonTypePrefix + nombre de elemento + AnonTypeSuffix

reserve *palabra*

Añade la palabra indicada a la lista de palabras reservadas. Esto garantiza que esa palabra se genere como nombre de clase o miembro.

Ejemplo:
```
reserve "while"
```

include *nombrearchivo*

Ejemplo:
```
include "Module.cpp"
```

incluye el archivo indicado como SPL de origen. Esto permite dividir su plantilla en varios archivos para editarlos y gestionarlos más fácilmente.

16.13.3 Variables

Los archivos SPL importantes exigen el uso de variables. Algunas variables vienen predefinidas por el generador de código y se pueden crear variables nuevas con solo asignarles valores.

El carácter **$** se usa cuando se **declara** o **usa** una variable, un nombre de variable siempre tiene el prefijo **$**.

© 2018 Altova GmbH

Los nombres de variable distinguen **entre mayúsculas y minúsculas**.

Tipos de variables:

- entero, que también se usa como binario, siendo 0 equivalente a false y cualquier otro valor equivale a true
- cadena
- objeto, que viene dado por MapForce
- iterador, ver instrucción foreach

El tipo de variable se declara en la primera asignación de valor:

```
[$x = 0]
```
ahora x es un entero.

```
[$x = "cadenaDePrueba"]
```
ahora x es una cadena.

Cadenas

Las constantes de cadena siempre aparecen entre comillas dobles, como en el ejemplo anterior. **\n** y **\t** dentro de comillas dobles se considera una línea nueva y una tabulación, respectivamente. **\"** es una comilla doble literal y **** es una barra diagonal inversa. Las constantes de cadena también pueden ocupar más de una línea.

Para la concatenación de cadenas se usa el carácter **&**:

```
[$BasePath = $outputpath & "/" & $JavaPackageDir]
```

Objetos

Los objetos representan la información que contiene el esquema XML así como las estructuras de bases de datos, los archivos de texto y las asignaciones. Los objetos tienen propiedades. Con el operador **.** puede acceder a estas propiedades. En SPL no puede crear objetos nuevos (vienen predefinidos por el generador de código y se derivan de la asignación de entrada), pero se pueden asignar objetos a variables.

Ejemplo:

```
class [=$class.Name]
```

Este ejemplo reproduce la palabra "class" seguida de un espacio y el valor de la propiedad **Name** del objeto **$class**.

16.13.4 Variables predefinidas

Después de que el generador de código termine de analizar un archivo de esquema, los objetos de esta tabla existen en el motor de plantillas.

Nombre	Tipo	Descripción
$schematype	entero	1 para DTD, 2 para XML Schema

Nombre	Tipo	Descripción
$TheLibrary	Biblioteca	La biblioteca que se deriva del esquema XML o de la DTD
$module	cadena	Nombre del esquema de origen sin la extensión
$outputpath	cadena	Ruta de acceso de salida especificada por el usuario o ruta de acceso de salida predeterminada

Para generación de código **C++** solamente:

Nombre	Tipo	Descripción
$domtype	entero	1 para MSXML, 2 para Xerces
$libtype	entero	1 para static LIB, 2 para DLL
$mfc	booleano	True si se marcó la casilla *Compatibilidad con MFC*
$VSVersion	entero	Especifica la versión de Visual Studio. Valores válidos: **0** Sin proyecto de Visual Studio **2008** Visual Studio 2008 **2010** Visual Studio 2010 **2013** Visual Studio 2013 **2015** Visual Studio 2015 **2017** Visual Studio 2017

Para generación de código **C#** solamente:

Nombre	Tipo	Descripción
$VSVersion	entero	Especifica la versión de Visual Studio. Valores válidos: **0** Sin proyecto de Visual Studio **2008** Visual Studio 2008 **2010** Visual Studio 2010 **2013** Visual Studio 2013 **2015** Visual Studio 2015 **2017** Visual Studio 2017

© 2018 Altova GmbH

16.13.5 Crear archivos de salida

Estas instrucciones sirven para crear archivos de salida con ayuda de la función de generación de código.

Recuerde que todas estas instrucciones deben estar dentro de un bloque delimitado por corchetes.

create *nombrearchivo*

crea un archivo nuevo. El archivo tiene que cerrarse con la instrucción **close**. El resultado se escribe en el archivo indicado.

Ejemplo:
```
[create $outputpath & "/" & $JavaPackageDir & "/" & $application.Name & ".java"]
package [=$JavaPackageName];

public class [=$application.Name]Application {
...
}
[close]
```

close

cierra el archivo de salida actual

=$*variable*

escribe el valor de la variable indicada en el archivo de salida actual.

Ejemplo:
```
[$x = 20+3]
El resultado de su cálculo es [=$x] - ¡que tenga un buen día!
```

- El archivo de salida incluirá:

```
El resultado de su cálculo es 23 - ¡que tenga un buen día!
```

write *cadena*

escribe la cadena en el archivo de salida actual.

Ejemplo:
```
[write "C" & $name]
```

Esto también se puede escribir de esta manera:
```
C[=$name]
```

filecopy *origen* **to** *destino*

copia el archivo de origen al archivo de destino, sin interpretarlo.

Ejemplo:
```
filecopy "java/mapforce/mapforce.png" to $outputpath & "/" & $JavaPackageDir &
```

```
"/mapforce.png"
```

16.13.6 Operadores

Los operadores de SPL funcionan como en casi todos los lenguajes de programación.

A continuación ofrecemos la lista de operadores de SPL ordenados por orden de prioridad (de mayor a menor):

`.`	Acceso a la propiedad del objeto
`( )`	Agrupación de expresiones
`true`	Constante binaria "true"
`false`	Constante binaria "false"
`&`	Concatenación de cadenas de texto
`-`	Signo para un número negativo
`not`	Negación lógica
`*`	Multiplicación
`/`	División
`%`	Resto
`+`	Suma
`-`	Resta
`<=`	Menor o igual que
`<`	Menor que
`>=`	Mayor o igual que
`>`	Mayor que
`=`	Igual
`<>`	No igual
`and`	Conjunción lógica (con evaluación en cortocircuito)
`or`	Disyunción lógica (con evaluación en cortocircuito)
`=`	Asignación

16.13.7 Condiciones

En SPL puede usar instrucciones `if` estándar. Esta es la sintaxis para estas instrucciones:

```
if condición
       instrucciones
else
       instrucciones
endif
```


© 2018 Altova GmbH

o sin la parte **else**:

```
if condición
	instrucciones
endif
```

Nota: observe que la condición no aparece entre paréntesis.
Al igual que en otros lenguajes de programación, las condiciones se construyen con operadores lógicos y de comparación.

Ejemplo:

```
[if $namespace.ContainsPublicClasses and $namespace.Prefix <> ""]
	lo que usted quiera ['inserta lo que usted quiera en el archivo de
resultados]
[endif]
```

Instrucción switch

SPL también incluye una instrucción de control para múltiples selecciones.

Sintaxis:

```
switch $variable
	case X:
		instrucciones
	case Y:
	case Z:
		instrucciones
	default:
		instrucciones
endswitch
```

Las etiquetas "case" son constantes o variables.

Al igual que C, SPL no admite el paso implícito de una etiqueta case a otra y, por tanto, no es necesario usar la instrucción `break`.

16.13.8 Colecciones y foreach

Colecciones e iteradores

Una colección contiene varios objetos, como una matriz normal y corriente. Los iteradores sirven para almacenar e incrementar índices de matriz al acceder a objetos.

Sintaxis:

```
foreach iterador in colección
	instrucciones
next
```

Ejemplo nº1:

```
[foreach $class in $classes
	if not $class.IsInternal
```

```
            ]      class [=$class.Name];
[     endif
next]
```

Ejemplo 2:

```
[foreach $i in 1 To 3
      Write "// Step " & $i & "\n"
      ‘ Do some work
next]
```

En la primera línea **$classes** es el **objeto global** de todos los tipos generados. Es una colección de diferentes objetos de clase.

La instrucción **foreach** recorre todos los elementos de **$classes** y en cada uno de ellos ejecuta el código que sigue a la instrucción hasta llegar a la instrucción **next**.

En cada iteración se asigna **$class** al siguiente objeto de clase. Simplemente se trabaja con el objeto de clase en lugar de usar `classes[i]->Name()` como en C++.

Los iteradores de colección tienen estas propiedades adicionales:

`Index`	El índice actual, empezando por 0
`IsFirst`	true si el objeto actual es el primer objeto de la colección (el índice es 0)
`IsLast`	true si el objeto actual es el último objeto de la colección
`Current`	El objecto actual (este es implícito si no se especifica y se puede omitir)

Ejemplo:

```
[foreach $enum in $facet.Enumeration
       if not $enum.IsFirst
              ], [
       endif
       ]"[=$enum.Value]"[
next]
```

Rutinas para la manipulación de colecciones:

collection **SortByName**(bAscending)
devuelve una colección cuyos elementos están ordenados por nombre (con distinción de mayúsculas y minúsculas) en orden ascendente o descendente.

collection **SortByNameNoCase**(bAscending)
devuelve una colección cuyos elementos están ordenados por nombre (sin distinción de mayúsculas y minúsculas) en orden ascendente o descendente.

Ejemplo:

```
$SortedNestedClassifier = $Class.nestedClassifier.SortByNameNoCase( true )
```

collection **SortByKind**(bAscending)
devuelve una colección cuyos elementos están ordenados por tipo (p. ej. clase, interfaz...) en orden ascendente o descendente.

© 2018 Altova GmbH

collection **SortByKindAndName**(bAscendingKind, bAscendingName)
devuelve una colección cuyos elementos están ordenados por tipo (p. ej. clase, interfaz...) en orden ascendente o descendente y si el tipo es idéntico al nombre (con distinción de mayúsculas y minúsculas en orden ascendente o descendente).

collection **SortByKindAndNameNoCase**(bAscending)
devuelve una colección cuyos elementos están ordenados por tipo (p. ej. clase, interfaz...) en orden ascendente o descendente y si el tipo es idéntico al nombre (sin distinción de mayúsculas y minúsculas en orden ascendente o descendente).

16.13.9 Subrutinas

El generador de código admite subrutinas en forma de procedimientos o funciones.

Características:
- Se pasan valores por valor y por referencia
- Parámetros locales/globales (locales dentro de las subrutinas)
- Variables locales
- Invocación
- Invocación recurrente (las subrutinas se pueden llamar a sí mismas)

16.13.9.1 Declaración de subrutinas

Subrutinas

Ejemplo de sintaxis:

```
Sub SimpleSub()
... lines of code
EndSub
```

- **Sub** es la palabra clave que denota el procedimiento.
- **SimpleSub** es el nombre asignado a la subrutina.
- Los **paréntesis** pueden contener una lista de parámetros.
- El bloque de código de una subrutina empieza inmediatamente después del paréntesis de cierre.
- **EndSub** denota el final del bloque de código.

Nota: no se permiten **declaraciones** de subrutinas recursivas o en cascada (es decir, una subrutina no puede contener otra subrutina).

Parámetros
Los parámetros también se pueden pasar con procedimientos usando esta sintaxis:

- Todos los parámetros deben ser variables.
- Las variables deben tener el prefijo **$**.
- Las variables locales se definen en una subrutina.
- Las variables globales se declaran de forma explícita, fuera de las subrutinas.

- Si hay varios parámetros, se separan con comas dentro de los paréntesis.
- Los parámetros pueden pasar valores.

Parámetros: pasar valores
Los parámetros se pueden pasar de dos maneras: por valor y por referencia, usando las palabras clave **ByVal** y **ByRef** respectivamente.

Sintaxis:
```
' definir sub CompleteSub()
[Sub CompleteSub( $param, ByVal $paramByValue, ByRef $paramByRef )
] ...
```

- **ByVal** especifica que el parámetro se pasa por valor. Recuerde que la mayoría del os objetos solamente se pueden pasar por referencia.
- **ByRef** especifica que el parámetro se pasa por referencia. Se trata del método predeterminado si no se especifica ByVal ni ByRef.

Valores devueltos de funciones
Para devolver un valor desde una subrutina use la instrucción **return**. Este tipo de función se puede llamar desde dentro de una expresión.

Ejemplo:
```
' definir una función
[Sub MakeQualifiedName( ByVal $namespacePrefix, ByVal $localName )
if $namespacePrefix = ""
  return $localName
else
  return $namespacePrefix & ":" & $localName
endif
EndSub
]
```

16.13.9.2 Invocación de subrutinas

Use **call** para invocar una subrutina, seguido del nombre y los parámetros del procedimiento (si procede).

```
       Call SimpleSub()
o
       Call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
```

Invocación de funciones
Para invocar una función (cualquier subrutina que contenga una instrucción **return**) use su nombre dentro de una expresión. No use la instrucción **call** para llamar a funciones.

Ejemplo:
```
       $QName = MakeQualifiedName($namespace, "entry")
```


© 2018 Altova GmbH

16.13.9.3 Ejemplo

Este ejemplo muestra la declaración e invocación de subrutinas.

```
[create $outputpath & $module & "output.txt"

' definir sub SimpleSub()
Sub SimpleSub()
]SimpleSub() called
[endsub

' ejecutar sub SimpleSub()
Call SimpleSub()

$ParamByValue       = "Original Value"
]ParamByValue       = [=$ParamByValue]
[$ParamByRef  = "Original Value"
]ParamByRef   = [=$ParamByRef]

' definir sub CompleteSub()
[Sub CompleteSub( $param, ByVal $paramByValue, ByRef $paramByRef )
]CompleteSub called.
       param = [=$param]
       paramByValue = [=$paramByValue]
       paramByRef = [=$paramByRef]
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
]      Set values inside Sub
[$ParamByRef = "Local Variable"
$paramByValue = "new value"
$paramByRef = "new value"
]CompleteSub finished.
[endsub

' ejecutar sub CompleteSub()
Call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
]
ParamByValue=[=$ParamByValue]
ParamByRef=[=$ParamByRef]
[
Close
]
```

16.13.10 Tipos integrados

Esta sección describe las propiedades de los tipos integrados usados en las variables predefinidas que describen el esquema analizado.

16.13.10.1 Library

Este objeto representa toda la biblioteca generada a partir del esquema XML o de la DTD.

Propiedad	Tipo	Descripción
SchemaNamespaces	ColecciónNamespace	Los espacios de nombres de esta biblioteca
SchemaFilename	string	Nombre del XSD o DTD del que se deriva esta biblioteca
SchemaType	integer	1 para la DTD, 2 para el esquema XML
Guid	string	Un ID único global
CodeName	string	El nombre de la biblioteca generada (derivado del nombre del archivo de esquema)

16.13.10.2 Namespace

Se genera un objeto namespace por cada espacio de nombres del esquema XML. Los componentes del esquema que no están en ningún espacio de nombres se incluyen en un objeto namespace especial con un NamespaceURI vacío. Recuerde que para las DTD también se derivan espacios de nombres de los atributos cuyo nombre comienza por "xmlns".

Propiedad	Tipo	Descripción
CodeName	string	Nombre para el código generado (derivado del prefijo)
LocalName	string	Prefijo del espacio de nombres
NamespaceURI	string	URI del espacio de nombres
Types	Colección Type	Todos los tipos incluidos en este espacio de nombres
Library	Library	La biblioteca que contiene este espacio de nombres

16.13.10.3 Type

Este objeto representa un tipo simple o complejo. Se usa para generar una clase en el lenguaje de destino.
Hay un tipo adicional por cada biblioteca que representa al documento, que tiene como miembros todos los elementos raíz posibles.
Los tipos anónimos tienen la propiedad LocalName vacía.

Propiedad	Tipo	Descripción
CodeName	string	Nombre para el código generado (derivado del nombre local o de la declaración de objeto primario)
LocalName	string	Nombre original en el esquema
Namespace	Namespace	Espacio de nombres que contiene este tipo

© 2018 Altova GmbH

Attributes	ColecciónMember	Atributos que contiene este tipo*
Elements	ColecciónMember	Elementos secundarios que contiene este tipo
IsSimpleType	boolean	True para tipos simples y false para tipos complejos
IsDerived	boolean	True si el tipo se deriva de otro tipo, lo cual también se representa con un objeto Type
IsDerivedByExtension	boolean	True si el tipo se deriva por extensión
IsDerivedByRestriction	boolean	True si el tipo se deriva por restricción
IsDerivedByUnion	boolean	True si el tipo se deriva mediante union
IsDerivedByList	boolean	True si el tipo se deriva mediante list
BaseType	Type	El tipo base de este tipo (si IsDerived es true)
IsDocumentRootType	boolean	True si este tipo representa al documento propiamente dicho
Library	Library	La biblioteca que contiene este tipo
IsFinal	boolean	True si se declaró como final en el esquema
IsMixed	boolean	True si este tipo puede tener contenido mixto
IsAbstract	boolean	True si este tipo se declaró como abstracto
IsGlobal	boolean	True si este tipo se declaró globalmente en el esquema
IsAnonymous	boolean	True si este tipo se declaró localmente en un elemento

Para tipos simples solamente:

Propiedad	Tipo	Descripción
IsNativeBound	boolean	True si existe un NativeBinding para este tipo
NativeBinding	NativeBinding	El NativeBinding para este tipo
Facets	Facets	Las facetas de este tipo
Whitespace	string	Acceso rápido a la faceta Whitespace

* Los tipos complejos que tienen contenido de texto (es decir, tipos que tienen contenido mixto y un complexType con simpleContent) tienen un miembro atributo adicional sin nombre que representa el contenido de texto.

16.13.10.4 Member

Este objeto representa a un atributo o elemento del esquema XML. Se usa para crear miembros de clases de tipos.

Propiedad	Tipo	Descripción
CodeName	string	El nombre para el código generado (derivado del nombre local o de la declaración del objeto primario)
LocalName	string	El nombre original en el esquema. Aparece vacía para el miembro especial que representa

		el contenido de texto de tipos complejos.
NamespaceURI	string	El URI de espacio de nombres de este elemento/atributo dentro de documentos/ secuencias de instancia XML.
DeclaringType	Type	El tipo que originalmente declara el miembro (igual a containingType para miembros no heredados)
ContainingType	Type	El tipo del cual este es miembro
DataType	Type	El tipo de datos del contenido de este miembro
Library	Library	La biblioteca que contiene el DataType de este miembro
IsAttribute	boolean	True para atributos, falso para elementos
IsOptional	boolean	True si minOccurs = 0 o si es un atributo opcional
IsRequired	boolean	True si minOccurs > 0 o si es un atributo obligatorio
IsFixed	boolean	True para atributos fijos, el valor está en la propiedad Default
IsDefault	boolean	True para los atributos con valor predeterminado, el valor está en la propiedad Default
IsNillable	boolean	True para elementos nillable
IsUseQualified	boolean	True si la propiedad NamespaceURI no está vacía
MinOccurs	integer	minOccurs como en el esquema. 1 para atributos obligatorios
MaxOccurs	integer	maxOccurs como en el esquema. 0 para atributos prohibidos y -1 para unbounded
Default	string	Valor predeterminado

16.13.10.5 NativeBinding

Este objeto representa el enlace de un tipo simple con un tipo nativo en el lenguaje de programación de destino, tal y como indica la asignación "schemanativetype".

Propiedad	**Tipo**	**Descripción**
ValueType	string	Tipo nativo
ValueHandler	string	Instancia de clase de formateador

16.13.10.6 Facets

Este objeto representa todas las facetas de un tipo simple.
Las facetas heredadas se combinan con las facetas declaradas de forma explícita. Si hay una faceta Length, entonces MinLength y MaxLength tienen el mismo valor.

Propiedad	**Tipo**	**Descripción**
DeclaringType	Type	El tipo en el que están declaradas las facetas

© 2018 Altova GmbH

`Whitespace`	`string`	"preserve", "collapse" o "replace"
`MinLength`	`integer`	Valor de faceta
`MaxLength`	`integer`	Valor de faceta
`MinInclusive`	`integer`	Valor de faceta
`MinExclusive`	`integer`	Valor de faceta
`MaxInclusive`	`integer`	Valor de faceta
`MaxExclusive`	`integer`	Valor de faceta
`TotalDigits`	`integer`	Valor de faceta
`FractionDigits`	`integer`	Valor de faceta
`List`	`Colección Facet`	Todas las facetas en forma de lista

Facet

Este objeto representa una sola faceta cuyo valor calculado está vigente en un tipo concreto.

Propiedad	**Tipo**	**Descripción**
`LocalName`	`string`	Nombre de la faceta
`NamespaceURI`	`string`	Espacio de nombres de la faceta
`FacetType`	`string`	Puede ser "normalization", "lexicalspace", "valuespace-length", "valuespace-enum" o "valuespace-range"
`DeclaringType`	`Type`	El tipo en el que se declara esta faceta
`FacetCheckerName`	`string`	El nombre del Facet Checker (de la asignación schemafacet)
`FacetValue`	`string o integer`	El valor real de esta faceta

Altova MapForce 2018 Enterprise Edition

The MapForce API

17 The MapForce API

The COM-based API of MapForce enables clients to access the functionality of MapForce from a custom code or application, and automate a wide range of tasks.

The MapForce COM API follows the common specifications for automation servers as set out by Microsoft. MapForce is automatically registered as a COM server object during installation. Once the COM server object is registered, you can invoke it from within applications and scripting languages that have programming support for COM calls. This makes it possible to access the MapForce API not only from development environments using .NET, C++ and Visual Basic, but also from scripting languages like JScript and VBScript.

Note: If you use the MapForce API to create an application that you intend to distribute to other clients, MapForce must be installed on each client computer. Also, your custom integration code must be deployed to (or your application installed on) each client computer.

© 2018 Altova GmbH

17.1 Overview

This overview of the API de MapForce provides you with the object model for the API and a description of the most important API concepts. The following topics are covered:

- Accessing the API
- The Object Model
- Error Handling
- Examples

17.1.1 Accessing the API

To access the MapForce COM API, a new instance of the `Application` object must be created in your application (or script). Once this object is created, you can interact with MapForce by invoking its methods and properties as required (for example, create a new document, open an existing document, generate mapping code, etc).

Prerequisites

To make the MapForce COM object available in your Visual Studio project, add a reference to the MapForce type library (.tlb) file. The following instructions are applicable to Visual Studio 2013, but are similar in other Visual Studio versions:

1. On the **Project** menu, click **Add Reference**.
2. Click **Browse** and select the **MapForce.tlb** file located in the MapForce installation folder.

A sample MapForce API client in C# is available at: **(My) Documents\Altova\MapForce2018 \MapForceExamples\API\C#**.

In Java, the MapForce API is available through Java-COM bridge libraries. These libraries are available in the MapForce installation folder: **C:\Program Files (x86)\Altova\MapForce2018 \JavaAPI** (note this path is valid when 32-bit MapForce runs on 64-bit Windows, otherwise adjust the path accordingly).

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `MapForceAPI.jar`: Java classes that wrap the MapForce automation interface
- `MapForceAPI_JavaDoc.zip`: a `Javadoc` file containing help documentation for the Java API

To allow access to the MapForce automation server directly from Java code, the libraries above must be in the Java `classpath`.

A sample MapForce API client in Java is available at: **(My) Documents\Altova\MapForce2018 \MapForceExamples\API\Java**.

In scripting languages such as JScript or VBScript, the MapForce COM object is accessible through the Microsoft Windows Script Host (see https://msdn.microsoft.com/en-us/library/9bbdkx3k.aspx). Such scripts can be written with a text editor, and do not need

compilation, since they are executed by the Windows Script Host packaged with Windows. (To check that the Windows Script Host is running, type `wscript.exe /?` at the command prompt). A sample MapForce API client in JScript is available at: **(My) Documents\Altova\MapForce2018\MapForceExamples\API\JScript**.

Note: For 32-bit MapForce, the registered name, or programmatic identifier (ProgId) of the COM object is `MapForce.Application`. For 64-bit MapForce, the name is `MapForce_x64.Application`.

Guidelines

The following guidelines should be considered in your client code:

- Do not hold references to objects in memory longer than you need them. If a user interacts between two calls of your client, then there is no guarantee that these references are still valid.
- Be aware that if your client code crashes, instances of MapForce may still remain in the system. For details on how to avoid error messages, see Error handling.
- Free references explicitly, if using languages such as C++.

Creating the Application object

The syntax to create the starting `Application` object depends on the programming language, as shown in the examples below:

Visual Basic

```
Dim objMapForce As MapForceLib.Application = New MapForceLib.Application
```

VBA

```
' Create a new instance of MapForce.
Dim objMapForce As Application
Set objMapForce = CreateObject("MapForce.Application")
```

VBScript

```
' Access a running instance, or create a new instance of MapForce.
Set objMapForce = GetObject("MapForce.Application");
```

C#

```
// Create a new instance of MapForce via its automation interface.
MapForceLib.Application objMapForce = new MapForceLib.Application();
```


© 2018 Altova GmbH

Java

```
// Start MapForce as COM server.
com.altova.automation.MapForce.Application objMapForce = new Application();
// COM servers start up invisible so we make it visible
objMapForce.setVisible(true);
```

JScript

```
// Access a running instance, or create a new instance of MapForce.
try
{
	objMapForce = WScript.GetObject ("", "MapForce.Application");
	// unhide application if it is a new instance
	objMapForce.Visible = true;
}
catch(err) { WScript.Echo ("Can't access or create MapForce.Application"); }
```

17.1.2 The Object Model

The starting point for every application which uses the MapForce API is the `Application` object. All other interfaces are accessed through the `Application` object as the starting point.

The object model of the MapForce API can be represented as follows (each indentation level indicates a child–parent relationship with the level directly above):

```
Application
   Options
   Project
      ProjectItem
   Documents
      Document
         MapForceView
         Mapping
            Component
               Datapoint
            Components
            Connection
         Mappings
   ErrorMarkers
      ErrorMarker
AppOutputLines
   AppOutputLine
      AppOutputLines
         ...
      AppOutputLineSymbol
Enumerations
```

For information about creating an instance the `Application` object, see Accessing the API. For reference to the objects exposed by the API, see Object Reference.

17.1.3 Error Handling

The MapForce API returns errors in two different ways. Every API method returns an `HRESULT`. This return value informs the caller about any malfunctions during the execution of the method. If the call was successful, the return value is equal to `S_OK`. C/C++ programmers generally use `HRESULT` to detect errors.

Visual Basic, scripting languages, and other high-level development environments do not give the programmer access to the returning `HRESULT` of a COM call. They use the second error-raising mechanism supported by the MapForce API, the `IErrorInfo` interface. If an error occurs, the API creates a new object that implements the `IErrorInfo` interface. The development environment takes this interface and fills its own error-handling mechanism with the provided information.

The following text describes how to deal with errors raised from the MapForce API in different development environments.

Visual Basic

A common way to handle errors in Visual Basic is to define an error handler. This error handler can be set with the `On Error` statement. Usually the handler displays an error message and does some cleanup to avoid spare references and any kind of resource leaks. Visual Basic fills its own `Err` object with the information from the `IErrorInfo` interface.

```
Sub Validate()
  'place variable declarations here

  'set error handler
  On Error GoTo ErrorHandler

  'if generation fails, program execution continues at ErrorHandler:
  objMapForce.ActiveDocument.GenerateXSLT()

  'additional code comes here

  'exit
  Exit Sub

  ErrorHandler:
  MsgBox("Error: " & (Err.Number - vbObjectError) & Chr(13) &
     "Description: " & Err.Description)
End Sub
```

JavaScript

The Microsoft implementation of JavaScript (JScript) provides a try-catch mechanism to deal with errors raised from COM calls. It is very similar to the VisualBasic approach, in that you also declare an error object containing the necessary information.

© 2018 Altova GmbH

```
function Generate()
{
   // please insert variable declarations here

   try
   {
      objMapForce.ActiveDocument.GenerateXSLT();
   }
   catch(Error)
   {
      sError = Error.description;
      nErrorCode = Error.number & 0xffff;
      return false;
   }

   return true;
}
```

C/C++

C/C++ gives you easy access to the `HRESULT` of the COM call and to the `IErrorInterface`.

```
HRESULT hr;

// Call GenerateXSLT() from the MapForce API
if(FAILED(hr = ipDocument->GenerateXSLT()))
{
   IErrorInfo *ipErrorInfo = Null;

   if(SUCCEEDED(::GetErrorInfo(0, &ipErrorInfo)))
   {
      BSTR   bstrDescr;
      ipErrorInfo->GetDescription(&bstrDescr);

      // handle Error information
      wprintf(L"Error message:\t%s\n",bstrDescr);
      ::SysFreeString(bstrDescr);

      // release Error info
      ipErrorInfo->Release();
   }
}
```

17.1.4 Examples

Programming languages differ in the way they support COM access. The following examples for C#, Java, and JScript will help you get started. The code listings in this section are available at **C:/Users/<username>/Documents/Altova/MapForce2018/MapForceExamples/API**.

17.1.4.1 Example C# Project

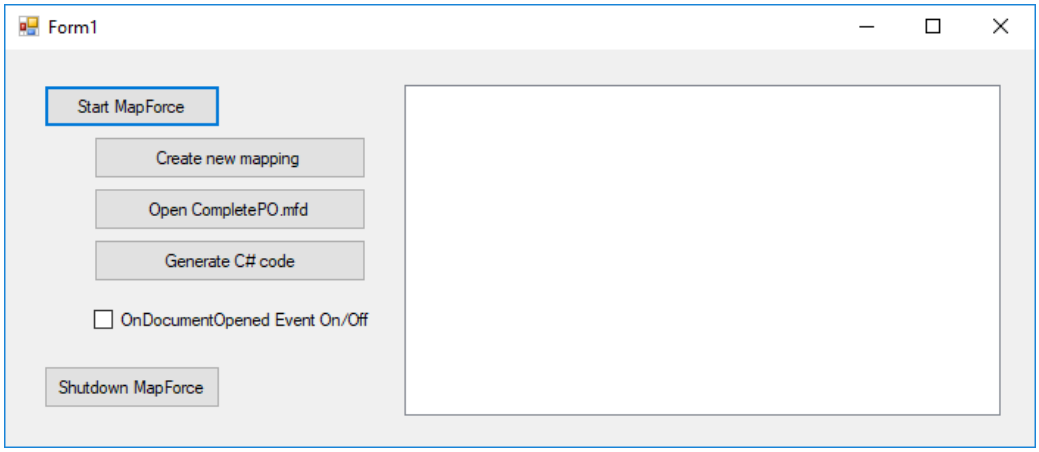
After you install MapForce, an example MapForce API client project for C# is available in the directory **C:/Users/<username>/Documents/Altova/MapForce2018/MapForceExamples/API**.

You can compile and run the project with Visual Studio 2008 or later. To compile and run the example, open the solution .sln file in Visual Studio and run **Debug | Start Debugging** or **Debug | Start Without Debugging**.

Note: If you have a 64-bit operating system and are using a 32-bit installation of MapForce, add the **x86** platform in the solution's Configuration Manager and build the sample using this configuration. A new x86 platform (for the active solution in Visual Studio) can be created in the New Solution Platform dialog (**Build | Configuration Manager | Active solution platform | <New...>**).

When you run the example, a Windows form is displayed, containing buttons that invoke basic MapForce operations:

- Start MapForce
- Create a new mapping design
- Open the CompletePO.mfd file from the **...\MapForceExamples** folder (note that you may need to adjust the path to point to the **\MapForceExamples** folder on your machine)
- Generate C# code in a temp directory
- Shut down MapForce



Code listing

The listing is commented for ease of understanding. The code essentially consists of a series of handlers for the buttons in the user interface shown above.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
```

```
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace WindowsFormsApplication2
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        // An instance of MapForce accessed via its automation interface.
        MapForceLib.Application MapForce;

        // Location of examples installed with MapForce
        String strExamplesFolder;

        private void Form1_Load(object sender, EventArgs e)
        {
        }

        // handler for the "Start MapForce" button
        private void StartMapForce_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
            {
                Cursor.Current = Cursors.WaitCursor;

                // if we have no MapForce instance, we create one a nd make it
visible.
                MapForce = new MapForceLib.Application();
                MapForce.Visible = true;

                // locate examples installed with MapForce.
                int majorVersionYear = MapForce.MajorVersion + 1998;
                strExamplesFolder =
Environment.GetEnvironmentVariable("USERPROFILE") + "\\My Documents\\Altova\
\MapForce" + Convert.ToString(majorVersionYear) + "\\MapForceExamples\\";

                Cursor.Current = Cursors.Default;
            }
            else
            {
                // if we have already an MapForce instance running we toggle
its visibility flag.
                MapForce.Visible = !MapForce.Visible;
            }
        }

        // handler for the "Open CompletePO.mfd" button
        private void openCompletePO_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
                StartMapForce_Click(null, null);

            // Open one of the sample files installed with the product.
            MapForce.OpenDocument(strExamplesFolder + "CompletePO.mfd");
        }
```

```
        // handler for the "Create new mapping" button
        private void newMapping_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
                StartMapForce_Click(null, null);

            // Create a new mapping
            MapForce.NewMapping();
        }

        // handler for the "Shutdown MapForce" button
        // shut-down application instance by explicitly releasing the COM
object.
        private void shutdownMapForce_Click(object sender, EventArgs e)
        {
            if (MapForce != null)
            {
                // allow shut-down of MapForce by releasing UI
                MapForce.Visible = false;

                // explicitly release COM object
                try
                {
                    while
(System.Runtime.InteropServices.Marshal.ReleaseComObject(MapForce) > 0) ;
                }
                finally
                {
                    // avoid later access to this object.
                    MapForce = null;
                }
            }
        }

        // handler for button "Generate C# Code"
        private void generateCppCode_Click(object sender, EventArgs e)
        {
            if (MapForce == null)
                listBoxMessages.Items.Add("start MapForce first.");
            // COM errors get returned to C# as exceptions. We use a try/catch
block to handle them.
            try
            {
                MapForceLib.Document doc = MapForce.ActiveDocument;

                listBoxMessages.Items.Add("Active document " + doc.Name);
                doc.GenerateCHashCode();

            }
            catch (Exception ex)
            {
                // The COM call was not successful.
                // Probably no application instance has been started or no
document is open.
                MessageBox.Show("COM error: " + ex.Message);
            }
        }

        delegate void addListBoxItem_delegate(string sText);
```


© 2018 Altova GmbH

```
        // called from the UI thread
        private void addListBoxItem(string sText)
        {
            listBoxMessages.Items.Add(sText);
        }
        // wrapper method to allow to call UI controls methods from a worker
thread
        void syncWithUIthread(Control ctrl, addListBoxItem_delegate
methodToInvoke, String sText)
        {
            // Control.Invoke: Executes on the UI thread, but calling thread
waits for completion before continuing.
            // Control.BeginInvoke: Executes on the UI thread, and calling
thread doesn't wait for completion.
            if (ctrl.InvokeRequired)
                ctrl.BeginInvoke(methodToInvoke, new Object[] { sText });
        }

        // event handler for OnDocumentOpened event
        private void handleOnDocumentOpened(MapForceLib.Document i_ipDocument)
        {
            String sText = "";

            if (i_ipDocument.Name.Length > 0)
                sText = "Document " + i_ipDocument.Name + " was opened!";
            else
                sText = "A new mapping was created.";

            // we need to synchronize the calling thread with the UI thread
because
            // the COM events are triggered from a working thread
            addListBoxItem_delegate methodToInvoke = new
addListBoxItem_delegate(addListBoxItem);
            // call syncWithUIthread with the following arguments:
            // 1 - listBoxMessages - list box control to display messages from
COM events
            // 2 - methodToInvoke  - a C# delegate which points to the method
which will be called from the UI thread
            // 3 - sText           - the text to be displayed in the list box
            syncWithUIthread(listBoxMessages, methodToInvoke, sText);
        }

        private void checkBoxEventOnOff_CheckedChanged(object sender,
EventArgs e)
        {
            if (MapForce != null)
            {
                if (checkBoxEventOnOff.Checked)
                    MapForce.OnDocumentOpened += new
MapForceLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentO
pened);
                else
                    MapForce.OnDocumentOpened -= new
MapForceLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentO
pened);
            }
        }
    }
}
```

17.1.4.2 Example Java Project

After you install MapForce, an example MapForce API client project for Java is available in the directory **C:/Users/<username>/Documents/Altova/MapForce2018/MapForceExamples/API**.

You can test the Java example directly from the command line, using the batch file `BuildAndRun.bat`, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

File list

The Java examples folder contains all the files required to run the example project. These files are listed below:

`AltovaAutomation.dll`	Java-COM bridge: DLL part
`AltovaAutomation.jar`	Java-COM bridge: Java library part
`MapForceAPI.jar`	Java classes of the MapForce API
`RunMapForce.java`	Java example source code
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`MapForceAPI_JavaDoc.zip`	`Javadoc` file containing help documentation for the Java API

What the example does

The example starts up MapForce and performs a few operations, including opening and closing documents. When done, MapForce stays open. You must close it manually.

Running the example from the command line

To run the example from the command line, open a command prompt window, go to the Java folder of the API Examples folder (*see above for location*), and then type:

```
buildAndRun.bat "<Path-to-the-Java-bin-folder>"
```

The Java binary folder must be that of a Java Development Kit (JDK) 7 or later installation on your computer.

Press the **Return** key. The Java source in `RunMapForce.java` will be compiled and then executed.

Loading the example in Eclipse

Open Eclipse and use the **File | Import... | General | Existing Projects into Workspace** command to add the Eclipse project file (`.project`) located in the Java folder of the API Examples folder (*see above for location*). The project `RunMapForce` will then appear in your Package Explorer or Navigator.

© 2018 Altova GmbH

Select the project and then the command **Run as | Java Application** to execute the example.

Note: You can select a class name or method of the Java API and press F1 to get help for that class or method.

Java source code listing

The Java source code in the example file `RunMapForce.java` is listed below with comments.

```
// access general JAVA-COM bridge classes
import java.util.Iterator;

import com.altova.automation.libs.*;

// access XMLSpy Java-COM bridge
import com.altova.automation.MapForce.*;
import com.altova.automation.MapForce.Enums.ENUMProgrammingLanguage;

/**
 * A simple example that starts XMLSpy COM server and performs a few
operations on it.
 * Feel free to extend.
 */
public class RunMapForce
{
   public static void main(String[] args)
   {
      // an instance of the application.
      Application mapforce = null;

      // instead of COM error handling use Java exception mechanism.
      try
      {
         // Start MapForce as COM server.
         mapforce = new Application();
         // COM servers start up invisible so we make it visible
         mapforce.setVisible(true);

         // The following lines attach to the application events using a
default implementation
         // for the events and override one of its methods.
         // If you want to override all document events it is better to derive
your listener class
         // from DocumentEvents and implement all methods of this interface.
         mapforce.addListener(new ApplicationEventsDefaultHandler()
         {
            @Override
            public void onDocumentOpened(Document i_ipDoc) throws
AutomationException
            {
               String name = i_ipDoc.getName();

               if (name.length() > 0)
```

```
                System.out.println("Document " + name + " was opened.");
             else
                System.out.println("A new mapping was created.");
          }
       });

       // Locate samples installed with the product.
       int majorVersionYear = mapforce.getMajorVersion() + 1998;
       String strExamplesFolder = System.getenv("USERPROFILE") + "\
\Documents\\Altova\\MapForce" + Integer.toString(majorVersionYear) + "\
\MapForceExamples\\";
       // create a new MapForce mapping and generate c++ code
       Document newDoc = mapforce.newMapping();
       ErrorMarkers err1 =
newDoc.generateCodeEx(ENUMProgrammingLanguage.eCpp);
       display(err1);
       // open CompletePO.mfd and generate c++ code
       Document doc = mapforce.openDocument(strExamplesFolder +
"CompletePO.mfd");
       ErrorMarkers err2 = doc.generateCodeEx(ENUMProgrammingLanguage.eCpp);
       display(err2);

       doc.close();
       doc = null;

       System.out.println("Watch MapForce!");
    }
    catch (AutomationException e)
    {
       // e.printStackTrace();
    }
    finally
    {
       // Make sure that MapForce can shut down properly.
       if (mapforce != null)
          mapforce.dispose();

       // Since the COM server was made visible and still is visible, it
will keep running
       // and needs to be closed manually.
       System.out.println("Now close MapForce!");
    }
 }

 public static void display(ErrorMarkers err) throws AutomationException
 {
    Iterator<ErrorMarker> itr = err.iterator();

    if (err.getCount() == 0)
        System.out.print("Code generation completed successfully.\n");

    while (itr.hasNext())
    {
       String sError = "";
        Object element = itr.next();
```


© 2018 Altova GmbH

```
            if (element instanceof ErrorMarker)
               sError = ((ErrorMarker)element).getText();
            System.out.print("Error text: " + sError + "\n");
         }
      }
   }
```

17.1.4.3 JScript Examples

After you install MapForce, an example MapForce API client project for JScript is available in the directory **C:/Users/<username>/Documents/Altova/MapForce2018/MapForceExamples/API**.

The example files can be run in one of two ways:

- *From the command line:*
  Open a command prompt window and type the name of one of the example scripts (for example, `Start.js`). The Windows Scripting Host that is packaged with Windows will execute the script.

- *From Windows Explorer:*
  In Windows Explorer, browse for the JScript file and double-click it. The Windows Scripting Host that is packaged with Windows will execute the script. After the script is executed, the command console gets closed automatically.

The following example files are included:

`Start.js`	Start Mapforce registered as an automation server or connect to a running instance (see Start Application).
`DocumentAccess.js`	Shows how to open, iterate and close documents (see Simple Document Access).
`GenerateCode.js`	Shows how to invoke code generation using JScript (see Generate Code).
`Readme.txt`	Provides basic help to run the scripts.

This documentation additionally includes a few extra JScript code listings:

- Example: Code Generation
- Example: Mapping Execution
- Example: Project Support

17.1.4.3.1 Start Application

The JScript code listing below starts the application and shuts it down. If an instance of the application is already running, the running instance will be returned.

Note: For 32-bit MapForce, the registered name, or programmatic identifier (ProgId) of the COM

object is `MapForce.Application`. For 64-bit MapForce, the name is `MapForce_x64.Application`.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM
object of an already
// running application might be returned.

try {   objMapForce = WScript.GetObject("", "MapForce.Application");   }
catch(err) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("",
"MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it
to visible.
objMapForce.Visible = true;

WScript.Echo(objMapForce.Edition + " has successfully started. ");

objMapForce.Visible = false; // will shutdown application if it has no more
COM connections
//objMapForce.Visible = true;   // will keep application running with UI
visible
```

The code listed above is available as a sample file (see JScript Examples). To run the script, start it from a command prompt window or from Windows Explorer.

17.1.4.3.2 Simple Document Access

The JScript listing below shows how to open documents, set a document as the active document, iterate through the open documents, and close documents.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM
object of an already
// running application might be returned.
try {   objMapForce = WScript.GetObject("", "MapForce.Application");   }
catch(err) {}
```


© 2018 Altova GmbH

```
if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("",
"MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it
to visible.
objMapForce.Visible = true;

// ***************************** code snippet for "Simple Document Access"
***********************

// Locate examples via USERPROFILE shell variable. The path needs to be
adapted to major release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
majorVersionYear = objMapForce.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\
\Documents\\Altova\\MapForce" + majorVersionYear + "\\MapForceExamples\\";

objMapForce.Documents.OpenDocument(strExampleFolder + "CompletePO.mfd");
objMapForce.Documents.OpenDocument(strExampleFolder +
"Altova_Hierarchical_DB.mfd");

// ***************************** code snippet for "Simple Document Access"
***********************

// ***************************** code snippet for "Iteration"
***********************************

// go through all open documents using a JScript Enumerator
for (var iterDocs = new Enumerator(objMapForce.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
   objName = iterDocs.item().Name;
   WScript.Echo("Document name: " + objName);
}

// go through all open documents using index-based access to the document
collection
for (i = objMapForce.Documents.Count; i > 0; i--)
   objMapForce.Documents.Item(i).Close();


// ***************************** code snippet for "Iteration"
***********************************

//objMapForce.Visible = false;      // will shutdown application if it has no
more COM connections
```

```
objMapForce.Visible = true;   // will keep application running with UI visible
```

The code listed above is available as a sample file (see JScript Examples). To run the script, start it from a command prompt window or from Windows Explorer.

17.1.4.3.3 Generate Code

The JScript listing below shows how to open documents, set a document as the active document, iterate through the open documents, and generate C++ code.

```
// Initialize application's COM object. This will start a new instance of the
application and
// return its main COM object. Depending on COM settings, a the main COM
object of an already
// running application might be returned.
try {   objMapForce = WScript.GetObject("", "MapForce.Application");   }
catch(err) {}

if( typeof( objMapForce ) == "undefined" )
{
   try   {   objMapForce = WScript.GetObject("",
"MapForce_x64.Application")   }
   catch(err)
   {
      WScript.Echo( "Can't access or create MapForce.Application" );
      WScript.Quit();
   }
}

// if newly started, the application will start without its UI visible. Set it
to visible.
objMapForce.Visible = true;

// ************************** code snippet for "Simple Document Access"
***********************

// Locate examples via USERPROFILE shell variable. The path needs to be
adapted to major release versions.
objWshShell = WScript.CreateObject("WScript.Shell");
majorVersionYear = objMapForce.MajorVersion + 1998
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\
\Documents\\Altova\\MapForce" + majorVersionYear + "\\MapForceExamples\\";

objMapForce.Documents.OpenDocument(strExampleFolder + "CompletePO.mfd");
//objMapForce.Documents.OpenDocument(strExampleFolder +
"Altova_Hierarchical_DB.mfd");
objMapForce.Documents.NewDocument();

// ************************** code snippet for "Simple Document Access"
***********************

// ************************** code snippet for "Iteration"
```


© 2018 Altova GmbH

```
************************************

objText = "";
// go through all open documents using a JScript Enumerator and generate c++
code
for (var iterDocs = new Enumerator(objMapForce.Documents); !iterDocs.atEnd();
iterDocs.moveNext())
{
   objText += "Generated c++ code result for document " + iterDocs.item().Name
+ " :\n";
   objErrorMarkers = iterDocs.item().generateCodeEx(1); //
ENUMProgrammingLanguage.eCpp = 1

   bSuccess = true;
   for (var iterErrorMarkers = new
Enumerator(objErrorMarkers); !iterErrorMarkers.atEnd();
iterErrorMarkers.moveNext())
   {
      bSuccess = false;
      objText += "\t" + iterErrorMarkers.item().Text + "\n";
   }

   if (bSuccess)
       objText += "\tCode generation completed successfully.\n";

   objText += "\n";
}

WScript.Echo(objText);

// go through all open documents using index-based access to the document
collection
for (i = objMapForce.Documents.Count; i > 0; i--)
   objMapForce.Documents.Item(i).Close();


// ***************************** code snippet for "Iteration"
************************************

//objMapForce.Visible = false;      // will shutdown application if it has no
more COM connections
objMapForce.Visible = true;   // will keep application running with UI visible
```

The code listed above is available as a sample file (see JScript Examples). To run the script, start it from a command prompt window or from Windows Explorer.

17.1.4.3.4 Example: Code Generation

The following JScript example shows how to load an existing document and generate different kinds of mapping code for it.

```
// ------------------- begin JScript example ----------------------
// Generate Code for existing mapping.
```

```
// works with Windows scripting host.

// ----------------- helper function ------------------
function Exit(strErrorText)
{
   WScript.Echo(strErrorText);
   WScript.Quit(-1);
}

function ERROR(strText, objErr)
{
   if (objErr != null)
      Exit ("ERROR: (" + (objErr.number & 0xffff) + ")" + objErr.description
+ " - " + strText);
   else
      Exit ("ERROR: " + strText);
}
// ------------------------------------------------------

// ----------------- MAIN ------------------

// ----- create the Shell and FileSystemObject of the windows scripting
try
{
   objWshShell = WScript.CreateObject("WScript.Shell");
   objFSO = WScript.CreateObject("Scripting.FileSystemObject");
}
catch(err)
   { Exit("Can't create WScript.Shell object"); }

// ----- open MapForce or access running instance and make it visible
try
{
   objMapForce = WScript.GetObject ("", "MapForce.Application");
   objMapForce.Visible = true;      // remove this line to perform background
processing
}
catch(err) { WScript.Echo ("Can't access or create MapForce.Application"); }

// ----- open an existing mapping. adapt this to your needs!
objMapForce.OpenDocument(objFSO.GetAbsolutePathName ("Test.mfd"));

// ----- access the mapping to have access to the code generation methods
var objDoc = objMapForce.ActiveDocument;

// ----- set the code generation output properties and call the code
generation methods.
// ----- adapt the output directories to your needs
try
{
   // ----- code generation uses some of these options
   var objOptions = objMapForce.Options;

   // ----- generate XSLT -----
   objOptions.XSLTDefaultOutputDirectory = "C:\\test\\TestCOMServer\\XSLT";
```


© 2018 Altova GmbH

```
   objDoc.GenerateXSLT();

   // ----- generate Java Code -----
   objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\Java";
   objDoc.GenerateJavaCode();

   // ----- generate CPP Code, use same cpp code options as the last time
-----
   objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\CPP";
   objDoc.GenerateCppCode();

   // ----- generate C# Code, use options C# code options as the last time
-----
   objOptions.CodeDefaultOutputDirectory = "C:\\test\\TestCOMServer\\CHash";
   objDoc.GenerateCHashCode();
}
catch (err)
   { ERROR ("while generating XSL or program code", err); }


// hide MapForce to allow it to shut down
objMapForce.Visible = false;

// -------------------- end example ---------------------
```

17.1.4.3.5 Example: Mapping Execution

The following JScript example shows how to load an existing document with a simple mapping, access its components, set input- and output-instance file names and execute the mapping.

```
/*
   This sample file performs the following operations:

   Load existing MapForce mapping document.
   Find source and target component.
   Set input and output instance filenames.
   Execute the transformation.

   Works with Windows scripting host.
*/

// ---- general helpers -----------------------------

function Exit( message )
{
   WScript.Echo( message );
   WScript.Quit(-1);
}

function ERROR( message, err )
{
   if( err != null )
      Exit( "ERROR: (" + (err.number & 0xffff) + ") " + err.description + " -
```

```
" + message );
   else
      Exit( "ERROR: " + message );
}



// ---- MapForce constants -----------------------

var eComponentUsageKind_Unknown     = 0;
var eComponentUsageKind_Instance    = 1;
var eComponentUsageKind_Input       = 2;
var eComponentUsageKind_Output      = 3;



// ---- MapForce helpers -------------------------

// Searches in the specified mapping for a component by name and returns it.
// If not found, throws an error.
function FindComponent( mapping, component_name )
{
   var components = mapping.Components;
   for( var i = 0 ; i < components.Count ; ++i )
   {
      var component = components.Item( i + 1 );
      if( component.Name == component_name )
         return component;
   }
   throw new Error( "Cannot find component with name " + component_name );
}



// Browses components in a mapping and returns the first one found acting as
// source component (i.e. having connections on its right side).
function GetFirstSourceComponent( mapping )
{
   var components = mapping.Components;
   for( var i = 0 ; i < components.Count ; ++i )
   {
      var component = components.Item( i + 1 );
      if( component.UsageKind == eComponentUsageKind_Instance &&
         component.HasOutgoingConnections )
      {
         return component;
      }
   }
   throw new Error( "Cannot find a source component" );
}

// Browses components in a mapping and returns the first one found acting as
// target component (i.e. having connections on its left side).
function GetFirstTargetComponent( mapping )
```


© 2018 Altova GmbH

```
{
   var components = mapping.Components;
   for( var i = 0 ; i < components.Count ; ++i )
   {
      var component = components.Item( i + 1 );
      if( component.UsageKind == eComponentUsageKind_Instance &&
         component.HasIncomingConnections )
      {
         return component;
      }
   }
   throw new Error( "Cannot find a target component" );
}



function IndentTextLines( s )
{
   return "\t" + s.replace( /\n/g, "\n\t" );
}

function GetAppoutputLineFullText( oAppoutputLine )
{
   var s = oAppoutputLine.GetLineText();
   var oAppoutputChildLines = oAppoutputLine.ChildLines;
   var i;

   for( i = 0 ; i < oAppoutputChildLines.Count ; ++i )
   {
      oAppoutputChildLine = oAppoutputChildLines.Item( i + 1 );
      sChilds = GetAppoutputLineFullText( oAppoutputChildLine );
      s += "\n" + IndentTextLines( sChilds );
   }

   return s;
}

// Create a nicely formatted string from AppOutputLines
function GetResultMessagesString( oAppoutputLines )
{
   var s1 = "Transformation result messages:\n";
   var oAppoutputLine;
   var i;

   for( i = 0 ; i < oAppoutputLines.Count ; ++i )
   {
      oAppoutputLine = oAppoutputLines.Item( i + 1 );
      s1 += GetAppoutputLineFullText( oAppoutputLine );
      s1 += "\n";
   }

   return s1;
}
```

```
// ---- MAIN -----------------------------------

var wshShell;
var fso;
var mapforce;

// create the Shell and FileSystemObject of the windows scripting system
try
{
   wshShell = WScript.CreateObject( "WScript.Shell" );
   fso = WScript.CreateObject( "Scripting.FileSystemObject" );
}
catch( err )
   { ERROR( "Can't create windows scripting objects", err ); }

// open MapForce or access currently running instance
try
{
   mapforce = WScript.GetObject( "", "MapForce.Application" );
}
catch( err )
   { ERROR( "Can't access or create MapForce.Application", err ); }

try
{
   // Make MapForce UI visible. This is an API requirement for output
generation.
  mapforce.Visible = true;

   // open an existing mapping.
   // **** adjust the examples path to your needs ! **************
   var sMapForceExamplesPath = fso.BuildPath(
            wshShell.SpecialFolders( "MyDocuments" ),
            "Altova\\MapForce2018\\MapForceExamples" );
   var sDocFilename = fso.BuildPath( sMapForceExamplesPath,
"PersonList.mfd" );
   var doc = mapforce.OpenDocument( sDocFilename );

   // Find existing components by name in the main mapping.
   // Note, the names of components may not be unique as a schema component's
name
   // is derived from its schema file name.
   var source_component = FindComponent( doc.MainMapping, "Employees" );
   var target_component = FindComponent( doc.MainMapping, "PersonList" );
   // If you do not know the names of the components for some reason, you
could
   // use the following functions instead of FindComponent.
   //var source_component = GetFirstSourceComponent( doc.MainMapping );
   //var target_component = GetFirstTargetComponent( doc.MainMapping );

   // specify the desired input and output files.
   source_component.InputInstanceFile = fso.BuildPath( sMapForceExamplesPath,
"Employees.xml" );
```


© 2018 Altova GmbH

```
   target_component.OutputInstanceFile =
fso.BuildPath( sMapForceExamplesPath, "test_transformation_results.xml" );

   // Perform the transformation.
   // You can use doc.GenerateOutput() if you do not need result messages.
   // If you have a mapping with more than one target component and you want
   // to execute the transformation only for one specific target component,
   // call target_component.GenerateOutput() instead.
   var result_messages = doc.GenerateOutputEx();

   var summary_info =
         "Transformation performed from " + source_component.InputInstanceFile
+ "\n" +
         "to " + target_component.OutputInstanceFile + "\n\n" +
         GetResultMessagesString( result_messages );
   WScript.Echo( summary_info );
}
catch( err )
{
   ERROR( "Failure", err );
}
```

17.1.4.3.6 Example: Project Support

The following JScript example shows how to use the MapForce API to automate tasks pertaining to MapForce projects. Before running the example, make sure to edit the variable `strSamplePath` so that it points to the **MapForceExamples** folder of your MapForce installation.

To successfully run all operations in this example below, you will need the Enterprise version of MapForce. If you have the Professional version running, comment out the lines that insert the WebService project.

```
// ////////////// global variables /////////////////
var objMapForce = null;
var objWshShell = null;
var objFSO = null;

// !!! adapt the following path to your needs. !!!
var strSamplePath = "C:\\Users\\<username>\\Documents\\Altova\\MapForce2018\
\MapForceExamples\\Tutorial\\";

// ////////////////////// Helpers ///////////////////////////////

function Exit(strErrorText)
{
   WScript.Echo(strErrorText);
   WScript.Quit(-1);
}

function ERROR(strText, objErr)
{
   if (objErr != null)
```

```
      Exit ("ERROR: (" + (objErr.number & 0xffff) + ")" + objErr.description
+ " - " + strText);
   else
      Exit ("ERROR: " + strText);
}

function CreateGlobalObjects ()
{
   // the Shell and FileSystemObject of the windows scripting host often
always useful
   try
   {
      objWshShell = WScript.CreateObject("WScript.Shell");
      objFSO = WScript.CreateObject("Scripting.FileSystemObject");
   }
   catch(err)
      { Exit("Can't create WScript.Shell object"); }

   // create the MapForce connection
   // if there is a running instance of MapForce (that never had a connection)
- use it
   // otherwise, we automatically create a new instance
   try
   {
      objMapForce = WScript.GetObject("", "MapForce.Application");
   }
   catch(err)
   {
      { Exit("Can't access or create MapForce.Application"); }
   }
}

// ------------------------------------------------------------
// print project tree items and their properties recursively.
// ------------------------------------------------------------
function PrintProjectTree( objProjectItemIter, strTab )
{
   while ( ! objProjectItemIter.atEnd() )
   {
      // get current project item
      objItem = objProjectItemIter.item();

      try
      {
         // ----- print common properties
         strGlobalText += strTab + "[" + objItem.Kind + "]" + objItem.Name +
"\n";

         // ----- print code generation properties, if available
         try
         {
            if ( objItem.CodeGenSettings_UseDefault )
               strGlobalText += strTab + "  Use default code generation
settings\n";
            else
```


© 2018 Altova GmbH

```
               strGlobalText += strTab + "  code generation language is " +
                                                    objItem.CodeGenSettings_Lan
guage +
                                " output folder is " +
objItem.CodeGenSettings_OutputFolder + "\n";
         }
         catch( err ) {}

         // ----- print WSDL settings, if available
         try
         {
            strGlobalText += strTab + "  WSDL File is " + objItem.WSDLFile +
                             " Qualified Name is " + objItem.QualifiedName +
"\n";
         }
         catch( err ) {}
      }
      catch( ex )
         { strGlobalText += strTab + "[" + objItem.Kind + "]\n" }

      // ---- recurse
      PrintProjectTree( new Enumerator( objItem ), strTab + '  ' );

      objProjectItemIter.moveNext();
   }
}

// ------------------------------------------------------------
// Load example project installed with MapForce.
// ------------------------------------------------------------
function LoadSampleProject()
{
   // close open project
   objProject = objMapForce.ActiveProject;
   if ( objProject != null )
      objProject.Close();

   // open sample project and iterate through it.
   // sump properties of all project items

   objProject = objMapForce.OpenProject(strSamplePath +
"MapForceExamples.mfp");
   strGlobalText = '';
   PrintProjectTree( new Enumerator (objProject), ' ' )
   WScript.Echo( strGlobalText );

   objProject.Close();
}

// ------------------------------------------------------------
// Create a new project with some folders, mappings and a
// Web service project.
// ------------------------------------------------------------
function CreateNewProject()
{
```

```
   try
   {
      // create new project and specify file to store it.
      objProject = objMapForce.NewProject(strSamplePath + "Sample.mfp");

      // create a simple folder structure
      objProject.CreateFolder( "New Folder 1");
      objFolder1 = objProject.Item(0);
      objFolder1.CreateFolder( "New Folder 2");
      objFolder2 = ( new Enumerator( objFolder1 ) ).item();   // an
alternative to Item(0)

      // add two different mappings to folder structure
      objFolder1.AddFile( strSamplePath + "DB_Altova_SQLXML.mfd");
      objMapForce.Documents.OpenDocument(strSamplePath +
"InspectionReport.mfd");
      objFolder2.AddActiveFile();

      // override code generation settings for this folder
      objFolder2.CodeGenSettings_UseDefault = false;
      objFolder2.CodeGenSettings_OutputFolder = strSamplePath + "SampleOutput"
      objFolder2.CodeGenSettings_Language = 1;      //C++

      // insert Web service project based on a wsdl file from the installed
examples
      objProject.InsertWebService( strSamplePath + "TimeService/
TimeService.wsdl",
                                   "{http://www.Nanonull.com/TimeService/}
TimeService",
                           "TimeServiceSoap",
                           true );
      objProject.Save();
      if ( ! objProject.Saved )
         WScript.Echo("problem occurred when saving project");

      // dump project tree
      strGlobalText = '';
      PrintProjectTree( new Enumerator (objProject), ' ' )
      WScript.Echo( strGlobalText );
   }
   catch (err)
   { ERROR("while creating new project", err ); }
}

// ------------------------------------------------------------
// Generate code for a project's sub-tree. Mix default code
// generation parameters and overloaded parameters.
// ------------------------------------------------------------
function GenerateCodeForNewProject()
{
   // since the Web service project contains only initial mappings,
   // we generate code only for our custom folder.
   // code generation parameters from project are used for Folder1,
   // whereas Folder2 provides overwritten values.
   objFolder = objProject.Item(0);
```


© 2018 Altova GmbH

```
   objFolder1.GenerateCode();
}

// ////////////////////// MAIN //////////////////////////////

CreateGlobalObjects();
objMapForce.Visible = true;

LoadSampleProject();
CreateNewProject();
GenerateCodeForNewProject();

// uncomment to shut down application when script ends
// objMapForce.Visible = false;
```

17.2 Object Reference

This section provides reference to the objects of the MapForce COM API. The objects are described in a generic manner, since the API may be used with virtually any language that supports calling a COM object. For language-specific examples, see Examples.

In Java, some syntax variations to the object names exist, as follows:

- ***Classes and class names***
  For every interface of the MapForce automation interface, a Java class exists with the name of the interface.

- ***Method names***
  Method names on the Java interface are the same as used on the COM interfaces, but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with `get` and `set` can be used. If a property does not support write-access, no setter method is available. Example: For the `Name` property of the `Document` interface, the Java methods `getName` and `setName` are available.

- ***Enumerations***
  For every enumeration defined in the automation interface, a Java enumeration is defined with the same name and values.

- ***Events and event handlers***
  For every interface in the automation interface that supports events, a Java interface with the same name plus '`Event`' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus `'DefaultHandler'`. For example:
  `Application`: Java class to access the application
  `ApplicationEvents`: Events interface for the Application
  `ApplicationEventsDefaultHandler`: Default handler for `ApplicationEvents`

17.2.1 Application

The `Application` interface is the interface to a MapForce application object. It represents the main access point for the MapForce application itself. This interface is the starting point to do any further operations with MapForce or to retrieve or create other MapForce related automation objects. For information about creating an instance the `Application` object, see Accessing the API.

Events
Events

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`
`Options`
`Project`

© 2018 Altova GmbH

Documents

Application status:
Visible
Name
Quit
Status
WindowHandle

MapForce designs:
NewDocument
OpenDocument
OpenURL
ActiveDocument

MapForce projects:
NewProject
OpenProject
ActiveProject

MapForce code generation:
HighlightSerializedMarker

Global resources:
GlobalResourceConfig
GlobalResourceFile

Version information:
Edition
IsAPISupported
MajorVersion
MinorVersion

17.2.1.1 Events

This object supports the following events:

OnDocumentOpened
OnProjectOpened
OnShutdown

17.2.1.1.1 OnDocumentOpened

Event: OnDocumentOpened (*i_objDocument* as Document)

Description
This event is triggered when an existing or new document is opened. The corresponding close event is Document.OnDocumentClosed.

17.2.1.1.2 OnProjectOpened

Event: `OnProjectOpened` (*`i_objProject`* as `Project`)

Description
This event is triggered when an existing or new project is loaded into the application. The corresponding close event is `Project.OnProjectClosed`.

17.2.1.1.3 OnShutdown

Event: `OnShutdown` ()

Description
This event is triggered when the application is shutting down.

17.2.1.2 ActiveDocument

Property: `ActiveDocument` as `Document` (read-only)

Description
Returns the automation object of the currently active document. This property returns the same as `Documents.ActiveDocument`.

Errors

1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.2.1.3 ActiveProject

Property: `ActiveProject` as `Project` (read-only)

Description
Returns the automation object of the currently active project.

Errors

1000	The application object is no longer valid.
1001	Invalid address for the return parameter was specified.

17.2.1.4 Application

Property: `Application` as `Application` (read-only)

Description

© 2018 Altova GmbH

Retrieves the application's top-level object.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.5 Documents

Property: `Documents` as `Documents` (read-only)

Description
Returns a collection of all currently open documents.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.6 Edition

Property: `Edition` as `String` (read-only)

Description
Returns the edition of the application, e.g. "Altova MapForce Enterprise Edition" for the enterprise edition.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.7 GlobalResourceConfig

Property: `GlobalResourceConfig` as `String`

Description
Gets or sets the name of the active global resource configuration file. Per default, the file is called GlobalResources.xml.

The configuration file can be renamed and saved to any location. You can therefore have multiple Global Resources XML files. However, only one of these Global Resources XML File can be active, per application, at one time, and only the definitions contained in this file will be available to the application.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.8 GlobalResourceFile

Property: `GlobalResourceFile` as `String`

Description
Gets or sets the global resource definition file. Per default the file is called GlobalResources.xml.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.9 HighlightSerializedMarker

Method: `HighlightSerializedMarker` (*`i_strSerializedMarker`* as `String`)

Description
Use this method to highlight a location in a mapping file that has been previously serialized. If the corresponding document has not already been loaded, it will be loaded first. See Document.GenerateCodeEx for a method to retrieve a serialized marker.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.
1007 The string passed in *i_strSerializedMarker* is not recognized as a serialized MapForce marker.
1008 The marker points to a location that is no longer valid.

17.2.1.10 IsAPISupported

Property: `IsAPISupported` as `Boolean` (read-only)

Description
Returns true if the API is supported in this version of MapForce.

Errors
1001 Invalid address for the return parameter was specified.

17.2.1.11 MajorVersion

Property: `MajorVersion` as `Long` (read-only)

Description
Gets the major version number of of MapForce. The version is calculated starting from 1998, and is incremented by one every year. For example, the major version is "18" for the release 2016.

© 2018 Altova GmbH

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.12 MinorVersion

Property: `MinorVersion` as `Long` (read-only)

Description
The minor version number of the product, e.g. 2 for 2006 R2 SP1.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.13 Name

Property: `Name` as `String` (read-only)

Description
The name of the application.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.14 NewDocument

Method: `NewDocument` () as `Document`

Description
Creates a new empty document. The newly opened document becomes the `ActiveDocument`. This method is a shortened form of `Documents.NewDocument`.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.15 NewProject

Method: `NewProject` () as `Project`

Description

Creates a new empty project. The current project is closed. The new project is accessible under `ActiveProject`.

Errors

1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.16 OpenDocument

Method: `OpenDocument` (*`i_strFileName`* as `String`) as `Document`

Description

Loads a previously saved document file and continues working on it. The newly opened document becomes the `ActiveDocument`. This method is a shorter form of `Documents.OpenDocument`.

Errors

1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.17 OpenProject

Method: `NewProject` () as `Project`

Description

Opens an existing Mapforce project (`*.mfp`). The current project is closed. The newly opened project is accessible under `ActiveProject`.

Errors

1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.18 OpenURL

Method: `OpenURL` (*`i_strURL`* as `String`, *`i_strUser`* as `String`, *`i_strPassword`* as `String`)

Description

Loads a previously saved document file from an URL location. Allows user name and password to be supplied.

Errors

1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

© 2018 Altova GmbH

17.2.1.19 Options

Property: `Options` as `Options` (read-only)

Description
This property gives access to options that configure the generation of code.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.20 Parent

Property: `Parent` as `Application` (read-only)

Description
The parent object according to the object model.

Errors
1000 The object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.21 Quit

Method: `Quit` ()

Description
Disconnects from MapForce to allow the application to shutdown. Calling this method is optional since MapForce keeps track of all external COM connections and automatically recognizes a disconnection. For more information on automatic shutdown see the `Visible` property.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.22 ServicePackVersion

Property: `ServicePackVersion` as `Long` (read-only)

Description
The service pack version number of the product, e.g. 1 for 2010 R2 SP1.

Errors

1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.23 Status

Property: `Status` as `Long` (read-only)

Description
The status of the application. It is one of the values of the `ENUMApplicationStatus` enumeration.

Errors
1001 Invalid address for the return parameter was specified.

17.2.1.24 Visible

Property: `Visible` as `Boolean`

Description
`True` if MapForce is displayed on the screen (though it might be covered by other applications or be iconized).

`False` if MapForce is hidden. The default value for MapForce when automatically started due to a request from the automation server `MapForce.Application` is `false`. In all other cases, the property is initialized to `true`.

An application instance that is visible is said to be controlled by the user (and possibly by clients connected via the automation interface). It will only shut down due to an explicit user request. To shut down an application instance, set its visibility to false and clear all references to this instance within your program. The application instance will shut down automatically when no further COM clients are holding references to it.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

17.2.1.25 WindowHandle

Property: `WindowHandle` () as `long` (read-only)

Description
Retrieve the application's Window Handle.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

© 2018 Altova GmbH

17.2.2 AppOutputLine

Represents a message line. In contrast to ErrorMarker, its structure is more detailed and can contain a collection of child lines, therefore forming a tree of message lines.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Line access:
`GetLineSeverity`
`GetLineSymbol`
`GetLineText`
`GetLineTextEx`
`GetLineTextWithChildren`
`GetLineTextWithChildrenEx`

A single AppOutputLine consists of one or more sub-lines.
Sub-line access:
`GetLineCount`

A sub-line consists of one or more cells.
Cell access:
`GetCellCountInLine`
`GetCellIcon`
`GetCellSymbol`
`GetCellText`
`GetCellTextDecoration`
`GetIsCellText`

Below an AppOutputLine there can be zero, one, or more child lines which themselves are of type AppOutputLine, which thus form a tree structure.

Child lines access:
`ChildLines`

17.2.2.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.2 ChildLines

Property: `ChildLines` as `AppOutputLines` (read-only)

Description
Returns a collection of the current line's direct child lines.

Errors
4100 The application object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.3 GetCellCountInLine

Method: `GetCellCountInLine` (*`nLine`* as `Long`) as `Long`

Description
Gets the number of cells in the sub-line indicated by *nLine* in the current AppOutputLine.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.4 GetCellIcon

Method: `GetCellIcon` (*`nLine`* as `Long`, *`nCell`* as `Long`) as `Long`

Description
Gets the icon of the cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine*.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.5 GetCellSymbol

Method: `GetCellSymbol` (*`nLine`* as `Long`, *`nCell`* as `Long`) as `AppOutputLineSymbol`

Description
Gets the symbol of the cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine*.

Errors
4100 The object is no longer valid.

© 2018 Altova GmbH

4101 Invalid address for the return parameter was specified.

17.2.2.6 GetCellText

Method: `GetCellText` (*`nLine`* as `Long`, *`nCell`* as `Long`) as `String`

Description
Gets the text of the cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine*.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.7 GetCellTextDecoration

Method: `GetCellTextDecoration` (*`nLine`* as `Long`, *`nCell`* as `Long`) as `Long`

Description
Gets the decoration of the text cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine*.
It can be one of the `ENUMAppOutputLine_TextDecoration` values.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.8 GetIsCellText

Method: `GetIsCellText` (*`nLine`* as `Long`, *`nCell`* as `Long`) as `Boolean`

Description
Returns true, if the cell indicated by *nCell* in the current AppOutputLine's sub-line indicated by *nLine* is a text cell.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.9 GetLineCount

Method: `GetLineCount` () as `Long`

Description

Gets the number of sub-lines the current line consists of.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.10 GetLineSeverity

Method: `GetLineSeverity` () as `Long`

Description
Gets the severity of the line. It can be one of the `ENUMAppOutputLine_Severity` values:

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.11 GetLineSymbol

Method: `GetLineSymbol` () as `AppOutputLineSymbol`

Description
Gets the symbol assigned to the whole line.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.12 GetLineText

Method: `GetLineText` () as `String`

Description
Gets the contents of the line as text.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.13 GetLineTextEx

Method: `GetLineTextEx` (*`psTextPartSeperator`* as `String`, *`psLineSeperator`* as `String`) as `String`

© 2018 Altova GmbH

Description
Gets the contents of the line as text using the specified part and line separators.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.14 GetLineTextWithChildren

Method: `GetLineTextWithChildren` () as `String`

Description
Gets the contents of the line including all child and descendant lines as text.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.15 GetLineTextWithChildrenEx

Method: `GetLineTextWithChildrenEx` (*`psPartSep`* as `String`, *`psLineSep`* as `String`, *`psTabSep`* as `String`, *`psItemSep`* as `String`) as `String`

Description
Gets the contents of the line including all child and descendant lines as text using the specified part, line, tab and item separators.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.2.16 Parent

Property: `Parent` as `AppOutputLines` (read-only)

Description
The parent object according to the object model.

Errors
4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

17.2.3 AppOutputLines

Represents a collection of AppOutputLine message lines.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Iterating through the collection:
`Count`
`Item`

17.2.3.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
4000 The object is no longer valid.
4001 Invalid address for the return parameter was specified.

17.2.3.2 Count

Property: `Count` as `Integer` (read-only)

Description
Retrieves the number of lines in the collection.

Errors
4000 The object is no longer valid.
4001 Invalid address for the return parameter was specified.

17.2.3.3 Item

Property: `Item` (*`nIndex`* as `Integer`) as `AppOutputLine` (read-only)

Description
Retrieves the line at `nIndex` from the collection. Indices start with 1.

Errors
4000 The object is no longer valid.

© 2018 Altova GmbH

4001 Invalid address for the return parameter was specified.

17.2.3.4 Parent

Property: `Parent` as `AppOutputLine` (read-only)

Description
The parent object according to the object model.

Errors
4000 The object is no longer valid.
4001 Invalid address for the return parameter was specified.

17.2.4 AppOutputLineSymbol

An AppOutputLineSymbol represents a link in an AppOutputLine message line which can be clicked in the MapForce Messages window.
It is applied to a cell of an AppOutputLine or to the whole line itself.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Access to AppOutputLineSymbol methods:
`GetSymbolHREF`
`GetSymbolID`
`IsSymbolHREF`

17.2.4.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.4.2 GetSymbolHREF

Method: `GetSymbolHREF` () as `String`

Description
If the symbol is of type URL, returns the URL as a string.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.4.3 GetSymbolID

Method: `GetSymbolHREF` () as `Long`

Description
Gets the ID of the symbol.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.4.4 IsSymbolHREF

Method: `IsSymbolHREF` () as `Boolean`

Description
Indicates if the symbol is of kind URL.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.4.5 Parent

Property: `Parent` as `Application` (read-only)

Description
The parent object according to the object model.

Errors
4200 The object is no longer valid.
4201 Invalid address for the return parameter was specified.

17.2.5 Component

A Component represents a MapForce component.

© 2018 Altova GmbH

Properties and Methods
Properties to navigate the object model:
Application
Parent

Component properties:
HasIncomingConnections
HasOutgoingConnections
CanChangeInputInstanceFile
CanChangeOutputInstanceFile
ComponentName

ID
IsParameterInputRequired
IsParameterSequence
Name
Preview
Schema
SubType
Type

Instance related properties:
InputInstanceFile
OutputInstanceFile

Datapoints:
GetRootDatapoint

Execution:
GenerateOutput

17.2.5.1 Application

Property: Application as Application (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.2 CanChangeInputInstanceFile

Property: CanChangeInputInstanceFile as Boolean (read-only)

Description
Indicates if the input instance file name can be changed.

Returns false if the component has a filename node and this node has a connection on its left (input) side, otherwise returns true.
If the component does not have a filename node, false is returned.

Errors

1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.3 CanChangeOutputInstanceFile

Property: CanChangeOutputInstanceFile as Boolean (read-only)

Description
Indicates if the output instance file name can be changed.

Returns false if the component has a filename node and this node has a connection on its left (input) side, otherwise returns true.
If the component does not have a filename node, false is returned.

Errors

1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.4 ComponentName

Property: ComponentName as `String`

Description
Gets or sets the component's name.

Errors

1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1246 The component does not support setting its name.
1247 Invalid component name.

17.2.5.5 GenerateOutput

Method: `GenerateOutput` (*[out]* *`pbError`* as `Boolean`) as `AppOutputLines`

Description
Generates the output file(s) defined in the mapping for the current component only, using a MapForce internal mapping language. The name(s) of the output file(s) are defined as property of the current component which is the output item in the mapping for this generation process.

Remarks
pbError is an output-only parameter. You will receive a value only if the calling language supports

© 2018 Altova GmbH

output parameters. If not, the value you pass here will remain unchanged when the function has finished.

Errors

1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1248	Generating output is only supported when the graphical user interface is visible.

17.2.5.6 GetRootDatapoint

Method: GetRootDatapoint(side as ENUMComponentDatapointSide, strNamespace as String, strLocalName as String, strParameterName as String) as Datapoint

Description

Gets a root datapoint on the left (input) or right (output) side of a component. To access children and descendants, the Datapoint object provides further methods.

The *side* parameter indicates if an input, or output, datapoint of a component is to be retrieved.

The specified namespace and local name, indicate the specific name of the node whose datapoint is to be retrieved. For components with structural information such as schema components, you will have to provide the namespace together with the local name, or you can just pass an empty string for the namespace.

File-based components like the schema component contain a special node on their root, the filename node. There, GetRootDatapoint can only find the filename node. You will have to pass namespace "`http://www.altova.com/mapforce`" and local name "`FileInstance`" to retrieve a datapoint of this node.

The specified parameter name should be an empty string unless the component in question is a function call component . Since a user-defined function might contain input or output parameters of the same structure, the function call component calling this user-defined function can have more than one root node with an identical namespace and local name.

They will then differ only by their parameter names, which are in fact the names of the according parameter components in the user-defined function mapping itself.

It is not mandatory to specify the parameter name, though. In that case, the method will return the first root datapoint matching the specified namespace and local name.

Errors

1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.
1231	Datapoint not found.

17.2.5.7 HasIncomingConnections

Property: `HasIncomingConnections` as `Boolean` (read-only)

Description
Indicates if the component has any incoming connections (on its left side) not including the filename node. An incoming connection on the filename node does not have any effect on the returned value.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.8 HasOutgoingConnections

Property: `HasOutgoingConnections` as `Boolean` (read-only)

Description
Indicates if the component has any outgoing connections (on its right side).

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.9 ID

Property: `ID` as `Unsigned Long` (read-only)

Description
Retrieves the component ID.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.10 InputInstanceFile

Property: `InputInstanceFile` as `String`

Description
Gets or sets the component's input instance file.

Errors
1200 The object is no longer valid.

© 2018 Altova GmbH

1201 Invalid address for the return parameter was specified.

17.2.5.11 IsParameterInputRequired

Property: IsParameterInputRequired as `Boolean`

Description
Gets or sets, if the input parameter component requires an ingoing connection on the function call component of the user-defined function this input parameter component is in.
This property works only for components, which are input parameter components.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1232 This operation works only for an input parameter component.
1240 Changing the document not allowed. It is read-only.

17.2.5.12 IsParameterSequence

Property: IsParameterSequence as `Boolean`

Description
Gets or sets, if the input or output parameter component supports sequences.
This property works only for components, which are input or output parameter components.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1233 This operation works only for an input or output parameter component.
1240 Changing the document not allowed. It is read-only.

17.2.5.13 Name

Property: `Name` as `String (read only)`

Description
Gets the component's name.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.14 OutputInstanceFile

Property: `OutputInstanceFile` as `String`

Description
Gets or sets the component's output instance file.

Trying to access the OutputInstanceFile of a component via the API does not return any data if the "File" connector of the component has been connected to another item in the mapping.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.15 Parent

Property: `Parent` as `Mapping` (read-only)

Description
The parent object according to the object model.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.16 Preview

Property: Preview as `Boolean`

Description
Gets or sets, if the component is the current preview component.

This property works only for components, which are target components in the document's main mapping. Only one target component in the main mapping can be the preview component at any time.

When setting this property, it is only possible to set it to true. This then will also implicitly set the Preview property of all other components to false.

If there is just a single target component in the main mapping, it is also the preview component.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.
1234 Only a target component in the main mapping can be set as preview component.
1235 A component cannot be set as non-preview component. Set another

© 2018 Altova GmbH

component as preview component instead.

17.2.5.17 Schema

Property: `Schema` as `String` (read-only)

Description
Retrieves the component's schema file name.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.18 SubType

Property: `SubType` as `ENUMComponentSubType` (read-only)

Description
Retrieves the component's sub type.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.19 Type

Property: `Type` as `ENUMComponentType` (read-only)

Description
Retrieves the component's type.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.5.20 UsageKind

Property: `UsageKind` as `ENUMUsageKind` (read-only)

Description
Retrieves the component's usage kind.

Errors
1200 The object is no longer valid.

1201 Invalid address for the return parameter was specified.

17.2.6 Components

Represents a collection of Component objects.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Iterating through the collection:
`Count`
`Item`

17.2.6.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.6.2 Count

Property: `Count` as `Integer` (read-only)

Description
Retrieves the number of components in the collection.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.6.3 Item

Property: `Item` (*`nIndex`* as `Integer`) as `Component` (read-only)

Description
Retrieves the component at `nIndex` from the collection. Indices start with 1.

© 2018 Altova GmbH

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.6.4 Parent

Property: `Parent` as `Mapping` (read-only)

Description
The parent object according to the object model.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.7 Connection

A Connection object represents a connector between two components.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Properties
`ConnectionType`

17.2.7.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
2100 The object is no longer valid.
2101 Invalid address for the return parameter was specified.

17.2.7.2 ConnectionType

Property: ConnectionType as ENUMConnectionType

Description
Gets or sets the connection's type.

Errors

2100 The application object is no longer valid.
2101 Invalid address for the return parameter was specified.
2102 Changing the document not allowed. It is read-only.
2103 Failed changing connection type.

17.2.7.3 Parent

Property: `Parent` as `Mapping` (read-only)

Description
The parent object according to the object model.

Errors

2100 The object is no longer valid.
2101 Invalid address for the return parameter was specified.

17.2.8 Datapoint

A Datapoint object represents an input or output icon of a component.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Methods
`GetChild`

17.2.8.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors

2000 The object is no longer valid.
2001 Invalid address for the return parameter was specified.

© 2018 Altova GmbH

17.2.8.2 GetChild

Method: GetChild(strNamespace as String, strLocalName as String, searchFlags as ENUMSearchDatapointFlags) as Datapoint

Description
Scans for a direct child datapoint of the current datapoint, by namespace and local name.

Search flags can be passed as combination of values (combined using binary OR) of the ENUMSearchDatapointFlags enumeration.

A schema component with elements that contain mixed content, each display an additional child node, the so-called text() node. To retrieve a datapoint of a text() node, you will have to pass an empty string in strNamespace as well as `"#text"` in strLocalName and eSearchDatapointElement in searchFlags.

Errors

2000	The application object is no longer valid.
2001	Invalid address for the return parameter was specified.
2002	Datapoint not found.

17.2.8.3 Parent

Property: `Parent` as `Component` (read-only)

Description
The parent object according to the object model.

Errors

2000	The object is no longer valid.
2001	Invalid address for the return parameter was specified.

17.2.9 Document

A Document object represents a MapForce document (a loaded MFD file).
A document contains a main mapping and zero or more local user-defined-function mappings.

Events
Events

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

File handling:
Activate

Close
FullName
Name
Path
Saved
Save
SaveAs

Mapping handling:
MainMapping
Mappings
CreateUserDefinedFunction

Component handling:
FindComponentByID

Code generation:
OutputSettings_ApplicationName
JavaSettings_BasePackageName

GenerateCHashCode
GenerateCodeEx
GenerateCppCode
GenerateJavaCode
GenerateXQuery
GenerateXSLT
GenerateXSLT2
HighlightSerializedMarker

Mapping execution:
GenerateOutput
GenerateOutputEx

View access:
MapForceView

Obsolete:
OutputSettings_Encoding

17.2.9.1 Events

This object supports the following events:

OnDocumentClosed
OnModifiedFlagChanged

17.2.9.1.1 OnDocumentClosed

Event: OnDocumentClosed (*i_objDocument* as Document)

© 2018 Altova GmbH

Description
This event is triggered when a document is closed. The document object passed into the event handler should not be accessed. The corresponding open event is `Application.OnDocumentOpened`.

17.2.9.1.2 OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (*`i_bIsModified`* as Boolean)

Description
This event is triggered when a document's modification status changes.

17.2.9.2 Activate

Method: `Activate` ()

Description
Makes this document the active document.

Errors
1200 The application object is no longer valid.

17.2.9.3 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.4 Close

Method: `Close` ()

Description
Closes the document without saving.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.5 CreateUserDefinedFunction

Method: CreateUserDefinedFunction(strFunctionName as String, strLibraryName as String, strSyntax as String, strDetails as String, bInlinedUse as Boolean) as Mapping

Description
Creates a user defined function in the current document.

Errors

1200	The application object is no longer valid.
1201	Invalid address for the return parameter was specified.
1208	Failed creating user-defined function.
1209	Changing the document not allowed. It is read-only.

17.2.9.6 FindComponentByID

Method: `FindComponentByID` (*nID* as `Unsigned Long`) as `Component`

Description
Searches in the whole document, so all its mappings, for the component with the specified id.

Errors

1200	The application object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.2.9.7 FullName

Property: `FullName` as `String`

Description
Path and name of the document file.

Errors

1200	The application object is no longer valid.
1201	Invalid address for the return parameter was specified.

17.2.9.8 GenerateCHashCode

Method: `GenerateCHashCode` ()

Description
Generate C# code that will perform the mapping. Uses the properties defined in `Application.Options` to configure code generation.

© 2018 Altova GmbH

Errors

1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1205 Error during code generation.

17.2.9.9 GenerateCodeEx

Method: `GenerateCodeEx` (*`i_nLanguage`* as `ENUMProgrammingLanguage`) as `ErrorMarkers`

Description

Generates code that will perform the mapping. The parameter *i_nLanguage* specifies the target language. The method returns an object that can be used to enumerate all messages created by the code generator. These are the same messages that get displayed in the Messages window of MapForce.

Errors

1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1205 Error during code generation.

17.2.9.10 GenerateCppCode

Method: `GenerateCppCode` ()

Description

Generates C++ code that will perform the mapping. Uses the properties defined in `Application.Options` to configure code generation.

Errors

1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1205 Error during code generation.

17.2.9.11 GenerateJavaCode

Method: `GenerateJavaCode` ()

Description

Generates Java code that will perform the mapping. Uses the properties defined in `Application.Options` to configure code generation.

Errors

1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1205 Error during code generation.

17.2.9.12 GenerateOutput

Method: `GenerateOutput` ()

Description
Generates all output files defined in the mapping using a MapForce internal mapping language. The names of the output files are defined as properties of the output items in the mapping.

Errors

1200	The application object is no longer valid.
1201	Invalid address for the return parameter was specified.
1206	Error during execution of mapping algorithm.
1210	Generating output is only supported when the graphical user interface is visible.

This method can only be used when the MapForce (running as a COM server) main window is visible, or is embedded with a graphical user interface. If the method is called while MapForce is not visible, then an error will occur.

17.2.9.13 GenerateOutputEx

Method: `GenerateOutputEx` () as `AppOutputLines`

Description
Generates all output files defined in the mapping using a MapForce internal mapping language. The names of the output files are defined as properties of the output items in the mapping. This method is identical to `GenerateOutput` except for its return value containing the resulting messages, warnings and errors arranged as trees of `AppOutputLine`s.

Errors

1200	The application object is no longer valid.
1201	Invalid address for the return parameter was specified.
1206	Error during execution of mapping algorithm.
1210	Generating output is only supported when the graphical user interface is visible.

This method can only be used when the MapForce (running as a COM server) main window is visible, or is embedded with a graphical user interface. If the method is called while MapForce is not visible, then an error will occur.

17.2.9.14 GenerateXQuery

Method: `GenerateXQuery` ()

Description
Generates mapping code as XQuery. Uses the properties defined in `Application.Options` to configure code generation.

Errors

© 2018 Altova GmbH

1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1204 Error during XSLT/XSLT2/XQuery code generation.

17.2.9.15 GenerateXSLT

Method: `GenerateXSLT` ()

Description
Generates mapping code as XSLT. Uses the properties defined in `Application.Options` to configure code generation.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1204 Error during XSLT/XSLT2/XQuery code generation.

17.2.9.16 GenerateXSLT2

Method: `GenerateXSLT2` ()

Description
Generates mapping code as XSLT2. Uses the properties defined in `Application.Options` to configure code generation.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1204 Error during XSLT/XSLT2/XQuery code generation.

17.2.9.17 HighlightSerializedMarker

Method: `HighlightSerializedMarker` (*`i_strSerializedMarker`* as `String`)

Description
Use this method to highlight a location in a mapping file that has been previously serialized. If the corresponding document is not already loaded, it will be loaded first. See `GenerateCodeEx` for a method to retrieve a serialized marker.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.
1007 The string passed in *i_strSerializedMarker* is not recognized a serialized MapForce marker.
1008 The marker points to a location that is no longer valid.

17.2.9.18 JavaSettings_BasePackageName

Property: `JavaSettings_BasePackageName` as `String`

Description
Sets or retrieves the base package name used when generating Java code. This property is available in UI-dialog for the Document Settings.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.19 MainMapping

Property: `MainMapping` as `Mapping` (read-only)

Description
Retrieves the main mapping of the document.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.20 MapForceView

Property: `MapForceView` as `MapForceView` (read-only)

Description
This property gives access to functionality specific to the MapForce view.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.21 Mappings

Property: `Mappings` as `Mappings` (read-only)

Description
Returns a collection of the mappings contained in the document.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

© 2018 Altova GmbH

17.2.9.22 Name

Property: `Name` as `String`

Description
Name of the document file without file path.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.23 OutputSettings_ApplicationName

Property: `OutputSettings_ApplicationName` as `String`

Description
Sets or retrieves the application name available in the Document Settings dialog.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.24 OutputSettings_Encoding (obsolete)

Property: `OutputSettings_Encoding` as `String`

Description
obsolete
This property is not supported anymore. Mapping output encoding settings do not exist anymore. Components have individual output encoding settings.

17.2.9.25 Parent

Property: `Parent` as `Documents` (read-only)

Description
The parent object according to the object model.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.26 Path

Property: `Path` as `String`

Description
Path of the document file without name.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.27 Save

Method: `Save` ()

Description
Save the document to the file defined by `Document.FullName`.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.28 SaveAs

Method: `SaveAs` (*`i_strFileName`* as `String`)

Description
Save document to specified file name, and set `Document.FullName` to this value if save operation was successful.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.9.29 Saved

Property: `Saved` as `Boolean` (read-only)

Description
`True` if the document was not modified since the last save operation, `False` otherwise.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

© 2018 Altova GmbH

17.2.10 Documents

Represents a collection of Document objects.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Open and create mappings:
`OpenDocument`
`NewDocument`

Iterating through the collection:
`Count`
`Item`
`ActiveDocument`

17.2.10.1 ActiveDocument

Property: `ActiveDocument` as `Document` (read-only)

Description
Retrieves the active document. If no document is open, `null` is returned.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.2 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.3 Count

Property: `Count` as `Integer` (read-only)

Description
Retrieves the number of documents in the collection.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.4 Item

Property: `Item` (*`nIndex`* as `Integer`) as `Document` (read-only)

Description
Retrieves the document at `nIndex` from the collection. Indices start with 1.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.5 NewDocument

Method: `NewDocument` () as `Document`

Description
Creates a new document, adds it to the end of the collection, and makes it the active document.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.10.6 OpenDocument

Method: `OpenDocument` (*`strFilePath`* as `String`) as `Document`

Description
Opens an existing mapping document (`*.mfd`). Adds the newly opened document to the end of the collection and makes it the active document.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

© 2018 Altova GmbH

17.2.10.7 Parent

Property: `Parent` as `Application` (read-only)

Description
The parent object according to the object model.

Errors
1600 The object is no longer valid.
1601 Invalid address for the return parameter was specified.

17.2.11 ErrorMarker

Represents a simple message line. In difference to AppOutputLine, error markers do not have a hierarchical structure.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Access to message information:
`DocumentFileName`
`ErrorLevel`
`Highlight`
`Serialization`
`Text`

17.2.11.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.11.2 DocumentFileName

Property: `DocumentFileName` as `String` (read-only)

Description
Retrieves the name of the mapping file that the error marker is associated with.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.11.3 ErrorLevel

Property: `ErrorLevel` as `ENUMCodeGenErrorLevel` (read-only)

Description
Retrieves the severity of the error.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.11.4 Highlight

Method: `Highlight` ()

Description
Highlights the item that the error marker is associated with. If the corresponding document is not open, it will be opened.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.
1008 The marker points to a location that is no longer valid.

17.2.11.5 Serialization

Property: `Serialization` as `String` (read-only)

Description
Serialize error marker into a string. Use this string in calls to Application.HighlightSerializedMarker or Document.HighlightSerializedMarker to highlight the marked item in the mapping. The string can be persisted and used in other instantiations of MapForce or its Control.

Errors
1900 The object is no longer valid.

© 2018 Altova GmbH

1901 Invalid address for the return parameter was specified.

17.2.11.6 Text

Property: `Text` as `String` (read-only)

Description
Retrieves the message text.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.11.7 Parent

Property: `Parent` as `ErrorMarkers` (read-only)

Description
The parent object according to the object model.

Errors
1900 The object is no longer valid.
1901 Invalid address for the return parameter was specified.

17.2.12 ErrorMarkers

Represents a collection of ErrorMarker objects.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Iterating through the collection:
`Count`
`Item`

17.2.12.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
1800 The object is no longer valid.

1801 Invalid address for the return parameter was specified.

17.2.12.2 Count

Property: `Count` as `Integer` (read-only)

Description
Retrieves the number of error markers in the collection.

Errors
1800 The object is no longer valid.
1801 Invalid address for the return parameter was specified.

17.2.12.3 Item

Property: `Item` (*`nIndex`* as `Integer`) as `ErrorMarker` (read-only)

Description
Retrieves the error marker at `nIndex` from the collection. Indices start with 1.

Errors
1800 The object is no longer valid.
1801 Invalid address for the return parameter was specified.

17.2.12.4 Parent

Property: `Parent` as `Application` (read-only)

Description
The parent object according to the object model.

Errors
1800 The object is no longer valid.
1801 Invalid address for the return parameter was specified.

17.2.13 MapForceView

Represents the current view in the MapForce Mapping tab for a document.
A document has exactly one MapForceView which displays the currently active mapping.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

© 2018 Altova GmbH

View activation and view properties:
Active
ShowItemTypes
ShowLibraryInFunctionHeader
HighlightMyConnections
HighlightMyConnectionsRecursivly

Mapping related properties:
ActiveMapping
ActiveMappingName

Adding items:
InsertWSDLCall
InsertXMLFile
InsertXMLSchema
InsertXMLSchemaWithSample

17.2.13.1 Active

Property: Active as Boolean

Description
Use this property to query if the mapping view is the active view, or set this view to be the active one.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.2 ActiveMapping

Property: ActiveMapping as Mapping

Description
Gets or sets the currently active mapping in the document this MapForceView belongs to.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.3 ActiveMappingName

Property: ActiveMappingName as String

Description

Gets or sets the currently active mapping by name in the document this MapForceView belongs to.

Errors

1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.4 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors

1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.5 HighlightMyConnections

Property: `HighlightMyConnections` as `Boolean`

Description
This property defines whether connections from the selected item only should be highlighed.

Errors

1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.6 HighlightMyConnectionsRecursivey

Property: `HighlightMyConnectionsRecursively` as `Boolean`

Description
This property defines if only the connections coming directly or indirectly from the selected item should be highlighed.

Errors

1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

© 2018 Altova GmbH

17.2.13.7 InsertWSDLCall

Method: `InsertWSDLCall (i_strWSDLFileName as String)`

Description
Adds a new WSDL call component to the mapping.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.8 InsertXMLFile (obsolete)

Method: `InsertXMLFile (i_strXMLFileName as String, i_strRootElement as String)`

Description
obsolete
MapForceView.InsertXMLFile is obsolete. Use Mapping.InsertXMLFile instead.

Adds a new component to the mapping.

The component's internal structure is determined by the schema referenced in the specified XML file.

The second parameter defines the root element of this schema, if there is more than one candidate.
When passing an empty string as root element, the root element of the xml file will be used. Otherwise if more candidates are available, a Select Root Element dialog will pop up regardless if MapForce is visible or not.

The specified XML file is used as the input sample to evaluate the mapping.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.9 InsertXMLSchema (obsolete)

Method: `InsertXMLSchema (i_strSchemaFileName as String, i_strRootElement as String)`

Description
obsolete
MapForceView.InsertXMLSchema is obsolete. Use Mapping.InsertXMLSchema instead.

Adds a new component to the mapping.

The component's internal structure is determined by the specified schema file.

The second parameter defines the root element of this schema if there is more than one candidate.
If the passed root element is an empty string and more candidates are available, a Select Root Element dialog will pop up regardless if MapForce is visible or not.

No XML input sample is assigned to this component.

Errors

1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.10 InsertXMLSchemaWithSample (obsolete)

Method: `InsertXMLSchemaWithSample` (*`i_strSchemaFileName`* as `String`, *`i_strXMLSampleName`* as `String`, *`i_strRootElement`* as `String`)

Description
obsolete
MapForceView.InsertXMLSchemaWithSample is obsolete. Use Mapping.InsertXMLFile instead. Notice, Mapping.InsertXMLFile does not require a parameter for passing the root element. The root element is automatically set as the xml file's root element name.

Adds a new component to the mapping.

The component's internal structure is determined by the specified schema file.

The second parameter is stored as the XML input sample for mapping evaluation.

The third parameter defines the root element of this schema if there is more than one candidate. When passing an empty string as root element, the root element of the xml sample file will be used.

Errors

1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.11 Parent

Property: `Parent` as `Document` (read-only)

Description
The parent object according to the object model.

Errors

1300 The object is no longer valid.
1301 Invalid address for the return parameter was specified.

© 2018 Altova GmbH

17.2.13.12 ShowItemTypes

Property: `ShowItemTypes` as `Boolean`

Description
This property defines if types of items should be shown in the mapping diagram.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.13.13 ShowLibraryInFunctionHeader

Property: `ShowLibraryInFunctionHeader` as `Boolean`

Description
This property defines whether the name of the function library should be part of function names.

Errors
1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

17.2.14 Mapping

A Mapping object represents a mapping in a document, so the main mapping, or a local user-defined-function mapping.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Mapping properties:
`IsMainMapping`
`Name`

Components in the mapping:
`Components`

Adding items:
`CreateConnection`
`InsertFunctionCall`
`InsertXMLFile`
`InsertXMLSchema`
`InsertXMLSchemaInputParameter`
`InsertXMLSchemaOutputParameter`

17.2.14.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.14.2 Components

Property: `Components` as `Components` (read-only)

Description
Returns a collection of all components in the current mapping.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.14.3 CreateConnection

Method: CreateConnection(DatapointFrom as Datapoint, DatapointTo as Datapoint) as Connection

Description
Creates a connection between the two supplied datapoints (DatapointFrom & DatapointTo).

It will fail to do so if the DatapointFrom is not an output-side datapoint, the DatapointTo is not an input-side datapoint, or a connection between these two datapoints already exists.

Errors
1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1240 Changing the document not allowed. It is read-only.
1241 Failed creating the connection.

17.2.14.4 InsertFunctionCall

Method: InsertFunctionCall(strFunctionName as String, strLibraryName as String) as Component

Description

© 2018 Altova GmbH

Inserts a function call component into the current mapping.

The specified library and function names indicate the function or user-defined function to be called.

Errors

1200	The application object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.
1242	Failed creating function call component.

17.2.14.5 InsertXMLFile

Method: InsertXMLFile(i_strFileName as String, i_strSchemaFileName as String) as Component

Description
Adds a new XML schema component to the mapping.

The component's internal structure is determined by the schema referenced in the specified XML file (i_strFileName) or, if the XML file does not reference a schema file, by the separately specified schema file (i_strSchemaFileName).

If the XML file has a schema file reference, then the parameter i_strSchemaFileName is ignored.

The root element of the XML file will be used in the component.

The specified XML file is used as the input sample to evaluate the mapping.

Errors

1200	The application object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.
1244	Failed creating component.

17.2.14.6 InsertXMLSchema

Method: InsertXMLSchema(i_strSchemaFileName as String, i_strXMLRootName as String) as Component

Description
Adds a new XML schema component to the mapping.

The component's internal structure is determined the specified schema file.

The second parameter defines the root element of this schema if there is more than one candidate.

If the passed root element is an empty string and more candidates are available, a Select Root Element dialog will pop up if MapForce is visible. If MapForce is invisible, no dialog will pop up and only an error is returned.

No XML input sample is assigned to this component.

Errors

1200	The application object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.
1244	Failed creating component.

17.2.14.7 InsertXMLSchemaInputParameter

Method: InsertXMLSchemaInputParameter(strParamName as String, strSchemaFileName as String, strXMLRootElementName as String) as Component

Description
Inserts an XML schema input parameter component into the current mapping.

The current mapping has to be a **user-defined** function. Trying to insert it (the schema input parameter) into the **main mapping** will fail.

strParamName is the name of the input parameter component to create and **strSchemaFileName** and **strXMLRootElementName** indicate the respective schema file and the root element of the schema file to be used.

If the passed root element is an empty string and more candidates are available, a Select Root Element dialog will pop up if MapForce is visible. If MapForce is invisible, no dialog will pop up and only an error is returned.

Errors

1200	The application object is no longer valid.
1201	Invalid address for the return parameter was specified.
1240	Changing the document not allowed. It is read-only.
1243	Failed creating parameter component.
1245	This operation is not supported for the main mapping.

17.2.14.8 InsertXMLSchemaOutputParameter

Method: InsertXMLSchemaOutputParameter(strParamName as String, strSchemaFileName as String, strXMLRootElementName as String) as Component

Description
Inserts an XML schema output parameter component into the current mapping.

The current mapping has to be a **user-defined** function. Trying to insert it (the schema output paramter) into the **main** mapping will fail.

strParamName is the name of the output parameter component to create and **strSchemaFileName** and **strXMLRootElementName** indicate the schema file and the root element of the schema file to be used respectively.

If the passed root element is an empty string and more candidates are available, a Select Root Element dialog will pop up if MapForce is visible. If MapForce is invisible, no dialog will pop up and only an error is returned.

© 2018 Altova GmbH

Errors

1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.
1240 Changing the document not allowed. It is read-only.
1243 Failed creating parameter component.
1245 This operation is not supported for the main mapping.

17.2.14.9 IsMainMapping

Property: `IsMainMapping` as `Boolean` (read-only)

Description
Indicates if the current mapping is the main mapping of the document the mapping is in.

True means it is the main mapping.
False means it is a user defined function (UDF).

Errors

1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.14.10 Name

Property: `Name` as `String` (read-only)

Description
The name of the mapping / user defined function (UDF).

Errors

1200 The application object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.14.11 Parent

Property: `Parent` as `Document` (read-only)

Description
The parent object according to the object model.

Errors

1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.15 Mappings

Represents a collection of Mapping objects.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Iterating through the collection:
`Count`
`Item`

17.2.15.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.15.2 Count

Property: `Count` as `Integer` (read-only)

Description
Retrieves the number of mappings in the collection.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.15.3 Item

Property: `Item` (*`nIndex`* as `Integer`) as `Mapping` (read-only)

Description
Retrieves the mapping at `nIndex` from the collection. Indices start with 1.

Errors
1200 The object is no longer valid.

© 2018 Altova GmbH

1201 Invalid address for the return parameter was specified.

17.2.15.4 Parent

Property: `Parent` as `Document` (read-only)

Description
The parent object according to the object model.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

17.2.16 Options

This object gives access to all MapForce options available in the **Tools | Options** dialog.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

General options:
`ShowLogoOnPrint`
`ShowLogoOnStartup`
`UseGradientBackground`

Options for code generation:
`DefaultOutputEncoding`
`DefaultOutputByteOrder`
`DefaultOutputByteOrderMark`
`XSLTDefaultOutputDirectory`
`CodeDefaultOutputDirectory`
`CPPSettings_DOMType`
`CPPSettings_GenerateVC6ProjectFile`
`CppSettings_GenerateVSProjectFile`
`CPPSettings_LibraryType`
`CPPSettings_UseMFC`
`CSharpSettings_ProjectType`

17.2.16.1 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the application's top-level object.

Errors

1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.2 CodeDefaultOutputDirectory

Property: `CodeDefaultOutputDirectory` as `String`

Description

Specifies the target directory where files generated by `Document.GenerateCppCode`, `Document.GenerateJavaCode` and `Document.GenerateCHashCode`, are placed.

Errors

1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.3 CPPSettings_DOMType

Property: `CPPSettings_DOMType` as `ENUMDOMType`

Description

Specifies the DOM type used by `Document.GenerateCppCode`.

Errors

1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.
1402 The parameter value is out of range
1403 The parameter value is not available anymore

17.2.16.4 CPPSettings_GenerateVC6ProjectFile

Property: `CPPSettings_GenerateVC6ProjectFile` as `Boolean`

Description

Specifies if VisualC++ 6.0 project files should be generated by `Document.GenerateCppCode`.

Errors

1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.5 CppSettings_GenerateVSProjectFile

Property: `CppSettings_GenerateVSProjectFile` as `ENUMProjectType`

Description

© 2018 Altova GmbH

Specifies the version of Visual Studio in which project files should be generated by `Document.GenerateCppCode`.

Errors

1400	The application object is no longer valid.
1401	Invalid address for the return parameter was specified.
1402	The parameter value is out of range
1403	The paramater value is not available anymore

17.2.16.6 CPPSettings_LibraryType

Property: `CPPSettings_LibraryType` as `ENUMLibType`

Description
Specifies the library type used by `Document.GenerateCppCode`.

Errors

1400	The application object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.2.16.7 CPPSettings_UseMFC

Property: `CPPSettings_UseMFC` as `Boolean`

Description
Specifies if MFC support should be used by C++ code generated by `Document.GenerateCppCode`.

Errors

1400	The application object is no longer valid.
1401	Invalid address for the return parameter was specified.

17.2.16.8 CSharpSettings_ProjectType

Property: `CSharpSettings_ProjectType` as `ENUMProjectType`

Description
Specifies the type of C# project used by `Document.GenerateCHashCode`.

Errors

1400	The application object is no longer valid.
1401	Invalid address for the return parameter was specified.
1402	The parameter value is out of range
1403	The paramater value is not available anymore

17.2.16.9 DefaultOutputByteOrder

Property: `DefaultOutputByteOrder` as `String`

Description
Byte order for the file encoding used for output files.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.10 DefaultOutputByteOrderMark

Property: `DefaultOutputByteOrderMark` as `Boolean`

Description
Indicates if a byte order mark (BOM), is to be included in the file encoding of output files.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.11 DefaultOutputEncoding

Property: `DefaultOutputEncoding` as `String`

Description
File encoding used for output files.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.12 GenerateWrapperClasses

Property: `GenerateWrapperClasses` as `Boolean`

Description
Indicates if wrapper classes are also to be generated when generating code.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

© 2018 Altova GmbH

17.2.16.13 JavaSettings_ApacheAxisVersion (obsolete)

Property: `JavaSettings_ApacheAxisVersion` as `ENUMApacheAxisVersion`

Description
Specifies the Apache Axis version to use when generating Java code for web service implementations with SOAP 1.1.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.14 Parent

Property: `Parent` as `Application` (read-only)

Description
The parent object according to the object model.

Errors
1400 The object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.15 ShowLogoOnPrint

Property: `ShowLogoOnPrint` as `Boolean`

Description
Show or hide the MapForce logo on printed outputs.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.16 ShowLogoOnStartup

Property: `ShowLogoOnStartup` as `Boolean`

Description
Show or hide the MapForce logo on application startup.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.17 UseGradientBackground

Property: `UseGradientBackground` as `Boolean`

Description
Set or retrieve the background color mode for a mapping window.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.16.18 XSLTDefaultOutputDirectory

Property: `XSLTDefaultOutputDirectory` as `String`

Description
Specifies the target directory where files generated by `Document.GenerateXSLT` are placed.

Errors
1400 The application object is no longer valid.
1401 Invalid address for the return parameter was specified.

17.2.17 Project

A Project object represents a project and its tree of project items in MapForce.

Events
Events

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

File handling:
`FullName`
`Name`
`Path`
`Saved`
`Save`
`Close`

Project tree navigation:
`Count`
`Item`
`_NewEnum`

© 2018 Altova GmbH

Project tree manipulation:
`AddActiveFile`
`AddFile`
`InsertWebService` (Enterprise edition only)
`CreateFolder`

Code-generation:
`Output_Folder`
`Output_Language`
`Output_TextEncoding`
`Java_BasePackageName`
`GenerateCode`
`GenerateCodeEx`
`GenerateCodeIn`
`GenerateCodeInEx`

For examples of how to use the properties and methods listed above, see Example: Project Support. For operations with Web services, the MapForce Enterprise edition is required.

17.2.17.1 Events

This object supports the following events:

`OnProjectClosed`

17.2.17.1.1 OnProjectClosed

Event: `OnProjectClosed` (*`i_objProject`* as `Project`)

Description
This event is triggered when the project is closed. The project object passed into the event handler should not be accessed. The corresponding open event is `Application.OnProjectOpened`.

17.2.17.2 _NewEnum

Property: `_NewEnum` () as `IUnknown` (read-only)

Description
This property supports language-specific standard enumeration.

Errors
1500 The object is no longer valid.

Examples

```
// ------------------------------------------------------------
```

```
// JScript sample - enumeration of a project's project items.
function AllChildrenOfProjectRoot()
{
    objProject = objMapForce.ActiveProject;
    if ( objProject != null )
    {
        for ( objProjectIter = new Enumerator(objProject); !
objProjectIter.atEnd(); objProjectIter.moveNext() )
        {
            objProjectItem = objProjectIter.item();

            // do something with project item here
        }
    }
}

// ------------------------------------------------------------
// JScript sample - iterate all project items, depth first.
function IterateProjectItemsRec(objProjectItemIter)
{
    while ( ! objProjectItemIter.atEnd() )
    {
        objProjectItem = objProjectItemIter.item();
        // do something with project item here

        IterateProjectItemsRec( new Enumerator(objProjectItem) );

        objProjectItemIter.moveNext();
    }
}
function IterateAllProjectItems()
{
    objProject = objMapForce.ActiveProject;
    if ( objProject != null )
    {
        IterateProjectItemsRec( new Enumerator(objProject) );
    }
}
```

17.2.17.3 AddActiveFile

Method: `AddActiveFile` () as `ProjectItem`

Description
Adds the currently open document to the mapping folder of the project's root.

Errors

1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.
1503	No active document is available.
1504	Active documents needs to be given a path name before it can be added to the project.

© 2018 Altova GmbH

1705 Mapping could not be assigned to project. Maybe it is already contained in the target folder.

17.2.17.4 AddFile

Method: `AddFile` (*`i_strFileName`* as `String`) as `ProjectItem`

Description
Adds the specified document to the mapping folder of the project's root.

Errors
1500 The object is no longer valid.
1501 The file name is empty.
Invalid address for the return parameter was specified.
1705 Mapping could not be assigned to project.
The file does not exist or is not a MapForce mapping.
Maybe the file is already assigned to the target folder.

17.2.17.5 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the top-level application object.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.6 Close

Method: `Close` ()

Description
Closes the project without saving.

Errors
1500 The object is no longer valid.

17.2.17.7 Count

Property: `Count` as `Integer` (read-only)

Description

Retrieves number of children of the project's root item.

Errors
1500 The object is no longer valid.

Examples
See Item or _NewEnum.

17.2.17.8 CreateFolder

Method: CreateFolder (*i_strFolderName* as String) as ProjectItem

Description
Creates a new folder as a child of the project's root item.

Errors
1500 The object is no longer valid.
1501 Invalid folder name or invalid address for the return parameter was specified.

17.2.17.9 FullName

Property: FullName as String (read-only)

Description
Path and name of the project file.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.10 GenerateCode

Method: GenerateCode ()

Description
Generates code for all project items of the project. The code language and output location is determined by properties of the project and project items.

Errors
1500 The object is no longer valid.
1706 Error during code generation

© 2018 Altova GmbH

17.2.17.11 GenerateCodeEx

Method: `GenerateCode` () `as` `ErrorMarkers`

Description
Generates code for all project items of the project. The code language and output location are determined by properties of the project and project items. An object that can be used to iterate through all messages issued by the code generation process is returned. These messages are the same as those shown in the *Messages* window of MapForce.

Errors

1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.
1706	Error during code generation

17.2.17.12 GenerateCodeIn

Method: `GenerateCodeIn` (*`i_nLanguage`* as `ENUMProgrammingLanguage`)

Description
Generates code for all project items of the project in the specified language. The output location is determined by properties of the project and project items.

Errors

1500	The object is no longer valid.
1706	Error during code generation

17.2.17.13 GenerateCodeInEx

Method: `GenerateCodeIn` (*`i_nLanguage`* as `ENUMProgrammingLanguage`) `as` `ErrorMarkers`

Description
Generates code for all project items of the project in the specified language. The output location is determined by properties of the project and project items. An object that can be used to iterate through all messages issued by the code generation process is returned. These messages are the same as those shown in the *Messages* window of MapForce.

Errors

1500	The object is no longer valid.
1501	Invalid address for the return parameter was specified.
1706	Error during code generation

17.2.17.14 InsertWebService

Method: `InsertWebService (i_strWSDLFile as String, i_strService as String, i_strPort as String, i_bGenerateMappings as Boolean) as ProjectItem`

Description
Inserts a new Web service project into the project's Web service folder. If `i_bGenerateMappings` is true, initial mapping documents for all ports get generated automatically.

Errors

1500	The object is no longer valid.
1501	WSDL file can not be found or is invalid. Service or port names are invalid. Invalid address for the return parameter was specified.
1503	Operation not supported by current edition.

17.2.17.15 Item

Property: `Item(i_nItemIndex as Integer) as ProjectItem` (read-only)

Description
Returns the child at `i_nItemIndex` position of the project's root. The index is zero-based. The largest valid index is `Count-1`. For an alternative to visit all children see `_NewEnum`.

Errors

1500	The object is no longer valid.

Examples

```
// ------------------------------------------------------------
// JScript code snippet - enumerate children using Count and Item.
for( nItemIndex = 0; nItemIndex < objProject.Count; nItemIndex++ )
{
    objProjectItem = objProject.Item(nItemIndex);
    // do something with project item here
}
```

17.2.17.16 Java_BasePackageName

Property: `Java_BasePackageName as String`

Description
Sets or gets the base package name of the Java packages that will be generated. This property is used only when generating Java code.

Errors

© 2018 Altova GmbH

1500 The object is no longer valid.
1501 Invalid package name specified.
Invalid address for the return parameter was specified.

17.2.17.17 Name

Property: `Name` as `String` (read-only)

Description
Name of the project file without file path.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.18 Output_Folder

Property: `Output_Folder` as `String`

Description
Sets or gets the default output folder used with `GenerateCode` and `GenerateCodeIn`. Project items can overwrite this value in their `CodeGenSettings_OutputFolder` property, when `CodeGenSettings_UseDefault` is set to false.

Errors
1500 The object is no longer valid.
1501 Invalid folder name specified.
Invalid address for the return parameter was specified.

17.2.17.19 Output_Language

Property: `Output_Language` as `ENUMProgrammingLanguage`

Description
Sets or gets the default language for code generation when using `GenerateCode`. Project items can overwrite this value in their `CodeGenSettings_OutputLanguage` property, when `CodeGenSettings_UseDefault` is set to false.

Errors
1500 The object is no longer valid.
1501 Invalid language specified.
Invalid address for the return parameter was specified.

17.2.17.20 Output_TextEncoding

Property: `Output_TextEncoding` as `String`

Description
Sets or gets the text encoding used when generating XML-based code.

Errors
1500 The object is no longer valid.
1501 Invalid text encoding specified.
Invalid address for the return parameter was specified.

17.2.17.21 Parent

Property: `Parent` as `Application` (read-only)

Description
The parent object according to the object model.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.22 Path

Property: `Path` as `String` (read-only)

Description
Path of the project file without name.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.17.23 Save

Method: `Save` ()

Description
Saves the project to the file defined by `FullName`.

Errors
1500 The object is no longer valid.
1502 Can't save to file.

© 2018 Altova GmbH

17.2.17.24 Saved

Property: `Saved` as Boolean (read-only)

Description
Tr ue if the project was not modified since the last Save operation, f al se otherwise.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

17.2.18 ProjectItem

A ProjectItem object represents one item in a project tree.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Project tree navigation:
`Count`
`Item`
`_NewEnum`

Project item properties:
`Kind`
`Name`
`WSDLFile` (only available to Web service project items)
`QualifiedName` (only available to Web service project items)

Project tree manipulation:
`AddActiveFile` (only available to folder items)
`AddFile` (only available to folder items)
`CreateFolder` (only available to folder items)
`CreateMappingForProject` (only available to Web service operations)
`Remove`

Document access:
`Open` (only available to mapping items and Web service operations)

Code-generation:
`CodeGenSettings_UseDefault`
`CodeGenSettings_OutputFolder`
`CodeGenSettings_Language`
`GenerateCode`
`GenerateCodeEx`

GenerateCodeIn
GenerateCodeInEx

For examples of how to use the properties and methods listed above, see Example: Project Support. For operations with Web services, the MapForce Enterprise edition is required.

17.2.18.1 _NewEnum

Property: `_NewEnum` () as `IUnknown` (read-only)

Description
This property supports language specific standard enumeration.

Errors
1700 The object is no longer valid.

Examples
See `Project.Item` or `Project._NewEnum`.

17.2.18.2 AddActiveFile

Method: `AddActiveFile` () as `ProjectItem`

Description
Adds the currently active document to this project item if it is a valid child. Otherwise, the document is added to the Mapping Folder of the project's root.

Errors
1700 The object is no longer valid.
1701 The file name is empty.
Invalid address for the return parameter was specified.
1703 No active document is available.
1704 Active documents needs to be given a path name before it can be added to the project.
1705 Mapping could not be assigned to project.
The file does not exist or is not a MapForce mapping.
Maybe the file is already assigned to the target folder.

17.2.18.3 AddFile

Method: `AddFile` (*`i_strFileName`* as `String`) as `ProjectItem`

Description
Adds the specified document to this project item if it is a valid child. Otherwise, the document is added to the Mapping Folder of the project's root.

Errors

© 2018 Altova GmbH

1700 The object is no longer valid.
1701 The file name is empty.
Invalid address for the return parameter was specified.
1705 Mapping could not be assigned to project.
The file does not exist or is not a MapForce mapping.
Maybe the file is already assigned to the target folder.

17.2.18.4 Application

Property: `Application` as `Application` (read-only)

Description
Retrieves the top-level application object.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.

17.2.18.5 CodeGenSettings_Language

Property: `CodeGenSettings_Language` as `ENUMProgrammingLanguage`

Description
Gets or sets the language to be used with `GenerateCode` or `Project.GenerateCode`. This property is consulted only if `CodeGenSettings_UseDefault` is set to false.

Errors
1700 The object is no longer valid.
1701 Invalid language or invalid address for the return parameter was specified.

17.2.18.6 CodeGenSettings_OutputFolder

Property: `CodeGenSettings_OutputFolder` as `String`

Description
Gets or sets the output directory to be used with `GenerateCode`, `GenerateCodeIn`, `Project.GenerateCode` or `Project.GenerateCodeIn`. This property is consulted only if `CodeGenSettings_UseDefault` is set to false.

Errors
1700 The object is no longer valid.
1701 An invalid output folder or an invalid address for the return parameter was specified.

17.2.18.7 CodeGenSettings_UseDefault

Property: `CodeGenSettings_UseDefault` as `Boolean`

Description
Gets or sets whether output directory and code language are used as defined by either (a) the parent folders, or (b) the project root. This property is used with calls to `GenerateCode`, `GenerateCodeIn`, `Project.GenerateCode` and Project.GenerateCodeIn. If this property is set to false, the values of `CodeGenSettings_OutputFolder` and `CodeGenSettings_Language` are used to generate code for this project item..

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.

17.2.18.8 Count

Property: `Count` as `Integer` (read-only)

Description
Retrieves number of children of this project item. Also see `Item`.

Errors
1700 The object is no longer valid.

Examples
See `Project.Item` or `Project._NewEnum`.

17.2.18.9 CreateFolder

Method: `CreateFolder` (*`i_strFolderName`* as `String`) as `ProjectItem`

Description
Creates a new folder as a child of this project item.

Errors
1700 The object is no longer valid.
1701 Invalid folder name or invalid address for the return parameter was specified.
1702 The project item does not support children.

17.2.18.10 CreateMappingForProject

Method: `CreateMappingForProject` (*`i_strFileName`* as `String`) as `ProjectItem`

© 2018 Altova GmbH

Description
Creates an initial mapping document for a Web service operation and saves it to `i_strFileName`. When using `Project.InsertWebService` you can use the `i_bGenerateMappings` flag to let MapForce automatically generate initial mappings for all ports.

Errors

1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.
1707	Cannot create new mapping. The project item does not support auto-creation of initial mappings or a mapping already exists.
1708	Operation not supported in current edition.

17.2.18.11 GenerateCode

Method: `GenerateCode` ()

Description
Generates code for this project item and its children. The code language and output location is determined by `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.

Errors

1700	The object is no longer valid.
1706	Error during code generation.

17.2.18.12 GenerateCodeEx

Method: `GenerateCode` () `as` `ErrorMarkers`

Description
Generates code for this project item and its children. The code language and output location are determined by `CodeGenSettings_UseDefault`, `CodeGenSettings_Language` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.

An object that can be used to iterate through all messages issued by the code generation process is returned. These messages are the same as those shown in the *Messages* window of MapForce.

Errors

1700	The object is no longer valid.
1701	Invalid address for the return parameter was specified.
1706	Error during code generation.

17.2.18.13 GenerateCodeIn

Method: `GenerateCodeIn` (*`i_nLanguage`* as `ENUMProgrammingLanguage`)

Description
Generates code for the project item and its children in the specified language. The output location is determined by `CodeGenSettings_UseDefault` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.

Errors

1700	The object is no longer valid.
1701	Invalid language specified.
1706	Error during code generation.

17.2.18.14 GenerateCodeInEx

Method: `GenerateCodeIn` (*`i_nLanguage`* as `ENUMProgrammingLanguage`) as `ErrorMarkers`

Description
Generates code for the project item and its children in the specified language. The output location is determined by `CodeGenSettings_UseDefault` and `CodeGenSettings_OutputFolder`. Children of this project item can have their own property settings related to code-generation.

An object that can be used to iterate through all messages issued by the code generation process is returned. These messages are the same as those shown in the *Messages* window of MapForce.

Errors

1700	The object is no longer valid.
1701	Invalid language specified or invalid address for the return parameter was specified.
1706	Error during code generation.

17.2.18.15 Item

Property: `Item(`*`i_nItemIndex`* as `Integer)` as `ProjectItem` (read-only)

Description
Returns the child at `i_nItemIndex` position of this project item. The index is zero-based. The largest valid index is `Count` - 1.
For an alternative to visit all children see `_NewEnum`.

Errors

1700	The object is no longer valid.

© 2018 Altova GmbH

Examples
See `Project.Item` or `Project._NewEnum`.

17.2.18.16 Kind

Property: `Kind` as `ENUMProjectItemType` (read-only)

Description
Retrieves the kind of the project item. Availability of some properties and the applicability of certain methods is restricted to specific kinds of project items. The description of all methods and properties contains information about these restrictions.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.

17.2.18.17 Name

Property: `Name` as `String`

Description
Retrieves or sets the name of a project item. The name of most items is read-only. Exceptions are user-created folders, the names of which can be altered after creation.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.
1702 Project item does not allow to alter its name.

17.2.18.18 Open

Method: `Open` () as `Document`

Description
Opens the project item as a document or makes the corresponding document the active one, if it is already open. The project item must be a MapForce mapping or, for Enterprise edition only, Web service operation.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.
1702 The project item does not refer to a MapForce mapping file.
1708 Operation not supported in current edition.

17.2.18.19 Parent

Property: `Parent` as `Project` (read-only)

Description
Retrieves the project that this item is a child of. Has the same effect as `Application.ActiveProject`.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.

17.2.18.20 QualifiedName

Property: `QualifiedName` as `String` (read-only)

Description
Retrieves the qualified name of a Web service item.

Errors
1700 The object is no longer valid.
1701 Invalid address for the return parameter was specified.
1702 The project item is not a part of a Web service.

17.2.18.21 Remove

Method: `Remove` ()

Description
Remove this project item and all its children from the project tree.

Errors
1700 The object is no longer valid.

17.2.18.22 WSDLFile

Property: `WSDLFile` as `String` (read-only)

Description
Retrieves the file name of the WSDL file defining the Web service that hosts the current project item.

Errors
1700 The object is no longer valid.

© 2018 Altova GmbH

1701 Invalid address for the return parameter was specified.
1702 The project item is not a part of a Web service.

17.3 Enumerations

This is a list of all enumerations used by the API de MapForce. If your scripting environment does not support enumerations, use the number-values instead.

17.3.1 ENUMApacheAxisVersion (obsolete)

Description
Enumeration values to select the Apache Axis version.

Possible values:

eApacheAxisVersion_Axis	= 1
eApacheAxisVersion_Axis2	= 2

17.3.2 ENUMApplicationStatus

Description
Enumeration values to indicate the status of the application.

Possible values:

eApplicationRunning	= 0
eApplicationAfterLicenseCheck	= 1
eApplicationBeforeLicenseCheck	= 2
eApplicationConcurrentLicenseCheckFailed	= 3

17.3.3 ENUMAppOutputLine_Severity

Description
Enumeration values to identify the severity of an AppOutputLine.

Possible values:

eSeverity_Undefined	= -1
eSeverity_Info	= 0
eSeverity_Warning	= 1
eSeverity_Error	= 2
eSeverity_CriticalError	= 3
eSeverity_Success	= 4
eSeverity_Summary	= 5
eSeverity_Progress	= 6
eSeverity_DataEdit	= 7
eSeverity_ParserInfo	= 8
eSeverity_PossibleInconsistencyWarning	= 9
eSeverity_Message	= 10
eSeverity_Document	= 11
eSeverity_Rest	= 12
eSeverity_NoSelect	= 13
eSeverity_Select	= 14

© 2018 Altova GmbH

eSeverity_Autoinsertion	= 15
eSeverity_GlobalResources_DefaultWarning	= 16

17.3.4 ENUMAppOutputLine_TextDecoration

Description
Enumeration values for the different kinds of text decoration of an AppOutputLine.

Possible values:

eTextDecorationDefault	= 0
eTextDecorationBold	= 1
eTextDecorationDebugValues	= 2
eTextDecorationDB_ObjectName	= 3
eTextDecorationDB_ObjectLink	= 4
eTextDecorationDB_ObjectKind	= 5
eTextDecorationDB_TimeoutValue	= 6
eTextDecorationFind_MatchingString	= 7
eTextDecorationValidation_Speclink	= 8
eTextDecorationValidation_ErrorPosition	= 9
eTextDecorationValidation_UnkownParam	= 10

17.3.5 ENUMCodeGenErrorLevel

Description
Enumeration values to identify severity of code generation messages.

Possible values:

eCodeGenErrorLevel_Information	= 0
eCodeGenErrorLevel_Warning	= 1
eCodeGenErrorLevel_Error	= 2
eCodeGenErrorLevel_Undefined	= 3

17.3.6 ENUMComponentDatapointSide

Description
Enumeration values to indicate the side of a datapoint on its component.

Possible values:

eDatapointSideInput	= 0
eDatapointSideOutput	= 1

See also
GetRootDatapoint

17.3.7 ENUMComponentSubType

Description
Enumeration values to indicate component sub types.

Possible values:

eComponentSubType_None	= 0
eComponentSubType_Text_EDI	= 1
eComponentSubType_Text_Flex	= 2
eComponentSubType_Text_CSVFLF	= 3

17.3.8 ENUMComponentType

Description
Enumeration values to indicate component types.

Possible values:

eComponentType_Unknown	= 0
eComponentType_XML	= 1
eComponentType_DB	= 2
eComponentType_Text	= 3
eComponentType_Excel	= 4
eComponentType_WSDL	= 5
eComponentType_XBRL	= 6

17.3.9 ENUMComponentUsageKind

Description
Enumeration values to indicate component usage kind.

Possible values:

eComponentUsageKind_Unknown	= 0
eComponentUsageKind_Instance	= 1
eComponentUsageKind_Input	= 2
eComponentUsageKind_Output	= 3
eComponentUsageKind_Variable	= 4

17.3.10 ENUMConnectionType

Description
Enumeration values to indicate the type of a connection.

Possible values:

eConnectionTypeTargetDriven	= 0
eConnectionTypeSourceDriven	= 1

© 2018 Altova GmbH

eConnectionTypeCopyAll = 2

See also
ConnectionType

17.3.11 ENUMDOMType

Description
Enumeration values to specify the DOM type used by generated C++ mapping code.

Possible values:

eDOMType_xerces3	= 2
eDOMType_msxml6	= 3

Obsolete values

eDOMType_msxml4	= 0
eDOMType_xerces	= 1

Obsolete in this context means that this value is not supported and should not be used.

`eDOMType_xerces3` indicates Xerces 3.x usage.

17.3.12 ENUMLibType

Description
Enumeration values to specify the library type used by the generated C++ mapping code.

Possible values:

eLibType_static	= 0
eLibType_dll	= 1

17.3.13 ENUMProgrammingLanguage

Description
Enumeration values to select a programming language.

Possible values:

eUndefinedLanguage	= -1
eJava	= 0
eCpp	= 1
eCSharp	= 2
eXSLT	= 3
eXSLT2	= 4
eXQuery	= 5

17.3.14 ENUMProjectItemType

WDescription
Enumeration to identify the different kinds of project items that can be children of Project or folder-like ProjectItems.

Possible values:

eProjectItemType_Invalid	= -1
eProjectItemType_MappingFolder	= 0
eProjectItemType_Mapping	= 1
eProjectItemType_WebServiceFolder	= 2
eProjectItemType_WebServiceRoot	= 3
eProjectItemType_WebServiceService	= 4
eProjectItemType_WebServicePort	= 5
eProjectItemType_WebServiceOperation	= 6
eProjectItemType_ExternalFolder	= 7
eProjectItemType_LibrarzFolder	= 8
eProjectItemType_ResourceFolder	= 9
eProjectItemType_VirtualFolder	= 10

See also
ProjectItem.Kind

17.3.15 ENUMProjectType

Description
Enumeration values to select a project type for generated C# and C++ mapping code.

Possible values:	eVisualStudio2008Project	= 5
	eVisualStudio2010Project	= 6
	eVisualStudio2013Project	= 7
	eVisualStudio2015Project	= 8
	eVisualStudio2017Project	= 9
Obsolete values	eVisualStudioProject	= 0
	eVisualStudio2003Project	= 1
	eBorlandProject	= 2
	eVisualStudio2005Project	= 4

Obsolete in this context means that this value is not supported and should not be used.

17.3.16 ENUMSearchDatapointFlags

Description
Enumeration values used as bit-flags; to be used as combination of flags when searching for a datapoint.

Possible values:

© 2018 Altova GmbH

eSearchDatapointElement	= 1
eSearchDatapointAttribute	= 2

See also
GetChild

17.3.17 ENUMViewMode

Description
Enumeration values to select a MapForce view.

Possible values:

eMapForceView	= 0
eXSLView	= 1
eOutputView	= 2

Altova MapForce 2018 Enterprise Edition

ActiveX Integration

18 ActiveX Integration

The MapForce user interface and the functionality described in this section can be integrated into custom applications that can consume ActiveX controls. ActiveX technology enables a wide variety of languages to be used for integration, such as C++, C#, VB.NET, HTML. (Note that ActiveX components integrated in HTML must be run with Microsoft Internet Explorer versions and platforms that support ActiveX). All components are full OLE Controls. Integration into Java is provided through wrapper classes.

> To integrate the ActiveX controls into your custom code, the MapForce Integration Package must be installed (see http://www.altova.com/ide_integration.html). Ensure that you install MapForce first, and then the MapForce Integration Package. Other prerequisites apply, depending on language and platform (see Prerequisites).

You can flexibly choose between two different levels of integration: application level and document level.

Integration at application level means embedding the complete interface of MapForce (including its menus, toolbars, panes, etc) as an ActiveX control into your custom application. For example, in the most simple scenario, your custom application could consist of only one form that embeds the MapForce graphical user interface. This approach is easier to implement than integration at document level but may not be suitable if you need flexibility to configure the MapForce graphical user interface according to your custom requirements.

Integration at document level means embedding MapForce into your own application piece-by-piece. This includes implementing not only the main MapForce control but also the main document editor window, and, optionally, any additional windows. This approach provides greater flexibility to configure the GUI, but requires advanced interaction with ActiveX controls in your language of choice.

The sections Integration at the Application Level and Integration at Document Level describe the key steps at these respective levels. The ActiveX Integration Examples section provides examples in C#, HTML, and Java. Looking through these examples will help you to make the right decisions quickly. The Object Reference section describes all COM objects that can be used for integration, together with their properties and methods.

For information about using MapForce as a Visual Studio plug-in, see MapForce in Visual Studio.

© 2018 Altova GmbH

18.1 Prerequisites

To integrate the MapForce ActiveX control into a custom application, the following must be installed on your computer:

- MapForce
- The MapForce Integration Package, available for download at http://www.altova.com/ide_integration.html

To integrate the 64-bit ActiveX control, install the 64-bit versions of MapForce and MapForce Integration Package. For applications developed under Microsoft .NET platform with Visual Studio, both the 32-bit and 64-bit versions of MapForce and MapForce Integration Package must be installed, as explained below.

Microsoft .NET (C#, VB.NET) with Visual Studio

To integrate the MapForce ActiveX control into a 32-bit application developed under Microsoft .NET, the following must be installed on your computer:

- Microsoft .NET Framework 4.0 or later
- Visual Studio 2008/2010/2012/2013/2015/2017
- MapForce 32-bit and MapForce Integration Package 32-bit
- The ActiveX controls must be added to the Visual Studio toolbox (see Adding the ActiveX Controls to the Toolbox).

If you want to integrate the 64-bit ActiveX control, the following prerequisites apply in addition to the ones above:

- MapForce 32-bit and MapForce Integration Package 32-bit must still be installed (this is required to provide the 32-bit ActiveX control to the Visual Studio designer, since Visual Studio runs on 32-bit)
- MapForce 64-bit and MapForce Integration Package 64-bit must be installed (provides the actual 64-bit ActiveX control to your custom application at runtime)
- In Visual Studio, create a 64-bit build configuration and build your application using this configuration. For an example, see Running the Sample C# Solution.

Java

To integrate the MapForce ActiveX control into Java application using the Eclipse development environment, the following must be installed on your computer:

- Java Runtime Environment (JRE) or Java Development Kit (JDK) 7 or later
- Eclipse
- MapForce and MapForce Integration Package

Note: To run the 64-bit version of the MapForce ActiveX control, use a 64-bit version of Eclipse, as well as the 64-bit version of MapForce and the MapForce Integration Package.

MapForce integration and deployment on client computers

If you create a .NET application and intend to distribute it to other clients, you will need to install the following on the client computer(s):

- MapForce
- The MapForce Integration Package
- The custom integration code or application.

© 2018 Altova GmbH

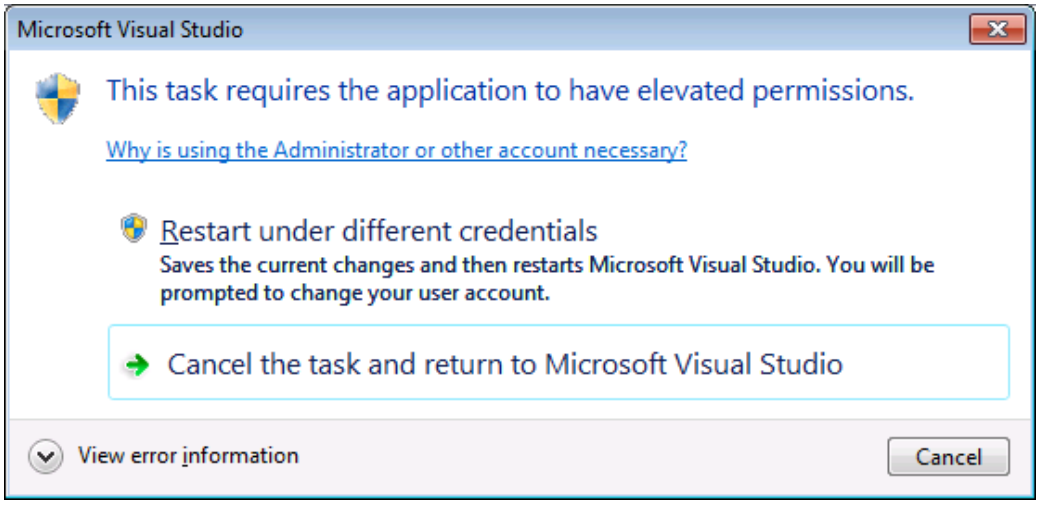
18.2 Adding the ActiveX Controls to the Toolbox

To use the MapForce ActiveX controls in an application developed with Visual Studio, the controls must first be added to the Visual Studio Toolbox, as follows:

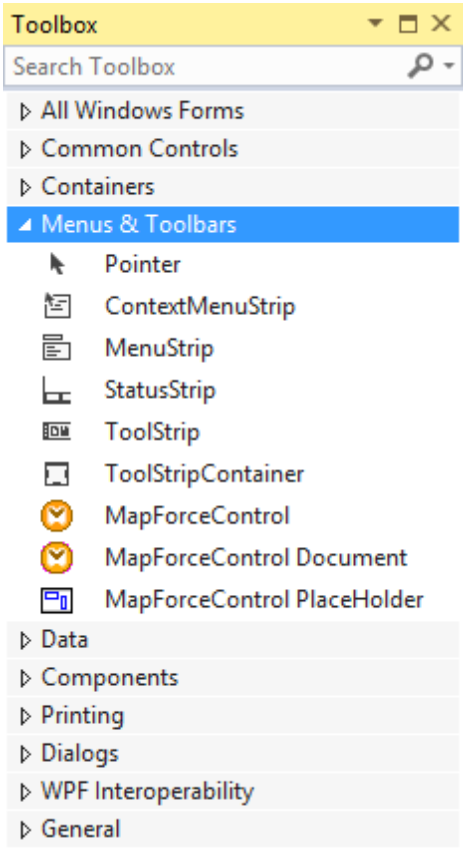
1. On the **Tools** menu of Visual Studio, click **Choose Toolbox Items**.
2. On the **COM Components** tab, select the check boxes next to the MapForceControl, MapForceControl Document, and MapForceControl Placeholder.

In case the controls above are not available, follow the steps below:

1. On the **COM Components** tab, click **Browse**, and select the **MapForceControl.ocx** file from the MapForce installation folder. Remember that the MapForce Integration Package must be installed; otherwise, this file is not available, see Prerequisites.
2. If prompted to restart Visual Studio with elevated permissions, click **Restart under different credentials**.



If the steps above were successful, the MapForce ActiveX controls become available in the Visual Studio Toolbox.



Note: For an application-level integration, only the **MapForceControl** ActiveX control is used (see Integration at Application Level). The **MapForceControl Document** and **MapForceControl Placeholder** controls are used for document-level integration (see Integration at Document Level).

© 2018 Altova GmbH

18.3 Integration at Application Level

Integration at application level allows you to embed the complete interface of MapForce into a window of your application. With this type of integration, you get the whole user interface of MapForce, including all menus, toolbars, the status bar, document windows, and helper windows. Customization of the application's user interface is restricted to what MapForce provides. This includes rearrangement and resizing of helper windows and customization of menus and toolbars.

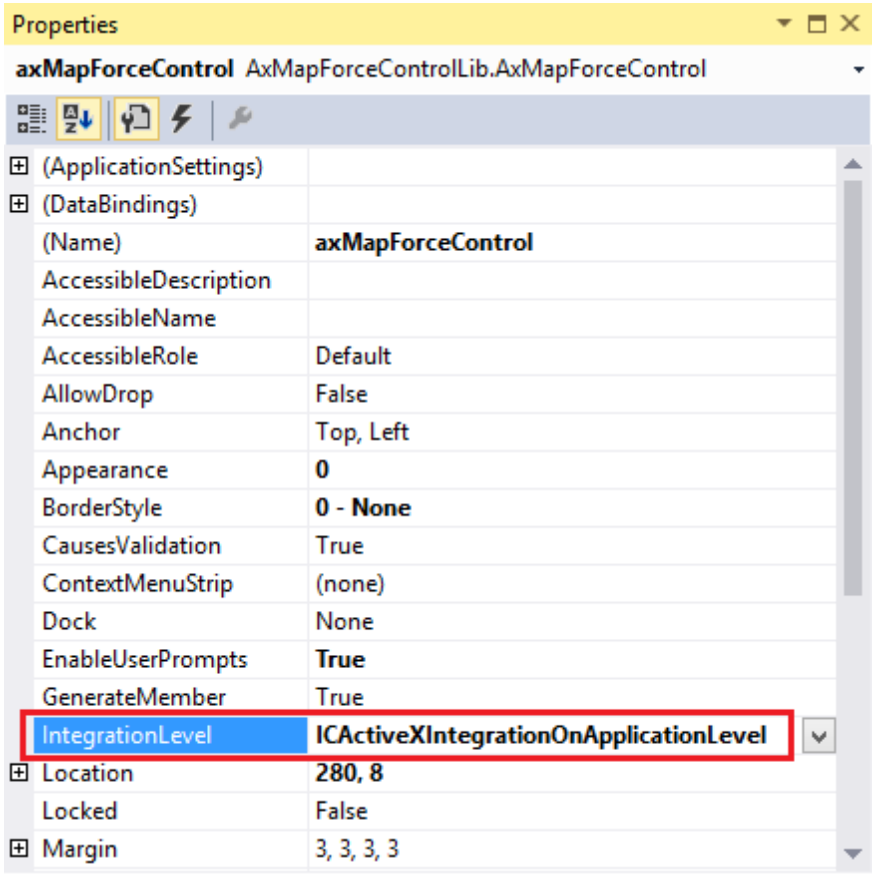
The only ActiveX control you need to integrate is `MapForceControl`. Do not instantiate or access `MapForceControlDocument` or `MapForceControlPlaceHolder` ActiveX controls when integrating at application-level.

If you have any initialization to do or if you want to automate some behaviour of MapForce, use the properties, methods, and events described for `MapForceControl`. Consider using `MapForceControl.Application` for more complex access to MapForce functionality.

For an example that shows how the MapForce application can be embedded in an HTML page, see HTML Integration at Application Level.

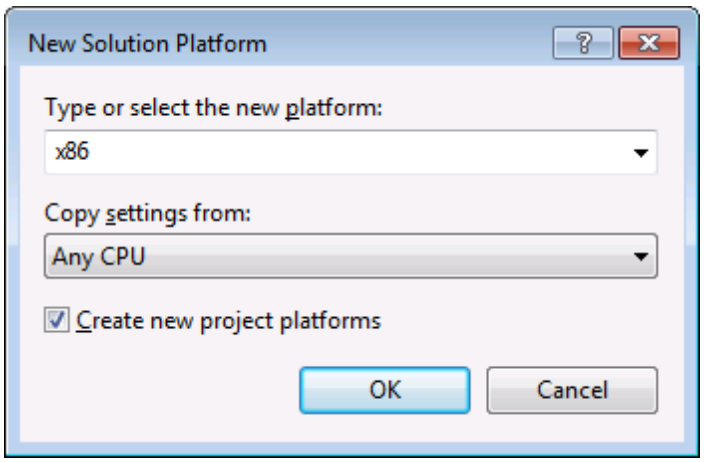
In C# or VB.NET with Visual Studio, the steps to create a basic, one-form application which integrates the MapForce ActiveX controls at application level are as follows:

1. Check that all prerequisites are met (see Prerequisites).
2. Create a new Visual Studio Windows Forms project with a new empty form.
3. If you have not done that already, add the ActiveX controls to the toolbox (see Adding the ActiveX Controls to the Toolbox).
4. Drag the **MapForceControl** from the toolbox onto your new form.
5. Select the **MapForceControl** on the form, and, in the Properties window, set the **IntegrationLevel** property to **ICActiveXIntegrationOnApplicationLevel**.



6. Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

   a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
   b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



You are now ready to build and run the solution in Visual Studio. Remember to build using the

 © 2018 Altova GmbH

configuration that matches your target platform (x86, x64).

18.4 Integration at Document Level

Compared to integration at application level, integration at document level is a more complex, yet more flexible way to embed MapForce functionality into your application by means of ActiveX controls. With this approach, your code can access selectively the following parts of the MapForce user interface:

- Document editing window
- Project window
- Libraries window
- Overview window
- Messages window

As mentioned in Integration at Application Level, for an ActiveX integration at application level, only one control is required, namely the **MapForceControl**. However, for an ActiveX integration at document level, functionality MapForce is provided by the following ActiveX controls:

1. **MapForceControl**
2. **MapForceControl Document**
3. **MapForceControl Placeholder**

These controls are supplied by the **MapForceControl.ocx** file available in the application installation folder of MapForce. When you develop the ActiveX integration with Visual Studio, you will need to add these controls to the Visual Studio toolbox (see Adding the ActiveX Controls to the Toolbox).

The basic steps to integrate the ActiveX controls at document level into your application are as follows:

1. First, instantiate **MapForceControl** in your application. Instantiating this control is mandatory; it enables support for the **MapForceControl Document** and **MapForceControl Placeholder** controls mentioned above. It is important to set the `IntegrationLevel` property to **ICActiveXIntegrationOnDocumentLevel** (or "1"). To hide the control from the user, set its **Visible** property to **False**.

Note: When integrating at document level, do not use the **Open** method of the **MapForceControl**; this might lead to unexpected results. Use the corresponding open methods of **MapForceControl Document** and **MapForceControl PlaceHolder** instead.

2. Create at least one instance of **MapForceControl Document** in your application. This control supplies the document editing window of MapForce to your application and can be instantiated multiple times if necessary.

   Use the method **Open** to load any existing file. To access document-related functionality, use the **Path** and **Save** or methods and properties accessible via the property **Document**.

Note: The control does not support a read-only mode. The value of the property **ReadOnly** is ignored.

3. Optionally, add to your application the **MapForceControl Placeholder** control for each

© 2018 Altova GmbH

additional window (other than the document window) that must be available to your application.

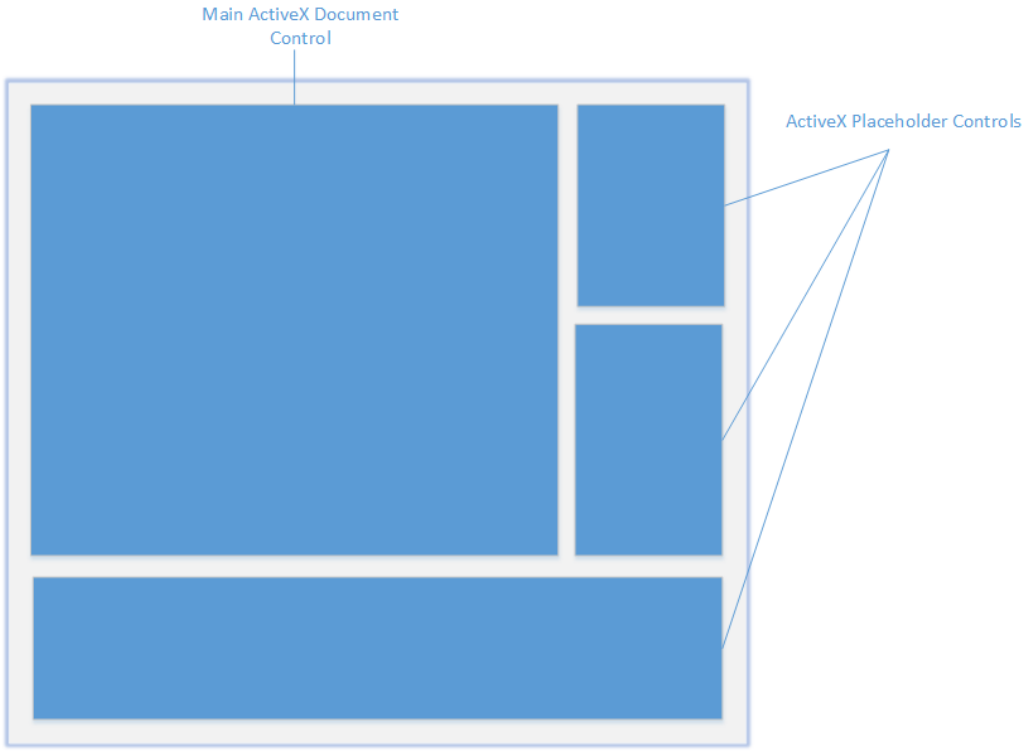
Instances of **MapForceControl PlaceHolder** allow you to selectively embed additional windows of MapForce into your application. The window kind (for example, Project window) is defined by the property **PlaceholderWindowID**. Therefore, to set the window kind, set the property **PlaceholderWindowID**. For valid window identifiers, see MapForceControlPlaceholderWindow.

Note: Use only one **MapForceControl PlaceHolder** for each window identifier.

For placeholder controls that select the MapForce project window, additional methods are available. Use **OpenProject** to load a MapForce project. Use the property Project and the methods and properties from the MapForce automation interface to perform any other project related operations.

For example, in C# or VB.NET with Visual Studio, the steps to create a basic, one-form application which integrates the MapForce ActiveX controls at document level could be similar to those listed below. Note that your application may be more complex if necessary; however, the instructions below are important to understand the minimum requirements for an ActiveX integration at document level.

1. Create a new Visual Studio Windows Forms project with a new empty form.
2. If you have not done that already, add the ActiveX controls to the toolbox (see Adding the ActiveX Controls to the Toolbox).
3. Drag the **MapForceControl** from the toolbox onto your new form.
4. Set the **IntegrationLevel** property of the **MapForceControl** to **ICActiveXIntegrationOnDocumentLevel**, and the **Visible** property to **False**. You can do this either from code or from the **Properties** window.
5. Drag the **MapForceControl Document** from the toolbox onto the form. This control provides the main document window of MapForce to your application, so you may need to resize it to a reasonable size for a document.
6. Optionally, add one or more **MapForceControl Placeholder** controls to the form (one for each additional window type that your application needs, for example, the **Project** window). You will typically want to place such additional placeholder controls either below or to the right or left of the main document control, for example:



7. Set the **PlaceholderWindowID** property of each **MapForceControl Placeholder** control to a valid window identifier. For the list of valid values, see MapForceControlPlaceholderWindow.
8. Add commands to your application (at minimum, you will need to open, save and close documents), as shown below.

Querying MapForce Commands

When you integrate at document level, no MapForce menu or toolbar is available to your application. Instead, you can retrieve the required commands, view their status, and execute them programmatically, as follows:

- To retrieve all available commands, use the `CommandsList` property of the **MapForceControl**.
- To retrieve commands organized according to their menu structure, use the `MainMenu` property.
- To retrieve commands organized by the toolbar in which they appear, use the `Toolbars` property.
- To send commands to MapForce, use the `Exec` method.
- To query if a command is currently enabled or disabled, use the `QueryStatus` method.

This enables you to flexibly integrate MapForce commands into your application's menus and toolbars.

Your installation of MapForce also provides you with command label images used within MapForce. See the folder **<ApplicationFolder>\Examples\ActiveX\Images** of your MapForce installation for icons in GIF format. The file names correspond to the command names as they are

 © 2018 Altova GmbH

listed in the Command Reference section.

General considerations

To automate the behaviour of MapForce, use the properties, methods, and events described for the **MapForceControl**, **MapForceControl Document**, and **MapForceControl Placeholder**.

For more complex access to MapForce functionality, consider using the following properties:

- `MapForceControl.Application`
- `MapForceControlDocument.Document`
- `MapForceControlPlaceHolder.Project`

These properties give you access to the MapForce automation interface (MapForceAPI)

Note: To open a document, always use `MapForceControlDocument.Open` or `MapForceControlDocument.New` on the appropriate document control. To open a project, always use `MapForceControlPlaceHolder.OpenProject` on a placeholder control embedding a MapForce project window.

For examples that show how to instantiate and access the necessary controls in different programming environments, see ActiveX Integration Examples.

18.5 ActiveX Integration Examples

This section contains examples of MapForce document-level integration using different container environments and programming languages. (The HTML section additionally contains examples of integration at application level.) Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX` of your MapForce installation.

18.5.1 C#

A basic ActiveX integration example solution for C# and Visual Studio is available in the folder `<ApplicationFolder>\Examples\ActiveX\C#`. Before you compile the source code and run the sample, make sure that all prerequisites are met (see Running the Sample C# Solution).

18.5.1.1 Running the Sample C# Solution

The sample Visual Studio solution available in the folder **<ApplicationFolder>\Examples\ActiveX\C#** illustrates how to consume the MapForce ActiveX controls. Before attempting to build and run this solution, note the following steps:

Step 1: Check the prerequisites

Visual Studio 2010 or later is required to open the sample solution. For the complete list of prerequisites, see Prerequisites.

Step 2: Copy the sample to a directory where you have write permissions

To avoid running Visual Studio as an Administrator, copy the source code to a directory where you have write permissions, instead of running it from the default location.

Step 3: Check and set all required control properties

The sample application contains one instance of MapForceControlDocument and several instances of MapForceControlPlaceHolder controls. Double-check that the following properties of these controls are set as shown in the table below:

Control name	Property	Property value
axMapForceControl	IntegrationLevel	ICActiveXIntegrationOnDocumentLevel
axMapForceControlLibrary	PlaceholderWindowID	0
axMapForceControlOutput	PlaceholderWindo	2

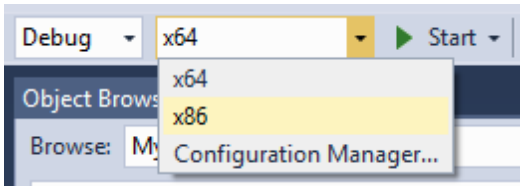
© 2018 Altova GmbH

Control name	Property	Property value
	wID	
axMapForceControlPreview	PlaceholderWindowID	1

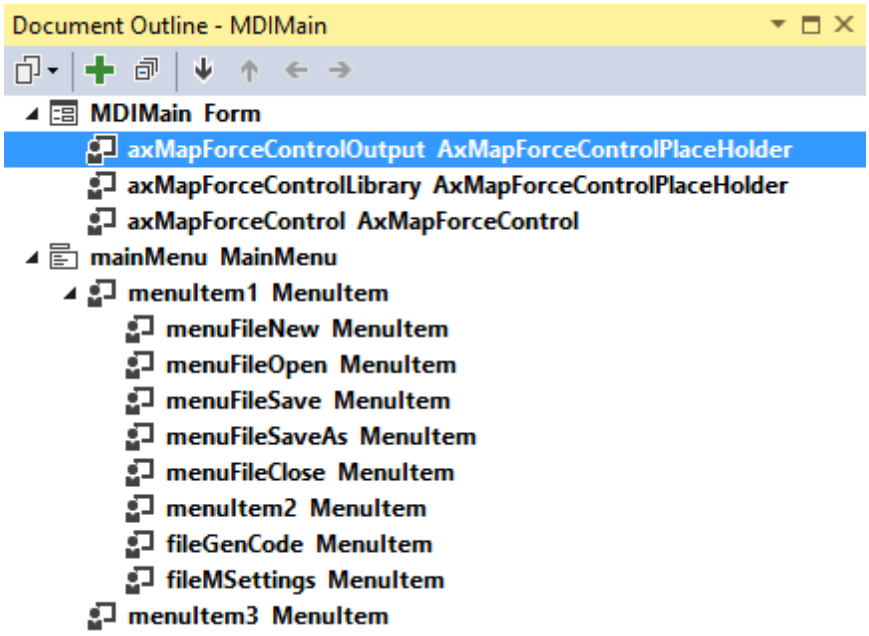
Here is how you can view or set the properties of an ActiveX control:

1. Open the **MDIMain.cs** form in the designer window.

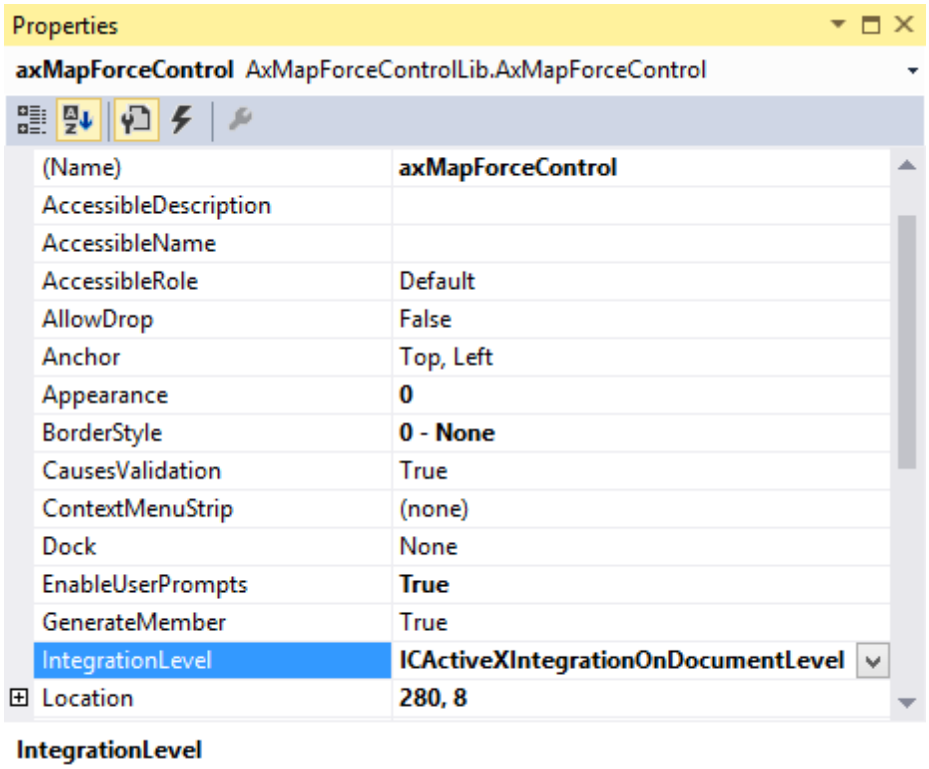
Note: On 64-bit Windows, it may be necessary to change the build configuration of the Visual Studio solution to "x86" **before** opening the designer window. If you need to build the sample as a 64-bit application, see Prerequisites.



2. Open the **Document Outline** window of Visual Studio (On the **View** menu, click **Other Windows | Document Outline**).



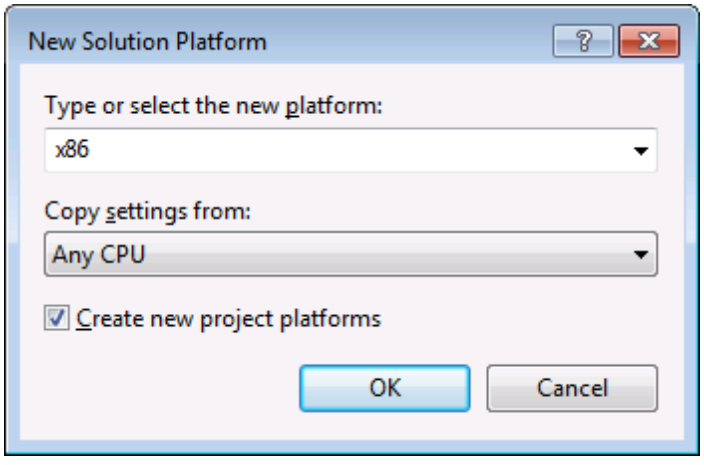
3. Click an ActiveX control in the **Document Outline** window, and edit its required property in the **Properties** window, for example:



Step 4: Set the build platform

- Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

  a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
  b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).

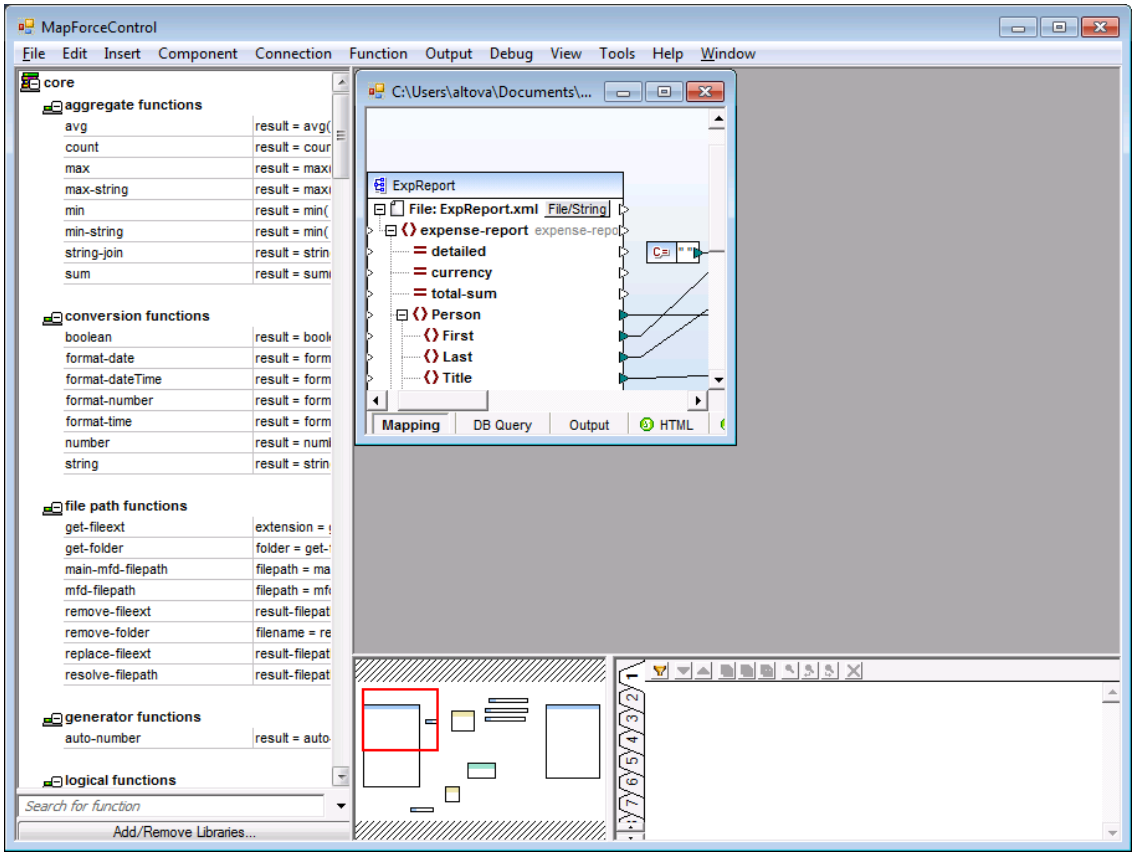


You are now ready to build and run the solution in Visual Studio. Remember to build using the

 © 2018 Altova GmbH

configuration that matches your target platform (x86, x64); otherwise, runtime errors might occur.

On running the sample, the main MDI Frame window is displayed. Use **File | Open** to open a mapping file (for example, `MarketingExpenses.mfd`, which is in the MapForce examples folder). The file is loaded and displayed in a new document child window:



Click to enlarge image

After you load the document, you can execute commands against the active document using the menu. Context menus are also available. You can also load additional documents. Save any modifications using the **File | Save** command.

18.5.1.2 Retrieving Command Information

The MapForceControl gives access to all commands of MapForce through its `CommandsList` , `MainMenu`, and `Toolbars` properties. The example project available in the folder **<ApplicationFolder>\Examples\ActiveX\C#** uses the `MainMenu` property to create the MapForce menu structure dynamically.

The code that gets the menu commands can be found in the `MDIMain` method in `MDIMain.cs` file:

```
public MDIMain()
{
    // ...
```

```
    // Get the MainMenu property of the control and create the menu structure
from it.
    MFLib.MapForceCommand objCommand = this.axMapForceControl.MainMenu;
    InsertMenuStructure(mainMenu, objCommand);
}
```

In the code listing above, `mainMenu` is the existing static menu of the main MDI Frame window. If you open the **MDIMain.cs** form in the Visual Studio Designer, you will notice that this menu contains two menu items: **File** and **Window**.



MDIMain.cs

The method `InsertMenuStructure` takes as parameters the `mainMenu` and the `objCommand` objects (the former is the existing static menu, while the latter contains the full menu structure retrieved from the MapForce ActiveX control). The retrieved MapForce menu structure is then merged into the existing static menu. Note that the menus **File**, **Project**, and **Window** are not added dynamically. This is intentional, because these menus deal with actively open documents, and they would require code which is beyond the scope of this example. The basic file management commands (create, open, save, bring into focus) are handled by the existing static menus **File** and **Window**. All other menus are inserted dynamically based on the information taken from the `MainMenu` property of the ActiveX control. The new menus are inserted after "File" but before "Window", i.e. starting at menu index 1.

The method `InsertMenuStructure` iterates through all top-level menus found in `MapForceCommand` object and adds a new menu item for each. Since each top-level menu has its own child menu items, a call to the method `InsertMenuCommand` takes place for each encountered child menu item. Furthermore, since each child menu item can have its own children menu items, and so on, the `InsertMenuCommand` method recurses into itself until no more child menu items exist.

The commands added dynamically are instances of the class `CustomMenuItem`, which is defined in `CustomMenuItem.cs`. This class is derived from `System.Windows.Forms.MenuItem` class and has an additional member to store the MapForce command ID.

```
public class CustomMenuItem : System.Windows.Forms.MenuItem
{
	public int m_MapForceCmdID;
}
```

All dynamically added commands (except those that are containers for other commands) get the same event handler `AltovaMenuItem_Click` which does the processing of the command:

© 2018 Altova GmbH

```
private void AltovaMenuItem_Click(object sender, EventArgs e)
{
  if(sender.GetType() ==
System.Type.GetType("MapForceApplication.CustomMenuItem"))
  {
    CustomMenuItem   customItem = (CustomMenuItem)sender;
    ProcessCommand(customItem.m_MapForceCmdID);
  }
}
```

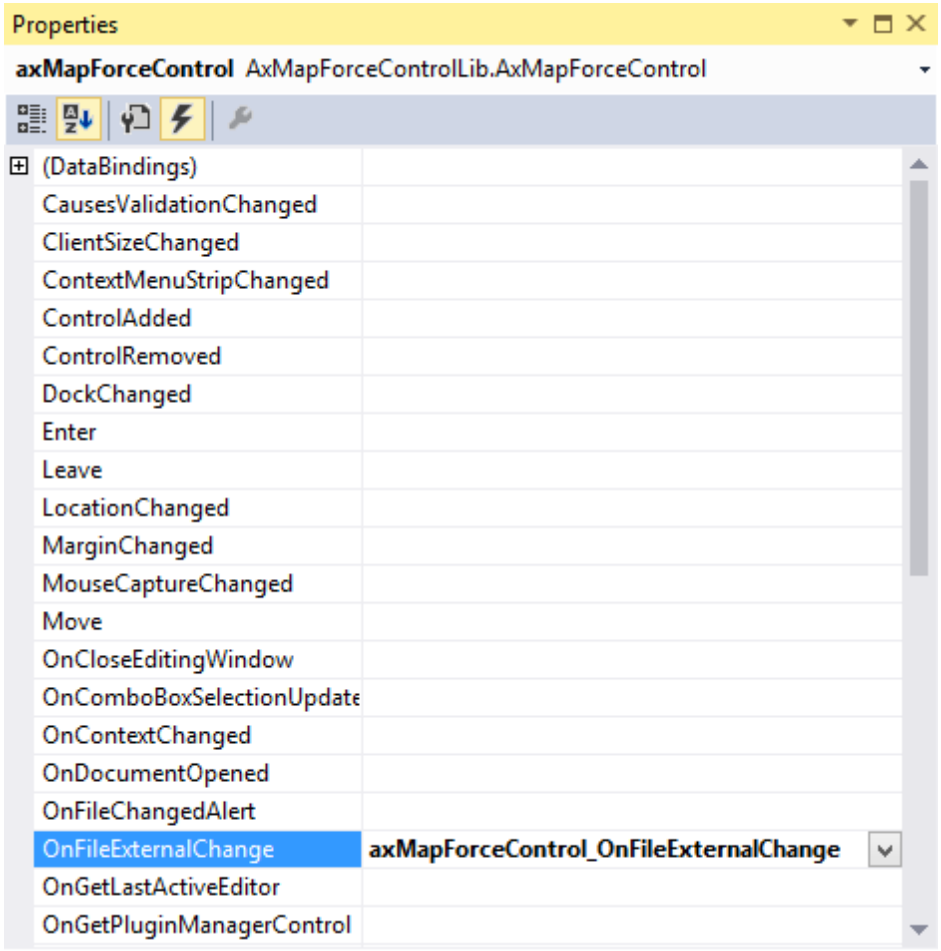
If the command is a container for other commands (that is, if it has child commands), it gets the event handler `AltovaSubMenu_Popup`. This handler queries the status of each child command and enables or disables it as required. This ensures that each command is enabled only when that is meaningful (for example, the **File | Save** menu item should be disabled if there is no active document open).

The method `ProcessCommand` delegates the execution either to the MapForceControl itself or to any active MapForce document loaded in a `MapForceControlDocument` control. This is necessary because the MapForceControl has no way to know which document is currently active in the hosting application.

```
private void ProcessCommand(int nID)
{
   MapForceDoc docMapForce = GetCurrentMapForceDoc();

   if(docMapForce != null)
      docMapForce.axMapForceControlDoc.Exec(nID);
   else
      axMapForceControl.Exec(nID);
}
```

18.5.1.3 Handling Events

Because all events in the MapForce library are based on connection points, you can use the C# delegate mechanism to provide the custom event handlers. You will always find a complete list of events on the property page of each control of the MapForce library. The image below shows the events of the main MapForceControl:



As you can see, the example project only overrides the `OnFileExternalChange` event. The creation of the C# delegate is done for you by the C# Framework. All you need to do is fill in the empty event handler.

For example, the handler implementation shown below turns off any file reloading and displays a message box to inform the user that a file loaded by the MapForceControl has been changed from outside:

```
private void axMapForceControl_OnFileExternalChange(object sender,
AxMapForceControlLib._DMapForceControlEvents_OnFileExternalChangeEvent e)
{
    MessageBox.Show("Attention: The file " + e.strPath + " has been changed
from outside\nbut reloading is turned off in the sample application!");

    // This turns off any file reloading:
    e.varRet = false;
}
```

 © 2018 Altova GmbH

18.5.2 HTML

The code listings in this section show how to integrate the MapForceControl at application level and document level. Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX\HTML` of your MapForce installation.

Note: ActiveX controls in an HTML page are supported only by Internet Explorer when it runs as a 32-bit application. When Internet Explorer 10 or 11 runs in 64-bit mode, it does not load ActiveX controls. The default browser security settings will normally block ActiveX, so you will need to explicitly allow blocked content to run on the page when prompted by Internet Explorer.

18.5.2.1 HTML Integration at Application Level

This example shows a simple integration of the MapForce control at application-level into a HTML page. The integration is described in the following sections:

- Instantiate a MapForceControl in HTML code.
- Implement buttons to load documents and automate code-generation tasks.
- Define actions for some application events.

The code for this example is available at the following location in your MapForce installation: `<ApplicationFolder>\Examples\ActiveX\HTML\MapForceActiveX_ApplicationLevel.htm`.

18.5.2.1.1 Instantiate the Control

The HTML `Object` tag is used to create an instance of the MapForceControl. The `Classid` is that of MapForceControl. Width and height specify the window size. No additional parameters are necessary, since application-level is the default.

```
<OBJECT id="objMapForceControl"
        Classid="clsid:A38637E9-5759-4456-A167-F01160CC22C1"
        width="800"
        height="500"
        VIEWASTEXT>
</OBJECT>
```

18.5.2.1.2 Add Button to Open Default Document

As a simple example of how to automate some tasks, we add a button to the page:

```
<input type="button" value="Open Marketing Expenses"
onclick="BtnOpenMEFile()">
```

When clicked, a predefined document will be opened in the MapForceControl. The `MakeAbsolutePath` method creates an absolute path using the location of the script as a base

path.

```
function BtnOpenMEFile()
{
   var strPath = MakeAbsolutePath("MarketingExpenses.mfd");
   var objDoc = objMapForceControl.Open(strPath );

   if (objDoc == null)
      alert("Unable to locate MarketingExpenses.mfd at: " +
objMapForceControl.BaseHref);
}
```

18.5.2.1.3 Add Buttons for Code Generation

Although code-generation for the active document is available via menus, we want to have buttons that will generate code without asking the user for the location of the output. The method is similar to that used in the previous section.

First come the buttons:

```
<input type="button" value="Generate XSLT" onclick="BtnGenerate( 0 )">
<input type="button" value="Generate Java" onclick="BtnGenerate( 1 )">
<input type="button" value="Generate C++" onclick="BtnGenerate( 2 )">
<input type="button" value="Generate C#" onclick="BtnGenerate( 3 )">
```

Then we provide the script that will generate the code into sub-folders of the currently defined default output folders.

```
// ------------------------------------------------------------------------
// generate code for active document into language-specific subfolders of
// the current default output dicrectory. No user intercation necessary.
function BtnGenerate(languageID)
{
   // get top-level object of automation interface
   var objApp = objMapForceControl.Application;

   // get the active document
   var objDocument = objApp.ActiveDocument;

   // retrive object to set the generation output path
   var objOptions = objApp.Options;

   if ( objDocument == null )
      alert( "no active document found" );
   else
   {
      objOptions.XSLTDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory = GetDefaultOutputDirectory();

      if (languageID == 0)
      {
```


© 2018 Altova GmbH

```
         objOptions.XSLTDefaultOutputDirectory =
objOptions.XSLTDefaultOutputDirectory + "\\XSLTGen";
         objDocument .GenerateXSLT();
      }
      else if (languageID == 1)
      {
         objOptions.CodeDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory + "/JavaCode";
         objDocument .GenerateJavaCode();
      }
      else if (languageID == 2)
      {
         objOptions.CodeDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory + "/CPPCode";
         objDocument .GenerateCppCode();
      }
      else if (languageID == 3)
      {
         objOptions.CodeDefaultOutputDirectory =
objOptions.CodeDefaultOutputDirectory + "/CSharpCode";
         objDocument .GenerateCHashCode();
      }
   }
}
```

18.5.2.1.4 Connect to Custom Events

The example implements two event callbacks for MapForceControl custom events to show the principle:

```
<!-- ------------------------------------------------------------ -->
<!--  custom event 'OnDocumentOpened" of MapForceControl object  -->
<SCRIPT FOR="objMapForceControl" event="OnDocumentOpened( objDocument )"
LANGUAGE="javascript">
   // alert("Document '" + objDocument.Name + "' opend!");
</SCRIPT>

<!-- ------------------------------------------------------------ -->
<!--  custom event 'OnDocumentClosed" of MapForceControl object  -->
<SCRIPT FOR="objMapForceControl" event="OnDocumentClosed( objDocument )"
LANGUAGE="javascript">
   // alert("Document '" + objDocument.Name + "' closed!");
</SCRIPT>
```

18.5.2.2 HTML Integration at Document Level

This example shows an integration of the MapForce control at document-level into a HTML page. The following topics are covered:

- Instantiate a MapForceControl ActiveX control object in HTML code
- Instantiate a MapForceControlDocument ActiveX control to allow editing a MapForce file
- Instantiate one MapForceControlPlaceHolder for a MapForceControl project window
- Instantiate one MapForceControlPlaceHolder to alternatively host one of the MapForce helper windows
- Create a simple custom toolbar for some heavy-used MapForce commands
- Add some more buttons that use the COM automation interface of MapForce
- Use event handlers to update command buttons

This example is available in its entirety in the file `MapForceActiveX_ApplicationLevel.htm` within the `<ApplicationFolder>\Examples\ActiveX\HTML\` folder of your MapForce installation.

18.5.2.2.1 Instantiate the MapForceControl

MapForceControlThe HTML `OBJECT` tag is used to create an instance of the MapForceControl. The Classid is that of MapForceControl. Width and height are set to 0 since we use this control as manager control without use for its user interface. The integration level is specified as a parameter within the `OBJECT` tag.

```
<object id="objMapForceX" classid="clsid:A38637E9-5759-4456-A167-F01160CC22C1"
 width="0" height="0" VIEWASTEXT>
   <param name="IntegrationLevel" value="1">
</object>
```

18.5.2.2.2 Create Editor Window

The HTML `OBJECT` tag is used to embed an editing window. The additional custom parameter specifies that the control is to be initialized with a new empty document.

```
<object id="objDoc1" classid="clsid:DFBB0871-DAFE-4502-BB66-08CEB7DF5255"
width="600" height="500" VIEWASTEXT>
   <param name="NewDocument">
</object>
```

18.5.2.2.3 Create Project Window

The HTML `OBJECT` tag is used to create a MapForceControlPlaceHolder window. The parameter defines the placeholder to show the MapForce project window.

```
<!-- ------------------------------------------------- -->
<!-- create project window placeholder control.        -->
<!-- initialize it with a project.                     -->
<object id="objProjectWindow" classid="clsid:FDEC3B04-05F2-427d-988C-
F03A85DE53C2" width="200" height="200" VIEWASTEXT>
   <param name="PlaceholderWindowID" value="3">
```


© 2018 Altova GmbH

```
</object>
```

18.5.2.2.4 Create Placeholder for Helper Windows

The **MapForceControlPlaceHolder** control is required to host an application helper window, see also Integration at Document Level. In the code listing below, the HTML `object` tag is used to instantiate a control that will host the **Libraries** window by default (`PlaceholderWindowID` is 0).

```
<!-- create helper window placeholder control.          -->
<!-- the editor with focus will automatically direct its -->
<!-- output to the appropriate helper window.           -->
<object id="objPlaceholderWindow" Classid="clsid:FDEC3B04-05F2-427d-988C-
F03A85DE53C2" width="200" height="200" VIEWASTEXT>
   <param name="PlaceholderWindowID" value="0">
   <param name="FileName" value="">
</object>
```

The example HTML page includes a few buttons which call the `BtnHelperWindow` method when clicked. The `BtnHelperWindow` method reassigns the `PlaceholderWindowID` of the control, and thus cause the ActiveX object to display a different helper window.

```
// specify which of the helper windows shall be shown in the placeholder
control.
function BtnHelperWindow(i_ePlaceholderWindowID)
{
   objPlaceholderWindow.PlaceholderWindowID = i_ePlaceholderWindowID;
}
```

For the list of possible values of `PlaceholderWindowID`, see `MapForceControlPlaceholderWindow`.

18.5.2.2.5 Create a Custom Toolbar

The example HTML page also includes a custom toolbar (intended as a replica of the MapForce menu). The custom toolbar consists of buttons with images of MapForce commands, for example:

```
<button id="btnInsertXML" title="Insert XML Schema/File"
onclick="BtnDoCommand(32393)">
   <img src="..\Images\ID_INSERT_XSD.gif" width="16" height="16" />
</button>
<button id="btnInsertDB" title="Insert Database"
onclick="BtnDoCommand(32389)">
   <img src="..\Images\ID_INSERT_DATABASE.gif" width="16" height="16" />
</button>
<button id="btnInsertEDI" title="Insert EDI" onclick="BtnDoCommand(32390)">
   <img src="..\Images\ID_INSERT_EDI.gif" width="16" height="16" />
</button>
<button id="btnInsertText" title="Insert Text file"
```

```
onclick="BtnDoCommand(32392)">
   <img src="..\Images\ID_INSERT_TXT.gif" width="16" height="16" />
</button>
```

The names of button images correspond to the command ID numbers, see Command Reference. On clicking the button, the corresponding command ID is sent to the main control and executed:

```
// perform any command specified by cmdID.
// command routing includes application, active document and view.
function BtnDoCommand(cmdID)
{
   objMapForceX.Exec( cmdID );
   msgtext.innerText = "Command " + cmdID + " performed.";
}
```

18.5.2.2.6 Create More Buttons

In the example, we add some more buttons to show some automation code.

```
<!-- add some buttons associated with above editor.              -->
<!-- generation of code is now implemented using the MapForce automation  -->
<!-- interface to select a target folder without prompting the user.      -->
<p>
   <input type="button" value="New File" onclick="BtnNewFile(objDoc1)">
   <input type="button" value="Save File" onclick="BtnSaveFile(objDoc1)">
   <input type="text" title="Path" id="strPath" width="150">
   <input type="button" value="Open MarketingExpenses"
onclick="BtnOpenMEFile(objDoc1)">
   <input type="button" value="Open MapForce Sample Project"
onclick="BtnOpenProjectFile(objDoc1)">
</p>
```

The corresponding JavaScript looks like this:

```
// ------------------------------------------------------------------
// open a new empty document in the specified document control window.
function BtnNewFile(objDocCtrl)
{
   objDocCtrl.OpenDocument("");
   objDocCtrl.setActive();
}

// ------------------------------------------------------------------
// Saves the current file in the specified document control window.
function BtnSaveFile(objDocCtrl)
{
   if(objDocCtrl.Path.length > 0)
objDocCtrl.SaveDocument();
   else
   {
if(strPath.value.length > 0)
```


© 2018 Altova GmbH

```
{
   objDocCtrl.Path = strPath.value;
   objDocCtrl.SaveDocument();
}
else
{
   alert("Please set path for the document first!");
   strPath.focus();
}
   }

   objDocCtrl.setActive();
}

// ------------------------------------------------------
// open a document in the specified document control window.
function BtnOpenMEFile(objDocCtrl)
{
   // do not use MapForceX.Application.OpenDocument(...) to open a document,
   // since then MapForceControl wouldn't know a control window to show
   // the document in. Instead:

   var strPath = MakeAbsolutePath("MarketingExpenses.mfd");
   var objDoc = objDocCtrl.OpenDocument(strPath);

   if (objDoc != null)
   {
objDocCtrl.setActive();
msgtext.innerText = "Opened mapping: " + strPath;
   }
   else
alert("Unable to open " + strPath);
}
```

18.5.2.2.7 Create Event Handler to Update Button Status

Availability of a command may vary with every mouse click or keystroke. The custom event `OnUpdateCmdUI` of MapForceControl gives us an opportunity to update the enabled/disabled state of buttons associated with MapForce commands. The method `MapForceControl.QueryStatus` is used to query whether a command is enabled or not.

```
<!--  custom event 'OnUpdateCmdUI" of MapForceControl object  -->
function objMapForceX::OnUpdateCmdUI()
{
   if ( document.readyState == "complete" )// 'complete'
   {
      // update status of buttons
      // set activity status of simulated toolbar
      GenerateXSLT.disabled =  ! (objDoc1.QueryStatus(32360) & 0x02);   // not
enabled
      GenerateJava.disabled =  ! (objDoc1.QueryStatus(32358) & 0x02);   // not
```

```
enabled
      GenerateCpp.disabled =   ! (objDoc1.QueryStatus(32356) & 0x02);   // not
enabled
      GenerateCSharp.disabled = ! (objDoc1.QueryStatus(32357) & 0x02);   //
not enabled

      btnInsertXML.disabled = ! (objDoc1.QueryStatus(32393) & 0x02);
      btnInsertDB.disabled = ! (objDoc1.QueryStatus(32389) & 0x02);
      btnInsertEDI.disabled = ! (objDoc1.QueryStatus(32390) & 0x02);
      btnInsertText.disabled = ! (objDoc1.QueryStatus(32392) & 0x02);

      btnInsertConstant.disabled = ! (objDoc1.QueryStatus(32388) & 0x02);
      btnInsertFilter.disabled = ! (objDoc1.QueryStatus(32391) & 0x02);
      btnInsertIFELSE.disabled = ! (objDoc1.QueryStatus(32394) & 0x02);
      btnInsertException.disabled = ! (objDoc1.QueryStatus(32311) & 0x02);

      btnFuncUserDef.disabled = ! (objDoc1.QueryStatus(32380) & 0x02);
      btnFuncUserDefSel.disabled = ! (objDoc1.QueryStatus(32381) & 0x02);
      btnFuncSettings.disabled = ! (objDoc1.QueryStatus(32387) & 0x02);
      btnInsertInput.disabled = ! (objDoc1.QueryStatus(32383) & 0x02);

      btnGenXSLT.disabled = ! (objDoc1.QueryStatus(32360) & 0x02);
      btnGenXSLT2.disabled = ! (objDoc1.QueryStatus(32361) & 0x02);
      btnGenXQuery.disabled = ! (objDoc1.QueryStatus(32359) & 0x02);
      btnGenCPP.disabled = ! (objDoc1.QueryStatus(32356) & 0x02);
      btnGenCSharp.disabled = ! (objDoc1.QueryStatus(32357) & 0x02);
      btnGenJava.disabled = ! (objDoc1.QueryStatus(32358) & 0x02);
   }
}
```

18.5.3 Java

MapForce ActiveX components can be accessed from Java code. Java integration is provided by the libraries listed below. These libraries are available in the folder `<ApplicationFolder>\Examples\JavaAPI` of your MapForce installation, after you have installed both MapForce and the MapForce Integration Package (see also Prerequisites).

- `AltovaAutomation.dll`: a JNI wrapper for Altova automation servers (in case of the 32-bit installation of MapForce)
- `AltovaAutomation_x64.dll`: a JNI wrapper for Altova automation servers (in case of the 64-bit installation of MapForce)
- `AltovaAutomation.jar`: Java classes to access Altova automation servers
- `MapForceActiveX.jar`: Java classes that wrap the MapForce ActiveX interface
- `MapForceActiveX_JavaDoc.zip`: a Javadoc file containing help documentation for the Java interface

Note: In order to use the Java ActiveX integration, the .dll and .jar files must be included in the Java class search path.

Example Java project

An example Java project is supplied with your product installation. You can test the Java project and modify and use it as you like. For more details, see Example Java Project.

© 2018 Altova GmbH

Rules for mapping the ActiveX Control names to Java

For the documentation of ActiveX controls, see Object Reference. Note that the object naming conventions are slightly different in Java compared to other languages. Namely, the rules for mapping between the ActiveX controls and the Java wrapper are as follows:

- ***Classes and class names***
  For every component of the MapForce ActiveX interface a Java class exists with the name of the component.

- ***Method names***
  Method names on the Java interface are the same as used on the COM interfaces but start with a small letter to conform to Java naming conventions. To access COM properties, Java methods that prefix the property name with get and set can be used. If a property does not support write-access, no setter method is available. Example: For the `IntegrationLevel` property of the MapForceControl, the Java methods `getIntegrationLevel` and `setIntegrationLevel` are available.

- ***Enumerations***
  For every enumeration defined in the ActiveX interface, a Java enumeration is defined with the same name and values.

- ***Events and event handlers***
  For every interface in the automation interface that supports events, a Java interface with the same name plus 'Event' is available. To simplify the overloading of single events, a Java class with default implementations for all events is provided. The name of this Java class is the name of the event interface plus '`DefaultHandler`'. For example:
  `MapForceControl`: Java class to access the application
  `MapForceControlEvents`: Events interface for the MapForceControl
  `MapForceControlEventsDefaultHandler`: Default handler for `MapForceControlEvents`

Exceptions to mapping rules

There are some exceptions to the rules listed above. These are listed below:

Interface	Changes in Java class
`MapForceControlDocument`, method `New`	Renamed to `newDocument`
`MapForceControlDocument`, method `OpenDocument`	Removed. Use the `Open` method
`MapForceControlDocument`, method `NewDocument`	Removed. Use the `newDocument` method
`MapForceControlDocument`, method `SaveDocument`	Removed. Use the `Save` method

This section

This section shows how some basic MapForce ActiveX functionality can be accessed from Java code. It is organized into the following sub-sections:

- Example Java Project

- Creating the ActiveX Controls
- Loading Data in the Controls
- Basic Event Handling
- Menus
- UI Update Event Handling
- Creating a MapForce Mapping Table

18.5.3.1 Example Java Project

The MapForce installation package contains an example Java project, located in the ActiveX Examples folder of the application folder: `<ApplicationFolder>\Examples\ActiveX\Java\`.

The Java example shows how to integrate the MapForceControl in a common desktop application created with Java. You can test it directly from the command line using the batch file `BuildAndRun.bat`, or you can compile and run the example project from within Eclipse. See below for instructions on how to use these procedures.

File list

The Java examples folder contains all the files required to run the example project. These files are listed below:

`.classpath`	Eclipse project helper file
`.project`	Eclipse project file
`AltovaAutomation.dll`	Java-COM bridge: DLL part (for the 32-bit installation)
`AltovaAutomation_x64.dll`	Java-COM bridge: DLL part (for the 64-bit installation)
`AltovaAutomation.jar`	Java-COM bridge: Java library part
`BuildAndRun.bat`	Batch file to compile and run example code from the command line prompt. Expects folder where Java Virtual Machine resides as parameter.
`MapForceActiveX.jar`	Java classes of the MapForce ActiveX control
`MapForceActiveX_JavaDoc.zip`	Javadoc file containing help documentation for the Java API
`MapForceContainer.java`	Java example source code
`MapForceContainerEventHandler.java`	Java example source code
`MapForceTable.java`	Java example source code

What the example does

The example places one MapForce document editor window, the MapForce project window, the MapForce library window and the MapForce validation window in an AWT frame window. It reads out the main menu defined for MapForce and creates an AWT menu with the same structure. You

© 2018 Altova GmbH

can use this menu or the project window to open and work with files in the document editor.

You can modify the example in any way you like.

The following specific features are described in code listings:

- Creating the ActiveX Controls: Starts MapForce, which is registered as an automation server, or activates MapForce if it is already running.
- Loading Data in the Controls: Locates one of the example documents installed with MapForce and opens it.
- Basic Event Handling: Changes the view of all open documents to Text View. The code also shows how to iterate through open documents.
- Menus: Validates the active document and shows the result in a message box. The code shows how to use output parameters.
- UI Update Event Handling: Shows how to handle MapForce events.
- Creating a MapForce Mapping Table: Shows how to create a MapForce mapping table and prepare it for modal activation.

Updating the path to the Examples folder

Before running the provided sample, you may need to edit the **MapForceContainer.java** file. Namely, check that the following path refers to the actual folder where the MapForce example files are stored on your operating system:

```
// Locate samples installed with the product.
final String strExamplesFolder = System.getenv( "USERPROFILE" ) + "\\Documents
\\Altova\\MapForce2018\\MapForceExamples\\";
```

Running the example from the command line

To run the example from the command line:

1. Check that all prerequisites are met (see Prerequisites).
2. Open a command prompt window, change the current directory to the sample Java project folder, and type:

   ```
   buildAndRun.bat "<Path-to-the-Java-bin-folder>"
   ```

3. Press **Enter**.

The Java source in `MapForceContainer.java` will be compiled and then executed.

Compiling and running the example in Eclipse

To import the sample Java project into Eclipse:

1. Check that all prerequisites are met (see Prerequisites).
2. On the **File** menu, click **Import**.
3. Select **Existing Projects into Workspace**, and browse for the Eclipse project file located at `<ApplicationFolder>\Examples\ActiveX\Java\`. Since you may not have write-access in this folder, it is recommended to select the **Copy projects into**

workspace check box on the Import dialog box.

To run the example application, right-click the project in Package Explorer and select the command **Run as | Java Application**.

Help for Java API classes is available through comments in code as well as the Javadoc view of Eclipse. To enable the Javadoc view in Eclipse, select the menu command **Window | Show View | JavaDoc**.

18.5.3.2 Creating the ActiveX Controls

The code listing below show how ActiveX controls can be created. The constructors will create the Java wrapper objects. Adding these Canvas-derived objects to a panel or to a frame will trigger the creation of the wrapped ActiveX object.

```
01   /**
02    * MapForce manager control - always needed
03    */
04   public static MapForceControl         mapForceControl = null;
05
06   /**
07    * MapForceDocument editing control
08    */
09   public static MapForceControlDocument     mapForceDocument = null;
10
11   /**
12    * Tool windows - MapForce place-holder controls
13    */
14   private static MapForceControlPlaceHolder   mapForceProjectToolWindow =
null;
15   private static MapForceControlPlaceHolder   mapForceValidationToolWindow =
null;
16   private static MapForceControlPlaceHolder   mapForceLibraryToolWindow =
null;
17
18   // Create the MapForce ActiveX control; the parameter determines that we
want
     // to place document controls and place-holder controls individually.
19   // It gives us full control over the menu, as well.
20      mapForceControl = new MapForceControl(
        ICActiveXIntegrationLevel.ICActiveXIntegrationOnDocumentLevel.getValue()
, false );
21
22      mapForceDocument = new MapForceControlDocument();
23      frame.add( mapForceDocument, BorderLayout.CENTER );
24
25
26   // Create a project window and open the sample project in it
27      mapForceProjectToolWindow = new MapForceControlPlaceHolder(
        MapForceControlPlaceholderWindow.MapForceXProjectWindow.getValue(),
        strExamplesFolder + "MapForceExamples.mfp" ) ;
28      mapForceProjectToolWindow.setPreferredSize( new Dimension( 200, 200 ) );
```


© 2018 Altova GmbH

18.5.3.3 Loading Data in the Controls

The code listing below show how data can be loaded in the ActiveX controls.

```
1    // Locate samples installed with the product.
2       final String strExamplesFolder = System.getenv( "USERPROFILE" ) +
        "\\Documents\\Altova\\MapForce2018\\MapForceExamples\\";
3       mapForceProjectToolWindow = new
MapForceControlPlaceHolder( MapForceControlPlaceholderWindow.MapForceXProjectWin
dow.getValue(), strExamplesFolder + "MapForceExamples.mfp" ) ;
```

18.5.3.4 Basic Event Handling

The code listing below shows how basic events can be handled. When calling the MapForceControl's open method, or when trying to open a file via the menu or Project tree, the onOpenedOrFocused event is sent to the attached event handler. The basic handling for this event is opening the file by calling the MapForceDocumentControl's open method.

```
01       // Open the Marketing file when button is pressed
02       btnMarkExp.addActionListener( new ActionListener() {
03         public void actionPerformed(ActionEvent e) {
04           try {
05             // Instruct the Document control to open the file - avoid calling
the open method of MapForceControl (see help)
06             mapForceDocument.open( strExamplesFolder +
"MarketingExpenses.mfd" );
07             mapForceDocument.requestFocusInWindow();
08           } catch (AutomationException e1) {
09             e1.printStackTrace();
10           }
11         }
12       } );
13       public void onOpenedOrFocused( String i_strFileName, boolean
i_bOpenWithThisControl, boolean i_bFileAlreadyOpened ) throws
AutomationException
14   {
15     // Handle the New/Open events coming from the Project tree or from the
menus
16     if ( !i_bFileAlreadyOpened )
17     {
18       // This is basically an SDI interface, so open the file in the already
existing document control
19       try {
20         MapForceContainer.mapForceDocument.open( i_strFileName );
21         MapForceContainer.mapForceDocument.requestFocusInWindow();
22       } catch (Exception e) {
23         e.printStackTrace();
24       }
25     }
26   }
```

18.5.3.5 Menus

The code listing below shows how menu items can be created. Each `MapForceCommand` object gets a corresponding `MenuItem` object, with the `ActionCommand` set to the ID of the command. The actions generated by all menu items are handled by the same function, which can perform specific handlings (like reinterpreting the closing mechanism) or can delegate the execution to the `MapForceControl` object by calling its `exec` method. The `menuMap` object that is filled during menu creation is used later (see section UI Update Event Handling).

```
01
02       // Load the file menu when the button is pressed
03       btnMenu.addActionListener( new ActionListener() {
04         public void actionPerformed(ActionEvent e) {
05           try {
06             // Create the menubar that will be attached to the frame
07             MenuBar mb = new MenuBar();
08             // Load the main menu's first item - the File menu
09             MapForceCommand xmlSpyMenu =
mapForceControl.getMainMenu().getSubCommands().getItem( 0 );
10             // Create Java menu items from the Commands objects
11             Menu fileMenu = new Menu();
12             handlerObject.fillMenu( fileMenu, xmlSpyMenu.getSubCommands() );
13             fileMenu.setLabel( xmlSpyMenu.getLabel().replace( "&", "" ) );
14             mb.add( fileMenu );
15             frame.setMenuBar( mb );
16             frame.validate();
17           } catch (AutomationException e1) {
18             e1.printStackTrace();
19           }
20           // Disable the button when the action has been performed
21           ((AbstractButton) e.getSource()).setEnabled( false );
22         }
23       } ) ;
24   /**
25    * Populates a menu with the commands and submenus contained in an
MapForceCommands object
26    */
27   public void fillMenu(Menu newMenu, MapForceCommands mapForceMenu) throws
AutomationException
28   {
29     // For each command/submenu in the mapForceMenu
30     for ( int i = 0 ; i < mapForceMenu.getCount() ; ++i  )
31     {
32       MapForceCommand mapForceCommand = mapForceMenu.getItem( i );
33       if ( mapForceCommand.getIsSeparator() )
34         newMenu.addSeparator();
35       else
36       {
37         MapForceCommands subCommands = mapForceCommand.getSubCommands();
38         // Is it a command (leaf), or a submenu?
39         if ( subCommands.isNull() || subCommands.getCount() == 0 )
40         {
41           // Command -> add it to the menu, set its ActionCommand to its ID
and store it in the menuMap
42           MenuItem mi = new
MenuItem( mapForceCommand.getLabel().replace( "&", "" ) );
```


© 2018 Altova GmbH

```
43            mi.setActionCommand( "" + mapForceCommand.getID() );
44            mi.addActionListener( this );
45            newMenu.add( mi );
46            menuMap.put( mapForceCommand.getID(), mi );
47          }
48          else
49          {
50            // Submenu -> create submenu and repeat recursively
51            Menu newSubMenu = new Menu();
52            fillMenu( newSubMenu, subCommands );
53            newSubMenu.setLabel( mapForceCommand.getLabel().replace( "&",
"" ) );
54            newMenu.add( newSubMenu );
55          }
56        }
57      }
58    }
59    /**
60     * Action handler for the menu items
61     * Called when the user selects a menu item; the item's action command
corresponds to the command table for MapForce
62     */
63    public void actionPerformed( ActionEvent e )
64    {
65      try
66      {
67        int iCmd = Integer.parseInt( e.getActionCommand() );
68        // Handle explicitly the Close commands
69        switch ( iCmd )
70        {
71          case 57602:       // Close
72          case 34050:       // Close All
73            MapForceContainer.initMapForceDocument();
74            break;
75          default:
76            MapForceContainer.mapForceControl.exec( iCmd );
77            break;
78        }
79      }
80      catch ( Exception ex )
81      {
82        ex.printStackTrace();
83      }
84
85    }
```

18.5.3.6 UI Update Event Handling

The code listing below shows how a UI-Update event handler can be created.

```
01 /**
02     * Call-back from the MapForceControl.
03     * Called to enable/disable commands
04     */
05    @Override
06    public void onUpdateCmdUI() throws AutomationException
07    {
```

```
08     // A command should be enabled if the result of queryStatus contains the
Supported (1) and Enabled (2) flags
09     for ( java.util.Map.Entry<Integer, MenuItem> pair : menuMap.entrySet() )
10
      pair.getValue().setEnabled( MapForceContainer.mapForceControl.queryStatus(
pair.getKey() ) > 2 );
11   }
12  /**
13    * Call-back from the MapForceControl.
14    * Usually called while enabling/disabling commands due to UI updates
15    */
16   @Override
17   public boolean onIsActiveEditor( String i_strFilePath ) throws
AutomationException
18   {
19     try {
20       return
MapForceContainer.mapForceDocument.getDocument().getFullName().equalsIgnoreCase(
i_strFilePath );
21     } catch ( Exception e ) {
22       return false;
23     }
24   }
```

18.5.3.7 Listing the Properties of a MapForce Mapping

The listing below shows how a Mapping object in MapForce can be loaded as a table and prepared for modal activation.

```
01 //access MapForce Java-COM bridge
02 import com.altova.automation.MapForce.*;
03 import com.altova.automation.MapForce.Component;
04 import com.altova.automation.MapForce.Enums.ENUMComponentUsageKind;
05
06 //access AWT and Swing components
07 import java.awt.*;
08 import javax.swing.*;
09 import javax.swing.table.*;
10
11
12 /**
13  * A simple example of a table control loading the structure from a Mapping
object.
14  * The class receives an Mapping object, loads its components in a JTable,
and prepares
15  * for modal activation.
16  *
17  * Feel free to modify and extend this sample.
18  *
19  * @author Altova GmbH
20  */
21 class MapForceTable extends JDialog
22 {
23   /**
24    * The table control
```


© 2018 Altova GmbH

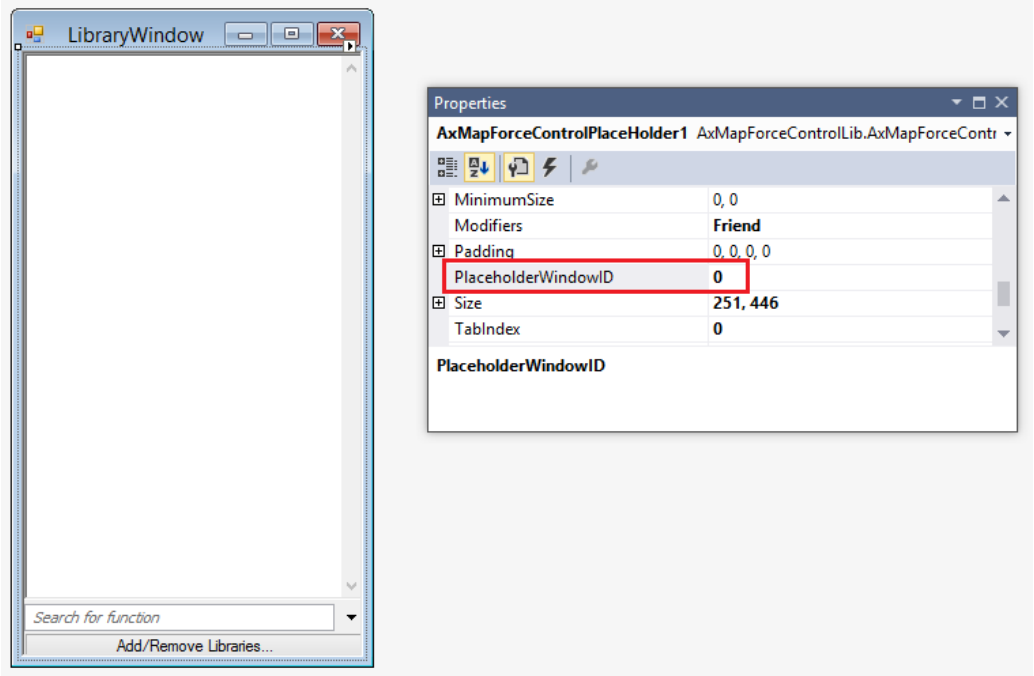
```
25    */
26   private JTable myTable;
27
28   /**
29    * Constructor that prepares the modal dialog containing the filled table
control
30    * @param mapping The data to be displayed in the table
31    * @param parent  Parent frame
32    */
33   public MapForceTable( Mapping mapping, Frame parent )
34   {
35     // Construct the modal dialog
36     super( parent, "MapForce component table", true );
37     // Build up the tree
38     fillTable( mapping );
39     // Arrange controls in the dialog
40     setContentPane( new JScrollPane( myTable ) );
41   }
42
43   /**
44    * Loads the components of a Mapping object in the table
45    * @param mapping Source data
46    */
47   private void fillTable( Mapping mapping)
48   {
49     try
50     {
51       // count how many Instance components do we have
52       int size = 0;
53       for (Component comp : mapping.getComponents())
54         if ( comp.getUsageKind() ==
ENUMComponentUsageKind.eComponentUsageKind_Instance )
55           ++size;
56
57       // Prepare data
58       final String[] columnNames = { "Component", "Has inputs", "Has
outputs", "Input file", "Output file", "Schema" };
59       final Object[][] data = new Object[size ][ 7 ] ;
60       int index = 0 ;
61       for (Component comp : mapping.getComponents())
62         if ( comp.getUsageKind() ==
ENUMComponentUsageKind.eComponentUsageKind_Instance )
63         {
64           int i = 0;
65           data[ index ][ i++ ] = comp.getName() ;
66           data[ index ][ i++ ] = new
Boolean( comp.getHasIncomingConnections() );
67           data[ index ][ i++ ] = new
Boolean( comp.getHasOutgoingConnections() );
68           data[ index ][ i++ ] = comp.getInputInstanceFile();
69           data[ index ][ i++ ] = comp.getOutputInstanceFile();
70           data[ index++ ][ i ] = comp.getSchema() ;
71         }
72
73       // Set up table
74       myTable = new JTable( new AbstractTableModel() {
75           public String getColumnName(int col) { return columnNames[col]; }
76           public int getRowCount() { return data.length; }
77           public int getColumnCount() { return columnNames.length; }
```

```
78            public Object getValueAt(int row, int col) { return data[row][col];
}
79            public boolean isCellEditable(int row, int col) { return false; }
80            public Class getColumnClass(int c) { return getValueAt(0,
c).getClass(); }
81        } );
82
83        // Set width
84        for( index = 0 ; index < columnNames.length ; ++index )
85          myTable.getColumnModel().getColumn( index ).setMinWidth( 80 );
86        myTable.getColumnModel().getColumn( 5 ).setMinWidth( 400 );
87      }
88      catch (Exception e)
89      {
90        e.printStackTrace();
91      }
92    }
93
94 }
```

18.5.4 VB.NET

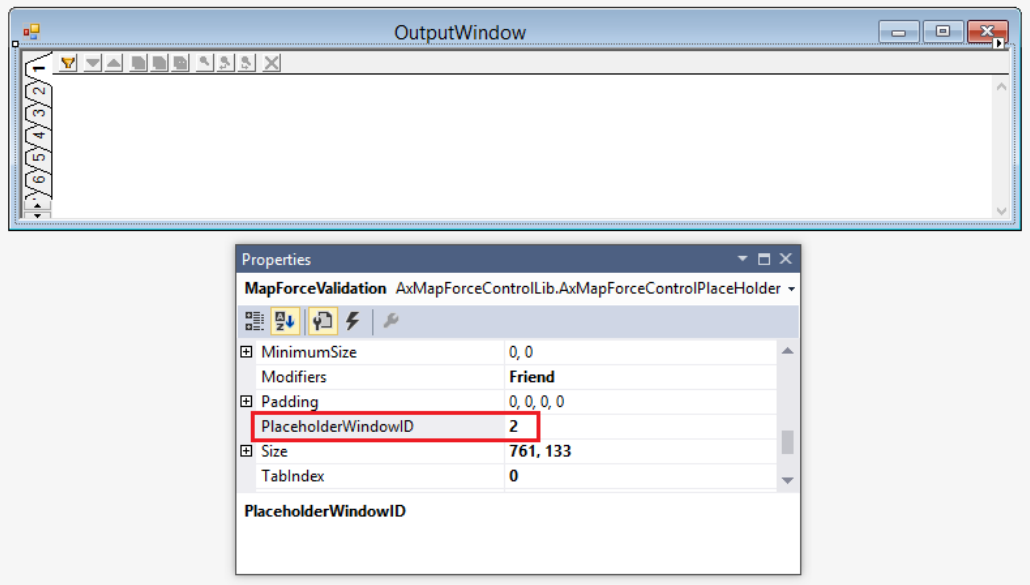
Source code which illustrates integration of MapForceControl into a VB.NET application can be found in the folder `<ApplicationFolder>\Examples\ActiveX\VB.NET` of your MapForce installation. The solution consists of three windows, as follows:

1. **MainWindow.vb** - the main document window, which also includes a basic application menu.
2. **LibraryWindow.vb** - the Library window. The contents of this window is populated by a Placeholder control which has the **PlaceholderWindowID** property set to 0 (this value instructs the control to display specifically the Library window).



3. **OutputWindow.vb** - the Messages (Output) window. The contents of this window is

populated by a Placeholder control which has the **PlaceholderWindowID** property set to 2 (this value instructs the control to display specifically the Output window).



Before attempting to build and run this solution, note the following steps:

Step 1: Check the prerequisites

For the list of prerequisites, see Prerequisites.

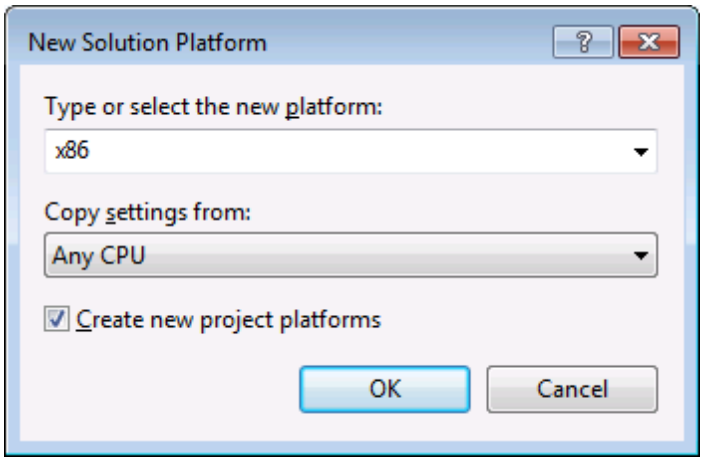
Step 2: Copy the sample to a directory where you have write permissions

To avoid running Visual Studio as an Administrator, copy the source code to a directory where you have write permissions, instead of running it from the default location.

Step 3: Set the build platform

- Create a build platform configuration that matches the platform under which you want to build (x86, x64). Here is how you can create the build configuration:

  a. Right-click the solution in Visual Studio, and select **Configuration Manager**.
  b. Under **Active solution platform**, select **New...** and then select the x86 or x64 configuration (in this example, **x86**).



You are now ready to build and run the solution in Visual Studio. Remember to build using the configuration that matches your target platform (x86, x64); otherwise, runtime errors might occur.

© 2018 Altova GmbH

18.6 Command Reference

This section lists the names and identifiers of all menu commands that are available within MapForce. Every sub-section lists the commands from the corresponding top-level menu of MapForce. The command tables are organized as follows:

- The "Menu Item" column shows the command's menu text as it appears in MapForce, to make it easier for you to identify the functionality behind the command.
- The "Command Name" column specifies the string that can be used to get an icon with the same name from **ActiveX\Images** folder of the MapForce installation directory.
- The "ID" column shows the numeric identifier of the column that must be supplied as argument to methods which execute or query this command.

To execute a command, use the `MapForceControl.Exec` or the `MapForceControlDocument.Exec` methods. To query the status of a command, use the `MapForceControl.QueryStatus` or `MapForceControlDocument.QueryStatus` methods.

Depending on the edition of MapForce you have installed, some of these commands might not be supported.

18.6.1 "File" Menu

The "File" menu has the following commands:

Menu item	Command name	ID
New...	ID_FILE_NEW	57600
Open...	ID_FILE_OPEN	57601
Save	ID_FILE_SAVE	57603
Save As...	ID_FILE_SAVE_AS	57604
Save All	ID_FILE_SAVEALL	32377
Reload	IDC_FILE_RELOAD	32467
Close	ID_WINDOW_CLOSE	32453
Close All	ID_WINDOW_CLOSEALL	32454
Print...	ID_FILE_PRINT	57607
Print Preview	ID_FILE_PRINT_PREVIEW	57609
Print Setup...	ID_FILE_PRINT_SETUP	57606
Validate Mapping	ID_MAPPING_VALIDATE	32347
Mapping Settings	ID_MAPPING_SETTINGS	32396
Generate Code in Selected Language	ID_FILE_GENERATE_SELECTED_COD	32362

Menu item	Command name	ID
	E	
XSLT 1.0	ID_FILE_GENERATEXSLT	32360
XSLT 2.0	ID_FILE_GENERATEXSLT2	32361
XQuery	ID_FILE_GENERATEXQUERY	32359
Java	ID_FILE_GENERATEJAVACODE	32358
C# (Sharp)	ID_FILE_GENERATECSCODE	32357
C++	ID_FILE_GENERATECPPCODE	32356
Compile to MapForce Server Execution File...	ID_FILE_CREATE_SERVER_EXECUTION_FILE	32517
Deploy to FlowForce Server...	ID_FILE_DEPLOY_MAPPING	32506
Generate Documentation...	ID_FILE_GENERATE_DOCUMENTATION	32468
Recent File	ID_FILE_MRU_FILE1	57616
Exit	ID_APP_EXIT	57665

18.6.2 "Edit" Menu

The "Edit" menu has the following commands:

Menu item	Command name	ID
Undo	ID_EDIT_UNDO	57643
Redo	ID_EDIT_REDO	57644
Find...	ID_EDIT_FIND	57636
Find Next	ID_EDIT_FINDNEXT	32349
Find Previous	ID_EDIT_FINDPREV	32350
Cut	ID_EDIT_CUT	57635
Copy	ID_EDIT_COPY	57634
Paste	ID_EDIT_PASTE	57637
Delete	ID_EDIT_CLEAR	57632
Select All	ID_EDIT_SELECT_ALL	57642

© 2018 Altova GmbH

18.6.3 "Insert" Menu

The "Insert" menu has the following commands:

Menu item	Command name	ID
XML Schema/File...	ID_INSERT_XSD	32393
Database...	ID_INSERT_DATABASE	32389
EDI...	ID_INSERT_EDI	32390
Text File...	ID_INSERT_TXT	32392
Web Service Function...	ID_INSERT_WEBSERVICE_FUNCTION	32319
Excel 2007+ File...	ID_INSERT_EXCEL	32376
XBRL Document...	ID_INSERT_XBRL	32469
JSON Schema/File...	ID_INSERT_JSON	32531
Insert Input...	ID_FUNCTION_INSERT_INPUT	32383
Insert Output...	ID_FUNCTION_INSERT_OUTPUT	32402
Constant...	ID_INSERT_CONSTANT	32388
Variable...	ID_INSERT_VARIABLE	32500
Join	ID_INSERT_JOIN	32581
Sort: Nodes/Rows	ID_INSERT_SORT	32444
Filter: Nodes/Rows	ID_INSERT_FILTER	32391
SQL-WHERE/ORDER	ID_INSERT_SQLWHERE_CONDITION	32351
Value-Map	ID_INSERT_VALUEMAP	32354
IF-Else Condition	ID_INSRT_CONDITION	32394
Exception	ID_INSERT_EXCEPTION	32311

18.6.4 "Project" Menu

The "Project" menu has the following commands:

Menu item	Command name	ID
Reload Project	ID_PROJECT_RELOAD	32476

Menu item	Command name	ID
Close Project	ID_FILE_CLOSEPROJECT	32355
Save Project	ID_FILE_SAVEPROJECT	32378
Add Files to Project...	ID_PROJECT_ADDFILESTOPROJECT	32420
Add Active File to Project	ID_PROJECT_ADDACTIVEFILETOPROJECT	32419
Create Folder...	ID_PROJECT_CREATE_FOLDER	32310
Open Mapping	ID_PROJECT_OPEN_MAPPING	32307
Create Mapping for Operation...	ID_PROJECT_CREATE_MAPPING_FOR_OPERATION	32399
Add Mapping File for Operation...	ID_PROJECT_ADD_MAPPING	32309
Insert Web Service...	ID_PROJECT_INSERT_WEBSERVICE	32306
Open File in XMLSpy	ID_PROJECT_OPEN_IN_XMLSPY	32305
Generate Code for Entire Project	ID_PROJECT_GENERATE_ALL	32303
XSLT 1.0	ID_PROJECT_GENERATEXSLTCODE_ENTIRE	32408
XSLT 2.0	ID_PROJECT_GENERATEXSLT2CODE_ENTIRE	32409
XQuery	ID_PROJECT_GENERATEXQUERYCODE_ENTIRE	32410
Java	ID_PROJECT_GENERATEJAVACODE_ENTIRE	32411
C# (Sharp)	ID_PROJECT_GENERATECSCODE_ENTIRE	32412
C++	ID_PROJECT_GENERATECPPCODE_ENTIRE	32413
Properties	ID_PROJECT_PROPERTIES	32404
Recent Project	ID_FILE_MRU_PROJECT1	32364

18.6.5 "Component" Menu

The "Component" menu has the following commands:

Menu item	Command name	ID
Change Root Element...	ID_COMPONENT_CHANGEROOTELEM	32334

© 2018 Altova GmbH

Menu item	Command name	ID
	ENT	
Edit Schema Definition in XMLSpy	ID_COMPONENT_EDIT_SCHEMA	32337
Edit FlexText Configuration	ID_COMPONENT_EDIT_MFT	32301
Add/Remove/Edit Database Objects...	ID_COMPONENT_SELECTTABLES	32346
Create Mapping to EDI X12 997	ID_COMPONENT_CREATE_MAPPING_TO_997	32483
Create Mapping to EDI X12 999	ID_COMPONENT_CREATE_MAPPING_TO_999	32484
Refresh	IDC_COMMAND_REFRESH_COMPONENT	32373
Add Duplicate Input Before	ID_COMPONENT_CREATE_DUPLICATE_ICON_BEFORE	32503
Add Duplicate Input After	ID_COMPONENT_CREATE_DUPLICATE_ICON	32335
Remove Duplicate	ID_COMPONENT_REMOVE_DUPLICATE_ICON	32339
Add Comment Before	ID_COMPONENT_ADD_COMMENT_BEFORE	32518
Add Comment After	ID_COMPONENT_ADD_COMMENT_AFTER	32519
Add Processing Instruction Before...	ID_COMPONENT_ADD_PI_BEFORE	32520
Add Processing Instruction After...	ID_COMPONENT_ADD_PI_AFTER	32521
Edit Processing Instruction Name...	ID_COMPONENT_EDIT_PI	32524
Delete Comment/Processing Instruction	ID_COMPONENT_REMOVE_COMMENT_PI	32522
Write Content as CDATA Section	ID_COMPONENT_TOGGLE_CDATA	32525
Database Table Actions	ID_POPUP_DATABASETABLEACTIONS	32400
Query Database...	ID_QUERY_DATABASE	32341
Align Tree Left	ID_COMPONENT_LEFTALIGNTREE	32338
Align Tree Right	ID_COMPONENT_RIGHTALIGNTREE	32340
Properties	ID_COMPONENT_PROPERTIES	32336

18.6.6 "Connection" Menu

The "Connection" menu has the following commands:

Menu item	Command name	ID
Auto Connect Matching Children	ID_CONNECTION_AUTOCONNECTCHILDREN	32342
Settings for Connect Matching Children	ID_CONNECTION_SETTINGS	32344
Connect Matching Children...	ID_CONNECTION_MAPCHILDELEMENTS	32343
Target Driven (Standard)	ID_POPUP_NORMALCONNECTION	32401
Copy-All (Copy Child Items)	ID_POPUP_NORMALWITHCHILDREN_CONNECTION	32460
Source Driven (Mixed Content)	ID_POPUP_ORDERBYSOURCECONNECTION	32403
Properties	ID_POPUP_CONNECTION_SETTINGS	32398

18.6.7 "Function" Menu

The "Function" menu has the following commands:

Menu item	Command name	ID
Create User-Defined Function...	ID_FUNCTION_CREATE_EMPTY	32380
Create User-Defined Function from Selection...	ID_FUNCTION_CREATE_FROM_SELECTION	32381
Function Settings	ID_FUNCTION_SETTINGS	32387
Remove Function	ID_FUNCTION_REMOVE	32385
Insert Input...	ID_FUNCTION_INSERT_INPUT	32383
Insert Output...	ID_FUNCTION_INSERT_OUTPUT	32402

18.6.8 "Output" Menu

The "Output" menu has the following commands:

© 2018 Altova GmbH

Menu item	Command name	ID
XSLT 1.0	ID_SELECT_LANGUAGE_XSLT	32433
XSLT 2.0	ID_SELECT_LANGUAGE_XSLT2	32434
XQuery	ID_SELECT_LANGUAGE_XQUERY	32432
Java	ID_SELECT_LANGUAGE_JAVA	32431
C# (Sharp)	ID_SELECT_LANGUAGE_CSHARP	32430
C++	ID_SELECT_LANGUAGE_CPP	32429
Built-In Execution Engine	ID_SELECT_LANGUAGE_BUILTIN	32490
Validate Output File	ID_XML_VALIDATE	32458
Save Output File...	IDC_FILE_SAVEGENERATEDOUTPUT	32321
Save All Output Files...	IDC_FILE_SAVEALLGENERATEDOUTPUT	32374
Regenerate Output	ID_REGENERATE_PREVIEW_OUTPUT	32480
Run SQL-Script	ID_TRANSFORM_RUN_SQL	32442
Insert/Remove Bookmark	ID_TOGGLE_BOOKMARK	32317
Next Bookmark	ID_GOTONEXTBOOKMARK	32315
Previous Bookmark	ID_GOTOPREVBOOKMARK	32314
Remove All Bookmarks	ID_REMOVEALLBOOKMARKS	32313
Pretty-Print XML Text	ID_PRETTY_PRINT_OUTPUT	32363
Text View Settings	ID_TEXTVIEWSETTINGSDIALOG	32472

18.6.9 "Debug" Menu

The "Debug" menu has the following commands:

Menu item	Command name	ID
Start Debugging	ID_DEBUG_START	32540
Stop Debugging	ID_DEBUG_STOP	32541
Step Into	ID_DEBUG_STEP_INTO	32545
Step Over	ID_DEBUG_STEP_OVER	32551
Step Out	ID_DEBUG_STEP_OUT	32552
Minimal Step	ID_DEBUG_STEP_NEXT_TRACE	32554

18.6.10 "View" Menu

The "View" menu has the following commands:

Menu item	Command name	ID
Show Annotations	ID_SHOW_ANNOTATION	32435
Show Types	ID_SHOW_TYPES	32437
Show Library in Function Header	ID_VIEW_SHOWLIBRARYINFUNCTIONHEADER	32448
Show Tips	ID_SHOW_TIPS	32436
XBRL Display Options	ID_VIEW_XBRL_DISPLAY_OPTIONS	32473
Show Selected Component Connectors	ID_VIEW_AUTOHIGHLIGHTCOMPONENTCONNECTIONS	32443
Show Connectors from Source to Target	ID_VIEW_RECURSIVEAUTOHIGHLIGHT	32447
Zoom...	ID_VIEW_ZOOM	32451
Back	ID_CMD_BACK	32479
Forward	ID_CMD_FORWARD	32478
Status Bar	ID_VIEW_STATUS_BAR	59393
Library Window	ID_VIEW_LIBRARY_WINDOW	32445
Messages	ID_VIEW_VALIDATION_OUTPUT	32450
Overview	ID_VIEW_OVERVIEW_WINDOW	32446
Project Window	ID_VIEW_PROJECT_WINDOW	32302
Values	ID_DEBUG_VIEW_VALUES_WINDOW	32544
Context	ID_DEBUG_VIEW_CONTEXT_WINDOW	32546
Breakpoints	ID_DEBUG_VIEW_DEBUGPOINTS_WINDOW	32547

18.6.11 "Tools" Menu

The "Tools" menu has the following commands:

Menu item	Command name	ID
Global Resources	IDC_GLOBALRESOURCES	37401

© 2018 Altova GmbH

Menu item	Command name	ID
<plugin not loaded>	IDC_GLOBALRESOURCES_SUBMENU ENTRY1	37408
Create Reversed Mapping	ID_CREATE_REVERSED_MAPPING	32489
Customize...	IDC_APP_TOOLS_CUSTOMIZE	32959
Options...	ID_TOOLS_OPTIONS	32441

18.6.12 "Window" Menu

The "Window" menu has the following commands:

Menu item	Command name	ID
Cascade	ID_WINDOW_CASCADE	57650
Tile Horizontal	ID_WINDOW_TILE_HORZ	57651
Tile Vertical	ID_WINDOW_TILE_VERT	57652

18.6.13 "Help" Menu

The "Help" menu has the following commands:

Menu item	Command name	ID
Table of Contents...	IDC_HELP_CONTENTS	32322
Index...	IDC_HELP_INDEX	32323
Search...	IDC_HELP_SEARCH	32324
Software Activation...	IDC_ACTIVATION	32701
Order Form...	IDC_OPEN_ORDER_PAGE	32326
Registration...	IDC_REGISTRATION	32330
Check for Updates...	IDC_CHECK_FOR_UPDATES	32700
Support Center...	IDC_OPEN_SUPPORT_PAGE	32327
FAQ on the Web...	IDC_SHOW_FAQ	32331
Download Components and Free Tools...	IDC_OPEN_COMPONENTS_PAGE	32325
MapForce on the Internet..	IDC_OPEN_XML_SPY_HOME	32328
MapForce Training...	IDC_OPEN_MAPFORCE_TRAINING_PAGE	32300

Menu item	Command name	ID
About MapForce...	ID_APP_ABOUT	57664

© 2018 Altova GmbH

18.7 Object Reference

Objects:
MapForceCommand
MapForceCommands
MapForceControl
MapForceControlDocument
MapForceControlPlaceHolder

To give access to standard MapForce functionality, objects of the **MapForce automation interface** can be accessed as well. See MapForceControl.Application, MapForceControlDocument.Document and MapForceControlPlaceHolder.Project for more information.

18.7.1 MapForceCommand

Properties:
`ID`
`Label`
`Name`
`IsSeparator`
`ToolTip`
`StatusText`
`Accelerator`
`SubCommands`

Description:
A command object can be one of the following: an executable command, a command container (for example, a menu, submenu, or toolbar), or a menu separator. To determine what kind of information is stored in the current `Command` object, query its `ID`, `IsSeparator`, and `SubCommands` properties, as follows.

The Command object is...	When...
An executable command	• `ID` is greater than zero • `IsSeparator` is false • `SubCommands` is empty
A command container	• `ID` is zero • `IsSeparator` is true • `SubCommands` contains a collection of `Command` objects.
Separator	• `ID` is zero • `IsSeparator` is true

18.7.1.1 Accelerator

Property: `Accelerator` as string

Description:
Returns the accelerator key defined for the command. If the command has no accelerator key assigned, this property returns the empty string. The string representation of the accelerator key has the following format:

```
[ALT+][CTRL+][SHIFT+]key
```

Where `key` is converted using the Windows Platform SDK function `GetKeyNameText`.

18.7.1.2 ID

Property: `ID` as long

Description:
This property gets the unique identifier of the command. A command's ID is required to execute the command (using `Exec`) or query its status (using `QueryStatus`). If the command is a container for other commands (for example, a top-level menu), or a separator, the ID is 0.

18.7.1.3 IsSeparator

Property: `IsSeparator` as boolean

Description:
The property returns `true` if the command object is a menu separator; `false` otherwise. See also Command.

18.7.1.4 Label

Property: `Label` as string

Description:
This property gets the text of the command as it is displayed in the graphical user interface of MapForce. If the command is a separator, "Label" is an empty string. This property may also return an empty string for some toolbar commands that do not have any GUI text associated with them.

18.7.1.5 Name

Property: `Name` as string

Description:
This property gets the unique name of the command. This value can be used to get the icon file of the command, where it is available. The available icon files can be found in the folder

© 2018 Altova GmbH

<ApplicationFolder>\Examples\ActiveX\Images of your MapForce installation.

18.7.1.6 StatusText

Property: `Label` as string

Description:
The status text is the text shown in the status bar of MapForce when the command is selected. It applies only to command objects that are not separators or containers of other commands; otherwise, the property is an empty string.

18.7.1.7 SubCommands

Property: `SubCommands` as `Commands`

Description:
The `SubCommands` property gets the collection of `Command` objects that are sub-commands of the current command. The property is applicable only to commands that are containers for other commands (menus, submenus, or toolbars). Such container commands have the `ID` set to 0, and the `IsSeparator` property set to `false`.

18.7.1.8 ToolTip

Property: `ToolTip` as string

Description:
This property gets the text that is shown as a tool-tip for each command. If the command does not have a tooltip text, the property returns an empty string.

18.7.2 MapForceCommands

Properties:
`Count`
`Item`

Description:
Collection of `Command` objects to get access to command labels and IDs of the MapForceControl. Those commands can be executed with the `Exec` method and their status can be queried with `QueryStatus`.

18.7.2.1 Count

Property: `Count` as long

Description:

Number of Command objects on this level of the collection.

18.7.2.2 Item

Property: Item (n as long) as Command

Description:
Gets the command with the index n in this collection. Index is 1-based.

18.7.3 MapForceControl

Properties:
IntegrationLevel
Appearance
Application
BorderStyle
CommandsList
CommandsStructure (deprecated)
EnableUserPrompts
MainMenu
Toolbars

Methods:
Open
Exec
QueryStatus

Events:
OnUpdateCmdUI
OnOpenedOrFocused
OnCloseEditingWindow
OnFileChangedAlert
OnContextChanged
OnDocumentOpened
OnValidationWindowUpdated

This object is a complete ActiveX control and should only be visible if the MapForce library is used in the Application Level mode.

```
CLSID: A38637E9-5759-4456-A167-F01160CC22C1
ProgID: Altova.MapForceControl
```

18.7.3.1 Properties

The following properties are defined:

IntegrationLevel

© 2018 Altova GmbH

EnableUserPrompts
Appearance
BorderStyle

Command related properties:
CommandsList
MainMenu
Toolbars
CommandsStructure (deprecated)

Access to MapForceAPI:
Application

18.7.3.1.1 Appearance

Property: Appearance as short

Dispatch Id: *-520*

Description:
A value not equal to 0 displays a client edge around the control. Default value is 0.

18.7.3.1.2 Application

Property: Application as Application

Dispatch Id: *1*

Description:
The Application property gives access to the Application object of the complete MapForce automation server API. The property is read-only.

18.7.3.1.3 BorderStyle

Property: BorderStyle as short

Dispatch Id: *-504*

Description:
A value of 1 displays the control with a thin border. Default value is 0.

18.7.3.1.4 CommandsList

Property: CommandList as Commands (read-only)

Dispatch Id: *1004*

Description:
This property returns a flat list of all commands defined available with MapForceControl. To get commands organized according to their menu structure, use `MainMenu`. To get toolbar commands, use `Toolbars`.

```
public void GetAllMapForceCommands()
{
    // Get all commands from the MapForce ActiveX control assigned to the
current form
    MapForceControlLib.MapForceCommands commands =
this.axMapForceControl1.CommandList;
    // Iterate through all commands
    for (int i = 0; i < commands.Count; i++)
    {
      // Get each command by index and output it to the console
      MapForceControlLib.MapForceCommand cmd =
axMapForceControl1.CommandList[i];
      Console.WriteLine("{0} {1} {2}", cmd.ID, cmd.Name,
cmd.Label.Replace("&", ""));
    }
}
```

C# example

18.7.3.1.5 EnableUserPrompts

Property: `EnableUserPrompts` as boolean

Dispatch Id: *1006*

Description:
Setting this property to *false*, disables user prompts in the control. The default value is *true*.

18.7.3.1.6 IntegrationLevel

Property: `IntegrationLevel` as `ICActiveXIntegrationLevel`

Dispatch Id: *1000*

Description:
The `IntegrationLevel` property determines the operation mode of the control. See also Integration at Application Level and Integration at Document Level for more information.

Note: It is important to set this property immediately after the creation of the `MapForceControl` object.

© 2018 Altova GmbH

18.7.3.1.7 MainMenu

Property: `MainMenu as Command` (read-only)

Dispatch Id: *1003*

Description:
This property provides information about the structure and commands available in the MapForceControl main menu, as a `Command` object. The `Command` object contains all available submenus of MapForce (for example "File", "Edit", "View" etc.). To access the submenu objects, use the `SubCommands` property of the `MainMenu` property. Each submenu is also a `Command` object. For each submenu, you can then further iterate through their `SubCommands` property in order to get their corresponding child commands and separators (this technique may be used, for example, to create the application menu programmatically). Note that some menu commands act as containers ("parents") for other menu commands, in which case they also have a `SubCommands` property. To get the structure of all menu commands programmatically, you will likely need to create a recursive function.

```
public void GetMapForceMenus()
{
    // Get the main menu from the MapForce ActiveX control assigned to the
current form
    MapForceControlLib.MapForceCommand mainMenu =
this.axMapForceControl1.MainMenu;

    // Loop through entries of the main menu (e.g. File, Edit, etc.)
    for (int i = 0; i < mainMenu.SubCommands.Count; i++)
    {
      MapForceControlLib.MapForceCommand menu = mainMenu.SubCommands[i];
      Console.WriteLine("{0} menu has {1} children items (including
separators)", menu.Label.Replace("&", ""), menu.SubCommands.Count);
    }
}
```

C# example

18.7.3.1.8 Toolbars

Property: `Toolbars as Commands` (read-only)

Dispatch Id: *1005*

Description:
This property provides information about the structure of MapForceControl toolbars, as a `Command` object. The `Command` object contains all available toolbars of MapForce. To access the toolbars, use the `SubCommands` property of the `Toolbars` property. Each toolbar is also a `Command` object. For each toolbar, you can then further iterate through their `SubCommands` property in order to get their commands (this technique may be used, for example, to create the application's toolbars programmatically).

```
public void GetMapForceToolbars()
{
    // Get the application toolbars from the MapForce ActiveX control assigned
to the current form
    MapForceControlLib.MapForceCommands toolbars =
this.axMapForceControl1.Toolbars;

    // Iterate through all toolbars
    for (int i = 0; i < toolbars.Count; i++)
    {
      MapForceControlLib.MapForceCommand toolbar = toolbars[i];
      Console.WriteLine();
      Console.WriteLine("The toolbar \"{0}\" has the following commands:",
toolbar.Label);

      // Iterate through all commands of this toolbar
      for (int j = 0; j < toolbar.SubCommands.Count; j++)
      {
         MapForceControlLib.MapForceCommand cmd = toolbar.SubCommands[j];
         // Output only command objects that are not separators
         if (!cmd.IsSeparator)
         {
            Console.WriteLine("{0}, {1}, {2}", cmd.ID, cmd.Name,
cmd.Label.Replace("&", ""));
         }
      }
    }
}
```

C# example

18.7.3.2 Methods

The following methods are defined:

Open
Exec
QueryStatus

18.7.3.2.1 Exec

Method: `Exec` (`nCmdID` as long) as boolean

Dispatch Id: *6*

Description:
This method calls the MapForce command with the ID `nCmdID`. If the command can be executed, the method returns `true`. To get a list of all available commands, use `CommandsList`. To retrieve the status of any command, use `QueryStatus`.

© 2018 Altova GmbH

18.7.3.2.2 Open

Method: `Open` (`strFilePath` as string) as boolean

Dispatch Id: *5*

Description:
The result of the method depends on the extension passed in the argument `strFilePath`. If the file extension is `.sps`, a new document is opened. If the file extension is `.svp`, the corresponding project is opened. If a different file extension is passed into the method, the control tries to load the file as a new component into the active document.

Do not use this method to load documents or projects when using the control in document-level integration mode. Instead, use `MapForceControlDocument.Open` and `MapForceControlPlaceHolder.OpenProject`.

18.7.3.2.3 QueryStatus

Method: `QueryStatus` (`nCmdID` as long) as long

Dispatch Id: *7*

Description:
`QueryStatus` returns the enabled/disabled and checked/unchecked status of the command specified by `nCmdID`. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.
1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if `QueryStatus` returns `0` the command ID is not recognized as a valid MapForce command. If `QueryStatus` returns a value of `1` or `5`, the command is disabled.

18.7.3.3 Events

The MapForceControl ActiveX control provides the following connection point events:

`OnUpdateCmdUI`
`OnOpenedOrFocused`
`OnCloseEditingWindow`
`OnFileChangedAlert`
`OnContextChanged`

`OnDocumentOpened`
`OnValidationWindowUpdated`

18.7.3.3.1 OnCloseEditingWindow

Event: `OnCloseEditingWindow` (i_strFilePath as String) as boolean

Dispatch Id: *1002*

Description:
This event is triggered when MapForce needs to close an already open document. As an answer to this event, clients should close the editor window associated with *i_strFilePath*. Returning *true* from this event indicates that the client has closed the document. Clients can return *false* if no specific handling is required and MapForceControl should try to close the editor and destroy the associated document control.

18.7.3.3.2 OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description:
This event is not used in MapForce

18.7.3.3.3 OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` as Document)

Dispatch Id: *1*

Description:
This event is triggered whenever a document is opened. The argument `objDocument` is a `Document` object from the MapForce automation interface and can be used to query for more details about the document, or perform additional operations. When integrating on document-level, it is often better to use the event `MapForceControlDocument.OnDocumentOpened` instead.

18.7.3.3.4 OnFileChangedAlert

Event: `OnFileChangedAlert` (`i_strFilePath` as String) as bool

Dispatch Id: *1001*

Description:
This event is triggered when a file loaded with MapForceControl is changed on the hard disk by another application. Clients should return true, if they handled the event, or false, if MapForce should handle it in its customary way, i.e. prompting the user for reload.

© 2018 Altova GmbH

18.7.3.3.5 OnLicenseProblem

Event: `OnLicenseProblem` (`i_strLicenseProblemText` as String)

Dispatch Id: *1005*

Description:
This event is triggered when MapForceControl detects that no valid license is available for this control. In case of restricted user licenses this can happen some time after the control has been initialized. Integrators should use this event to disable access to this control's functionality. After returning from this event, the control will block access to its functionality (e.g. show empty windows in its controls and return errors on requests).

18.7.3.3.6 OnOpenedOrFocused

Event: `OnOpenedOrFocused` (`i_strFilePath` as String, `i_bOpenWithThisControl` as bool)

Dispatch Id: *1000*

Description:
When integrating at application level, this event informs clients that a document has been opened, or made active by MapForce.

When integrating at document level, this event instructs the client to open the file `i_strFilePath` in a document window. If the file is already open, the corresponding document window should be made the active window.

if `i_bOpenWithThisControl` is `true`, the document must be opened with MapForceControl, since internal access is required. Otherwise, the file can be opened with different editors.

18.7.3.3.7 OnToolWindowUpdated

Event: `OnToolWindowUpdated(pToolWnd` as long `)`

Dispatch Id: *1006*

Description:
This event is triggered when the tool window is updated.

18.7.3.3.8 OnUpdateCmdUI

Event: `OnUpdateCmdUI ()`

Dispatch Id: *1003*

Description:
Called frequently to give integrators a good opportunity to check status of MapForce commands using `MapForceControl.QueryStatus`. Do not perform long operations in this callback.

18.7.3.3.9 OnValidationWindowUpdated

Event: `OnValidationWindowUpdated` ()

Dispatch Id: *3*

Description:
This event is triggered whenever the validation output window is updated with new information.

18.7.4 MapForceControlDocument

Properties:
`Appearance`
`BorderStyle`
`Document`
`IsModified`
`Path`
`ReadOnly`

Methods:
`Exec`
`New`
`Open`
`QueryStatus`
`Reload`
`Save`
`SaveAs`

Events:
`OnDocumentOpened`
`OnDocumentClosed`
`OnModifiedFlagChanged`
`OnContextChanged`
`OnFileChangedAlert`
`OnActivate`

If the MapForceControl is integrated in the Document Level mode each document is displayed in an own object of type `MapForceControlDocument`. The `MapForceControlDocument` contains only one document at the time but can be reused to display different files one after another.

This object is a complete ActiveX control.

```
CLSID: DFBB0871-DAFE-4502-BB66-08CEB7DF5255
ProgID: Altova.MapForceControlDocument
```


© 2018 Altova GmbH

18.7.4.1 Properties

The following properties are defined:

`ReadOnly`
`IsModified`
`Path`
`Appearance`
`BorderStyle`

Access to MapForceAPI:
`Document`

18.7.4.1.1 Appearance

Property: `Appearance` as short

Dispatch Id: *-520*

Description:
A value not equal to `0` displays a client edge around the document control. Default value is `0`.

18.7.4.1.2 BorderStyle

Property: `BorderStyle` as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

18.7.4.1.3 Document

Property: `Document` as Document

Dispatch Id: *1*

Description:
The `Document` property gives access to the `Document` object of the MapForce automation server API. This interface provides additional functionality which can be used with the document loaded in the control. The property is read-only.

18.7.4.1.4 IsModified

Property: `IsModified` as boolean (read-only)

Dispatch Id: *1006*

Description:
`IsModified` is *true* if the document content has changed since the last open, reload or save operation. It is *false*, otherwise.

18.7.4.1.5 Path

Property: `Path` as string

Dispatch Id: *1005*

Description:
Sets or gets the full path name of the document loaded into the control.

18.7.4.1.6 ReadOnly

Property: `ReadOnly` as boolean

Dispatch Id: *1007*

Description:
Using this property you can turn on and off the read-only mode of the document. If `ReadOnly` is `true` it is not possible to do any modifications.

18.7.4.2 Methods

The following methods are defined:

Document handling:
`New`
`Open`
`Reload`
`Save`
`SaveAs`

Command Handling:
`Exec`
`QueryStatus`

© 2018 Altova GmbH

18.7.4.2.1 Exec

Method: `Exec` (`nCmdID` as long) as boolean

Dispatch Id: *8*

Description:
`Exec` calls the MapForce command with the ID `nCmdID`. If the command can be executed, the method returns `true`. This method should be called only if there is currently an active document available in the application.

To get commands organized according to their menu structure, use the `MainMenu` property of MapForceControl. To get toolbar commands, use the `Toolbars` property of the MapForceControl.

18.7.4.2.2 New

Method: `New` () as boolean

Dispatch Id: *1000*

Description:
This method initializes a new document inside the control..

18.7.4.2.3 Open

Method: `Open` (`strFileName` as string) as boolean

Dispatch Id: *1001*

Description:
`Open` loads the file `strFileName` as the new document into the control.

18.7.4.2.4 QueryStatus

Method: `QueryStatus` (`nCmdID` as long) as long

Dispatch Id: *9*

Description:
`QueryStatus` returns the enabled/disabled and checked/unchecked status of the command specified by `nCmdID`. The status is returned as a bit mask.

Bit	Value	Name	Meaning
0	1	Supported	Set if the command is supported.

1	2	Enabled	Set if the command is enabled (can be executed).
2	4	Checked	Set if the command is checked.

This means that if `QueryStatus` returns `0` the command ID is not recognized as a valid MapForce command. If `QueryStatus` returns a value of `1` or `5` the command is disabled. The client should call the `QueryStatus` method of the document control if there is currently an active document available in the application.

18.7.4.2.5 Reload

Method: `Reload` () as boolean

Dispatch Id: *1002*

Description:
`Reload` updates the document content from the file system.

18.7.4.2.6 Save

Method: `Save` () as boolean

Dispatch Id: *1003*

Description:
`Save` saves the current document at the location `Path`.

18.7.4.2.7 SaveAs

Method: `SaveAs` (`strFileName` as string) as boolean

Dispatch Id: *1004*

Description:
`SaveAs` sets `Path` to *strFileName* and then saves the document to this location.

18.7.4.3 Events

The MapForceControlDocument ActiveX control provides following connection point events:

`OnDocumentOpened`
`OnDocumentClosed`
`OnModifiedFlagChanged`
`OnContextChanged`
`OnFileChangedAlert`
`OnActivate`

© 2018 Altova GmbH

OnSetEditorTitle

18.7.4.3.1 OnActivate

Event: `OnActivate` ()

Dispatch Id: *1005*

Description:
This event is triggered when the document control is activated, has the focus, and is ready for user input.

18.7.4.3.2 OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description: None

18.7.4.3.3 OnDocumentClosed

Event: `OnDocumentClosed` (`objDocument` as Document)

Dispatch Id: *1001*

Description:
This event is triggered whenever the document loaded into this control is closed. The argument `objDocument` is a `Document` object from the MapForce automation interface and should be used with care.

18.7.4.3.4 OnDocumentOpened

Event: `OnDocumentOpened` (`objDocument` as Document)

Dispatch Id: *1000*

Description:
This event is triggered whenever a document is opened in this control. The argument `objDocument` is a `Document` object from the MapForce automation interface, and can be used to query for more details about the document, or perform additional operations.

18.7.4.3.5 OnDocumentSaveAs

Event: `OnContextDocumentSaveAs` (`i_strFileName` as String)

Dispatch Id: *1007*

Description:
This event is triggered when this document gets internally saved under a new name.

18.7.4.3.6 OnFileChangedAlert

Event: `OnFileChangedAlert` () as bool

Dispatch Id: *1003*

Description:
This event is triggered when the file loaded into this document control is changed on the hard disk by another application. Clients should return true, if they handled the event, or false, if MapForce should handle it in its customary way, i.e. prompting the user for reload.

18.7.4.3.7 OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` as boolean)

Dispatch Id: *1002*

Description:
This event gets triggered whenever the document changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the document contents differs from the original content, and *false*, otherwise.

18.7.4.3.8 OnSetEditorTitle

Event: OnSetEditorTitle ()

Dispatch Id: *1006*

Description:
This event is being raised when the contained document is being internally renamed.

18.7.5 MapForceControlPlaceHolder

Properties available for all kinds of placeholder windows:
`PlaceholderWindowID`

© 2018 Altova GmbH

Properties for project placeholder window:
`Project`

Methods for project placeholder window:
`OpenProject`
`CloseProject`

The `MapForceControlPlaceHolder` control is used to show the additional MapForce windows like Overview, Library or Project window. It is used like any other ActiveX control and can be placed anywhere in the client application.

```
CLSID: FDEC3B04-05F2-427d-988C-F03A85DE53C2
ProgID: Altova.MapForceControlPlaceHolder
```

18.7.5.1 Properties

The following properties are defined:

`PlaceholderWindowID`

Access to MapForceAPI:
`Project`

18.7.5.1.1 Label

Property: `Label` as String (read-only)

Dispatch Id: *1001*

Description:
This property gives access to the title of the placeholder. The property is read-only.

18.7.5.1.2 PlaceholderWindowID

Property: `PlaceholderWindowID as MapForceControlPlaceholderWindow`

Dispatch Id: *1*

Description:
This property specifies which MapForce window should be displayed in the client area of the control. The `PlaceholderWindowID` can be set at any time to any valid value of the `MapForceControlPlaceholderWindow` enumeration. The control changes its state immediately and shows the new MapForce window.

18.7.5.1.3 Project

Property: `Project` as Project (read-only)

Dispatch Id: *2*

Description:
The `Project` property gives access to the `Project` object of the MapForce automation server API. This interface provides additional functionality which can be used with the project loaded into the control. The property will return a valid project interface only if the placeholder window has `PlaceholderWindowID` with a value of `MapForceXProjectWindow (=3)`. The property is read-only.

18.7.5.2 Methods

The following method is defined:

`OpenProject`
`CloseProject`

18.7.5.2.1 OpenProject

Method: `OpenProject` (`strFileName` as string) as boolean

Dispatch Id: *3*

Description:
`OpenProject` loads the file `strFileName` as the new project into the control. The method will fail if the placeholder window has a `PlaceholderWindowID` different to `XMLSpyXProjectWindow (=3)`.

18.7.5.2.2 CloseProject

Method: `CloseProject` ()

Dispatch Id: *4*

Description:
`CloseProject` closes the project loaded the control. The method will fail if the placeholder window has a `PlaceholderWindowID` different to `MapForceXProjectWindow (=3)`.

© 2018 Altova GmbH

18.7.5.3 Events

The MapForceControlPlaceholder ActiveX control provides following connection point events:

`OnModifiedFlagChanged`

18.7.5.3.1 OnModifiedFlagChanged

Event: `OnModifiedFlagChanged` (`i_bIsModified` as boolean)

Dispatch Id: *1*

Description:
This event gets triggered only for placeholder controls with a `PlaceholderWindowID` of `MapForceXProjectWindow (=3).` The event is fired whenever the project content changes between modified and unmodified state. The parameter *i_bIsModifed* is *true* if the project contents differs from the original content, and *false*, otherwise.

18.7.5.3.2 OnSetLabel

Event: `OnSetLabel(i_strNewLabel` as string`)`

Dispatch Id: *1000*

Description:
Raised when the title of the placeholder window is changed.

18.7.6 Enumerations

The following enumerations are defined:

`ICActiveXIntegrationLevel`
`MapForceControlPlaceholderWindow`

18.7.6.1 ICActiveXIntegrationLevel

Possible values for the IntegrationLevel property of the MapForceControl.

```
ICActiveXIntegrationOnApplicationLevel  = 0
ICActiveXIntegrationOnDocumentLevel     = 1
```

18.7.6.2 MapForceControlPlaceholderWindow

This enumeration contains the list of the supported additional MapForce windows.

```
MapForceXNoWindow                = -1
MapForceXLibraryWindow           = 0
MapForceXOverviewWindow          = 1
MapForceXValidationWindow        = 2
MapForceXProjectWindow           = 3
MapForceXDebuggerValuesWindow    = 4
MapForceXDebuggerContextWindow   = 5
MapForceXDebuggerPointsWindow    = 6
```


© 2018 Altova GmbH

Altova MapForce 2018 Enterprise Edition

Anexos

19 Anexos

Estos anexos contienen datos técnicos sobre MapForce e información importante sobre las licencias. Cada anexo incluye varios apartados:

Datos técnicos

- Requisitos de SO y memoria
- Analizador XML de Altova
- Motores XSLT y XQuery de Altova
- Compatibilidad con Unicode
- Uso de Internet
- Medición de licencias

Información sobre licencias

- Distribución electrónica de software
- Derechos de propiedad intelectual
- Contrato de licencia para el usuario final de Altova

© 2018 Altova GmbH

19.1 Información sobre motores

Esta sección ofrece información sobre las características de implementación del validador XML de Altova y de los motores XSLT 1.0, XSLT 2.0 y XQuery de Altova.

19.1.1 Información sobre motores XSLT y XQuery

Los motores XSLT y XQuery de MapForce siguen las especificaciones del W3C y, por tanto, son más estrictos que otros motores anteriores de Altova, como los de las versiones antiguas de XMLSpy. Por consiguiente, MapForce señala algunos errores leves que antes no se notificaban en la versión anterior de estos motores.

Por ejemplo:

- Se notifica un error de tipo (`err:XPTY0018`) si el resultado de un operador de ruta de acceso contiene tanto nodos como no nodos.
- Se notifica un error de tipo (`err:XPTY0019`) si `E1` en una expresión XPath `E1/E2` no da como resultado una secuencia de nodos.

Si encuentra este tipo de errores, modifique el documento XSLT/XQuery o el documento de instancia según corresponda.

Esta sección describe características relacionadas con la implementación de los motores e incluye estos apartados:

- XSLT 1.0
- XSLT 2.0
- XQuery 1.0

19.1.1.1 XSLT 1.0

El motor XSLT 1.0 de MapForce cumple con la recomendación XSLT 1.0 del 16 de noviembre de 1999 y con la recomendación XPath 1.0 del 16 de noviembre de 1999, ambas del W3C.

Nota sobre la implementación
Cuando el atributo `method` de `xsl:output` tiene el valor HTML o si selecciona de forma predeterminada el formato de salida HTML, los caracteres especiales del archivo XML o XSLT se insertan en el documento HTML como referencias de caracteres HTML. Por ejemplo, el carácter U+00A0 (la referencia de carácter hexadecimal para un espacio de no separación) se inserta en el código HTML como referencia de carácter (` ` o ` `) o como referencia de entidad (` `).

19.1.1.2 XSLT 2.0

Temas de este apartado:

- Especificaciones con las que cumple el motor
- Compatibilidad con versiones antiguas
- Espacios de nombres
- Compatibilidad con esquemas
- Comportamiento propio de esta implementación

Especificaciones

El motor XSLT 2.0 de MapForce cumple con la recomendación XSLT 2.0 del 23 de enero de 2007 y la recomendación XPath 2.0 del 14 de diciembre de 2010, ambas del W3C.

Compatibilidad con versiones antiguas

El motor XSLT 2.0 es compatible con versiones previas. Esto solamente es relevante cuando se utiliza el motor XSLT 2.0 para procesar una hoja de estilos XSLT 1.0. Tenga en cuenta que los resultados obtenidos con el motor XSLT 1.0 pueden ser diferentes a los obtenidos con el motor XSLT 2.0 en modo de compatibilidad con versiones antiguas.

Espacios de nombres

En su hoja de estilos XSLT 2.0 debe declarar estos espacios de nombres para poder usar los constructores de tipo y las funciones disponibles en XSLT 2.0. Los prefijos que aparecen a continuación son los que se suelen usar, pero puede usar otros prefijos si quiere.

Espacio de nombres	Prefijo	URI del espacio de nombres
Tipos XML Schema	`xs:`	`http://www.w3.org/2001/XMLSchema`
Funciones XPath 2.0	`fn:`	`http://www.w3.org/2005/xpath-functions`

Estos espacios de nombres se suelen declarar en el elemento `xsl:stylesheet` o en el elemento `xsl:transform`:

```
<xsl:stylesheet version="2.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:fn="http://www.w3.org/2005/xpath-functions"
    ...
</xsl:stylesheet>
```

Es necesario tener en cuenta que:

- El motor XSLT 2.0 utiliza el espacio de nombres Funciones XPath 2.0 y XQuery 1.0

© 2018 Altova GmbH

como **espacio de nombres de funciones predeterminado**. Esto significa que puede usar funciones XPath 2.0 y XSLT 2.0 en su hoja de estilos sin prefijos. Si declara el espacio de nombres Funciones XPath 2.0 en su hoja de estilos con un prefijo, podrá usar el prefijo asignado en la declaración.

- Cuando se usan constructores de tipo y tipos del espacio de nombres XML Schema, el prefijo utilizado en la declaración de espacio de nombres se debe usar en la llamada al constructor de tipo (por ejemplo, `xs:date`).
- Algunas funciones XPath 2.0 se llaman igual que algunos tipos de datos de XML Schema. Por ejemplo, las funciones XPath `fn:string` y `fn:boolean` y los tipos de datos de XML Schema `xs:string` y `xs:boolean`. Por tanto, si usa la expresión `string('Hello')`, la expresión se evalúa como `fn:string('Hello')` y no como `xs:string('Hello')`.

Compatibilidad con esquemas

El motor XSLT 2.0 está preparado para esquemas de modo que puede usar tipos de esquema definidos por el usuario y la instrucción `xsl:validate`.

Comportamiento propio de esta implementación

Más abajo puede ver cómo se ocupa el motor XSLT 2.0 de algunos aspectos del comportamiento de las funciones XSLT 2.0 relacionadas con esta implementación.

`xsl:result-document`

También son compatibles estas codificaciones específicas de Altova: `x-base16tobinary` y `x-base64tobinary`.

`function-available`

Esta función mira si hay funciones del ámbito disponibles (funciones XSLT, XPath y de extensión).

`unparsed-text`

El atributo `href` acepta (i) rutas de acceso relativas para archivos que estén en la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles estas codificaciones específicas de Altova: `x-binarytobase16` y `x-binarytobase64`.

`unparsed-text-available`

El atributo `href` acepta (i) rutas de acceso relativas para archivos que estén en la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles estas codificaciones específicas de Altova: `x-binarytobase16` y `x-binarytobase64`.

Nota: estos valores de codificación estaban implementados en el ya descatalogado AltovaXML pero ya no se utilizan (son obsoletos): `base16tobinary`, `base64tobinary`, `binarytobase16` y `binarytobase64`.

19.1.1.3 XQuery 1.0

Temas de este apartado:

- Especificaciones con las que cumple el motor
- Compatibilidad con esquemas
- Codificación
- Espacios de nombres
- Fuentes XML y validación
- Comprobación de tipos estática y dinámica
- Módulos biblioteca
- Funciones externas
- Intercalaciones
- Precisión de datos numéricos
- Compatibilidad con instrucciones XQuery

Especificaciones compatibles

El motor XQuery 1.0 de MapForce cumple con la recomendación XQuery 1.0 del 14 de diciembre de 2010 del W3C. El estándar XQuery concede libertad a la hora de implementar muchas características. A continuación explicamos cómo se implementaron estas características en el motor XQuery 1.0 de MapForce.

Compatibilidad con esquemas

El motor XQuery 1.0 está preparado para esquemas.

Codificación

El motor XQuery 1.0 es compatible con las codificaciones de caracteres UTF-8 y UTF-16.

Espacios de nombres

Se predefinen estos URI de espacios de nombres y sus enlaces asociados.

Espacio de nombres	Prefijo	URI del espacio de nombres
Tipos XML Schema	`xs:`	`http://www.w3.org/2001/XMLSchema`
Schema instance	`xsi:`	`http://www.w3.org/2001/XMLSchema-instance`
Funciones integradas	`fn:`	`http://www.w3.org/2005/xpath-functions`
Funciones locales	`local:`	`http://www.w3.org/2005/xquery-local-functions`

Es importante tener en cuenta que:

© 2018 Altova GmbH

- El motor XQuery 1.0 entiende que los prefijos de la tabla anterior están enlazados con los correspondientes espacios de nombres.
- Como el espacio de nombres de funciones integradas (ver tabla) es el espacio de nombres de funciones predeterminado de XQuery, no es necesario usar el prefijo `fn:` cuando se invocan funciones integradas (por ejemplo, `string("Hello")` llamará a la función `fn:string`). No obstante, el prefijo `fn:` se puede utilizar para llamar a una función integrada sin necesidad de declarar el espacio de nombres en el prólogo de la consulta (por ejemplo: `fn:string("Hello")`).
- Puede cambiar el espacio de nombres de funciones predeterminado declarando la expresión `default function namespace` en el prólogo de la consulta.
- Cuando use tipos del espacio de nombres XML Schema, puede usar el prefijo `xs:` sin necesidad de declarar los espacios de nombres de forma explícita ni enlazar estos prefijos a los espacios de nombres en el prólogo de la consulta. (Ejemplo: `xs:date` y `xs:yearMonthDuration`.) Si quiere usar otros prefijos para el espacio de nombres de XML Schema, estos se deben declarar en el prólogo de la consulta. (Ejemplo: `declare namespace alt = "http://www.w3.org/2001/XMLSchema"; alt:date("2004-10-04")`.)
- Recuerde que los tipos de datos `untypedAtomic`, `dayTimeDuration` y `yearMonthDuration` se movieron del espacio de nombres XPath Datatypes al espacio de nombres XML Schema (es decir, ahora es `xs:yearMonthDuration`.)

Si se asignaron mal los espacios de nombres para funciones, constructores de tipo, pruebas de nodo, etc., se emite un error. Sin embargo, recuerde que algunas funciones se llaman igual que los tipos de datos de esquema (p. ej. `fn:string` y `fn:boolean`.) (Se definen `xs:string` y `xs:boolean`.) El prefijo del espacio de nombres determina si se usa la función o el constructor de tipo.

Documento XML de origen y validación

Los documentos XML que se utilizan para ejecutar un documento XQuery con el motor XQuery 1.0 deben tener un formato XML correcto. Sin embargo, no es necesario que sean válidos con respecto a un esquema XML. Si el archivo no es válido, el archivo no válido se carga sin información de esquema. Si el archivo XML está asociado a un esquema externo y es válido con respecto a dicho esquema, se genera información posterior a la validación de esquema, que se utilizará para evaluar la consulta.

Comprobación de tipos estática y dinámica

En la fase de análisis estático se revisan aspectos de la consulta como la sintaxis, si existen referencias externas (p. ej. para módulos), si las funciones y variables que se invocan están definidas, etc. Si se detecta un error en la fase de análisis estático, se notifica y la ejecución se interrumpe.

La comprobación dinámica de tipos se realiza en tiempo de ejecución, cuando la consulta se ejecuta. Si un tipo no es compatible con los requisitos de una operación, se emite un error. Por ejemplo, la expresión `xs:string("1") + 1` devuelve un error porque la operación de suma no se puede llevar a cabo en un operando de tipo `xs:string`.

Módulos biblioteca

Los módulos biblioteca almacenan funciones y variables para poder volver a utilizarlas. El motor XQuery 1.0 es compatible con el uso de módulos almacenados en un **solo archivo XQuery externo**. Dicho archivo de módulo debe incluir una declaración `module` en su prólogo que apunte a un espacio de nombres de destino Por ejemplo:

```
module namespace libns="urn:module-library";
declare variable $libns:company := "Altova";
declare function libns:webaddress() { "http://www.altova.com" };
```

Todas las funciones y variables declaradas en el módulo pertenecen al espacio de nombres asociado al módulo. El módulo se importa en un archivo XQuery con la instrucción `import module` del prólogo de la consulta. La instrucción `import module` solamente importa funciones y variables declaradas directamente en el archivo de módulo biblioteca. Por ejemplo:

```
import module namespace modlib = "urn:module-library" at "modulefilename.xq";

if        ($modlib:company = "Altova")
then       modlib:webaddress()
else       error("No match found.")
```

Funciones externas

Las funciones externas son incompatibles con el motor XQuery 1.0, es decir, todas las expresiones que usen la palabra clave `external`. Por ejemplo:

```
declare function hoo($param as xs:integer) as xs:string external;
```

Intercalaciones

La intercalación predeterminada es la intercalación de puntos de código Unicode, que compara las cadenas de texto según sus puntos de código Unicode. Otras intercalaciones compatibles son las intercalaciones ICU que se enumeran aquí. Para usar una intercalación concreta, indique su URI tal y como aparece en la lista de intercalaciones compatibles. Las comparaciones de cadenas de texto, incluidas las comparaciones para las funciones `fn:max` y `fn:min`, se harán según la intercalación especificada. Si no se indica la opción de intercalación, se utiliza la intercalación de puntos de código Unicode predeterminada.

Precisión de tipos numéricos

- El tipo de datos `xs:integer` es de precisión arbitraria, es decir, puede representar un número de dígitos cualquiera.
- El tipo de datos `xs:decimal` tiene un límite de 20 dígitos después del punto decimal.
- Los tipos de datos `xs:float` y `xs:double` tienen una precisión limitada de 15 dígitos.

Compatibilidad con instrucciones XQuery

La instrucción `Pragma` no es compatible. Si se encuentra, se ignora y en su lugar se evalúa la

© 2018 Altova GmbH

expresión de reserva.

19.1.2 Funciones XSTL y XPath/XQuery

Esta sección enumera las funciones de extensión de Altova y otras funciones de extensión que se pueden utilizar con expresiones XPath y XQuery. Las funciones de extensión de Altova se pueden usar con los motores XSLT y XQuery de Altova y ofrecen algunas funciones más aparte de las que están disponibles en las bibliotecas de funciones definidas en los estándares del W3C.

Aspectos generales

Es necesario tener en cuenta estos puntos generales:

- A las funciones de las bibliotecas de funciones `core` definidas en las especificaciones W3C se les puede llamar sin un prefijo. Esto se debe a que los motores XSLT y XQuery leen funciones sin prefijo como si pertenecieran a un espacio de nombres de funciones predeterminado, que es el que se especifica en las especificaciones de las funciones XPath y XQuery`http://www.w3.org/2005/xpath-functions`. Si este espacio de nombres se declara explícitamente en un documento XSLT o XQuery, el prefijo utilizado en la declaración de espacio de nombres también se puede usar en el nombre de las funciones.
- Por lo general, si una función espera como argumento una secuencia de un elemento y se suministra una secuencia de más de un elemento, entonces se devuelve un error.
- Todas las comparaciones de cadena se realizan usando la intercalación de puntos de código Unicode.
- Los resultados que son QName se serializan de esta forma `[prefijo:]nombrelocal`.

Precisión de xs:decimal

El término *precisión* hace referencia al número de dígitos del número y la especificación exige un mínimo de 18 dígitos. Para las operaciones de división que dan un resultado de tipo `xs:decimal`, la precisión es de 19 dígitos después del punto decimal sin redondear.

Uso horario implícito

Cuando hay que comparar dos valores `date`, `time` o `dateTime`, es necesario conocer el uso horario de los valores que se deben comparar. Cuando el uso horario no se conoce de forma explícita, se usa el uso horario implícito, que se toma del reloj del sistema. Para probar cuál es su valor puede utilizar la función `fn:implicit-timezone()`.

Intercalaciones

La intercalación predeterminada es la intercalación de puntos de código Unicode, que compara las cadenas de texto según sus puntos de código Unicode. El motor usa el algoritmo de intercalación Unicode. Otras intercalaciones compatibles son las intercalaciones ICU que aparecen más abajo. Para usar una intercalación indique su URI tal y como aparece en la tabla más abajo. Las comparaciones de cadenas de texto (incluidas las que usan las funciones `fn:max` y `fn:min`) se harán según la intercalación especificada. Si no se especifica la opción de intercalación, se usa la intercalación predeterminada de puntos de código Unicode.

Idioma	Identificadores URI
da: danés	da_DK
de: alemán	de_AT, de_BE, de_CH, de_DE, de_LI, de_LU
en: inglés	en_AS, en_AU, en_BB, en_BE, en_BM, en_BW, en_BZ, en_CA, en_GB, en_GU, en_HK, en_IE, en_IN, en_JM, en_MH, en_MP, en_MT, en_MU, en_NA, en_NZ, en_PH, en_PK, en_SG, en_TT, en_UM, en_US, en_VI, en_ZA, en_ZW
es: español	es_419, es_AR, es_BO, es_CL, es_CO, es_CR, es_DO, es_EC, es_ES, es_GQ, es_GT, es_HN, es_MX, es_NI, es_PA, es_PE, es_PR, es_PY, es_SV, es_US, es_UY, es_VE
fr: francés	fr_BE, fr_BF, fr_BI, fr_BJ, fr_BL, fr_CA, fr_CD, fr_CF, fr_CG, fr_CH, fr_CI, fr_CM, fr_DJ, fr_FR, fr_GA, fr_GN, fr_GP, fr_GQ, fr_KM, fr_LU, fr_MC, fr_MF, fr_MG, fr_ML, fr_MQ, fr_NE, fr_RE, fr_RW, fr_SN, fr_TD, fr_TG
it: italiano	it_CH, it_IT
ja: jaoonés	ja_JP
nb: noruego bokmål	nb_NO
nl: holandés	nl_AW, nl_BE, nl_NL
nn: noruego nynorsk	nn_NO
pt: portugués	pt_AO, pt_BR, pt_GW, pt_MZ, pt_PT, pt_ST
ru: ruso	ru_MD, ru_RU, ru_UA
sv: sueco	sv_FI, sv_SE

Eje del espacio de nombres
El eje del espacio de nombres se dejó de utilizar en XPath 2.0. Sin embargo, los ejes de espacio de nombres son compatibles con la aplicación. Para acceder a la información sobre el espacio de nombres con mecanismos de XPath 2.0, utilice las funciones `fn:in-scope-prefixes()`, `fn:namespace-uri()` y `fn:namespace-uri-for-prefix()`.

19.1.2.1 Funciones de extensión de Altova

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** `http://www.altova.com/xslt-extensions` y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

© 2018 Altova GmbH

Las funciones definidas en las especificaciones XPath/XQuery Functions del W3C se pueden usar en (i) expresiones XPath en contextos XSLT y en (ii) expresiones XQuery en documentos XQuery. En esta documentación las funciones que se pueden usar en el primer contexto (XPath en XSLT) llevan el símbolo **XP** y se les llama funciones XPath. Las funciones que se pueden usar en contextos XQuery llevan el símbolo **XQ** y funcionan como funciones XQuery. Las especificaciones XSLT del W3C también definen funciones que se pueden usar en expresiones XPath en documentos XSLT. Estas funciones llevan el símbolo **XSLT** y se les denomina funciones XSLT. Por cada función se indica en qué versión de XPath/XQuery y XSLT se puede usar (ver símbolos más abajo). Las funciones de las bibliotecas de funciones XPath/XQuery y XSLT aparecen sin prefijo. Las funciones de extensión de otras bibliotecas, como las funciones de extensión de Altova, aparecen con un prefijo.

Funciones XPath *(en expresiones XPath en XSLT):*	**XP1** **XP2** **XP3.1**
Funciones XSLT *(en expresiones XPath en XSLT):*	**XSLT1** **XSLT2** **XSLT3**
Funciones XQuery *(en expresiones XQuery en XQuery):*	**XQ1** **XQ3.1**

Funciones XSLT

Las funciones XSLT solo se pueden utilizar en expresiones XPath en un contexto XSLT (igual que las funciones XSLT 2.0 `current-group()` o `key()`). Estas funciones no están pensadas para contextos no XSLT (p. ej. contextos XQuery) y, por tanto, no funcionarán en contextos que no sean XSLT. Recuerde que las funciones XSLT para XBRL solamente se pueden utilizar con ediciones de los productos de Altova compatibles con XBRL.

Funciones XPath/XQuery

Las funciones XPath/XQuery se pueden utilizar en expresiones XPath, en contextos XSLT y en expresiones XQuery:

- Funciones de fecha y hora
- Funciones de geoubicación
- Funciones relacionadas con imágenes
- Funciones numéricas
- Funciones de secuencia
- Funciones de cadena
- Funciones varias

Funciones para gráficos (solo en las ediciones Enterprise y Server Edition)

Las funciones de extensión para gráficos de Altova son compatibles con las ediciones Enterprise y Server Edition de los productos de Altova solamente. Estas funciones permiten generar gráficos a partir de datos XML.

19.1.2.1.1 Funciones XSLT

Las **funciones de extensión XSLT** pueden utilizarse en expresiones XPath en contextos XSLT y no funcionan en contextos que no sean XSLT (por ejemplo, en contextos XQuery).

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** `http://www.altova.com/xslt-extensions` y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath *(en expresiones XPath en XSLT):*	XP1 XP2 XP3.1
Funciones XSLT *(en expresiones XPath en XSLT):*	XSLT1 XSLT2 XSLT3
Funciones XQuery *(en expresiones XQuery en XQuery):*	XQ1 XQ3.1

Funciones estándar

▼ distinct-nodes [altova:]

altova:distinct-nodes(*node()****)** como **node()*** XSLT1 XSLT2 XSLT3

Toma un conjunto de nodos como entrada y devuelve el mismo conjunto menos los nodos que tengan el mismo valor (es decir, devuelve los nodos que son únicos). La comparación se hace con la función XPath/XQuery `fn:deep-equal`.

⊟ *Ejemplo*

- **altova:distinct-nodes**(`country`) devuelve todos los nodos secundarios `country` excepto los que tengan el mismo valor.

▼ evaluate [altova:]

altova:evaluate(ExpresiónXPath *como xs:string***[, ValorDe$p1, ... ValorDe$pN])** XSLT1 XSLT2 XSLT3

Toma una expresión XPath, pasada como cadena, como argumento obligatorio. Devuelve el resultado de la expresión evaluada. Por ejemplo, **evaluate('//Name[1]')** devuelve el contenido del primer elemento `Name` del documento. Observe que para pasar la expresión `//Name[1]` como cadena basta con ponerla entre comillas simples.

© 2018 Altova GmbH

La función `altova:evaluate` puede tomar más argumentos, que son los valores de las variables del ámbito que se llaman `p1`, `p2`, `p3`... `pN`. Recuerde que (i) las variables deben definirse con nombres de tipo `pX`, siendo `X` un entero; (ii) los argumentos de la función `altova:evaluate` (*ver firma más abajo*), a partir del segundo argumento, ofrecen los valores de las variables, correspondiendo la secuencia de argumentos a la secuencia numérica de variables: `p1` corresponde a `pN` y el segundo argumento será el valor de la variable `p1`, el tercer argumento al de la variable `p2`, y así sucesivamente; (iii) los valores de las variables deben ser de tipo `item*`.

- *Ejemplo*

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath, 10, 20, 'hi')" />
```

da el resultado `"hi 20 10"`

En el ejemplo anterior puede observar que:

- El segundo argumento de la expresión `altova:evaluate` es el valor asignado a la variable `$p1`, el tercer argumento es el valor asignado a la variable `$p2` y así sucesivamente.
- Observe que el cuarto argumento de la función es un valor de cadena porque va entre comillas simples.
- El atributo `select` del elemento `xs:variable` suministra la expresión XPath. Como esta expresión debe ser de tipo `xs:string`, se pone entre comillas simples.

- *Más ejemplos*

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, //Name[1])" />
  ```
  El resultado es el valor del primer elemento `Name`.

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, '//Name[1]')" />
  ```
  El resultado es "`//Name[1]`*"*

La función de extensión `altova:evaluate()` es muy práctica cuando una expresión XPath de la hoja de estilos XSLT contiene partes que se deben evaluar de forma dinámica. Por ejemplo, imagine que el usuario selecciona un criterio de ordenación y este criterio se almacena en el atributo `UserReq/@sortkey`. En la hoja de estilos podría tener esta expresión:

```
      <xsl:sort select="altova:evaluate(../UserReq/@sortkey)"
order="ascending"/>
```

La función `altova:evaluate()` lee el atributo `sortkey` del elemento secundario `UserReq` del primario del nodo de contexto. Imagine que el valor del atributo `sortkey` es `Price`. En ese caso, la función `altova:evaluate()` devuelve `Price`, que se convierte en el valor del atributo `select`:

```
<xsl:sort select="Price" order="ascending"/>
```

Si esta instrucción `sort` aparece dentro del contexto de un elemento llamado `Order`, entonces los elementos `Order` se ordenan según el valor de los secundarios `Price`. Otra opción es que, si el valor de `@sortkey` fuera `Date`, por ejemplo, entonces los elementos `Order` se ordenarían según el valor de los secundarios `Date`. Es decir, el criterio de ordenación para `Order` se selecciona del atributo `sortkey` en tiempo de ejecución. Esto no sería posible con una expresión como:

```
<xsl:sort select="../UserReq/@sortkey" order="ascending"/>
```

En este caso, el criterio de ordenación sería el propio atributo `sortkey`, no `Price` ni `Date` (ni otro contenido actual de `sortkey`).

Nota: el contexto estático incluye espacios de nombres, tipos y funciones (pero no variables) del entorno de llamada. El URI base y el espacio de nombres predeterminado se heredan.

Más ejemplos

- Variables estáticas: `<xsl:value-of select="$i3, $i2, $i1" />`
  El resultado es los valores de las tres variables.

- Expresión XPath dinámica con variables dinámicas:
  `<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />`
  `<xsl:value-of select="altova:evaluate($xpath, 10, 20, 30)" />`
  El resultado es "30 20 10"

- Expresión XPath dinámica sin variables dinámicas:
  `<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />`
  `<xsl:value-of select="altova:evaluate($xpath)" />`
  Error: no se definió la variable para $p3.

encode-for-rtf [altova:]

altova:encode-for-rtf(entrada *como xs:string*, conservarEspaciosEnBlanco *como xs:boolean*, conservarLíneasNuevas *como xs:boolean***)** como **xs:string** XSLT2 XSLT3

Convierte la cadena de entrada en código para RTF. Los espacios en blanco y las líneas nuevas se conservan o no dependiendo del valor booleano especificado para los correspondientes parámetros.

[Subir]

Funciones XBRL

Las funciones XBRL de Altova solo funcionan en las ediciones de los productos de Altova que son compatibles con XBRL.

xbrl-footnotes [altova:]

altova:xbrl-footnotes(*node()***)** como **node()*** XSLT2 XSLT3

© 2018 Altova GmbH

Toma un nodo como argumento de entrada y devuelve el conjunto de nodos de nota al pie XBRL al que hace referencia el nodo de entrada.

▼ xbrl-labels [altova:]

altova:xbrl-labels(*xs:QName***, *xs:string***)** como **node()*** XSLT2 XSLT3
Toma dos argumentos de entrada: un nombre de nodo y la ubicación del archivo de taxonomía en el que está el nodo. La función devuelve los nodos de etiqueta XBRL asociados al nodo de entrada.

[Subir]

19.1.2.1.2 Funciones XPath/XQuery: de fecha y hora

Las funciones de extensión de fecha y hora de Altova se pueden usar en expresiones XPath y XQuery y permiten procesar datos almacenados en tipos de datos XML Schema de fecha y hora. Estas funciones se pueden usar con los **motores XPath 3.0 y XQuery 3.0** de Altova y están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **http://www.altova.com/xslt-extensions** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

▼ Funciones agrupadas según su funcionalidad

- Agregar una duración a xs:dateTime y devolver xs:dateTime
- Agregar una duración a xs:date y devolver xs:date
- Agregar una duración a xs:time y devolver xs:time
- Recuperar duraciones y aplicarles formato
- Quitar la zona horaria de las funciones que generan la fecha/hora actual
- Devolver el día de la semana de una fecha como número entero
- Devolver el número de semana de una fecha como número entero

- Generar la fecha, la hora y el tipo de duración a partir de los componentes léxicos de cada tipo
- Construir un tipo date, dateTime o a partir de la cadena de entrada
- Funciones para calcular la edad

▼ Funciones por orden alfabético

altova:add-days-to-date
altova:add-days-to-dateTime
altova:add-hours-to-dateTime
altova:add-hours-to-time
altova:add-minutes-to-dateTime
altova:add-minutes-to-time
altova:add-months-to-date
altova:add-months-to-dateTime
altova:add-seconds-to-dateTime
altova:add-seconds-to-time
altova:add-years-to-date
altova:add-years-to-dateTime
altova:age
altova:age-details
altova:build-date
altova:build-duration
altova:build-time
altova:current-dateTime-no-TZ
altova:current-date-no-TZ
altova:current-time-no-TZ
altova:format-duration
altova:parse-date
altova:parse-dateTime
altova:parse-duration
altova:parse-time
altova:weekday-from-date
altova:weekday-from-dateTime
altova:weeknumber-from-date
altova:weeknumber-from-dateTime

[Subir]

Agregar una duración a `xs:dateTime` XP3.1 XQ3.1

Estas funciones sirven para agregar una duración a **`xs:dateTime`** y devuelven **`xs:dateTime`**. El tipo `xs:dateTime` tiene el formato `SSAA-MM-DDThh:mm:ss.sss`. Se trata de la concatenación de los formatos `xs:date` y `xs:time` separados por la letra `T`. Si quiere, puede usar un sufijo de zona horaria`+01:00 (por ejemplo)`.

▼ add-years-to-dateTime [altova:]

`altova:add-years-to-dateTime(FechaHora` *`as xs:dateTime`*`, `**`Años`** *`as xs:integer`*`)`
como **`xs:dateTime`** XP3.1 XQ3.1

Añade una duración en años un valor de fecha y hora. El segundo argumento es el número de años que se debe añadir al valor de fecha y hora dado como primer argumento. El

© 2018 Altova GmbH

resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2024-01-15T14:00:00
- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -4) devuelve 2010-01-15T14:00:00

add-months-to-dateTime [altova:]

altova:add-months-to-dateTime(FechaHora *as xs:dateTime*, **Meses** *as xs:integer*)
como **xs:dateTime** XP3.1 XQ3.1
Añade una duración en meses a un valor de fecha y hora. El segundo argumento es el número de meses que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2014-11-15T14:00:00
- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -2) devuelve 2013-11-15T14:00:00

add-days-to-dateTime [altova:]

altova:add-days-to-dateTime(FechaHora *as xs:dateTime*, **Días** *as xs:integer*)
como **xs:dateTime** XP3.1 XQ3.1
Añade una duración en días a un valor de fecha y hora. El segundo argumento es el número de días que se deben añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2014-01-25T14:00:00
- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -8) devuelve 2014-01-07T14:00:00

add-hours-to-dateTime [altova:]

altova:add-hours-to-dateTime(FechaHora *as xs:dateTime*, **Horas** *as xs:integer*)
como **xs:dateTime** XP3.1 XQ3.1
Añade una duración en horas a un valor de fecha y hora. El segundo argumento es el número de horas que se deben añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), 10) devuelve 2014-01-15T23:00:00

- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), -8) devuelve 2014-01-15T05:00:00

▼ add-minutes-to-dateTime [altova:]

altova:add-minutes-to-dateTime(FechaHora *as xs:dateTime*, **Minutos** *as xs:integer*) como **xs:dateTime** XP3.1 XQ3.1

Añade una duración en minutos a un valor de fecha y hora. El segundo argumento es el número de minutos que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

▣ *Ejemplos*

- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), 45) devuelve 2014-01-15T14:55:00
- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), -5) devuelve 2014-01-15T14:05:00

▼ add-seconds-to-dateTime [altova:]

altova:add-seconds-to-dateTime(FechaHora *as xs:dateTime*, **Segundos** *as xs:integer*) como **xs:dateTime** XP3.1 XQ3.1

Añade una duración en segundos a un valor de fecha y hora. El segundo argumento es el número de segundos que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

▣ *Ejemplos*

- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), 20) devuelve 2014-01-15T14:00:30
- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), -5) devuelve 2014-01-15T14:00:05

[Subir]

Recuperar duraciones y aplicarles formato XP3.1 XQ3.1

Estas funciones añaden una duración a **xs:date** y devuelven **xs:date**. El tipo xs:date tiene el formato SSAA-MM-DD.

▼ format-duration [altova:]

altova:format-duration(Duración *como xs:duration*, **Imagen** *como xs:string*) como **xs:string** XP3.1 XQ3.1

Aplica formato a una duración, que se suministra como primer argumento, en base a la cadena de imagen dada como segundo argumento. El resultado es una cadena de texto cuyo formato se ajusta a la cadena de imagen.

▣ *Ejemplos*

© 2018 Altova GmbH

- **altova:format-duration**(xs:duration("P2DT2H53M11.7S"), "Días:[D01] Horas:[H01] Minutos:[m01] Segundos:[s01] Fracciones:[f0]") devuelve "Días:02 Horas:02 Minutos:53 Segundos:11 Fracciones:7"
- **altova:format-duration**(xs:duration("P3M2DT2H53M11.7S"), "Meses:[M01] Días:[D01] Horas:[H01] Minutos:[m01]") devuelve "Meses:03 Días:02 Horas:02 Minutos:53"

▼ parse-duration [altova:]

altova:parse-duration(CadenaEntrada *como xs:string*, **Imagen** *como xs:string*) como **xs:duration** XP3.1 XQ3.1

Toma una cadena con patrón como primer argumento y una cadena de imagen como segundo argumento. La cadena de entrada se analiza en base a la cadena de imagen y se devuelve un xs:duration.

Ejemplos

- **altova:parse-duration**("Días:02 Horas:02 Minutos:53 Segundos:11 Fracciones:7"), "Días:[D01] Horas:[H01] Minutes:[m01] Segundos:[s01] Fracciones:[f0]") devuelve "P2DT2H53M11.7S"
- **altova:parse-duration**("Meses:03 Días:02 Horas:02 Minutos:53 Segundos:11 Fracciones:7", "Meses:[M01] Días:[D01] Horas:[H01] Minutos:[m01]") devuelve "P3M2DT2H53M"

[Subir]

Agregar una duración a xs:date XP3.1 XQ3.1

Estas funciones agregan una duración a **xs:date** y devuelven **xs:date**. El tipo xs:date tiene el formato SSAA-MM-DD.

▼ add-years-to-date [altova:]

altova:add-years-to-date(Fecha *as xs:date*, **Años** *as xs:integer*) como **xs:date** XP3.1 XQ3.1

Añade una duración en años a una fecha. El segundo parámetro es el número de años que se debe añadir a la fecha dada como primer argumento. El resultado es de tipo xs:date.

Ejemplos

- **altova:add-years-to-date**(xs:date("2014-01-15"), 10) devuelve 2024-01-15
- **altova:add-years-to-date**(xs:date("2014-01-15"), -4) devuelve 2010-01-15

▼ add-months-to-date [altova:]

altova:add-months-to-date(Fecha *as xs:date*, **Meses** *as xs:integer*) como **xs:date** XP3.1 XQ3.1

Añade una duración en meses a una fecha. El segundo argumento es el número de meses que se debe añadir a la fecha dada como primer argumento. El resultado es de tipo xs:date.

Ejemplos

- **altova:add-months-to-date**(xs:date("2014-01-15"), 10) devuelve 2014-11-15

- **altova:add-months-to-date**(xs:date("2014-01-15"), -2) devuelve 2013-11-15

▼ add-days-to-date [altova:]

altova:add-days-to-date(Fecha *as xs:date*, **Días** *as xs:integer***)** como **xs:date**
XP3.1 XQ3.1
Añade una duración en días a una fecha. El segundo argumento es el número de días que se deben añadir a la fecha dad como primer argumento. El resultado es de tipo xs:date.

⊟ *Ejemplos*

- **altova:add-days-to-date**(xs:date("2014-01-15"), 10) devuelve 2014-01-25
- **altova:add-days-to-date**(xs:date("2014-01-15"), -8) devuelve 2014-01-07

[Subir]

Agregar una duración a xs:time XP3.1 XQ3.1

Estas funciones agregan una duración a **xs:time** y devuelven **xs:time**. El tipo xs:time tiene un formato léxico de este tipo hh:mm:ss.sss. Si quiere, puede añadir un sufijo de zona horaria. La letra Z indica (UTC). Las demás zonas horarias se representan con la diferencia que hay entre ellas y la zona UTC: +hh:mm o -hh:mm. Si falta el valor de zona horaria, se entiende que se desconoce (no se da por hecho que es UTC)

▼ add-hours-to-time [altova:]

altova:add-hours-to-time(Hora *as xs:time*, **Horas** *as xs:integer***)** como **xs:time**
XP3.1 XQ3.1
Añade una duración en horas a una hora. El segundo argumento es el número de horas que se debe añadir a la hora dada como primer argumento. El resultado es de tipo xs:time.

⊟ *Ejemplos*

- **altova:add-hours-to-time**(xs:time("11:00:00"), 10) devuelve 21:00:00
- **altova:add-hours-to-time**(xs:time("11:00:00"), -7) devuelve 04:00:00

▼ add-minutes-to-time [altova:]

altova:add-minutes-to-time(Hora *as xs:time*, **Minutos** *as xs:integer***)** como **xs:time** XP3.1 XQ3.1
Añade una duración en minutos a una hora. El segundo argumento es el número de minutos que se debe añadir a la hora dada como primer argumento. El resultado es de tipo xs:time.

⊟ *Ejemplos*

- **altova:add-minutes-to-time**(xs:time("14:10:00"), 45) devuelve 14:55:00
- **altova:add-minutes-to-time**(xs:time("14:10:00"), -5) devuelve 14:05:00

© 2018 Altova GmbH

▼ add-seconds-to-time [altova:]

altova:add-seconds-to-time(Hora *as xs:time*, **Segundos** *as xs:integer***)** como **xs:time** XP3.1 XQ3.1

Añade una duración en segundos a una hora. El segundo argumento es el número de segundos que se debe añadir a la hora dada como primer argumento. El resultado es de tipo `xs:time`. El componente **Segundos** puede estar comprendido entre `0` y `59.999`.

⊟ *Ejemplos*

- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20) devuelve 14:00:20
- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20.895) devuelve 14:00:20.895

[Subir]

Quitar la parte de zona horaria de los tipos de datos date/time XP3.1 XQ3.1

Estas funciones quitan la zona horaria de los valores **xs:dateTime**, **xs:date** o **xs:time** actuales. Tenga en cuenta que la diferencia entre `xs:dateTime` y `xs:dateTimeStamp` es que en esta última la parte de zona horaria es obligatoria (mientras que en la primera es opcional). Es decir, el formato de un valor `xs:dateTimeStamp` puede ser `SSAA-MM-DDThh:mm:ss.sss±hh:mm` o `SSAA-MM-DDThh:mm:ss.sssZ`. Si la fecha y la hora se leen del reloj del sistema como `xs:dateTimeStamp`, la función `current-dateTime-no-TZ()` se puede usar para quitar la zona horaria.

▼ current-dateTime-no-TZ [altova:]

altova:current-dateTime-no-TZ() como **xs:dateTime** XP3.1 XQ3.1

Esta función no toma ningún argumento. Quita la parte de zona horaria de `current-dateTime()` (que es la fecha y hora actual según el reloj del sistema) y devuelve un valor de tipo `xs:dateTime`.

⊟ *Ejemplos*

Si la fecha y hora actual es **2014-01-15T14:00:00+01:00**:

- **altova:current-dateTime-no-TZ**() devuelve 2014-01-15T14:00:00

▼ current-date-no-TZ [altova:]

altova:current-date-no-TZ() como **xs:date** XP3.1 XQ3.1

Esta función no toma ningún argumento. Quita la parte de zona horaria de la función `current-date()` (que es la fecha actual según el reloj del sistema) y devuelve un valor de tipo `xs:date`.

⊟ *Ejemplos*

Si la fecha actual es **2014-01-15+01:00**:

- **altova:current-date-no-TZ**() devuelve 2014-01-15

▼ current-time-no-TZ [altova:]

altova:current-time-no-TZ() as **xs:time** XP3.1 XQ3.1
Esta función no toma ningún argumento. Quita la parte de zona horaria de `current-time()` (que es la hora actual según el reloj del sistema) y devuelve un valor de tipo `xs:time`.

⊟ *Ejemplos*

Si la hora actual es **14:00:00+01:00**:

- **altova:current-time-no-TZ**() devuelve `14:00:00`

[Subir]

Obtener el día de la semana de `xs:dateTime` o `xs:date` XP3.1 XQ3.1

Estas funciones obtienen el día de la semana (como entero) de `xs:dateTime` o `xs:date`. Los días de la semana se numeran del `1` al `7` (usando el formato EE UU, es decir `Domingo =1`). En el formato europeo la semana empieza el lunes (es decir, `Lunes=1`). Para establecer el formato EE UU (`Domingo=1`) use el entero `0` allí donde se acepte un entero para indicar el formato.

▼ weekday-from-dateTime [altova:]

altova:weekday-from-dateTime(FechaHora *as xs:dateTime*) como **xs:integer** XP3.1 XQ3.1
Toma una fecha con hora como único argumento y devuelve el día de la semana de la fecha dada como número entero. Los días de la semana se numeran del 1 al 7 empezando por `Domingo=1`. Si necesita usar el formato europeo (donde `Lunes=1`), utilice la otra firma de esta función (*ver más abajo*).

⊟ *Ejemplos*

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00")) devuelve `2`, lo cual significa "Lunes".

altova:weekday-from-dateTime(DateTime *as xs:dateTime*, **Formato** *as xs:integer*) como **xs:integer** XP3.1 XQ3.1
Toma una fecha con hora como primer argumento y devuelve el día de la semana de la fecha dada como número entero. Los días de la semana se numeran del 1 al 7 empezando por `Lunes=1`. Si el segundo argumento (**Formato**) es `0`, entonces los días de la semana se numeran del 1 al 7 empezando por `Domingo=1`. Si el segundo argumento es un entero distinto de `0`, entonces `Lunes=1`. Si falta el segundo argumento, la función se lee como en la firma anterior (*ver más arriba*).

⊟ *Ejemplos*

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 1) devuelve `1`, lo cual significa "Lunes".

© 2018 Altova GmbH

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 4) devuelve 1, lo cual significa "Lunes".
- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 0) devuelve 2, lo cual significa "Lunes".

▼ weekday-from-date [altova:]

altova:weekday-from-date(Fecha *as xs:date***)** como **xs:integer** XP3.1 XQ3.1
Toma una fecha como único argumento y devuelve el día de la semana de la fecha dada como número entero. Los días de la semana se numeran del 1 al 7 empezando por Domingo=1. Si necesita usar el formato europeo (donde Lunes=1), utilice la otra firma de esta función (*ver más abajo*).

▣ *Ejemplos*

- **altova:weekday-from-date**(xs:date("2014-02-03+01:00")) devuelve 2, lo cual significa "Lunes".

altova:weekday-from-date(Fecha *as xs:date*, **Formato** *as xs:integer***)** como **xs:integer** XP3.1 XQ3.1
Toma una fecha como primer argumento y devuelve el día de la semana de la fecha dada como número entero. Los días de la semana se numeran del 1 al 7 empezando por Lunes=1. Si el segundo argumento (**Formato**) es 0, entonces los días de la semana se numeran del 1 al 7 empezando por Domingo=1. Si el segundo argumento es un entero distinto de 0, entonces Lunes=1. Si falta el segundo argumento, la función se lee como en la firma anterior (*ver más arriba*).

▣ *Ejemplos*

- **altova:weekday-from-date**(xs:date("2014-02-03"), 1) devuelve 1, lo cual significa "Lunes"
- **altova:weekday-from-date**(xs:date("2014-02-03"), 4) devuelve 1, lo cual significa "Lunes"
- **altova:weekday-from-date**(xs:date("2014-02-03"), 0) devuelve 2, lo cual significa "Lunes"

[Subir]

Devolver el número de semana de xs:dateTime o xs:date XP2 XQ1 XP3.1 XQ3.1

Estas funciones devuelven el número de semana (como número entero) de xs:dateTime o xs:date. El número de la semana está disponible en el formato de calendario estadounidense, europeo e islámico. La razón de que los números de semana difieran en cada uno de estos calendarios es que en cada uno de ellos se considera un día diferente para el inicio de la semana (p. ej. en el formato estadounidense el primer día de la semana es el domingo).

▼ weeknumber-from-date [altova:]

altova:weeknumber-from-date(Fecha *como xs:date*, **Calendario** *como xs:integer***)**

como **xs:integer** XP2 XQ1 XP3.1 XQ3.1

Devuelve como número entero el número de semana del argumento **Fecha** dado. El segundo argumento (**Calendario**) indica el sistema de calendario que se debe seguir.
Estos son los valores permitidos para el argumento **Calendario**:

- **0 = Calendario estadounidense** (la semana comienza el domingo)
- **1 = Calendario estándar ISO o europeo** (la semana comienza el lunes)
- **2 = Calendario islámico** (la semana comienza el sábado)

El valor predeterminado es **0**.

Ejemplos

- `altova:weeknumber-from-date(xs:date("2014-03-23"), 0)` devuelve `13`
- `altova:weeknumber-from-date(xs:date("2014-03-23"), 1)` devuelve `12`
- `altova:weeknumber-from-date(xs:date("2014-03-23"), 2)` devuelve `13`
- `altova:weeknumber-from-date(xs:date("2014-03-23") )` devuelve `13`

El día de la fecha de los ejemplos anteriores (`2014-03-23`) es un domingo. Por tanto, en este caso, el calendario estadounidense y el islámico van una semana por delante del calendario europeo.

weeknumber-from-dateTime [altova:]

altova:weeknumber-from-dateTime(FechaHora *como xs:dateTime*, **Calendario** *como xs:integer***)** como **xs:integer** XP2 XQ1 XP3.1 XQ3.1

Devuelve como entero el día de la semana del argumento **FechaHora** dado. El segundo argumento (**Calendario**) indica el sistema de calendario que se debe seguir.
Estos son los valores permitidos para el argumento **Calendario**:

- **0 = Calendario estadounidense** (la semana comienza el domingo)
- **1 = Calendario estándar ISO o europeo** (la semana comienza el lunes)
- **2 = Calendario islámico** (la semana comienza el sábado)

El valor predeterminado es **0**.

Ejemplos

- `altova:weeknumber-from-dateTime(xs:dateTime("2014-03-23T00:00:00"), 0)` devuelve `13`
- `altova:weeknumber-from-dateTime(xs:dateTime("2014-03-23T00:00:00"), 1)` devuelve `12`
- `altova:weeknumber-from-dateTime(xs:dateTime("2014-03-23T00:00:00"), 2)` devuelve `13`
- `altova:weeknumber-from-dateTime(xs:dateTime("2014-03-23T00:00:00") )` devuelve `13`

El día de `dateTime` de los ejemplos anteriores (`2014-03-23T00:00:00`) es un domingo. Por tanto, en este caso, el calendario estadounidense y el islámico van una semana por delante del calendario europeo.

© 2018 Altova GmbH

[Subir]

Generar tipos de datos de fecha, hora y duración a partir de sus componentes léxicos XP3.1 XQ3.1

Estas funciones toman los componentes léxicos de los tipos de datos `xs:date`, `xs:time` y `xs:duration` como argumentos de entrada y los combinan para generar el tipo de datos correspondiente.

▼ build-date [altova:]

altova:build-date(Año *as xs:integer*, **Mes** *as xs:integer*, **Fecha** *as xs:integer*) como **xs:date** XP3.1 XQ3.1

Los argumentos son el año, el mes y la fecha respectivamente. Se combinan para generar un valor de tipo `xs:date`. Los valores de los enteros deben estar en el intervalo de esa fecha en particular. Por ejemplo, el segundo argumento (para el mes) no puede ser mayor que `12`.

⊟ *Ejemplos*

- **altova:build-date**(2014, 2, 03) devuelve 2014-02-03

▼ build-time [altova:]

altova:build-time(Horas *as xs:integer*, **Minutos** *as xs:integer*, **Segundos** *as xs:integer*) como **xs:time** XP3.1 XQ3.1

El primer, segundo y tercer argumentos son la hora (`0` - `23`), los minutos (`0` - `59`) y los segundos (`0` - `59`) respectivamente. Se combinan para generar un valor de tipo `xs:time`. Los valores de los enteros deben estar dentro del intervalo correcto de esa parte de tiempo concreta. Por ejemplo, el segundo argumento (`Minutos`) no puede ser mayor que `59`. Para añadir la parte de uso horario al valor, use la firma que aparece más abajo.

⊟ *Ejemplos*

- **altova:build-time**(23, 4, 57) devuelve 23:04:57

▼ build-duration [altova:]

altova:build-duration(Años *as xs:integer*, **Meses** *as xs:integer*) como **xs:yearMonthDuration** XP3.1 XQ3.1

Toma dos argumentos para generar un valor de tipo `xs:yearMonthDuration`. El primer argumento da la parte `Years` del valor de duración, mientras que el segundo da la parte `Months`. Si el segundo (`Months`) es mayor o igual que `12`, el entero se divide por 12. El cociente se añade al primer argumento para aportar la parte `Years` del valor de duración, mientras que el resto (de la división) da la parte `Months`. Para generar una duración de tipo `xs:dayTimeDuration`, consulte la firma siguiente.

⊟ *Ejemplos*

- **altova:build-duration**(2, 10) devuelve P2Y10M
- **altova:build-duration**(14, 27) devuelve P16Y3M
- **altova:build-duration**(2, 24) devuelve P4Y

altova:build-duration(Días *as xs:integer*, **Horas** *as xs:integer*, **Minutos** *as xs:integer*, **Segundos** *as xs:integer*) como **xs:dayTimeDuration** XP3.1 XQ3.1
Toma cuatro argumentos y los combina para generar un valor de tipo `xs:dayTimeDuration`. El primer argumento da la parte `Days` del valor de duración, el segundo, el tercero y el cuarto dan las partes `Hours`, `Minutes` y `Seconds` respectivamente. Los tres argumentos de tiempo se convierten a un valor equivalente en cuanto a la unidad mayor siguiente y el resultado se utiliza para calcular el valor total de la duración. Por ejemplo, 72 segundos se convierte en `1M+12S` (1 minuto y 12 segundos) y este valor se usa para calcular el valor total de la duración. Para generar una duración de tipo `xs:yearMonthDuration`, consulte la firma anterior.

Ejemplos

- `altova:build-duration(2, 10, 3, 56)` devuelve `P2DT10H3M56S`
- `altova:build-duration(1, 0, 100, 0)` devuelve `P1DT1H40M`
- `altova:build-duration(1, 0, 0, 3600)` devuelve `P1DT1H`

[Subir]

Construir tipos de datos date, dateTime y time a partir de una cadena de entrada XP2 XQ1 XP3.1 XQ3.1

Estas funciones toman cadenas como argumentos y construyen tipos de datos `xs:date`, `xs:dateTime` o `xs:time`. La cadena de entrada se analiza para los componentes del tipo de datos en función del argumento patrón dado.

parse-date [altova:]

altova:parse-date(Fecha *como xs:string*, **PatrónFecha** *como xs:string*) como **xs:date** XP2 XQ1 XP3.1 XQ3.1
Devuelve la cadena de entrada **Fecha** como valor **xs:date**. El segundo argumento (**PatrónFecha**) indica el patrón (secuencia de componentes) de la cadena de entrada. El argumento **PatrónFecha** se describe con los especificadores que aparecen a continuación y con cualquier separador de componentes (consulte los ejemplos más abajo).

D	Día
M	Mes
Y	Año

El patrón **PatrónFecha** debe coincidir con el patrón de **Fecha**. Como el resultado es de tipo **xs:date**, el resultado siempre tendrá el formato léxico **YYYY-MM-DD**.

Ejemplos

- `altova:parse-date(xs:string("09-12-2014"), "[D]-[M]-[Y]")` devuelve `2014-12-09`
- `altova:parse-date(xs:string("09-12-2014"), "[M]-[D]-[Y]")` devuelve `2014-09-12`
- `altova:parse-date("06/03/2014", "[M]/[D]/[Y]")` devuelve `2014-06-03`
- `altova:parse-date("06 03 2014", "[M] [D] [Y]")` devuelve `2014-06-03`

© 2018 Altova GmbH

- **altova:parse-date**("6 3 2014", "[M] [D] [Y]") devuelve 2014-06-03

▼ parse-dateTime [altova:]

altova:parse-dateTime(FechaHora *como xs:string*, **PatrónFechaHora** *como xs:string*) como **xs:dateTime** XP2 XQ1 XP3.1 XQ3.1

Devuelve la cadena de entrada **FechaHora** como valor **xs:dateTime**. El segundo argumento (**PatrónFechaHora**) indica el patrón (secuencia de componentes) de la cadena de entrada. El argumento **PatrónFechaHora** se describe con los especificadores que aparecen a continuación y con cualquier separador de componentes (consulte los ejemplos más abajo).

D	Día
M	Mes
Y	Año
H	Hora
m	minutos
s	segundos

El patrón **PatrónFechaHora** debe coincidir con el patrón de **FechaHora**. Como el resultado es de tipo **xs:dateTime**, el resultado siempre tendrá el formato léxico **YYYY-MM-DDTHH:mm:ss**.

▣ *Ejemplos*

- **altova:parse-dateTime**(xs:string("09-12-2014 13:56:24"), "[M]-[D]-[Y] [H]:[m]:[s]") devuelve 2014-09-12T13:56:24
- **altova:parse-dateTime**("time=13:56:24; date=09-12-2014", "time=[H]:[m]:[s]; date=[D]-[M]-[Y]") devuelve 2014-12-09T13:56:24

▼ parse-time [altova:]

altova:parse-time(Hora *como xs:string*, **PatrónHora** *como xs:string*) como **xs:time** XP2 XQ1 XP3.1 XQ3.1

Devuelve la cadena de entrada **Hora** como valor **xs:time**. El segundo argumento (**PatrónHora**) indica el patrón (secuencia de componentes) de la cadena de entrada. El argumento **PatrónHora** se describe con los especificadores que aparecen a continuación y con cualquier separador de componentes (consulte los ejemplos más abajo).

H	Hora
m	minutos
s	segundos

El patrón **PatrónHora** debe coincidir con el patrón de **Hora**. Como el resultado es de tipo **xs:time**, el resultado siempre tendrá el formato léxico **HH:mm:ss**.

▣ *Ejemplos*

- **altova:parse-time**(xs:string("13:56:24"), "[H]:[m]:[s]") devuelve

13:56:24
- **altova:parse-time**("13-56-24", "[H]-[m]") devuelve 13:56:00
- **altova:parse-time**("time=13h56m24s", "time=[H]h[m]m[s]s") devuelve 13:56:24
- **altova:parse-time**("time=24s56m13h", "time=[s]s[m]m[H]h") devuelve 13:56:24

[Subir]

Funciones para calcular la edad XP3.1 XQ3.1

Estas funciones devuelven la edad que se calcula obteniendo la diferencia (i) entre la fecha del argumento de entrada y la fecha actual o (ii) entre las fechas de los dos argumentos de entrada. La función **age** devuelve la edad en años, mientras que la función **age-details** devuelve la edad en forma de una secuencia de tres enteros (años, meses y días).

▼ age [altova:]

altova:age(FechaInicio *as xs:date***)** como **xs:integer** XP3.1 XQ3.1

Devuelve un entero que es la edad *en años* de algún objeto, contando a partir de la fecha de inicio dada como argumento y hasta la fecha actual (tomada del reloj del sistema). Si el argumento de entrada es un año o más después que la fecha actual, el valor devuelto será negativo.

⊟ *Ejemplos*

Si la fecha actual es **2014-01-15**:

- **altova:age**(xs:date("2013-01-15")) devuelve 1
- **altova:age**(xs:date("2013-01-16")) devuelve 0
- **altova:age**(xs:date("2015-01-15")) devuelve -1
- **altova:age**(xs:date("2015-01-14")) devuelve 0

altova:age(FechaInicio *as xs:date*, **FechaFinal** *as xs:date***)** como **xs:integer** XP3.1 XQ3.1

Devuelve un entero que es la edad *en años* de algún objeto, contando a partir de la fecha de inicio dada como primer argumento y hasta la fecha dada como segundo argumento. El valor devuelto será negativo si el primer argumento es un año o más después que el segundo argumento.

⊟ *Ejemplos*

- **altova:age**(xs:date("2000-01-15"), xs:date("2010-01-15")) devuelve 10
- **altova:age**(xs:date("2000-01-15"), current-date()) devuelve 14 si la fecha actual es 2014-01-15
- **altova:age**(xs:date("2014-01-15"), xs:date("2010-01-15")) devuelve -4

© 2018 Altova GmbH

▼ age-details [altova:]

altova:age-details(FechaEntrada *as xs:date***)** como **(xs:integer)*** XP3.1 XQ3.1
Devuelve tres enteros que son los años, meses y días respectivamente que hay entre la fecha dada como argumento y la fecha actual (tomada del reloj del sistema). La suma del valor devuelto nos da el tiempo total transcurrido entre ambas fechas (entre la fecha dada y la fecha actual). La fecha de entrada puede tener un valor anterior o posterior a la fecha actual, pero esto no se indica en el valor devuelto por medio de un signo negativo o positivo. El valor devuelto siempre es positivo.

⊟ *Ejemplos*

Si la fecha actual es **2014-01-15**:

- `altova:age-details(xs:date("2014-01-16"))` devuelve `(0 0 1)`
- `altova:age-details(xs:date("2014-01-14"))` devuelve `(0 0 1)`
- `altova:age-details(xs:date("2013-01-16"))` devuelve `(1 0 1)`
- `altova:age-details(current-date())` devuelve `(0 0 0`

altova:age-details(Fecha1 *as xs:date*, **Fecha2** *as xs:date***)** como **(xs:integer)*** XP3.1 XQ3.1
Devuelve tres enteros que son los años, meses y días que hay entre las dos fechas dadas por los argumentos. La suma del valor devuelto nos da el tiempo total transcurrido entre las dos fechas de entrada. Da igual cuál de las dos fechas se da como primer argumento, la más antigua o la más reciente. El valor devuelto no indica si la fecha de entrada es anterior o posterior a la fecha actual. Es decir, el valor devuelto siempre es positivo.

⊟ *Ejemplos*

- `altova:age-details(xs:date("2014-01-16"), xs:date("2014-01-15"))` devuelve `(0 0 1)`
- `altova:age-details(xs:date("2014-01-15"), xs:date("2014-01-16"))` devuelve `(0 0 1)`

[Subir]

19.1.2.1.3 Funciones XPath/XQuery: de geoubicación

Las funciones de extensión XPath/XQuery de geoubicación son compatibles con la versión actual de MapForce y se pueden utilizar en (i) expresiones XPath en contextos XSLT o (ii) expresiones XQuery en documentos XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **http://www.altova.com/xslt-extensions** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto

para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath *(en expresiones XPath en XSLT):*	XP1 XP2 XP3.1
Funciones XSLT *(en expresiones XPath en XSLT):*	XSLT1 XSLT2 XSLT3
Funciones XQuery *(en expresiones XQuery en XQuery):*	XQ1 XQ3.1

▼ parse-geolocation [altova:]

altova:parse-geolocation(CadenaEntradaGeoubicación *como xs:string***)** como **xs:decimal+** XP3.1 XQ3.1

Analiza el argumento `CadenaEntradaGeoubicación` y devuelve la latitud y la longitud (en ese orden) de la geoubicación en forma de secuencia de dos elementos `xs:decimal`. Más abajo puede ver en qué formatos se puede suministrar la cadena de entrada de la geoubicación.

Nota: la función `image-exif-data` y el atributo `@Geolocation` de los metadatos Exif se pueden utilizar para suministrar la cadena de entrada de la geoubicación (*ver ejemplos*).

⊟ *Ejemplos*

- `altova:parse-geolocation("33.33 -22.22")` devuelve la secuencia de dos `xs:decimals (33.33, 22.22)`
- `altova:parse-geolocation("48°51'29.6""N 24°17'40.2""")` devuelve la secuencia de dos `xs:decimals (48.8582222222222, 24.2945)`
- `altova:parse-geolocation('48°51''29.6"N 24°17''40.2"')` devuelve la secuencia de dos `xs:decimals (48.8582222222222, 24.2945)`
- `altova:parse-geolocation( image-exif-data(//MisImágenes/Imagen20141130.01)/@Geolocation )` devuelve una secuencia de dos `xs:decimals`

⊟ *Formato de las cadenas de entrada de geoubicaciones:*

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `W/E`)
  `D°M'S.SS"N/S D°M'S.SS"W/E`

© 2018 Altova GmbH

Ejemplo: **33°55'11.11"N 22°44'55.25"W**

- Grados, minutos y segundos decimales + prefijo (+/-). El signo + para (**N/W**) es opcional
  +/-D°M'S.SS" +/-D°M'S.SS"
  Ejemplo: **33°55'11.11" -22°44'55.25"**

- Grados y minutos decimales + orientación como sufijo (N/S, W/E)
  D°M.MM'N/S D°M.MM'W/E
  Ejemplo: **33°55.55'N 22°44.44'W**

- Grados y minutos decimales + prefijo (+/-). El signo + para (**N/W**) es opcional
  +/-D°M.MM' +/-D°M.MM'
  Ejemplo: **+33°55.55' -22°44.44'**

- Grados decimales + orientación como sufijo (N/S, W/E)
  D.DDN/S D.DDW/E
  Ejemplo: **33.33N 22.22W**

- Grados decimales + prefijo (+/-). El signo + para (**N/W**) es opcional
  +/-D.DD +/-D.DD
  Ejemplo: **33.33 -22.22**

Ejemplos de combinación de formatos:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151°13'11.73"E

geolocation-distance-km [altova:]

altova:geolocation-distance-km(CadenaEntradaGeoubicación-1 *como xs:string*, **CadenaEntradaGeoubicación-2** *como xs:string*) como **xs:decimal** XP3.1 XQ3.1
Calcula la distancia en km que existe entre dos geoubicaciones. El formato que puede utilizarse para dar las cadenas de entrada aparece más abajo. Los valores de latitud están comprendidos entre +90 y -90 (N a S). Los valores de longitud están comprendidos entre +180 y -180 (E a W).

Nota: la función `image-exif-data` y el atributo de metadatos Exif `@Geolocation` pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- **`altova:geolocation-distance-km`**(`"33.33  -22.22"`, `"48°51'29.6""N  24° 17'40.2"""`) devuelve el `xs:decimal` `4183.08132372392`

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `W/E`)
  `D°M'S.SS"N/S  D°M'S.SS"W/E`
  Ejemplo: **`33°55'11.11"N  22°44'55.25"W`**

- Grados, minutos y segundos decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D°M'S.SS"  +/-D°M'S.SS"`
  Ejemplo: **`33°55'11.11"  -22°44'55.25"`**

- Grados y minutos decimales + orientación como sufijo (`N/S`, `W/E`)
  `D°M.MM'N/S  D°M.MM'W/E`
  Ejemplo: **`33°55.55'N  22°44.44'W`**

- Grados y minutos decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D°M.MM'  +/-D°M.MM'`
  Ejemplo: **`+33°55.55'  -22°44.44'`**

- Grados decimales + orientación como sufijo (`N/S`, `W/E`)
  `D.DDN/S  D.DDW/E`
  Ejemplo: **`33.33N  22.22W`**

- Grados decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D.DD  +/-D.DD`
  Ejemplo: **`33.33  -22.22`**

Ejemplos de combinación de formatos:
`33.33N  -22°44'55.25"`
`33.33  22°44'55.25"W`
`33.33  22.45`

© 2018 Altova GmbH

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (`GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151°13'11.73"E

geolocation-distance-mi [altova:]

altova:geolocation-distance-mi(CadenaEntradaGeoubicación-1 *como xs:string*, CadenaEntradaGeoubicación-2 *como xs:string***)** como **xs:decimal** XP3.1 XQ3.1

Calcula la distancia en millas que existe entre dos geoubicaciones. El formato que puede utilizarse para dar las cadenas de entrada aparece más abajo. Los valores de latitud están comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud están comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: la función `image-exif-data` y el atributo de metadatos Exif `@Geolocation` pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- **altova:geolocation-distance-mi**(`"33.33 -22.22"`, `"48°51'29.6""N 24° 17'40.2"""`) devuelve el `xs:decimal` `2599.40652340653`

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `W/E`)
  D°M'S.SS"N/S D°M'S.SS"W/E
  Ejemplo: **33°55'11.11"N 22°44'55.25"W**

- Grados, minutos y segundos decimales + prefijo (`+/-`). El signo + para (**N/W**)

es opcional
+/-D°M'S.SS" +/-D°M'S.SS"
Ejemplo: **33°55'11.11" -22°44'55.25"**

- Grados y minutos decimales + orientación como sufijo (N/S, W/E)
  D°M.MM'N/S D°M.MM'W/E
  Ejemplo: **33°55.55'N 22°44.44'W**

- Grados y minutos decimales + prefijo (+/-). El signo + para (**N/W**) es opcional
  +/-D°M.MM' +/-D°M.MM'
  Ejemplo: **+33°55.55' -22°44.44'**

- Grados decimales + orientación como sufijo (N/S, W/E)
  D.DDN/S D.DDW/E
  Ejemplo: **33.33N 22.22W**

- Grados decimales + prefijo (+/-). El signo + para (**N/W**) es opcional
  +/-D.DD +/-D.DD
  Ejemplo: **33.33 -22.22**

Ejemplos de combinación de formatos:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (GPSLatitude, GPSLatitudeRef, GPSLongitude, GPSLongitudeRef) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151°13'11.73"E

geolocation-within-polygon [altova:]

altova:geolocation-within-polygon(Geoubicación *como xs:string*, **((PuntoDePolígono** *como xs:string*)+)) como **xs:boolean** XP3.1 XQ3.1

Determina si **Geoubicación** (primer argumento) está dentro del área poligonal descrita por los argumentos **PuntoDePolígono**. Si los argumentos PuntoDePolígono no forman una figura cerrada (la figura se cierra cuando el primer y el último punto son el mismo), entonces el primer punto se añade implícitamente como último punto a fin de cerrar la figura. Todos los argumentos (Geoubicación y PuntoDePolígono+) se dan como cadenas de entrada de geoubicación (*formatos permitidos más abajo*). Si el argumento Geoubicación está dentro del área poligonal, entonces la función devuelve true(). De lo contrario, devuelve false(). Los valores de latitud están comprendidos entre +90 y -90 (N a S). Los valores de longitud

 © 2018 Altova GmbH

están comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: la función `image-exif-data` y el atributo de metadatos Exif `@Geolocation` pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

Ejemplos

- **`altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24", "58 -32"))`** devuelve `true()`
- **`altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48 24"))`** devuelve `true()`
- **`altova:geolocation-within-polygon("33 -22", ("58 -32", "-78 -55", "48°51'29.6""N  24°17'40.2""")`** devuelve `true()`

Formato de las cadenas de entrada de geoubicaciones:

La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `W/E`)
  `D°M'S.SS"N/S  D°M'S.SS"W/E`
  Ejemplo: **`33°55'11.11"N  22°44'55.25"W`**

- Grados, minutos y segundos decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D°M'S.SS"  +/-D°M'S.SS"`
  Ejemplo: **`33°55'11.11"  -22°44'55.25"`**

- Grados y minutos decimales + orientación como sufijo (`N/S`, `W/E`)
  `D°M.MM'N/S  D°M.MM'W/E`
  Ejemplo: **`33°55.55'N  22°44.44'W`**

- Grados y minutos decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D°M.MM'  +/-D°M.MM'`
  Ejemplo: **`+33°55.55'  -22°44.44'`**

- Grados decimales + orientación como sufijo (`N/S`, `W/E`)
  `D.DDN/S  D.DDW/E`
  Ejemplo: **`33.33N  22.22W`**

- Grados decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D.DD  +/-D.DD`

Ejemplo: **33.33 -22.22**

Ejemplos de combinación de formatos:
33.33N -22°44'55.25"
33.33 22°44'55.25"W
33.33 22.45

- *Atributo Exif de Altova: Geolocation*

  El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (`GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151°13'11.73"E

- geolocation-within-rectangle [altova:]

  altova:geolocation-within-rectangle(Geoubicación *como xs:string*, **ÁnguloRectángulo-1** *como xs:string*, **ÁnguloRectángulo-2** *como xs:string***)** como **xs:boolean** XP3.1 XQ3.1

  Determina si **Geoubicación** (primer argumento) está dentro del rectángulo definido por el segundo y el tercer argumento (**ÁnguloRectángulo-1** y **ÁnguloRectángulo-2**), que indican ángulos opuestos del rectángulo. Todos los argumentos de la función se dan como cadenas de entrada de geoubicación (*formatos permitidos más abajo*). Si el argumento `Geoubicación` está dentro del rectángulo, entonces la función devuelve `true()`. De lo contrario, devuelve `false()`. Los valores de latitud están comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud están comprendidos entre `+180` y `-180` (`E` a `W`).

  Nota: la función `image-exif-data` y el atributo de metadatos Exif `@Geolocation` pueden utilizarse para suministrar las cadenas de entrada de geoubicaciones.

  - *Ejemplos*

    - **altova:geolocation-within-rectangle(**"33 -22", "58 -32", "-48 24") devuelve `true()`
    - **altova:geolocation-within-rectangle(**"33 -22", "58 -32", "48 24") devuelve `false()`
    - **altova:geolocation-within-rectangle(**"33 -22", "58 -32", "48°51'29.6""S 24°17'40.2""") devuelve `true()`

  - *Formato de las cadenas de entrada de geoubicaciones:*

    La cadena de entrada de la geoubicación debe contener la latitud y la longitud (en ese orden) se paradas por un espacio en blanco. Ambas pueden estar en cualquier formato de los que se indican más abajo y puede combinar formatos distintos. Es decir, la

© 2018 Altova GmbH

latitud puede estar en un formato y la longitud en otro. Los valores de la latitud deben estar comprendidos entre `+90` y `-90` (`N` a `S`). Los valores de longitud deben estar comprendidos entre `+180` y `-180` (`E` a `W`).

Nota: si utiliza comillas simples o dobles para delimitar el argumento de la cadena de entrada, esto dará lugar a un conflicto con las comillas simples o dobles que se utilizan, respectivamente, para indicar los valores de los minutos y los segundos. Si esto ocurre, debe añadir caracteres de escape a las comillas utilizadas para los minutos y segundos (esto se hace duplicando las comillas.

- Grados, minutos y segundos decimales + orientación como sufijo (`N/S`, `W/E`)
  `D°M'S.SS"N/S  D°M'S.SS"W/E`
  Ejemplo: **`33°55'11.11"N  22°44'55.25"W`**

- Grados, minutos y segundos decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D°M'S.SS"  +/-D°M'S.SS"`
  Ejemplo: **`33°55'11.11"  -22°44'55.25"`**

- Grados y minutos decimales + orientación como sufijo (`N/S`, `W/E`)
  `D°M.MM'N/S  D°M.MM'W/E`
  Ejemplo: **`33°55.55'N  22°44.44'W`**

- Grados y minutos decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D°M.MM'  +/-D°M.MM'`
  Ejemplo: **`+33°55.55'  -22°44.44'`**

- Grados decimales + orientación como sufijo (`N/S`, `W/E`)
  `D.DDN/S  D.DDW/E`
  Ejemplo: **`33.33N  22.22W`**

- Grados decimales + prefijo (`+/-`). El signo + para (**`N/W`**) es opcional
  `+/-D.DD  +/-D.DD`
  Ejemplo: **`33.33  -22.22`**

Ejemplos de combinación de formatos:
`33.33N  -22°44'55.25"`
`33.33  22°44'55.25"W`
`33.33  22.45`

Atributo Exif de Altova: Geolocation

El motor XPath/XQuery de Altova genera el atributo personalizado **`Geolocation`** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (`GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151°13'11.73"E

[Subir]

19.1.2.1.4 Funciones XPath/XQuery: relacionadas con imágenes

Las funciones de extensión XPath/XQuery para trabajar con imágenes son compatibles con la versión actual de MapForce y se pueden utilizar en (i) expresiones XPath en contextos XSLT o (ii) expresiones XQuery en documentos XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** `http://www.altova.com/xslt-extensions` y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

▼ suggested-image-file-extension [altova:]

altova:suggested-image-file-extension(CadenaBase64 *como string***)** como **string?**
XP3.1 XQ3.1

Toma la codificación base64 de un archivo de imagen como argumento y devuelve la extensión de archivo de la imagen registrada en la codificación base64 de la imagen. El valor devuelto es una sugerencia basada en la información sobre el tipo de imagen disponible en la codificación. Si esta información no está disponible, entonces devuelve una cadena vacía. Esta función es muy práctica a la hora de guardar una imagen base64 como archivo y recuperar de forma dinámica una extensión de archivo adecuada.

⊟ *Ejemplos*

- **altova:suggested-image-file-extension**(/MisImágenes/TeléfonoMóvil/Imagen20141130.01) devuelve `'jpg'`
- **altova:suggested-image-file-extension**($XML1/Personal/Persona/@photo) devuelve `''`

 © 2018 Altova GmbH

En los ejemplos anteriores, se da por hecho que los nodos suministrados como argumento de la función contienen una imagen codificada en base64. El primer ejemplo recupera `jpg` como tipo de imagen y como extensión de archivo. En el segundo ejemplo, la codificación base64 dada no ofrece información sobre la extensión del archivo.

▼ image-exif-data [altova:]

altova:image-exif-data(CadenaBinariaBase64 *como string***)** como **element?** XP3.1 XQ3.1

Toma una imagen JPEG codificada en base64 como argumento y devuelve un elemento llamado **Exif** que contiene los metadatos Exif de la imagen. Los metadatos Exif se crean como pares atributo-valor del elemento `Exif`. El nombre de los atributos son las etiquetas de datos Exif encontradas en la codificación base64. La lista de etiquetas Exif aparece más abajo. Si en lo datos Exif hay etiquetas de terceros, estas etiquetas y sus valores también se devuelven en un par atributo-valor. Además de las etiquetas de metadatos Exif estándar (*lista más abajo*), también se generan pares atributo-valor de Altova. Estos atributos Exif de Altova también se enumeran más abajo.

⊟ *Ejemplos*

- Para acceder a un atributo, utilice la función de esta manera:
  ```
  image-exif-data(//MisImágenes/Imagen20141130.01)/@GPSLatitude
  image-exif-data(//MisImágenes/Imagen20141130.01)/@Geolocation
  ```
- Para acceder a todos los atributos, utilice la función de esta manera:
  ```
  image-exif-data(//MisImágenes/Imagen20141130.01)/@*
  ```
- Para acceder al nombre de todos los atributos, utilice esta expresión:
  ```
  for $i in image-exif-data(//MisImágenes/Imagen20141130.01)/@* return name($i)
  ```
  Esto es muy práctico a la hora de averiguar el nombre de los atributos que devuelve la función.

⊟ *Atributo Exif de Altova: Geolocation*

El motor XPath/XQuery de Altova genera el atributo personalizado **Geolocation** a partir de las etiquetas de metadatos Exif estándar. Este atributo es una concatenación de cuatro etiquetas Exif (`GPSLatitude`, `GPSLatitudeRef`, `GPSLongitude`, `GPSLongitudeRef`) seguidas de unidades:

GPSLatitude	GPSLatitudeRef	GPSLongitude	GPSLongitudeRef	Geolocation
33 51 21.91	S	151 13 11.73	E	33°51'21.91"S 151°13'11.73"E

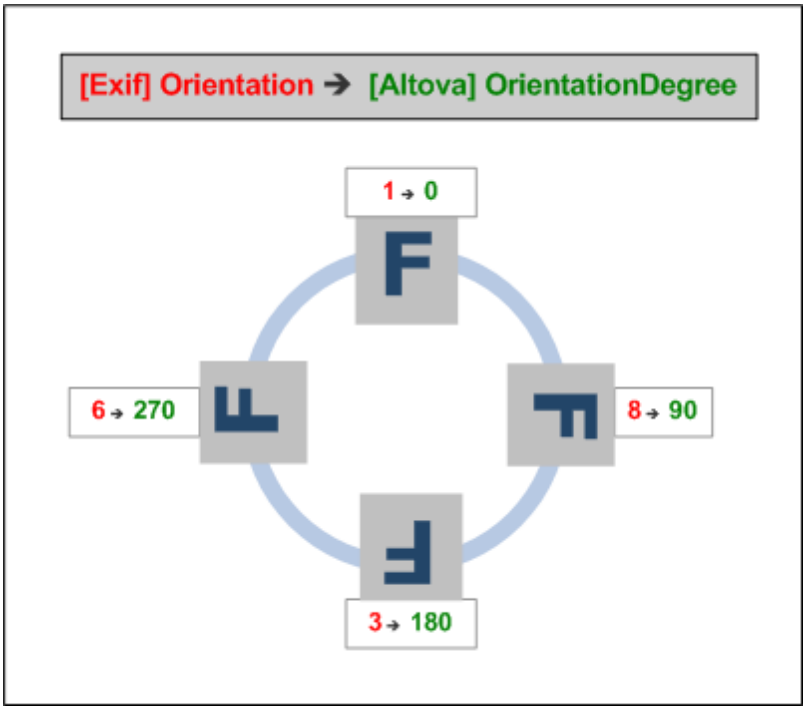
⊟ *Atributo Exif de Altova: OrientationDegree*

El motor XPath/XQuery de Altova genera el atributo personalizado **OrientationDegree** a partir de la etiqueta de metadatos Exif **Orientation**.

Este atributo transforma el valor entero de la etiqueta Exif `Orientation` (1, 8, 3 o 6) en

el correspondiente valor en grados (`0`, `90`, `180`, `270`), tal y como describe el diagrama más abajo.

Debe tener en cuenta que los valores `2`, `4`, `5`, `7` de `Orientation` no se pueden traducir. Estas orientaciones se obtienen invirtiendo la imagen `1` en su eje central vertical para obtener la imagen con un valor de `2` e invirtiendo después esta imagen por pasos de 90 grados en el sentido de las agujas del reloj para obtener los valores de `7`, `4` y `5`, respectivamente.



- *Lista de etiquetas Exif estándar*
  - `ImageWidth`
  - `ImageLength`
  - `BitsPerSample`
  - `Compression`
  - `PhotometricInterpretation`
  - `Orientation`
  - `SamplesPerPixel`
  - `PlanarConfiguration`
  - `YCbCrSubSampling`
  - `YCbCrPositioning`
  - `XResolution`
  - `YResolution`
  - `ResolutionUnit`
  - `StripOffsets`
  - `RowsPerStrip`
  - `StripByteCounts`
  - `JPEGInterchangeFormat`
  - `JPEGInterchangeFormatLength`

© 2018 Altova GmbH

- TransferFunction
- WhitePoint
- PrimaryChromaticities
- YCbCrCoefficients
- ReferenceBlackWhite
- DateTime
- ImageDescription
- Make
- Model
- Software
- Artist
- Copyright

- ExifVersion
- FlashpixVersion
- ColorSpace
- ComponentsConfiguration
- CompressedBitsPerPixel
- PixelXDimension
- PixelYDimension
- MakerNote
- UserComment
- RelatedSoundFile
- DateTimeOriginal
- DateTimeDigitized
- SubSecTime
- SubSecTimeOriginal
- SubSecTimeDigitized
- ExposureTime
- FNumber
- ExposureProgram
- SpectralSensitivity
- ISOSpeedRatings
- OECF
- ShutterSpeedValue
- ApertureValue
- BrightnessValue
- ExposureBiasValue
- MaxApertureValue
- SubjectDistance
- MeteringMode
- LightSource
- Flash
- FocalLength
- SubjectArea
- FlashEnergy
- SpatialFrequencyResponse
- FocalPlaneXResolution
- FocalPlaneYResolution
- FocalPlaneResolutionUnit
- SubjectLocation
- ExposureIndex
- SensingMethod
- FileSource

- SceneType
- CFAPattern
- CustomRendered
- ExposureMode
- WhiteBalance
- DigitalZoomRatio
- FocalLengthIn35mmFilm
- SceneCaptureType
- GainControl
- Contrast
- Saturation
- Sharpness
- DeviceSettingDescription
- SubjectDistanceRange
- ImageUniqueID

- GPSVersionID
- GPSLatitudeRef
- GPSLatitude
- GPSLongitudeRef
- GPSLongitude
- GPSAltitudeRef
- GPSAltitude
- GPSTimeStamp
- GPSSatellites
- GPSStatus
- GPSMeasureMode
- GPSDOP
- GPSSpeedRef
- GPSSpeed
- GPSTrackRef
- GPSTrack
- GPSImgDirectionRef
- GPSImgDirection
- GPSMapDatum
- GPSDestLatitudeRef
- GPSDestLatitude
- GPSDestLongitudeRef
- GPSDestLongitude
- GPSDestBearingRef
- GPSDestBearing
- GPSDestDistanceRef
- GPSDestDistance
- GPSProcessingMethod
- GPSAreaInformation
- GPSDateStamp
- GPSDifferential

[Subir]

© 2018 Altova GmbH

19.1.2.1.5 Funciones XPath/XQuery: numéricas

Las funciones de extensión numéricas de Altova pueden utilizarse en expresiones XPath y XQuery y ofrecen funciones adicionales para el procesamiento de datos. Estas funciones se pueden usar con los motores **XPath 3.0** y **XQuery 3.0** de Altova. Están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** `http://www.altova.com/xslt-extensions` y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

Funciones de numeración automática

▼ generate-auto-number [altova:]

altova:generate-auto-number(ID *como xs:string*, **EmpiezaPor** *como xs:double*, **Incremento** *como xs:double*, **RestaurarAlCambiar** *como xs:string*) como `xs:integer`
XP1 XP2 XQ1 XP3.1 XQ3.1

Genera un número cada vez que se llama a la función. El primer número, que se genera cuando se llama a la función por primera vez, viene dado por el argumento `EmpiezaPor`. Cada llamada posterior genera un número nuevo, que se incrementa en función del valor especificado en el argumento `Incremento`. De hecho, la función `generate-auto-number` crea un contador llamado como indique el argumento `ID` y este contador se incrementa cada vez que se llama a la función. Si el valor del argumento `RestaurarAlCambiar` cambia con respecto al valor que tenía en la llamada anterior, entonces el valor del número que se debe generar se restablece con el valor de `EmpiezaPor`. También puede restablecer la numeración automática con la función `altova:reset-auto-number`.

⊟ *Ejemplo*

- `altova:generate-auto-number("ChapterNumber", 1, 1, "SomeString")`

Devuelve un número cada vez que se llama a la función, empezando por `1` y con un incremento de `1` con cada llamada a función. Si el cuarto argumento continúa siendo `"SomeString"` en las llamadas posteriores, el incremento continuará. Cuando cambie

el valor del cuarto argumento, se restaura el valor 1 del contador (llamado `ChapterNumber`). El valor de `ChapterNumber` también se puede restaurar llamando a la función `altova:reset-auto-number("ChapterNumber")`.

▼ reset-auto-number [altova:]

altova:reset-auto-number(ID *como xs:string*) XP1 XP2 XQ1 XP3.1 XQ3.1

Esta función restaura el número del contador de numeración automática especificado en el argumento `ID`. El número se reemplaza con el número indicado en el argumento `EmpiezaPor` de la función `altova:generate-auto-number` que creó el contador especificado en el argumento `ID`.

▣ *Ejemplos*

- **altova:reset-auto-number**(`"ChapterNumber"`) restablece el número del contador de numeración automática llamado `ChapterNumber` que se creó con la función `altova:generate-auto-number`. El número se reemplaza con el valor del argumento `EmpiezaPor` de la función `altova:generate-auto-number` que creó `ChapterNumber`.

[Subir]

Funciones numéricas

▼ hex-string-to-integer [altova:]

altova:hex-string-to-integer(CadenaHex *as xs:string*) como **xs:integer** XP3.1 XQ3.1

Toma un argumento de cadena que es el equivalente Base-16 de un entero del sistema decimal (Base-10) y devuelve un entero decimal.

▣ *Ejemplos*

- **altova:hex-string-to-integer**(`'1'`) devuelve `1`
- **altova:hex-string-to-integer**(`'9'`) devuelve `9`
- **altova:hex-string-to-integer**(`'A'`) devuelve `10`
- **altova:hex-string-to-integer**(`'B'`) devuelve `11`
- **altova:hex-string-to-integer**(`'F'`) devuelve `15`
- **altova:hex-string-to-integer**(`'G'`) devuelve un error
- **altova:hex-string-to-integer**(`'10'`) devuelve `16`
- **altova:hex-string-to-integer**(`'01'`) devuelve `1`
- **altova:hex-string-to-integer**(`'20'`) devuelve `32`
- **altova:hex-string-to-integer**(`'21'`) devuelve `33`
- **altova:hex-string-to-integer**(`'5A'`) devuelve `90`
- **altova:hex-string-to-integer**(`'USA'`) devuelve un error

▼ integer-to-hex-string [altova:]

altova:integer-to-hex-string(Entero *as xs:integer*) como **xs:string** XP3.1 XQ3.1

Toma el argumento **Entero** y devuelve su equivalente Base-16 en forma de cadena.

© 2018 Altova GmbH

Ejemplos

- `altova:integer-to-hex-string(1)` devuelve `'1'`
- `altova:integer-to-hex-string(9)` devuelve `'9'`
- `altova:integer-to-hex-string(10)` devuelve `'A'`
- `altova:integer-to-hex-string(11)` devuelve `'B'`
- `altova:integer-to-hex-string(15)` devuelve `'F'`
- `altova:integer-to-hex-string(16)` devuelve `'10'`
- `altova:integer-to-hex-string(32)` devuelve `'20'`
- `altova:integer-to-hex-string(33)` devuelve `'21'`
- `altova:integer-to-hex-string(90)` devuelve `'5A'`

[Subir]

Funciones de formato numérico

generate-auto-number [altova:]

`altova:generate-auto-number(ID como xs:string, EmpiezaPor como xs:double, Incremento como xs:double, RestaurarAlCambiar como xs:string)` como `xs:integer`

XP1 XP2 XQ1 XP3.1 XQ3.1

Genera un número cada vez que se llama a la función. El primer número, que se genera cuando se llama a la función por primera vez, viene dado por el argumento `EmpiezaPor`. Cada llamada posterior genera un número nuevo, que se incrementa en función del valor especificado en el argumento `Incremento`. De hecho, la función `generate-auto-number` crea un contador llamado como indique el argumento `ID` y este contador se incrementa cada vez que se llama a la función. Si el valor del argumento `RestaurarAlCambiar` cambia con respecto al valor que tenía en la llamada anterior, entonces el valor del número que se debe generar se restablece con el valor de `EmpiezaPor`. También puede restablecer la numeración automática con la función `altova:reset-auto-number`.

Ejemplo

- `altova:generate-auto-number("ChapterNumber", 1, 1, "SomeString")`
  Devuelve un número cada vez que se llama a la función, empezando por `1` y con un incremento de `1` con cada llamada a función. Si el cuarto argumento continúa siendo `"SomeString"` en las llamadas posteriores, el incremento continuará. Cuando cambie el valor del cuarto argumento, se restaura el valor 1 del contador (llamado `ChapterNumber`). El valor de `ChapterNumber` también se puede restaurar llamando a la función `altova:reset-auto-number("ChapterNumber")`.

[Subir]

19.1.2.1.6 Funciones XPath/XQuery: de secuencia

Las funciones de extensión de Altova para trabajar con secuencias pueden utilizarse en expresiones XPath y XQuery y ofrecen funciones adicionales para el procesamiento de datos. Estas funciones se pueden usar con los motores **XPath 3.0** y **XQuery 3.0** de Altova. Están

disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** `http://www.altova.com/xslt-extensions` y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

▼ attributes [altova:]

altova:attributes(NombreAtributo *as xs:string***)** como **attribute()*** XP3.1 XQ3.1
Devuelve todos los atributos cuyo nombre local coincida con el nombre dado como argumento de entrada (`NombreAtributo`). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje `attribute::`.

Ejemplos

- **altova:attributes**(`"MiAtributo"`) devuelve `MiAtributo()*`

altova:attributes(NombreAtributo *as xs:string*, **OpcionesBúsqueda** *as xs:string***)** como **attribute()*** XP3.1 XQ3.1
Devuelve todos los atributos cuyo nombre local coincida con el nombre dado como argumento de entrada (`NombreAtributo`). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje `attribute::`. El segundo argumento es una cadena con marcas de búsqueda. Estas son las marcas disponibles:

r = habilita la búsqueda de expresiones regulares. En este caso, `NombreAtributo` debe ser una cadena de búsqueda de expresión regular;
i = la búsqueda no tiene en cuenta el uso de mayúsculas y minúsculas;
p = incluye el prefijo de espacio de nombres en la búsqueda. En este caso, `NombreAtributo` debe contener el prefijo de espacio de nombres (p. ej.: `MiAtributo`).

Las marcas pueden escribirse en cualquier orden y no hace falta utilizar todas. Si usa marcas no válidas, se genera un error. También puede usar una cadena vacía para el segundo argumento. Esto tiene el mismo efecto que usar solo el primer argumento. Sin embargo, no está permitido usar una secuencia vacía.

Ejemplos

© 2018 Altova GmbH

- **altova:attributes**("MiAtributo", "rip") devuelve MiAtributo()*
- **altova:attributes**("MiAtributo", "pri") devuelve MiAtributo()*
- **altova:attributes**("MiAtributo", "") devuelve MiAtributo()*
- **altova:attributes**("MiAtributo", "Rip") devuelve un error de marca desconocida.
- **altova:attributes**("MiAtributo",) devuelve un error diciendo que falta el segundo argumento.

▼ elements [altova:]

altova:elements(NombreElemento *as xs:string***)** como **elemento()*** XP3.1 XQ3.1
Devuelve todos los elementos cuyo nombre local coincida con el nombre dado como argumento de entrada (NombreElemento). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje child::.

▣ *Ejemplos*

- **altova:elements**("MiElemento") devuelve MiElemento()*

altova:elements(NombreElemento *as xs:string*, **OpcionesBúsqueda** *as xs:string***)** como **elemento()*** XP3.1 XQ3.1
Devuelve todos los elementos cuyo nombre local coincida con el nombre dado como argumento de entrada (NombreElemento). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje child::. El segundo argumento es una cadena con marcas de búsqueda. Estas son las marcas disponibles:

r = habilita la búsqueda de expresiones regulares. En este caso, NombreElemento debe ser una cadena de búsqueda de expresión regular;
i = la búsqueda no tiene en cuenta el uso de mayúsculas y minúsculas;
p = incluye el prefijo de espacio de nombres en la búsqueda. En este caso, NombreElemento debe contener el prefijo de espacio de nombres (p. ej.: MiElemento).

Las marcas pueden escribirse en cualquier orden y no hace falta utilizar todas. Si usa marcas no válidas, se genera un error. También puede usar una cadena vacía para el segundo argumento. Esto tiene el mismo efecto que usar solo el primer argumento. Sin embargo, no está permitido usar una secuencia vacía.

▣ *Ejemplos*

- **altova:elements**("MiElemento", "rip") devuelve MiElemento()*
- **altova:elements**("MiElemento", "pri") devuelve MiElemento()*
- **altova:elements**("MiElemento", "") devuelve MiElemento()*
- **altova:elements**("MiElemento", "Rip") devuelve un error de marca desconocida.
- **altova:elements**("MiElemento",) devuelve un error diciendo que falta el segundo argumento.

▼ find-first [altova:]

altova:find-first((Secuencia *como item()**, **(Condición(Elemento-Secuencia** *como xs:boolean***))** como **item()?** XP3.1 XQ3.1

Esta función toma dos argumentos. El primero es una secuencia de uno o varios elementos de cualquier tipo de datos. El segundo argumento, **Condición**, es una referencia a una función XPath que toma un argumento (es decir, su aridad es 1) y devuelve un valor binario. Cada elemento de **Secuencia** se envía a su vez a la función a la que se hace referencia en **Condición**. Nota: recuerde que esta función solo toma un argumento. El primer elemento de **Secuencia** que consiga que la función de **Condición** dé **true()** como resultado se devuelve como resultado de **find-first** y la iteración se detiene.

Ejemplos

- **altova:find-first**(5 to 10, function($a) {$a mod 2 = 0}) devuelve xs:integer 6

El argumento **Condición** remite a la función inline XPath 3.0 **function()**, que declara una función inline llamada **$a** y después la define. Cada elemento del argumento Secuencia de **find-first** se envía a su vez como valor de entrada a **$a**. El valor de entrada se prueba en la condición en la definición de función ($a mod 2 = 0). El primer valor de entrada que cumpla la condición se devuelve como resultado de **find-first** (en este caso **6**).

- **altova:find-first**((1 to 10), (function($a) {$a+3=7})) devuelve xs:integer 4

Más ejemplos

Si existe el archivo **C:\Temp\Customers.xml**:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve xs:string C:\Temp\Customers.xml

Si no existe el archivo **C:\Temp\Customers.xml** pero existe **http://www.altova.com/index.html**:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve xs:string http://www.altova.com/index.html

Si no existe el archivo **C:\Temp\Customers.xml** y tampoco existe **http://www.altova.com/index.html**:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) no devuelve ningún resultado

Notas sobre los ejemplos anteriores

- La función XPath 3.0 doc-available toma un solo argumento de cadena, que se usa como URI, y devuelve true si en el URI dado se encuentra un nodo de documento. El documento que está en el URI dado debe ser un documento XML.
- La función doc-available se puede usar para **Condición**, el segundo argumento de find-first, porque solamente toma un argumento (aridad=1), porque toma un item() como entrada (una cadena que se usa como URI) y devuelve un valor binario.

© 2018 Altova GmbH

- Recuerde que solamente se hace referencia a la función `doc-available` pero no se le llama. El sufijo `#1` que se anexa a la función indica una función cuya aridad es 1. Es decir, `doc-available#1` simplemente significa "*Utilizar la función doc-availabe() que tiene aridad=1, pasándole como solo argumento a su vez cada uno de los elementos de la primera secuencia.*" Como resultado, se pasarán las dos cadenas a **`doc-available()`**, que utiliza la cadena como URI y prueba si existe un nodo de documento en el URI. Si existe, entonces `doc-available()` da como resultado `true()` y esa cadena se devuelve como resultado de la función **`find-first`**. Nota sobre la función `doc-available()`: las rutas de acceso relativas se resuelven en relación al URI base actual, que es por defecto el URI del documento XML desde el que se carga la función.

▼ find-first-combination [altova:]

`altova:find-first-combination((Sec-01` *`como item()*`*`),` **`(Sec-02`** *`como item()*`*`),` **`(Condición( Elem-Sec-01, Elem-Sec-02`** *`como xs:boolean`*`))`** como **`item()*`** **XP3.1** **XQ3.1**

Esta función toma tres argumentos:

- Los dos primeros (**`Sec-01`** y **`Sec-02`**) son secuencias de uno o más elementos de cualquier tipo de datos.
- El tercero (**`Condición`**) es una referencia a una función XPath que toma dos argumentos (su aridad es 2) y devuelve un valor binario.

Los elementos de **`Sec-01`** y **`Sec-02`** se pasan en pares ordenados (cada par está formado por un elemento de cada secuencia) como argumentos de la función de **`Condición`**. Los pares se ordenan de la siguiente manera:

```
Si   Sec-01 = X1, X2, X3 ... Xn
Y  Sec-02 = Y1, Y2, Y3 ... Yn
Entonces (X1 Y1), (X1 Y2), (X1 Y3) ... (X1 Yn), (X2 Y1), (X2 Y2) ... (Xn
Yn)
```

El primer par ordenado que consiga que la función de **`Condición`** dé como resultado **`true()`** se devuelve como resultado de **`find-first-combination`**. Recuerde que (i) si la función de **`Condición`** recorre los pares de argumentos dados y no consigue dar **`true()`** como resultado ni una vez, entonces **`find-first-combination`** devuelve *Sin resultados;* (ii) el resultado de **`find-first-combination`** siempre será un par de elementos (de cualquier tipo de datos) o ningún elemento.

⊟ *Ejemplos*

- **`altova:find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32})` devuelve la secuencia de `xs:integers (11, 21)`
- **`altova:find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33})` devuelve la secuencia de `xs:integers (11, 22)`
- **`altova:find-first-pair`**`(11 to 20, 21 to 30, function($a, $b) {$a+$b = 34})` devuelve la secuencia de `xs:integers (11, 23)`

▼ find-first-pair [altova:]

`altova:find-first-pair((Sec-01` *`como item()*`*`),` **`(Sec-02`** *`como item()*`*`),`

(Condición(Elem-Sec-01, Elem-Sec-02 *como xs:boolean*)) como **item()*** XP3.1 XQ3.1
Esta función toma tres argumentos:

- Los dos primeros (**Sec-01** y **Sec-02**) son secuencias de uno o más elementos de cualquier tipo de datos.
- El tercero (**Condición**) es una referencia a una función XPath que toma dos argumentos (su aridad es 2) y devuelve un valor binario.

Los elementos de **Sec-01** y **Sec-02** se pasan en pares ordenados como argumentos de la función de **Condición**. Los pares se ordenan de la siguiente manera:

```
Si   Sec-01 = X1, X2, X3 ... Xn
Y  Sec-02 = Y1, Y2, Y3 ... Yn
Entonces (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

El primer par ordenado que consiga que la función de **Condición** dé como resultado **true()** se devuelve como resultado de **find-first-pair**. Recuerde que (i) si la función de **Condición** recorre los pares de argumentos dados y no consigue dar **true()** como resultado ni una vez, entonces **find-first-pair** devuelve *Sin resultados;* (ii) el resultado de **find-first-pair** siempre será un par de elementos (de cualquier tipo de datos) o ningún elemento.

Ejemplos

- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) devuelve la secuencia de xs:integers (11, 21)
- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) devuelve *Sin resultados*

Observe que en los dos ejemplos anteriores el orden de los pares es: (11, 21) (12, 22) (13, 23)...(20, 30). Por ese motivo el segundo ejemplo no obtiene resultados (porque ningún par ordenado consigue sumar 33).

find-first-pair-pos [altova:]

altova:find-first-pair-pos((Sec-01 *como item()**), **(Sec-02** *como item()**), **(Condición(Elem-Sec-01, Elem-Sec-02** *como xs:boolean*)) como **xs:integer** XP3.1 XQ3.1
Esta función toma tres argumentos:

- Los dos primeros (**Sec-01** y **Sec-02**) son secuencias de uno o más elementos de cualquier tipo de datos.
- El tercero (**Condición**) es una referencia a una función XPath que toma dos argumentos (su aridad es 2) y devuelve un valor binario.

Los elementos de **Sec-01** y **Sec-02** se pasan en pares ordenados como argumentos de la función de **Condición**. Los pares se ordenan de la siguiente manera:

```
Si   Sec-01 = X1, X2, X3 ... Xn
Y  Sec-02 = Y1, Y2, Y3 ... Yn
Entonces (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```


© 2018 Altova GmbH

La posición de índice del primer par ordenado que consiga que la función de **Condición** dé como resultado **true()** se devuelve como resultado de **find-first-pair-pos**. Recuerde que si la función de **Condición** recorre los pares de argumentos dados y no da como resultado **true()** ni una sola vez, entonces **find-first-pair-pos** devuelve *Sin resultados*.

Ejemplos

- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) devuelve 1
- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) devuelve *Sin resultados*

Observe que en los dos ejemplos anteriores el orden de los pares es: (11, 21) (12, 22) (13, 23)...(20, 30). En el primer ejemplo el primer par consigue que la función de **Condición** dé como resultado **true()** y, por tanto, se devuelve la posición de índice que tienen en la secuencia (**1**). El segundo ejemplo, sin embargo, devuelve *Sin resultados* porque ningún par consigue sumar 33.

find-first-pos [altova:]

altova:find-first-pos((Secuencia *como item()**)**,* **(Condición(** **Elem-Sec** *como xs:boolean***))** como **xs:integer** XP3.1 XQ3.1

Esta función toma dos argumentos. El primer argumento es una secuencia de uno o varios elementos de cualquier tipo. El segundo argumento (**Condición**) es una referencia a una función XPath que toma un argumento (su aridad es 1) y devuelve un valor binario. Cada elemento de **Secuencia** se envía a su vez a la función a la que se hace referencia en **Condición**. (Recuerde que esta función toma un solo argumento.) El primer elemento de **Secuencia** que consiga que la función de **Condición** dé como resultado **true()** devuelve la posición de índice que tiene en **Secuencia** como resultado de **find-first-pos** y la iteración se detiene.

Ejemplos

- **altova:find-first-pos**(5 to 10, function($a) {$a mod 2 = 0}) devuelve xs:integer 2

El argumento **Condición** hace referencia a la función inline XPath 3.0 **function()**, que declara una función inline llamada **$a** y después la define. Cada elemento del argumento **Sequence** de **find-first-pos** se pasa a su vez como valor de entrada de **$a**. El valor de entrada se prueba en la condición de la definición de función ($a mod 2 = 0). La posición de índice que tiene en la secuencia el primer valor de entrada que cumple la condición se devuelve como resultado de **find-first-pos** (en este caso es la posición de índice **2**, porque **6** es el primer valor (de la secuencia) que cumple la condición y su posición de índice en la secuencia es **2**).

- **altova:find-first-pos**((2 to 10), (function($a) {$a+3=7})) devuelve xs:integer 3

Más ejemplos

Si existe el archivo **C:\Temp\Customers.xml**:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve 1

Si no existe el archivo **C:\Temp\Customers.xml** pero existe **http://www.altova.com/index.html**:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve 2

Si no existe el archivo **C:\Temp\Customers.xml** y tampoco existe **http://www.altova.com/index.html**:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) no devuelve ningún resultado

Notas sobre los ejemplos anteriores

- La función XPath 3.0 `doc-available` toma un solo argumento de cadena, que se usa como URI, y devuelve `true` si en el URI dado se encuentra un nodo de documento. El documento que está en el URI dado debe ser un documento XML.
- La función `doc-available` se puede usar para **Condición**, el segundo argumento de `find-first-pos`, porque solamente toma un argumento (aridad=1), porque toma un `item()` como entrada (una cadena que se usa como URI) y devuelve un valor binario.
- Recuerde que solamente se hace referencia a la función `doc-available` pero no se le llama. El sufijo `#1` que se anexa a la función indica una función cuya aridad es 1. Es decir, `doc-available#1` simplemente significa "*Utilizar la función doc-availabe() que tiene aridad=1, pasándole como solo argumento a su vez cada uno de los elementos de la primera secuencia.*" Como resultado, se pasarán las dos cadenas a **doc-available()**, que utiliza la cadena como URI y prueba si existe un nodo de documento en el URI. Si existe, entonces `doc-available()` da como resultado `true()` y esa cadena se devuelve como resultado de la función **find-first-pos**. Nota sobre la función `doc-available()`: las rutas de acceso relativas se resuelven en relación al URI base actual, que es por defecto el URI del documento XML desde el que se carga la función.

▼ substitute-empty [altova:]

altova:substitute-empty(PrimeraSecuencia *as item()**,* **SegundaSecuencia** *as item()***)** como **item()*** XP3.1 XQ3.1

Si **PrimeraSecuencia** está vacío, la función devuelve **SegundaSecuencia**. Si **PrimeraSecuencia** no está vacío, la función devuelve **PrimeraSecuencia**.

Ejemplos

- **altova:substitute-empty**((1,2,3), (4,5,6)) devuelve (1,2,3)
- **altova:substitute-empty**((), (4,5,6)) devuelve (4,5,6)

© 2018 Altova GmbH

19.1.2.1.7 Funciones XPath/XQuery: de cadena

Las funciones de extensión de Altova para trabajar con cadenas pueden utilizarse en expresiones XPath y XQuery y ofrecen funciones adicionales para el procesamiento de datos. Estas funciones se pueden usar con los motores **XPath 3.0** y **XQuery 3.0** de Altova. Están disponibles en contextos XPath/XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** `http://www.altova.com/xslt-extensions` y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

▼ camel-case [altova:]

altova:camel-case(CadenaEntrada *como xs:string***)** como **xs:string** XP3.1 XQ3.1
Devuelve la cadena de entrada **CadenaEntrada** escrita en CamelCase. La cadena se analiza usando la expresión regular `'\s'` (que es la forma abreviada del carácter espacio en blanco). El primer carácter que no sea un espacio en blanco situado después de un espacio en blanco o de una secuencia de espacios en blanco consecutivos se pondrá en mayúsculas. El primer carácter de la cadena de salida se pondrá en mayúsculas.

⊟ *Ejemplos*

- **altova:camel-case**("max") devuelve Max
- **altova:camel-case**("max max") devuelve Max Max
- **altova:camel-case**("file01.xml") devuelve File01.xml
- **altova:camel-case**("file01.xml file02.xml") devuelve File01.xml File02.xml
- **altova:camel-case**("file01.xml file02.xml") devuelve File01.xml File02.xml
- **altova:camel-case**("file01.xml -file02.xml") devuelve File01.xml -file02.xml

altova:camel-case(CadenaEntrada *como xs:string*, **CaracteresDivisión** *como xs:string*, **EsExpReg** *como xs:boolean***)** como **xs:string** XP3.1 XQ3.1

Devuelve la cadena de entrada **CadenaEntrada** escrita en CamelCase usando los **CaracteresDivisón** para determinar qué caracteres desencadenan el siguiente uso de mayúsculas. El argumento **CaracteresDivisón** se usa como expresión regular **cuando EsExpReg = true()** o como caracteres planos cuando r **EsExpReg = false()**. El primer carácter de la cadensa de salida se escribe con mayúsculas.

Ejemplos

- `altova:camel-case("setname getname", "set|get", true())` devuelve `setName getName`
- `altova:camel-case("altova\documents\testcases", "\", false())` devuelve `Altova\Documents\Testcases`

char [altova:]

altova:char(Posición *as xs:integer*) como **xs:string** XP3.1 XQ3.1

Devuelve una cadena que contiene el carácter que está en la posición indicada por el argumento **Posición** en la cadena que se obtiene al convertir el valor del elemento de contexto en `xs:string`. La cadena resultante estará vacía si en la posición indicada no existe ningún carácter.

Ejemplos

Si el elemento de contexto es **1234ABCD**:

- `altova:char(2)` devuelve `2`
- `altova:char(5)` devuelve `A`
- `altova:char(9)` devuelve la cadena vacía
- `altova:char(-2)` devuelve la cadena vacía

altova:char(CadenaEntrada *as xs:string*, **Posición** *as xs:integer*) como **xs:string** XP3.1 XQ3.1

Devuelve una cadena que contiene el carácter que está en la posición indicada por el argumento **Posición** en la cadena dada por el argumento **CadenaEntrada**. La cadena resultante estará vacía si en la posición indicada no existe ningún carácter.

Ejemplos

- `altova:char("2014-01-15", 5)` devuelve `-`
- `altova:char("USA", 1)` devuelve `U`
- `altova:char("USA", 1)` devuelve la cadena vacía
- `altova:char("USA", -2)` devuelve la cadena vacía

first-chars [altova:]

altova:first-chars(X *as xs:integer*) como **xs:string** XP3.1 XQ3.1

Devuelve una cadena que contiene los **X** primeros caracteres de la cadena que se obtiene al convertir el valor del elemento de contexto en `xs:string`.

Ejemplos

© 2018 Altova GmbH

Si el elemento de contexto es **1234ABCD**:

- **altova:first-chars**(2) devuelve 12
- **altova:first-chars**(5) devuelve 1234A
- **altova:first-chars**(9) devuelve 1234ABCD

altova:first-chars(CadenaEntrada *as xs:string*, **X** *as xs:integer***)** como **xs:string** XP3.1 XQ3.1
Devuelve una cadena que contiene los **X** primeros caracteres de la cadena dada como argumento **CadenaEntrada**.

Ejemplos

- **altova:first-chars**("2014-01-15", 5) devuelve 2014-
- **altova:first-chars**("USA", 1) devuelve U

last-chars [altova:]

altova:last-chars(X *as xs:integer***)** como **xs:string** XP3.1 XQ3.1
Devuelve una cadena que contiene los X últimos caracteres de la cadena que se obtiene al convertir el valor del elemento de contexto en `xs:string`.

Ejemplos

Si el elemento de contexto es **1234ABCD**:

- **altova:last-chars**(2) devuelve CD
- **altova:last-chars**(5) devuelve 4ABCD
- **altova:last-chars**(9) devuelve 1234ABCD

altova:last-chars(CadenaEntrada *as xs:string*, **X** *as xs:integer***)** como **xs:string** XP3.1 XQ3.1
Devuelve una cadena que contiene los **X** últimos caracteres de la cadena dada como argumento **CadenaEntrada**.

Ejemplos

- **altova:last-chars**("2014-01-15", 5) devuelve 01-15-
- **altova:last-chars**("USA", 10) devuelve USA

pad-string-left [altova:]

altova:pad-string-left(CadenaParaRellenar *como xs:string*, **LongitudCadena** *como xs:integer*, **CarácterRelleno** *como xs:string***)** como **xs:string** XP3.1 XQ3.1
El argumento `CarácterRelleno` es un solo carácter. Se añade a la izquierda de la cadena para aumentar el número de caracteres de la `CadenaParaRellenar`, de modo que este número equivalga al valor entero del argumento `LongitudCadena`. El argumento `LongitudCadena` puede tener cualquier valor entero (positivo o negativo), pero el relleno solo se lleva a cabo si el valor de `LongitudCadena` es mayor que el número de caracteres de `CadenaParaRellenar`. Si `CadenaParaRellenar` tiene más caracteres que el valor de

`LongitudCadena`, entonces `CadenaParaRellenar` se deja como está.

Ejemplos

- **`altova:pad-string-left`**`('AP', 1, 'Z')` devuelve `'AP'`
- **`altova:pad-string-left`**`('AP', 2, 'Z')` devuelve `'AP'`
- **`altova:pad-string-left`**`('AP', 3, 'Z')` devuelve `'ZAP'`
- **`altova:pad-string-left`**`('AP', 4, 'Z')` devuelve `'ZZAP'`
- **`altova:pad-string-left`**`('AP', -3, 'Z')` devuelve `'AP'`
- **`altova:pad-string-left`**`('AP', 3, 'YZ')` devuelve un error indicando que el carácter de relleno es demasiado largo.

pad-string-right [altova:]

`altova:pad-string-right(CadenaParaRellenar` *como xs:string*, **`LongitudCadena`** *como xs:integer*, **`CarácterRelleno`** *como xs:string*`)` como **`xs:string`** XP3.1 XQ3.1

El argumento `CarácterRelleno` es un solo carácter. Se añade a la derecha de la cadena para aumentar el número de caracteres de la `CadenaParaRellenar`, de modo que este número equivalga al valor entero del argumento `LongitudCadena`. El argumento `LongitudCadena` puede tener cualquier valor entero (positivo o negativo), pero el relleno solo se lleva a cabo si el valor de `LongitudCadena` es mayor que el número de caracteres de `CadenaParaRellenar`. Si `CadenaParaRellenar` tiene más caracteres que el valor de `LongitudCadena`, entonces `CadenaParaRellenar` se deja como está.

Ejemplos

- **`altova:pad-string-right`**`('AP', 1, 'Z')` devuelve `'AP'`
- **`altova:pad-string-right`**`('AP', 2, 'Z')` devuelve `'AP'`
- **`altova:pad-string-right`**`('AP', 3, 'Z')` devuelve `'APZ'`
- **`altova:pad-string-right`**`('AP', 4, 'Z')` devuelve `'APZZ'`
- **`altova:pad-string-right`**`('AP', -3, 'Z')` devuelve `'AP'`
- **`altova:pad-string-right`**`('AP', 3, 'YZ')` devuelve un error indicando que el carácter de relleno es demasiado largo.

repeat-string [altova:]

`altova:repeat-string(CadenaEntrada` *as xs:string*, **`Repeticiones`** *as xs:integer*`)` como **`xs:string`** XP2 XQ1 XP3.1 XQ3.1

Genera una cadena que está compuesta por el primer argumento `CadenaEntrada` repetida tantas veces como indique el argumento `Repeticiones`.

Ejemplo

- **`altova:repeat-string`**`("Altova #", 3)`
  devuelve `Altova #Altova #Altova #"`

substring-after-last [altova:]

`altova:substring-after-last(CadenaPrincipal` *as xs:string*, **`CadenaPrueba`** *as xs:string*`)` como **`xs:string`** XP3.1 XQ3.1

© 2018 Altova GmbH

Si **CadenaPrueba** se encuentra en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de **CadenaPrueba** en **CadenaPrincipal**. Si **CadenaPrueba** no está en **CadenaPrincipal**, entonces devuelve la cadena vacía. Si **CadenaPrueba** es una cadena vacía, entonces devuelve la **CadenaPrincipal** entera. Si **CadenaPrueba** aparece varias veces en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de la última **CadenaPrueba**.

Ejemplos

- **altova:substring-after-last**('ABCDEFGH', 'B') devuelve 'CDEFGH'
- **altova:substring-after-last**('ABCDEFGH', 'BC') devuelve 'DEFGH'
- **altova:substring-after-last**('ABCDEFGH', 'BD') devuelve ''
- **altova:substring-after-last**('ABCDEFGH', 'Z') devuelve ''
- **altova:substring-after-last**('ABCDEFGH', '') devuelve 'ABCDEFGH'
- **altova:substring-after-last**('ABCD-ABCD', 'B') devuelve 'CD'
- **altova:substring-after-last**('ABCD-ABCD-ABCD', 'BCD') devuelve ''

substring-before-last [altova:]

altova:substring-before-last(CadenaPrincipal *as xs:string*, **CadenaPrueba** *as xs:string***)** como **xs:string** XP3.1 XQ3.1

Si **CadenaPrueba** se encuentra en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de **CadenaPrueba** en **CadenaPrincipal**. Si **CadenaPrueba** no está en **CadenaPrincipal**, entonces devuelve la cadena vacía. Si **CadenaPrueba** es una cadena vacía, entonces devuelve la **CadenaPrincipal** entera. Si **CadenaPrueba** aparece varias veces en **CadenaPrincipal**, la función devuelve la subcadena que aparece antes de la última **CadenaPrueba**.

Ejemplos

- **altova:substring-before-last**('ABCDEFGH', 'B') devuelve 'A'
- **altova:substring-before-last**('ABCDEFGH', 'BC') devuelve 'A'
- **altova:substring-before-last**('ABCDEFGH', 'BD') devuelve ''
- **altova:substring-before-last**('ABCDEFGH', 'Z') devuelve ''
- **altova:substring-before-last**('ABCDEFGH', '') devuelve ''
- **altova:substring-before-last**('ABCD-ABCD', 'B') devuelve 'ABCD-A'
- **altova:substring-before-last**('ABCD-ABCD-ABCD', 'ABCD') devuelve 'ABCD-ABCD-'

substring-pos [altova:]

altova:substring-pos(Cadena *as xs:string*, **CadenaBúsqueda** *as xs:string***)** como **xs:integer** XP3.1 XQ3.1

Devuelve la posición de carácter de la primera instancia de **CadenaBúsqueda** en **Cadena**. La posición de carácter se devuelve como número entero. El primer carácter de **CadenaBúsqueda** tiene la posición 1. Si **CadenaBúsqueda** no aparece dentro de **Cadena**, la función devuelve el entero `0`. Para buscar la segunda instancia de **CadenaBúsqueda**, etc. use la otra firma de esta función.

Ejemplos

- **altova:substring-pos**('Altova', 'to') devuelve 3
- **altova:substring-pos**('Altova', 'tov') devuelve 3

- **altova:substring-pos**('Altova', 'tv') devuelve 0
- **altova:substring-pos**('AltovaAltova', 'to') devuelve 3

altova:substring-pos(Cadena *as xs:string*, **CadenaBúsqueda** *as xs:string*, **Entero** *as xs:integer***)** como **xs:integer** XP3.1 XQ3.1
Devuelve la posición de carácter de **CadenaBúsqueda** en **Cadena**. La búsqueda de **CadenaBúsqueda** empieza en la posición de carácter dada por el argumento **Entero** (es decir, no se busca en la subcadena anterior a esta posición). El entero devuelto, sin embargo, es la posición que la cadena encontrada tiene en **Cadena**. Esta firma es muy práctica si quiere buscar la segunda posición, etc. de una cadena que aparece varias veces dentro de **Cadena**. Si **CadenaBúsqueda** no aparece en **Cadena**, la función devuelve el entero 0.

Ejemplos

- **altova:substring-pos**('Altova', 'to', 1) devuelve 3
- **altova:substring-pos**('Altova', 'to', 3) devuelve 3
- **altova:substring-pos**('Altova', 'to', 4) devuelve 0
- **altova:substring-pos**('Altova-Altova', 'to', 0) devuelve 3
- **altova:substring-pos**('Altova-Altova', 'to', 4) devuelve 10

▼ trim-string [altova:]

altova:trim-string(CadenaEntrada *as xs:string***)** como **xs:string** XP3.1 XQ3.1
Esta función toma un argumento xs:string, quita los espacios en blanco iniciales y finales y devuelve un xs:string "recortado".

Ejemplos

- **altova:trim-string**(" Hello World ")) devuelve "Hello World"
- **altova:trim-string**("Hello World ")) devuelve "Hello World"
- **altova:trim-string**(" Hello World")) devuelve "Hello World"
- **altova:trim-string**("Hello World")) devuelve "Hello World"
- **altova:trim-string**("Hello World")) devuelve "Hello World"

▼ trim-string-left [altova:]

altova:trim-string-left(CadenaEntrada *as xs:string***)** como **xs:string** XP3.1 XQ3.1
Esta función toma un argumento xs:string, quita los espacios en blanco iniciales y devuelve un xs:string recortado por la izquierda.

Ejemplos

- **altova:trim-string-left**(" Hello World ")) devuelve "Hello World "
- **altova:trim-string-left**("Hello World ")) devuelve "Hello World "
- **altova:trim-string-left**(" Hello World")) devuelve "Hello World"
- **altova:trim-string-left**("Hello World")) devuelve "Hello World"
- **altova:trim-string-left**("Hello World")) devuelve "Hello World"

▼ trim-string-right [altova:]

altova:trim-string-right(CadenaEntrada *as xs:string***)** como **xs:string** XP3.1

© 2018 Altova GmbH

XQ3.1
Esta función toma un argumento `xs:string`, quita los espacios en blanco finales y devuelve una cadena `xs:string` recortada por la derecha.

Ejemplos

- **altova:trim-string-right**(" Hello World ")) devuelve " Hello World"
- **altova:trim-string-right**("Hello World ")) devuelve "Hello World"
- **altova:trim-string-right**(" Hello World")) devuelve " Hello World"
- **altova:trim-string-right**("Hello World")) devuelve "Hello World"
- **altova:trim-string-right**("Hello World")) devuelve "Hello World"

19.1.2.1.8 Funciones XPath/XQuery varias

Estas funciones de extensión XPath/XQuery generales son compatibles con la versión actual de MapForce y se pueden usar en (i) expresiones XPath en contextos XSLT o (ii) en expresiones XQuery en documentos XQuery.

Nota sobre el nombre de las funciones y lenguajes

Puede utilizar todas las funciones de extensión de Altova en sus expresiones XPath/XQuery. Con ellas conseguirá funciones adicionales no disponibles en la biblioteca de funciones estándar de XPath, XQuery y XSLT. Las funciones de extensión de Altova están en el **espacio de nombres** **http://www.altova.com/xslt-extensions** y en esta sección se presentan con el prefijo, que se supone estará enlazado al espacio de nombres señalado. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3.1
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3.1

▼ get-temp-folder [altova:]

altova:get-temp-folder() como `xs:string` XP2 XQ1 XP3.1 XQ3.1
Esta función no toma ningún argumento. Devuelve la ruta de acceso de la carpeta temporal del usuario actual.

Ejemplo

- **altova:get-temp-folder**()
  en un equipo Windows devuelve (más o menos) `C:\Usuarios\<Usuario>\AppData\Local\Temp\` como valor de tipo `xs:string`.

▼ generate-guid [altova:]

altova:generate-guid() as **xs:string** XP2 XQ1 XP3.1 XQ3.1

Genera una cadena única de la interfaz gráfica del usuario.

⊞ *Ejemplo*

- **altova:generate-guid**() devuelve (por ejemplo) 85F971DA-17F3-4E4E-994E-99137873ACCD

[Subir]

19.1.2.1.9 Funciones de extensión para gráficos

Las funciones para gráficos que aparecen a continuación sirven para crear, generar y guardar gráficos como imágenes. Esta funciones son compatibles con la versión actual de su producto de Altova. No obstante, tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Las funciones para gráficos son **funciones XPath** (no XSLT) y las hay de dos tipos:

- Funciones para generar y guardar gráficos
- Funciones para crear gráficos

Nota: las funciones para gráficos solamente son compatibles con los **productos servidor** de Altova y con las ediciones **Enterprise Edition** de las herramientas de escritorio de Altova.

Nota: en los **productos servidor** de Altova los formatos de imagen compatibles para gráficos son `jpg`, `png` y `bmp`. La opción más recomendable es `png` porque no pierde información y es un formato comprimido. En las ediciones **Enterprise Edition** de las herramientas de escritorio de Altova, los formatos compatibles son `jpg`, `png`, `bmp` y `gif`.

Funciones para generar y guardar gráficos

Estas funciones toman el objeto del gráfico (obtenido con las funciones de creación de gráficos) y generan una imagen o guardan una imagen en un archivo.

generate-chart-image ($chart, $width, $height, $encoding) as atomic

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `create-chart`
- `$width` y `$height` deben especificarse con una unidad de longitud
- `$encoding` puede ser `binarytobase64` o `binarytobase16`

La función devuelve la imagen del gráfico en la codificación elegida.

generate-chart-image (`$chart, $width, $height, $encoding, $imagetype`) `as atomic`

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `create-chart`
- `$width` y `$height` deben especificarse con una unidad de longitud
- `$encoding` puede ser `base64Binary` o `hexBinary`
- `$imagetype` puede ser uno de estos formatos de imagen: `png`, `gif`, `bmp`, `jpg`, `jpeg`. Recuerde que el formato `gif` no es compatible con los productos servidor de Altova (*ver nota al principio de este apartado*)

La función devuelve la imagen del gráfico en la codificación y formato de imagen elegidos.

save-chart-image (`$chart, $filename, $width, $height`) `as empty()` ***(solo en Windows)***

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `create-chart`
- `$filename` es la ruta de acceso del archivo y el nombre de archivo donde se debe guardar la imagen del gráfico
- `$width` y `$height` deben especificarse con una unidad de longitud

La función guarda la imagen del gráfico en el archivo especificado en `$filename`.

save-chart-image (`$chart, $filename, $width, $height, $imagetype`) `as empty()` ***(solo en Windows)***

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `create-chart`
- `$filename` es la ruta de acceso del archivo y el nombre de archivo donde se debe guardar la imagen del gráfico
- `$width` y `$height` deben especificarse con una unidad de longitud
- `$imagetype` puede ser uno de estos formatos de imagen: `png`, `gif`, `bmp`, `jpg`, `jpeg`. Recuerde que el formato `gif` no es compatible con los productos servidor de Altova (*ver nota al principio de este apartado*)

La función guarda la imagen del gráfico en el archivo especificado en `$filename` en el formato de imagen elegido.

Funciones para crear gráficos

Puede usar estas funciones para crear gráficos.

create-chart($chart-config, $chart-data-series*) como componente de extensión de gráficos

donde

- $chart-config es el componente de extensión chart-config obtenido con la función create-chart-config o con la función create-chart-config-from-xml
- $chart-data-series es el componente de extensión chart-data-series obtenido con la función create-chart-data-series o con la función create-chart-data-series-from-rows

La función devuelve un componente de extensión de gráficos, que se crea a partir de los datos suministrados con los argumentos.

create-chart-config($type-name, $title) como componente de extensión de gráficos chart-config

donde

- $type-name especifica el tipo de gráfico que se debe crear: Pie, Pie3d, BarChart, BarChart3d, BarChart3dGrouped, LineChart, ValueLineChart, RoundGauge, BarGauge
- $title es el nombre del gráfico

La función devuelve un componente de extensión chart-config que contiene los datos de configuración del gráfico.

create-chart-config-from-xml($xml-struct) como componente de extensión de gráficos chart-config

donde

- $xml-struct es la estructura XML que contiene los datos de configuración del gráfico

La función devuelve un componente de extensión chart-config que contiene los datos de configuración del gráfico. Estos datos se suministran en un fragmento de código XML.

create-chart-data-series($series-name?, $x-values*, $y-values*) como componente de extensión de gráficos chart-data-series

donde

© 2018 Altova GmbH

- `$series-name` especifica el nombre de la serie
- `$x-values` presenta la lista de valores del eje X
- `$y-values` presenta la lista de valores del eje Y

La función devuelve un componente de extensión chart-data-series que contiene los datos necesarios para generar el gráfico: es decir, el nombre de las series y los datos de los ejes.

`create-chart-data-row``(x, y1, y2, y3, ...) como componente de extensión de gráficos chart-data-x-`*`N`*`y-row`

donde

- `x` es el valor de la columna del eje X de la fila de datos del gráfico
- `yN` son los valores de las columnas del eje Y

La función devuelve un componente de extensión chart-data-x-Ny-row, que contiene los datos para la columna del eje X y las columnas del eje Y de una sola serie.

`create-chart-data-series-from-rows``($series-names as xs:string*, $row*) como componente de extensión de gráficos chart-data-series`

donde

- `$series-name` es el nombre de la series que se debe crear
- `$row` es el componente de extensión chart-data-x-Ny-row que se debe crear como serie

La función devuelve un componente de extensión chart-data-series, que contiene los datos para el eje X y el eje Y de la serie.

`create-chart-layer``($chart-config, $chart-data-series*) como componente de extensión de gráficos chart-layer`

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `create-chart-config` o con la función `create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series que se obtiene con la función `create-chart-data-series` o con la función `create-chart-data-series-from-rows`

La función devuelve un componente de extensión chart-layer, que contiene los datos de la capa de gráfico.

create-multi-layer-chart(`$chart-config, $chart-data-series*, $chart-layer*`)

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `create-chart-config` o con la función `create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series obtenido con la función `create-chart-data-series` o con la función `create-chart-data-series-from-rows`
- `$chart-layer` es el componente de extensión de gráficos multicapas obtenido con la función `create-chart-layer`

La función devuelve un componente de extensión de gráficos multicapas.

create-multi-layer-chart(`$chart-config, $chart-data-series*, $chart-layer*, xs:boolean $mergecategoryvalues`)

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `create-chart-config` o con la función `create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series obtenido con la función `create-chart-data-series` o con la función `create-chart-data-series-from-rows`
- `$chart-layer` es el componente de extensión de gráficos multicapa obtenido con la función `create-chart-layer`

La función devuelve un componente de extensión de gráficos multicapa.

19.1.2.1.9.1 Estructura XML de los datos de gráficos

A continuación puede ver un fragmento de código XML con datos de gráfico, tal y como aparecería para las funciones de extensión para gráficos de Altova. Esto afecta al aspecto del gráfico. No todos los elementos se utilizan para todos los tipos de gráfico. Por ejemplo, el elemento `<Pie>` se omite en los gráficos de barras.

Nota: las funciones para gráficos son compatibles solamente con las ediciones **Enterprise** y **Server**.

```
<chart-config>
        <General
        SettingsVersion="1"   debe darse
        ChartKind="BarChart"   Pie, Pie3d, BarChart, StackedBarChart, BarChart3d,
BarChart3dGrouped, LineChart, ValueLineChart, AreaChart, StackedAreaChart, RoundGauge,
BarGauge, CandleStick
        BKColor="#ffffff"   Color
        BKColorGradientEnd="#ffffff"   Color. En caso de degradado, BKColor y
BKColorGradientEnd definen el color del degradado
```

```
BKMode="#ffffff"   Solid, HorzGradient, VertGradient
BKFile="Ruta+NombreArchivo"   Cadena. Si el archivo existe, su contenido se dibuja sobre el fondo.
BKFileMode="Stretch"   Stretch, ZoomToFit, Center, Tile
ShowBorder="1"   Bool
PlotBorderColor="#000000"   Color
PlotBKColor="#ffffff"   Color
Title=""   Cadena de texto
ShowLegend="1"   Bool
OutsideMargin="3.%"   PorcentajeOPíxel
TitleToPlotMargin="3.%"   PorcentajeOPíxel
LegendToPlotMargin="3.%"   PorcentajeOPíxel
Orientation="vert"   Enums. Los valores posibles son: vert, horz
>
<TitleFont
   Color="#000000"   Color
   Name="Tahoma"   Cadena de texto
   Bold="1"   Bool
   Italic="0"   Bool
   Underline="0"   Bool
   MinFontHeight="10.pt"   TamañoFuente (solo valores pt)
   Size="8.%"   TamañoFuente />
<LegendFont
   Color="#000000"
   Name="Tahoma"
   Bold="0"
   Italic="0"
   Underline="0"
   MinFontHeight="10.pt"
   Size="3.5%" />
<AxisLabelFont
   Color="#000000"
   Name="Tahoma"
   Bold="1"
   Italic="0"
   Underline="0"
   MinFontHeight="10.pt"
   Size="5.%" />
</General>

<Line
      ConnectionShapeSize="1.%"   PorcentajeOPixel
      DrawFilledConnectionShapes="1"   Bool
      DrawOutlineConnectionShapes="0"   Bool
      DrawSlashConnectionShapes="0"   Bool
      DrawBackslashConnectionShapes="0"   Bool
/>

<Bar
      ShowShadow="1"   Bool
      ShadowColor="#a0a0a0"   Color
      OutlineColor="#000000"   Color
      ShowOutline="1"   Bool
/>
```

<Area
Transparency="0" *UINT (0-255) 255 es totalmente transparente y 0 es opaco*
OutlineColor="#000000" *Color*
ShowOutline="1" *Bool*
/>

<CandleStick
FillHighClose="0" *Bool. Si es 0, el cuerpo está vacío. Si es 1, FillColorHighClose se usa para el cuerpo de la vela*
FillColorHighClose="#ffffff" *Color. Para el cuerpo de la vela cuando el valor de close > que el valor de open*
FillHighOpenWithSeriesColor="1" *Bool. Si es true, el color de la serie se usa para rellenar el cuerpo de la vela cuando el valor de open > que el valor de close*
FillColorHighOpen="#000000" *Color. Para el cuerpo de la vela cuando el valor de open > que el valor de close y* FillHighOpenWithSeriesColor *es false*
/>

<Colors *Combinación de colores definida por el usuario. Este elemento está vacío por defecto, excepto el estilo, y no tiene atributos Color*
UseSubsequentColors ="1" *Booleano. Si es* 0*, entonces se superpone el color. Si es* 1*, se usan los colores siguientes de la capa de gráfico anterior*
Style="User" *Valores posibles: "Default", "Grayscale", "Colorful", "Pastel", "User"*
Colors="#52aca0" *Color: solamente se añade para el conjunto de colores definido por el usuario*
Colors1="#d3c15d" *Color: solamente se añade para el conjunto de colores definido por el usuario*
Colors2="#8971d8" *Color: solamente se añade para el conjunto de colores definido por el usuario*
...
ColorsN="" *Cada conjunto de colores puede tener un máximo de diez colores, de* Colors *a* Colors9
</Colors>

<Pie
ShowLabels="1" *Bool*
OutlineColor="#404040" *Color*
ShowOutline="1" *Bool*
StartAngle="0." *Double*
Clockwise="1" *Bool*
Draw2dHighlights="1" *Bool*
Transparency="0" *Int (De 0 a 255: 0 es opaco, 255 es totalmente transparente)*
DropShadowColor="#c0c0c0" *Color*
DropShadowSize="5.%" *PorcentajeOPíxel*
PieHeight="10.%" *PorcentajeOPíxel. Los valores de píxel pueden ser diferente en el resultado debido a la inclinación 3D*
Tilt="40.0" *Double (De 10 a 90: la inclinación 3D en grados de un gráfico circular 3D)*
ShowDropShadow="1" *Bool*
ChartToLabelMargin="10.%" *PorcentajeOPíxel*

© 2018 Altova GmbH

```
                AddValueToLabel="0"  Bool
                AddPercentToLabel="0"  Bool
                AddPercentToLabels_DecimalDigits="0" UINT ( 0 – 2 )
                >
                <LabelFont
                        Color="#000000"
                        Name="Arial"
                        Bold="0"
                        Italic="0"
                        Underline="0"
                        MinFontHeight="10.pt"
                        Size="4.%"    />
        </Pie>

        <XY>
                <XAxis  Axis
                        AutoRange="1"  Bool
                        AutoRangeIncludesZero="1"  Bool
                        RangeFrom="0."  Double: intervalo manual
                        RangeTill="1."  Double : intervalo manual
                        LabelToAxisMargin="3.%"  PorcentajeOPíxel
                        AxisLabel=""  Cadena de texto
                        AxisColor="#000000"  Color
                        AxisGridColor="#e6e6e6"  Color
                        ShowGrid="1"  Bool
                        UseAutoTick="1"  Bool
                        ManualTickInterval="1."  Double
                        AxisToChartMargin="0.px"  PorcentajeOPíxel
                        TickSize="3.px"  PorcentajeOPíxel
                        ShowTicks="1"  Bool
                        ShowValues="1"  Bool
                        AxisPosition="LeftOrBottom"  Enums: "LeftOrBottom",
"RightOrTop", "AtValue"
                        AxisPositionAtValue = "0"  Double
                        >
                        <ValueFont
                                Color="#000000"
                                Name="Tahoma"
                                Bold="0"
                                Italic="0"
                                Underline="0"
                                MinFontHeight="10.pt"
                                Size="3.%"    />
                </XAxis>
                <YAxis  Eje (igual que XAxis)
                        AutoRange="1"
                        AutoRangeIncludesZero="1"
                        RangeFrom="0."
                        RangeTill="1."
                        LabelToAxisMargin="3.%"
                        AxisLabel=""
                        AxisColor="#000000"
                        AxisGridColor="#e6e6e6"
                        ShowGrid="1"
                        UseAutoTick="1"
```

```
                        ManualTickInterval="1."
                        AxisToChartMargin="0.px"
                        TickSize="3.px"
                        ShowTicks="1"   Bool
                        ShowValues="1"   Bool
                        AxisPosition="LeftOrBottom"   Enums: "LeftOrBottom",
"RightOrTop", "AtValue"
                        AxisPositionAtValue = "0"   Double
                        >
                        <ValueFont
                               Color="#000000"
                               Name="Tahoma"
                               Bold="0"
                               Italic="0"
                               Underline="0"
                               MinFontHeight="10.pt"
                               Size="3.%"/>
                </YAxis>
        </XY>

        <XY3d
                AxisAutoSize="1"   Bool: Si es false, XSize y YSize definen la relación de
aspecto de los ejes x e y. Si es true, la relación de aspecto es igual a la ventana del gráfico
                XSize="100.%"   PorcentajeOPíxel. Los valores en píxel pueden ser diferentes
en el resultado debido a la inclinación 3D y a la opción de ajustar al tamaño
                YSize="100.%"   PorcentajeOPíxel. Los valores en píxel pueden ser diferentes
en el resultado debido a la inclinación 3D y a la opción de ajustar al tamaño
                SeriesMargin="30.%"   PorcentajeOPíxel. Los valores en píxel pueden ser
diferentes en el resultado debido a la inclinación 3D y a la opción de ajustar al tamaño
                Tilt="20."   Double. De -90 a +90 grados
                Rot="20."   Double. De -359 a +359 grados
                FoV="50.">   Double. Campo de visión: de 1 a 120 grados
                >
                <ZAxis
                        AutoRange="1"
                        AutoRangeIncludesZero="1"
                        RangeFrom="0."
                        RangeTill="1."
                        LabelToAxisMargin="3.%"
                        AxisLabel=""
                        AxisColor="#000000"
                        AxisGridColor="#e6e6e6"
                        ShowGrid="1"
                        UseAutoTick="1"
                        ManualTickInterval="1."
                        AxisToChartMargin="0.px"
                        TickSize="3.px" >
                        <ValueFont
                               Color="#000000"
                               Name="Tahoma"
                               Bold="0"
                               Italic="0"
                               Underline="0"
                               MinFontHeight="10.pt"
                               Size="3.%"/>
```


© 2018 Altova GmbH

```
        </ZAxis>
    </XY3d>

    <Gauge
        MinVal="0."  Double
        MaxVal="100."  Double
        MinAngle="225"  UINT: -359-359
        SweepAngle="270"  UINT: 1-359
        BorderToTick="1.%"  PorcentajeOPíxel
        MajorTickWidth="3.px"  PorcentajeOPíxel
        MajorTickLength="4.%"  PorcentajeOPíxel
        MinorTickWidth="1.px"  PorcentajeOPíxel
        MinorTickLength="3.%"  PorcentajeOPíxel
        BorderColor="#a0a0a0"  Color
        FillColor="#303535"  Color
        MajorTickColor="#a0c0b0"  Color
        MinorTickColor="#a0c0b0"  Color
        BorderWidth="2.%"  PorcentajeOPíxel
        NeedleBaseWidth="1.5%"  PorcentajeOPíxel
        NeedleBaseRadius="5.%"  PorcentajeOPíxel
        NeedleColor="#f00000"  Color
        NeedleBaseColor="#141414"  Color
        TickToTickValueMargin="5.%"  PorcentajeOPíxel
        MajorTickStep="10."  Double
        MinorTickStep="5."  Double
        RoundGaugeBorderToColorRange="0.%"  PorcentajeOPíxel
        RoundGaugeColorRangeWidth ="6.%"  PorcentajeOPíxel
        BarGaugeRadius="5.%"  PorcentajeOPíxel
        BarGaugeMaxHeight="20.%"  PorcentajeOPíxel
        RoundGaugeNeedleLength="45.%"  PorcentajeOPíxel
        BarGaugeNeedleLength="3.%"  PorcentajeOPíxel
        >
        <TicksFont
            Color="#a0c0b0"
            Name="Tahoma"
            Bold="0"
            Italic="0"
            Underline="0"
            MinFontHeight="10.pt"
            Size="4.%"
        />
        <ColorRanges>  Intervalos de color definidos por el usuario. Está vacío por
defecto y no tiene elementos secundarios
            <Entry
                From="50. "  Double
                FillWithColor="1"  Bool
                Color="#00ff00"  Color
            />
            <Entry
                From="50.0"
                FillWithColor="1"
                Color="#ff0000"
            />
            ...
```

```
            </ColorRanges>
        </Gauge>
</chart-config>
```

19.1.2.1.9.2 Ejemplo: funciones para gráficos

A continuación puede ver un ejemplo de documento XSLT que muestra cómo usar las funciones de extensión para gráficos de Altova. Después ofrecemos un documento XML y una captura de pantalla de la imagen generada cuando el documento XML se procesa con el documento XSLT usando el motor XSLT 2.0.

Nota: las funciones para gráficos solamente son compatibles con las ediciones **Enterprise Edition** y **Server** de los productos de Altova.

Nota: para más información sobre las tablas de datos del gráfico, consulte la documentación de las herramientas XMLSpy y StyleVision de Altova.

Documento XSLT

Este documento XSLT usa las funciones de extensión para gráficos de Altova para generar un gráfico circular. Este XSLT se puede usar para procesar el documento XML que aparece más abajo.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:altovaext="http://www.altova.com/xslt-extensions"
    exclude-result-prefixes="#all">
    <xsl:output version="4.0" method="html" indent="yes" encoding="UTF-8"/>
    <xsl:template match="/">
        <html>
            <head>
                <title>
                    <xsl:text>HTML Page with Embedded Chart</xsl:text>
                </title>
            </head>
            <body>
                <xsl:for-each select="/Data/Region[1]">
                    <xsl:variable name="extChartConfig" as="item()*">
                        <xsl:variable name="ext-chart-settings" as="item()*">
                            <chart-config>
                                <General
                                    SettingsVersion="1"
                                    ChartKind="Pie3d"
                                    BKColor="#ffffff"
                                    ShowBorder="1"
                                    PlotBorderColor="#000000"
                                    PlotBKColor="#ffffff"
```

```
                                    Title="{@id}"
                                    ShowLegend="1"
                                    OutsideMargin="3.2%"
                                    TitleToPlotMargin="3.%"
                                    LegendToPlotMargin="6.%"
                                    >
                                    <TitleFont
                                        Color="#023d7d"
                                        Name="Tahoma"
                                        Bold="1"
                                        Italic="0"
                                        Underline="0"
                                        MinFontHeight="10.pt"
                                        Size="8.%" />
                                </General>
                            </chart-config>
                        </xsl:variable>
                        <xsl:sequence select="altovaext:create-chart-config-from-
xml( $ext-chart-settings )"/>
                    </xsl:variable>
                    <xsl:variable name="chartDataSeries" as="item()*">
                        <xsl:variable name="chartDataRows" as="item()*">
                            <xsl:for-each select="(Year)">
                                <xsl:sequence select="altovaext:create-chart-data-
row( (@id), ( .) )"/>
                            </xsl:for-each>
                        </xsl:variable>
                        <xsl:variable name="chartDataSeriesNames" as="xs:string*"
select=" ( (&quot;Series 1&quot;), &apos;&apos; )[1]"/>
                        <xsl:sequence
                            select="altovaext:create-chart-data-series-from-
rows( $chartDataSeriesNames, $chartDataRows)"/>
                    </xsl:variable>
                    <xsl:variable name="ChartObj" select="altovaext:create-
chart( $extChartConfig, ( $chartDataSeries), false() )"/>
                    <xsl:variable name="sChartFileName" select="'mychart1.png'"/>
                    <img src="{$sChartFileName, altovaext:save-chart-
image( $ChartObj, $sChartFileName, 400, 400 ) }"/>
                </xsl:for-each>
            </body>
        </html>
    </xsl:template>
</xsl:stylesheet>
```

Documento XML

Este documento XML se puede procesar con el XSLT que aparece arriba. Los datos del documento XML se usan para generar el gráfico circular de la imagen que aparece más abajo.

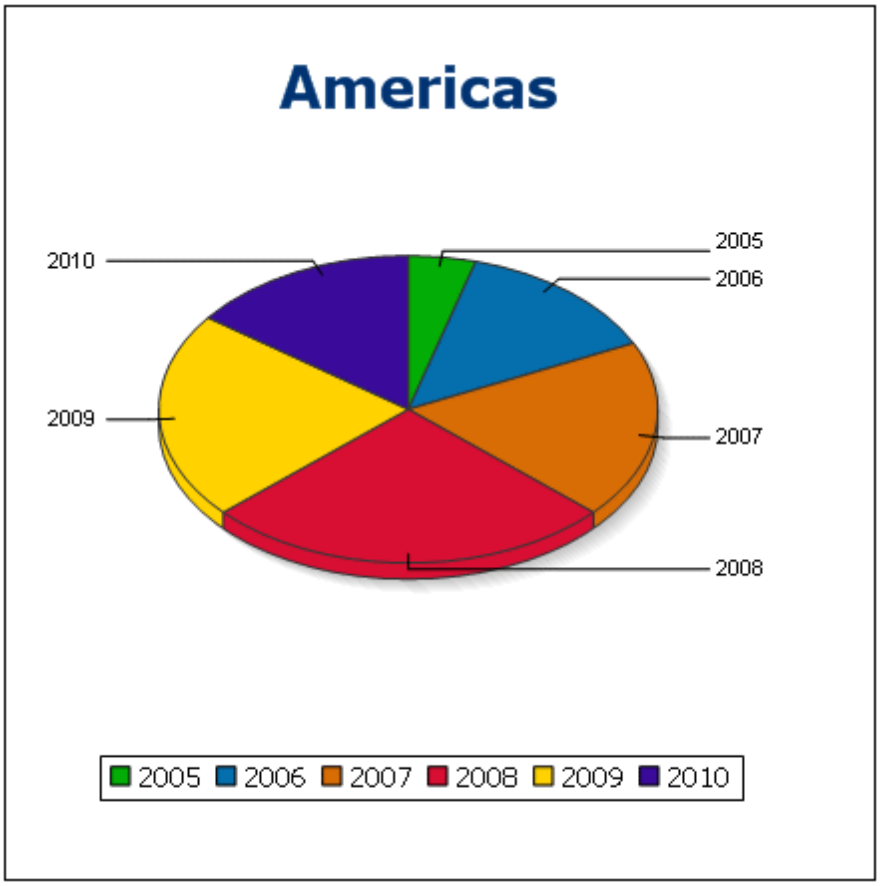
```
<?xml version="1.0" encoding="UTF-8"?>
<Data xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="YearlySales.xsd">
```

```
    <ChartType>Pie Chart 2D</ChartType>
    <Region id="Americas">
        <Year id="2005">30000</Year>
        <Year id="2006">90000</Year>
        <Year id="2007">120000</Year>
        <Year id="2008">180000</Year>
        <Year id="2009">140000</Year>
        <Year id="2010">100000</Year>
    </Region>
    <Region id="Europe">
        <Year id="2005">50000</Year>
        <Year id="2006">60000</Year>
        <Year id="2007">80000</Year>
        <Year id="2008">100000</Year>
        <Year id="2009">95000</Year>
        <Year id="2010">80000</Year>
    </Region>
    <Region id="Asia">
        <Year id="2005">10000</Year>
        <Year id="2006">25000</Year>
        <Year id="2007">70000</Year>
        <Year id="2008">110000</Year>
        <Year id="2009">125000</Year>
        <Year id="2010">150000</Year>
    </Region>
</Data>
```

Imagen de salida

Este gráfico circular se generó al procesar el documento XML anterior con el archivo XSLT que aparece al principio de este apartado.

© 2018 Altova GmbH



19.1.2.2 Funciones de extensión varias

Los lenguajes de programación como Java y C# ofrecen varias funciones predefinidas que no están disponibles como funciones XQuery/XPath ni XSLT. Un ejemplo son las funciones matemáticas de Java `sin()` y `cos()`. Si los diseñadores de hojas de estilos XSLT y consultas XQuery tuvieran acceso a estas funciones, el área de aplicación de sus hojas de estilos y consultas aumentaría y su trabajo sería un poco más sencillo.

Los motores XSLT y XQuery de los productos de Altova admiten el uso de funciones de extensión en Java y .NET, así como scripts MSXSL para XSLT.

Esta sección describe cómo usar funciones de extensión y scripts MSXSL en hojas de estilos XSLT y documentos XQuery. Las funciones de extensión pueden organizarse en varios grupos:

- Funciones de extensión Java
- Funciones de extensión .NET
- Scripts MSXSL para XSLT

En los apartados de esta sección nos ocupamos de tres aspectos fundamentales: (i) cómo se

llaman las funciones en sus respectivas bibliotecas, (ii) qué reglas deben seguirse para convertir los argumentos de una llamada a función en el formato de entrada necesario de la función y (iii) qué reglas deben seguirse para la conversión del tipo devuelto.

Requisitos

Para que estas funciones de extensión funcionen es necesario tener Java Runtime Environment (para las funciones Java) y .NET Framework 2.0 o superior (para las funciones .NET) instalado en el equipo que ejecuta la transformación XSLT o XQuery.

19.1.2.2.1 Funciones de extensión Java

Puede usar una función de extensión Java dentro de una expresión XPath o XQuery para invocar un constructor Java o llamar a un método Java (estático o de instancia).

Un campo de una clase Java se trata como un método sin argumentos. Un campo puede ser estático o de instancia. Más adelante describimos cómo se accede a los campos estáticos y de instancia.

Este apartado tiene varias partes:

- Constructores Java
- Métodos estáticos y campos estáticos
- Métodos de instancia y campos de instancia
- Tipos de datos: conversión de XPath/XQuery en Java
- Tipos de datos: conversión de Java en XPath/XQuery

Formato de la función de extensión

La función de extensión de la expresión XPath/XQuery debe tener este formato `prefijo:nombreFunción()`.

- La parte `prefijo:` identifica la función de extensión como función Java. Lo hace asociando la función de extensión con una declaración de espacio de nombres del ámbito, cuyo URI debe empezar por `java:` (*ver ejemplos más abajo*). La declaración de espacio de nombres debe identificar una clase Java, por ejemplo: `xmlns:myns="java:java.lang.Math"`. Sin embargo, también puede ser simplemente: `xmlns:myns="java"` (sin los dos puntos), dejando la identificación de la clase Java a la parte `nombreFunción()` de la función de extensión.
- La parte `nombreFunción()`identifica el método Java al que se llama y presenta los argumentos para el método (*ver ejemplos más abajo*). Sin embargo, si el URI de espacio de nombres identificado por la parte `prefijo:` no identifica una clase Java (*ver punto anterior*), entonces la clase Java debe identificarse en la parte `nombreFunción()`, antes de la clase y separada de la clase por un punto (*ver el segundo ejemplo XSLT que aparece más abajo*).

Nota: la clase a la que se llama debe estar en la ruta de acceso de clase del equipo.

© 2018 Altova GmbH

Ejemplo de código XSLT

Aquí ofrecemos dos ejemplos de cómo se puede llamar a un método estático. En el primer ejemplo, el nombre de la clase (`java.lang.Math`) se incluye en el URI de espacio de nombres y, por tanto, no puede estar en la parte `nombreFunción()`. En el segundo ejemplo, la parte `prefijo:` presenta el prefijo `java:` mientras que la parte `nombreFunción()` identifica la clase y el método.

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jmath="java"
              select="jmath:java.lang.Math.cos(3.14)" />
```

El método nombrado en la función de extensión (`cos()`) debe coincidir con el nombre de un método estático público de la clase Java nombrada (`java.lang.Math`).

Ejemplo de código XQuery

Aquí puede ver un ejemplo de código XQuery similar al código XSLT anterior:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

Clases Java definidas por el usuario

Si creó sus propias clases Java, a los métodos de estas clases se les llama de otra manera, dependiendo de: (i) si a las clases se accede por medio de un archivo JAR o de un archivo de clases y (ii) si estos archivos están en el directorio actual (el directorio del documento XSLT o XQuery). Para más información consulte los apartados Archivos de clases definidos por el usuario y Archivos Jar definidos por el usuario. Recuerde que debe especificar las rutas de acceso de los archivos de clases que no están en el directorio actual y de todos los archivos JAR.

19.1.2.2.1.1 Archivos de clases definidos por el usuario

Si se accede a las clases por medio de un archivo de clases, entonces hay cuatro posibilidades:

- El archivo de clases está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el paquete Java. (*ver ejemplo*.)
- El archivo de clases no está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el archivo de clases. (*ver ejemplo*.)
- El archivo de clases está en un paquete. El archivo XSLT/XQuery está en una carpeta cualquiera. (*ver ejemplo*.)

- El archivo de clases no está en un paquete. El archivo XSLT/XQuery está una carpeta cualquiera. (*ver ejemplo*.)

Imaginemos que tenemos un archivo de clases que no está en un paquete y que está en la misma carpeta que el documento XSLT/XQuery. En este caso, puesto que en la carpeta se encuentran todas las clases, no es necesario especificar la ubicación del archivo. La sintaxis que se utiliza para identificar una clase es esta:

```
java:nombreClase
```

donde

`java`: indica que se está llamando a una función definida por el usuario (por defecto se cargan las clases Java del directorio actual)
`nombreClase` es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método.

El archivo de clases está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el paquete Java

El código que aparece a continuación llama al método `getVehicleType()` de la clase `Car` del paquete `com.altova.extfunc`. El paquete `com.altova.extfunc` está en la carpeta `JavaProject`. El archivo XSLT también está en la carpeta `JavaProject`.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:com.altova.extfunc.Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

El archivo de clases no está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el archivo de clases

El código que aparece a continuación llama al método `getVehicleType()` de la clase `Car` del paquete `com.altova.extfunc`. El archivo de clases `Car` está en esta carpeta: `JavaProject/com/altova/extfunc`. El archivo XSLT también está en la carpeta `JavaProject/com/altova/extfunc`.

© 2018 Altova GmbH

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

El archivo de clases está en un paquete. El archivo XSLT/XQuery está en una carpeta cualquiera

El código que aparece a continuación llama al método `getCarColor()` de la clase `Car` del paquete `com.altova.extfunc`. El paquete `com.altova.extfunc` está en la carpeta `JavaProject`. El archivo XSLT está en otra carpeta cualquiera. En este caso debe especificarse la ubicación del paquete dentro del URI como una cadena de consulta. La sintaxis es esta:

```
java:nombreClase[?ruta=uri-del-paquete]
```

donde

`java`: indica que se está llamando a una función Java definida por el usuario
`uri-del-paquete` es el URI del paquete Java
`nombreClase` es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método. El ejemplo de código que aparece a continuación explica cómo se accede a un archivo de clases que está ubicado en un directorio que no es el directorio actual.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:com.altova.extfunc.Car?path=file:///C:/
JavaProject/" >

<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>
```

```
</xsl:stylesheet>
```

El archivo de clases no está en un paquete. El archivo XSLT/XQuery está una carpeta cualquiera

El código que aparece a continuación llama al método `getCarColor()`de la clase `Car` del paquete `com.altova.extfunc`. El paquete `com.altova.extfunc` está en la carpeta `JavaProject`. El archivo XSLT está en otra carpeta cualquiera. En este caso debe especificarse la ubicación del paquete dentro del URI como una cadena de consulta. La sintaxis es esta:

```
java:nombreClase[?ruta=uri-del-archivoClases]
```

donde

`java`: indica que se está llamando a una función Java definida por el usuario
`uri-del-archivoClases` es el URI de la carpeta donde se ubica el archivo de clases
`nombreClase` es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método. El ejemplo de código que aparece a continuación explica cómo se accede a un archivo de clases que está ubicado en un directorio que no es el directorio actual.

```
<xsl:stylesheet version="2.0"
      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:fn="http://www.w3.org/2005/xpath-functions"
      xmlns:car="java:Car?path=file:///C:/JavaProject/com/altova/
extfunc/" >

<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

Nota: cuando se presenta una ruta de acceso por medio de una función de extensión, la ruta de acceso se añade al ClassLoader.

19.1.2.2.1.2 Archivos JAR definidos por el usuario

Si se accede a las clases por medio de un archivo JAR, entonces se debe especificar el URI del archivo JAR usando esta sintaxis:

```
      xmlns:claseEspacioNombres="java:nombreClase?ruta=jar:uri-del-
archivoJar!/"
```


© 2018 Altova GmbH

Para la llamada al método se usa el prefijo del URI de espacio de nombres que identifica la clase: `claseEspacioNombres:método()`

En la sintaxis anterior:

`java:` indica que se está llamando a una función de Java
`nombreClase` es el nombre de la clase definida por el usuario
`?` es el separador entre el nombre de la clase y la ruta de acceso
`ruta=jar:` indica que se ofrece una ruta de acceso a un archivo JAR
`uri-del-archivoJar` es el URI del archivo JAR
`!/` es el delimitador final de la ruta de acceso
`claseEspacioNombres:método()` es la llamada al método

Otra opción es dar el nombre de la clase con la llamada al método. Por ejemplo:

```
xmlns:ns1="java:docx.layout.pages?path=jar:file:///c:/projects/
docs/docx.jar!/"
ns1:main()

xmlns:ns2="java?path=jar:file:///c:/projects/docs/docx.jar!/"
ns2:docx.layout.pages.main()
```

Y aquí puede ver un ejemplo de XSLT que usa un archivo JAR para llamar a una función de extensión Java:

```
<xsl:stylesheet version="2.0"
      xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
      xmlns:xs="http://www.w3.org/2001/XMLSchema"
      xmlns:fn="http://www.w3.org/2005/xpath-functions"
      xmlns:car="java?path=jar:file:///C:/test/Car1.jar!/" >
<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
      <xsl:variable name="myCar" select="car:Car1.new('red')" />
      <a><xsl:value-of select="car:Car1.getCarColor($myCar)"/></a>
</xsl:template>

<xsl:template match="car"/>

</xsl:stylesheet>
```

Nota: cuando se presenta una ruta de acceso por medio de una función de extensión, la ruta de acceso se añade al ClassLoader.

19.1.2.2.1.3 Constructores

Una función de extensión se puede usar para llamar a un constructor Java. A todos los constructores se les llama con la pseudofunción `new()`.

Si el resultado de una llamada a un constructor Java se puede convertir de manera implícita a tipos de datos XPath/XQuery, entonces la llamada a la función de extensión Java devuelve una secuencia que es un tipo de datos XPath/XQuery. Si el resultado de una llamada a un constructor Java no se puede convertir a un tipo de datos XPath/XQuery adecuado, entonces el constructor crea un objeto Java contenido con un tipo que es el nombre de la clase que devuelve ese objeto Java. Por ejemplo, si se llama a un constructor para la clase `java.util.Date` (`java.util.Date.new()`), entonces se devuelve un objeto que tiene el tipo `java.util.Date`. Puede que el formato léxico del objeto devuelto no coincida con el formato léxico de un tipo de datos XPath y, por tanto, su valor debe convertirse al formato léxico del tipo de datos XPath pertinente y después al tipo de datos XPath.

Puede hacer dos cosas con el objeto Java creado por un constructor:

- Puede asignar el objeto a una variable:
```
<xsl:variable name="currentdate" select="date:new()"
xmlns:date="java:java.util.Date" />
```
- Puede pasar el objeto a una función de extensión (*ver métodos de instancia y campos de instancia*):
```
<xsl:value-of select="date:toString(date:new())"
xmlns:date="java:java.util.Date" />
```

19.1.2.2.1.4 Métodos estáticos y campos estáticos

La llamada a un método estático la hace directamente su nombre Java y se hace presentando los argumentos para el método. A los campos estáticos (es decir, los métodos que no toman argumentos), como los campos de valor constante `E` y `PI`, se accede sin especificar ningún argumento.

Ejemplos de código XSLT

Aquí puede ver varios ejemplos de cómo se llama a métodos y campos estáticos:

```
<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos(3.14)" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:cos( jMath:PI() )" />

<xsl:value-of xmlns:jMath="java:java.lang.Math"
              select="jMath:E() * jMath:cos(3.14)" />
```

Observe que las funciones de extensión anteriores tienen el formato `prefijo:nombreFunción()`. En los tres ejemplos anteriores, el prefijo es `jMath:`, que está asociado al URI de espacio de nombres `java:java.lang.Math`. (El URI de espacio de nombres debe empezar por `java:`. En los ejemplos anteriores se extiende para contener el nombre de la clase (`java.lang.Math`).) La parte `nombreFunción()` de las funciones de extensión debe coincidir con el nombre de una clase pública (p. ej. `java.lang.Math`) seguido del nombre de un método estático público con sus argumentos (como `cos(3.14)`) o de un campo estático público (como `PI()`).

© 2018 Altova GmbH

En los tres ejemplos anteriores, el nombre de la clase se incluyó en el URI de espacio de nombres. Si no estuviera en el URI de espacio de nombres, se incluiría en la parte `nombreFunción()` de la función de extensión. Por ejemplo:

```
<xsl:value-of xmlns:java="java:"
              select="java:java.lang.Math.cos(3.14)" />
```

Ejemplo de XQuery

Un ejemplo de XQuery similar sería:

```
<cosine xmlns:jMath="java:java.lang.Math">
   {jMath:cos(3.14)}
</cosine>
```

19.1.2.2.1.5 Métodos de instancia y campos de instancia

A un método de instancia se le pasa un objeto Java como primer argumento de la llamada a método. Dicho objeto Java suele crearse usando una función de extensión (por ejemplo, una llamada a un constructor) o un parámetro o una variable de hoja de estilos. Un ejemplo de código XSLT de este tipo sería:

```
<xsl:stylesheet version="1.0" exclude-result-prefixes="date"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:date="java:java.util.Date"
  xmlns:jlang="java:java.lang">
  <xsl:param name="CurrentDate" select="date:new()"/>
  <xsl:template match="/">
      <enrollment institution-id="Altova School"
                date="{date:toString($CurrentDate)}"
                type="
{jlang:Object.toString(jlang:Object.getClass( date:new() ))}">
      </enrollment>
  </xsl:template>
</xsl:stylesheet>
```

En el ejemplo anterior el valor del nodo `enrollment/@type` se crea de la siguiente manera:

1. Se crea un objeto con un constructor para la clase `java.util.Date` (con el constructor `date:new()`).
2. Este objeto Java se pasa como argumento del método `jlang.Object.getClass`.
3. El objeto que obtiene el método `getClass` se pasa como argumento al método `jlang.Object.toString`.

El resultado (el valor de `@type`) será una cadena con este valor: `java.util.Date`.

En teoría, un campo de instancia es diferente de un método de instancia porque al campo de instancia no se pasa como argumento un objeto Java propiamente dicho. En su lugar se pasa como argumento un parámetro o variable. Sin embargo, el parámetro o la variable puede contener el valor devuelto por un objeto Java. Por ejemplo, el parámetro `CurrentDate` toma el valor que

devolvió un constructor para la clase `java.util.Date`. Este valor se pasa después como argumento al método de instancia `date:toString` a fin de suministrar el valor de `/enrollment/@date`.

19.1.2.2.1.6 Tipos de datos: conversión de XPath/XQuery en Java

Cuando se llama a una función Java desde dentro de una expresión XPath/XQuery, el tipo de datos de los argumentos de la función es importante a la hora de determinar a cuál de las clases Java que tienen el mismo nombre se llama.

En Java se siguen estas reglas:

- Si hay más de un método Java con el mismo nombre, pero cada método tiene un número diferente de argumentos, entonces se selecciona el método Java que mejor se ajusta al número de argumentos de la llamada a función.
- Los tipos de datos de cadena, numéricos y booleanos de XPath/XQuery (*ver lista más abajo*) se convierten de forma implícita en el tipo de datos Java correspondiente. Si el tipo XPath/XQuery suministrado se puede convertir a más de un tipo Java (p. ej. `xs:integer`), entonces se selecciona el tipo Java que se declaró para el método seleccionado. Por ejemplo, si el método Java al que se llama es `fx(decimal)` y el tipo de datos XPath/XQuery suministrado es `xs:integer`, entonces `xs:integer` se convierte en el tipo de datos Java `decimal`.

La tabla que aparece a continuación enumera las conversiones implícitas de los tipos de cadena, numéricos y booleanos XPath/XQuery en tipos de datos Java.

`xs:string`	`java.lang.String`
`xs:boolean`	`boolean` (primitivo), `java.lang.Boolean`
`xs:integer`	int, long, short, byte, float, double y sus clases contenedoras, como `java.lang.Integer`
`xs:float`	`float` (primitivo), `java.lang.Float`, `double` (primitivo)
`xs:double`	`double` (primitivo), `java.lang.Double`
`xs:decimal`	`float` (primitivo), `java.lang.Float`, `double`(primitivo), `java.lang.Double`

Los subtipos de los tipos de datos XML Schema de la tabla anterior (que se usan en XPath y XQuery) también se convierten en los tipos Java correspondientes al tipo antecesor del subtipo.

En algunos casos quizás no sea posible seleccionar el método Java correcto usando la información dada. Por ejemplo, imagine que:

- El argumento presentado es un valor `xs:untypedAtomic` de 10 y está destinado al método `mimétodo(float)`.
- Sin embargo, hay otro método en la clase que toma un argumento de otro tipo de datos: `mimétodo(double)`.
- Puesto que los métodos tienen el mismo nombre y el tipo suministrado

 © 2018 Altova GmbH

(`xs:untypedAtomic`) se puede convertir correctamente tanto en `float` como en `double`, es posible que `xs:untypedAtomic` se convierta en `double` en lugar de en `float`.

- Por consiguiente, el método seleccionado no será el método necesario y quizás no produzca el resultado esperado. Una solución es crear un método definido por el usuario con un nombre diferente y usar ese método.

Los tipos que no aparecen en la lista anterior (p. ej. `xs:date`) no se convertirán y generarán un error. No obstante, tenga en cuenta que en algunos casos, es posible crear el tipo Java necesario usando un constructor Java.

19.1.2.2.1.7 Tipos de datos: conversión de Java en XPath/XQuery

Cuando un método Java devuelve un valor y el tipo de datos del valor es un tipo de cadena, numérico o booleano, entonces se convierte en el tipo de datos XPath/XQuery correspondiente. Por ejemplo, los tipos de datos Java `java.lang.Boolean` y `boolean` se convierten en `xsd:boolean`.

Las matrices unidimensionales devueltas por las funciones se extienden en una secuencia. Las matrices multidimensionales no se convierten y, por tanto, deberían ser contenidas.

Cuando se devuelve un objeto Java contenido o un tipo de datos que no es de cadena, numérico ni booleano, puede garantizar la conversión del tipo XPath/XQuery necesario usando primero un método Java (p. ej. `toString`) para convertir el objeto Java en una cadena. En XPath/XQuery la cadena se puede modificar para ajustarse a la representación léxica del tipo necesario y convertirse después en dicho tipo (usando la expresión `cast as`, por ejemplo).

19.1.2.2.2 Funciones de extensión .NET

Si trabaja en la plataforma .NET desde un equipo Windows, puede usar funciones de extensión escritas en cualquier lenguaje .NET (p. ej. C#). Una función de extensión .NET se puede usar dentro de una expresión XPath/XQuery para invocar un constructor, una propiedad o un método (estático o de instancia) de una clase .NET.

A una propiedad de una clase .NET se le llama usando la sintaxis `get_NombrePropiedad()`.

Este apartado tiene varias partes:

- Constructores
- Métodos estáticos y campos estáticos
- Métodos de instancia y campos de instancia
- Tipos de datos: conversión de XPath/XQuery en .NET
- Tipos de datos: conversión de .NET en XPath/XQuery

Formato de la función de extensión

La función de extensión de la expresión XPath/XQuery debe tener este formato `prefijo:nombreFunción()`.

- La parte `prefijo:` está asociada a un URI que identifica la clase .NET.
- La parte `nombreFunción()` identifica el constructor, la propiedad o el método (estático o de instancia) dentro de la clase .NET y, si es necesario, suministra los argumentos.
- El URI debe empezar por `clitype:` (que identifica la función como función de extensión .NET).
- El formato `prefijo:nombreFunción()` de la función de extensión se puede usar con clases del sistema y con clases de un ensamblado cargado. No obstante, si se tiene que cargar una clase, será necesario suministrar parámetros que contengan la información necesaria.

Parámetros

Para cargar un ensamblado se usan estos parámetros:

`asm`	El nombre del ensamblado que se debe cargar.
`ver`	El número de versión (máximo cuatro enteros separados por puntos).
`sn`	El símbolo de clave del nombre seguro del ensamblado (16 dígitos hexadecimales).
`from`	Un URI que da la ubicación del ensamblado (DLL) que se debe cargar. Si el URI es relativo, es relativo al archivo XSLT o XQuery. Si está presente este parámetro, se ignoran los demás parámetros.
`partialname`	El nombre parcial del ensamblado. Se suministra a `Assembly.LoadWith.PartialName()`, que intentará cargar el ensamblado. Si está presente el parámetro `partialname`, se ignoran los demás parámetros.
`loc`	La configuración regional, por ejemplo, `en-US`. La configuración predeterminada es `neutral`.

Si el ensamblado se debe cargar desde un archivo DLL, use el parámetro `from` y omita el parámetro `sn`. Si el ensamblado se debe cargar desde el caché general de ensamblados (GAC), use el parámetro `sn` y omita el parámetro `from`.

Debe insertar un signo de interrogación final antes del primer parámetro y los parámetros deben separarse con un punto y coma (;). El nombre de parámetro da su valor con un signo igual (=), como en el ejemplo que aparece más abajo.

Ejemplos de declaraciones de espacios de nombres

Esto es un ejemplo de una declaración de espacio de nombres en XSLT que identifica la clase del sistema `System.Environment`:

© 2018 Altova GmbH

```
xmlns:myns="clitype:System.Environment"
```

Esto es un ejemplo de una declaración de espacio de nombres en XSLT que identifica la clase que se debe cargar como `Trade.Forward.Scrip`:

```
xmlns:myns="clitype:Trade.Forward.Scrip?asm=forward;version=10.6.2.1"
```

Esto es un ejemplo de una declaración de espacio de nombres en XQuery que identifica la clase del sistema `MyManagedDLL.testClass`:. Existen dos tipos de clases:

1. Cuando el ensamblado se carga desde el GAC:
   ```
   declare namespace cs="clitype:MyManagedDLL.testClass?asm=MyManagedDLL;
          ver=1.2.3.4;loc=neutral;sn=b9f091b72dccfba8";
   ```

2. Cuando el ensamblado se carga desde el archivo DLL (ver las referencias parciales y completas):
   ```
          declare namespace cs="clitype:MyManagedDLL.testClass?from=file:///
   C:/Altova
          Projects/extFunctions/MyManagedDLL.dll;

      declare namespace cs="clitype:MyManagedDLL.testClass?
      from=MyManagedDLL.dll;
   ```

Ejemplo de código XSLT

Aquí puede ver un ejemplo de código XSLT que llama a funciones de la clase del sistema `System.Math`:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes" />
   <xsl:template match="/">
      <math xmlns:math="clitype:System.Math">
         <sqrt><xsl:value-of select="math:Sqrt(9)"/></sqrt>
         <pi><xsl:value-of select="math:PI()"/></pi>
         <e><xsl:value-of select="math:E()"/></e>
         <pow><xsl:value-of select="math:Pow(math:PI(), math:E())"/></pow>
      </math>
   </xsl:template>
</xsl:stylesheet>
```

La declaración de espacio de nombres del elemento `math` asocia el prefijo `math:` al URI `clitype:System.Math`. La parte inicial `clitype:` del URI indica que lo que sigue identifica una clase del sistema o una clase cargada. El prefijo `math:` de las expresiones XPath asocia las funciones de extensión al URI (y, por extensión, a la clase) `System.Math`. Las funciones de extensión identifican métodos en la clase `System.Math` y presenta argumentos cuando es necesario.

Ejemplo de código XQuery

Aquí puede ver un fragmento de código XQuery similar al ejemplo anterior:

```
<math xmlns:math="clitype:System.Math">
   {math:Sqrt(9)}
</math>
```

Tal y como ocurre con el código XSLT anterior, la declaración de espacio de nombres identifica la clase .NET, en este caso una clase del sistema. La expresión XQuery identifica el método al que se debe llamar y presenta el argumento.

19.1.2.2.2.1 Constructores

Una función de extensión se puede usar para llamar a un constructor .NET. A todos los constructores se les llama con la pseudofunción `new()`. Si hay más de un constructor para una clase, entonces se selecciona el constructor que más se ajusta al número de argumentos suministrados. Si no se encuentra ningún constructor que coincida con los argumentos suministrados, entonces se genera el error `"No constructor found"`.

Constructores que devuelven tipos de datos XPath/XQuery

Si el resultado de una llamada a un constructor .NET se puede convertir de forma implícita en tipos de datos XPath/XQuery, entonces la función de extensión .NET devuelve una secuencia que es un tipo de datos XPath/XQuery.

Constructores que devuelven objetos .NET

Si el resultado de una llamada a un constructor .NET no se puede convertir a un tipo de datos XPath/XQuery adecuado, entonces el constructr crea un objeto .NET contenido con un tipo que es el nombre de la clase que devuelve dicho objeto. Por ejemplo, si se llama al constructor para la clase `System.DateTime` (con `System.DateTime.new()`), entonces se devuelve un objeto que tiene un tipo `System.DateTime`.

Puede que el formato léxico del objeto devuelto no coincida con el formato léxico de un tipo de datos XPath. En estos casos, el valor devuelto (i) debe convertirse al formato léxico del tipo de datos XPath pertinente y (ii) debe convertirse en el tipo de datos XPath necesario.

Se pueden hacer tres cosas con un objeto .NET creado con un constructor:

- Se puede usar dentro de una variable:
  ```
  <xsl:variable name="currentdate" select="date:new(2008, 4, 29)"
  xmlns:date="clitype:System.DateTime" />
  ```
- Se puede pasar a una función de extensión (ver *Métodos de instancia y campos de instancia*):

```
<xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
xmlns:date="clitype:System.DateTime" />
```

- Se puede convertir en un tipo de cadena, numérico o booleano:

```
<xsl:value-of select="xs:integer(data:get_Month(date:new(2008, 4, 29)))"
xmlns:date="clitype:System.DateTime" />
```

19.1.2.2.2.2 Metodos estáticos y campos estáticos

La llamada a un método estático la hace directamente su nombre y se hace presentando los argumentos para el método. El nombre usado en la llamada debe ser el mismo que un método estático público de la clase especificada. Si el nombre del método y el número de argumentos que se dio en la llamada a función coincide con algún método de la clase, entonces los tipos de los argumentos presentados se evalúan para encontrar el resultado ideal. Si no se encuentra ninguna coincidencia, se emite un error.

Nota: un campo de una clase .NET se trata como si fuera un método sin argumentos. Para llamar a una propiedad se usa la sintaxis `get_nombrePropiedad()`.

Ejemplos

Este ejemplo de código XSLT muestra una llamada a un método con un argumento (`System.Math.Sin(arg)`):

```
<xsl:value-of select="math:Sin(30)" xmlns:math="clitype:System.Math"/>
```

Este ejemplo de código XSLT muestra una llamada a un campo (que se trata como si fuera un método sin argumentos) (`System.Double.MaxValue()`):

```
<xsl:value-of select="double:MaxValue()"
xmlns:double="clitype:System.Double"/>
```

Este ejemplo de código XSLT muestra una llamada a una propiedad (la sintaxis es `get_nombrePropiedad()`) (`System.String()`):

```
<xsl:value-of select="string:get_Length('my string')"
xmlns:string="clitype:System.String"/>
```

Este ejemplo de código XQuery muestra una llamada a un método con un argumento (`System.Math.Sin(arg)`):

```
<sin xmlns:math="clitype:System.Math">
   { math:Sin(30) }
</sin>
```

19.1.2.2.2.3 Métodos de instancia y campos de instancia

Un método de instancia es un método al que se le pasa un objeto .NET como primer argumento de la llamada al método. Este objeto .NET se suele crear usando una función de extensión (por ejemplo, una llamada a un constructor) o un parámetro o una variable de una hoja de estilos. Un ejemplo de código XSLT para este tipo de método sería:

```
<xsl:stylesheet version="2.0"
   xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions">
   <xsl:output method="xml" omit-xml-declaration="yes"/>
   <xsl:template match="/">
      <xsl:variable name="releasedate"
         select="date:new(2008, 4, 29)"
         xmlns:date="clitype:System.DateTime"/>
      <doc>
         <date>
            <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
               xmlns:date="clitype:System.DateTime"/>
         </date>
         <date>
            <xsl:value-of select="date:ToString($releasedate)"
               xmlns:date="clitype:System.DateTime"/>
         </date>
      </doc>
   </xsl:template>
</xsl:stylesheet>
```

En el ejemplo anterior, se usó un constructor `System.DateTime` (`new(2008, 4, 29)`) para crear un objeto .NET de tipo `System.DateTime`. Este objeto se creó dos veces, una vez como valor de la variable `releasedate`, y otra vez como primer y único argumento del método `System.DateTime.ToString()`. Al método de instancia `System.DateTime.ToString()` se le llama dos veces, ambas con el constructor `System.DateTime` (`new(2008, 4, 29)`) como primer y único argumento. En una de estas instancias, se usó la variable `releasedate` para obtener el objeto .NET.

Métodos de instancia y campos de instancia

La diferencia entre un método de instancia y un campo de instancia es solo teórica. En un método de instancia, se pasa directamente un objeto .NET como argumento. En un campo de instancia, se pasa un parámetro o una variable (aunque el parámetro o la variable puede contener un objeto .NET). Por ejemplo, en el código del ejemplo anterior, la variable `releasedate` contiene un objeto .NET y esta es la variable que se pasa como argumento de `ToString()` en el segundo constructor de elemento `date`. Por tanto, la instancia `ToString()` del primer elemento `date` es un método de instancia, mientras que la segunda se considera un campo de instancia. El resultado es el mismo en ambos casos.

© 2018 Altova GmbH

19.1.2.2.2.4 Tipos de datos: conversión de XPath/XQuery en .NET

Cuando se usa una función de extensión .NET dentro de una expresión XPath/XQuery, los tipos de datos de los argumentos de la función son importantes para determinar a cuál de los métodos .NET que tienen el mismo nombre se está llamando.

En .NET se siguen estas normas:

- Si en una clase hay varios métodos que tienen el mismo nombre, solamente se pueden seleccionar los métodos que tienen el mismo número de argumentos que la llamada a función.
- Los tipos de datos de cadena, numéricos y booleanos XPath/XQuery (*ver lista más abajo*) se convierten de forma implícita en el tipo de datos .NET correspondiente. Si el tipo XPath/XQuery suministrado se puede convertir en más de un tipo .NET (p. ej. `xs:integer`), entonces se selecciona el tipo .NET que se declaró para el método seleccionado. Por ejemplo, si el método .NET al que se está llamando es `fx(double)` y el tipo de datos XPath/XQuery suministrado es `xs:integer`, entonces se convierte `xs:integer` en el tipo de datos .NET `double`.

La tabla que aparece a continuación enumera las conversiones implícitas de los tipos de cadena, numéricos y booleanos XPath/XQuery en tipos de datos .NET.

`xs:string`	`StringValue, string`
`xs:boolean`	`BooleanValue, bool`
`xs:integer`	`IntegerValue, decimal, long, integer, short, byte, double, float`
`xs:float`	`FloatValue, float, double`
`xs:double`	`DoubleValue, double`
`xs:decimal`	`DecimalValue, decimal, double, float`

Los subtipos de los tipos de datos XML Schema de la tabla anterior (que se usan en XPath y XQuery) también se convierten en los tipos .NET correspondientes al tipo antecesor del subtipo.

En algunos casos quizás no sea posible seleccionar el método .NET correcto usando la información dada. Por ejemplo, imagine que:

- El argumento presentado es un valor `xs:untypedAtomic` de 10 y está destinado al método `mimétodo(float)`.
- Sin embargo, hay otro método en la clase que toma un argumento de otro tipo de datos: `mimétodo(double)`.
- Puesto que los métodos tienen el mismo nombre y el tipo suministrado (`xs:untypedAtomic`) se puede convertir correctamente tanto en `float` como en `double`, es posible que `xs:untypedAtomic` se convierta en `double` en lugar de en `float`.
- Por consiguiente, el método seleccionado no será el método necesario y puede que no produzca el resultado esperado. Una solución es crear un método definido por el usuario con un nombre diferente y usar ese método.

Los tipos que no aparecen en la lista anterior (p. ej. `xs:date`) no se convertirán y generarán un

error.

19.1.2.2.5 Tipos de datos: conversión de .NET en XPath/XQuery

Cuando un método .NET devuelve un valor y el tipo de datos del valor es un tipo de cadena, numérico o booleano, entonces se convierte en el tipo de datos XPath/XQuery correspondiente. Por ejemplo, el tipo de datos .NET `decimal` se convierte en `xsd:decimal`.

Cuando se devuelve un objeto .NET o un tipo de datos que no es de cadena, numérico ni booleano, puede garantizar la conversión del tipo XPath/XQuery necesario usando primero un método .NET (p. ej. `System.DateTime.ToString()`) para convertir el objeto .NET en una cadena. En XPath/XQuery la cadena se puede modificar para ajustarse a la representación léxica del tipo necesario y convertirse después en dicho tipo (usando la expresión `cast as`, por ejemplo).

19.1.2.2.3 Scripts MSXSL para XSLT

El elemento `<msxsl:script>` contiene funciones y variables definidas por el usuario a las que se puede llamar desde dentro de expresiones XPath en la hoja de estilos XSLT. El elemento `<msxsl:script>` es un elemento de nivel superior, es decir, debe ser un elemento secundario de `<xsl:stylesheet>` o `<xsl:transform>`.

El elemento `<msxsl:script>` debe estar en el espacio de nombres `urn:schemas-microsoft-com:xslt` (*ver ejemplo más abajo*).

Lenguaje de scripting y espacio de nombres

El lenguaje de scripting utilizado dentro del bloque se especifica en el atributo `language` del elemento `<msxsl:script>` y el espacio de nombres que se debe usar para las llamadas a función desde expresiones XPath se identifica con el atributo `implements-prefix`:

```
<msxsl:script language="lenguaje-de-scripting" implements-prefix="prefijo-
espacioNombres-usuario">

   función-1 o variable-1
   ...
   función-n o variable-n

</msxsl:script>
```

El elemento `<msxsl:script>` interactúa con Windows Scripting Runtime, de modo que dentro del elemento `<msxsl:script>` solamente se pueden usar lenguajes que estén instalados en el equipo. Para poder usar scripts MSXSL es necesario tener **instalada la plataforma .NET Framework 2.0 (o superior)**. Por tanto, los lenguajes de scripting .NET se pueden usar dentro del elemento `<msxsl:script>`.

© 2018 Altova GmbH

El atributo `language` admite los mismos valores que el atributo `language` del elemento HTML `<script>`. Si no se especifica el atributo `language`, entonce se asume Microsoft JScript por defecto.

El atributo `implements-prefix` toma un valor que es un prefijo de un espacio de nombres declarado dentro del ámbito. Este espacio de nombres suele ser un espacio de nombres de usuario que se reservó para una biblioteca de funciones. Todas las funciones y variables definidas dentro del elemento `<msxsl:script>` están en el espacio de nombres identificado por el prefijo indicado en el atributo `implements-prefix`. Cuando se llama a una función desde dentro de una expresión XPath, el nombre de función completo debe estar en el mismo espacio de nombres que la definición de función.

Ejemplo

Aquí puede ver un ejemplo de una hoja de estilos XSLT que usa una función definida dentro de un elemento `<msxsl:script>`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   xmlns:xs="http://www.w3.org/2001/XMLSchema"
   xmlns:fn="http://www.w3.org/2005/xpath-functions"
   xmlns:msxsl="urn:schemas-microsoft-com:xslt"
   xmlns:user="http://mycompany.com/mynamespace">

  <msxsl:script language="VBScript" implements-prefix="user">
    <![CDATA[
    ' Input: A currency value: the wholesale price
    ' Returns: The retail price: the input value plus 20% margin,
    ' rounded to the nearest cent
    dim a as integer  = 13
    Function AddMargin(WholesalePrice) as integer
      AddMargin = WholesalePrice * 1.2 + a
    End Function
  ]]>
  </msxsl:script>

  <xsl:template match="/">
    <html>
      <body>
        <p>
          <b>Total Retail Price =
            $<xsl:value-of select="user:AddMargin(50)"/>
          </b>
          <br/>
          <b>Total Wholesale Price =
            $<xsl:value-of select="50"/>
          </b>
        </p>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

Tipos de datos

Los valores de los parámetros que se pasan dentro y fuera del bloque de script solamente pueden ser tipos de datos XPath. Esta restricción no afecta a los datos que se pasan las funciones y variables situadas dentro del bloque de script.

Ensamblados

Puede importar un ensamblado al script usando el elemento `msxsl:assembly`. El ensamblado se identifica con un nombre o un URI. El ensamblado se importa cuando se compila la hoja de estilos. Aquí puede ver cómo se usa el elemento `msxsl:assembly`:

```
<msxsl:script>
       <msxsl:assembly name="miEnsamblado.nombreEnsamblado" />
       <msxsl:assembly href="rutaDelEnsamblado" />

       ...

</msxsl:script>
```

El nombre de ensamblado puede ser un nombre completo, como:

```
"system.Math, Version=3.1.4500.1 Culture=neutral
PublicKeyToken=a46b3f648229c514"
```

o un nombre abreviado, como `"miEnsamblado.Draw"`.

Espacios de nombres

Puede declarar espacios de nombres con el elemento `msxsl:using`. Esto permite escribir las clases del ensamblado en el script sin sus espacios de nombres, lo cual le permitirá ahorrar mucho tiempo. Aquí puede ver cómo se usa el elemento `msxsl:using` para declarar espacios de nombres.

```
<msxsl:script>
       <msxsl:using namespace="ENmiEnsamblado.NombreEspaciodenombres" />

       ...

</msxsl:script>
```

El valor del atributo `namespace` es el nombre del espacio de nombres.

© 2018 Altova GmbH

19.2 Datos técnicos

Esta sección incluye información general práctica sobre aspectos técnicos de su software. Consta de varios apartados:

- Requisitos de OS y memoria
- Analizador XML de Altova
- Motores XSLT y XQuery de Altova
- Compatibilidad con Unicode
- Uso de Internet

19.2.1 Requisitos de SO y memoria

Sistema operativo

Las aplicaciones de software de Altova están disponibles en estas plataformas:

- Windows 7 SP1 con actualización de la plataforma, Windows 8, Windows 10
- Windows Server 2008 R2 SP1 con actualización de la plataforma o superior

Memoria

Puesto que el software está escrito en C++ no necesita tanto espacio como un JRE y suele necesitar menos memoria que otras aplicaciones similares basadas en Java. No obstante, todos los documentos se cargan en memoria por completo, para poder analizarlos completamente y mejorar la velocidad de visualización y edición. Los requisitos de memoria aumentan en función del tamaño del documento.

Los requisitos de memoria también vienen dados por el historial de operaciones Deshacer. Cuando se cortan y pegan secciones grandes de documentos de gran tamaño, la memoria disponible se puede agotar rápidamente.

19.2.2 Validador XML de Altova

Al abrir un documento XML, la aplicación usa su validador integrado para comprobar si el formato XML es correcto, validar el documento con un esquema (si viene dado) y generar árboles e infosets. El validador XML también se usa para ofrecerle ayuda de edición inteligente mientras edita documentos y para mostrar de forma dinámica los errores de validación (si los hay).

El validador XML implementa la recomendación final de la especificación XML Schema 1.0 y 1.1 del W3C. Además Altova actualiza continuamente el validador los avances del grupo de trabajo XML Schema del W3C.

19.2.3 Motores XSLT y XQuery de Altova

Los productos de Altova usan los motores XSLT 1.0, XSLT 2.0, XSLT 3.0, XQuery 1.0 y XQuery 3.0 de Altova. Para más información sobre el comportamiento de cada motor, consulte el anexo *Información sobre motores de Altova.*

Nota: Altova MapForce genera código con los motores XSLT 1.0, XSLT 2.0 y XQuery 1.0.

19.2.4 Compatibilidad con Unicode

Los productos XML de Altova son completamente compatibles con Unicode. Para editar un documento XML también necesitará una fuente compatible con los caracteres Unicode utilizados por el documento.

Tenga en cuenta que la mayoría de las fuentes contienen solamente un subconjunto muy concreto de caracteres Unicode y, por tanto, están destinadas a un sistema de escritura concreto. Si algunos caracteres aparecen desfigurados, el motivo puede ser que la fuente seleccionada no contiene los glifos necesarios. Por tanto, es recomendable tener una fuente que abarque todos los caracteres Unicode. Sobre todo si edita documentos XML en varios idiomas o sistemas de escritura. Una fuente Unicode que suele venir con los equipos Windows es la fuente Arial Unicode MS.

En la carpeta `/Examples` de la carpeta de su aplicación puede encontrar un archivo XHTML llamado `UnicodeUTF-8.html` que incluye esta frase en gran número de idiomas y sistemas de escritura diferentes:

- *When the world wants to talk, it speaks Unicode*
- *Cuando el mundo quiere conversar, habla Unicode*
- *Wenn die Welt miteinander spricht, spricht sie Unicode*
- 世界的に話すなら、Unicode です。

Abra este archivo XHTML y observe el potencial de Unicode.

19.2.5 Uso de Internet

Las aplicaciones de Altova inician conexiones a Internet en estos casos:

- Si hace clic en el botón **Solicitar una clave de evaluación GRATUITA** del cuadro de diálogo "Activación del software" (**Ayuda | Activación del software**), los campos del cuadro de diálogo de activación del software se transfieren a nuestro servidor web por medio de una conexión HTTP corriente (puerto 80) y le enviamos el código de evaluación gratuito por correo electrónico.
- En algunos productos de Altova puede abrir un archivo por Internet (**Archivo | Abrir | Cambiar a URL**). En este caso, el documento se recupera usando uno de estos protocolos y conexiones: HTTP (normalmente por el puerto 80), FTP (normalmente por el puerto 20/21) o HTTPS (normalmente por el puerto 443). También puede ejecutar un servidor HTTP en el puerto 8080. (En el cuadro de diálogo "Abrir URL", después del

© 2018 Altova GmbH

nombre de servidor escriba dos puntos y el número de puerto.)

- Si abre un documento XML que hace referencia a un documento DTD o esquema XML y el documento se especifica a través de una URL, el documento de esquema al que se hace referencia también se recupera a través de una conexión HTTP (puerto 80) o cualquier otro protocolo (ver punto anterior). El documento de esquema también se recupera para validar el archivo XML. Recuerde que la validación puede realizarse automáticamente nada más abrir el documento, si seleccionó esta opción en la sección *Archivo* del cuadro de diálogo "Opciones" (**Herramientas | Opciones**).
- En las aplicaciones de Altova que trabajen con WSDL y SOAP, las conexiones a servicios web son definidas por documentos WSDL.
- Si usa el comando **Archivo | Enviar por correo electrónico** de MapForce, el texto seleccionado actualmente o el archivo se envía con el programa de correo electrónico instalado en el equipo.
- Durante la activación del software y la búsqueda de actualizaciones, tal y como se describe en el contrato de licencia de software de Altova.

19.3 Información sobre licencias

En esta sección encontrará:

- información sobre la distribución de este producto de software
- información sobre activación del software y medición de licencias
- información sobre derechos de propiedad intelectual de este producto de software
- el contrato de licencia para el usuario final que rige el uso de este producto de software

Los términos del contrato de licencia que aceptó al instalar el producto de software son vinculantes, por lo que rogamos lea atentamente toda esta información.

19.3.1 Distribución electrónica de software

Este producto está disponible por distribución electrónica de software, un método de distribución que ofrece ventajas únicas:

- Puede evaluar el software de forma totalmente gratuita antes de decidir si compra el producto.
- Si decide comprarlo, puede hacer un pedido en línea en el sitio web de Altova y conseguir en pocos minutos el software con licencia.
- Si realiza el pedido en línea, siempre recibirá la versión más reciente de nuestro software.
- El paquete de instalación del producto incluye un sistema de ayuda en pantalla totalmente integrado. La versión más reciente del manual del usuario está disponible en www.altova.com (i) en formato HTML y (ii) en formato PDF para descargar e imprimir si lo desea.

Período de evaluación de 30 días

Después de descargar el producto de software, puede probarlo de forma totalmente gratuita durante un plazo de 30 días. Pasados unos 20 días, el software empieza a recordarle que no tiene una licencia. El mensaje de aviso aparece una sola vez cada vez que se inicie la aplicación. Para seguir utilizando el programa una vez pasado el plazo de 30 días, deberá comprar una licencia permanente y aceptar el contrato de licencia de software de Altova, que se entrega en forma de código clave. La licencia puede comprarse directamente en la tienda en línea del sitio web de Altova. Después de comprar la licencia recibirá el código clave, que debe introducir en el cuadro de diálogo "Activación del software" para desbloquear el producto de forma permanente.

Distribuir la versión de evaluación a otros usuarios de su organización

Si desea distribuir la versión de evaluación en la red de su compañía o si desea usarlo en un PC que no está conectado a Internet, solamente puede distribuir los programas de instalación (siempre y cuando no se modifiquen de forma alguna). Todo usuario que acceda al instalador debe solicitar su propio código clave de evaluación (de 30 días). Una vez pasado este plazo de 30

© 2018 Altova GmbH

días, todos los usuarios deben comprar también una licencia para poder seguir usando el producto.

Para más información consulte el contrato de licencia de software de Altova que aparece al final de esta sección.

19.3.2 Activación del software y medición de licencias

Durante el proceso de activación del software de Altova, puede que la aplicación utilice su red interna y su conexión a Internet para transmitir datos relacionados con la licencia durante la instalación, registro, uso o actualización del software a un servidor de licencias operado por Altova y para validar la autenticidad de los datos relacionados con la licencia y proteger a Altova de un uso ilegítimo del software y mejorar el servicio a los clientes. La activación es posible gracias al intercambio de datos de la licencia (como el sistema operativo, la dirección IP, la fecha y hora, la versión del software, el nombre del equipo, etc.) entre su equipo y el servidor de licencias de Altova.

Su producto incluye un módulo integrado de medición de licencias que le ayudará a evitar infracciones del contrato de licencia para el usuario final. Puede comprar una licencia de un solo usuario o de varios usuarios para el producto de software y el módulo de medición de licencias se asegura de que no se utiliza un número de licencias mayor al permitido.

Esta tecnología de medición de licencias usa su red de área local (LAN) para comunicarse con las instancias de la aplicación que se ejecutan en equipos diferentes.

Licencia de un solo usuario

Cuando se inicia la aplicación, se inicia el proceso de medición de licencias y el software envía un breve datagrama de multidifusión para averiguar si hay otras instancias del producto activas en otros equipos del mismo segmento de red al mismo tiempo. Si no recibe ninguna respuesta, la aplicación abre un puerto para escuchar a otras instancias de la aplicación.

Licencia de varios usuarios

Si se usa más de una instancia de la aplicación dentro de la misma red LAN, estas instancias se comunicarán entre ellas al iniciarse. Estas instancias intercambian códigos claves para que ayudarle a no sobrepasar por error el número máximo de licencias concurrentes. Se trata de la misma tecnología de medición de licencias que suele utilizarse en Unix y en otras herramientas de desarrollo de bases de datos. Gracias a ella puede comprar licencias de varios usuarios de uso concurrente a un precio razonable.

Las aplicaciones se diseñaron de tal modo que envían pocos paquetes pequeños de red y no cargan demasiado su red. Los puertos TCP/IP (2799) utilizados por su producto de Altova están registrados oficialmente en la IANA (para más información consulte el sitio web de la IANA www.iana.org) y nuestro módulo de medición de licencias es una tecnología probada y eficaz.

Si usa un servidor de seguridad, puede notar las comunicaciones del puerto 2799 entre los equipos que ejecutan los productos de Altova. Si quiere, puede bloquear ese tráfico, siempre y cuando esto no resulte en una infracción del contrato de licencia.

También notará que su producto de Altova ofrece varias funciones prácticas si está conectado a Internet. Estas funciones no tienen nada que ver con la tecnología de medición de licencias.

19.3.3 Derechos de propiedad intelectual

El software de Altova y sus copias (si tiene permiso de Altova para realizar copias) es propiedad intelectual de Altova y de sus proveedores. La estructura, la organización y el código del software se considera secreto comercial e información confidencial de Altova y de sus proveedores. El software está protegido por las leyes de derechos de autor, como la ley de derechos de autor de EE UU, tratados internacionales y la legislación vigente del país donde se utiliza, entre otras. Altova conserva los derechos de propiedad de todas las patentes, derechos de autor, secretos comerciales, marcas registradas y otros derechos de propiedad intelectual pertenecientes al software y los derechos de propiedad de Altova abarcan también imágenes, fotografías, animaciones, vídeos, audio, música, texto y otros applets incorporados al software y al material impreso que viene con el software. Las notificaciones de infracción de dichos derechos de autor debe enviarse al agente de derechos de autor de Altova, cuyos datos de contacto aparecen en el sitio web de Altova.

El software de Altova contiene software de terceros que también está protegido por las leyes de propiedad intelectual, incluida, entre otras, la legislación de derechos de autor mencionada en http://www.altova.com/es/legal_3rdparty.html.

Los demás nombres y marcas registradas son propiedad de sus respectivos propietarios.

19.3.4 Contrato de licencia para el usuario final

- Altova End User License Agreement: http://www.altova.com/eula
- Altova Privacy Policy: http://www.altova.com/privacy

© 2018 Altova GmbH

Altova MapForce 2018 Enterprise Edition

Glosario

20 Glosario

Este glosario incluye una lista de términos utilizados en MapForce.

© 2018 Altova GmbH

20.1 A

Asignación de datos

Un diseño de asignación de datos (o simplemente *asignación de datos*) es la representación visual de cómo se deben transformar datos de un formato a otro. Una asignación está formada por componentes, que se van añadiendo al área de asignación de MapForce para crear las transformaciones de datos (p. ej. para convertir documentos XML que siguen un esquema en documentos que siguen otro esquema). Una asignación válida está formada por uno o varios componentes de origen que están conectados a uno o varios componentes de destino. La asignación se puede ejecutar y MapForce ofrece una vista previa del resultado. También puede generar código desde MapForce y ejecutar la asignación en una aplicación externa. Por último, puede compilar la asignación en un archivo de ejecución de MapForce y automatizar la ejecución con MapForce Server o FlowForce Server. Los diseños de asignación de MapForce tienen la extensión de archivo **.mfd**.

20.2 C

Componente

En MapForce un componente es el elemento gráfico que representa visualmente la estructura (esquema) de los datos o el elemento que determina cómo se deben transformar los datos (funciones). Los componentes son piezas fundamentales necesarias para construir una asignación. En el área de asignación de MapForce los componentes aparecen en forma de rectángulo. Estos son algunos ejemplos de componente:

- Constantes
- Bases de datos
- Filtros
- Condiciones
- Archivos planos (archivos de texto, CSV o de longitud fija)
- Funciones
- Documentos EDI (UN/EDIFACT, ANSI X12, HL7)
- Archivos Excel 2007+
- Componentes de entrada simples
- Componentes de salida simples
- Variables
- Documentos XBRL
- Esquemas XML y documentos DTD

Componente de entrada

Un componente de entrada es un componente de MapForce que permite pasar valore simples a una asignación de datos. Los componentes de entrada se suelen utilizar para pasar nombres de archivo y otros valores de cadena a una asignación de datos en tiempo de ejecución. Los componentes de entrada no se deben confundir con los componentes de origen.

Componente de salida

Un componente de salida (o de salida simple) es un componente de MapForce que permite recuperar un valor de cadena de la asignación de datos. Los componentes de salida son uno de los muchos tipos de componente de destino, pero no son sinónimos.

Componente de origen

Un componente de salida es un componente donde MapForce lee datos. Al ejecutar la asignación, MapForce lee lo datos que aporta el conector del componente de origen, los convierte al tipo elegido y los envía al conector del componente de destino.

© 2018 Altova GmbH

Componente de destino

Un componente de destino es un componente donde MapForce escribe datos. Al ejecutarse la asignación, el componente de destino da a MapForce instrucciones para generar archivos de salida o generar el resultado como valor de cadena para poder procesarlo en una aplicación externa. El componente de destino es lo opuesto de un componente de origen.

Componente de combinación

Un componente de combinación es un componente de MapForce que permite unir varias estructuras en la asignación de acuerdo con las condiciones definidas por el usuario. Devuelve la asociación (conjunto combinado) de elementos que cumplen con la condición. Las combinaciones son de gran ayuda a la hora de combinar datos de dos estructuras que tienen un campo en común (p. ej. una identidad).

Conexión

Una conexión es la línea que se puede dibujar para unir dos conectores. Al dibujar la línea estamos dando a MapForce instrucciones de transformar lo datos de una forma determinada (p. ej. leer datos de un documento XML y escribirlos en otro documento XML).

Conector

Un conector es un pequeño triángulo situado a la izquierda o derecha de un componente. Los conectores situados a la izquierda del componente ofrecen puntos de entrada para el componente. Los situados a la derecha ofrecen puntos de salida desde el componente.

Contexto primario (parent-context)

parent-context es un argumento opcional de algunas funciones de agregado de la biblioteca de funciones **core** de MapForce (como las funciones `min`, `max`, `avg` y `count`). En un componente de origen que tiene varias secuencias jerárquicas, el contexto primario determina en qué conjunto de nodos debe operar la función.

20.3 F

FLF (*Fixed Length Field* o campo de longitud fija)

Frecuente formato de texto donde los datos se separan por campos de longitud fija (p. ej. los primeros cinco caracteres de cada fila representan un ID de transacción y los 20 caracteres siguientes representan una descripción de la transacción).

FlexText

FlexText es un módulo de MapForce Enterprise Edition que sirve para convertir datos de archivos de texto heredados y no estándar de gran complejidad en otros formatos compatibles con MapForce y viceversa.

© 2018 Altova GmbH

20.4 M

MapForce

MapForce es un entorno IDE multiuso basado en Windows que sirve para transformar datos de un formato en otro o pasar datos de un esquema a otro mediante operaciones visuales de arrastrar y colocar. Para ello ofrece una interfaz gráfica donde el usuario no necesita crear código de programación. De hecho, MapForce genera automáticamente el código de programa necesario para realizar la transformación de datos propiamente dicha (o la asignación de datos). Si prefiere no generar código de programa, puede simplemente ejecutar la transformación con el lenguaje de transformación integrado de MapForce (disponible en las ediciones Professional y Enterprise).

MFF

Extensión de archivo de los archivos de funciones de MapForce.

MFD

Extensión de archivo de los documentos de diseño de MapForce (asignaciones de datos).

MFP

Extensión de archivo de los archivos de proyecto de MapForce.

MFT

Extensión de archivo de los documentos plantilla de MapForce FlextText.

20.5 R

Recursos globales

Los recursos globales de Altova es una característica que permite hacer referencia a archivos, carpetas y bases de datos y poder reutilizar este tipo de recursos, configurarlos y ponerlos a disposición de otras aplicaciones de Altova.

© 2018 Altova GmbH

Índice

2

2005R3,
modo de compatibilidad, 1440
2018, 1479

A

A - Z,
componente de ordenación, 207
abs,
como función de MapForce (en lang | math functions), 1036
como función de MapForce (en xpath2 | numeric functions), 1055
acos,
como función de MapForce (en lang | math functions), 1036
Active, 1531
ActiveDocument, 1490, 1525
ActiveX,
integration at application level, 1577
integration at document level, 1580
integration prerequisites, 1573
ActiveX controls,
adding to the Visual Studio Toolbox, 1575
add,
como función de MapForce (en core | math functions), 986
ADO,
como interfaz de conexión de datos, 357
configurar una conexión, 364
ADO.NET,
configurar una conexión, 369
age,
como función de MapForce (en lang | datetime functions), 1021
Agregar,
biblioteca personal, 943
Análisis de impacto,
hoja de estilos SPS, 1231
Analizador,
integrado en los productos de Altova, 1735
Analizador XML,
información sobre, 1735
Analizador XML de Altova,
información sobre, 1735
Anidadas,
funciones definidas por el usuario, 903
ANSI X12,
personalizar la estructura del mensaje, 783
Ant,
crear un proyecto de Eclipse con, 1329
Any,
xs:any, 339
API,
documentation, 1460
overview, 1461
API de MapForce, 1460
overview, 1461
API de MapForce Type Library, 1488
Application, 1488
for Documents, 1525
Application object, 1463, 1489, 1490
AppOutputLine, 1497
AppOutputLines, 1502
AppOutputLineSymbol, 1503
Archivo,
como botón en componentes, 167
como botón en un componente, 104
como menú de aplicación, 1285
Archivo de biblioteca,
mff, 943
Archivo mff,
configurar, 943
Archivo plano,
asignación, 607
Archivo/Cadena,
como botón en componentes, 167
Archivo/cadena de texto,
como botón en un componente, 104
Archivo: (predeterminado),
como nombre de nodo raíz, 167
Archivo: <dinámico>,
como nombre de nodo raíz, 167
Archivos,
varios a partir de Excel, 177
Archivos CSV,
agregar o quitar campos, 616
como componente de destino, 616
como componente de origen, 616
configurar codificación, 616

Archivos CSV,
vista previa, 616
Archivos de ejemplo,
ubicación en disco, 32
Archivos de texto,
agregar o quitar campos, 627
asignar datos, 621
como componente de destino, 627
como componente de origen, 627
configurar codificación, 627
establecer carácter de relleno, 621
establecer el tamaño del campo fijo, 621
vista previa, 627
Archivos EDI,
como componentes de asignación, 695
Archivos JSON,
como componentes de origen y destino, 802
Archivos XML,
generar a partir de Excel, 177
generar a partir de registros de BD, 175
generar a partir de un solo origen XML, 174
AS2,
integración con los productos de Altova, 1184
integration with Altova products, 1184
ASC X12, 692, 706
Asignación,
basada en el destino, 152
basada en el origen (contenido mixto), 145
crear, 75
depuración, 294
documentar, 1226
estándar, 152
estándar con secundarios, 152
hojas de estilos SPS predefinidas para la documentación, 1231
secuencia de procesamiento, 282
validar, 81
Asignación basda en el destino, 145
Asignación de datos,
definición, 1743
formato de archivo plano, 607
Asignación de valores,
pasar datos sin modificarlos, 234
tabla de búsqueda, 230
tabla de búsqueda (propiedades), 237
Asignaciones,
procesamiento automatizado, 1170
tipos, 154
asin,
como función de MapForce (en lang | math functions), 1036
atan,
como función de MapForce (en lang | math functions), 1036
ATTLIST,
URI de espacio de nombres de DTD, 329
auto-format,
como función de MapForce (en edifact functions), 1019
Automático/a,
carga de bibliotecas, 943
Automatizado,
procesamiento, 1170
auto-number,
como función de MapForce (en core | generator functions), 980
avg,
función de MaForce (en core | aggregate functions), 962
Axis2,
configurar para servicios web SOAP, 1080, 1097
Ayuda,
como menú de aplicación, 1304

B

Background,
with gradient, 1546
Basada en el destino,
asignación, 152
Basada en el origen,
asignación de contenido mixto, 145
diferencias con la asignación estándar, 152
Base de datos,
asignar esquema XML a un campo, 553
consulta, 564
escribir archivos XML en, 556
generar valores secuenciales y únicos para, 460
generar varios archivos XML a partir de, 175
Base de datos Access,
actualización basada en condición IF, 621
Base package name,
for Java, 1522
Bases de datos,
actualizar datos de tabla, 475
actualizar e insertar datos de tabla, 478
agregar a la asignación, 433
combinar datos en, 478

© 2018 Altova GmbH

Bases de datos,
como recursos globales, 1203
compatibles, 356
ejecutar asignaciones en, 457
eliminar datos de tabla, 494
insertar datos en una tabla, 466
insertar datos en varias tablas vinculadas, 470
revertir transacciones, 503
usar inserción masiva, 504
base-uri,
como función de MapForce (en xpath2 | accessors library), 1048
BD,
ORDER BY, 537
Biblioteca, 1454
agregar funciones XQuery, 937
agregar personal, 943
carga automática, 943
Biblioteca personal,
agregar, 943
boolean,
como función de MapForce (en core | conversion functions), 967
Booleano,
resultado si bool = false, 903
Buscar,
archivos en la ventana Proyecto, 129
elementos dentro de componentes de asignación, 102

C

C#,
code generation, 1518
código, 1312
configuración, 1440
enumeration, 1568
error handling, 1464
generar código de asignación de datos en, 1322
generar servicios web en, 1083
integrar código generado, 1343
integration of MapForce, 1584
options, 1542, 1543
C++,
code generation, 1519
código, 1312
configuración, 1440
enumeration, 1567
error handling, 1464
generar código de asignación de datos en, 1317
integrar código generado, 1345
options, 1542, 1543
Cadena,
redistribuir datos de, 243
redistribuir datos desde, 243, 246
serializar datos a, 243, 251
Campo,
clave en archivos de texto, 612
Campos vacíos,
en archivos CSV, 616
en archivos de texto planos, 627
Canal RSS,
asignación de datos, 1135
capitalize,
como función de MapForce (en lang | string functions), 1040
Carácter de relleno,
en archivos de texto planos, 627
Carpeta,
disposición en carpetas (consulta de la BD), 571
CDATA, 337
ceiling,
como función de MapForce (en core | math functions), 986
Ceros iniciales,
conservar durante conversión, 738
Certificado,
firmas digitales, 352
Certificados digitales,
de confianza en Linux, 1147
de confianza en OS X / macOS, 1150
de confianza en Windows, 1151
en asignaciones de MapForce, 1145
exportar de Windows, 1155
gestión en Windows, 1155
transferir a Linux, 1161
transferir a OS X / macOS, 1162
char-from-code,
como función de MapForce (en core | string functions), 1009
Class ID,
in MapForce integration, 1591
Clave,
campos clave en archivos de texto, 612
clave de ordenación, 207
Clave de ordenación,

Clave de ordenación,
componente de ordenación, 207
Clave foránea,
en asignaciones de BD, 441, 451
Clave principal,
en asignaciones de BD, 441, 451
generar durante acción de inserción en BD, 466
generar para un componente de BD, 460
Close, 1517
project, 1549
code generation, 1527, 1529
C#, 1518
C++, 1519
enumerations, 1564, 1567, 1568, 1569
Java, 1519
options, 1523
options for, 1542, 1543, 1544, 1545
sample, 1477
XSLT, 1521
code-from-char,
como función de MapForce (en core | string functions), 1009
Code-generation,
options for, 1495, 1546
Codificación,
cambiar en archivos CSV, 616
cambiar en archivos de texto planos, 627
Código,
funciones inline y tamaño del código, 897
integrar el código de MapForce, 1340
SPL, 1442
tipos integrados, 1453
Código generado,
generar excepciones desde, 239
columnname-to-index,
as MapForce function, 1046
COM API,
documentation, 1460
Combinación,
de archivos XML, 343
Comentario de bloque, 568
Comentario de línea, 568
Comentarios,
agregar a archivos de destino, 336
Comillas,
en archivos CSV, 616
Comodines,
xs:any - xs:anyAtrribute, 339
Compatibilidad con QName, 332
Compatibilidad con Unicode,
de los productos de Altova, 1736
Compatibilidad del código,
v2005R3, 1440
Compilador,
configuración, 1440
Compleja,
entrada definida por el usuario, 909
función definida por el usuario, 908, 915
salida definida por el usuario, 916
Complejo/a,
funciones inline, 897
Complemento de MapForce para Eclipse,
acceso a menús y funciones frecuentes, 1265
ampliar características, 1278
cambiar a la perspectiva MapForce, 1262
configuración para la generación automática de código, 1275
crear asignaciones nuevas, 1270
crear un proyecto de MapForce/Eclipse, 1268
importar asignaciones a un proyecto de Eclipse, 1272
información, 1256, 1262, 1268
instalación, 1257
punto de extensión, 1278
trabajar con asignaciones y proyectos, 1268
Complemento de MapForce para Visual Studio,
acceso a menús y funciones frecuentes, 1251
habilitar, 1247
información, 1246
trabajar con asignaciones y proyectos, 1249
Complemento para Visual Studio,
ejecutar MapForce como, 1246
Component, 1504
Componente,
como menú de aplicación, 1293
definición, 1744
elementos eliminados, 124
ordenar datos, 207
Componente de destino,
definición, 1744
Componente de entrada,
definición, 1744
Componente de origen,
definición, 1744
Componente de salida,
definición, 1744
Componentes,
agregar a la asignación, 75

© 2018 Altova GmbH

Componentes,
alinear, 103
buscar, 102
cambiar configuración, 104
introducción, 101
secuencia de procesamiento, 282
Components, 1512
concat,
(como función) ejemplo de uso, 621
como función de MapForce (en core | string functions), 1009
Condiciones If-Else,
agregar a la asignación, 213
Conector,
definición, 1744
ver historial de valores procesados, 307
Conectores,
copia total, 154
Conexión,
como menú de aplicación, 1295
definición, 1744
Conexión de base de datos,
configurar, 357
ejemplo de configuración, 386
iniciar el asistente, 359
reutilizar desde recursos globales, 385
Conexiones,
conservar después de que cambie el elemento raíz, 119
mover a un componente distinto, 119
tipos, 154
Configuración,
C++ y C#, 1440
firma digital, 352
fuentes de texto (consulta de la BD), 582
vista de resultados (consulta de la BD), 582
Configuración de codificación,
en resultados XML, 324
Configuración de componentes Excel 2007+,
cambiar, 828
Configuración de la generación de código,
definir para todo el proyecto, 131
definir para una carpeta del proyecto, 133
Configurar,
archivo mff, 943
opciones del editor SQL, 580
Conmutador,
como función de FlexText, 667
Connection, 1513
Conservar datos,
cuando se use una asignación de valores, 234
Conservar datos sin modificarlos,
al pasar por una asignación de valores, 234
Consolidar datos,
combinación de archivos XML, 343
Consulta de BD,
convertir texto en comentario, 568
marcadores, 570
Consulta de la BD,
filtrar tablas, 573
generar SQL, 571, 576
opciones de fuentes de texto, 582
opciones de la vista de resultados, 582
regiones, 570
contains,
como función de MapForce (en core | string functions), 1010
Contexto (ventana),
información, 301, 309
Contexto primario,
definición, 1744
Contexto prioritario,
establecer en funciones, 285
Contraer,
regiones, 570
Contraseña,
firma digital, 352
Contrato de licencia para el usuario final, 1738, 1740
Controladores de BD,
resumen, 361
Controladores ODBC,
comprobar disponibilidad, 376
convert-to-utc,
como función de MapForce (en lang | datetime functions), 1021
Copia total,
conectores, 154
método de asignación, 145
resolver / eliminar conectores, 154
y filtros, 154
cos,
como función de MapForce (en lang | math functions), 1037
Count, 1502, 1512, 1526, 1540
como función de MapForce (en core | aggregate functions), 963
count-substring,
como función de MapForce (en lang | string functions), 1040

© 2018 Altova GmbH

Crear,
regiones, 570
create-guid,
como función de MapForce (en lang | generator functions), 1035
Criterio de ordenación,
cambiar, 207
CSV,
como origen de la asignación, 607
crear jerarquías (claves), 612
crear varias filas, 610
current,
como función de MapForce (en xslt | xslt functions library), 1060
current-date,
como función de MapForce (en xpath2 | context functions), 1050
current-dateTime,
como función de MapForce (en xpath2 | context functions), 1050
current-time,
como función de MapForce (en xpath2 | context functions), 1050

D

Datapoint, 1514
date-from-datetime,
como función de MapForce (en lang | datetime functions), 1022
datetime-add,
como función de MapForce (en lang | datetime functions), 1022
datetime-diff,
como función de MapForce (en lang | datetime functions), 1023
datetime-from-date-and-time,
como función de MapForce (en lang | datetime functions), 1024
datetime-from-parts,
como función de MapForce (en lang | datetime functions), 1025
datetime-to-xlsx,
as MapForce function, 1047
date-to-xlsx,
as MapForce function, 1046
Datos de tabla,
ordenar, 207
day-from-datetime,
como función de MapForce (en lang | datetime functions), 1026
day-from-duration,
como función de MapForce (en lang | datetime functions), 1027
Declaración XML,
suprimirla en resultado, 324
default-collation,
como función de MapForce (en xpath2 | context functions), 1051
Definida por el usuario,
entrada compleja, 909
salida compleja, 916
Definidas por el usuario,
funciones anidadas, 903
funciones complejas, 908, 915
resultado si bool = false, 903
Definido/a por el usuario,
función de búsqueda, 899
función estándar, 899
función inline / estándar, 897
Degradado,
fondo, 1440
degrees,
como función de MapForce (en lang | math functions), 1037
Delimitador,
cambiar en archivos CSV, 616
cambiar en archivos de texto planos, 627
Depuración,
con puntos de interrupción, 294
configuración, 319
detener, 299
información, 301
iniciar, 299
por pasos, 294
preparar, 298
restricciones, 294
Destino,
a partir de varios orígenes, 343
componente FlexText, 651
Disposición,
explorador, 571
distinct-values,
como función de MapForce (en core | sequence functions), 993
Distribución,
de productos de software de Altova, 1738, 1740
divide,

© 2018 Altova GmbH

divide,
como función de MapForce (en core | math functions), 986
divide-integer,
como función de MapForce (en lang | math functions), 1037
Dividir una vez,
como función de FlexText, 661
usando el modo "delimitado (basado en línea)", 664
usando el modo "delimitado (flotante)", 663
usando el modo "delimitado (la línea empieza con)", 665
usando el modo "longitud fija", 662
División repetida,
función de FlexText, 652
usando el modlo "delimitado (basado en línea)", 656
usando el modo "delimitado (flotante)", 654
usando el modo "delimitado (la línea empieza con)", 658
usando el modo "longitud fija", 654
Dll,
configuración del compilador, 1440
Document, 1515
closing, 1517
como función de MapForce (en xslt | xslt functions library), 1060
creating new, 1493, 1526
filename, 1523
on closing, 1516
on opening, 1489
opening, 1494, 1526
path and name of, 1518
path to, 1524
retrieving active document, 1525
save, 1524
save as, 1524
Documentación,
definir hojas de estilos SPS, 1234
Documentación general,
hoja de estilos SPS, 1231
Documentar,
asignaciones, 1226
Document-level,
examples of integration of XMLSpy, 1584
Documents, 1491, 1525
retrieving, 1526
total number in collection, 1526
DOM type,
enumerations for C++, 1567
for C++, 1542
DoTransform.bat,
ejecutar con RaptorXML Server, 1169
DSN de archivo,
configurar, 376
DSN de sistema,
configurar, 376
DSN de usuario,
configurar, 376
DTD,
origen y destino, 329
Duplicado de entrada,
agregar, 1293
Duplicar entradas, 46
duration-add,
como función de MapForce (en lang | datetime functions), 1027
duration-from-parts,
como función de MapForce (en lang | datetime functions), 1028
duration-subtract,
como función de MapForce (en lang | datetime functions), 1028

E

Eclipse,
generar código de asignación de datos para, 1326
EDI,
ASC X12, 706
IATA PADIS, 719
NCPDP SCRIPT, 722
personalizar validación, 790
separadores, 694
X12 HIPAA, 711
Edición,
como menú de aplicación, 1288
EDIFACT, 692
asignar a XML, 749
entradas combinadas, 768
terminología, 694
Edition, 1491
Editor SQL,
configuración general, 580
convertir texto en comentario, 568
crear regiones, 570
insertar comentarios, 568
insertar marcadores, 570
insertar regiones, 570
margen de marcadores, 570

© 2018 Altova GmbH

Editor SQL,
quitar comentarios, 568
quitar marcadores, 570
quitar regiones, 570
usar marcadores, 570
usar regiones, 570
Ejemplo,
asignación recursiva definida por el usuario, 920
element-available,
como función de MapForce (en xslt | xslt functions library), 1060
Elemento,
recursivo en XML Schema, 920
Elementos,
ausentes, 124
Elementos marcados,
ausentes, 124
Eliminar,
conexiones de copia total, 154
eliminaciones - elementos ausentes, 124
empty,
como función de MapForce (en lang | string functions), 1041
Encoding,
default for output files, 1544
Entrada,
parámetros opcionales, 903
valor predeterminado, 903
Entrada de la asignación,
especificar nombre de archivo personalizado como, 172
especificar varios archivos como, 167, 170, 172
Entradas combinadas, 768
Enumerations, 1564, 1565, 1566, 1567, 1568
for MapForce View, 1569
in MapForceControl, 1641
Envuelta,
firma digital, 352
equal,
como función de MapForce (en core | logical functions), 983
equal-or-greater,
como función de MapForce (en core | logical functions), 983
equal-or-less,
como función de MapForce (en core | logical functions), 983
Error handling,
general description, 1464
Errores de memoria,
soluciones, 79
ErrorMarkers, 1527, 1529
Espacios de nombres,
declarar personalizados, 345
y comodines (xs:any), 339
Esquema,
cambiar la referencia de ruta de acceso, 138
elementos recursivos, 920
generador de código, 1312
generar para un archivo XML, 323
Esquemas,
asignación de XML y esquemas, 323
Estándar,
asignación con secundarios, 152
asignación de contenido mixto, 152
diferencias con la asignación basada en el origen, 152
método de asignación, 145
Events,
of Document, 1516
Events for Project, 1547
Excel 2007+,
agregar como componente de asignación, 816
asignación de datos de canal RSS, 1135
asignar a XML, 831
asignar datos a, 246
asignar datos de BD a, 833
como componente de asignación, 817
como componente de origen o destino, 814
escribir datos de asignación en rangos de celdas, 821
generar varios archivos XML a partir de, 177
leer datos de asignación de rangos de celdas, 821
seleccionar rangos de celdas del componente de asignación, 822
Excepción,
memoria insuficiente, 1439
Excepciones,
agregar, 239
exists,
como función de MapForce (en core | sequence functions), 994
exp,
como función de MapForce (en lang | math functions), 1037
Expandir,
regiones, 570
Explorador,
aplicar filtros, 573
consulta de la BD, 571
filtrado, 573
Expresiones regulares,
como parámetro de la función "match-pattern", 958

© 2018 Altova GmbH

Expresiones regulares,
como parámetro de la función "tokenize-regexp", 958
dividir el componente de FlexText con, 685
en MapForce FlexText, 685
usar en condiciones de conmutador de FlexText, 687
Extensiones de Altova,
funciones para gráficos, 1652

F

false,
como función de MapForce (en xpath2 | boolean functions), 1049
Fija,
archivos de longitud fija (asignación de), 607
Filas,
asignar filas en archivos de texto, 612
de Excel, 177
Filtrar,
combinación de archivos XML, 343
conector de copa total, 154
datos de componentes, 213
de bases de datos, 540
explorador, 573
objetos de BD en la asignación, 438
objetos de la BD, 573
tablas de base de datos, 213
Filtros,
agregar a la asignación, 213
find-substring,
como función de MapForce (en lang | string functions), 1041
Firebird,
conectarse por JDBC, 389
conectarse por ODBC, 386
Firma,
configuración, 352
Firma digital,
activar, 348
configuración, 352
crear en resultado XML, 324
Firmar,
firma digital, 348
first-items,
como función de MapForce (en core | sequence functions), 995
FlexText,
cambiar configuración del componente, 649
como componente de destino, 651
contenedores, 634
definición, 1746
información general, 634
interfaz, 634
tutorial, 637
FLF,
definición, 1746
floor,
como función de MapForce (en core | math functions), 987
Fondo,
degradado, 1440
format-date,
como función de MapForce (en core | conversion functions), 967
format-dateTime,
como función de MapForce (en core | conversion functions), 968
format-guid-string,
como función de MapForce (en lang | string functions), 1041
format-number,
como función de MapForce (en core | conversion functions), 971
format-time,
como función de MapForce (en core | conversion functions), 973
FullName, 1518, 1550
Función,
agregar función XQuery personal, 937
como menú de aplicación, 1296
compleja (inline), 897
de búsqueda definida por el usuario, 899
definida por el usuario, 922
definida por el usuario anidada, 903
estándar definida por el usuario, 899
inline, 897
Función definida por el usuario,
recursiva, 922
Funciones de extensión .NET,
campos de instancia, 1730
campos estáticos, 1729
constructores, 1728
conversiones de tipos de datos, 1731, 1732
métodos de instancia, 1730
métodos estáticos, 1729
para XSLT y XQuery, 1725
resumen, 1725

© 2018 Altova GmbH

Funciones de extensión .NET,
tipos de datos .NET en XPath/XQuery, 1732
tipos de datos XPath/XQuery en .NET, 1731
Funciones de extensión .NET para XSLT y XQuery,
ver Funciones de extensión .NET, 1725
Funciones de extensión en scripts MSXSL,
msxsl:script, 1732
Funciones de extensión Java,
archivos de clases definidos por el usuario, 1717
archivos JAR definidos por el usuario, 1720
campos de instancia, 1723
campos estáticos, 1722
constructores, 1721
conversiones de tipos de datos, 1724, 1725
métodos de instancia, 1723
métodos estáticos, 1722
para XSLT y XQuery, 1716
resumen, 1716
tipos de datos Java en XPath/XQuery, 1725
tipos de datos XPath/XQuery en Java, 1724
Funciones de extensión Java para XSLT y XQuery,
ver Funciones de extensión Java, 1716
Funciones de extensión para XSLT y XQuery, 1715
Funciones de FlexText,
conmutador, 667
dividir una vez, 661
división repetida, 652
guardar como CSV (delimitado), 674
guardar como FLF (longitud fija), 681
guardar valor, 684
nodo, 671
omitir, 673
Funciones definidas por el usuario,
abrir, 888
crear, 888
eliminar, 888
importar, 888
influir en el orden de los parámetros, 888
reutilizar, 888
Funciones para gráficos,
ejemplo, 1712
estructura de datos de gráficos, 1706
fragmento de código, 1702
Funciones utilizadas por, 1231
function-available,
como función de MapForce (en xslt | xslt functions library), 1061
Functions,
adding as mapping components, 883
adding parameters to, 883
deleting parameters from, 883
finding in the Libraries window, 883
finding occurences in active mapping, 883
viewing the argument data type of, 883
viewing the description of, 883

G

Generador de código, 1312
Generar,
código a partir de un esquema, 1312
código y funciones inline, 897
firma digital, 348
generate-id,
como función de MapForce (en xslt | xslt functions library), 1061
generate-sequence,
como función de MapForce (en core | sequence functions), 995
get-fileext,
como función de MapForce (en core | file path functions), 978
get-folder,
como función de MapForce (en core | file path functions), 978
GetRootDatapoint, 1565
Gradients,
in background, 1546
greater,
como función de MapForce (en core | logical functions), 983
group-adjacent,
como función de MapForce (en core | sequence functions), 995
group-by,
como función de MapForce (en core | sequence functions), 997
group-ending-with,
como función de MapForce (en core | sequence functions), 999
group-into-blocks,
como función de MapForce (en core | sequence functions), 1000
group-starting-with,
como función de MapForce (en core | sequence functions), 1000
Guardar como CSV (delimitado),

© 2018 Altova GmbH

Guardar como CSV (delimitado),
como función de FlexText, 674
Guardar como FLF (longitud fija),
como función de FlexText, 681
Guardar valor,
como función de FlexText, 684

H

Health Level 7,
ejemplo, 878
Herramientas,
como menú de aplicación, 1301
HighlightMyConnections, 1532
HighlightMyConnectionsRecursively, 1532
HIPAA,
generar la confirmación de implementación X12 999, 710
X12, 711
HIPAA X12, 692
HL7, 692
HL7 2.6 a 3.x,
ejemplo, 878
Hoja de cálculo,
convertir en archivos de salida XML, 177
Hojas de cálculo Excel 2007+,
escribir datos en, 819
leer datos de, 819
Hojas de estilos,
definir para la documentación, 1234
hour-from-datetime,
como función de MapForce (en lang | datetime functions), 1029
hour-from-duration,
como función de MapForce (en lang | datetime functions), 1030
HRESULT,
and error handling, 1464
HTML,
documentación de la asignación, 1226
integration of MapForce, 1593
vista previa del resultado de la asignación como, 1221
HTML example,
of MapForceControl integration, 1591, 1592, 1593
HTTPS,
llamar a servicios web por, 1145

I

IATA PADIS, 692, 719
IBM DB2,
conectarse por ODBC, 390
escribir en campos de tipo XML, 553
leer campos de tipo XML, 553
IBM DB2 for i,
conectarse por ODBC, 395
IBM Informix,
conectarse por JDBC, 398
Iconos,
en la ventana Mensajes de la consulta de BD, 576
en la ventana Resultados de la consulta de BD, 576
IIS,
configurar para servicios web, 1083
implicit-timezone,
como función de MapForce (en xpath2 | context functions), 1051
index-to-columnname,
as MapForce function, 1047
Información general, 1735
Información legal, 1738
Información sobre derechos de autor, 1738
Información técnica, 1735
Inline,
funciones y tamaño del código, 897
Inline / estándar,
funciones definidas por el usuario, 897
Inline XBRL,
asignar datos de, 841
Insertar,
comentario de bloque, 568
comentario de línea, 568
comentarios, 568
como menú de aplicación, 1289
componente WHERE de SQL, 537
marcadores, 570
regiones, 570
InsertXMLFile, 1533
InsertXMLSchema, 1533
InsertXMLSchemaWithSample, 1534
Instancia,
cambiar la referencia de ruta de acceso, 138
Instrucciones de procesamiento,

© 2018 Altova GmbH

Instrucciones de procesamiento,
agregar a archivos de destino, 336
Instrucciones de procesamiento y comentarios,
asignación, 146
Integración,
en C#, 1343
en C++, 1345
en Java, 1341
Integrar código de MapForce, 1340
Integrating,
MapForce in applications, 1572
Intercalación,
componente de ordenación, 207
intercalación local, 207
punto de código unicode, 207
Intercalación local, 207
Introducción,
generador de código, 1313
is-not-null,
como función de MapForce (en db functions), 1017
is-null,
como función de MapForce (en db functions), 1017
is-xsi-nil,
como función de MapForce (en core | node functions), 989
Item, 1502, 1512, 1526, 1540
item-at,
como función de MapForce (en core | sequence functions), 1001
items-from-till,
como función de MapForce (en core | sequence functions), 1001

J

Java, 1598
code generation, 1519
código, 1312
configuración, 1440
generar código de asignación de datos en, 1326
generar servicios web en, 1097
integrar código generado, 1341
options, 1522, 1542, 1545
JavaScript,
error handling, 1464
JDBC,
como interfaz de conexión de datos, 357
conectarse a Teradata, 428
configurar una conexión (Linux), 432
configurar una conexión (macOS), 432
configurar una conexión (Windows), 379
configurar una conexión Oracle en OS X Yosemite, 432
control de referencias en el código generado, 1329
Jerarquía,
a partir de archivos de texto, 612
JScript,
code-generation sample, 1477
JSON,
asignación de datos de servicios web, 1131
asignación de propiedades adicionales, 808
ejemplo de asignación de datos, 808

L

last,
como función de MapForce (en xpath2 | context functions), 1051
last-items,
como función de MapForce (en core | sequence functions), 1002
leapyear,
como función de MapForce (en lang | datetime functions), 1030
left,
como función de MapForce (en lang | string functions), 1041
left-trim,
como función de MapForce (en lang | string functions), 1042
Lenguaje de transformación,
seleccionar, 80
less,
como función de MapForce (en core | logical functions), 984
Lib,
configuración del compilador, 1440
Libraries window,
finding functions in, 883
Library type,
enumerations for C++, 1567
for C++, 1542, 1543
Licencia, 1740
información sobre, 1738
Licencia del producto de software, 1740
Linux,

© 2018 Altova GmbH

Linux,
bases de datos compatibles, 430
certificados de servidor de confianza, 1147
configurar conexiones de BD en, 430
ejecutar asignaciones con llamadas a servicios web por HTTPS, 1161
implementar archivos de ejecución de servidor en, 430
transferir certificados clientes a, 1161
Llamar a gráficos,
hoja de estilos SPS, 1231
local-name-from-QName,
como función de MapForce (en core | QName functions), 991
log,
como función de MapForce (en lang | math functions), 1037
log10,
como función de MapForce (en lang | math functions), 1038
logical-and,
como función de MapForce (en core | logical functions), 984
logical-not,
como función de MapForce (en core | logical functions), 984
logical-or,
como función de MapForce (en core | logical functions), 985
logical-xor,
como función de MapForce (en lang | logical functions), 1035
Logo,
display on startup, 1545
option for printing, 1545
Lotes,
automatización de procesamiento, 1170
lowercase,
como función de MapForce (en lang | string functions), 1042

M

macOS,
bases de datos compatibles, 430
configurar conexiones de BD en, 430
implementar archivos de ejecución de servidor en, 430
main-mfd-filepath,
como función de MapForce (en core | file path functions), 978
MapForce,
API, 1460
conceptos básicos, 21
integration, 1572
introducción, 14
parent, 1534
MapForce integration,
example of, 1591, 1592, 1593
MapForce Server,
automatizar asignaciones, 1170
compilar asignaciones para, 1176
generar excepciones desde, 239
MapForce view,
enumerations for, 1569
MapForceCommand,
in MapForceControl, 1621
MapForceCommands,
in MapForceControl, 1623
MapForceControl, 1624
documentation of, 1572
example of integration at application level, 1591, 1592, 1593
examples of integration at document level, 1584
integration using C#, 1584
integration using HTML, 1593
integration using Visual Basic, 1608
object reference, 1621
MapForceControlDocument, 1632
MapForceControlPlaceHolder, 1638
MapForceView, 1522, 1530
application, 1532
Mapping,
inserting XML file, 1533
inserting XML Schema file, 1533
Marcadores,
insertar, 570
margen de marcadores, 570
navegar, 570
quitar, 570
MariaDB,
conectarse por ODBC, 399
match-pattern,
como función de MapForce (en lang | string functions), 1042
max,
como función de MapForce (en core | aggregate functions), 963
como función de MapForce (en lang | math functions), 1038
max-string,
como función de MapForce (en core | aggregate functions), 964
Medición de licencias,

© 2018 Altova GmbH

Medición de licencias,
en los productos de Altova, 1739
Memoria,
insuficiente - excepciones, 1439
Memoria insuficiente, 1439
Mensaje SOAP,
agregar un UsernameToken, 1128
configurar TTL, 1128
Mensajes,
iconos de la consulta de BD, 576
ventana de la consulta de BD, 576
MERGE,
instrucción en código SQL generado con MapForce, 478, 485
Método,
reservar nombre, 1439
Métodos de asignación,
basado en el destino, 145
estándar, 145
estándar / mixto / copia total, 145
MFC support,
fo C++, 1543
mfd,
extensión de archivo, 1747
mfd-filepath,
como función de MapForce (en core | file path functions), 978
mff,
archivo de biblioteca, 943
archivo mff.xsd, 943
extensión de archivo, 1747
mfp,
extensión de archivo, 1747
mft,
extensión de archivo, 1747
Microsoft Access,
conectarse por ADO, 364, 401
Microsoft Excel,
véase "Excel 2007+", 814
Microsoft SharePoint Server,
agregar archivos como componentes desde, 76
Microsoft SQL Server,
conectarse por ADO, 404
conectarse por ODBC, 406
millisecond-from-datetime,
como función de MapForce (en lang | datetime functions), 1030
millisecond-from-duration,
como función de MapForce (en lang | datetime functions), 1031
min,
como función de MapForce (en core | aggregate functions), 964
como función de MapForce (en lang | math functions), 1038
min-string,
como función de MapForce (en core | aggregate functions), 965
minute-from-datetime,
como función de MapForce (en lang | datetime functions), 1031
minute-from-duration,
como función de MapForce (en lang | datetime functions), 1031
Mixto,
asignación basada en el origen, 145
asignación de contenido, 145
asignación estándar, 152
ejemplo de asignación de contenido, 151
método de asignación de contenido, 145
Modo,
compatible con v2005R3, 1440
Modo de compatibilidad,
v2005R3, 1440
modulus,
como función de MapForce (en core | math functions), 987
month-from-datetime,
como función de MapForce (en lang | datetime functions), 1031
month-from-duration,
como función de MapForce (en lang | datetime functions), 1032
Motor integrado,
definición, 80
uso, 80
Motores,
de los productos de Altova, 1736
multiply,
como función de MapForce (en core | math functions), 987
MySQL,
conectarse por ODBC, 409

N

Name, 1493, 1523, 1553
namespace-uri-form-QName,

© 2018 Altova GmbH

namespace-uri-form-QName,
como función de MapForce (en core | QName functions), 993
Navegar,
marcadores, 570
NCPDP SCRIPT,
componente de asignación, 722
negative,
como función de MapForce (en lang | logical functions), 1035
NewDocument, 1493, 1526
NewProject, 1493
Nillable,
como atributo en esquemas XML, 332
no compatible, 1439
node-name,
como función de MapForce (en core | node functions), 989
como función de MapForce (en xpath2 | accessors library), 1048
node-name (función),
alternativas de uso, 256
Nodo,
como función de FlexText, 671
Nombre de la aplicación, 1440
Nombres de archivo,
especificar como parámetros de entrada, 172
Nombres de nodo,
asignar, 256
normalize-space,
como función de MapForce (en core | string functions), 1010
not-equal,
como función de MapForce (en core | logical functions), 985
not-exists,
como función de MapForce (en core | sequence functions), 1002
now,
como función de MapForce (en lang | datetime functions), 1032
NULL,
gestionar en asignaciones de BD, 507
number,
como función de MapForce (en core | conversion functions), 973
numeric,
como función de MapForce (en lang | logical functions), 1035

O

Object,
reference, 1488
Object model,
overview, 1463
Object tree navigation,
Application, 1490, 1495
AppOutputLine, 1497, 1501
AppOutputLines, 1502, 1503
AppOutputLineSymbol, 1503, 1504
Component, 1505, 1510
Components, 1512, 1513
Document, 1517, 1523
Documents, 1525, 1527
ErrorMarker, 1527, 1529
ErrorMarkers, 1529, 1530
MapForceView, 1532, 1534
Mapping, 1536, 1539
Mappings, 1540, 1541
Options, 1541, 1545
Project, 1549, 1554
ProjectItem, 1557, 1562
Objetos de BD,
agregar a la asignación, 438
filtrar, 438
quitar de la asignación, 438
Objetos globales,
de SPL, 1445
Obsolete, 1523
ODBC,
como interfaz de conexión de datos, 357
conectarse a MariaDB, 399
conectarse a Teradata, 423
configurar una conexión, 376
OLE DB,
como interfaz de conexión de datos, 357
Omitir,
como función de FlexText, 673
OnDocumentClosed, 1516
OnDocumentOpened, 1489
OnProjectClosed, 1547
OnProjectOpened, 1490
OOXML,
como formato Excel 2007 predeterminado, 814

Opcional,
parámetros de entrada, 903
Opciones,
fuentes de texto (consulta de la BD), 582
vista de resultados (consulta de la BD), 582
OpenDocument, 1494, 1526
OpenProject, 1494
Options, 1495, 1541
for code generation, 1523, 1544
for Java, 1522
Oracle,
conectarse por JDBC, 415
conectarse por ODBC, 410
escribir en campos de tipo XML, 553
leer campos de tipo XML, 553
Orden,
de procesamiento de los componentes, 282
Ordenar,
bases de datos, 540
columnas en la ventana de resultados, 576
componente de ordenación, 207
datos en la ventana de resultados, 576
tablas de la consulta de la BD, 571
Ordenar datos,
componente de ordenación, 207
ORDER BY,
componente where de SQL, 537
OS X / macOS,
certificados de servidor de confianza, 1150
ejecutar asignaciones con llamadas a servicios web por HTTPS, 1162
transferir certificados clientes a, 1162
Output directory,
for code-generation files, 1542
for XSLT generated output, 1546
Output encoding,
default used, 1544
Overview,
of API de MapForce, 1461

P

PADIS,
IATA, 719
pad-string-left,
como función de MapForce (en lang | string functions), 1042
pad-string-right,
como función de MapForce (en lang | string functions), 1043
Paquete base, 1440
Paquetes de taxonomías XBRL,
agregar, 843
Parámetro,
opcional, 903
salida, 903
Parent, 1495, 1501, 1503, 1504, 1510, 1513, 1527, 1529, 1530, 1539, 1541, 1554
parse-date,
como función de MapForce (en core | conversion functions), 974
parse-dateTime,
como función de MapForce (en core | conversion functions), 974
parse-number,
como función de MapForce (en core | conversion functions), 976
parse-time,
como función de MapForce (en core | conversion functions), 977
Pasar datos,
sin modificarlos a través de una asignación de valores, 234
Path, 1524, 1554
PDF,
documentación de la asignación, 1226
vista previa del resultado de la asignación como, 1221
Período de evaluación,
de los productos de software de Altova, 1738, 1740
Personales,
funciones XQuery, 937
pi,
como función de MapForce (en lang | math functions), 1038
Plataformas,
para productos de Altova, 1735
Posición del depurador,
ver valor actual de, 307
position,
como función de MapForce (en core | sequence functions), 1003
positive,
como función de MapForce (en lang | logical functions), 1035
PostgreSQL,
conectarse directamente (de forma nativa), 383
conectarse por ODBC, 417

© 2018 Altova GmbH

pow,
como función de MapForce (en lang | math functions), 1038
Predeterminado,
valor de entrada, 903
Procesador XQuery,
de los productos de Altova, 1736
Procesadores XSLT,
de los productos de Altova, 1736
Procesamiento,
automatizar asignaciones, 1170
Programming language,
enumerations for, 1567
Progress OpenEdge (base de datos),
conectarse por JDBC, 421
conectarse por ODBC, 419
Project, 1546
creating new, 1493
file name, 1553
file name and path, 1550
on opening, 1490
opening, 1494
path with filename, 1554
saving, 1554, 1555
Project type,
enumerations for C#, 1568
for C#, 1543
for Java, 1545
Propiedades,
tabla de asignación de valores, 237
Protocolos,
compatibles con WSDL, 1067
Proyecto,
como menú de aplicación, 1291
Proyecto de servicio web,
crear, 1070
Proyectos,
abrir, 129
buscar, 129
cerrar, 129
crear, 130
Punto de código,
intercalación, 207
Puntos de interrupción,
agregar, 304
eliminar, 304
información, 301
Puntos de interrupción (ventana),
información, 301, 311

Q

QName,
como función de MapForce (en core | QName functions), 991
QName-as-string,
como función de MapForce (en lang | QName functions), 1040
Quit, 1495
Quitar,
comentario de bloque, 568
comentario de línea, 568
comentarios, 568
conexiones de copia total, 154
marcadores, 570
regiones, 570

R

radians,
como función de MapForce (en lang | math functions), 1039
random,
como función de MapForce (en lang | math functions), 1039
RaptorXML Server,
ejecutar una transformación, 1169
Recursivo/a,
asignación definida por el usuario, 920
función definida por el usuario, 922
llamada en funciones, 897
Recursos globales,
crear, 1202
ejemplos de uso, 1209, 1210, 1212, 1216
en distintos entornos de ejecución, 1207
introducción, 1201
Referencia, 1284
Regiones,
contraer, 570
crear, 570
expandir, 570
insertar, 570
quitar, 570
Relaciones de BD,
conservar en asignaciones, 441
definir en asignaciones, 451

© 2018 Altova GmbH

remove-fileext,
como función de MapForce (en core | file path functions), 979
remove-folder,
como función de MapForce (en core | file path functions), 979
remove-timezone,
como función de MapForce (en lang | datetime functions), 1032
repeat-string,
como función de MapForce (en lang | string functions), 1043
replace,
como función de MapForce (en lang | string functions), 1043
replace-fileext,
como función de MapForce (en core | file path functions), 979
replicate-item,
como función de MapForce (en core | sequence functions), 1006
replicate-sequence,
como función de MapForce (en core | sequence functions), 1008
Requisitos de memoria, 1735
Reservar,
nombre de método, 1439
resolve-filepath,
como función de MapForce (en core | file path functions), 979
resolve-uri,
como función de MapForce (en xpath2 | anyURI functions), 1048
Resultado de la asignación,
generar varios archivos como, 167, 172
Resultado XML,
cambiar el esquema, 324
cambiar el nombre del archivo de instancia, 324
cambiar la configuración de codificación, 324
crear firma digital, 324
Resultados,
como menú de aplicación, 1297
guardar, 84
iconos de la consulta de BD, 576
validar, 83
ventana de la consulta de BD, 576
vista previa, 84
Retener datos,
al pasar por una asignación de valores, 234
reversefind-substring,
como función de MapForce (en lang | string functions), 1043
right,
como función de MapForce (en lang | string functions), 1044
right-trim,
como función de MapForce (en lang | string functions), 1044
round,
como función de MapForce (en core | math functions), 988
round-half-to-even,
como función de MapForce (en xpath2 | numeric functions), 1055
round-precision,
como función de MapForce (en core | math functions), 988
RTF,
documentación de la asignación, 1226
vista previa del resultado de la asignación como, 1221
Rutas de acceso de archivos,
corregir referencias rotas, 142
de bases de datos basadas en archivos, 140
en el código generado, 143
relativas o absolutas, 143
relativas y absolutas, 138
Rutas de acceso en el código generado,
convertir en rutas absolutas, 97

S

Salida,
definida por el usuario si bool = false, 903
parámetro, 903
SAP IDoc,
agregar componentes, 725
asigar a XML, 725
asignar datos, 725
SAP IDocs, 692
Save, 1524, 1554
SaveAs, 1524
Saved, 1524, 1555
schemanativetype, 1443
Scripts en XSLT/XQuery,
ver Funciones de extensión, 1715
Search,
functions in the Libraries window, 883
Sección,
CDATA, 337

© 2018 Altova GmbH

second-from-datetime,
como función de MapForce (en lang | datetime functions), 1032
second-from-duration,
como función de MapForce (en lang | datetime functions), 1033
Secuencia,
de procesamiento de los componentes, 282
Secuencia de procesamiento,
de los componentes, 282
Seleccionar,
datos de tabla (consulta de BD), 576
Separada,
firma digital, 352
Separadores,
EDI, 694
Servicios web,
agregar llamadas de tipo REST, 1131
agregar un UsernameToken, 1128
asignación de datos de canal RSS, 1135
asignación de datos JSON, 1131
como componentes de MapForce, 1106
configurar el plazo de tiempo de espera, 1122
credenciales de autenticación HTTP básicas, 1126
ejemplo (de tipo REST), 1131
ejemplo (de tipo WSDL), 1140
firmar con certificados digitales HTTPS, 1126
habilitar autenticación preventiva, 1126
habilitar WS-Security, 1128
implementar, 1064
llamar desde MapForce, 1106
llamar por HTTPS, 1145
requisitos, 1064
Servicios web (REST),
agregar en la asignación, 1108
Servicios web (SOAP),
generar con MapForce, 1083, 1097
Servicios web (WSDL),
agregar en la asignación, 1121
set-empty,
como función de MapForce (en core | sequence functions), 1008
set-null,
como función de MapForce (en db functions), 1017
set-xsi-nil,
como función de MapForce (en core | node functions), 990
ShowItemTypes, 1535
ShowLibraryFunctionHeader, 1535
shutdown,
of application, 1495, 1496
Signo de interrogación,
elementos ausentes, 124
sin,
como función de MapForce (en lang | math functions), 1039
skip-first-items,
como función de MapForce (en core | sequence functions), 1008
SO,
para productos de Altova, 1735
SOAP,
definir un error, 1078
información sobre Java, 1080
SPL, 1442
bloques de código, 1442
condiciones, 1448
foreach, 1449
objetos globales, 1445
subrutinas, 1451
usar archivos, 1447
variables, 1444
SPS,
hojas de estilos definidas por el usuario, 1234
hojas de estilos predefinidas para documentar asignaciones, 1231
SQL,
cargar desde scripts, 566
ejecutar instrucciones, 566, 568
escribir instrucciones, 567
exportar instrucciones como scripts SQL, 568
generar instrucciones, 566, 567
importar scripts SQL, 568
SQL Server,
conectarse por ADO, 364
conectarse por ADO.NET, 369
escribir en campos de tipo XML, 553
leer campos de tipo XML, 553
SQL WHERE,
componente (insertar), 537
ORDER BY, 537
SQL WHERE/ORDER,
componente de MapForce, 540
SQLite,
asignar datos a, 251, 514
asignar datos desde, 246
cambiar la ruta de acceso de la BD por una ruta de acceso absoluta en el código generado, 143

© 2018 Altova GmbH

SQLite,
configurar una conexión (Linux), 432
configurar una conexión (macOS), 432
configurar una conexión (Windows), 384
escribir archivos XML en, 556
usar una ruta de acceso absoluta o relativa, 140
sqrt,
como función de MapForce (en lang | math functions), 1039
starts-with,
como función de MapForce (en core | string functions), 1010
startup,
of application, 1496
static-node-annotation,
como función de MapForce (en core | node functions), 990
static-node-name,
como función de MapForce (en core | node functions), 990
Status, 1496, 1564
string,
como función de MapForce (en core | conversion functions), 977
como función de MapForce (en xpath2 | accessors library), 1048
string-as-QName,
como función de MapForce (en lang | QName functions), 1040
string-compare,
como función de MapForce (en lang | string functions), 1044
string-compare-ignore-case,
como función de MapForce (en lang | string functions), 1044
string-join,
como función de MapForce (en core | aggregate functions), 965
string-length,
como función de MapForce (en core | string functions), 1010
Stylevision,
definir hojas de estilos SPS para asignaciones, 1234
substitute-missing,
como función de MapForce (en core | sequence functions), 1008
substitute-missing-with-xsi-nil,
como función de MapForce (en core | node functions), 991
substitute-null,
como función de MapForce (en db functions), 1018
substring,
como función de MapForce (en core | string functions), 1010
substring-after,
como función de MapForce (en core | string functions), 1011
substring-before,
como función de MapForce (en core | string functions), 1011
subtract,
como función de MapForce (en core | math functions), 988
sum,
como función de MapForce (en core | aggregate functions), 966
Superposición de datos,
información, 301
Sybase,
conectarse por JDBC, 422
system-property,
como función de MapForce (en xslt | xslt functions library), 1061

T

Tabla,
tabla de búsqueda (asignación de valores), 230
Tabla de búsqueda,
propiedades, 237
tabla de asignación de valores, 230
tan,
como función de MapForce (en lang | math functions), 1039
Teradata,
conectarse por JDBC, 428
conectarse por ODBC, 423
Terminología,
EDIFACT, 694
Texto,
archivos de texto (definir campos clave), 612
asignación de datos de archivos de texto, 607
time-from-datetime,
como función de MapForce (en lang | datetime functions), 1033
time-to-xlsx,
as MapForce function, 1047
timezone,
como función de MapForce (en lang | datetime functions), 1033
Tipo complejo,
ordenar, 207
Tipo simple,

© 2018 Altova GmbH

Tipo simple,
ordenar, 207
Tipos,
conexiones, 154
derivados (xsi:type), 329
integrados, 1453
Tipos derivados,
asignaciones, 329
to-date,
como función de MapForce (en edifact functions), 1019
to-datetime,
como función de MapForce (en edifact functions), 1019
to-duration,
como función de MapForce (en edifact functions), 1020
tokenize,
como función de MapForce (en core | string functions), 1011
tokenize-by-length,
como función de MapForce (en core | string functions), 1013
tokenize-regexp,
como función de MapForce (en core | string functions), 1015
Tomcat,
configurar para servicios web SOAP, 1080, 1097
to-time,
como función de MapForce (en edifact functions), 1020
TRADACOMS, 692
archivos de configuración, 739
como componente de asignación, 728, 734
convertir en XML, 740
reglas de validación, 735
Transformaciones,
RaptorXML Server, 1169
Transformar,
datos de entrada (asignación de valores), 230
translate (in core | string functions),
as MapForce function, 1016
Transmisión por secuencias de datos,
definición, 79
true,
como función de MapForce (en xpath2 | boolean functions), 1049

U

UN/EDIFACT, 692
asignar a XML, 749
personalizar la estructura del mensaje, 776
terminología, 694
unary-minus,
como función de MapForce (en lang | math functions), 1039
Unicode,
intercalación de punto de código, 207
reemplazar caracteres especiales, 458
unparsed-entity-uri,
como función de MapForce (en xslt | xslt functions library), 1062
uppercase,
como función de MapForce (en lang | string functions), 1045
URI,
de documentos DTD, 329
y nombres QName, 332
URI de espacio de nombres,
DTD, 329
y nombres QName, 332
URL,
agregar archivos como componentes desde, 76
Uso de Internet,
en los productos de Altova, 1736

V

Validador,
de los productos de Altova, 1735
Validar,
diseño de asignación, 81
resultados de la asignación, 83
Valor,
predeterminado, 903
Valores (ventana),
información, 301, 307
pestaña Contexto, 307
pestaña Historial, 307
pestaña Relacionados, 307
Valores null,
gestionar en componentes de BD, 459
Variables,
agregar a la asignación, 197
cambiar el ámbito de, 200
de SPL, 1444
ejemplos de uso, 202, 203, 205

© 2018 Altova GmbH

Variables,
introducción, 195
Varios archivos XML,
a partir de un Excel, 177
Varios elementos consecutivos,
Edifact - X12, 768
Varios orígenes,
en un solo destino, 343
Version, 1492, 1493
View,
of MapForce, 1530
Visible, 1496
Vista,
como menú de aplicación, 1299
Vista Explorador,
generar instrucciones SQL, 574
opciones del menú contextual, 574
Visual Basic,
error handling, 1464
integration of MapForce, 1608
Visual Studio,
adding the MapForce ActiveX Controls to the toolbox, 1575
generar código de asignación de datos para, 1317, 1322

W

WADL, 1122
WebDAV Server,
agregar archivos como componentes desde, 76
weekday,
como función de MapForce (en lang | datetime functions), 1033
weeknumber,
como función de MapForce (en lang | datetime functions), 1034
WHERE,
componente WHERE de SQL, 537
WindowHandle, 1496
Windows,
certificados de servidor de confianza, 1151
compatibilidad con productos de Altova, 1735
ejecutar asignaciones con llamadas a servicios web por HTTPS, 1164
implementar archivos de ejecución de servidor en, 430
Word,
documentación de la asignación, 1226
Word 2007+,
vista previa del resultado de la asignación como, 1221
WSDL,
compatibilidad con versión 2.0, 1067
llamar a un servicio web desde una asignación, 1140
protocolos compatibles, 1067

X

X12,
entradas combinadas, 768
generar la confirmación de implementación X12 999, 710
generar la confirmación funcional X12 997, 708
HIPAA, 711
xbrl-measure-currency,
as MapForce function, 1045
xbrl-measure-pure,
as MapForce function, 1046
xbrl-measure-shares,
as MapForce function, 1046
Xerces,
compatibilidad, 1440
xlsx-to-date,
as MapForce function, 1047
xlsx-to-datetime,
as MapForce function, 1047
xlsx-to-time,
as MapForce function, 1048
XML,
asignar a UN/EDIFACT, 749
asignar datos CSV a XML, 607
asignar datos Excel 2007+ a, 831
como destino de la asignación, 607
configuración de la firma, 352
convertir TRADACOMS, 740
escribir en campo de BD, 556
XML a XML, 323
XML canónica,
firma digital, 352
XML data,
escribir en campos de tipo XML, 553
leer campos de tipo XML, 553
XMLSpy command table, 1611
XQuery,
agregar funciones personales, 937
funciones de extensión, 1715

© 2018 Altova GmbH

XQuery,
vista previa del código generado, 95
xs: any (xs:anyAttribute), 339
xsi:nil,
como atributo en instancias XML, 332
xsi:type,
asignación de tipos derivados, 329
XSLT,
agregar funciones personales, 931
code generation, 1521
espacio de nombres de plantilla, 931
funciones de extensión, 1715
options, 1546
quitar funciones personales, 931
vista previa del código generado, 94

Y

year-from-datetime,
como función de MapForce (en lang | datetime functions), 1034
year-from-duration,
como función de MapForce (en lang | datetime functions), 1034

Z

Z to A,
componente de ordenación, 207

© 2018 Altova GmbH