Manual del usuario y referencia del programador

Manual del usuario y referencia de Altova MapForce 2014

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: 2014

© 2014 Altova GmbH

Tabla de contenido

1 MapForce 2014 3

2 Características nuevas 6

3 Introducción a MapForce 12

3.1 Terminología ... 14

4 MapForce Server: automatizar asignaciones 22

5 RaptorXML Server 24

6 Tutorial de MapForce 26

6.1 Configuración del entorno de asignación ... 28

6.1.1 Agregar componentes al panel Asignación ... 29

6.2 Crear una asignación ... 33

6.2.1 Asignaciones entre elementos de un esquema ... 34

6.2.2 Uso de las funciones en las asignaciones de datos ... 37

6.2.3 Filtrado de datos ... 41

6.3 Generación de código XSLT 1.0 y 2.0 ... 45

6.4 Trabajar con varios esquemas y documentos de destino ... 46

6.4.1 Crear un segundo componente de destino ... 47

6.4.2 Ver y generar el resultado de varios esquemas de destino ... 50

6.5 Asignar varios componentes de origen a uno de destino ... 52

6.5.1 Crear la asignación ... 54

6.5.2 Duplicar elementos de entrada ... 57

6.6 Trabajar con varios archivos de entrada/salida ... 62

6.6.1 Procesar varios archivos por cada componente de entrada/salida ... 63

6.7 Asignación de datos entre una BD y un esquema ... 67

6.7.1 Conectarse a una base de datos Access ... 70

6.7.2 Asignar los datos de la base de datos ... 72

7 Interfaz del usuario 76

7.1 Ventana Bibliotecas ... 80
7.2 Ventana Proyecto ... 82
7.3 Panel Asignación ... 86
7.4 Panel XSLT / XSLT2 / XQuery ... 88
7.5 Panel Consulta de la BD ... 89
7.5.1 Ventana Explorador ... 90
7.5.2 Editor SQL ... 91
7.5.3 Pestaña Resultados ... 92
7.5.4 Pestaña Mensajes ... 93
7.6 Panel Resultados / motor de ejecución BUILTIN ... 94
7.7 Paneles relacionados con StyleVision ... 97
7.8 Ventana Vista general ... 98
7.9 Ventana Mensajes ... 99

8 Funcionamiento general 102

8.1 Mover y conservar conectores ... 103
8.2 Elementos desaparecidos ... 108
8.3 Vista previa de resultados (con el motor integrado) ... 112
8.4 Validar la asignación y su resultado ... 114
8.5 Compilar una asignación de MapForce ... 117
8.6 Implementar una asignación de MapForce ... 118
8.7 Parámetros de la línea de comandos ... 121
8.8 Archivos de catálogo en MapForce ... 127
8.9 Documentar proyectos de asignación ... 132
8.9.1 Hojas de estilos predefinidas ... 138
8.9.2 Diseño de documentación definido por el usuario ... 142
8.10 Archivos SPS en la vista previa de MapForce ... 144
8.10.1 Asignar un archivo SPS a un componente ... 146

9 Asignaciones entre componentes 150

9.1 Métodos de asignación de datos ... 155
9.1.1 Basada en el destino (asignación estándar) ... 156
9.1.2 Basada en el origen (asignación de contenido mixto) ... 157
Asignación de contenido mixto ... 157

Ejemplo de contenido mixto ... 162
Usar asignaciones estándar en elementos de contenido mixto ... 163
9.1.3 Conexiones de copia total ... 165
9.2 Configuración de la conexión ... 168
9.3 Conexiones y resultados de la asignación ... 170
9.4 Secuencia de procesamiento de los componentes ... 171
9.5 Asignaciones en cadena / componentes de paso a través ... 174
9.5.1 Asignaciones en cadena: paso a través activo ... 176
9.5.2 Asignaciones en cadena: paso a través inactivo ... 181
9.5.3 Ejemplo de asignación en cadena ... 186
9.6 Uso de las funciones ... 189
9.7 Bucles, grupos y jerarquías ... 192
9.8 Reglas y estrategias de asignación de datos ... 193

10 Datos de origen y destino 200

10.1 Datos XML y esquemas XML ... 201
10.1.1 Usar documentos DTD como componentes de esquema ... 202
10.1.2 Asignar tipos derivados de XML Schema ... 203
10.1.3 Uso de nombres QName ... 205
10.1.4 Valores nulos / que pueden ser nulos ... 208
10.1.5 Comentarios e instrucciones de procesamiento ... 212
10.1.6 Secciones CDATA ... 214
10.1.7 Comodines xs:any / xs:anyAttribute ... 216
10.1.8 Firmas digitales ... 221
Opciones de configuración de la firma XML ... 225
10.2 Trabajar con bases de datos ... 230
10.2.1 Instalar clientes y controladores de BD ... 232
SQL Server ... 232
SQL Server 2000 ... 232
SQL Server 2005 ... 232
SQL Server 2008 ... 233
Oracle ... 233
Oracle 9i ... 233
Oracle 10g ... 233
Oracle 11g ... 233
Instalar el cliente Oracle ... 233
IBM DB2 ... 234
DB2 Versión 8 / 9 ... 234
DB2 for i 5.4 ... 235
MySQL ... 235
MySQL 4 / 5 ... 235

PostgreSQL ... 235
PostgreSQL 8.x ... 235
Sybase ... 236
Sybase 12 ... 236
Microsoft Access ... 236
Firebird ... 236
Firebird 2.0.5 / 2.1.2 ... 236
10.2.2 Tutorial: asignar datos XML a bases de datos ... 238
Configurar la asignación de datos XML a BD ... 238
Insertar bases de datos: personalizar la vista previa de las tablas ... 242
Componentes y relaciones de tablas ... 244
Acción de base de datos: inserción ... 246
Insertar datos en una tabla de BD ... 247
Insertar tablas y tablas secundarias relacionadas ... 250
Acción de base de datos: actualización ... 254
Actualizar campos de BD ... 254
Actualizar y añadir registros nuevos ... 257
Actualizar y eliminar datos secundarios ... 261
Acción de base de datos: eliminación ... 264
Acción de base de datos: omisión ... 267
Generar valores de salida de la BD ... 269
10.2.3 Uso del asistente para la conexión ... 270
10.2.4 Conexiones JDBC ... 277
10.2.5 Acciones de tabla, configuración de clave y transacciones ... 281
Acciones de tabla ... 282
Tiempos de espera de ejecución ... 285
Transferencias masivas (MapForce Server) ... 286
10.2.6 Comparación de características de BD ... 291
MS Access ... 291
MS SQL Server ... 292
Oracle ... 294
MySQL ... 295
Sybase ... 296
IBM DB2 ... 297
10.2.7 Relaciones de BD: cómo conservarlas/descartarlas ... 300
10.2.8 Relaciones locales: crear relaciones de BD ... 303
10.2.9 Asignación de datos de BD de gran tamaño ... 307
Importar toda la base de datos ... 308
Importar solo una parte de la base de datos ... 309
10.2.10 Funciones de procesamiento de valores NULL ... 311
10.2.11 Componente WHERE/ORDER de SQL ... 313
Operadores WHERE/ORDER de SQL ... 316

10.2.12 Asignaciones entre datos XML y de BD ... 319
10.2.13 IBM DB2: asignaciones entre datos XML y de BD ... 324
Consultar y asignar datos en IBM DB2 ... 329
Asignar datos XML: bases de datos IBM DB2 como destino ... 331
Asignar datos de una BD a otra ... 333
10.2.14 SQL Server 2005: asignaciones de datos XML ... 337
10.2.15 Consultar BD en el panel de consulta ... 342
Seleccionar y conectarse a la base de datos ... 342
Seleccionar un recurso global de base de datos ... 346
Consultar los datos ... 349
Consulta de la BD: editor SQL ... 350
Generar instrucciones SQL ... 351
Ejecutar instrucciones SQL ... 352
Guardar y abrir scripts SQL ... 353
Características del editor SQL ... 353
Finalización automática ... 354
Convertir texto en comentario ... 354
Marcadores ... 356
Insertar regiones ... 356
Consulta de la BD: ventana Explorador ... 357
Filtrar y buscar objetos de la BD ... 359
Opciones contextuales de la ventana Explorador ... 361
Consulta de la BD: pestañas Resultados/Mensajes ... 362
Consulta de la BD: configuración ... 365
Opciones del editor SQL ... 365
Generación ... 366
Finalización automática ... 367
Vista de resultados ... 368
Fuentes ... 368
10.2.16 Instrucciones SELECT como tablas virtuales ... 370
Crear instrucciones SELECT ... 370
Parámetros de la instrucción SELECT ... 374
Ejemplo de instrucción SELECT ... 377
10.2.17 Procedimientos almacenados ... 379
Insertar procedimientos almacenados en componentes de BD ... 381
Casos de uso ... 383
Procedimientos almacenados y relaciones locales ... 383
Procedimientos almacenados como origen de datos ... 384
Procedimientos almacenados sin parámetros de entrada ... 385
Llamada con parámetros: entrada/salida ... 388
Componentes de origen y relaciones locales ... 391
Procedimientos almacenados en componentes de destino ... 395
Usar procedimientos almacenados para generar claves principales ... 398

10.3 Asignar archivos de texto/CSV ... 403
10.3.1 Asignar archivos CSV a archivos XML ... 404
10.3.2 Asignar datos XML a archivos CSV iterando en los elementos ... 407
10.3.3 Crear jerarquías a partir de archivos CSV y FLF ... 410
10.3.4 Opciones de los archivos CSV ... 414
10.3.5 Asignar archivos de longitud fija (a una BD) ... 418
Opciones de los archivos de longitud fija ... 425
10.3.6 Asignar datos de BD a archivos de texto/CSV ... 430
10.4 MapForce FlexText ... 432
10.4.1 Introducción ... 433
10.4.2 Tutorial de MapForce FlexText ... 436
10.4.3 Crear condiciones de división ... 439
10.4.4 Definir varias condiciones por contenedor/fragmento ... 441
10.4.5 Usar plantillas de FlexText en MapForce ... 446
Orígenes de datos en FlexText ... 449
10.4.6 Usar FlexText como componente de destino ... 450
10.4.7 Referencia del usuario de FlexText ... 452
División repetida ... 452
Modo: longitud fija ... 453
Modo: delimitado (flotante) ... 454
Modo: delimitado (basado en línea) ... 456
Modo: delimitado (la línea empieza con) ... 458
Dividir una vez ... 459
Modo: longitud fija ... 461
Modo: delimitado (flotante) ... 462
Modo: delimitado (basado en línea) ... 463
Conmutador ... 464
Nodo ... 467
Omitir ... 469
Guardar como CSV (delimitado) ... 470
Guardar como FLF (longitud fija) ... 476
Guardar valor ... 478
10.5 EDI: UN/EDIFACT, ANSI X12, HL7, PADIS, SAP IDoc ... 480
10.5.1 Terminología EDI ... 482
10.5.2 Asignaciones UN/EDIFACT - XML Schema ... 483
10.5.3 Validación de componentes EDI ... 494
10.5.4 X12: Confirmación funcional 997 ... 498
10.5.5 X12: Confirmación de implementación 999 ... 500
10.5.6 Segmento TA1 ... 502
10.5.7 Varios mensajes EDI por componente ... 504
10.5.8 UN/EDIFACT y ANSI X12 como destino ... 507
Validación de componentes de destino UN/EDIFACT ... 510

Validación de componentes de destino ANSI X12 ... 512
10.5.9 Personalizar un mensaje EDIFACT ... 515
Configurar el ejemplo de personalización ... 515
Personalización global ... 517
Personalización local ... 518
Ejemplo ... 520
10.5.10 Personalizar transacciones ANSI X12 ... 523
Archivos de origen X12 y varios campos idénticos ... 523
Configurar el ejemplo de personalización ... 524
Personalización global ... 526
Personalización local ... 527
Ejemplo ... 528
10.5.11 Nodos distintos combinando entradas ... 530
10.5.12 Actualizar archivos de configuración ... 533
10.5.13 SAP IDocs ... 535
10.5.14 IATA PADIS ... 537
10.5.15 HIPAA X12 ... 540
Transacciones HIPAA ... 544
10.6 Archivos MS OOXML Excel 2007+ ... 547
10.6.1 Definir los elementos asignables del libro Excel ... 548
10.6.2 Usar nombres Excel para rangos de celdas y tablas ... 554
10.6.3 Agregar, definir y mover rangos de filas ... 558
10.6.4 Seleccionar un rango de celdas (opciones) ... 564
10.6.5 Asignar archivos Excel a XML ... 568
10.6.6 Asignar datos de BD a Excel ... 571
10.6.7 Añadir datos a hojas Excel con formato previo ... 576
10.7 Asignación de datos XBRL ... 579
10.7.1 Insertar documentos XBRL ... 580
10.7.2 Componentes XBRL ... 584
Hipercubos XBRL ... 593
Función de asignación de valores XBRL ... 599
10.7.3 Ejemplo de asignación de datos Excel a XBRL ... 601
Cómo se hizo ... 601
10.7.4 Ejemplo de asignación de datos de BD a XBRL ... 605
10.8 Asignación de datos entre HL7 v3 y esquemas XML ... 607

11 Consejos prácticos 612

11.1 Filtrar datos: recuperar datos dinámicos ... 614
11.2 Componentes de filtrado: consejos ... 617

11.3 Componentes de ordenación: ordenar secuencias de entrada 619
11.4 Asignación de valores: transformar datos de entrada 628
11.4.1 Pasar datos sin cambiarlos a través de una asignación de valores 633
11.4.2 Propiedades de las asignaciones de valores 637
11.5 Reemplazar caracteres especiales en datos de BD 639
11.6 Funciones de agregado: min, max, sum, count, avg 641
11.7 Asignar varias tablas a un archivo XML 643
11.8 Elemento raíz de los documentos de destino 645
11.9 Comparación booleana de los nodos de entrada 646
11.10 Nodo de contexto prioritario 648
11.11 Unir varios archivos en un solo archivo de destino 651
11.12 Línea de comandos: definir parámetros de entrada 653
11.13 Parámetros de entrada: configuración predeterminada y vista previa 655
11.14 Nombres de los componentes 659
11.15 Filtrar datos de base de datos por fecha 661
11.16 Comprobación, posición y agrupamiento de nodos 662
11.16.1 Asignar nodos que faltan con not-exists 664
11.16.2 Posición de los elementos de contexto en una secuencia 666
11.16.3 Agrupamiento de nodos / contenido de nodos 669
11.17 Excepciones 676
11.18 Asignación recursiva definida por el usuario 678
11.18.1 Definir una función recursiva definida por el usuario 680

12 Recursos globales 688

12.1 Recursos globales: archivos 689
12.1.1 Definir y agregar recursos globales 690
12.1.2 Asignar un recurso global 693
12.1.3 Usar y activar un recurso global 695
12.2 Recursos globales: carpetas 697
12.3 Recursos globales: flujo de trabajo en la aplicación 701
12.3.1 Iniciar el flujo de trabajo en la aplicación 707
12.4 Recursos globales: bases de datos 711
12.5 Recursos globales: propiedades 717

13 Archivos de entrada/salida dinámicos por componente 722

13.1 Nombres de archivo dinámicos: entrada/salida 724

13.2 Nombres de archivo dinámicos como parámetros de entrada 727
13.3 Crear varios archivos XML a partir de un solo XML de origen 729
13.4 Crear varios archivos XML a partir de una tabla 732
13.5 Crear varios archivos XML a partir de Excel 734
13.6 Rutas de acceso de archivo relativas y absolutas 740

14 Variables intermedias 746

14.1 Variables: casos de uso 752

15 Bibliotecas y funciones 756

15.1 Definir funciones definidas por el usuario 757
15.1.1 Parámetros de las funciones 763
15.1.2 Funciones definidas por el usuario inline y normales 767
15.1.3 Crear una función de búsqueda sencilla 769
15.1.4 Función definida por el usuario compleja: con un nodo XML como entrada . 774
Definir componentes de entrada complejos 775
15.1.5 Función definida por el usuario compleja: con un nodo XML como salida ... 780
Definir componentes de salida complejos 780
15.1.6 Ejemplo de función definida por el usuario 786
15.2 Agregar funciones XSLT y XQuery personales 793
15.2.1 Agregar clases Java y funciones .NET DLL personales 794
15.2.2 Agregar funciones XSLT 1.0 personales 796
15.2.3 Agregar funciones XSLT 2.0 personales 800
15.2.4 Agregar funciones XQuery personales 801
15.2.5 Funciones de agregado: sumar nodos en XSLT 1.0/2.0 802
15.3 Agregar bibliotecas personales de funciones Java, C# y C++ 805
15.3.1 Configurar el archivo MFF 807
15.3.2 Definir la interfaz de usuario del componente 810
15.3.3 Detalles de la implementación de la función 812
15.3.4 Cómo escribir sus propias bibliotecas 813
Crear una biblioteca Java 813
Crear una biblioteca C# 815
Crear una biblioteca C++ 817
15.4 Detalles de las funciones Java y .NET 820
15.5 Referencia de funciones 823
15.5.1 Biblioteca core 825
aggregate (funciones de agregado) 825
conversion (funciones de conversión) 828

file path (funciones de ruta de acceso) ... 837
generator (funciones generador) ... 839
logical (funciones lógicas) ... 841
math (funciones matemáticas) ... 843
node (funciones de nodo) ... 846
sequence (funciones de secuencia) ... 848
string (funciones de cadena) ... 852
Ejemplos de la función tokenize ... 856
Expresiones regulares ... 860
15.5.2 Biblioteca db ... 864
15.5.3 Biblioteca edifact ... 866
15.5.4 Biblioteca lang ... 868
funciones QName ... 868
generator (funciones generador) ... 868
datetime (funciones de fecha y hora) ... 869
logical (funciones lógicas) ... 880
math (funciones matemáticas) ... 881
string (funciones de cadena) ... 883
15.5.5 Biblioteca xbrl ... 886
15.5.6 Biblioteca xlsx ... 887
15.5.7 Biblioteca xpath2 ... 888
accessors (descriptores de acceso) ... 888
anyURI ... 888
boolean (funciones booleanas) ... 889
constructores ... 889
context (funciones de contexto) ... 890
duration, date and time (funciones de duración, fecha y hora) ... 891
node (funciones de nodo) ... 893
numeric (funciones numéricas) ... 894
qname-related (funciones para QName) ... 895
string (funciones de cadena) ... 895
15.5.8 Biblioteca xslt ... 898
xpath ... 898
xslt ... 900

16 Implementar servicios web 904

16.1 Información sobre WSDL: protocolos compatibles ... 905
16.2 Crear proyectos de servicio web a partir de archivos WSDL ... 908
16.3 Generar servicios web Java con MapForce ... 911

16.3.1 Usar el servicio web: operación getPerson ... 912
16.3.2 Usar el servicio web: operación putPerson ... 914
16.4 Generar servicios web C# con MapForce ... 916
16.5 Elementos de error fault del servicio web ... 918

17 Llamar a servicios web 922

17.1 Llamar a la función de servicio web getCityTime ... 923
17.2 Llamar al servicio web getPerson ... 929

18 Complemento de MapForce para MS Visual Studio 932

18.1 Abrir archivos de MapForce en Visual Studio ... 933
18.2 Diferencias en MapForce para Visual Studio ... 936

19 Complemento de MapForce para Eclipse 940

19.1 Instalar el complemento de MapForce para Eclipse ... 941
19.2 Iniciar Eclipse y usar el complemento de MapForce ... 947
19.3 MapForce: editor, vista y perspectivas ... 950
19.4 Importar la carpeta de ejemplos de MapForce ... 951
19.5 Crear archivos de MapForce nuevos ... 953
19.6 Generación de código con MapForce ... 954
19.6.1 Generar código de asignación a mano ... 955
19.6.2 Usar proyectos de MapForce/Eclipse para generación automática ... 956
19.6.3 Agregar la naturaleza MapForce a un proyecto de Eclipse ... 960
19.7 Ampliar el complemento de MapForce ... 961

20 Referencia del usuario 966

20.1 Archivo ... 967
20.2 Edición ... 973
20.3 Insertar ... 974
20.4 Proyecto ... 978
20.5 Componente ... 981
20.6 Conexión ... 989
20.7 Función ... 994
20.8 Resultados ... 997

20.9 Vista 999
20.10 Herramientas 1002
20.11 Ventanas 1012
20.12 Ayuda 1013
20.12.1 Contenido, índice, buscar 1014
20.12.2 Activación, formulario de pedido, registro, actualizaciones 1015
20.12.3 Otros comandos 1016

21 Generador de código 1018

21.1 Introducción al generador de código 1019
21.2 Novedades 1021
21.3 Generar código de programa 1023
21.3.1 Generar código Java 1026
21.3.2 Generar código C# 1028
21.3.3 Generar código C++ 1031
21.4 Ejemplo de asignación para generación de código 1034
21.5 Integrar código de MapForce en su aplicación 1037
21.5.1 Código de MapForce en aplicaciones Java 1038
21.5.2 Código de MapForce en aplicaciones C# 1040
21.5.3 Código de MapForce en aplicaciones C++ 1042
21.5.4 Compatibilidad con secuencias de datos 1044
21.6 Usar la biblioteca de código generado 1048
21.6.1 Esquema de ejemplo 1050
21.6.2 Usar la biblioteca de código Java generado 1051
21.6.3 Usar la biblioteca de código C++ generado 1058
21.6.4 Usar la biblioteca de código C# generado 1066
21.7 Consejos para la generación de código 1073
21.8 Opciones del generador de código 1075
21.9 SPL: el lenguaje de programación Spy 1076
21.9.1 Estructura básica de SPL 1077
21.9.2 Declaraciones 1078
21.9.3 Variables 1080
21.9.4 Variables predefinidas 1081
21.9.5 Crear archivos de salida 1082
21.9.6 Operadores 1083
21.9.7 Condiciones 1084
21.9.8 Colecciones y foreach 1085
21.9.9 Subrutinas 1087
Declaración de subrutina 1087

Invocación de subrutina ... 1088
Ejemplo de subrutina ... 1088
21.9.10 Tipos integrados ... 1091
Library ... 1091
Namespace ... 1091
Type ... 1091
Member ... 1092
NativeBinding ... 1093
Facets ... 1093

22 The MapForce API 1096

22.1 Overview ... 1097
22.1.1 Object model ... 1098
22.1.2 Example: Code-Generation ... 1099
22.1.3 Example: Mapping Execution ... 1101
22.1.4 Example: Project Support ... 1105
22.1.5 Error handling ... 1110
22.1.6 Programming Languages ... 1112
C# ... 1112
Java ... 1118
Example Java Project ... 1119
JScript ... 1122
Start application ... 1123
Simple document access ... 1124
Generate code ... 1125
22.2 Object Reference ... 1128
22.2.1 Application ... 1129
Events ... 1130
OnDocumentOpened ... 1130
OnProjectOpened ... 1130
OnShutdown ... 1130
ActiveDocument ... 1131
ActiveProject ... 1131
Application ... 1131
Documents ... 1131
Edition ... 1131
GlobalResourceConfig ... 1132
GlobalResourceFile ... 1132
HighlightSerializedMarker ... 1132
IsAPISupported ... 1133
MajorVersion ... 1133

MinorVersion ... 1133
Name ... 1133
NewDocument ... 1133
NewProject ... 1134
OpenDocument ... 1134
OpenProject ... 1134
OpenURL ... 1134
Options ... 1135
Parent ... 1135
Quit ... 1135
ServicePackVersion ... 1135
Status ... 1136
Visible ... 1136
WindowHandle ... 1136
22.2.2 AppOutputLine ... 1137
Application ... 1137
ChildLines ... 1137
GetCellCountInLine ... 1138
GetCellIcon ... 1138
GetCellSymbol ... 1138
GetCellText ... 1138
GetCellTextDecoration ... 1139
GetIsCellText ... 1139
GetLineCount ... 1139
GetLineSeverity ... 1139
GetLineSymbol ... 1140
GetLineText ... 1140
GetLineTextEx ... 1140
GetLineTextWithChildren ... 1140
GetLineTextWithChildrenEx ... 1140
Parent ... 1141
22.2.3 AppOutputLines ... 1142
Application ... 1142
Count ... 1142
Item ... 1142
Parent ... 1142
22.2.4 AppOutputLineSymbol ... 1144
Application ... 1144
GetSymbolHREF ... 1144
GetSymbolID ... 1144
IsSymbolHREF ... 1145

Parent ... 1145
22.2.5 Component ... 1146
Application ... 1146
CanChangeInputInstanceFile ... 1146
CanChangeOutputInstanceFile ... 1147
ComponentName ... 1147
GenerateOutput ... 1147
GetRootDatapoint ... 1148
HasIncomingConnections ... 1148
HasOutgoingConnections ... 1149
ID ... 1149
InputInstanceFile ... 1149
IsParameterInputRequired ... 1149
IsParameterSequence ... 1149
Name ... 1150
OutputInstanceFile ... 1150
Parent ... 1150
Preview ... 1150
Schema ... 1151
SubType ... 1151
Type ... 1151
UsageKind ... 1152
22.2.6 Components ... 1153
Application ... 1153
Count ... 1153
Item ... 1153
Parent ... 1153
22.2.7 Connection ... 1155
Application ... 1155
ConnectionType ... 1155
Parent ... 1155
22.2.8 Datapoint ... 1156
Application ... 1156
GetChild ... 1156
Parent ... 1156
22.2.9 Document ... 1158
Events ... 1159
OnDocumentClosed ... 1159
OnModifiedFlagChanged ... 1159
Activate ... 1159
Application ... 1159

Close ... 1159
CreateUserDefinedFunction ... 1160
FindComponentByID ... 1160
FullName ... 1160
GenerateCHashCode ... 1160
GenerateCodeEx ... 1161
GenerateCppCode ... 1161
GenerateJavaCode ... 1161
GenerateOutput ... 1162
GenerateOutputEx ... 1162
GenerateXQuery ... 1163
GenerateXSLT ... 1163
GenerateXSLT2 ... 1163
HighlightSerializedMarker ... 1163
JavaSettings_BasePackageName ... 1164
MainMapping ... 1164
MapForceView ... 1164
Mappings ... 1164
Name ... 1165
OutputSettings_ApplicationName ... 1165
OutputSettings_Encoding (obsolete) ... 1165
Parent ... 1165
Path ... 1166
Save ... 1166
SaveAs ... 1166
Saved ... 1166
22.2.10 Documents ... 1167
ActiveDocument ... 1167
Application ... 1167
Count ... 1167
Item ... 1168
NewDocument ... 1168
OpenDocument ... 1168
Parent ... 1168
22.2.11 ErrorMarker ... 1169
Application ... 1169
DocumentFileName ... 1169
ErrorLevel ... 1169
Highlight ... 1170
Serialization ... 1170
Text ... 1170

Parent ... 1170
22.2.12 ErrorMarkers ... 1171
Application ... 1171
Count ... 1171
Item ... 1171
Parent ... 1171
22.2.13 MapForceView ... 1173
Active ... 1173
ActiveMapping ... 1173
ActiveMappingName ... 1174
Application ... 1174
HighlightMyConnections ... 1174
HighlightMyConnectionsRecursivey ... 1174
InsertWSDLCall ... 1174
InsertXMLFile (obsolete) ... 1175
InsertXMLSchema (obsolete) ... 1175
InsertXMLSchemaWithSample (obsolete) ... 1176
Parent ... 1176
ShowItemTypes ... 1176
ShowLibraryInFunctionHeader ... 1176
22.2.14 Mapping ... 1178
Application ... 1178
Components ... 1178
CreateConnection ... 1178
InsertFunctionCall ... 1179
InsertXMLFile ... 1179
InsertXMLSchema ... 1180
InsertXMLSchemaInputParameter ... 1180
InsertXMLSchemaOutputParameter ... 1180
IsMainMapping ... 1181
Name ... 1181
Parent ... 1181
22.2.15 Mappings ... 1183
Application ... 1183
Count ... 1183
Item ... 1183
Parent ... 1183
22.2.16 Options ... 1185
Application ... 1185
CodeDefaultOutputDirectory ... 1185
CPPSettings_DOMType ... 1186

CPPSettings_GenerateVC6ProjectFile ... 1186
CppSettings_GenerateVSProjectFile ... 1186
CPPSettings_LibraryType ... 1187
CPPSettings_UseMFC ... 1187
CSharpSettings_ProjectType ... 1187
DefaultOutputByteOrder ... 1187
DefaultOutputByteOrderMark ... 1188
DefaultOutputEncoding ... 1188
GenerateWrapperClasses ... 1188
JavaSettings_ApacheAxisVersion (obsolete) ... 1189
Parent ... 1189
ShowLogoOnPrint ... 1189
ShowLogoOnStartup ... 1189
UseGradientBackground ... 1189
XSLTDefaultOutputDirectory ... 1190
22.2.17 Project (Enterprise or Professional Edition) ... 1191
Events ... 1191
OnProjectClosed ... 1192
_NewEnum ... 1192
AddActiveFile ... 1193
AddFile ... 1193
Application ... 1193
Close ... 1193
Count ... 1194
CreateFolder ... 1194
FullName ... 1194
GenerateCode ... 1194
GenerateCodeEx ... 1195
GenerateCodeIn ... 1195
GenerateCodeInEx ... 1195
InsertWebService ... 1195
Item ... 1196
Java_BasePackageName ... 1196
Name ... 1196
Output_Folder ... 1197
Output_Language ... 1197
Output_TextEncoding ... 1197
Parent ... 1197
Path ... 1198
Save ... 1198
Saved ... 1198

22.2.18 ProjectItem (Enterprise or Professional Edition) 1199
_NewEnum 1199
AddActiveFile 1200
AddFile 1200
Application 1200
CodeGenSettings_Language 1201
CodeGenSettings_OutputFolder 1201
CodeGenSettings_UseDefault 1201
Count 1201
CreateFolder 1202
CreateMappingForProject 1202
GenerateCode 1202
GenerateCodeEx 1202
GenerateCodeIn 1203
GenerateCodeInEx 1203
Item 1203
Kind 1204
Name 1204
Open 1204
Parent 1205
QualifiedName 1205
Remove 1205
WSDLFile 1205
22.3 Enumerations 1206
22.3.1 ENUMApacheAxisVersion (obsolete) 1207
22.3.2 ENUMApplicationStatus 1208
22.3.3 ENUMAppOutputLine_Severity 1209
22.3.4 ENUMAppOutputLine_TextDecoration 1210
22.3.5 ENUMCodeGenErrorLevel 1211
22.3.6 ENUMComponentDatapointSide 1212
22.3.7 ENUMComponentSubType 1213
22.3.8 ENUMComponentType 1214
22.3.9 ENUMComponentUsageKind 1215
22.3.10 ENUMConnectionType 1216
22.3.11 ENUMDOMType 1217
22.3.12 ENUMLibType 1218
22.3.13 ENUMProgrammingLanguage 1219
22.3.14 ENUMProjectItemType 1220
22.3.15 ENUMProjectType 1221
22.3.16 ENUMSearchDatapointFlags 1222
22.3.17 ENUMViewMode 1223

23 ActiveX Integration 1226

23.1 Integration at Application Level ... 1227
23.2 Integration at Document Level ... 1228
23.2.1 Use MapForceControl ... 1229
23.2.2 Use MapForceControlDocument ... 1230
23.2.3 Use MapForceControlPlaceHolder ... 1231
23.2.4 Query MapForce Commands ... 1232
23.3 Programming Languages ... 1233
23.3.1 C# ... 1234
Introduction ... 1234
Placing the MapForceControl ... 1234
Adding the Placeholder Control ... 1235
Retrieving Command Information ... 1237
Handling Events ... 1239
Testing the Example ... 1240
23.3.2 HTML ... 1243
Integration at Application Level ... 1243
Instantiate the Control ... 1243
Add Button to Open Default Document ... 1243
Add Buttons for Code Generation ... 1244
Connect to Custom Events ... 1245
Integration at Document Level ... 1245
Instantiate the MapForceControl ... 1245
Create Editor Window ... 1246
Create Project Window ... 1246
Create Placeholder for Helper Windows ... 1246
Create a Custom Toolbar ... 1247
Create More Buttons ... 1247
Create Event Handler to Update Button Status ... 1249
23.3.3 Java ... 1250
Example Java Project ... 1251
Creating the ActiveX Controls ... 1259
Loading Data in the Controls ... 1259
Basic Event Handling ... 1260
Menus ... 1260
UI Update Event Handling ... 1262
Listing the Properties of a MapForce Mapping ... 1263
23.3.4 Visual Basic ... 1265
23.4 Command Table for MapForce ... 1266
23.4.1 File Menu ... 1267
23.4.2 Edit Menu ... 1268

23.4.3 Insert Menu ... 1269
23.4.4 Project Menu ... 1270
23.4.5 Component Menu ... 1271
23.4.6 Connection Menu ... 1272
23.4.7 Function Menu ... 1273
23.4.8 Output Menu ... 1274
23.4.9 View Menu ... 1275
23.4.10 Tools Menu ... 1276
23.4.11 Window Menu ... 1277
23.4.12 Help Menu ... 1278
23.4.13 Commands Not in Main Menu ... 1279
23.5 Accessing MapForceAPI ... 1280
23.6 Object Reference ... 1281
23.6.1 MapForceCommand ... 1282
Accelerator ... 1282
ID ... 1282
IsSeparator ... 1282
Label ... 1283
StatusText ... 1283
SubCommands ... 1283
ToolTip ... 1283
23.6.2 MapForceCommands ... 1284
Count ... 1284
Item ... 1284
23.6.3 MapForceControl ... 1285
Properties ... 1285
Appearance ... 1286
Application ... 1286
BorderStyle ... 1286
CommandsList ... 1286
EnableUserPrompts ... 1286
IntegrationLevel ... 1286
MainMenu ... 1287
Toolbars ... 1287
Methods ... 1287
Exec ... 1287
Open ... 1287
QueryStatus ... 1288
Events ... 1288
OnCloseEditingWindow ... 1288
OnContextChanged ... 1289
OnDocumentOpened ... 1289
OnFileChangedAlert ... 1289
OnLicenseProblem ... 1289
OnOpenedOrFocused ... 1289

OnToolWindowUpdated ... 1290
OnUpdateCmdUI ... 1290
OnValidationWindowUpdated ... 1290
23.6.4 MapForceControlDocument ... 1291
Properties ... 1291
Appearance ... 1291
BorderStyle ... 1292
Document ... 1292
IsModified ... 1292
Path ... 1292
ReadOnly ... 1292
Methods ... 1293
Exec ... 1293
New ... 1293
Open ... 1293
QueryStatus ... 1293
Reload ... 1294
Save ... 1294
SaveAs ... 1294
Events ... 1294
OnActivate ... 1295
OnContextChanged ... 1295
OnDocumentClosed ... 1295
OnDocumentOpened ... 1295
OnDocumentSaveAs ... 1295
OnFileChangedAlert ... 1295
OnModifiedFlagChanged ... 1296
OnSetEditorTitle ... 1296
23.6.5 MapForceControlPlaceHolder ... 1297
Properties ... 1297
Label ... 1297
PlaceholderWindowID ... 1297
Project ... 1297
Methods ... 1298
OpenProject ... 1298
CloseProject ... 1298
Events ... 1298
OnModifiedFlagChanged ... 1298
OnSetLabel ... 1299
23.6.6 Enumerations ... 1300
ICActiveXIntegrationLevel ... 1300
MapForceControlPlaceholderWindow ... 1300

24 Anexos 1302

24.1 Información sobre motores de Altova ... 1303
24.1.1 Motor XSLT 1.0: información de implementación ... 1304
24.1.2 Motor XSLT 2.0: información de implementación ... 1306

Información general ... 1306
Elementos y funciones XSLT 2.0 ... 1308
24.1.3 Motor XQuery 1.0: información de implementación ... 1309
24.1.4 Funciones XPath 2.0 y XQuery 1.0 ... 1312
Información general ... 1312
Funciones compatibles ... 1313
24.1.5 Funciones de extensión XSLT y XQuery ... 1317
Funciones de extensión de Altova ... 1317
Funciones XSLT generales ... 1318
Funciones XPath generales ... 1322
Funciones para gráficos (XPath) ... 1323
Estructura XML de los datos de gráfico ... 1327
Ejemplo: funciones de gráficos ... 1332
Funciones de extensión Java ... 1335
Archivos de clases definidos por el usuario ... 1336
Archivos JAR definidos por el usuario ... 1339
Java: constructores ... 1340
Java: métodos estáticos y campos estáticos ... 1341
Java: métodos de instancia y campos de instancia ... 1342
Tipos de datos: XPath/XQuery a Java ... 1342
Tipos de datos: Java a XPath/XQuery ... 1343
Funciones de extensión .NET ... 1344
.NET: constructores ... 1346
.NET: métodos estáticos y campos estáticos ... 1347
.NET: métodos de instancia y campos de instancia ... 1348
Tipos de datos: XPath/XQuery a .NET ... 1348
Tipos de datos: .NET a XPath/XQuery ... 1349
Scripts MSXSL para XSLT ... 1350
24.2 Datos técnicos ... 1353
24.2.1 Requisitos de SO y memoria ... 1354
24.2.2 Validador XML de Altova ... 1355
24.2.3 Motores XSLT y XQuery de Altova ... 1356
24.2.4 Compatibilidad con Unicode ... 1357
24.2.5 Uso de Internet ... 1358
24.3 Información sobre licencias ... 1359
24.3.1 Distribución electrónica de software ... 1360
24.3.2 Activación del software y medición de licencias ... 1361
24.3.3 Derechos de propiedad intelectual ... 1362
24.3.4 Contrato de licencia para el usuario final de Altova ... 1363

Altova MapForce 2014

MapForce 2014

1 MapForce 2014

MapForce® 2014 Enterprise Edition es una herramienta visual de asignación de datos para proyectos de integración de datos avanzados. MapForce® es una aplicación de Windows (de 32 y 64 bits) compatible con Windows 8, Windows 7, Windows Vista, Windows XP y Windows Server 2003/2008/2012. Las ediciones Enterprise y Professional ofrecen compatibilidad con 64 bits.

Última actualización: 17/03/2014

Altova MapForce 2014

Características nuevas

2 Características nuevas

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
- Opción para definir valores de tiempo de espera para la ejecución de bases de datos y llamadas a servicios web
- Opción para conservar los conectores después de eliminar componentes
- Transferencia masiva de datos de BD (acción `Insert all` masiva)
- Opción que resalta 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 insertar comillas en archivos CSV siempre

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

- Altova publica el rapidísimo motor de transformación RaptorXML 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

 © 2014 Altova GmbH

- 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 FLF files (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 los componentes de destino gracias a las StyleVision Power Stylesheets, con gráficos de StyleVision.
-
- 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 compatibles
- Mejoras en la función **`auto-number`**
- Nuevas funciones de uso horario: **`remove-timezone`** y **`convert-to-utc`**

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: Windows Server 2003/2008, Windows XP, Windows Vista y Windows 7
- 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
- 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

 © 2014 Altova GmbH

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**:

- 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#
- Función para generar esquemas XML para archivos XML automáticamente
- Compatibilidad con SOAP 1.2 en servicios web
- Nueva opción para divisiones repetidas en FlexText: "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 (IBM DB2 y otras)
- Consulta directa de bases de datos
- Filtro SQL-WHERE y asistente para instrucciones SQL

- Compatibilidad total con EDI X12 versiones 3040 - 5030
- Compatibilidad total con mensajes UN/EDIFACT de los directorios 93A - 06B
- Mejoras en el generador de código y en la función de documentación

© 2014 Altova GmbH

Altova MapForce 2014

Introducción a MapForce

3 Introducción a MapForce

Sitio web de Altova: Introducción a MapForce

¿Qué es una asignación de datos?
Básicamente el contenido de un componente se asigna (o transforma) a otro componente. Los archivos XML, los archivos de texto, las bases de datos, los archivos Excel 2007 (o superior), los mensajes EDI y los archivos XBRL se pueden asignar a otros documentos XML, archivos de texto CSV, archivos EDI, archivos Excel 2007 o superior y documentos XBRL y bases de datos. La transformación se realiza por medio de hojas de estilos XSLT 1.0 o 2.0 generadas automáticamente y con ayuda del motor de ejecución integrado en el modo de vista previa o del código de programa generado.

Además un solo componente de la asignación puede procesar varios archivos de entrada de un directorio y generar varios archivos para un solo componente.

Cuando se crea una transformación XSLT, se asigna un **esquema de origen** a un **esquema de destino**. Por tanto, los elementos/atributos del esquema de origen se *conectan* con otros elementos/atributos del esquema de destino. Puesto que el archivo de esquema está asociado (y define) a un documento XML de instancia, al final lo que se consigue es crear una asignación de datos entre dos documentos XML.

MapForce® ofrece estas características y funciones:

- Asignación gráfica de datos entre varios esquemas XML de origen y destino

  En la edición **Professional Edition** puede usar estos archivos como componentes de origen y destino:
  - Archivos planos: delimitados por comas (CSV) y de longitud fija
  - Bases de datos relacionales

  Y en la edición **Enterprise Edition** también puede usar estos archivos como componentes de origen y destino:
  - Archivos EDI (UN/EDIFACT, ANSI X12 incluidos los formatos HIPAA, HL7 2.x, IATA PADIS y SAP IDocs)
  - Archivos de FlexText™
  - Archivos Office Open XML Excel 2007 (y superior)
  - Archivos de instancia XBRL y taxonomías XBRL
- Generación automática de código XSLT 1.0 y 2.0

  En las ediciones **Enterprise** y **Professional** también puede generar automáticamente:
  - Código XQuery
  - Código Java, C# y C++
  - Compatibilidad con versiones de 64 bits

- Transformación instantánea y vista previa de todas las asignaciones de datos, sin necesidad de generar ni compilar código
- Vista previa de componentes intermedios de una asignación en cadena de dos o más componentes conectados a un componente de destino (vista previa de paso a través).
- Vista previa del resultado de los componentes de destino usando archivos PSP de StyleVision
- Potente generador visual de funciones para crear funciones definidas por el usuario
- Acceso a la interfaz del usuario y a las funciones de MapForce a través de la API de

 © 2014 Altova GmbH

MapForce (control ActiveX)
- Definición de bibliotecas XSLT 1.0 y 2.0 personalizadas
- Compatibilidad con funciones XPath 2.0 en XSLT 2.0 y XQuery
- Definición de funciones/componentes definidos por el usuario, con entradas/salidas complejas
- Asignaciones basadas en origen/de contenido mixto y conexiones de copia total
- Retención automática de conectores de asignación de nodos/elementos que faltan
- Compatibilidad con las versiones 3.x de HL7, puesto que están basadas en XML Schema

La edición **Professional Edition** también incluye estas características:
- Asignación de datos entre documentos XML y bases de datos (IBM DB2 y otras BD)
- Consulta directa de bases de datos
- Filtro WHERE de SQL y asistente para instrucciones SQL
- Instrucciones SELECT de SQL como origen de datos de la asignación
- Integración de funciones personalizadas C++, Java y C#
- Funciones de gestión de proyecto para agrupar asignaciones
- Complemento de MapForce para Eclipse 3.8 / 4.2 / 4.3
- Complemento de MapForce para Microsoft Visual Studio 2005/2008/2010/2012/2013
- Documentación del diseño de la asignación

Y la edición **Enterprise Edition** incluye estas características adicionales:
- Creación de proyectos de servicios web SOAP 1.1 y 1.2 y asignación de operaciones de servicios web desde archivos WSDL 1.1 y 2.0
- Llamada directa a funciones de servicios web
- FlexText™: procesamiento avanzado de archivos heredados

Además la edición **Professional** y **Enterprise Edition** se pueden integrar en Eclipse 3.8 / 4.2 / 4.3 y Microsoft Visual Studio 2005/2008/2010/2012/2013.

Todas las transformaciones tienen lugar en un solo área de trabajo, donde puede mezclar varios componentes de origen y varios componentes de salida. El área de trabajo ofrece también una extensa biblioteca de funciones que se puede ampliar y permite realizar cualquier tipo de manipulación de datos.

3.1 Terminología

A continuación definimos algunos términos clave utilizados en esta documentación.

Biblioteca
Una *biblioteca* es una colección de funciones visible en la ventana Bibliotecas. Hay varios tipos de funciones, como funciones básicas, funciones propias de determinados lenguajes de programación, funciones definidas por el usuario y funciones personalizadas. Para más información, consulte el apartado sobre el uso de las funciones.

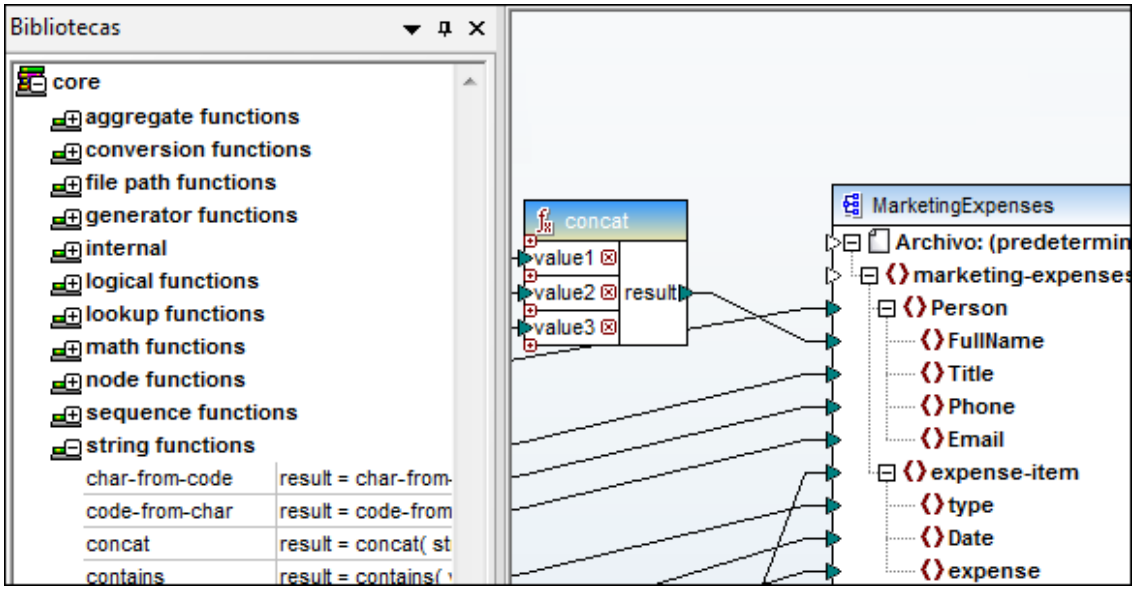
Componente
En MapForce muchos de los elementos gráficos que se pueden insertar/importar o colocar en el panel **Asignación** se convierten en *componentes*. Los componentes incluyen unos pequeños iconos en forma de **triángulo** que sirven para realizar asignaciones de datos entre los componentes de origen y los de destino, creando conexiones entre ellos.

Cuando se colocan en el panel **Asignación** estos archivos se convierten en componentes:

- Esquemas y archivos DTD: se convierten en esquemas de origen/destino
- Bases de datos: se convierten en bases de datos de origen/destino
- Archivos planos: tanto archivos CSV como otros archivos de texto
- Documentos EDI en formato UN/EDIFACT, ANSI X12 y HL7: se convierten en componentes de origen/destino
- Archivos Excel 2007 (y superior): se convierten en componentes de origen/destino
- Documentos XBRL: componentes de origen/destino
- Tipos de funciones: XSLT/XSLT2, funciones XQuery, Java, C# y C++, así como constantes, filtros y condiciones

Función
Una *función* es un componente predefinido que realiza operaciones en los datos, p. ej. la función **`concat`** (*imagen siguiente*). Las funciones tienen **parámetros** de entrada y/o salida y cada parámetro cuenta con un icono de entrada/salida. Las funciones están disponibles en la ventana Bibliotecas y se agrupan de forma lógica. Si pulsa **Ctrl+F** puede buscar una función en la ventana Bibliotecas. Si arrastra una función desde la ventana Bibliotecas hasta el panel **Asignación**, se crea un componente de tipo función. Para más información, consulte el apartado sobre el uso de las funciones.

 © 2014 Altova GmbH

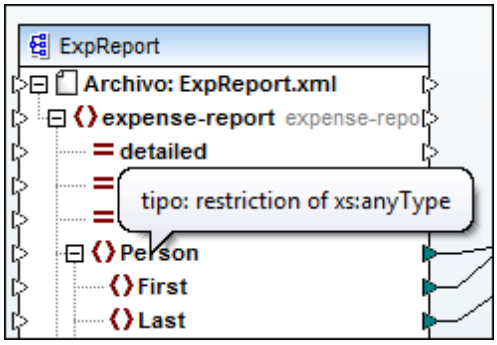


Con Java seleccionado

Elemento

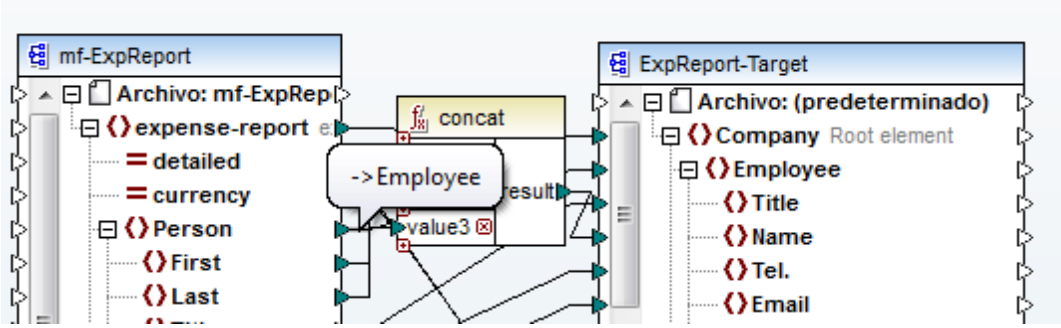
Un *elemento* representa los datos que se pueden asignar de un componente a otro. Un elemento puede ser de varios tipos: **elemento**, **atributo**, campo de BD, segmento EDI, campo Excel o ítem XBRL.

Cada **elemento** tiene un icono de **entrada** y otro de **salida**. No es obligatorio realizar asignaciones entre elementos del mismo tipo (p. ej. puede realizar una asignación entre un atributo y un elemento).



Icono de entrada / salida

Los pequeños triángulos que aparecen en los componentes son los denominados iconos de ***entrada*** y ***salida***. Cuando se hace clic en uno de estos iconos y se arrastra, se crea un **conector** que conecta el icono con el icono donde se "suelta" el conector. El conector **representa una asignación** entre los dos conjuntos de datos que los iconos representan. Para más información consulte la sección Asignaciones entre componentes.



Conector

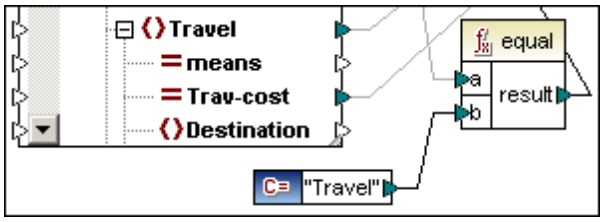
Un *conector* es la **línea** que une dos iconos. El conector representa la **asignación** existente entre los dos conjuntos de datos que los iconos representan. Para más información consulte la sección Asignaciones entre componentes.

Puede definir varios tipos de conectores:

- Conectores basados en el destino (estándar): consulte el apartado Asignaciones basadas en el destino / de contenido mixto frente a asignaciones estándar
- Conectores de copia total: consulte el apartado Conexiones de copia total
- Basadas en el origen (de contenido mixto): consulte el apartado Asignación basada en el origen (de contenido mixto)

Constante

Una *constante* es un componente que suministra datos fijos a un icono de entrada de una función o componente. Por ejemplo, en la imagen siguiente la cadena de texto "Travel" está conectada al parámetro **b** de la función **equal**. Los datos se introducen en un cuadro de diálogo al crear o hacer doble clic en el componente. Las constante tienen un icono de salida, pero no de entrada. Puede elegir entre estos tipos de constantes: cadena, número y demás opciones (cadena).



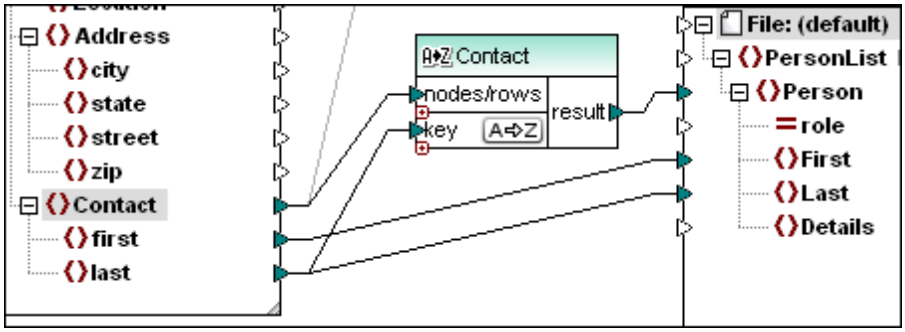
Variable

Una *variable* inserta una variable intermedia, que equivale a una función normal (no inline) definida por el usuario. Las variables son componentes estructurales, no tienen archivos de instancia y se utilizan para simplificar el proceso de asignación de datos.

Componente de ordenación

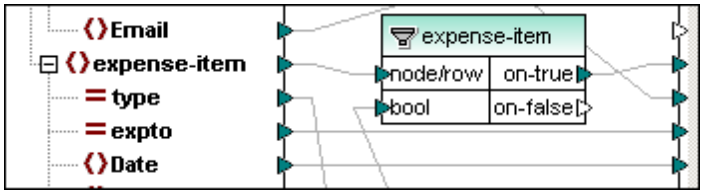
Un *componente de ordenación* ordena los datos de entrada según la clave de ordenación definida/ asignada por el usuario. El criterio de ordenación se puede cambiar haciendo clic en el icono **A=>Z** del parámetro **key** (clave) del componente de ordenación (*imagen siguiente*).

 © 2014 Altova GmbH



Filtro: nodos/filas

Un *filtro* es un componente que filtra los datos usando dos parámetros de entrada y dos salida: los parámetros de entrada **node/row** y **bool** y los de salida **on-true** y **on-false**. Si el parámetro booleano **bool** es `true`, entonces el valor/contenido del parámetro **node/row** se pasa al parámetro **on-true**.

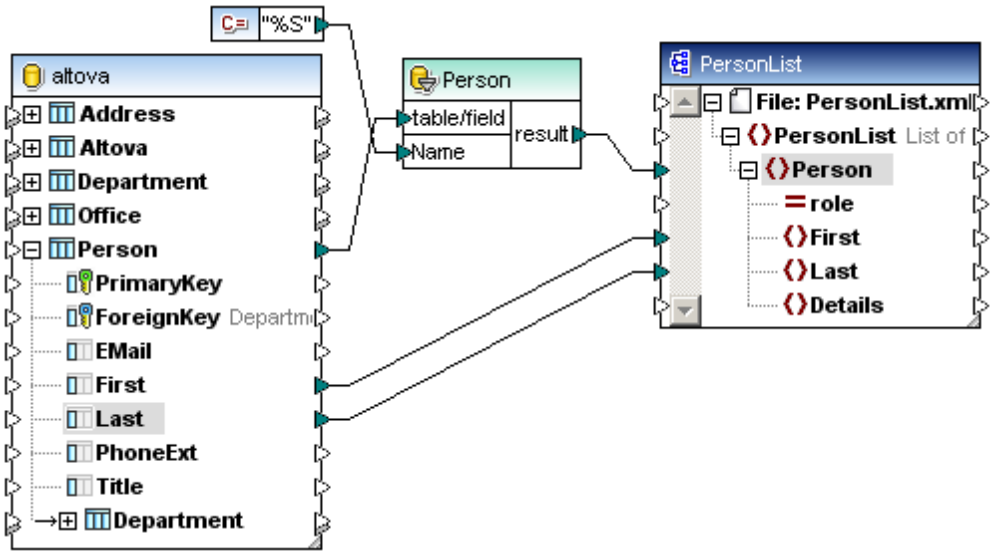


El parámetro de salida **on-false** produce el grupo de nodos definido por la asignación. Para más información consulte el apartado Trabajar con varios esquemas y documentos de destino.

Condición WHERE/ORDER de SQL

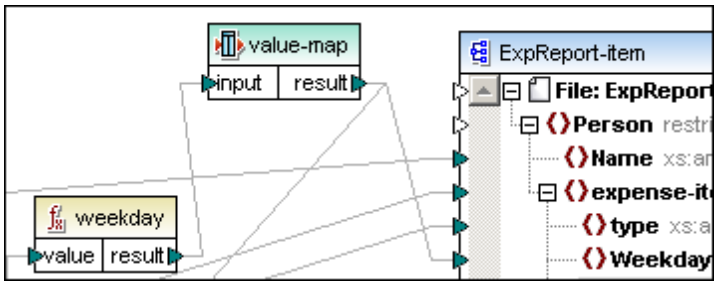
El componente *WHERE/ORDER de SQL* sirve para filtrar datos de una BD de forma condicional. Haga doble clic en este componente para introducir la instrucción WHERE/ORDER de SQL. El componente WHERE de SQL tiene varias partes:

- La instrucción **SELECT**, que se genera automáticamente cuando se establece la conexión con una tabla de BD
- La cláusula **WHERE**, que se introduce manualmente en el cuadro de texto SQL WHERE. Tenga en cuenta que las claves foráneas se incluyen automáticamente en la instrucción SELECT.
- La cláusula **ORDER BY**



Asignación de valores

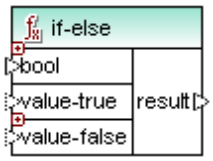
Una *asignación de valores* sirve para transformar un conjunto de datos de entrada en un conjunto diferente de datos de salida, usando un tipo de tabla de búsqueda.



Al hacer doble clic en este componente se abre la tabla de asignación de valores. La columna izquierda de la tabla define la entrada, mientras que la columna derecha define los datos transformados que desea producir.

Condición `IF-Else`

Una *condición* es un componente que sirve para pasar diferentes conjuntos de datos dependiendo del resultado de una condición predefinida. El nombre del componente es **`if-else`**. Para más información, consulte este apartado de la *Referencia del usuario*.



- El primer parámetro de entrada es un booleano (**`bool`**), que contiene los datos que se buscan.
- El parámetro de entrada **`value-true`** suministra los datos que se deben pasar si la condición da **`true`** como resultado
- El parámetro **`value-false`** suministra los datos que se deben pasar si la condición da

 © 2014 Altova GmbH

false como resultado.

- El parámetro de salida **result** genera los datos suministrados por los parámetros de entrada **value-true**/**value-false**.

La condición **IF-Else** ahora se puede ampliar. Esto significa que puede buscar varios valores **IF** a la vez y usar el parámetro **otherwise** para dar salida al valor / a la condición **Else**. Para más información consulte este apartado de la *Referencia del usuario*.

Altova MapForce 2014

MapForce Server: automatizar asignaciones

4 MapForce Server: automatizar asignaciones

MapForce Server es un producto servidor que se ejecuta en servidores de alta velocidad en sistemas operativos MS Windows, Linux y Mac OS X. MapForce Server funciona como módulo de FlowForce Server pero también se puede usar como producto servidor independiente.

MapForce Server le ayudará a automatizar sus asignaciones en FlowForce Server. Para ello es necesario implementar la asignación, definir un trabajo de FlowForce y ejecutarlo para generar el resultado de la asignación. Para más información consulte el tutorial de FlowForce Server.

So utiliza MapForce Server como producto servidor independiente, entonces debe compilar la asignación de MapForce desde la línea de comandos o con el comando correspondiente. La asignación compilada se puede ejecutar después con el comando `run` de la línea de comandos de MapForce Server.

MapForce Server es compatible con estos sistemas operativos:

- Windows Server 2003, 2008 R2 o superior
- Windows XP con Service Pack 3, Windows 7, Windows 8 o superior
- Linux (CentOS 6, Debian 6 y Ubuntu 12.10 o superior)
- Mac OS X

  Altova ofrece versiones Windows de 32 y 64 bits de MapForce Server.

Limitaciones de MapForce Server:

- no admite el uso de firmas XML
- no admite el uso de recursos globales a través de la interfaz de COM
- las conexiones de base de datos ODBC y ADO solamente se pueden usar en Windows. Los demás sistemas operativos se conectan con JDBC automáticamente.

MapForce Server está incluido en el paquete de instalación de FlowForce Server e interactúa con el servidor de FlowForce para definir trabajos, desencadenadores, usuarios, etc.

MapForce Server como producto independiente:
MapForce Server también se puede usar como producto independiente y su paquete de instalación se puede descargar del sitio web de Altova.

Las ediciones Windows, Linux y Mac OS X de MapForce Server incluyen el servidor Altova LicenseServer, necesario para gestionar las licencias de los productos servidor de Altova.

 © 2014 Altova GmbH

Altova MapForce 2014

RaptorXML Server

5 RaptorXML Server

Altova RaptorXML Server (en adelante *RaptorXML*) es el rapidísimo procesador XML y XBRL de tercera generación de Altova. Este procesador está optimizado pensando en los entornos de informática en paralelo y para los estándares más recientes. Al estar diseñado para ser altamente compatible con múltiples plataformas, este procesador aprovecha la actual omnipresencia de equipos multi-CPU para ofrecer un rapidísimo procesamiento de datos XBRL y XML.

Altova ofrece dos ediciones de RaptorXML y todas ellas se pueden descargar e instalar desde la página de descargas del sitio web de Altova:

- **RaptorXML Server** es un motor de procesamiento XML compatible con los estándares XML, XML Schema, XSLT, XPath, XQuery, etc.
- **RaptorXML+XBRL Server** incluye todas las características de RaptorXML Server más funciones de procesamiento y validación de todos los estándares XBRL.

Limitaciones de RaptorXML:

- no admite el uso de firmas XML
- no admite el uso de recursos globales a través de la interfaz de COM
- las conexiones de base de datos ODBC y ADO solamente se pueden usar en Windows. Los demás sistemas operativos se conectan con JDBC automáticamente.

Descarga
Descargue e instale RaptorXML desde la página de descargas del sitio web de Altova.

Licencias
Las ediciones Windows, Linux y Mac OS X de RaptorXML también incluyen el servidor de licencias Altova LicenseServer, necesario para asignar licencias a los productos servidor de Altova.

Ejecutar asignaciones usando RaptorXML Server
Durante la generación de código XSLT 1.0, XSLT 2.0 o XQuery, MapForce genera un archivo por lotes llamado DoTransform.bat en la carpeta de salida elegida por el usuario para la generación de código.

Al ejecutarse, este archivo por lotes llama a RaptorXML y ejecuta la transformación XSLT/XQuery en el servidor.

Si tiene pensado ejecutar o automatizar asignaciones de MapForce en un servidor, consulte la documentación de MapForce Server.

Nota: los modos de vista previa de MapForce (los botones **XSLT**, **XSLT1** y **XQuery** y el botón de **Resultados**) funcionan igual que siempre y usan motores internos.

 © 2014 Altova GmbH

Altova MapForce 2014

Tutorial de MapForce

6 Tutorial de MapForce

Este tutorial le enseñará a realizar varias tareas en MapForce 2014, ofreciéndole una introducción a sus principales funciones.

El objetivo de este tutorial es realizar una asignación de datos entre un informe de gastos de viaje de los empleados de una compañía y un informe más complejo de la compañía. En el ejemplo de nuestro tutorial, cada empleado rellena los campos de su propio informe de gastos. Más tarde los datos de este informe se asignan al informe de la compañía, que se envía al departamento administrativo. Después se pueden añadir más datos y el resultado final es un informe de gastos estandarizado.

En este tutorial aprenderá a:

- Configurar el entorno de asignación
- Asignar los datos del archivo XML de origen (el informe de gastos personal del empleado) al documento de salida (el informe de gastos de la compañía)
- Aplicar filtros a los datos de origen
- Generar un archivo de transformación XSLT
- Transformar los datos de origen en datos de destino usando el archivo XSLT generado

Instalación y configuración
Para completar este tutorial asumimos que tiene instalado MapForce correctamente en su equipo y que recibió un código clave de evaluación gratis (o es usuario registrado del producto). La versión de prueba de MapForce es completamente funcional, pero está limitada a un máximo de 30 días. Puede comprar una licencia normal en nuestra tienda en línea o a uno de nuestros distribuidores.

Archivos de ejemplo del tutorial
El tutorial utiliza estos componentes:

- Esquemas de origen y destino
- Una base de datos MS Access como origen de datos
- Varias funciones (p. ej. **`concat`**, **`filter`** y **`equal`**) y constantes

Todos los archivos utilizados en este tutorial están disponibles en la carpeta `C:\Documents and Settings\All Users\Application Data\Altova`. Cuando se inicia la aplicación por primera vez, los archivos de ejemplo del usuario se copian en la carpeta `...\MapForceExamples\Tutorial`. Por tanto, es importante que no mueva, edite ni elimine los archivos de ejemplo de la carpeta inicial `...\All Users\...`

Los archivos XSLT y los archivos XML transformados también están en esa carpeta. Estos son los archivos que usamos en este tutorial:

Para el informe de gastos personal del empleado:

- `Tut-ExpReport.mfd` La asignación de datos del informe de gastos (con un solo componente de destino)

 © 2014 Altova GmbH

- `Tut-ExpReport-multi.mfd` La asignación de datos del informe de gastos (con varios componentes de destino)
- `PersonDB.mfd` La asignación de datos de empleados, usando una BD MS Access como origen de datos
- `mf-ExpReport.xml` El documento XML de instancia con los gastos personales del empleado
- `mf-ExpReport.xsd` El archivo de esquema asociado

Para el informe de gastos de la compañía

- `ExpReport-Target.xml` El documento de instancia XML con los gastos de la compañía
- `ExpReport-Target.xsd` El archivo de esquema asociado

Rutas de acceso de los archivos en Windows XP, Windows Vista, Windows 7 y Windows 8

Las rutas de acceso de archivo utilizadas en esta documentación no son iguales para todos los sistemas operativos.

- *Carpeta (Mis) Documentos:* la carpeta *Mis documentos* de Windows XP es la carpeta *Documentos* de Windows Vista, Windows 7 y Windows 8. Esta carpeta se encuentra por defecto en las ubicaciones que aparecen a continuación. Los archivos de ejemplo se encuentran por lo general en una subcarpeta de la carpeta *(Mis) Documentos*.

Windows XP	`C:/Documents and Settings/<nombreUsuario>/Mis Documentos`
Windows Vista, Windows 7, Windows 8	`C:/Usuarios/<nombreUsuario>/Documentos`

- *Carpeta de la aplicación:* la carpeta de la aplicación es la carpeta en la que se encuentra su aplicación de Altova. Esta es la ruta de acceso predeterminada de la carpeta de la aplicación:

Windows XP	`C:/Archivos de programa/Altova`
Windows Vista, Windows 7, Windows 8	`C:/Archivos de programa/Altova`
Paquete de 32 bits en sistemas operativos de Windows de 64 bits (XP, Vista, 7)	`C:/Archivos de programa(x86)/Altova`

Nota: MapForce también es compatible con Windows Server 2003, Windows 2008 y Windows Server 2012.

6.1 Configuración del entorno de asignación

Esta sección explica cómo se definen los esquemas de origen y destino que queremos usar para la asignación.

Objetivo

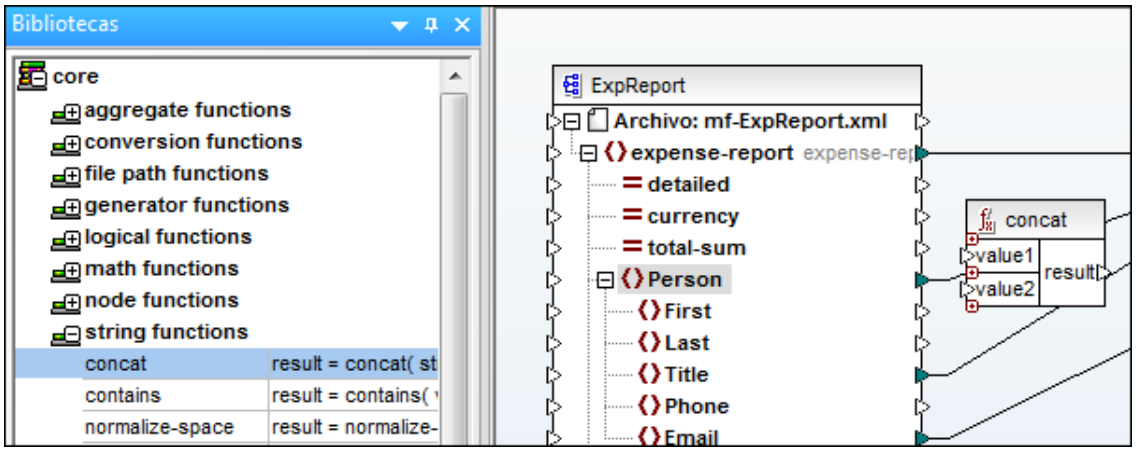
En esta sección del tutorial aprenderá a configurar el entorno de asignación en MapForce. En concreto, aprenderá a:

- Crear los componentes de esquema de origen y destino
- Definir el archivo XML de origen
- Seleccionar el elemento raíz del esquema de destino

Comandos utilizados en esta sección

Insertar archivo o esquema XML: haga clic en este icono para abrir el cuadro de diálogo Abrir de Windows y seleccionar el archivo de su sistema de archivos.

Nota: Al pie de cada imagen de la ventana Bibliotecas aparece una anotación (p. ej. ***Con XSLT seleccionado***). Esta anotación indica el lenguaje de salida/destino seleccionado y que se utiliza al cambiar al panel **Resultados** para obtener una vista previa del resultado de la asignación de datos. El lenguaje de salida/destino seleccionado también determina qué funciones están disponibles en la ventana Bibliotecas.



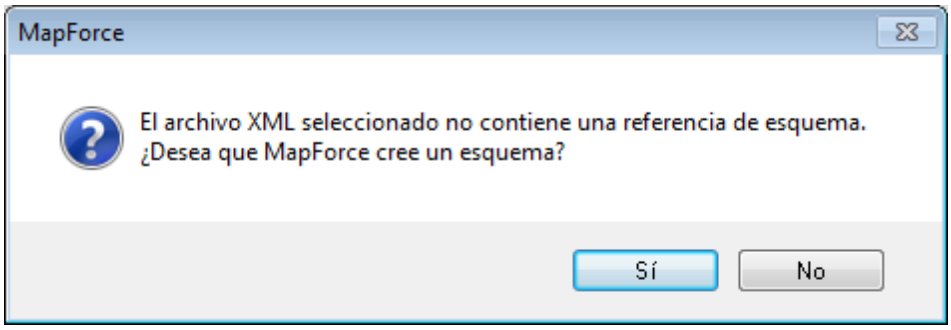
Con XSLT seleccionado

 © 2014 Altova GmbH

6.1.1 Agregar componentes al panel Asignación

Después de iniciar MapForce debe agregar archivos de origen y destino en el panel **Asignación**. Esto también se puede hacer arrastrando archivos desde el explorador de Windows hasta el panel **Asignación**.

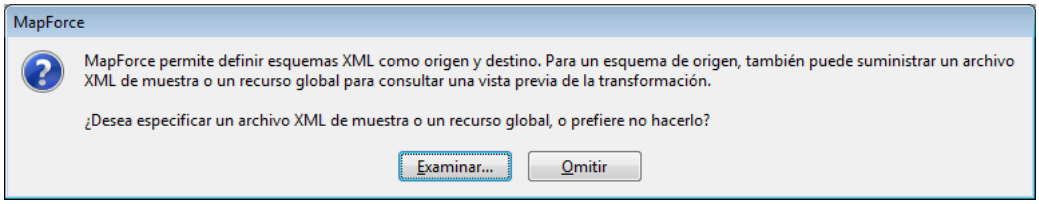
MapForce puede generar automáticamente un esquema XML a partir de un archivo XML ya existente si no está disponible el esquema XML. Si MapForce no encuentra ningún esquema XML asociado al archivo XML insertado con el comando **Insertar archivo o esquema XML**, aparece automáticamente este cuadro de diálogo.



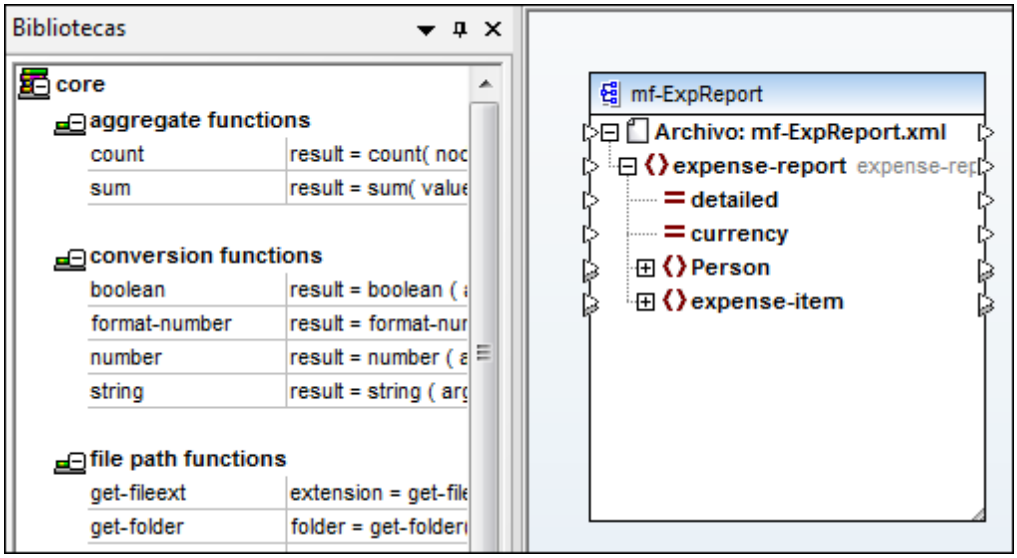
Cuando se genera un esquema a partir de un archivo XML, los tipos de datos de los elementos/atributos deben deducirse del documento XML de instancia y puede que no sean exactamente los tipos deseados. Por eso, recomendamos que revise el esquema generado y compruebe si representa con precisión los datos de instancia.

Para crear el componente de esquema de origen:

1. Haga clic en el icono **Insertar archivo o esquema XML** o seleccione la opción de menú **Insertar | Archivo o esquema XML...**.
2. En el cuadro de diálogo Abrir, navegue hasta la subcarpeta `Tutorial` de la carpeta `...\MapForce2014\MapForceExamples` y seleccione el archivo `mf-ExpReport.xsd`. Aparece un cuadro de diálogo solicitando un archivo XML de muestra para poder generar una vista previa.

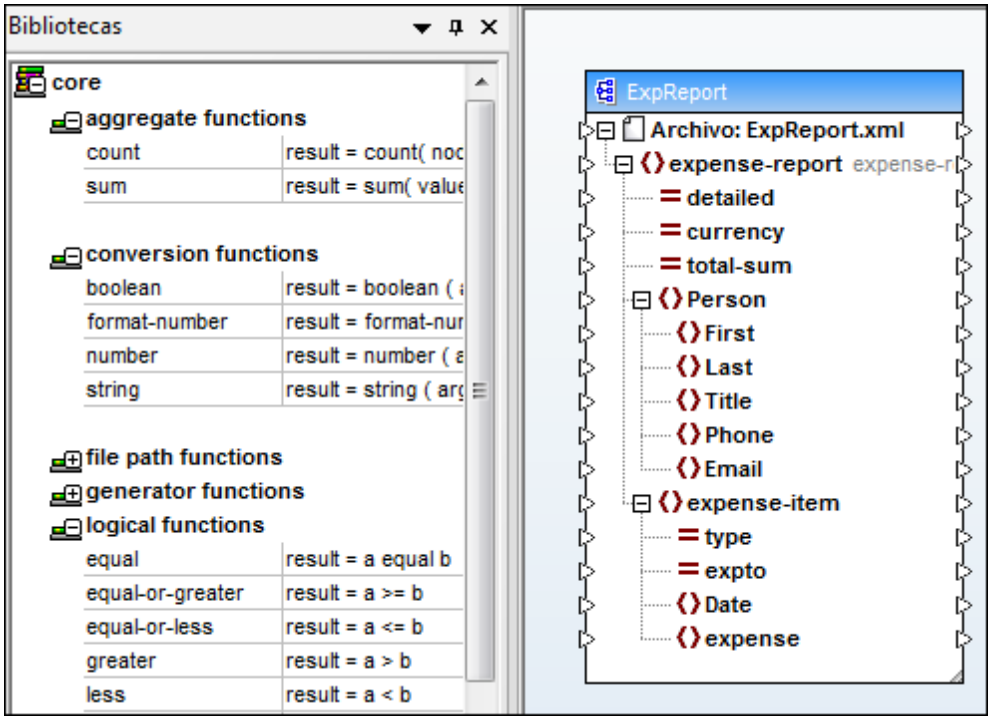


3. Pulse el botón **Examinar...** y seleccione el archivo **`mf-ExpReport.xml`**.
   El componente de esquema de origen aparece ahora en el panel **Asignación**.



Con XSLT seleccionado

4. Haga clic en el elemento **`expense-report`** (del componente) y pulse la tecla ***** del teclado numérico para ver todos los elementos.
5. Haga clic en la esquina inferior derecha del componente y ajuste su tamaño arrastrando el mouse.
   Nota: si hace doble clic en la esquina inferior derecha , el componente se ajusta a un tamaño óptimo que muestra todos sus elementos.



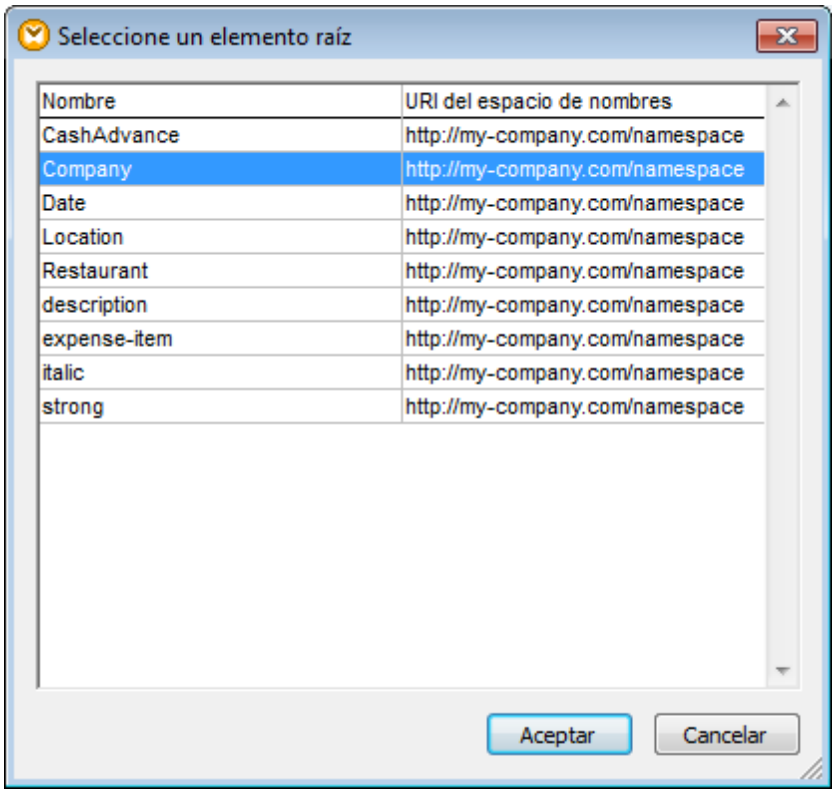
Con XSLT seleccionado

Para crear el componente de esquema de destino:

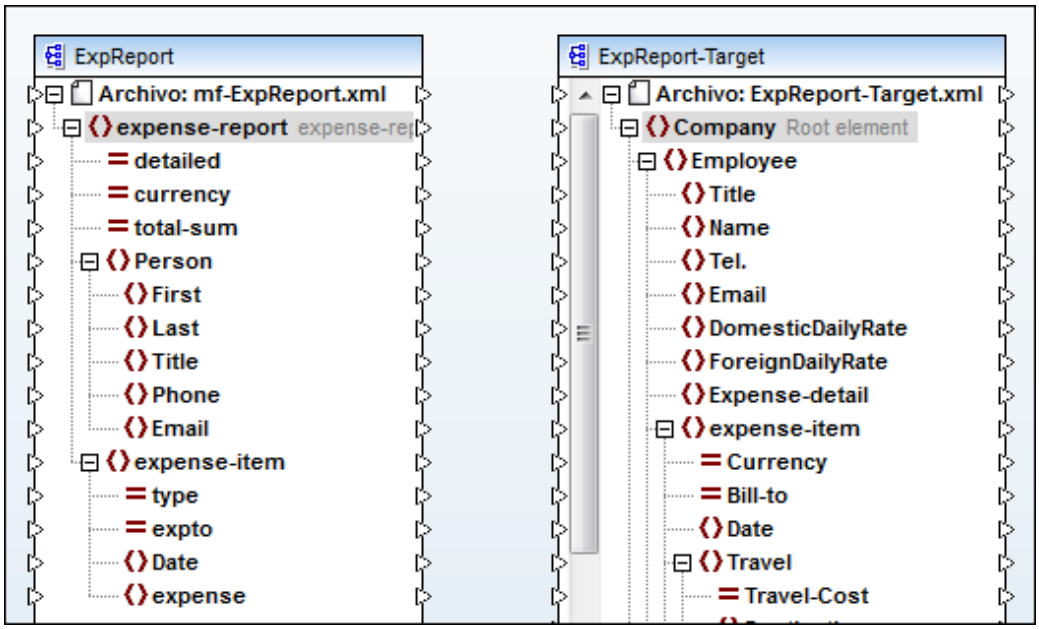
1. Haga clic en el icono **Insertar archivo o esquema XML** o seleccione la opción de menú **Insertar | Archivo o esquema XML...**.

 © 2014 Altova GmbH

2. Seleccione el archivo **`ExpReport-Target.xsd`** en el cuadro de diálogo Abrir.
   Aparece un cuadro de diálogo solicitando un archivo XML de muestra para este esquema.
3. Haga clic en el botón **Omitir** y seleccione **`Company`** como elemento raíz del documento de destino.



   El componente de esquema de destino aparece ahora en el panel **Asignación**.
4. Haga clic en la entrada **`Company`** y pulse la tecla ***** del teclado numérico para ver todos sus elementos.
5. Haga doble clic en la esquina inferior derecha para ajustar el tamaño del componente.



Ya estamos listos para empezar a crear asignaciones de datos entre los elementos del esquema de origen y el de destino.

Nota: al arrastrar los componentes por el panel **Asignación** aparecen líneas de alineación automática que le ayudan a poner el componente en la posición adecuada.

© 2014 Altova GmbH

6.2 Crear una asignación

En la sección anterior, definimos los componentes de origen y destino de la asignación. Ahora empezaremos la asignación de datos propiamente dicha.

Objetivo
Aprender a realizar asignaciones entre componentes de origen y destino y mejorar el resultado de la asignación usando funciones y filtros.

- Usar conectores para asignar elementos de esquema a otros
- Usar una función **`concat`** para combinar elementos de los datos de origen
- Filtrar los datos de origen para pasar solamente ciertos gastos al informe de destino

Comandos utilizados en esta sección

Conectar automáticamente los secundarios equivalentes: haga clic en este icono para activar/desactivar la conexión automática de nodos secundarios equivalentes.

Insertar constante: haga clic en este icono para agregar un componente de constante al panel **Asignación**.

Filtro: nodos/filas: haga clic en este icono para agregar un componente de filtrado al panel **Asignación**.

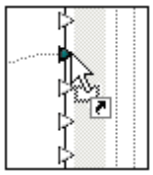
Sitio web de Altova: Integración de datos y asignación de datos XML

6.2.1 Asignaciones entre elementos de un esquema

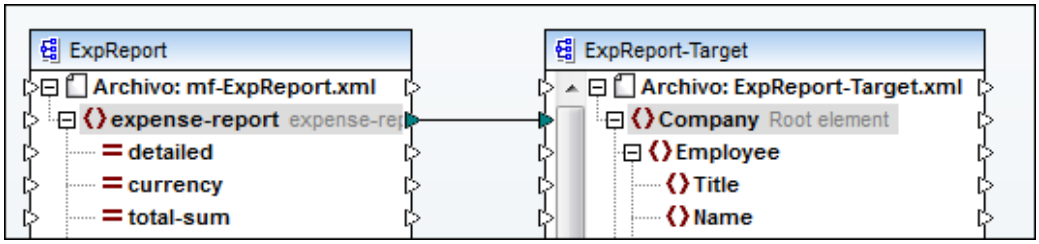
En este apartado aprenderá a definir asignaciones entre los elementos del esquema de origen y los del esquema de destino.

Para realizar asignaciones entre los esquemas `mf-ExpReport` y `ExpReport-Target`:

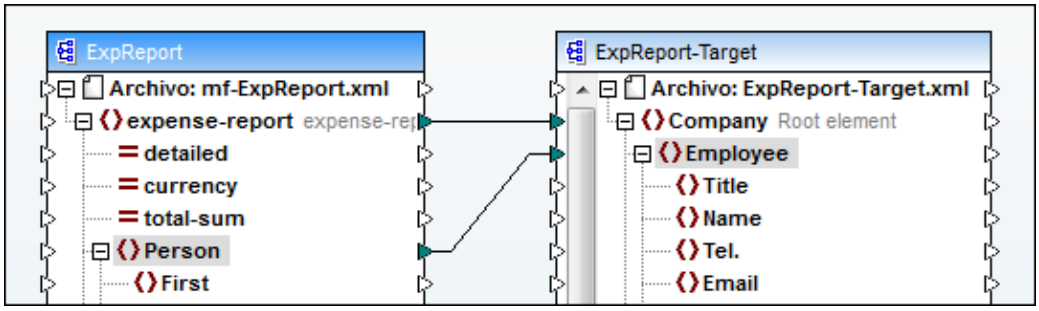
1. Haga clic en el elemento **`expense-report`** del esquema de origen **`mf-ExpReport`** y arrastre el mouse.
   Automáticamente aparece una línea de conexión que surge del icono de salida del elemento y está unida al puntero del mouse, que ahora tiene otra forma.
2. Mueva el puntero del mouse hasta el elemento **`Company`** del esquema de destino **`ExpReport-Target`** y "suelte" el conector en cuanto el puntero del mouse tenga otra vez forma de flecha. Si el conector se puede soltar, aparece un pequeño icono de enlace debajo del puntero del mouse y el icono de entrada y el nombre del elemento del componente de destino aparecen resaltados.



   Se crea un conector entre los esquemas de origen y destino. Esto significa que existe una asignación de datos entre el esquema de origen y el documento de destino.

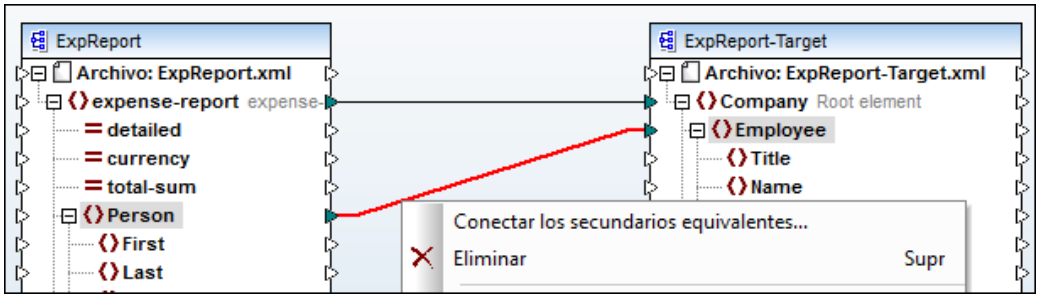


3. Use el método que acabamos de describir para crear una asignación entre los elementos **`Person`** y **`Employee`**.

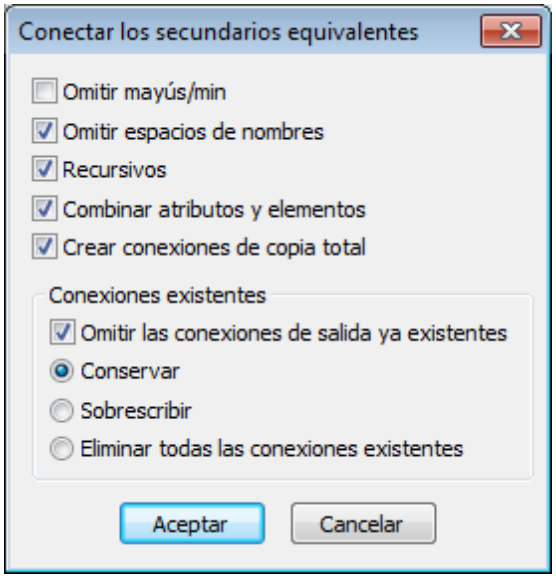


   Si está activo el icono **Conectar automáticamente los secundarios equivalentes** , entonces los elementos **`Title`** y **`Email`** se conectan automáticamente. Si no es así:
4. Haga clic con el botón secundario en el conector de **`Person`** y seleccione la opción **Conectar los secundarios equivalentes** en el menú contextual.

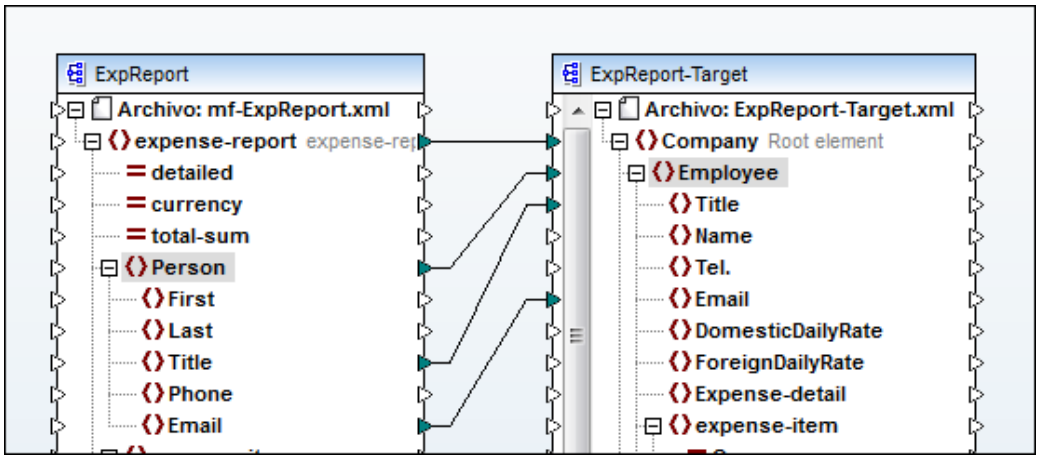
 © 2014 Altova GmbH



Aparece el cuadro de diálogo "Conectar los secundarios equivalentes".



5. Marque las opciones que aparecen en la imagen anterior y haga clic en **Aceptar**. Para más información consulte el apartado Propiedades del conector.



Automáticamente se crean asignaciones de datos entre los elementos **`Title`** y **`Email`** de ambos esquemas.

6. Cambie al panel **Resultados** para ver el resultado de la asignación.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xmlns="http://my-company.com/namespace" xmlns:xsi="htt
3    <Employee>
4      <Title>Project Manager</Title>
5      <Email>f.landis@nanonull.com</Email>
6    </Employee>
7  </Company>
8
```

Observe que los campos **`Title`** y **`Email`** contienen datos del documento XML de instancia.

7. Vuelva al panel **Asignación** para seguir trabajando en la asignación de datos.

Nota: las opciones seleccionadas en el cuadro de diálogo "Conectar los secundarios equivalentes" se conservan hasta que las cambie. Estas opciones se aplican desde el menú contextual o haciendo clic en el icono **Conectar automáticamente los secundarios equivalente**, que sirve para activar/desactivar esta opción.

© 2014 Altova GmbH

6.2.2 Uso de las funciones en las asignaciones de datos

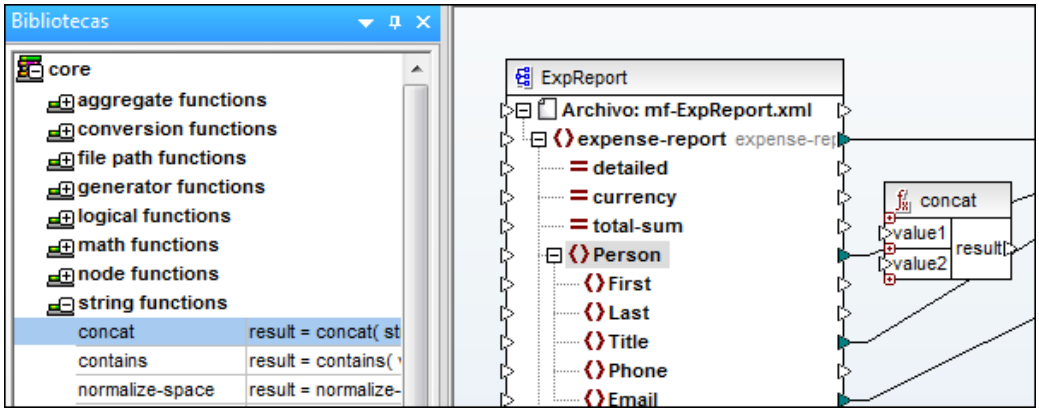
El objetivo de este apartado es combinar dos conjuntos de datos del esquema de origen y colocar el resultado en un solo elemento del documento de destino. Para ello será necesario:

- Usar la función **`concat`** para combinar los elementos **`First`** y **`Last`** del esquema de origen
- Usar la función **`constant`** para suministrar el carácter de espacio necesario para separar ambos elementos
- Colocar el resultado de este proceso en el elemento **`Name`** del esquema de destino

Tenga en cuenta que, por motivos de claridad, algunas de las asignaciones definidas en el paso anterior no aparecen en las imágenes siguientes.

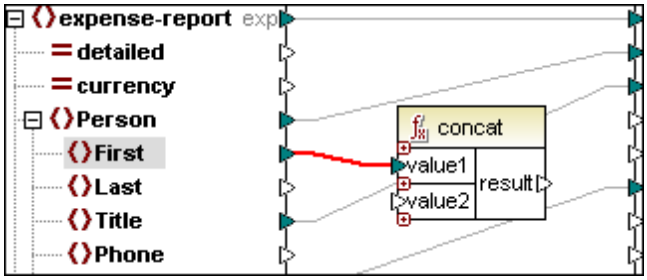
Para combinar elementos usando funciones:

1. En la ventana Bibliotecas, expanda el grupo **string functions** de la biblioteca **core**, haga clic en la entrada **`concat`** y arrástrela al panel **Asignación**.

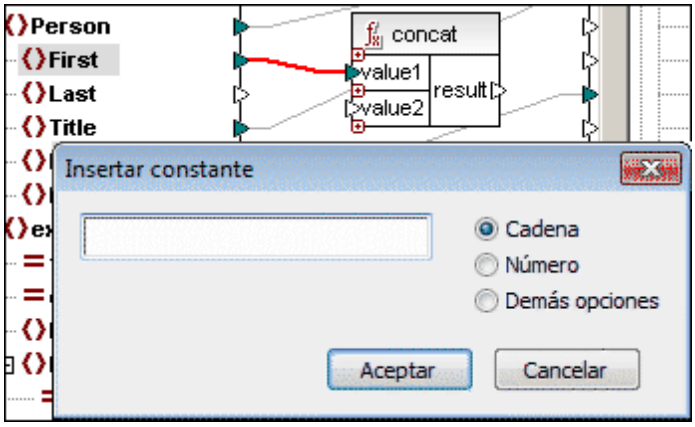


Con XSLT seleccionado

2. En el componente **`mf-ExpReport`** seleccione el elemento **`First`** y cree una conexión (arrastrando el puntero mientras mantiene pulsado el botón del mouse) con la entrada **`value1`** del componente **`concat`**.

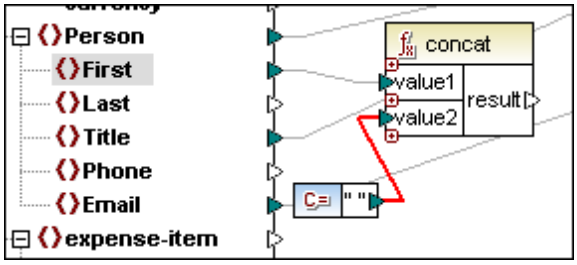


3. Haga clic con el botón secundario en el fondo del panel, cerca de la entrada **`value2`** y seleccione la opción **Insertar constante...** del menú contextual para insertar un componente de constante.

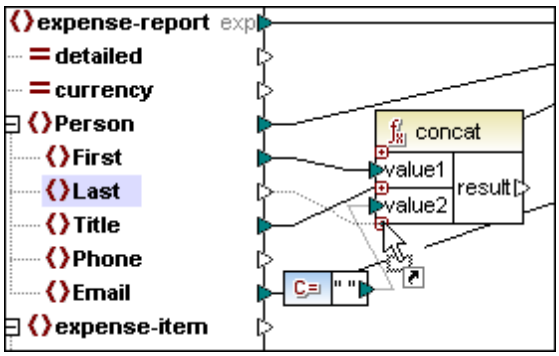


4. Introduzca un carácter de espacio en el cuadro de texto y haga clic en **Aceptar**. Ahora aparece el componente de constante en el área de trabajo. Su contenido aparece junto al icono de salida.

5. Cree una conexión entre el componente de constante y la entrada **`value2`** del componente **`concat`**.

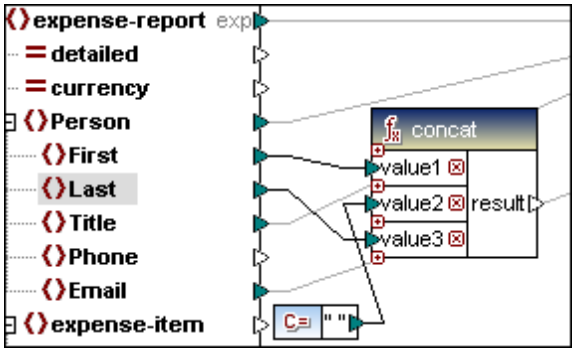


6. En el componente **`mf-ExpReport`** haga clic en el elemento **`Last`** y arrastre una línea de conexión hasta el icono "**+**" de la función **`concat`**, justo debajo de la entrada **`value2`**. El puntero del mouse cambia de forma, indicando cuándo puede "soltar" el conector.

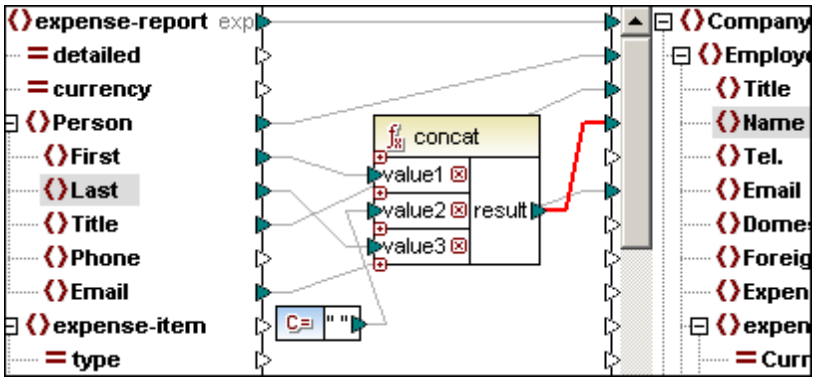


Automáticamente se amplía la función **`concat`** con un elemento más (es decir, un valor más), al que se conecta el elemento **`Last`**.

 © 2014 Altova GmbH



7. Conecte el icono del valor **`result`** del componente **`concat`** con el elemento **`Name`** del esquema de destino.



8. Cambie al panel **Resultados** para ver el resultado de la asignación actual.

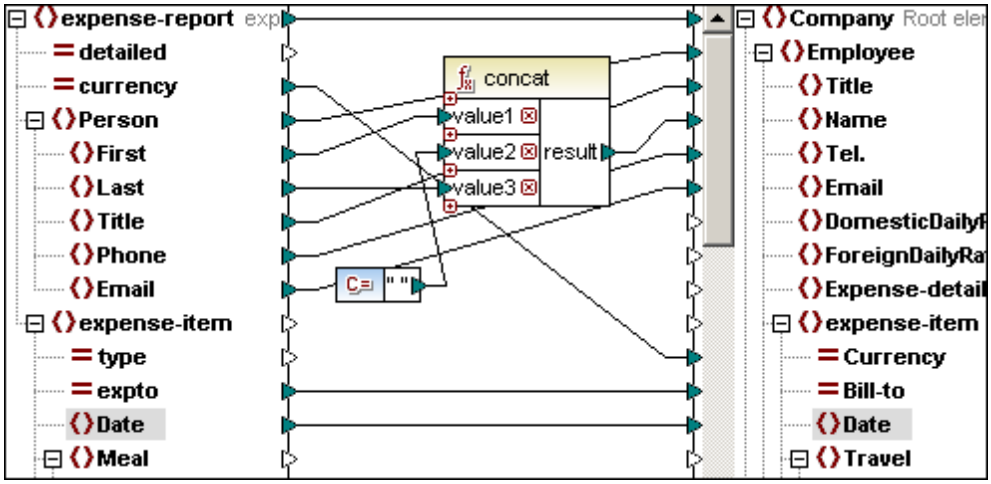
```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     <Employee>
4       <Title>Project Manager</Title>
5       <Name>Fred Landis</Name>
6       <Email>f.landis@nanonull.com</Email>
7     </Employee>
8   </Company>
9
```

Observe que el nombre del elemento **`Person`** (*Fred Landis*) ahora aparece con la etiqueta **`Name`**. El nombre (**`First`**) y el apellido (**`Last`**) se separaron con un carácter de espacio.

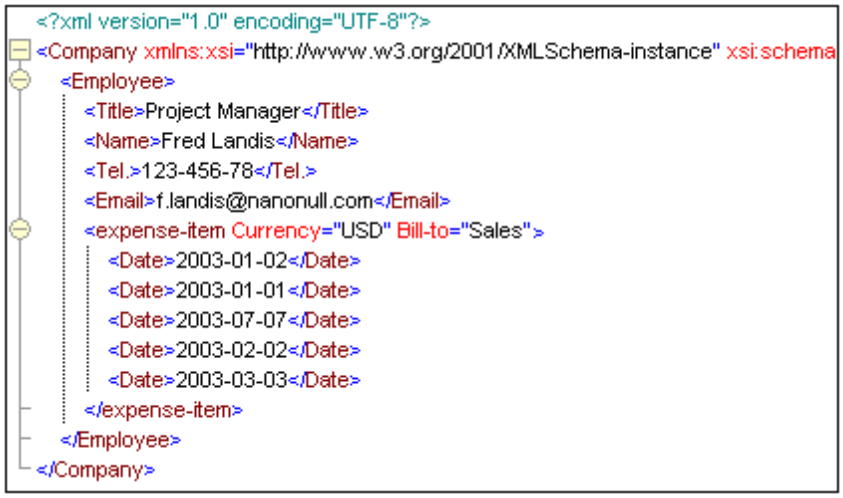
Asignación de los demás datos personales

Ahora cree asignaciones entre estos elementos:

- Entre **`currency`** y **`Currency`**
- Entre **`Phone`** y **`Tel.`**
- Entre **`expto`** y **`Bill-to`**
- Entre **`Date`** y **`Date`**



Cuando termine de crear estas asignaciones, cambie al panel **Resultados** para ver el resultado.



Ahora tenemos cinco elementos que proceden del archivo XML de instancia asociado.

Nota: puede reunir varias funciones en una sola función definida por el usuario para optimizar el uso de la pantalla. Para ver un ejemplo de cómo unir la función **`concat`** y la constante en una sola función definida por el usuario, consulte el apartado Funciones definidas por el usuario.

 © 2014 Altova GmbH

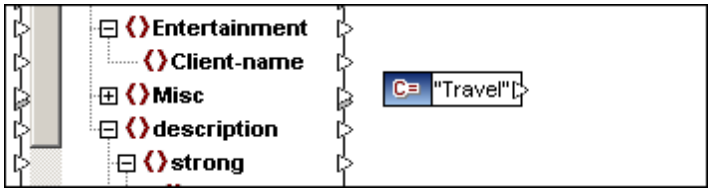
6.2.3 Filtrado de datos

El objetivo de este apartado es aplicar un filtro para eliminar los gastos **`Lodging`** y **`Meal`** y pasar solamente los gastos **`Travel`** al documento/esquema de destino. Para ello será necesario:

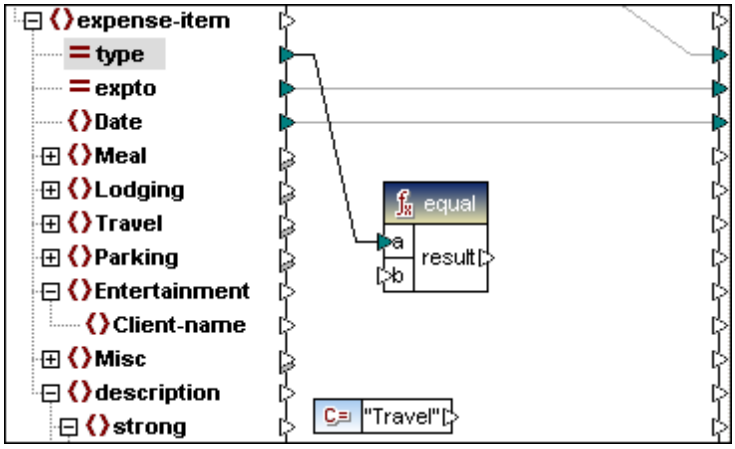
- Usar la función **`equal`** para probar el valor de un elemento de origen
- Usar una **constante** para suministrar la cadena de comparación que debe probarse
- Usar el **componente de filtrado** que pasa los datos de **`Travel`**, si el valor de entrada **`bool`** da **`true`** como resultado
- Colocar el resultado de este proceso en el elemento **`expense-item`** documento/esquema de destino.

Para filtrar los datos:

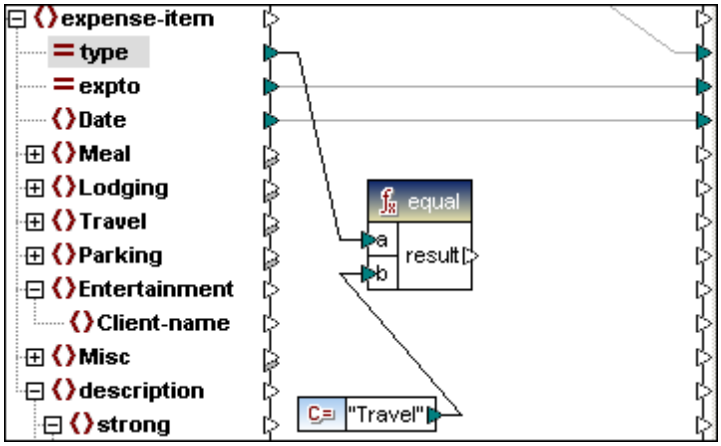
1. Haga clic en el icono **Insertar constante** e introduzca la cadena **`Travel`** en el campo de entrada.



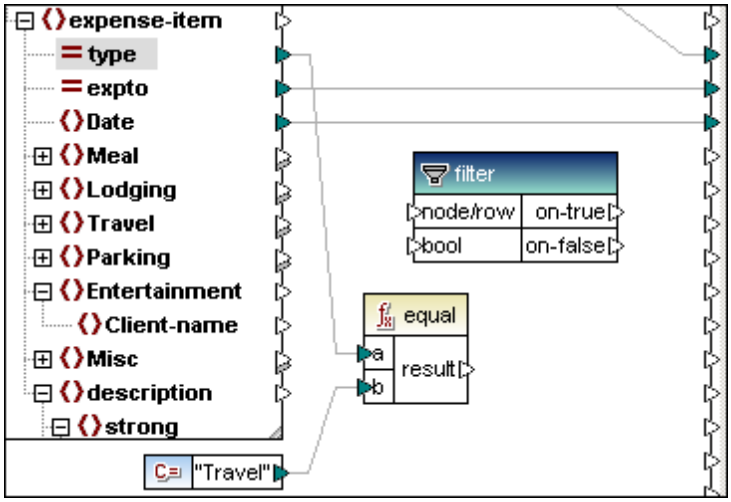
2. En la ventana Bibliotecas, expanda el grupo **logical functions** de la biblioteca **core** y arrastre la función lógica **`equal`** hasta el panel **Asignación**.
3. Conecte el elemento **`type`** (de **`expense-item`**) del esquema de origen con el parámetro **`a`** de la función **`equal`**.



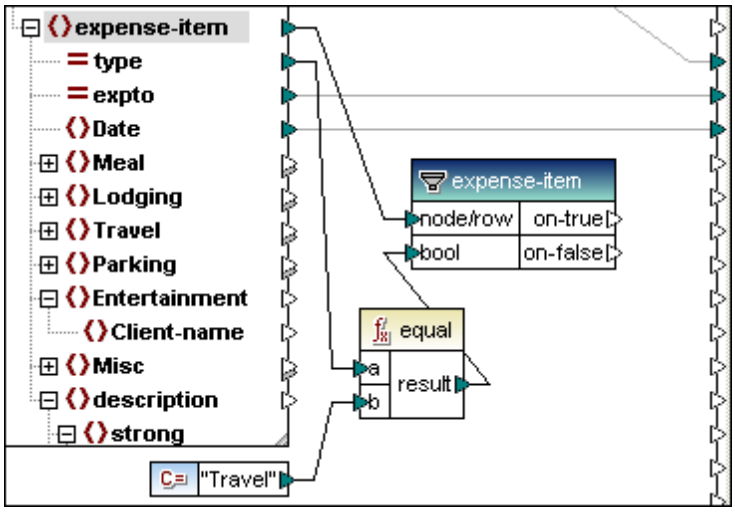
4. Conecte el icono de salida de la **constante** "Travel" con el parámetro **`b`** de la función **`equal`**.



5. Seleccione la opción de menú **Insertar | Filtro: nodos/filas**.



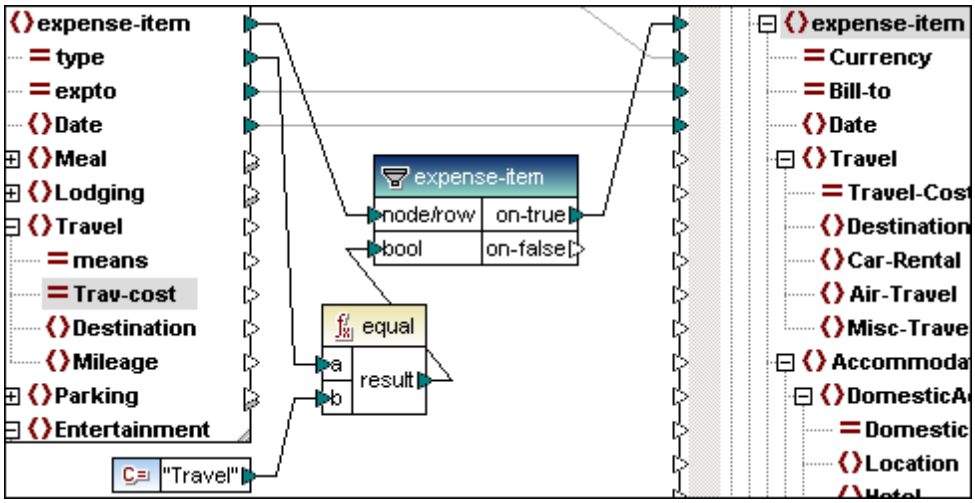
6. Conecte el icono de salida **`result`** de la función **`equal`** con el parámetro **`bool`** del componente **`filter`**.
7. Conecte el elemento **`expense-item`** del esquema de origen con el parámetro **`node/row`** del componente **`filter`**.



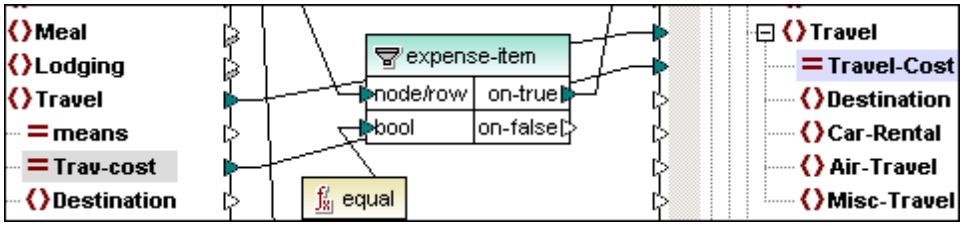
Observe cómo el nombre del componente de filtrado ahora pasa a ser **`expense-item`**.

 © 2014 Altova GmbH

8. Conecte el icono **`on-true`** del componente de **filtrado** con el elemento **`expense-item`** del documento de destino.



9. Conecte el elemento **`Travel`** del esquema de origen con el elemento **`Travel`** del esquema de destino.
10. Conecte el elemento **`Trav-cost`** con el elemento **`Travel-Cost`** del esquema de destino.



11. Cambie al 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:schema
3     <Employee>
4       <Title>Project Manager</Title>
5       <Name>Fred Landis</Name>
6       <Tel.>123-456-78</Tel.>
7       <Email>f.landis@nanonull.com</Email>
8       <expense-item Currency="USD" Bill-to="Development">
9         <Date>2003-01-02</Date>
10        <Travel Travel-Cost="337.88"/>
11      </expense-item>
12      <expense-item Currency="USD" Bill-to="Accounting">
13        <Date>2003-07-07</Date>
14        <Travel Travel-Cost="1014.22"/>
15      </expense-item>
16      <expense-item Currency="USD" Bill-to="Marketing">
17        <Date>2003-02-02</Date>
18        <Travel Travel-Cost="2000"/>
19      </expense-item>
20    </Employee>
21  </Company>
22
```

Nota: el parámetro **`on-false`** del componente de filtrado da como resultado el juego de nodos asignado por el parámetro **`on-true`**. En este ejemplo, serían todos los elementos de gasto que no sean gastos de viaje (es decir, que no sean de tipo **`travel`**).

El número de elementos de gasto se ha reducido a tres. Si compara los resultados con el archivo **`mf-ExpReport.xml`**, observará que solamente se conservan los registros **`Travel`** y que los registros **`Lodging`** y **`Meal`** se eliminaron aplicando el filtro.

© 2014 Altova GmbH

6.3 Generación de código XSLT 1.0 y 2.0

Una vez creada la asignación de datos puede pasar a transformar los datos de origen. MapForce puede generar varias versiones de código XSLT: 1.0 y 2.0.

Objetivo

En esta sección del tutorial aprenderá a obtener una vista previa, generar y guardar el código XSLT y a ejecutar el código XSLT generado. En concreto aprenderá a:

- Generar y guardar el código XSLT en la versión que prefiera
- Ejecutar el archivo por lotes de transformación

Comandos utilizados en esta sección

Archivo | Generar código en: seleccione esta opción para elegir el lenguaje de salida (es decir,. **XSLT 1.0** o **XSLT 2.0**). Al hacer clic en este comando, se abre el cuadro de diálogo "Buscar carpeta" donde puede seleccionar la carpeta donde desea guardar el código XSLT generado.

Para generar código XSLT:

1. Seleccione el comando de menú **Archivo | Generar código en | XSLT 1.0** o **XSLT 2.0**.
2. Elija en qué carpeta desea guardar el archivo XSLT generado y haga clic en **Aceptar**. Aparece un mensaje confirmando que el código se generó correctamente.
3. Navegue hasta la carpeta indicada y encontrará el archivo XSLT con el nombre `MappingMapToExpReport-Target.xslt` (es decir, el nombre será: `MappingMapTo<NombreEsquemaDestino>`).

Para transformar el informe personal de gastos en un informe de gastos de la compañía:

1. Descargue e instale RaptorXML de la página de descargas del sitio web de Altova.
2. Inicie el archivo por lotes `DoTransform.bat` ubicado en la carpeta donde se guardó el código.

Como resultado se genera el archivo de salida `ExpReport-Target.xml` en la carpeta `...\Tutorial`.

Tenga en cuenta que quizás sea necesario añadir la ubicación de la instalación de RaptorXML Server a la variable PATH de las variables de entorno. La documentación de RaptorXML está disponible en el sitio web de Altova.

6.4 Trabajar con varios esquemas y documentos de destino

En esta sección explicamos cómo crear otro esquema/documento de destino en la asignación de datos del tutorial (`Tut-ExpReport.mfd`). En este otro esquema colocaremos los registros que no sean gastos de viaje.

Objetivo
En esta sección del tutorial aprenderá a añadir otro esquema de destino y a generar resultados para varios esquemas de destino. En concreto aprenderá a:

- Crear otro componente de esquema de destino
- Filtrar los registros del informe que no sean gastos de viaje
- Definir varios esquemas de destino con el mismo nombre
- Ver los resultados de cada componente de destino
- Generar código XSLT para varios esquemas de destino
- Generar código de programa para varios esquemas de destino

Comandos utilizados en esta sección

Insertar archivo o esquema XML: haga clic en este icono para abrir el cuadro de diálogo Abrir de Windows y seleccionar el archivo de su sistema de archivos. **Propiedades**: a las que puede tener acceso desde el menú **Componente**. En los esquemas de origen y destino este comando sirve para definir, entre otras propiedades, el archivo de esquema, el archivo XML de entrada y el archivo XML de salida del componente.

Vista previa: aparece en la barra de título de los componentes cuando se definen varios archivos de destino. Haga clic en este icono para elegir el componente cuya vista previa desea obtener.

Guardar el archivo de salida: situado en el menú/panel **Resultados**. Haga clic en este icono para abrir el cuadro de diálogo "Guardar como" y elegir dónde se guardan los resultados generados.

Validar archivo de salida: situado en el menú/panel **Resultados**. Haga clic en este icono para comprobar si los resultados generados son válidos. El resultado de la validación aparece en la ventana Mensajes.

Archivo | Generar código en: seleccione esta opción para elegir el lenguaje de salida (es decir,. **XSLT 1.0** o **XSLT 2.0**). Al hacer clic en este comando, se abre el cuadro de diálogo "Buscar carpeta" donde puede seleccionar la carpeta donde desea guardar el código XSLT generado.

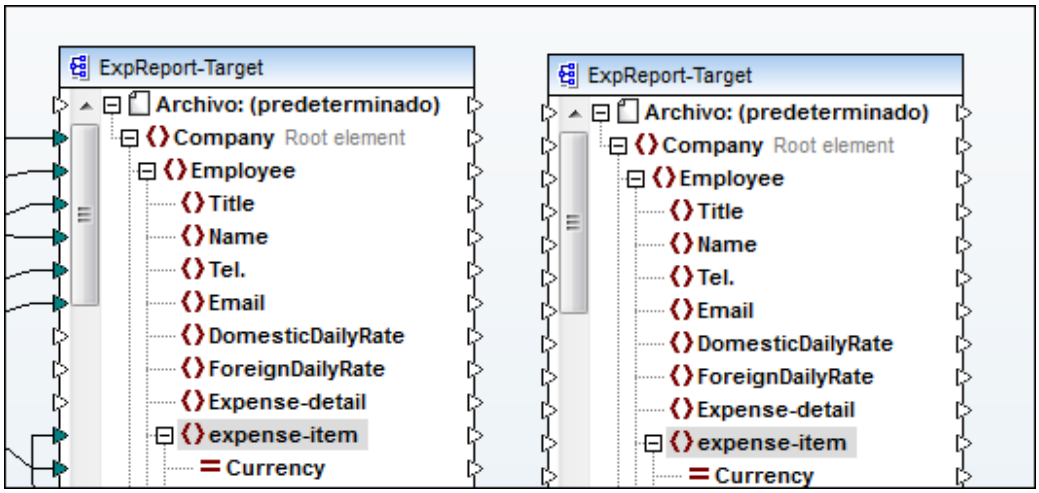
 © 2014 Altova GmbH

6.4.1 Crear un segundo componente de destino

En este apartado aprenderá a crear otro componente de esquema de destino que filtra los datos que son de tipo **`travel`** (es decir, solamente gastos de viaje).

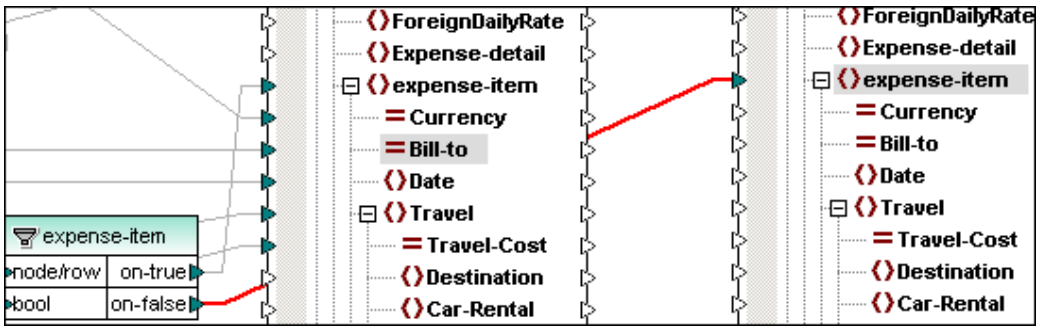
Para crear el segundo componente de esquema de destino:

1. Haga clic en el icono **Insertar archivo o esquema XML**.
2. Seleccione el archivo `ExpReport-Target.xsd` en el cuadro de diálogo "Abrir".
   Aparece un mensaje solicitando un archivo XML de muestra para este esquema.
3. Haga clic en **Omitir** y seleccione **`Company`** como elemento raíz del documento de destino.
   El componente de esquema de destino aparece ahora en el panel **Asignación**.
4. Haga clic en la entrada **`Company`** y pulse la tecla ***** del teclado numérico para ver todos sus elementos.
5. Haga clic en el icono de la esquina superior derecha del componente y ajuste su tamaño. Después coloque los componentes de esquema de tal forma que pueda verlos bien y trabajar con ellos fácilmente.
   En el panel **Asignación** ahora tenemos un esquema de origen (**`mf-expReport`**) y dos esquemas de destino (llamados **`ExpReport-Target`**).

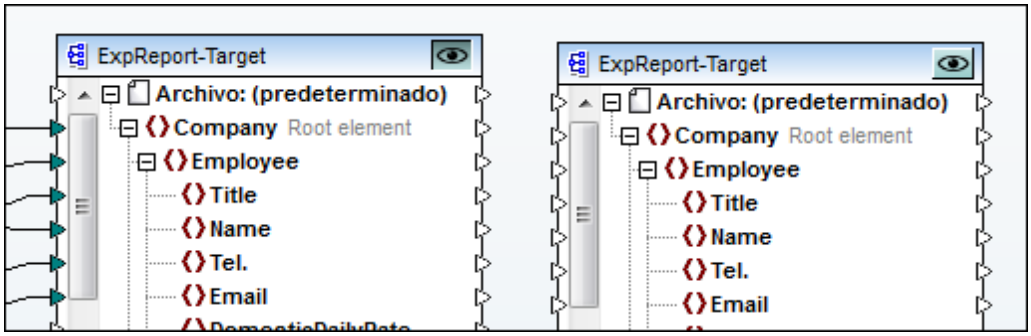


Para filtrar los datos que no son gastos de viaje y eliminarlos:

1. Conecte el icono **`on-false`** del componente de filtrado con el elemento **`expense-item`** del nuevo esquema de destino.



   Aparece un mensaje que confirma que ahora está trabajando con varios esquemas/documentos de destino.
2. Haga clic en **Aceptar** para confirmar.



Ahora aparece un icono **Vista previa** en la barra de título de los componentes de esquema de destino.

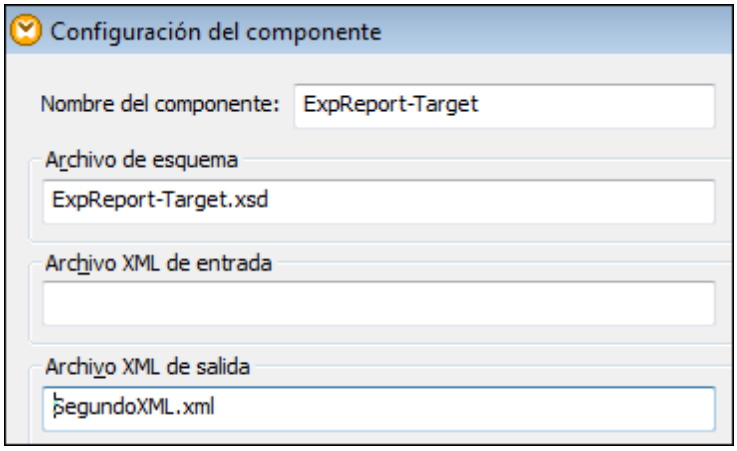
Al hacer clic en este icono se define qué esquema de destino se debe representar cuando se cambia al panel **XSLT**, **XSLT2**, **XQuery**, o **Resultados**.

Definir varios esquemas de destino con el mismo nombre para la generación de código

En este ejemplo los dos esquemas de destino tienen el mismo nombre. Por eso tenemos que asegurarnos de que al generar el código no sobrescribimos el primer archivo de salida con el segundo archivo de salida. Si genera código XSLT no hace falta preocuparse por esto.

Para definir el archivo XML de salida:

1. Haga clic con el botón secundario en el **segundo** esquema/documento de destino y seleccione **Propiedades** en el menú contextual.
2. Escriba un nombre de archivo en el campo *Archivo XML de salida*. Por ejemplo `...\MapForceExamples\Tutorial\SegundoXML.xml`.



3. Haga clic en **Aceptar** para cerrar el cuadro de diálogo "Configuración del componente". Recomendamos que inserte la ruta de acceso absoluta si desea generar código. Para este ejemplo usamos la ruta de acceso de instalación predeterminada de MapForce. Observe que ahora el nombre del archivo de salida que aparece en el componente es `SegundoXML.xml`.

Crear asignaciones para el resto de los datos del informe de gastos

Ahora cree las asignaciones que aparecen más abajo entre el **esquema de origen** y el **segundo esquema de destino**. Se trata de las mismas conexiones que creó entre el esquema

de origen y el primer esquema de destino:

- Entre **Person** y **Employee**
- Entre **Title** y **Title**
- Entre **Phone** y **Tel.**
- Entre **Email** y **Email**
- Entre **currency** y **Currency**
- Entre **expto** y **Bill-to**
- Entre **Date** y **Date**

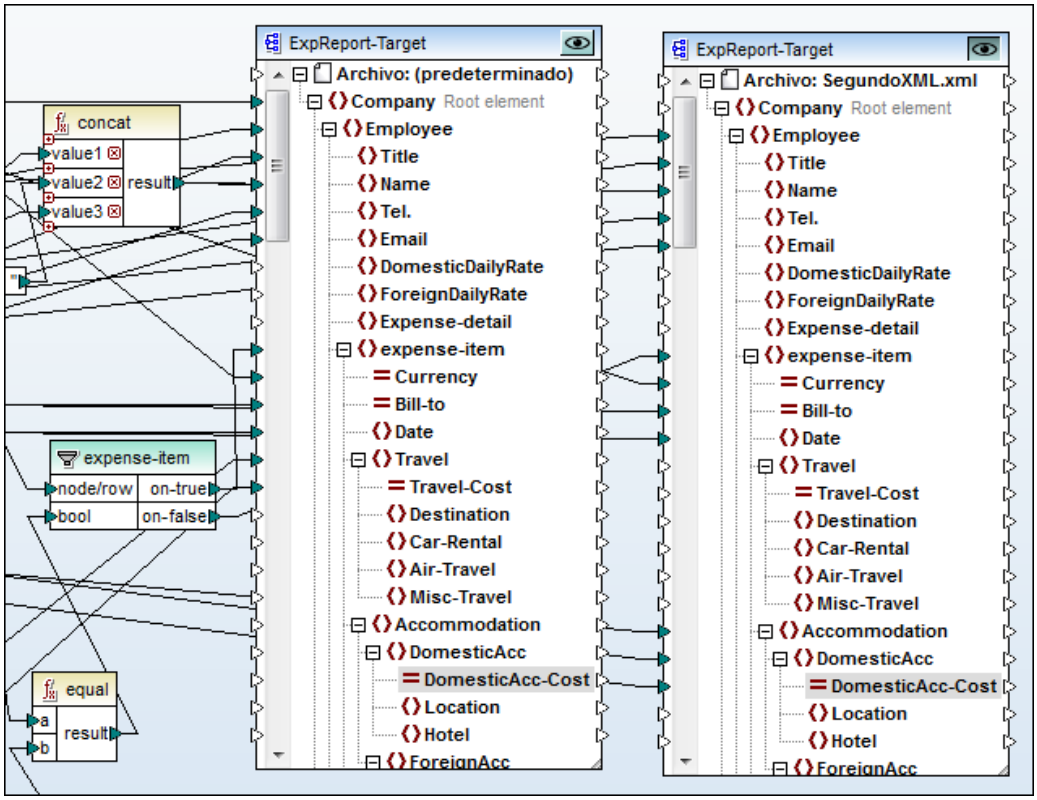
Después cree esta asignación entre la función **concat** y el **segundo esquema de destino**:

- Entre **result** y **Name**

Para crear asignaciones entre el resto de gastos que no son gastos de viaje:

Primero asegúrese de que la opción **Conectar los secundarios equivalentes** no está activada y después

1. Conecte el elemento **Lodging** del esquema de origen con el elemento **Accommodation** del segundo esquema de destino.
2. Conecte el elemento **Lodging** del esquema de origen con **DomesticAcc**.
3. Conecte el elemento **Lodge-Cost** del esquema de origen con **DomesticAcc-Cost**

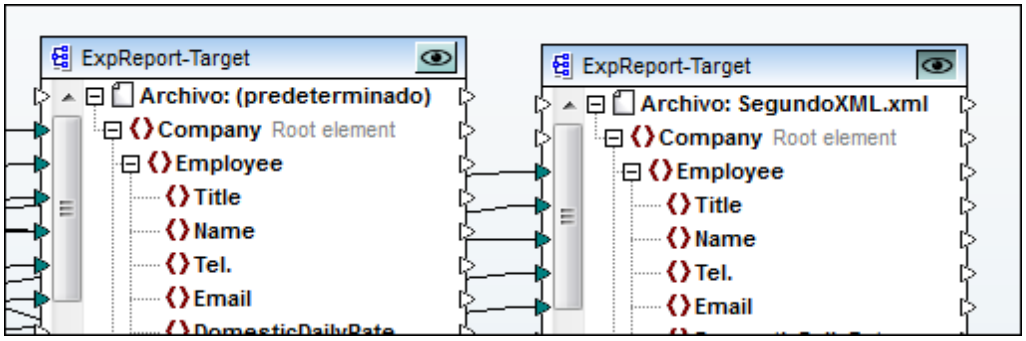


6.4.2 Ver y generar el resultado de varios esquemas de destino

Si hace clic en el icono **Vista previa** puede elegir para cuál de los dos esquemas de destino se genera la vista previa.

Para ver el resultado XSLT de uno de los esquemas de destino:

1. Haga clic en el icono **Vista previa** de la barra de título del segundo componente de esquema.



2. Cambie al panel **Resultados**.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Company xmlns="http://my-company.com/namespace" xmlns:xsi="http://www.w:
3     <Employee>
4       <Title>Project Manager</Title>
5       <Name>Fred Landis</Name>
6       <Tel.>123-456-78</Tel.>
7       <Email>f.landis@nanonull.com</Email>
8       <expense-item Currency="USD" Bill-to="Sales">
9         <Date>2003-01-01</Date>
10        <Accommodation>
11          <DomesticAcc DomesticAcc-Cost="121.2"/>
12        </Accommodation>
13      </expense-item>
14      <expense-item Currency="USD" Bill-to="Sales">
15        <Date>2003-03-03</Date>
16      </expense-item>
17    </Employee>
18  </Company>
19
```

Observe que el resultado XML contiene dos registros que se pasarán al departamento de ventas (con el atributo Bill-to="Sales"): un gasto por valor de 121.2 USD cuyo concepto es DomesticAcc y otro que solo incluye la fecha. Este registro procede del elemento `Meal`. Por ahora no hay ninguna asignación entre los gastos de tipo **`Meal`** y los gastos de tipo **`DomesticAcc`**, pero si la hubiera, el gasto no aparecería en los resultados porque la instancia XML no incluye la cuantía del gasto.

Nota: puede guardar estos datos XML haciendo clic en el icono **Guardar resultado generado** de la barra de herramientas .

El archivo XML de instancia resultante también se puede validar con el esquema de destino, haciendo clic en el icono **Validar archivo de salida** .

 © 2014 Altova GmbH

Para generar código XSLT 1.0 / XSLT 2.0 para varios esquemas de destino:

1. Seleccione el comando **Archivo | Generar código en | XSLT 1.0 (o XSLT 2.0)**.
2. Elija la carpeta donde desea guardar los archivos XSLT generados y haga clic en **Aceptar**.
   Aparece un mensaje confirmando que los archivos se generaron correctamente.
3. Navegue hasta la carpeta correspondiente y encontrará dos archivos XSLT llamados `MappingExpReport-Target.xslt` y `MappingExpReport-Target2.xslt`.

En la carpeta carpeta donde se guardaron los archivos XSLT también hay un archivo por lotes llamado `DoTransform.bat`, que utiliza RaptorXML para transformar el archivo XML.

Para transformar el informe personal de gastos en un informe de gastos de la compañía:

1. Descargue e instale RaptorXML de la página de descargas del sitio web de Altova.
2. Inicie el archivo por lotes `DoTransform.bat` ubicado en la carpeta donde se guardó el código.

Como resultado se genera el archivo de salida `ExpReport-Target.xml` en la carpeta `...\Tutorial`.

Tenga en cuenta que quizás sea necesario añadir la ubicación de la instalación de RaptorXML Server a la variable PATH de las variables de entorno. La documentación de RaptorXML está disponible en el sitio web de Altova.

Para generar código de programa para varios esquemas de destino:

1. Seleccione el comando **Archivo | Generar código en | XQuery, Java, C# o C++**.
2. Elija la carpeta donde desea guardar los archivos generados y haga clic en **Aceptar**.
   Aparece un mensaje confirmando que los archivos se generaron correctamente.
3. Navegue hasta la carpeta correspondiente y compile el proyecto.
4. Compile y ejecute el código de programa con su compilador.
   La aplicación genera dos archivos XML.

Nota: MapForce genera un archivo de proyecto **JBuilder** y scripts **Ant** para ayudarle a compilar el código Java. Para más información consulte el apartado sobre conexiones JDBC.

6.5 Asignar varios componentes de origen a uno de destino

En esta sección crearemos asignaciones entre dos sencillos informes personales de gastos de viaje y un solo informe de la compañía. Este ejemplo es una versión simplificada de la asignación en la que trabajamos en la sección anterior.

Nota: otra manera de hacer esto consiste en usar archivos de entrada/salida dinámicos. Para ver un ejemplo concreto, consulte el apartado Nombres de archivo dinámicos: entrada/salida.

Objetivo
En esta sección del tutorial aprenderá a combinar **dos informes personales de gastos de viajes** en un solo informe de gastos de viaje de la compañía. En concreto aprenderá a:

- Realizar asignaciones entre componentes de los esquemas (repaso de las secciones anteriores)
- Duplicar elementos de entrada
- Quitar elementos duplicados

Comandos utilizados en esta sección

Nuevo...: haga clic en este icono para abrir el cuadro de diálogo "Archivo nuevo" donde puede crear un archivo de asignación nuevo.

Insertar archivo o esquema XML: haga clic en este icono para abrir el cuadro de diálogo Abrir de Windows y seleccionar el archivo de su sistema de archivos.

Conectar automáticamente los secundarios equivalentes: haga clic en este icono para activar/desactivar la conexión automática de nodos secundarios equivalentes.

Duplicar la entrada: comando del menú contextual que aparece al hacer clic con el botón secundario en un elemento de un componente. Haga clic en este comando para duplicar el elemento seleccionado.

Quitar el duplicado: comando del menú contextual que aparece al hacer clic con el botón secundario en un elemento duplicado de un componente. Haga clic en este comando para quitar el duplicado seleccionado del componente.

Archivos de ejemplo utilizados en esta sección
Recuerde que los archivos de ejemplo utilizados en esta sección se optimizaron para mostrar cómo se asignan datos de dos archivos XML de entrada a un solo elemento del esquema de destino y no pretenden ser ejemplos realistas.

- `mf-ExpReport.xml` Archivo XML de entrada utilizado en la sección anterior
- `mf-ExpReport2.xml` El segundo archivo XML de entrada
- `mf-ExpReport-combined.xml` El archivo resultante cuando la asignación se realiza correctamente
- `ExpReport-combined.xsd` El archivo de esquema de destino en el que se combinarán los dos archivos XML de entrada
- `Tut-ExpReport-msource.mfd` El archivo de asignación de este ejemplo

Nota: los archivos utilizados en esta sección se pueden encontrar en la carpeta . . .

 © 2014 Altova GmbH

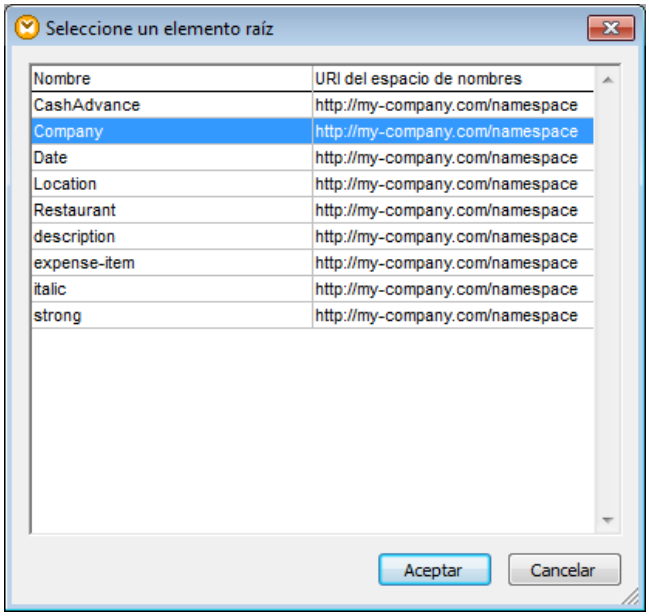
\MapForceExamples\Tutorial.

6.5.1 Crear la asignación

A continuación repasamos todo lo aprendido en las secciones anteriores. Esta asignación es el archivo `Tut-ExpReport-msource.mfd` disponible en la carpeta `...\MapForceExamples\Tutorial`.

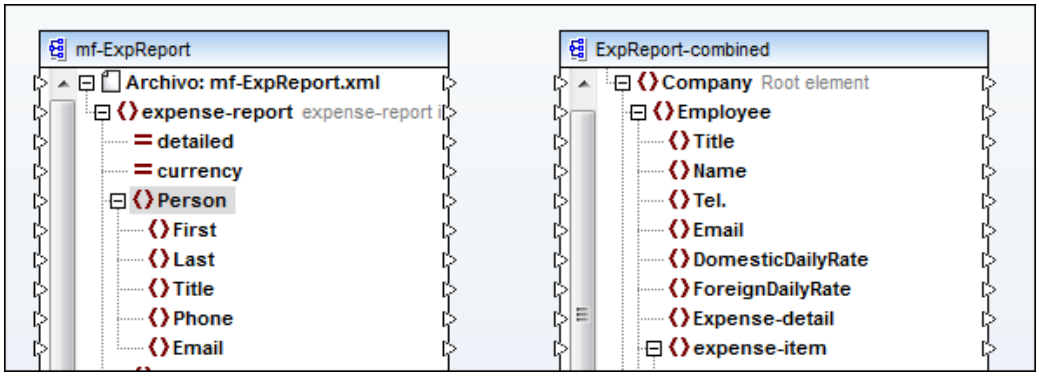
Para crear el entorno de asignación de datos:

1. Haga clic en el icono **Nuevo** de la barra de herramientas estándar para abrir el cuadro de diálogo "Archivo nuevo".
2. Haga clic en el icono **Asignación** y después en **Aceptar** para crear una asignación nueva.
3. Haga clic en el icono **Insertar archivo o esquema XML**.
4. En la carpeta `Tutorial` del directorio `MapForceExamples` seleccione el archivo `mf-ExpReport.xsd`**.** En el cuadro de diálogo que aparece haga clic en **Examinar** y seleccione el archivo `mf-ExpReport.xml` como archivo XML de instancia.
5. Haga clic en la entrada **`expense-report`** y pulse la tecla * del teclado numérico para ver todos sus elementos. Ajuste el tamaño del componente si es necesario.
6. Haga clic en el icono **Insertar archivo o esquema XML**.
7. Seleccione el archivo `ExpReport-combined.xsd`.
   La aplicación le solicita un archivo XML de muestra para este esquema.
8. Haga clic en **Omitir** y seleccione **`Company`** como elemento raíz del documento de destino.



   El componente de esquema de destino aparece ahora en el panel **Asignación**.
9. Haga clic en la entrada **`Company`**, pulse la tecla ***** del teclado numérico para ver todos sus elementos y ajuste el tamaño del componente si es necesario.

 © 2014 Altova GmbH

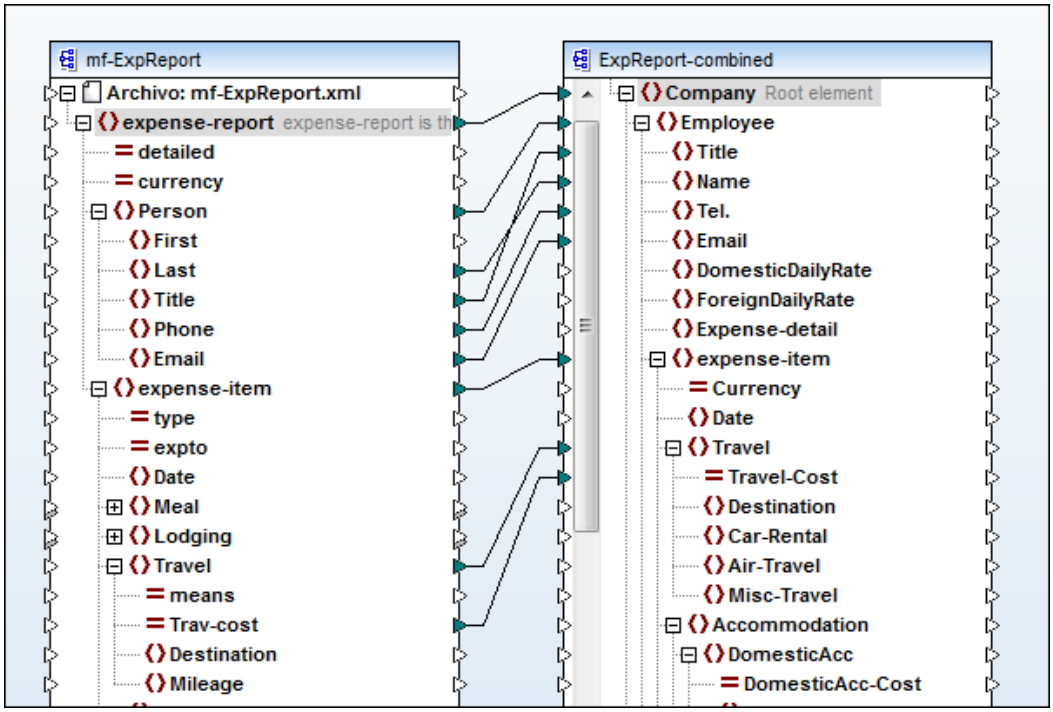


Asignación de los componentes

Ahora cree las asignaciones que aparecen a continuación, pero antes compruebe que el icono **Conectar automáticamente secundarios equivalentes** está desactivado:

- Entre **`Expense-report`** y **`Company`**
- Entre **`Person`** y **`Employee`**
- Entre **`Last`** y **`Name`**
- Entre **`Title`** y **`Title`**
- Entre **`Phone`** y **`Tel.`**
- Entre **`Email`** y **`Email`**
- Entre **`expense-item`** y **`expense-item`**
- Entre **`Travel`** y **`Travel`**
- Entre **`Trav-cost`** y **`Travel-Cost`**

Aquí puede ver las asignaciones:



Cambie al panel **Resultados** para ver el resultado de esta asignación.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:sche
3     <Employee>
4       <Title>Project Manager</Title>
5       <Name>Landis</Name>
6       <Tel.>123-456-78</Tel.>
7       <Email>f.landis@nanonull.com</Email>
8       <expense-item>
9         <Travel Travel-Cost="337.88"/>
10      </expense-item>
11      <expense-item/>
12      <expense-item>
13        <Travel Travel-Cost="1014.22"/>
14      </expense-item>
15      <expense-item>
16        <Travel Travel-Cost="2000"/>
17      </expense-item>
18      <expense-item/>
19    </Employee>
20  </Company>
21
```

Nota: observe que cuando los elementos secundarios de un elemento primario asignado existen en el archivo de origen, pero no se asignan al esquema de destino, en el archivo de salida se crean elementos/etiquetas **`<expense-item/>` vacíos**. En este caso solamente se asignaron los gastos de viaje (**`Travel`**) del primario **`expense-item`**. Sin embargo, hay otros dos elementos en la lista: **`Lodging`** y **`Meal`**. Cada uno de estos elementos genera una etiqueta **`expense-item`** vacía.

Si prefiere no generar etiquetas vacías, cree un filtro como el descrito en el apartado Filtrado de datos del tutorial o conecte el elemento **`Travel`** con el elemento **`expense-item`**.

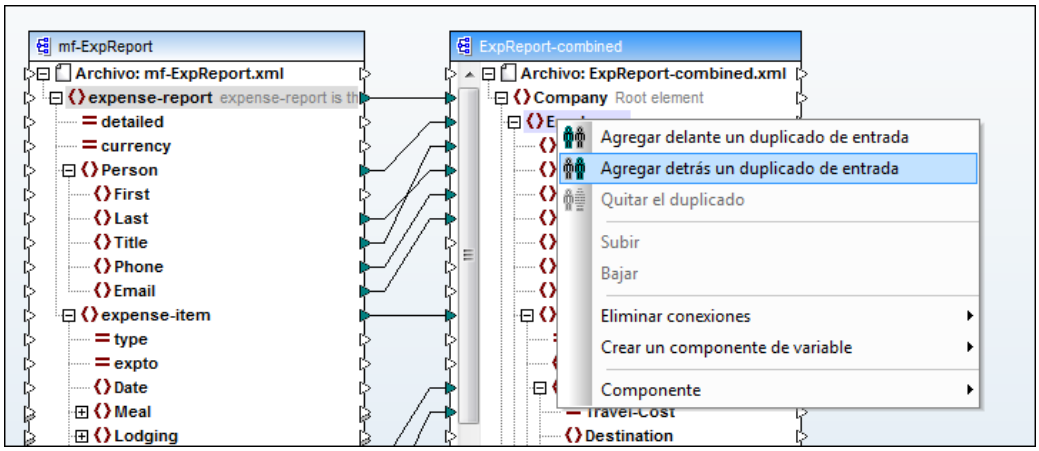
© 2014 Altova GmbH

6.5.2 Duplicar elementos de entrada

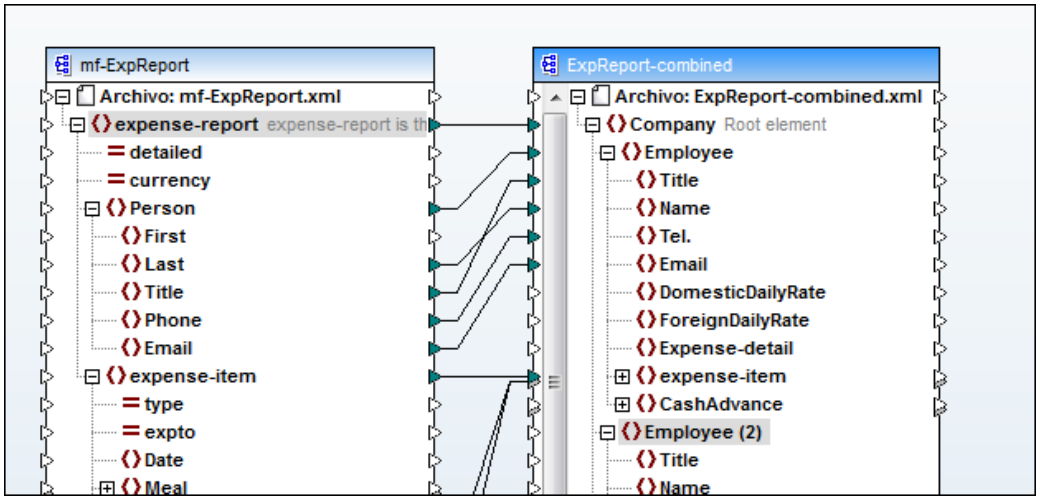
Para poder realizar asignaciones entre componentes de origen diferentes y un solo elemento de destino es necesario duplicar los **elementos de entrada** del componente de destino. Esto se consigue añadiendo un segundo archivo XML de origen y creando asignaciones desde sus elementos hasta la "misma" entrada del elemento duplicado del archivo XML de destino.

Duplicar elementos de entrada:

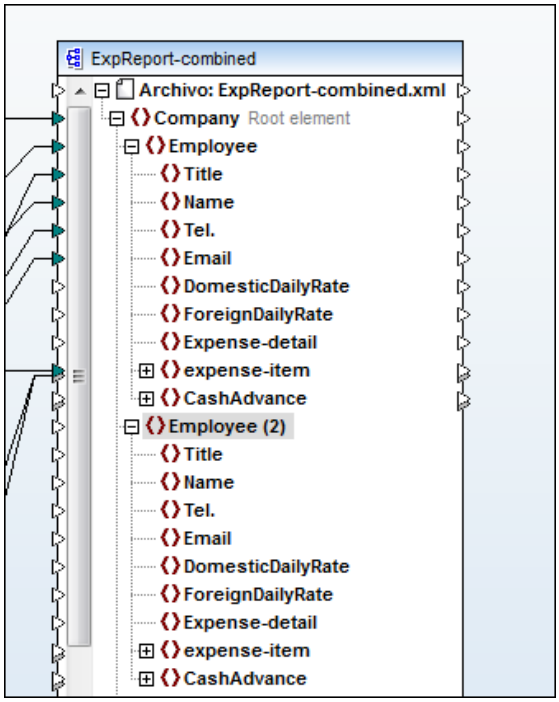
1. Haga clic con el botón secundario en el elemento **`Employee`** del archivo XML de destino.
2. En el menú contextual seleccione la opción **Agregar detrás un duplicado de entrada**.



Como resultado se añade otro elemento **`Employee`** al componente: el elemento **`Employee(2)`**.



3. Haga clic en el icono + para ver sus elementos.
   La **estructura** del nuevo elemento **`Employee`** es una copia exacta del original, excepto que los elementos duplicados no tienen iconos de salida.



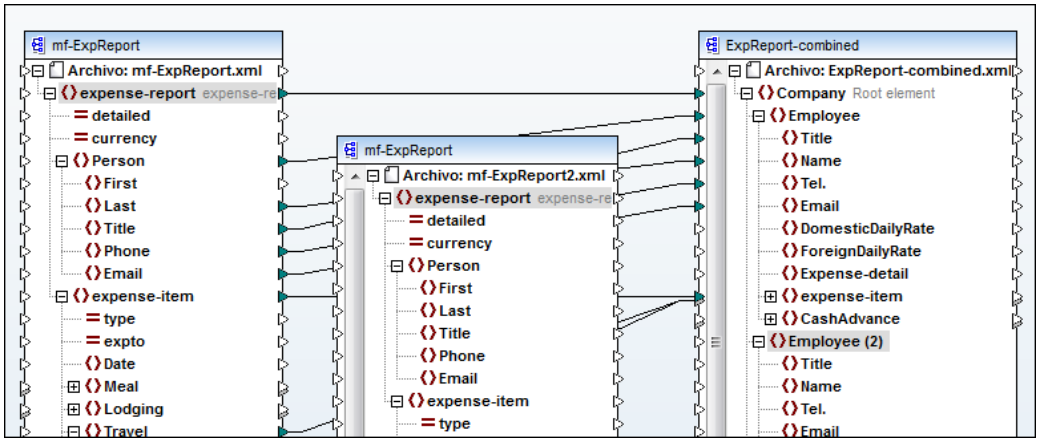
Ahora puede usar estos elementos duplicados como destino del **segundo** archivo XML de origen.

Insertar el segundo archivo XML de instancia

Para insertar el segundo archivo XML de instancia utilizamos el mismo método que con el archivo de esquema XML.

Para insertar otro componente de origen:

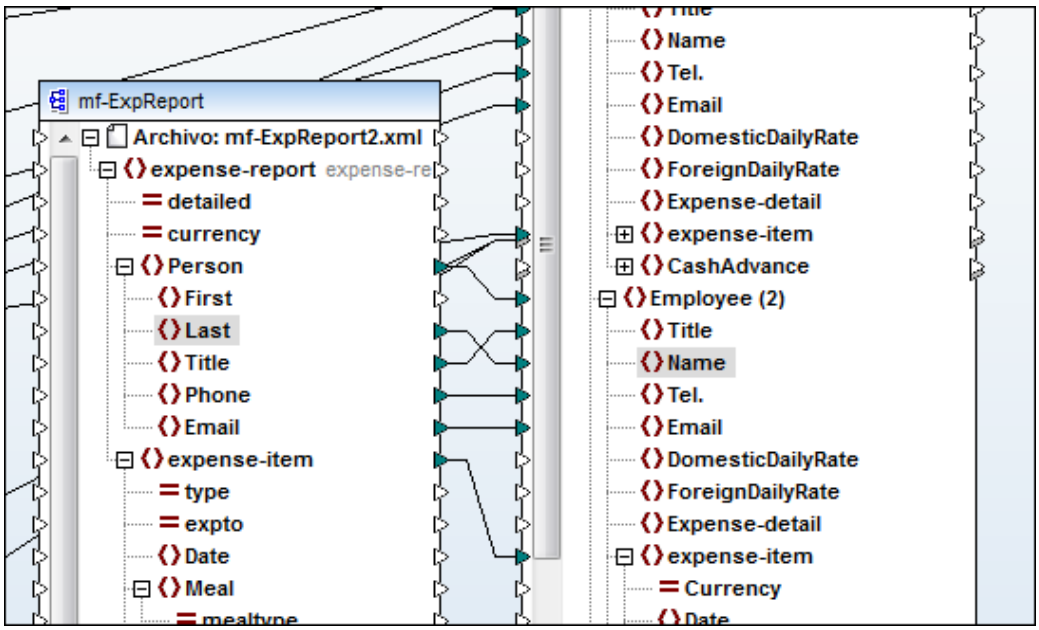
1. Haga clic en el icono **Insertar archivo o esquema XML** .
2. Seleccione el archivo `mf-ExpReport.xsd`. En el cuadro de diálogo que aparece, haga clic **Examinar...** y después seleccione el archivo `mf-ExpReport2.xml` como archivo XML de instancia.
3. Haga clic en la entrada **`expense-report`**, pulse la tecla ***** del teclado numérico para ver todos sus elementos y ajuste el tamaño del componente si es necesario.
   Por motivos de claridad el componente nuevo se colocó entre los dos componentes ya existentes:



 © 2014 Altova GmbH

4. Cree las mismas asignaciones que definimos para el primer archivo XML de origen:
   - Entre **`Person`** y **`Employee(2)`**
   - Entre **`Last`** y **`Name`**
   - Entre **`Title`** y **`Title`**
   - Entre **`Phone`** y **`Tel.`**
   - Entre **`Email`** y **`Email`**
   - Entre **`expense-item`** y **`expense-item`**

   Desplácese un poco más abajo y cree estas asignaciones:
   - Entre **`Travel`** y **`Travel`**
   - Entre **`Trav-cost`** y **`Travel-Cost`**



5. Cambie al 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:schemal
3     <Employee>
4       <Title>Project Manager</Title>
5       <Name>Landis</Name>
6       <Tel.>123-456-78</Tel.>
7       <Email>f.landis@nanonull.com</Email>
8       <expense-item>
9         <Travel Travel-Cost="337.88"/>
10      </expense-item>
11      <expense-item/>
12      <expense-item>
13        <Travel Travel-Cost="1014.22"/>
14      </expense-item>
15      <expense-item>
16        <Travel Travel-Cost="2000"/>
17      </expense-item>
18      <expense-item/>
19    </Employee>
20    <Employee>
21      <Title>Manager</Title>
22      <Name>Johnson</Name>
23      <Tel.>456-789-123</Tel.>
24      <Email>j.john@nanonull.com</Email>
25      <expense-item>
26        <Travel Travel-Cost="150.44"/>
27      </expense-item>
28      <expense-item/>
29      <expense-item>
30        <Travel Travel-Cost="1020"/>
31      </expense-item>
32      <expense-item>
33        <Travel Travel-Cost="70"/>
34      </expense-item>
35    </Employee>
36  </Company>
37
```

Los datos del segundo informe de gastos se añadieron al archivo de salida. Johnson y sus gastos de viaje aparecen ahora junto a los gastos de Fred Landis en el informe de gastos de la compañía.

Para guardar los resultados generados:

- Haga clic en el icono **Guardar resultado generado** que aparece en la barra de herramientas, en el panel **Resultados**.
  El archivo resultante (`mf-ExpReport-combined.xml`) está disponible en la carpeta `...\MapForceExamples\Tutorial`.

Para quitar elementos duplicados:

- Haga clic con el botón secundario en el elemento duplicado y seleccione **Quitar el duplicado** en el menú contextual.

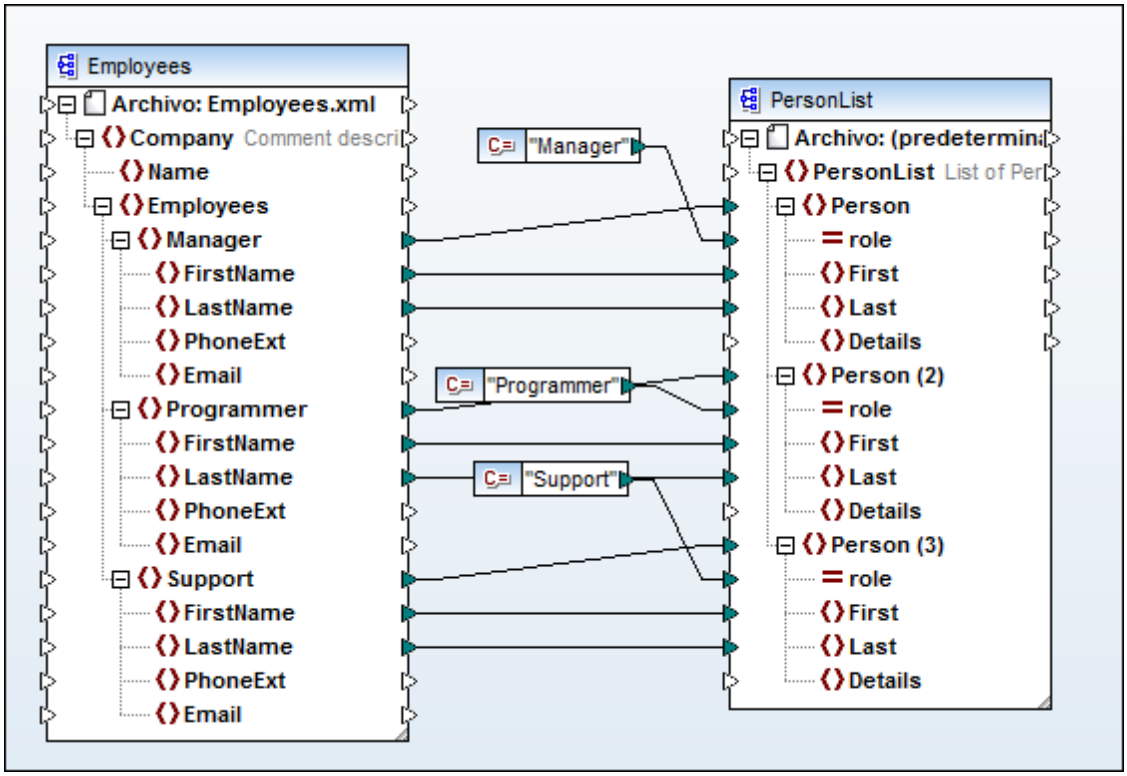
Ejemplo

Para ver otro ejemplo con elementos duplicados, consulte el archivo de muestra `PersonList.mfd`

 © 2014 Altova GmbH

disponible en la carpeta `...\MapForceExamples`.

En el ejemplo `PersonList.mfd` se asignaron diferentes elementos del documento de origen al "mismo" elemento del esquema de destino y algunos elementos (como **`Manager`**, etc.) se asignaron a un elemento genérico usando el atributo **`role`**.




```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <PersonList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="
    C:/.../Altova/MapForce2011/MapForceExamples/PersonList.xsd">
3     <Person role="Manager">
4       <First>Vernon</First>
5       <Last>Callaby</Last>
6     </Person>
7     <Person role="Programmer">
8       <First>Frank</First>
9       <Last>Further</Last>
10    </Person>
11    <Person role="Support">
12      <First>Loby</First>
13      <Last>Matise</Last>
14    </Person>
15    <Person role="Support">
16      <First>Susi</First>
17      <Last>Sanna</Last>
18    </Person>
19  </PersonList>
```

6.6 Trabajar con varios archivos de entrada/salida

En esta sección aprenderemos a usar las funciones de MapForce para trabajar con varios archivos de entrada/salida. Tenga en cuenta que estas funciones no son compatibles con XSLT 1.0 y XQuery.

Un solo componente de origen puede procesar dos documentos de entrada, mientras que un solo componente de destino puede generar dos archivos de salida. El ejemplo utilizado en esta sección se creó en el apartado Filtrado de datos y también se usó como punto de partida en la sección Cómo asignar varios componentes de origen a un solo componente de destino.

Objetivo
En esta sección del tutorial aprenderá a crear una asignación en la que el componente de origen procesa dos archivos XML de entrada y el componente de destino genera dos archivos XML de salida.

Comandos utilizados en esta sección

Guardar todos los archivos de salida...: situado en el menú **Resultados**. Este comando sirve para guardar todos los archivos asignados en el panel **Asignación**.

Archivos de ejemplo utilizados en esta sección

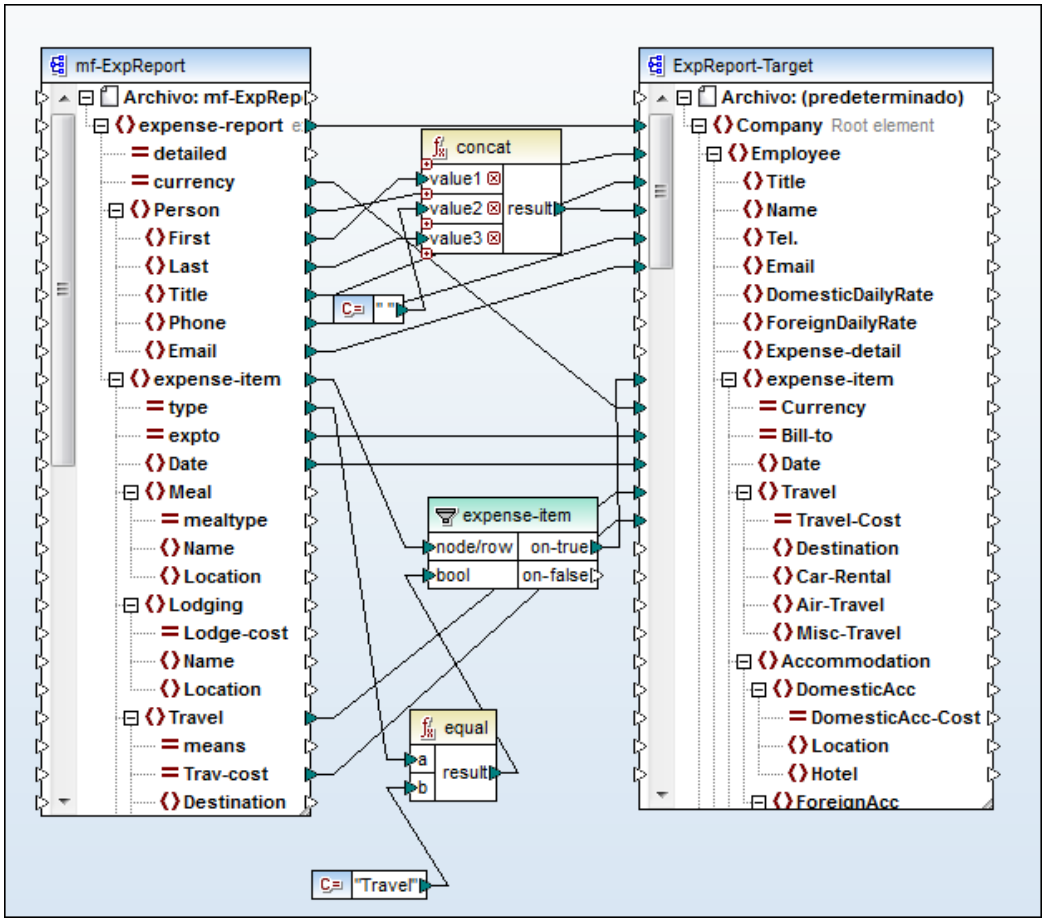
- `mf-ExpReport.xml` Archivo XML de entrada utilizado en la sección anterior
- `mf-ExpReport2.xml` El segundo archivo XML de entrada
- `Tut-ExpReport-multi.mfd` El archivo de asignación para este ejemplo

Nota: los archivos utilizados en esta sección también están en la carpeta `...\MapForceExamples\Tutorial`.

 © 2014 Altova GmbH

6.6.1 Procesar varios archivos por cada componente de entrada/salida

En este ejemplo modificaremos el archivo `Tut-ExpReport.mfd` de la carpeta `...\MapForceExamples` y lo guardaremos con otro nombre.



Antes de continuar, preste atención a estos elementos de cada componente:

- El elemento **`Archivo:mf-ExpReport.xml`** del componente **mf-ExpReport** muestra el archivo XML de entrada/salida. Si el archivo de entrada/salida es el mismo, aparece solamente un elemento. Si son diferentes, aparece un elemento llamado **`NombreArchivoEntrada;NombreArchivoSalida`**.
  Este elemento se rellena automáticamente al asignar el archivo XML de instancia al esquema XML.
- El elemento **`Archivo:(predeterminado)`** del componente **ExpReport-Target** indica que no se asignó ningún archivo de instancia al esquema XML. Por tanto, se usará un valor predeterminado cuando se ejecute la asignación.

Procesar varios archivos

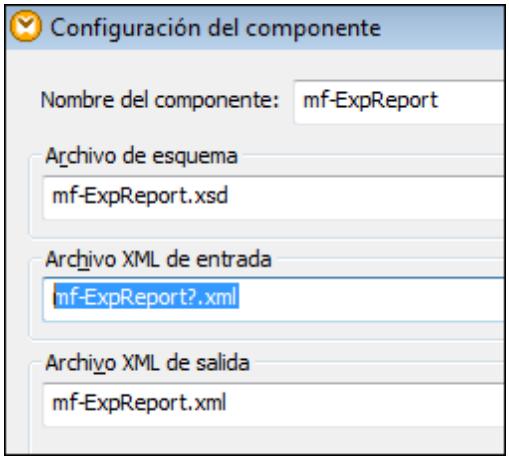
Para procesar varios archivos MapForce utiliza el carácter comodín "?" en el nombre de archivo del archivo XML de entrada. El carácter "?" puede remplazarse con un carácter o no remplazarse.

Para procesar varios archivos:

Después de abrir el archivo `Tut-ExpReport.mfd` de la carpeta `...\MapForceExamples`, haga clic

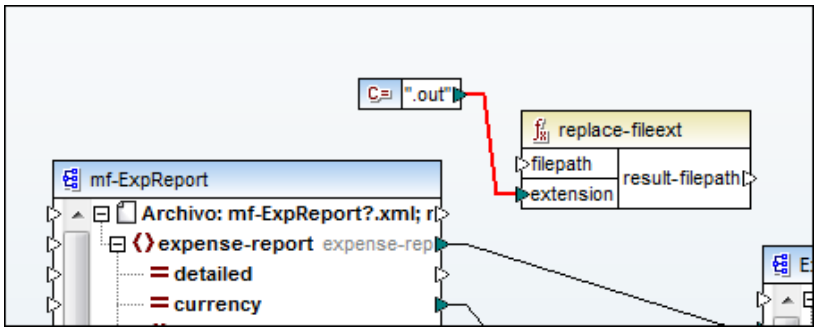
en el icono **XSLT2** de la barra de herramientas.

1. Haga doble clic en la barra de título del componente **mf-ExpReport** de la izquierda.
2. Escriba `mf-expReport?.xml` en el campo *Archivo XML de entrada*.

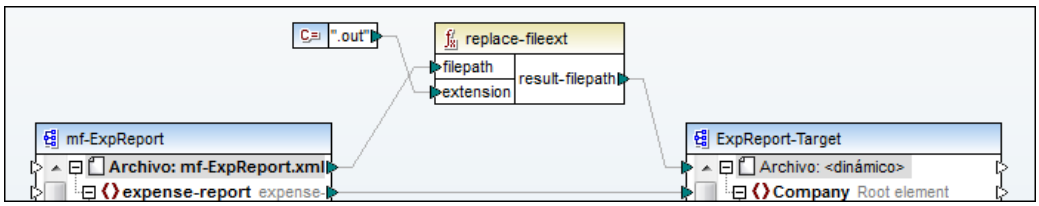


Los nombres de archivo admiten los caracteres comodín **?** y *. Observe que se introdujo una ruta de acceso relativa porque el archivo `Tut-ExpReport.mfd` está disponible en la carpeta `...\Tutorial` (si quiere puede escribir una ruta de acceso absoluta).

3. Inserte la función **`replace-fileext`** del grupo **file path functions** de la biblioteca **core** y después inserte un componente de constante.
4. Escriba `.out` en el componente de constante y conéctelo al parámetro **`extension`** de la función.



5. Conecte el elemento `Archivo:mf-ExpReport?.xml` del componente con el parámetro **`filepath`** de la función.
6. Conecte el parámetro **`result-filepath`** de la función con el elemento **`Archivo: (predeterminado)`** del componente de destino.



El elemento **`Archivo:`** del componente de destino se convierte en **`Archivo: <dinámico>`**.

7. Cambie al panel **Resultados** para ver el resultado.
   Observe que el panel **Resultados** ahora incluye la barra de herramientas *Vista previa*, que muestra los resultados de cada archivo XML de entrada. Por ejemplo, en la imagen

siguiente aparece la vista previa 1 de 2.

Vista previa 1 de 2 C:\MapForceExamples\Tutorial\mf-ExpReport.out

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xmlns="http://my-company.com/namespace" xmlns:xsi="
   http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="
   http://my-company.com/namespace
   C:/MapForceExamples/Tutorial/ExpReport-Target.xsd">
3    <Employee>
4      <Title>Project Manager</Title>
5      <Name>Fred Landis</Name>
6      <Tel.>123-456-78</Tel.>
7      <Email>f.landis@nanonull.com</Email>
8      <expense-item Currency="USD" Bill-to="Development">
9        <Date>2003-01-02</Date>
10       <Travel Travel-Cost="337.88"/>
11     </expense-item>
12     <expense-item Currency="USD" Bill-to="Accounting">
13       <Date>2003-07-07</Date>
14       <Travel Travel-Cost="1014.22"/>
15     </expense-item>
16     <expense-item Currency="USD" Bill-to="Marketing">
17       <Date>2003-02-02</Date>
18       <Travel Travel-Cost="2000"/>
19     </expense-item>
20   </Employee>
21 </Company>
```

8. Haga clic en el cuadro desplegable para seleccionar el resultado del segundo archivo XML de entrada.
   Observe que el cuadro combinado muestra todos los archivos XML de origen, pero su extensión `*.xml` se reemplazó con la extensión `*.out`.

Vista previa 2 de 2 C:\MapForceExamples\Tutorial\mf-ExpReport2.out

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xmlns="http://my-company.com/namespace" xmlns:xsi="
   http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="
   http://my-company.com/namespace
   C:/MapForceExamples/Tutorial/ExpReport-Target.xsd">
3    <Employee>
4      <Title>Manager</Title>
5      <Name>James Johnson</Name>
6      <Tel.>456-789-123</Tel.>
7      <Email>j.john@nanonull.com</Email>
8      <expense-item Currency="Euro" Bill-to="Sales">
9        <Date>2004-02-03</Date>
10       <Travel Travel-Cost="150.44"/>
11     </expense-item>
12     <expense-item Currency="Euro" Bill-to="Operations">
13       <Date>2004-08-08</Date>
14       <Travel Travel-Cost="1020"/>
15     </expense-item>
16     <expense-item Currency="Euro" Bill-to="Support">
17       <Date>2004-03-03</Date>
18       <Travel Travel-Cost="70"/>
19     </expense-item>
20   </Employee>
21 </Company>
```

Haga clic en el icono **Guardar todos** para guardar todos los archivos de la ventana

de vista previa sin necesidad de generar código. Aparece un cuadro de diálogo preguntando si los archivos de salida que estén en la misma ubicación se pueden sobrescribir.

9. Guarde el archivo de asignación con otro nombre.

Nota: para más información sobre cómo trabajar con archivos dinámicos consulte la sección Archivos de entrada/salida dinámicos por componente.

© 2014 Altova GmbH

6.7 Asignación de datos entre una BD y un esquema

Esta sección explica cómo usar una sencilla base de datos Microsoft Acces como origen de datos para asignar datos de BD a un esquema.

Objetivo
En esta sección aprenderá a insertar una BD en una asignación de MapForce y a asignar las tablas de esta BD al componente de esquema de destino. En concreto aprenderá a:

- Conectarse a una base de datos Microsoft Access
- Asignar tablas de BD a los nodos del esquema XML

Comandos utilizados en esta sección

Nuevo...: haga clic en este icono para abrir el cuadro de diálogo "Archivo nuevo" donde puede crear un archivo de asignación nuevo.

Insertar base de datos: haga clic en este comando para abrir el cuadro de diálogo "Seleccione una base de datos", donde puede crear una conexión con la BD.

Insertar archivo o esquema XML: haga clic en este icono para abrir el cuadro de diálogo Abrir de Windows y seleccionar el archivo de su sistema de archivos.

Conectar automáticamente los secundarios equivalentes: haga clic en este icono para activar/desactivar la conexión automática de nodos secundarios equivalentes.

Generar código en: situado en el menú **Archivo**. Al hacer clic en este comando se abre un submenú donde puede elegir el tipo de código que desea generar (es decir, **Java, C++** o **C#**). El comando abre el cuadro de diálogo "Buscar carpeta" donde puede seleccionar la ubicación donde se debe guardar el código generado.

Las bases de datos compatibles y sus objetos raíz aparecen a continuación. Altova procura ofrecer compatibilidad con otras bases de datos ODBC/ADO, 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. MapForce es compatible con los archivos lógicos de la base de datos IBM iSeries y los muestra como si fueran vistas.

En la lista también aparece el objeto raíz disponible por cada base de datos compatible.

Base de datos (con soporte nativo)	Objeto raíz
MS Access 2003, 2007, 2010, 2013	database
MS SQL Server 2000, 2005, 2008 y 2012	database
MS SQL Server 2005, 2008 y 2012	schema
Oracle 9i, 10g y 11g	schema
MySQL 4.x, 5.x y 5.5.28	database
PostgreSQL 8.0, 8.1, 8.2, 8.3, 9.0.10, 9.1.6, 9.2.1	database
Sybase ASE12 y ASE15	database

MS Access 2003, 2007, 2010, 2013	database
IBM DB2 8.x, 9.5, 9.7 y 10.1	schema
IBM DB2 para iSeries 6.1 y 7.1	schema
IBM Informix 11.70	database

Notas: cuando se conecte a una base de datos Sybase a través de JDBC, la instrucción SELECT no puede recuperar datos. Se trata de un problema conocido de controlador. Utilice otro controlador para conectarse a la BD.

Si recibe este mensaje de error cuando se conecte a la BD a través del cliente SQL nativo:
Error de conexión de base de datos.
Motivo: no se puede conectar a la base de datos. Error de inicio de sesión. El inicio de sesión se realiza desde un dominio que no es de confianza y no se puede utilizar con autenticación de Windows.

Entonces siga estas instrucciones:

Para una conexión ADO a un MS SQL Server con controlador SQL Server Native Client 10.0 debe establecer estos valores de propiedades en la pestaña *Todas* del cuadro de diálogo "Propiedades de vínculo de datos":
(i) el valor *Seguridad integrada* debe ser un carácter de espacio
(ii) el valor de *Información de seguridad persistente* debe ser `true`.

MapForce es compatible con archivos lógicos de la base de datos IBM iSeries y presenta los archivos lógicos como vistas de la BD.

El controlador ODBC 5.2 es necesario si usa procedimientos sin parámetros de salida con MySQL 5.5.

Cuando genere archivos JAR a partir de código Java generado, recuerde que las entradas CLASSPATH para el control de la BD no se añaden automáticamente al archivo de manifiesto. Debe editar el archivo MANIFEST.MF a mano añadiendo el JAR externo del controlador de la BD a la CLASSPATH (p. ej. añada esta línea al archivo de manifiesto `Class-Path: mysql-connector-java-5.1.20-bin.jar`).

Nota sobre compatibilidad con IBM Informix:
Informix admite conexiones ADO, JDBC y ODBC.

La implementación no es compatible con tipos de datos de objetos grandes en ninguno de los lenguajes de generación de código. MapForce emite un mensaje de error (durante la generación de código) si se usan estos tipos de datos.

La tabla que aparece a continuación muestra el tipo de BD que se crea cuando insertamos BD en MapForce, así como sus limitaciones y métodos de conexión.

	Métodos de conexión

Bases de datos compatibles	**Restricciones ODBC** (ODBC no admite claves únicas)	**Restricciones ADO**
Microsoft Access (ADO)	OK No se admiten claves primarias ni foráneas.	OK *
MS SQL Server (ADO)	OK	OK *
Oracle (ODBC)	OK, Limitaciones: tablas que contienen columnas de tipo CLOB, BLOB, BFILE; tablas XML	OK, Limitaciones: tablas que contienen columnas de tipo CLOB, BLOB, BFILE; tablas XML; datos del propietario, las restricciones de identidad no se leen de la BD
MySQL (ODBC)	OK *	OK ⌖
Sybase (ODBC)	OK *	OK
IBM DB2 (ODBC)	OK *	OK

* **Método de conexión recomendado**

⌖ **MySQL: cuando cree la conexión ADO basada en ODBC recomendamos que utilice el DNS de usuario o del sistema**.

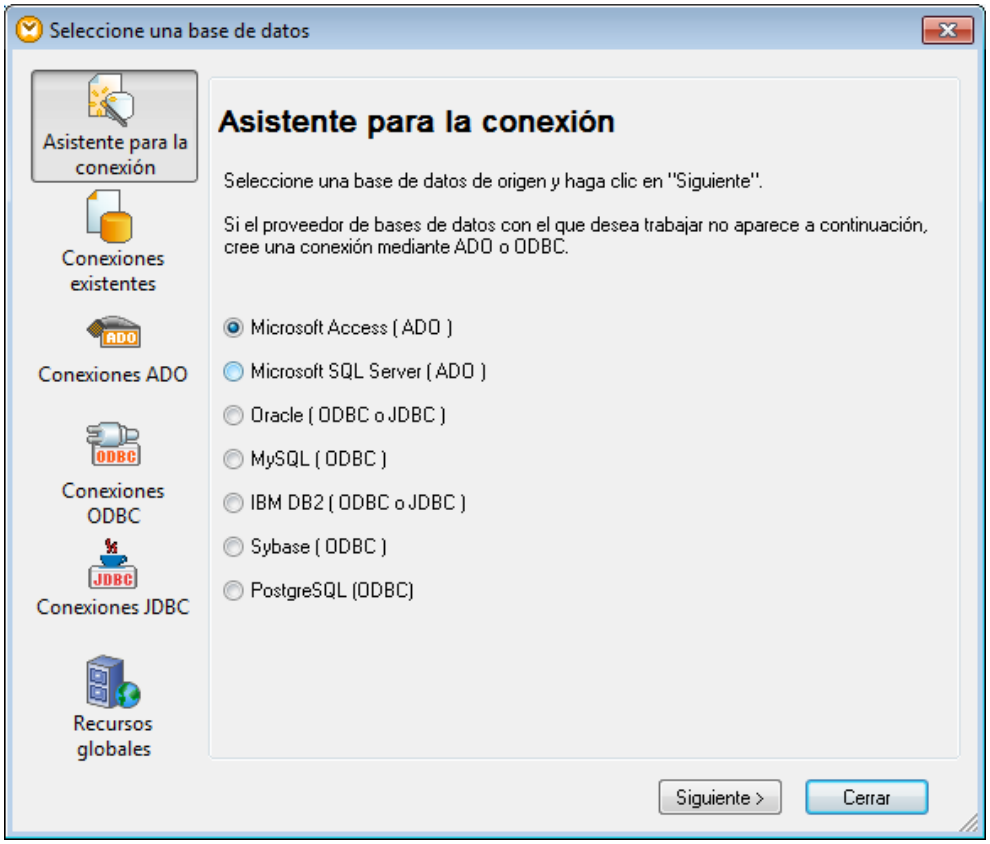
\- **No disponible**

Nota: también puede insertar bases de datos como recursos globales. Para más información consulte el apartado Recursos globales.

6.7.1 Conectarse a una base de datos Access

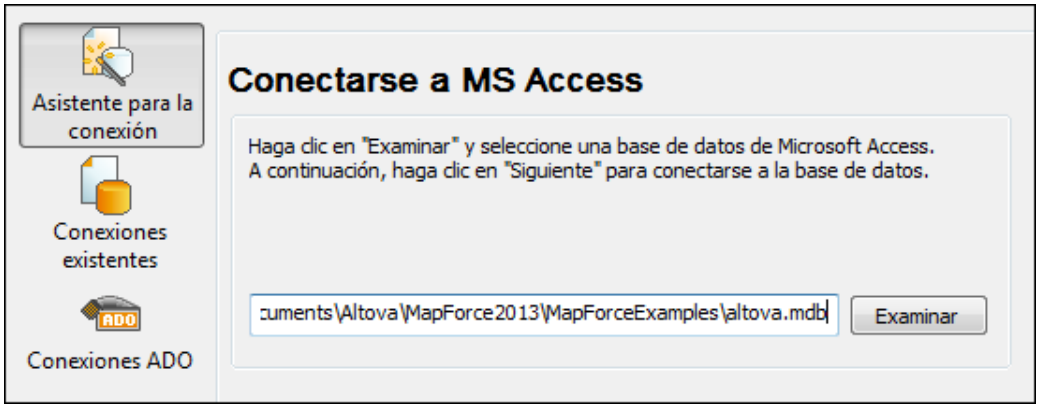
Para crear un componente de base de datos en MapForce:

1. Seleccione **Archivo | Nuevo | Asignación** en MapForce para crear una asignación.
2. En la barra de herramientas haga clic en uno de estos iconos: **Java, C#, C++** o **BUILTIN**.
3. Después haga clic en el icono **Insertar base de datos** .
4. Aparece el cuadro de diálogo "Seleccione una base de datos". Haga clic en el botón **Asistente para la conexión**, a la izquierda del cuadro de diálogo.

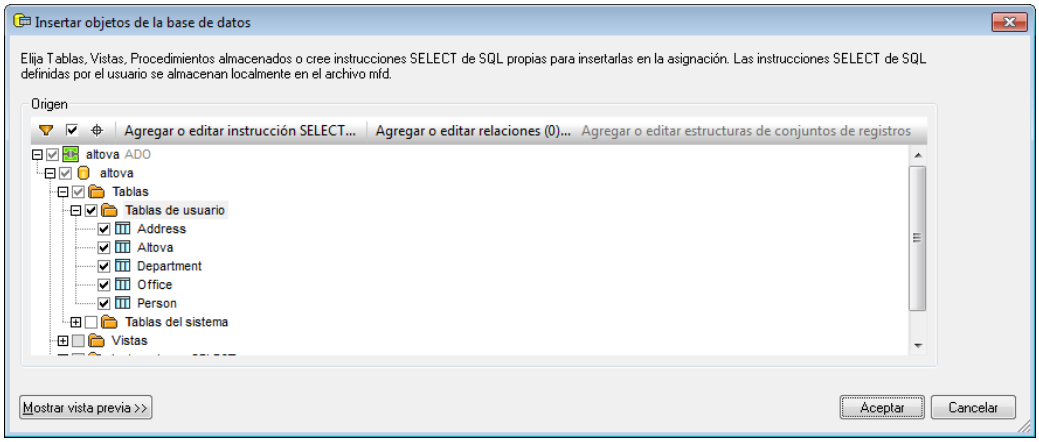


5. Seleccione el botón de opción **Microsoft Access (ADO)**.
6. Haga clic en **Siguiente** para continuar.
7. Ahora pulse el botón **Examinar** y seleccione la BD que desea usar como origen de datos. En este caso seleccionamos `altova.mdb` en la carpeta `...\MapForceExamples\Tutorial`.
   En el cuadro de texto aparece la cadena de conexión.

 © 2014 Altova GmbH



8. Haga clic en el botón **Conectarse**.



9. Aparece el cuadro de diálogo "Insertar objetos de la base de datos". Marque la casilla *Tablas de usuario* y después haga clic en **Aceptar** para insertar el componente (de esquema) de base de datos. Como resultado se seleccionan todas las tablas de la carpeta *Tablas de usuario*.

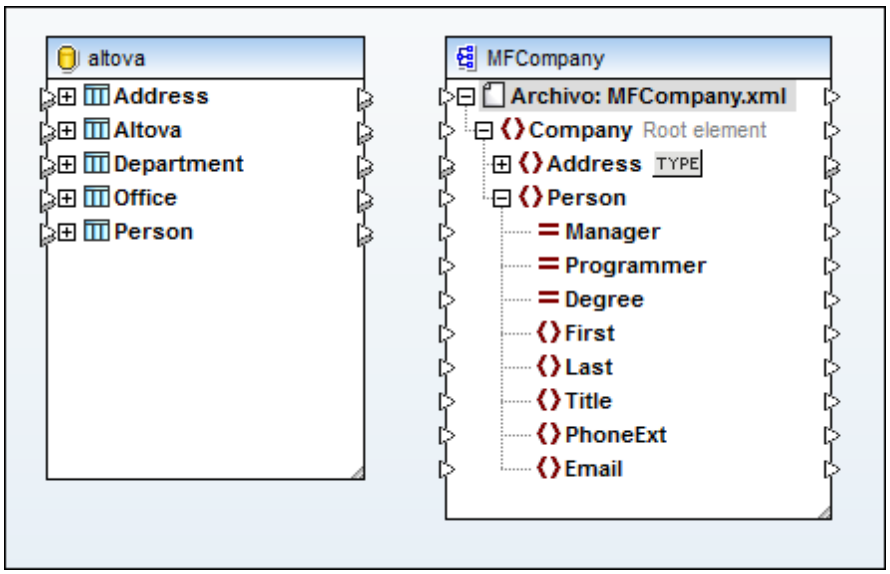
El componente de BD aparece en el panel **Asignación**. Ahora puede crear asignaciones entre los elementos de la BD y un esquema de destino.

6.7.2 Asignar los datos de la base de datos

Después de insertar el componente de BD en el panel **Asignación** podemos añadir el componente de esquema de destino y crear asignaciones entre ambos.

Para insertar el documento/esquema de destino:

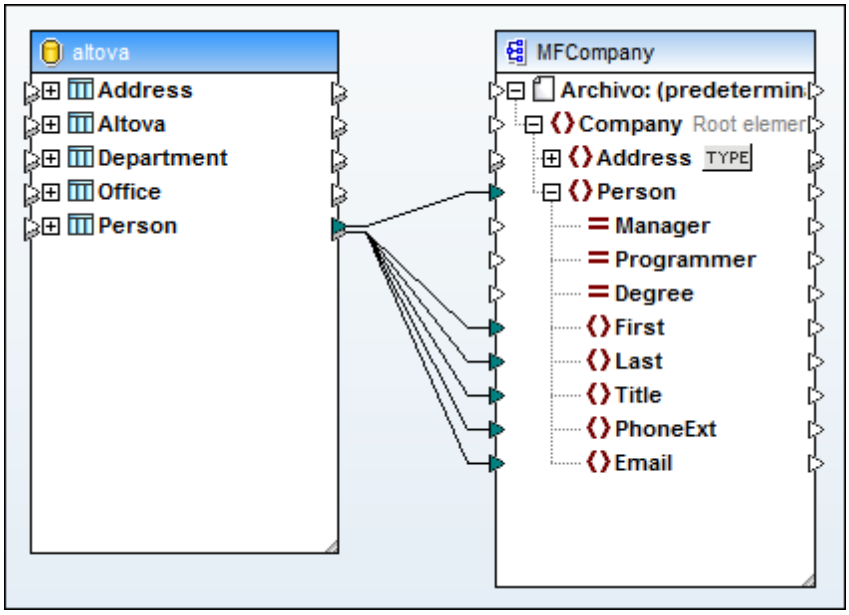
1. Haga clic en el icono **Insertar archivo o esquema XML** y seleccione el esquema `MFCompany.xsd`.
2. Haga clic en **Omitir** cuando aparezca el cuadro de diálogo solicitando un archivo XML de muestra.
3. Seleccione **`Company`** como elemento raíz y expanda todos sus elementos.
   Ya puede empezar a asignar los datos de la BD al esquema XML.



Para asignar datos de base de datos a un esquema/documento en MapForce:

1. Active el icono **Conectar automáticamente los secundarios equivalentes** .
2. Haga clic en el elemento tabla **`Person`** del componente de BD y conéctelo con el elemento **`Person`** de **MFCompany**.
   Observe que también se crean conexiones entre los elementos que tienen el mismo nombre en ambos componentes.

 © 2014 Altova GmbH



4. Guarde el archivo de MapForce (con el nombre `PersonDB`, por ejemplo).
5. Cambie al panel **Resultados** para ver el resultado de esta asignación. El motor de ejecución integrado (BUILTIN) genera resultados instantáneamente, sin necesidad de generar ni compilar código.

Para generar código Java y el archivo XML resultante:

1. Seleccione la opción de menú **Archivo | Generar código en | Java**.
2. Seleccione el directorio donde desea guardar los archivos Java y pulse **Aceptar**.
   En la ventana Mensajes aparece el mensaje *El código se generó correctamente*.
3. Compile el código generado y ejecútelo.
   Este es el archivo `MFCompany.xml` creado.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Company xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     <Person>
4       <First>Vernon</First>
5       <Last>Callaby</Last>
6       <Title>Office Manager</Title>
7       <PhoneExt>582</PhoneExt>
8     </Person>
9     <Person>
10      <First>Frank</First>
11      <Last>Further</Last>
12      <Title>Accounts Receivable</Title>
13      <PhoneExt>471</PhoneExt>
14    </Person>
15    <Person>
16      <First>Loby</First>
17      <Last>Matise</Last>
18      <Title>Accounting Manager</Title>
19      <PhoneExt>963</PhoneExt>
20    </Person>
```

Para ver ejemplos más complejos de asignaciones de datos entre BD y esquemas usando (i) varios archivos de origen y (ii) bases de datos planas y jerárquicas, consulte los archivos

`DB_Altova_SQLXML.mfd` y `DB_Altova_Hierarchical.mfd` situados en la carpeta `...\MapForceExamples` de MapForce.

© 2014 Altova GmbH

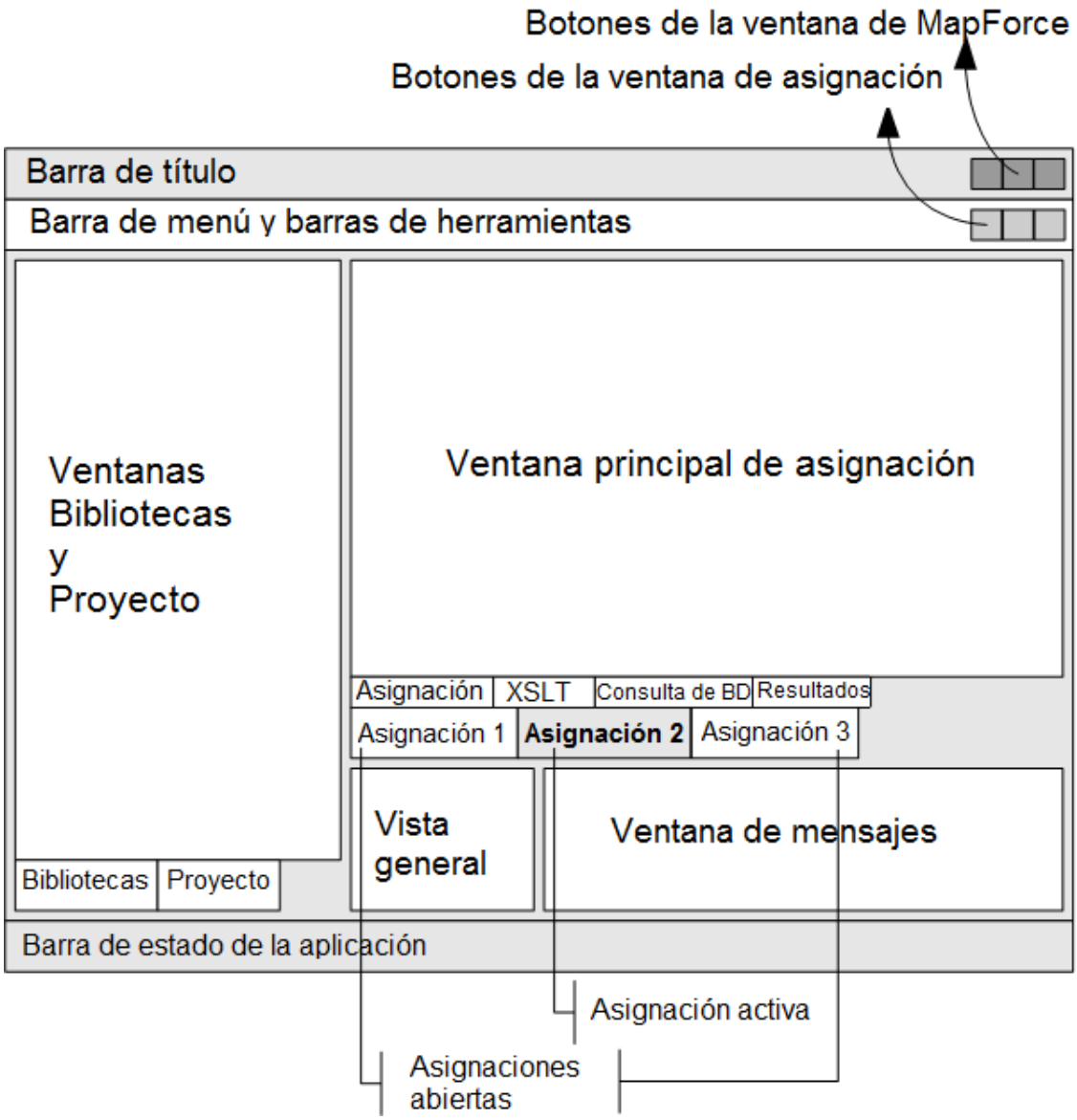
Altova MapForce 2014

Interfaz del usuario

7 Interfaz del usuario

MapForce se divide en tres partes:

- la sección izquierda, que incluye la ventana Bibliotecas y la ventana Proyecto
- la ventana de asignación principal (compuesta por los paneles **Asignación**, **XSLT**, **XSLT2**, **XQuery**, **Consulta de BD**, y **Resultados**)
- y, por último, en la parte inferior, las ventanas Vista general y Mensajes.



Barra de título

La barra de título muestra el nombre de la aplicación (es decir, MapForce) seguida del nombre del

 © 2014 Altova GmbH

diseño activo en la ventana de asignación. Los botones que permiten controlar la ventana de la aplicación MapForce están en la esquina superior derecha.

Barra de menú y barras de herramientas

La barra de menú muestra los diferentes menús de la aplicación. Cada barra de herramientas incluye un grupo de iconos que equivalen a comandos de MapForce. La posición de la barra de menú y de las barras de herramientas se puede cambiar. Para ello haga clic en el identificador de la barra y arrástrela hasta la posición deseada.

Ventanas Bibliotecas y Proyecto

La ventana Bibliotecas muestra grupos de funciones diferentes, dependiendo del lenguaje de salida seleccionado. Las funciones se pueden arrastrar a la ventana de asignación.

La ventana Proyecto muestra todos los archivos y carpetas relacionados con el proyecto o el servicio web respectivamente. Puede añadir carpetas y archivos al proyecto y definir opciones de configuración predeterminadas del proyecto con el menú **Proyecto**.

Ventana de asignación

La ventana de asignación muestra los elementos gráficos utilizados para crear la asignación (transformación) entre los componentes. El **componente de origen** muestra la estructura del esquema de origen y el **componente de destino** muestra la estructura del esquema de destino. Los **conectores** conectan los **iconos** de entrada y salida de cada elemento del esquema. Los **elementos** del esquema pueden ser atributos o elementos.

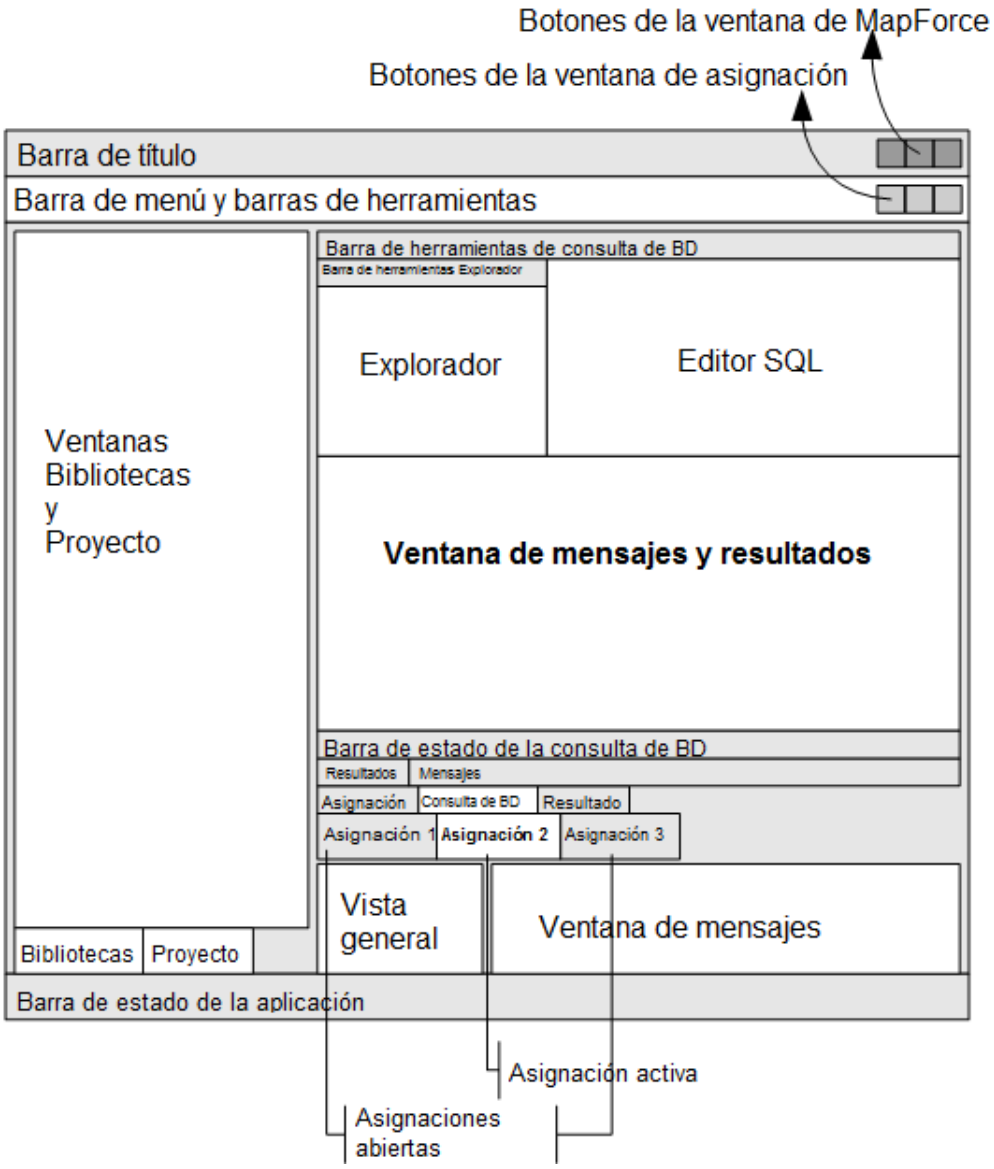
En la parte inferior de la ventana de asignación hay varias pestañas que abren estos paneles:

- El panel **XSLT** , **XSLT2** y **XQuery** ofrece una vista previa del código de transformación en el lenguaje seleccionado.
- El panel **Consulta de la BD** sirve para consultar todas las bases de datos compatibles. Las consultas/acciones definidas en este panel son independientes del resto de paneles y vistas de MapForce y no se guardan en el archivo `*.mfd`.

  En el panel **Consulta de la BD** puede realizar **varias consultas de BD** por asignación. También puede tener varias conexiones de BD activas al mismo tiempo en cada panel de consulta.
- El panel **Resultados** ofrece una vista previa de los datos transformados (o asignados) en una vista de texto.
- Los paneles **HTML, RTF, PDF y Word 2007+** muestran los datos del componente de destino como documentos HTML, RTF, PDF o Word 2007+ si el componente tiene asociado un archivo SPS (StyleVision Power Stylesheet).

Ventana de consulta de BD

El panel **Consulta de BD** incluye varias ventanas, barras de herramientas y una barra de estado para conectarse y trabajar con bases de datos (*imagen siguiente*).



En el panel **Consulta de BD** puede consultar todas las bases de datos compatibles. Las consultas/acciones definidas en este panel son independientes del resto de paneles y vistas de MapForce y no se guardan en el archivo `*.mfd`.

El panel **Consulta de BD** está formado por:

- La ventana Explorador, que muestra información sobre la conexión y las tablas de la BD y ofrece un resumen de los objetos de la BD. Por ejemplo incluye datos sobre las restricciones de la BD como, por ejemplo, las columnas de clave primaria o foránea. En las bases de datos IBM DB2 versión 9, el Explorador también muestra en una carpeta todos los esquemas XML registrados en la BD.
- La ventana Editor SQL, que se utiliza para escribir y ejecutar instrucciones SQL. El Editor SQL ofrece funciones de generación automática y finalización automática de instrucciones SQL, definición de regiones e inserción de comentarios.
- El panel inferior formado por la pestaña *Resultados*, que muestra los resultados de la

 © 2014 Altova GmbH

consulta en forma de tabla, y la pestaña *Mensajes* que muestra advertencias y errores.

En el panel Consulta de la BD puede realizar **varias consultas de BD** por asignación. También puede tener varias conexiones de BD activas al mismo tiempo en cada panel de consulta.

Barra de estado del panel Consulta de la BD

La barra de estado del panel **Consulta de la BD** está situada en la parte inferior de la pestaña *Resultados*. Esta barra de estados ofrece información sobre el progreso de la consulta: si la recuperación de datos se ejecutó correctamente, si se anuló o si el usuario la detuvo. Además muestra el número de filas y columnas recuperadas e informa del tiempo necesario para recuperar los datos y ejecutar la consulta.

Ventanas Vista general y Mensajes

La ventana Vista general muestra todo el área de trabajo en tamaño reducido e incluye un rectángulo de color rojo que puede arrastrar por la superficie para navegar por la asignación.

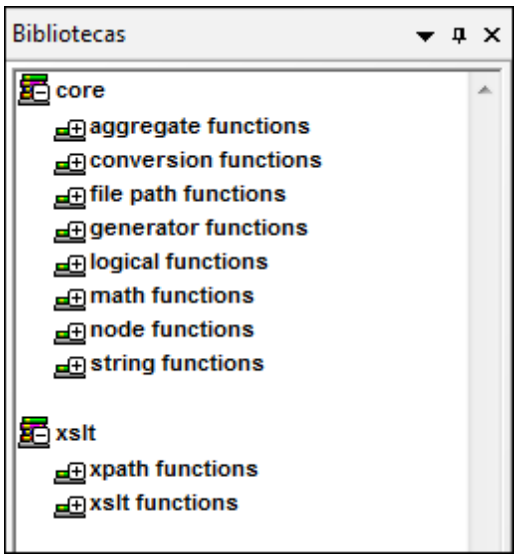
La ventana Mensajes muestra mensajes de advertencia y errores producidos durante el proceso de asignación. Si hace clic en un mensaje, el elemento afectado se resalta en la ventana de asignación.

Barra de estado de la aplicación

La barra de estado de la aplicación está situada en la parte inferior de la ventana de la aplicación y ofrece información sobre esta. Cuando pase el puntero del mouse por un icono o comando, la barra de estado ofrece información sobre el comando. Es importante no confundir la barra de estado de la aplicación con la barra de estado del panel **Consulta de la BD**. Si usa la versión de 64 bits de MapForce, en la barra de estado podrá observar que el nombre de la aplicación tiene el sufijo (x64).

7.1 Ventana Bibliotecas

La ventana **Bibliotecas** muestra las bibliotecas de funciones disponibles para el lenguaje de programación seleccionado, así como todas las funciones de dichas bibliotecas. Además incluye una breve descripción de cada función. Estas funciones se pueden arrastrar desde esta ventana hasta el panel **Asignación**. Una vez insertadas en el panel **Asignación**, se convierten en componentes de la asignación.



Con XSLT seleccionado

Las bibliotecas estándar **core, lang, xpath2, edifact, xbrl, xlsx** y **xslt** se cargan automáticamente y el usuario no necesita añadirlas. La biblioteca **core** agrupa las funciones que se pueden utilizar para producir todo tipo de datos de salida: XSLT, XQuery, Java, C#, C++,. Las demás bibliotecas (**xslt**, **xslt2**, **xpath2**, **lang**, etc.) contienen funciones asociadas al tipo de datos de salida elegido.

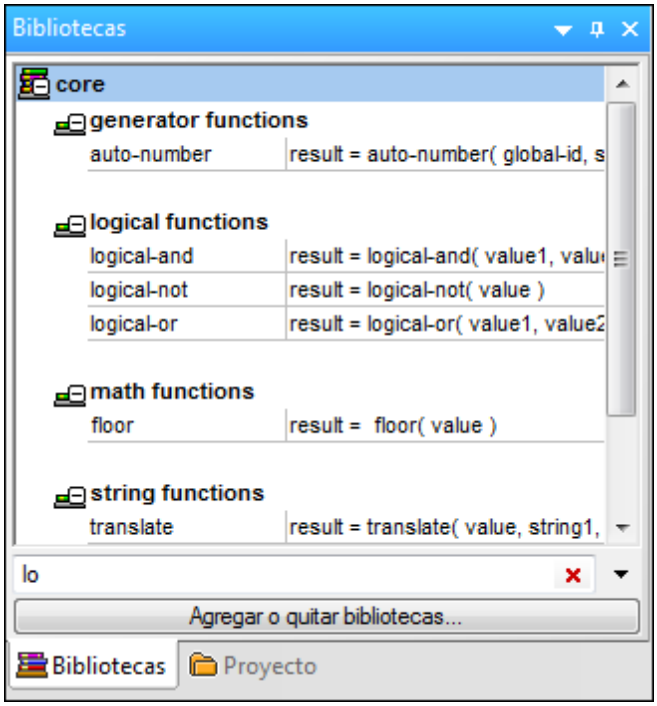
Nota: la biblioteca **xpath2** y sus funciones están disponibles tanto con XSLT 2.0 como con XQuery. Si selecciona el icono **Motor de ejecución integrado** BUILT IN, las funciones disponibles son las mismas que las de los lenguajes Java, C# y C++.

Si selecciona	Se habilitan
XSLT	bibliotecas **core** y **xslt** (funciones XPath 1.0 y XSLT 1.0)
XSLT2	bibliotecas **core**, **xpath2** y **xslt**
XQuery	bibliotecas **core** y **xpath2**

Limitaciones relacionadas con XPath 2.0: no están disponibles varias funciones XPath 2.0 relacionadas con secuencias.

Buscar funciones en la ventana Bibliotecas
La parte inferior de la ventana **Bibliotecas** incluye un campo de búsqueda para buscar funciones.

 © 2014 Altova GmbH

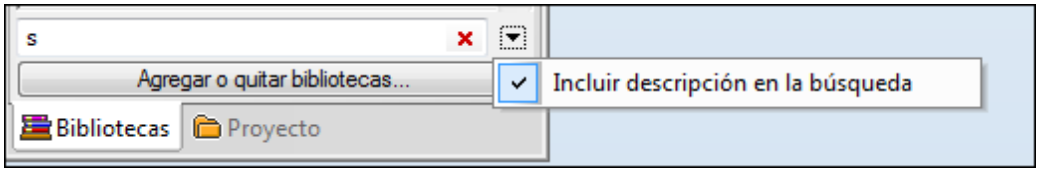


Con XSLT seleccionado

Si pulsa la tecla **Esc** se elimina el texto del campo de búsqueda. También puede eliminarlo haciendo clic en el icono rojo **x**.

Para buscar una función en la ventana Bibliotecas:

1. Haga clic en la ventana Bibliotecas y escriba los caracteres que busca. Por ejemplo, "lo".
   En la ventana Bibliotecas aparecen ahora las funciones que contienen estos caracteres (cada una dentro de su biblioteca).
2. Haga clic en la punta de flecha situada junto al campo de búsqueda y seleccione **Incluir descripción en la búsqueda** si desea incluir el texto de la descripción de la función en la búsqueda.



Agregar bibliotecas de funciones nuevas

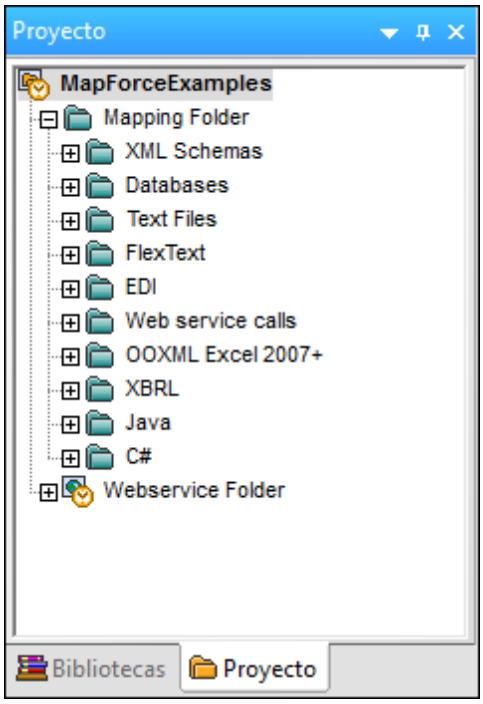
En MapForce puede crear e integrar sus propias bibliotecas de funciones. Para más información consulte los apartados Agregar bibliotecas de funciones personalizadas, Agregar funciones XSLT 1.0 personalizadas, Agregar funciones XSLT 1.0 personalizadas y Funciones definidas por el usuario.

Nota: puede definir funciones/bibliotecas personalizadas para Java, C# y C++, así como para XSLT y XSLT 2.

7.2 Ventana Proyecto

MapForce es compatible con la especificación de interfaz de múltiples documentos, permitiendo al usuario agrupar sus asignaciones en proyectos de asignación. Los archivos de proyecto tienen la extensión `*.mfp`. La ventana Proyecto muestra todos los archivos y carpetas que se agregaron al proyecto.

Nota: si pulsa las teclas **Ctrl+F** puede buscar elementos en la ventana Proyecto.



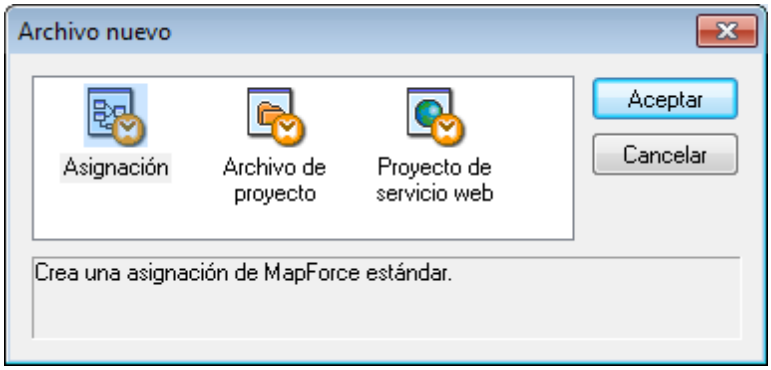
Puede definir dos tipos de proyectos:

- Un proyecto que agrupa varias asignaciones (o proyecto estándar)
- Un proyecto que agrupa asignaciones relacionadas que componen un proyecto de asignación WSDL

En ambos casos puede generar código para todo el proyecto.

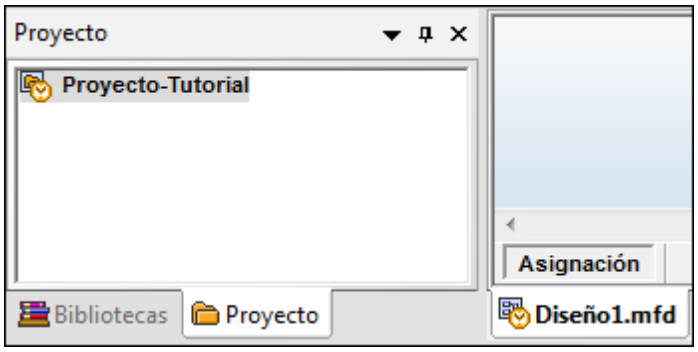
Para crear un proyecto nuevo:

1. Seleccione **Archivo | Nuevo** y haga doble clic en el icono **Archivo de proyecto**.



 © 2014 Altova GmbH

2. Introduzca el nombre del proyecto en el cuadro de diálogo "Guardar el proyecto como" y haga clic en **Guardar**.
   En la ventana Proyecto aparece ahora una carpeta de proyecto.



3. Seleccione **Archivo | Nuevo** y haga doble clic en el icono **Asignación**.
   En el panel **Asignación** se abre un archivo de asignación nuevo llamado `Diseño nuevo1`.

Crear proyectos de servicio web

Para crear un proyecto de servicio web es necesario tener un archivo WSDL. Además, en un proyecto de servicio web, el resultado del proceso de generación de código es un servicio web completo. Después solamente hay que compilar el código generado e implementar el servicio web en el servidor web.

Cada operación definida en el archivo WSDL se presenta en un archivo de asignación diferente. Para más información consulte el apartado Implementar servicios web.

Para crear un proyecto de servicio web:

1. Seleccione **Archivo | Nuevo** y haga doble clic en el icono **Proyecto de servicio web**.
2. Haga clic en el botón **Examinar** para buscar el archivo de definición de servicio web (el archivo `*.wsdl`).
   El resto de campos del cuadro de diálogo se rellenan automáticamente.

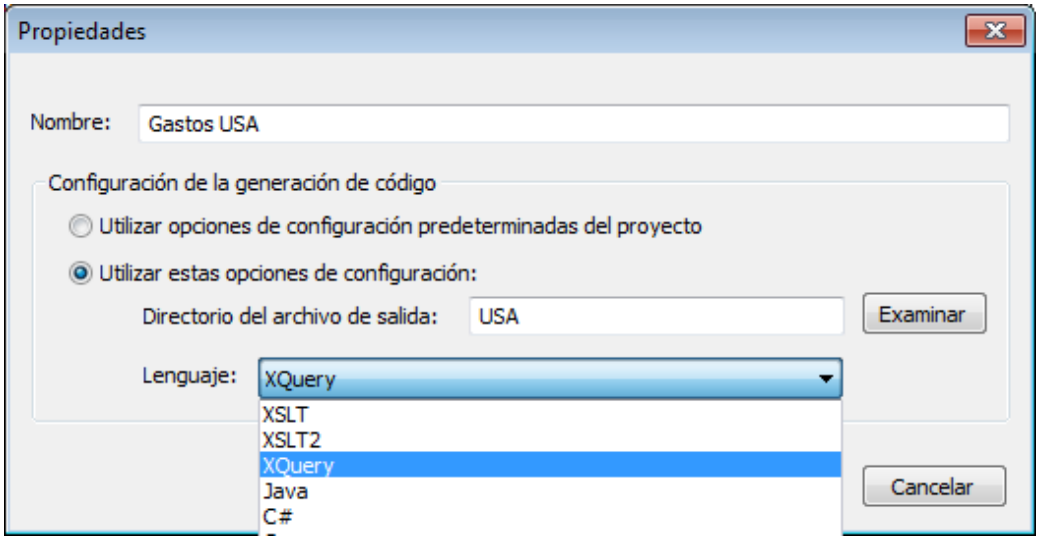
Opciones de proyecto

Una vez creado el proyecto, puede añadir archivos y definir una estructura de carpetas dentro del proyecto. Para ello use los comandos del menú **Proyecto** o del menú contextual que aparece al hacer clic con el botón secundario en el nombre del proyecto en la ventana Proyecto.

Agregar archivos al proyecto... abre el cuadro de diálogo Abrir de Windows, donde puede buscar las asignaciones de MapForce (archivos `*.mfd`) que desea añadir al proyecto.

Agregar archivo activo al proyecto añade al proyecto la asignación que está activa en la ventana de asignación. Si el archivo ya forma parte del proyecto, aparece un mensaje de advertencia.

Crear carpeta abre el cuadro de diálogo **Propiedades**, donde puede poner un nombre a la carpeta y elegir si se usan las opciones predeterminadas del proyecto o definir opciones de configuración nuevas para la carpeta (el directorio del archivo de salida y el lenguaje). Cuando termine haga clic en **Aceptar**.



Para agregar asignaciones al proyecto:
Tiene dos opciones:

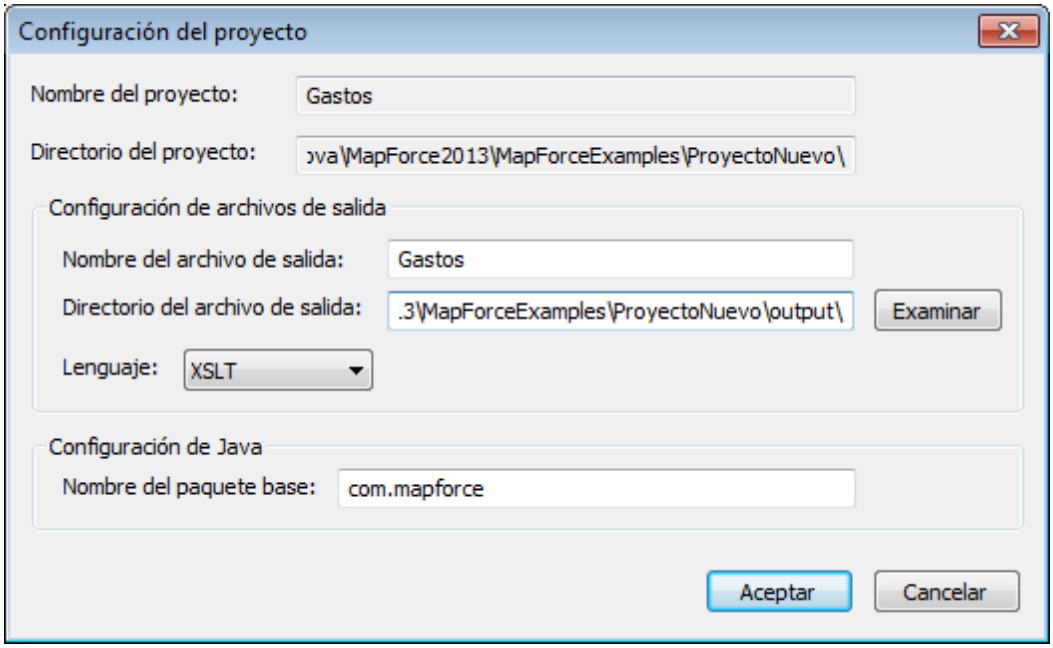
- Seleccione **Proyecto | Agregar archivo activo al proyecto**.
  El archivo activo se añade al proyecto. El nombre de la asignación aparece en la ventana Proyectos, debajo del nombre del proyecto.
- O seleccione **Proyecto | Agregar archivos al proyecto**.
  En el cuadro de diálogo que aparece puede buscar los archivos que desea añadir al proyecto.

Para quitar una asignación del proyecto:
Tiene dos opciones:

- En la ventana Proyectos haga clic con el botón secundario en el icono de la asignación y seleccione **Eliminar** en el menú contextual.
- En la ventana Proyectos haga clic en el icono de la asignación y pulse la tecla **Supr**.

Configuración del proyecto
Para abrir el cuadro de diálogo "Configuración del proyecto" haga clic con el botón secundario en el nombre del proyecto en la ventana Proyectos. Después seleccione **Propiedades** en el menú contextual o **Proyecto | Propiedades**.

 © 2014 Altova GmbH

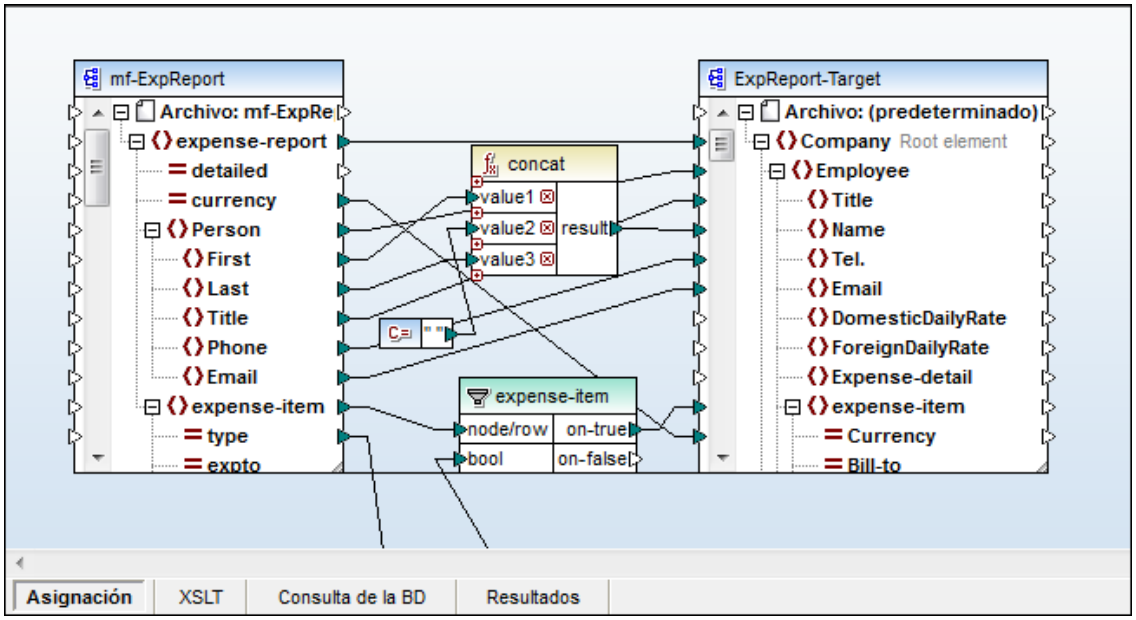


El *Directorio del proyecto* es la carpeta donde se guarda el archivo de proyecto **`*.mfp`**.

En este cuadro de diálogo puede cambiar el nombre del archivo de salida, el directorio del archivo de salida y el lenguaje de salida (recuadro de opciones *Configuración de archivos de salida).* También puede definir el nombre del paquete base en el recuadro de opciones *Configuración de Java.* Tenga en cuenta que el nombre del proyecto y el directorio del proyecto no se pueden cambiar una vez creado el proyecto.

7.3 Panel Asignación

El panel **Asignación** es el área de trabajo de MapForce donde se crean las asignaciones de datos.



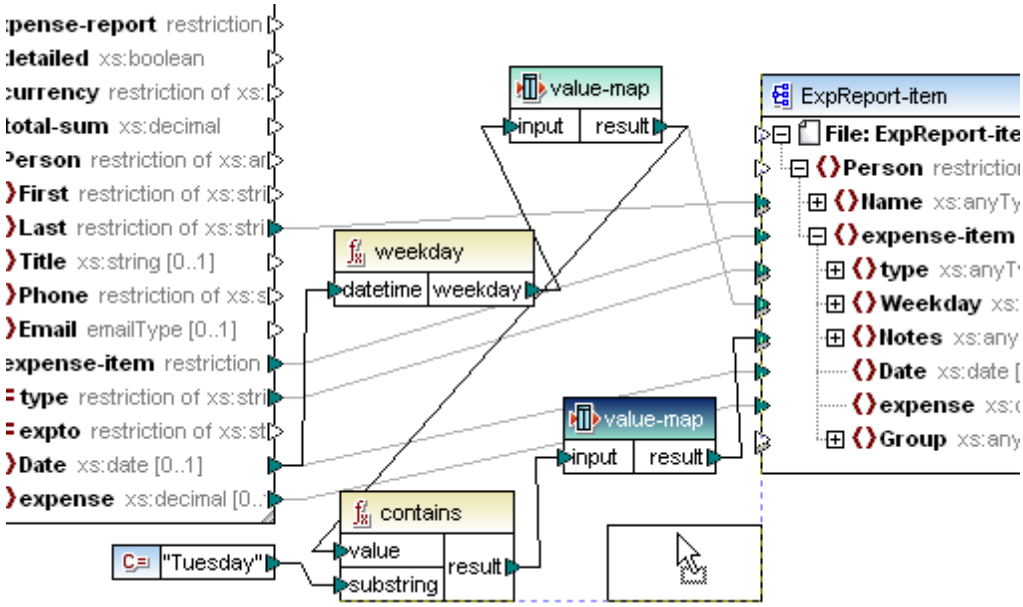
El panel **Asignación** muestra los elementos gráficos utilizados para crear la asignación (transformación) entre los componentes. Los conectores conectan los iconos de entrada y salida de cada elemento del esquema. Los elementos del esquema pueden ser atributos o elementos.

Alineación de componentes: líneas de ajuste

Cuando mueva los componentes por el panel **Asignación**, aparecen líneas de guía de alineación automática que ayudan a alinear el componente con los demás componentes de la vista. Esta opción se puede habilitar/deshabilitar en **Herramientas | Opciones | General.**

En la imagen siguiente puede ver que se está moviendo el componente **`value-map`** de la parte inferior. Las líneas de guía muestran que este componente se puede alinear con la función **`contains`** y con el componente **`ExpReport-item`**.

 © 2014 Altova GmbH


pense-report restriction
detailed xs:boolean
currency restriction of xs:
total-sum xs:decimal
Person restriction of xs:ar
First restriction of xs:stri
Last restriction of xs:stri
Title xs:string [0..1]
Phone restriction of xs:s
Email emailType [0..1]
expense-item restriction
type restriction of xs:stri
expto restriction of xs:st
Date xs:date [0..1]
expense xs:decimal [0..
value-map
input
result
weekday
datetime
weekday
value-map
input
result
contains
value
substring
result
"Tuesday"
ExpReport-item
File: ExpReport-ite
Person restrictio
Name xs:anyTy
expense-item
type xs:anyT
Weekday xs:
Notes xs:any
Date xs:date [
expense xs:
Group xs:any

7.4 Panel XSLT / XSLT2 / XQuery

El panel **XSLT / XSLT2 / XQuery** ofrece una vista previa de la transformación en el lenguaje seleccionado.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <!--
3   This file was generated by Altova MapForce 2013r2
4
5   YOU SHOULD NOT MODIFY THIS FILE, BECAUSE IT WILL BE
6   OVERWRITTEN WHEN YOU RE-RUN CODE GENERATION.
7
8   Refer to the Altova MapForce Documentation for further details.
9   http://www.altova.com/mapforce
10  -->
11  <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:xs="
    http://www.w3.org/2001/XMLSchema" exclude-result-prefixes="xs">
12    <xsl:output method="xml" encoding="UTF-8" indent="yes"/>
13    <xsl:template match="/">
14      <Company xmlns="http://my-company.com/namespace">
15        <xsl:attribute name="xsi:schemaLocation" namespace="http://www.w3.org/2001/XMLSchema-instance">
    http://my-company.com/namespace
    C:/Documents/Altova/MapForce2013/MapForceExamples/Tutorial/ExpReport-Target.xsd</xsl:attribute>
16        <xsl:for-each select="*[local-name()='expense-report' and namespace-uri()='']">
17          <xsl:variable name="var4_cur" select="."/>
18          <xsl:variable name="var1_resultof_first" select="*[local-name()='Person' and namespace-uri()='']"/>
19          <Employee>
20            <Title>
21              <xsl:value-of select="string($var1_resultof_first/*[local-name()='Title' and namespace-uri()=''])"/>
```

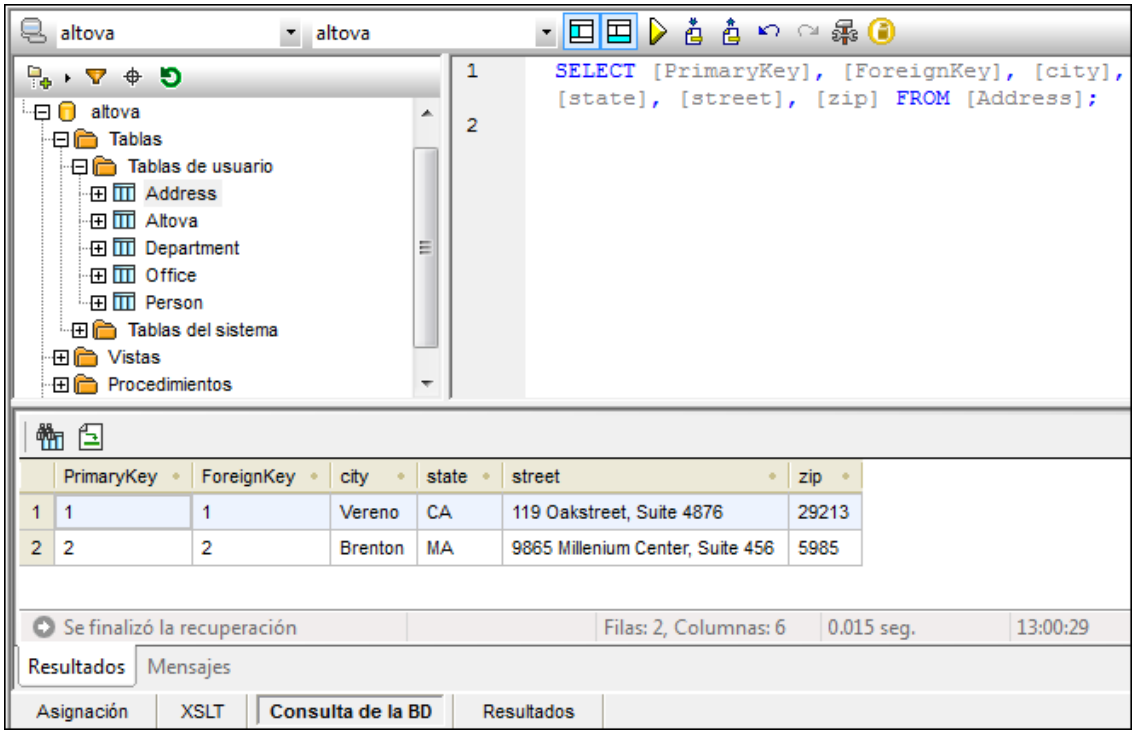
Asignación | XSLT | Consulta de la BD | Resultados

Nota: si desea cambiar el lenguaje de salida, primero debe volver al panel **Asignación** y seleccionar otro lenguaje en la barra de herramientas o en el menú **Resultados**.

© 2014 Altova GmbH

7.5 Panel Consulta de la BD

El panel **Consulta de la BD** sirve para consultar todas las bases de datos compatibles. Las consultas/acciones definidas en este panel son independientes del resto de paneles y vistas de MapForce y no se guardan en el archivo `*.mfd`.



En el panel **Consulta de la BD** puede realizar **varias consultas de BD** por asignación. También puede tener varias conexiones de BD activas al mismo tiempo en cada panel de consulta.

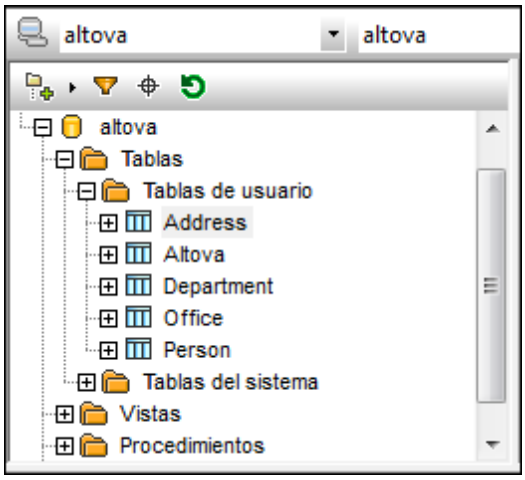
El panel **Consulta de la BD** tiene cuatro partes:

- La ventana Explorador, situada a la izquierda, ofrece información sobre la conexión y muestra las tablas de la BD
- La ventana Editor SQL, situada a la derecha del Explorador, sirve para escribir y ejecutar consultas SQL
- La ventana Resultados, situada en la parte inferior, muestra los resultados de la consulta en forma de tabla
- La ventana Messages, situada también en la parte inferior, muestra mensajes de error y advertencia

La barra superior del panel **Consulta de la BD** contiene los controles de conexión, que sirven para definir las BD de trabajo y los esquemas de BD.

7.5.1 Ventana Explorador

Por cada conexión de BD, la ventana Explorador ofrece un resumen de los objetos disponibles en la BD. Por ejemplo incluye datos sobre las restricciones de la BD como, por ejemplo, las columnas de clave primaria o foránea. En las bases de datos IBM DB2 versión 9, el Explorador también muestra en una carpeta todos los esquemas XML registrados en la BD.



Puede personalizar la ventana Explorador para:

- cambiar la **disposición** en que se presentan los objetos de la BD
- **buscar** objetos concretos en la BD usando el localizador de objetos
- **filtrar** los objetos que se presentan en la vista
- **actualizar** el objeto raíz del origen de datos activo

Disposición de las carpetas
La ventana Explorador puede presentar los objetos de la BD de varias formas diferentes:

- **En carpetas**: los objetos de la BD se agrupan en carpetas dependiendo del tipo de objeto y se presentan en forma de árbol jerárquico. Esta es la opción predeterminada.
- **Sin esquemas**: similar a la **Disposición en carpetas**. La diferencia es que en la disposición sin esquemas no hay carpetas para los esquemas de la BD.
- **Sin carpetas**: los objetos de la BD se presentan en un estructura jerárquica pero sin usar carpetas.
- **Disposición nivelada**: los objetos se dividen según el tipo en el primer nivel jerárquico. Por ejemplo, en lugar de mostrar todas las columnas en una tabla, las columnas se muestran en una carpeta diferente llamada *Columnas*.
- **Dependencias de tablas**: las tablas se agrupan según su relación con otras tablas. Las categorías son: tablas con claves foráneas, tablas utilizadas por claves foráneas, tablas con referencias desde claves foráneas y tablas que no tienen relación con ninguna otra tabla.

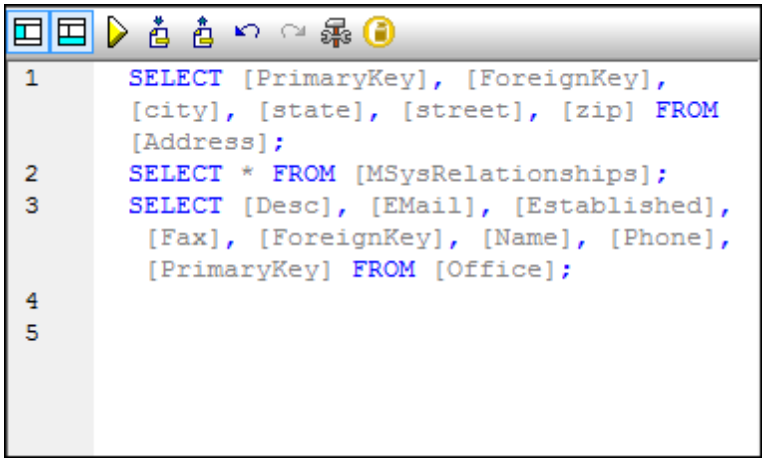
La disposición predeterminada **En carpetas** muestra los objetos de la BD de forma jerárquica. Si hace clic con el botón secundario en un objeto de la ventana Explorador, aparece un menú contextual con diferentes opciones, dependiendo del tipo de objeto seleccionado.

 © 2014 Altova GmbH

7.5.2 Editor SQL

La ventana Editor SQL sirve para escribir y ejecutar instrucciones SQL. Ofrece las siguientes funciones:

- Generación automática de instrucciones SQL (arrastrando objetos desde el Explorador hasta el Editor SQL)
- Finalización automática de instrucciones SQL mientras se escriben instrucciones SQL
- Definición de regiones
- Inserción de comentarios



La barra de herramientas del Editor SQL contiene estos iconos:

Mostrar/ocultar el explorador: habilita/deshabilita la ventana Explorador.

Mostrar/ocultar resultados: habilita/deshabilita la ventana Resultados.

Ejecutar (F5): al hacer clic en este icono se ejecutan las instrucciones SQL seleccionadas en el Editor SQL. Si existen varias instrucciones y no hay ninguna seleccionada, se ejecutan todas las instrucciones.

Deshacer: permite deshacer los cambios realizados en el Editor SQL un número ilimitado de veces.

Rehacer: permite rehacer las operaciones deshechas previamente. Puede recorrer el historial de operaciones deshacer con este comando.

Importar archivo SQL: abre un archivo SQL en el Editor SQL. Después se puede ejecutar el archivo SQL.

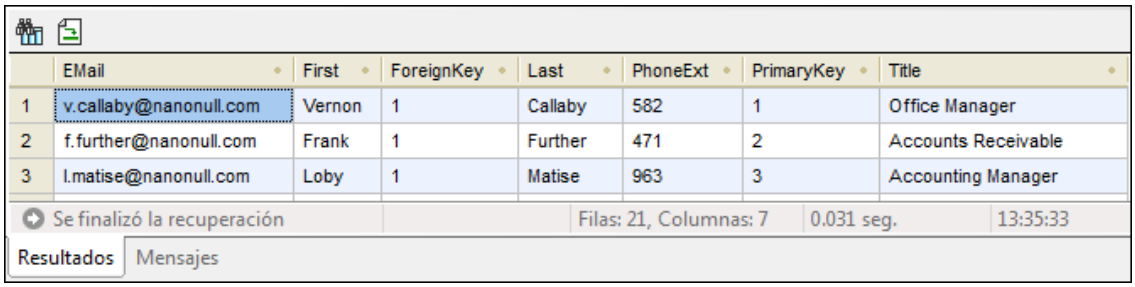
Exportar archivo SQL: guarda las consultas SQL para poder usarlas más tarde.

Abrir en DatabaseSpy: inicia la herramienta de Altova DatabaseSpy y abre el script en el Editor SQL de DatabaseSpy.

Opciones: abre el cuadro de diálogo "Opciones" donde puede definir las opciones de configuración del Editor SQL.

7.5.3 Pestaña Resultados

La pestaña *Resultados* del Editor SQL muestra el conjunto de registros recuperado por la consulta de la BD.



	EMail	First	ForeignKey	Last	PhoneExt	PrimaryKey	Title
1	v.callaby@nanonull.com	Vernon	1	Callaby	582	1	Office Manager
2	f.further@nanonull.com	Frank	1	Further	471	2	Accounts Receivable
3	l.matise@nanonull.com	Loby	1	Matise	963	3	Accounting Manager

Se finalizó la recuperación | Filas: 21, Columnas: 7 | 0.031 seg. | 13:35:33

Resultados | Mensajes

Opciones de la barra de herramientas (en modo de recuperación)
La pestaña *Resultados* tiene una barra de herramientas que permite pasar de los resultados a las instrucciones SQL y recuperar datos de la BD con gran facilidad.

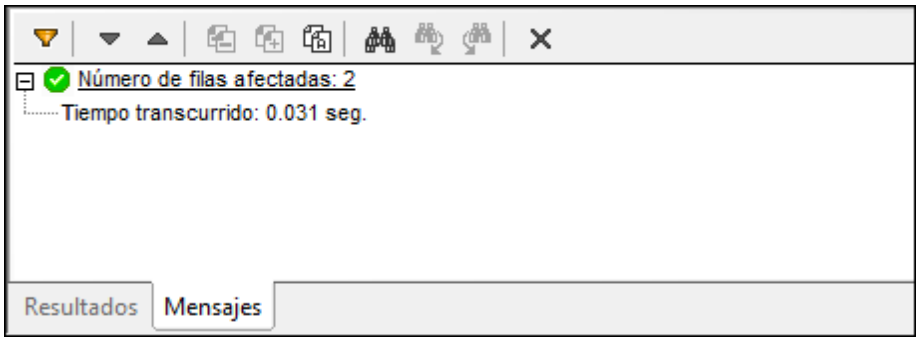
Buscar: busca una cadena de texto en la pestaña *Resultados*. Puede continuar la búsqueda pulsando **F3**.

Ir a la instrucción: resalta en el editor SQL el grupo de instrucciones SQL que produjo el resultado seleccionado.

© 2014 Altova GmbH

7.5.4 Pestaña Mensajes

La pestaña *Mensajes* ofrece información sobre las instrucciones SQL ejecutadas y muestra mensajes de error y advertencia.



Puede personalizar la vista de la pestaña *Mensajes* usando la función de filtrado. También puede pasar de un mensaje a otro con los botones **Siguiente** y **Anterior**. La pestaña *Mensajes* también tiene una función de búsqueda y una opción para copiar el texto en el portapapeles.

Opciones de la barra de herramientas
La pestaña *Mensajes* tiene una barra de herramientas que permite pasar de un mensaje a otro y filtrar los mensaje que aparecen en la pestaña.

Filtrar: al hacer clic en este icono aparece un menú donde puede seleccionar que tipo de mensaje se muestra (**Resumen**, **Operación correcta**, **Advertencia**, **Error**). También hay una opción para mostrar/ocultar 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 repetición anterior del término de búsqueda indicado en el cuadro de diálogo Buscar.

Buscar siguiente: pasa a la siguiente repetición del término de búsqueda indicado en el cuadro de diálogo Buscar.

Borrar: borra todos los mensajes de la pestaña *Mensajes*.

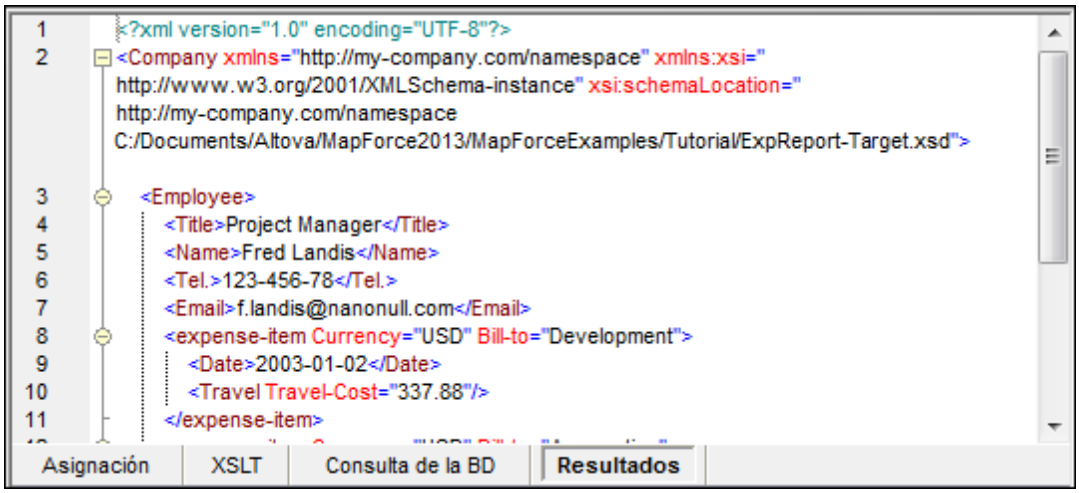
Nota: estos comandos también están disponibles en el menú contextual que aparece cuando se hace clic con el botón secundario junto a un mensaje.

7.6 Panel Resultados / motor de ejecución BUILTIN

El resultado de una asignación se presenta inmediatamente en el panel **Resultados**, usando el motor XSLT o XQuery de Altova, dependiendo del lenguaje seleccionado.

Si selecciona uno de los lenguajes de programación (Java, C++ o C#) o la opción **BUILTIN**, el motor de ejecución integrado genera el resultado a partir de los componentes de entrada: archivos o esquemas XML, archivos de texto, bases de datos, etc. El resultado que aparece en el panel **Resultados** es el mismo resultado que se obtiene si se genera, compila y ejecuta el código Java, C++ o C#.

Nota: el resultado generado por el motor de ejecución integrado **BUILTIN** es una transformación instantánea de los datos de BD, texto o EDI sin necesidad de generar ni compilar código de programa. Recomendamos que use el panel Resultados hasta alcanzar los resultados deseados y después genere el código de programa.



Dependiendo del tipo de componente de **destino**, el panel **Resultados** puede mostrar diferentes cosas:

- **El componente de destino es un documento o esquema XML**
  La imagen siguiente muestra los resultados de la asignación `DB_CompletePO.mfd`, disponible en la carpeta `...\MapForceExamples`. En esta asignación se usa un esquema/documento XML y una BD como componentes de origen.

 © 2014 Altova GmbH

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <CompletePO xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     <Customer>
4       <Number>3</Number>
5       <FirstName>Ted</FirstName>
6       <LastName>Little</LastName>
7       <Address>
13    </Customer>
14    <LineItems>
15      <LineItem>
16        <Article>
17          <Number>3</Number>
18          <Name>Pants</Name>
19          <SinglePrice>34</SinglePrice>
20          <Amount>5</Amount>
21          <Price>170</Price>
22        </Article>
23      </LineItem>
24      <LineItem>
25        <Article>
32      </LineItem>
33    </LineItems>
34  </CompletePO>
```

El archivo XML resultante se puede guardar haciendo clic en el icono **Guardar resultado generado** . El archivo XML resultante también se puede validar con el esquema referenciado haciendo clic en el icono **Validar archivo de salida** .

- **El componente de destino es una BD**
  En este caso se ejecuta la asignación a la BD de destino, teniendo en cuenta las acciones de tabla definidas.

```
/*
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:\Documents\Altova\MapForce2012\MapForceExamples\CustomersTarget.mdb;
Provider=Microsoft.Jet.OLEDB.4.0
*/

SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey]
FROM [Altova]
-->>> %PrimaryKey1%

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', '%PrimaryKey1%')
```

Al hacer clic en el icono **Ejecutar script SQL** (en la barra de herramientas del panel **Resultados**) se ejecuta la instrucción SQL SELECT y se presenta un informe con todas las acciones de BD:

– Instrucciones SQL que se ejecutaron en la BD de destino
  SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey]
  FROM [Altova]

```
-->>> OK. Una fila o más.
```

– Acciones de tabla varias, si se produjeron

```
INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Organization Chart', 1)
-->>> OK. 1 fila.
```

– Resultados de cada instrucción SQL

`-->>> OK. n fila(s).` se ejecutaron correctamente o el mensaje `Error en la ejecución` y detalles del error.

```
SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey]
FROM [Altova]
-->>> OK. Una fila o más.

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', 2)
-->>> OK. 1 fila(s).
```

- **El componente de destino es un archivo de texto**
  El motor de ejecución integrado puede generar resultados para estos archivos de texto:
  – Archivos CSV (valores separados por comas) y para otros delimitadores
  – Archivos FLF (archivos de longitud fija)

```
1  "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Albert,Aldrich,582,A.Ald
2  "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Bert,Bander,471,b.bander
3  "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Clive,Clovis,963,c.clovi
4  "Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,5588339,Dave,Durne
5  "Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,5588339,Eve,Ellas,
6  "Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,Fred,Fortunas,95
7  "Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,Gerry,Gundall,65
```

  – Archivos UN/EDIFACT y ANSI X12
  – Una plantilla de FlexText que define la estructura y el contenido que se debe usar como componente de destino

- **El componente de destino es un archivo OOXML Excel 2007 (o superior)**
  Es necesario tener instalado Microsoft Excel 2007 (o superior) en el equipo si desea ver los resultados de documentos Excel 2007 (o superior) en el panel **Resultados**. Si usa una versión anterior de Microsoft Excel o no tiene Excel instalado en el equipo, puede usar archivos Excel (o superior) en sus asignaciones, lo que no puede hacer es ver los resultados en el panel **Resultados**, pero sí puede guardar los resultados con el comando **Guardar archivo de salida**.

	A	B	C	D
1	Alex	Martin		
2	George	Hammer		
3	Jessica	Bander		
4	Lui	King		
5				
6				

Altova

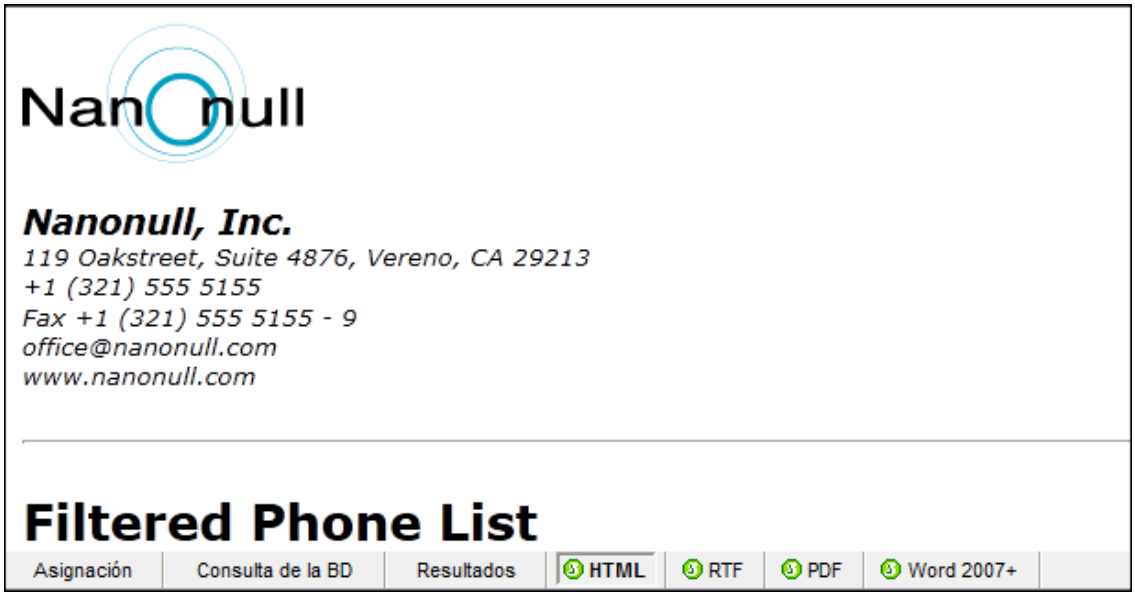
Teclas de acceso rápido para el panel Resultados (del teclado y del teclado numérico)

CTRL y "+" > acercarse (agrandar el texto)
CTRL y "-" > alejarse (reducir el tamaño del texto)
CTRL y "0" > restaura el nivel de zoom al nivel estándar
CTRL y rueda del mouse hacia adelante/atrás > acercarse/alejarse

 © 2014 Altova GmbH

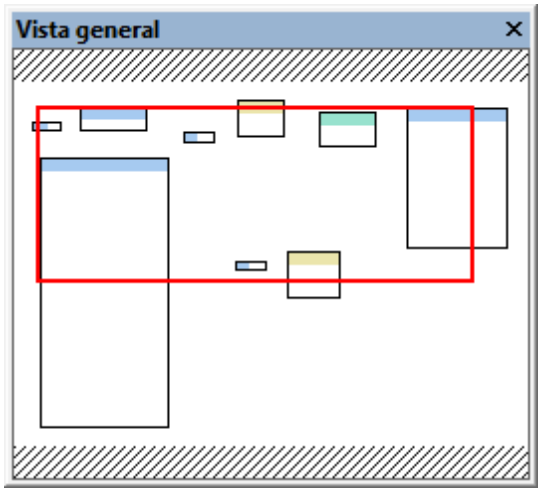
7.7 Paneles relacionados con StyleVision

Si el componente de destino tiene asociado un archivo SPS (StyleVision Power Stylesheet), los paneles **HTML**, **RTF**, **PDF** y **Word 2007+** muestran los datos del componente de destino como documentos HTML, RTF, PDF y Word 2007+ respectivamente.



7.8 Ventana Vista general

La ventana Vista general es un panel de navegación que sirve para desplazarse por asignaciones de gran tamaño. El rectángulo rojo indica el área de la asignación que está visible en el panel **Asignación**. Puede arrastrar este rectángulo por la ventana Vista general para ajustar la parte visible de la asignación en el panel **Asignación**.

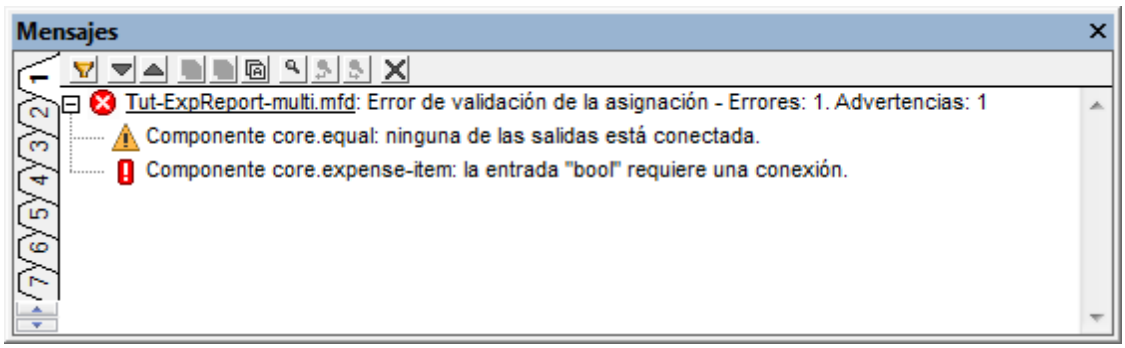


Cuando se hace clic en la ventana Vista general, el recuadro rojo muestra la parte central de la asignación.

© 2014 Altova GmbH

7.9 Ventana Mensajes

La ventana Mensajes muestra mensajes, errores y advertencias. Estos mensajes se generan al hacer clic en la pestaña **Resultados** o al validar la asignación.



Altova MapForce 2014

Funcionamiento general

8 Funcionamiento general

Este apartado describe el funcionamiento de varias tareas que se pueden realizar en MapForce:

- Mover y conservar conectores
- Cómo ocuparse de elementos desaparecidos
- Usar el motor de transformación integrado
- Validar la asignación y el resultado de la asignación
- Compilar una asignación de MapForce
- Implementar una asignación de MapForce
- Parámetros de la línea de comandos
- Usar los archivos de catálogo
- Documentar proyectos de asignación de datos
- Usar las hojas de estilos Power Stylesheets de StyleVision para representar datos

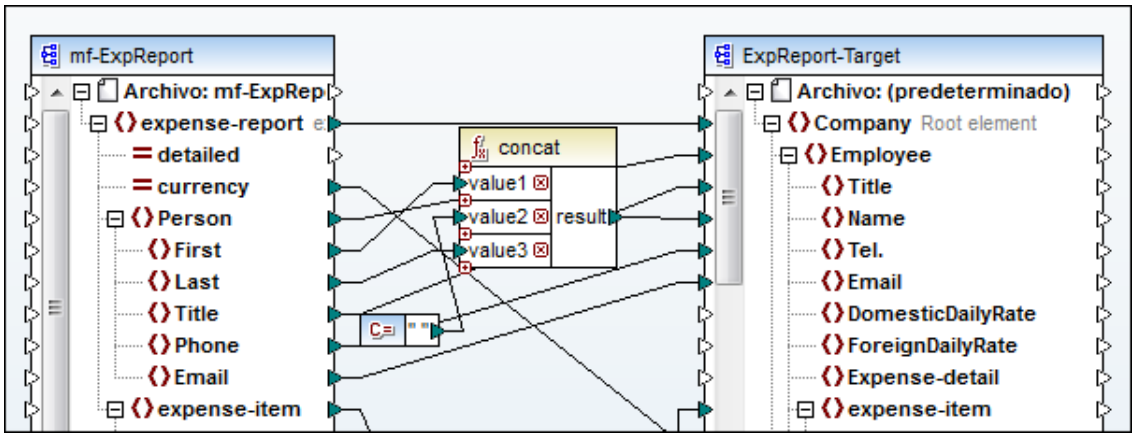
© 2014 Altova GmbH

8.1 Mover y conservar conectores

Cambiar conexiones (y efectos en los conectores secundarios)
Cuando se mueve a otro elemento un conector que tiene conectores secundarios, MapForce automáticamente ajusta los conectores secundarios equivalentes a la nueva posición del conector primario. No debemos confundir esta característica con la opción de conexión automática de secundarios equivalentes porque en este caso se utiliza otra regla para crear las conexiones.

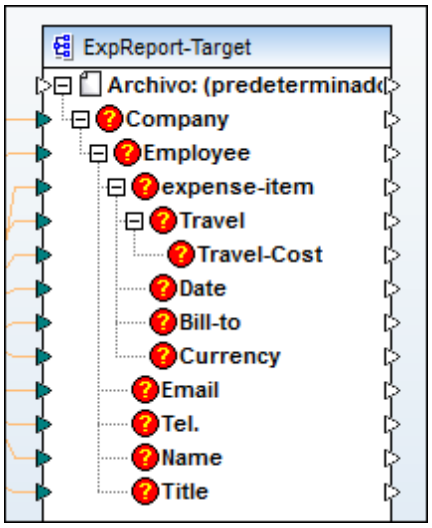
Esta característica es muy práctica si, por ejemplo, tiene una asignación y cambia el elemento raíz del esquema de destino. Esto le obligaría a crear otra vez las conexiones de los elementos secundarios a mano.

Para este ejemplo usamos el archivo `Tut-ExpReport.mfd` guardado en la carpeta `...\MapForceExamples\Tutorial`.



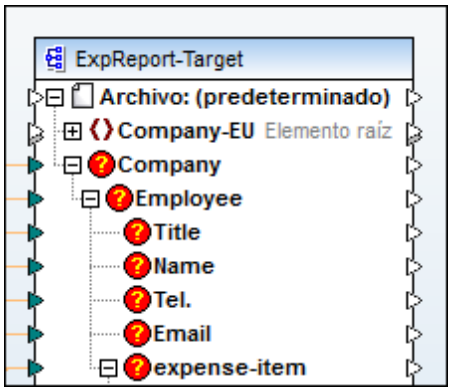
Si por cualquier motivo, el elemento raíz del esquema de destino cambia (a **`Company-EU`**, por ejemplo) recibimos un mensaje avisando de que el archivo ha cambiado.

1. Haga clic en el botón **Volver a cargar** para cargar otra vez el esquema que se actualizó. Ahora aparecen señalados muchos nodos que faltan como consecuencia del cambio de elemento raíz.



2. Haga clic con el botón secundario en el título del componente de destino y seleccione **Cambiar de elemento raíz** en el menú contextual.

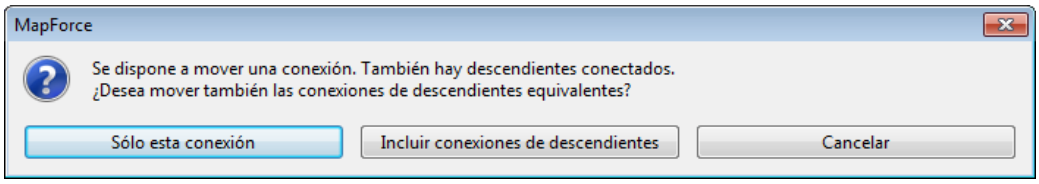
3. Seleccione el nuevo elemento raíz (**`Company-EU`**) y haga clic en **Aceptar**.



El elemento raíz **`Company-EU`** ahora aparece al principio del componente.

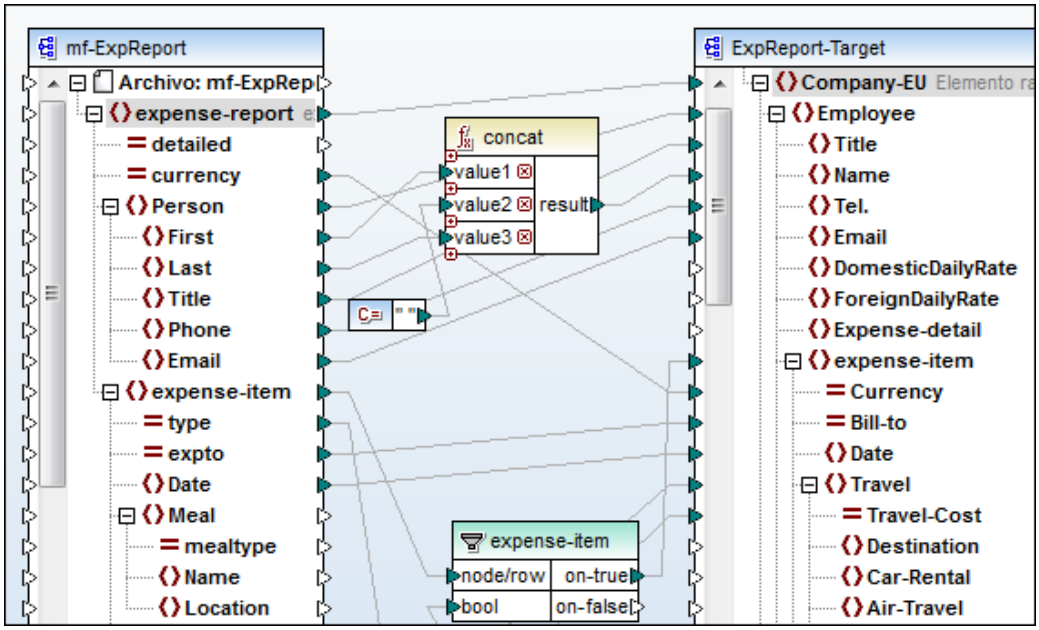
4. Haga clic en el conector del elemento **`Company`** y arrástrelo hasta el nuevo elemento raíz **`Company-EU`**.

Aparece un mensaje preguntando qué conectores desea mover.



5. Haga clic en el botón **Incluir conexiones de descendientes** si desea actualizar también los conectores secundarios.
Los nodos que faltaban (que aparecían con un signo de interrogación) desaparecen y todos los conectores se asignan a los elementos secundarios equivalentes del elemento raíz nuevo.

 © 2014 Altova GmbH



Nota importante:

Si el elemento/nodo de **destino** de la asignación tiene el mismo nombre (que el nodo de origen) pero está en un espacio de nombres distinto, el mensaje incluye otro botón más, llamado **Incluir descendientes y asignar espacio de nombres**.

Al hacer clic en este botón, los conectores secundarios que tienen el mismo espacio de nombres que el nodo primario de origen se mueven a los mismos nodos secundarios del nodo que tiene un espacio de nombres diferente. Es decir, si la única diferencia entre los nodos primarios es el espacio de nombres, entonces esa es la única diferencia permitida entre los nodos secundarios si se asignan automáticamente.

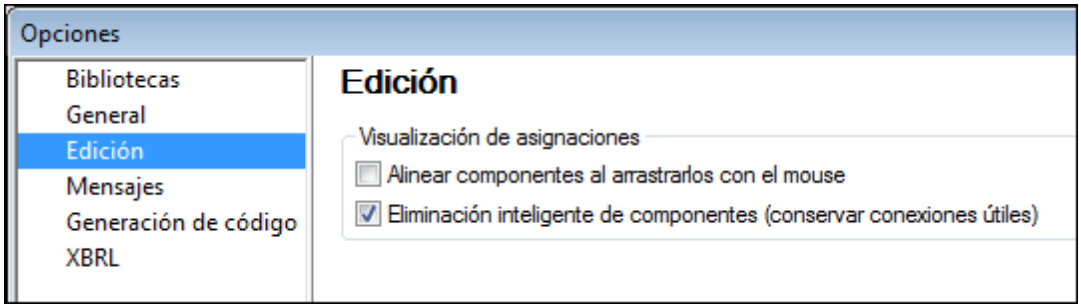
También puede cambiar de elemento raíz haciendo clic en el componente y seleccionando la opción de menú **Componente | Cambiar de elemento raíz**.

Conservar conectores tras la eliminación de componentes

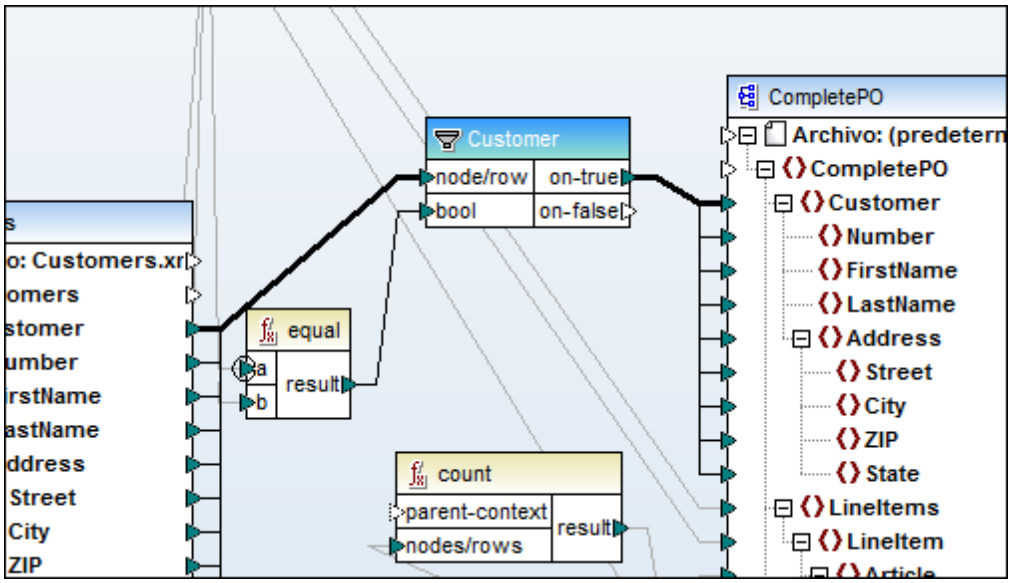
Ahora puede decidir qué ocurre cuando se elimina un componente que tiene varias conexiones (secundarias) con otro componente, como por ejemplo un filtro o componente de ordenación. Esta opción es muy práctica porque permite conservar todos los conectores secundarios, en lugar de tener que restaurarlos a mano uno a uno.

Tras eliminar el componente podrá elegir entre conservar/restablecer las conexiones secundarias y eliminar todos los conectores secundarios inmediatamente.

Para ver cuál es la acción predeterminada actual, haga clic en **Herramientas | Opciones**, abra la pestaña *Edición* del cuadro de diálogo "Opciones" y compruebe el estado de la casilla *Eliminación inteligente de componentes (conservar conexiones útiles).* Esta casilla está **desactivada** por defecto (es decir, las conexiones siempre se restablecen tras la eliminación del componente).

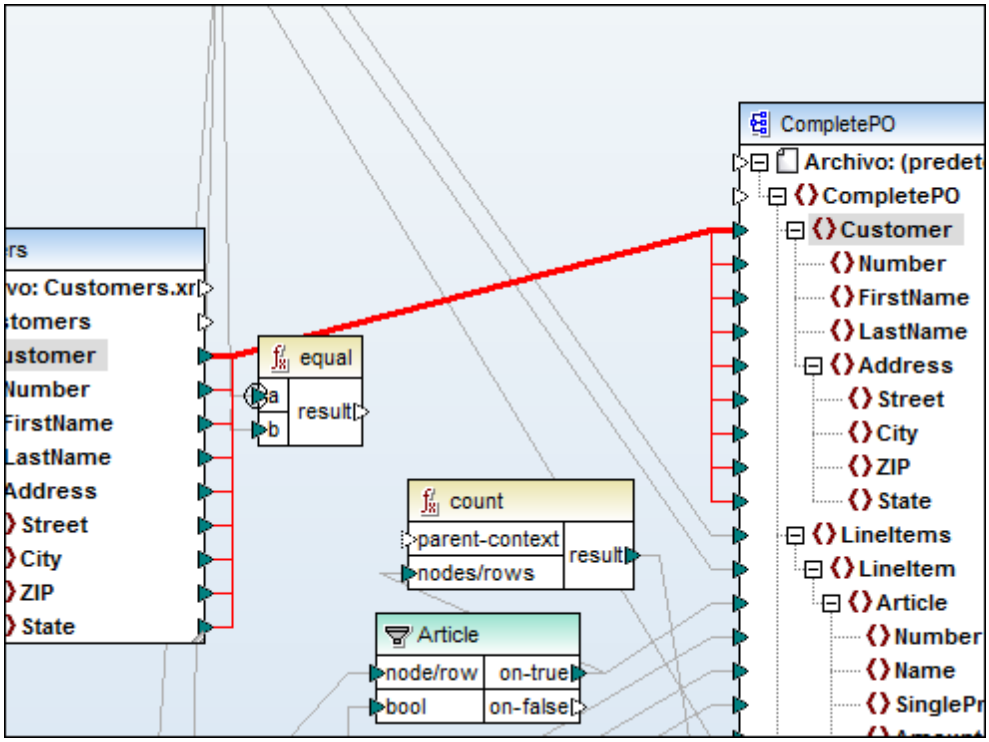


Por ejemplo, tome la asignación `CompletePO.mfd` de la carpeta `...\MapForceExamples` y active la casilla *Eliminación inteligente de componentes (conservar conexiones útiles).* Observe que el filtro **`Customer`** es una conexión de copia total que tiene muchos elementos secundarios conectados.



Si eliminamos el filtro **`Customer`** aparece un aviso preguntando si de verdad queremos eliminarlo. Hacemos clic en **Sí** y el filtro se elimina. Sin embargo, sus conectores secundarios se conservan.

 © 2014 Altova GmbH



Observe que el resto de conectores siguen seleccionados (en color rojo). Si también quiere eliminar estos conectores, pulse la tecla **Supr**.

Si hace clic en cualquier parte del área de trabajo, los conectores dejan de estar seleccionados.

Si, por el contrario, la casilla *Eliminación inteligente de componentes (conservar conexiones útiles)* estuviese **desactivada**, al eliminar el filtro se habrían eliminado también todos los conectores secundarios.

Nota: si un componente de filtrado tiene conectada tanto la salida `on-true` como la salida `on-false`, entonces se conservan los conectores de ambas salidas.

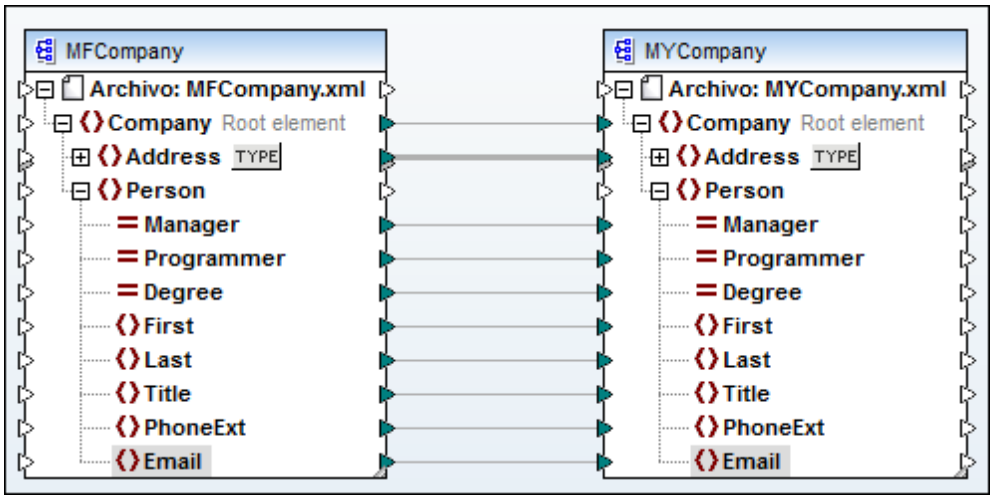
8.2 Elementos desaparecidos

Es posible que, pasado un tiempo, la estructura de uno de los componentes de la asignación cambie. Por ejemplo, puede que se eliminen elementos/atributos del esquema XML o que se añadan elementos nuevos. MapForce usa marcadores de posición para conservar todos los conectores (y otros datos de conexión importantes) cuando se eliminan elementos.

Ejemplo
Para este ejemplo usamos el archivo de esquema `MFCompany.xsd`. Primero creamos una copia de este archivo de esquema y la llamamos `MyCompany.xsd`.

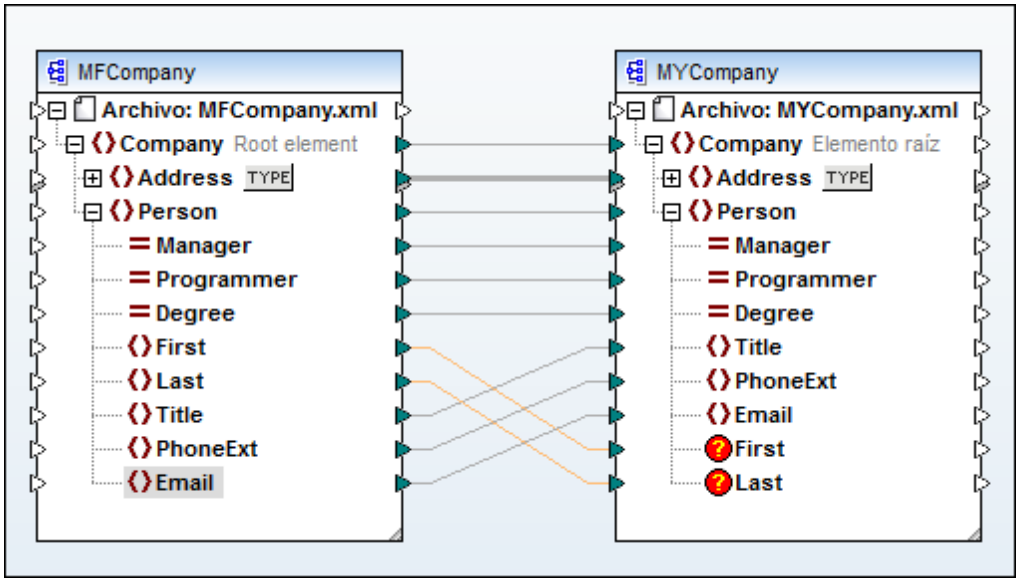
Creamos un componente de origen con el esquema `MFCompany.xsd` y el archivo de instancia `MFCompany.xml`. Después creamos un componente de destino con el esquema `MyCompany.xsd` y el archivo de instancia `MyCompany.xml`.

Creamos un conector entre los elementos **`Company`** de los componentes. Si está activa la opción **Conectar los secundarios equivalentes**, se crean automáticamente conectores entre los elementos secundarios equivalentes de los elementos **`Company`**.



Ahora abrimos `MyCompany.xsd` en XMLSpy y eliminamos los elementos **`First`** y **`Last`** del esquema. Al volver a MapForce, aparece un cuadro de diálogo informándonos de que se realizaron cambios en uno de los archivos. Hacemos clic en **Volver a cargar** y los componentes de la asignación se actualizan.

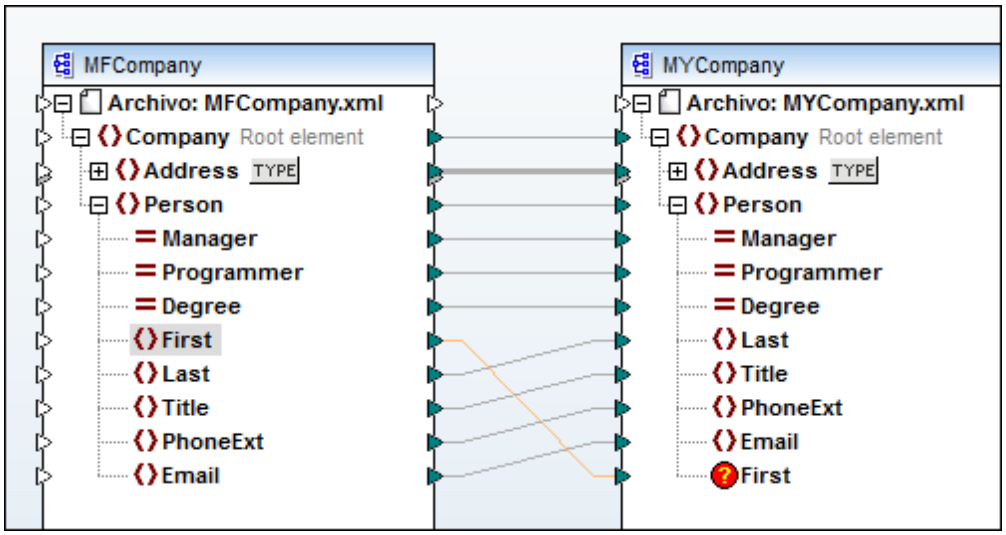
 © 2014 Altova GmbH



Los **elementos** eliminados y sus **conectores** ahora aparecen señalados en color naranja en el componente **MyCompany**. Ahora podemos volver a conectar los conectores a otros elementos o eliminarlos.

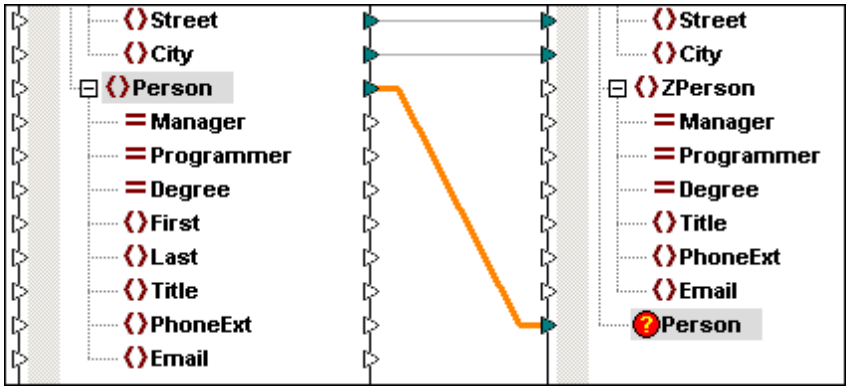
Recuerde que puede ver una vista previa de la asignación (o generar código), pero si lo hace en este momento, en la ventana Mensajes aparecen advertencias. Durante la generación de código y en la vista previa de resultados se ignoran los elementos que faltan.

Si hacemos clic en uno de los conectores resaltados y lo eliminamos, también se elimina del componente el elemento que falta (p. ej. **`Last`** en el componente **MyCompany**).

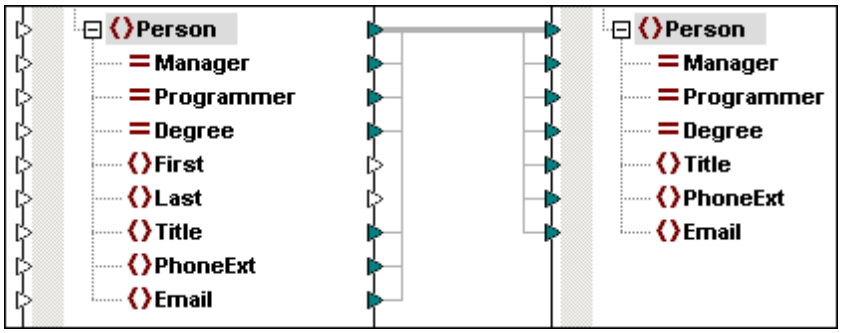


Qué ocurre cuando un elemento cambia de nombre

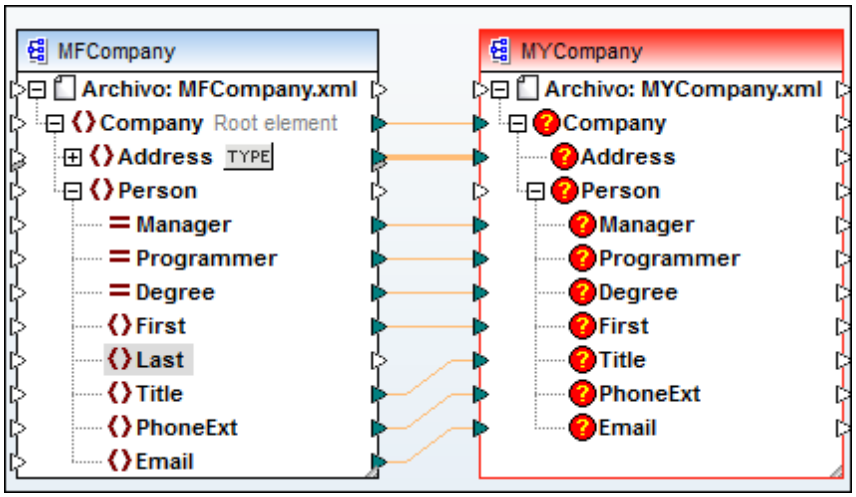
Si se cambia el nombre de un elemento primario (p. ej. imaginemos que ahora **`Person`** se llama **`ZPerson`**), entonces el conector original del elemento primario se conserva y los elementos secundarios y sus conectores se eliminan.



Qué ocurre con las conexiones de copia total cuando falta un elemento
Las conexiones de copia total se tratan igual que las conexiones normales, con la única diferencia de que no se conservan los conectores a los elementos que faltan.



Qué ocurre cuando desaparece el origen de datos o cambia de nombre
Si el origen de datos de un componente (es decir, el esquema, la base de datos, etc.) se elimina o cambia de nombre, entonces aparecen resaltados todos los elementos que contenía. El componente aparece ahora con un marco rojo, lo cual indica que no existe una conexión válida con un esquema o una BD. Esto significa que no se puede obtener una vista previa de los resultados ni generar código.

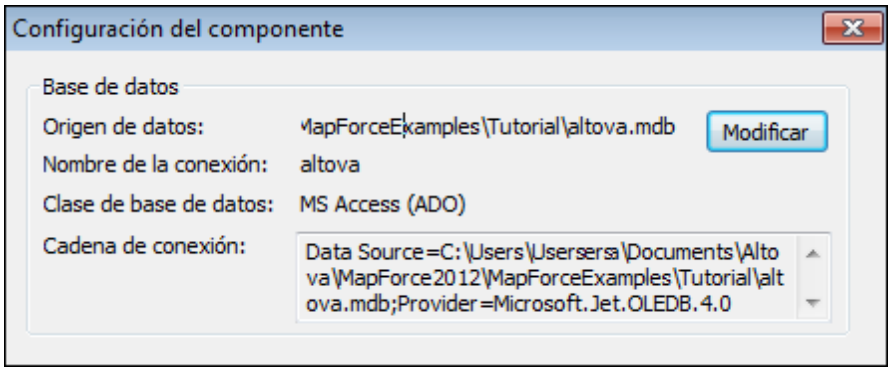


Si pasa el puntero del mouse sobre el componente que aparece en rojo, se abre un mensaje con información.

 © 2014 Altova GmbH

Al hacer doble clic en la barra del título del componente que aparece en rojo se abre el cuadro de diálogo "Configuración del componente". Pulse el botón **Examinar...** del grupo *Archivo de esquema* y seleccione otro esquema o una copia de seguridad del esquema utilizado hasta ahora. Para más información consulte el apartado Componente de la *Referencia del usuario*.

Si el componente es una BD, haga clic en el botón **Modificar** del cuadro de diálogo "Configuración del componente". Seleccione otra BD o elija otras tablas para el componente. MapForce conservará los conectores de las tablas que tengan el mismo nombre.



Todas las conexiones válidas/correctas (y, si es un componente de BD, todos los datos relevantes) se conservan cuando seleccionamos un esquema o una BD que tiene la misma estructura.

8.3 Vista previa de resultados (con el motor integrado)

Gracias al motor de ejecución integrado puede obtener una vista previa inmediata de los resultados y puede guardarlos en un archivo, sin necesidad de generar código de programa ni compilarlo. Para ello basta con hacer clic en la pestaña del panel **Resultados**. En este panel también puede usar el comando **Buscar** para encontrar datos XML o instrucciones SQL.

Con el icono **BUILTIN** puede seleccionar el motor de ejecución integrado como destino de los resultados. El motor de ejecución integrado también sirve para obtener ver una vista previa de los resultados cuando está seleccionado un lenguaje de programación (Java, C#, C++). Tras hacer clic en el icono **BUILTIN** y pasar al panel **Resultados**, se habilitan todas las opciones estándar del panel **Resultados**, es decir los comandos **Validar archivo de salida**, **Guardar el archivo de salida...**, **Guardar todos los archivos de salida...**, etc.

Transmisión de datos por secuencias
El motor de ejecución integrado es el único destino que admite la **transmisión por secuencias** de datos XML, CSV y FLF. Esto significa que los archivos de **salida** pueden tener un tamaño ilimitado. Si el archivo de salida es muy grande, aparece el botón **Cargar más** en la parte inferior del panel **Resultados**. Al hacer clic en este botón, se anexa otro trozo a los datos visibles. El botón **Pretty-print** se activa cuando todo el archivo está cargado en el panel **Resultados**.

Firmas digitales:
En MapForce también puede crear firmas XML digitales para archivos de salida XML y XBRL. Las firmas digitales solamente se pueden generar si elige la opción BUILTIN. Al hacer clic en la pestaña del panel **Resultados** se crea la firma XML del archivo de resultados. Para más información consulte el apartado Firmas digitales.

Vista previa del resultado de la transformación
En el menú **Resultados** (o con los iconos de la barra de herramientas Selección del lenguaje) puede definir el lenguaje que MapForce utiliza para transformar los datos del archivo de entrada.

Estas son las opciones disponibles:

- XSLT
- XSLT2

La vista previa del código XSLT se puede ver en el panel **XSLT/XSLT2** y se genera y guarda con la opción de menú **Archivo | Generar código en | XSLT 1.0** o **XSLT 2.0** respectivamente.

- XQuery

La velocidad de ejecución del código XQuery 1.0 es considerablemente mayor que la del código XSLT 1.0 / 2.0.

- Java
- C#
- C++

 © 2014 Altova GmbH

El código de programa Java, C# o C++ es aún más rápido que XQuery porque se compila antes de ejecutarse.

Vista previa del código de programa

Antes de generar código de programa es recomendable ver una vista previa del resultado del código XSLT/XQuery usando el motor de ejecución integrado.

Para ver una vista previa del resultado del código XSLT /XQuery :

Abra el archivo `*.mfd` en MapForce y después:

1. Haga clic en uno de estos tres botones:
   - En el botón **XSLT** para la vista previa del código XSLT 1.0 generado.
   - En el botón **XSLT2** para la vista previa del código XSLT 2.0 generado.
   - En el botón **XQuery** para la vista previa del código XQuery generado.
2. Cambie al panel **Resultados** ver el resultado de la asignación.

Transformar el archivo XML de origen: DoTransform

Cuando genere código XSLT/XQuery, se genera también un archivo por lotes llamado `DoTransform.bat`. Este archivo se guarda en la carpeta donde se generó el archivo XSLT/XQuery y utiliza RaptorXML para transformar el archivo XML de origen.

Para transformar el archivo XML usando el código XSLT/XQuery generado:

1. Descargue e instale RaptorXML de la página de descargas del sitio web de Altova.
2. Inicie el archivo por lotes `DoTransform.bat` ubicado en la carpeta donde se guardó el código.

Como resultado se genera el archivo de salida `ExpReport-Target.xml` en la carpeta `...\Tutorial`.

Tenga en cuenta que quizás sea necesario añadir la ubicación de la instalación de RaptorXML Server a la variable PATH de las variables de entorno. La documentación de RaptorXML está disponible en el sitio web de Altova.

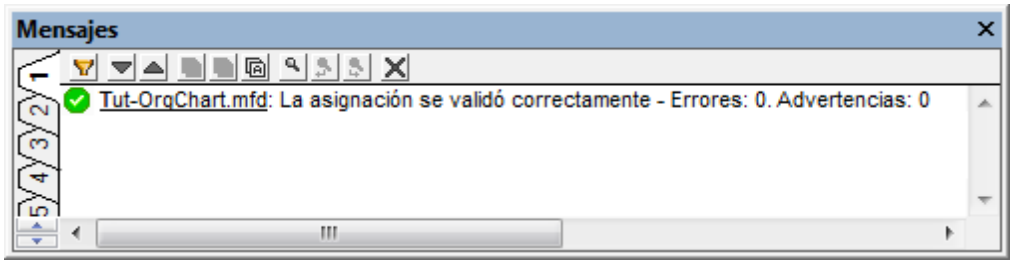
8.4 Validar la asignación y su resultado

No es obligatorio crear conexiones entre todos y cada uno de los componentes y funciones del panel **Asignación**. Este panel es un área de trabajo donde puede colocar todos los componentes que quiera. El código XSLT 1.0, XSLT 2, XQuery, Java, C# o C++ se genera solamente para los componentes que tengan conexiones válidas.

Los componentes que **no estén conectados** no generan mensajes de error ni advertencia.

Los componentes conectados sólo parcialmente pueden generar dos tipos de mensajes:

- Si un **icono de entrada** de una función está sin conectar, se genera un mensaje de error y la transformación se detiene.
- Si el **icono de salida** de una función está sin conectar, se genera una advertencia y el proceso de transformación continúa. El componente infractor y sus datos se omiten y no se asignan al documento de destino.



Si su proyecto contiene varios archivos de asignación, puede usar las distintas pestañas de la ventana Mensajes. Haga clic en una de las pestañas numeradas de la ventana Mensajes y después haga clic en la pestaña del panel **Resultados** de otra asignación. El mensaje de validación aparece ahora en la pestaña seleccionada. El mensaje original de la pestaña 1 se conserva.

Además, la ventana Mensajes incluye una barra de herramientas con iconos para:

- filtrar los mensajes según su tipo (errores o advertencias)
- desplazarse por los mensajes
- copiar el texto de los mensajes en el portapapeles
- buscar una cadena de texto concreta en los mensajes
- borrar los mensajes de la ventana

Para validar una asignación:

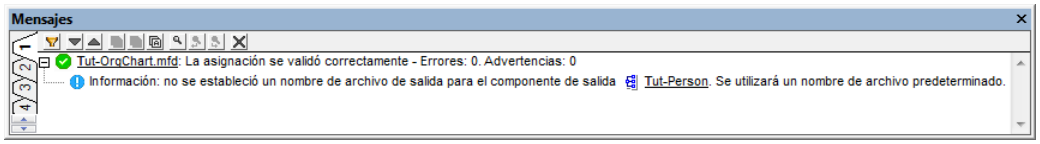
- Haga clic en el icono **Validar asignación** de la barra de herramientas o seleccione la opción de menú **Archivo | Validar asignación**.
  Aparece un mensaje de validación en la ventana Mensajes.

Mensajes de validación

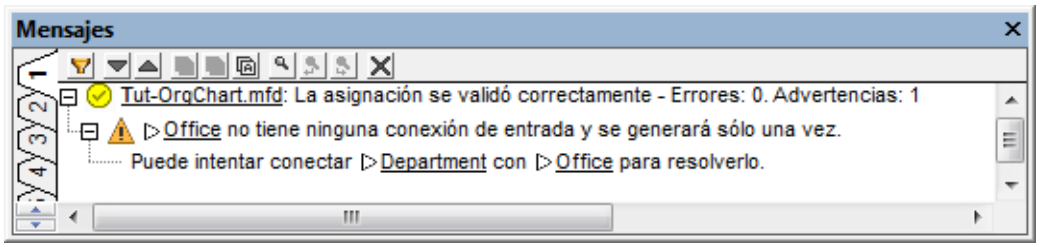
Los mensajes de validación aparecen en la ventana Mensajes e indican si la asignación es válida o no. Esta ventana también ofrece mensajes de error/advertencia e información sobre la asignación. Hay dos tipos de mensajes de validación:

- El mensaje *La asignación se validó correctamente - Errores: 0. Advertencias: n*

 © 2014 Altova GmbH

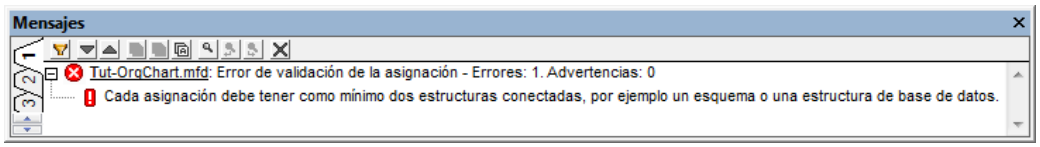


Las **advertencias** alertan de riesgos sin impedir que continúe el proceso de asignación y de vista previa de los resultados de la transformación. Por tanto, una asignación válida tiene cero errores pero puede tener varias advertencias.



- El mensaje *Error de validación de la asignación - Errores: x. Validaciones: y.*

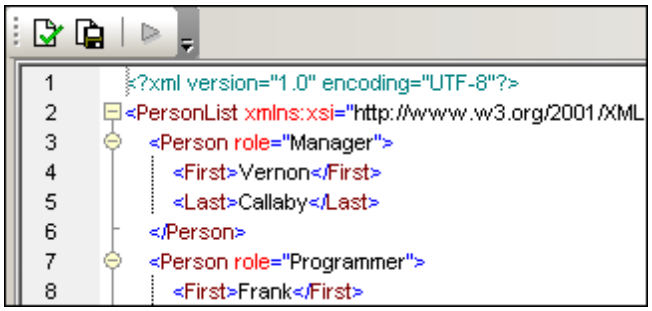
  Los **errores** detienen el proceso de transformación y emiten un mensaje de error. Si existe un error de este tipo no se puede obtener una vista previa de resultados ni se puede generar código XSLT/XQuery. Al hacer clic en el mensaje en la ventana Mensajes, el componente/icono infractor se señala en el panel **Asignación**.



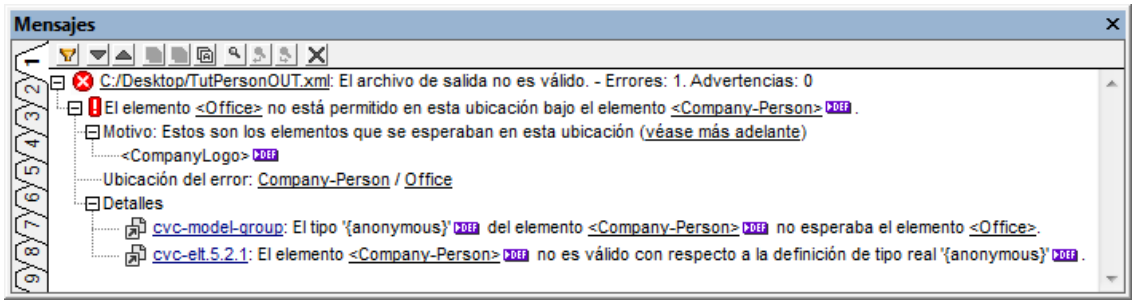
Validar los resultados de una asignación

Al hacer clic en la pestaña del panel **Resultados**, el motor XSLT 1.0/2.0, XQuery o BUILTIN transforma los datos y reproduce los resultados en una vista Texto.

Si los datos están asignados a un esquema XML, el documento XML resultante se puede validar con el esquema subyacente. Si el componente de destino es un archivo EDI, el resultado se puede validar con la especificación EDI. Para más información, consulte el apartado UN/EDIFACT y ANSI X12 como componentes de destino.



El resultado de la validación aparece en la ventana Mensajes. Si los datos son válidos, el mensaje ofrece información sobre los errores detectados (*imagen siguiente*).

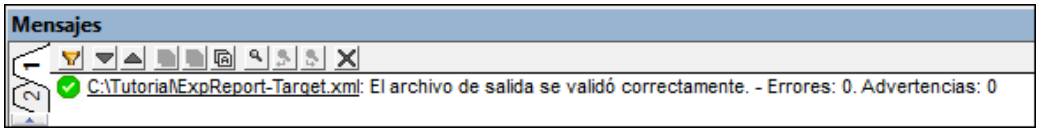


El mensaje de validación contiene algunos hipervínculos que ofrecen más información:

- Al hacer clic en la ruta de acceso del archivo se abre el resultado de la trasformación en el panel **Resultados** de MapForce.
- Al hacer clic en el hipervínculo `<NombreElemento>` se resalta el elemento en el panel **Resultados**.
- Al hacer clic en el icono se abre la definición del elemento en XMLSpy (si está instalado).
- Al hacer clic en los hipervínculos del subapartado *Detalles* (p. ej. en `cvc-model-group`) se abre una descripción de la regla de validación correspondiente en el sitio web http://www.w3.org/.

Para validar los resultados de una asignación:

- Haga clic en el icono **Validar** para validar el documento con el esquema subyacente. Si los datos son válidos aparece un mensaje que lo confirma (*imagen siguiente*). De lo contrario, aparece un mensaje con información sobre los errores.



Nota: la casilla *Agregar referencia de esquema o DTD* del cuadro de diálogo "Configuración del componente" sirve para añadir la ruta de acceso del archivo de esquema XML referenciado al elemento raíz del archivo XML de salida. La ruta de acceso sirve para definir la ubicación del archivo de esquema al que hace referencia el archivo XML de instancia. Esto permite validar el archivo de instancia de salida cuando se ejecuta la asignación. En este campo puede introducir una dirección http:// o una ruta de acceso absoluta/relativa.

 © 2014 Altova GmbH

8.5 Compilar una asignación de MapForce

Objetivo: Compilar una asignación de datos en un archivo de ejecución de MapForce Server para poder ejecutarlo con MapForce Server.

Hay dos maneras de compilar una asignación de datos en un archivo de ejecución de MapForce Server:

Para compilar una asignación desde la línea de comandos de MapForce:
Ejecute MapForce y especifique el archivo de asignación y de datos y la opción `/COMPILE` de la línea de comandos. El archivo de ejecución de MapForce Server se crea en el directorio donde está el archivo de asignación de datos.

Para compilar una asignación desde la interfaz gráfica de MapForce:

1. Abra una asignación en MapForce (p. ej. `miAsignación.mfd`.)
2. Seleccione la opción de menú **Archivo | Compilar en archivo de ejecución de MapForce Server**.
3. Seleccione la carpeta en la que desea guardar el archivo `.mfx` y, si quiere, cambie el nombre de archivo.
4. Haga clic en **Guardar**.

El archivo de ejecución de MapForce Server `MiAsignación.mfx` se genera en la ubicación elegida.

Para ejecutar el archivo de ejecución de MapForce Server:
Consulte la documentación de MapForce Server sobre el uso del comando `run` de la línea de comandos.

8.6 Implementar una asignación de MapForce

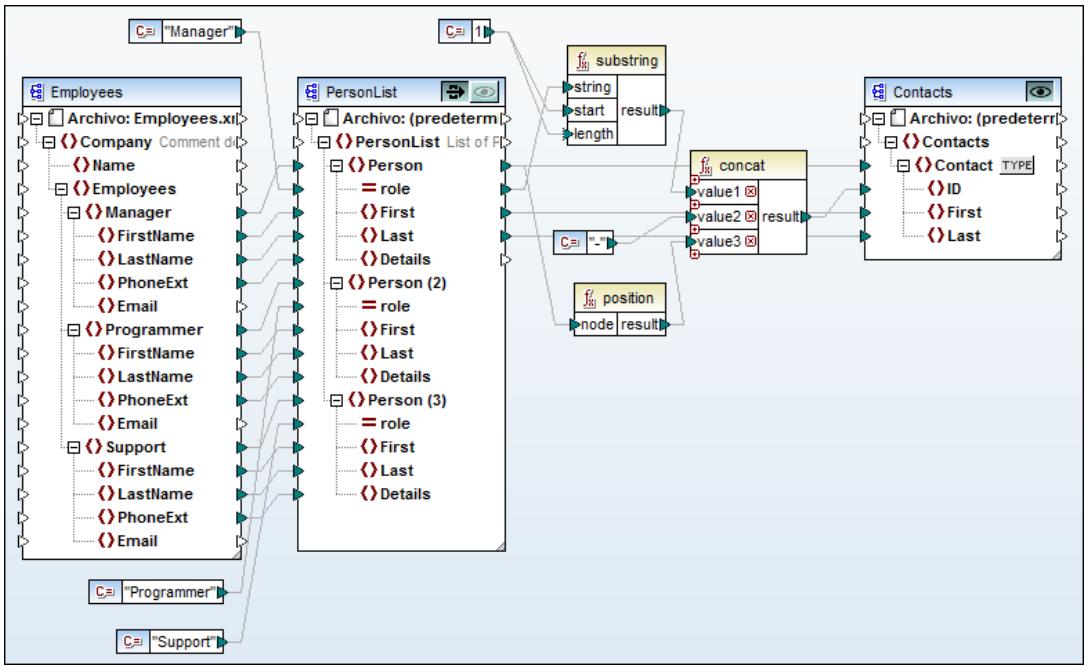
Objetivo: implementar una asignación de datos de MapForce en Altova FlowForce Server.

Implementar una asignación de datos significa que MapForce agrupa todos los recursos de la asignación (es decir, los recursos que usa la asignación) en un objeto y lo pasa al servidor o equipo donde se ejecuta FlowForce Server.

Nota: cuando implemente una asignación en FlowForce Server, compruebe que el motor seleccionado es el motor integrado, es decir, haga clic en el icono **BUILTIN**.

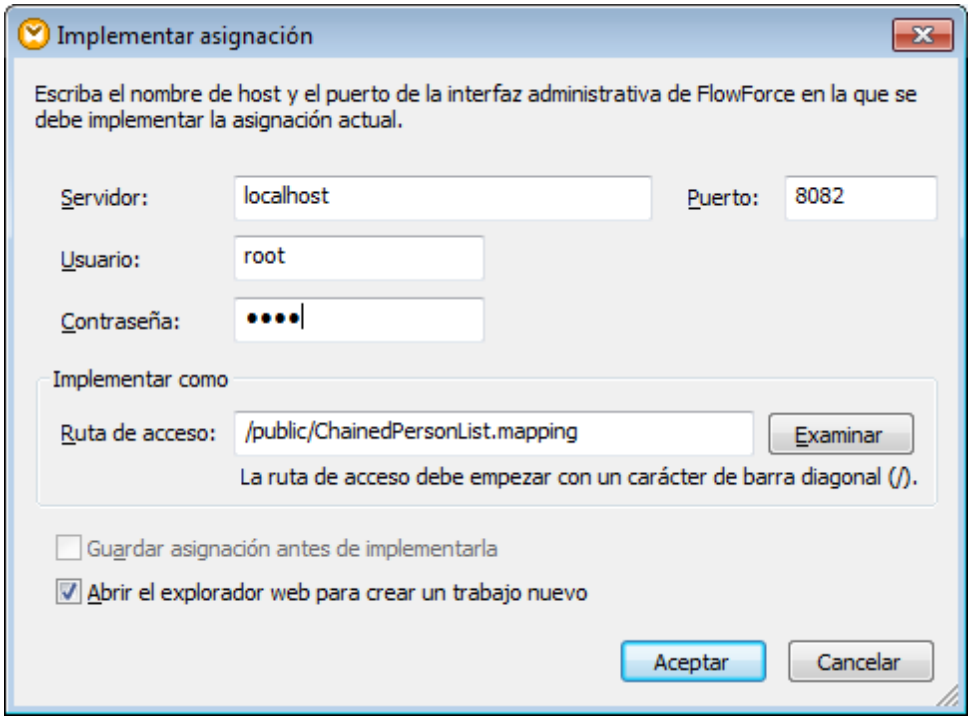
Para implementar una asignación en MapForce

1. Abra, por ejemplo, el archivo de asignación `ChainedPersonList.mfd` en MapForce.

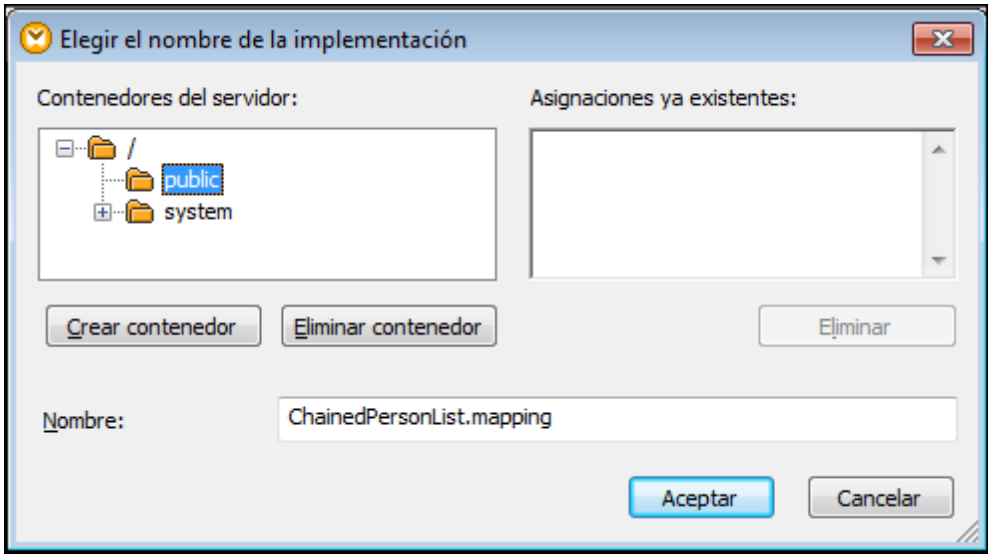


2. Seleccione la opción de menú **Archivo | Implementar en FlowForce Server**.
3. Escriba el nombre de servidor y el puerto de la interfaz de administración web en los campos correspondientes (p. ej. `localhost` y `8082` si FlowForce se ejecuta en el mismo equipo y si usa el puerto predeterminado).
4. Escriba el nombre de usuario y la contraseña necesaria para acceder al servidor (p. ej. `root` y `root`).

 © 2014 Altova GmbH



5. Opcionalmente, haga clic en el botón **Examinar** para definir donde colocar las asignaciones dentro del sistema de objetos del servidor de FlowForce (la ubicación predeterminada es `public`) y después haga clic en **Aceptar**.
   Asegúrese de que está activa la casilla *Abrir el explorador web para crear un trabajo nuevo*.



6. Haga clic en **Aceptar** para implementar la asignación.

   La ventana de mensajes notifica si la asignación se implementó correctamente. La interfaz de administración de FlowForce se abre automáticamente en el explorador web y aparece una página de creación de trabajos con varios campos de entrada ya rellenos.

7. El siguiente paso es definir el resto del trabajo (es decir, los desencadenadores, las credencias y los pasos de la ejecución). Para más información sobre esta parte del proceso consulte la documentación de FlowForce Server.

 © 2014 Altova GmbH

8.7 Parámetros de la línea de comandos

Sintaxis general de la línea de comandos:

MapForce.exe *nombrearchivo* [*/destino* [*dirsalida*] *opciones*

- Los corchetes **[...]** denotan parámetros opcionales.
- Las llaves **{...}** denotan un grupo de parámetros que contiene varias opciones.
- El símbolo de barra vertical **|** denota OR, p. ej /XSLT o /JAVA

MapForce.exe devuelve un código de salida 0 si la ejecución de la línea de comandos se completa correctamente. Cualquier otro valor indica un error. Puede usar el comando IF ERRORLEVEL en archivos por lotes para buscar el código de error.

nombrearchivo
El archivo MFD o MFP que se debe cargar. **Si la ruta de acceso o el nombre de archivo contiene un espacio, entonces ponga la ruta o el nombre entre comillas, p. ej. `"c:\Archivos de programa\...\NombreArchivo"`**

destino
El destino de la generación de código

/COMPILE: *opcionescompile*	compila una asignación en un archivo de ejecución de MapForce Server (`.mfx`) *opcionescompile* **JDBC**, **NOXMLSIGNATURES** **JDBC** transforma todas las conexiones de BD en JDBC usando el controlador JDBC y la URL de la BD definida en el cuadro de diálogo "Configuración del componente". **NOXMLSIGNATURES** suprime la generación de firmas digitales en el archivo de ejecución de MapForce Server (las firmas digitales no son compatibles con MapForce Server)
/GENERATE	usa la configuración de la carpeta actual para generar código de proyecto para todas las asignaciones del archivo de proyecto. Debe utilizar el archivo de proyecto MFP como *nombrearchivo*
/XSLT	genera código XSLT 1.0
/XSLT2	genera código XSLT 2.0
/XQuery	genera código XQuery 1.0
/JAVA	genera una aplicación Java
/CS	genera una aplicación C# usando las opciones de configuración de la asignación
/CS:*csopciones*	genera una aplicación C# usando estas opciones: *csopciones* { **VS2010** | **VS2008** | **VS2005** }

VS2010 genera una aplicación C# con archivo de solución para Visual Studio 2010

VS2008 genera una aplicación C# con archivo de solución para Visual Studio 2008

VS2005 genera una aplicación C# con archivo de solución para Visual Studio 2005

/CPP genera una aplicación C++ usando las opciones de configuración de la asignación

/CPP:*cpopciones* genera una aplicación C++ usando estas opciones:

cppopciones { **VC10** | **VC9** | **VC8** },{ **MSXML** | **XERCES** | **XERCES3** }, { **LIB** | **DLL** },{ **MFC** | **NoMFC** }

VC10 genera una aplicación C++ con archivo de solución para Visual Studio 2010

VC9 genera una aplicación C++ con archivo de solución para Visual Studio 2008

VC8 genera una aplicación C++ con archivo de solución para Visual Studio 2005

MSXML genera código C++ usando MSXML 6.0

XERCES genera código C++ usando Apache Xerces 2.x (2.6 y superior)

XERCES3 genera código C++ usando Apache Xerces 3.x

LIB genera código C++ para bibliotecas estáticas

DLL genera código C++ para bibliotecas de vínculos dinámicos

MFC genera código C++ compatible con MFC (Microsoft Foundation Classes)

NoMFC genera código C++ no compatible con MFC

dirsalida
Para la generación de código, el directorio donde se debe colocar el código de asignación generado. *dirsalida* es opcional. Si no se indica una ruta de acceso de salida, entonces se usa el directorio de trabajo/actual.

opciones
Especifica varias opciones:

/**GLOBALRESOURCEFILE** *nombrearchivorecglob* usa los recursos globales definidos en el archivo de recursos globales especificado

/**GLOBALRESOURCECONFIG** *configrecglob* usa la configuración de recursos globales especificada

/**LIBRARY** *nombrebibl* se utiliza con un lenguaje de destino de

 © 2014 Altova GmbH

	generación de código para especificar bibliotecas de funciones adicionales. Puede especificar esta opción más de una vez para cargar varias bibliotecas. Estas bibliotecas se añaden temporalmente (para esta sesión) a las bibliotecas desde Herramientas->Opciones->Bibliotecas.
/**LOG** *nombrearchivolog*	genera un archivo de registro. *nombrearchivolog* puede ser un nombre de ruta de acceso completo (es decir, el directorio y el nombre de archivo del registro) pero el directorio debe existir si se indica una ruta de acceso completa. De lo contrario no se puede generar el archivo de registro. Si solamente se indica el nombre de archivo, entonces el archivo se crea en el directorio *dirsalida*.

Notas:
En los parámetros de la línea de comandos las rutas de acceso relativas se interpretan como relativas al directorio de trabajo, es decir, al directorio actual de la aplicación que llama a MapForce. Esto afecta a la ruta de acceso de archivo MFD indicada en los parámetros *nombrearchivo*, *nombrearchivolog* y *nombrearchivorecglob*.

Si define una ruta de acceso absoluta, se ignora el directorio de trabajo. La ruta de acceso absoluta se utiliza tal y como se indicara.

Evite usar la barra diagonal inversa final y las comillas de cierre (\") en la línea de comandos p. ej. "C:\Mi Directorio\". El analizador de la línea de comandos interpreta estos dos caracteres como comillas dobles literales. Utilice dos barras diagonales inversas \\ si hay espacios en la ruta de acceso y necesita usar comillas (p. ej. "c:\Mi Directorio\\") o evite totalmente el uso de espacio y, por tanto, el uso de comillas (p. ej. c:\MiDirectorio).

Ejemplos:

MapForce.exe *nombrearchivo* inicia MapForce y abre el archivo definido por *nombrearchivo*.

I)
generar todos los archivos XSLT y un archivo de registro:
MapForce.exe *nombrearchivo.mfd* **/XSLT** *dirsalida* **/LOG** *nombrearchivolog*

II)
generar una aplicación Java y un archivo de registro:
MapForce.exe *nombrearchivo.mfd* **/JAVA** *dirsalida* **/LOG** *nombrearchivolog*

III)
generar una aplicación C# y un archivo de registro:
MapForce.exe *nombrearchivo.mfd* **/CS** *dirsalida* **/LOG** *nombrearchivolog*

IV)
generar una aplicación C++ usando la configuración de generación de código indicada en las opciones de la aplicación y genera un archivo de registro:
MapForce.exe *nombrearchivo.mfd* **/CPP** *dirsalida* **/LOG** *nombrearchivolog*

V)
generar una aplicación C++ usando el modificador /CPP, invalidando las opciones del compilador C++
MapForce.exe *nombrearchivo.mfd* **/CPP:(MSXML|XERCES),(LIB|DLL),(MFC|NoMFC)** *dirsalida* **[/LOG** *nombrearchivolog* **]**

MapForce.exe *nombrearchivo.mfd* **/CPP:MSXML,LIB,MFC**
genera la aplicación C++ usando la primera opción en cada caso:

- compilar para C++
- usar MSXML
- generar código para bibliotecas estáticas
- código generado compatible con MFC

MapForce.exe *nombrearchivo.mfd* **/CPP:XERCES,DLL,NoMFC** *dirsalida* **/LOG** *nombrearchivolog*
genera la aplicación C++ usando la segunda opción en cada caso:

- compilar para C++
- usar XERCES
- generar código para bibliotecas dinámicas
- código generado no compatible con MFC
- crear un archivo de registro en el directorio de salida *dirsalida* con el nombre de archivo *nombrearchivolog*

VI)
generar todos los archivos de resultados usando los recursos globales del archivo de recursos globales de la configuración especificada
Mapforce.exe *nombrearchivo.mfd dirsalida* **/GLOBALRESOURCEFILE** *nombrearchivorecursoglobal* **/GLOBALRESOURCECONFIG** *nombreconfiguración*

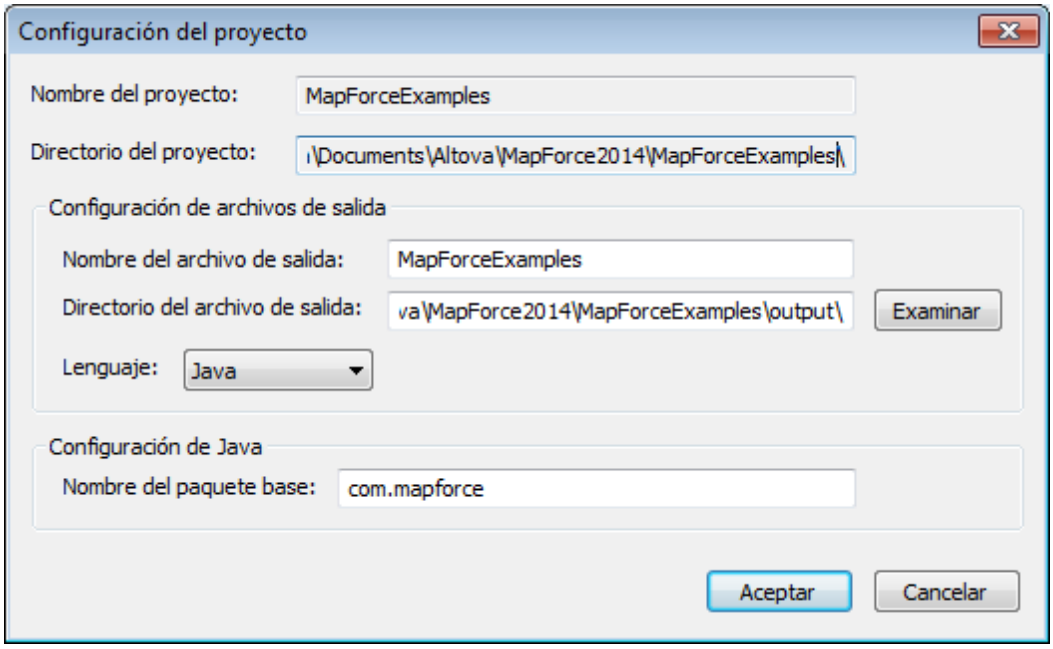
VII)
generar código de proyecto para todas las asignaciones del archivo de proyecto usando la configuración actual de la carpeta de proyecto
MapForce.exe *nombrearchivo.mfp* **/GENERATE /LOG** *nombrearchivolog*

Si se genera código para todo el proyecto, debe utilizar el nombre de archivo de proyecto (p. ej. `MapForceExamples.mfp`) como *nombrearchivo.*
El lenguaje de generación de código, así como la ruta de acceso de salida para las asignaciones de cada carpeta, vienen dados por el cuadro de diálogo Propiedades de cada carpeta cuando **no se especifica** *dirsalida*.

Si se usa el parámetro *dirsalida* en la generación de proyecto (con **/GENERATE**), entonces se sobrescribe la ruta de acceso del directorio *dirsalida* de la **carpeta raíz** del proyecto con la entrada suministrada en la línea de comandos. Este es el directorio que se utiliza si selecciona la

 © 2014 Altova GmbH

opción *Utilizar opciones de configuración predeterminadas del proyecto* del cuadro de diálogo "Propiedades" de una carpeta nueva (cuando se añade una carpeta nueva al proyecto).



VIII)
generar código de proyecto en Java para todas las asignaciones del archivo de proyecto
MapForce.exe *nombrearchivo.mfp* **/JAVA /LOG** *nombrearchivolog*

Se ignora el lenguaje de generación de código definido por las opciones de configuración de la carpeta y se usa Java para todas las asignaciones.

IX)
generar archivos de resultados de un proyecto de asignación (Java) compilado previamente, usando el archivo ejecutable JAR, y definir los archivos de entrada/salida en la línea de comandos.
java -jar *archivoasignación.jar* /**InputFileName** *nombrearchivoentrada* /**OutputFileName** *nombrearchivosalida*

Nota: los **parámetros** /InputFileName y /OutputFileName son el nombre de los componentes de entrada especiales de la asignación de MapForce que pueden utilizarse como parámetros en la ejecución de la línea de comandos.

Para aprender a definir los parámetros de entrada y salida (para suministrar los nombres de archivo) cuando se ejecuta la línea de comandos, consulte el apartado Línea de comandos: definir archivos de entrada.

X)
producir informes con formato de una asignación usando StyleVision y generar documentos HTML y PDF de la asignación.
MapForce.exe *nombrearchivo.mfd* **/BUILTIN** *dirsalida* **/STYLEVISION:***SVOpciones* **/LOG** *nombrearchivolog*

MapForce.exe *miAsignación*.mfd /**BUILTIN** /**STYLEVISION**:HTML,PDF /**LOG** miarchivolog.txt

XI)
usar un proyecto de asignación ya existente (FNAP.mfd) pero usar un archivo de instancia de entrada diferente y generar un archivo de salida diferente a los especificados en el proyecto de asignación.

Este ejemplo aprovecha el hecho de que los componentes tienen nombres, que se pueden usar para identificarlos unívocamente, y les asigna diferentes archivos de instancia/salida durante la ejecución de la línea de comandos.
Mapforce.exe *nombrearchivo.mfd* **/BUILTIN /LOG** *nombrearchivolog* ***/-*** *parámetrostiempoejecución* **/Parámetro-nombre1 parámetro-valor1 Parámetro-nombre2 parámetro-valor2**.

Mapforce.exe FNAP.mfd /**BUILTIN** /**LOG** *log.txt* ***/-*** */Altova_Hierarchical Altova_hierarchical2.xml /Altova_Hierarchical_targ Altova_out.xml*

Nota: Altova_hierarchical2.xml como instancia de entrada nueva del componente de origen ***Altova_Hierarchical*** y
Altova_out.xml es el nuevo nombre de archivo de destino/salida del componente de destino ***Altova_Hierarchical_targ***

Para más información, consulte el apartado Línea de comandos: nombres de los componentes.

 © 2014 Altova GmbH

8.8 Archivos de catálogo en MapForce

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\CommonMapForce
`%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 actual.

 © 2014 Altova GmbH

`%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="IDPúblico del Recurso" uri="URL del archivo local"/>`
- `<system systemId="IDdeSistema del Recurso" uri="URL del archivo local"/>`
- `<uri name="nombreDeArchivo" uri="URL del archivo identificado con el nombre de archivo"/>`
- `<rewriteURI uriStartString="StartString del URI que se debe volver a escribir" rewritePrefix="Cadena que debe sustituir a StartString"/>`
- `<rewriteSystem systemIdStartString="StartString del IDdeSistema" rewritePrefix="Cadena de sustitución para localizar el recurso localmente"/>`

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.

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

 © 2014 Altova GmbH

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.

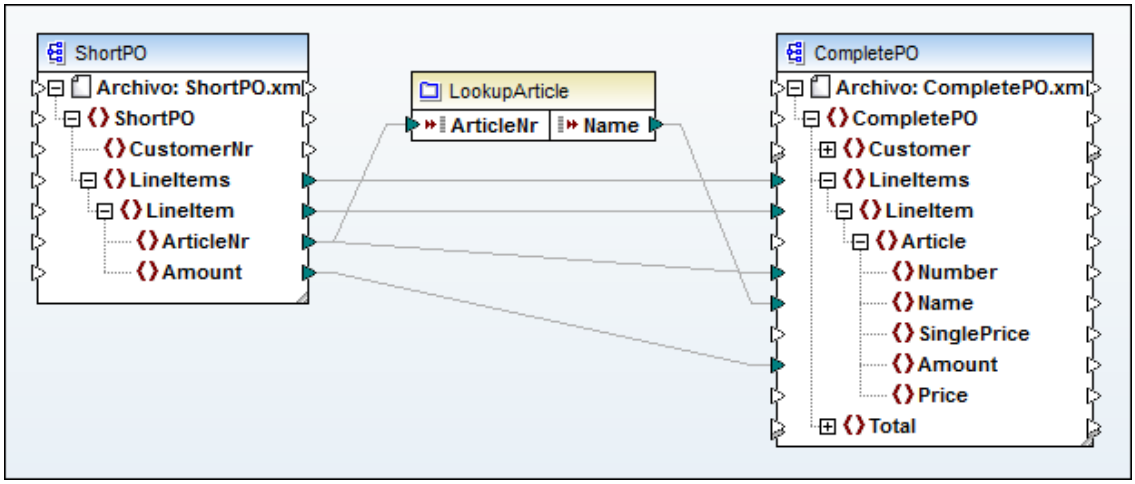
8.9 Documentar proyectos de asignación

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. Para más información consulte el siguiente apartado, Hojas de estilos predefinidas.

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.

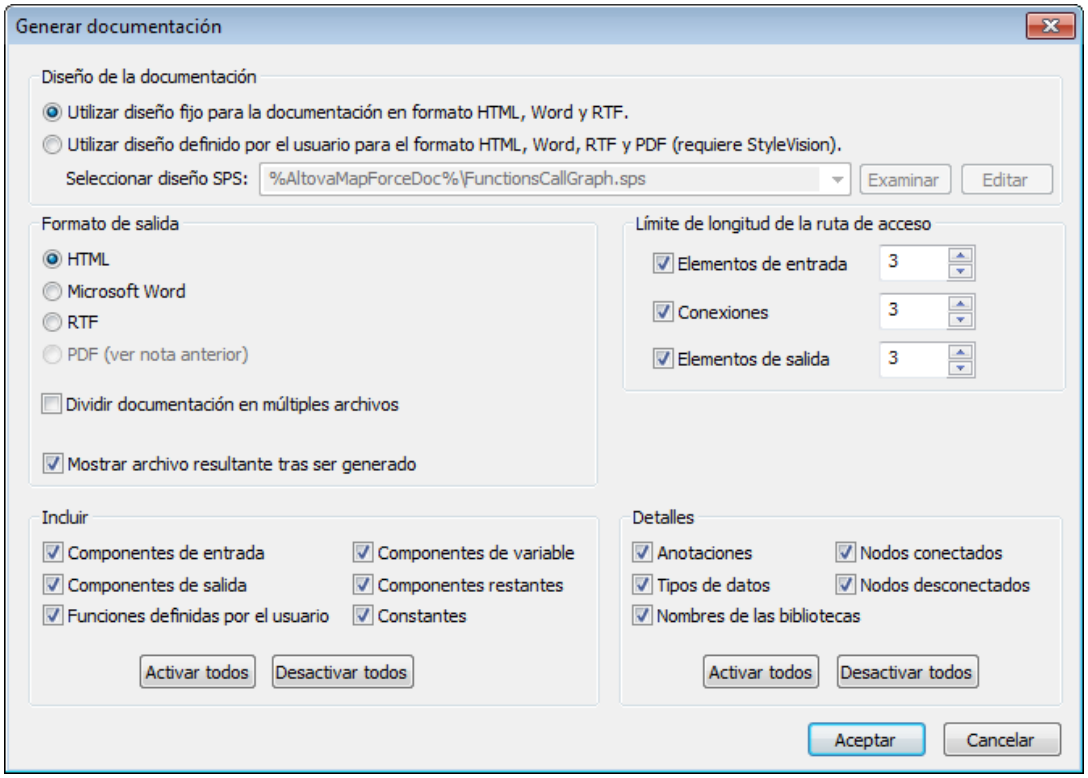
La imagen que aparee a continuación muestra un fragmento de la asignación `Lookup-standard.mfd`, disponible en la carpeta `...\MapForceExamples`.



Para generar la documentación de esta asignación:

- Seleccione la opción de menú **Archivo | Generar documentación**.
  Aparece este cuadro de diálogo:

 © 2014 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 `...\Mis Documentos\Altova\MapForce2014\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 `...MapForce2014\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.

 © 2014 Altova GmbH

lookup-standard.mfd (C:\Altova\MapForce2013\MapForceExamples\lookup-standard.mfd)

Entrada **ShortPO** (ShortPO.xsd)

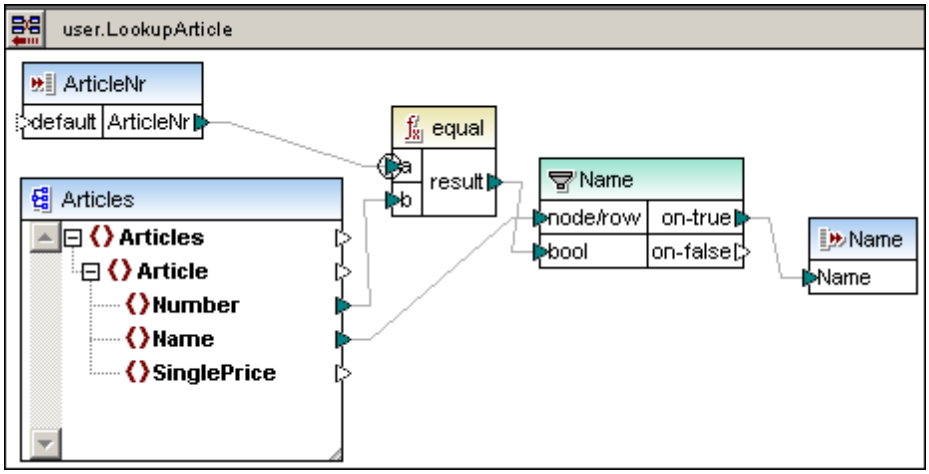
Nodos	Conexiones	
Archivo: ShortPO.xml Tipo: string		
ShortPO Tipo: restriction of xs:anyType [0..1]		
ShortPO/CustomerNr Tipo: xs:integer		
ShortPO/LineItems Tipo: restriction of xs:anyType	*directa*	CompletePO/LineItems Tipo: restriction of xs:anyType
ShortPO/LineItems/LineItem Tipo: restriction of 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 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 =>	

8.9.1 Hojas de estilos predefinidas

Gráfico de llamadas
Si usamos el archivo SPS **`FunctionsCallGraph.sps`**. para documentar la asignación, la documentación resultante muestra los gráficos de llamadas de la asignación principal y de todas las funciones definidas por el usuario.

A continuación puede ver la documentación de gráfico de llamadas de la asignación **`PersonListByBranchOffice.mfd`**.

```
Expand 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
```

Funciones utilizadas por...
Si usamos el archivo SPS **`FunctionsUsedBy.sps`**. para documentar la asignación, la documentación resultante muestra todas las funciones presentes en la asignación y su uso directo o indirecto en otras funciones. Este tipo de documentación puede ser muy útil a la hora de planificar cambios en funciones definidas por el usuario y determinar qué funciones pueden verse afectadas por dichos cambios.

 © 2014 Altova GmbH

A continuación puede ver la documentación de funciones utilizadas por la asignación **`PersonListByBranchOffice.mfd`**.

Library **core**

Function	Directly used by	Indirectly used by
core.concat	user.Person2Details	Main mapping user.LookupPerson
core.equal	Main mapping user.EqualAnd	user.LookupPerson
core.filter	Main mapping user.LookupPerson	
core.logical-and	user.EqualAnd	Main mapping user.LookupPerson

Library **user**

Function	Directly used by	Indirectly used by
user.EqualAnd	user.LookupPerson	Main mapping
user.LookupPerson	Main mapping	
user.Person2Details	user.LookupPerson	Main mapping

Análisis de impacto

Si usamos el archivo SPS **`ImpactAnalysis.sps`** para documentar la asignación, la documentación resultante enumera cada conexión entre nodos de entrada y salida una por una y puede ser práctica en el proceso de análisis de impacto con herramientas de modelado.

A continuación puede ver la documentación de análisis de impacto para la asignación **`PersonListByBranchOffice.mfd`**.

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

Si usamos el archivo SPS **`OverallDocumentation.sps`** para documentar la asignación, obtenemos la documentación más detallada posible.

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)

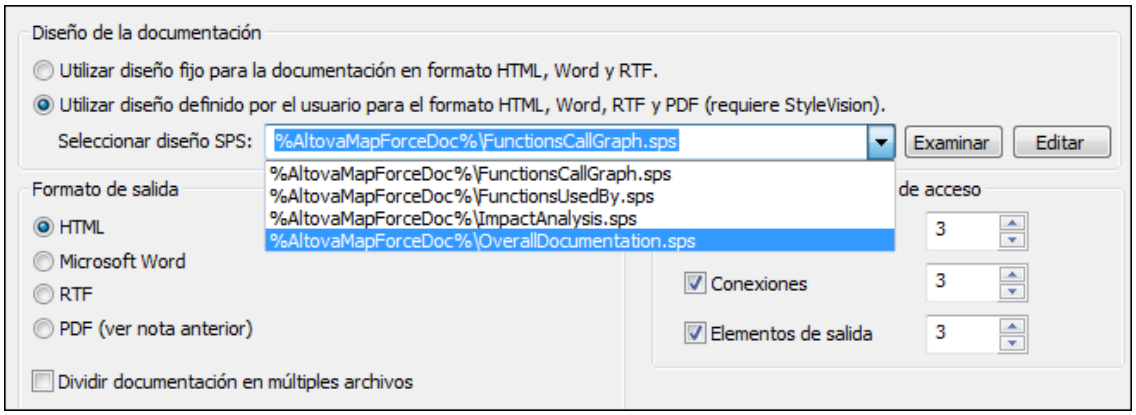
Nodes	Connections	
File: BranchOffices.xml Type: string		
BranchOffices Type: restriction of xs:anyType [0..1]		
BranchOffices/Name Type: restriction of xs:string		
BranchOffices/Office Type: restriction of xs:anyType [0..∞]	**core.filter => node/row | on-true =>**	PersonList Type: restriction of xs:anyType [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

© 2014 Altova GmbH

8.9.2 Diseño de documentación definido por el usuario

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`.



Estos archivos SPS predefinidos están en la carpeta `...MapForce2014\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 `...\Mis Documentos\Altova\MapForce2014\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.

8.10 Archivos SPS en la vista previa de MapForce

MapForce puede ofrecer una vista previa / representar componentes de esquema XML y XBRL usando archivos SPS (StyleVision Power Stylesheet). Esto significa que MapForce genera una vista previa de los datos del componente de destino en documentos HTML, RTF, PDF o Word 2007+. Si usa la edición Enterprise Edition de StyleVision, MapForce también puede representar datos de gráficos en estas vistas previas.

Para generar una vista previa de este tipo, es necesario tener instalado Altova StyleVision en el equipo.

Los archivos SPS (StyleVision Power Stylesheets) se crean en StyleVision y se asignan a un componente de MapForce. Estas hojas de estilos no se pueden editar ni modificar en MapForce, pero se pueden abrir en StyleVision desde MapForce. En la edición de 64 bits de MapForce, los documentos Word 2007+ y RTF de vista previa se abren en una aplicación no incrustada.

Por tanto, si StyleVision está instalado en el equipo y el componente de destino tiene asociado un archivo SPS, en MapForce hay varios paneles adicionales:

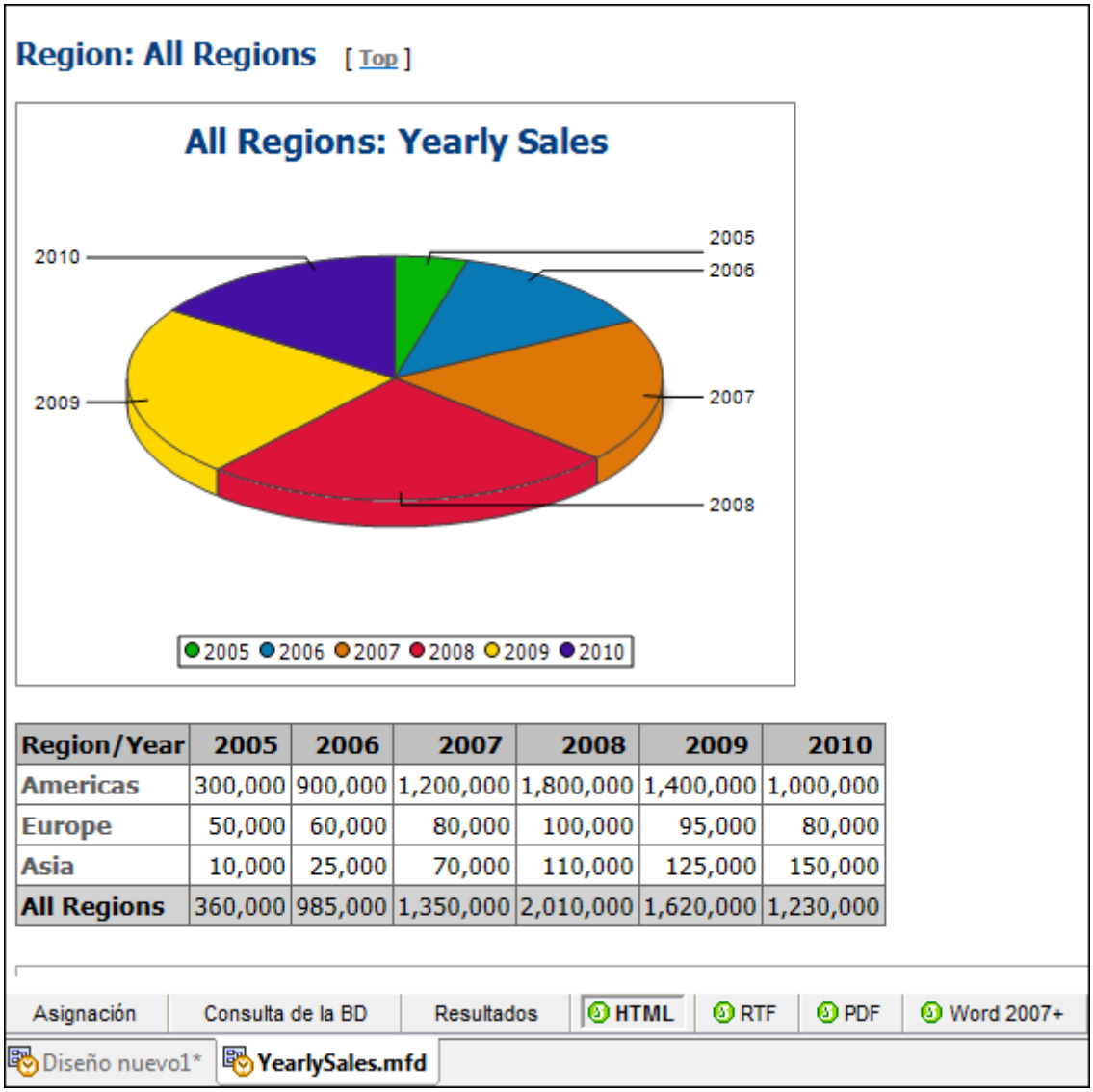
- el panel **HTML**
- el panel **RTF** (vista previa en RTF)
- el panel **PDF** (requiere FOP versión 0.93 / 1.0)
- el panel **Word 2007+** (vista previa en MS Word 2007 o superior)

Nota: dependiendo de qué edición de StyleVision tenga instalada, MapForce ofrece unos paneles u otros. Si se trata de StyleVision Enterprise Edition, MapForce incluye todos los paneles mencionados. Cuando vea estos paneles adicionales, significa que el componente de destino tiene asociado un archivo SPS. En la carpeta `...\MapForceExamples` encontrará varios ejemplos.

Si se trata de StyleVision Professional Edition, MapForce incluye los paneles **HTML** y **RTF** solamente.

Nota: si la asignación contiene varios componentes enlazados (es decir, si se trata de una asignación en cadena) la vista previa HTML, RTF, etc. solamente está disponible para el componente que tenga asociado un archivo SPS y que tenga activado el botón de vista previa .

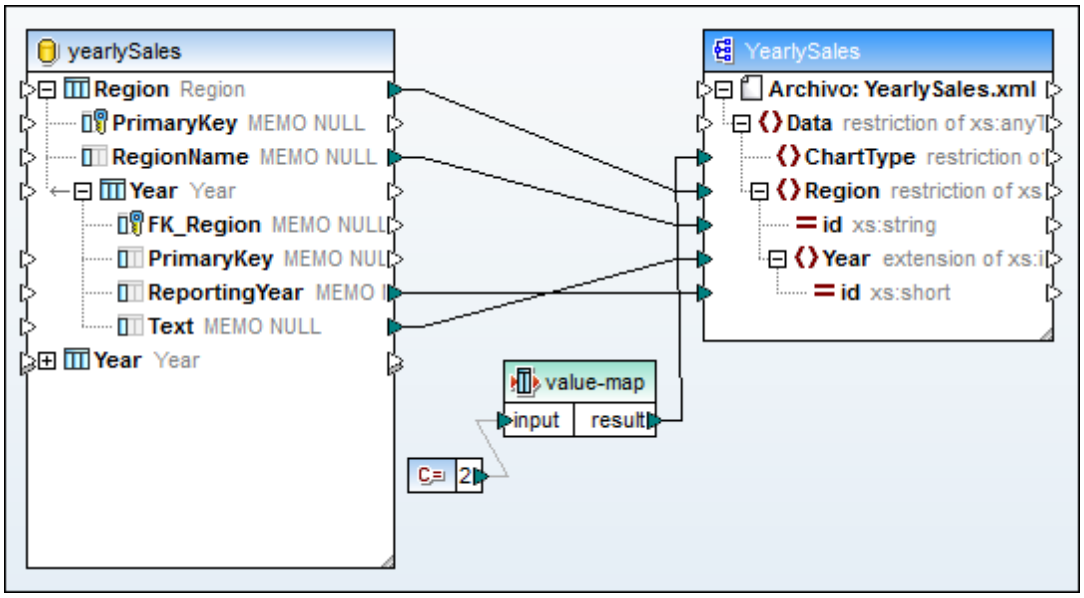
 © 2014 Altova GmbH



Region/Year	2005	2006	2007	2008	2009	2010
Americas	300,000	900,000	1,200,000	1,800,000	1,400,000	1,000,000
Europe	50,000	60,000	80,000	100,000	95,000	80,000
Asia	10,000	25,000	70,000	110,000	125,000	150,000
All Regions	360,000	985,000	1,350,000	2,010,000	1,620,000	1,230,000

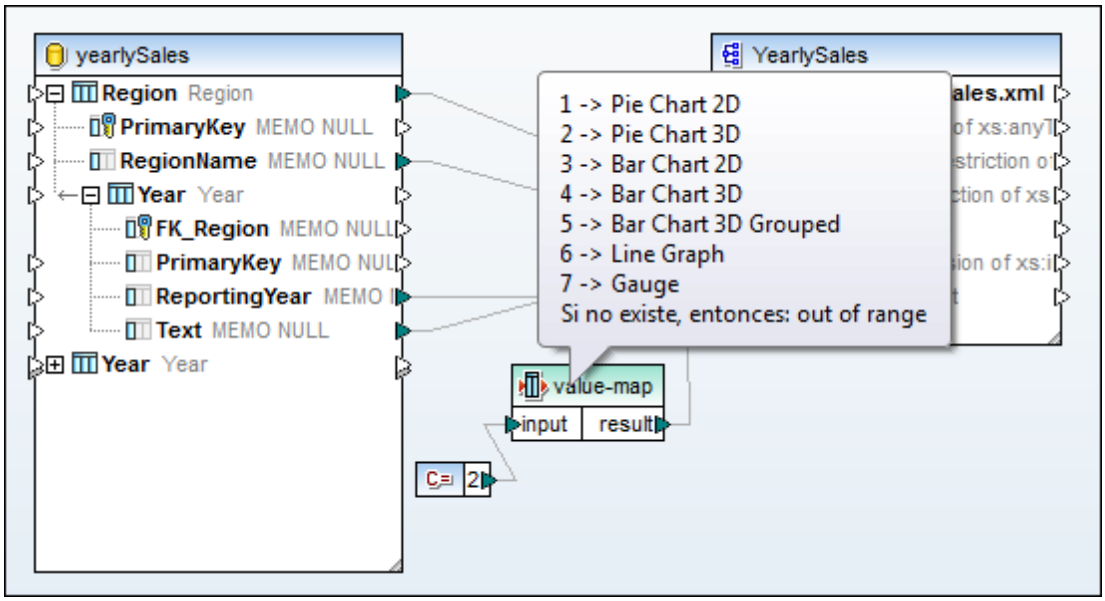
8.10.1 Asignar un archivo SPS a un componente

En este apartado usamos como ejemplo el archivo `YearlySales.mfd`, disponible en la carpeta `...\MapForceExamples\Tutorial`.



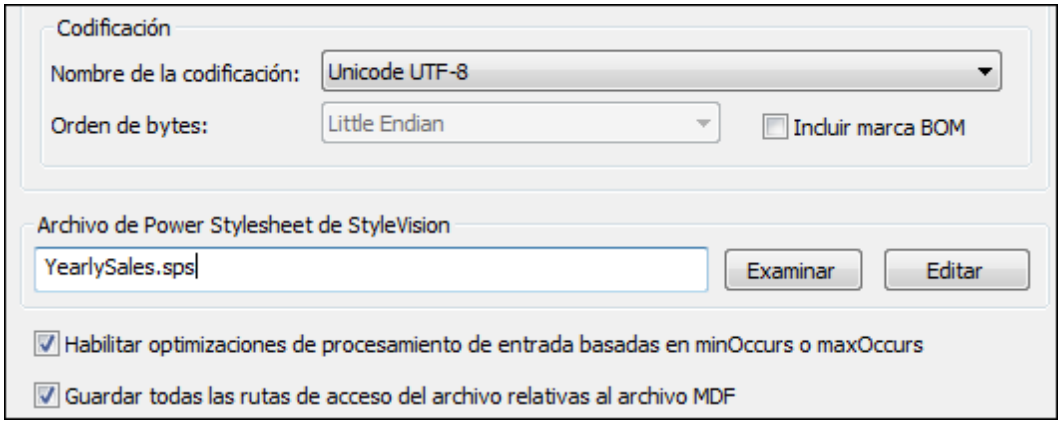
El número de la constante define el tipo de gráfico que se usa para representar los datos en el panel elegido de StyleVision.

Al pasar el puntero del mouse sobre la asignación de valores **value-map**, la información emergente enumera todos los gráficos posibles (en este caso, se trata de un gráfico circular 3D).

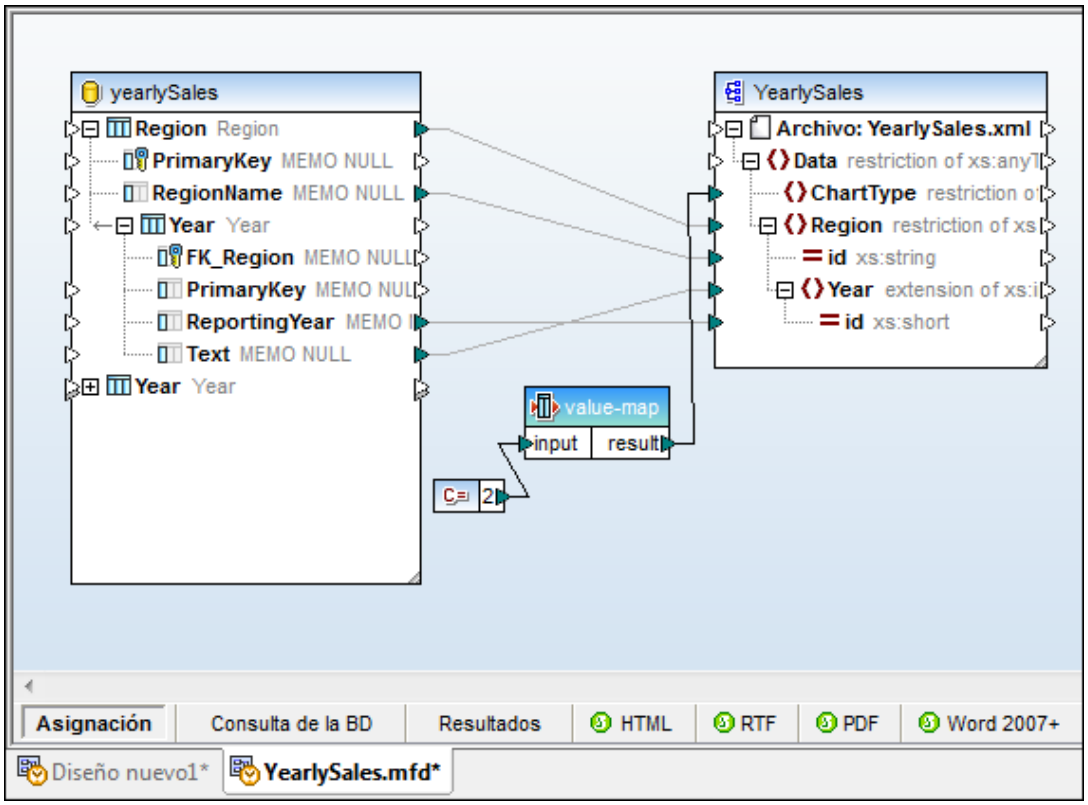


Para asignar una plantilla SPS a un componente de MapForce:
Una vez recibida (o diseñada) la plantilla en StyleVision, podemos usarla en MapForce:

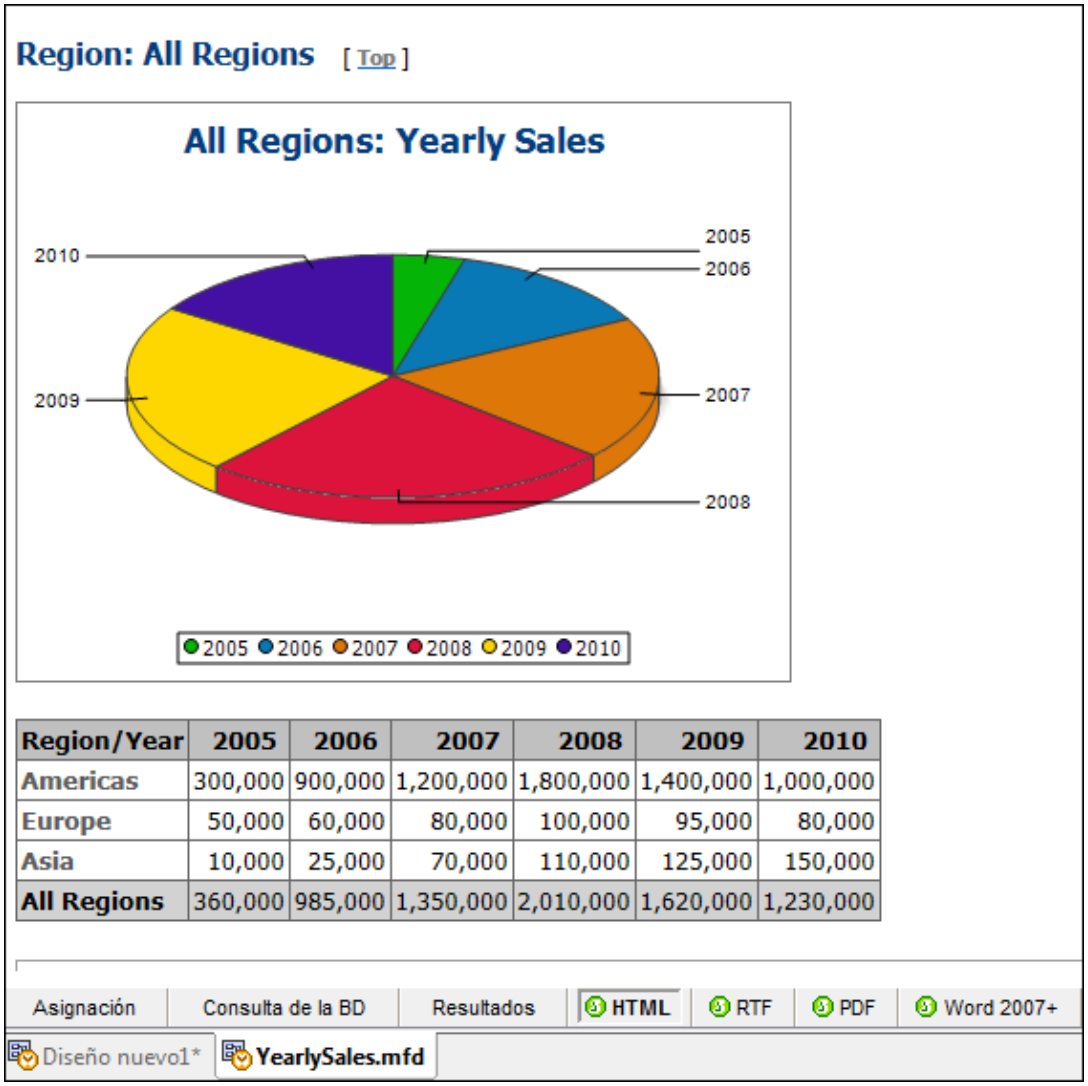
1. Haga doble clic en la barra de título del componente al que quiere asignar el archivo SPS (p. ej. **YearlySales**). Aparece el cuadro de diálogo "Configuración del componente".



2. En el grupo *Archivo de Power Stylesheet de StyleVision* haga clic en **Examinar**, seleccione el archivo SPS que desea usar y haga clic en **Aceptar**.



Como resultado aparecen las pestañas de los paneles **HTML**, **RTF**, **PDF** y **Word 2007+** a la derecha de la pestaña **Resultados**. Al hacer clic en una de estas pestañas, los datos del componente se representan en el formato correspondiente. En la imagen siguiente, por ejemplo, los datos se presentan en formato HTML.



Region/Year	2005	2006	2007	2008	2009	2010
Americas	300,000	900,000	1,200,000	1,800,000	1,400,000	1,000,000
Europe	50,000	60,000	80,000	100,000	95,000	80,000
Asia	10,000	25,000	70,000	110,000	125,000	150,000
All Regions	360,000	985,000	1,350,000	2,010,000	1,620,000	1,230,000

Nota: si en el cuadro de diálogo "Configuración del componente" hace clic en el botón **Crear...**, MapForce solicita un nombre para el archivo SPS y abre StyleVision, donde puede crear el archivo SPS. Si el componente ya tiene asignado un archivo SPS, el cuadro de diálogo "Configuración del componente" ofrece el botón **Editar**, en lugar de **Crear...**.

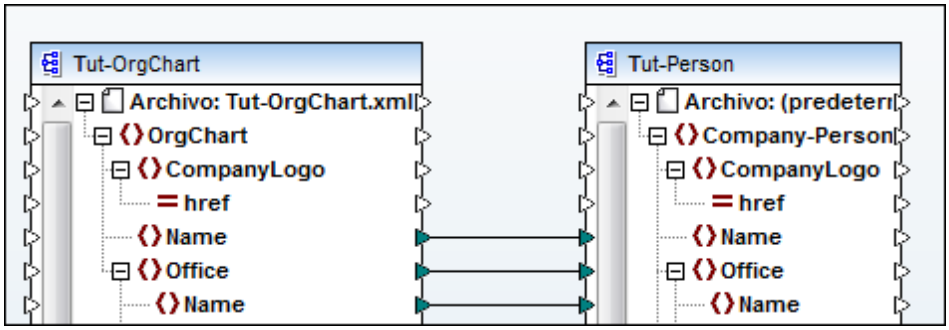
 © 2014 Altova GmbH

Altova MapForce 2014

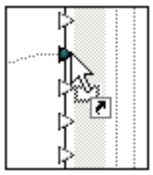
Asignaciones entre componentes

9 Asignaciones entre componentes

Un **conector** traza la **asignación** entre dos conjuntos de datos y permite que los datos (valores) de **origen** aparezcan (o se transformen) en el componente de destino (p. ej. en el documento/ esquema o base de datos de destino).



Los **componentes** y las **funciones** tienen pequeños triángulos de conexión llamados **iconos de entrada/salida**. Estos iconos están situados a la izquierda y a la derecha de todos los **elementos** "asignables", es decir, los que se pueden asignar a otros elementos. Cuando se hace clic en un icono y se arrastra el mouse, se crea un **conector de asignación**. Este conector se puede soltar en otro icono o nombre de elemento. Al pasar el mouse por un icono donde se puede soltar el conector, el icono del mouse cambia de forma.



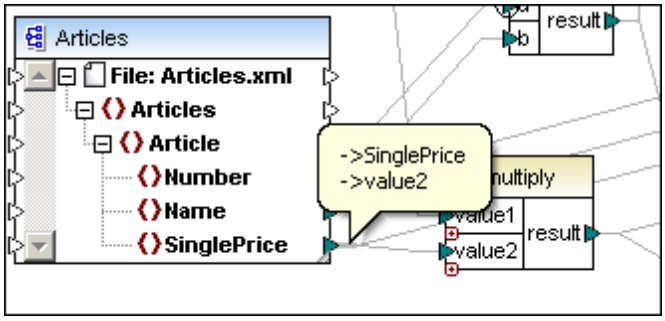
Para seleccionar el icono del elemento/atributo también puede hacer clic en su **nombre**. Igualmente, si suelta el conector en el nombre del elemento/atributo de destino, se selecciona automáticamente su icono.

Los **iconos de entrada** solamente pueden tener un conector. Si intenta conectar otro conector, aparece un mensaje preguntando si desea reemplazar el conector existente con el nuevo conector o si desea **duplicar** el icono de entrada.

Los **iconos de salida** pueden tener varios conectores, cada uno conectado a un icono de entrada diferente.

Si pasa el puntero del mouse sobre la línea de un conector, cerca del icono de entrada/salida, el conector se resalta y aparece un mensaje emergente. El mensaje incluye el nombre del elemento de destino. Si definió varios conectores para el mismo icono de salida, el mensaje emergente puede incluir un máximo de 10 nombres. La imagen siguiente, por ejemplo, muestra que los elementos de destino del conector son el elemento **`SinglePrice`** y el parámetro de entrada **`value2`** de la función **`multiply`**.

 © 2014 Altova GmbH

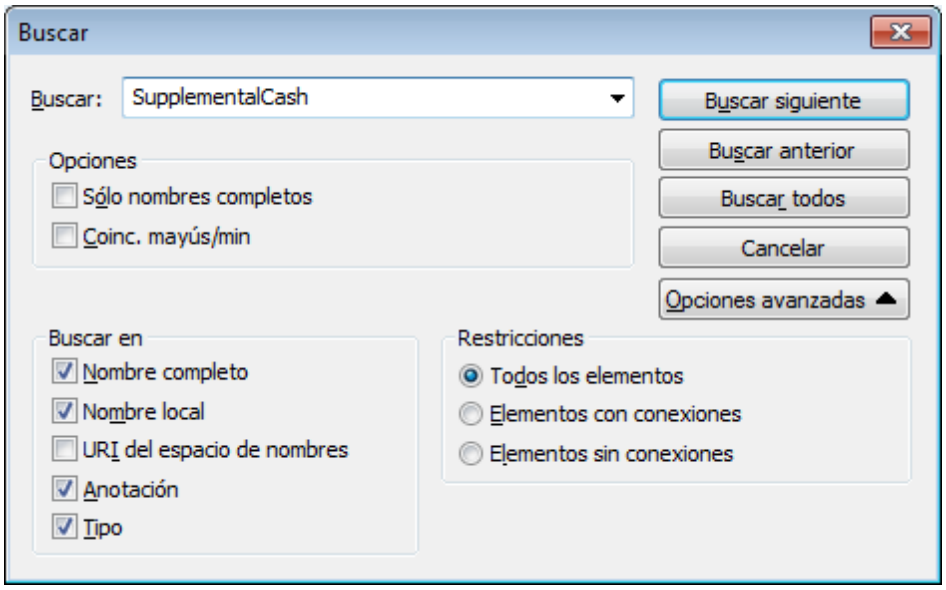


Para **mover** un conector de un elemento a otro elemento, haga clic el icono de salida del conector y arrástrelo hasta su nueva posición.

Para crear un **duplicado del conector** desde el mismo elemento de origen hasta otro elemento de destino diferente, haga clic en el icono de entrada del elemento de destino inicial y arrástrelo hasta el otro elemento de destino mientras pulsa la tecla **Ctrl**.

Para buscar un elemento/nodo concreto en un componente:

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



Las opciones avanzadas de este cuadro de diálogo sirven para definir en qué elementos/nodos se realiza la búsqueda y a limitar las opciones de búsqueda a cierto tipo de conexiones.

Conexión automática de elementos

Si activa el icono **Conectar automáticamente los secundarios equivalentes** y después crea un conector entre dos elementos que tienen secundarios, todos los secundarios que tengan el mismo nombre se conectan automáticamente.

Número de conectores

La mayoría de los componentes tienen iconos de entrada/salida:

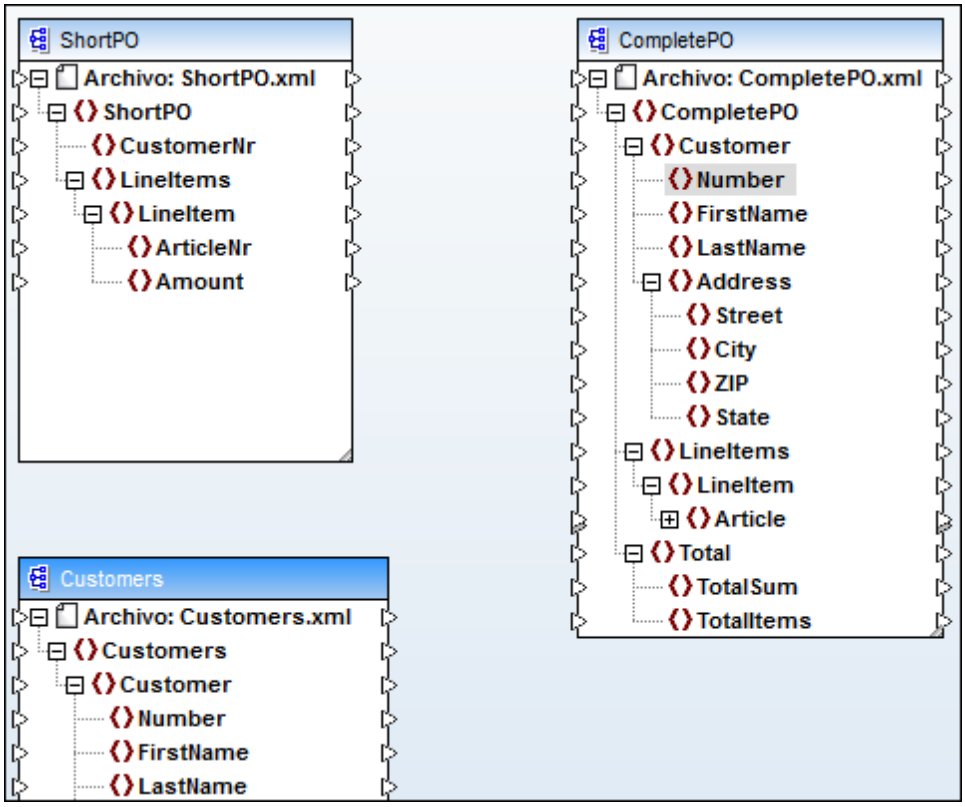
- Los elementos de **esquema** (atributos/elementos) tienen un icono de entrada y otro de salida.
- Los elementos de **base de datos** tienen un icono de entrada y otro de salida.
- Los componentes de esquema, de base de datos y de otro tipo presentes dentro de funciones definidas por el usuario solamente tienen iconos de salida.
- Los elementos duplicados solamente tienen iconos de entrada. Así puede asignarles varios componentes de origen. Para más información consulte el apartado Duplicar elementos de entrada.
- En las **funciones**, cada **parámetro** tiene **un solo icono**, ya sea de entrada o salida. Por ejemplo, la función **add** tiene dos (o más) iconos de entrada y sólo uno de salida.
- Algunos componentes especiales pueden tener un solo icono o varios. Por ejemplo, las **constantes** solamente tienen un icono de salida.

Elementos/nodos/iconos obligatorios (en componentes de destino)
A fin de ayudarle a la hora de crear asignaciones de datos, MapForce resalta los nodos/elementos obligatorios en los componentes de destino en color naranja. Entre los componentes obligatorios se encuentran:

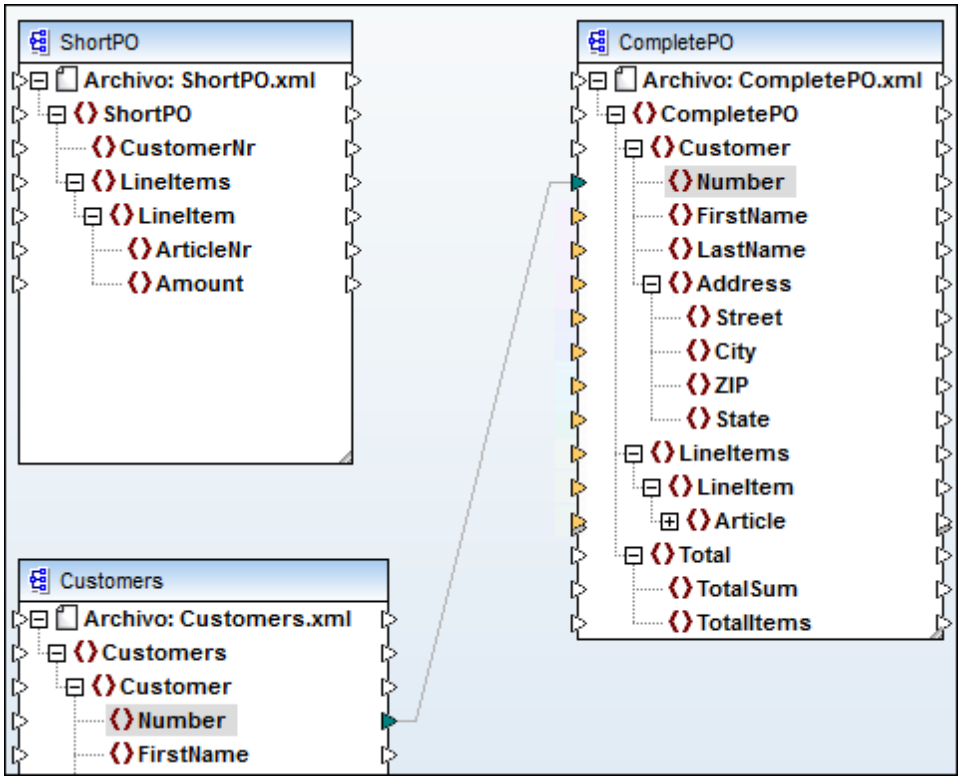
- Los elementos cuyo parámetro `minOccurs` es igual o mayor que `1` (en los componentes EDI y XML).
- Los campos que no admiten valores `null` (en los componentes de BD).
- Todos los nodos de las llamadas y respuestas WSDL.
- Los nodos XBRL definidos como obligatorios.
- Algunos parámetros concretos de las funciones. Por ejemplo, si un parámetro de entrada de la función está conectado (de un filtro, por ejemplo), el otro parámetro también debe estar conectado.
- El nombre de las hojas de cálculo en los documentos Excel.

Ejemplo:
Para crear la asignación de datos `CompletePO.mfd` (disponible en la carpeta `...\MapForce2014\MapForceExamples`) hace falta insertar estos componentes de esquema XML:

 © 2014 Altova GmbH



Ahora imagine que conectamos el nodo **`Number`** del componente **Customers** con el nodo **`Number`** del componente **CompletePO**. Nada más crear la conexión, MapForce resalta automáticamente los nodos obligatorios del componente de destino **CompletePO**. Observe que el nodo contraído **`Article`** también se resalta.



Mover y restablecer conectores

Consulte el apartado Mover y restablecer conectores para aprender a mover grupos de conectores y para saber qué ocurre cuando se eliminan componentes (p. ej. filtros) que tienen muchos conectores secundarios.

 © 2014 Altova GmbH

9.1 Métodos de asignación de datos

En MapForce puede usar varios métodos para asignar datos:

- Asignaciones basadas en el destino (estándar),
- Asignaciones basadas en el origen (asignación de contenido mixto) y
- Conexiones de copia total (copia de elementos secundarios).

Conectores y sus propiedades

- Cuando haga clic en un conector, este se resalta en color rojo.
- Cuando el conector está resaltado en rojo y pulse la tecla **Supr**, el conector se elimina.
- Cuando haga clic con el botón secundario en un conector, aparece un menú contextual.
- Cuando haga doble clic en un conector, se abre el cuadro de diálogo Configuración de la conexión.

Visualización de los conectores

En MapForce puede elegir qué conectores se muestran en la ventana de asignación.

Pulsando el icono **Mostrar los conectores del componente seleccionado** puede:

- ver resaltados en negro todos los conectores de la asignación o
- ver resaltados en negro los conectores relacionados con el componente seleccionado. Los demás conectores aparecen atenuados.

Pulsando el icono **Mostrar los conectores desde su origen a su destino** puede:

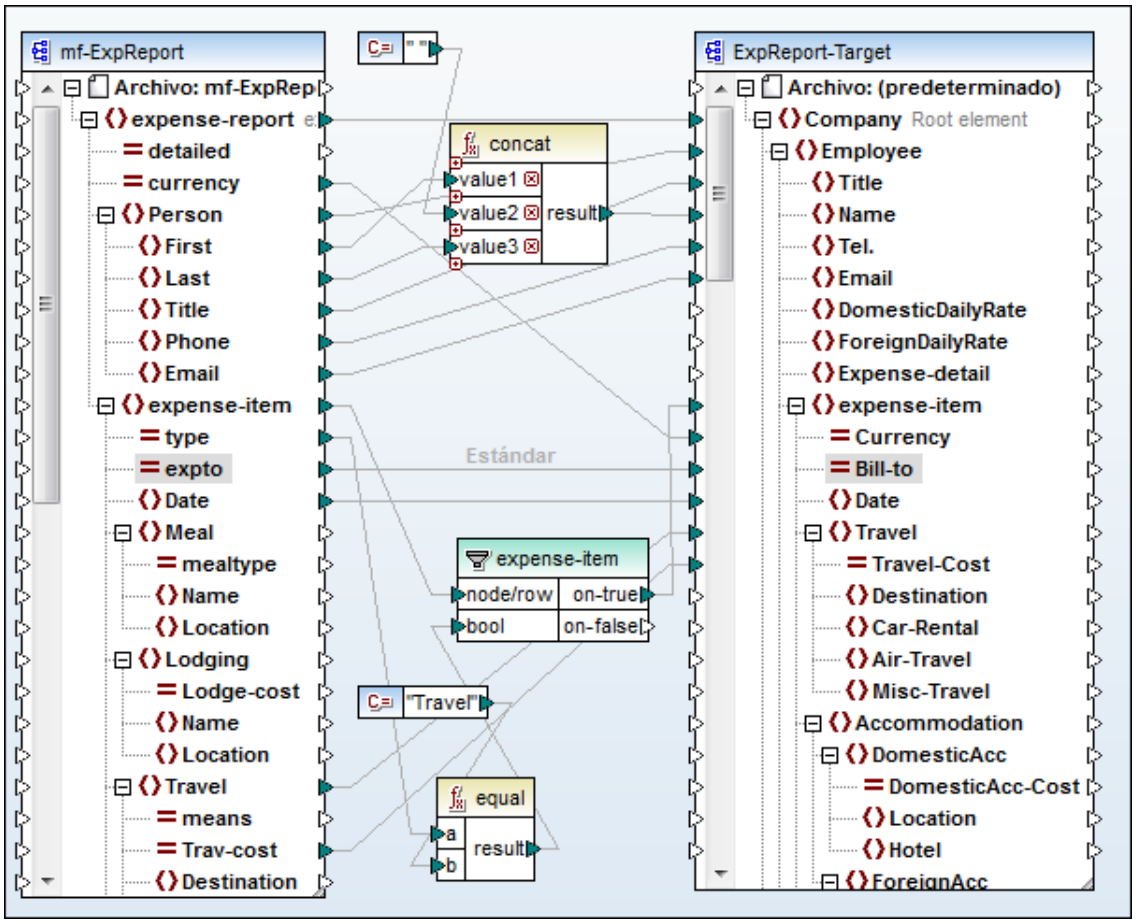
- ver resaltados en negro los conectores que están conectados **directamente** con el componente seleccionado o
- ver resaltados en negro los conectores relacionados con el componente seleccionado, desde el componente de origen hasta los de destino.

9.1.1 Basada en el destino (asignación estándar)

Las asignaciones basadas en el destino son el método de asignación estándar de MapForce. En este tipo de asignaciones los resultados siguen el orden de los nodos de destino.

- No se admite/no se puede asignar el contenido de nodos de texto de contenido mixto.
- El orden de los nodos secundarios depende del archivo de esquema de destino.

Las asignaciones estándar se dibujan con una línea sólida.

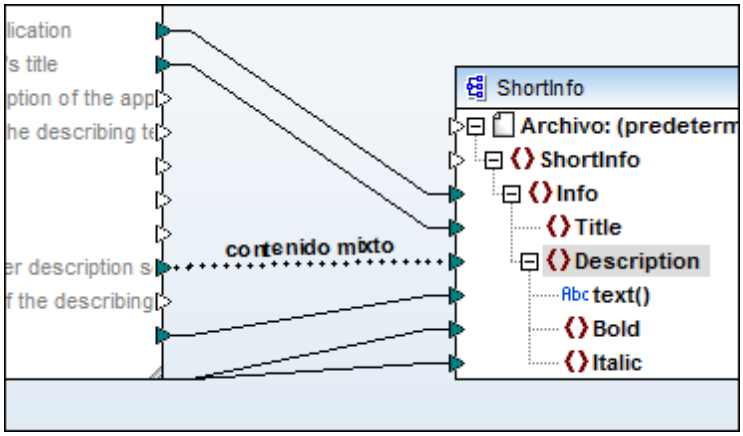


 © 2014 Altova GmbH

9.1.2 Basada en el origen (asignación de contenido mixto)

Las asignaciones basadas en el origen (asignaciones de contenido mixto) permiten asignar los nodos de texto y secundarios en el mismo orden que tenían en el archivo XML de **origen** automáticamente.

- Se admite/se puede asignar el contenido de nodos de texto de contenido mixto
- El orden de los nodos secundarios depende del archivo XML de origen.



Las asignaciones de contenido mixto se dibujan con una línea discontinua.

Las asignaciones basadas en el origen (de contenido mixto) también se pueden aplicar a los elementos **complexType** de XML Schema. Los nodos secundarios se asignarán en el orden que tenían en el archivo XML de origen.

En las asignaciones basadas en el origen (de contenido mixto) puede usar:

- como componentes de **origen**:
  - elementos **complexType** de XML Schema (incluidos elementos de contenido mixto, es decir `mixed=true`)
  - elementos **complexType** de XML Schema (incluidos elementos de contenido mixto) en esquemas incrustados de un campo de BD
- como componentes de destino:
  - elementos **complexType** de XML Schema (incluidos elementos de contenido mixto)
  - elementos **complexType** de XML Schema (incluidos elementos de contenido mixto) en esquemas incrustados de un campo de BD

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

Asignación de contenido mixto

Para este ejemplo utilizamos el archivo `Tut-OrgChart.mfd`, que está en la carpeta `...\MapForceExamples\Tutorial`.

Archivo XML de origen

La imagen que aparece a continuación muestra un fragmento del archivo `Tut-OrgChart.xml`. Para este ejemplo nos vamos a centrar en el elemento de contenido mixto **`para`**, así como en sus nodos secundarios **`bold`** e **`italic`**.

Observe que el elemento **`para`** también contiene una instrucción de procesamiento (`sort alpha-`

`ascending`) y un comentario (`Company details...`), que también se pueden asignar.

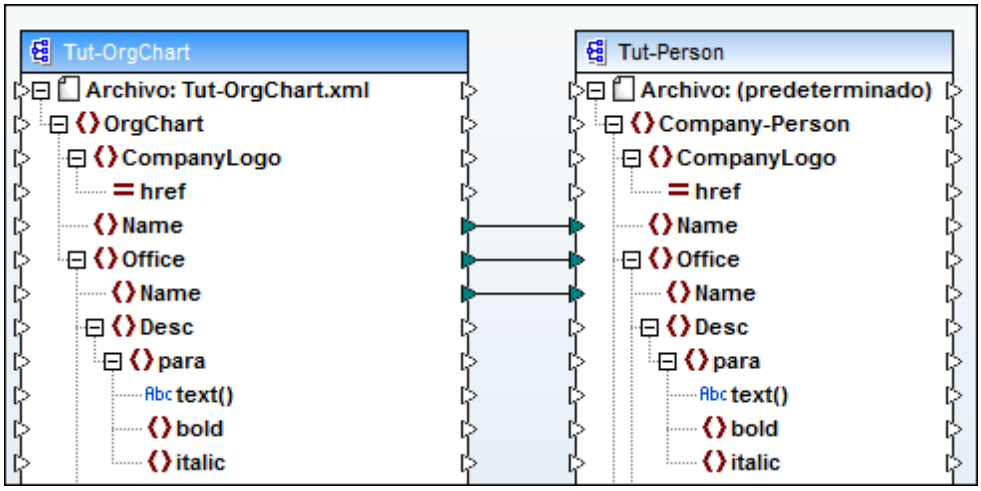
```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2005 sp2 U (http://www.altova.com) by Mr. Nobody (Altova GmbH) -->
<OrgChart xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="Tut-OrgChart.xsd">
  <CompanyLogo href="nanonull.gif"/>
  <Name>Organization 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>
```

Observe el orden en el que aparecen los nodos **`text`** y los nodos **`bold`**/**`italic`** de `Nanonull, Inc.` en el archivo XML:

```
<para> The company...
<bold>Vereno</bold>in 1995 ...
<italic>multi-core...conn_mixedContent.BMPFebruary 1999

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

Estado inicial de la asignación

Al abrir el archivo `Tut-Orgchart.mfd` este es el estado inicial de la asignación:



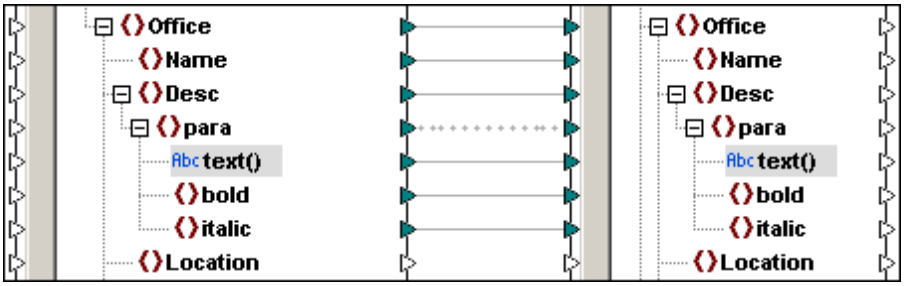
Resultado de esta asignación

El resultado de esta asignación inicial aparece a continuación. Observe que el resultado incluye `Organization Chart` y el nombre de cada una de las 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
```

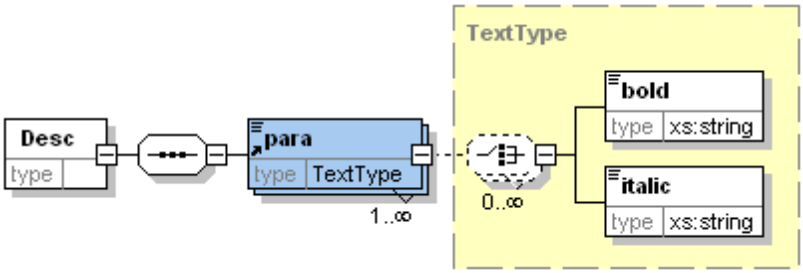
Asignación del elemento `para`

En la imagen siguiente podemos ver un ejemplo de asignación de contenido mixto. El elemento **`para`** tiene contenido mixto y el conector está dibujado con una línea **discontinua**. El nodo **`text()`** contiene los datos de texto y debe tener una asignación si queremos que aparezca en el componente de destino.



Si hace clic con el botón secundario en el conector y selecciona **Propiedades** en el menú contextual, puede añadir una anotación o etiqueta al conector. Para más información consulte el apartado Conexión de la *Referencia del usuario*.

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



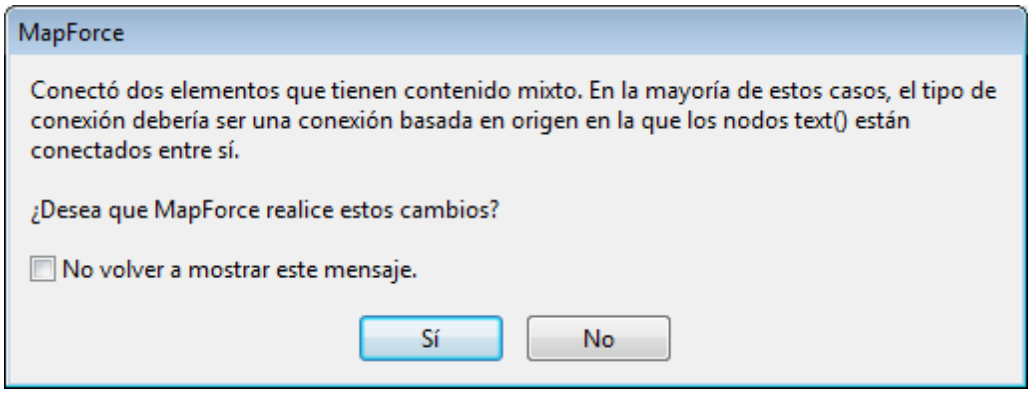
Fijémonos en las siguientes propiedades del elemento **`para`**:

- **para** es un elemento complexType con mixed="true" y su tipo es TextType
- los elementos **bold** e **italic** son de tipo "xs:string" y no se definieron como recursivos (es decir, ni **bold** ni **italic** son de tipo TextType)
- los elementos **bold** e **italic** pueden aparecer un número ilimitado de veces y en cualquier orden dentro del elemento **para**
- dentro del elemento **para** puede aparecer un número ilimitado de nodos de texto, intercalados con un número ilimitado de elementos **bold** e **italic**

Para crear conexiones de contenido mixto entre los elementos:

1. Active la conexión automática de secundarios equivalentes (**Conexión | Conectar automáticamente los secundarios equivalentes**).
2. Conecte el elemento **para** del esquema de origen con el elemento **para** del esquema de destino.

   Aparece un mensaje preguntando si MapForce debe definir los conectores como conectores basados en el origen.

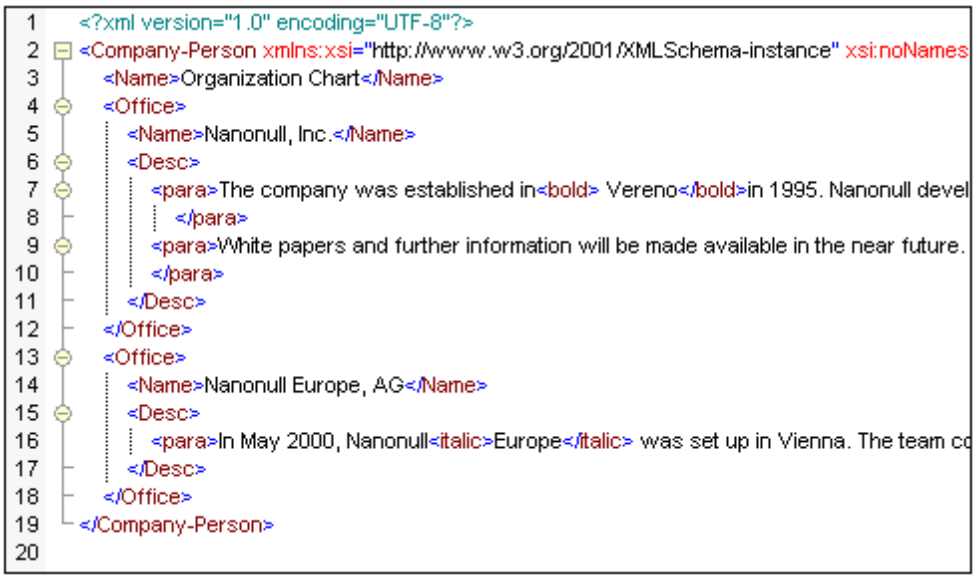


3. Haga clic en **Sí** para crear una conexión de contenido mixto.

   Nota: el elemento **para** tiene contenido mixto, por eso aparece este mensaje en este momento. Este mensaje también aparece si conecta los elementos **para** sin tener activada la opción **Conectar automáticamente los secundarios equivalentes**.

   Se crea una asignación entre todos los elementos secundarios de **para**. El conector que une los elementos **para** es una línea discontinua. Esto indica que es de contenido mixto.

4. Cambie al panel **Resultados** para ver el resultado de la asignación.



5. Haga clic en el icono **Ajuste automático de línea** de la barra de herramientas para ver todo el texto.

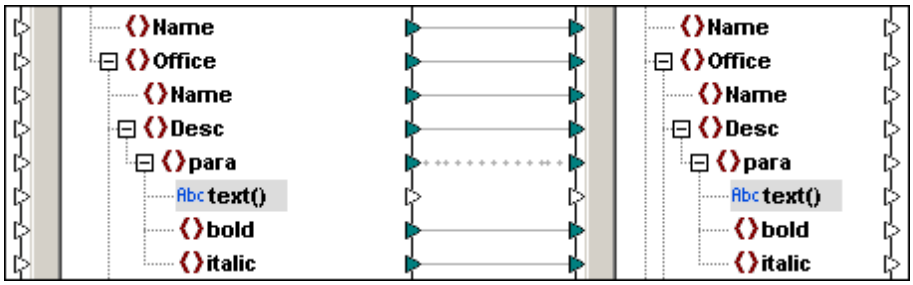
```
3     <Name>Organization Chart</Name>
4     <Office>
5       <Name>Nanonull, Inc.</Name>
6       <Desc>
7         <para>The company was established in<bold> Vereno</bold>in 1995. Nanonull develops
    nanoelectronic technologies for<italic>multi-core processors.</italic>February 1999 saw the
    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
```

La asignación del texto de contenido mixto de la descripción de cada oficina se realizó correctamente. El texto, así como el contenido de las etiquetas **`bold`** e **`italic`** se asignó tal y como aparece en el archivo XML de **origen**.

6. Vuelva al panel **Asignación**.

Para quitar los nodos de texto de los elementos de contenido mixto:

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



2. Cambie al panel **Resultados** para ver el resultado.

```
5        <Name>Nanonull, Inc.</Name>
6        <Desc>
7          <para>
8            <bold> Vereno</bold>
9            <italic>multi-core processors.</italic>
10           <bold>Nano-grid.</bold>
11           <italic>offshore</italic>
12         </para>
13         <para/>
14       </Desc>
15     </Office>
16     <Office>
17       <Name>Nanonull Europe, AG</Name>
18       <Desc>
19         <para>
20           <italic>Europe</italic>
21           <bold> five research scientists </bold>
22         </para>
23       </Desc>
```

Resultado:

- se eliminan todos los nodos de texto del elemento **`para`**
- se conserva el contenido textual asignado de las etiquetas `bold` e `italic`
- el **orden** de los elementos `bold` e `italic` se corresponde con el que tenían en el archivo XML de origen

Ejemplo de contenido mixto

Para este ejemplo utilizamos el archivo `ShortApplicationInfo.mfd`, guardado en la carpeta `...\MapForceExamples`.

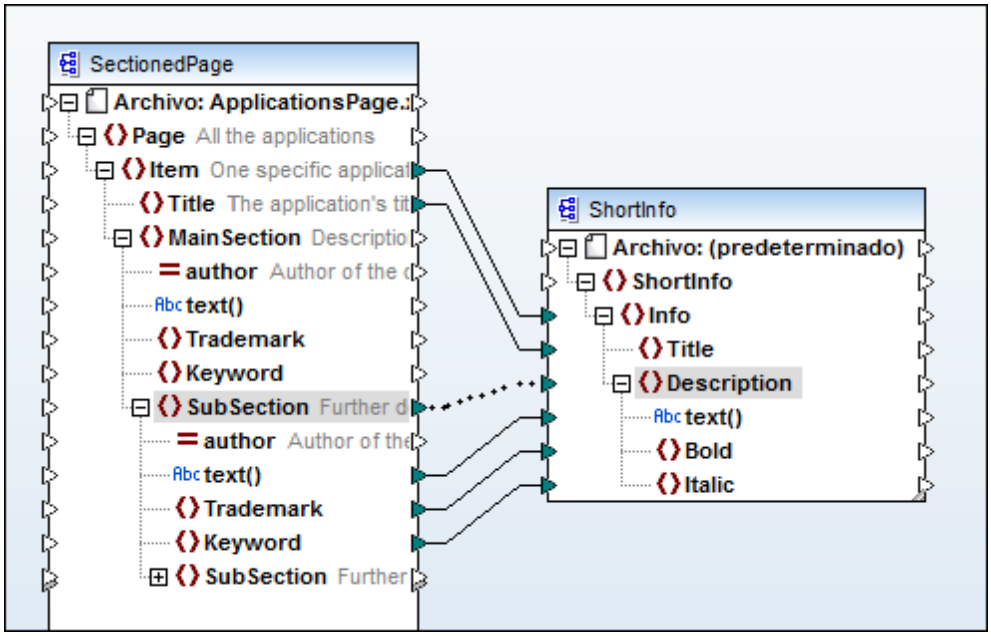
Aquí puede ver un pequeño fragmento del archivo XML de origen utilizado en este ejemplo.

```
<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 la asignación. Observe que:

- el conector del elemento **`SubSection`** es de contenido mixto y está asignado al elemento **`Description`** del archivo/esquema XML de destino.
- hay una asignación entre los nodos **`text()`**
- el texto del elemento **`Trademark`** está asignado al elemento **`Bold`** del componente de destino
- el texto del elemento **`Keyword`** está asignado al elemento **`Italic`** del componente de destino

 © 2014 Altova GmbH



Resultado de la asignación

La asignación del texto de contenido mixto de cada descripción se realizó correctamente. El texto, así como el contenido de las etiquetas `bold` e `italic`, se asignó en el orden que tenían en el archivo XML de origen.

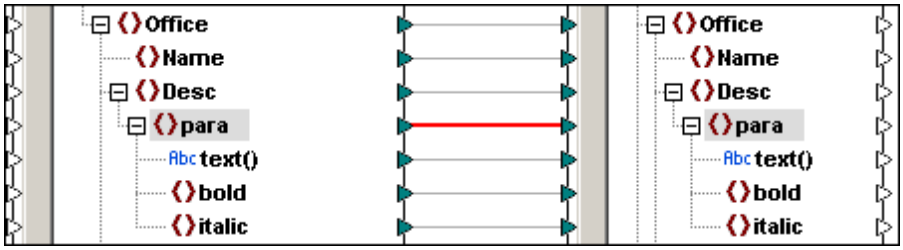
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ShortInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="
   C:/PROGRA~1/Altova/MapForce2005/MapForceExamples/ShortInfo.xsd">
3    <Info>
4      <Title>XMLSpy</Title>
5      <Description>Altova <Bold>XMLSpy</Bold> 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>
```

Usar asignaciones estándar en elementos de contenido mixto

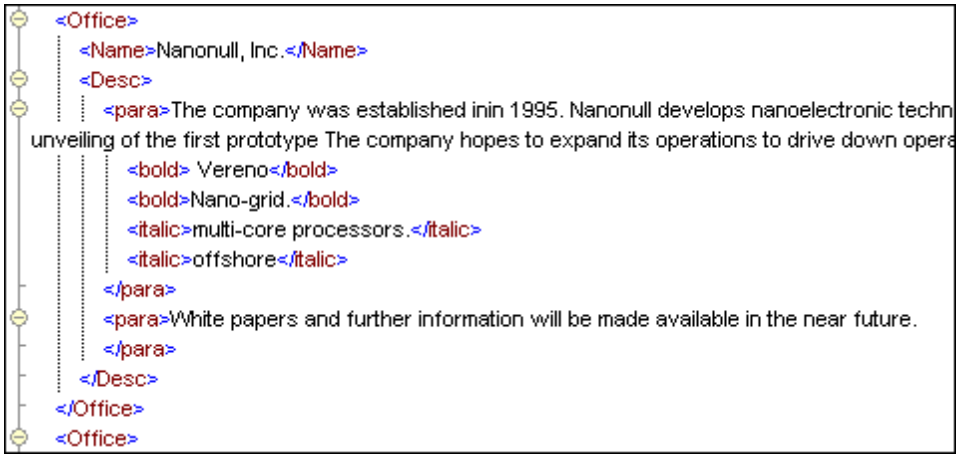
Esta sección describe los resultados que se obtienen al definir asignaciones **estándar** (usando conectores estándar) para nodos de **contenido mixto**. Para este ejemplo utilizamos el archivo `Tut-OrgChart.mfd`, que está en la carpeta `...\MapForceExamples\Tutorial`.

Para crear conexiones estándar entre elementos de contenido mixto:

1. Cree un conector entre los dos elementos **`para`**.
   Aparece un mensaje preguntando si MapForce debe definir los conectores como conectores basados en el origen.
2. Haga clic en **No** para crear una asignación estándar.



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



Resultado
Si asignamos elementos de contenido mixto usando asignaciones estándar, obtenemos este resultado:

- Se admite/se puede asignar el contenido de **`Text()`**
- Las etiquetas iniciales/finales de los elementos secundarios (**`bold`** e **`italic`**) se quitan del nodo de texto.
- Los nodos secundarios aparecen **después** del nodo de texto.
- El **orden** de los nodos secundarios depende del orden de los elementos del archivo/esquema XML de **destino**.

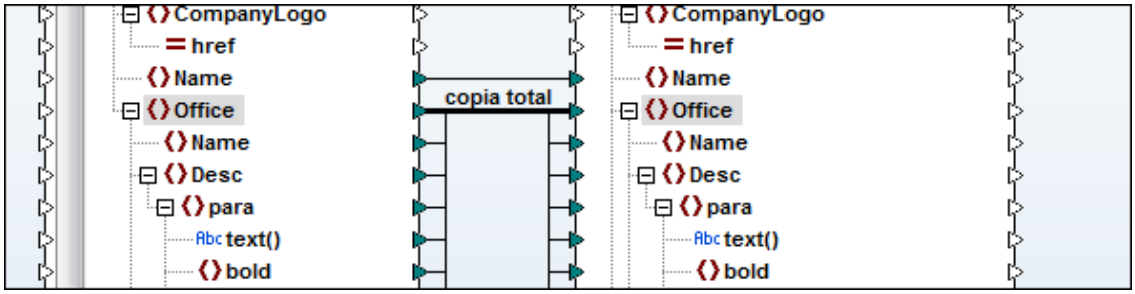
Es decir, por cada elemento **`para`**, se presenta primero el nodo **`text()`**, después todos los elementos **`bold`** y después todos los elementos **`italic`**.

 © 2014 Altova GmbH

9.1.3 Conexiones de copia total

Este tipo de conexión permite trabajar en un área de trabajo más sencillo y conectar automáticamente **todos** los elementos que sean idénticos. Esto significa que los elementos de origen se **copian** en el componente de destino si el elemento de origen y el de destino tienen **el mismo tipo** y si el elemento de destino es de tipo **`xs:anyType`**.

Si el tipo de origen y el tipo de destino **no son idénticos** y el tipo de destino no es **`xs:anyType`**, los datos de origen se transfieren/asignan a los elementos de destino que tengan el mismo nombre y el mismo nivel dentro de la jerarquía. Si el nombre de los elementos de destino es diferente, entonces no se crea la asignación.



Los **conectores de copia total** se dibujan con una línea sólida gruesa que conecta todos los elementos que son idénticos.

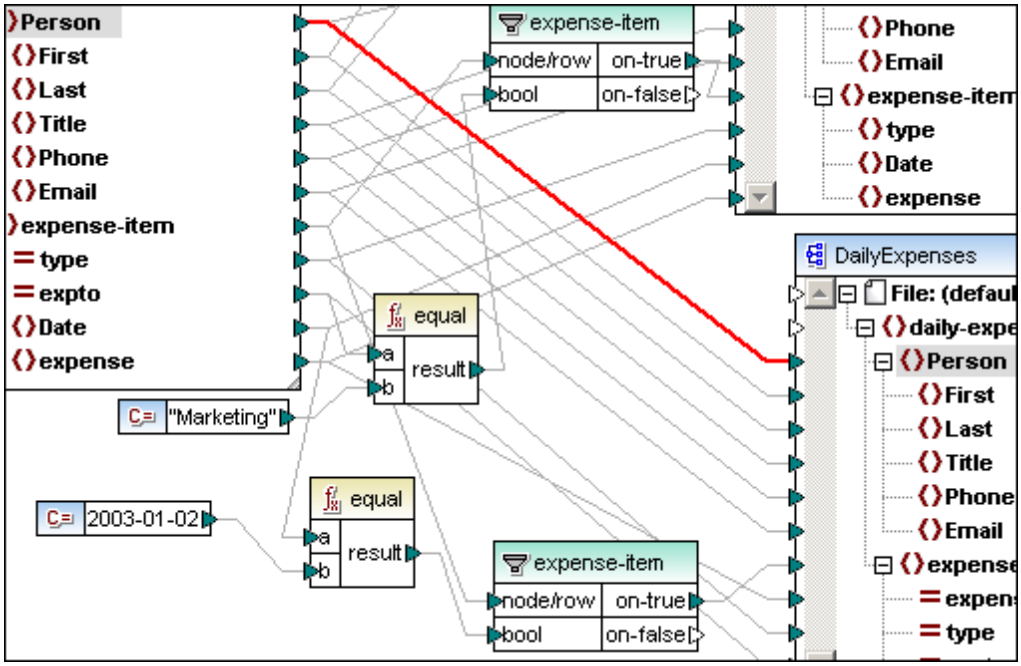
Tenga en cuenta que solamente se compara el nombre de los elementos secundarios, no su tipo.

Puede usar conexiones de copia total para crear asignaciones (i) entre elementos complexType de esquemas XML y (ii) componentes complejos (esquemas XML, bases de datos, archivos EDI) y componentes/funciones definidas por el usuario que contienen los mismos parámetros complejos.

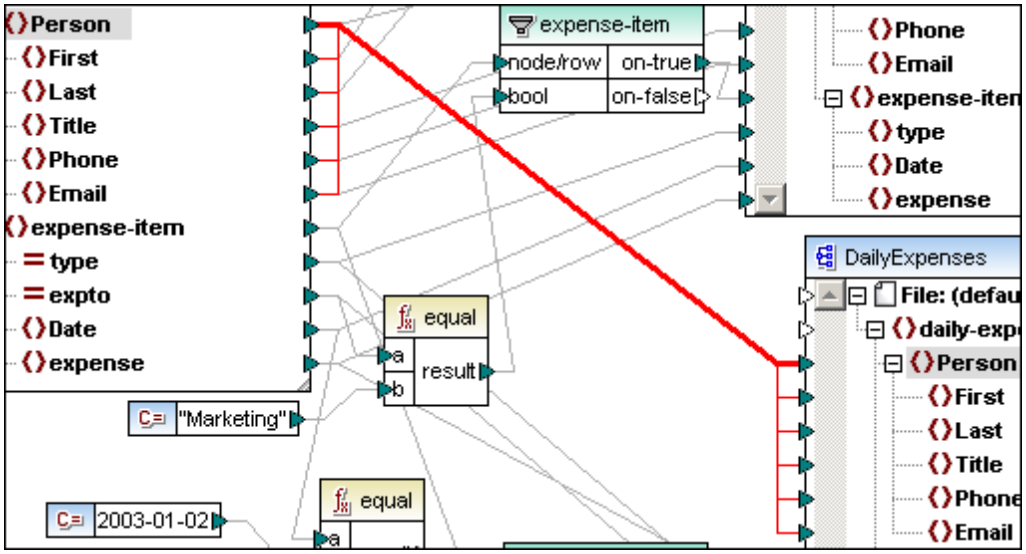
Para el ejemplo que aparece a continuación usamos el archivo `MarketingAndDailyExpenses.mfd`, guardado en la carpeta `...\MapForceExamples`.

Para definir una copia de conexión total:

1. Haga clic con el botón secundario en un conector (p. ej. en el conector de **`Person`**) y seleccione **Copia total** en el menú contextual.
   Aparece un mensaje recordándole que la conexión de copia total reemplazará todas las conexiones a los elementos secundarios de destino.



2. Haga clic en **Aceptar** para crear conectores de copia total.



Se crean conectores entre todos los elementos de origen y destino que tienen el mismo nombre.

Nota:

- Cuando se eliminan conexiones de destino, se eliminan también los conectores desde otros componentes de origen o desde funciones.
- Este tipo de conexión no se puede crear entre un elemento y el elemento raíz de un componente de esquema.
- En las conexiones de copia total no puede eliminar un conector y no otros ni reconectar los conectores por separado.

Para resolver o eliminar conectores de copia total:

1. Conecte cualquier elemento con un elemento secundario de la copia de conexión total

del componente de destino.
Aparece un diálogo informándole de que en el elemento de destino solamente puede haber un conector. Haga clic en **Reemplazar la conexión**.

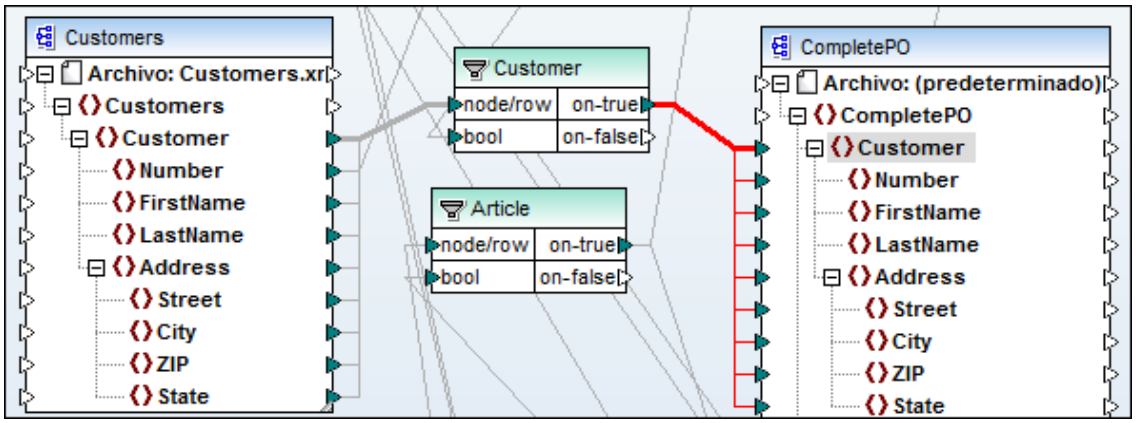
2. Aparece otro cuadro de diálogo. Haga clic en **Resolver la conexión de copia total**.
La conexión de copia total se sustituye por varios conectores.

Conexiones de copia total y funciones definidas por el usuario
Cuando cree una conexión de copia total entre un esquema y un parámetro de una función definida por el usuario, es obligatorio que los dos componentes estén basados en el mismo esquema. Sin embargo, no es necesario que tengan el mismo elemento raíz. Consulte el apartado Definir componentes de salida complejos para ver un ejemplo.

Conexiones de copia total y filtros
También puede crear conexiones de copia total que pasen por componentes de filtrado, siempre y cuando el componente de origen:

- esté formado por datos estructurados, es decir, un componente de esquema, de BD o EDI,
- reciba datos de un parámetro de salida complejo de una función definida por el usuario o de un servicio web, o
- reciba datos de otro componente de filtrado.

Al componente de destino se pasan solamente los datos filtrados.



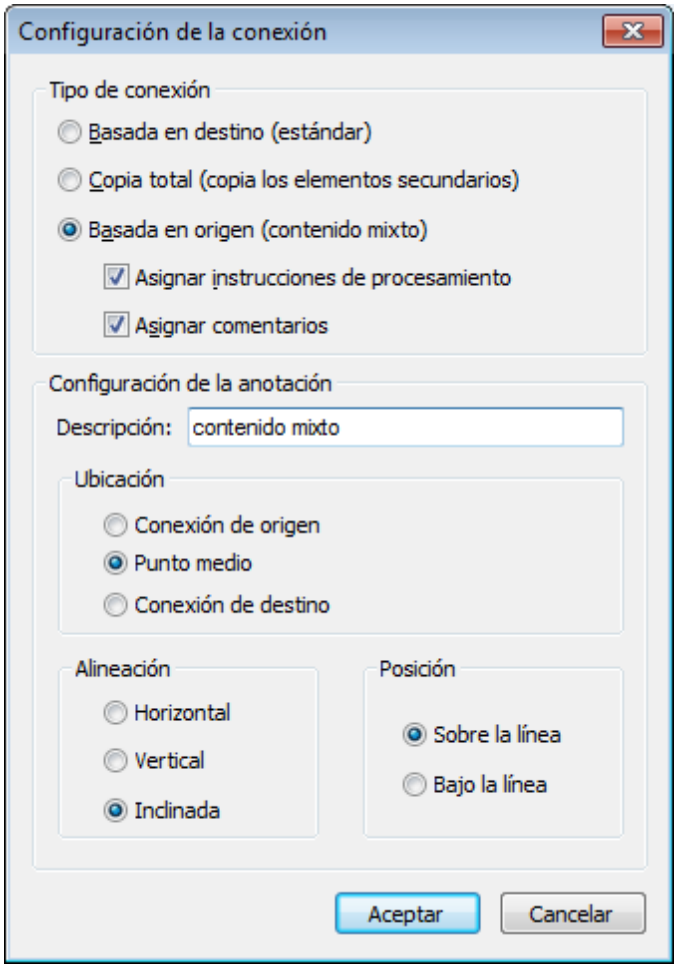
Para definir una conexión de copia total a través de un componente de filtrado:

1. Cree un conector desde el elemento `on-true/on-false` hasta el elemento de destino (p. ej. `Customer`).
2. Haga clic con el botón secundario en el conector y seleccione **Copia total (copia los elementos secundarios)** en el menú contextual.
Se crea un componente de copia total entre los elementos que tienen el mismo nombre.

   Nota: consulte el apartado Mover y restablecer conectores para conocer las posibilidades a la hora de eliminar componentes de filtrado.

9.2 Configuración de la conexión

Si hace clic con el botón secundario en un conector y selecciona **Propiedades** en el menú contextual o hace doble clic en el título del conector, se abre el cuadro de diálogo "Configuración de la conexión". Aquí puede definir opciones de configuración para el conector. Recuerde que si una opción aparece atenuada es porque no está disponible en ese caso concreto.

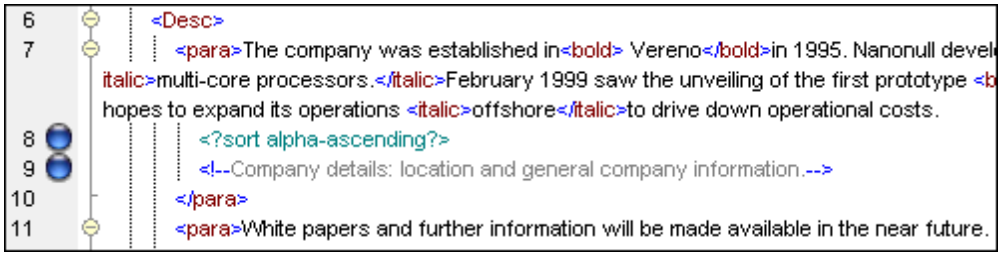


Tipo de conexión
Para los elementos complexType (incluso para los que no contienen nodos de texto), puede elegir uno de estos tipos de conexión:

- **Basada en destino (estándar):** el conector se convierte en una asignación estándar (ver el apartado Basada en el destino (asignación estándar)).
- **Copia total (copia los elementos secundarios):** el conector se convierte en una asignación de copia total. Es decir, todos los elementos del componente de origen y destino que son idénticos se conectan automáticamente (ver el apartado Conexiones de copia total).
- **Basada en origen (contenido mixto):** el conector se convierte en una asignación basada en el origen (de contenido mixto). También puede seleccionar qué otros elementos quiere asignar, siempre y cuando sean **elementos secundarios** del elemento asignado en el archivo XML de origen.

 © 2014 Altova GmbH

Marque las casillas *Asignar instrucciones de procesamiento* y *Asignar componentes* para incluir esos datos en el archivo de salida.



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

Configuración de la anotación

A cada conector puede ponerle una etiqueta, lo cual permite comentar en detalle la asignación. Las opciones de los recuadros *Ubicación*, *Alineación* y *Posición* se habilitan nada más rellenar el campo *Descripción*. Puede crear anotaciones para **todos los tipos de conexión**.

Para añadir una anotación a un conector:

1. Haga doble clic en el conector. Se abre el cuadro de diálogo "Configuración de la conexión".
2. Escriba el nombre del conector seleccionado en el campo *Descripción*.
   Se habilitan las demás opciones del grupo *Configuración de la anotación*.
3. Defina la *ubicación*, *alineación* y *posición* de la anotación.
4. Active el icono **Mostrar anotaciones** de la barra de herramientas Opciones de vista.

Nota: las anotaciones se pueden ver aunque el icono **Mostrar anotaciones** está desactivado.

Para ello active el icono **Mostrar información sobre herramientas** y pase el puntero del mouse sobre el conector.

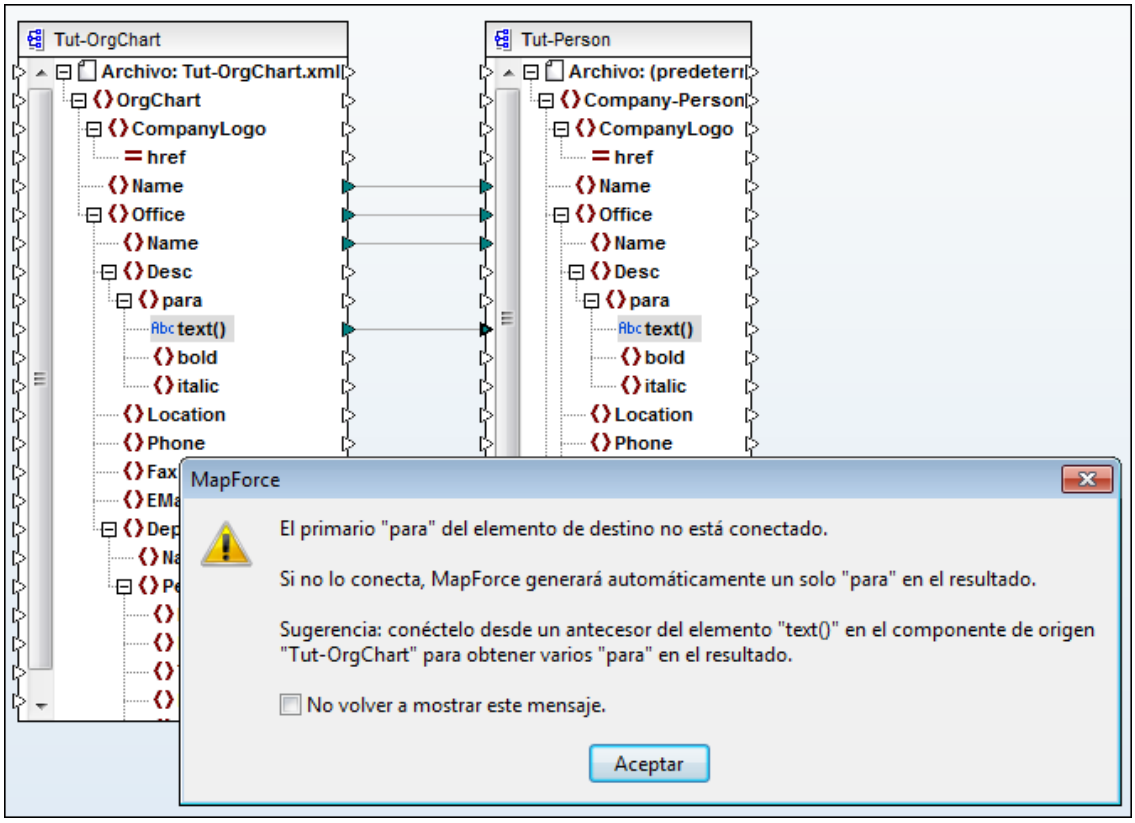
9.3 Conexiones y resultados de la asignación

Cuando las conexiones entre elementos de origen y destino se crean a mano, MapForce analiza automáticamente los posibles resultados de la asignación. Por eso, si crea una asignación entre dos elementos secundarios, puede que aparezca un mensaje sugiriendo que conecte también el primario del elemento de origen con el primario del elemento de destino.

Así evitamos tener un solo elemento secundario en la vista previa de los resultados de la asignación en el panel **Resultados**. Este suele ser el caso cuando el nodo de origen incluye una secuencia de elementos en lugar de un solo valor.

Para este ejemplo utilizamos el archivo `Tut-OrgChart.mfd` guardado en la carpeta `...\MapForceExamples\Tutorial`.

Si conectamos el elemento de origen **`text()`** con el elemento de destino **`text()`**, aparece un mensaje informándonos de que el elemento primario **`para`** no está conectado y que **`para`** aparecerá una sola vez en el archivo de salida. Para generar varios elementos **`para`** en el archivo de salida, cree una asignación entre los elementos **`para`** de origen y destino.



 © 2014 Altova GmbH

9.4 Secuencia de procesamiento de los componentes

En MapForce puede trabajar con asignaciones que tienen varios componentes de destino. Cada componente de destino tiene un botón de vista previa, lo cual permite ver los resultados de la asignación para cada componente.

Si la asignación se ejecuta desde la línea de comandos (o se ejecuta el código generado), entonces se ejecuta la asignación entera, independientemente del botón de vista previa seleccionado, y se genera el resultado de todos los componentes de destino.

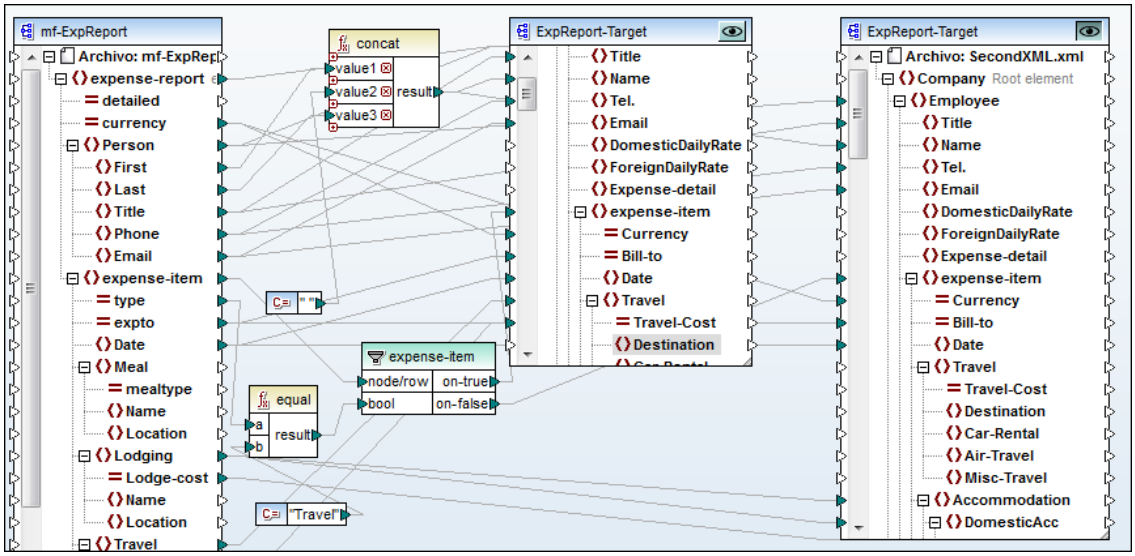
El orden en el que se procesan los componentes de destino se puede alterar cambiando la posición de los componentes de destino en el panel **Asignación**. La **posición** de un componente viene definida por la posición de su ángulo superior izquierdo.

Los componentes de destino se procesan según su posición Y-X en la pantalla, de arriba a abajo y de izquierda a derecha.

- Si hay dos componentes con la misma posición vertical, tiene prioridad el componente situado más a la izquierda.
- Si hay dos componentes con la misma posición horizontal, tiene prioridad el componente situado más arriba.
- En el caso poco probable de que dos componentes tengan exactamente la misma posición vertical y horizontal, se usa automáticamente el Id. único del componente. Esto garantiza un orden bien definido, que no se puede cambiar.

En la imagen siguiente aparece la asignación `Tut-ExpReport-multi.mfd`, que está en la carpeta `...\MapForceExamples\Tutorial`.

Ambos componentes de destino (**ExpReport-Target**) tienen la misma posición **vertical** y el botón de vista previa está activo en el componente de destino de la derecha.



Tras seleccionar **XSLT2** y generar el código:

- Primero se procesa el componente de destino de la izquierda, que genera el archivo `ExpReport.xml`.

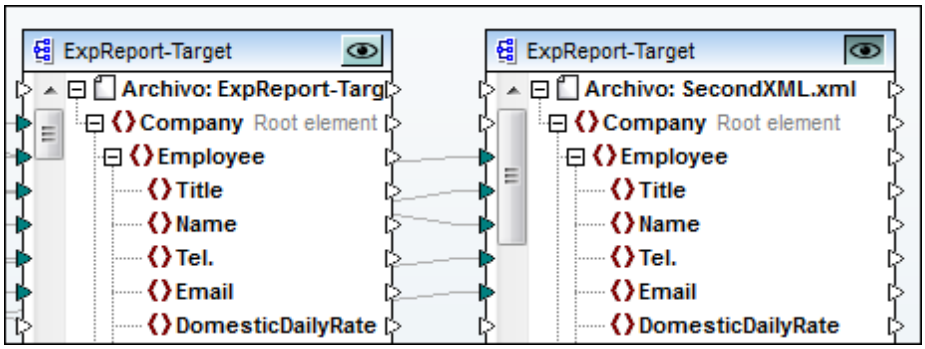
- Después se procesa el componente de la derecha, que genera el archivo `SecondXML.xml`.

Para comprobarlo, abra el archivo `DoTransform.bat` (situado en la carpeta de salida indicada), que muestra el orden en el que se generan los archivos de salida. `ExpReport-Target.xml` es el primer archivo de salida generado por el archivo por lotes y `SecondXML.xml` es el segundo.

```
@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 de la izquierda y póngalo debajo del componente de la derecha.



2. Vuelva a generar el código y mire 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 se genera primero `SecondXML.xml` y después `ExpReport-Target.xml`.

Asignaciones en cadena

La secuencia de procesamiento descrita en este apartado es la utilizada para las asignaciones en cadena. Sin embargo, el grupo de la asignación en cadena se entiende como una sola unidad. Por eso, cambiar la posición del componente intermedio o final de la asignación en cadena no

tiene efecto alguno en la secuencia de procesamiento.

La posición de los componentes de destino **finales** determina el orden de procesamiento solamente si se trata de una asignación con varias cadenas o componentes de destino.

- Si hay dos componentes finales con la misma posición vertical, tiene prioridad el componente situado más a la izquierda.
- Si hay dos componentes con la misma posición horizontal, tiene prioridad el componente situado más arriba.
- En el caso poco probable de que dos componentes finales tengan exactamente la misma posición vertical y horizontal, se usa automáticamente el Id. único del componente. Esto garantiza un orden bien definido, que no se puede cambiar.

9.5 Asignaciones en cadena / componentes de paso a través

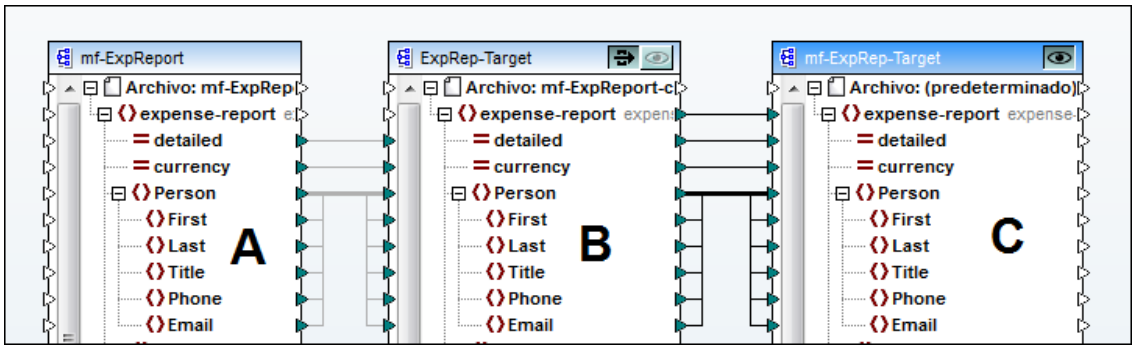
En MapForce puede trabajar con asignaciones compuestas por varios componentes encadenados. Las asignaciones en cadena son asignaciones en las que al menos uno de los componentes funciona tanto como componente de origen como de destino. Dichos componentes crean resultados que se utilizan como entrada para el siguiente paso de la cadena de asignación. Estos componentes se denominan componentes *intermedios*.

Las asignaciones en cadena presentan una característica particular conocida como *paso a través.* Esta característica sirve para crear archivos de salida de los componentes intermedios y obtener una vista previa de las distintas etapas de una asignación en cadena en el panel **Resultados**. Para obtener una vista previa de las asignaciones se usa el motor de ejecución integrado, que utiliza archivos temporales para generar los resultados.

Si la asignación se ejecuta desde la línea de comandos (o se ejecuta el código generado), entonces se ejecuta la cadena de asignación entera, independientemente de los archivos XML de entrada y salida asignados al componente intermedio. Además, el resultado de un paso anterior de la cadena de asignación se usa como entrada para el siguiente paso.

Nota: la característica *paso a través* solamente es compatible con componentes intermedios basados en un archivo (XML, CSV, TXT, etc.). Los componentes basados en BD pueden ser componentes intermedios pero no incluyen el botón de paso a través. El componente intermedio se vuelve a generar al ver la vista previa o al generar código. Esto no se puede hacer con una BD, porque habría que borrarla antes de volver a generarla.

En el ejemplo de la imagen siguiente puede ver una asignación en cadena con tres componentes (A, B y C). El componente C es el componente de destino. El componente B (**ExpRep-Target**) es el componente *intermedio* porque tiene conexiones de entrada y salida.



Recuerde que si la asignación en cadena se ejecuta desde la línea de comandos (o se ejecuta el código generado) la asignación ejecuta todos los pasos en el orden correcto y genera los archivos de salida pertinentes.

Botón de vista previa

Observe que los componentes B y C tienen un *botón de vista previa*. Con estos botones puede ver los resultados intermedios de B o el resultado final de la asignación en cadena de C en el motor de ejecución integrado. Pulse uno de los dos botones de vista previa y cambie al panel **Resultados** para ver el resultado del paso elegido.

Si en el componente intermedio está activado el botón de paso a través, se deshabilita automáticamente el botón de vista previa. Por eso, si desea ver una vista previa de un

 © 2014 Altova GmbH

componente intermedio, desactive primero el botón de paso a través y después pulse el botón de vista previa.

Botón de paso a través

En la barra de título del componente intermedio B hay otro botón, denominado *botón de paso a través*.

Si el botón de paso a través está **activo** , se usan todos los datos (del componente A al B y después al C) para la vista previa de resultados. Se crean 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 de paso a través está **desactivado** , se ejecuta solamente una parte de la cadena de asignación. Dependiendo de qué botones de vista previa están activos, MapForce genera uno de estos dos resultados:

- el resultado de asignar el componente A al componente B (si está activo el botón de vista previa del componente B). Es decir, la cadena de asignación se para en el componente B y el componente C no se usa en absoluto.
- el resultado de asignar el componente intermedio B al componente C (si está activado el botón de vista previa del componente C). Cuando se desactiva el paso a través, se interrumpe la asignación automática del componente B y solamente se ejecuta la parte derecha de la cadena de asignación. El componente A no se utiliza.

  Nota: si la asignación se ejecuta desde la línea de comandos (o se ejecuta el código generado), entonces se genera el resultado de todos los componentes, independientemente de qué botón de vista previa está activo y del estado del botón de paso a través del componente B.

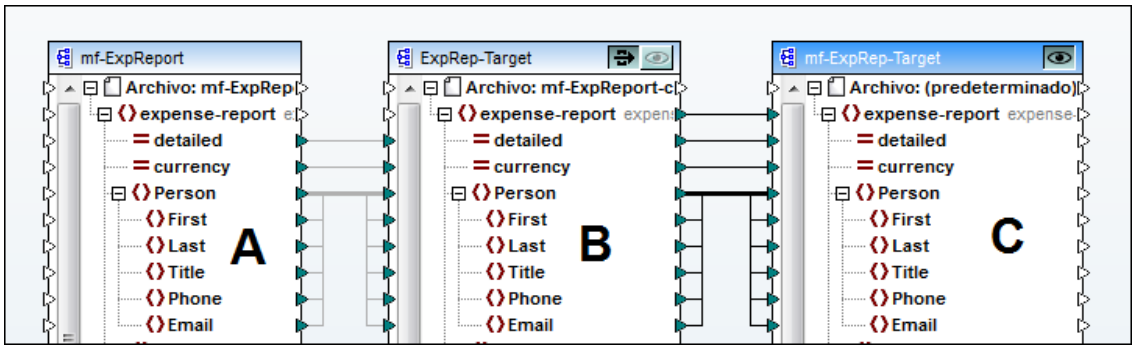
En nuestro ejemplo se generarán dos archivos de resultados. Esto se debe a que MapForce analiza automáticamente las dependencias entre todos los componentes y genera todos los resultados de los componentes intermedios y de destino en el orden correcto.

Puesto que el botón de paso a través no está activo, es crucial que el componente intermedio B tenga nombres de archivo idénticos en los campos *Archivo XML de entrada* y *Archivo XML de salida* del cuadro de diálogo "Configuración del componente".

Para más información sobre el ejemplo anterior siga leyendo esta sección. Los apartados siguientes también explican cómo se transfieren los datos de origen cuando el botón de paso a través está activo o inactivo. Para ver otro ejemplo, consulte el apartado Ejemplo de asignación en cadena.

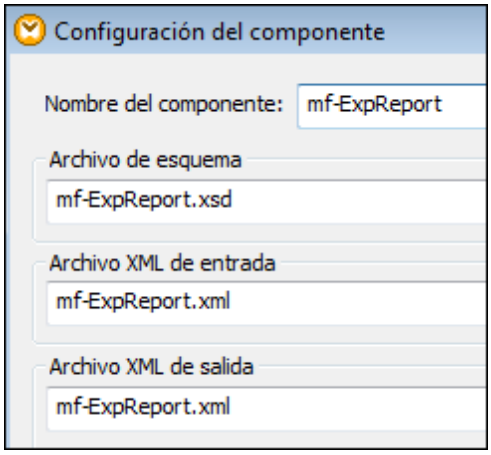
9.5.1 Asignaciones en cadena: paso a través activo

Para este ejemplo usamos el archivo `Tut-ExpReport-chain.mfd`, disponible en la carpeta `...\MapForceExamples\Tutorial`.



La asignación del archivo `Tut-ExpReport-chain.mfd` (*imagen anterior*) se configuró de la siguiente manera:

- El **componente A** suministra todos los datos necesarios para la asignación, a partir de un archivo XML de muestra. El archivo XML (`mf-ExpReport.xml`) aparece en el campo *Archivo XML de entrada* del cuadro de diálogo "Configuración del componente". El nombre del archivo XML de salida se inserta automáticamente en el campo *Archivo XML de salida* (se usa el mismo nombre para el archivo de entrada y de salida).

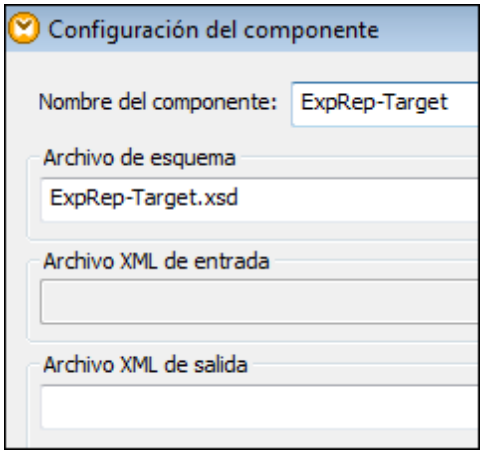


- El **componente intermedio B** con el paso a través activo:
  cuando el paso a través está activo, el campo *Archivo XML de entrada* del componente intermedio se desactiva automáticamente. No es necesario tener un nombre para el archivo porque los datos intermedios se almacenan en archivos temporales.

  Si no se define un archivo XML de salida, se usará un nombre de archivo predeterminado. Si se define el archivo XML de salida, se usa el mismo nombre para el archivo de salida intermedio.

  Recuerde que los componentes intermedios pueden tener nombres de archivo dinámicos (es decir, conectores al elemento **`Archivo:`** e incluso comodines de nombre de archivo). Para más información consulte el apartado Archivos de entrada/salida dinámicos por componente.

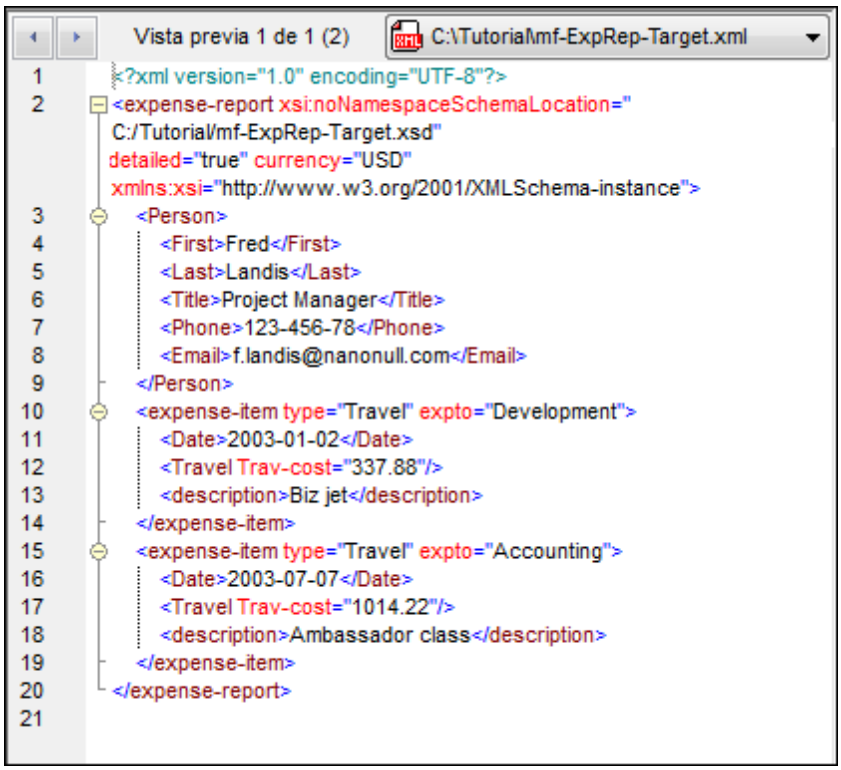
 © 2014 Altova GmbH



- El **componente final C** no tiene asociado un archivo XML de salida. El botón de vista previa del componente C está activo.

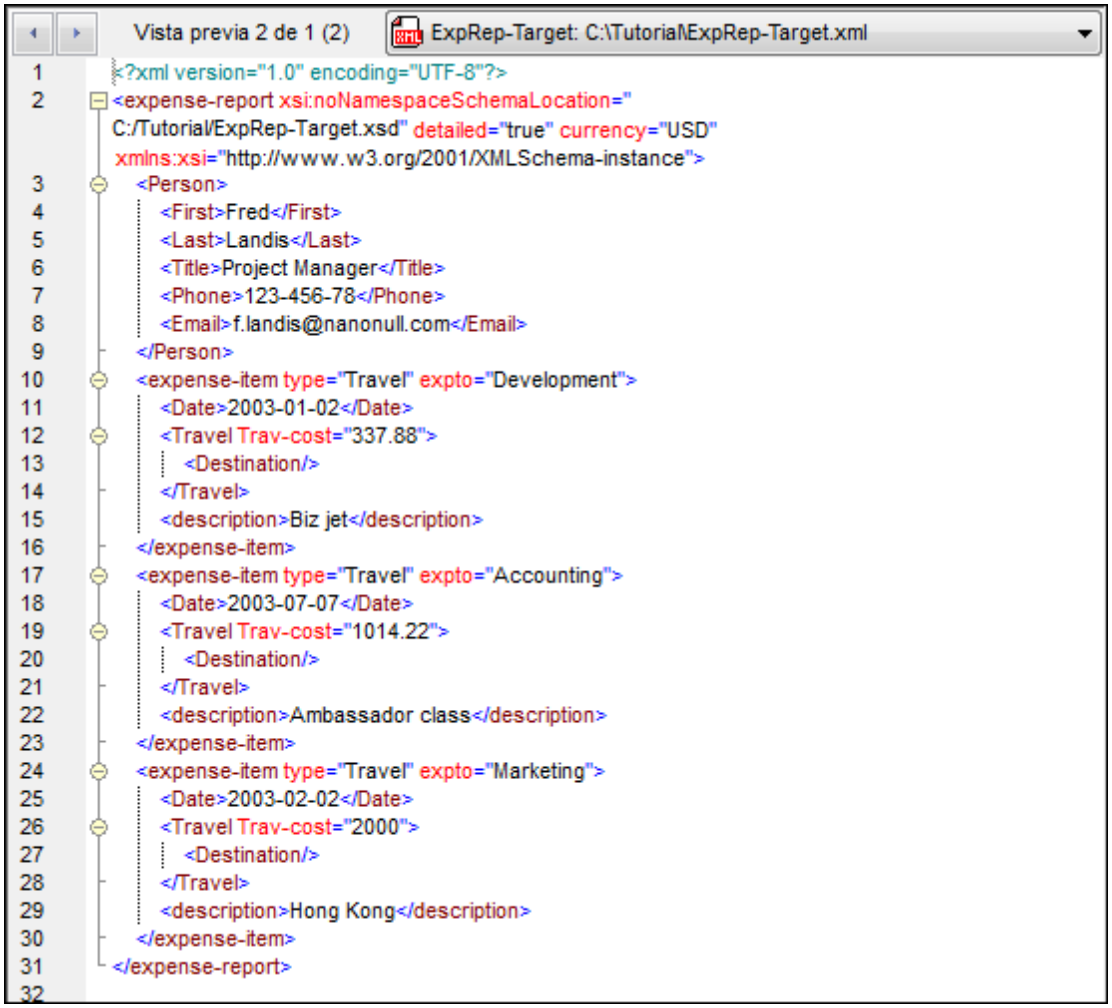
Pase al panel **Resultados** para ver una vista previa de los resultados en el motor de ejecución integrado.

Vista previa 1:
Muestra el resultado final de asignar los datos del componente A al componente final C (a través del componente intermedio B). Se trata de un archivo con todos los gastos de viaje por un importe inferior a 1500.

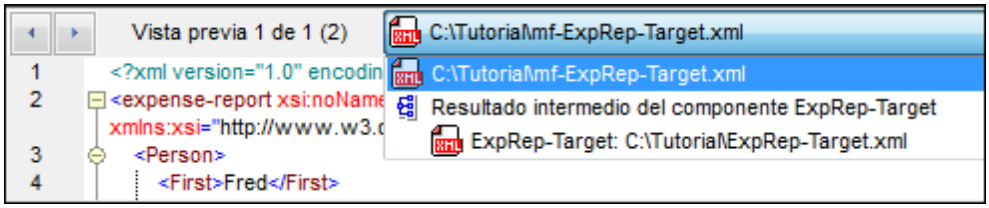


Vista previa 2:
Muestra el resultado de asignar los datos del componente A al componente intermedio B (es decir, todos los gastos de viaje). `ExpRep-Target.xml` es un nombre de archivo predeterminado

que se genera automáticamente porque no se introdujo un nombre de archivo en el campo *Archivo XML de salida* del componente B.

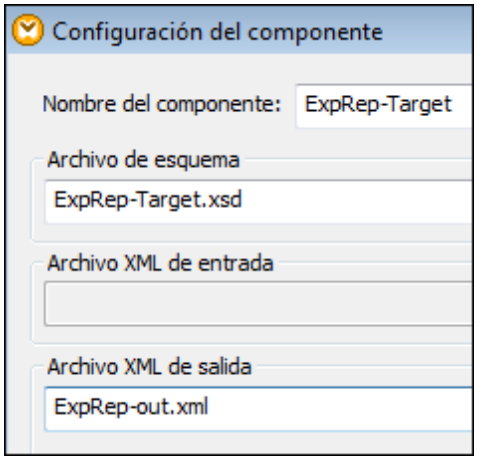


Nota importante:
Cada resultado tiene una ventana de vista previa propia. Haga clic en la flecha de la lista desplegable para ver los demás resultados.

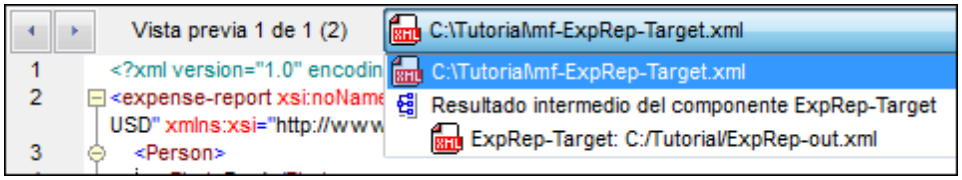


En la lista desplegable aparecen todos los archivos de resultados. El resultado final aparece al principio de la lista y el resultado intermedio al final. Haga clic en el resultado que desea ver o use las teclas de dirección del teclado para navegar por la lista y pulse **Entrar**.

 © 2014 Altova GmbH



Si define un archivo XML de salida para el componente intermedio B (p. ej. `ExpRep-out.xml`, como en la imagen anterior) los datos intermedios del componente B se guardan en un archivo con ese nombre (`ExpRep-out.xml`).



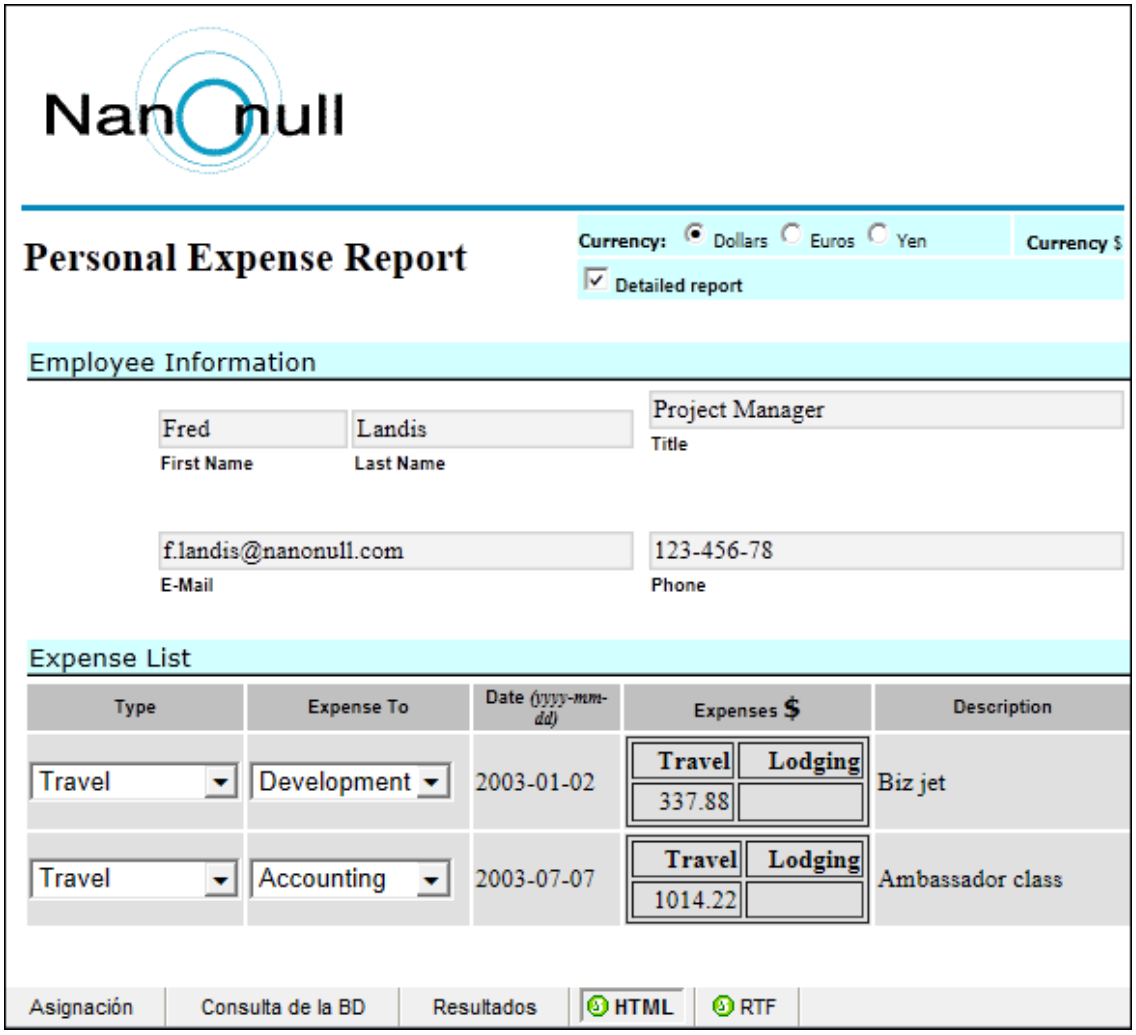
Cuando está activo el botón de paso a través, los archivos creados por el componente intermedio se guardan **automáticamente** como archivos temporales y se usan para seguir procesando los resultados de ese componente.

La opción de configuración *Escribir directamente en los archivos de salida finales* (**Herramientas | Opciones | General**) determina si los archivos intermedios se guardan como archivos temporales o como archivos físicos. Para los componentes intermedios se usa un nombre de archivo predeterminado a la hora de guardar el resultado intermedio, a no ser que se suministre/ asigne un nombre de archivo dinámico.

En la barra superior del panel **Resultados**, *Vista previa X* ***de 1*** hace referencia al número de componentes de destino finales (en este caso 1). Por su parte, Vista previa ... **(2)** hace referencia al número total de resultados, incluidos los resultados de los componentes intermedios.

Visualizar los resultados con StyleVision

Si asoció un archivo SPS al componente de destino, puede ver los datos resultantes en los paneles **HTML/RTF/PDF/Word** de StyleVision de MapForce.

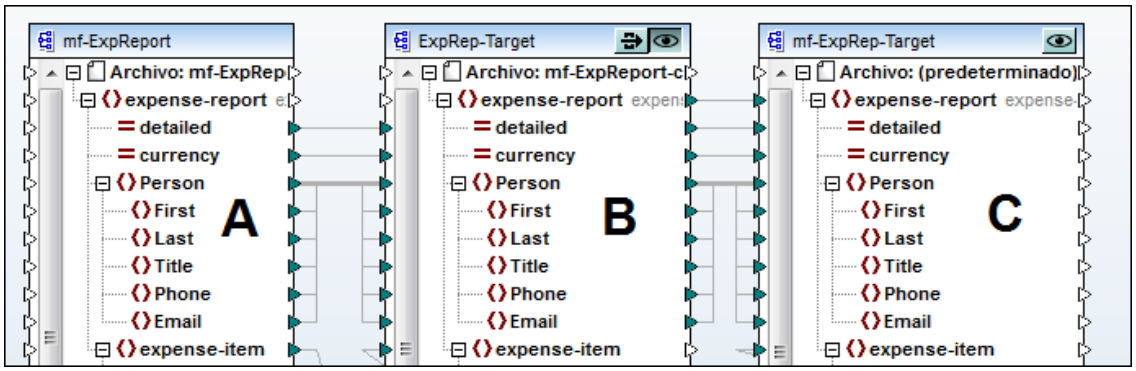


Recuerde que en los paneles de StyleVision de MapForce solamente se representan los resultados de los componentes de destino finales de la asignación en cadena. En estos paneles no se puede ver el resultado de componentes intermedios.

 © 2014 Altova GmbH

9.5.2 Asignaciones en cadena: paso a través inactivo

El archivo de ejemplo `Tut-ExpReport-chain.mfd` da resultados diferentes si el botón de paso a través del componente B está **inactivo**.



La transferencia automática de datos desde el componente A hasta el componente C (pasando por B) se interrumpe al desactivar el botón de paso a través. Los botones de vista previa de los componentes B y C determinan ahora qué parte de la cadena de asignación se genera.

MapForce genera los resultados del componente cuyo botón de vista previa está activo.

- Si está activo el botón de vista previa del componente **B**, entonces se genera el resultado de asignar los datos del componente A al B. El componente C no se utiliza.

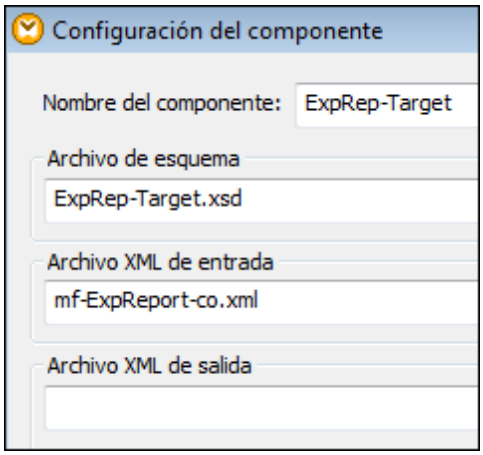
  Al cambiar al panel **Resultados** se pueden ver los resultados en el motor de ejecución integrado.

  Vista previa:
  Muestra el resultado de asignar los datos del componente A al componente intermedio B (es decir, todos los gastos de viaje).

Vista previa 2 de 1 (2) ExpRep-Target: C:\Tutorial\ExpRep-Target.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <expense-report xsi:noNamespaceSchemaLocation="
   C:/Tutorial/ExpRep-Target.xsd" detailed="true" currency="USD"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
3    <Person>
4      <First>Fred</First>
5      <Last>Landis</Last>
6      <Title>Project Manager</Title>
7      <Phone>123-456-78</Phone>
8      <Email>f.landis@nanonull.com</Email>
9    </Person>
10   <expense-item type="Travel" expto="Development">
11     <Date>2003-01-02</Date>
12     <Travel Trav-cost="337.88">
13       <Destination/>
14     </Travel>
15     <description>Biz jet</description>
16   </expense-item>
17   <expense-item type="Travel" expto="Accounting">
18     <Date>2003-07-07</Date>
19     <Travel Trav-cost="1014.22">
20       <Destination/>
21     </Travel>
22     <description>Ambassador class</description>
23   </expense-item>
24   <expense-item type="Travel" expto="Marketing">
25     <Date>2003-02-02</Date>
26     <Travel Trav-cost="2000">
27       <Destination/>
28     </Travel>
29     <description>Hong Kong</description>
30   </expense-item>
31 </expense-report>
32
```

- Por el contrario, si está activo el botón de vista previa del componente **C**, MapForce asigna los datos del componente intermedio B al componente C (el componente A no se utiliza). El componente B tiene asociado un archivo XML de entrada (`mf-ExpReport-co.xml`). Consulte el subapartado Guardar el resultado intermedio de una asignación que aparece más abajo.



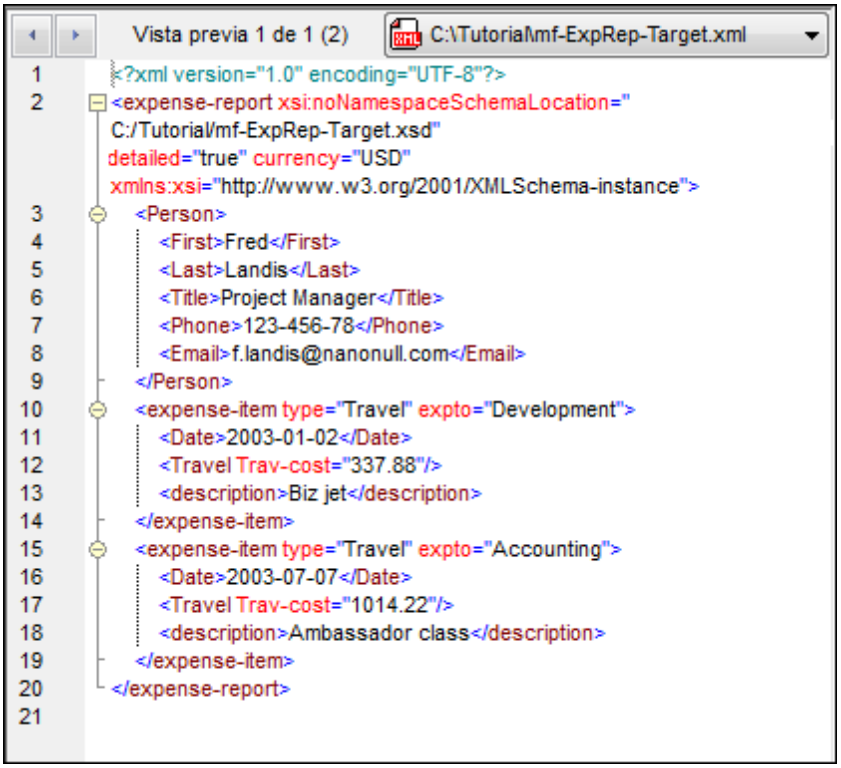
MapForce abre el archivo intermedio y asigna sus datos al componente C. Si existe el

archivo de entrada del componente B, la asignación produce resultados y se puede ejecutar. Si falta el archivo de entrada, MapForce genera un mensaje de error.

Cuando el botón de paso a través está inactivo, se habilita el campo *Archivo XML de entrada* (*imagen anterior*).

Observe la diferencia con el caso en el que el componente B tenía activado el botón de paso a través. En ese caso el campo *Archivo XML de entrada* se deshabilitó automáticamente.

Vista previa:
Muestra el resultado de asignar los datos del componente intermedio B al componente de destino C (es decir, los gastos de viaje por un importe inferior a 1500).



Nota:
Si esta asignación se ejecuta desde la línea de comandos (o se ejecuta el código generado), MapForce intenta generar el resultado de los componentes B y C, independientemente de qué botón de vista previa esté activo y del estado del botón de paso a través. El estado del botón de vista previa no tiene efecto alguno en este caso.

Puesto que el archivo XML de entrada no es el mismo que el archivo XML de salida (porque este campo está vacío), la cadena de asignación está rota y no se pueden generar resultados para el componente C. Para poder generar código, los campos *Archivo XML de entrada* y *Archivo XML de salida* deben ser idénticos.

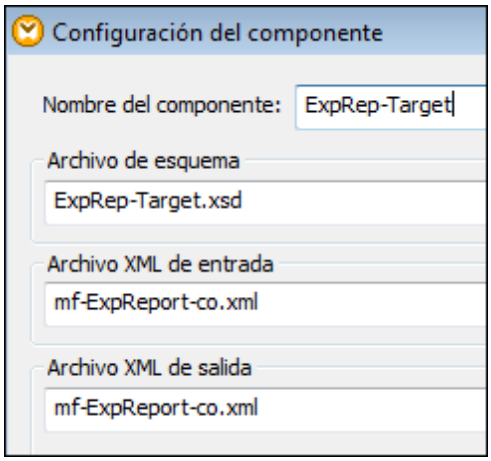
Guardar el resultado intermedio de una asignación

Para poder acceder al archivo de entrada del componente intermedio cuando está inactivo el botón de paso a través, es necesario guardar en un archivo el resultado de asignar los datos del

componente A al componente B. Este nombre de archivo se pone luego en el campo *Archivo XML de entrada* del componente B. Sólo en ese caso se pueden presentar los datos en el componente C final.

Para guardar en un archivo los resultados de asignar datos al componente B:

1. Pulse el botón de vista previa del componente B y después pase al panel **Resultados**.
2. Haga clic en el icono **Guardar resultado generado** de la barra de herramientas. Guarde el archivo XML con un nombre, p. ej. `mf-ExpReport-co.xml`.
3. Vuelva al panel **Asignación**. Haga doble clic en la barra de título del componente B para abrir el cuadro de diálogo "Configuración del componente". Copie el nombre del archivo en el campo *Archivo XML de entrada* y haga clic en **Aceptar**.

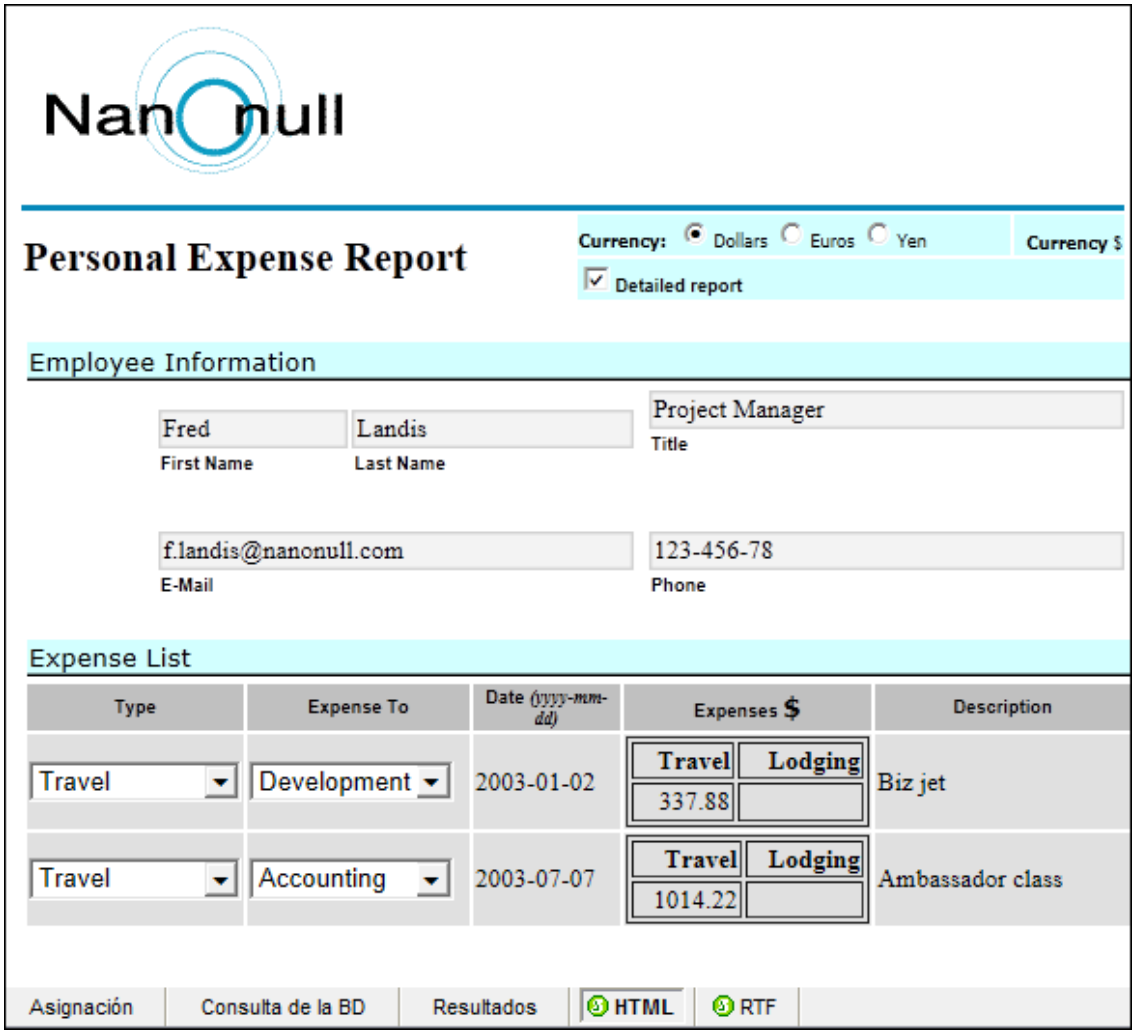


Nota: los archivos XML de entrada y salida **deben** tener el mismo nombre y ambos **deben estar presentes** si desea generar código y ejecutar la asignación desde la línea de comandos.

Visualizar los resultados con StyleVision

Si asoció un archivo SPS al componente de destino, puede ver los datos resultantes en los paneles **HTML/RTF/PDF/Word** de StyleVision de MapForce.

 © 2014 Altova GmbH

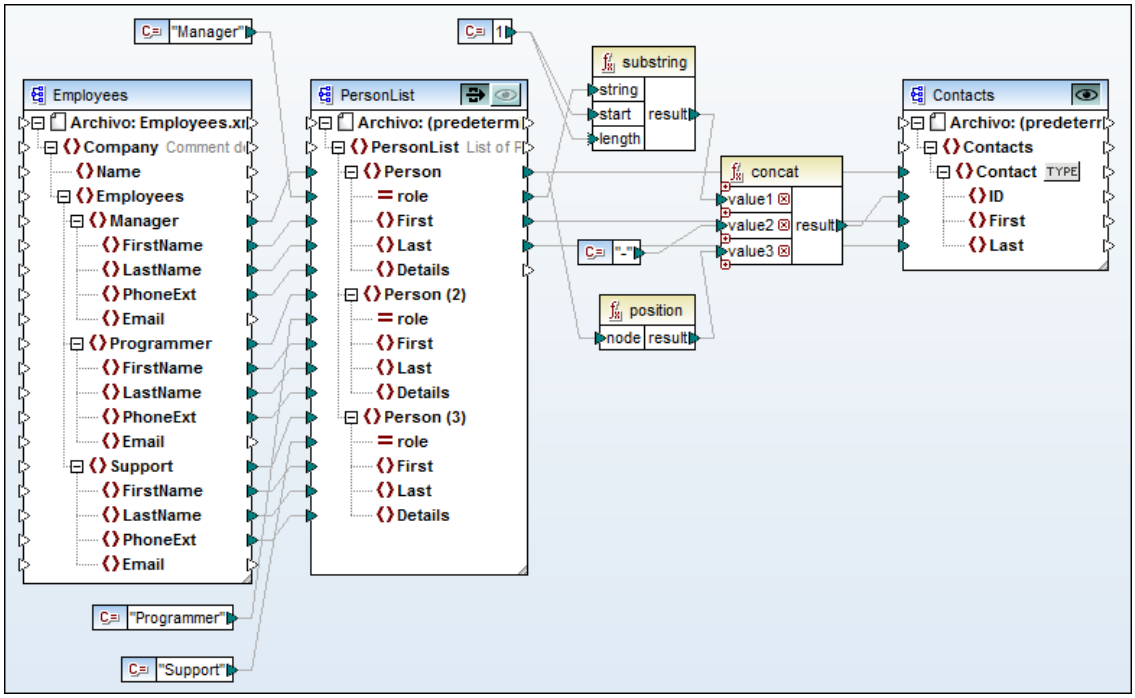

NanoNull
Personal Expense Report
Currency: Dollars Euros Yen
Currency $
Detailed report
Employee Information
Fred
First Name
Landis
Last Name
Project Manager
Title
f.landis@nanonull.com
E-Mail
123-456-78
Phone
Expense List
Type
Expense To
Date (yyyy-mm-dd)
Expenses $
Description
Travel
Development
2003-01-02
Travel
Lodging
337.88
Biz jet
Travel
Accounting
2003-07-07
Travel
Lodging
1014.22
Ambassador class
Asignación
Consulta de la BD
Resultados
HTML
RTF

9.5.3 Ejemplo de asignación en cadena

El ejemplo que describimos a continuación está disponible en el archivo `ChainedPersonList.mfd`, que está en la carpeta `...\MapForceExamples`.

Objetivo:
Crear dos documentos de empleados, uno para el departamento de recursos humanos y otro para el de contabilidad.

- en el documento destinado al departamento de contabilidad, cada empleado tiene un identificador único.
- en el documento destinado al departamento de recursos humanos aparecen los datos personales de cada empleado y su extensión telefónica.



Componentes:

(1) Employees es el componente de origen
El archivo de instancia `Employees.xml` contiene datos de cuatro personas y sus funciones en este orden: manager, programador y soporte.

(2) PersonList (cuyo resultado será el documento para recursos humanos) es el componente intermedio

- A los datos de los empleados se añade el atributo **`role`** y se elimina la posición en la jerarquía que existía en el componente **Employees**.
- Se activa el botón de paso a través.

(3) Contacts: (cuyo resultado será el documento para contabilidad) es el componente de destino/final

- A los datos de contacto (elemento **`Contact`**) se añade el elemento **`ID`** para garantizar que cada persona tiene un identificador único.

 © 2014 Altova GmbH

Cómo funciona está asignación:

PersonList:

- El elemento **`person`** se duplicó **dos veces**, para incluir los tres tipos de funciones (roles) que existen en la compañía.
- El nombre del atributo **`role`** se añade como cadena de texto usando **constantes** (en el mismo orden en el que aparecían en el componente **Employees**).

Contacts:

- La función **`substring`** separa el primer carácter del atributo **`role`** y lo envía a la función **`concat`**.
- La función **`position`** recorre todos los nodos **`Person`**, les asigna un número secuencial (empezando por 1) y lo envía a la función **`concat`**.
- La función **`concat`** combina el carácter de la función **`substring`** con un guión (del componente de constante) y el número de la función **`position`**. Después envía este valor al elemento **`ID`** del componente **Contacts**.

Resultados:

Resultado del componente **PersonList** (para recursos humanos)

Vista previa 2 de 1 (2) PersonList: C:\MapForceExamples\PersonList.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <PersonList xsi:noNamespaceSchemaLocation="C:/MapForceExamples/PersonList.x
   http://www.w3.org/2001/XMLSchema-instance">
3    <Person role="Manager">
4      <First>Vernon</First>
5      <Last>Callaby</Last>
6      <Details>582</Details>
7    </Person>
8    <Person role="Programmer">
9      <First>Frank</First>
10     <Last>Further</Last>
11     <Details>471</Details>
12   </Person>
13   <Person role="Support">
14     <First>Loby</First>
15     <Last>Matise</Last>
16     <Details>963</Details>
17   </Person>
18   <Person role="Support">
19     <First>Susi</First>
20     <Last>Sanna</Last>
21     <Details>753</Details>
22   </Person>
23 </PersonList>
24
```

Resultado del componente **Contacts** (para contabilidad)

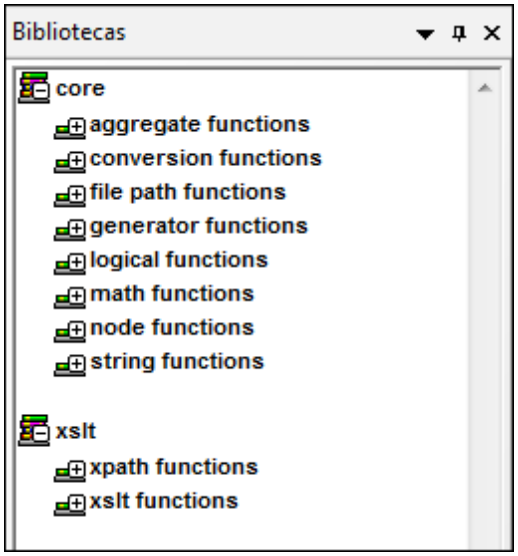
Vista previa 1 de 1 (2) C:\MapForceExamples\Contacts.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Contacts xsi:noNamespaceSchemaLocation="C:/MapForceExamples/
   http://www.w3.org/2001/XMLSchema-instance">
3    <Contact>
4      <ID>M-1</ID>
5      <First>Vernon</First>
6      <Last>Callaby</Last>
7    </Contact>
8    <Contact>
9      <ID>P-2</ID>
10     <First>Frank</First>
11     <Last>Further</Last>
12   </Contact>
13   <Contact>
14     <ID>S-3</ID>
15     <First>Loby</First>
16     <Last>Matise</Last>
17   </Contact>
18   <Contact>
19     <ID>S-4</ID>
20     <First>Susi</First>
21     <Last>Sanna</Last>
22   </Contact>
23 </Contacts>
24
```

 © 2014 Altova GmbH

9.6 Uso de las funciones

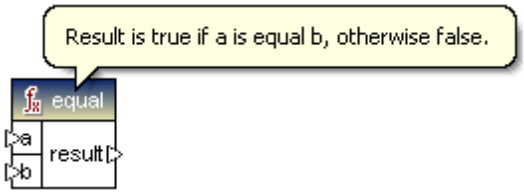
En la ventana Bibliotecas aparecen todas las funciones disponibles para el lenguaje seleccionado (XSLT/XSLT2/BUILTIN/XQ/Java/C#/C++). Para ver/ocultar las funciones use los iconos + y - de cada biblioteca.



Con XSLT seleccionado

Información sobre las funciones

Si hace clic en el icono **Mostrar información sobre herramientas** puede activar/desactivar la descripción de las funciones. Si activa este icono, la descripción aparece al pasar el puntero del mouse sobre el título de la función.



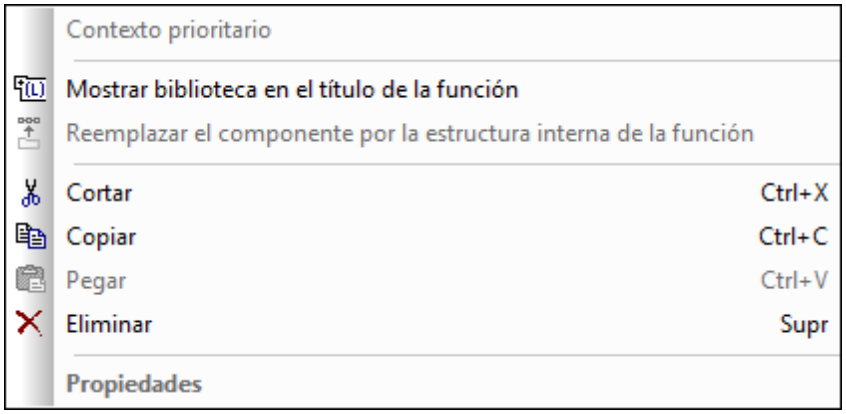
Para usar una función en la ventana de la asignación:

1. Seleccione el **lenguaje de programación** para el que tiene pensado generar código (haciendo clic en el icono correspondiente de la barra de herramientas).
2. Haga clic en el **nombre de la función** en la ventana Bibliotecas y arrástrela hasta el panel **Asignación**.
3. Conecte los parámetros de entrada y salida de la función con los nodos pertinentes.

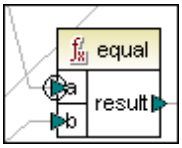
Recuerde que si pasa el puntero del mouse sobre la expresión "`result=....`" de la ventana Bibliotecas, aparece un mensaje emergente que describe la función con más detalle.

Menú contextual de las funciones

Al hacer clic con el botón secundario en una función situada en el panel **Asignación**, se abre este menú contextual:



Contexto prioritario Cuando se aplica una función a varios elementos de un esquema o BD, MapForce necesita conocer cuál será el nodo de contexto. Los demás elementos se procesarán en relación a dicho nodo. Esto se consigue designando el elemento (o nodo) como contexto prioritario. El nodo designado como prioritario aparece dentro de un círculo. Para más información, consulte el apartado Contexto prioritario de la *Referencia del usuario*.



Mostrar biblioteca en el título de la función En el nombre del componente aparece el nombre de la biblioteca a la que pertenece la función.

Reemplazar el componente por la estructura interna de la función Reemplaza el componente definido por el usuario por las partes que lo constituyen. No disponible para las funciones.

Cortar/Copiar/Pegar/Eliminar Comandos de edición estándar de MS Windows que permiten cortar, copiar, etc. cualquier componente o función visible en el panel **Asignación**. Se conservarán todos los conectores excepto los que se deban reemplazar.

Propiedades Disponible solamente para componentes definidos por el usuario.

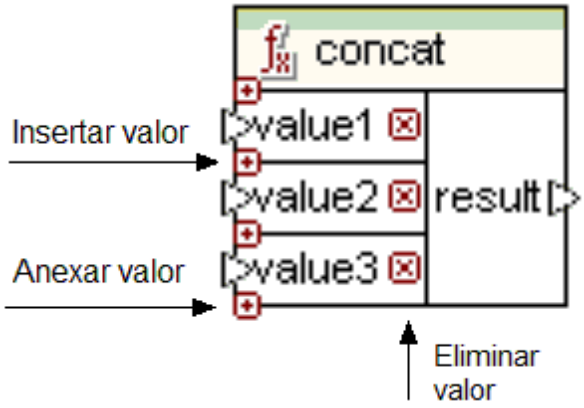
Funciones ampliables

Hay varias funciones que se pueden ampliar: p. ej. las funciones **`concat`**, **`logical-and`**, **`logical-or`** e **`IF-ELSE`**. Esto significa que en este tipo de funciones puede insertar/anexar/eliminar parámetros dependiendo de sus requisitos.

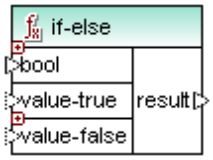
Al hacer clic en el icono **+** se inserta/anexa el mismo tipo de parámetro. Asimismo, al hacer clic en la marca **x** se elimina el parámetro.

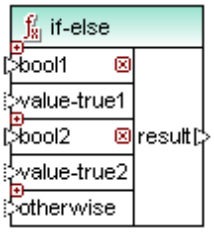
Nota: al soltar un conector en el icono **+** de la función, se inserta/anexa otro parámetro y la conexión se crea automáticamente.

 © 2014 Altova GmbH



Los parámetros de prueba IF de la función **IF-Else** también se pueden ampliar de esta forma.





9.7 Bucles, grupos y jerarquías

En MapForce hay varias formas de recorrer las jerarquías de origen y destino. Esto significa que puede definir cómo se recorren los datos mediante bucles y cómo se agrupan los conjuntos de datos.

Aquí puede ver los apartados que describen estos métodos. Tenga en cuenta que estos ejemplos no aparecen uno detrás del otro, sino que se encuentran en apartados diferentes de la documentación.

Asignar datos XML a archivos CSV o archivos de texto de longitud fija

Asignar archivos CSV a XML

Crear jerarquías a partir de archivos CSV y de texto de longitud fija

Asignación de valores (tabla de consulta)

Asignar varias tablas a un solo archivo XML

Usar MapForce para crear relaciones de BD

Contexto prioritario

 © 2014 Altova GmbH

9.8 Reglas y estrategias de asignación de datos

Por lo general, MapForce asigna datos de forma intuitiva, pero en algunos casos puede haber demasiados elementos en los resultados de la asignación o puede que falten elementos. Este apartado se ocupa de estos y otros casos.

Regla general:
Por regla general, cuando hay una conexión entre un elemento de origen y otro de destino se crea un elemento de destino por cada elemento de origen.

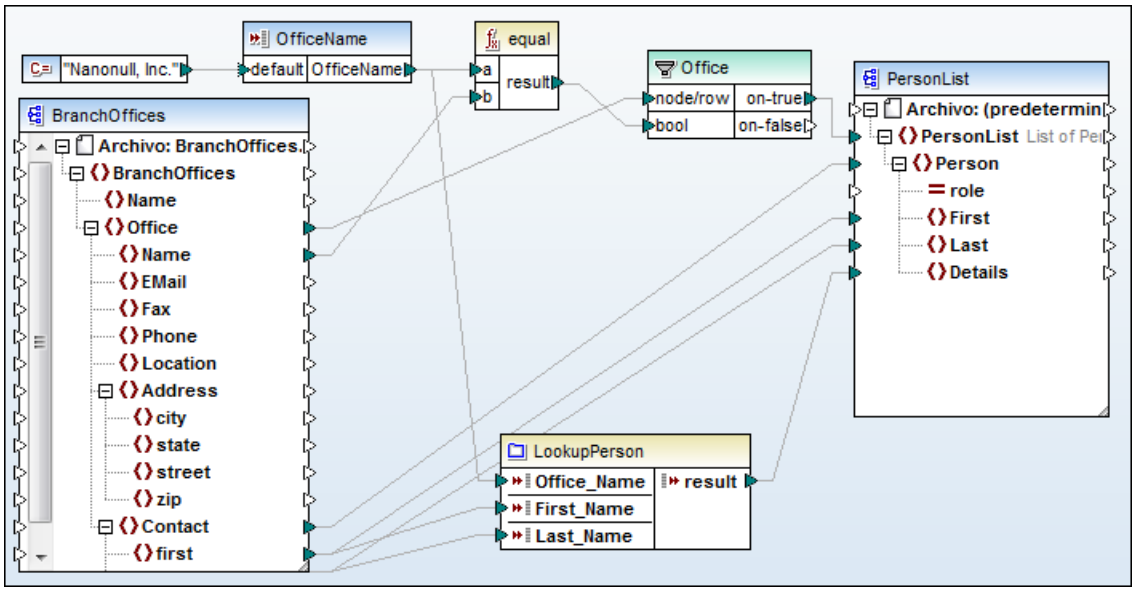
Si el elemento de origen tiene contenido simple (p. ej. una cadena de texto, un entero, etc.) y el de destino acepta contenido simple, entonces se copia el contenido y se convierte el tipo de datos, si es necesario.

Sin embargo, hay algunas excepciones.

Elementos *actuales* y elementos de *contexto*
MapForce muestra la estructura de un esquema, BD o archivo EDI como una jerarquía de elementos asignables dentro del componente. Por cada nodo de la jerarquía (del esquema) puede haber varias instancias del nodo (o ninguna) en el archivo XML de instancia o en la BD.

Por ejemplo, en la asignación `PersonListByBranchOffice.mfd` observe que en el componente de origen (**BranchOffices**) el nodo **`first`** aparece solo una vez (bajo el nodo **`Contact`**). Y, sin embargo, en el archivo de instancia `BranchOffices.xml` hay varios elementos **`first`** y varios elementos **`Contact`** con contenido diferente y situados bajo diferentes elementos primarios **`Office`**.

Dependiendo del **contexto** actual (del nodo de **destino**), se selecciona un elemento de origen u otro a la hora de copiar los datos en el elemento de destino (por medio de un conector).



Este contexto viene definido por el nodo de destino actual y sus conexiones con nodos

antecesores:

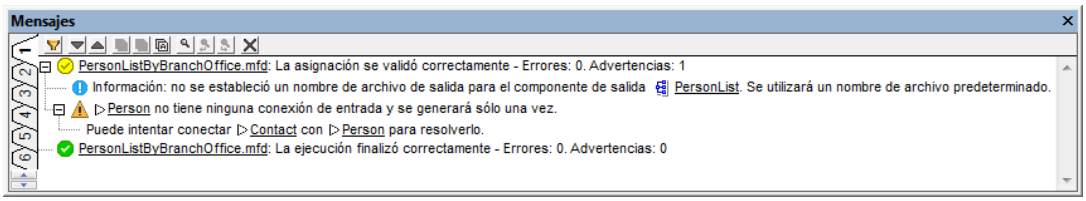
- En un principio, el contexto solamente contiene los componentes de origen, pero no contiene ningún nodo concreto. Al evaluar la asignación, MapForce procesa primero el nodo **raíz de destino** (**`PersonList`**) y después sigue con el resto de la jerarquía.
- El conector del nodo de **destino** llega hasta todos los elementos de origen conectados a él directa o indirectamente, incluso a través de las funciones existentes entre los dos componentes. 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, en un principio, 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.

Si esto mismo se aplica al caso concreto de la asignación anterior (`PersonListByBranchOffice.mfd`):

- La conexión entre el nodo de origen **`Office`** a través del filtro (`Office`) y el nodo de destino **`PersonList`** define una oficina determinada como contexto de todo el documento de destino (porque **`PersonList`** es el elemento raíz del componente de destino). El nombre de la oficina lo suministra el componente de entrada, que tiene el valor predeterminado `Nanonull, Inc`.
- Todas las conexiones/los datos que llegan a los **descendientes** del elemento raíz **`PersonList`** se ven afectados automáticamente por la condición de filtro, porque la oficina seleccionada está en el contexto.

- La conexión entre el nodo de origen **`Contact`** y el nodo de destino **`Person`** da como resultado un elemento `Person` por cada elemento `Contact` del XML de origen (regla general). Por cada elemento `Person` se añade un elemento `Contact` concreto al contexto, a partir del cual se crean los secundarios de `Person`.
- La conexión entre el nodo **`first`** y el nodo de destino **`First`** selecciona el primer elemento `name` del elemento `Contact` actual y lo escribe en el elemento `First` de destino.

  Si suprimimos la conexión entre el nodo de origen **`Contact`** y el de destino **`Person`**, entonces se crea un solo elemento `Person` con varios secundarios `First`, `Last` y `Details`. En casos así MapForce emite un mensaje de advertencia (*imagen siguiente*) y sugiere cómo solucionar el problema, p. ej. *Puede intentar conectar* `Contact` *con* `Person` *para resolverlo.*



Secuencias
MapForce muestra la estructura de un esquema, BD o archivo EDI como una jerarquía de elementos asignables dentro del componente.

Dependiendo del contexto (de destino), cada **elemento asignable** de un componente de **origen** puede representar:

- un solo nodo de instancia del archivo (o BD) de entrada asociado o
- una **secuencia** de entre 0 y **varios nodos de instancia** del archivo (o BD) de entrada asociado

Si se conecta una secuencia al nodo de destino, se crea un bucle para crear tantos nodos de destino como nodos de origen haya.

Si se crea un filtro entre la secuencia y el nodo de destino, la condición bool se comprueba en cada nodo de entrada, es decir, en cada elemento de la secuencia. Concretamente se comprueba si en cada secuencia existe como mínimo un bool cuyo resultado sea true. El contexto prioritario puede influir en el orden de evaluación (*ver más abajo*).

Las condiciones de filtro afectan automáticamente a todos los nodos descendientes.

Nota: si el esquema de origen especifica que un nodo determinado debe aparecer exactamente una vez, MapForce puede quitar el bucle y tomar solamente el primer elemento, porque sabe que este elemento debe existir. Esta optimización se puede deshabilitar en el cuadro de diálogo "Configuración del componente" (desactivando la casilla *Optimización de procesamiento de datos de entrada basada en minOccurs/maxOccurs*).

Los **parámetros de entrada** de las **funciones** (de las funciones normales que no son funciones de secuencia) funcionan igual que los nodos de destino: si una secuencia está conectada a una función de este tipo, se crea un bucle alrededor de la llamada a función, que produce tantos resultados como elementos haya en la secuencia.

Si una secuencia está conectada a **varias funciones** de este tipo, MapForce crea bucles anidados que procesan el **producto cartesiano** de todos los datos de entrada. Esto no es ideal, así que a la función (y a los demás parámetros enlazados a elementos actuales de primarios o a otros parámetros) deberíamos conectar una sola secuencia formada por varios elementos.

Nota: si conectamos una secuencia vacía a este tipo de función (p. ej. a una función **`concat`**), el resultado obtenido será una **secuencia vacía**, que no producirá nodos de salida. Si en el archivo de salida no se crean resultados porque no hay datos de entrada, puede usar la función **`substitute-missing`** para insertar un valor de sustitución.

Las funciones que tiene **secuencias como parámetros de entrada** son las únicas funciones que pueden producir resultados si la secuencia de entrada está **vacía**:

- las funciones **`exists`**, **`not-exists`** y **`substitute-missing`** (así como sus alias **`is-not-null`**, **`is-null`** y **`substitute-null`**)
- las funciones de **agregado** (**`sum`**, **`count`**, etc.)
- las funciones definidas por el usuario **corrientes** que aceptan secuencias (es decir, funciones no inline)

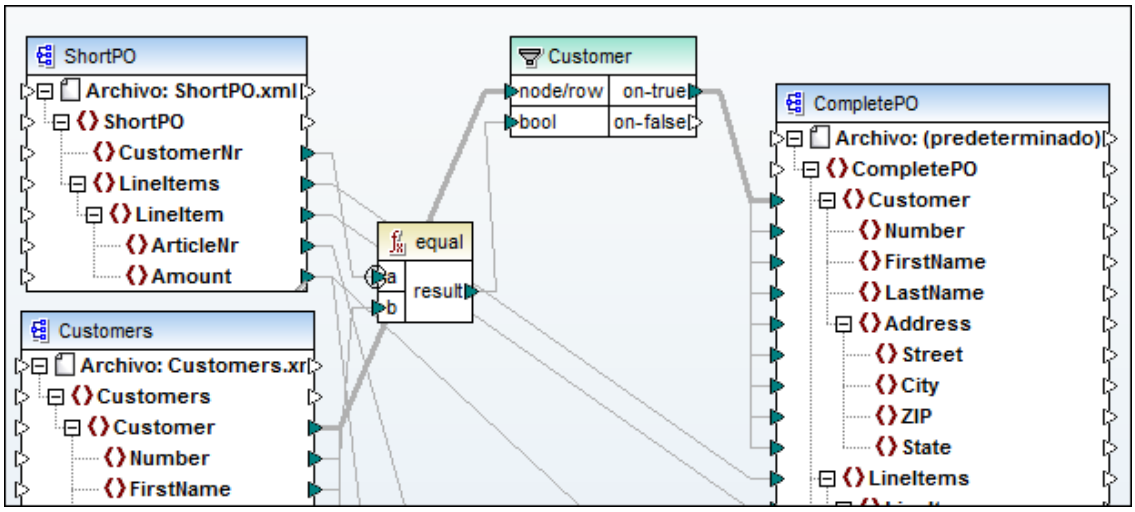
La secuencia de entrada de este tipo de funciones siempre se evalúa independientemente del

nodo de destino actual del contexto de sus antecesores. Estos significa que si hay componentes de filtrado o WHERE de SQL conectados a este tipo de funciones, estos componentes no afectan a las demás conexiones.

Contexto prioritario

Por lo general, los parámetros de las funciones se evalúan de arriba a abajo. No obstante, existe la posibilidad de definir un parámetro como **contexto prioritario**, para que este parámetro se evalúe antes que los demás.

En las funciones conectadas al parámetro de entrada bool de condiciones de **filtro**, el contexto prioritario no afecta solamente a la función de comparación sino también a la evaluación del filtro. Por tanto, se pueden unir dos secuencias de origen (ver las secuencias **`CustomerNo`** y **`Number`** del archivo de asignación `CompletePO.mfd`).



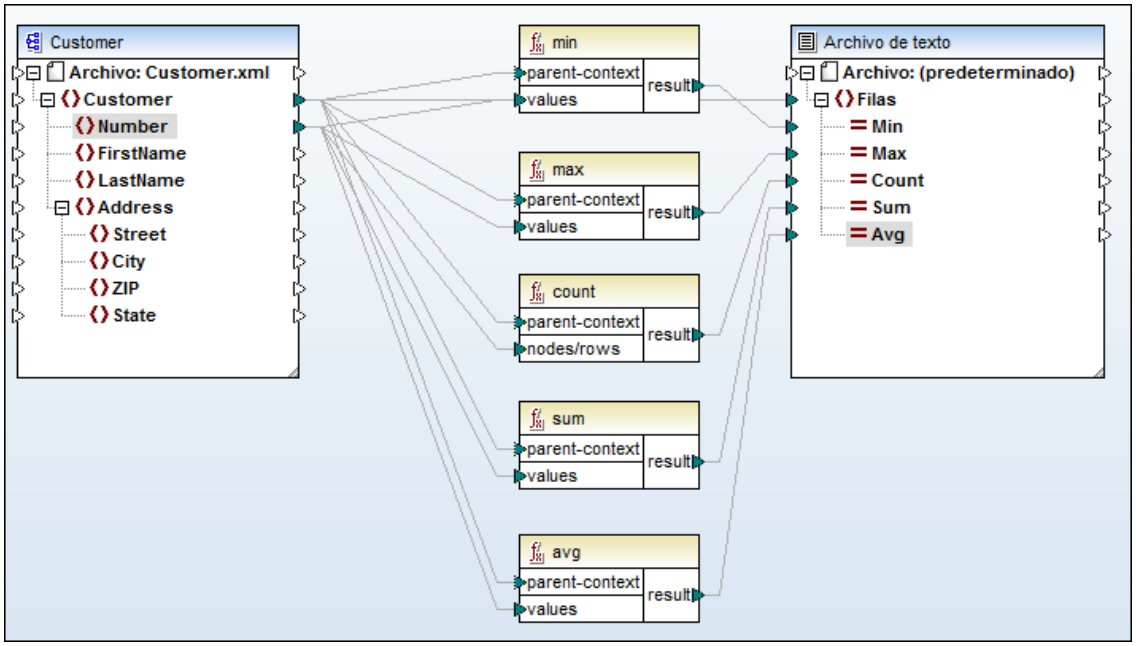
En este ejemplo el contexto prioritario obliga a evaluar **`ShortPO`**/**`CustomerNr`** antes de recorrer y filtrar los nodos de **`Customer`** del componente **Customers**. Para más información, consulte el ejemplo del apartado Nodo/elemento con contexto prioritario.

Invalidar el contexto

Algunas funciones de agregado tienen el parámetro de entrada opcional `parent-context`.

Si este parámetro de entrada no se conecta, no tiene efecto alguno y la función se evalúa en el contexto normal de las secuencias de entrada (es decir, en el contexto del primario del nodo de destino).

 © 2014 Altova GmbH



Si el parámetro de entrada `parent-context` está conectado a un nodo de origen, la función se evalúa **para cada** nodo `parent-context` y produce un resultado separado por cada repetición.

Excepciones de la regla general (por cada elemento de origen se crea un elemento de destino)

- El elemento raíz XML de destino se crea siempre una sola vez. Si está conectado a una secuencia, se repite solamente el contenido del elemento, pero no del elemento raíz propiamente dicho. No obstante, puede que el resultado no sea válido. Si los atributos del elemento raíz también están conectados, la serialización XML dará error en el tiempo de ejecución. Por ello, no aconsejamos conectar una secuencia al elemento raíz. Para crear varios archivos de salida, conecte la secuencia al nodo **`Archivo:`**, a través de alguna función que genere los nombres de archivo.
- Otros tipos de nodos, como los atributos XML, los campos de BD o los componentes de salida situados dentro de una función definida por el usuario, admiten un solo valor.

Poner en el contexto varios nodos del mismo componente de origen
Esto puede ser necesario en algunos casos especiales y se puede hacer usando variables intermedias.

Altova MapForce 2014

Datos de origen y destino

10 Datos de origen y destino

Esta sección describe los tipos de componentes de origen y destino compatibles con las asignaciones de datos de MapForce.

- XML y esquemas XML
- Bases de datos
- Archivos de texto y archivos CSV
- Archivos de MapForce Flextext
- Archivos EDI - UN/EDIFACT, ANSI X12, HL7, IATA PADIS, SAP IDoc
- Archivos MS OOXML Excel 2007+
- Archivos XBRL

© 2014 Altova GmbH

10.1 Datos XML y esquemas XML

Esta sección explica algunos conceptos algo más complicados que la simple asignación de archivos XML y esquemas XML. La asignación básica de datos XML a archivos XML o esquemas XML se explica en detalle en el Tutorial.

A continuación tratamos estos conceptos:

- Usar documentos DTD como componentes de esquema
- Asignación de datos a tipos XML Schema derivados
- Uso de QName
- Valores nulos / que pueden ser nulos
- Comentarios e instrucciones de procesamiento
- Comodines: xs:any

10.1.1 Usar documentos DTD como componentes de esquema

A partir de la versión 2006sp2 MapForce admite el uso de documentos DTD como componentes de origen y destino. Para hacer posible la asignación de datos, los URI de espacio de nombres se extraen de las declaraciones de atributo `xmlns`.

Agregar el URI de espacio de nombres de la DTD
Sin embargo, algunas DTD (p. ej. las DTD utilizadas por StyleVision) contienen declaraciones de atributo `xmlns*` sin URI de espacio de nombres. Estas DTD deben ampliarse antes de poder ser utilizadas en MapForce.

La DTD debe modificarse definiendo el atributo `xmlns` con el URI de espacio de nombres. Por ejemplo:

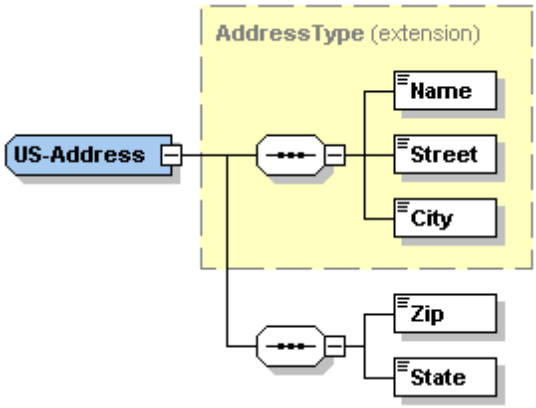
```
<!ATTLIST fo:root
        xmlns:fo CDATA #FIXED 'http://www.w3.org/1999/XSL/Format'
        ...
>
```

 © 2014 Altova GmbH

10.1.2 Asignar tipos derivados de XML Schema

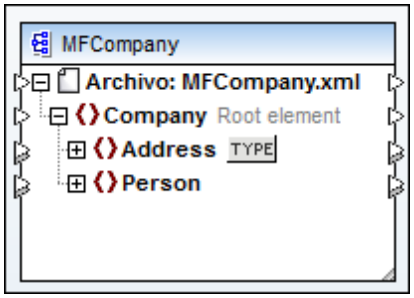
En MapForce puede realizar asignaciones entre tipos derivados de un tipo complejo. Los tipos derivados son tipos complejos de un esquema XML que usan el atributo `xsi:type` para identificar los tipos derivados.

La imagen siguiente muestra la definición del tipo derivado **`US-Address`** en XMLSpy. El tipo base (o el tipo complejo original) es en este caso **`AddressType`**. Para crear el tipo derivado **`US-Address`** se añadieron dos elementos al tipo base original.

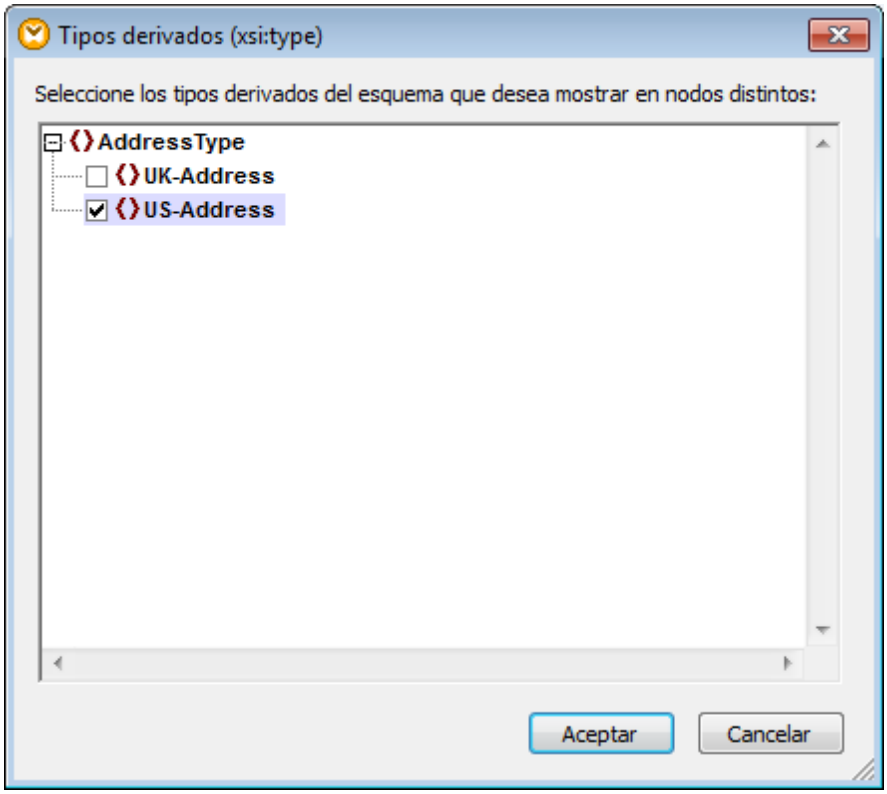


Asignaciones de datos a tipos derivados en MapForce:

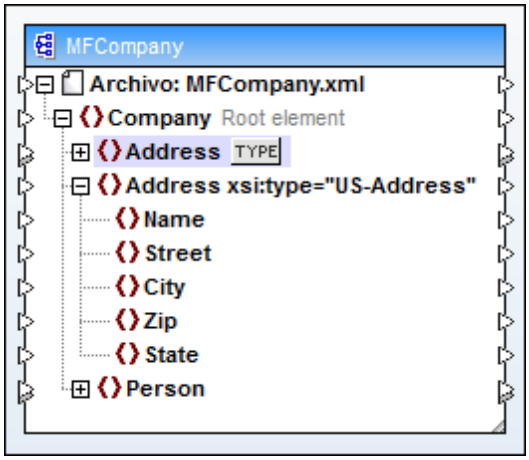
1. Inserte el esquema XML `MFCompany.xsd` (de la carpeta `...\Tutorial`) haga clic en **Omitir** y seleccione el elemento raíz `Company`.



2. Haga clic en el botón **TYPE** situado a la derecha del elemento **`Address`** y aparece un cuadro de diálogo con todos los tipos derivados que existen en el componente de esquema.



3. Marque la casilla del tipo derivado que desea usar, p. ej. **`US-Address`** y confirme la selección con **Aceptar**.
   En el componente se añade el elemento nuevo **`Address xsi:type="US-Address"`**.
4. Haga clic en el botón de expansión para ver todos los elementos asignables del elemento.



5. Ahora puede asignar los datos de este elemento al componente de destino.

Nota: puede incluir/insertar varios tipos derivados seleccionándolos en el cuadro de diálogo "Tipos derivados" y cada uno lleva su propio elemento `xsi:type`.

 © 2014 Altova GmbH

10.1.3 Uso de nombres QName

Los QName (nombres completos) permiten usar los URI de espacio de nombres utilizados en archivos XML y XBRL y hacerles referencia. Hay dos tipos de nombres QName, con prefijo y sin prefijo.

NombreConPrefijo	Prefijo **":"**	ParteLocal
NombreSinPrefijo		ParteLocal siendo ParteLocal un nombre de elemento o atributo.

```
<Doc xmlns:x="http://myCompany.com">

    <x:part>

</Doc>
```

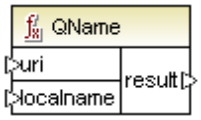
x es la referencia de espacio de nombres a **`"http://myCompany.com"`** y `<x:part>` es un QName válido porque:

x es el prefijo de espacio de nombres
part es la parteLocal, es decir, el nombre del elemento.

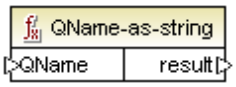
En las asignaciones entre componentes de origen y destino los prefijos de QName se resuelven.

MapForce ofrece varias funciones para trabajar con nombres QName (en la biblioteca de funciones **Lang**):

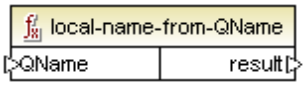
QName
Construye un QName a partir de un URI de espacio de nombres y una parte local. Utilice esta función para crear un QName en un componente de destino. Los parámetros **`uri`** y **`localname`** pueden venir dados por una constante.



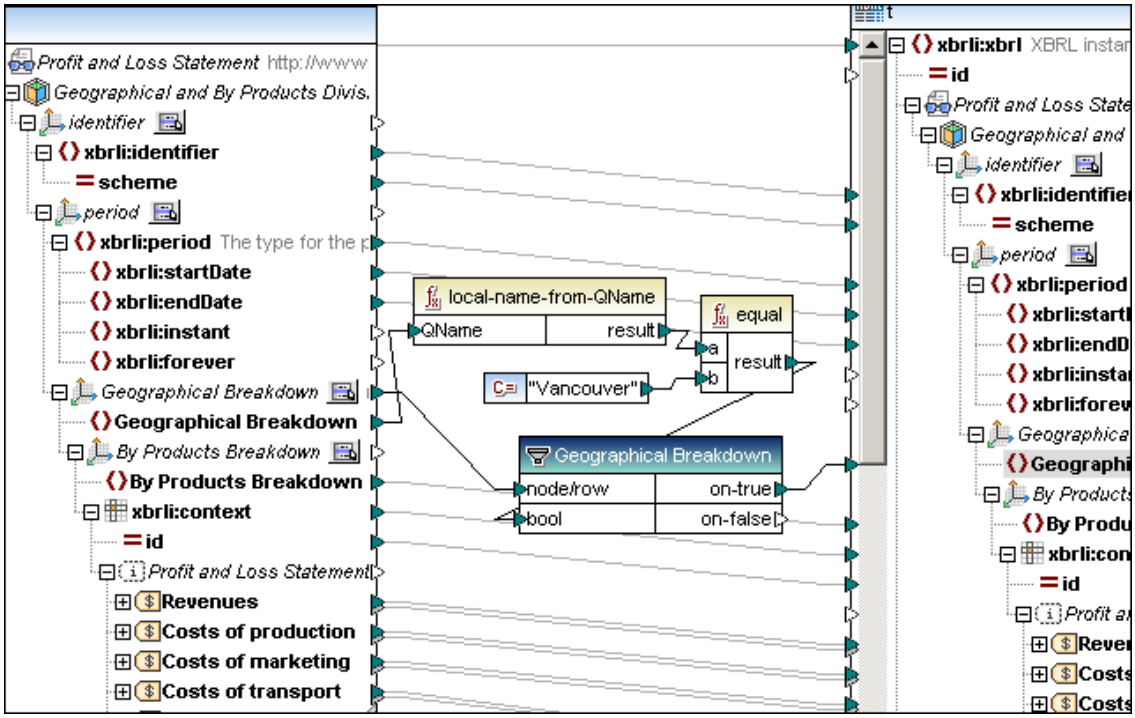
QName-as-string
Convierte un QName en una cadena de formato `{http://myCompany.com}local`.



local-name-from-QName
Extrae el nombre local de un QName.



Esta función es muy práctica a la hora de crear asignaciones entre documentos XBRL con hipercubos.



Por ejemplo, la asignación anterior filtra los hechos en los que el **nombre local** del **contenido** del miembro explícito (**`d-g:Vancouver`**) es igual a "Vancouver". Observe que el contenido del miembro es en sí mismo un QName.

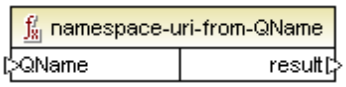
```
<xbrli:context id="Year2007_Vancouver_BeveragesTotal">
  <xbrli:entity>
    <xbrli:identifier scheme="http://www.stockexchange/ticker">ACME</xbrli:identifier>
    <xbrli:segment>
      <xbrldi:explicitMember dimension="d-g:GeographicalBreakdown">d-g:Vancouver</xbrl
      <xbrldi:explicitMember dimension="d-b:ByProductsBreakdown">d-b:BeveragesTotal</x
    </xbrli:segment>
```

Todos los hechos que pertenecen a la dimensión **`GeographicalBreakdown`** se filtran y se pasan 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>
```

 © 2014 Altova GmbH

namespace-uri-from-QName
Extrae el URI de espacio de nombres de un QName.



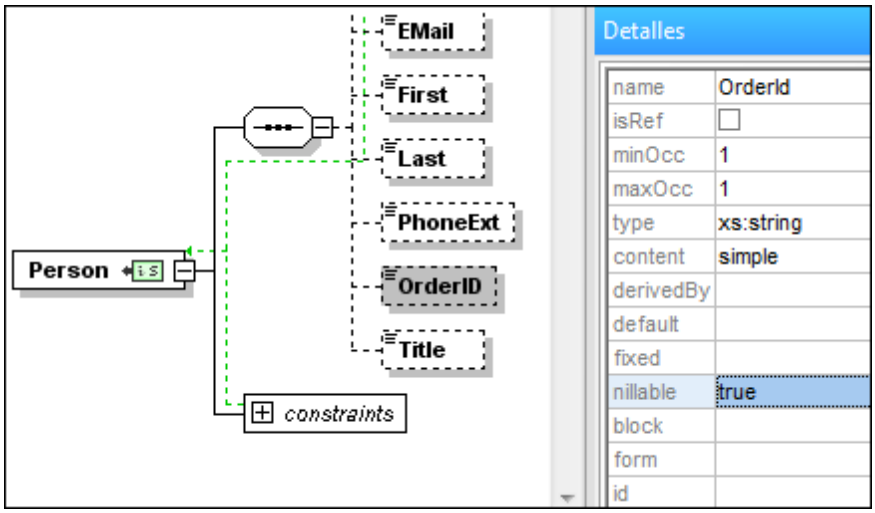
10.1.4 Valores nulos / que pueden ser nulos

Según la especificación XML Schema, un elemento es válido a pesar de no tener contenido (a pesar de tener un tipo de contenido que no permite contenido vacío) si en el esquema se definió el atributo `nillable="true"` para dicho elemento.

El **atributo** `xsi:nil="true"` del documento XML de **instancia** se utiliza para indicar que el elemento existe pero no tiene contenido.

Tenga en cuenta que esto afecta solamente a los valores de los elementos, no a los atributos. El elemento que tenga el atributo `xsi:nil="true"` puede carecer de contenido textual o de elementos secundarios, pero puede tener más atributos.

El atributo `nillable="true"` del **esquema XML** puede estar presente tanto en el componente de origen como en el de destino.



El atributo `xsi:nil="true"` del **archivo XML de instancia** solamente está presente en el archivo de instancia.

```
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` no se representa de forma explícita en la asignación gráfica de MapForce porque en la mayoría de los casos se procesa automáticamente: en MapForce existe un nodo llamado **`nilled`** (que tiene el atributo `xsi:nil="true"`), pero su contenido no existe.

 © 2014 Altova GmbH

Usar como origen de la asignación de datos elementos que pueden ser nulos
Cuando una asignación lee datos **desde** un elemento XML o XBRL que puede ser nulo, MapForce comprueba automáticamente qué valor tiene el atributo `xsi:nil`. Si su valor es `true`, su contenido se trata como si no existiera.

Cuando se crea una asignación **basada en el destino** entre un elemento de origen nulo y otro de destino con contenido simple (un solo valor con atributos opcionales, pero sin elementos secundarios), el atributo `xsi:nil` del elemento de origen se inserta en el elemento de destino (p. ej. <OrderID xsi:nil=" true"/>.

Cuando se crea una asignación de copia total entre un elemento de origen nulo y otro de destino, el atributo `xsi:nil` del elemento de origen se inserta en el elemento de destino (p. ej. <OrderID xsi:nil="true"/>.)

Funciones y elementos nulos
Si conecta la función **`exists`** a un elemento de origen nulo, la función devuelve `true` para todos los elementos porque el nodo de **elemento** existe aunque no tenga contenido.

Para comprobar expresamente si el elemento de origen tiene `xsi:nil` con valor `true`, use la función **`is-xsi-nil`**. Esta función devuelve `true` si el elemento tiene `xsi:nil=true`.

Las funciones que esperan valores simples (como **`multiply`**, **`concat`**, etc.) y se usan en elementos con `xsi:nil=true` no devuelven resultados porque no existe contenido de elemento y no se puede extraer ningún valor. En este caso, las funciones se comportan como si el nodo de origen no existiera.

Para sustituir el valor de un elemento de origen nulo (que no existe) por algo en concreto, use la función **`substitute-missing`**.

Las funciones que generan `xsi:nil` no se pueden pasar mediante funciones ni componentes que solo operan en valores (p. ej. la función **`if-else`**).

Asignar elementos `xsi:nil` a un campo de BD
Cuando se asigna un elemento XML nulo a una columna de BD; se escribe un valor NULL. También puede usar la función **`set-null`** si quiere dar el valor NULL al campo de la BD.

Usar como destino de la asignación elementos que pueden ser nulos
Cuando se crea una asignación basada en el destino entre un elemento de origen nulo y un elemento de destino que puede ser nulo de **contenido simple** (un solo valor con atributos opcionales, pero sin elementos secundarios), el atributo `xsi:nil` se inserta en el elemento de destino <OrderID xsi:nil="true"/>.

Si el elemento XML de **origen no tiene** un atributo `xsi:nil` con valor `true`, entonces el contenido del elemento se asigna al elemento de destino de forma normal.

Cuando se crea una asignación entre un elemento de origen nulo y un elemento de destino que puede ser nulo de **contenido complejo** (con elementos secundarios), el atributo `xsi:nil` no se escribe automáticamente en el elemento de destino porque MapForce no puede saber en ese momento si le seguirán elementos secundarios o no. Para copiar el atributo `xsi:nil` en el

elemento de destino, defina una **conexión de copia total**.

Cuando se crea una asignación entre una **secuencia vacía** o un valor `NULL` de base de datos y un elemento de destino, el elemento no se crea en el destino, independientemente de su atributo `nil`.

Funciones y elementos nulos

Para forzar la creación de un elemento de destino vacío con `xsi:nil="true"`, conecte la función **`set-xsi-nil`** al elemento de destino directamente, tanto si el elemento de destino es de tipo simple como si es de tipo complejo.

Use la función **`substitute-missing-with-xsi-nil`** para insertar el atributo `xsi:nil` en el destino si no hay ningún valor disponible en el origen de la asignación. Esto puede pasar si el nodo de origen no existe o si existe un cálculo (p. ej. **`multiply`**) relacionado con un nodo de origen nulo y que, por tanto, no produce ningún resultado.

Esta función también se puede usar para crear elementos con `xsi:nil="true"` a partir de valores `NULL` de BD. Esta función solamente funciona con nodos de contenido simple.

Para crear `xsi:nil` en un elemento de **tipo complejo**:

1. Cree la asignación al nodo de destino que tiene contenido.
2. Inserte un componente de filtrado en la conexión con el nodo de destino y defina la condición en caso de que no haya contenido.
3. Añada un duplicado de entrada del nodo de destino.
4. Conecte la función **`set-xsi-nil`** al duplicado del nodo de destino a través de otro componente de filtrado.
5. Inserte la función **`logical-not`** para negar la condición y conéctela al filtro situado en la conexión entre **`set-xsi-nil`** y el duplicado del nodo de destino.

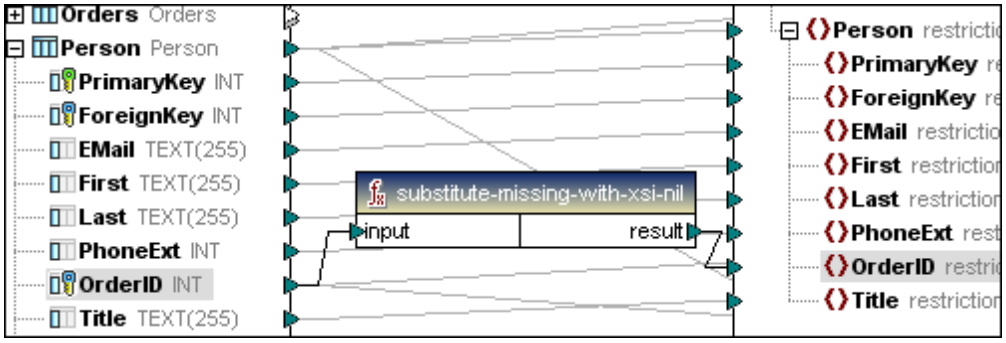
Asignar campos `NULL` de componentes de base de datos

Si asigna un campo de BD con valor `NULL` a un elemento que puede ser nulo de un esquema XML, solamente se generan los elementos de destino que contienen datos de BD. Los elementos de campos de BD con valor `NULL` no se crean en el componente de destino.

Si conecta la función **`exists`** a este tipo de elementos de origen, la función devuelve el valor `false` para los campos **`NULL`**.

Para forzar la creación de todos los elementos en el componente de destino:

Use la función **`substitute-missing-with-xsi-nil`** del grupo de funciones **node** de la biblioteca **core**.



Ahora se crean elementos de destino para todos los campos de la BD.

- Todos los campos de la BD con valor NULL (o inexistentes) ahora contienen <OrderID xsi:nil="true"/> en el elemento de destino.
- Los datos de los campos de la BD se asignan directamente al elemento de destino (p. ej. <OrderID>1</OrderID>.)

Para buscar/visualizar los campos NULL de un componente de base de datos abra el panel **Consulta de la BD** y ejecute una consulta en la tabla de la BD. Los campos NULL aparecen como `[NULL]` en el panel de resultados.

	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

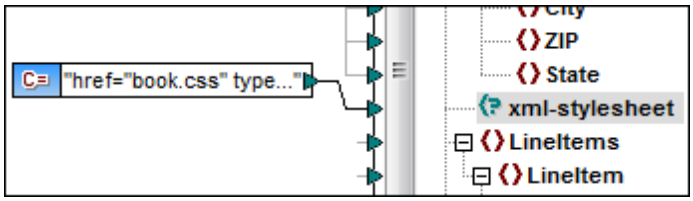
10.1.5 Comentarios e instrucciones de procesamiento

En los componentes XML de destino puede insertar comentarios e instrucciones de procesamiento. Estas últimas sirven para pasar información a las aplicaciones que continúan procesando los documentos XML

Nota: no se pueden definir comentarios ni instrucciones de procesamiento para nodos que forman parte de un grupo con asignaciones de copia total.

Para insertar una instrucción de procesamiento:

1. Haga clic con el botón secundario en un elemento del componente de destino y seleccione **Comentario/instrucción de procesamiento**. Después elija una de las opciones disponibles (**Agregar delante, agregar detrás**, etc.) en el menú.
2. Introduzca el nombre (de destino) de la instrucción de procesamiento en el cuadro de diálogo (p. ej. `xml-stylesheet`) y pulse **Aceptar** para confirmar.
   Esto añade un nodo con este nombre al componente.

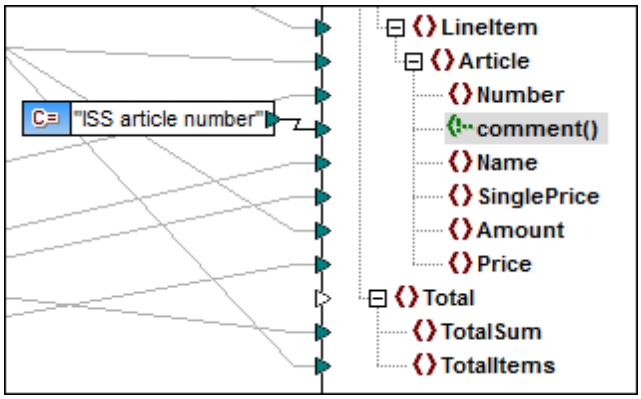


3. Ahora puede usar un componente de constante para suministrar el valor del atributo de la instrucción de procesamiento (p. ej. `href="book.css" Type="text/css"`).

Nota: puede añadir más de una instrucción de procesamiento delante o detrás de cualquier elemento del componente de destino.

Para insertar un comentario:

1. Haga clic con el botón secundario en un elemento del componente de destino y seleccione **Comentario/instrucción de procesamiento**. Después elija una de las opciones disponibles (**Agregar delante, agregar detrás**, etc.) en el menú.



   Esto añade el nodo de comentario (`<!--comment()`) al componente.
2. Ahora puede usar un componente de constante para suministrar el texto del comentario o conectar un nodo de origen al nodo de comentario.

Nota: no se puede añadir más de un comentario antes o después de un nodo de destino.

 © 2014 Altova GmbH

Para crear varios comentarios, utilice la función *duplicado de entrada*.

Para eliminar un comentario o una instrucción de procesamiento:

- Haga clic con el botón secundario en el nodo, seleccione **Comentario/instrucción de procesamiento** y después haga clic en la opción correspondiente en el menú contextual.

10.1.6 Secciones CDATA

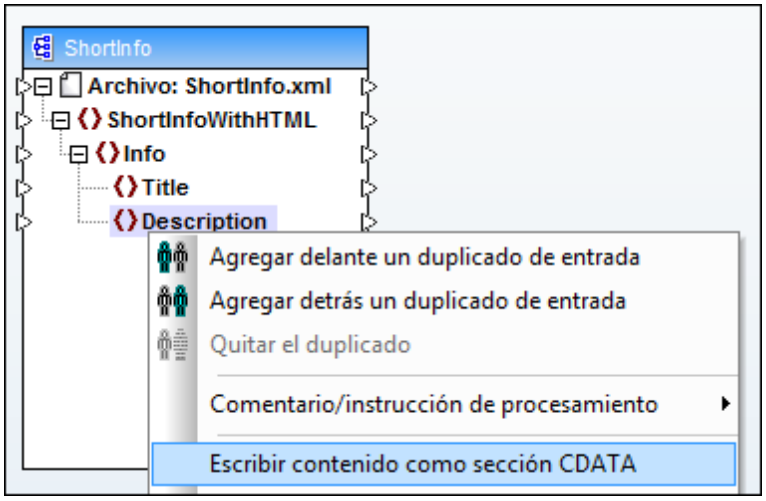
Las secciones CDATA sirven para excluir bloques de texto que contienen caracteres que de lo contrario serían interpretados como código. Las secciones CDATA empiezan con `<![CDATA[` y terminan con `]]>`.

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 bases de datos
- elementos XML secundarios de dimensiones con tipo de un destino XBRL

Para crear una sección CDATA:

1. Haga clic con el botón secundario en el nodo de destino que desea definir como sección CDATA y elija **Escribir contenido como sección CDATA** en el menú contextual.



2. Aparece un aviso advirtiendo de que los datos de entrada no deben tener el delimitador de cierre `]]>`.
   Haga clic en **Aceptar** para cerrar el aviso.
3. El icono `[C..` aparece bajo la etiqueta de elemento para indicar que ahora el nodo está definido como sección CDATA.

Nota: también puede definir secciones CDATA en nodos duplicados y en nodos xsi:type.

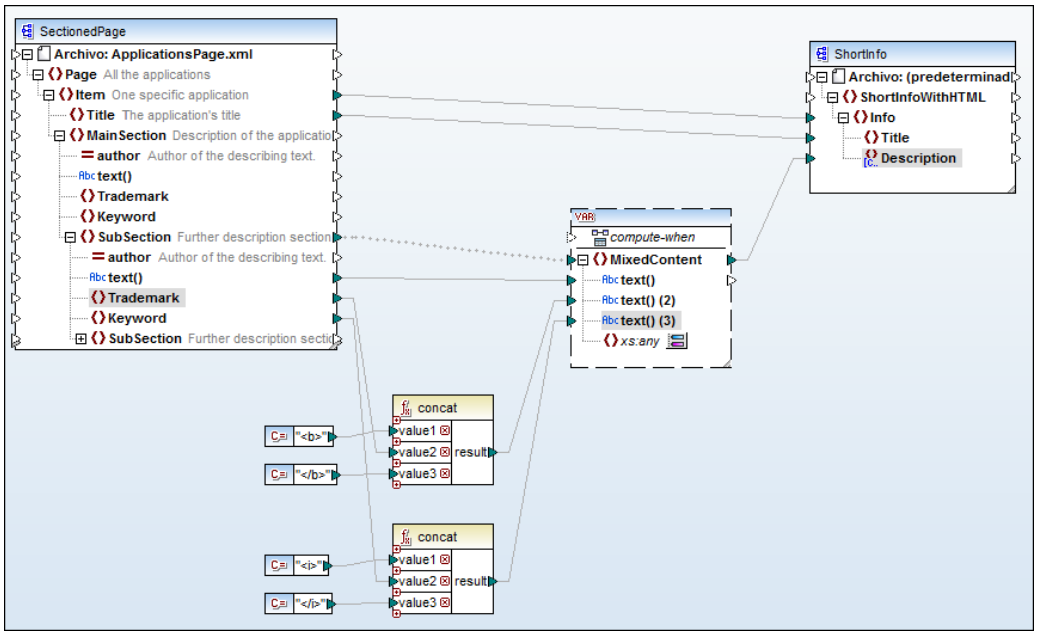
Ejemplo

La asignación `HTMLinCDATA.mfd` de la carpeta `...\MapForceExamples` muestra un ejemplo de cómo y cuándo usar secciones CDATA.
En este ejemplo:

- Las etiquetas de negrita (`<b>`) y (`</b>`) se añaden al contenido del elemento de origen **`Trademark`**.

 © 2014 Altova GmbH

- Las etiquetas de cursiva (`<i>`) y (`</i>`) se añaden al contenido del elemento de origen **`Keyword`**.
- Los datos resultantes se pasan a nodos **`text()`** duplicados en el orden en el mismo orden en que aparecen en el documento de origen. Esto se debe a que el conector del elemento **`Subsection`** se definió como nodo basado en origen (contenido mixto).
- Después el resultado del nodo **`MixedContent`** se pasa al nodo **`Description`** del componente de destino **`ShortInfo`**, que se definió como sección CDATA.

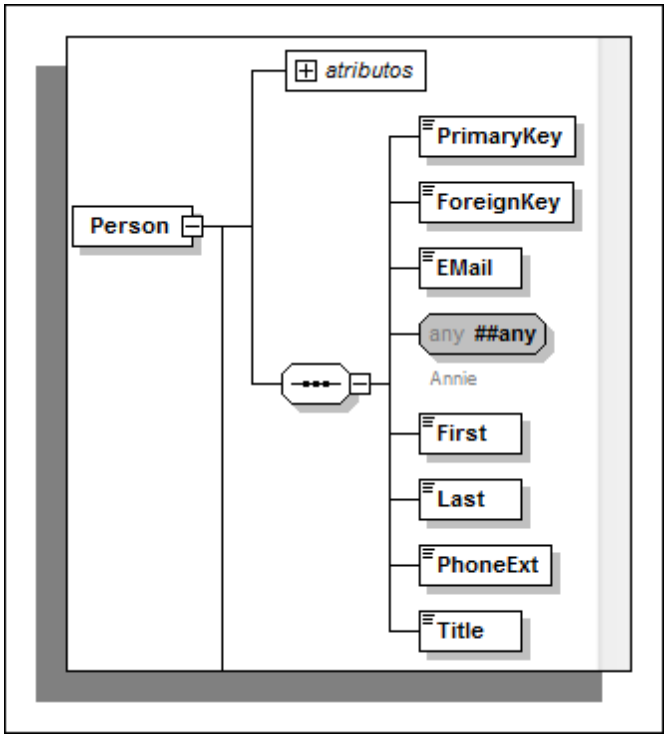


Al hacer clic en la pestaña **Resultados** aparece la sección CDATA que contiene el texto con marcado.

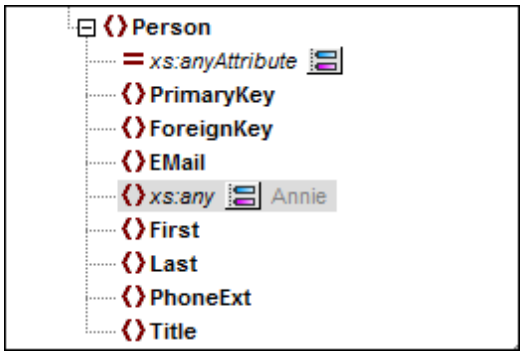
```
7    <Info>
8      <Title>MapForce</Title>
9      <Description><![CDATA[Altova <b>MapForce</b> 2014 Enterprise Edition is the premier <i>XML</i>
/ <i>database</i> / <i>flat file</i> / <i>EDI</i> data mapping tool that auto-generates mapping code in
<i>XSLT</i> 1.0/2.0, <i>XQuery</i> , <i>Java</i> , <i>C++</i> and <i>C#</i> . It is the definitive tool for
data integration and information leverage.]]></Description>
10   </Info>
```

10.1.7 Comodines xs:any / xs:anyAttribute

Los comodines `xs:any` (y `xs:anyAttribute`) permiten usar cualquier elemento o atributo del esquema XML correspondiente. Por ejemplo, en la imagen siguiente puede ver el elemento "any" en la vista **Esquema** de XMLSpy.

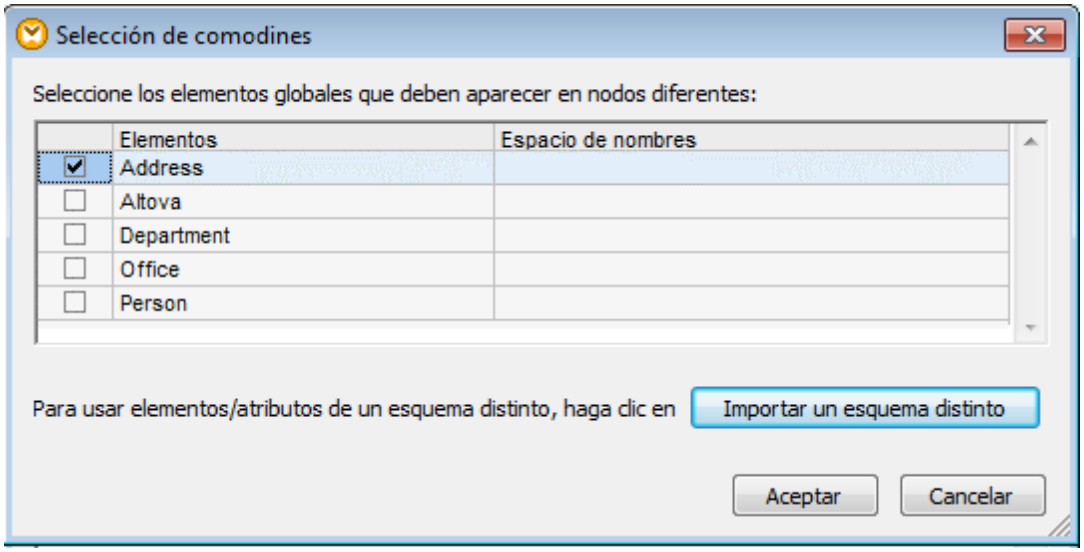


En MapForce, la estructura de este esquema puede verse haciendo clic en el botón de selección situado a la derecha del elemento `xs:any` (y de `xs:anyAttribute`).

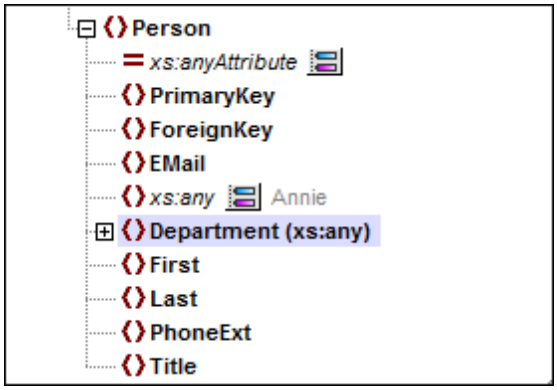


Al hacer clic en el botón de selección de *xs:any* se abre el cuadro de diálogo "Selección de comodines", que enumera los elementos/atributos globales declarados en el esquema actual.

 © 2014 Altova GmbH



Marque las casillas correspondientes y pulse **Aceptar**. Los elementos/atributos seleccionados (y sus secundarios si los hubiera) se insertan en el componente en la posición elegida.



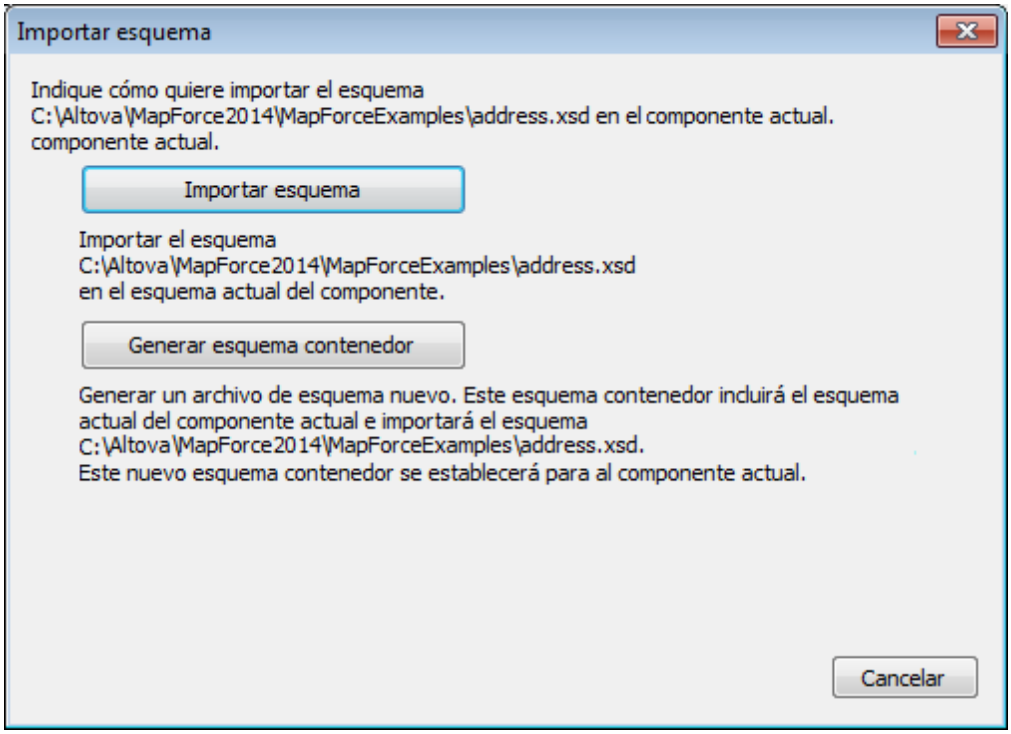
Ahora puede crear asignaciones entre estos nodos y cualquier otro elemento.

Para eliminar un elemento comodín:

- Haga clic en el botón de selección y desactive las casillas de los elementos globales seleccionados.

Para usar elementos de otro esquema como comodín:

1. Haga clic en el botón **Importar un esquema distinto** del cuadro de diálogo "Selección de comodines".
2. Seleccione el esquema cuyos elementos desea importar.
   Ahora aparece un cuadro de diálogo donde puede elegir si importa el esquema adicional en el esquema actual o si prefiere generar un esquema contenedor nuevo que contenga referencias a ambos esquemas.

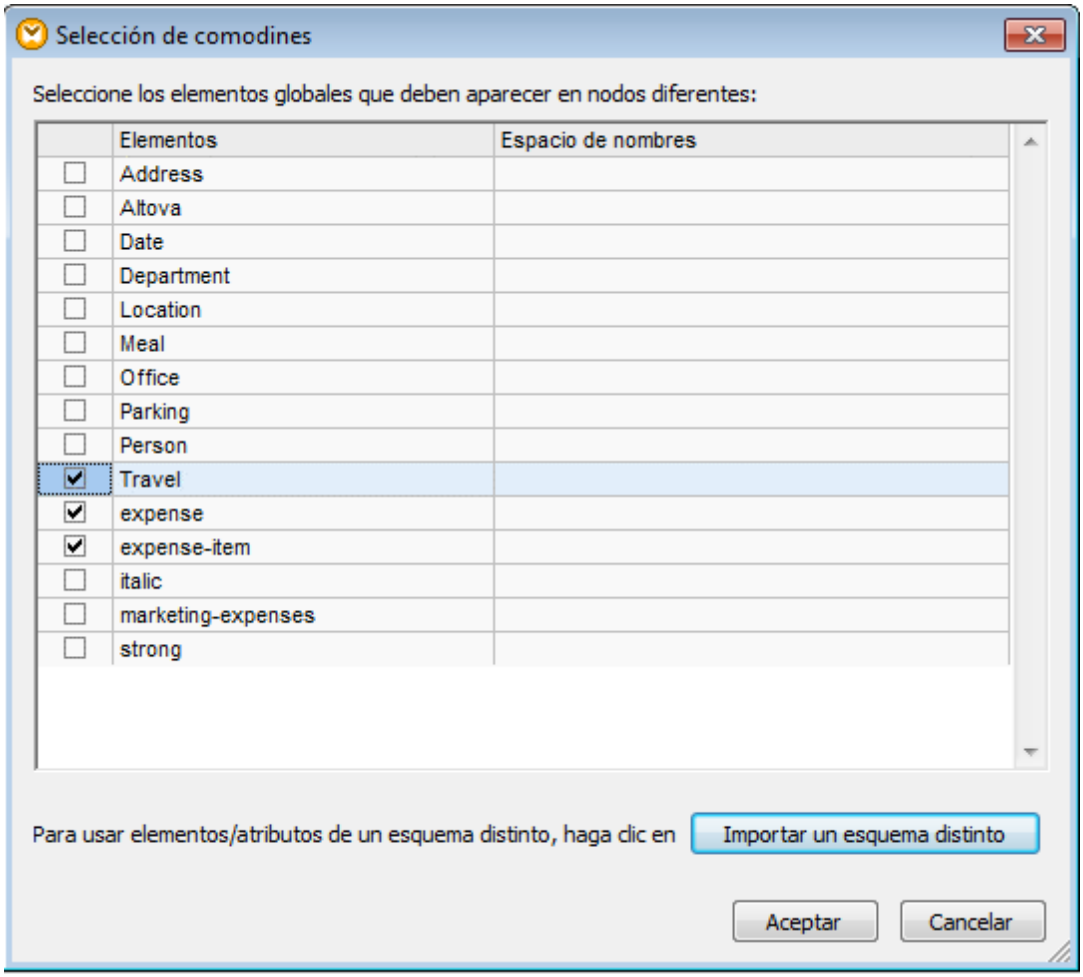


3. Haga clic en el botón correspondiente.

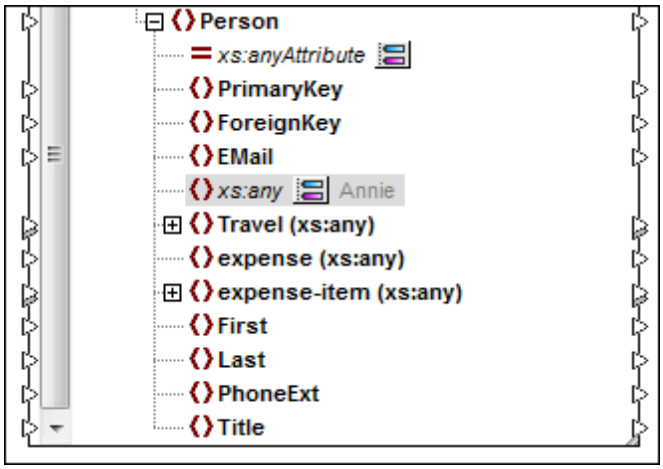
Importar esquema

La imagen siguiente muestra los elementos elementos globales disponibles si el esquema importado es el esquema **HasExpenses.xsd** guardado en la carpeta ...\MapForceExamples. Para este ejemplo seleccionamos los elementos `Travel`, `expense` y `expense-item`.

 © 2014 Altova GmbH



Tras hacer clic en **Aceptar**, estos tres elementos (y sus secundarios si los tienen) se añaden al componente, justo debajo del nodo comodín *xs:any*.



Cada uno de los nodos importados tiene la anotación **(xs:any)**, indicando que se trata de un nodo importado.

Nota: si importa un esquema, el esquema se **sobrescribe**. Por tanto, necesitará tener

derechos de usuario para hacerlo. Si se trata de un esquema remoto que abrió usando el botón **Cambiar a URL**, el esquema no se puede sobrescribir y por tanto no se puede importar.

Generar esquema contenedor

1. Haga clic en el botón **Generar esquema contenedor** del cuadro de diálogo "Importar esquema", escriba el nombre del nuevo esquema contenedor y pulse **Guardar**.
   El esquema contenedor recibe un nombre predeterminado tipo `XXXX-wrapper.xsd`. Este nuevo esquema contenedor incluirá el esquema actual e importará el otro esquema.

   MapForce muestra un aviso y le pregunta si quiere que la ubicación del esquema (de la Configuración del componente) haga referencia al esquema contenedor o si prefiere que apunte al esquema principal anterior.

2. Haga clic en **No** si prefiere conservar la referencia al esquema contenedor. O haga clic en **Sí** para cambiar la ubicación del esquema por la del esquema principal anterior.
   Los nodos importados se añaden al componente, justo debajo del nodo comodín.

Nota: la función **Generar esquema contenedor** permite usar esquemas remotos (cargados desde una URL) y añadir elementos de otro esquema al componente actual.

 © 2014 Altova GmbH

10.1.8 Firmas digitales

Las firmas digitales son una especificación del W3C para la firma digital de documentos XML con un código cifrado que se puede usar para comprobar si se realizaron cambios en el documento XML. La función de firma XML de MapForce es compatible con certificados de tipo RSA-SHA1 y DSA-SHA1.

Para más información consulte la especificación del W3C (http://www.w3.org/TR/xmldsig-core/).

En MapForce puede crear firmas XML digitales para archivos de salida XML y XBRL. Las firmas digitales solamente se pueden generar si el destino de los resultados es BUILTIN y solamente en la vista previa. La firma del archivo de resultados generado se crea al abrir el panel **Resultados** y cuando se guarda el archivo de resultados.

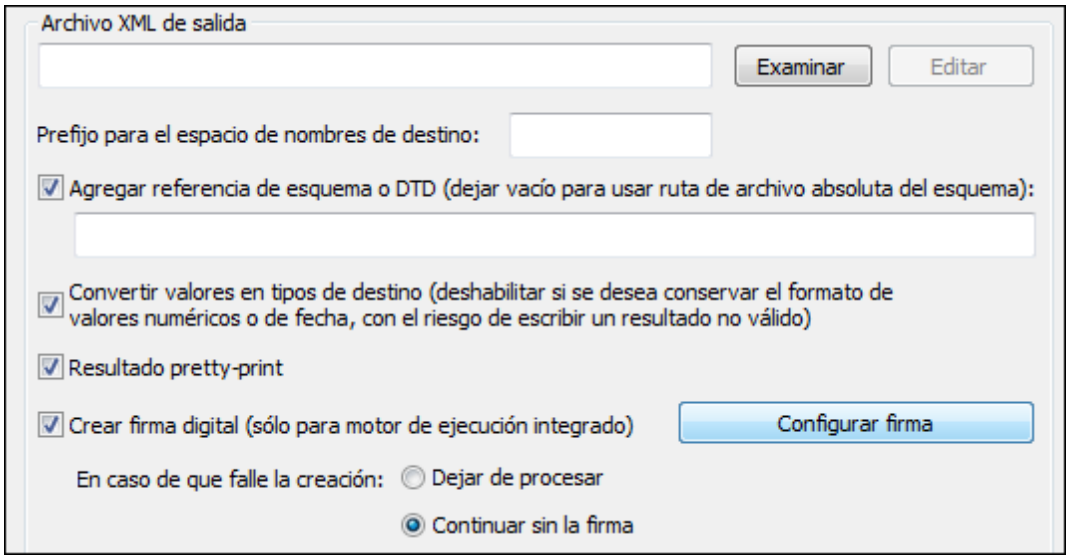
Nota: MapForce Server no es compatible con firmas digitales.

Las firmas digitales se pueden incrustar como último elemento del documento de salida o se pueden almacenar en un archivo de firma separado.

- Si la firma es *envuelta*, entonces es el último elemento secundario del elemento raíz del archivo XML.
- Si la firma es *separada*, entonces se genera un archivo de firma.

Para activar la generación de firmas digitales

1. Abra el cuadro de diálogo "Configuración del componente" del componente de salida (haciendo doble clic en su título o seleccionando **Componente** | **Propiedades**).
2. Marque la casilla *Crear firma digital*.



3. Aparece el cuadro de diálogo Configuración de las firmas XML.
4. Elija las opciones de configuración de la firma y, cuando termine, haga clic en **Aceptar**.

Para cambiar la configuración de las firmas digitales:

1. Abra el cuadro de diálogo "Configuración del componente" del componente de salida.
2. Haga clic en el botón **Configurar firma** para abrir el cuadro de diálogo Configuración de

las firmas XML.

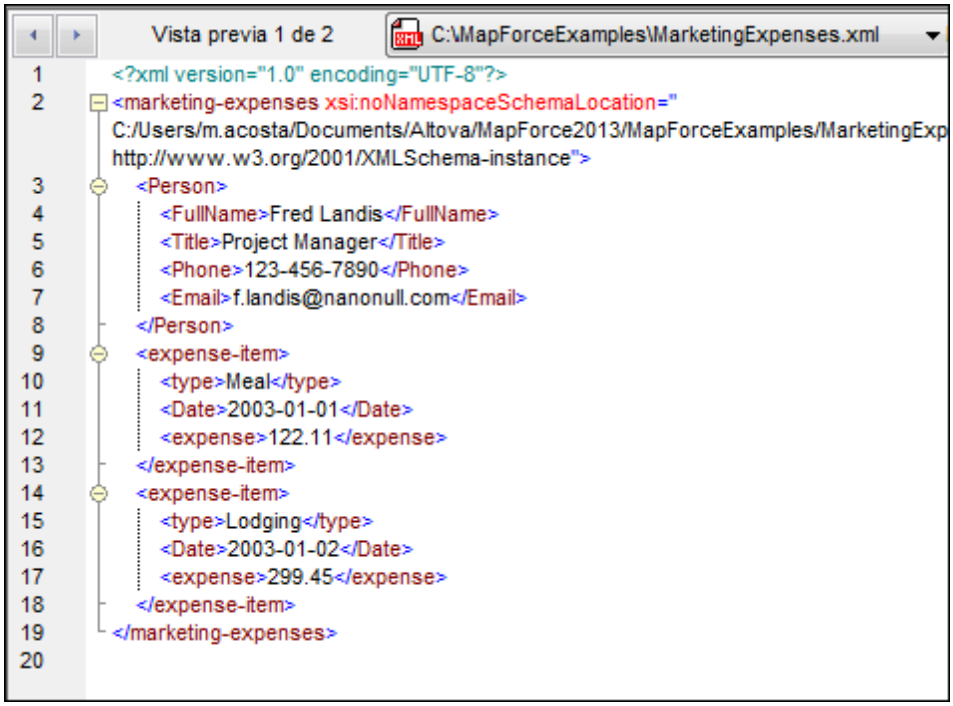
3. Elija las nuevas opciones de configuración y, cuando termine, haga clic en **Aceptar**.

Usemos como ejemplo el archivo `MarketingExpenses_DetachedSignature.mfd`, disponible en la carpeta `...\MapForceExamples`.

1. Haga doble clic en el componente de destino **MarketingExpenses**.
2. Pulse el botón **Configurar firma** para ver la configuración de la firma.
3. Abra el panel **Resultados** para ver el resultado de la asignación.

Se generan dos archivos de salida para la vista previa:

- El archivo XML `MarketingExpenses.xml`, que es el resultado de la asignación.



- El archivo `MarketingExpenses.xml.xsig`, que es un archivo de firma digital temporal generado por el componente de destino.

 © 2014 Altova GmbH

Vista previa 2 de 2 Firma XML: C:\MapForceExamples\MarketingExpenses.xml.xsig

El archivo de firma digital que se muestra es un archivo temporal para un archivo de resultados temporal. Cuando se guarde el resultado, se creará y guardará la firma para el archivo de resultados generado.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsig:Signature xmlns:xsig="http://www.w3.org/2000/09/xmldsig#">
3    <xsig:SignedInfo>
4      <xsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"></
   xsig:CanonicalizationMethod>
5      <xsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1"></
   xsig:SignatureMethod>
6      <xsig:Reference URI="~mfD607.tmp">
7        <xsig:Transforms>
8          <xsig:Transform Algorithm="http://www.altova.com/xml-signatures/strip-whitespaces"></
   xsig:Transform>
9        </xsig:Transforms>
10       <xsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></xsig:DigestMethod>
11       <xsig:DigestValue>veVnvxXlz80YrxYl8rZPNAkl5/o=</xsig:DigestValue>
12     </xsig:Reference>
13   </xsig:SignedInfo>
14   <xsig:SignatureValue>+t/rJnlN9UxMdax8740SNSaakvE=</xsig:SignatureValue>
15 </xsig:Signature>
```

Para generar el archivo de firma:

1. Haga clic en el icono **Guardar todos los resultados generados** de la barra de herramientas del panel **Resultados**.
   Esto genera los archivos `.XML` y `.XSIG` en el directorio de salida.

Por su parte, el archivo `MarketingExpenses_EnvelopedSignature.mfd`, disponible en la carpeta `...\MapForceExamples` muestra el resultado que se obtiene al utilizar una firma *envuelta*.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <marketing-expenses xmlns:xsig="http://www.w3.org/2000/09/xmldsig#" xsi:noNamespaceSchemaLocation="
   C:/DOCUME~1/MYDOCU~1/Altova/MapForce2011/MapForceExamples/MarketingExpensesSigned.xsd" xmlns:xsi="
   http://www.w3.org/2001/XMLSchema-instance">
3    <Person>
4      <FullName>Fred Landis</FullName>
5      <Title>Project Manager</Title>
6      <Phone>123-456-7890</Phone>
7      <Email>f.landis@nanonull.com</Email>
8    </Person>
9    <expense-item>
10     <type>Meal</type>
11     <Date>2003-01-01</Date>
12     <expense>122.11</expense>
13   </expense-item>
14   <expense-item>
15     <type>Lodging</type>
16     <Date>2003-01-02</Date>
17     <expense>299.45</expense>
18   </expense-item>
19   <xsig:Signature>
20     <xsig:SignedInfo>
21       <xsig:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
22       <xsig:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#hmac-sha1"/>
23       <xsig:Reference URI="">
24         <xsig:Transforms>
25           <xsig:Transform Algorithm="http://www.altova.com/xml-signatures/strip-whitespaces"/>
26           <xsig:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/>
27         </xsig:Transforms>
28         <xsig:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>
29         <xsig:DigestValue>Hvl5SnMsLf8RBsgy7flj5+leD0M=</xsig:DigestValue>
30       </xsig:Reference>
31     </xsig:SignedInfo>
32     <xsig:SignatureValue>a+uml6qhJ1SN1J00/rKV7h99Y6g=</xsig:SignatureValue>
33   </xsig:Signature>
34 </marketing-expenses>
```

Validez del documento XML

Si una firma XML se incrusta en el documento XML, se añade al documento un elemento `Signature` del espacio de nombres `http://www.w3.org/2000/09/xmldsig#`. Para que el documento siga siendo válido con respecto al esquema, este debe contener las declaraciones de elemento pertinentes.

MapForce incrusta la firma en el documento XML cuando se selecciona la opción *Firma envuelta*: el elemento **`Signature`** se crea como último elemento secundario del elemento raíz (o de documento).

Si prefiere no modificar el esquema asociado a su documento XML, MapForce puede crear la firma en un archivo externo (opción *Firma separada*).

A continuación puede ver fragmentos de esquemas XML que ilustran cómo se puede autorizar el elemento `Signature` de una firma envuelta. Puede usar estos ejemplos como base a la hora de modificar sus propios esquemas.

En el primer fragmento, el esquema XML Signature se importa en el esquema del usuario. El esquema XML Signature está en está dirección `http://www.w3.org/TR/xmldsig-core/`

xmldsig-core-schema.xsd

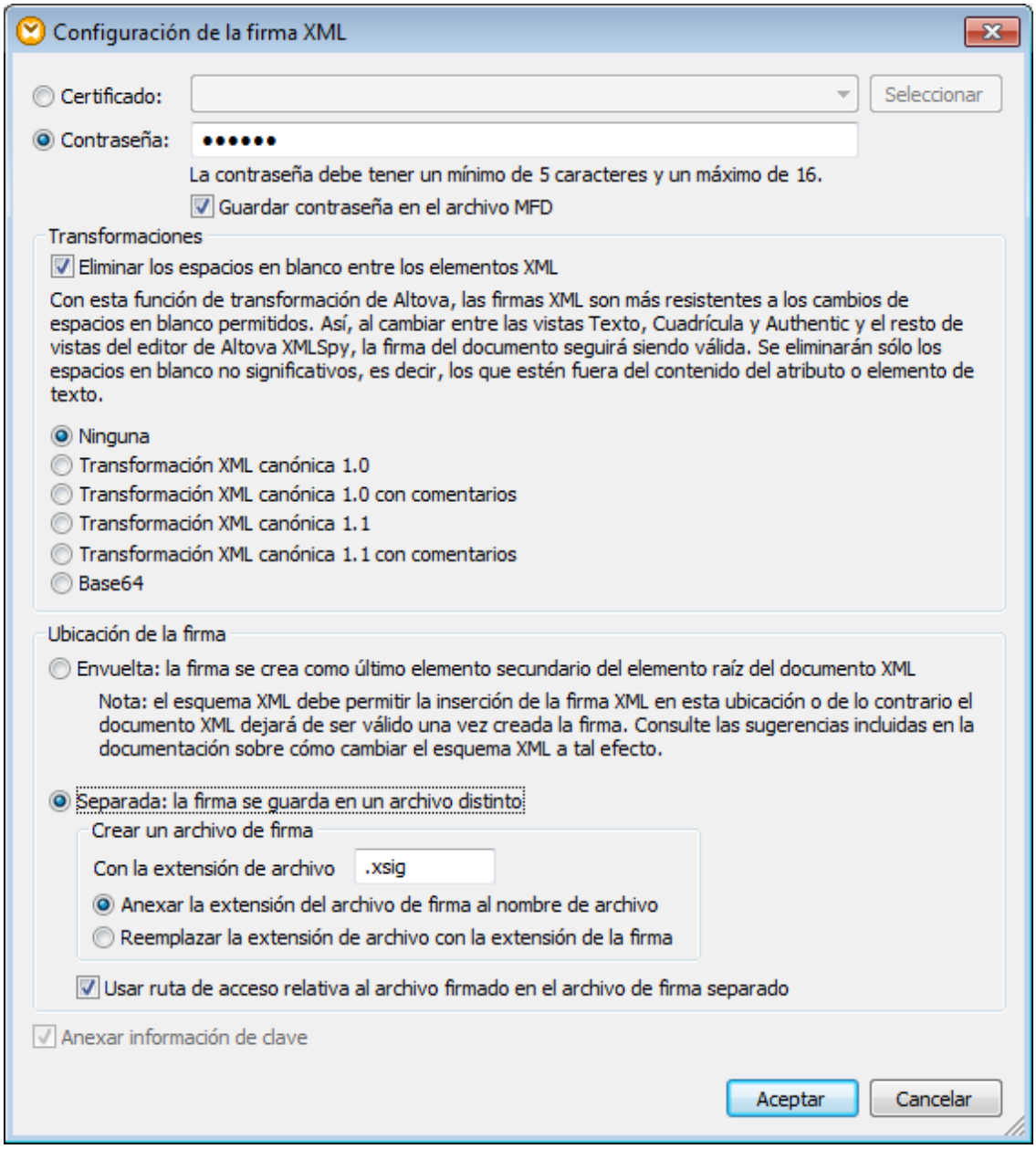
```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:xsig="http://www.w3.org/2000/09/xmldsig#"
           elementFormDefault="qualified"
           attributeFormDefault="unqualified">
   <xs:import namespace="http://www.w3.org/2000/09/xmldsig#"
              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>
```

El segundo fragmento muestra una alternativa: añadir un elemento comodín genérico que coincida con todos los elementos de otros espacios de nombres. Si al atributo `processContents` le damos el valor `lax` ,el validador pasará por alto este elemento (porque no se encuentra ninguna declaración de elemento coincidente). Por consiguiente el usuario no necesita hacer referencia al esquema XML Signatures. El inconveniente de esta opción, sin embargo, es que entonces se puede añadir cualquier elemento (no sólo el elemento `Signature`) en la ubicación especificada en el 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>
```

Opciones de configuración de la firma XML

Las opciones de configuración de la firma de cada componente se definen en el cuadro de diálogo "Opciones de la firma XML" (*ver imagen*) y se almacenan en el archivo MFD cuando este se guarda.



Método de autenticación: certificado o contraseña
La firma puede estar basada en un certificado o en una contraseña. Seleccione el botón de opción del método que prefiera utilizar.

- *Certificado:*
  Si desea usar un certificado, este debe tener una clave privada y debe estar ubicado en un almacén de certificados accesible. La firma se genera usando la clave privada del certificado. Para verificar la firma, será necesario tener acceso al certificado (o a su versión de clave pública). La clave pública del certificado se usa para verificar la firma. Para seleccionar el certificado de clave pública/privada que desea usar, haga clic en el botón **Seleccionar** y busque el certificado.
- *Contraseña:*

Escriba una contraseña con una longitud de entre 5 y 16 caracteres. Esta contraseña se utiliza para crear la firma y será necesaria para verificarla. El botón **Aceptar** del cuadro de diálogo no se habilita hasta que introduzca la contraseña.

- *Guardar contraseña en el archivo MFD:*
  Si marca esta casilla, la contraseña introducida en el campo *Contraseña* se guarda en forma confusa en el archivo MFD, es decir, la contraseña está cifrada, en un formato ilegible. Recuerde que cualquiera con acceso al archivo MFD puede crear firmas usando esta contraseña.

Transformaciones

Los datos XML se transforman y los resultados de la transformación se utilizan para crear la firma. Puede especificar el algoritmo de canonización que se debe aplicar a los datos XML del archivo (el contenido del elemento `SignedInfo`) antes de realizar los cálculos de la firma. A continuación señalamos las diferencias entre los diferentes algoritmos:

- *Transformación XML canónica con o sin comentarios:*
  Si los comentarios se incluyen en el cálculo de la firma y se realizan cambios en los comentarios de los datos XML, la verificación de la firma dará un error. Si no se incluyen en el cálculo de la firma, entonces sí puede añadir o modificar comentarios después de firmar el documento y la verificación de la firma será válida.

  Nota: la opción "*...con comentarios*" solamente está disponible si elige una firma *separada*.

- *Base64:*
  Si marca esta opción, se considera que el elemento raíz (o de documento) del documento XML tiene codificación Base64 y se lee en su forma binaria. Si el elemento raíz no tiene codificación Base64, se devuelve un mensaje de error o el elemento se lee como vacío, dependiendo del tipo de elemento.

- *Ninguna:*
  Si marca esta opción, no se realiza ninguna transformación y los datos XML del archivo binario guardados en disco se transmiten directamente para crear la firma. Cualquier cambio posterior genera un error de verificación.

  No obstante, si está seleccionada la casilla *Eliminar los espacios en blanco entre los elementos XML*, entonces se eliminan todos los espacios en blanco y no se tienen en cuenta los cambios en los espacios en blanco.

  Una diferencia importante entre la opción *Ninguna* y cualquier opción *Transformación canónica,* es que esta última produce una secuencia de datos XML, en la que se normalizan determinadas diferencias como, por ejemplo, el orden de los atributos. Es decir, la transformación canónica normalizará todos los cambios que afecten al orden de los atributos (y se podrá verificar la firma correctamente), mientras que con la opción *Ninguna* esos cambios no permiten verificar la firma.

  De toda maneras, recuerde que si la firma está incrustada en el documento (firma *envuelta*), se realizará una transformación canónica predeterminada. De modo que los datos XML se usarán tal y como están (sin transformarse) cuando (i) se trata de una firma *separada*, (ii) se selecciona la opción *Ninguna* y (iii) está desactivada la casilla *Eliminar espacios en blanco*.

Ubicación de la firma
La firma puede ubicarse dentro del archivo XML o en un archivo separado. Estas son las opciones disponibles:

- *Envuelta:*
  El elemento de firma se crea como último elemento secundario del elemento raíz (de documento). Recuerde que el esquema XML asociado debe contener la definición del elemento `Signature` o, de lo contrario, el archivo XML de salida no será válido. Para más información, lea el apartado anterior.
- *Separada:*
  La firma XML se crea en un archivo separado. En este caso, puede especificar la extensión del archivo de firma y si (i) la extensión de archivo se anexa al nombre del archivo XML (p. ej. `test.xml.xsig`) o (ii) la extensión de archivo reemplaza la extensión XML del archivo XML (p. ej. `test.xsig`). También puede indicar si la referencia al archivo XML dentro del archivo de firma es una ruta de archivo relativa o absoluta.

Nota: las firmas XML para archivos de esquema XML (`.xsd`) y para archivos XBRL solamente se pueden crear en archivos de firma externos. Para los archivos WSDL, las firmas se pueden crear en archivos externos o como firmas envueltas dentro del archivo WSDL.

Nota: si la firma XML se crea en un archivo separado, el archivo XML y el archivo de firma se asocian por medio de una referencia presente en el archivo de firma. Por consiguiente, para poder verificar la firma, el archivo activo debe ser el archivo de firma, no el archivo XML.

Anexar información de clave
Si la firma está basada en un certificado, entonces está disponible la opción *Anexar información de clave*. Si la firma se basa en una contraseña, esta opción se deshabilita.

Si marca esta casilla, la información de clave pública se coloca dentro de la firma. De lo contrario, esa información de clave no se incluye en la firma. La ventaja de incluir la información de clave en la firma es que el certificado (en concreto, su información de clave pública) no será necesario en el proceso de verificación (puesto que la información de clave está presente en la firma).

Opciones de configuración de firma no válidas
MapForce no puede firmar digitalmente un archivo de salida si las opciones de configuración de la firma no son válidas. Esto ocurre si:

- no se puede acceder al certificado seleccionado o el certificado no es adecuado para la firma de documentos XML o
- no se establece ninguna contraseña (no marca la casilla *Guardar contraseña en el archivo MFD*).

En estos casos, al hacer clic en la pestaña del panel **Resultados**, MapForce emite estos mensajes:

- para la firma:

  Indique la contraseña que desea utilizar para firmar el resultado del componente XXXX

- para el certificado:

 © 2014 Altova GmbH

Elija el almacén que contiene el certificado que desea utilizar para firmar el resultado del componente XXX

Si no elige la contraseña/el certificado, entonces puede pasar dos cosas: o bien se detiene el procesamiento o bien continúa sin la firma. Esto también se configura en el cuadro de diálogo "Configuración del componente" (opción *Dejar de procesar* y opción *Continuar sin la firma*, respectivamente).

Si la asignación se ejecuta desde la línea de comandos, no aparece ningún cuadro de diálogo. En este caso pueden pasar dos cosas: o bien se detiene la ejecución y se emite un error o bien continúa sin la firma.

10.2 Trabajar con bases de datos

Sitio web de Altova: Asignación de datos de BD

MapForce 2014 ofrece potentes funciones de asignación de datos entre bases de datos y archivos XML, archivos planos, EDI, Excel 2007+, XBRL, servicios web e incluso bases de datos de otro tipo.

MapForce tiene en cuenta las restricciones de clave principal y foránea y genera transacciones que garantizan la integridad de los datos.

Las bases de datos compatibles y sus objetos raíz aparecen a continuación. Altova procura ofrecer compatibilidad con otras bases de datos ODBC/ADO, 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. MapForce es compatible con los archivos lógicos de la base de datos IBM iSeries y los muestra como si fueran vistas.

En la lista también aparece el objeto raíz disponible por cada base de datos compatible.

Base de datos (con soporte nativo)	Objeto raíz
MS Access 2003, 2007, 2010, 2013	database
MS SQL Server 2000, 2005, 2008 y 2012	database
MS SQL Server 2005, 2008 y 2012	schema
Oracle 9i, 10g y 11g	schema
MySQL 4.x, 5.x y 5.5.28	database
PostgreSQL 8.0, 8.1, 8.2, 8.3, 9.0.10, 9.1.6, 9.2.1	database
Sybase ASE12 y ASE15	database
IBM DB2 8.x, 9.5, 9.7 y 10.1	schema
IBM DB2 para iSeries 6.1 y 7.1	schema
IBM Informix 11.70	database

Para poder utilizar bases de datos es necesario seleccionar Java, C#, C++ o BUILTIN (el motor de ejecución integrado). MapForce solamente puede generar código **XQuery** para archivos XML de origen/destino.

Notas: cuando se conecte a una base de datos Sybase a través de JDBC, la instrucción SELECT no puede recuperar datos. Se trata de un problema conocido de controlador. Utilice otro controlador para conectarse a la BD.

Si recibe este mensaje de error cuando se conecte a la BD a través del cliente SQL nativo:
Error de conexión de base de datos.
Motivo: no se puede conectar a la base de datos. Error de inicio de sesión. El inicio de sesión se realiza desde un dominio que no es de confianza y no se puede utilizar con autenticación de Windows.

 © 2014 Altova GmbH

Entonces siga estas instrucciones:

Para una conexión ADO a un MS SQL Server con controlador SQL Server Native Client 10.0 debe establecer estos valores de propiedades en la pestaña *Todas* del cuadro de diálogo "Propiedades de vínculo de datos":
(i) el valor *Seguridad integrada* debe ser un carácter de espacio
(ii) el valor de *Información de seguridad persistente* debe ser `true`.

MapForce es compatible con archivos lógicos de la base de datos IBM iSeries y presenta los archivos lógicos como vistas de la BD.

El controlador ODBC 5.2 es necesario si usa procedimientos sin parámetros de salida con MySQL 5.5.

Cuando genere archivos JAR a partir de código Java generado, recuerde que las entradas CLASSPATH para el control de la BD no se añaden automáticamente al archivo de manifiesto. Debe editar el archivo MANIFEST.MF a mano añadiendo el JAR externo del controlador de la BD a la CLASSPATH (p. ej. añada esta línea al archivo de manifiesto `Class-Path: mysql-connector-java-5.1.20-bin.jar`).

Nota sobre compatibilidad con IBM Informix:
Informix admite conexiones ADO, JDBC y ODBC.

La implementación no es compatible con tipos de datos de objetos grandes en ninguno de los lenguajes de generación de código. MapForce emite un mensaje de error (durante la generación de código) si se usan estos tipos de datos.

Cuando genere archivos JAR a partir del código Java generado, recuerde que las entradas classpath para el controlador de BD no se añaden automáticamente al archivo de manifiesto. Debe editar manualmente el archivo MANIFEST.MF añadiendo el JAR externo del controlador de BD a la entrada classpath. P. ej. anexe al archivo de manifiesto esta línea: `Class-Path: mysql-connector-java-5.1.20-bin.jar`.

10.2.1 Instalar clientes y controladores de BD

El objetivo de este documento es ayudar al usuario final con la instalación de los clientes de las bases de datos compatibles con la aplicación. Este documento no pretende ser un recurso definitivo sino una recomendación y se facilita sin garantía alguna.

No obstante, recomendamos evitar la instalación manual de la base de datos y que su administrador de base de datos instale el software cliente.

En los apartados siguientes encontrará información sobre la instalación de varios clientes/ controladores de BD.

Requisito: la base de datos debe estar instalada en su entorno y es necesario tener acceso al servidor de BD a través de una red LAN.

SQL Server

Bases de datos Microsoft SQL:

SQL Server 2000
SQL Server 2005
SQL Server 2008

SQL Server 2000

- Pídale el cliente de BD Microsoft SQL Server a su administrador de base de datos e instálelo.
- Inicie la herramienta Enterprise Manager, haga clic con el botón secundario en *SQL SERVER Group*, elija la opción *New SQL Server Registration* y añada un servidor.
- Pídale los datos de inicio de sesión (nombre de usuario y contraseña) a su administrador de base de datos para poder establecer la conexión.
- Elija la opción *SQL Server authentication*, escriba el nombre de usuario y la contraseña y después añada SQL Server a un grupo SQL Server Group.
- Expanda el árbol situado a la izquierda y seleccione una base de datos: Microsoft SQL Server->SQL Server Group->{Nombre del servidor}->Databases->{Nombre de la BD}

SQL Server 2005

- Pídale el cliente de BD Microsoft SQL Server a su administrador de base de datos e instálelo.
- En el diálogo "Components to install" seleccione *Client Components* y *Document Components.*
- No seleccione ningún componente y haga clic en **Advanced**.
- Seleccione *Client Components* (se instalará todo el contenido)
- Seleccione la opción "Documentation, Samples, and..." (se instalará todo el contenido)
- Haga clic en **Next** y después en **Install**.

 © 2014 Altova GmbH

SQL Server 2008

- Pídale el cliente de BD Microsoft SQL Server a su administrador de base de datos e instálelo.

Oracle

Bases de datos Oracle:

Oracle 9
Oracle 10
Oracle 11

Oracle 9i

- Pídale el cliente de Oracle a su administrador de base de datos e instálelo.
- Copie los archivos **`sqlnet.ora`** y **`tnsnames.ora`** del paquete en la carpeta `...\ORACLE\ora92\network\ADMIN`.
- Pídale los datos de inicio de sesión (nombre de usuario y contraseña) a su administrador de base de datos para conectarse a la base de datos.

Puede conseguir más descargas en http://www.oracle.com/technology/software/products/oracle9i/index.html.

Oracle 10g

- Pídale el cliente de Oracle a su administrador de base de datos e instálelo.
- Durante la instalación se inicia el asistente Oracle Net Configuration Assistant.
- Haga clic en **Next** en todos los cuadros de diálogo, sin cambiar ninguna de las opciones predeterminadas.
- Uno de los cuadros de diálogo le pedirá el nombre de **servicio** y otro le pedirá el nombre del **servidor**. Escriba el servicio y el nombre de servidor.
- Use el puerto estándar 1521 o pregunte a su administrador cuál es el número de puerto que debe usar.
- Pídale los datos de inicio de sesión (nombre de usuario y contraseña) a su administrador de base de datos para conectarse a la base de datos.

Puede conseguir más descargas en http://www.oracle.com/technology/software/products/database/index.html.

Oracle 11g

- Pídale el cliente de Oracle a su administrador de base de datos e instálelo.
- Siga las instrucciones de instalación.
- Pídale los datos de inicio de sesión (nombre de usuario y contraseña) a su administrador de base de datos para conectarse a la base de datos.

Puede conseguir más descargas en http://www.oracle.com/technology/software/products/database/index.html.

Instalar el cliente Oracle

Estas son las instrucciones para instalar una conexión nueva con una base de datos Oracle existente en su red local. El asistente de configuración sigue el mismo orden que en la instalación del servicio NET durante la instalación inicial del cliente Oracle.

1. Seleccione la opción de menú **Programs | Oracle - OraHome92 | Configuration and**

migration tools | Net Configuration Assistant.
Se abre el asistente Oracle Net Configuration Assistant.

2. Haga clic en el botón de opción *Local Net Service Name configuration* y haga clic en **Next**.
3. Haga clic en **Add** para agregar un nombre de servicio net nuevo y después haga clic en **Next**.
4. Seleccione la versión de Oracle que instaló (p. ej **Oracle 8i** or later) y haga clic en **Next**.
5. Escriba el nombre de servicio (Service name) de la base de datos a la que desea conectarse (p. ej. TestDB) y haga clic en **Next**. El nombre de servicio de la BD suele ser su nombre de base de datos global.
6. Seleccione el protocolo de red utilizado para acceder a la BD (p. ej. TCP) y haga clic en **Next**.
7. Escriba el nombre de host del equipo en el que se instaló la BD y el número de puerto. Haga clic en **Next** para continuar.
8. Seleccione el botón de opción *Yes* para probar la conexión de la BD y después haga clic en **Next**.
9. Puede cambiar los parámetros de inicio de sesión si la prueba da un error (haciendo clic en el botón **Change Login** y probando la conexión otra vez). Para continuar haga clic en **Next**.
10. Escriba el nombre de servicio de red en el campo *Net Service Name* (aquí puede usar el nombre que quiera). Se trata del nombre que debe introducir en el campo *Base de datos* del cuadro de diálogo "Inicio de sesión de Oracle" de MapForce. Para continuar haga clic en **Next**.
11. Para cerrar el cuadro de diálogo haga clic en **Next**.

IBM DB2

Bases de datos IBM DB2:

DB2 versión 8 / 9
DB2 for i 5.4

DB2 Versión 8 / 9

Requisito: debe tener instalado JRE (Java Runtime Environment) 1.4.2 o superior.

- Pídale el paquete de instalación del cliente IBM DB2 a su administrador de BD e instálelo. Siga las instrucciones de instalación.
- Cuando termine la instalación, inicie el asistente Configuration Assistant para configurar la base de datos.
  Haga clic en IBMDB2->setup tools->Configuration Assistant.
  Aparece un mensaje preguntando si desea añadir una base de datos.
  Haga clic en **Yes** y se inicia el asistente Add Database Wizard.
  Elija la opción *Search the network*, después en **Next**, después en *Add system*, después en **Discover** y seleccione el nombre del sistema.
- Añada una base de datos:
  Inicie IBMDB2->General administration Tools->Control Center.
  Expanda el árbol: Control center->All Cataloged Systems->{Nombre del servidor}->Instances->{nombre_instancia}->Databases.
  Haga clic con el botón secundario en Databases y elija la opción de menú **Add**.
  Haga clic en **Discover**, seleccione una base de datos y haga clic en **Accept**.
- Registre la base de datos como origen de datos:
  Inicie el asistente Configuration Assistant y seleccione una base de datos.
  En la lista de la izquierda seleccione *data Source*.
  Elija la opción *Register this database for ODBC* y marque la casilla *As user data source*.

 © 2014 Altova GmbH

Haga clic en **Finish**.

- Pruebe la conexión con la base de datos:
  Inicie el asistente Configuration Assistant y seleccione una base de datos.
  Elija la opción de menú Selected->Test connection.
  Seleccione CLI, ODBC como tipo de conexión.
  Pídale el nombre de usuario y la contraseña a su administrador de base de datos y haga clic en **Test connection**.
- No olvide marcar la casilla *Remember Password*. Si no marca esta casilla, las opciones de configuración se pierden.

DB2 for i 5.4

Pida a su administrador de base de datos las instrucciones para acceder al servidor de la base de datos. Tenga en cuenta que iSeries es el nombre nuevo de los equipos AS400.

MySQL

Bases de datos MySQL

MySQL 4 / 5

MySQL 4 / 5

En el caso de las bases de datos MySQL no es necesario instalar un cliente, sino un controlador de base de datos.

¿Cómo instalar el controlador ODBC?

- Instale el controlador ODBC más reciente
- Inicie el Panel de control y seleccione **Herramientas administrativas**->**Origen de datos (ODBC)**
- Seleccione **DNS de usuario** y haga clic en **Agregar**.
- Seleccione la versión más reciente de *Controlador ODBC de MySQL* y haga clic en **Finalizar**.
- Rellene los campos *Nombre del origen de datos*, *Servidor*, *Usuario*, *Contraseña* y *Base de datos*.
- Haga clic en **Probar** y, si la conexión funciona, haga clic en **Aceptar**.

Puede encontrar más descargas en la página http://dev.mysql.com/downloads/connector/odbc/.

PostgreSQL

Bases de datos PostgreSQL

PostgreSQL 8.x

PostgreSQL 8.x

En el caso de las bases de datos PostgreSQL no es necesario instalar un cliente, sino un controlador de base de datos.

- Instale el controlador ODBC: ejecute **psqlodbc.msi** y siga las instrucciones.

Puede encontrar más descargas en la página http://www.postgresql.org/ftp/odbc/versions/msi

Sybase

Bases de datos Sybase:

Sybase 12

Sybase 12

Pídale a su administrador de base de datos el paquete cliente Sybase e instálelo.

- Instale el cliente Sybase y siga las instrucciones de instalación
- Quizás sea necesario editar el archivo `c:\sybase\ini\sql.ini`
- Inicie el Panel de control y seleccione **Herramientas administrativas**->**Origen de datos (ODBC)**
- Seleccione **DNS de usuario** y haga clic en **Agregar**.
- Seleccione la versión más reciente de *Controlador ODBC de Sybase ASE* y haga clic en **Finalizar**.
- Escriba el nombre del archivo DSN en el campo *Nombre del origen de datos*.
- Escriba Winsock en el campo *Nombre de la biblioteca de red.*
- Escriba "<nombre del servidor>,2048" en el campo *Dirección de red*.
- Escriba el <nombre de la base de datos> en el campo *Nombre de la base de datos.*
- Haga clic en **Probar** y, si la conexión funciona, haga clic en **Aceptar**.

Para más información visite la página http://www.sybase.com/products/databasemanagement/adaptiveserverenterprise.

Las entradas del archivo **`sql.ini`** suelen tener este formato:

```
[<nombreServidor>]
master=TCP, <nombreServidor>,<númeroPuerto>
query=TCP, <nombreServidor>,< númeroPuerto>
```

Microsoft Access

Microsoft Access 2004 y 2007

Para usar estas bases de datos no es necesario instalar clientes ni controladores. A estas bases de datos puede acceder directamente o a través de una red LAN.

Firebird

Bases de datos Firebird:

Firebird 2.0.5 / 2.1.2

Firebird 2.0.5 / 2.1.2

El controlador ODBC de Firebird exige instalar un DLL cliente. Por tanto, es necesario instalar primero el cliente y después el controlador.

- Ejecute el instalador ejecutable para Windows de Full Classic.
  Puede seleccionar la instalación cliente solamente.
- Ejecute **Full Install.exe** para instalar el controlador.
- Inicie el Panel de control y seleccione **Herramientas administrativas**->**Origen de datos (ODBC)**
- Seleccione **DNS de usuario** y haga clic en **Agregar**.
- Seleccione *Controlador de Firebird* y haga clic en **Finalizar**.

 © 2014 Altova GmbH

- Rellene los campos *Nombre del origen de datos*, *Base de datos, Cuenta de base de datos y Contraseña*.
- Seleccione **fbClient.dll** y haga clic en **Aceptar**.

Descargue el cliente para la versión 2.0.5
http://www.firebirdsql.org/index.php?op=files&id=engine_205

Descargue el cliente para la versión 2.1.2
http://www.firebirdsql.org/index.php?op=files&id=engine_212

Descargar el controlador ODBC para ambas versiones
http://www.firebirdsql.org/index.php?op=files&id=odbc

10.2.2 Tutorial: asignar datos XML a bases de datos

En este tutorial aprenderá a:

- Configurar la asignación de datos XML a BD
- Insertar datos en tablas de BD
- Actualizar
- Eliminar
- Omitir
- Generar valores de salida de la BD

Archivos de ejemplo
En este tutorial se utilizan estos archivos de ejemplo:

- Altova_Hierarchical.xsd • El archivo de esquema jerárquico, que contiene restricciones de identidad
- Altova-cmpy.xml • El archivo de datos de la compañía Altova que ofrece los datos XML
- Altova.mdb • El archivo de base de datos de Altova, que funciona como base de datos de destino

Todos estos archivos de ejemplo están en la carpeta `...\MapForceExamples\Tutorial`.

Nota: en esta sección utilizamos la base de datos **`Altova.mdb`** en multitud de ocasiones para explicar el uso de BD como componentes de destino en MapForce. Recomendamos que haga una copia de seguridad del archivo antes de trabajar con los ejemplos de esta sección. Para conseguir los mismos resultados que en los ejemplos de este tutorial es necesario empezar cada apartado con la base de datos en su estado original.

MapForce puede realizar asignaciones de datos de archivos Access protegidos con contraseña si los añade seleccionando una de las opciones **ODBC** del cuadro de diálogo "Seleccione una base de datos". En el asistente para la conexión puede introducir el nombre de usuario y la contraseña.

Configurar la asignación de datos XML a BD

La asignación de datos entre XML y bases de datos es similar al resto de asignaciones de datos vistas hasta ahora en este manual. **Nota:** si el lenguaje de destino seleccionado es XSLT, XSLT2 o XQuery, las bases de datos no se pueden usar como componentes de origen/destino. XSLT no es compatible con consultas de base de datos.

Objetivo
En este apartado del tutorial aprenderá a añadir un componente de base de datos y a conectarlo a la base de datos que representa.

Comandos utilizados en este apartado

Motor de ejecución integrado: este comando está disponible en la barra de herramientas "Selección del lenguaje" y en el menú **Resultados**. Haga clic en este comando para seleccionar el motor de ejecución integrado como formato de salida.

Insertar archivo o esquema XML: haga clic en este icono para abrir el cuadro de diálogo Abrir de Windows y seleccionar el archivo de su sistema de archivos.

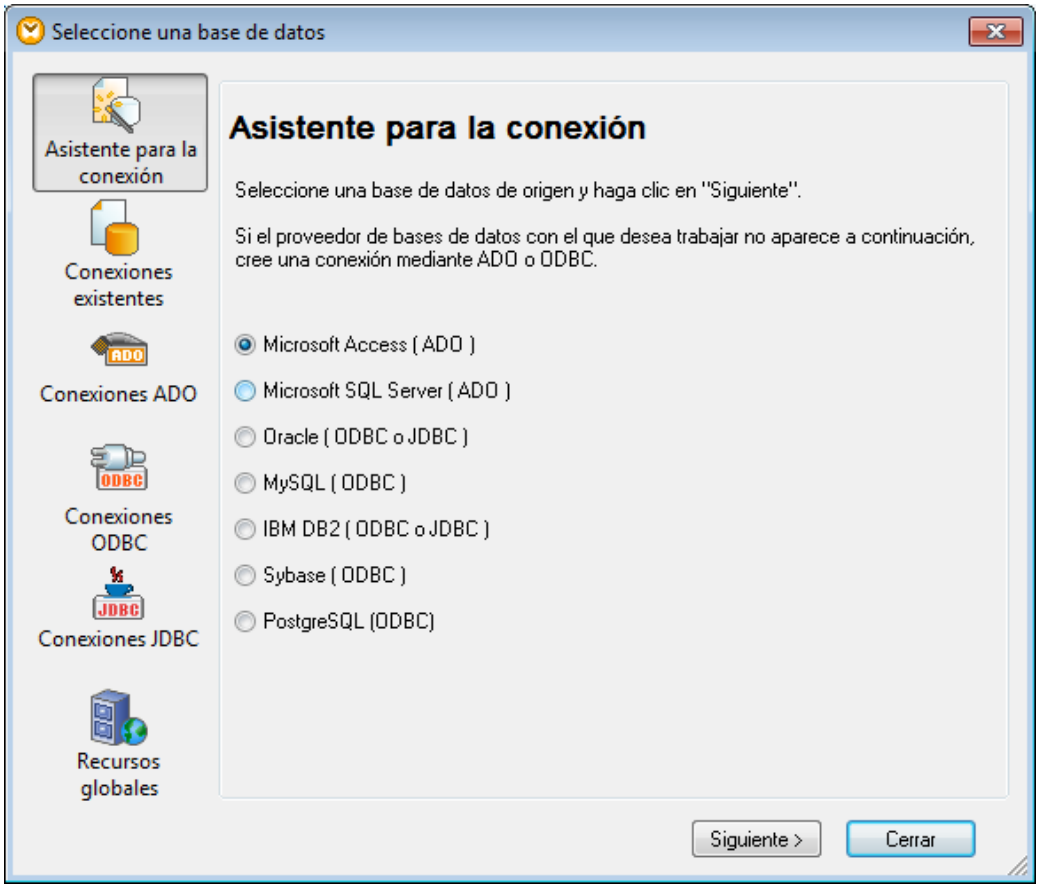
Insertar base de datos: este comando está disponible en la barra de herramientas

 © 2014 Altova GmbH

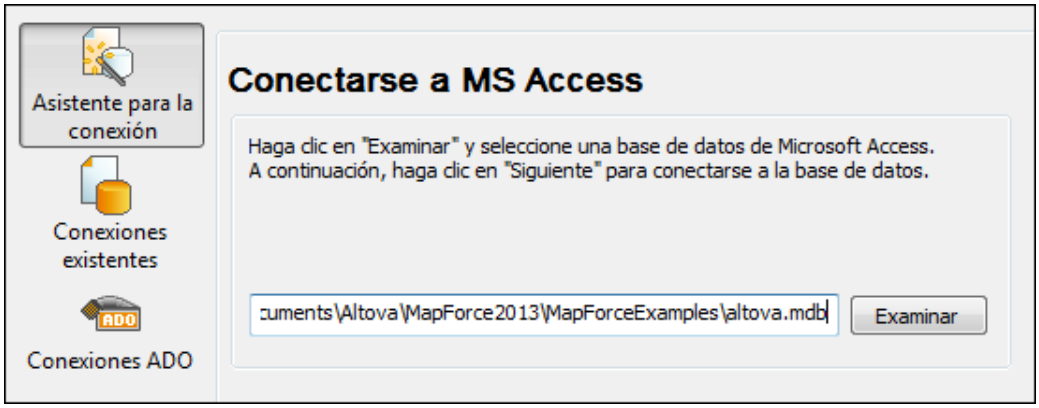
"Insertar componentes" y en el menú **Insertar**. Haga clic en este comando para abrir el cuadro de diálogo "Seleccione una base de datos" donde puede definir una conexión de base de datos y seleccionar las tablas de BD que desea incluir en la asignación de datos de MapForce.

Para configurar el entorno de asignación y conectarse a una base de datos MS Access:

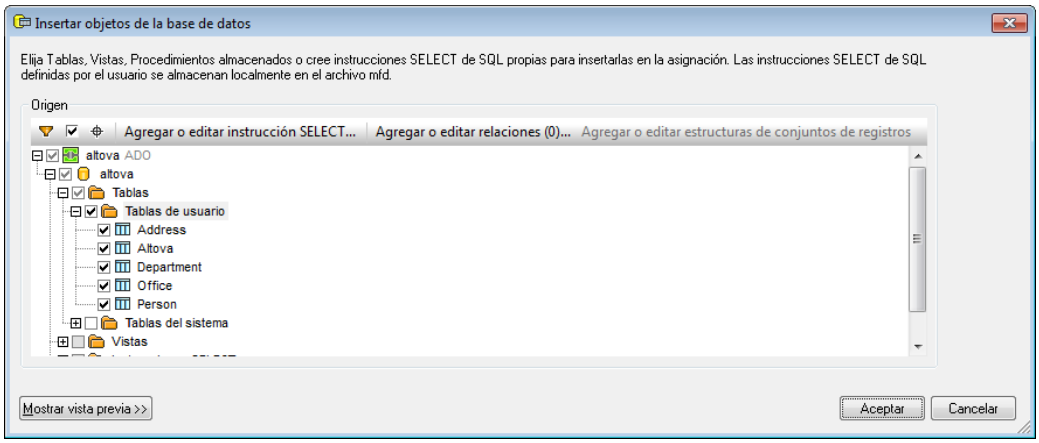
1. En la barra de herramientas seleccione el lenguaje de programación (Java, C#, C++ o BUILTIN) en que desea generar código. Al seleccionar el lenguaje de programación, la biblioteca de funciones correspondiente se carga en la ventana Bibliotecas.
2. Haga clic en el icono **Insertar archivo o esquema XML** y seleccione el archivo `Altova_Hierarchical.xsd` de la carpeta `MapForceExamples`.
3. Aparece un cuadro de diálogo. Haga clic en el botón **Examinar...** y seleccione el archivo `Altova-cmpy.xml` como archivo XML de instancia.
4. En la ventana de asignación haga clic en el elemento **`Altova`** del componente **Altova_Hierarchical** y pulse la tecla * del teclado numérico para ver todos sus elementos. Ajuste el tamaño del componente si es necesario.
5. Haga clic en el icono **Insertar base de datos** y en el cuadro de diálogo "Seleccione una base de datos" haga clic en el icono **Asistente para la conexión**.
6. Seleccione la opción *Microsoft Access (ADO)* y haga clic en **Siguiente**.



7. Haga clic en el botón **Examinar** y seleccione la base de datos `altova.mdb` de la carpeta `...\MapForceExamples\Tutorial`. Después haga clic en **Conectarse**.

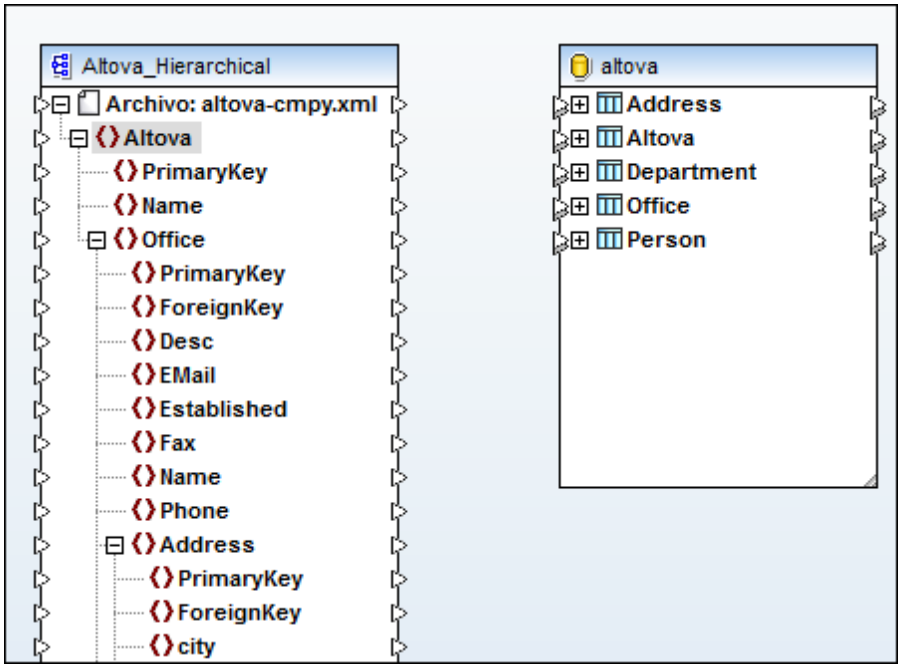


Aparece el cuadro de diálogo "Insertar objetos de la base de datos", donde puede seleccionar qué tablas, vistas y tablas del sistema aparecen en el componente de base de datos. Para ver una vista previa de las tablas seleccionadas haga clic en el botón **Mostrar vista previa** situado en la parte inferior del cuadro de diálogo.

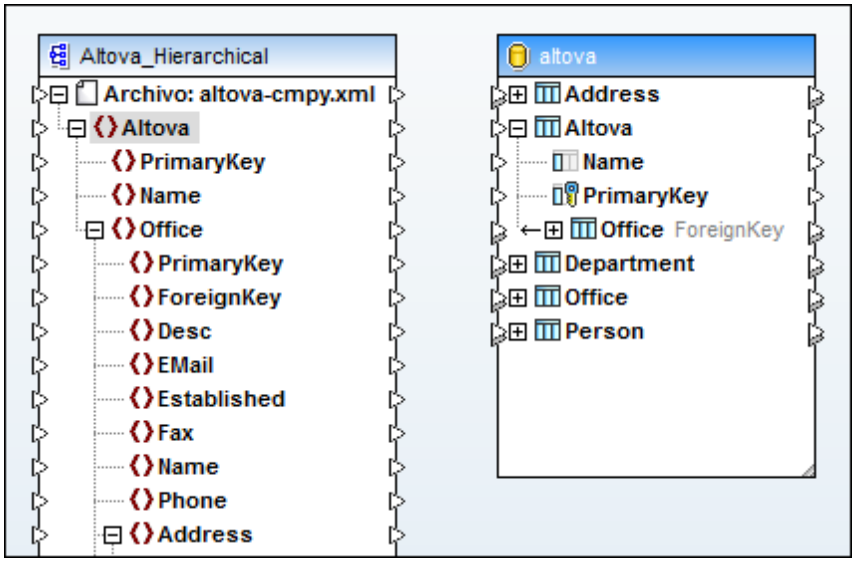


8. Marque la casilla *Tablas de usuario* para seleccionar todas las tablas de usuario y haga clic en **Aceptar** para insertar la base de datos.

 © 2014 Altova GmbH



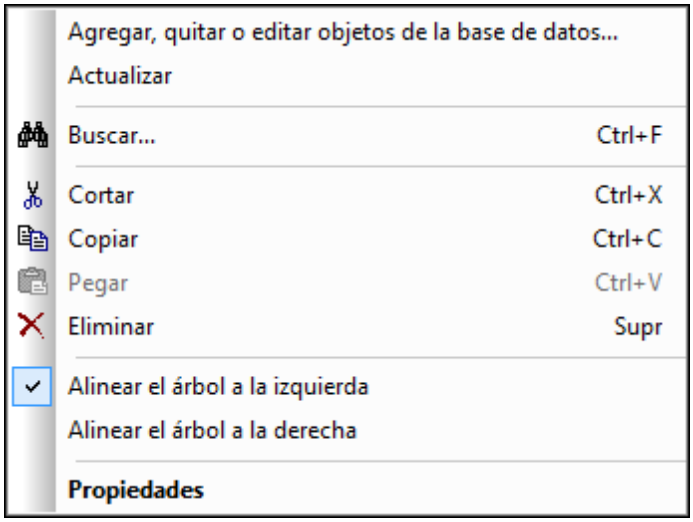
9. En el componente de base de datos, haga clic en el icono de expansión + del elemento **`Altova`** para ver los campos de la tabla.



Cambiar la configuración de la base de datos

Para cambiar la configuración de la base de datos haga clic con el botón secundario en el componente de base de datos y seleccione:

- **Agregar, quitar o editar objetos de la base de datos** para agregar/eliminar tablas del componente.
- **Propiedades** para cambiar el componente de base de datos (haga clic en el botón **Cambiar** y seleccione otra BD en el asistente).



Insertar bases de datos: personalizar la vista previa de las tablas

El cuadro de diálogo "Insertar objetos de la base de datos" incluye una barra de herramientas con comandos para personalizar y buscar elementos en el panel *Origen*.

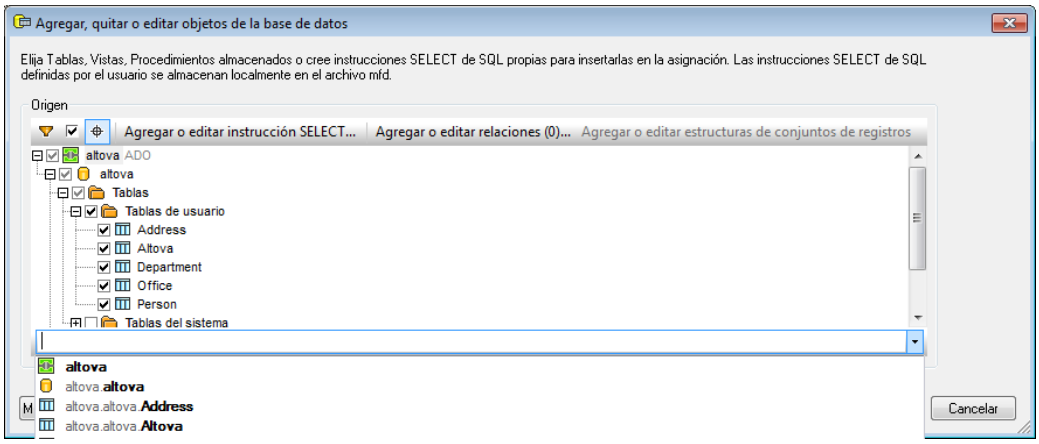
El **localizador de objetos** sirve para buscar elementos en la base de datos.

Con este icono puede **filtrar** las tablas según los caracteres que contengan.

El icono **Mostrar objetos seleccionados** muestra solamente los objetos cuya casilla está marcada.

Buscar elementos de la base de datos con el localizador de objetos:

1. Haga clic en el icono del **localizador de objetos** o pulse **Ctrl+L** para buscar elementos en la base de datos.
   Aparece una lista desplegable con todos los elementos disponibles.



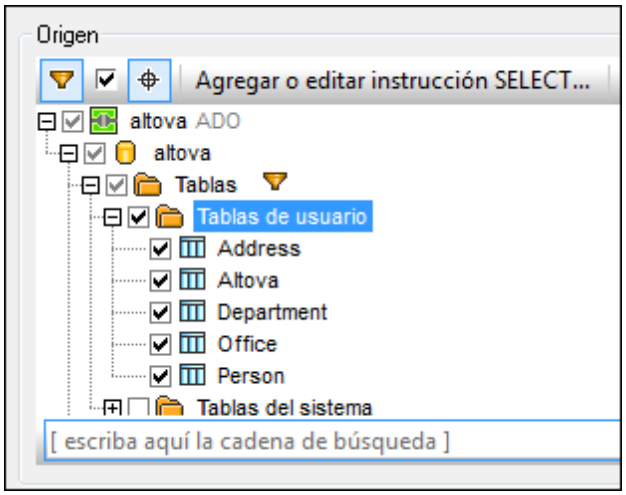
2. Introduzca la cadena de texto que quiere buscar o seleccione un elemento de la lista desplegable.

Filtrar objetos en la ventana de vista previa:

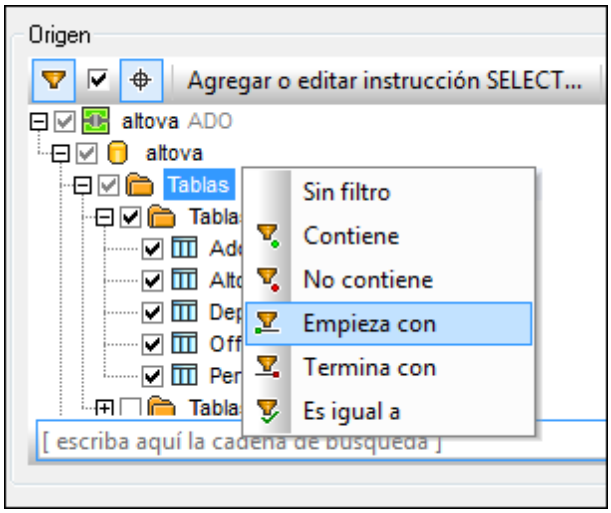
Puede filtrar los esquemas, las tablas y las vistas por su nombre. La función de filtrado no

diferencia entre mayúsculas y minúsculas.

1. Haga clic en el icono **Filtrar contenido de la carpeta** de la barra de herramientas para activar la función de filtro. Junto a las carpetas aparece un icono en forma de embudo.

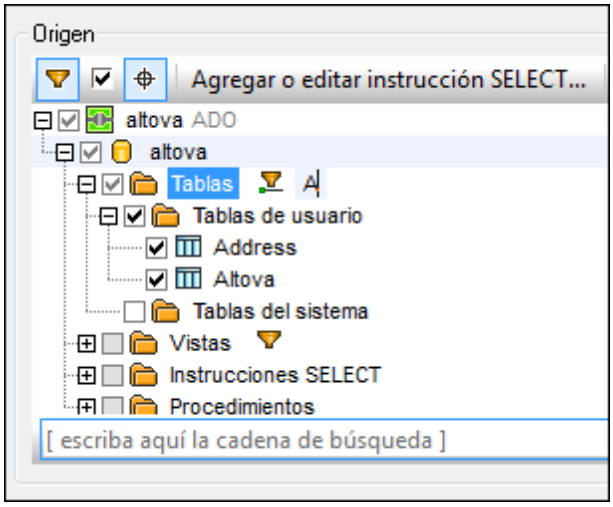


2. Haga clic en el icono situado junto a la carpeta que contiene los objetos que desea filtrar. Aparece un menú donde puede seleccionar un criterio para el filtro.



Tras seleccionar un criterio aparece un campo vacío junto al icono.

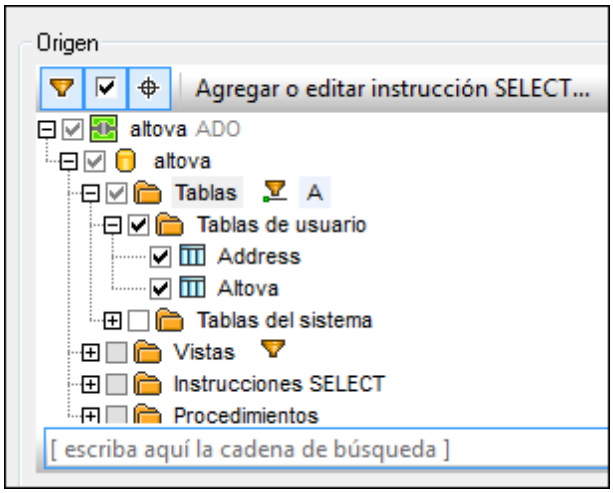
3. Escriba los caracteres que quiere usar para filtrar los objetos (p. ej. la letra A).



A medida que se teclea en este campo, los objetos de la tabla se actualizan automáticamente.

Mostrar solamente los objetos seleccionados

1. Si hace clic en el icono **Mostrar objetos seleccionados** ☑, se muestran solamente las tablas cuya casilla de verificación está marcada.



Componentes y relaciones de tablas

En el componente de base de datos puede identificar fácilmente las relaciones de tabla. El componente de base de datos muestra **cada tabla** de la base de datos como **tabla raíz** y las demás tablas aparecen debajo, en una estructura en forma de árbol.

 © 2014 Altova GmbH

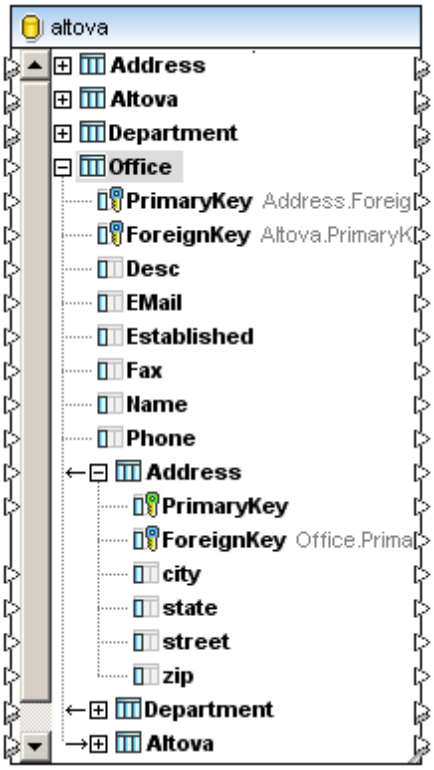
A las tablas cuyo nombre está visible en la imagen anterior las llamamos **tablas raíz** porque están en el nivel superior o raíz de la vista en forma de árbol. Si expandimos estos nombres de tabla podemos ver todas las tablas relacionadas con ellas. Las tablas raíz suelen aparecer por orden alfabético y su presentación no guarda relación con la relación real de las tablas.

Cuando cree consultas/asignaciones de bases de datos que tienen relaciones, incluidas las bases de datos SQL/XML de formato plano, asegúrese de crear asignaciones **entre las tablas** que aparecen **bajo una de las tablas raíz** o de lo contrario se pierden las relaciones de tabla.

La consulta que aparece más abajo muestra la tabla raíz **`Office`** de la base de datos **Altova**. Las flechas que aparecen a la izquierda de los iconos +/-, así como las guías de sangría, muestran las relaciones de tabla.

Empezando pues por la tabla **`Office`** y de arriba a abajo podemos ver que:

- La **flecha apuntando a la izquierda** denota una tabla secundaria de la tabla. Es decir, **`Address`** es una tabla secundaria de la tabla **`Office`**.
- **`Department`** es una tabla secundaria de **`Office`** y una tabla del mismo nivel que **`Address`** (son tablas *hermanas*). Por eso ambas tienen la misma guía de sangría.
- **`Person`** es una tabla secundaria de **`Department`**.
- La **flecha apuntando a la derecha** denota una tabla primaria de la tabla. Es decir, **`Altova`** es la tabla primaria de **`Office`** (o dicho de otra manera, **`Office`** es tabla secundaria de **`Altova`**).



¿Qué tablas raíz debería usar cuando asigne los datos?
Cuando cree asignaciones de datos a tablas de BD, asegúrese de que crea asignaciones usando la tabla correspondiente como tabla de nivel superior o tabla raíz.

Por ejemplo: imagine que solamente quiere insertar o actualizar los datos de la tabla **`Person`**. Para ello debería tomar la tabla **`Person`** como tabla raíz y crear asignaciones entre los elementos de origen y los campos de **`Person`** que desea actualizar.

Si quiere actualizar los datos de **`Department`** y **`Person`** pero conservar las relaciones existentes entre estas dos tablas, use **`Department`** como tabla raíz y cree asignaciones entre los elementos de ambas tablas.

Acción de base de datos: inserción

En este apartado del tutorial explicamos cómo insertar datos en una base de datos. Usaremos los datos de origen de un archivo XML y su esquema XML para generar scripts SQL que llevan a cabo ciertas acciones de base de datos.

Objetivo
En este apartado del tutorial aprenderá a usar MapForce para insertar datos en una base de datos. En concreto aprenderá a:

- Agregar una nueva oficina **`orgchart`** a la tabla **`Altova`** de su base de datos de ejemplo
- Insertar tablas de oficina relacionadas al nuevo registro **`orgchart`**.

Comandos utilizados en este apartado

Insertar archivo o esquema XML: haga clic en este icono para abrir el cuadro de diálogo Abrir de Windows y seleccionar el archivo de su sistema de archivos.

Insertar base de datos: este comando está disponible en la barra de herramientas "Insertar componentes" y en el menú **Insertar**. Haga clic en este comando para abrir el cuadro de diálogo "Seleccione una base de datos" donde puede definir una conexión de base de datos y seleccionar las tablas de BD que desea incluir en la asignación de datos de MapForce.

Acciones de tabla de la base de datos: este comando aparece en el menú **Componente** de un componente de base de datos y aparece en forma de botón junto al nombre de la tabla en el componente. Haga clic en este comando para abrir el cuadro de diálogo "Acciones de tabla de la base de datos", donde puede definir las acciones que se deben ejecutar para cada registro de la BD.

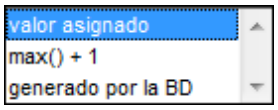
Ejecutar script SQL: este comando está en la barra de herramientas "Vista previa del resultado" y en el menú **Resultados**. Haga clic en este comando para ejecutar el script SQL que se generó en el panel **Resultados**.

Volver a generar archivo de salida: este comando está situado en la barra de herramientas "Vista previa del resultado" y en el menú Resultados. Haga clic en este comando para restablecer el script SQL y así poder ejecutarlo una vez más.

Insertar datos en una tabla de BD

En este primer ejemplo añadiremos una nueva oficina `orgchart` a la tabla **`Altova`**. Por cada registro se creará una clave principal nueva.

Configuración de clave principal
La configuración de clave principal para la acción **Insertar** se define en los cuadros combinados situados a la derecha de cada campo.



Puede elegir una de estas tres opciones:

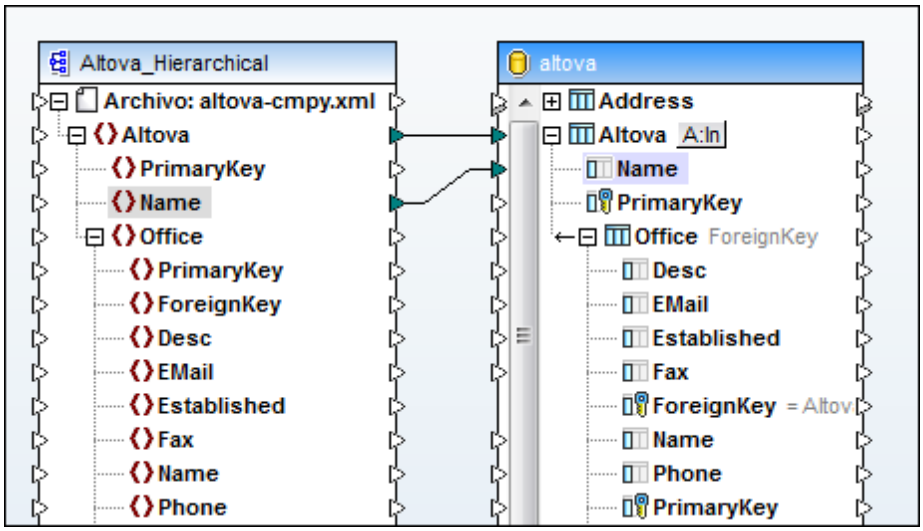
- **valor asignado**: permite asignar los datos de origen al campo de la BD directamente y es la opción predeterminada para todos los campos de la BD. También se puede usar un procedimiento almacenado para suministrar un valor de clave definiendo una relación (consulte el apartado Usar procedimientos almacenados para generar claves principales).
- **max() + 1**: genera los valores de clave a partir de las claves ya existentes en la BD, de modo que los registros nuevos se anexan automáticamente a los registros ya existentes.
- **generado por la BD**: la base de datos utiliza la función `IDENTITY` para generar los valores de clave.

Para insertar valores en una tabla de la BD desde un archivo XML:

1. Cree un archivo de asignación nuevo e inserte un componente de esquema basado en `Altova_Hierarchical.xsd`, seleccione `Altova-cmpy.xml` como archivo XML de instancia del componente de esquema e inserte un componente de base de datos basado en `altova.mdb` (siguiendo los pasos del apartado Configurar la asignación de datos XML a BD).

2. Ahora cree estas asignaciones:
   - Entre **`Altova`** y **`Altova`**
   - Entre **`Nombre`** y **`Nombre`**

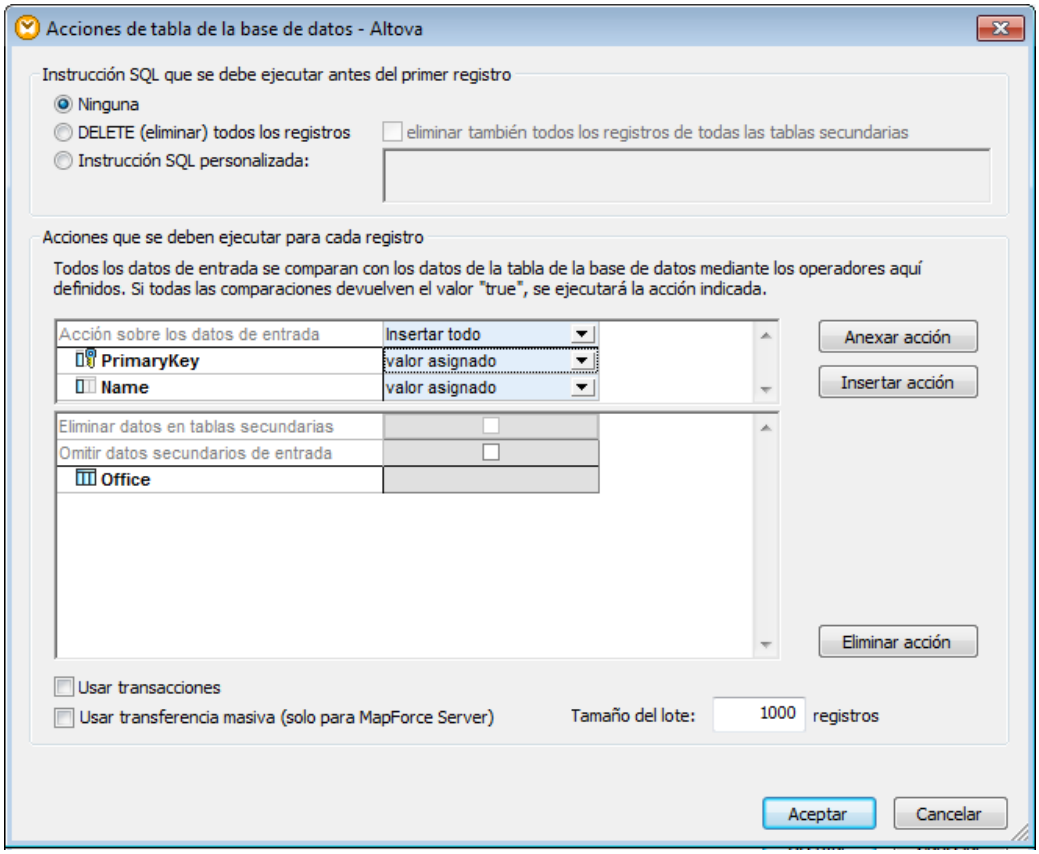
Nota: si se crean asignaciones automáticas entre los elementos secundarios, significa que está activa la opción **Conexión automática de secundarios**. Seleccione la opción de menú **Edición | Deshacer** y después haga clic en **Conexión | Conectar automáticamente los secundarios equivalentes** para deshabilitar esta opción.



Observe que a la derecha del nombre de la tabla **`Altova`** aparece un icono (**`A:In`**) que indica la **acción de tabla** definida para la tabla en ese momento (es decir, **`Acción:Insertar`**). Al hacer clic en este icono aparece el cuadro de diálogo "Acciones de tabla de la base de datos", donde puede definir las acciones de tabla.

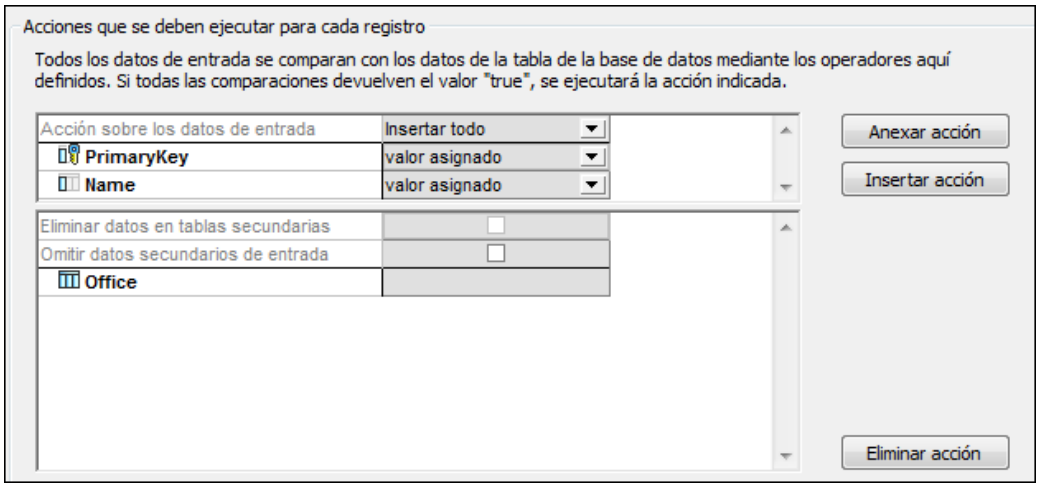
3. Haga clic en el icono A:In de la tabla **`Altova`**.
   En el cuadro de diálogo "Acciones de tabla de la base de datos", en el grupo *Acciones que se deben ejecutar para cada registro* aparece en este momento una sola acción de tabla: *Insertar todo*.

 © 2014 Altova GmbH



La acción de tabla *Insertar todo* insertará en la base de datos todos los registros con todos los **campos asignados** de la tabla actual. El siguiente paso es definir cómo se genera el campo **`PrimaryKey`** nuevo para esta acción.

4. Haga clic en el cuadro combinado del campo **`PrimaryKey`** y seleccione la opción **`max() + 1`**.



5. Haga clic en **Aceptar** para confirmar.
6. Abra el panel **Resultados** para ver el script SQL que genera esta asignación.

```
/*
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:\Documents\Altova\MapForce2012\MapForceExamples\CustomersTarget.mdb;
Provider=Microsoft.Jet.OLEDB.4.0
*/

SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey]
FROM [Altova]
-->>> %PrimaryKey1%

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', '%PrimaryKey1%')
```

El script ejecuta la asignación de datos a la base de datos de destino, teniendo en cuenta las acciones de tabla definidas. Los scripts SQL se pueden volver a ejecutar desde el panel **Resultados**. Para ello haga clic en el icono **Volver a generar archivo de salida** .

7. Haga clic en el icono **Ejecutar script SQL** de la barra de herramientas para ejecutar el script e insertar los datos de la tabla en la base de datos. Si el script se ejecuta correctamente, aparece un mensaje de confirmación en la ventana Mensajes.
8. Abra la base de datos `Altova.mdb` en DatabaseSpy o en MS Access para ver los resultados.

Altova : Table

	PrimaryKey	Name
+	1	Organization Chart
+	2	Microtech OrgChart

En la tabla **`Altova`** se insertó el nuevo registro `Microtech OrgChart` con la nueva `PrimaryKey 2`. Los datos de este registro proceden del archivo XML de entrada.

9. Vuelva a MapForce.

   Observe que en la ventana **Resultados** aparece un mensaje que confirma lo que ocurrió cuando se procesó el script SQL.

```
SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey]
FROM [Altova]
-->>> OK. Una fila o más.

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', 2)
-->>> OK. 1 fila(s).
```

Nota: si quiere insertar grandes cantidades de filas en una BD más rápido, puede activar la opción *Usar transferencia masiva*.

Insertar tablas y tablas secundarias relacionadas

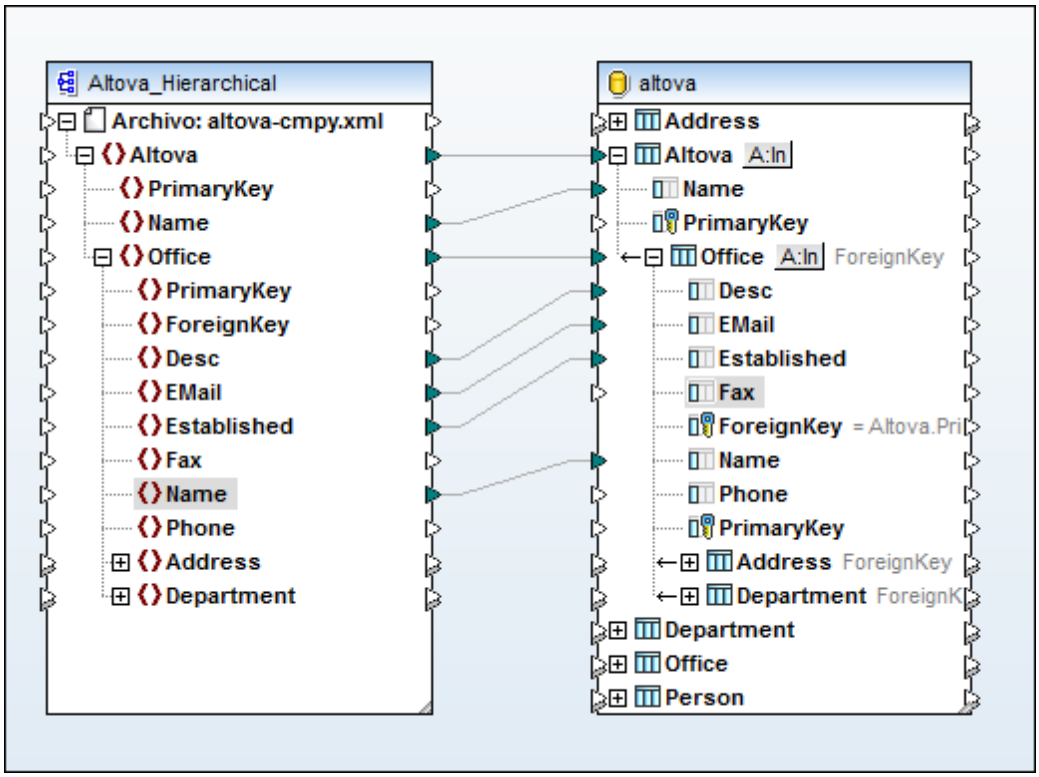
En este ejemplo usamos como base el ejemplo del apartado anterior y lo ampliamos insertando en la tabla primaria **`Altova`** las tablas secundarias de **`Office`**.

Las relaciones de tabla se generan automáticamente si se crean asignaciones entre las tablas secundarias de una tabla raíz. En este caso, se crean asignaciones entre los campos **`Office`** que aparecen directamente bajo la tabla primaria o raíz **`Altova`**.

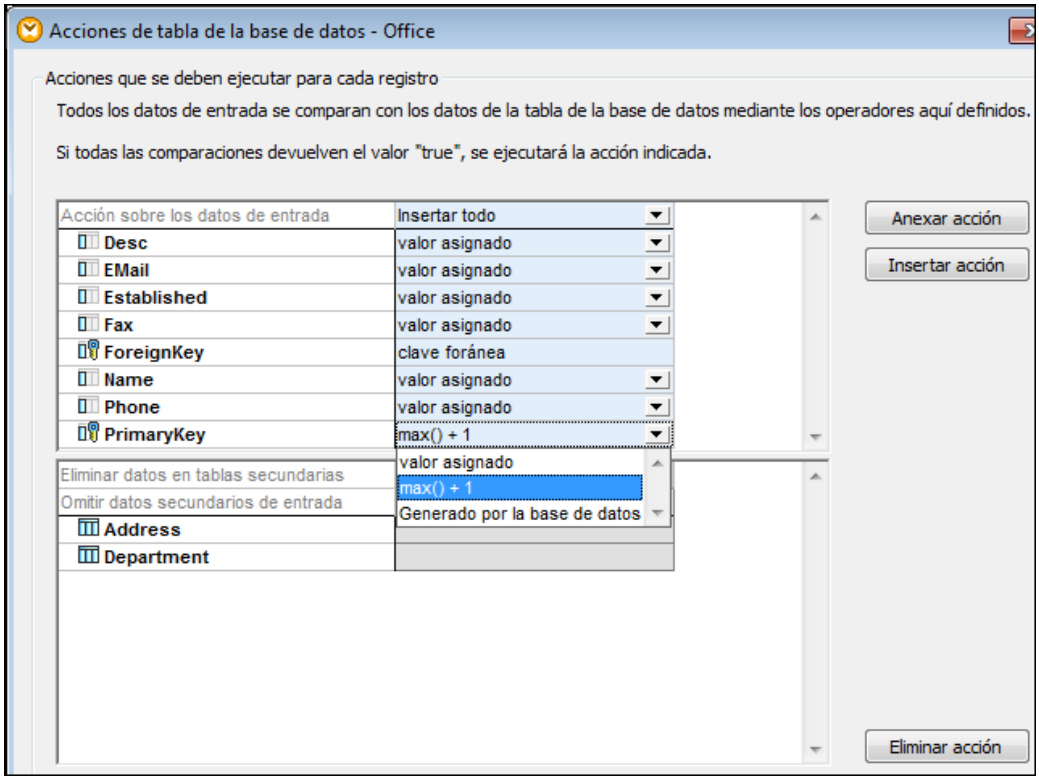
Para insertar tablas y tablas secundarias relacionadas:

1. Cree un archivo de asignación nuevo, inserte un componente de esquema basado en `Altova_Hierarchical.xsd`, seleccione el archivo XML de instancia `Altova-cmpy.xml` e inserte un componente de base de datos basado en `altova.mdb` (siguiendo los pasos del apartado Configurar la asignación de datos XML a BD).
2. Cree asignaciones entre los elementos **`Altova`** y **`Name`** (siguiendo los pasos del apartado anterior).
3. Haga clic en el icono + de los elementos **`Office`** de ambos componentes para ver los campos de tabla de **`Office`**.
4. Cree estas asignaciones entre los dos componentes:

   Entre **`Office`** y **`Office`**
   Entre **`Desc`** y **`Desc`**
   Entre **`Email`** y **`Email`**
   Entre **`Established`** y **`Established`**
   Entre **`Name`** y **`Name`**



5. Haga clic en el icono A:In de la tabla **`Office`**.
   Aparece el cuadro de diálogo "Acciones de tabla de la base de datos".
6. Haga clic en el cuadro combinado de **`PrimaryKey`** y seleccione la opción **`max()+1`**. Haga clic en **Aceptar** para confirmar.



7. Abra el panel **Resultados** para ver el script SQL resultante.

```
/*
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:\Documents\Altova\MapForce2013\MapForceExamples\altova.mdb;Provider=Microsoft.Jet.OLEDB.4.0
*/

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', 1)

SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Office]
-->>> %PrimaryKey1%

INSERT INTO [Office] ([ForeignKey], [Desc], [EMail], [Established], [Name], [PrimaryKey]) VALUES (1, 'Microtechnology products are
currently the bleeding edge of computer technology.', 'office@microtech.com', '1992-04-01', 'Microtech, Inc.', '%PrimaryKey1%')

SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Office]
-->>> %PrimaryKey2%

INSERT INTO [Office] ([ForeignKey], [Desc], [EMail], [Established], [Name], [PrimaryKey]) VALUES (1, 'Microtech established its new
office on Feb 30', 'nextoffice@microtech.com', '2001-03-01', 'Microtech Partners, Inc.', '%PrimaryKey2%')
```

8. Haga clic en el icono **Ejecutar script SQL** para ejecutar el script e insertar las tablas nuevas.
9. Abra la base de datos en DatabaseSpy o MS Access y haga doble clic en la tabla **`Altova`** para ver el resultado.

 © 2014 Altova GmbH

Altova : Table

PrimaryKey	Name
1	Organization Chart
2	Microtech OrgChart

PrimaryKey	Desc	EMail	Established
1	The company was es	office@nanonul	1992-04-01
2	On March 1st, 2000,	nextoffice@nan	2001-03-01

PrimaryKey	Desc	EMail	Established
3	Microtechnology proc	office@microte	1992-04-01
4	Microtech establishe	nextoffice@mic	2001-03-01

A la tabla **Microtech OrgChart** se añadieron dos oficinas nuevas.

10. Haga doble clic en la tabla **Office** para ver el resultado más detalladamente.

Office : Table

PrimaryKey	ForeignKey	Desc	EMail
1	1	The company was establis	office@nanonul
2	1	On March 1st, 2000, Nanoi	nextoffice@nan
3	2	Microtechnology products	office@microte
4	2	Microtech established its r	nextoffice@mic

Las nuevas oficinas añadidas tienen la clave principal **3** y **4** respectivamente. Ambas tablas están relacionadas con la tabla **Altova** por medio de la clave foránea **2**, que hace referencia al registro **Microtech OrgChart**.

```
SELECT IIF(MAX([Altova].[PrimaryKey]) IS NULL,0,MAX([Altova].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Altova]
 -->>> OK. Una fila o más.

INSERT INTO [Altova] ([Name], [PrimaryKey]) VALUES ('Microtech OrgChart', 2)
-->>> OK. 1 fila(s).

SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Office]
 -->>> OK. Una fila o más.

INSERT INTO [Office] ([ForeignKey], [Desc], [EMail], [Established], [Name], [PrimaryKey]) VALUES (2,
'Microtechnology products are currently the bleeding edge of computer technology.', 'office@microtech.com',
'1992-04-01', 'Microtech, Inc.', 3)
-->>> OK. 1 fila(s).

SELECT IIF(MAX([Office].[PrimaryKey]) IS NULL,0,MAX([Office].[PrimaryKey]))+1 AS [PrimaryKey] FROM [Office]
 -->>> OK. Una fila o más.

INSERT INTO [Office] ([ForeignKey], [Desc], [EMail], [Established], [Name], [PrimaryKey]) VALUES (2, 'Microtech
established its new office on Feb 30', 'nextoffice@microtech.com', '2001-03-01', 'Microtech Partners, Inc.', 4)
-->>> OK. 1 fila(s).
```

Acción de base de datos: actualización

En este apartado del tutorial explicamos cómo actualizar los datos de una BD desde MapForce.

Objetivo

En este apartado aprenderá a usar MapForce para actualizar los datos de una tabla de la BD. En concreto aprenderá a:

- Actualizar los campos de una tabla concreta
- Agregar registros nuevos a una tabla
- Usar la condición **Actualizar si...**

Comandos utilizados en este apartado

Insertar archivo o esquema XML: haga clic en este icono para abrir el cuadro de diálogo Abrir de Windows y seleccionar el archivo de su sistema de archivos.

Insertar base de datos: este comando está disponible en la barra de herramientas "Insertar componentes" y en el menú **Insertar**. Haga clic en este comando para abrir el cuadro de diálogo "Seleccione una base de datos" donde puede definir una conexión de base de datos y seleccionar las tablas de BD que desea incluir en la asignación de datos de MapForce.

Conectar automáticamente los secundarios equivalentes: haga clic en este icono para activar/desactivar la conexión automática de nodos secundarios equivalentes.

Acciones de tabla de la base de datos: este comando aparece en el menú **Componente** de un componente de base de datos y aparece en forma de botón junto al nombre de la tabla en el componente. Haga clic en este comando para abrir el cuadro de diálogo "Acciones de tabla de la base de datos", donde puede definir las acciones que se deben ejecutar para cada registro de la BD.

Ejecutar script SQL: este comando está en la barra de herramientas "Vista previa del resultado" y en el menú **Resultados**. Haga clic en este comando para ejecutar el script SQL que se generó en el panel **Resultados**.

Archivos utilizados en este apartado

- `Altova_Hierarchical.xsd`
- `altova-cmpy.xml`
- `altova-cmpy-extra.xml`
- `altova.mdb`

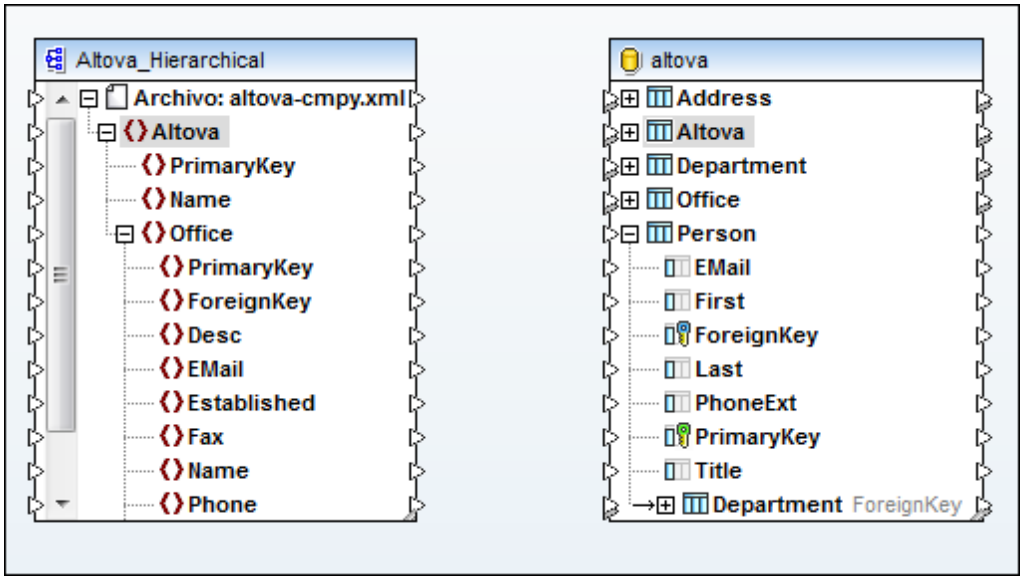
Actualizar campos de BD

En este primer ejemplo actualizaremos los registros ya existentes de la tabla **`Person`**. Para ello crearemos asignaciones entre los datos de origen XML y la tabla raíz **`Person`**.

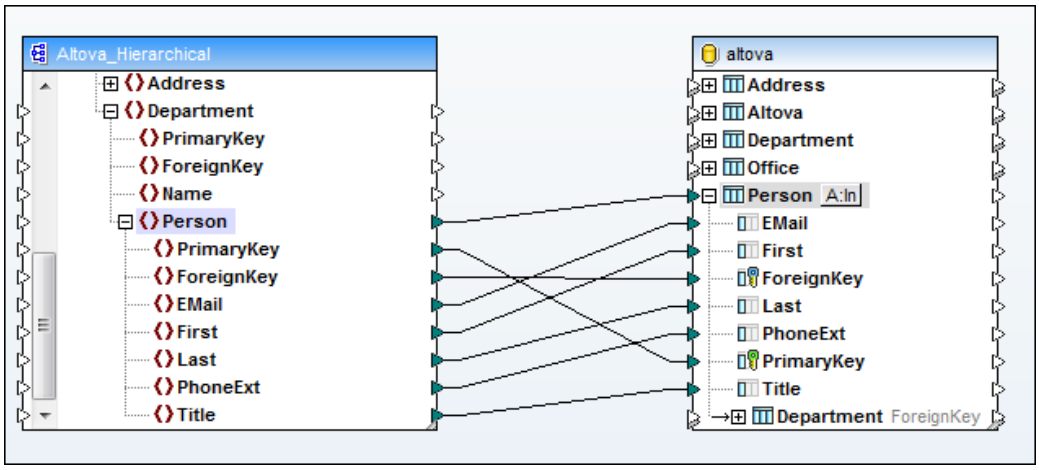
Para actualizar la tabla `Person`:

1. Inserte un componente de esquema basado en `Altova_Hierarchical.xsd` y asígnele el archivo de entrada `altova-cmpy.xml`.
2. Inserte un componente de base de datos basado en `altova.mdb`.

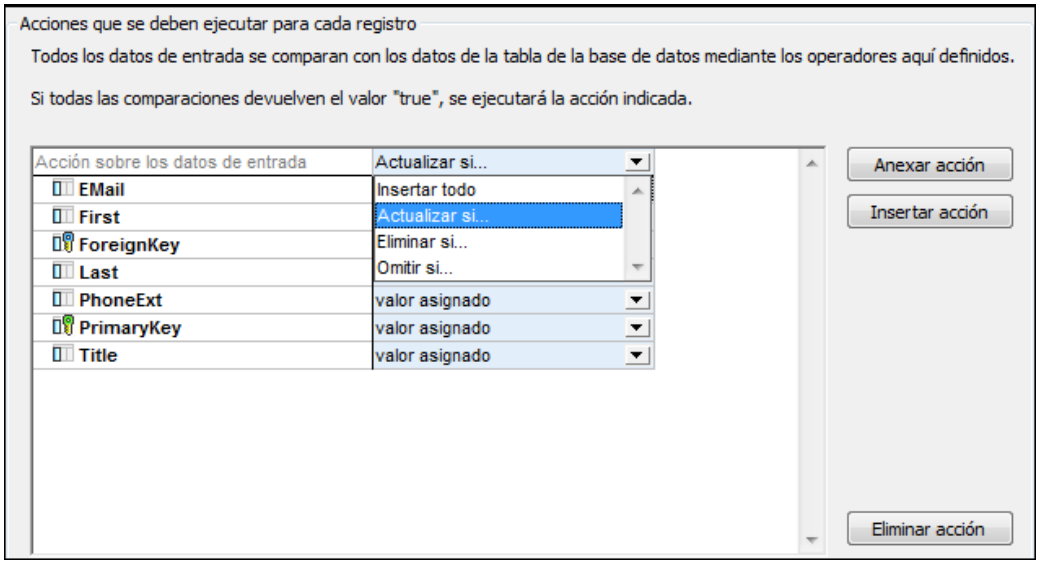
 © 2014 Altova GmbH



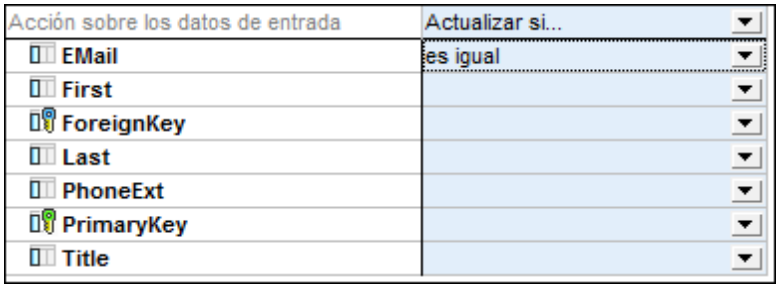
3. Active la conexión automática de secundarios equivalentes (con el icono).
4. Expanda los elementos **`Office`** y **`Department`** del componente de esquema. Haga clic en el elemento **`Person`** y conéctelo con el elemento **`Person`** del componente de base de datos. Asegúrese de que lo conecta con la tabla raíz **`Person`**.
   Se crean asignaciones automáticas entre los elementos secundarios equivalentes.



5. Haga clic en el icono A:In de Person para abrir el cuadro de diálogo "Acciones de tabla de la base de datos".
6. En el grupo *Acciones que se deben ejecutar para cada registro*, haga clic en el primer cuadro combinado y seleccione la opción **Actualizar si...**.

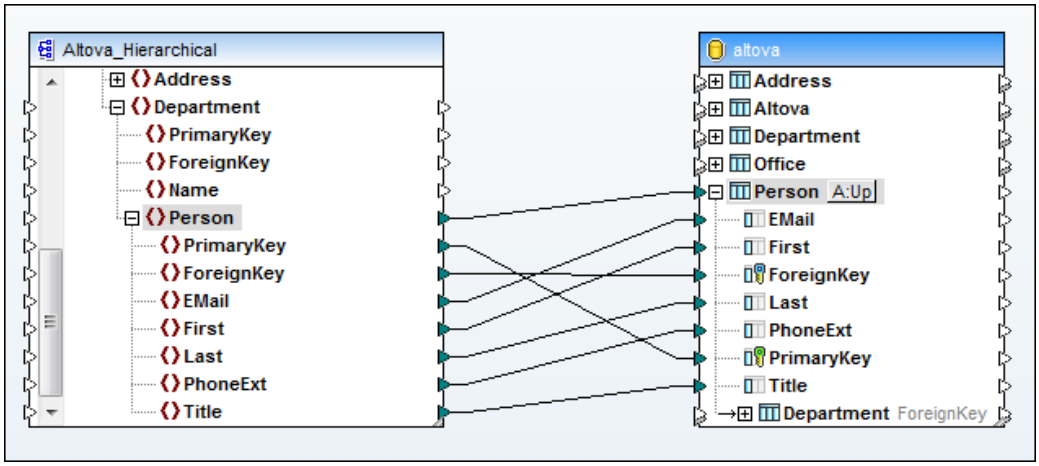


7. Haga clic en el cuadro combinado de la entrada **`PrimaryKey`**, seleccione la opción **`es igual`** y haga clic en **Aceptar** para confirmar.



Las claves principales (**`PrimaryKey`**) de origen y destino se comparan usando el operador `igual que`. Si las claves son idénticas, entonces se actualizan todos los campos asignados de las tablas **`Person`**.

El icono de acción de tabla **A:In** se convierte ahora en el icono **A:Up** (*Acción: Actualizar*) porque se seleccionó la acción de tabla **Actualizar si...**.



8. Abra el panel **Resultados** para ver el script SQL generado por la asignación. El script ejecuta la asignación en la BD de destino, teniendo en cuenta las acciones de tabla definidas.

 © 2014 Altova GmbH

9. Haga clic en el icono **Ejecutar script SQL** de la barra de herramientas para ejecutar el script y actualizar los registros de la BD. Si el script se ejecuta correctamente, aparece un mensaje de confirmación en la ventana Mensajes.
10. Abra la base de datos `altova.mdb` en DatabaseSpy o MS Access y haga doble clic en la tabla **`Person`** para ver el resultado.
    Observe que se actualizaron todos los registros de **`Person`** de la base de datos.

Person : Table

PrimaryKey	ForeignKey	EMail	First	Last	PhoneExt
1	1	A.Aldrich@mici	Albert	Aldrich	582
2	1	b.bander@micr	Bert	Bander	471
3	1	c.clovis@micro	Clive	Clovis	963
4	2	d.Durnell@micr	Dave	Durnelll	621
5	2	e.ellas@microt	Eve	Ellas	753
6	3	f.fortunas@mici	Fred	Fortunas	951
7	3	g.gundall@micr	Gerry	Gundall	654
8	3	h.hardy@micro	Harry	Hardy	852
9	3	i.idilko@microte	Ingrid	Idilko	951
10	3	j.judy@microte	June	Judy	753
11	3	k.krove@microt	Karl	Krove	334

Actualizar y añadir registros nuevos

En este ejemplo, algo más complicado, actualizaremos los registros de las tablas **`Department`** y **`Person`** y además añadiremos los registros de **`Person`** del archivo XML de entrada que falten en la BD. La tabla raíz utilizada en este ejemplo es la tabla **`Department`**.

Archivos utilizados en este ejemplo

- `Altova_Hierarchical.xsd`
- `altova-cmpy-`**`extra`**`.xml` (el archivo XML de instancia de `Altova_hierarchical.xsd`)
- `altova.mdb`

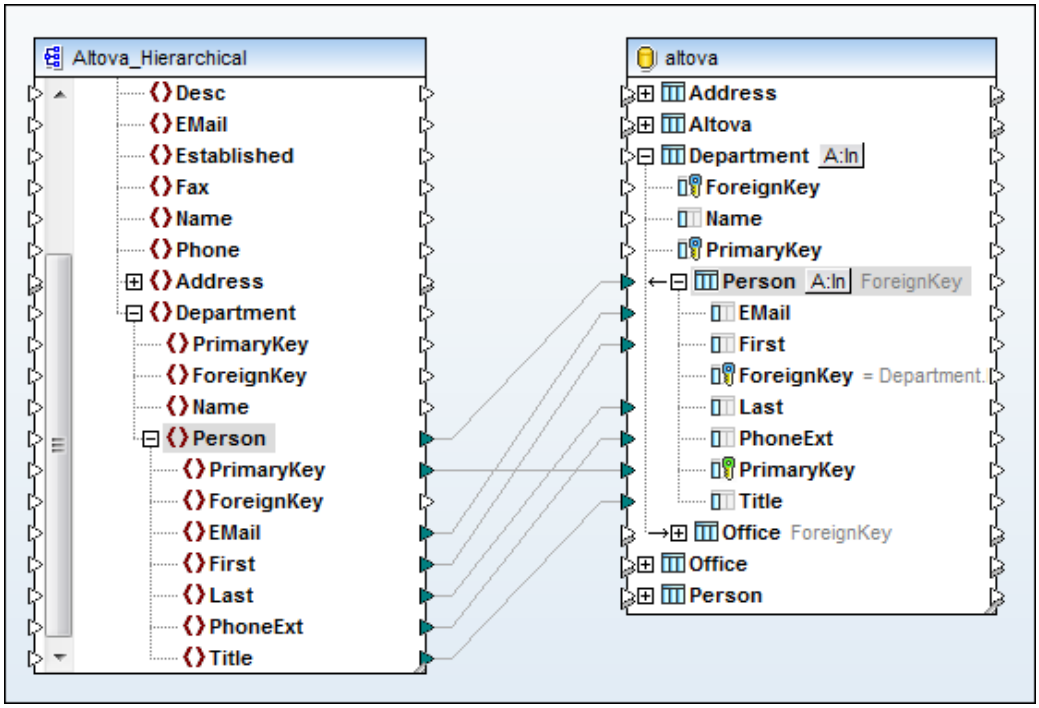
Objetivo

- **actualizar** los registros **`Name`** de la tabla **`Department`**
- **actualizar** los registros ya existentes de la tabla **`Person`**
- **insertar** los registros que falten en la tabla **`Person`**

Para asignar datos a las tablas `Department` y `Person`:

1. Inserte un componente de esquema basado en `Altova_Hierarchical.xsd` (siga las instrucciones de este apartado) y asígnele el archivo XML de instancia `altova-cmpy-extra.xml`.
2. Inserte un componente de base de datos basado en `altova.mdb` (siga las instrucciones de este apartado).
3. Active la conexión automática de secundarios equivalentes (haga clic en el icono).
4. En el componente de base de datos expanda el elemento **`Department`** y su elemento secundario **`Person`**.
5. En el componente de esquema expanda el elemento **`Office`**, después **`Department`** y después **`Person`**. Haga clic en el elemento **`Person`** y conéctelo con la tabla **`Person`** de la

BD. Asegúrese de conectarlo a la tabla **Person** que está anidada dentro de la tabla **Department**.
Se crean conexiones automáticas entre los secundarios equivalentes.

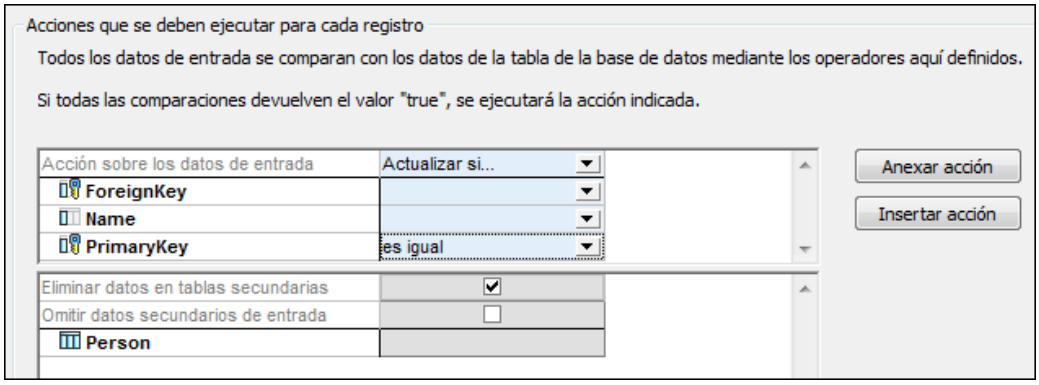


Definir las acciones de tabla

Las claves principales (`PrimaryKey`) de origen y destino de las tablas se comparan con el operador `igual que`. Si las claves son idénticas, entonces se actualizan los campos asignados de las tablas **Department** y **Person**. Si las claves no son idénticas, entonces se procesa la siguiente acción de tabla (es decir, la acción **Insertar restantes**).

Para definir la acción de tabla para la tabla Department:

1. Haga clic en el icono A:In de la tabla **Department** para abrir el cuadro de diálogo "Acciones de tabla de la base de datos".
2. En el grupo *Acciones de tabla que se deben ejecutar...* haga clic en el primer cuadro combinado y seleccione la opción **Actualizar si...**.
3. Haga clic en el cuadro combinado de la entrada `PrimaryKey` y seleccione la opción **es igual**. Para confirmar haga clic en **Aceptar**.

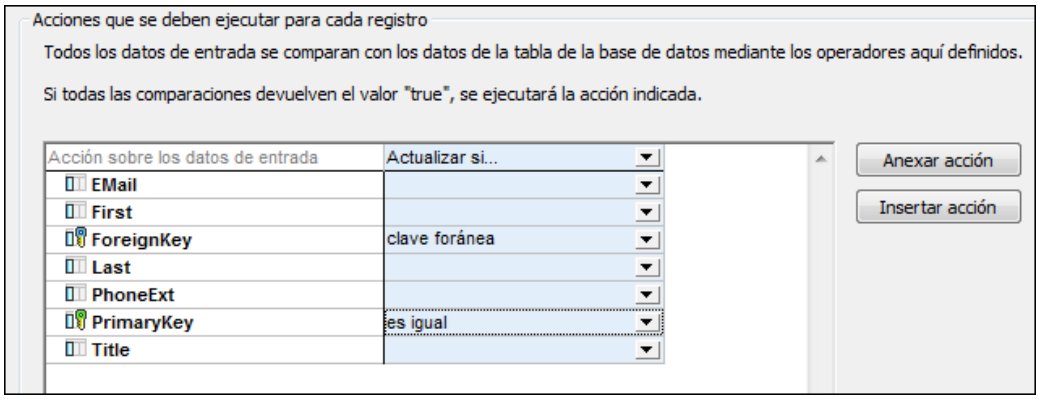


 © 2014 Altova GmbH

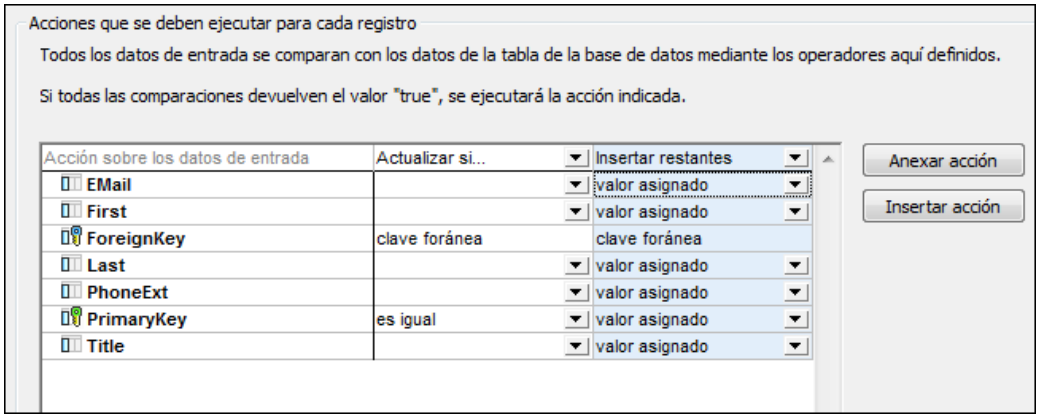
Las claves principales (`PrimaryKey`) de origen y destino se comparan usando el operador `igual que`. Si las claves son idénticas, se actualizarán todos los campos asignados de **`Department`**.

Para definir la acción de tabla para la tabla Person:

1. Haga clic en el icono A:In de la tabla **`Person`** para abrir el cuadro de diálogo "Acciones de tabla de la base de datos".
2. Haga clic en el primer cuadro combinado del grupo *Acciones de tabla que se deben ejecutar...* y seleccione la opción **Actualizar si...**.
3. Haga clic en el cuadro combinado de la entrada `PrimaryKey` y seleccione la opción **es igual**.



4. Haga clic en el botón **Anexar acción** para anexar una columna de acción de tabla nueva.
   La acción de tabla **Insertar restantes** se inserta automáticamente en el panel. Si no se puede procesar la acción **Actualizar si...**, entonces se procesa esta otra acción de tabla.



5. Haga clic en **Aceptar** para terminar de definir las acciones de tabla. Observe cómo el icono de acción de tabla ahora es **A:Up,In** (*Acción: actualizar, insertar*).

6. Abra el panel **Resultados** para ver el script SQL resultante.

```
UPDATE [Department] SET [ForeignKey] = 1, [Name] = 'Admin' WHERE ([Department].[PrimaryKey]=1)

SELECT [ForeignKey], [PrimaryKey] FROM [Department] WHERE ([PrimaryKey]=1)
-->>> %ForeignKey1%

-->>> %PrimaryKey1%

DELETE FROM [Person] WHERE EXISTS(SELECT * FROM [Department] WHERE [Department].[PrimaryKey] =
[Person].[ForeignKey] AND ([Department].[PrimaryKey]='%PrimaryKey1%'))

UPDATE [Person] SET [EMail] = 'A.Aldrich@microtech.com', [First] = 'Albert', [Last] = 'Aldrich', [PhoneExt] = 582
, [Title] = 'Manager' WHERE ([Person].[ForeignKey] = '%PrimaryKey1%') AND ([Person].[PrimaryKey]=1)
```

Nota: el script ejecuta la asignación a la base de datos de destino teniendo en cuenta las acciones de tabla definidas.

Secuencia de procesamiento

La condición **Actualizar si...** comprueba si las claves principales de los elementos de origen y destino son idénticas.

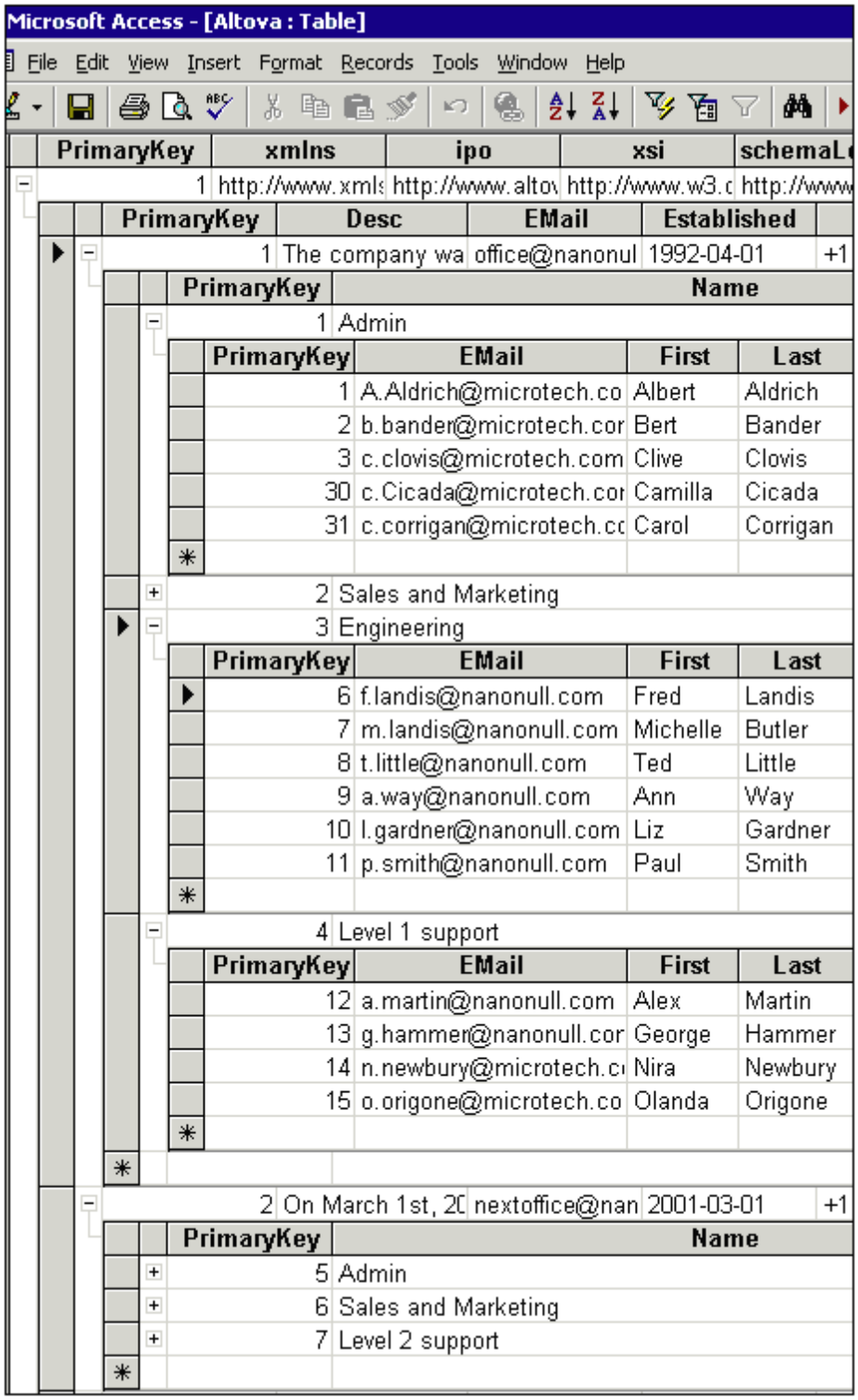
Tabla **`Department`**:

- Si la **condición es verdadera**, se actualizan todos los registros de **`Department`** cuya clave es idéntica a la clave principal de origen. Si se encuentran registros en la BD que no tienen elementos equivalentes en el archivo de origen, estos registros se conservan tal y como están (p. ej. la tabla **`Engineering`**).
- Si la **condición es falsa** (es decir, si la clave principal de origen no coincide con la clave principal de la BD de destino), no se actualiza ningún registro de la tabla **`Department`**.

Tabla **`Person`**:

- Si la **condición es verdadera**, se actualizan todos los registros de **`Person`** cuya clave es idéntica a la clave principal de origen. Si se encuentran registros en la BD que no tienen elementos equivalentes en el archivo de origen, estos registros se conservan tal y como están.
- Si la **condición es falsa** (es decir, si la clave principal de origen no coincide con la clave principal de la BD de destino), MapForce pasa a la siguiente columna de acción de tabla: la acción **Insertar restantes**. Esta acción inserta los registros de **`Person`** que la tabla **`Person`** de la BD no tiene. En este ejemplo, se añaden dos registros nuevos al departamento **`Admin`** cuyas claves principales son 30 y 31 respectivamente.

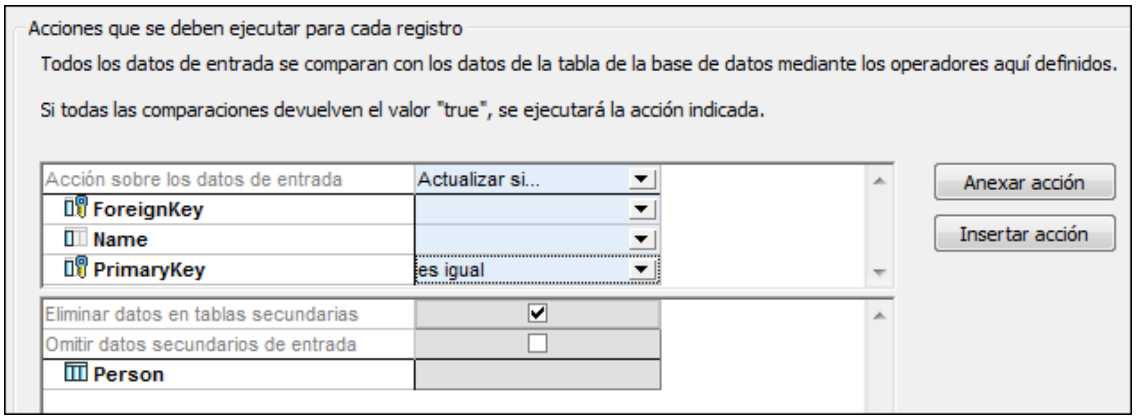
 © 2014 Altova GmbH



Actualizar y eliminar datos secundarios

En este apartado describimos lo que ocurre al usar la condición **Actualizar si...** en una tabla primaria cuyas tablas secundarias tienen definida otra acción de tabla (es decir, **Insertar todo**, **Actualizar si...**, **Eliminar si...**). En todos estos ejemplos, excepto uno, está activa la opción

Eliminar datos en tablas secundarias. Para estos ejemplos puede usar la asignación del apartado anterior.



Archivos utilizados en este ejemplo:

- `Altova_hierarchical.xsd`
- `Altova-cmpy-extra.xml`
- `Altova.mdb`

La **tabla primaria** (**`Department`**) tiene definidas estas acciones de tabla:

Acción sobre los datos de entrada	**Actualizar si...**
`PrimaryKey`	`es igual`
`ForeignKey`	
`Name`	
Eliminar datos en tablas secundarias	☑ *
Omitir datos secundarios de entrada	☐

* Esta opción siempre debe estar desactivada para la acción **Eliminar si...** (*ver la tabla que aparece más abajo*).

El resultado de la asignación depende de qué acción seleccione para la tabla secundaria (**`Person`**):

Insertar todo...	**Actualizar si...**	**Eliminar si...**	
Actualiza los datos de la tabla primaria (los registros de `Department`).			
Elimina los datos secundarios de las tablas que cumplen la condición **Actualizar si...** (los registros de **`Person`**).		Si **está activa** la opción *Eliminar datos en tablas secundarias,* la asignación elimina los datos secundarios (los registros de **`Person`**) de todos los **`Department`**. Todos los registros de **`Person`** se eliminan en cada **`Department`** cuya clave principal coincida con la del XML de origen (es decir, se eliminan incluso los registros de **`Person`** de la BD	Si **no está activa** la opción *Eliminar datos en tablas secundarias*, la asignación elimina los datos secundarios de las tablas que cumplen la condición **Actualizar si...** (registros de **`Person`**).

 © 2014 Altova GmbH

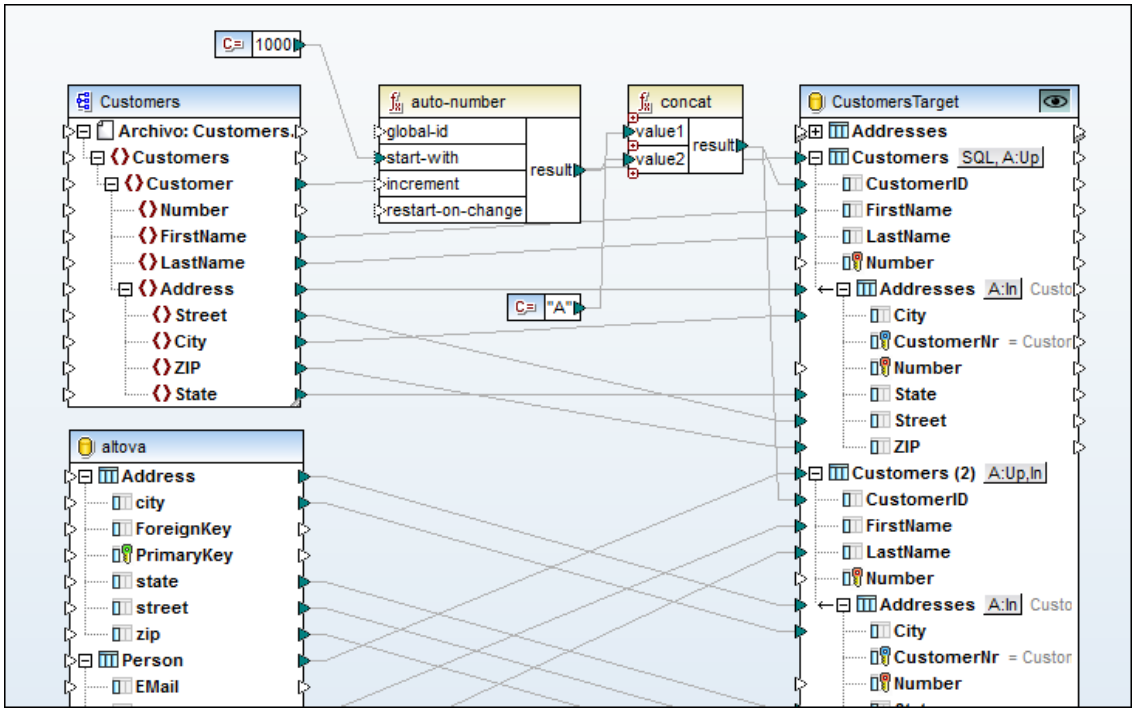
		que no tienen equivalentes en el XML de origen).	
Conserva las tablas de la BD que no tienen equivalentes en el XML de entrada (porque no existen claves de origen para la comparación **Actualizar si...**). En este caso se trataría de la tabla **`Engineering`**.			
Inserta en la BD todos los registros de **`Person`** del archivo XML de entrada. También se insertan registros nuevos que no existen todavía en la BD.	La condición **Actualizar si...** definida para la tabla **`Person`** no se cumple porque todos los registros de **`Person`** se eliminaron al activar la opción *Eliminar datos en tablas secundarias.* Es imposible comparar la clave principal de la BD con la del XML de origen porque las claves de la BD se eliminaron. No se actualiza ningún registro.	Los datos de la tabla secundaria (registros de **`Person`**) se eliminan antes de que se ejecute la acción de tabla **Eliminar si...**. No se elimina ningún registro.	Se conservan todos los registros de la BD que no tienen la clave de **`Person`**.

Para ver otro ejemplo con transacciones Insert y Update consulte el archivo `Customers_DB.mfd`, disponible en la carpeta `...\MapForceExamples`. Este ejemplo ilustra cómo se pueden asignar datos XML y de base de datos a una base de datos de destino.

En el archivo `Customers_DB.mfd` hay asignaciones de datos:

- Entre el esquema XML de origen y la BD de destino:
  Las tablas **`Customers`** y **`Addresses`** existen en la BD de destino. Estas tablas se actualizan con datos nuevos del archivo XML de origen. Para encontrar las filas correctas de la BD se usan los elementos **`FirstName`** y **`LastName`**.
- Entre la BD de origen y la BD de destino:
  Los datos de las tablas **`Addresses`** y **`Person`** vienen dados por la BD de origen y se insertan en la BD de destino. Se crea un duplicado de la tabla de destino (**`Customers`**).

  Se crea el campo **`CustomerID`** para cada registro, con un valor inicial de **`A1000`**.



Acción de base de datos: eliminación

La acción de tabla **Eliminar si...** se utiliza para eliminar datos de las tablas de forma selectiva. Esto se consigue seleccionando los campos/elementos de los componentes de origen y destino que se deben comparar. Dependiendo del resultado de esta comparación, la acción de tabla se ejecuta o no.

Nota: es importante no confundir esta acción de tabla con la opción *Eliminar datos en tablas secundarias* del cuadro de diálogo "Acciones de tabla de la base de datos". La acción de tabla **Eliminar si...** solamente afecta a la tabla para la que se define esta acción y no afecta a ninguna otra tabla.

Objetivo

En este apartado del tutorial aprenderá a usar MapForce para eliminar datos de las tablas de la BD. En concreto aprenderá a:

- Eliminar los registros de **`Person`** ya existentes en la BD
- Insertar registros de **`Person`** nuevos desde el archivo XML de entrada

Comandos utilizados en este apartado

Insertar archivo o esquema XML: haga clic en este icono para abrir el cuadro de diálogo Abrir de Windows y seleccionar el archivo de su sistema de archivos.

Insertar base de datos: este comando está disponible en la barra de herramientas "Insertar componentes" y en el menú **Insertar**. Haga clic en este comando para abrir el cuadro de diálogo "Seleccione una base de datos" donde puede definir una conexión de base de datos y seleccionar las tablas de BD que desea incluir en la asignación de datos de MapForce.

 © 2014 Altova GmbH

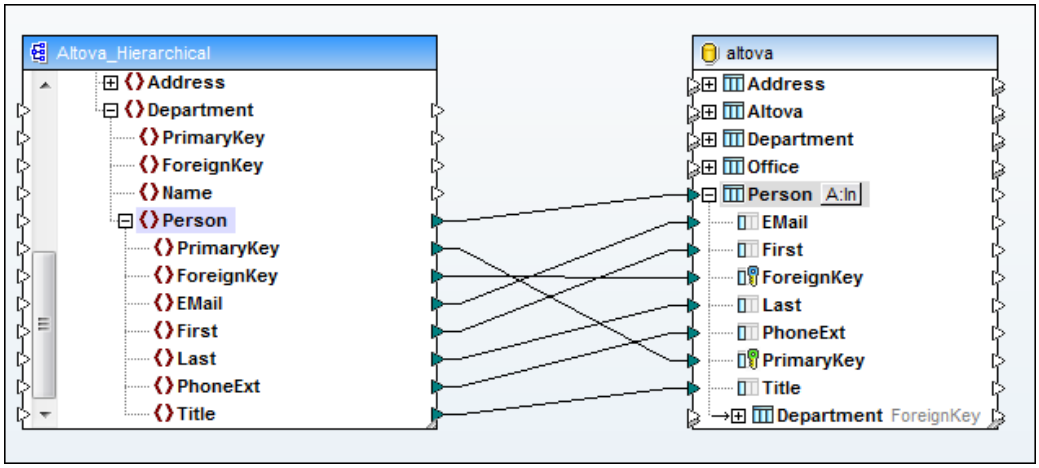
Conectar automáticamente los secundarios equivalentes: haga clic en este icono para activar/desactivar la conexión automática de nodos secundarios equivalentes.

Acciones de tabla de la base de datos: este comando aparece en el menú **Componente** de un componente de base de datos y aparece en forma de botón junto al nombre de la tabla en el componente. Haga clic en este comando para abrir el cuadro de diálogo "Acciones de tabla de la base de datos", donde puede definir las acciones que se deben ejecutar para cada registro de la BD.

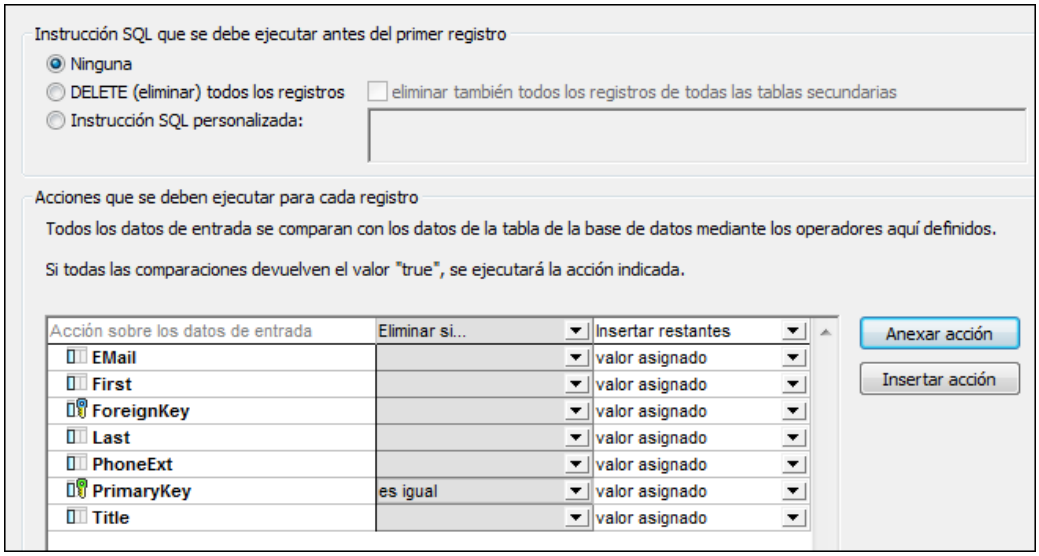
Ejecutar script SQL: este comando está en la barra de herramientas "Vista previa del resultado" y en el menú **Resultados**. Haga clic en este comando para ejecutar el script SQL que se generó en el panel **Resultados**.

Para configurar la asignación y definir las acciones de tabla de la BD para Eliminar si...:

1. Inserte el componente de esquema basado en `Altova_Hierarchical.xsd` y asígnele el archivo XML de entrada `altova-cmpy-extra.xml`.
2. Inserte el componente de base de datos basado en `altova.mdb` y en todas sus tablas de usuario.



3. Seleccione la opción de menú **Conexión | Conectar automáticamente los secundarios equivalente** (si no está activa ya esta opción).
4. Haga clic en el elemento **`Person`** del componente de esquema XML y conéctelo con la tabla **`Person`** de la base de datos. Asegúrese de que lo conecta con la tabla raíz **`Person`**. Se conectan automáticamente todos los elementos secundarios equivalentes.
5. Haga clic en el icono A:In de la tabla **`Person`** para abrir el cuadro de diálogo "Acciones de tabla de la base de datos".
6. Haga clic en el primer cuadro combinado (*Acción sobre los datos de entrada*) y seleccione **Eliminar si...**.
7. Haga clic en el botón **Anexar acción**.
   Se inserta una columna nueva de acción de tabla llamada **Insertar restantes**.



8. Haga clic en el cuadro combinado `PrimaryKey` y seleccione la opción **es igual**.
   Las claves principales de origen y destino se comparan usando el operador `igual que`.
   Si las claves son idénticas, el registro se elimina.
   La acción **Insertar restantes** crea registros nuevos en la tabla **`Person`** con los elementos del archivo XML de origen que no tienen equivalentes en la BD.

9. Haga clic en **Aceptar** para confirmar.
   El icono de acción de tabla ahora es **A:De,In** (*Acción: eliminar, insertar*)

   Nota: si el cuadro combinado de alguna entrada `PrimaryKey` o `ForeignKey` no incluye opciones (es decir, la lista desplegable está vacía), es necesario cambiar las propiedades de control de clave (consulte el apartado Acciones de tabla: configuración de clave).

10. Abra el panel **Resultados** para ver el script SQL resultante. El script ejecuta la asignación a la base de datos de destino teniendo en cuenta las acciones de tabla definidas.

11. Haga clic en el icono **Ejecutar script SQL** de la barra de herramientas para ejecutar el script y actualizar los registros de la BD.
12. Abra la base de datos `altova.mdb` en DatabaseSpy o MS Access y haga doble clic en la tabla **`Person`** para ver los resultados.

 © 2014 Altova GmbH

Person : Table

PrimaryKey	ForeignKey	EMail	First	Last
6	3	f.landis@nanonull.com	Fred	Landis
7	3	m.landis@nanonull.co	Michelle	Butler
8	3	t.little@nanonull.com	Ted	Little
9	3	a.way@nanonull.com	Ann	Way
10	3	l.gardner@nanonull.cc	Liz	Gardner
11	3	p.smith@nanonull.cor	Paul	Smith
12	4	a.martin@nanonull.co	Alex	Martin
13	4	g.hammer@nanonull.c	George	Hammer
30	1	c.Cicada@microtech.	Camilla	Cicada
31	1	c.corrigan@microtech	Carol	Corrigan

Secuencia de procesamiento

La condición **Eliminar si...** comprueba si las claves principales de los elementos de origen y destino son idénticos.

Tabla **`Person`**:

- Si la **condición es verdadera**, se eliminan los registros de **`Person`** cuya clave principal es idéntica a la del elemento de origen. Si en la BD existen registros que no tienen equivalentes en el archivo XML de origen, estos registros se conservan tal y como están.
- Si la **condición es falsa** (es decir, la clave principal de los elementos de origen no coincide con la de los registros de la BD de destino), MapForce pasa a la acción de tabla **Insertar restantes**. Esta acción inserta los registros del elemento **`Person`** que no existen en la tabla **`Person`**. En este ejemplo, dos registros nuevos se añaden al departamento **`Administration`**, que tienen la clave principal 30 y 31 respectivamente.

En el apartado Actualizar y eliminar datos secundarios hay dos ejemplos más de la acción de tabla **Eliminar si...**.

Acción de base de datos: omisión

La acción de tabla **Omitir si...** se utiliza para omitir de forma selectiva determinados registros creados por la asignación. Esta acción genera una instrucción SELECT con la condición especificada. Si esta instrucción encuentra los datos, se omite el registro de entrada asignado.

Esta acción de tabla se puede usar junto con una acción **Insertar restantes** para omitir todos los registros de entrada que ya existen en la BD e insertar registros nuevos.

Objetivo

En este apartado del tutorial aprenderá a:

- omitir registros duplicados de la tabla **`Person`** de la BD que proceden del archivo XML de entrada
- insertar elementos del archivo XML de entrada como registros nuevos en la tabla **`Person`**

Comandos utilizados en este apartado

Insertar archivo o esquema XML: haga clic en este icono para abrir el cuadro de diálogo Abrir de Windows y seleccionar el archivo de su sistema de archivos.

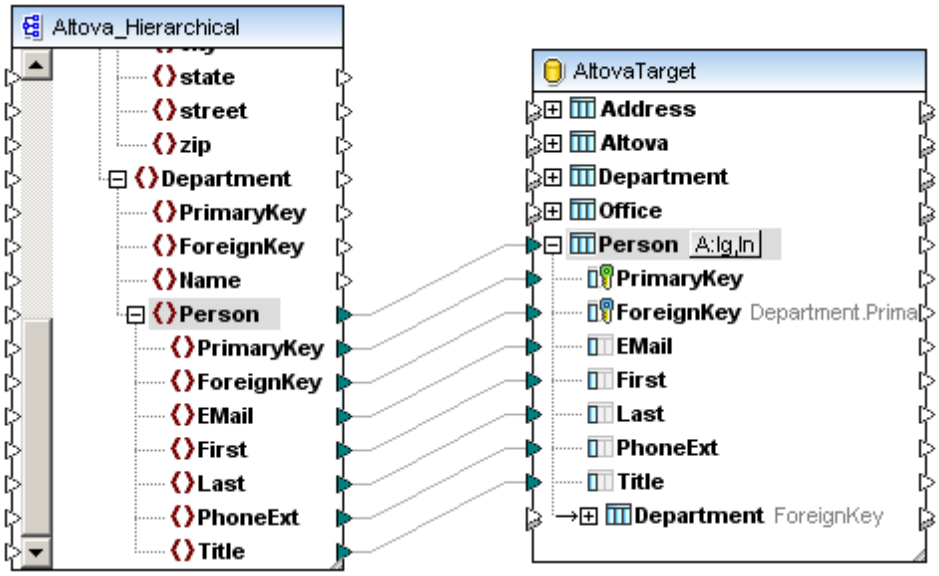
Insertar base de datos: este comando está disponible en la barra de herramientas "Insertar componentes" y en el menú **Insertar**. Haga clic en este comando para abrir el cuadro de diálogo "Seleccione una base de datos" donde puede definir una conexión de base de datos y seleccionar las tablas de BD que desea incluir en la asignación de datos de MapForce.

A:In **Acciones de tabla de la base de datos**: este comando aparece en el menú **Componente** de un componente de base de datos y aparece en forma de botón junto al nombre de la tabla en el componente. Haga clic en este comando para abrir el cuadro de diálogo "Acciones de tabla de la base de datos", donde puede definir las acciones que se deben ejecutar para cada registro de la BD.

Conectar automáticamente los secundarios equivalentes: haga clic en este icono para activar/desactivar la conexión automática de nodos secundarios equivalentes.

Para configurar la asignación y definir las acciones de tabla de la BD para Omitir si...:

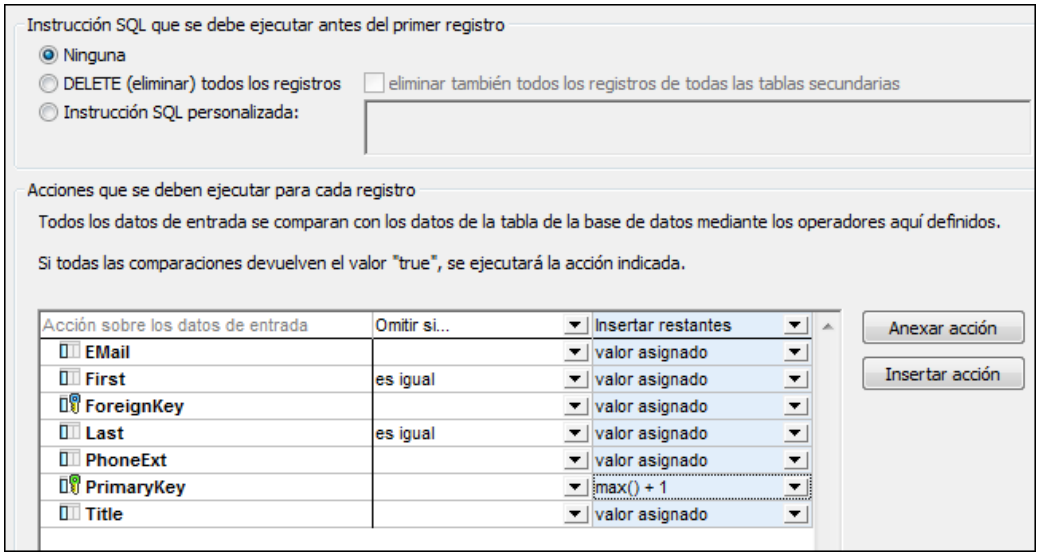
1. Inserte el componente de esquema basado en `Altova_Hierarchical.xsd` y asígnele el archivo XML de entrada `altova-cmpy-extra.xml`.
2. Inserte el componente de base de datos basado en `altova.mdb` y en todas sus tablas de usuario.



3. Seleccione la opción de menú **Conexión | Conectar automáticamente los secundarios equivalente** (si no está activa ya esta opción).
4. Haga clic en el elemento **`Person`** del archivo XML y conéctelo con la tabla **`Person`** de la BD. Asegúrese de que lo conecta con la tabla raíz **`Person`**.
   Se crean asignaciones automáticas entre los elementos secundarios equivalentes.
5. Haga clic en el icono A:In de la tabla **`Person`** para abrir el cuadro de diálogo "Acciones de tabla de la BD".
6. Haga clic en el primer cuadro combinado (*Acción sobre los datos de entrada*) y seleccione **Omitir si...**.

 © 2014 Altova GmbH

7. Haga clic en el botón **Anexar acción**.
   Se inserta una columna nueva de acción de tabla llamada **Insertar restantes**.



8. Haga clic en el cuadro combinado de **`Last`** y **`First`** y seleccione la opción **`max() + 1`**.
9. Haga clic en el cuadro combinado de **`First`** y seleccione la opción **`es igual`**. Haga lo mismo con el cuadro combinado de **`Last`**.
10. Para confirmar haga clic en **Aceptar**.

Secuencia de procesamiento

La acción de tabla **Omitir si...** compara los campos **`First`** y **`Last`** del XML de origen con los campos **`First`** y **`Last`** de la BD:

- Si se encuentran los datos de ambos campos en la BD, los datos asignados se omiten y no se procesa la acción de tabla **Insertar restantes**.
- Si no se encuentran los datos de los campos **`First`** y **`Last`**, entonces se genera un nuevo registro en la BD por medio de la entrada **`max()+ 1`** de **`PrimaryKey`**.

Generar valores de salida de la BD

MapForce ofrece funciones para generar valores para campos de BD que no tienen datos de entrada del componente de origen (ya sea de esquema o base de datos o archivo EDI). Esta característica funciona tanto en los lenguajes de programación compatibles como con el motor de ejecución integrado.

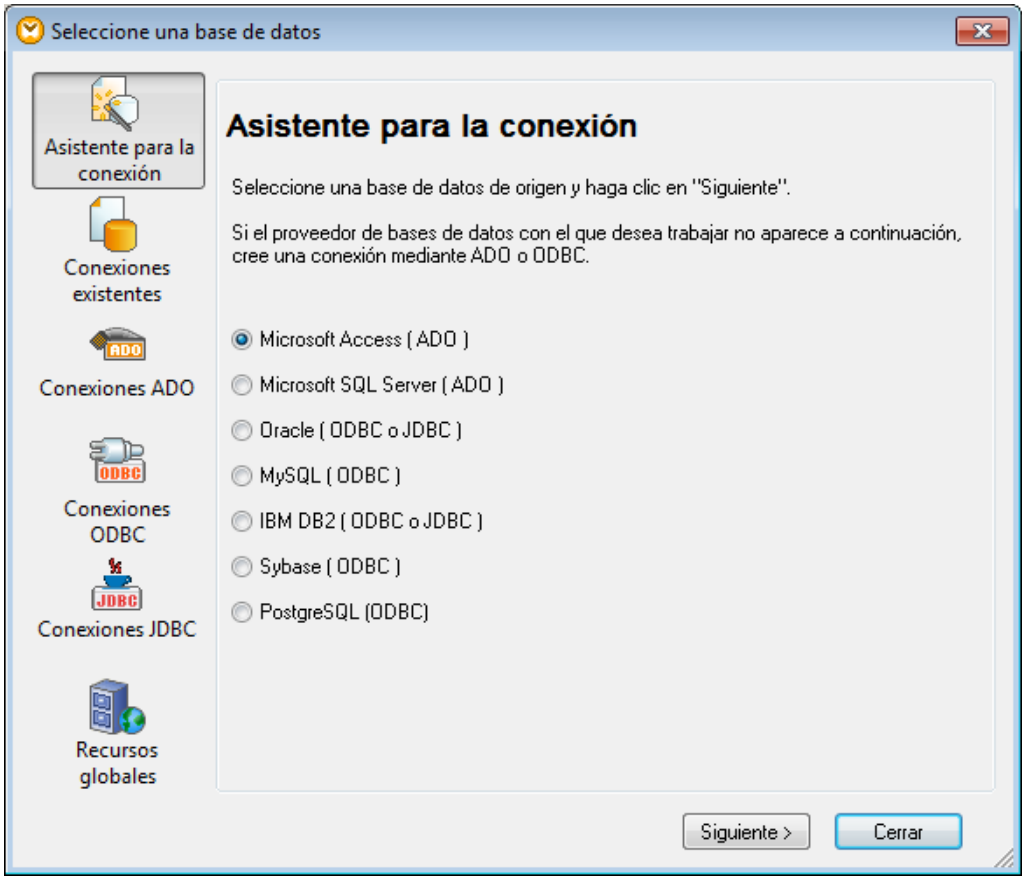
Las funciones que pueden generar valores para campos de BD son:

`auto-number` (funciones **core | generator**), que también está disponible para el motor de ejecución integrado, se suele usar para generar valores de clave principal para un campo numérico.

`create-guid` (funciones **lang | generator**) crea un identificador global único (en forma de cadena con codificación hexadecimal) para el campo de la BD.

10.2.3 Uso del asistente para la conexión

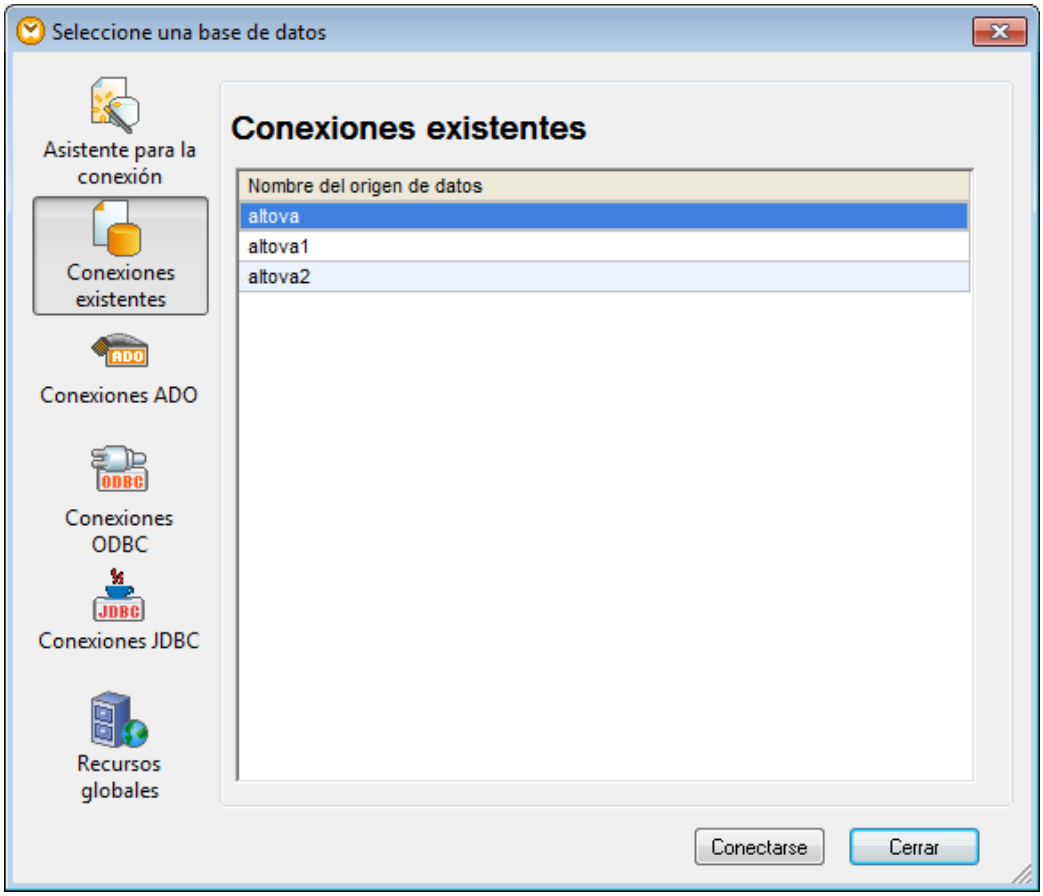
El **Asistente para la conexión** simplifica el proceso de conexión a la BD y ofrece varios tipos de conexión estándar. En el asistente puede crear conexiones con rapidez a cualquier tipo de BD.



El panel **Conexiones existentes** enumera todas las conexiones de BD que están **activas** actualmente.

- Seleccione una conexión de la lista y haga clic en el botón **Conectarse**.

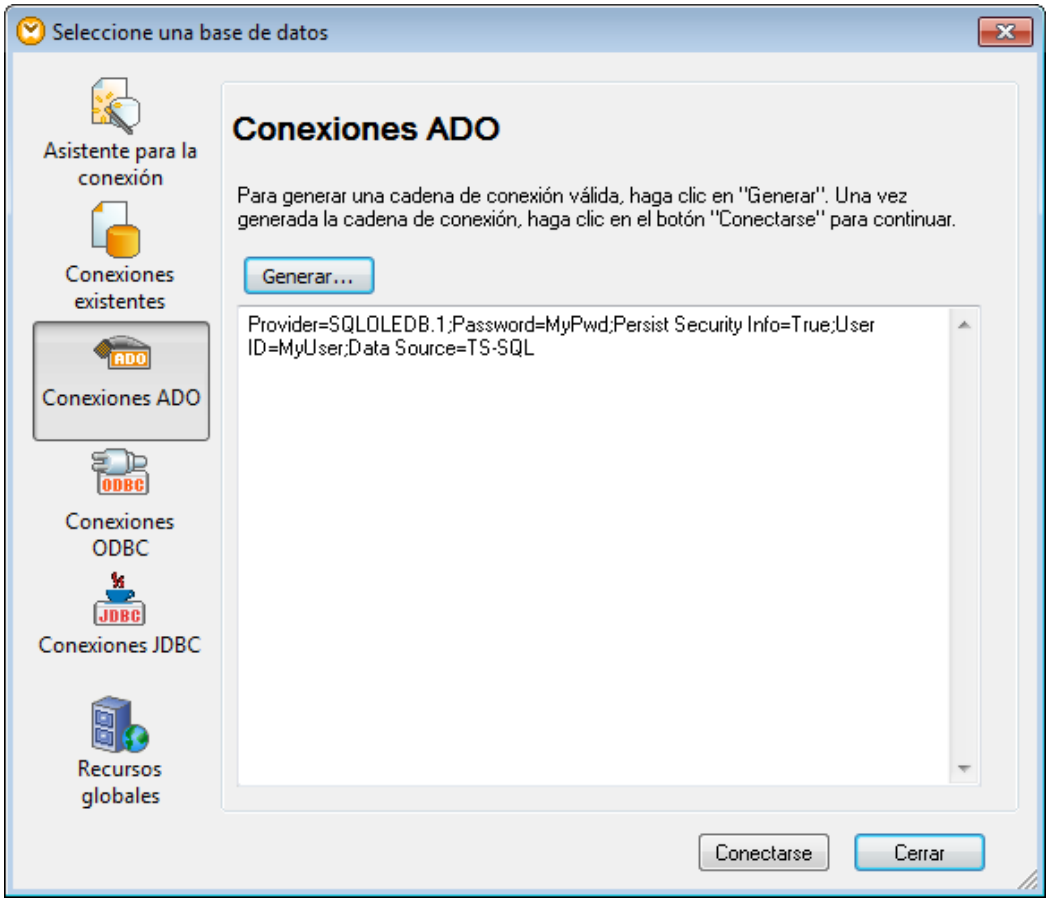
 © 2014 Altova GmbH



En el panel **Conexiones ADO** puede crear una conexión ADO (objetos de datos ActiveX) a una BD.

- Haga clic en el botón **Generar** para crear la cadena de conexión. Después de definirla, la cadena de conexión aparece en la ventana. Para terminar haga clic en el botón **Conectarse**.

  La conexión ADO se puede crear sin ningún paso preliminar (como crear un DSN). Para conectarse a bases de datos MS Access utilice siempre una conexión ADO porque ODBC no admite relaciones.



En el panel **Conexiones ODBC** puede crear una conexión ODBC (conectividad abierta de bases de datos) a una base de datos. Puede elegir una de estas opciones:

- *DSN (nombre de origen de datos) de sistema:* este tipo de DSN puede ser utilizado por cualquier usuario. La información DSN se almacena en el registro.
- *DSN de usuario:* este tipo de DSN se crea para un usuario determinado y también se almacena en el registro.
- *DSN de archivo:* para este tipo de DSN la información se almacena en un archivo de texto con la extensión `.dsn`.

 © 2014 Altova GmbH

Nota: para crear una conexión ODBC antes es necesario crear un DSN. El cuadro de lista de la parte inferior del panel enumera todos los DSN creados previamente.

Al seleccionar el botón de opción DSN de sistema o DSN de usuario, aparecen varios iconos en la parte inferior del cuadro de diálogo. Con estos iconos puede crear o editar los DSN ya existentes:

Crear DSN nuevo: crea un DSN nuevo.

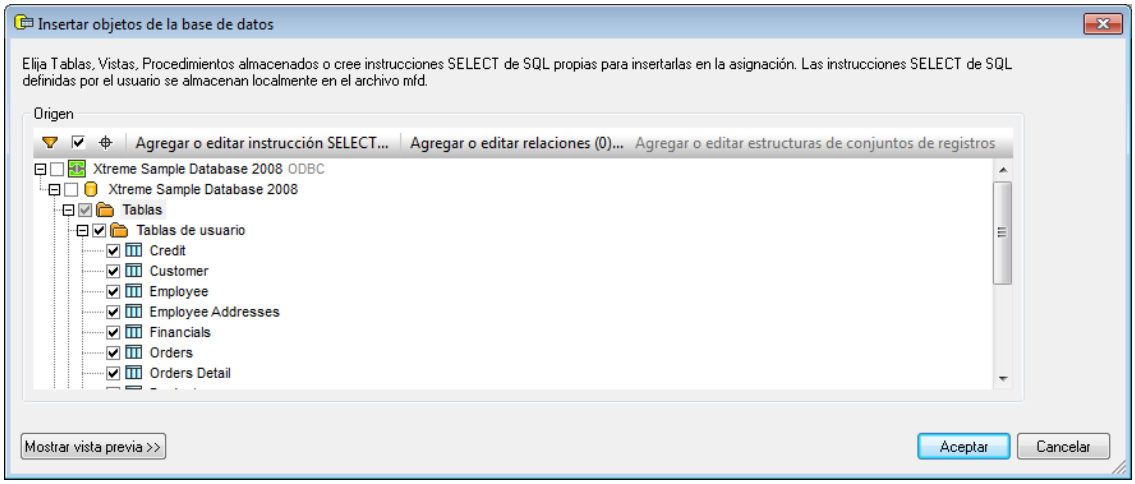
Editar DSN seleccionado: permite cambiar la configuración del DSN que está seleccionado en el cuadro de lista.

Eliminar DSN seleccionado: elimina el DSN que está seleccionado en el cuadro de lista.

Actualizar la lista: actualiza los DSN que aparecen en el cuadro de lista.

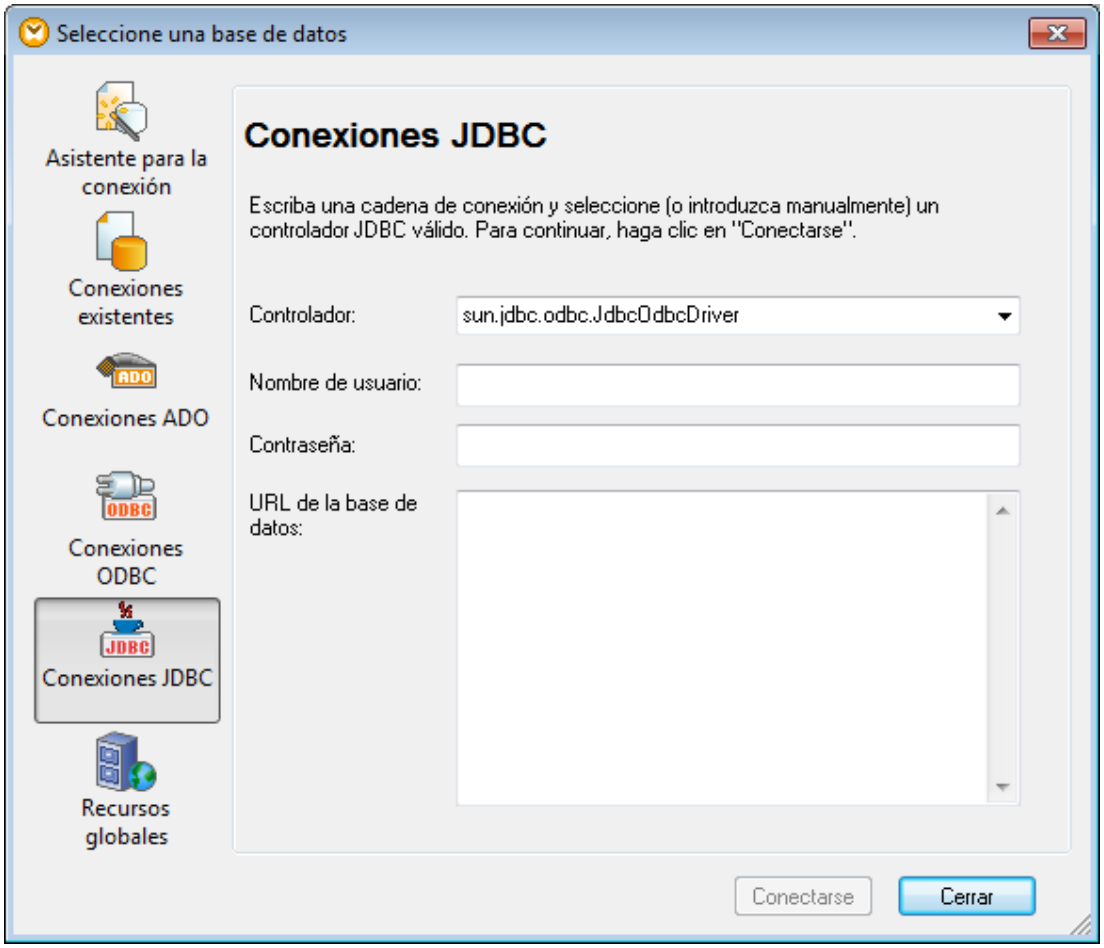
Al hacer doble clic en uno de los DSN aparece el cuadro de diálogo de inicio de sesión de la BD correspondiente.

Introduzca los datos de inicio de sesión y haga clic en **OK**.

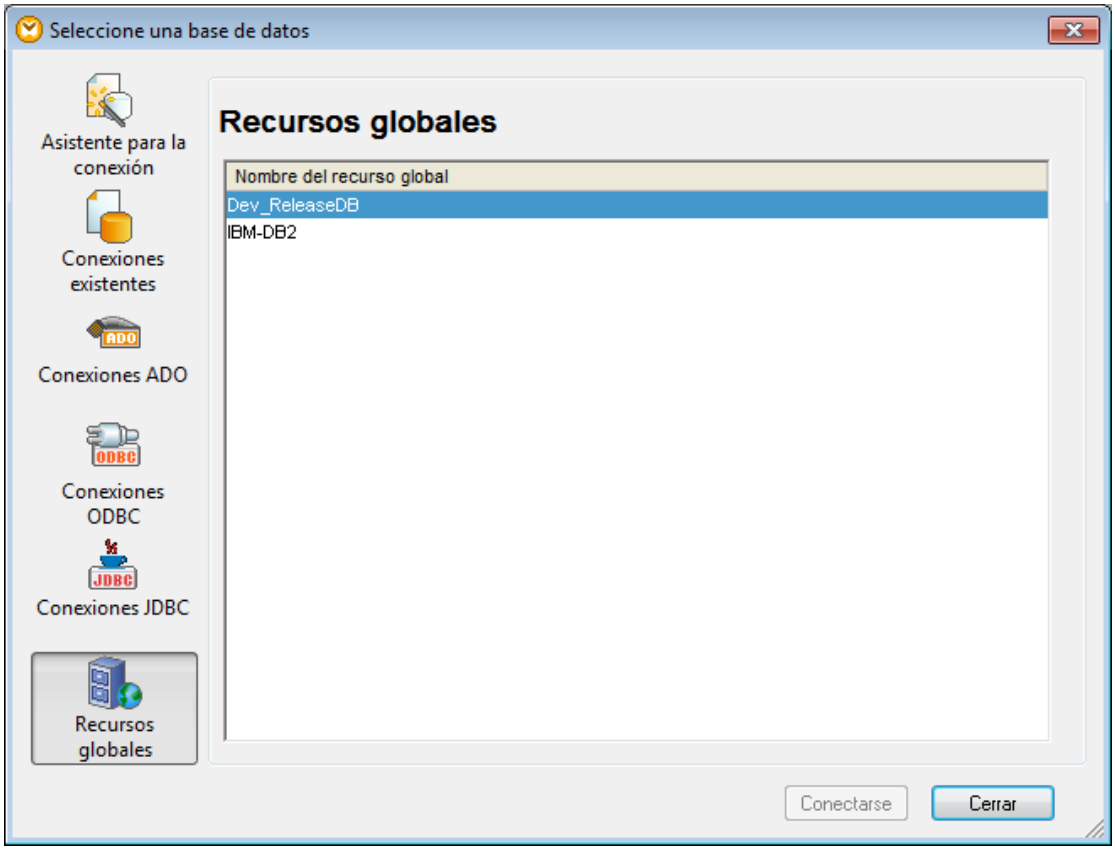


El panel **Conexiones JDBC** permite establecer una conexión a una base de datos con ayuda de un controlador JDBC. Para más información, consulte el apartado Conexiones JDBC.

 © 2014 Altova GmbH



En el panel **Recursos globales** puede seleccionar un alias de base de datos de los recursos globales definidos. Para más información consulte el apartado Recursos globales: bases de datos.



 © 2014 Altova GmbH

10.2.4 Conexiones JDBC

En este apartado describimos cómo se establece una conexión con una BD con ayuda de un controlador JDBC. Las instrucciones que aparecen a continuación describen una conexión a una base de datos IBM DB2 pero también sirven para otros tipos de BD que admitan conexiones JDBC.

Requisitos

Antes de conectarse a la BD por JDBC:

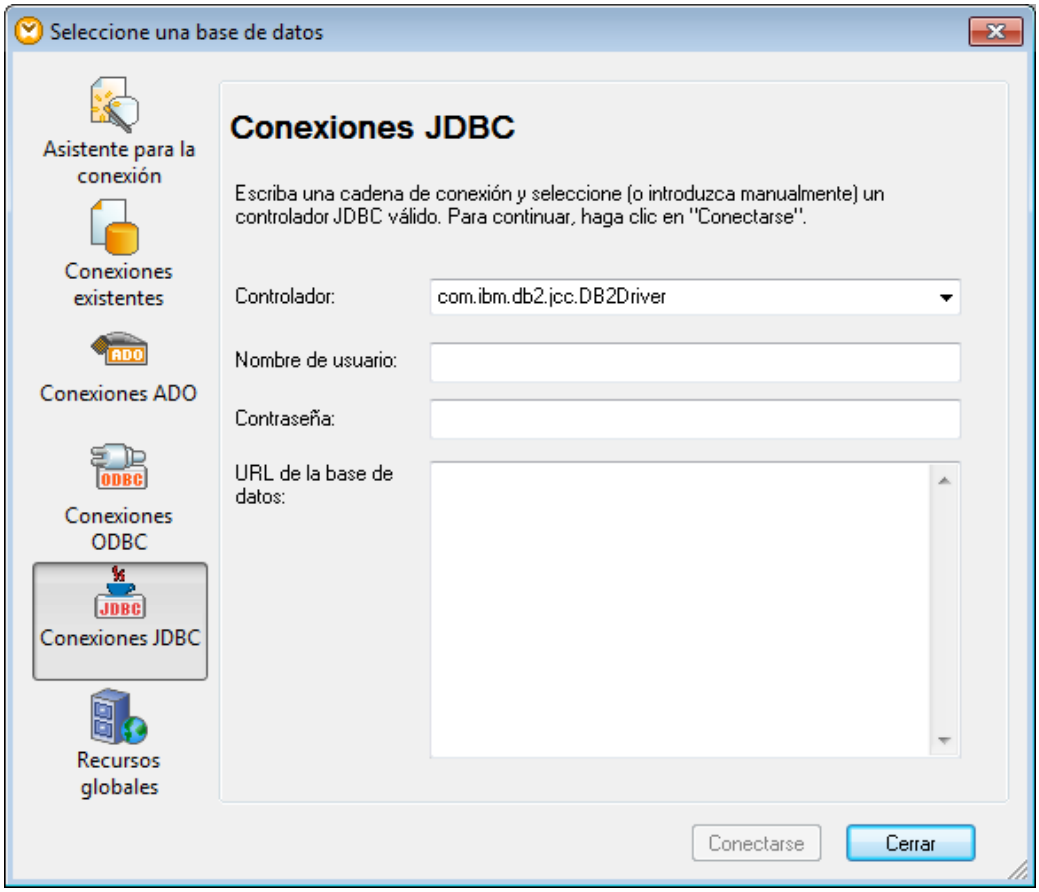
1. Compruebe que está instalado Java Runtime Environment (JRE). Use un JRE de 32 bits si el equipo es de 32 bits o un JRE de 64 bits si el equipo es de 64 bits.
2. Instale un controlador JDBC. No hay ningún requisito de instalación especial. Basta con copiar el controlador en un directorio local, por ejemplo `c:\jdbc`. Recuerde que los controladores JDBC (que son archivos Jar) son independientes de la plataforma y los controladores funcionan en sistemas operativos de 32 y 64 bits.
3. Configure la variable `CLASSPATH` para que incluya la ubicación donde está el controlador JDBC. (La aplicación lee la variable de entorno `CLASSPATH` para localizar el controlador JDBC). Para acceder a la variable `CLASSPATH` y editarla, haga clic en **Inicio | Panel de control | Sistema | Opciones avanzadas | Variables de entorno**. En el cuadro de diálogo "Variables de entorno" que aparece seleccione la variable de entorno de usuario `CLASSPATH` o de sistema `CLASSPATH` y haga clic en el botón **Edición**. Añada la ruta de acceso del controlador JDBC a la variable `CLASSPATH`. Por ejemplo: `CLASSPATH=C:\jdbc\sqljdbc.jar; C:\jdbc\db2jcc.jar;`.
4. Cierre sesión y vuelva a iniciar sesión para que se activen los cambios.

Nota: cuando se instala todo el cliente IBM DB2 u Oracle, los controladores JDBC del paquete pertinente se incluyen en la variable `CLASSPATH` automáticamente. Para más información lea el resto de este apartado.

Conexión por medio de controladores JDBC

Para conectarse a la BD usando una conexión JDBC seleccione **Insertar | Base de datos**:

1. Seleccione **Conexiones JDBC** en el cuadro de diálogo "Seleccione una base de datos". Aparece el panel **Conexiones JDBC** (*imagen siguiente*).



2. Seleccione el controlador JDCB de la lista desplegable *Controlador:* (la lista enumera todos los controladores detectados, es decir, los de la variable de entorno `CLASSPATH`). Escriba la URL de la BD, el nombre de usuario y la contraseña. A continuación ofrecemos la sintaxis de direcciones URL para las bases de datos más utilizadas (cada una con un ejemplo).

Oracle

```
jdbc:oracle:thin:[usuario/contraseña]@//[host][:puerto]/SID
```

```
jdbc:oracle:thin:@//abcd234/ORA11
```

IBM DB2

```
jdbc:db2://nombre_host:puerto/nombreBD
```

```
jdbc:db2://MiDB2:50000/boz
```

MySQL

```
jdbc:mysql://nombre_host:puerto/nombreBD
```

```
jdbc:mysql://MiDB2:3306/moz
```

MSSQL

```
jdbc:sqlserver:// host:puerto;databasename=nombre;user=usuario;password=contras eña
```

```
jdbc:sqlserver://abcd38:1433;databasename=coz
```

 © 2014 Altova GmbH

	`jdbc:sqlserver://Q5;DatabaseName=coz;SelectMethod=Cursor`
PostgreSQL	`jdbc:postgresql://host:puerto/baseDatos`
	`jdbc:postgresql://abc993:5432/qanoz`
Sybase	`jdbc:sybase:Tds:host:puerto/nombreBD`
	`jdbc:sybase:Tds:abc12:2048/QUE`

3. Haga clic en el botón **Conectarse** para establecer la conexión. Se establece la conexión y aparece un cuadro de diálogo donde puede seleccionar los datos de la BD que desea usar. Para más información consulte el apartado Selección de los datos de la BD.

Instrucciones paso a paso para bases de datos MSSQL, MySQL, PostGre y otras BD no XML

Estas son las instrucciones para conectarse a bases de datos no XML a través de JDBC:

1. Archivos JAR JDBC (archivos de controladores): copie los archivos a una carpeta local. Después añada la ruta de acceso y el nombre de archivo a la variable de Windows `CLASSPATH`. Por ejemplo: `C:\jdbc\sqljdbc.jar; C:\jdbc\db2jcc.jar;`.
2. Cierre sesión y vuelva a iniciar sesión para que se activen los cambios realizados en `CLASSPATH`.
3. Inicie MapForce y abra el cuadro de diálogo "Seleccione una base de datos".
4. En el panel **Conexiones JDBC** los controladores JDBC detectados se enumeran en la lista desplegable del cuadro combinado *Controlador*. Si la lista está vacía, compruebe que el archivo `altovadb.jar` está en la carpeta: `C:\Archivos de programa\Altova\Common2014\jar`.
5. Conéctese a la BD siguiendo las instrucciones dadas más arriba.

Instrucciones paso a paso para bases de datos Oracle

Estas son las instrucciones para conectarse a una BD Oracle a través de JDBC. Si no necesita las características XML y XDB de la BD Oracle, siga las instrucciones del párrafo anterior para BD no XML. La carpeta de instalación del cliente de Oracle se indica con el marcador de posición: `%ORACLE_HOME%`.

1. Instale el software cliente de Oracle con las características OCI y ODBC habilitadas. Si ya tiene instalado un cliente de Oracle, compruebe si estos dos archivos JAR están en el equipo:

   ```
   %ORACLE_HOME%\LIB\xmlparserv2.jar
   %ORACLE_HOME%\RDBMS\jlib\xdb.jar
   ```

2. Añada estos archivos a la variable de entorno Windows `CLASSPATH`:

   ```
   %ORACLE_HOME%\jdbc\lib\ojdbc6.jar
   %ORACLE_HOME%\LIB\xmlparserv2.jar
   %ORACLE_HOME%\RDBMS\jlib\xdb.jar
   ```

3. Cierre sesión y vuelva a iniciar sesión para que se activen los cambios realizados en `CLASSPATH`.
4. Inicie MapForce y abra el cuadro de diálogo "Seleccione una base de datos".
5. En el panel **Conexiones JDCB** los controladores JDCB detectados se enumeran en la lista desplegable del cuadro combinado *Controlador*. Si la lista está vacía, compruebe que el archivo `altovadb.jar` está presente en la carpeta: `C:\Archivos de programa\Altova\Common2014\jar`.
6. Conéctese a la BD siguiendo las instrucciones dadas al principio de este apartado.

Instrucciones paso a paso para bases de datos IBM DB2

Estas son las instrucciones para conectarse a una BD IBM DB2 a través de JDBC.

1. Si ya tienen instalado un cliente IBM DB2, no es necesario hacer nada porque la variable `CLASSPATH` ya se configuró durante el proceso de instalación del cliente.
2. Si no tiene instalado el cliente IBM DB2, añada los archivos JAR del controlador JDBC para IBM DB2 `db2jcc.jar` y `db2jcc_license_cu.jar` a la variable de entorno Windows `CLASSPATH`. Cierre sesión y vuelva a iniciar sesión para que se activen los cambios realizados en `CLASSPATH`.
3. Inicie MapForce y abra el cuadro de diálogo "Seleccione una base de datos".
4. En el panel **Conexiones JDBC** los controladores JDBC detectados se enumeran en la lista desplegable del cuadro combinado *Controlador*. Si la lista está vacía, compruebe que el archivo `altovadb.jar` está en la carpeta: `C:\Archivos de programa\Altova\Common2014\jar`.
5. Conéctese a la BD siguiendo las instrucciones dadas más arriba.

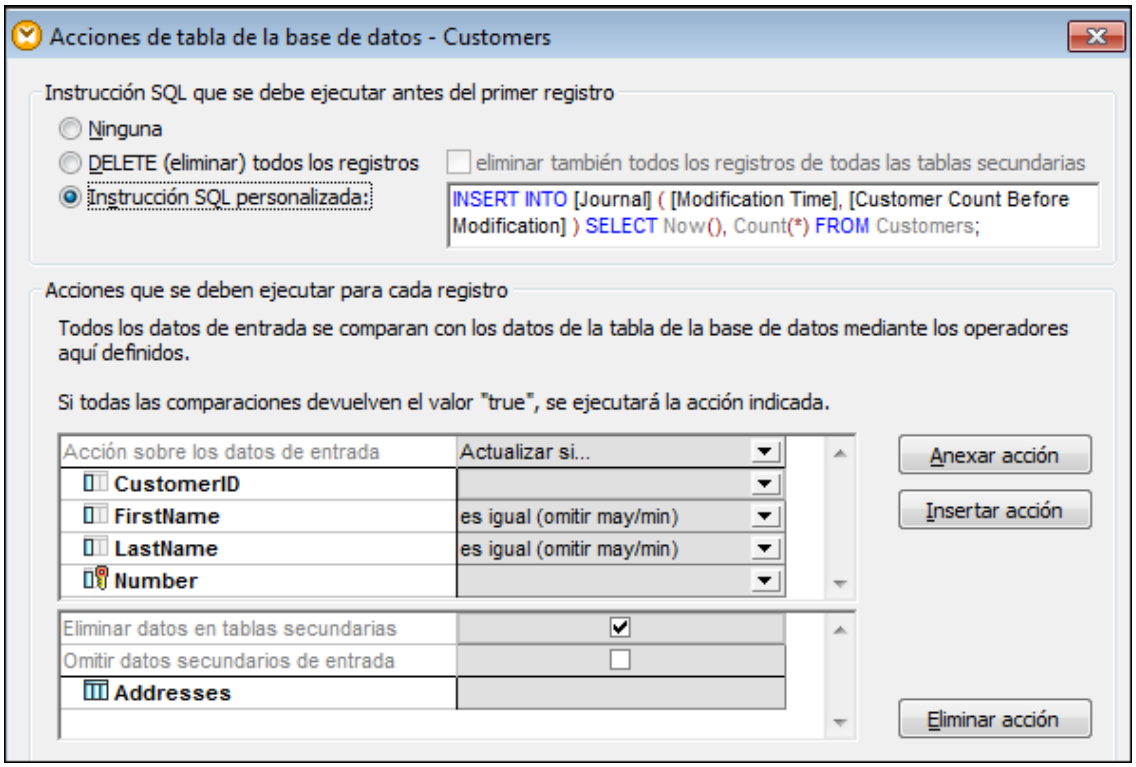
Nota: si se conecta a una BD a través de JDBC y los controladores no devuelven suficiente información: (i) no se pueden editar los datos de las tablas que no tengan una clave principal y (ii) el comando **Ejecutar para editar datos** del Editor SQL no funciona.

Si se conecta usando JDBC y utiliza una instrucción SELECT con el comando Agregar/Quitar tablas para recuperar datos, la instrucción SELECT no puede incluir el punto y coma final.

 © 2014 Altova GmbH

10.2.5 Acciones de tabla, configuración de clave y transacciones

En el cuadro de diálogo "Acciones de tabla de la base de datos" puede (i) especificar la instrucción SQL que desea ejecutar antes de las acciones de tabla, (ii) indicar qué acciones de tabla se ejecutan en cada registro suministrado por la asignación a la tabla de destino y (iii) especificar la configuración de clave que se debe usar a la hora de insertar registros nuevos en la BD.



Instrucción SQL que se debe ejecutar antes del primer registro
En este grupo de opciones puede definir las instrucciones SQL que se ejecutan antes de recuperar el primer registro de la BD. Puede elegir una de estas tres opciones:

- *Ninguna* (opción predeterminada): no se lleva a cabo ninguna acción.
- *DELETE (eliminar) todos los registros*: se eliminan todos los registros de la tabla seleccionada antes de ejecutar las acciones de tabla definidas en el grupo de opciones inferior. Si también desea eliminar los datos almacenados en las tablas secundarias de la tabla seleccionada, marque la casilla *eliminar también todos los registros de todas las tablas secundarias*.
- *Instrucción SQL personalizada:* escriba la instrucción SQL personalizada que desea ejecutar en toda la tabla.

Nota: las BD MS Access, MySQL y Oracle no admiten el uso de varias instrucciones SQL. El uso de múltiples instrucciones SQL con otros tipos de BD depende del método de conexión y del controlador utilizado para la conexión.

Acciones que se deben ejecutar para cada registro
Las acciones de tabla definen cómo se procesan los registros asignados. Hay varias opciones:

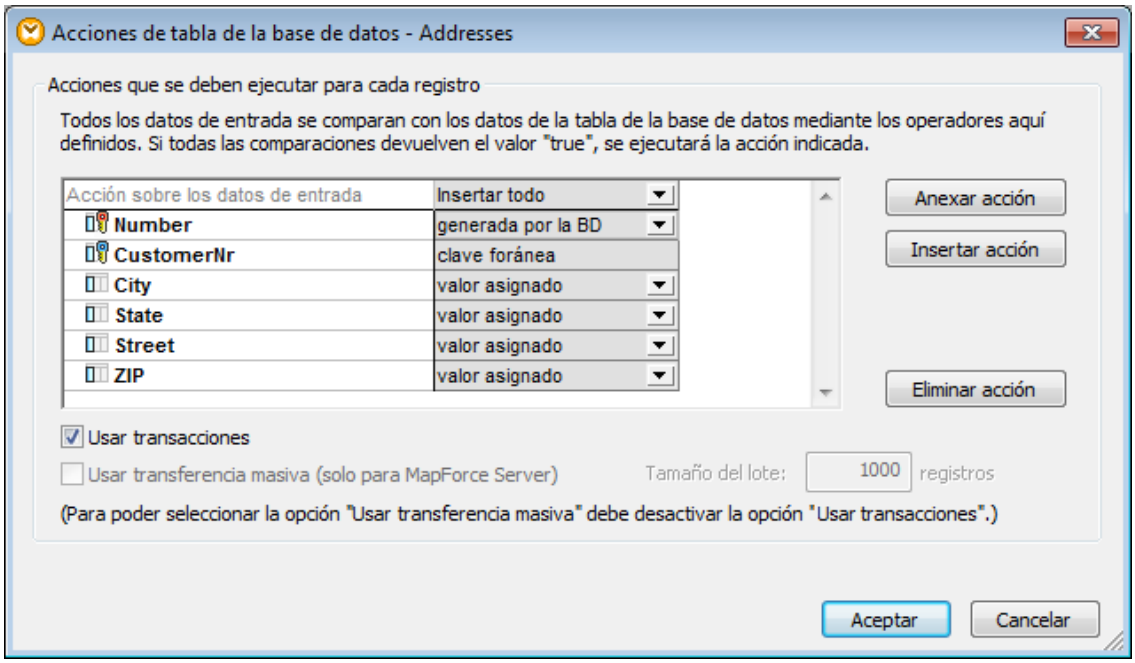
Insertar, **Actualizar**, **Eliminar** y **Omitir**. La acción de tabla se selecciona en el cuadro combinado *Acción sobre los datos de entrada*.

Para más información consulte el apartado Acciones de tabla.

Acciones de tabla

En el grupo de opciones *Acciones que se deben ejecutar para cada registro* puede definir acciones de tabla que determinarán cómo se procesan los registros asignados a la tabla de destino. MapForce ofrece las acciones de tabla: **Insertar**, **Actualizar**, **Eliminar** y **Omitir**.

Las acciones de tabla **Actualizar**, **Eliminar** y **Omitir** necesitan una **condición** que determine en qué registros se lleva a cabo la acción. La condición compara campos de origen y destino para confirmar si la acción de tabla se debe ejecutar o no. La acción de tabla **Insertar** no requiere el uso de una condición, pero tiene una **configuración de clave** que define cómo se genera la clave principal.



En este cuadro de diálogo, por cada columna de la BD que tenga una asignación, puede definir:

- los campos que se deben comparar (p. ej. PrimaryKey, FirstName etc.)
- los operadores utilizados para la comparación (es igual, es igual omitiendo mayús/minús) y
- la acción que se debe llevar a cabo si se cumplen todas las condiciones de cada columna.

Las acciones de tabla definidas en este cuadro de diálogo se procesan de izquierda a derecha. En el ejemplo de la imagen anterior, por ejemplo, primero se procesa la columna **Actualizar si...**. Si la condición de la acción **Actualizar** no se cumple, entonces se procesa la siguiente acción (en este caso la acción **Insertar restantes**).

- Para que la acción de tabla se ejecute, deben cumplirse **todas** las condiciones definidas en la columna de la acción. Si es así, se actualizan todos los campos que **tienen una**

asignación (es decir, cuando existe un conector entre el campo de origen y de destino en el panel **Asignación**). Las demás acciones de tabla (las situadas a la derecha de la acción cuya condición se cumple) se ignoran.

- Si no se cumple una condición de la columna, entonces se pasa por alto la acción de tabla y se procesa la siguiente acción de tabla.
- Si no se cumple ninguna condición, no se procesa ninguna acción de tabla.

Nota: MapForce no ejecuta nunca las acciones de tabla definidas después de las acciones **Insertar todo/Insertar restantes** porque las acciones de inserción no incluyen condiciones. Cuando existen tales acciones de tabla aparece un mensaje advirtiendo que las acciones se eliminarán.

Eliminar datos en tablas secundarias:

- Esta opción se activa por defecto al seleccionar la acción **Actualizar si...**.
- Es necesaria si cree que el número de registros del archivo de origen puede ser distinto al número de registros de la BD de destino.
- Le ayuda a mantener la BD sincronizada (sin datos *huérfanos* en las tablas secundarias)

Efecto:

- La condición de **Actualizar si...** se cumple si existe una clave (o cualquier otro campo) equivalente en el archivo XML de origen. Se eliminan todos los datos secundarios de la tabla primaria.
- La acción **Actualizar si...** selecciona la tabla primaria (y las tablas secundarias) en la que se procesa la opción *Eliminar datos en tablas secundarias*.
- Si la condición de **Actualizar si...** no se cumple en la tabla primaria (es decir, no existe una clave/campo equivalente en el XML de origen), no se eliminan los datos secundarios.
- Los registros de la BD que no tienen un equivalente en el archivo de origen no se eliminan de la BD, sino que se conservan.

Omitir datos secundarios de entrada:
Utilice esta opción si quiere actualizar datos de una tabla sin cambiar los registros/las tablas secundarias de dicha tabla.

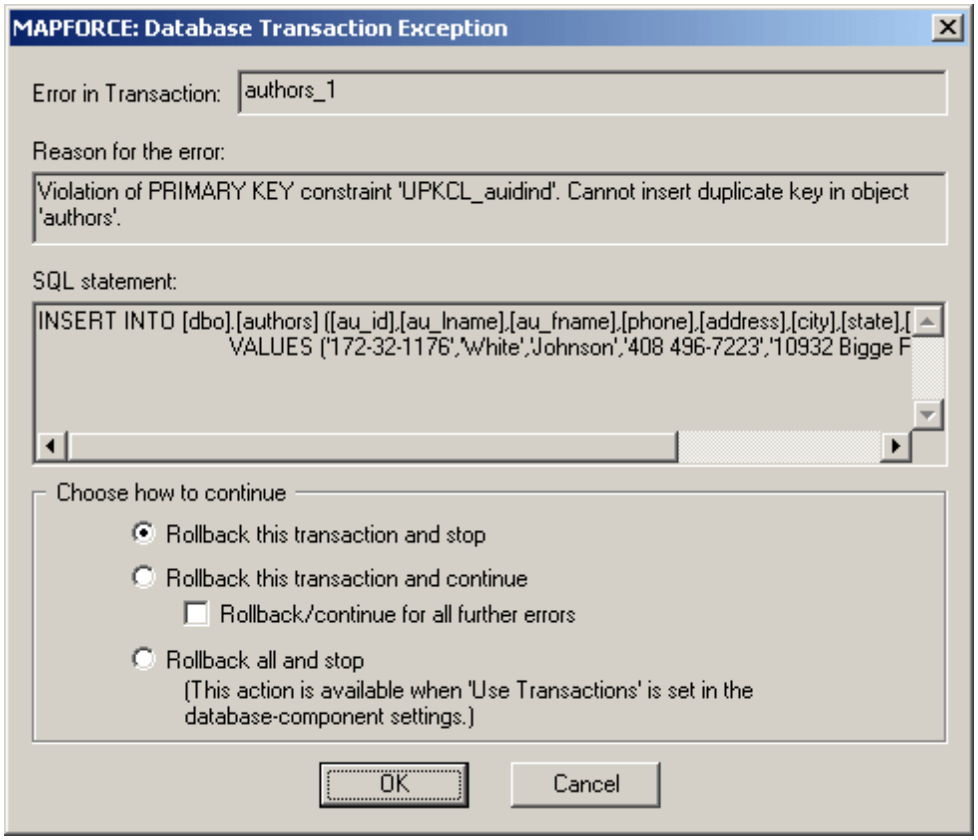
Por ejemplo, imaginemos que su asignación consta de 3 registros de origen y 2 registros de BD de destino. En este caso debería:

- definir una condición **Actualizar si...** para actualizar los registros ya existentes
- marcar la casilla *Omitir datos secundarios de entrada* de la columna **Actualizar si...** para pasar por alto los registros secundarios relacionados
- definir una condición **Insertar restantes** para los registros nuevos que se deban insertar.

Usar transacciones:
La casilla *Usar transacciones* permite definir qué ocurre si por cualquier motivo no se puede ejecutar una acción de BD. Cuando esto ocurre, aparece un cuadro de diálogo que le pregunta qué desea hacer. En este cuadro de diálogo puede seleccionar varias opciones y continuar. Si activa esta opción, la acción de tabla se puede revertir cuando ocurra un error.

Esta opción también se puede activar desde el cuadro de diálogo "Configuración del componente" (opción *Usar transacciones* situada al final del cuadro de diálogo), pero en este caso se revierten

todas las acciones de tabla si se produce un error.



Si ocurre un error y **no está activada** la casilla *Usar transacciones* en el cuadro de diálogo de acciones de tabla ni en el de configuración del componente:

- La ejecución se detiene en el punto donde ocurre el error. Todas las instrucciones SQL anteriores a ese punto se ejecutan y los resultados se almacenan en la BD.

Si ocurre un error y está activada la casilla *Usar transacciones* en el cuadro de diálogo de **configuración del componente**:

- La ejecución se detiene en el punto donde ocurre el error. Todas las instrucciones SQL anteriores se revierten y no se realizan cambios en la BD.

Si ocurre un error y está activada la casilla *Usar transacciones* en el cuadro de diálogo de **acciones de tabla**:

- Aparece el cuadro de diálogo "Excepción de transacción" con la opción *Revertir todas las transacciones y detener la operación* **deshabilitada**. Puede revertir la instrucción SQL que produjo el error para esa **tabla** concreta.

Si ocurre un error y está activada la casilla *Usar transacciones* tanto en el cuadro de diálogo de acciones de tabla como en el de configuración del componente:

- Aparece el cuadro de diálogo "Excepción de transacción" con la opción *Revertir todas las transacciones y detener la operación* **habilitada**. Puede revertir todas las instrucciones SQL anteriores ejecutadas en la BD y en todas sus tablas.

Si hace clic en el botón **Cancelar** del cuadro de diálogo "Excepción de transacción", la

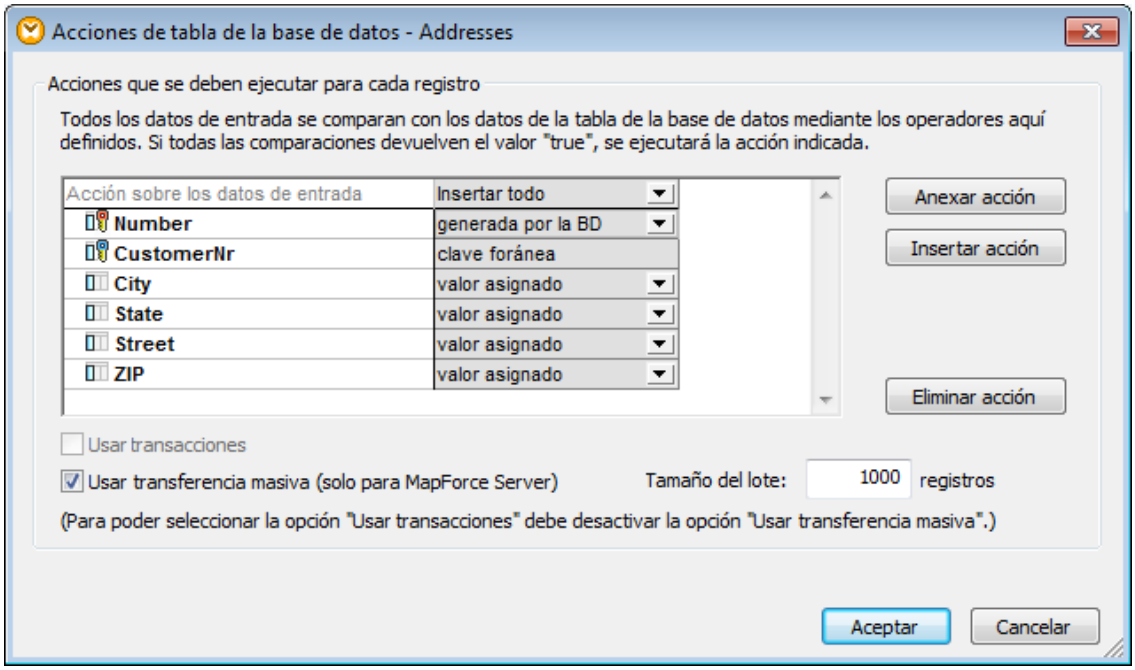
 © 2014 Altova GmbH

instrucción SQL actual se revierte y se detiene la ejecución.

Nota: el cuadro de diálogo "Excepción de transacción" solamente aparece cuando la transformación se lleva a cabo de forma **interactiva**. El **código generado** realiza una reversión (y se detiene) cuando se encuentra el primer error.

Usar transferencia masiva

La opción *Usar transferencia masiva* sirve para cargar datos de un archivo de datos (`txt`, `csv`, `dat`, etc.) en una tabla de BD. Gracias a esta opción podrá acelerar significativamente el proceso de inserción de datos porque solamente es necesario ejecutar una instrucción. No obstante, tenga en cuenta que esta opción solamente está disponible si el destino elegido es el motor de ejecución integrado (`BUILTIN`) y la ejecución se ejecuta con MapForce Server.



Esta opción solamente se puede seleccionar si está desactivada la casilla *Usar transacciones.* Para más información consulte el apartado Transferencias masivas (MapForce Server).

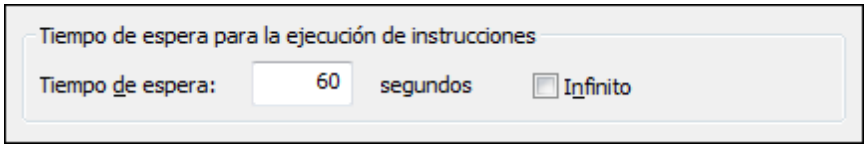
Tiempos de espera de ejecución

El tiempo de espera de la ejecución define cuánto pueden tardar las instrucciones en ejecutarse por cada base de datos. Y, si sobrepasan el tiempo de espera, las instrucciones se anulan. Todos los productos de Altova que permiten acceder a bases de datos ofrecen una opción para definir un tiempo de espera de ejecución a la hora de acceder a bases de datos. La opción surte efecto en la pestaña *Resultados* y es compatible con código C#, Java y C++.

Cuando se usa una BD como componente de destino, algunos factores pueden provocar que se supere el tiempo de espera de ejecución, como por ejemplo el tráfico y la disponibilidad del servidor, la presencia de desencadenadores de larga ejecución, etc. Por eso MapForce ofrece una opción para definir el tiempo de espera de la ejecución. Una vez superado este tiempo de espera de ejecución, MapForce cierra la conexión con la BD.

Para definir el tiempo de espera de la ejecución:

1. Haga clic con el botón secundario en un componente de BD y elija el comando **Propiedades**.
2. Haga clic en el campo *Tiempo de espera* y escriba el valor correspondiente.



Nota: el valor predeterminado es `60` segundos. La opción *Infinito* equivale a muchos años.

Transferencias masivas (MapForce Server)

La opción *Usar transferencia masiva* sirve para insertar datos de un componente de MapForce (`txt`, `csv`, `dat`, etc.) en una tabla de BD a gran velocidad. Con esta opción puede acelerar significativamente el proceso de inserción de datos porque basta con ejecutar una sola instrucción.

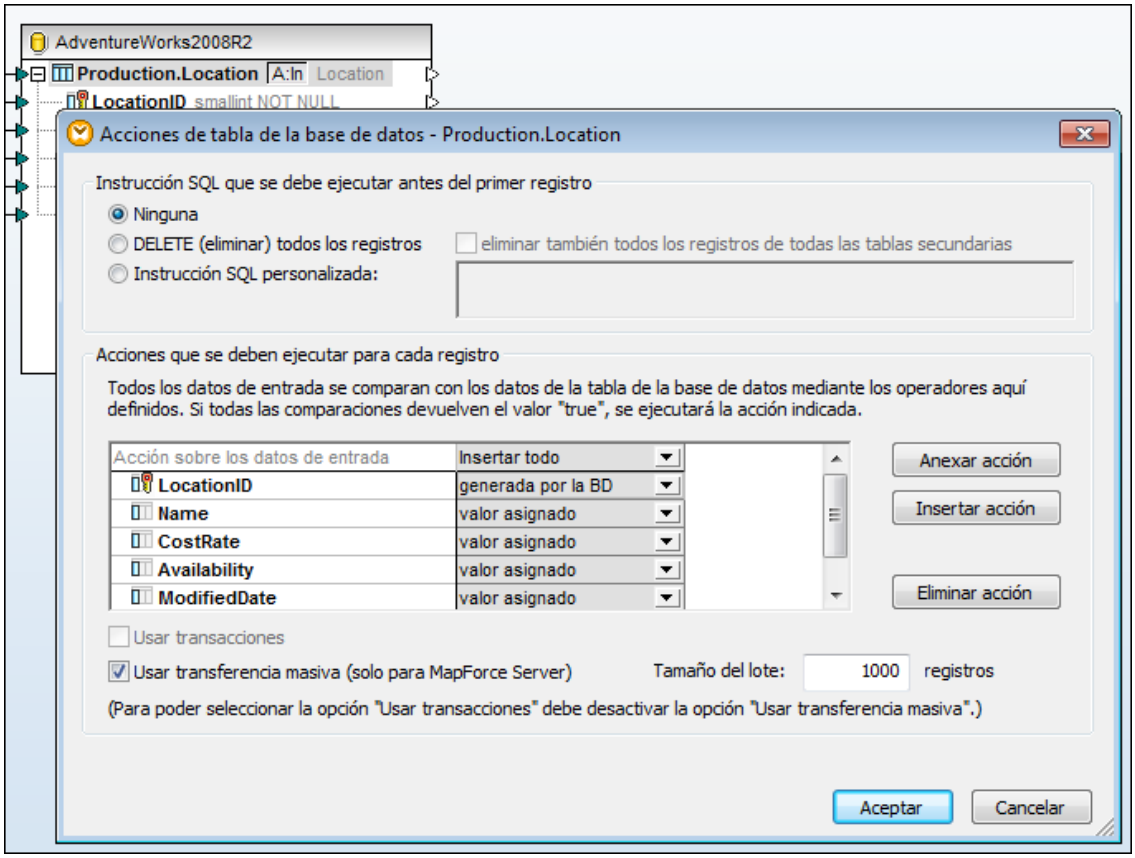
No obstante, es necesario cumplir algunos requisitos para poder usar transferencias masivas:

- Solamente se pueden usar con el motor de ejecución integrado (`BUILT-IN`) de MapForce.
- Solamente se pueden ejecutar en MapForce Server (es necesario proporcionar/implementar la asignación de datos con el comando **Compilar en archivo de ejecución de MapForce Server**).
- Si la asignación se implementa en FlowForce Server, con el comando **Implementar en FlowForce Server**, después puede definir un trabajo que ejecute la asignación en función de los desencadenadores definidos en FlowForce. Después se ejecuta el trabajo en MapForce Server.
- La licencia de MapForce Server no puede estar limitada a ejecución de subprocesos (*single thread execution*) en un equipo multinúcleo.
- Solo funcionan con acciones de tabla `Insertar todo`.
- La tabla en la que se deben cargar datos de forma masiva debe ser una tabla *leaf*, es decir, una tabla que está en el nivel más inferior de la jerarquía de la BD.
- En la asignación no debería haber tablas, vistas ni procedimientos almacenados relacionados que hagan referencia a la tabla.

MapForce genera una instrucción de transferencia masiva de forma interna que permite al servidor de la BD rellenar la tabla con el formato de archivo reconocido y ejecutar la instrucción de transferencia masiva una vez.

La opción *Usar transferencia masiva* está disponible en el cuadro de diálogo "Acciones de tabla de la base de datos", que aparece al hacer clic en el icono de acción de tabla (p. ej.) situado junto al nodo pertinente del componente de BD de destino (*imagen siguiente*). El campo *Tamaño del lote* define el número de registros que deben insertarse por acción.

 © 2014 Altova GmbH



Las acciones `Insertar todo` masivas son compatibles con los controladores que admiten el uso de inserciones masivas en condiciones WHERE. Tenga en cuenta que si selecciona la opción *Usar transferencia masiva (solo para MapForce Server)* la opción *Usar transacciones* se desactiva automáticamente. Para poder habilitar el procesamiento de transacciones es necesario desactivar la casilla *Usar transferencia masiva.*

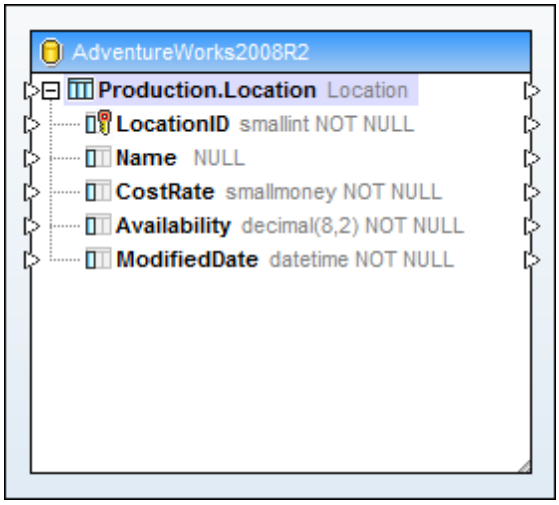
SQL Server es la única base de datos compatible con conexiones ADO, ODBC y JDBC. Las demás bases de datos con compatibles con conexiones ODBC.

En el siguiente ejemplo se usa MS SQL 2008 y la base de datos **`AdventureWorks`** que se puede descargar del sitio web de CodePlex. El objetivo del ejemplo es explicar cómo definir transferencias masivas en MapForce.

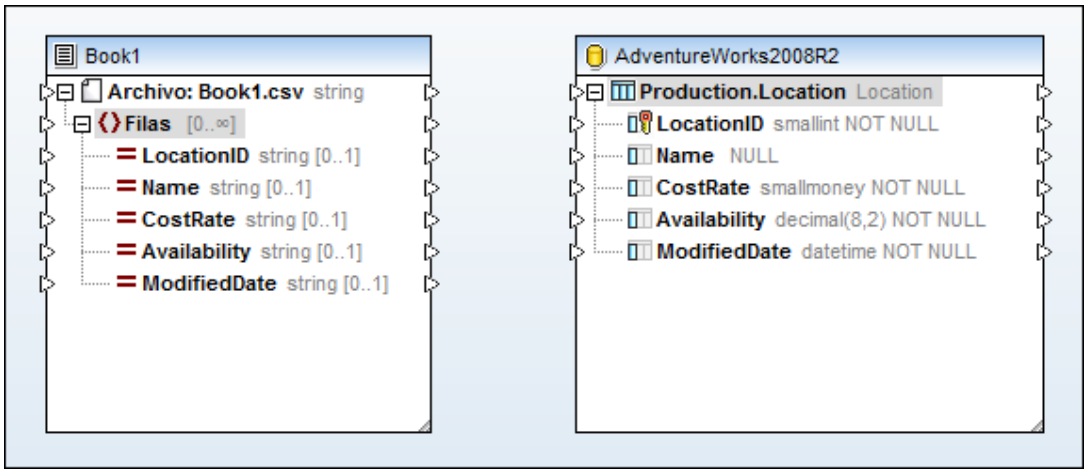
Para definir una inserción masiva:

Para empezar compruebe que está seleccionado el motor de ejecución integrado de MapForce (BUILT IN).

1. Inserte un componente de base de datos en una asignación (con el comando **Insertar | Base de datos**). Para este ejemplo insertamos la tabla **`Production.Location`**.

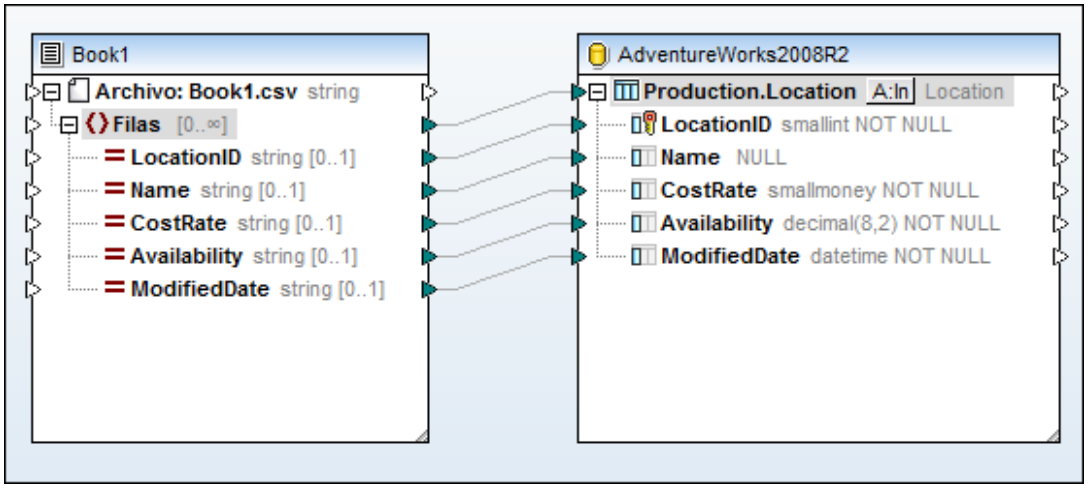


2. Inserte el archivo de datos que quiere usar como origen para la inserción masiva (p. ej. un archivo de texto). Compruebe que los tipos de datos de los componentes de origen y destino son idénticos.



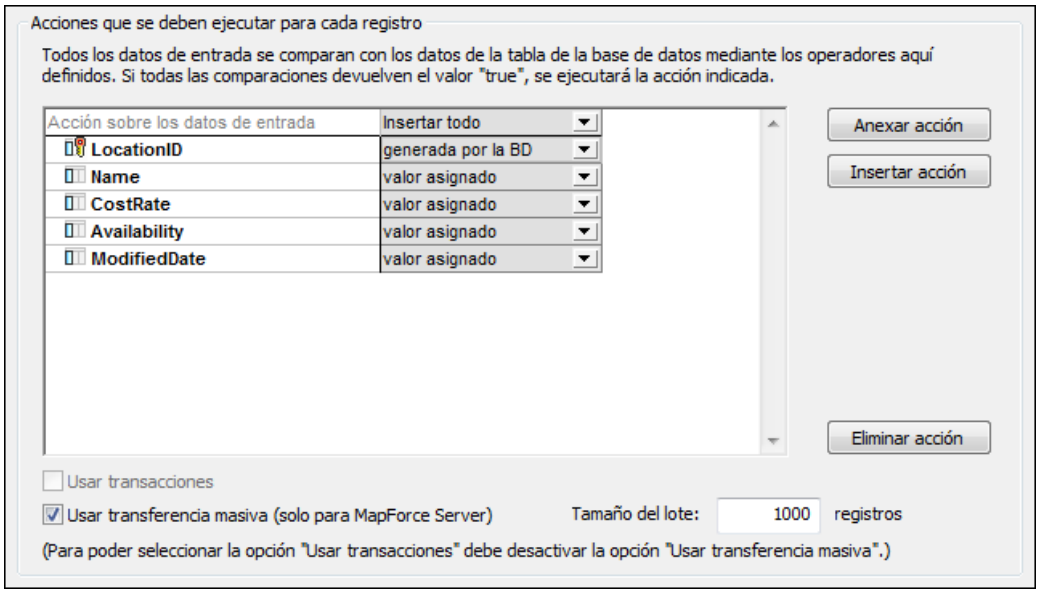
3. Conecte el nodo **`Filas`** con el nombre de la tabla (**`Production.Location`**) del componente de base de datos.
   Los demás elementos se conectan automáticamente (si está activa la opción **Conexión automática de secundarios equivalentes**).

 © 2014 Altova GmbH



Observe que el botón de acciones de base de datos aparece a la derecha del nombre de la tabla.

4. Haga clic en el botón de acciones de base de datos A:In, marque la casilla *Usar transferencia masiva* y haga clic en **Aceptar** para confirmar.



La inserción masiva ya está definida en MapForce. El siguiente paso es compilar la asignación para poder ejecutarla.

Objetivo: Compilar una asignación de datos en un archivo de ejecución de MapForce Server para poder ejecutarlo con MapForce Server.

Hay dos maneras de compilar una asignación de datos en un archivo de ejecución de MapForce Server:

Para compilar una asignación desde la línea de comandos de MapForce:

Ejecute MapForce y especifique el archivo de asignación y de datos y la opción `/COMPILE` de la línea de comandos. El archivo de ejecución de MapForce Server se crea en el directorio donde está el archivo de asignación de datos.

Para compilar una asignación desde la interfaz gráfica de MapForce:

1. Abra una asignación en MapForce (p. ej. `miAsignación.mfd`.)
2. Seleccione la opción de menú **Archivo | Compilar en archivo de ejecución de MapForce Server**.
3. Seleccione la carpeta en la que desea guardar el archivo `.mfx` y, si quiere, cambie el nombre de archivo.
4. Haga clic en **Guardar**.

El archivo de ejecución de MapForce Server `MiAsignación.mfx` se genera en la ubicación elegida.

Para ejecutar el archivo de ejecución de MapForce Server:

Consulte la documentación de MapForce Server sobre el uso del comando `run` de la línea de comandos.

Nota: para ejecutar el comando **`run`** correctamente en MapForce Server, la casilla ***Limit to single thread execution*** *(limitar a ejecución de subprocesos)* debe estar **desactivada** en la pestaña **Server Management** de Altova LicenseServer.

Si el comando **`run`** se ejecuta correctamente en MapForce Server, la inserción masiva habrá insertado los registros especificados.

 © 2014 Altova GmbH

10.2.6 Comparación de características de BD

Este apartado incluye información sobre las funciones de asignación de datos que MapForce ofrece para cada tipo de base de datos compatible.

Cada uno de los subapartados siguientes incluye:

- Información general sobre la base de datos, como problemas de autenticación, etc.
- Tipos de conexión compatibles
- Funciones SQL compatibles para esquemas, instrucciones de combinación, etc.
- Métodos de transacción compatibles

MS Access

	MS Access	Compatibilidad	Notas
General:			
	motor de BD como servicio	No	implementado en el proveedor OLEDB o controlador ODBC
	autenticación propia	Sí	la autenticación es posible
	autenticación de confianza	No	
Conexión:			
	OLE DB	Sí	
	problemas con la cadena de conexión OLE DB	No	
	ODBC	Sí	
	problemas con la cadena de conexión ODBC	DBQ	debe aplicarse
	problemas con la cadena de conexión ODBC	DATABASE	no se puede aplicar
	JDBC	Sí	a través de ODBC
	problemas con URL JDBC	No	
	uso de instrucciones iniciales	No	
	uso de instrucciones finales	No	
SQL:			
	DB-object-name qualification	[] o ""	
	compatibilidad con esquemas de BD	No	no compatible con Access
	identidad	Sí	
	MF: valor de identidad leído	@@IDENTITY	
	sub select	Sí	
	JOIN	No	compatibilidad limitada
	MF: función upper	Ucase(..)	

Ejecución de			

instrucciones SQL:			
	ejecutar múltiples instrucciones en una	No	
	separador de comandos	--	
	gestión especial de errores	No	
	recuperar tipos de parámetros	?	
Transacciones:			
	transacciones planas	Sí	
	iniciar transacción plana mediante ejecución de comando SQL	No	
	transacciones anidadas	No	no compatible con Access
	establecer aislamiento de transacción	No	no compatible con Access
	MF: comenzar transacción	llamada API	
	MF: confirmar transacción	llamada API	
	MF: revertir transacción	llamada API	
	MF: establecer punto de almacenamiento	--	
	MF: reversión a punto de almacenamiento	--	
	uso de instrucciones iniciales	No	

La semántica del tipo de datos `datetime` es diferente en XML Schema y en Access. En XML Schema la fecha es obligatoria, mientras que en Access es opcional. Puesto que MapForce usa tipos de datos XML Schema internamente, se evita que el usuario pueda asignar campos `xs:time` a un campo `datetime` de Access directamente.

La solución es usar la función **`datetime-from-date-and-time`** y usar la fecha **`1899-12-30`**. Se trata de la fecha que Access usa internamente para los valores `only-time`.

MS SQL Server

	MS SQLServer	Compatibilidad	Notas
General:			
	motor de BD como servicio	Sí	
	autenticación propia	Sí	
	autenticación de confianza	Sí	
Conexión:			
	OLE DB	Sí	
	ODBC	Sí	
	problemas con la cadena de conexión ODBC	Select Method=Cursor	debe aplicarse

 © 2014 Altova GmbH

	JDBC	Sí	
	uso de instrucciones iniciales	No	
	uso de instrucciones finales	No	
SQL:			
	DB-object-name qualification	[] o ""	
	compatibilidad con esquemas de BD	Sí	
	identidad	Sí	
	MF: valor de identidad leído	@@IDENTITY	
	sub select	Sí	
	JOIN	Sí	
	MF: función upper	UPPER()	

Ejecución de instrucciones SQL:			
	ejecutar múltiples instrucciones en una	Sí	
	separador de comandos	';' o 'GO'	
	gestión especial de errores	No	
	recuperar tipos de parámetros	Sí	con límites
	problemas al usar ?	Sí	tipo de datos DATETIME incompatible cuando se usa ODBC
Transacciones:			
	transacciones planas	Sí	
	iniciar transacción plana mediante ejecución de comando SQL	Sí	obligatorio cuando se usan transacciones anidadas. No se puede mezclar control de transacciones por API con comandos de transacciones SQL
	transacciones anidadas	Sí	a través de SAVEPOINT
	establecer aislamiento de transacción	Sí	
	MF: comenzar transacción	BEGIN TRANSACTION	
	MF: confirmar transacción	COMMIT TRANSACTION	
	MF: revertir transacción	ROLLBACK TRANSACTION	
	MF: establecer punto de almacenamiento	SAVEPOINT	
	MF: reversión a punto de almacenamiento	ROLLBACK TO	
	uso de instrucciones iniciales	No	

Oracle

	Oracle	Compatibilidad	Notas
General:			
	motor de BD como servicio	Sí	
	autenticación propia	Sí	
	autenticación de confianza	No	
Conexión:			
	OLE DB	No	no compatible con MapForce
	ODBC	Sí	
	problemas con la cadena de conexión ODBC	DATABASE	no se debe aplicar
	JDBC	Sí	
	uso de instrucciones iniciales	No	
	uso de instrucciones finales	No	
SQL:			
	DB-object-name qualification	""	
	compatibilidad con esquemas de BD	Sí	
	identidad	No	debe usar disparadores
	MF: valor de identidad leído	incompatible	
	sub select	Sí	
	JOIN	Sí	
	MF: función upper	UPPER()	

Ejecución de instrucciones SQL:			
	ejecutar múltiples instrucciones en una	No	
	separador de comandos	--	
	gestión especial de errores	No	
	recuperar tipos de parámetros		
	problemas al usar ?	No	
Transacciones:			
	transacciones planas	Sí	
	iniciar transacción plana mediante ejecución de comando SQL	No	
	transacciones anidadas	Sí	a través de SAVEPOINT
	establecer aislamiento de transacción	Sí	
	MF: comenzar transacción	llamada API	

 © 2014 Altova GmbH

	MF: confirmar transacción	llamada API	
	MF: revertir transacción	llamada API	
	MF: establecer punto de almacenamiento	SAVEPOINT	
	MF: reversión a punto de almacenamiento	ROLLBACK TO SAVEPOINT	
	uso de instrucciones iniciales	No	

MySQL

	MySQL	Compatibilidad	Notas
General:			
	motor de BD como servicio	Sí	
	autenticación propia	Sí	
	autenticación de confianza	No	
	problemas especiales	TYPE=INNODB	para tablas cuando se usan relaciones, transacciones
Conexión:			
	OLE DB	No	incompatible con MapForce
	ODBC	Sí	
	JDBC	Sí	a través de ODBC
	uso de instrucciones iniciales	No	
	uso de instrucciones finales	No	
SQL:			
	DB-object-name qualification	``	
	compatibilidad con esquemas de BD	Sí	
	identidad	Sí	
	MF: valor de identidad leído	@@IDENTITY	
	sub select	No	necesaria la implementación especial para DELETE
	JOIN	Sí	
	MF: función upper	UPPER()	

Ejecución de instrucciones SQL:			
	ejecutar múltiples instrucciones en una	No	
	separador de comandos	--	
	gestión especial de errores	No	

	recuperar tipos de parámetros	Sí	con límites
	problemas especiales al usar ?	No	
Transacciones:			
	transacciones planas	Sí	
	iniciar transacción plana mediante ejecución de comando SQL	No	
	transacciones anidadas	No	MySQL no produce un error y continúa si no existen transacciones anidadas
	establecer aislamiento de transacción	Sí	
	MF: comenzar transacción	llamada API	
	MF: confirmar transacción	llamada API	
	MF: revertir transacción	llamada API	
	MF: establecer punto de almacenamiento	SAVEPOINT	
	MF: reversión a punto de almacenamiento	ROLLBACK TO	
	uso de instrucciones iniciales	SET AUTOCOMMIT=0	

Sybase

Tenga en cuenta que cuando se conecte a una base de datos Sybase a través de JDBC, la instrucción SELECT no puede recuperar datos. Se trata de un problema conocido del controlador. Por ello, rogamos utilice otro controlador para conectarse a la BD.

	Sybase	Compatibilidad	Notas
General:			
	motor de BD como servicio	Sí	
	autenticación propia	Sí	
	autenticación de confianza	No	
Conexión:			
	OLE DB	No	incompatible con MapForce
	ODBC	Sí	
	problemas con la cadena de conexión ODBC	Select Method=Cursor	debe aplicarse
	JDBC	Sí	a través de ODBC
	uso de instrucciones iniciales	No	
	uso de instrucciones finales	No	
SQL:			
	DB-object-name qualification		
	compatibilidad con esquemas de BD	Sí	

 © 2014 Altova GmbH

	identidad	Sí	
	MF: valor de identidad leído	@@IDENTITY	
	sub select	y	
	JOIN	y	
	MF: función upper	UPPER()	

Ejecución de instrucciones SQL:			
	ejecutar múltiples instrucciones en una	Sí	
	separador de comandos	No	
	gestión especial de errores	No	
	recuperar tipos de parámetros		
	problemas especiales al usar ?	Sí	cuando se usa ODBC las constantes de cadena solamente admiten caracteres ASCII-127
Transacciones:			
	transacciones planas	Sí	
	iniciar transacción plana mediante ejecución de comando SQL	incompatible con MAPFORCE	
	transacciones anidadas	No	Sybase no produce un error y continúa si no existen transacciones anidadas
	establecer aislamiento de transacción	Sí	
	MF: comenzar transacción	llamada API	
	MF: confirmar transacción	llamada API	
	MF: revertir transacción	llamada API	
	MF: establecer punto de almacenamiento	SAVE TRANSACTION	
	MF: reversión a punto de almacenamiento	ROLLBACK	
	uso de instrucciones iniciales	No	

Tras definir las relaciones entre las tablas usando los procedimientos Sybase `sp_primarykey` y `sp_foreignkey`, es necesario usar `ALTER TABLE` para añadir una restricción a la tabla que describa la relación de clave foránea y que las relaciones de clave foránea/principal aparezcan en MapForce.

IBM DB2

	IBM DB2	Compatibilidad	Notas

General:			
	motor de BD como servicio	Sí	
	autenticación propia	Sí	usa cuentas de usuario local de Windows
	autenticación de confianza	Sí	ver punto anterior
Conexión:			
	OLE DB	No	Incompatible con MapForce
	ODBC	Sí	
	JDBC	Sí	a través de ODBC
	uso de instrucciones iniciales	No	
	uso de instrucciones finales	No	
SQL:			
	DB-object-name qualification	""	
	compatibilidad con esquemas de BD	Sí	
	identidad	Sí	
	MF: valor de identidad leído	identity_val_local()	
	sub select	Sí	
	JOIN	Sí	
	MF: función upper	UPPER()	

Ejecución de instrucciones SQL:			
	ejecutar múltiples instrucciones en una	No	
	separador de comandos	--	
	gestión especial de errores	Sí	
	recuperar tipos de parámetros	No	
	problemas especiales al usar ?	No	
Transacciones:			
	transacciones planas	Sí	
	iniciar transacción plana mediante ejecución de comando SQL	No	
	transacciones anidadas	No	Incompatible con DB2
	establecer aislamiento de transacción	Sí	
	MF: comenzar transacción	llamada API	
	MF: confirmar transacción	llamada API	
	MF: revertir transacción	llamada API	
	MF: establecer punto de almacenamiento	--	Incompatible con DB2
	MF: reversión a punto de almacenamiento	--	Incompatible con DB2

 © 2014 Altova GmbH

	uso de instrucciones iniciales	No	

10.2.7 Relaciones de BD: cómo conservarlas/descartarlas

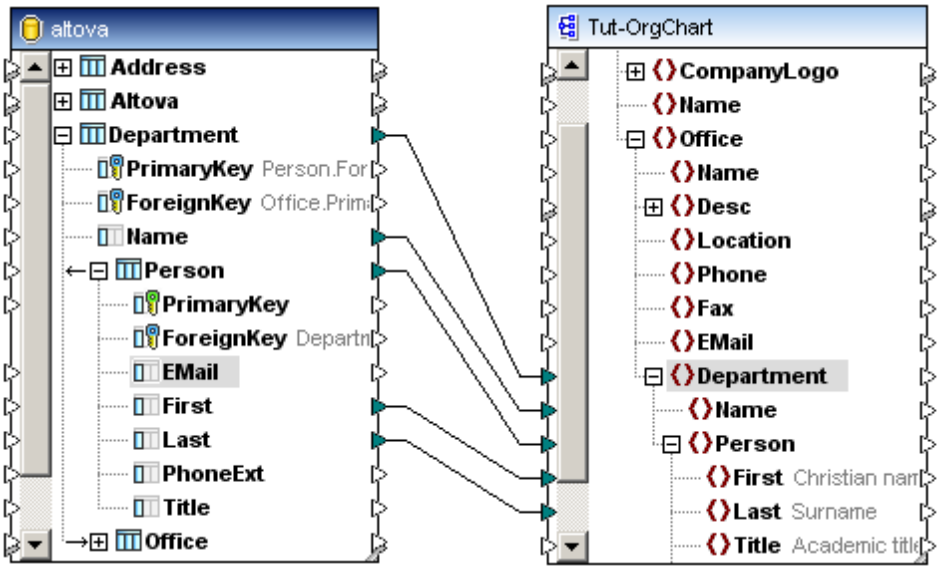
Este apartado describe cómo mantener intactas las relaciones de BD que existen entre las tablas y cómo descartar dichas relaciones de tabla.

Conservar y usar relaciones de base de datos

Para conservar las relaciones existentes entre las tablas de la BD es necesario crear asignaciones de datos entre los campos situados bajo **una** de las tablas **raíz** (p. ej. **`Department`** en la imagen siguiente). La estructura completa de la BD aparece representada de forma jerárquica debajo de cada tabla raíz del componente de base de datos.

Para más información sobre la estructura jerárquica de los componentes de base de datos consulte el apartado Componentes y relaciones de tabla.

- **`Department | Name`** están asignados a los campos situados bajo la tabla raíz **`Department`**.
- **`Person | First`** y **`Person | Last`** están asignados a los campos de la tabla **`Person`**, que está debajo de la tabla raíz **`Department`**.



Resultado de la asignación de la imagen anterior

El archivo de salida incluye el nombre de las personas que trabajan en cada departamento.

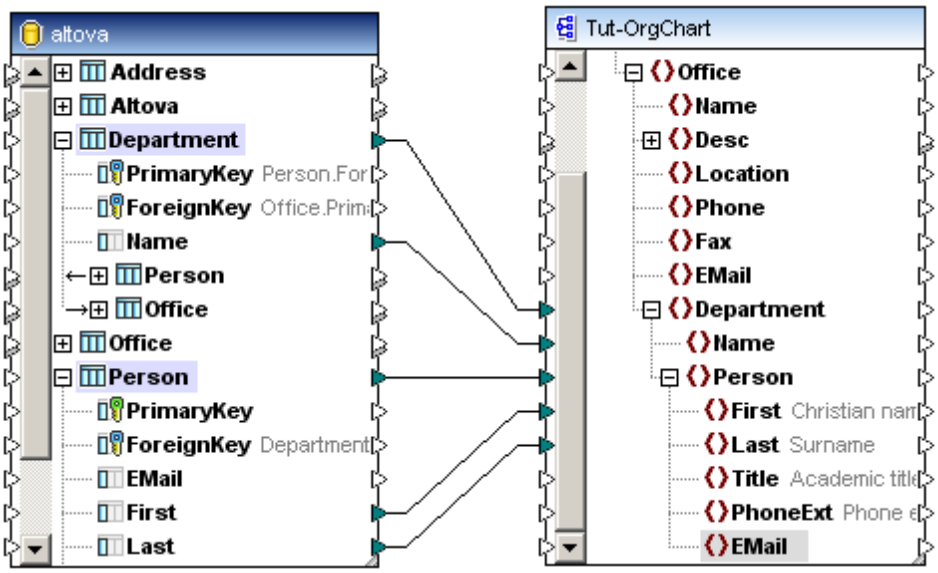
 © 2014 Altova GmbH

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <OrgChart xsi:noNamespaceSchemaLocation="
    C:/DOCUME~1/MYDOCU~1/Altova/MapForce2010/MapForceExamples
    http://www.w3.org/2001/XMLSchema-instance">
3     <Office>
4       <Department>
5         <Name>Administration</Name>
6         <Person>
7           <First>Vernon</First>
8           <Last>Callaby</Last>
9         </Person>
10        <Person>
11          <First>Frank</First>
12          <Last>Further</Last>
13        </Person>
14        <Person>
15          <First>Loby</First>
16          <Last>Matise</Last>
17        </Person>
```

Descartar relaciones de base de datos

Este método requiere que existan asignaciones entre los distintos campos situados bajo diferentes tablas **raíz** del componente de base de datos.

- **`Department | Name`** está asignado a un campo situado bajo la tabla raíz **`Department`**.
- **`Person | First`** y **`Person | Last`** están asignados a campos situados bajo la tabla raíz **`Person`**.



Resultado de la asignación de la imagen anterior

El archivo de salida incluye el nombre de todos las personas de todos los departamentos. Las relaciones de tabla se ignoran y se incluyen todas las personas en cada departamento.

Departamento Administration:

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <OrgChart xsi:noNamespaceSchemaLocation="
    C:/DOCUME~1/MYDOCU~1/Altova/MapForce2010/MapForceExamples
    http://www.w3.org/2001/XMLSchema-instance">
3     <Office>
4       <Department>
5         <Name>Administration</Name>
6         <Person>
7           <First>Vernon</First>
8           <Last>Callaby</Last>
9         </Person>
10        <Person>
11          <First>Frank</First>
12          <Last>Further</Last>
13        </Person>
14        <Person>
15          <First>Loby</First>
16          <Last>Matise</Last>
17        </Person>
```

Departamento Marketing:

```
91        <Department>
92          <Name>Marketing</Name>
93          <Person>
94            <First>Vernon</First>
95            <Last>Callaby</Last>
96          </Person>
97          <Person>
98            <First>Frank</First>
99            <Last>Further</Last>
100         </Person>
```


© 2014 Altova GmbH

10.2.8 Relaciones locales: crear relaciones de BD

En MapForce puede consultar o crear datos de BD relacionados incluso si las relaciones no existen explícitamente en la BD. Puede definir relaciones de clave principal/foránea entre las tablas de un componente sin que las relaciones afecten a las relaciones subyacentes de la BD.

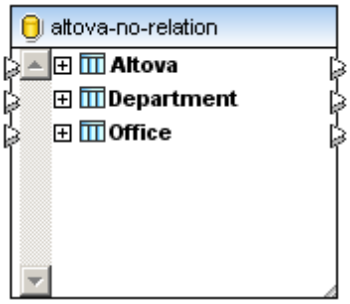
En MapForce estas relaciones se denominan ***relaciones locales***:

- puede usar cualquier campo de la BD como clave principal o foránea
- puede crear relaciones nuevas que no existen en la BD

Puede definir relaciones locales para:

- tablas de la BD
- vistas de la BD
- procedimientos almacenados y funciones
- instrucciones SELECT definidas por el usuario

En este ejemplo usamos la base de datos MS Access `altova-no-relation.mdb`, que es una versión simplificada de la base de datos `altova.mdb`. Las tablas **`Person`** y **`Address`**, así como las relaciones de tabla restantes, se eliminaron de la BD en MS Access.



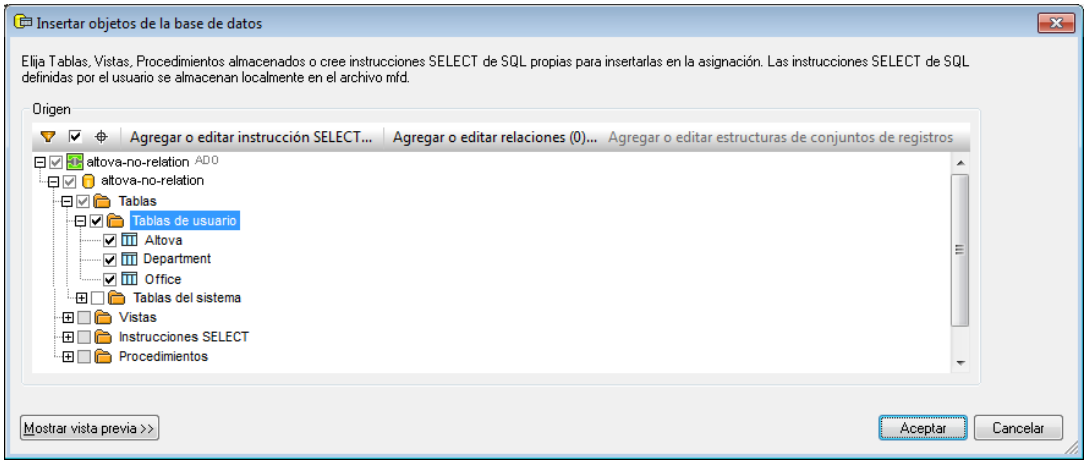
El objetivo de este ejemplo es ver las oficinas de Altova y ver los departamentos de cada una de las oficinas.

Ninguna tabla de la estructura en forma de árbol de `altova-no-relation` tiene tablas secundarias, es decir, todas las tablas están al mismo nivel raíz. El contenido de cada tabla se limita a los campos que contiene. No obstante, en MapForce podemos extraer datos relacionados de la BD incluso si no se definieron relaciones explícitamente.

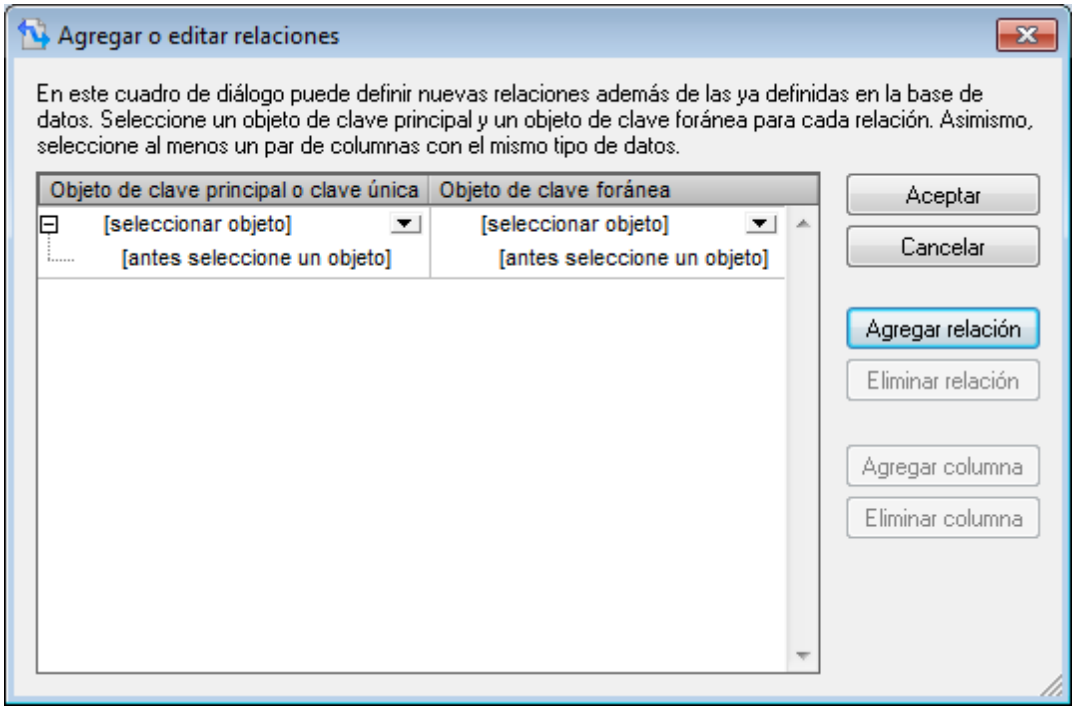
Cómo se crean las relaciones locales

Puede definir las relaciones locales en el momento en que inserta la BD. También puede hacer clic con el botón secundario en el componente de base de datos y seleccionar la opción **Agregar, quitar o editar objetos de la BD** en el menú contextual.

1. Inserte un componente de base de datos.
2. En el asistente para la conexión haga clic en **Microsoft Access** y después en **Siguiente**.
3. Haga clic en **Examinar** y seleccione `altova.mdb`. Aparece el cuadro de diálogo "Insertar objetos de la base de datos".
   Marque la casilla de *Tablas de usuario* para seleccionar las tres tablas.

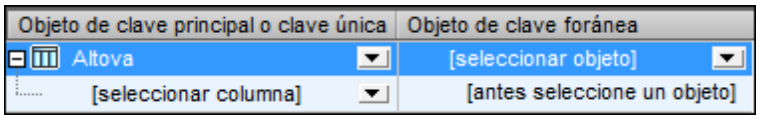


4. Haga clic en el botón **Agregar o editar relaciones** de la barra de herramientas. Aparece el cuadro de diálogo "Agregar o editar relaciones".
5. Haga clic en el botón **Agregar relación**.



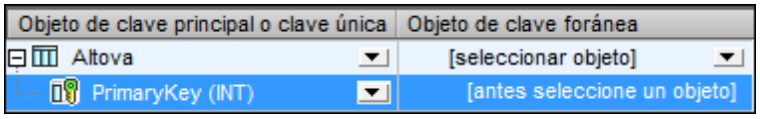
Aparecen dos cuadros combinados donde puede seleccionar las tablas u objetos de la BD para los que desea crear las relaciones. El cuadro combinado de la izquierda es la tabla de clave única/principal. El de la derecha corresponde a la tabla de clave foránea. El objeto de clave única/principal será el objeto primario en MapForce y el objeto de clave foránea se presentará como secundario del objeto primario.

6. Haga clic en el cuadro combinado de la izquierda y seleccione la tabla **`Altova`**.

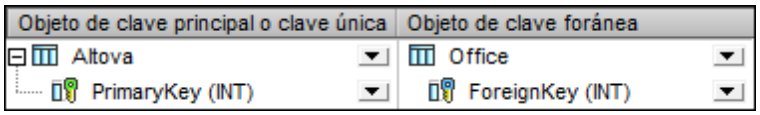


7. Haga clic en el cuadro combinado `[seleccionar columna]` y seleccione la opción

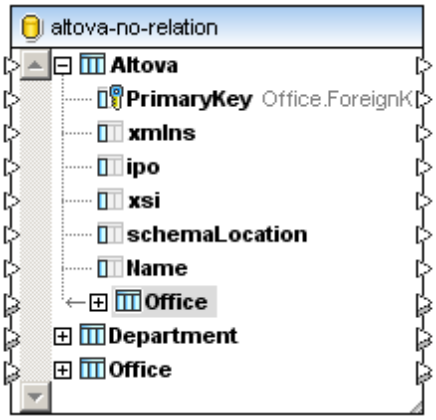
PrimaryKey.



8. En el cuadro combinado de la derecha seleccione la tabla **`Office`** y la columna **`ForeignKey`**.

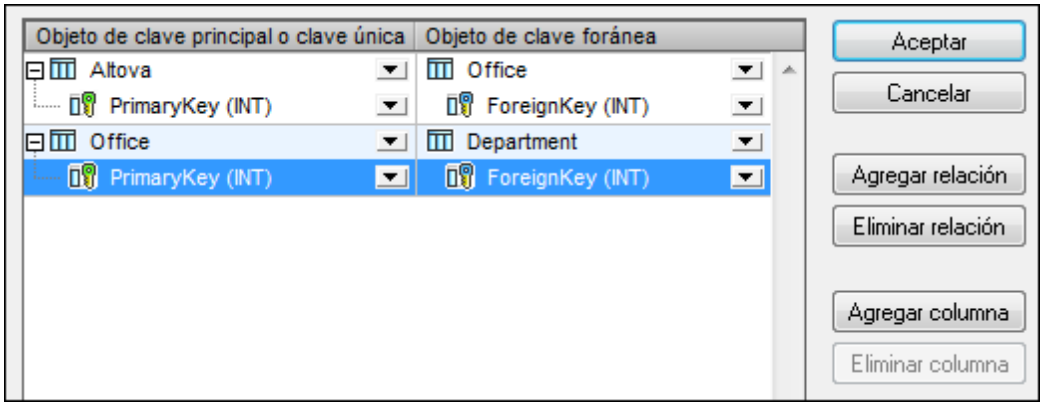


9. Haga clic en el botón **Aceptar** para terminar de definir la relación local. Haga clic en **Aceptar** otra vez para insertar la base de datos en el panel **Asignación**.



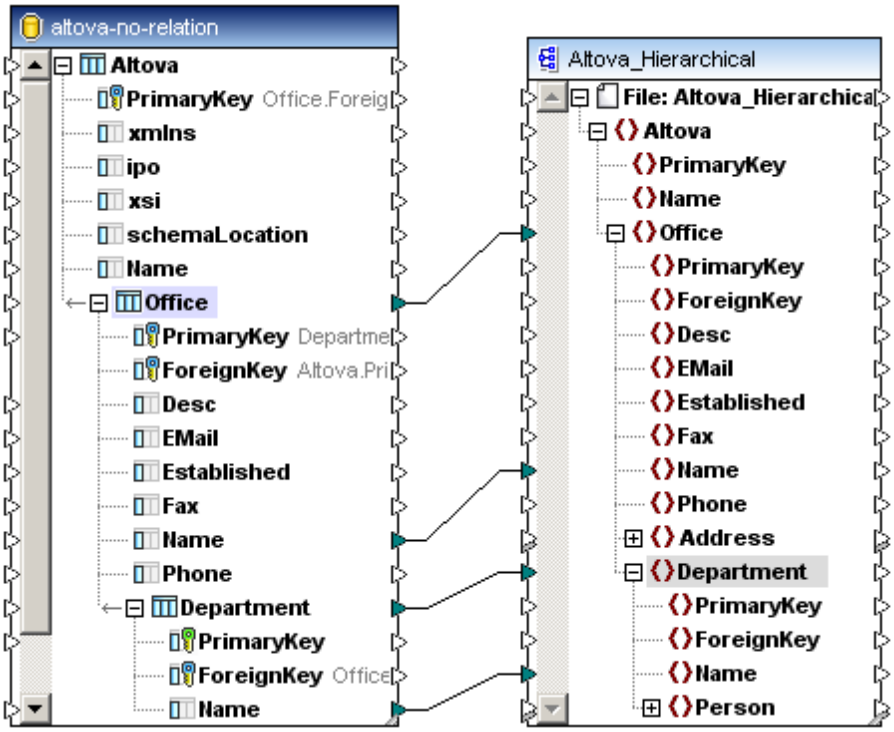
Al hacer clic en el icono + de la tabla **`Altova`** podemos ver la relación que existe entre las tablas **`Altova`** y **`Office`**. La tabla **`Office`** aparece como tabla relacionada bajo la tabla Altova y tiene su propio icono +.
Use el mismo método para crear una relación entre las tablas **`Office`** y **`Department`**.

10. Haga clic con el botón secundario en el componente de base de datos y en el menú contextual seleccione **Agregar, quitar o editar objetos de la BD**.
En el cuadro de diálogo que aparece haga clic en el botón **Agregar o editar relaciones** de la barra de herramientas.



Cuando cree las asignaciones de datos, es importante recordar que para conservar las relaciones existentes entre las tablas es necesario conectar las tablas situadas bajo una de las tablas raíz

(en este caso, **`Altova`**).



Tras definir las asignaciones de datos que aparecen en la imagen anterior, abra el panel **Resultados** para ver el resultado inmediatamente. Si el lenguaje elegido es XSLT, XSLT2 o XQuery, no se puede generar una vista previa de los datos de la BD. En ese caso se genera un mensaje de error y el componente de BD aparece atenuado.

Por cada elemento **`Office`**, el resultado de la asignación incluye el nombre de la oficina y después todos los departamentos de la oficina

```
<Altova xsi:noNamespaceSchemaLocation="C:/DOCUME~1/MYDOCU~1/Altov
  <Office>
    <Name>Nanonull, Inc.</Name>
    <Department>
      <Name>Administration</Name>
    </Department>
    <Department>
      <Name>Marketing</Name>
    </Department>
    <Department>
      <Name>Engineering</Name>
    </Department>
    <Department>
      <Name>IT &amp; Technical Support</Name>
    </Department>
  </Office>
  <Office>
    <Name>Nanonull Partners, Inc.</Name>
    <Department>
      <Name>Administration</Name>
```

 © 2014 Altova GmbH

10.2.9 Asignación de datos de BD de gran tamaño

Cuando utilice bases de datos con muchas tablas, MapForce muestra todas las relaciones existentes entre las tablas importadas de toda la base de datos. Esto se debe a que la aplicación no puede decidir automáticamente qué tablas se usarán en el proceso de asignación y cuáles no, por lo que intenta abarcar todas las posibilidades.

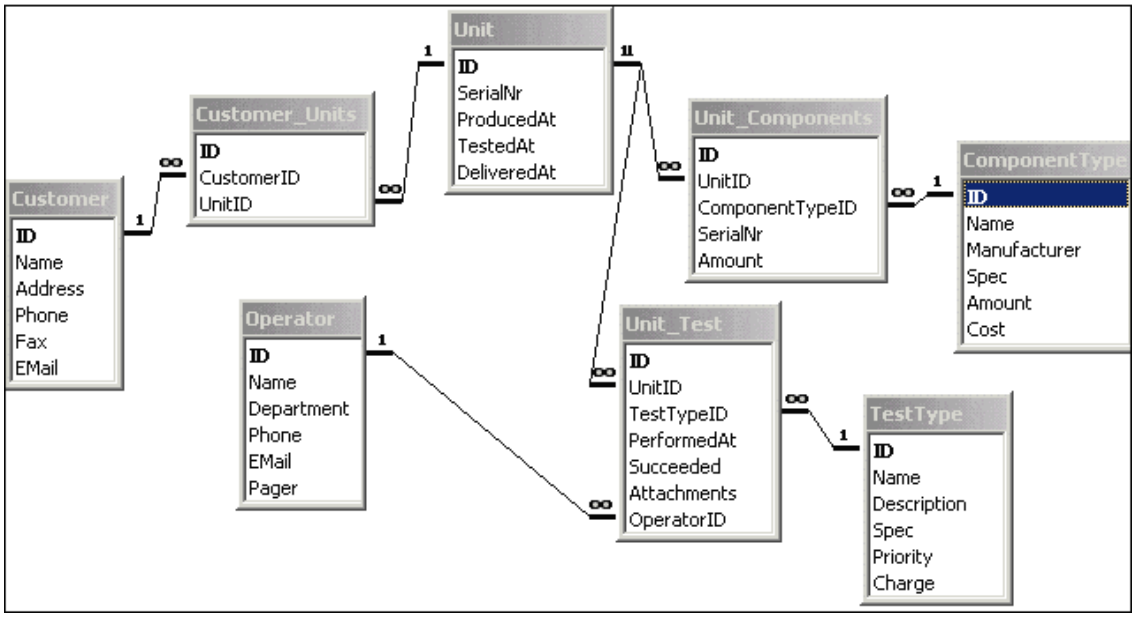
Sin embargo, esto puede dar lugar a estructuras en forma de árbol demasiado grandes si todas las tablas se seleccionan en un solo componente de base de datos. No obstante, se pueden crear varios componentes de base de datos a partir de la misma BD que usen/importen solamente aquellas tablas necesarias para el proceso de asignación. Este método también permite un proceso de asignación más intuitivo.

Ejemplo:
Imaginemos que en una compañía de fabricación se montan varios componentes para producir unidades definidas por el cliente. Antes del reparto, las unidades pasan por una prueba de unidad y todos los resultados se almacenan en una base de datos.

En un momento determinado de la fase de pruebas de prototipos se descubre que un lote de los componentes es defectuoso y dicho lote debe retirarse. El objetivo de la asignación es generar una lista con todos los clientes a quienes afecta la retirada del producto para enviarles una carta.

En este caso la asignación define que:
Por cada **`ComponentType`** de nombre = "**`Prototype`**" y por cada **`Manufacturer`** de nombre = "**`Noname`**",
seleccionar todos los **`Customer`** relacionados y sus datos.

La imagen siguiente muestra el diagrama de relación de este ejemplo de base de datos:



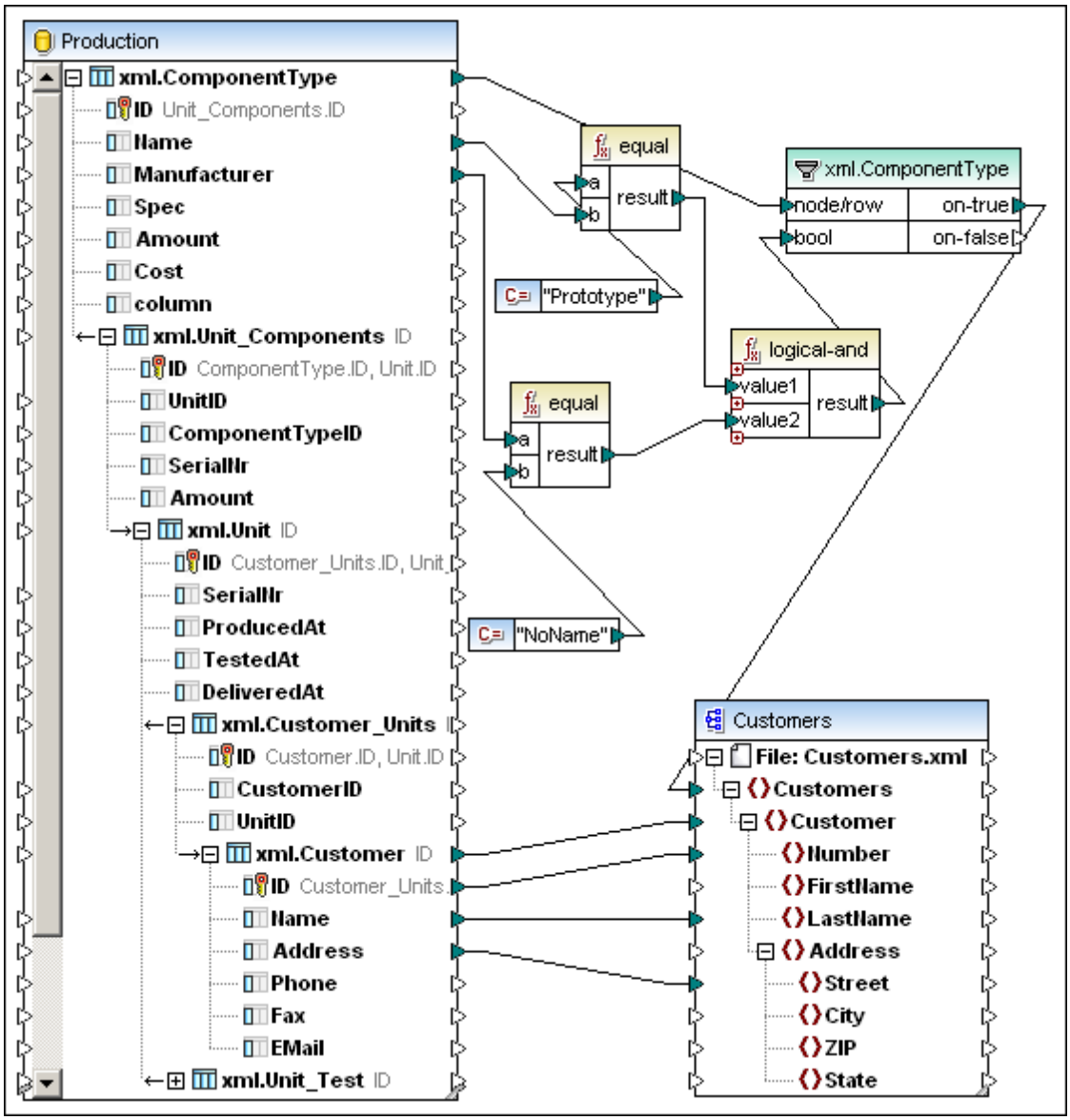
En la carpeta `MapForceExamples` encontrará el script SQL `CreateProductionsTables.sql`, con el que puede crear esta base de datos en SQL Server 2008.

Importar toda la base de datos

Opción nº1:
Importar toda la base de datos, es decir, todas las tablas que contiene. En la imagen siguiente, el componente de base de datos **Production** contiene todas las tablas de la BD y cada tabla aparece como tabla raíz junto con sus tablas relacionadas.

Si usamos la tabla **`ComponentType`** como tabla raíz, la asignación filtra los datos según estos criterios:

- el componente cuyo **`name`** es **`Prototype`** Y
- el **`Manufacturer`** de nombre **`NoName`** junto con
- el **`Customer ID`** correspondiente y los datos de dirección (**`address`**)



 © 2014 Altova GmbH

Importar solo una parte de la base de datos

Opción nº2:
Importar solamente las tablas necesarias para extraer la información que nos interesa, es decir, para recuperar:

- todas las unidades defectuosas
- todos los clientes a los que se suministraron dichas unidades

Inserte dos componentes de base de datos (basados en la misma BD), cada uno de ellos con un conjunto de tablas distinto.

En el componente 1 inserte estas tablas:

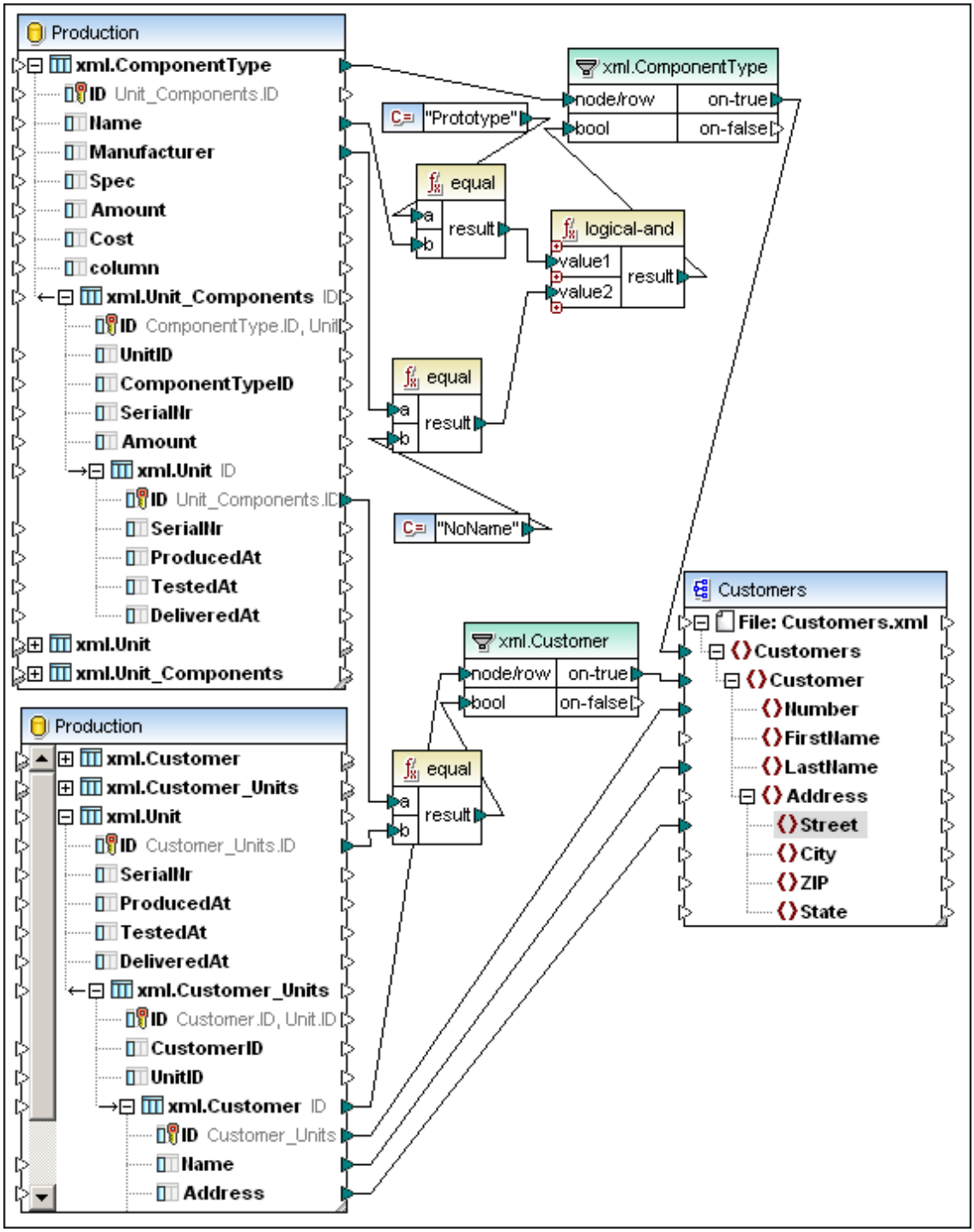
- **`ComponentType`**
- **`Unit_Components`**
- **`Unit`**

En el componente 2 inserte estas tablas:

- **`Unit`**
- **`Customer_Units`**
- **`Customer`**

Proceso de asignación:

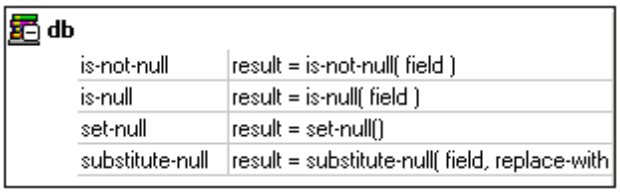
- filtre los componentes cuyo **`Name`** es **`Prototype`** y el **`Manufacterer`** de nombre **`NoName`** (Componente 1)
- use la función **`equal`** para comparar el **`UnitID`** del componente 1 con el **`UnitID`** del componente 2
- si los **`UnitID`** son iguales, use el componente de filtrado para pasar los datos de **`Customer`** del componente 2 al archivo XML de destino **Customers**.

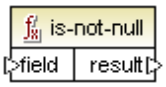


 © 2014 Altova GmbH

10.2.10 Funciones de procesamiento de valores NULL

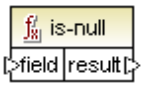
La biblioteca **db** de MapForce incluye varias funciones de procesamiento de valores NULL.





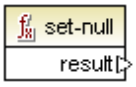
is-not-null

Si el campo es NULL, la función devuelve **`false`**. De lo contrario devuelve **`true`**.



is-null

Si el campo es NULL, la función devuelve **`true`**. De lo contrario devuelve **`false`**.

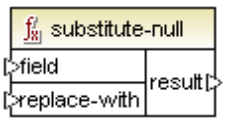


set-null

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:

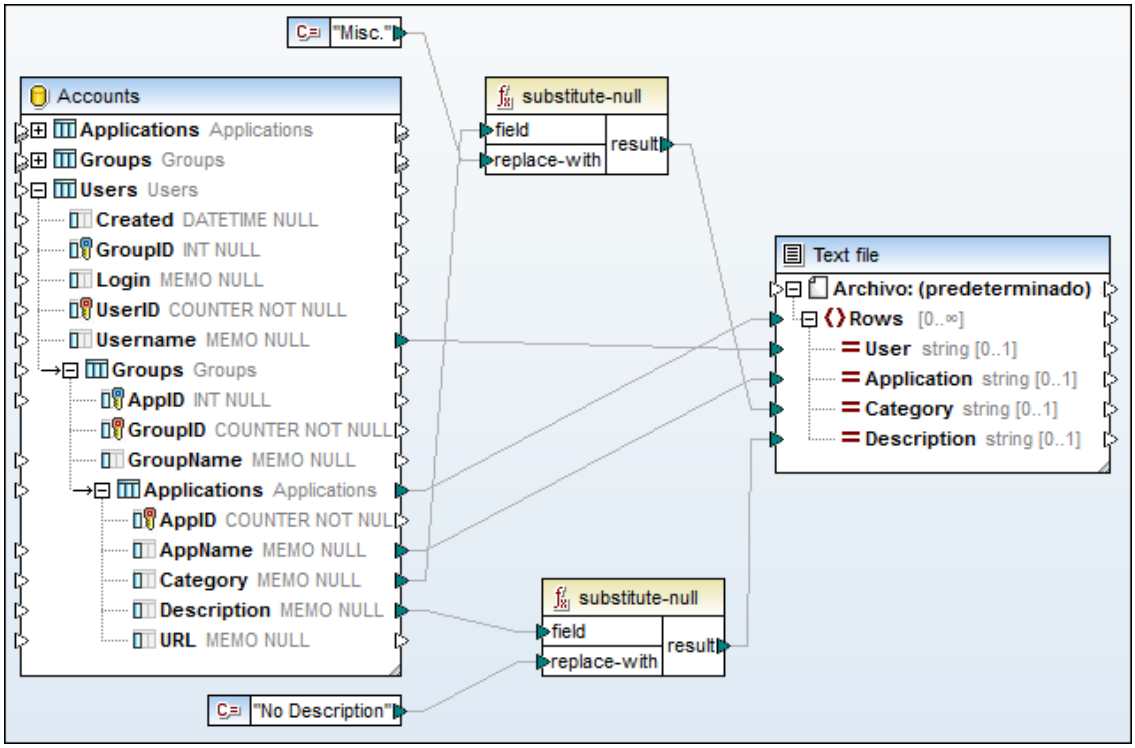
- Si conecta **`set-null`** a otra función, no se llama a la otra función en absoluto. Si conecta **`set-null`** a una función de secuencia (p. ej. **`count`**), se llama a la función con una secuencia vacía.
- Si conecta **`set-null`** a funciones especiales, los filtros y las condiciones **`IF-Else`** funcionan normalmente y los campos se configuran como NULL.
- Si conecta **`set-null`** a un elemento **`simpleType`**, el elemento no se crea en el componente de destino.
- No está permitido conectar **`set-null`** a un elemento **`complexType`**, a una tabla ni a una fila. Cuando esto ocurre aparece un error de validación.



substitute-null

Se utiliza 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`.

 © 2014 Altova GmbH

10.2.11 Componente WHERE/ORDER de SQL

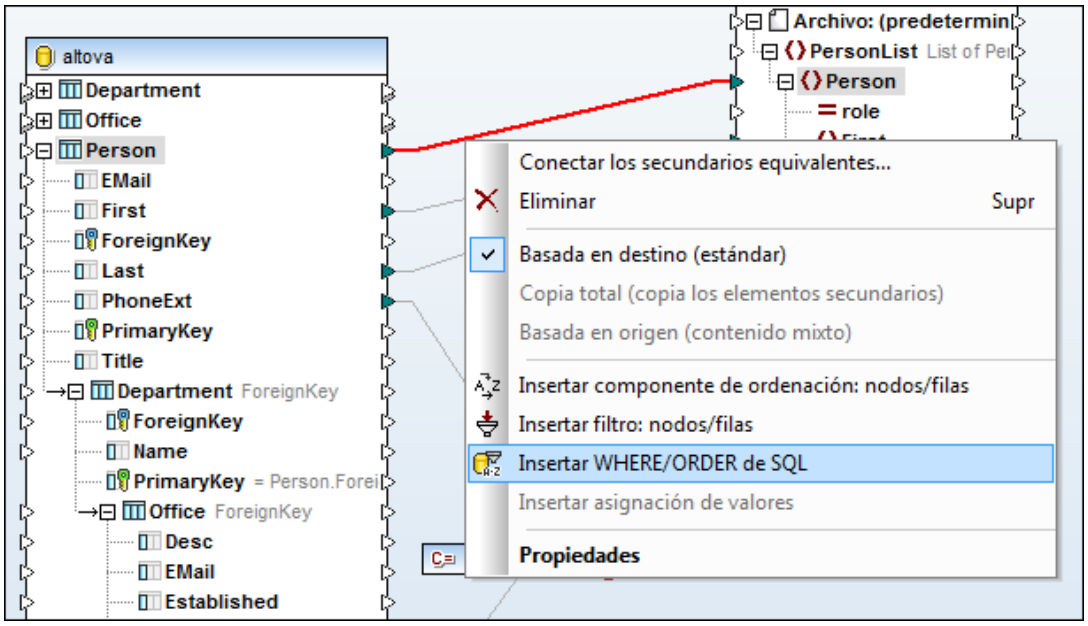
En MapForce puede filtrar y ordenar datos de BD con el componente WHERE/ORDER de SQL. En este apartado usamos el archivo de ejemplo `DB_PhoneList.mfd` , disponible en la carpeta `...\MapForceExamples`.

El componente WHERE/ORDER de SQL consta de varios elementos:

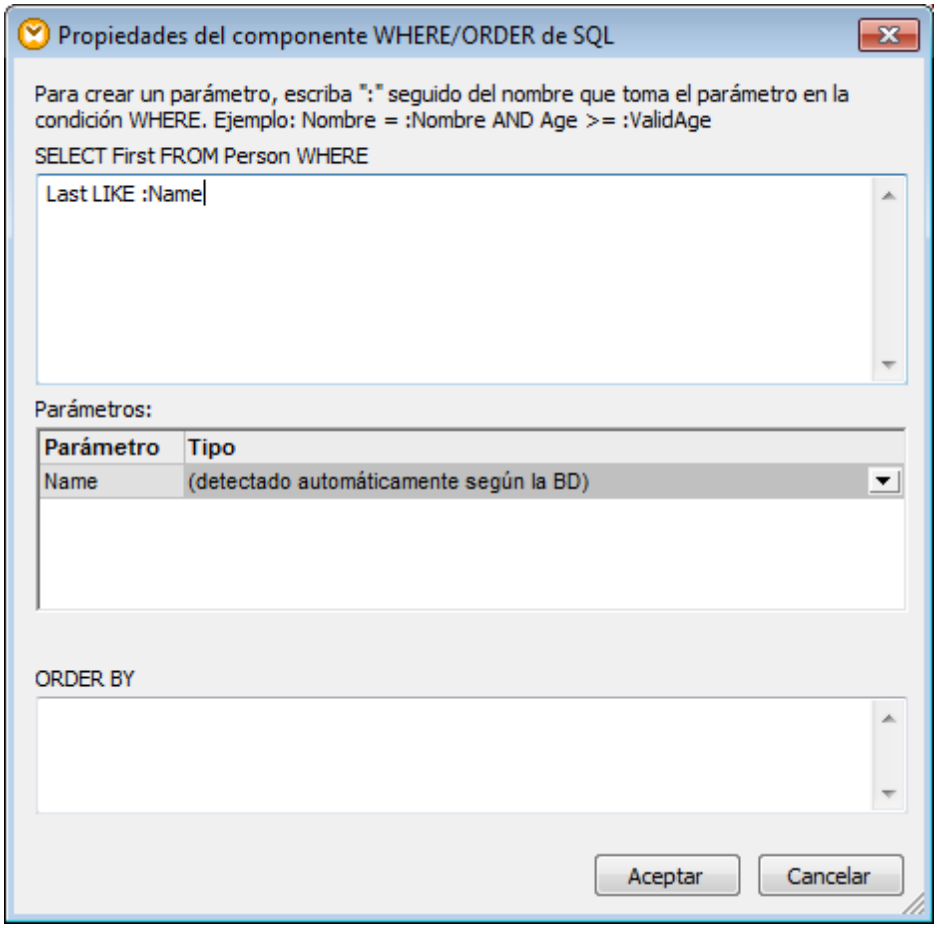
- la instrucción **SELECT** que se genera automáticamente cuando el componente se conecta a una tabla o campo de una BD.
- la cláusula **WHERE** que se introduce manualmente en el cuadro de texto SELECT. Tenga en cuenta que las claves foráneas se incluyen automáticamente en la condición WHERE.
- la cláusula **ORDER BY**

Para insertar un componente WHERE/ORDER de SQL:

1. Haga clic con el botón secundario en el conector existente entre la tabla y el nodo de destino.



2. Seleccione el comando **Insertar WHERE/ORDER de SQL** del menú contextual.
   Se abre el cuadro de diálogo "Propiedades del componente WHERE/ORDER de SQL".
3. Haga clic en el primer cuadro de texto y escriba la consulta WHERE.
   La instrucción SELECT que aparece justo encima del primer cuadro de texto se genera automáticamente cuando se crea la conexión entre la tabla/el campo y el icono de entrada **`table/field`** del componente WHERE/ORDER de SQL.

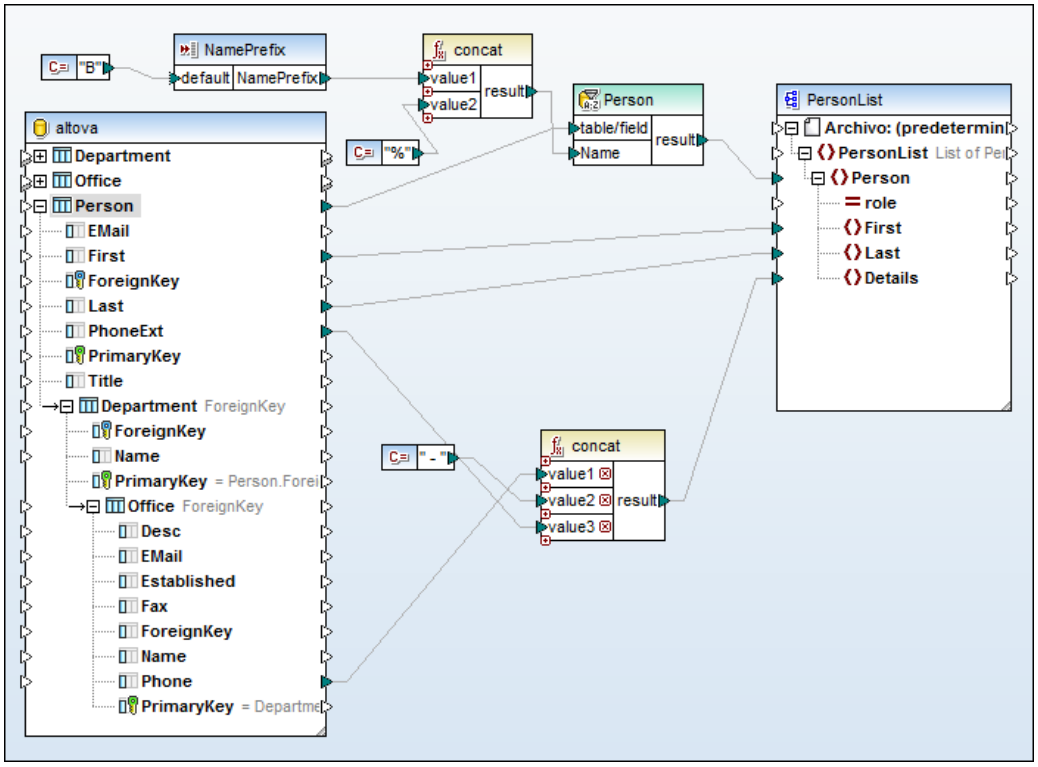


La instrucción WHERE del primer cuadro de texto define un parámetro llamado **`Name`** que usa la palabra clave LIKE para buscar un patrón en el campo **`Last`** de la tabla **`Person`**. En este caso hay una constante que suministra la cadena de búsqueda (**B**) que permite encontrar todas las personas (**`Person`**) cuyo apellido (**`Last`**) empieza por B.

El carácter comodín % (de la constante) denota un número cualquiera de caracteres.

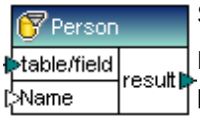
El panel *Parámetros* sirve para especificar el tipo de datos de un parámetro definido en la consulta (en el cuadro de diálogo aparece un mensaje de advertencia cuando el componente WHERE/ORDER no está conectado a ninguna BD).

4. Escriba `First` en el panel *ORDER BY* para que los registros resultantes se almacenen ordenados según el nombre (**`First`**).



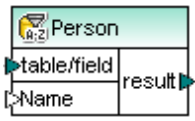
Nota: en la barra de título del componente WHERE/ORDER de SQL un icono indica qué parámetros se definieron:

Se definió una cláusula WHERE, p. ej. `First > "C" AND Last > "C"`



Se definió una cláusula WHERE, p. ej. `Last LIKE :Name`

El parámetro definido por el usuario (**`Name`**) aparece bajo el parámetro **`table/field`**.



Se definió una cláusula WHERE, p, ej. `Last LIKE :Name`

Además se definió una cláusula ORDER BY, p. ej. `ORDER BY Last`. La ordenación se indica con el icono **`A-Z`**.

Al pasar el puntero del mouse por el título del componente WHERE/ORDER aparece información sobre las cláusulas definidas en el componente.

El resultado de esta asignación es que todas las personas (campo **`Person`**) cuyo apellido (campo **`Last`**) empieza por la letra B aparecen ordenados por nombre (campo **`First`**) en orden alfabético.

```
3   <Person>
4     <First>Jessica</First>
5     <Last>Bander</Last>
6     <Details>+1 (321) 555 5155 - 241</Details>
7   </Person>
8   <Person>
9     <First>Michelle</First>
10    <Last>Butler</Last>
11    <Details>+1 (321) 555 5155 - 654</Details>
12  </Person>
13  <Person>
14    <First>Theo</First>
15    <Last>Bone</Last>
16    <Details>+1 (927) 555 0094 - 331</Details>
17  </Person>
```

Operadores WHERE/ORDER de SQL

A continuación enumeramos los operadores que se pueden utilizar dentro de los componentes WHERE/ORDER:

Operador	Descripción
=	igual a
<>	desigual a
<	menor que
>	mayor que
>=	mayor que/igual
<=	menor que/igual
IN	recupera un valor conocido de una columna
LIKE	busca un patrón determinado
BETWEEN	busca dentro del intervalo

Comodines:
El comodín % se usa para definir un número cualquiera de caracteres dentro de un patrón (igual que el comodín * utilizado en otros programas). Por ejemplo, %r recupera todos los registro que terminan con la letra "r".

Datos XML:
Los comandos **XQuery** también se pueden utilizar para consultar bases de datos compatibles con el almacenamiento y la consulta de datos de BD XML, como IBM DB2.

P. ej. **xmlexists**('$c/Client/Address[zip>"55116"]' passing USER.CLIENTS.CONTACTINFO AS "c")

Select * FROM Person WHERE
First > "C" AND Last > "C"

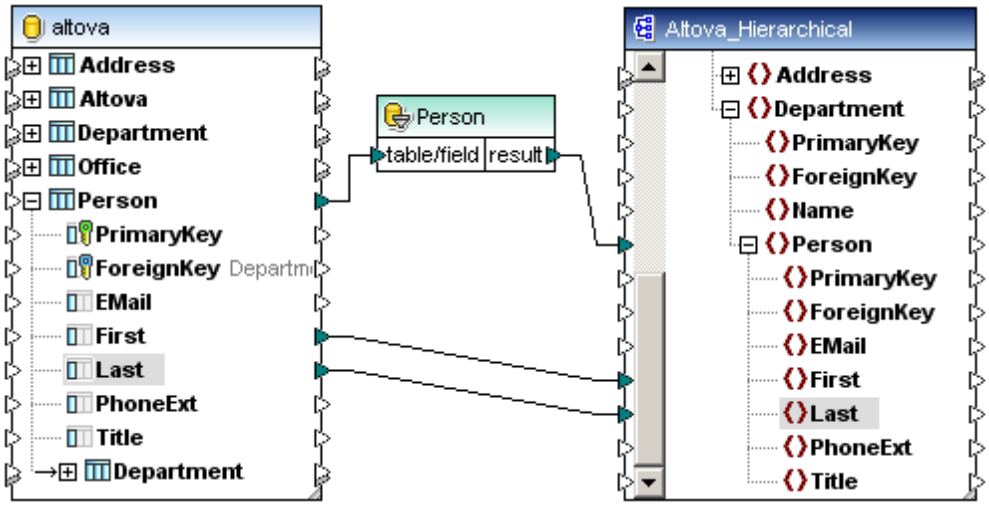
Recupera los registros cuyo contenido de **`First`** y **`Last`** es mayor que la letra C. Es decir, recupera todos los nombres a partir de Callaby (dados por la base de datos altova.mdb).

Observe con atención la colocación de los conectores:

- El parámetro **`table/field`** está conectado con la tabla que se debe consultar (la tabla **`Person`**).

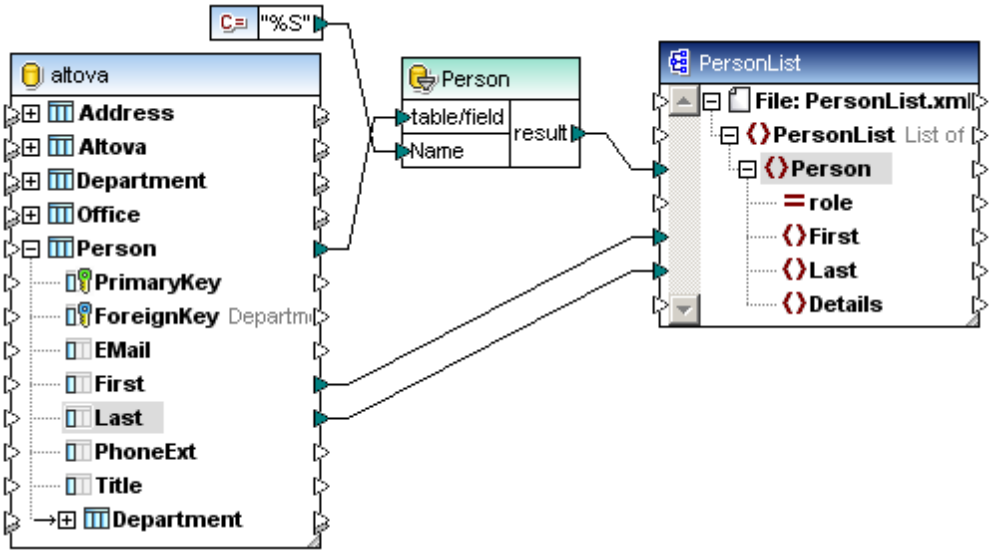
 © 2014 Altova GmbH

- El parámetro **result** está conectado a un elemento primario de los campos que se deben consultar/filtrar (el elemento **Person**). Los campos **First** y **Last** están conectados a elementos secundarios del componente de destino. De lo contrario no aparecerían en el resultado de la asignación.



Select from Person WHERE
Last LIKE :Name

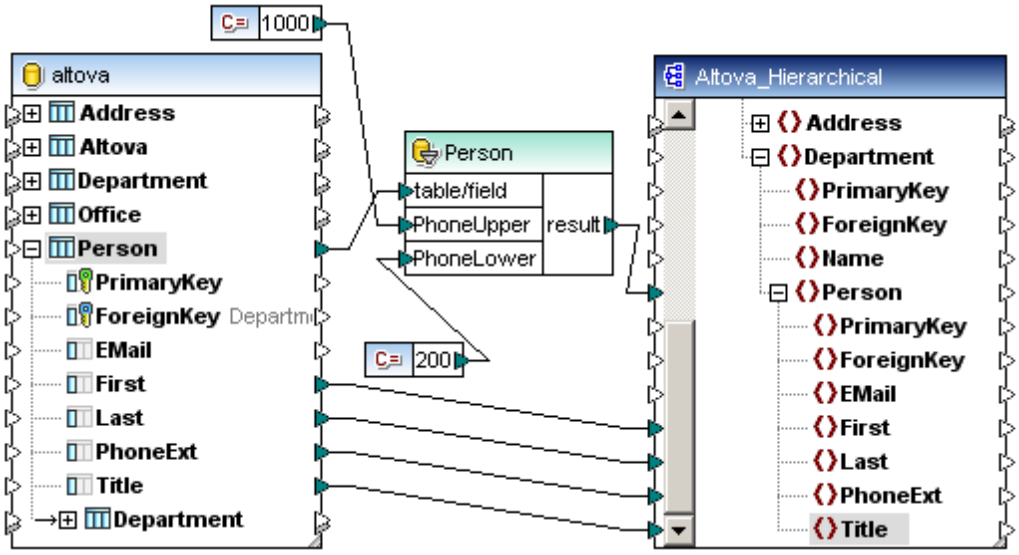
Esta instrucción WHERE crea un parámetro **Name** que aparece como parámetro del componente WHERE/ORDER de SQL. En este caso, el parámetro **Name** se usa para buscar un patrón en la columna **Last**. El comodín % denota un número cualquiera de caracteres.



La constante suministra los valores que se deben buscar en todos los apellidos (**Last**) que terminen con S (es decir **%S**).

Select from Person WHERE
PhoneExt < :PhoneUpper and PhoneExt > :PhoneLower

Esta cláusula WHERE crea dos parámetros: **PhoneUpper** y **PhoneLower**, con los que se comparan los valores actuales de **PhoneExt**. El valor de **PhoneUpper** y de **PhoneLower** viene dado por constantes (**1000** y **200** respectivamente).



Nota: la cláusula WHERE de este ejemplo también se podría expresar con el operador BETWEEN:

Select from Person WHERE
PhoneExt BETWEEN :PhoneUpper and :PhoneLower

 © 2014 Altova GmbH

10.2.12 Asignaciones entre datos XML y de BD

MapForce permite realizar asignaciones entre datos XML y varios tipos de BD, como MS Access e IBM DB2 versión 9. En este apartado describimos el archivo de asignación `xml2access.mfd`, disponible en la carpeta `...\Tutorial`.

Las asignaciones de datos se pueden crear entre documentos XML y los campos de BD **`string`**, **`varchar`** y **`memo`** que tengan la suficiente longitud. Recuerde que la codificación de caracteres de estos documentos siempre es la codificación del campo **`string`** subyacente de la BD. Si el campo no almacena texto como Unicode, algunos caracteres no se pueden representar. Solamente los campos XML nativos de IBM DB2 versión 2 son totalmente compatibles con todas las codificaciones XML.

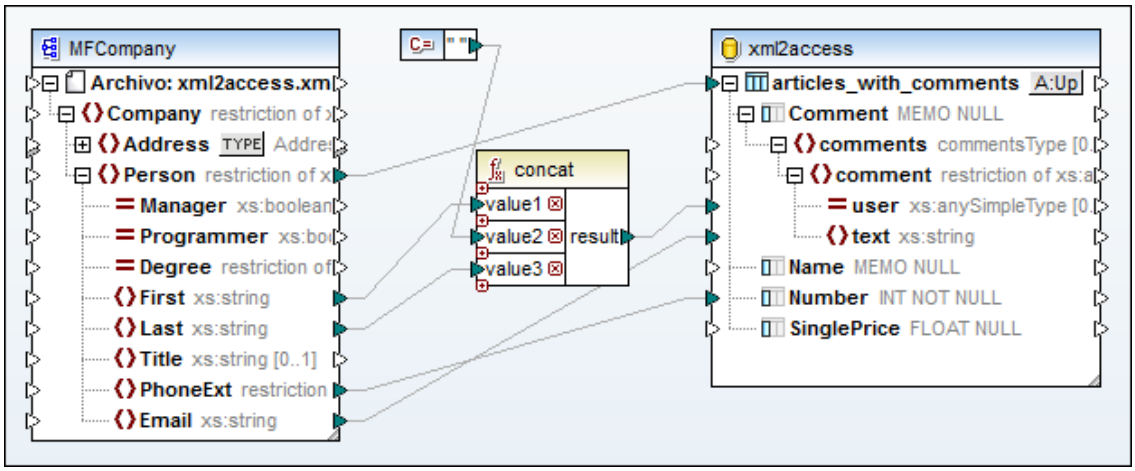
Para que los nodos XML aparezcan en el componente de BD, que permite asignar los datos a elementos XML, es necesario **atribuir a la BD** un esquema XML. Después puede proceder a crear asignaciones igual que siempre.

Por el momento, MapForce permite asignar **un solo esquema** XML por cada columna de la BD y seleccionar **un solo elemento raíz** para cada esquema.

Abra el archivo de asignación `xml2access.mfd`, disponible en la carpeta `...\Tutorial`.

Este ejemplo de asignación:

- Actualiza los datos XML de la columna **`Comment`** de la BD de destino siempre y cuando:
- El contenido del elemento **`PhoneExt`** y del campo **`Number`** sea el mismo.

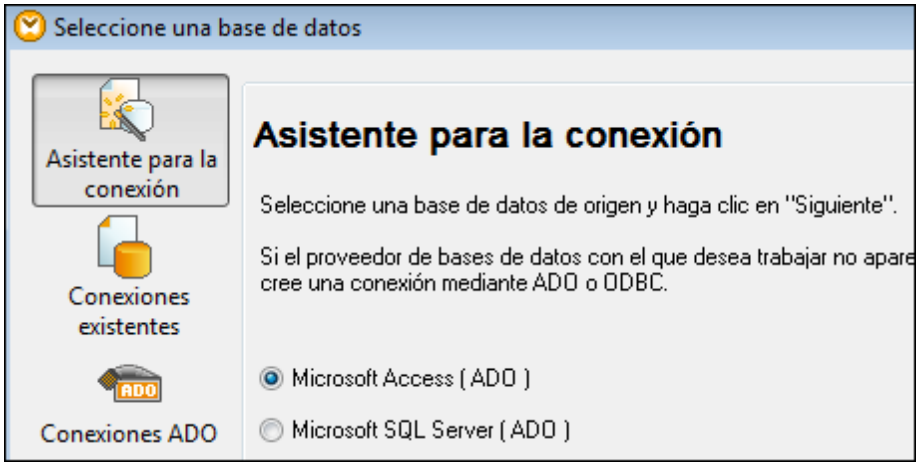


Para insertar el componente de datos de origen:

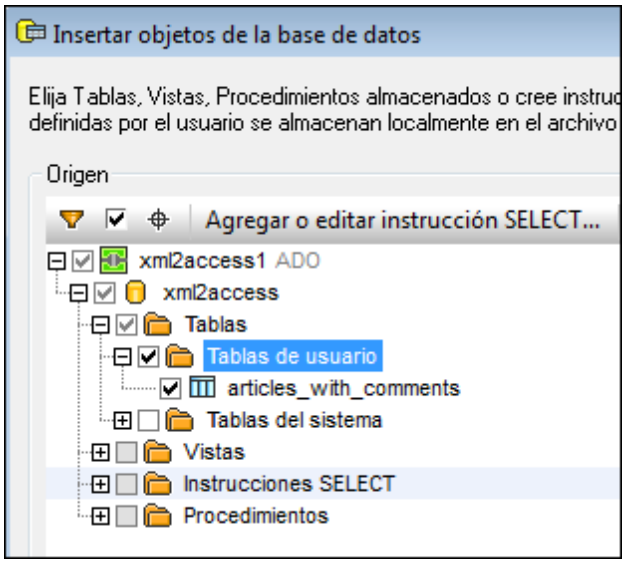
1. Haga clic en el icono **Insertar archivo o esquema XML** y seleccione el esquema `MFCompany.xsd`.
2. En el aviso que aparece haga clic en **Examinar...** y seleccione el archivo `xml2access.mfd`.

Para insertar la BD de destino y asignarle datos:

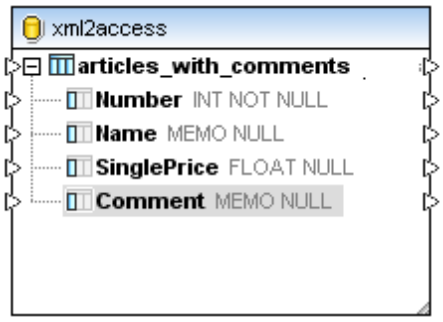
1. Haga clic en el icono **Insertar base de datos**. En el cuadro de diálogo que aparece haga clic en *MS Access (ADO)* y después en **Siguiente**.



2. Haga clic en el botón **Examinar**, seleccione el archivo `xml2access.mdb` de la carpeta `...\Tutorial` y después haga clic en **Siguiente**.



3. Aparece el cuadro de diálogo "Insertar objetos de la base de datos". Marque la casilla de la tabla de usuario **`articles_with_comments`** y haga clic en **Aceptar**.

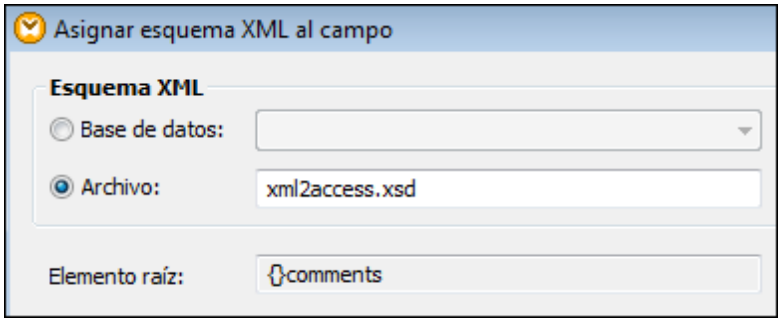


Para asignar un esquema XML a una BD:

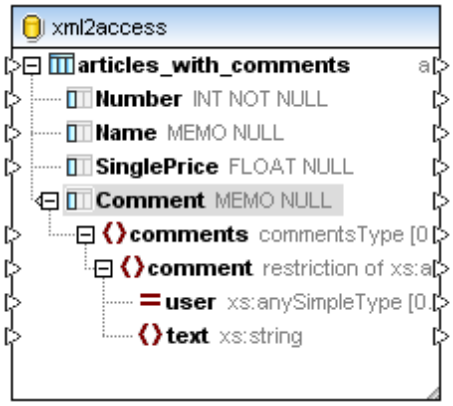
1. Haga clic con el botón secundario en la columna **`Comment`** de la BD de destino y seleccione el comando **Asignar esquema XML al campo...**.
2. Haga clic en el botón de opción *Archivo*, seleccione `xml2access.xsd` y haga clic en

 © 2014 Altova GmbH

Aceptar.

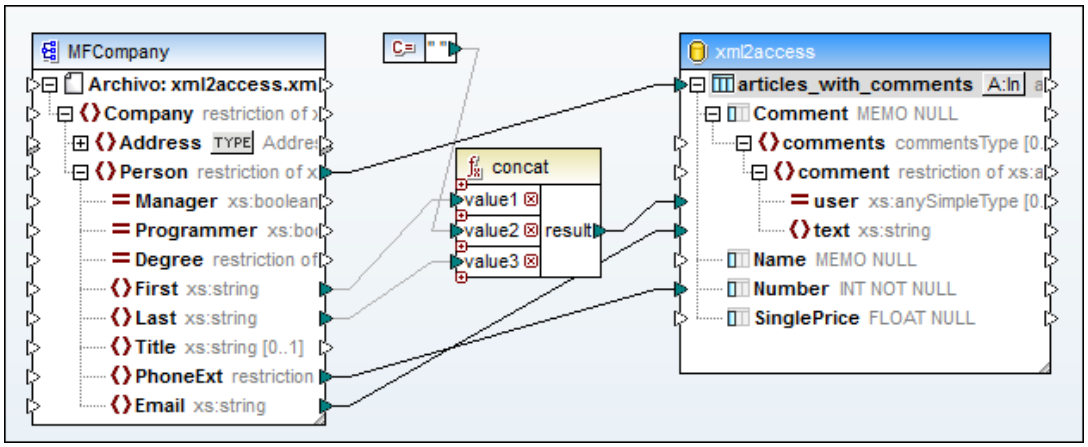


3. Expanda la columna **Comment** para poder ver sus elementos y conectarlos.



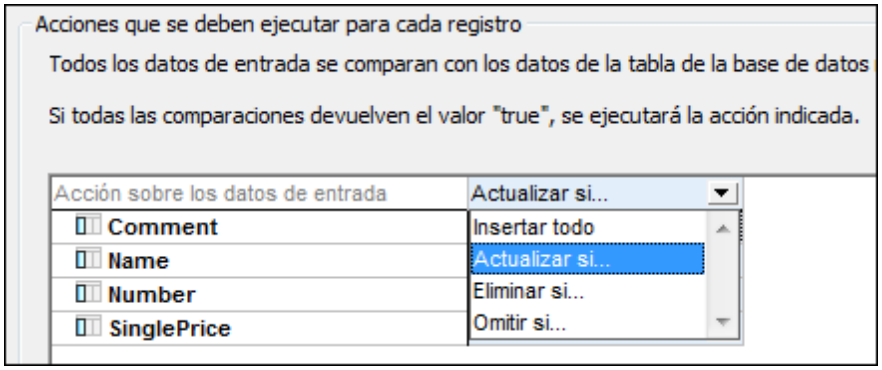
Bajo la columna **Comment** aparece el nodo de esquema XML en forma de árbol con todos los elementos asignables. Ya puede asignar datos XML a la BD.

4. Cree las conexiones que aparecen en la imagen.

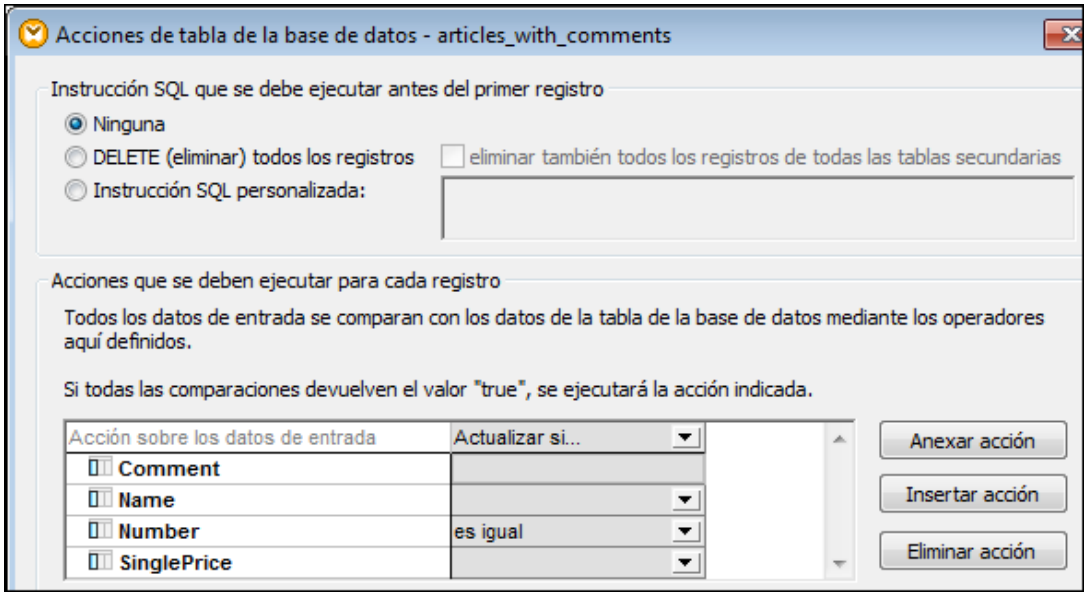


Para definir las acciones de tabla y asignar los datos:

1. Haga clic en el icono de acción de tabla A:In de la tabla **articles_with_comments**.
2. Haga clic en el primer cuadro combinado (**Insertar todo**) y seleccione **Actualizar si...**.



3. Haga clic en el cuadro combinado de **`Number`** y seleccione **es igual**. Haga clic en **Aceptar** para confirmar.



4. Abra el panel **Resultados** y haga clic en el icono **Ajuste automático de línea** para ver bien el script SQL resultante.
5. Haga clic en el icono **Ejecutar script SQL** para insertar los datos XML en la base de datos.

 © 2014 Altova GmbH

```
UPDATE [articles_with_comments] SET [Comment] = '<comments><comment user="Vernon
Callaby"><text>v.callaby@nanonull.com</text></comment></comments>' WHERE (
[articles_with_comments].[Number]=1)
-->>> OK. 1 fila(s).

UPDATE [articles_with_comments] SET [Comment] = '<comments><comment user="Frank
Further"><text>f.further@nanonull.com</text></comment></comments>' WHERE (
[articles_with_comments].[Number]=2)
-->>> OK. 1 fila(s).

UPDATE [articles_with_comments] SET [Comment] = '<comments><comment user="Loby
Matise"><text>l.matise@nanonull.com</text></comment></comments>' WHERE (
[articles_with_comments].[Number]=3)
-->>> OK. 1 fila(s).
```

10.2.13 IBM DB2: asignaciones entre datos XML y de BD

MapForce permite realizar asignaciones entre datos XML y bases de datos IBM DB2 versión 9. Para los ejemplos utilizados en esta sección es necesario tener acceso a una base de datos IBM DB2. Los demás archivos utilizados en esta sección están en la carpeta `...\Tutorial`.

Para que los nodos XML aparezcan en el componente de BD y así poder asignar los datos a elementos XML, es necesario **atribuir a la BD** un esquema XML. Después puede proceder a crear asignaciones igual que siempre.

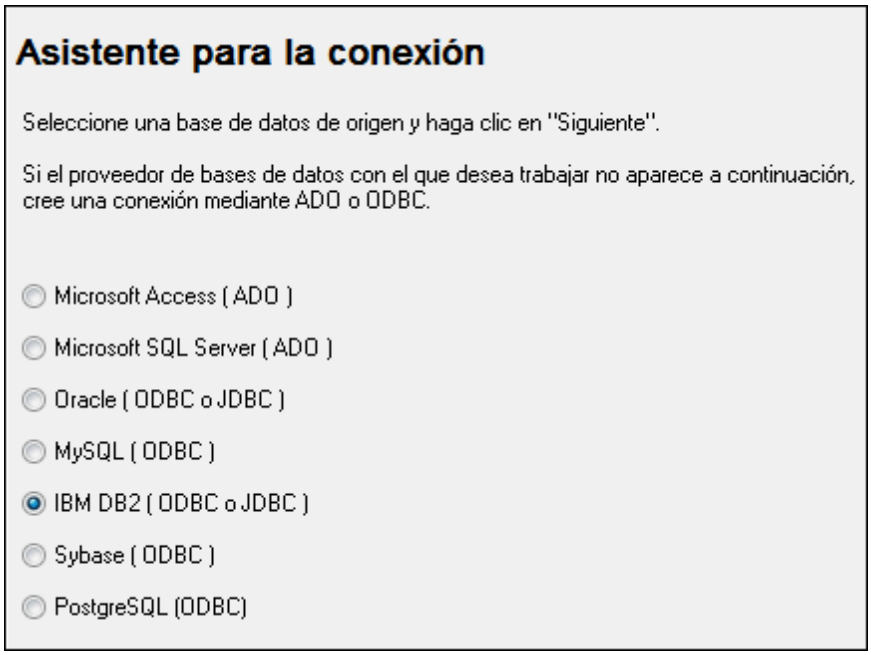
Por el momento, MapForce permite asignar **un solo esquema** XML por cada columna de la BD y seleccionar **un solo elemento raíz** para cada esquema.

Nota: al añadir un origen de datos ODBC para IBM iSeries (anteriormente AS/400), se establece un marcador que habilita tiempos de espera de consultas. Para cargar archivos de asignación en MapForce es necesario **deshabilitar** esta opción.

Al añadir un origen de datos ODBC para el controlador iSeries Access ODBC, se abre el cuadro de diálogo "iSeries Access for Windows ODBC Setup". Seleccione la pestaña **Performance**, haga clic en el botón **Advanced** y después desactive la casilla *Allow query timeout*.

Configurar e insertar una base de datos IBM DB2:

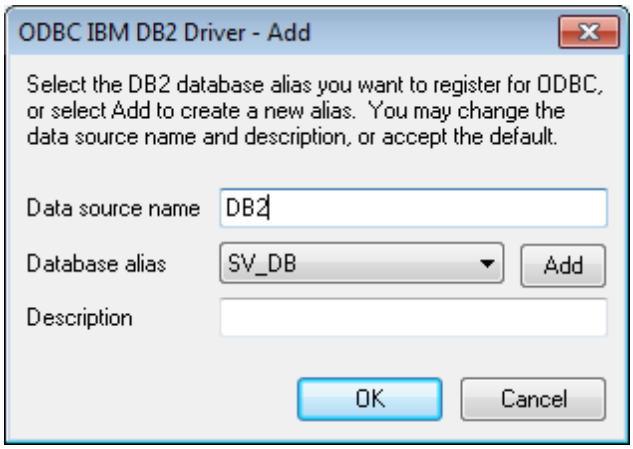
1. En la barra de herramientas de MapForce seleccione uno de estos iconos: **Java**, **C#, C++** o **BUILTIN**.
2. Haga clic en el icono **Insertar base de datos** .



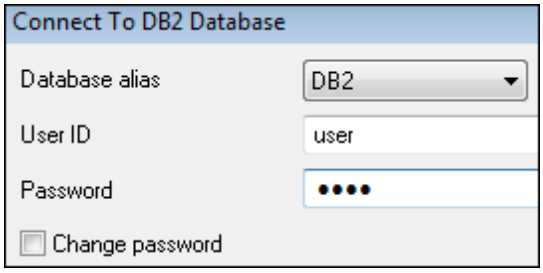
3. Haga clic en el botón de opción *IBM DB2 (ODBC o JDBC)* y después en **Siguiente**. Seleccione *ODBC* en la siguiente pantalla y haga clic en **Siguiente**.
4. Haga clic en la opción *Crear un nombre del origen de datos...* y seleccione un controlador de la lista del cuadro combinado.

 © 2014 Altova GmbH

5. Haga clic en **Conectarse**. Aparece el cuadro de diálogo "ODBC IBM DB Driver - Add".
6. Escriba el nombre de origen de datos, seleccione el alias de BD (o añádalo) y haga clic en **OK**.



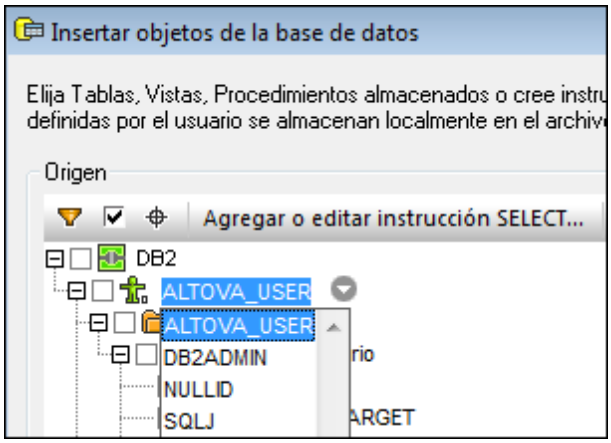
7. Introduzca los datos de inicio de sesión y haga clic en **OK** para conectarse a la BD.



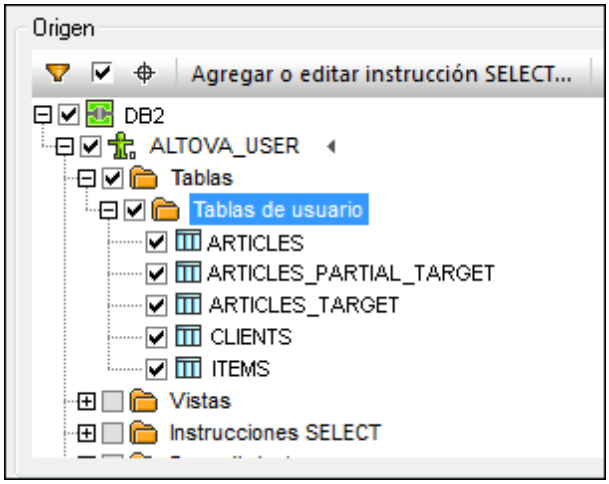
Aparece otro cuadro de diálogo donde puede elegir cómo se comunica el origen de datos con el servidor (de forma nativa o por la API ODBC).

Al hacer clic en **OK** se abre el cuadro de diálogo "Insertar objetos de la base de datos"

8. Haga clic en el icono para seleccionar un esquema de base de datos, p. ej. **`ALTOVA_USER`**.

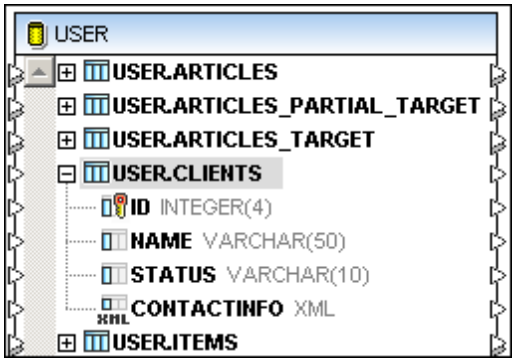


El cuadro de diálogo ahora incluye todas las tablas de **`ALTOVA_USER`**.



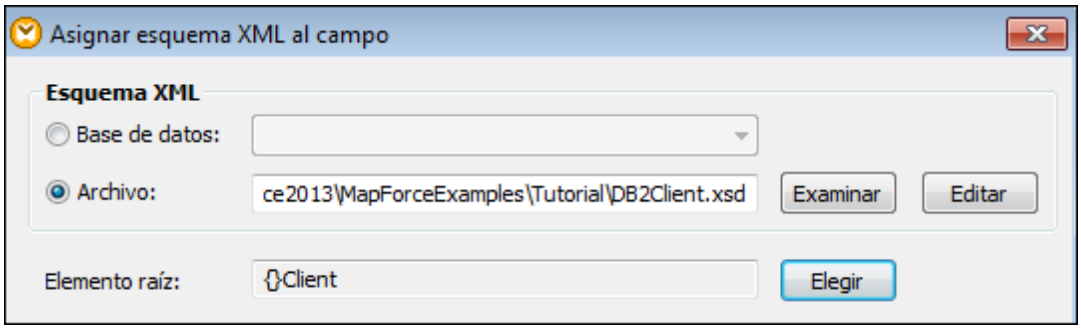
9. Marque la casilla *Tablas de usuario* para seleccionar todas las tablas secundarias y haga clic en **Aceptar** para insertar el componente de base de datos.
10. Haga clic en el icono de expansión de la tabla **`USER.CLIENTS`** para ver su contenido. Observe que la columna **`CONTACTINFO`** es una columna de tipo **`XML`**.

 © 2014 Altova GmbH

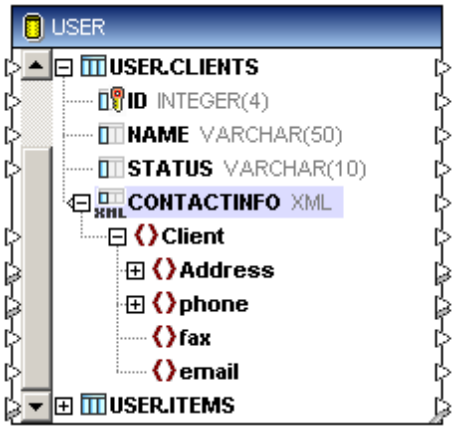


Asignar un esquema XML a un archivo XML:

1. Haga clic con el botón secundario en la columna **CONTACTINFO** de la tabla **USER.CLIENTS** y seleccione el comando **Asignar esquema XML al campo...**.
2. Con el botón de opción *Base de datos* puede seleccionar un esquema ya guardado (registrado) en la base de datos. Con el botón de opción *Archivo* puede seleccionar un esquema local. Para este ejemplo seleccionamos el esquema XML `DB2Client.xsd`, guardado en la carpeta `...\Tutorial`.

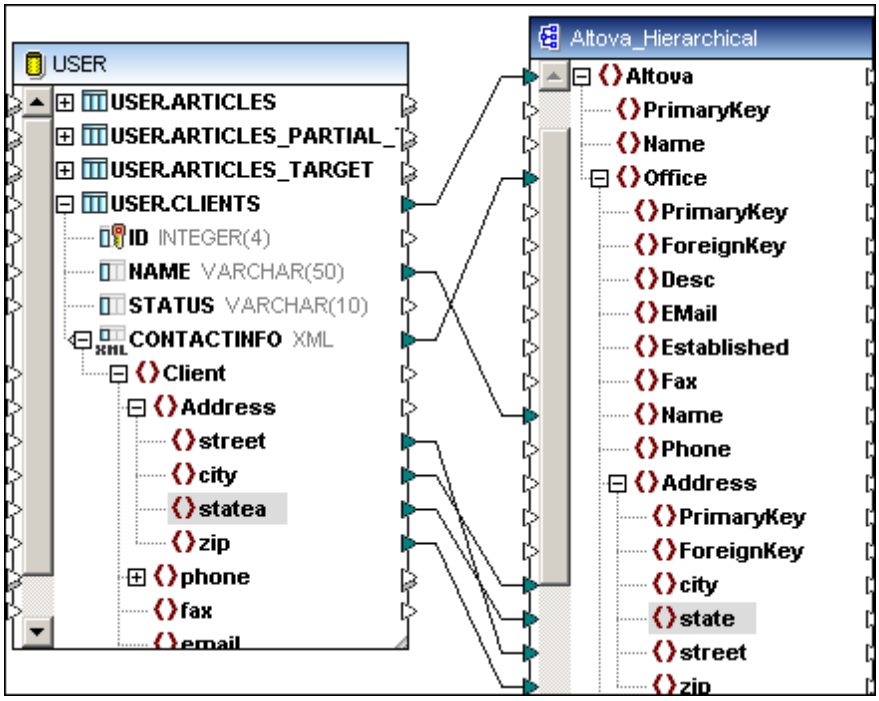


3. Elija qué elemento raíz del esquema debe aparecer en el componente (p. ej. **Cliente**) y haga clic en **Aceptar**.

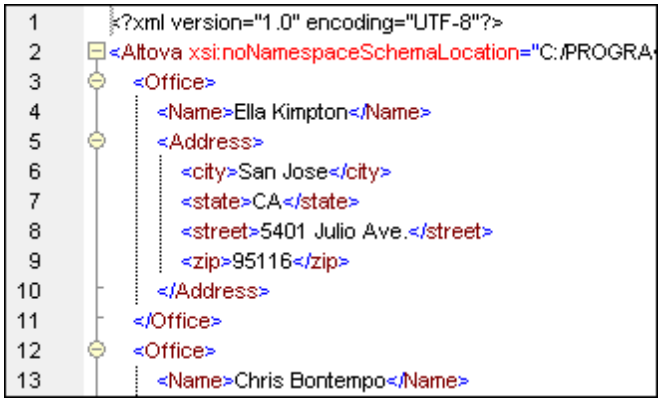


El elemento **Cliente** aparece ahora debajo de la columna **CONTACTINFO**. Haga clic en el icono de expansión para ver la estructura del esquema.

4. Cree las asignaciones que aparecen en la siguiente imagen entre los elementos del esquema y el componente de destino, que en este caso es un documento XML.



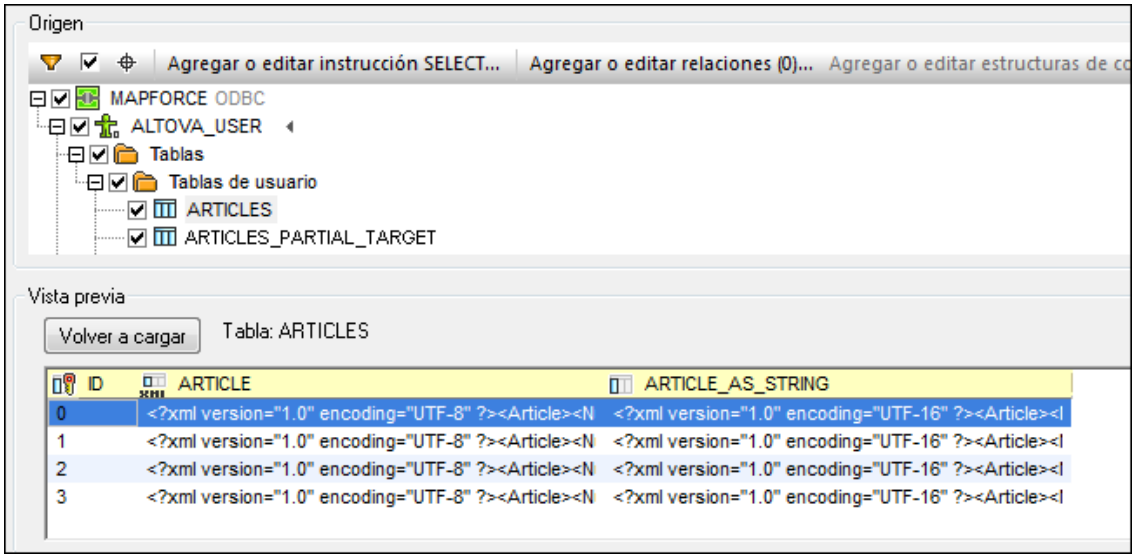
5. Abra el panel **Resultados** para ver el resultado de la asignación.



Vista previa del contenido de las tablas

En la parte inferior el cuadro de diálogo "Insertar objetos de la base de datos" el botón **Vista previa** sirve para ver los datos de la tabla antes de insertarla. Cuando el panel de vista previa está abierto, el botón se convierte en el botón **Volver a cargar**. Si una columna de la tabla es de tipo XML, pase el puntero del mouse por la columna en la ventana de vista previa para ver su contenido XML.

 © 2014 Altova GmbH



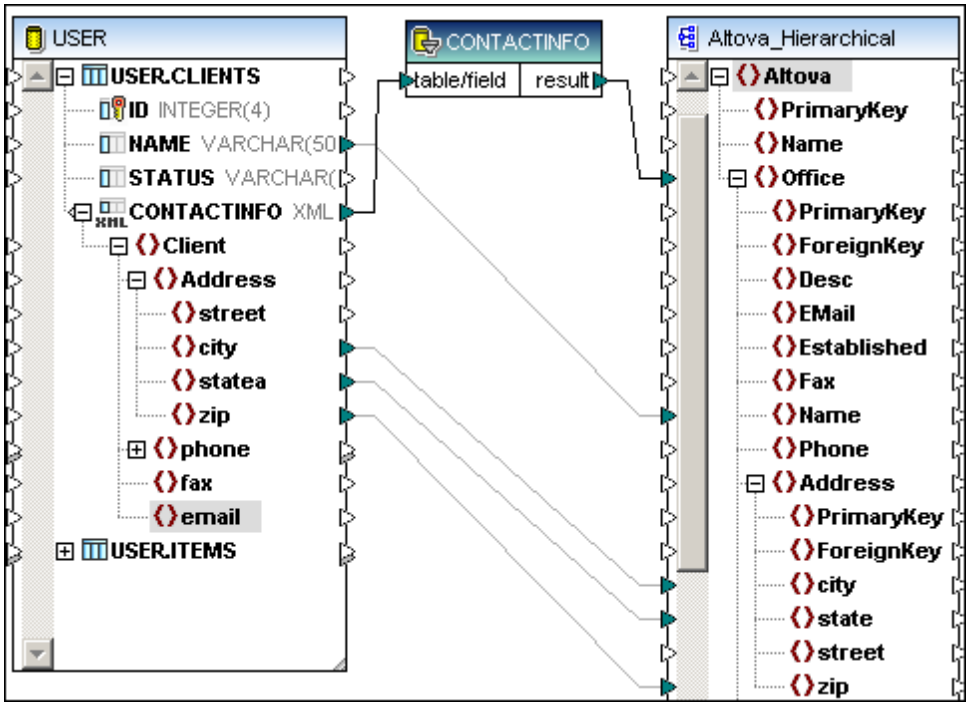
Consultar y asignar datos en IBM DB2

MapForce permite consultar datos XML en bases de datos IBM DB2 usando componentes WHERE de SQL, así como asignar los datos del conjunto de registros a otros componentes. Para más información consulte la sección Componente WHERE/ORDER de SQL , donde se explica cómo insertar y usar este tipo de componente. En este apartado describimos cómo se consultan y asignan datos XML desde una base de datos IBM DB2.

No olvide que también puede consultar bases de datos en el panel **Consulta de la BD** directamente. Lea la sección Consulta directa de bases de datos (panel Consulta de la BD) para obtener más información.

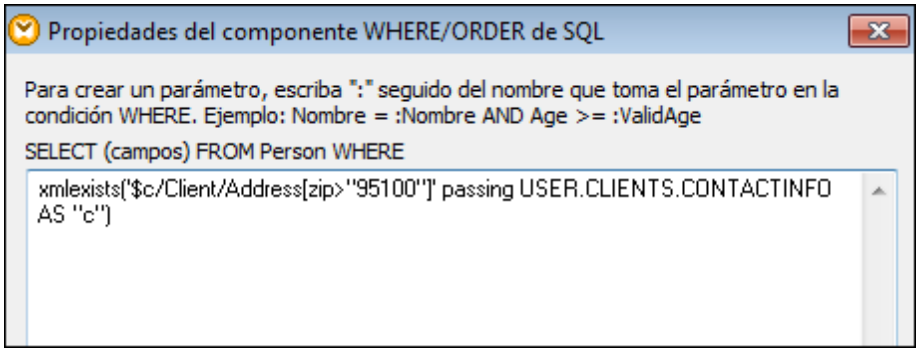
Tras insertar la base de datos DB2 y asignar el esquema XML a la columna **`CONTACTINFO`**:

1. Haga clic en el icono **WHERE de SQL** de la barra de herramientas para insertar el componente WHERE de SQL.
2. Conecte la columna **`CONTACTINFO`** de la base de datos de origen al parámetro **`table/field`** del componente WHERE de SQL.
3. Conecte el parámetro **`result`** del componente WHERE de SQL a un elemento del componente de destino (p. ej. **`Office`**).



Observe cómo el componente WHERE ahora se llama **`CONTACTINFO`**.

4. Haga doble clic en el componente **`CONTACTINFO`** para crear la consulta.
5. Escriba la consulta WHERE SQL/XML, p. ej. **xmlexists('$c/Client/Address[zip>"95100"]' passing USER.CLIENTS.CONTACTINFO AS "c")**



Observe que la primera parte de la instrucción SELECT (`SELECT (campos) FROM USER CLIENTS WHERE`) se genera automáticamente cuando se conectan los conectores de entrada/salida al componente de origen/destino.

Esta consulta da como resultado los registros cuyo código ZIP es mayor que 95100 en el archivo XML.

La función **xmlexists** permite navegar por el documento XML usando una expresión XPath (p. ej. '$c**/Client/Address[zip>"95100"]**') y probar una condición. Para más información sobre las funciones SQL/XML consulte la página web DB2 Information Centre.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Altova xsi:noNamespaceSchemaLocation="C:/PROGRA~1/Altova
3     <Office>
4       <Name>Ella Kimpton</Name>
5       <Address>
6         <city>San Jose</city>
7         <state>CA</state>
8         <zip>95116</zip>
9       </Address>
10    </Office>
11    <Office>
12      <Name>Chris Bontempo</Name>
13      <Address>
14        <city>San Jose</city>
15        <zip>95124</zip>
16      </Address>
17    </Office>
```

Asignar datos XML: bases de datos IBM DB2 como destino

En este apartado explicamos cómo asignar datos XML a un documento XML de destino almacenado en una base de datos IBM DB2. Para este ejemplo es necesario tener acceso a una base de datos IBM DB2. Los demás archivos utilizados en este apartado están disponibles en la carpeta `...\Tutorial`.

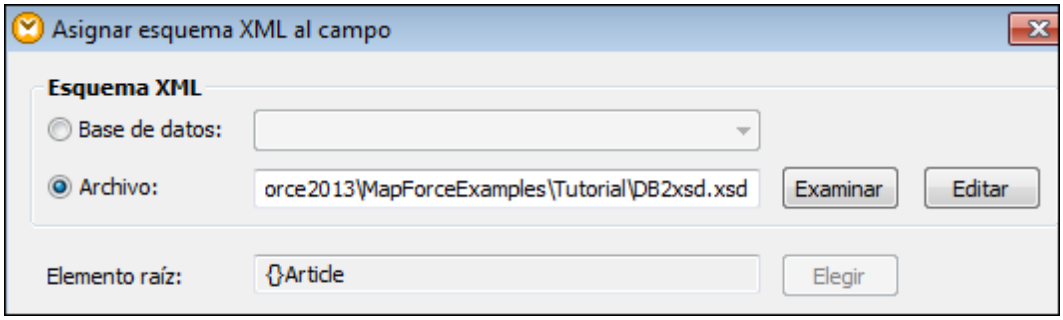
No olvide que también puede consultar bases de datos en el panel **Consulta de la BD** directamente. Lea la sección Consulta directa de bases de datos (panel Consulta de la BD) para obtener más información.

Para insertar el componente de origen:

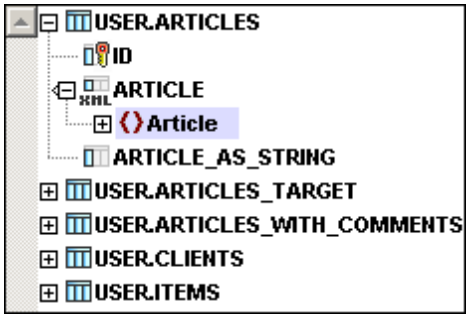
1. Haga clic en el icono **Insertar archivo o esquema XML** y seleccione el esquema `DB2asTarget.xsd`.
2. En el aviso que aparece haga clic en el botón **Examinar** y seleccion el archivo `DB2asTarget.xml` de la carpeta `...\Tutorial`.
3. Seleccione **`Articles`** como elemento raíz.

Para insertar la BD de destino y asignarle datos:

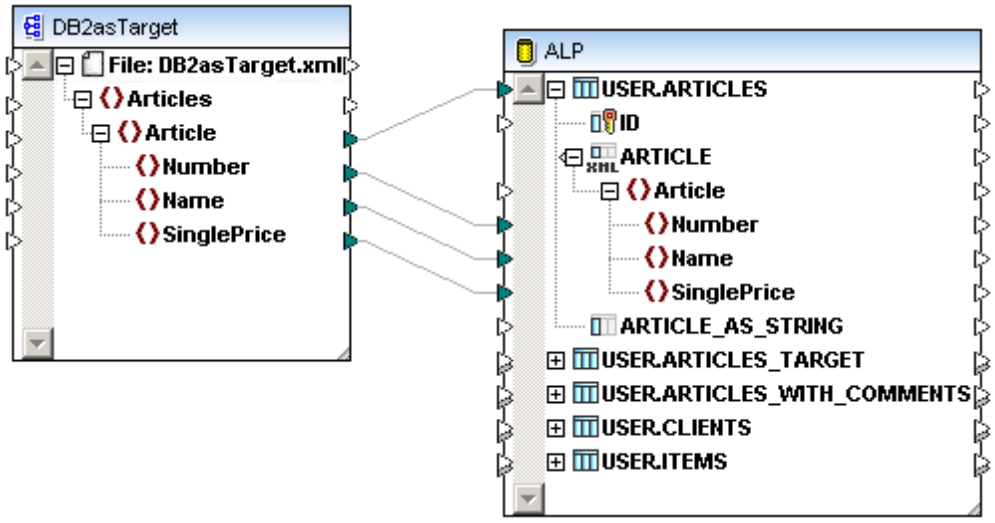
1. Inserte la BD siguiendo las instrucciones del apartado IBM DB2: asignaciones entre datos XML y de BD.
2. Haga clic con el botón secundario en la columna **`ARTICLE`** de la base de datos y seleccione el comando **Asignar esquema XML al campo...**.
3. Seleccione el botón de opción *Archivo*, busque el archivo `DB2xsd.xsd`, elija el elemento raíz **Article** y haga clic en **Aceptar**.



4. Expanda el elemento **`Article`** para poder crear conexiones con otros elementos.



5. Cree las conexiones que aparecen a continuación entre el componente de origen y el de destino.



6. Abra el panel **Resultados** y haga clic en el icono **Ajuste automático de línea** para ver mejor el script SQL.

 © 2014 Altova GmbH

```
8   INSERT INTO "USER"."ARTICLES" ("ARTICLE")
9       VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
    xsi:noNamespaceSchemaLocation="C:\Program
    Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>11</Number><Name>
    sneakers</Name><SinglePrice>99</SinglePrice></Article>')
10
11  INSERT INTO "USER"."ARTICLES" ("ARTICLE")
12      VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
    xsi:noNamespaceSchemaLocation="C:\Program
    Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>22</Number><Name>
    band</Name><SinglePrice>2.9</SinglePrice></Article>')
```

Aquí puede ver una vista previa de los datos XML que se insertarán en la BD.

7. Haga clic en el icono **Ejecutar script SQL** para insertar los datos en la BD.

```
1   /*
2   Las instrucciones SQL que aparecen a continuación se ejecutaron durante la función "Gene
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 e
    herramienta de consultas SQL.
5   */
6   INSERT INTO "USER"."ARTICLES" ("ARTICLE")
7       VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
8   xsi:noNamespaceSchemaLocation="C:\Program
9   Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
10  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>11</Number><Name>
    ers</Name><SinglePrice>99</SinglePrice></Article>')
    -->>> OK. Una fila o más.
```

En el panel **Resultados** puede comprobar si las instrucciones se ejecutaron correctamente o no.

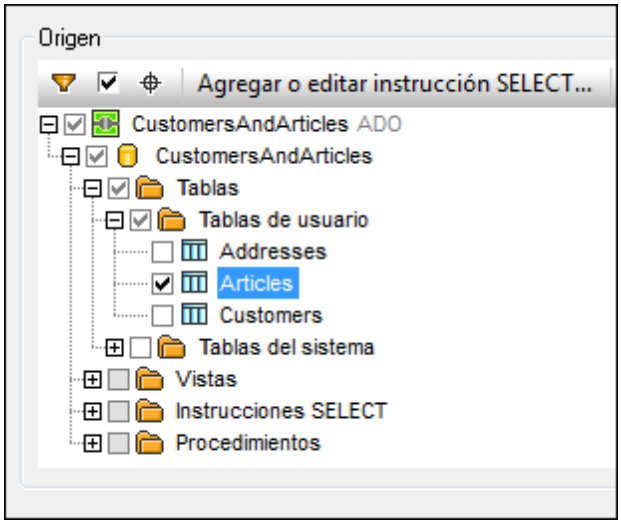
Asignar datos de una BD a otra

En el ejemplo de este apartado asignamos datos XML de una base de datos MS Access a un documento XML de destino almacenado en una base de datos IBM DB2. Para este ejemplo es necesario tener acceso a una base de datos IBM DB2. Los demás archivos utilizados en el ejemplo están disponibles en la carpeta `...\MapforceExampes` o en la carpeta `...\MapforceExampes\Tutorial`.

No olvide que también puede consultar bases de datos en el panel **Consulta de la BD** directamente. Lea la sección Consulta directa de bases de datos (panel Consulta de la BD) para obtener más información.

Para insertar la base de datos MS Access de origen:

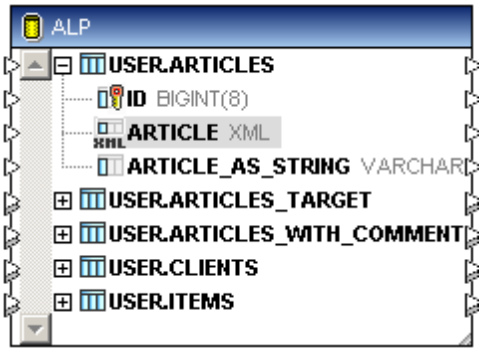
1. Haga clic en el icono **Insertar base de datos** y seleccione la base de datos `CustomersAndArticles.mdb` de la carpeta `...\MapForceExamples`.
2. Seleccione la tabla de usuario **`Articles`** y haga clic en **Aceptar**.



La tabla de la BD se inserta en forma de componente de base de datos.

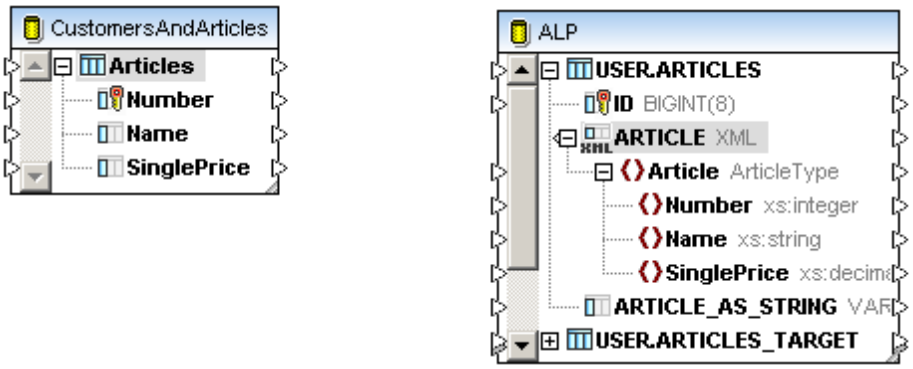
Para insertar la base de datos IBM DB2 de destino y asignarle datos:

1. Inserte la BD siguiendo las instrucciones de este apartado.

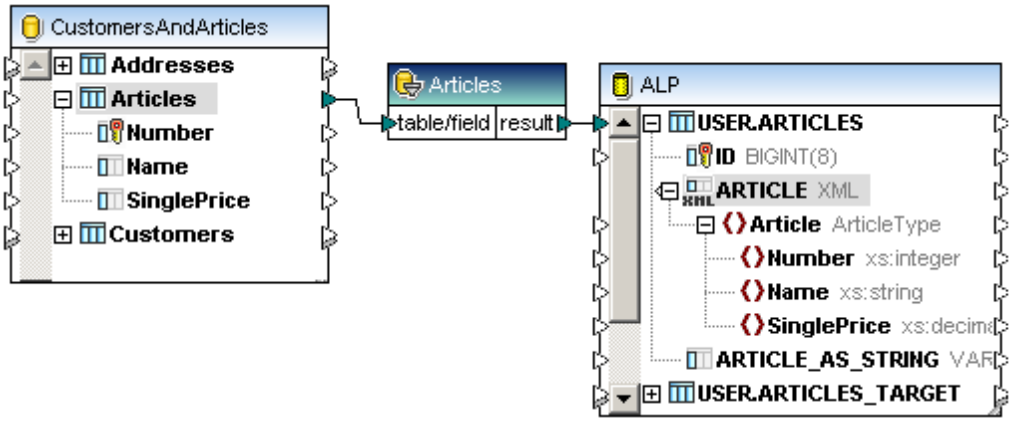


2. Haga clic con el botón secundario en la columna **`ARTICLE`** de la BD de destino y seleccione el comando **Asignar esquema XML al campo...** del menú contextual.
3. Seleccione el botón de opción *Archivo*, elija el archivo de esquema `DB2xsd.xsd` y haga clic en **Aceptar**.

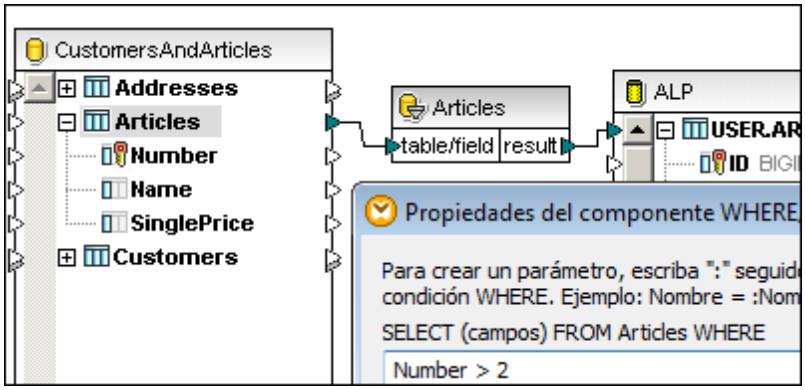
 © 2014 Altova GmbH



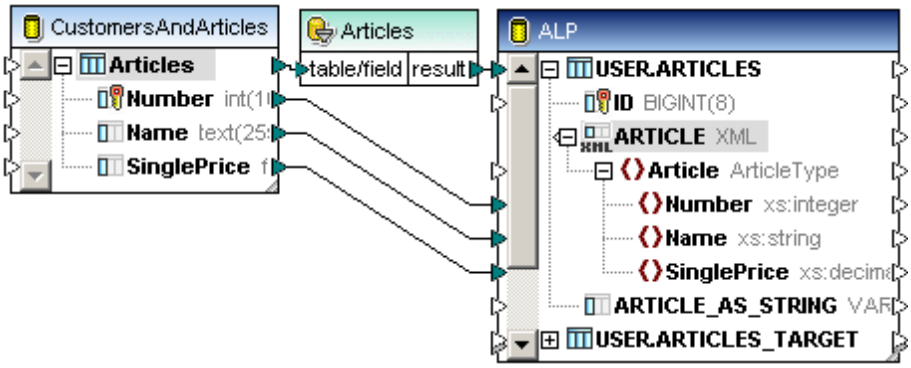
4. Haga clic en el icono WHERE/ORDER de SQL para insertar un componente WHERE de SQL.
5. Conecte el elemento **`Articles`** de la BD de origen al parámetro **`table/field`** del componente WHERE de SQL.
6. Conecte el parámetro **`result`** con el elemento **`USER.ARTICLES`** del componente de destino.



7. Haga doble clic en el componente WHERE de SQL llamado **Articles** y escriba la cláusula Where `Number > 2` en el recuadro. Para confirmar haga clic en **Aceptar**.



8. Asigne los elementos **`Number`**, **`Name`** y **`SinglePrice`** del componente de origen a la BD de destino.



9. Abra el panel **Resultados** para ver una vista previa de la asignación y haga clic en el icono **Ejecutar el script SQL** para insertar los datos en la BD de destino.

```
INSERT INTO "USER"."ARTICLES" ("ARTICLE")
    VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
xsi:noNamespaceSchemaLocation="C:\Program
Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>3</Number><Name>Pants<
/Name><SinglePrice>34</SinglePrice></Article>')

INSERT INTO "USER"."ARTICLES" ("ARTICLE")
    VALUES ('<?xml version="1.0" encoding="UTF-8"?><Article
xsi:noNamespaceSchemaLocation="C:\Program
Files\Altova\MapForce2007\MapForceExamples\Tutorial\DB2xsd.xsd"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"><Number>4</Number><Name>Jacket
</Name><SinglePrice>5750</SinglePrice></Article>')
```

10.2.14 SQL Server 2005: asignaciones de datos XML

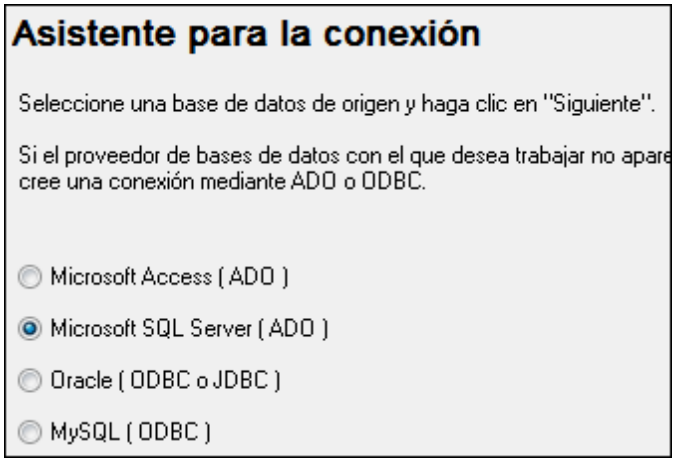
En MapForce puede realizar asignaciones de datos entre archivos XML y bases de datos SQL Server 2005 (o superior). Para el ejemplo de este apartado es necesario tener acceso a una base de datos SQL Server 2005.

Para que los nodos XML aparezcan en el componente de BD y así poder asignar los datos a elementos XML, es necesario **atribuir a la BD** un esquema XML. Después puede proceder a crear asignaciones igual que siempre.

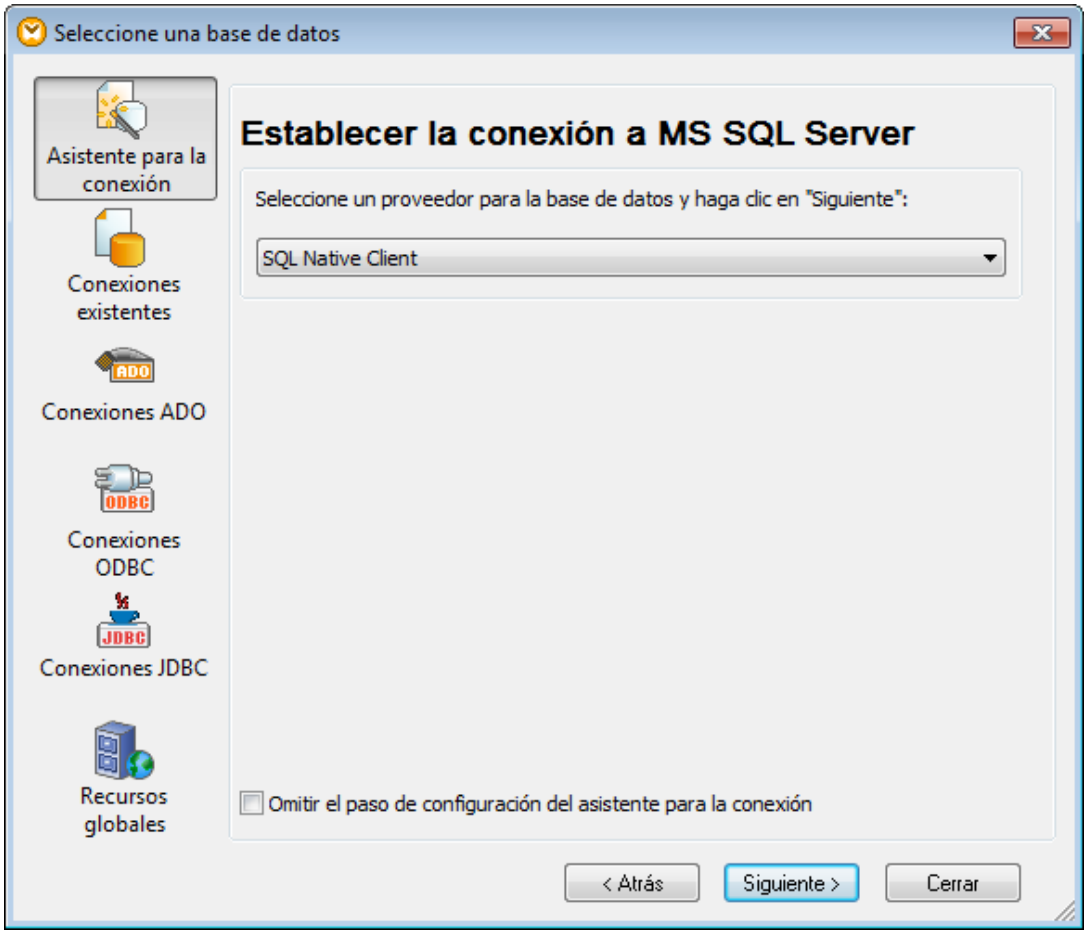
Por el momento, MapForce permite asignar **un solo esquema** XML por cada columna de la BD y seleccionar **un solo elemento raíz** para cada esquema.

Configurar e insertar una base de datos SQL Server 2005:

1. Seleccione el lenguaje de destino en la barra de herramientas: Java, C#, C++ o BUILTIN.
2. Haga clic en el icono **Insertar base de datos** de la barra de herramientas.

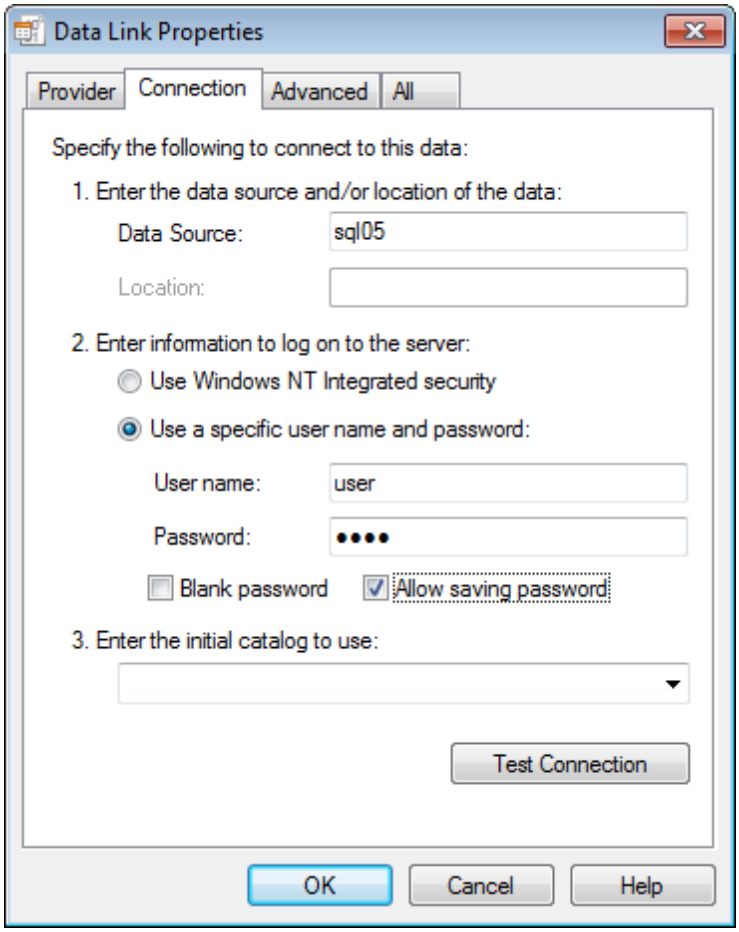


3. Seleccione el botón de opción *Microsoft SQL Server (ADO)* y haga clic en **Siguiente**.
4. Seleccione un controlador de la lista desplegable de controladores.

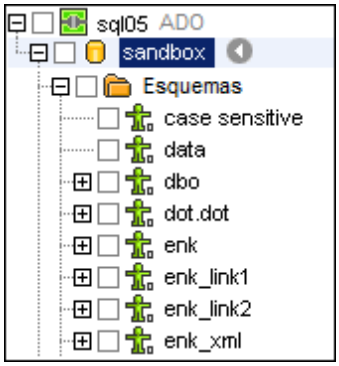


5. Haga clic en **Siguiente** para definir las propiedades de vínculo de datos.
6. Escriba el *Origen de datos*, el *Usuario* y la *Contraseña*. No se olvide de marcar la casilla *Permitir guardar contraseña*.

 © 2014 Altova GmbH

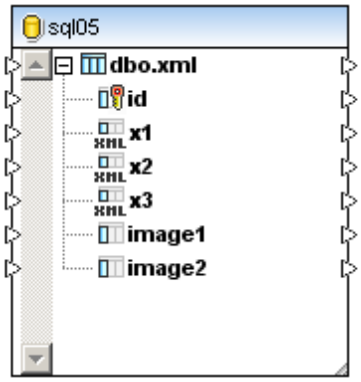


7. Haga clic en el botón **Probar conexión** y después en **Aceptar** para insertar el componente.
8. Haga clic en el icono para seleccionar el esquema de BD (p. ej. el esquema sandbox).



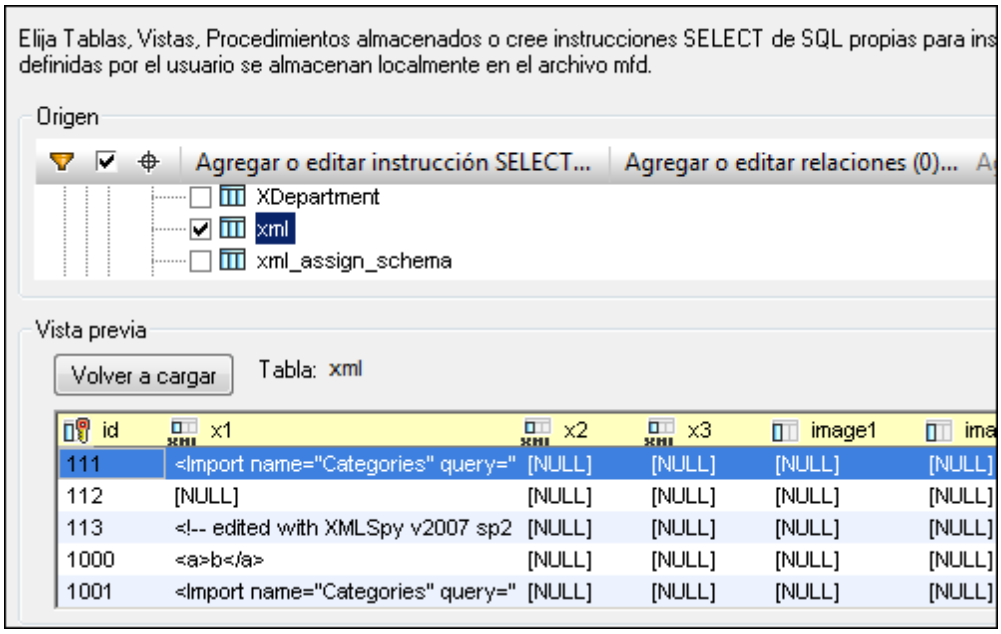
9. Marque las casillas de las tablas que desea insertar (p. ej. `xml`) y después haga clic en **Aceptar**.
   Nota: recuerde que con el botón **Vista previa** puede obtener una vista previa del contenido de la tabla antes de insertar el componente.

La tabla **xml** se inserta como componente de base de datos en el panel **Asignación**.



Observe que las columnas **x1**, **x2** y **x3** son de tipo **XML**.

Vista previa de la tabla **xml**:

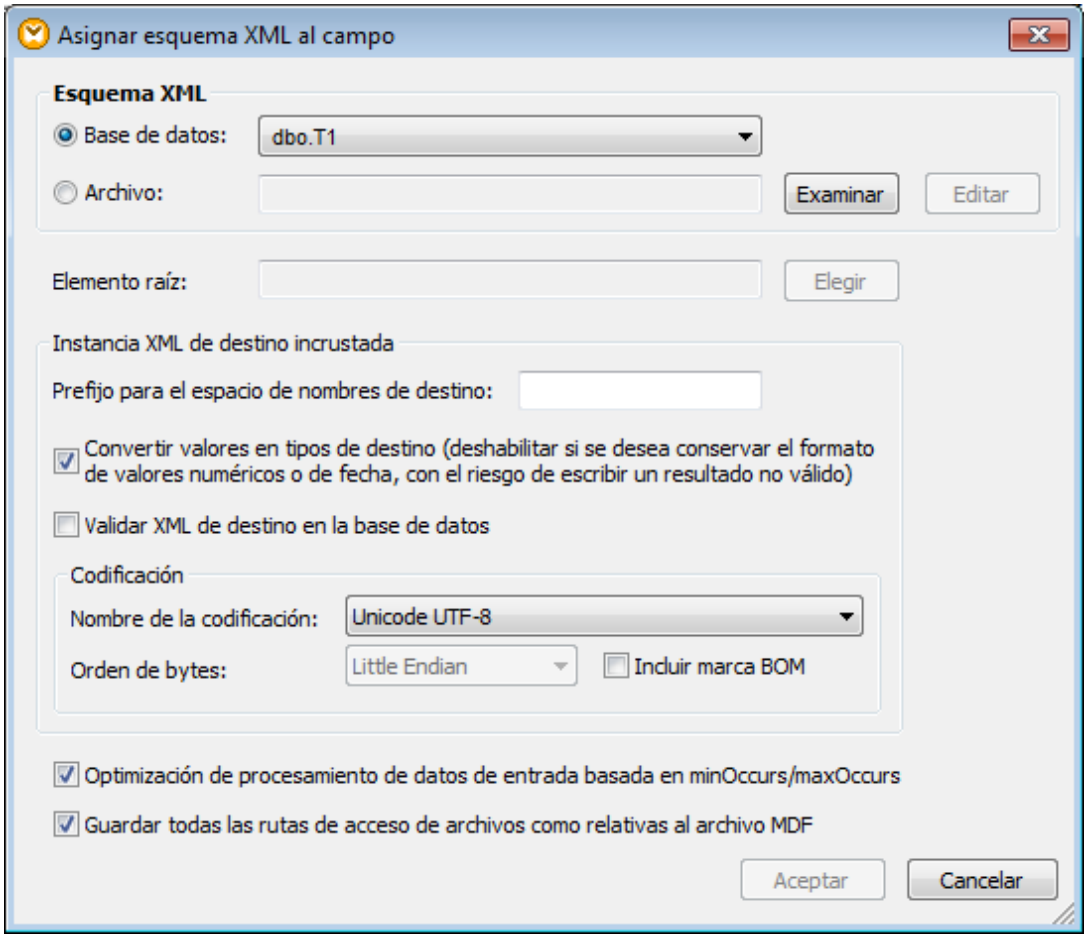


Asignar un esquema XML a un archivo XML:

1. Haga clic con el botón secundario en la columna a la que desea asignar el esquema (p. ej. **x1**) y seleccione **Asignar esquema XML al campo...** en el menú contextual.
2. En el cuadro de diálogo que aparece seleccione el botón de opción *Base de datos*. En el cuadro combinado puede elegir uno de los esquemas guardados (registrados) en la BD.

 © 2014 Altova GmbH

Con el botón de opción *Archivo* puede seleccionar un esquema local.



3. Para terminar haga clic en **Aceptar**.

10.2.15 Consultar BD en el panel de consulta

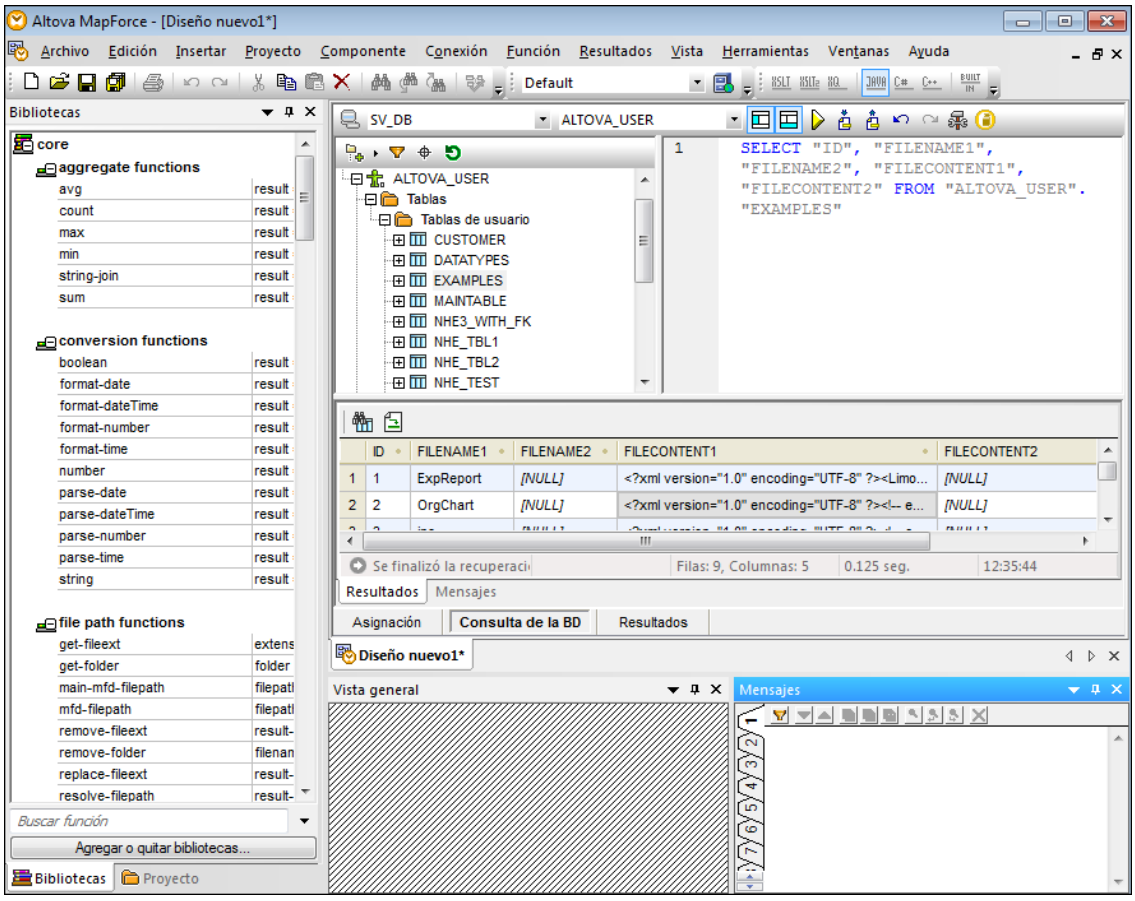
MapForce ofrece un panel especial para la consulta directa de bases de datos. Las consultas y acciones definidas en este panel son independientes de los demás paneles de MapForce y no se guardan en el archivo .MFD.

Esto significa que por cada sesión/asignación puede realizar varias consultas en el panel **Consulta de la BD**. Además puede tener activas varias conexiones a bases de datos diferentes.

Este panel se divide en varias ventanas:

- La ventana **Explorador** (a la izquierda) ofrece información sobre la conexión y muestra las tablas de la BD
- La ventana **Editor SQL** (a la derecha) es donde se escriben las consultas SQL
- La pestaña **Resultados** (en la ventana inferior) muestra los resultados de la consulta en forma de tabla
- La pestaña **Mensajes** (en la ventana inferior) muestra mensajes de error y de advertencia

La barra superior del panel **Consulta de la BD** incluye iconos y comandos para definir las bases de datos de trabajo, la conexión y los esquemas de base de datos.



Con Java seleccionado

Seleccionar y conectarse a la base de datos

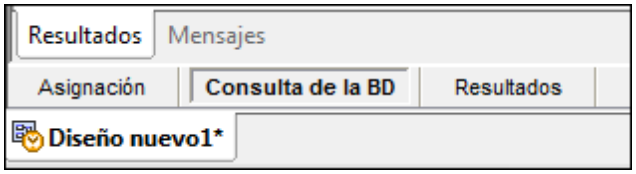
Para poder consultar una base de datos antes es necesario conectarse a ella. Recuerde que en el panel **Consulta de la BD** puede tener activas varias conexiones. Para cambiar de base de datos haga clic en el cuadro combinado del origen de datos.

 © 2014 Altova GmbH

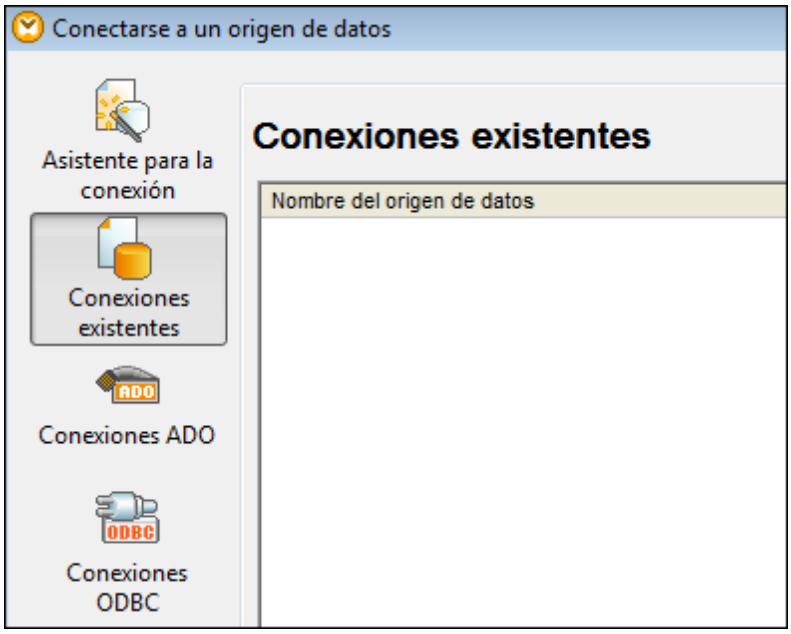
Los ejemplos utilizados en este apartado utilizan una base de datos de muestra que viene con IBM DB2 versión 9.

Para conectarse a una base de datos:

1. Haga clic en la pestaña **Consulta de la BD** para abrir el panel.



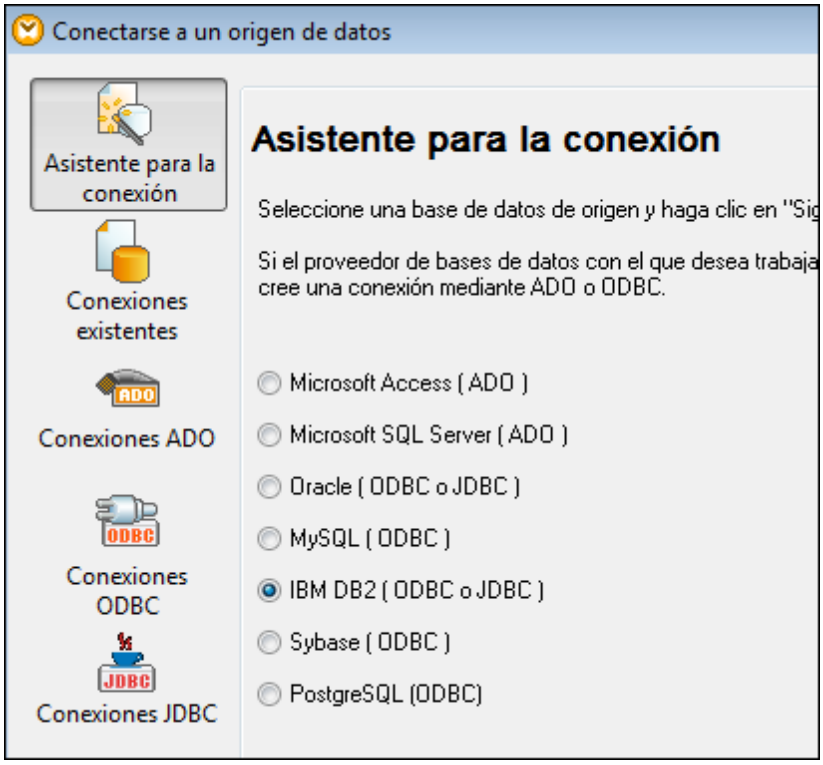
2. Haga clic en el icono de la barra de herramientas para seleccionar el origen de datos.



Esto abre el asistente para la conexión, desde donde puede conectarse a cualquier tipo de BD. Si tiene abiertos archivos de asignación que contienen conexiones a bases de datos, estas conexiones aparecen en el panel **Conexiones existentes** (*imagen anterior*). Para este ejemplo asumimos que no hay ninguna conexión activa.

En el panel **Recursos globales** puede seleccionar bases de datos que se definieron previamente como recursos globales. Consulte el apartado Recursos globales: bases de datos para más información.

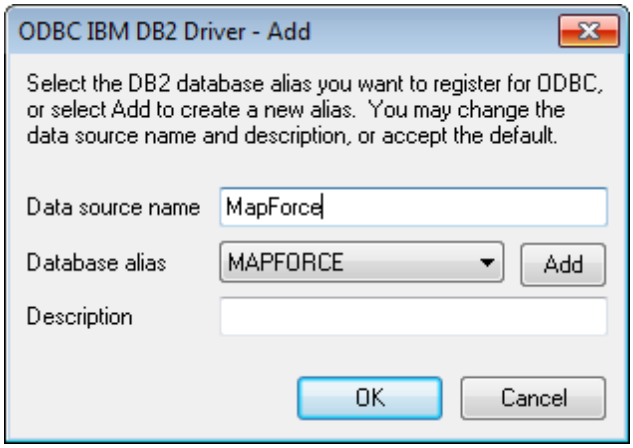
3. Haga clic en el icono **Asistente para la conexión** (*imagen siguiente*) para crear una conexión nueva.



4. Seleccione el tipo de BD a la que desea conectarse (p. ej. *IBM DB2*) y haga clic en **Siguiente**.

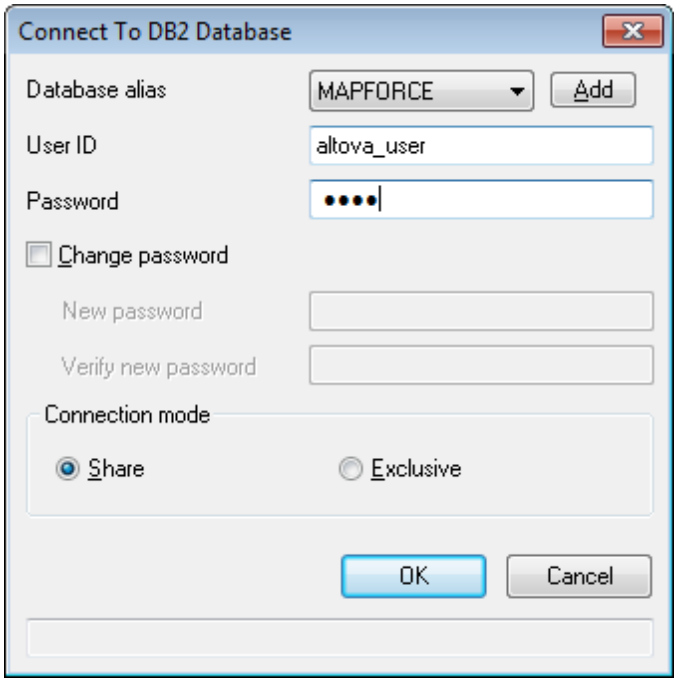
5. Si desea crear un nuevo nombre de origen de datos (DSN), haga clic en **Siguiente**. De lo contrario, haga clic en la opción *Utilizar un DSN ya existente*.

 © 2014 Altova GmbH

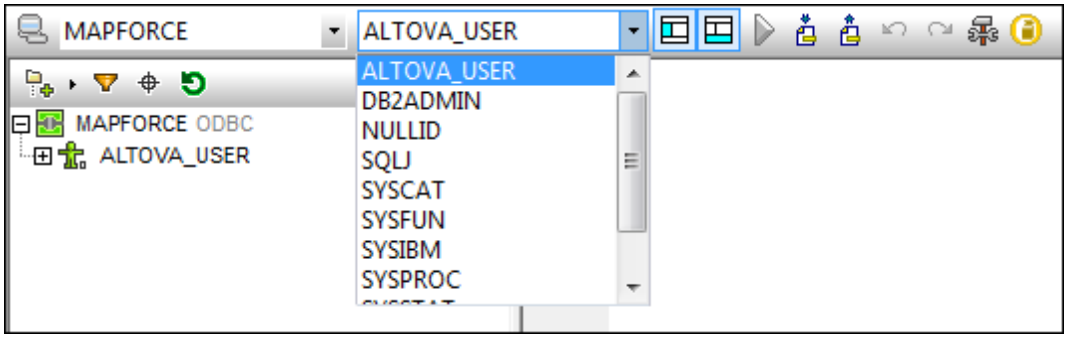


Rellene los campos del siguiente cuadro de diálogo y haga clic en **OK**. Aparece otro cuadro de diálogo donde puede elegir el modo de conexión con el servidor (de forma nativa o por la API de ODBC).

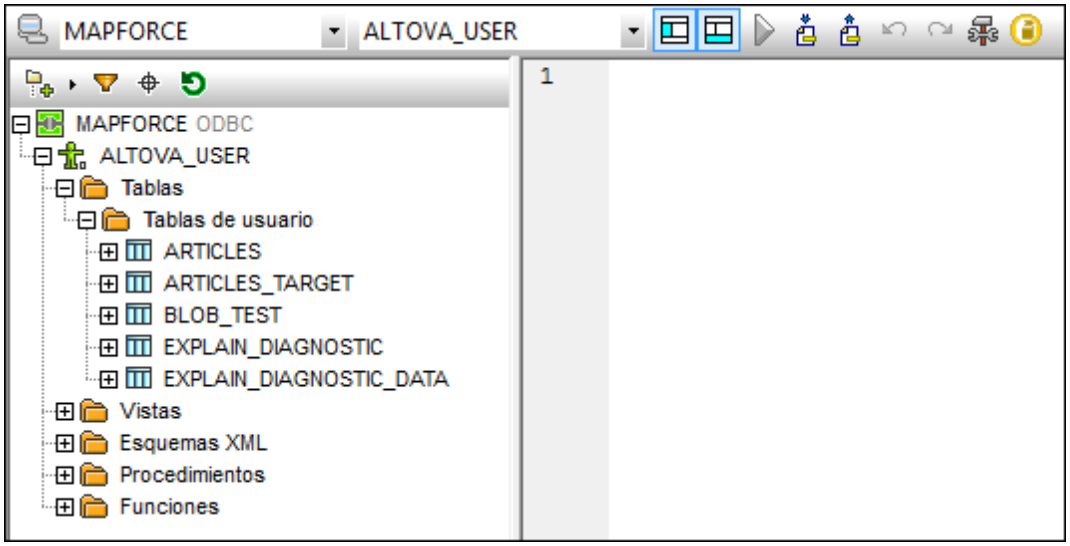
6. Escriba el DSN y el alias de la BD y haga clic en **OK** para continuar.



7. Introduzca el nombre de usuario y contraseña y haga clic en **OK**.
   Como resultado se establece la conexión a la base de datos DB2.



8. En la barra de herramientas superior, haga clic en el cuadro combinado de la derecha y seleccione el objeto raíz (p. ej. el esquema de BD ALTOVA_USER).



Bajo la carpeta *Tablas* aparecen ahora las tablas de la BD. Los objetos raíz de las diferentes bases de datos se enumeran en esta tabla:

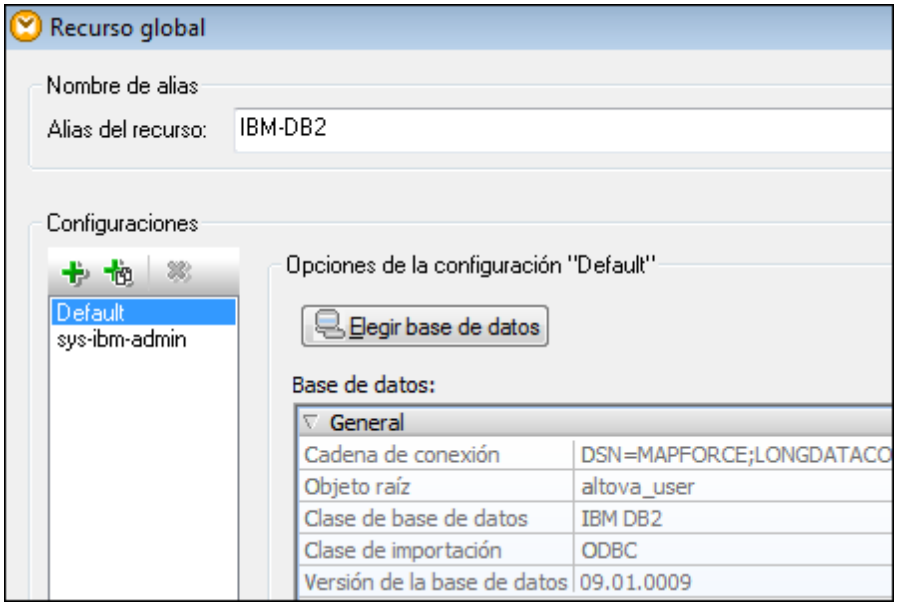
MS SQL Server	database
Oracle	schema
MS Access	database
MySQL	database
DB2	schema
Sybase	database

Seleccionar un recurso global de base de datos

El panel **Recursos globales** del cuadro de diálogo de conexión permite seleccionar una de las bases de datos definidas previamente como recurso global. Consulte el apartado Recursos globales: bases de datos para más información sobre cómo definir una base de datos como recurso global.

Los recursos globales pueden utilizar la misma BD pero tener tablas distintas en cada configuración.
La configuración predeterminada **Default** tiene acceso a las tablas/al objeto raíz `altova_user` de

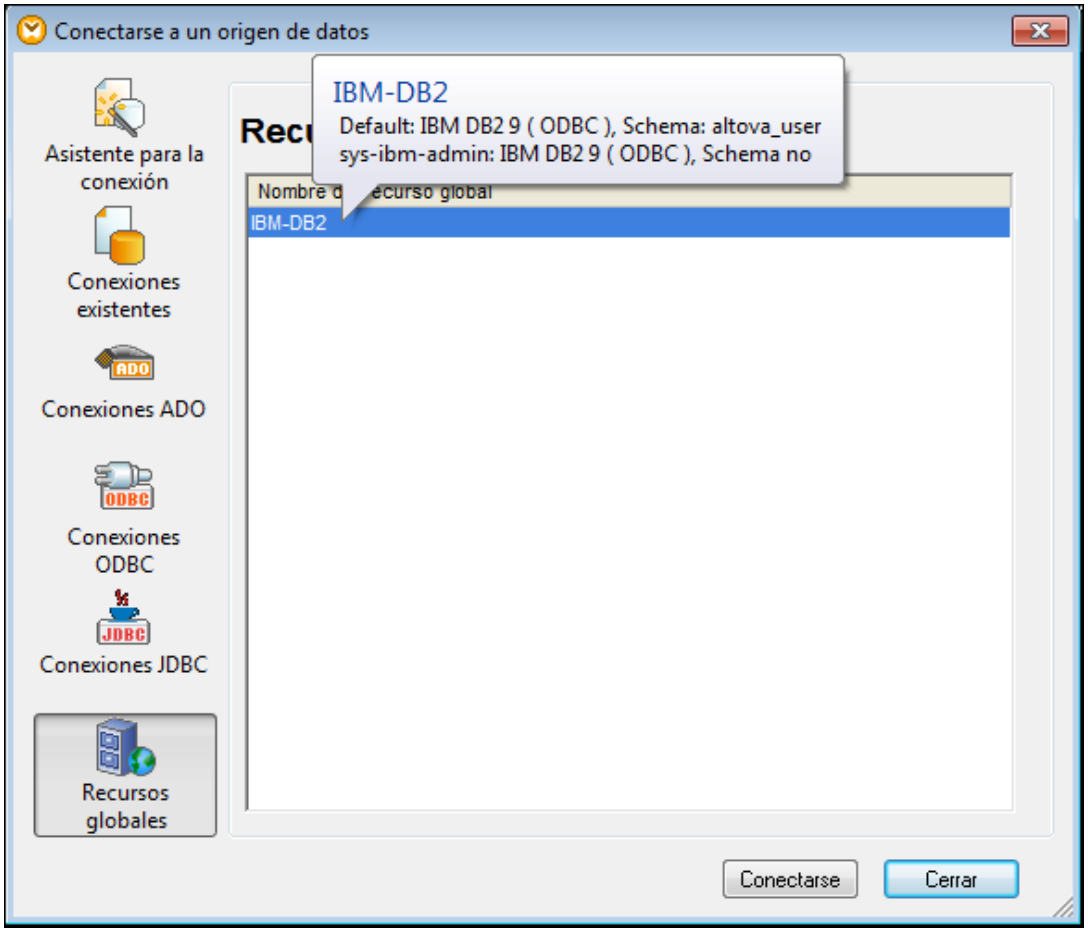
la base de datos DB2 (*imagen siguiente*).



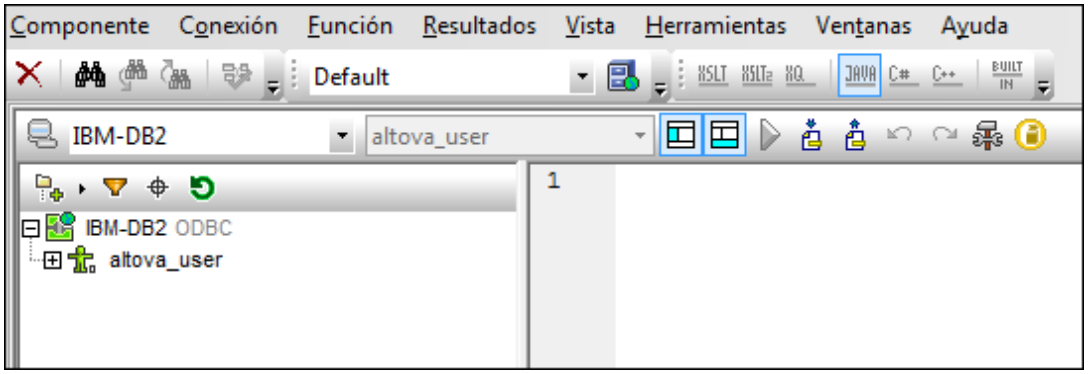
Sin embargo, la configuración **sys-ibm-admin** accede a la misma BD pero a las tablas/al objeto raíz **`SYSIBMADM`** (*imagen siguiente*).

Para cambiar de configuración de recurso global en el panel Consulta de la BD:

1. Abra el panel **Consulta de la BD**.
2. Haga clic en el icono de conexión de la barra de herramientas y haga clic en el icono **Recursos globales**.



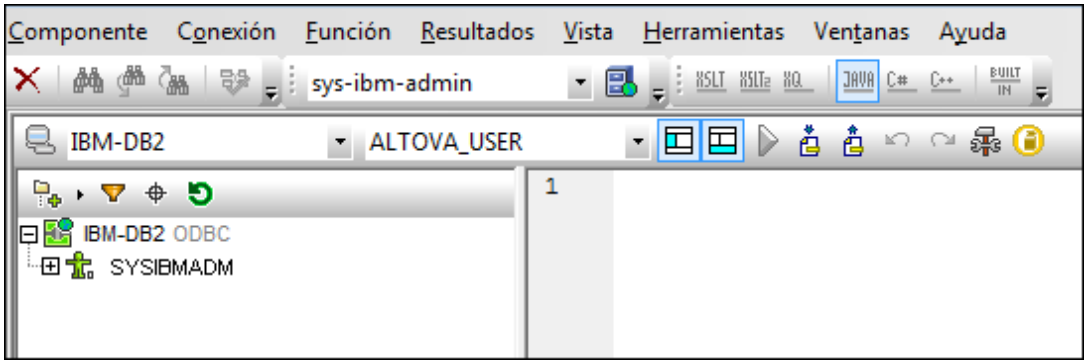
3. Seleccione el nombre del recurso global (p. ej. IBM-DB2) y haga clic en **Conectarse**.



El nombre del recurso global aparece ahora en el cuadro combinado de la izquierda y en el de la derecha aparece atenuado el objeto raíz **SYSCAT**.

4. Haga clic en el cuadro combinado de recursos globales y seleccione la configuración **sys-ibm-admin** (*imagen siguiente*).
   Aparece el cuadro de diálogo "Cambio en la configuración. Volver a cargar" y le pregunta si desea volver a cargar el recurso.
5. Haga clic en el botón **Volver a cargar** para cambiar de configuración.

 © 2014 Altova GmbH

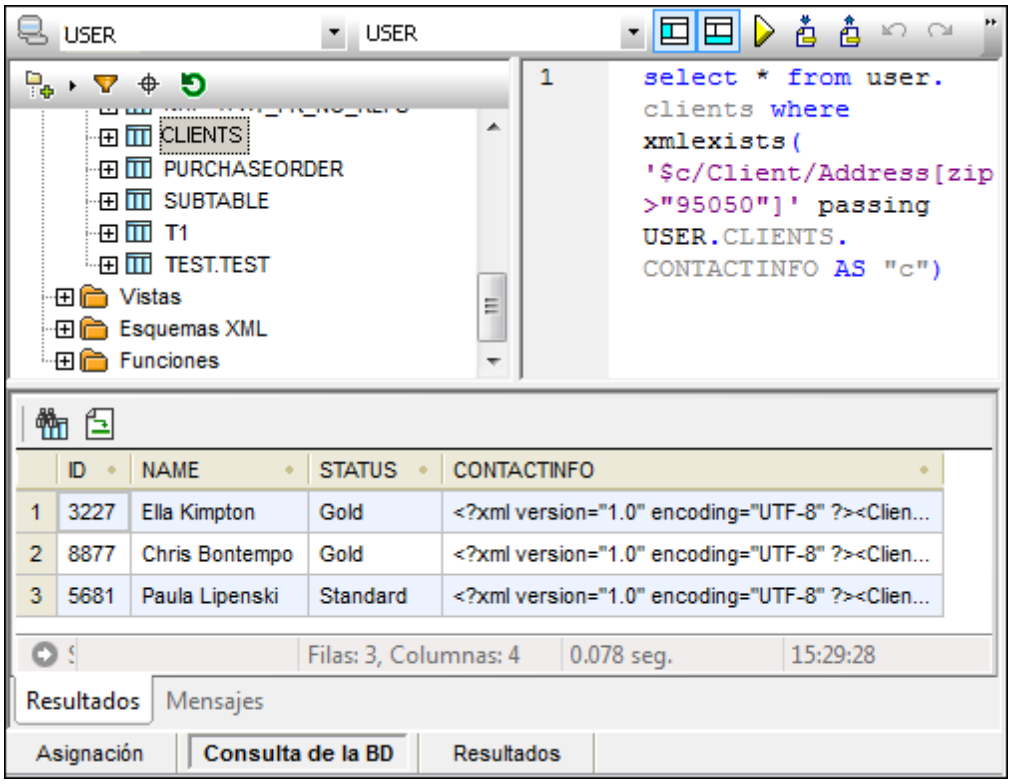


Ahora aparece el nuevo objeto raíz y puede consultar sus tablas.

Consultar los datos

Para consultar los datos de la BD:
Tras conectarse a la BD siguiendo las instrucciones del apartado Seleccionar y conectarse a la base de datos:

1. Haga clic en el icono **Importar archivo SQL** de la barra de herramientas y seleccione el archivo `db2-query.sql` de la carpeta `...\Tutorial`. También puede introducir la instrucción SQL a mano en la ventana del editor SQL si no desea usar el archivo suministrado.
2. Haga clic en el icono **Ejecutar**.



MapForce recupera los datos de la BD y los muestra en forma de tabla en la pestaña ***Resultados*** del panel inferior. Observe que la barra de estado muestra el modo actual (**Recuperación**) y otros datos sobre el conjunto de resultados.

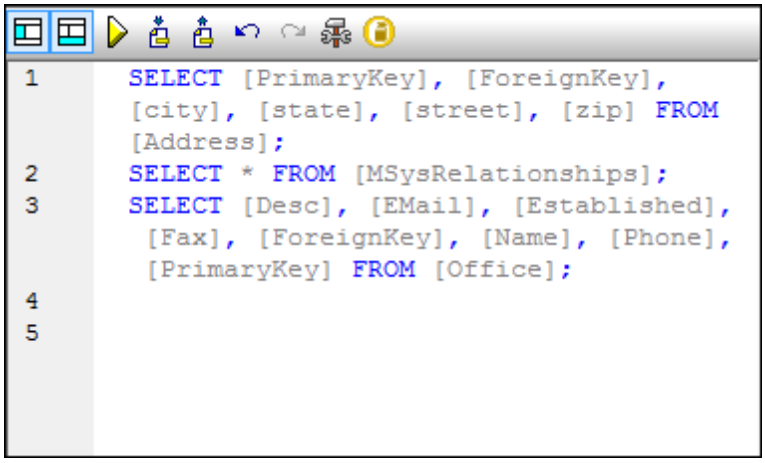
El modo *Recuperación* permite copiar, seleccionar y ordenar los datos de la pestaña ***Resultados***. Para ello haga clic con el botón secundario y seleccione la opción pertinente del menú contextual. Consulte el apartado Consulta de la BD: pestañas Resultados/Mensajes para más información.

Consulta de la BD: editor SQL

El **Editor SQL** sirve para escribir y ejecutar instrucciones SQL.

Esta son las características del **Editor SQL**:

- **generación automática** de instrucciones SQL mediante operaciones de arrastrar y colocar
- **finalización automática** de instrucciones SELECT SQL
- definición de **regiones**
- inserción de **comentarios** de línea o de bloque



La barra de herramientas del **Editor SQL** incluye estos iconos/comandos:

Mostrar/ocultar el explorador: activa o desactiva la ventana **Explorador**.

Mostrar/ocultar resultados: activa o desactiva la pestaña ***Resultados***.

Ejecutar (F5): ejecuta las instrucciones SQL seleccionadas en ese momento. Si existen varias instrucciones y ninguna está seleccionada, se ejecutan todas.

Deshacer: permite deshacer un número ilimitado de cambios realizados en la ventana **Editor SQL**.

Rehacer: permite rehacer cambios deshechos previamente. Puede recorrer el historial de cambios realizados con estos dos comandos.

Importar archivo SQL: abre un archivo SQL en el **Editor SQL**. Después se puede ejecutar este archivo SQL.

 © 2014 Altova GmbH

Exportar archivo SQL: guarda las consultas SQL en un archivo para poder utilizarlas más adelante.

Abrir en DatabaseSpy: inicia DatabaseSpy y abre el script SQL en la ventana **Editor SQL**.

Opciones: abre el cuadro de diálogo Opciones donde puede definir las opciones de configuración del **Editor SQL**.

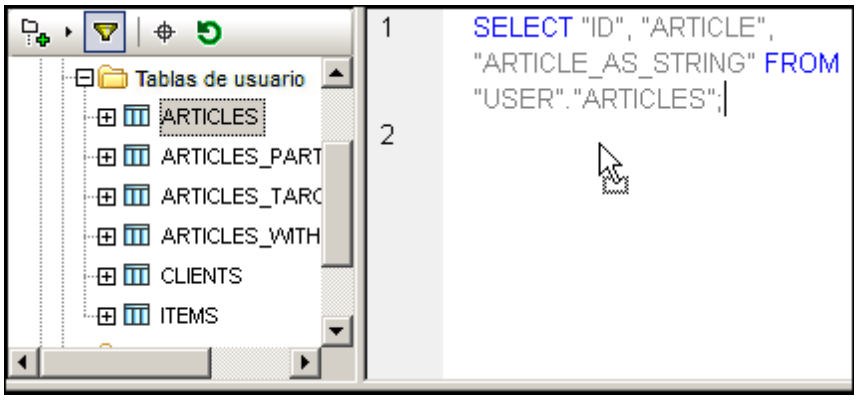
Generar instrucciones SQL

El **Editor SQL** puede generar automáticamente instrucciones SQL basadas en las tablas y columnas de la ventana **Explorador**. Esto se puede conseguir de dos maneras diferentes:

- Arrastrando un objeto de la BD desde la ventana **Explorador** hasta la ventana **Editor SQL**.
- Haciendo clic con el botón secundario en un objeto de la BD en la ventana **Explorador** y seleccionando un comando del menú contextual.

Para generar instrucciones SQL mediante operaciones de arrastrar y colocar:

1. Haga clic en la tabla **ARTICLES** de la ventana **Explorador** y arrástrela hasta la ventana **Editor SQL**.



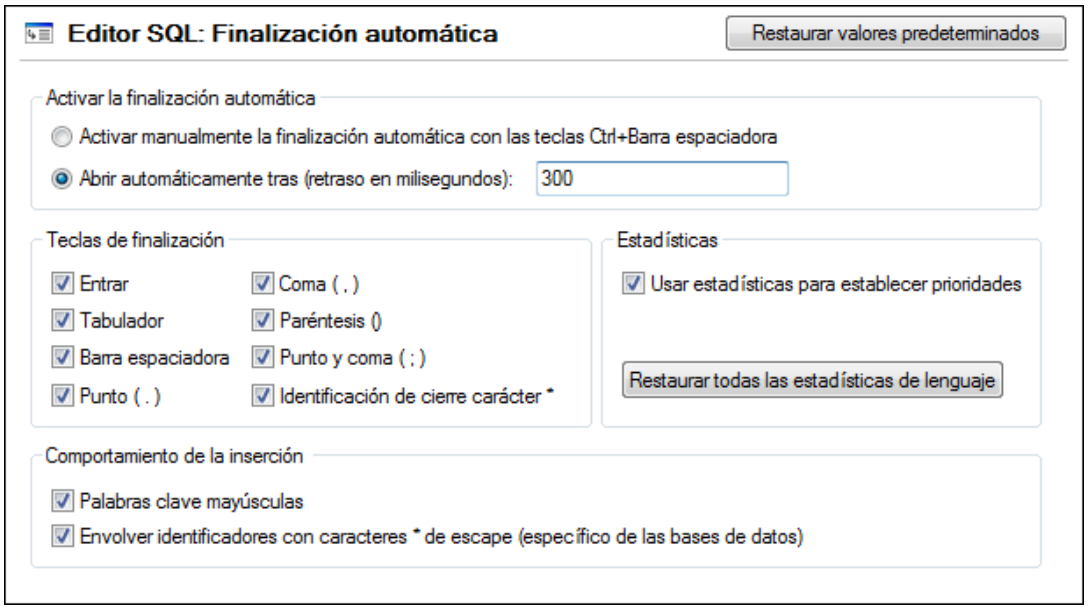
Aparece una instrucción SQL en la ventana **Editor SQL**.

Para generar instrucciones SQL con el menú contextual:

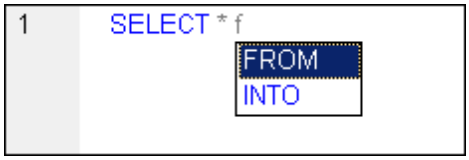
1. Haga clic con el botón secundario en un objeto de la BD de la ventana **Explorador** y seleccione **Mostrar en editor SQL | SELECT**.

Para crear instrucciones SQL a mano con ayuda de la función de finalización automática:

1. Haga clic en el icono **Opciones** de la barra de herramientas y seleccione **Editor SQL | Finalización automática** en el panel lateral.
2. Defina las opciones de finalización automática que desea utilizar.



3. Empiece a teclear su instrucción SQL en el **Editor SQL**.



Aparece una lista emergente de finalización automática. Pulse la tecla **Entrar** si desea insertar/usar la opción resaltada.

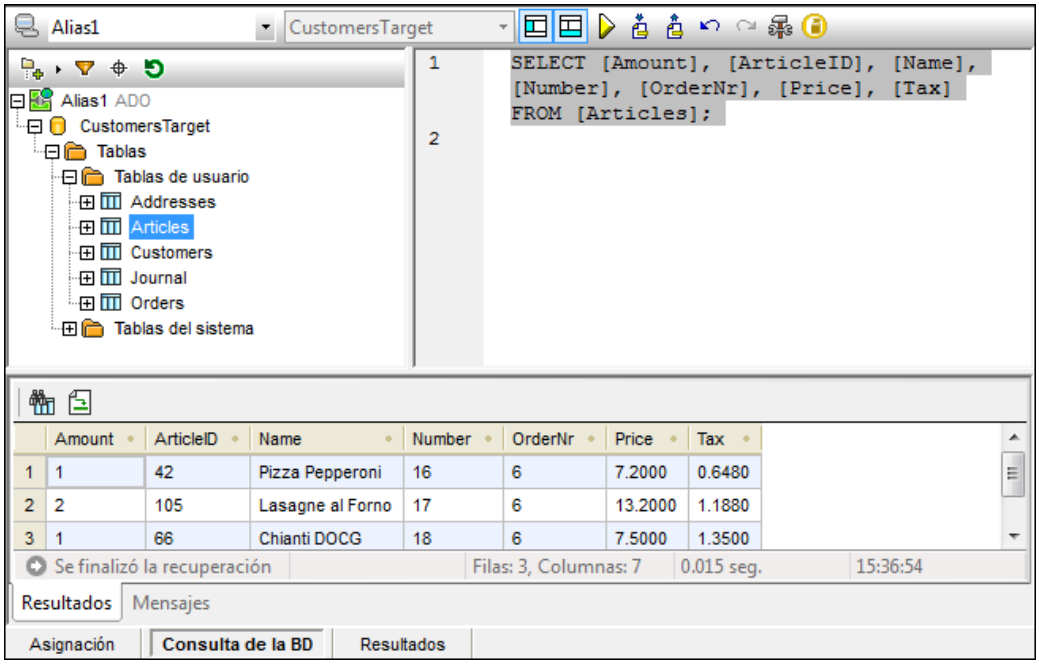
Ejecutar instrucciones SQL

Las instrucciones SQL se pueden ejecutar directamente desde el **Editor SQL**.

Para ejecutar instrucciones SQL desde la ventana Editor SQL:

1. Escriba o seleccione una instrucción SQL en el **Editor SQL**.
2. Haga clic en el icono **Ejecutar** .

 © 2014 Altova GmbH



Si no existe una conexión a un origen de datos, aparece un mensaje de aviso preguntándole si desea conectarse a la BD.

3. Haga clic en **Sí** para conectarse al origen de datos.

Para seleccionar una sola instrucción SQL:

- Marque la instrucción con el mouse.
- Apunte el cursor del mouse al margen de numeración de líneas del **Editor SQL** y haga clic para seleccionar toda la instrucción.
- Haga clic tres veces en la instrucción.

Guardar y abrir scripts SQL

Las instrucciones SQL de la ventana **Editor SQL** se pueden guardar en un archivo para utilizarlas más adelante.

Para guardar el contenido de la ventana Editor SQL en un archivo:

1. Haga clic en el icono **Exportar archivo SQL** y escriba el nombre del script SQL.

Para abrir un archivo SQL ya existente:

1. Haga clic en el icono **Importar archivo SQL** y seleccione el archivo SQL que desea abrir en la ventana SQL.

Características del editor SQL

En el **Editor SQL** puede editar instrucciones SQL igual que en cualquier otro editor de texto. Sin embargo, el **Editor SQL** ofrece algunas características especiales:

- finalización automática
- creación de comentarios
- marcadores
- regiones

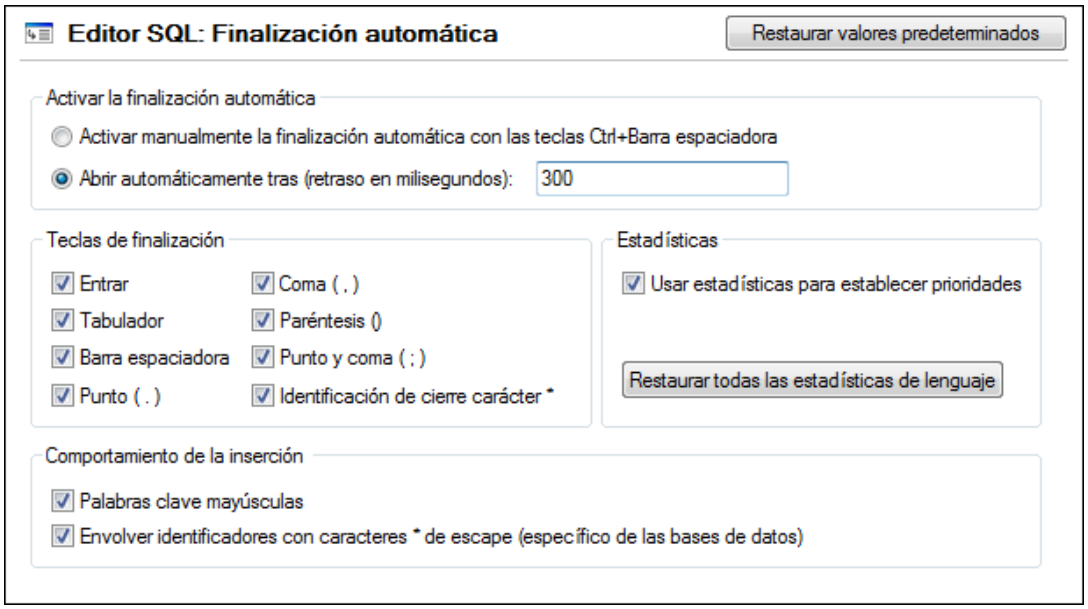
Finalización automática

En el Editor SQL puede escribir instrucciones SQL con ayuda de la función de finalización automática, que sugiere palabras clave, tipos de datos, identificadores, separadores y operadores mientras teclea, según el tipo de instrucción que se esté creando en cada momento. La función de finalización automática es compatible con estas bases de datos:

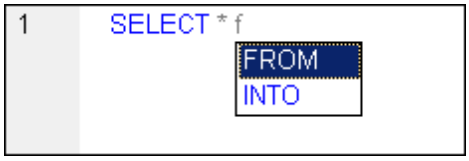
- MS SQL Server 2000
- MS SQL Server 2005
- MS Access 2003
- IBM DB2 9

Para activar la función de finalización automática:

1. Haga clic en el icono **Opciones** de la barra de herramientas.
   Aparece el cuadro de diálogo "Opciones". En el panel izquierdo seleccione **Editor SQL | Finalización automática**.
2. Defina las opciones de finalización automática que desea utilizar.



3. Empiece a teclear su instrucción SQL en el **Editor SQL**.



Aparece una lista emergente de finalización automática. Pulse la tecla **Entrar** si desea insertar/usar la opción resaltada.

Convertir texto en comentario

En MapForce puede convertir instrucciones, parte de instrucciones o grupos de instrucciones del **Editor SQL** en comentarios. Las instrucciones convertidas en comentarios se ignoran cuando se

 © 2014 Altova GmbH

ejecuta el script SQL.

Para convertir una parte del texto en comentario:

1. Seleccione una instrucción o parte de la instrucción.

```
1   select * from user.clients where xmlexists(
    '$c/Client/Address[zip>"95050"]' passing USER
    .CLIENTS.CONTACTINFO AS "c")
2
3   SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU", "SRP",
    "COMMENTS" FROM "USER"."ITEMS";
4
5   SELECT [PrimaryKey], [ForeignKey], [Name] FROM [
    Department];
```

2. Haga clic con el botón secundario en el texto seleccionado y elija la opción **Convertir el bloque en comentario/quitar comentario**.

```
1   select * from user.clients where xmlexists(
    '$c/Client/Address[zip>"95050"]' passing USER
    .CLIENTS.CONTACTINFO AS "c")
2
3   /*SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU",
    "SRP", "COMMENTS" FROM "USER"."ITEMS";*/
4
5   SELECT [PrimaryKey], [ForeignKey], [Name] FROM [
    Department];
```

La instrucción se convierte en comentario y se ignorará durante la ejecución.

Para convertir una línea en comentario:

1. Haga clic con el botón secundario en la posición que desea convertir en comentario y seleccione la opción **Convertir la línea en comentario/quitar comentario**.

```
3   SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU", "SRP",
    "COMMENTS" FROM "USER"."ITEMS";
4
5   SELECT [PrimaryKey], --[ForeignKey], [Name] FROM
    [Department];
6
```

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 secundario y seleccione la opción **Convertir el bloque (o la línea) en comentario/quitar comentario**.

Marcadores

Los marcadores sirven para marcar puntos de interés en los scripts de gran tamaño.

Para insertar un marcador:

1. Haga clic con el botón secundario en la línea en la que desea insertar el marcador y seleccione **Insertar o quitar marcador** en el menú contextual.

```
1   select * from user.clients where xmlexists(
    '$c/Client/Address[zip>"95050"]' passing USER
    .CLIENTS.CONTACTINFO AS "c")
2
3   SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU",
    "SRP", "COMMENTS" FROM "USER"."ITEMS";
4
5
6   SELECT [PrimaryKey], [ForeignKey], [Name] FROM [
    Department];
7
8
9   SELECT "MED", "DECISION", "NAME", "UPDATE_TIME",
    "POLICY" FROM "SYSTOOLS"."POLICY";
10
```

El icono aparece en el margen de la línea que incluye el marcador.

Para quitar un marcador:

1. Haga clic con el botón secundario en la línea que incluye el marcador 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 secundario y seleccione **Ir al siguiente marcador**.
- Para ir al marcador anterior, haga clic con el botón secundario y seleccione **Ir al marcador anterior**.

Para quitar todos los marcadores:

- Haga clic con el botón secundario en el **Editor SQL** y seleccione **Quitar todos los marcadores**.

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 `--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`.

 © 2014 Altova GmbH

Para crear una región:

1. En el **Editor SQL** seleccione las instrucciones que desea insertar en una región.

```
1   select * from user.clients where xmlexists(
    '$c/Client/Address[zip>"95050"]' passing USER
    .CLIENTS.CONTACTINFO AS "c")
2
3   SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU",
    "SRP", "COMMENTS" FROM "USER"."ITEMS";
4
5
6   SELECT [PrimaryKey], [ForeignKey], [Name] FROM [
    Department];
7   SELECT "MED", "DECISION", "NAME", "UPDATE_TIME",
    "POLICY" FROM "SYSTOOLS"."POLICY";
8
```

2. Haga clic con el botón secundario y seleccione la opción **Insertar región** del menú contextual.
   El área seleccionada se convierte en una región que se puede expandir o contraer.

```
1   -- region
2   select * from user.clients where xmlexists(
    '$c/Client/Address[zip>"95050"]' passing USER
    .CLIENTS.CONTACTINFO AS "c")
3   -- endregion
4
5   SELECT "ID", "BRANDNAME", "ITEMNAME", "SKU",
    "SRP", "COMMENTS" FROM "USER"."ITEMS";
6
7
8   SELECT [PrimaryKey], [ForeignKey], [Name] FROM [
    Department];
9   SELECT "MED", "DECISION", "NAME", "UPDATE_TIME",
```

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`.

Consulta de la BD: ventana Explorador

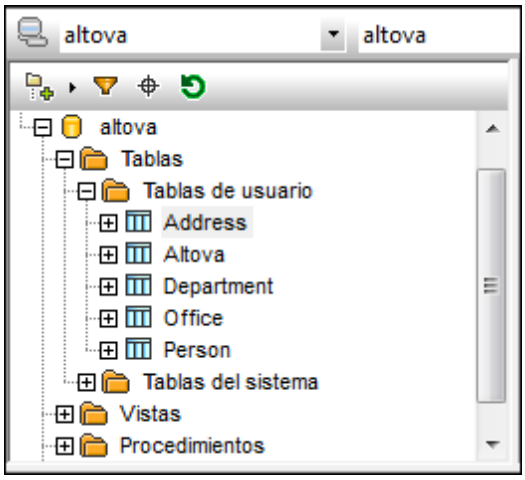
La ventana **Explorador** ofrece un resumen de los objetos de la BS, incluida la información sobre restricciones (p. ej. una columna es una clave principal o foránea). En las bases de datos IBM DB2, el **Explorador** también muestra en una carpeta separada los esquemas XML registrados en la BD.

Además puede personalizar la ventana **Explorador** para:

- visualizar los objetos de la BD en varias **disposiciones** distintas
- **buscar** determinados objetos de la BD con el localizador de objetos
- **filtrar** el número de elementos que aparecen en el **Explorador**

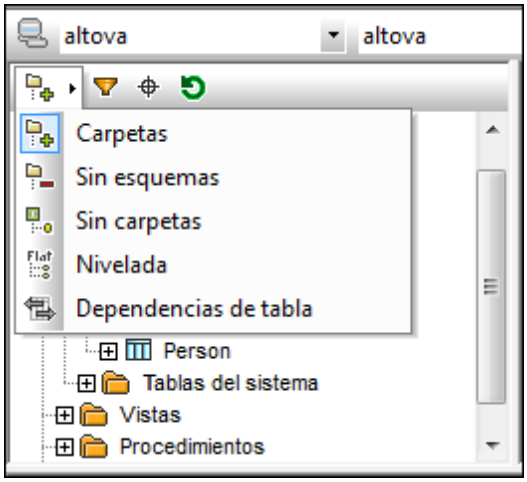
- **actualizar** el objeto raíz del origen de datos activo

  La *disposición en carpetas* predeterminada muestra los objetos de la BD de forma jerárquica. Al hacer clic con el botón secundario en un objeto aparece un menú contextual, cuyas opciones dependen del tipo de objeto seleccionado.



Para seleccionar la disposición de los objetos en el Explorador:

- Haga clic en el icono de la barra de herramientas del **Explorador** y seleccione la disposición que prefiera en la lista desplegable.
  Observe que el icono cambia con cada opción.



Disposición de los objetos en la ventana Explorador
La ventana **Explorador** puede mostrar los objetos de la BD en varias disposiciones diferentes:

- La ***disposición en carpetas*** organiza los objetos de la BD en carpetas y en una estructura jerárquica en forma de árbol que depende del tipo de objeto (se trata de la disposición predeterminada).
- La ***disposición sin esquemas*** es igual que la disposición en carpetas excepto que no hay una carpeta para los esquemas de BD. Es decir, las tablas no se ordenan por su esquema de BD .
- La ***disposición sin carpetas*** muestra los objetos de la BD de forma jerárquica pero sin

usar carpetas.

- La ***disposición nivelada*** divide los objetos de la BD por tipo en el primer nivel jarárquico. Por ejemplo, en lugar de mostrar las columnas dentro de su tabla, todas las columnas aparecen en la carpeta *Columnas*.
- La disposición por ***dependencias de tabla*** ordena las tablas según su relación con otras tablas. Las categorías son tablas con claves foráneas, tablas a las que hacen referencia claves foráneas, tablas sin relaciones, etc.

Para ordenar las tablas por tablas de usuario y del sistema:

1. Haga clic con el botón secundario en la carpeta *Tablas* de la ventana **Explorador**. Aparece un menú contextual.
2. Seleccione el comando **Ordenar por tablas de usuario y del sistema**.
   Las tablas se ordenan por orden alfabético en las carpetas *Tablas de usuario* y *Tablas del sistema*.

   Nota: esta función está disponible solamente en la disposición en carpetas, disposición sin esquemas y disposición nivelada.

Para actualizar el objeto raíz del origen de datos activo:

- Haga clic en el icono **Actualizar** de la barra de herramientas.

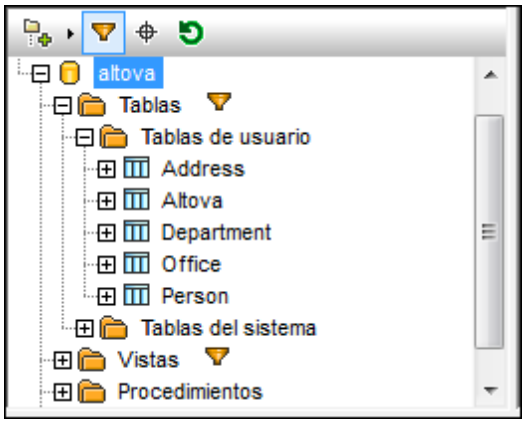
Filtrar y buscar objetos de la BD

En la ventana **Explorador** puede filtrar los esquemas, las tablas y las vistas por nombre o parte del nombre. Los objetos se filtran en tiempo real mientras se teclean caracteres en el filtro. Por defecto, la función de filtrado no diferencia entre mayúsculas y minúsculas.

Nota: la función de filtrado no funciona con la disposición sin carpetas.

Para filtrar objetos en la ventana Explorador:

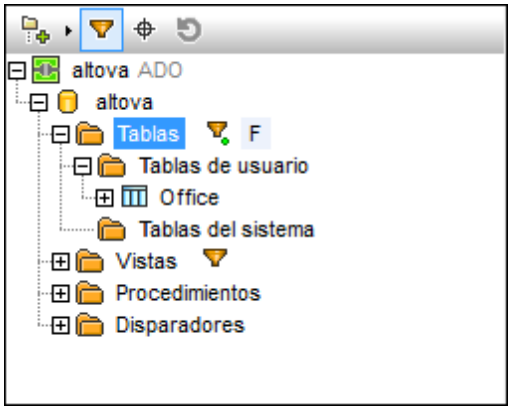
1. Haga clic en el icono **Filtrar contenido de carpeta** de la barra de herramientas. Junto a las carpetas aparecen iconos en forma de embudo.



2. Haga clic en el icono de la carpeta que desea filtrar y seleccione una opción de filtrado en el menú emergente (p. ej. **Contiene**).

Junto al icono aparece un campo de texto vacío.

3. Escriba la cadena de texto que desea usar para filtrar el contenido de la carpeta (p. ej. **F**). El filtro se aplica en tiempo real a medida que se teclea.



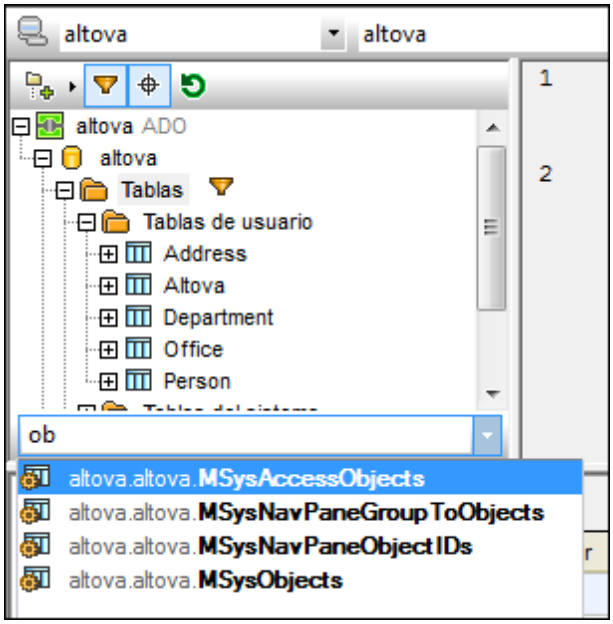
Buscar objetos de la base de datos

Para buscar un elemento de la BD por su nombre, puede usar la función de filtrado o el ***localizador de objetos*** de la ventana **Explorador**.

Para buscar elementos de la BD con el localizador de objetos:

1. Haga clic en el icono **Localizador de objetos** de la ventana **Explorador**.
   Aparece una lista desplegable en la parte inferior de la ventana **Explorador**.
2. Escriba la cadena de texto que desea buscar (p. ej. **ob**).
   Haga clic en la flecha desplegable y aparece una lista con los elementos que contienen la cadena de búsqueda..

 © 2014 Altova GmbH



3. Haga clic en el objeto pertinente para verlo en la ventana **Explorador**.

Opciones contextuales de la ventana Explorador

En la ventana **Explorador** hay varios menús contextuales:

- El que aparece al hacer clic con el botón secundario en el **objeto raíz**
- El que aparece al hacer clic con el botón secundario en una **carpeta**, p. ej. la carpeta *Tablas de usuario*
- El que aparece al hacer clic con el botón secundario en un **objeto de la BD**, p. ej. en la tabla **ARTICLES**

Haga clic con el botón secundario en el objeto raíz para **actualizar** la base de datos.

Haga clic con el botón secundario en una carpeta para:
Expandir | Elementos del mismo nivel | Elementos secundarios
Contraer | Elementos del mismo nivel | Elementos secundarios

Haga clic con el botón secundario en un objeto de la BD para:
Mostrar el objeto en el editor SQL.

Nota: la sintaxis de las instrucciones puede variar dependiendo del tipo de BD que esté usando. En los ejemplos que ofrecemos más abajo usamos Microsoft SQL Server 2000. Utilice **Mayús+Clic** y **Ctrl+Clic** para seleccionar varios objetos de la BD al mismo tiempo.

Al hacer clic con el botón secundario en una **tabla** aparece un menú contextual con estas opciones:

- **Mostrar en editor SQL | SELECT**: crea una instrucción SELECT que recupera los datos de toda las columnas de la tabla de origen.
  P. ej. SELECT "ID", "ARTICLE", "ARTICLE_AS_STRING" FROM "USER"."ARTICLES";
- **Mostrar en editor SQL | Nombre**: devuelve el nombre de la tabla. Los nombres aparecen en líneas distintas, separadas por comas.

- **Mostrar en editor SQL | Ruta de acceso**: devuelve la ruta completa de las tablas, es decir, `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla`. También puede seleccionar varias tablas. Las rutas de acceso aparecen en líneas distintas, separadas por comas.

Al hacer clic con el botón secundario en una **columna** aparece un menú contextual con estas opciones:

- **Mostrar en editor SQL | SELECT**: crea una instrucción SELECT que recupera los datos de las columnas seleccionados de la tabla primaria.
  P. ej. SELECT "ARTICLE" FROM "USER"."ARTICLES";
- **Mostrar en editor SQL | Nombre**: devuelve el nombre de las columnas seleccionadas. Los nombres aparecen en líneas distintas, separadas por comas.
- **Mostrar en editor SQL | Ruta de acceso**: devuelve la ruta completa de las columnas, es decir, `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla.NombreColumna`. También puede seleccionar varias columnas. Las rutas de acceso aparecen en líneas distintas, separadas por comas.

Al hacer clic con el botón secundario en una **restricción** aparece un menú contextual con estas opciones:

- **Mostrar en editor SQL | Nombre**: devuelve el nombre de la restricción seleccionada. Los nombres aparecen en líneas distintas, separadas por comas.
- **Mostrar en editor SQL | Ruta de acceso**: devuelve la ruta completa de la restricción, es decir, `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla.NombreRestricción`. Las rutas de acceso aparecen en líneas distintas, separadas por comas.

  P. ej "USER"."ARTICLES_PARTIAL_TARGET"."CC1175533269606"

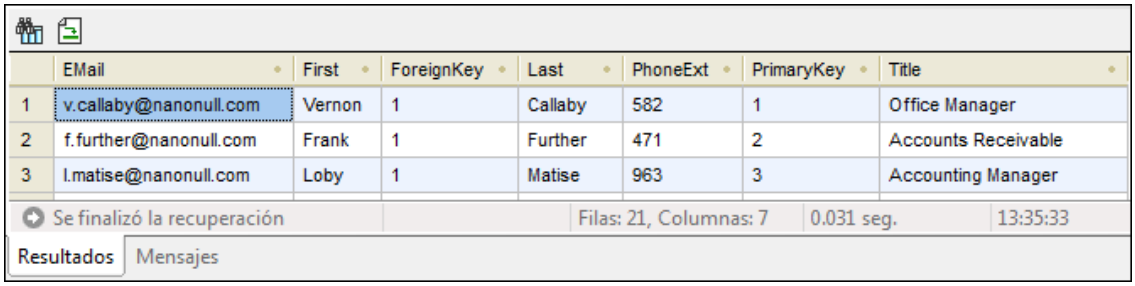
Al hacer clic con el botón secundario en un **índice** aparece un menú contextual con estas opciones:

- **Mostrar en editor SQL | Nombre**: devuelve el nombre del índice seleccionado. Los nombres aparecen en líneas distintas, separadas por comas.
- **Mostrar en editor SQL | Ruta de acceso**: devuelve la ruta completa del índice, es decir, `NombreOrigenDatos.NombreBaseDatos.NombreEsquema.NombreTabla.NombreÍndice`. Las rutas de acceso aparecen en líneas distintas, separadas por comas.

Consulta de la BD: pestañas Resultados/Mensajes

La pestaña ***Resultados*** del **Editor SQL** muestra el conjunto de registros recuperado como resultado de una consulta de BD.

 © 2014 Altova GmbH



	EMail	First	ForeignKey	Last	PhoneExt	PrimaryKey	Title
1	v.callaby@nanonull.com	Vernon	1	Callaby	582	1	Office Manager
2	f.further@nanonull.com	Frank	1	Further	471	2	Accounts Receivable
3	l.matise@nanonull.com	Loby	1	Matise	963	3	Accounting Manager

Se finalizó la recuperación | Filas: 21, Columnas: 7 | 0.031 seg. | 13:35:33

Resultados | Mensajes

Seleccionar y copiar datos en la pestaña *Resultados*:

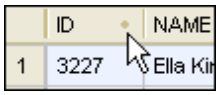
Hay varias maneras de seleccionar datos en esta pestaña para copiarlos y pegarlos en otras aplicaciones.

- Haga clic en el encabezado de una columna para seleccionar los datos de la columna.
- Haga clic en el número de fila para seleccionar los datos de la fila.
- Haga clic en las celdas que le interesen (pulsando la tecla **Ctrl** mientras hace clic en las celdas puede seleccionar varias celdas a la vez). Si una columna/celda contiene datos XML también puede copiar estos datos.
- Haga clic con el botón secundario y seleccione la opción **Copiar celdas seleccionadas** en el menú contextual.

  Nota: también puede seleccionar datos haciendo clic con el botón secundario y seleccionando la opción **Seleccionar | Fila** o **Columna** o **Todo.**

Para ordenar los datos de la pestaña *Resultados*:

- Haga clic con el botón secundario en la columna que desea ordenar y seleccione **Orden | Ascendente** o **Descendente** en el menú contextual.
- Haga clic en el icono de ordenación del encabezado de columna



	ID	NAME
1	3227	Ella Kir

  Los datos se ordenan según el contenido de la columna ordenada.

Para restaurar el criterio de ordenación predeterminado:

- Haga clic con el botón secundario dentro de la tabla y elija la opción **Orden | Restaurar valor predeterminado** en el menú contextual.

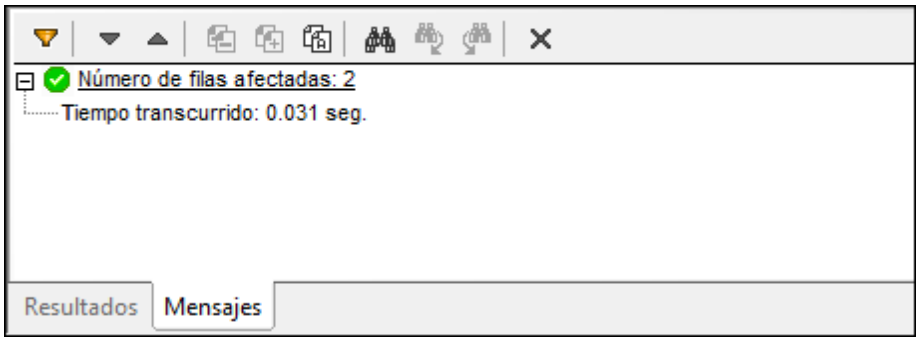
Pestaña *Resultados*: opciones de la barra de herramientas

La pestaña ***Resultados*** incluye una barra de herramientas con comandos para navegar por los resultados y las instrucciones SQL. Estos comandos permiten recuperar con facilidad los datos de la BD.

Buscar: busca la cadena de búsqueda en la pestaña ***Resultados***. Pulse **F3** para seguir buscando el término de búsqueda.

Ir a la instrucción: pasa a la ventana **Editor SQL** y resalta el grupo de instrucciones SQL que produjeron el resultado actual.

La pestaña ***Mensajes*** del **Editor SQL** ofrece información concreta sobre las instrucciones SQL ejecutadas previamente así como mensajes de error y advertencia.



Puede usar varios filtros para personalizar la vista de la pestaña ***Mensajes*** o usar los botones **Siguiente** y **Anterior** para examinar la pestaña fila por fila. La pestaña ***Mensajes*** también ofrece una función de **búsqueda** y varios comandos para copiar el texto en el portapapeles.

Pestaña *Mensajes*: opciones de la barra de herramientas

La ventana ***Mensajes*** incluye una barra de herramientas con opciones para navegar por los mensajes y para filtrar los mensajes.

Filtrar: al hacer clic en este icono aparece un menú donde puede elegir qué información aparece en la pestaña (**Resumen**, **Operación correcta**, **Advertencia**, **Error**). También puede activar/desactivar todas estas opciones con un solo clic con **Activar todos** / **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 aparición anterior de la cadena de búsqueda indicada en el cuadro de diálogo Buscar.

Buscar siguiente: pasa a la siguiente aparición de la cadena de búsqueda indicada en el cuadro de diálogo Buscar.

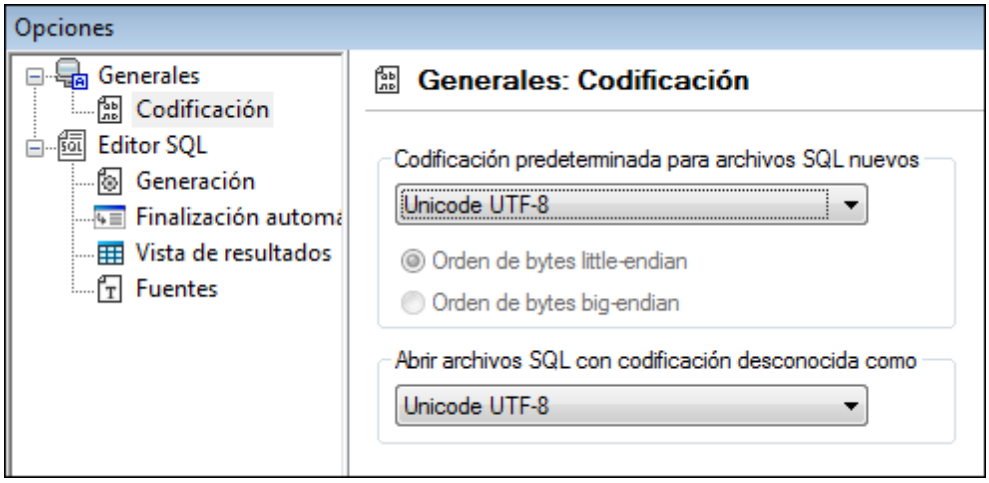
Borrar: borra todos los mensajes que aparecen en la pestaña ***Mensajes***.

Nota: estos comandos también están disponibles en el menú contextual de la pestaña ***Mensajes***.

 © 2014 Altova GmbH

Consulta de la BD: configuración

En la página **Codificación** del cuadro de diálogo "Opciones" puede definir varias opciones de codificación de archivos.



Codificación predeterminada para archivos XML nuevos
Define la codificación predeterminada de los archivos nuevos. Si selecciona una codificación predeterminada de 2 o 4 bytes (es decir, UTF-16, UCS-2 o UCS-4), también puede elegir si el orden de bytes es little-endian o big-endian.

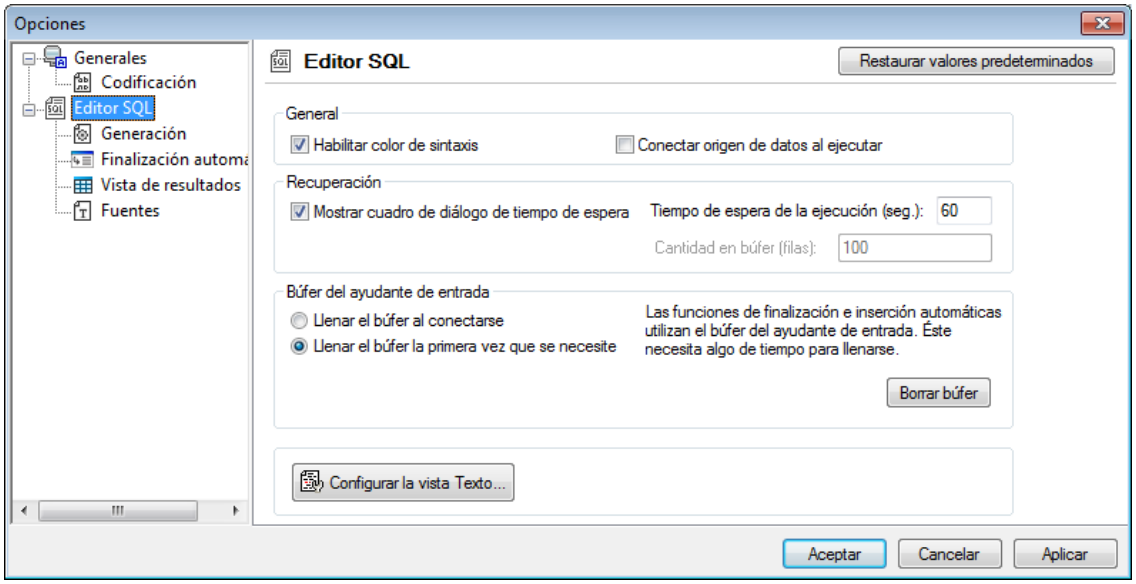
La codificación de los archivos ya existentes se mantiene. Esta codificación solamente afecta a los archivos SQL nuevos.

Abrir archivos SQL con codificación desconocida como
Puede seleccionar qué codificación se utiliza para abrir los archivos SQL cuya codificación no se especifica o no se pueda detectar.

Nota: los archivos SQL que no tienen especificación de codificación se guardan correctamente con la codificación UTF-8.

Opciones del editor SQL

En la página principal del cuadro de diálogo "Opciones" del **Editor SQL** puede definir el aspecto del editor y las opciones de finalización automática. El resto de las funciones del **Editor SQL** se pueden configurar en las páginas Generación, Finalización automática, Vista de resultados y Fuentes del cuadro de diálogo.



Generales
Mientras teclea en el **Editor SQL**, la función de finalización automática le sugiere palabras clave SQL y nombres de objetos de la BD en una lista desplegable. En el grupo de opciones *General* puede activar determinadas funciones de finalización automática. Estas funciones se pueden configurar después en la página ***Finalización automática*** del cuadro de diálogo "Opciones".

El *color de sintaxis* resalta los diferentes elementos de la sintaxis SQL usando varios colores.

Si marca la casilla *Conectar origen de datos al ejecutar,* el **Editor SQL** se conecta al origen de datos correspondiente de forma automática cuando se ejecuta un archivo SQL y el origen de datos no está conectado.

Recuperación
En el grupo de opciones *Recuperación* puede especificar el tiempo máximo permitido (en segundos) para la ejecución SQL en la opción *Tiempo de espera de la ejecución...*

Si marca la casilla *Mostrar cuadro de diálogo de tiempo de espera*, podrá cambiar la configuración del tiempo de espera cuando se exceda el tiempo de ejecución permitido.

Búfer del ayudante de entrada
En el grupo de opciones *Búfer del ayudante de entrada* puede definir cómo se llena el búfer del ayudante de entrada: al conectarse o solamente la primera vez que se necesite.

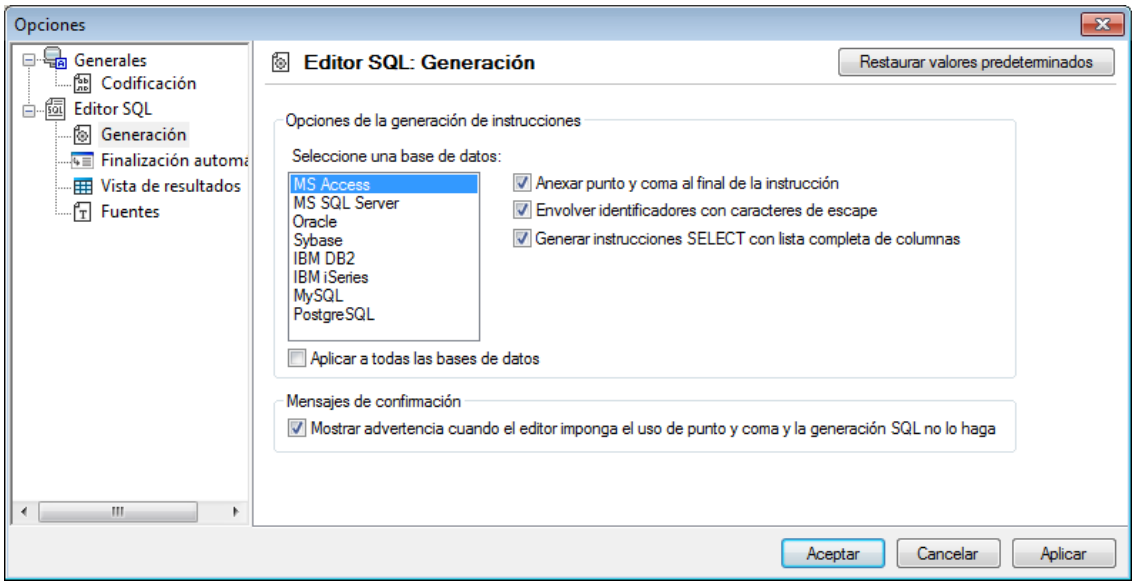
Configurar la vista Texto
Haga clic en este botón para definir la configuración de la vista Texto: defina sus márgenes, guías visuales, etc.

Generación

En la página ***Generación*** del cuadro de diálogo "Opciones" puede definir la sintaxis de generación de instrucciones para todos los tipos de BD compatibles.

Haga clic en una BD del cuadro de lista para definir su sintaxis con las tres casillas situadas a la derecha.
Marque la casilla "Aplicar a todas las bases de datos" situada bajo el cuadro de lista para aplicar

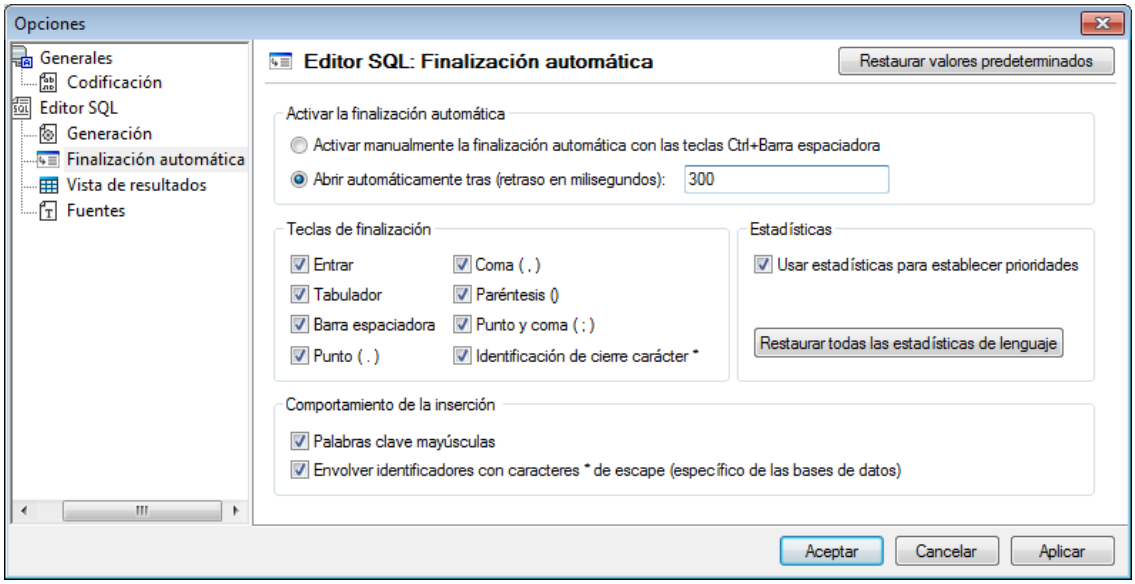
las opciones de las tres casillas a todas las BD de la lista.



Tenga en cuenta que si marca esta casilla, no podrá editar datos de bases de datos Oracle e IBM iSeries y DB2 a través de conexiones JDBC.

Finalización automática

En la página **Finalización automática** del cuadro de diálogo "Opciones" puede configurar las opciones de finalización automática.



La finalización automática se puede activar automática o manualmente.

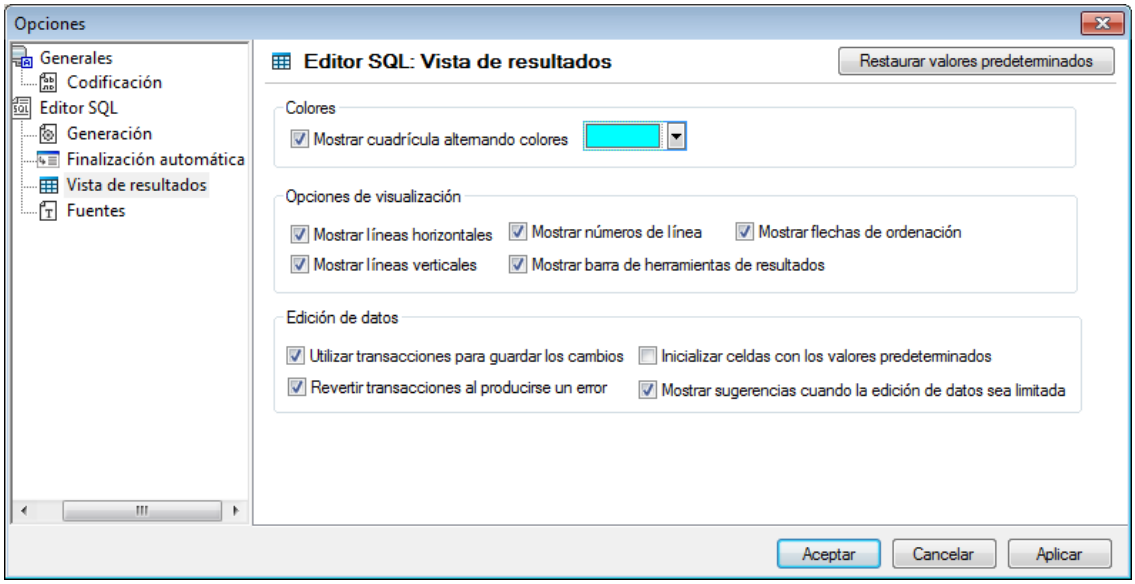
En el grupo de opciones *Teclas de finalización* puede especificar qué caracteres insertan la palabra clave y cierran la ventana de finalización automática.

En el grupo de opciones *Comportamiento de la inserción* puede definir si las palabras clave se

insertan en mayúsculas o minúsculas y si los identificadores se deben insertar entre caracteres de escape.

Vista de resultados

En la página ***Vista de resultados*** del cuadro de diálogo "Opciones" puede configurar el aspecto de la pestaña *Resultados* del **Editor SQL** que muestra los resultados de las consultas.



Colores
En este grupo de opciones puede elegir si la tabla de resultados se muestra en formato cuadrícula, alternando filas blancas con filas de cualquier otro color.

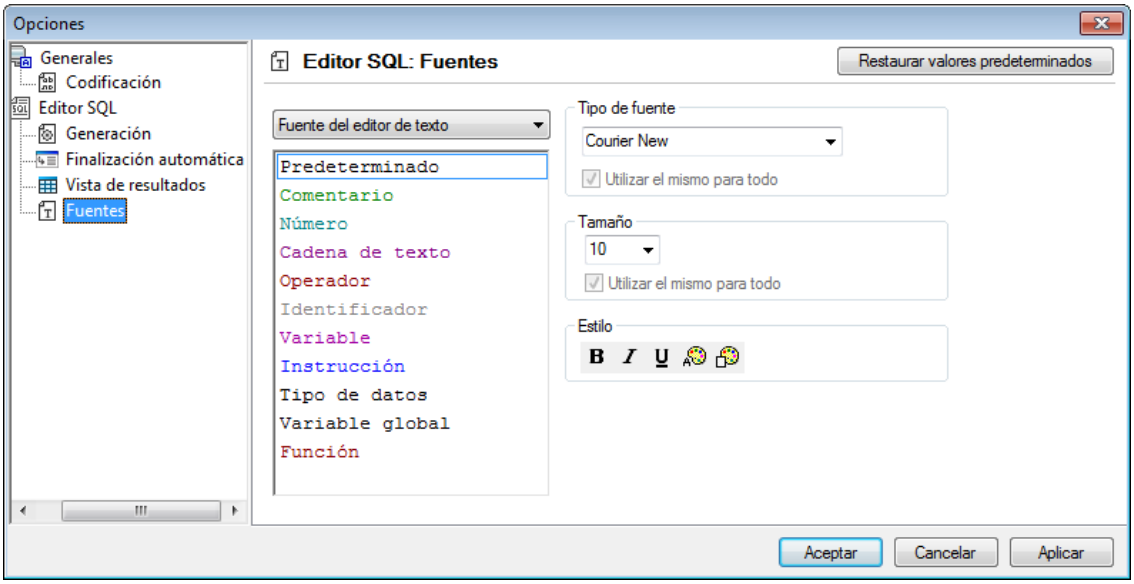
Opciones de visualización
En este grupo de opciones puede definir las líneas horizontales y verticales de la cuadrícula así como los números de línea. También puede elegir si se incluye la barra de herramientas o no.

Edición de datos
En este grupo de opciones puede configurar las opciones de transacción, si las celdas deben llenarse con valores predeterminados y si aparece una advertencia cuando la edición de datos es limitada.

Fuentes

En la página **Fuentes** del cuadro de diálogo "Opciones" puede configurar el color y la fuente de varias partes de las instrucciones SQL.

 © 2014 Altova GmbH



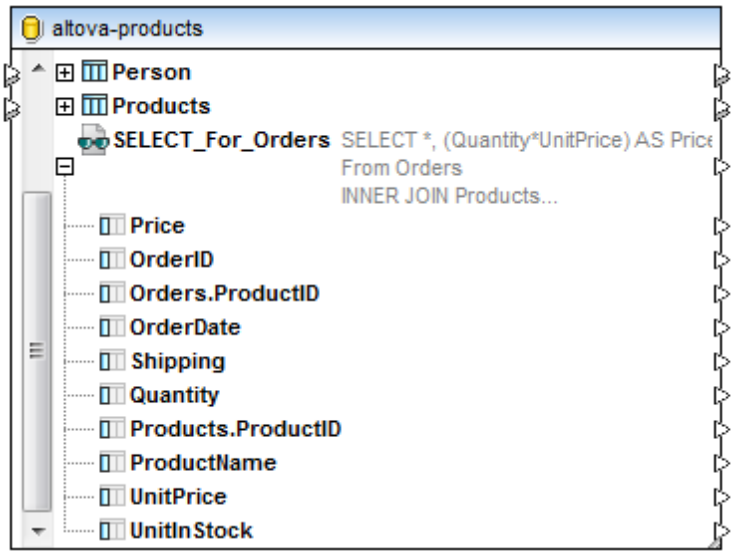
El cuadro de lista incluye los diferentes elementos de las instrucciones SQL. Puede definir una fuente, un estilo y un tamaño para el texto que aparece en el editor SQL. Tenga en cuenta que la fuente y el tamaño seleccionados se aplican a todos los elementos de las instrucciones SQL.

No obstante, cada elemento puede tener un estilo propio (color, peso de la fuente, etc.) si se habilita la función de color de sintaxis. Haga clic en el botón **Restaurar valores predeterminados** para devolver el aspecto del **Editor SQL** a su configuración original.

10.2.16 Instrucciones 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 se crean cuando se insertan las tablas de la BD en MapForce.
- Las instrucciones SELECT pueden tener parámetros que usan la misma sintaxis que el componente WHERE/ORDER de SQL.



Nota: la instrucción SELECT se puede ver dentro del componente. Para definir cuántas líneas de la instrucción deben estar visibles en el componente, seleccione **Herramientas | Opciones**, haga clic en **General** y escriba el número de líneas en la opción *Limitar la presentación de anotaciones a X líneas* del grupo de opciones *Visualización de asignaciones*.

Esta sección utiliza como ejemplo la base de datos `altova-products.mdb` disponible en la carpeta `...\MapForceExamples\Tutorial`. También utiliza el archivo de asignación `select-component.mfd`, que está guardado en la misma carpeta.

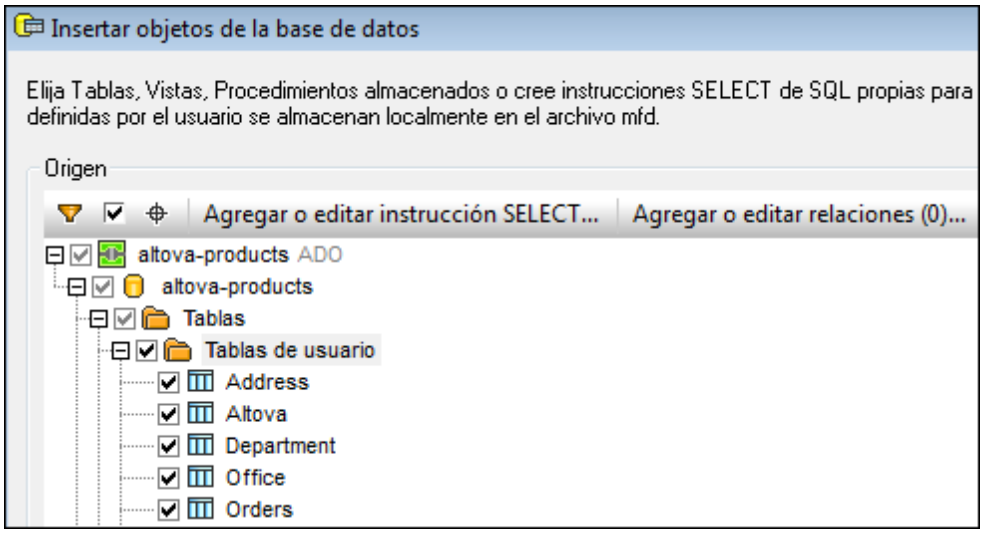
Crear instrucciones SELECT

Este apartado utiliza como ejemplo la base de datos `altova-products.mdb` disponible en la carpeta `...\MapForceExamples\Tutorial`.

Para crear una instrucción SELECT dentro de un componente de BD:

1. Haga clic en el icono **Insertar base de datos** de la barra de herramientas.

2. Seleccione el botón de opción *Microsoft Access* y haga clic en **Siguiente**.
3. Haga clic en el botón **Examinar** para seleccionar la BD que desea usar como origen de datos (`altova-products.mdb`) y después haga clic en **Conectarse**.
4. Marque la casilla *Tablas de usuario* para seleccionar todas las tablas de la BD.

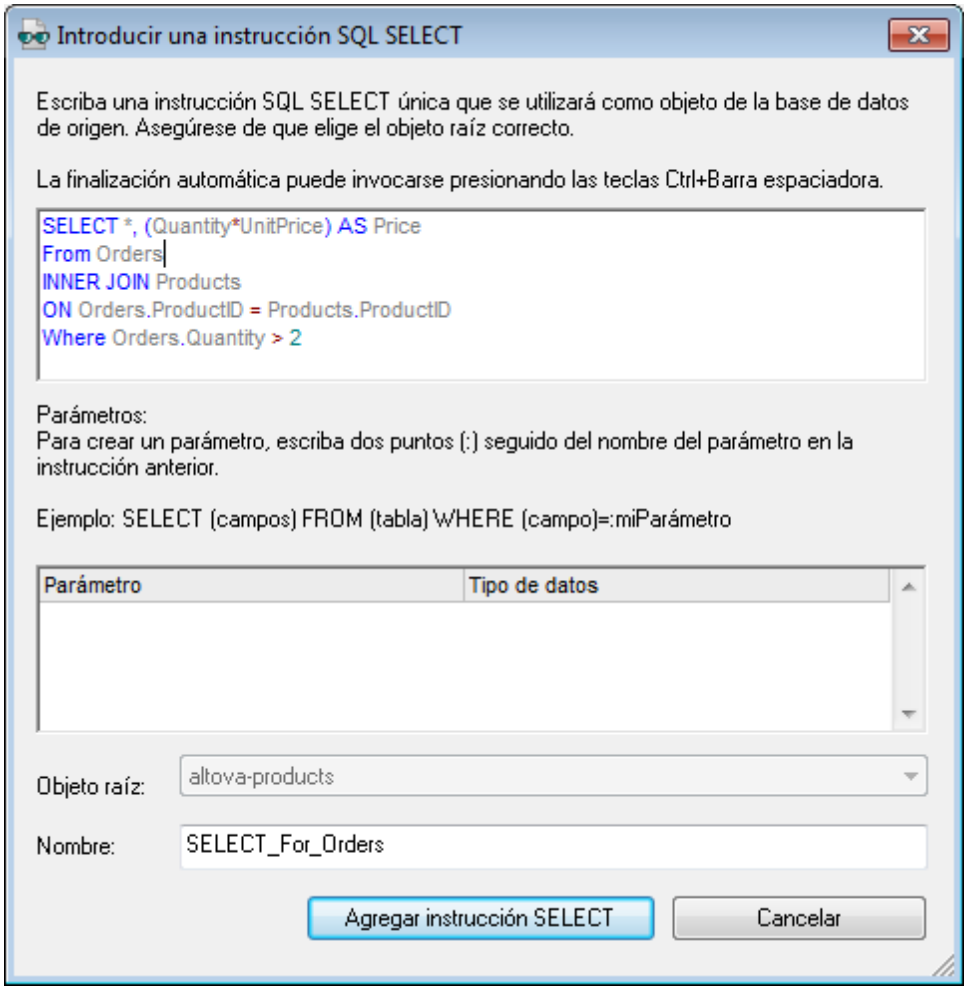


5. Haga clic con el botón secundario en la tabla **`Orders`** y seleccione el comando **Generar y agregar una instrucción SELECT** del menú contextual (o haga clic en el botón **Agregar o editar instrucción SELECT** de la barra de herramientas). Aparece un cuadro de diálogo con una instrucción SELECT predeterminada.
6. Edite la instrucción según sus requisitos. Para este ejemplo utilizamos esta instrucción:

```
SELECT *, (Quantity*UnitPrice) AS Price
From Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
Where Orders.Quantity > 2
```

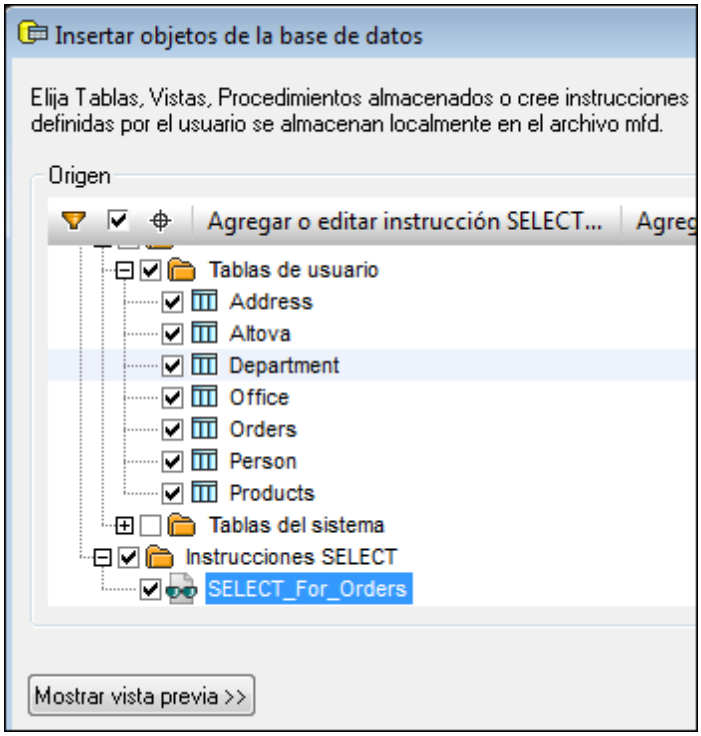
Observe que todas las expresiones calculadas de la instrucción SELECT deben tener un nombre de correlación único (como "AS Price" en el ejemplo anterior) para poder considerarse elementos asignables.

Nota: si se conecta a una base de datos Oracle o IBM DB2 a través de JDBC y usa una instrucción SELECT con el comando **Agregar o quitar tabla** para recuperar datos, la instrucción SELECT no puede llevar un punto y coma al final.



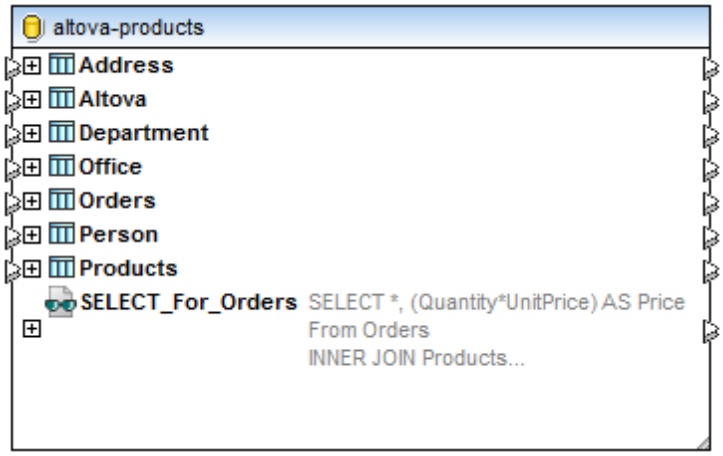
7. Haga clic en el botón **Agregar instrucción SELECT** para insertar el componente.

 © 2014 Altova GmbH



La "tabla virtual" se añade a la carpeta *Instrucciones SELECT* del cuadro de diálogo.

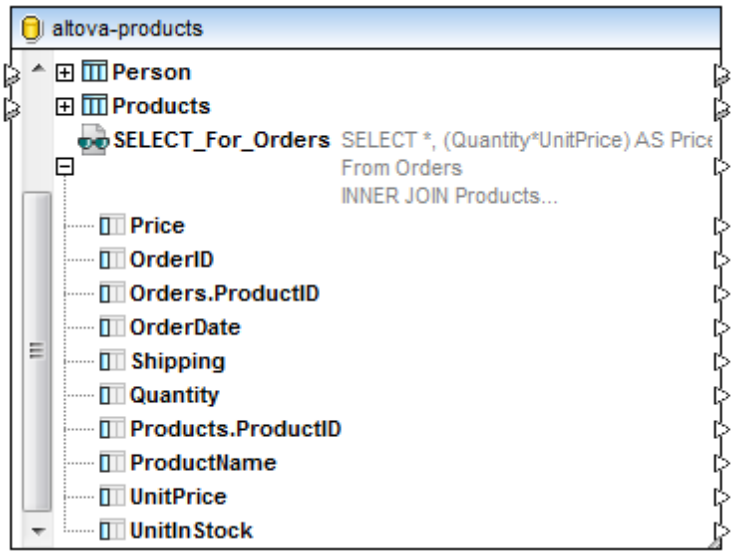
8. Para terminar haga clic en **Aceptar** y el componente se inserta en el área de asignación.



¿Qué se insertó?

- Todas las tablas de la base de datos **altova-products**, desde **`Address`** hasta **`Products`**.
- La tabla virtual **`SELECT_For_Orders`** (tabla del conjunto de resultados) que contiene todas las columnas seleccionadas o expresiones definidas por la instrucción SELECT.

1. Haga clic en el icono de expansión de la tabla virtual **`Select_For_Orders`**.



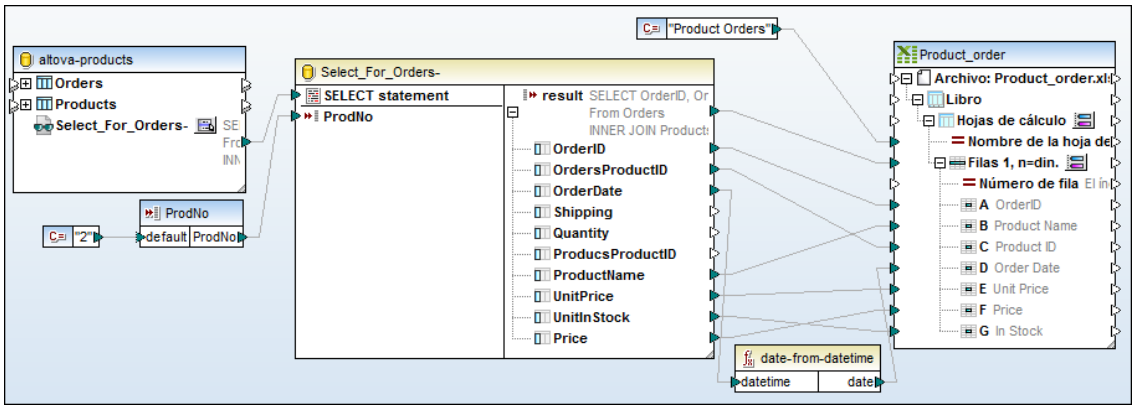
Las columnas de ambas tablas, definidas por la combinación interna, se pueden asignar a otros componentes de la asignación.

Nota: el campo **`Price`** es el producto de dos campos (**`Quantity`** y **`UnitPrice`**) y en realidad es un nombre de correlación: SELECT *, (Quantity*UnitPrice) AS Price (*ver la primera línea de la instrucción SELECT*).

Parámetros de la instrucción SELECT

El apartado anterior explica cómo se inserta una instrucción SELECT en un componente de BD y muestra qué campos contiene el componente cuando la instrucción SELECT se inserta en el área de asignación.

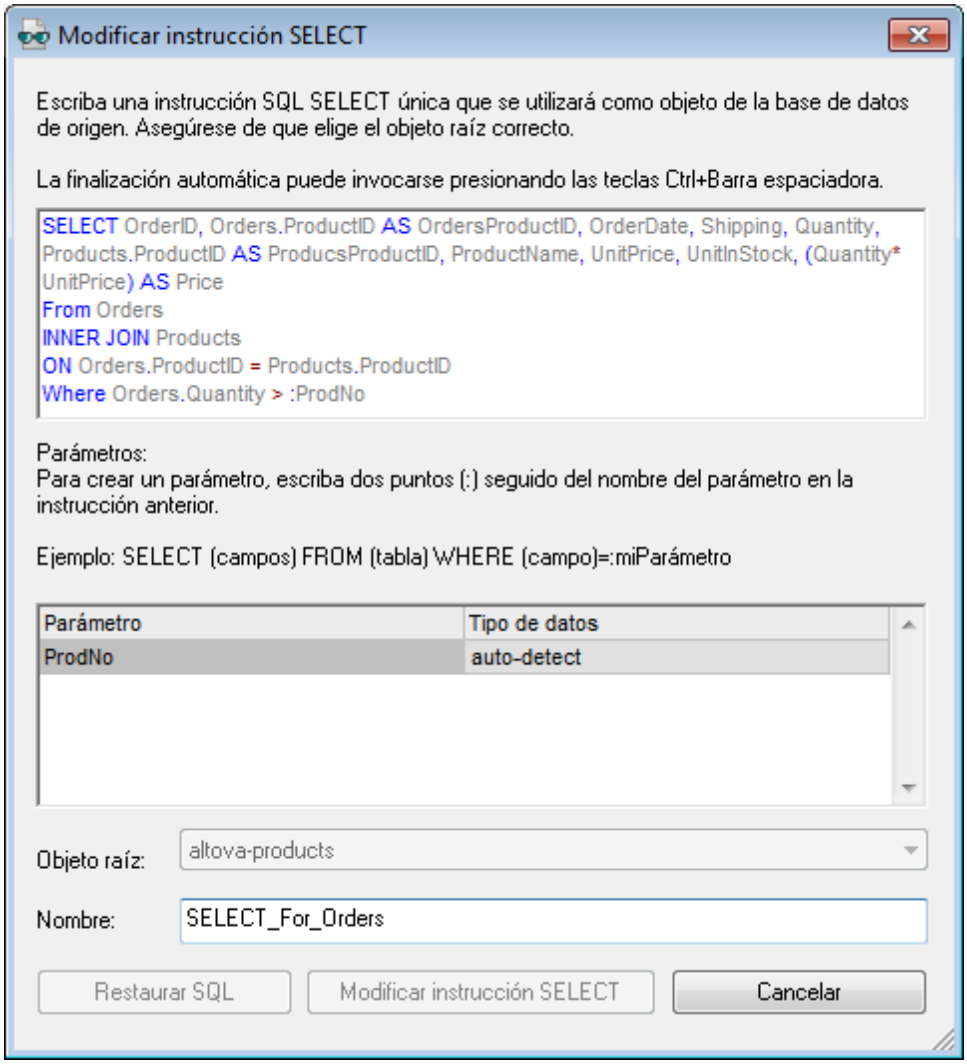
Este apartado usa otra vez la base de datos `altova-products.mdb` y el archivo de asignación `select-component.mfd` (*imagen siguiente*), disponible en la carpeta `...\MapForceExamples\Tutorial`. El objetivo de esta asignación es recuperar los datos de la tabla **`Order`** de la BD y asignarlos a una hoja de cálculo OOXML Excel 2007. En la asignación se usa un parámetro instrucción SELECT para pasar el parámetro **`ProdNo`** de forma externa (p. ej. a través de una llamada a la línea de comandos o por medio de un trabajo de FlowForce Server).



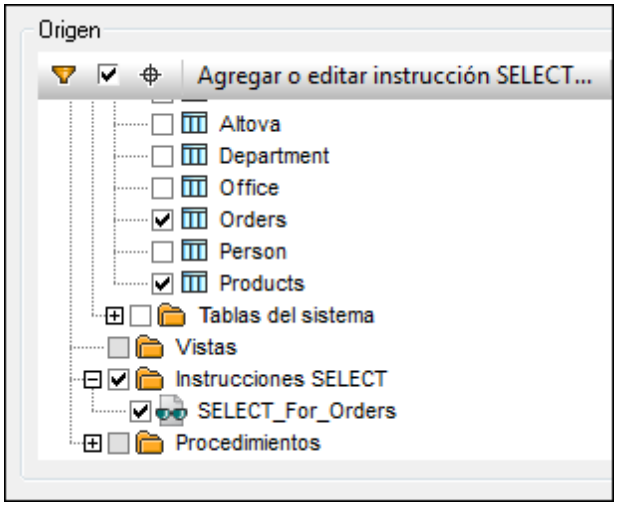
Para reproducir la asignación anterior desde cero:

1. Inserte la base de datos `altova-products.mdb` de la carpeta `.../MapForceExamples/tutorial`.
2. Marque las casillas de las tablas **`Products`** y **`Order`**.
3. Haga clic en el botón **Agregar o editar instrucción SELECT** de la barra de iconos y escriba esta instrucción:

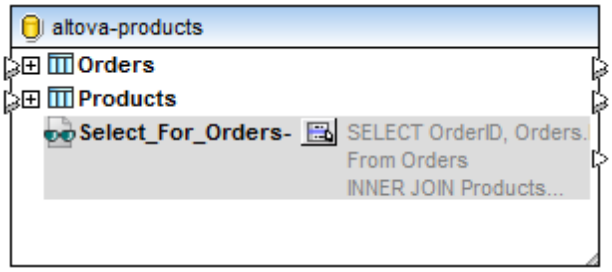
```
SELECT OrderID, Orders.ProductID AS OrdersProductID, OrderDate, Shipping, Quantity,
Products.ProductID AS ProducsProductID, ProductName, UnitPrice, UnitInStock,
(Quantity*UnitPrice) AS Price
From Orders
INNER JOIN Products
ON Orders.ProductID = Products.ProductID
Where Orders.Quantity > :ProdNo
```



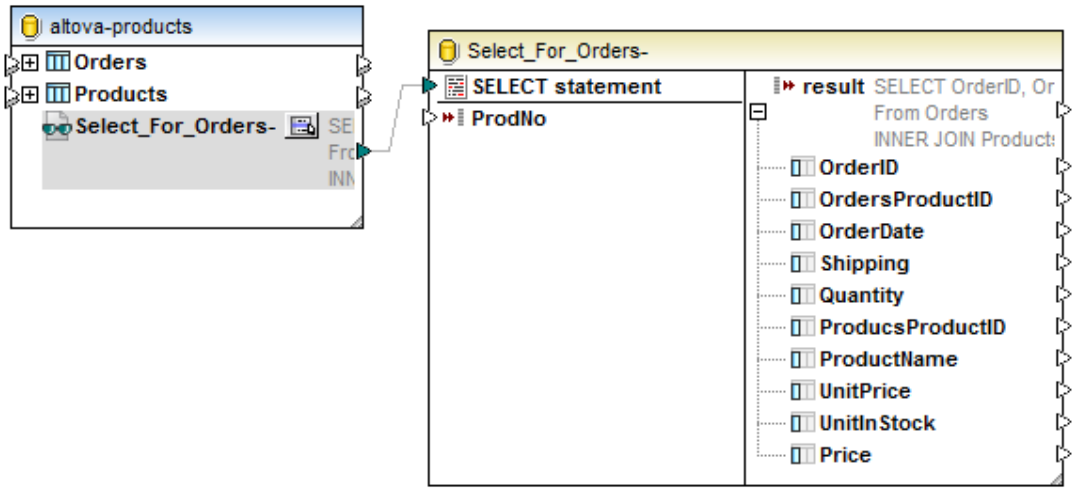
   Observe que en cuanto se introduce el nombre de parámetro `:ProdNo` se rellena automáticamente la columna *Parámetro* del recuadro inferior.
4. Introduzca el nombre de la instrucción SELECT en el campo *Nombre*.
5. Haga clic en el botón **Agregar instrucción SELECT**.
   La instrucción se inserta en la carpeta *Instrucciones SELECT*.



6. Haga clic en **Aceptar** para insertar el componente de BD en la asignación.

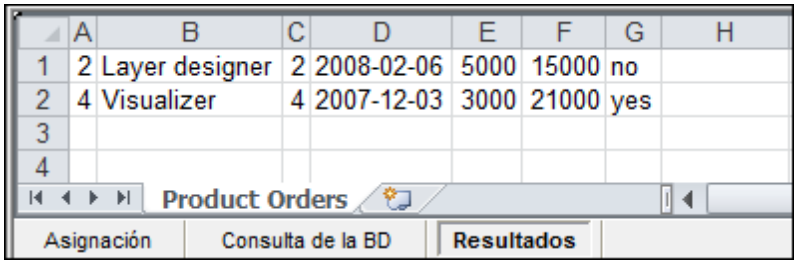


7. Haga clic en el botón y en el menú contextual que aparece seleccione **Insertar llamada con parámetros**.
   Esto inserta el componente de origen de datos **Select_For_Orders**.



   En este componente puede definir los parámetros de entrada del componente a la izquierda (es decir, **`ProdNo`**) y a la derecha el resultado de la tabla virtual que desea asignar al componente de destino.
8. Inserte el resto de los componentes (una hoja de Excel vacía, un componente de entrada y la función **`date-from-datetime`**) que aparecen en la primera imagen de este apartado.
9. Conecte los componentes tal y como aparecen en la imagen y abra el panel **Resultados** para ver el resultado de la asignación.

 © 2014 Altova GmbH



	A	B	C	D	E	F	G	H
1	2	Layer designer	2	2008-02-06	5000	15000	no	
2	4	Visualizer	4	2007-12-03	3000	21000	yes	
3								
4								

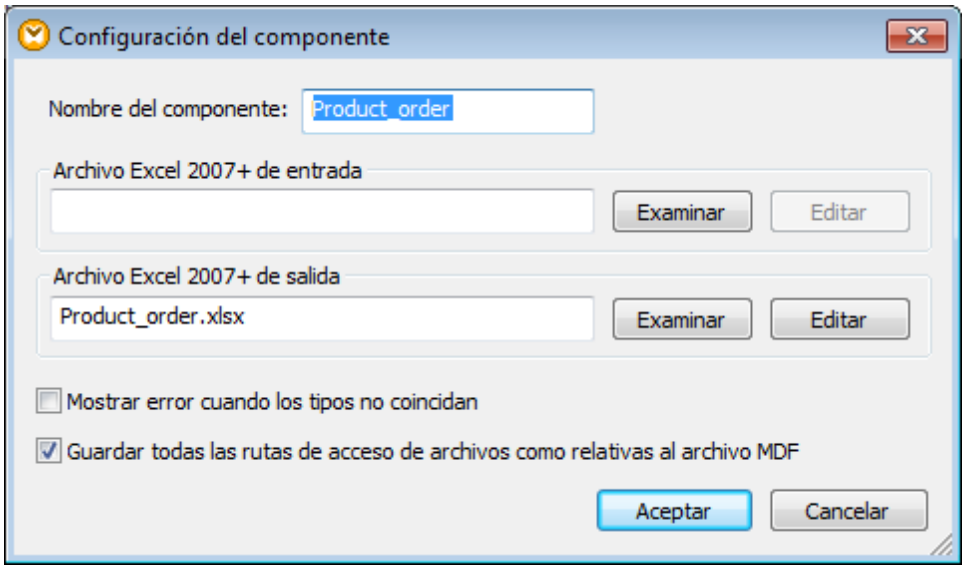
Product Orders

Asignación | Consulta de la BD | Resultados

Notas:

- El componente de entrada **`ProdNo`** define el parámetro de entrada que se debe utilizar cuando la asignación se ejecuta desde la línea de comandos. Cambie el número del componente de constante para ver una vista previa diferente de la asignación.
- Para más información sobre asignaciones de datos de hojas de cálculo Excel 2007, consulte el apartado Asignaciones de datos MS OOXML Excel 2007 (o superior).

  Recuerde que si hace doble clic en la barra de título del componente Excel, aparece un cuadro de diálogo donde puede definir el nombre del archivo de entrada y de salida del componente Excel.



- La función **`date-from-datetime`** convierte el formato de fecha de la BD al formato de fecha simple.
- En el panel **Resultados** puede ver una vista previa del contenido del archivo Excel de salida.

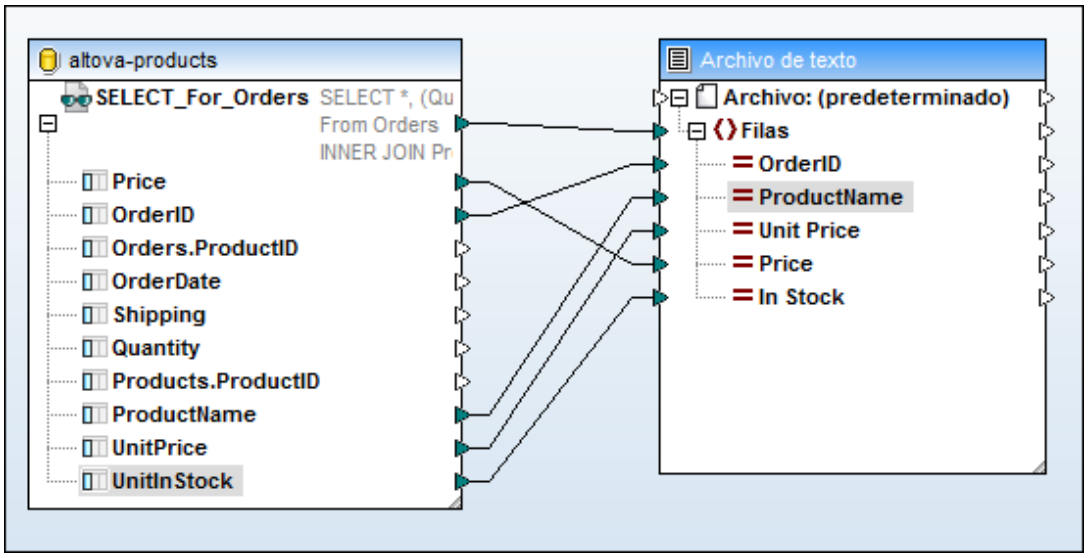
Ejemplo de instrucción SELECT

En el apartado interior explicamos cómo insertar una instrucción SELECT en un componente de base de datos y qué campos contiene el componente cuando se inserta en el área de asignación. Este ejemplo usa la misma base de datos que el apartado anterior (es decir, `altova-products.mdb` de la carpeta `...\MapForceExamples\Tutorial`).

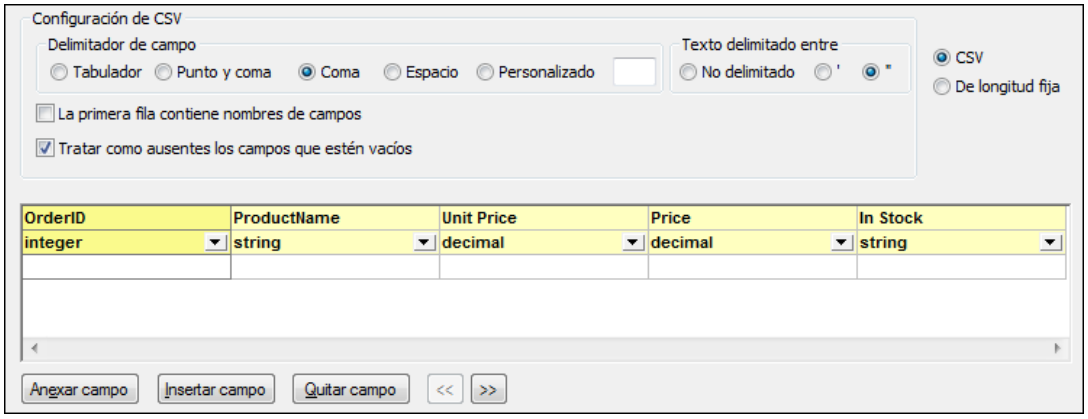
En la asignación de este ejemplo:

- se crea una instrucción SELECT "altova-products" usando una instrucción INNER JOIN.
- se asignan datos de la BD a un archivo de texto/CSV vacío

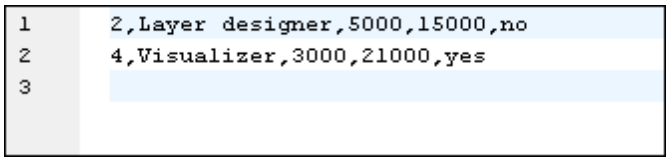
1. Inserte la instrucción SELECT (siguiendo los pasos explicados previamente).



2. Inserte un componente de texto, ponga nombres a sus elementos y seleccione los tipos de los diferentes campos (es decir, integer, string, etc.)



3. Cree las conexiones que aparecen en la primera imagen entre los dos componentes y abra el panel **Resultados** para ver el resultado de la instrucción.



10.2.17 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: para explicar la implementación de procedimientos almacenados en MapForce en esta sección usamos MS SQL 2008 y la base de datos **AdventureWorks**, que se puede descargar del sitio web de CodePlex.

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.

Controladores
Por lo general no es buena idea usar controladores puentes como el puente de ODBC-ADO de Microsoft o ODBC-JDBC de Sun. Este último se considera experimental y no se debería usar en

un entorno de producción. Por eso recomendamos utilizar siempre que se pueda la versión más reciente del controlador nativo de BD que mantenga el proveedor de la BD.

Microsoft SQL Server 2000, 2005, 2008
Compatibles con MapForce: procedimientos almacenados, funciones escalares, funciones con valores de tabla.

El controlador ADO "Microsoft OLE DB Provider for SQL Server" viene con el sistema operativo para ofrecer funciones SQL Server básicas pero está escrito en SQL Server 7 (o anterior). Este controlador no es compatible con MapForce. Use el controlador más reciente que viene con el servidor de la BD.

La API ADO tiene una compatibilidad limitada con algunos tipos de datos introducidos por SQL Server 2008 (`datetime2`, `datetimeoffset`). Si detecta problemas de truncamiento de datos con los nuevos tipos `time` cuando use ADO con SQL Server Native Client, puede establecer el argumento de cadena de conexión DataTypeCompatibility=80 o utilice ODBC.

El antiguo controlador JDBC del paquete "com.microsoft.jdbc.sqlserver" (SQL Server 2000) no es compatible. Los procedimientos de SQL Server tienen un parámetro de devolución implícito del tipo int null, que se puede asignar. Si el procedimiento omite una instrucción RETURN, el valor resultantes es 0.

IBM DB2 8, 9 y IBM DB2 for i 6.1 y i 7.1
Compatibles con MapForce: procedimientos almacenados, funciones escalares, funciones con valores de tabla.
Los valores devueltos de procedimientos almacenados de DB2 no son compatibles con MapForce porque no se pueden leer desde la API de BD utilizada en MapForce.

Las funciones con valores de fila (RETURNS ROW) no son compatibles.
Recomendamos instalar como mínimo "IBM_DB2 9.7 Fix Pack 3a" para evitar un problema confirmado del controlador JDBC que se produce cuando se leen errores/advertencias tras la ejecución. Esto también corrige un problema detectado en el proveedor ADO que causa una fila de conjuntos de resultados que falta.

Oracle 9i, 10g, 11g
Compatibles con MapForce: procedimientos almacenados, funciones escalares, funciones con valores de tabla.

Microsoft OLE DB Provider for Oracle es muy antiguo y no es compatible con MapForce. Utilice un controlador de Oracle más actual.
Oracle tiene una manera especial de devolver conjuntos de resultados al cliente utilizando parámetros de salida de tipo REF CURSOR. Esto es compatible con MapForce en el caso de los procedimientos almacenados pero no en el caso de las funciones. Por tanto, los nombres y números de los conjuntos de registros siempre son fijos para los procedimientos almacenados de Oracle.

Sybase 12
Compatibles con MapForce: solamente procedimientos almacenados, las funciones no son compatibles.

 © 2014 Altova GmbH

MySQL 4, 5
MySQL no es totalmente compatible con los procedimientos almacenados (no puede recuperar parámetros).

Compatibles con MapForce: pueden funcionar los procedimientos almacenados sin parámetros que solamente devuelven conjuntos de registros. Las funciones no son compatibles.

PostgreSQL 8
PostgreSQL no tiene procedimientos almacenados, todo se hace con funciones.

Compatibles con MapForce: funciones escalares, funciones con valores de tabla y con valores de fila.

En PostgreSQL los parámetros definidas en una función describen las columnas del conjunto de resultados. MapForce utiliza esta información automáticamente: no es necesaria la detección por ejecución ni la entrada manual de conjuntos de registros. Los parámetros de tipo refcursor no son compatibles.

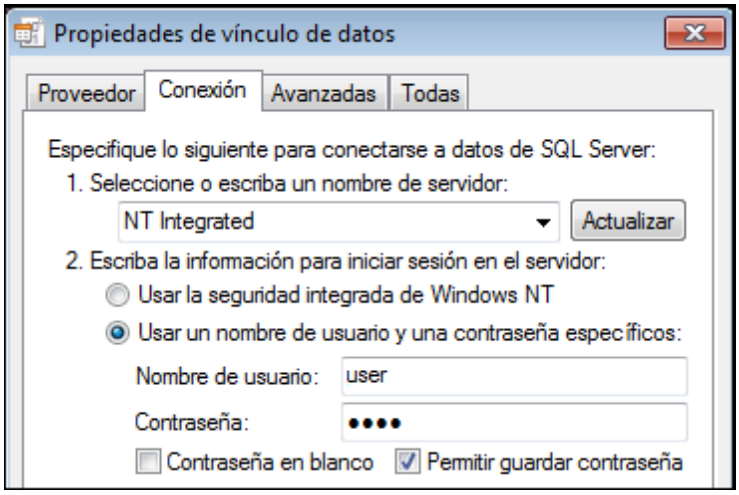
Microsoft Access 2003, 2007
Access tiene una implementación muy limitada para procedimientos almacenados y no puede consultar procedimientos almacenados ya existentes. Por tanto, esta característica no es compatible con Access.

Insertar procedimientos almacenados en componentes de BD

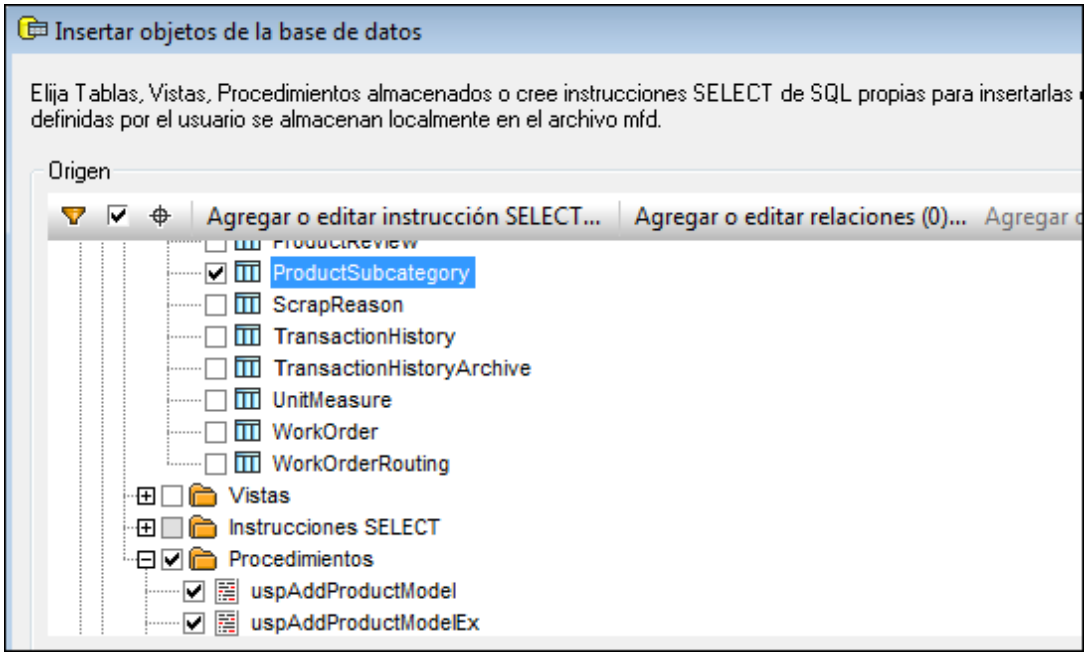
Los procedimientos almacenados se pueden incorporar a un componente de BD cuando el componente se inserta en el área de asignación.

Para insertar un componente de BD que contiene procedimientos almacenados:

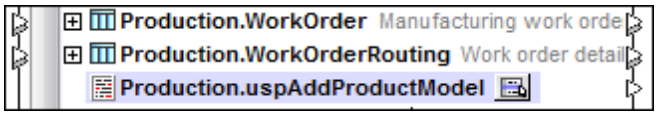
1. Haga clic en el icono **Insertar base de datos** o seleccione la opción de menú **Archivo | Insertar base de datos**.
2. Use el Asistente para la conexión para conectarse a la base de datos.
3. Aparece el cuadro de diálogo "Propiedades de vínculo de datos". Rellene los campos de la pestaña *Conexión* 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.



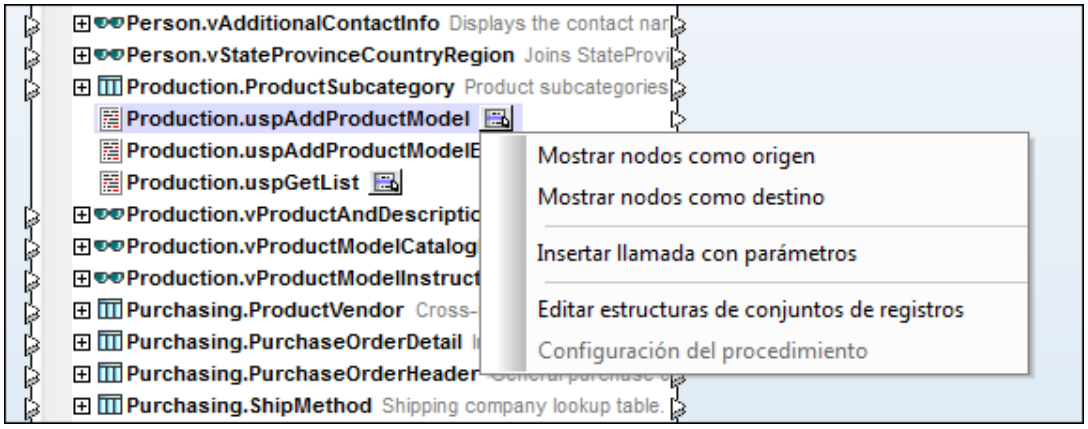
5. Expanda la carpeta *Procedimientos* para seleccionar los procedimientos almacenados que desea insertar y después haga clic en **Aceptar**.



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.

 © 2014 Altova GmbH



Casos de uso

A continuación enumeramos los casos de uso más frecuentes relacionados con procedimientos almacenados.

Quiero...	Consulte este apartado
... llamar a un procedimiento almacenado para **recuperar** datos de una base de datos y asignarlos a otro componente. P. ej. quiero 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

Procedimientos almacenados y relaciones locales

Puede usar 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
- 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).

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	Componentes de origen y relaciones locales

 © 2014 Altova GmbH

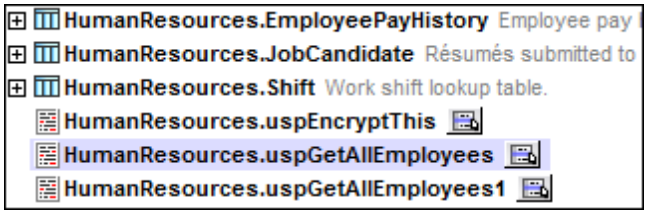
desde una tabla o vista de la misma BD o desde el resultado de otro procedimiento	

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**.



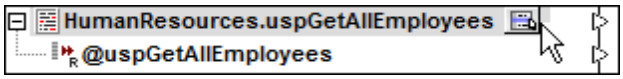
Procedimiento almacenado:

```
PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment;
```

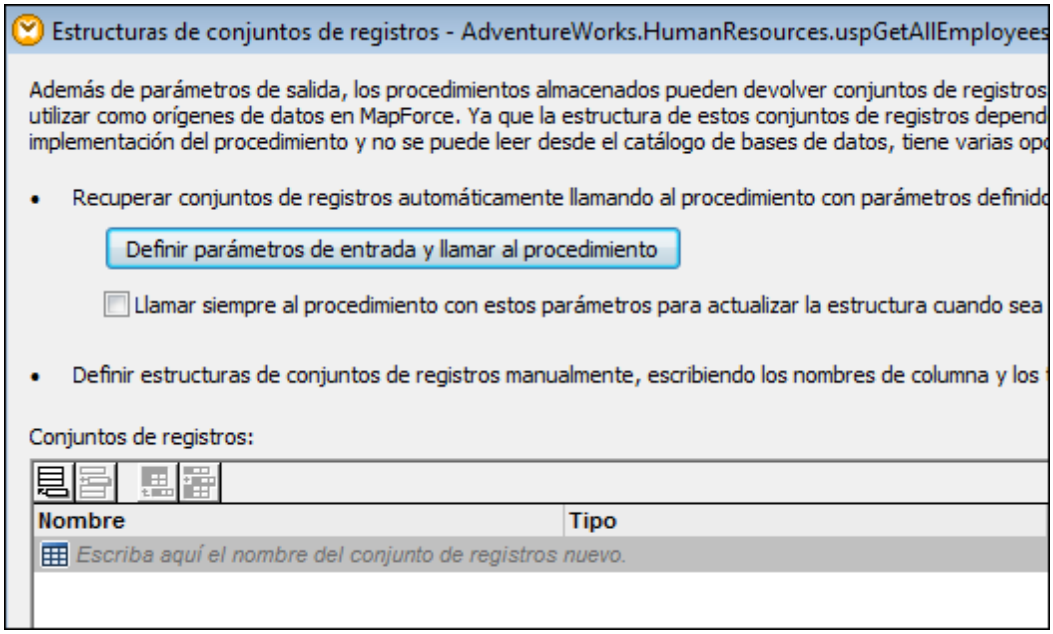
MapForce no puede leer las columnas del conjunto de registros desde el catálogo de la BD directamente. Antes deben ser detectadas ejecutando el procedimiento almacenado una vez o definiéndolas a mano.

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:

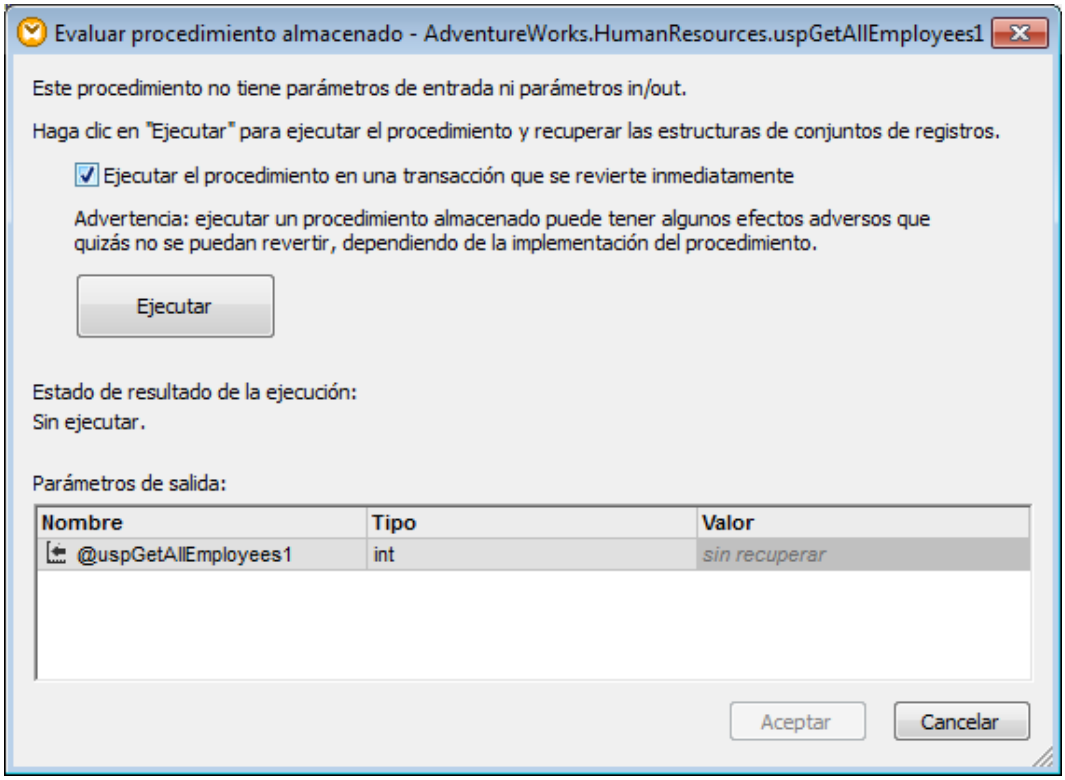
1. Haga clic en el botón del procedimiento almacenado y seleccione la opción **Mostrar nodos como origen** del menú contextual.

   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**.

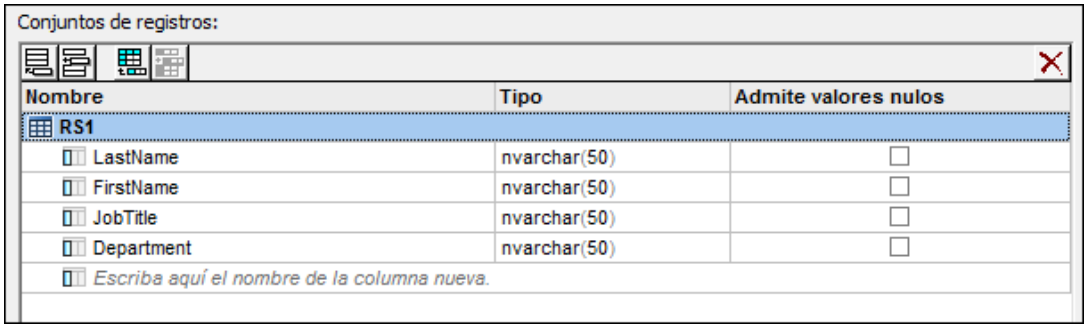


Aparece el cuadro de diálogo "Evaluar procedimiento almacenado".

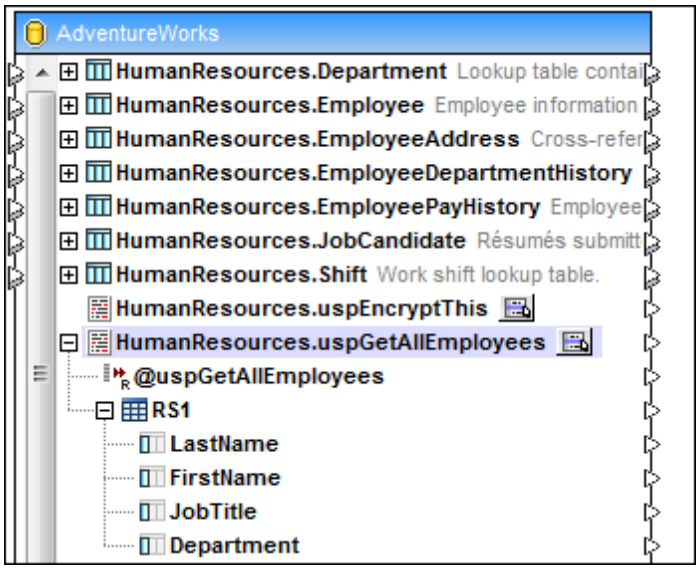


3. Haga clic en el botón **Ejecutar** y después en **Aceptar**.
   Los campos del conjunto de registros aparecen en el panel *Conjuntos de registros* del cuadro de diálogo.

 © 2014 Altova GmbH

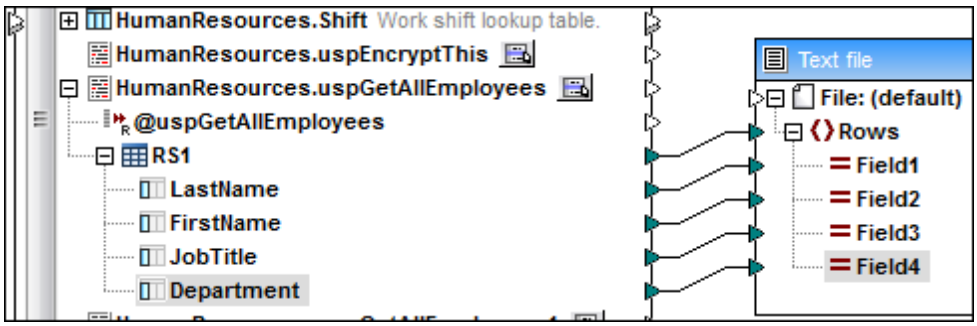


4. Haga clic otra vez en el botón **Aceptar** para terminar de definir el conjunto de registros. Las columnas (**`LastName`**, etc.) aparecen en forma de nodos bajo el nodo de conjunto de registros **`RS1`** (*Record Set 1*). (Si no las ve, haga clic en el icono + para expandir el conjunto de registros).



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.

```
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**.

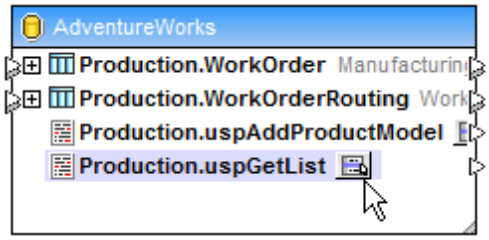
Llamada con parámetros: entrada/salida

Los procedimientos almacenados también se pueden usar como llamadas de tipo función. De esta manera puede:

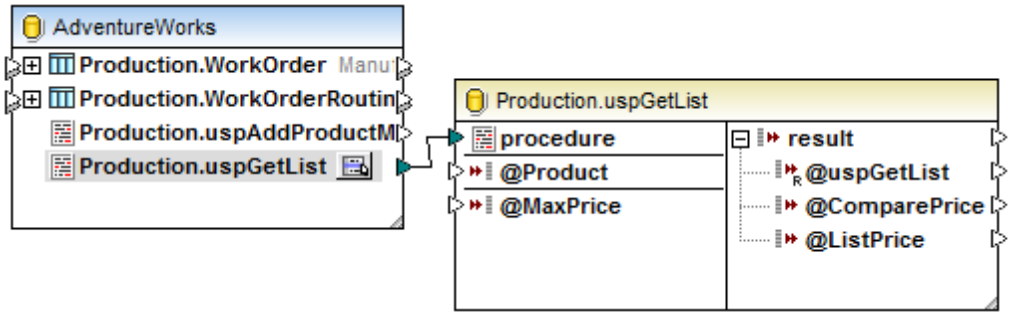
- suministrar datos de entrada al procedimiento
- ejecutar el procedimiento
- asignar los datos de salida del procedimiento a otros componentes

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

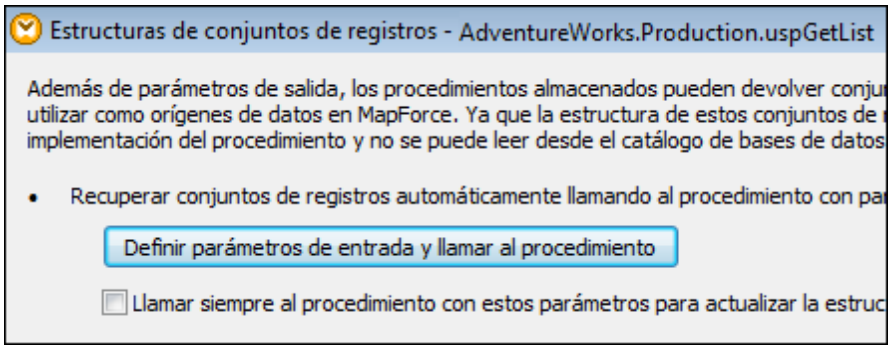
 © 2014 Altova GmbH

automáticamente al elemento **`procedure`** del componente.

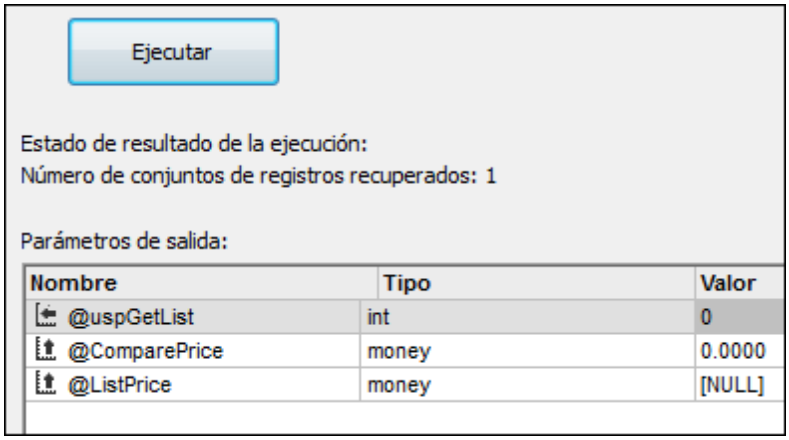
Los parámetros de entrada del procedimiento aparecen a la izquierda y los de salida a la derecha. En concreto, este procedimiento almacenado devuelve parámetros de salida y un conjunto de registros. Sin embargo, debemos definir su estructura antes de poder verla y utilizarla en MapForce.

Para definir la estructura del conjunto de registros:

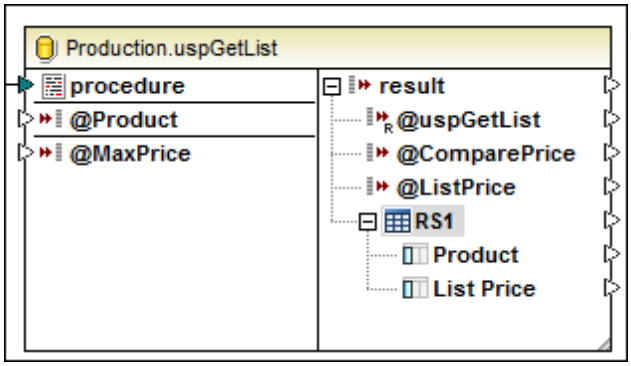
1. Haga clic en el botón interactivo del procedimiento almacenado **`Production.uspGetList`**.
2. Haga clic en el botón **Definir parámetros de entrada y llamar al procedimiento**. En el nuevo cuadro de diálogo haga clic en **Ejecutar**.



Los valores de los parámetros de salida devueltos aparecen en la tabla inferior. Se recuperó un conjunto de registros.



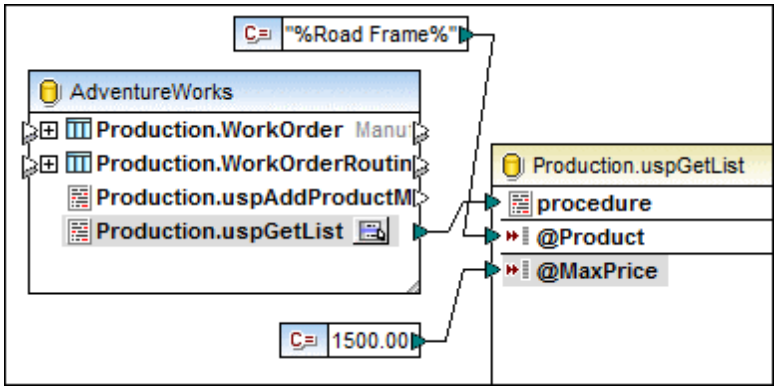
3. Haga clic **Aceptar** para confirmar y otra vez en **Aceptar** para cerrar el cuadro de diálogo "Estructuras de conjuntos de registros".



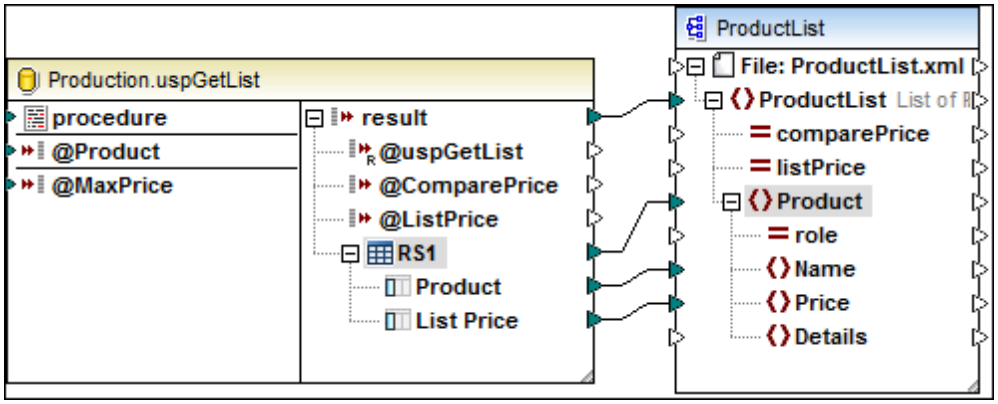
El conjunto de registros **RS1** se añadió a la sección derecha del componente de procedimiento.

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.

 © 2014 Altova GmbH

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <ProductList xsi:noNamespaceSchemaLocation="C:/Tools/sp/ProductList.xsd"
3     <Product>
4       <Name>HL Road Frame - Black, 58</Name>
5       <Price>1431.5</Price>
6     </Product>
7     <Product>
8       <Name>HL Road Frame - Red, 58</Name>
9       <Price>1431.5</Price>
10    </Product>
11    <Product>
12      <Name>HL Road Frame - Red, 62</Name>
13      <Price>1431.5</Price>
14    </Product>
15    <Product>
16      <Name>HL Road Frame - Red, 44</Name>
17      <Price>1431.5</Price>
```

Componentes de origen y relaciones locales

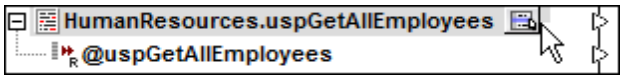
Utilice este método si desea combinar los datos dados por un conjunto de registros de un procedimiento almacenado con los datos de otra tabla con la que no hay ninguna relación directa.

```
PROCEDURE HumanResources.uspGetAllEmployees
AS
    SELECT LastName, FirstName, JobTitle, Department
    FROM HumanResources.vEmployeeDepartment;
```

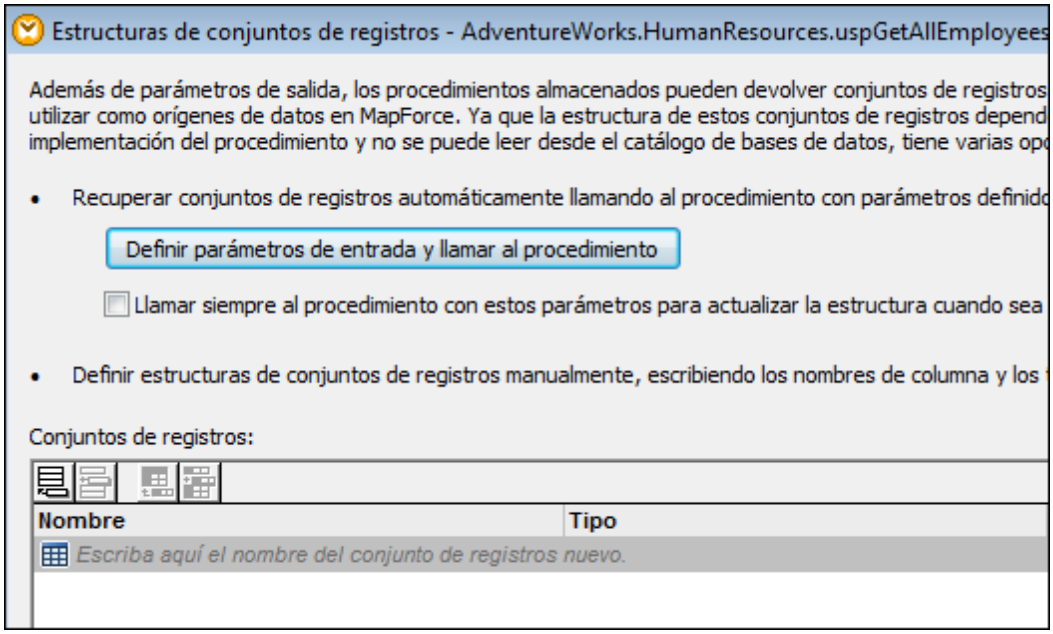
MapForce no puede leer las columnas del conjunto de registros desde el catálogo de la BD directamente. Antes deben ser detectadas ejecutando el procedimiento almacenado una vez o definiéndolas a mano.

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:

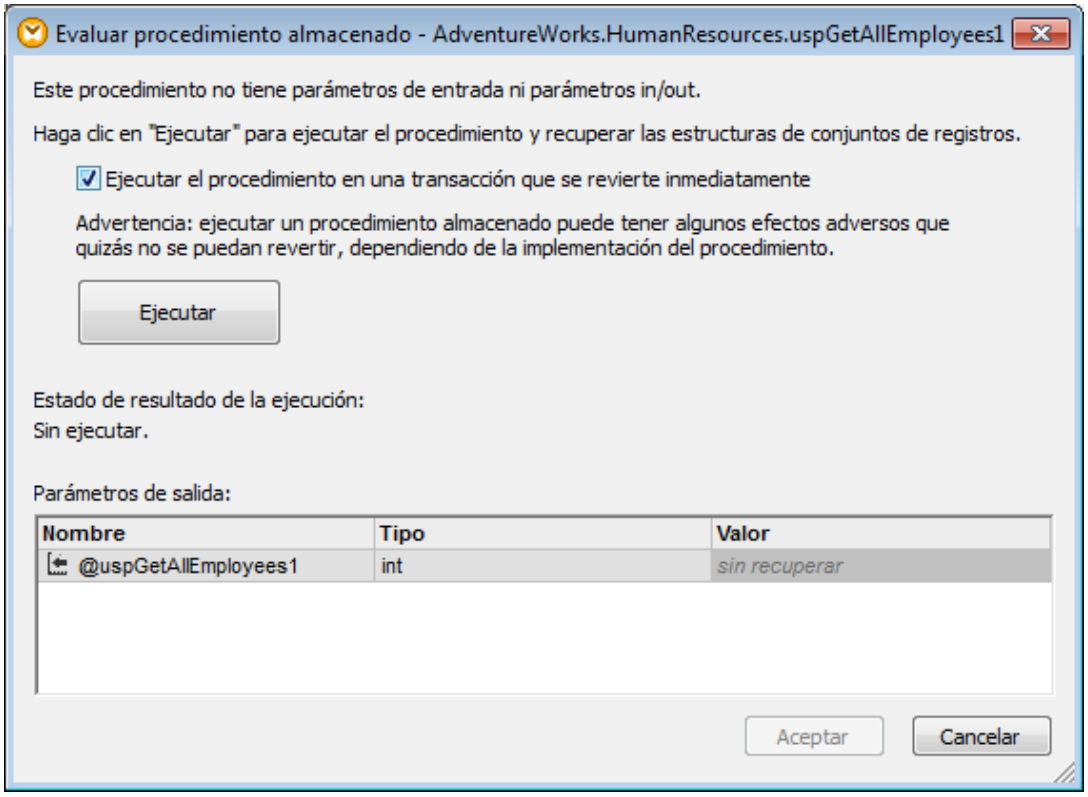
1. Haga clic en el botón interactivo del procedimiento almacenado y seleccione la opción **Mostrar nodos como origen**.



   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**.

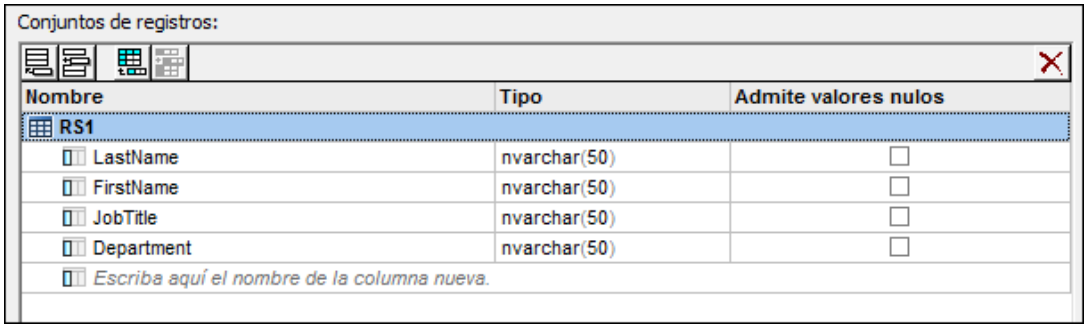


Se abre el cuadro de diálogo "Evaluar procedimiento almacenado".

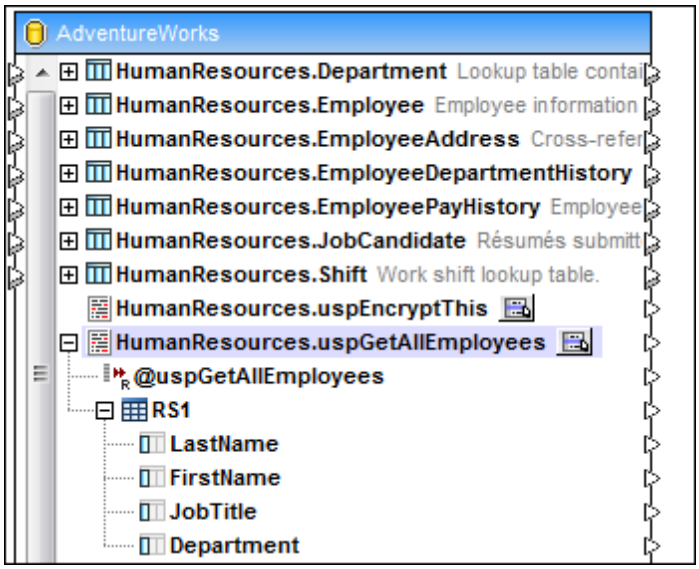


3. Haga clic en el botón **Ejecutar** y después en **Aceptar**.
   Los campos del conjunto de registros aparecen ahora en el panel *Conjuntos de registros* del cuadro de diálogo.

 © 2014 Altova GmbH

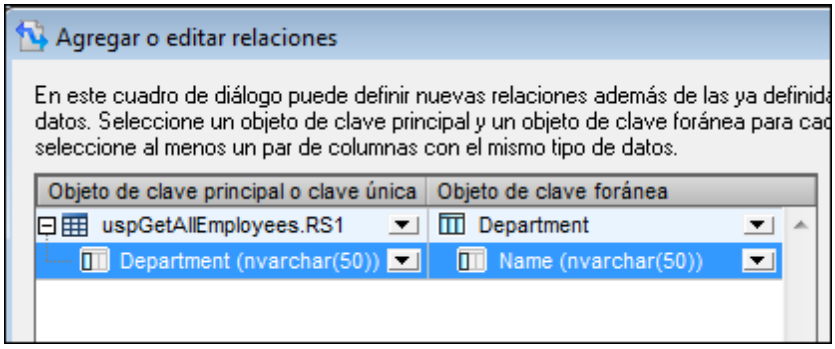


4. Haga clic otra vez en el botón **Aceptar** para terminar de definir el conjunto de registros. Las columnas del conjunto de registros (**`LastName`**, etc.) aparecen como nodos bajo el conjunto de registros **`RS1`** (*Record Set 1*). (Si no las ve, haga clic en el icono + para expandir el conjunto de registros).

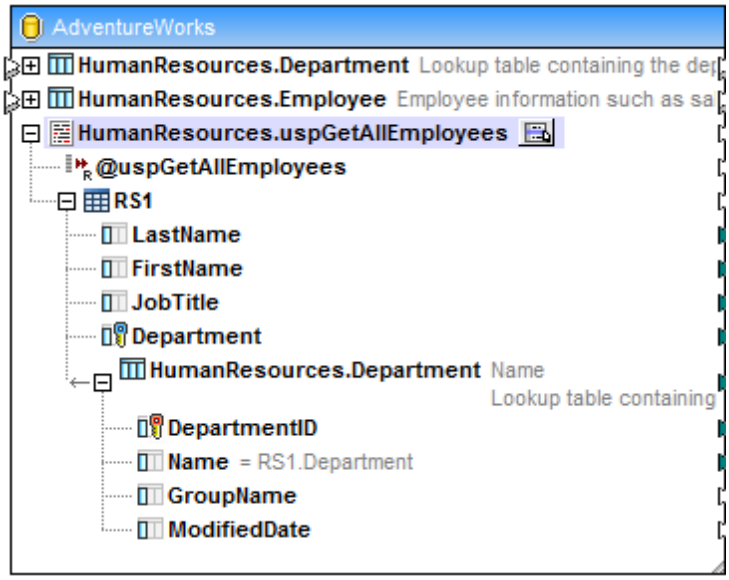


Definir una relación local con otra tabla:

1. Haga clic con el botón secundario 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.

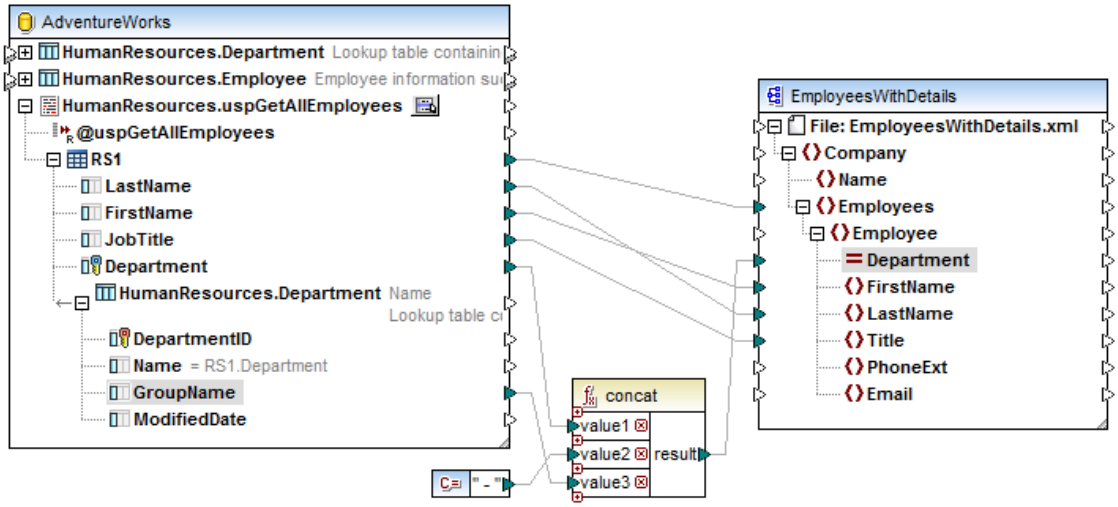


La tabla **`Department`** aparece ahora como elemento secundario del procedimiento almacenado.

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.

 © 2014 Altova GmbH



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

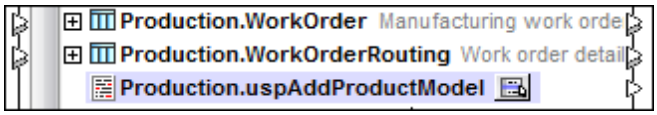
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xsi:noNamespaceSchemaLocation="C:/Users/Documents/Altova/Ma
3    <Employees>
4      <Employee Department="Production - Manufacturing">
5        <FirstName>Guy</FirstName>
6        <LastName>Gilbert</LastName>
7        <Title>Production Technician - WC60</Title>
8      </Employee>
9    </Employees>
10   <Employees>
11     <Employee Department="Marketing - Sales and Marketing">
12       <FirstName>Kevin</FirstName>
13       <LastName>Brown</LastName>
14       <Title>Marketing Assistant</Title>
15     </Employee>
16   </Employees>
17   <Employees>
18     <Employee Department="Engineering - Research and Development">
19       <FirstName>Roberto</FirstName>
20       <LastName>Tamburello</LastName>
21       <Title>Engineering Manager</Title>
22     </Employee>
```

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:

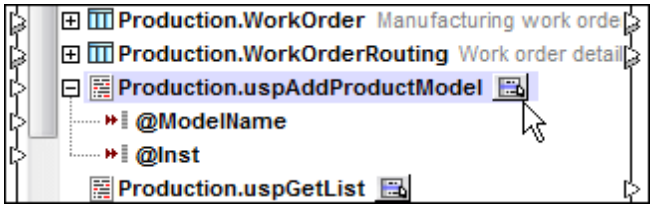
```
PROCEDURE Production.uspAddProductModel
@ModelName nvarchar(50),
@Inst xml

as
INSERT INTO [AdventureWorks].[Production].[ProductModel]
           ([Name]
           --,[CatalogDescription]
           ,[Instructions]
           ,[rowguid]
           ,[ModifiedDate])
     VALUES
           (@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:

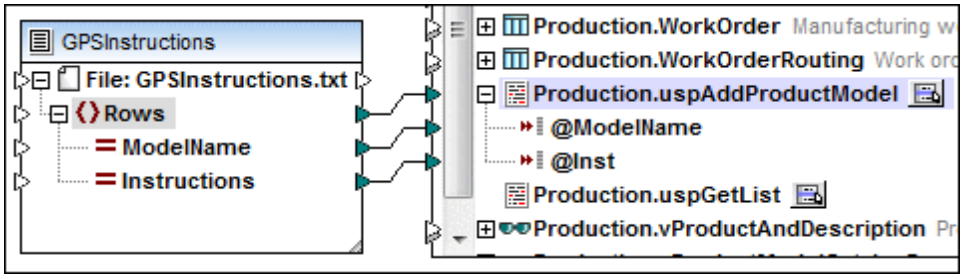
Tras insertar el componente de base de datos **AdventureWorks**, seleccionar las tablas **`Production`** e incluir los procedimientos almacenados:

1. Haga clic en el botón interactivo de procedimiento almacenado y seleccione la opción **Mostrar nodos como origen**.



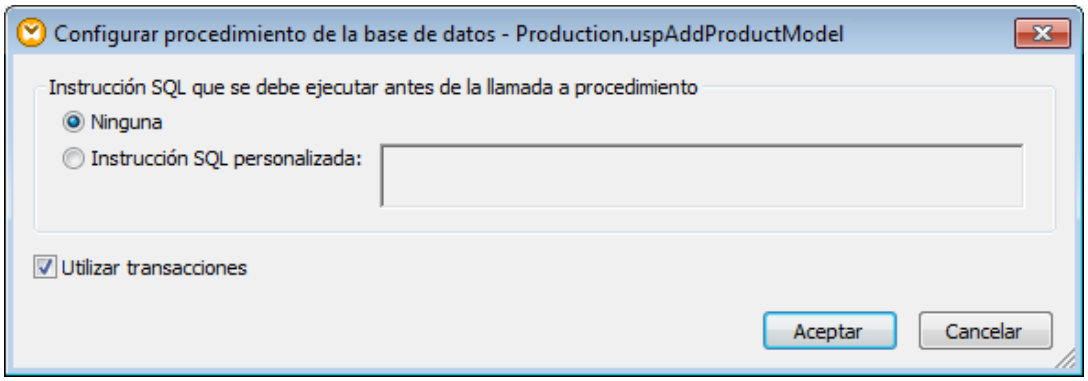
   Esto inserta los parámetros de entrada **`@ModelName`** e **`@Inst`** bajo el nombre del parámetro de entrada. En el componente de destino solamente están disponibles los iconos de entrada de los parámetros de entrada.
2. Inserte un componente de origen (p. ej. un archivo de texto, un archivo XML, etc.) y asigne a los iconos de entrada del procedimiento almacenado los elementos que deben suministrar los datos a los parámetros de entrada.

 © 2014 Altova GmbH



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.

Usar procedimientos almacenados para generar claves principales

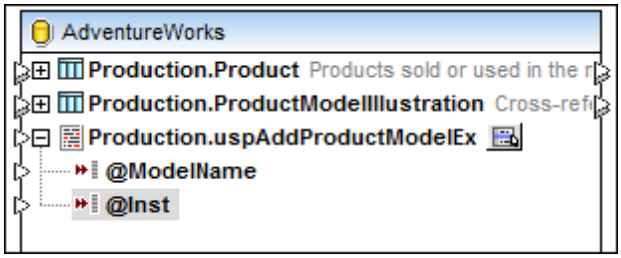
Utilice este método si el procedimiento almacenado realiza cambios en la BD y también quiere usar el parámetro de salida del procedimiento para generar una clave principal en otra tabla.

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]
           ,[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:

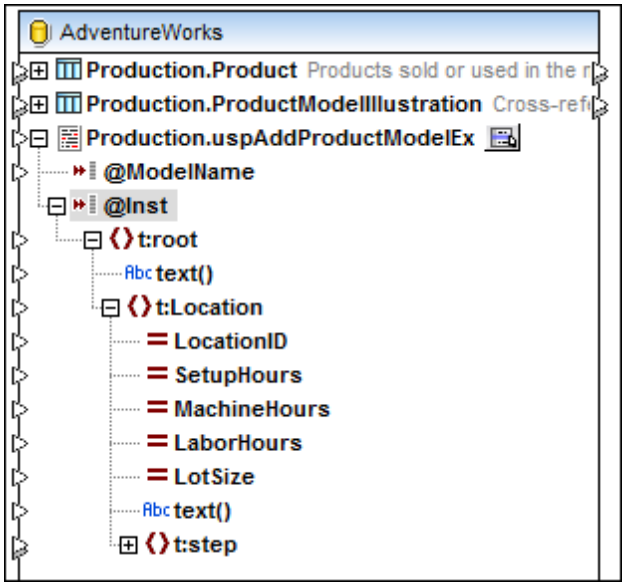
1. Haga clic en el botón interactivo del procedimiento y seleccione **Mostrar nodos como destino** en el menú contextual.



   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 secundario 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**.

 © 2014 Altova GmbH

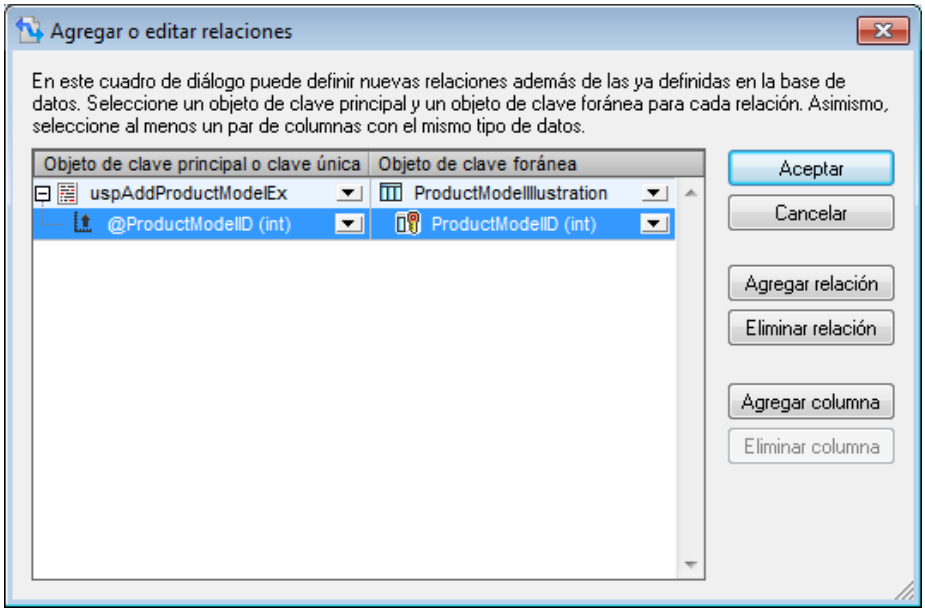


Esto añade los elementos y atributos del esquema XML al componente. El parámetro **`@ModelName`** y todos los parámetros **`@Inst`** ya están disponibles en el componente.

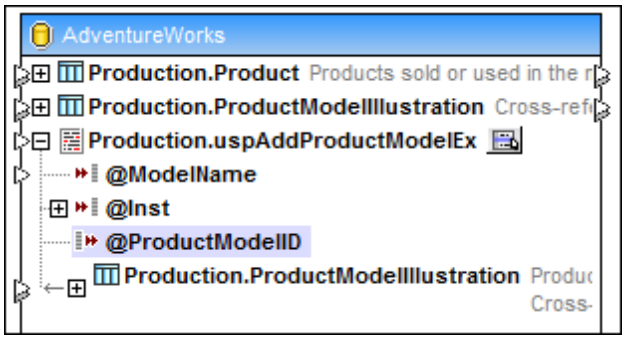
Ahora queremos definir una relación local con una tabla que no tiene conexión directa con la tabla referenciada por los parámetros del procedimiento almacenado (**`production.product`**).

Definir una relación local con una tabla en la que desea generar una clave principal:

1. Haga clic con el botón secundario 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.

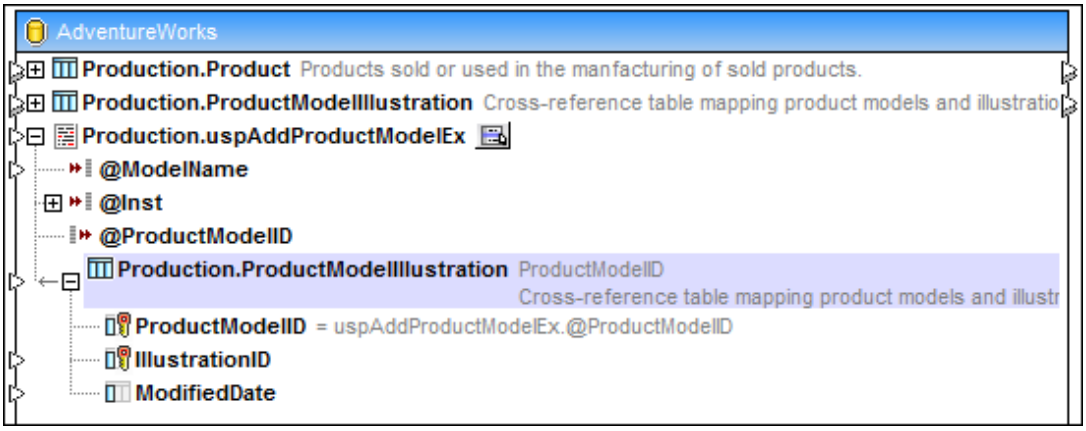


5. Haga clic en el botón **Aceptar** para confirmar.



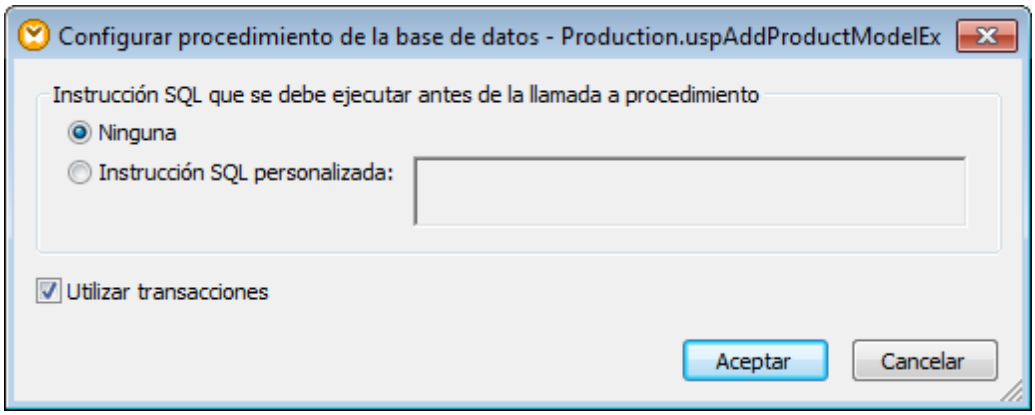
- El parámetro de salida del procedimiento almacenado (**`ProductModelID`**) se añadió al procedimiento almacenado para indicar que se usará en la relación local, pero no tiene iconos de entrada ni salida.
- La tabla **`ProductModelIllustration`** también se añadió como elemento secundario del procedimiento 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.

 © 2014 Altova GmbH



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**.

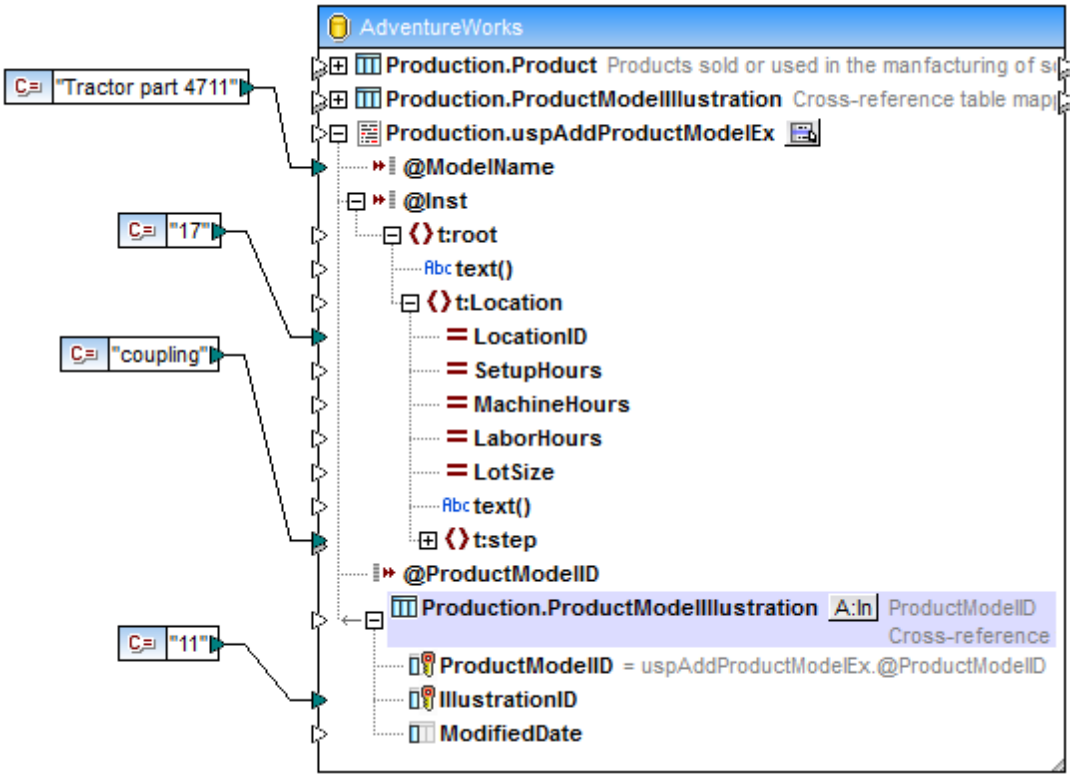


Definir la transacción para el procedimiento almacenado garantiza que tanto la recuperación de la clave como la inserción del registro sucedan durante la misma transacción.

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.

 © 2014 Altova GmbH

10.3 Asignar archivos de texto/CSV

MapForce permite realizar asignaciones de datos usando archivos planos, es decir, archivos de texto y CSV como componentes de origen y destino. Para poder trabajar con este tipo de archivos es necesario seleccionar uno de estos lenguajes: Java, C# o C++ o el motor de ejecución integrado.

Formatos compatibles de archivo de texto:

- Archivos CSV (valores separados por comas o por otros delimitadores)
- Archivos FLF (campos de longitud fija)
- Otros archivos de texto cuya estructura se defina en una plantilla de FlexText (solamente compatible con la edición Enterprise Edition)

En MapForce puede realizar asignaciones de datos entre:

- esquemas XML y archivos planos
- bases de datos y archivos planos
- archivos UN/EDIFACT/ANSI X12 y archivos planos
- una plantilla de FlexText que defina la estructura y el contenido del archivo, creada en el módulo FlexText e insertada como componente. Para más información consulte la sección MapForce FlexText.

Hay dos maneras de generar / guardar los datos de archivos planos asignados:

- Abriendo el panel **Resultados** (lo cual genera una vista previa con el motor de ejecución integrado) y seleccionando la opción de menú **Resultados | Guardar el archivo de salida**. O haciendo clic en el icono para guardar el resultado.
- Seleccionando la opción **Archivo | Generar código en | Java, C# o C++** y compilando y ejecutando el código generado.

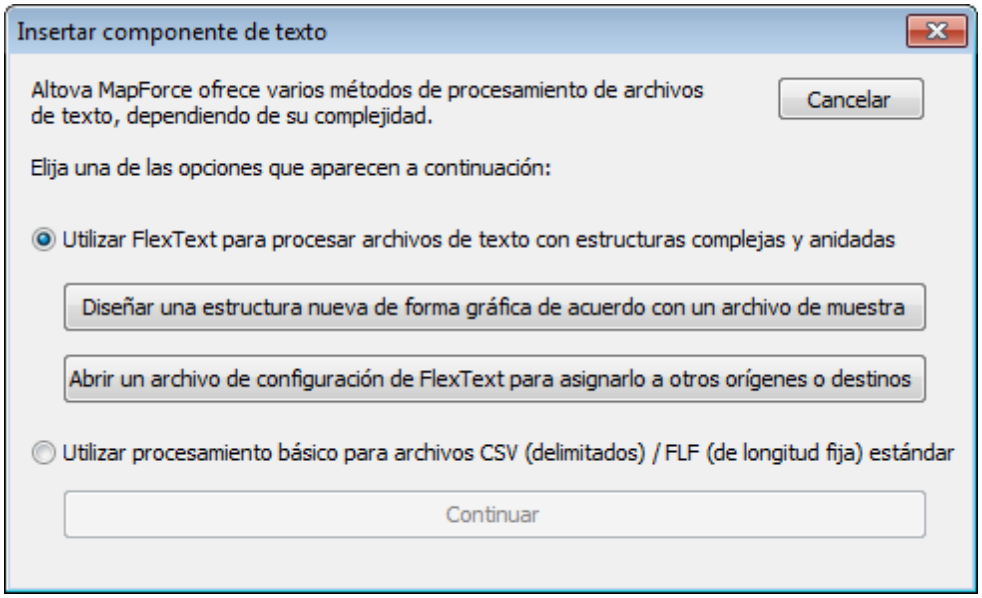
Nota: los ejemplos que aparecen en los apartados siguientes utilizan archivos CSV como componentes de origen/destino, pero también funcionan con archivos de texto de longitud fija. La única diferencia es que la longitud fija de los campos deben definirse manualmente (consulte el apartado Asignar archivos de longitud fija para más información).

10.3.1 Asignar archivos CSV a archivos XML

En este apartado explicamos cómo crear asignaciones de datos entre un sencillo archivo CSV y un archivo XML basado en el esquema `MFCompany.xsd`. Todos los archivos utilizados en este y en los siguientes apartados están disponibles en la carpeta `...\MapForceExamples\Tutorial`.

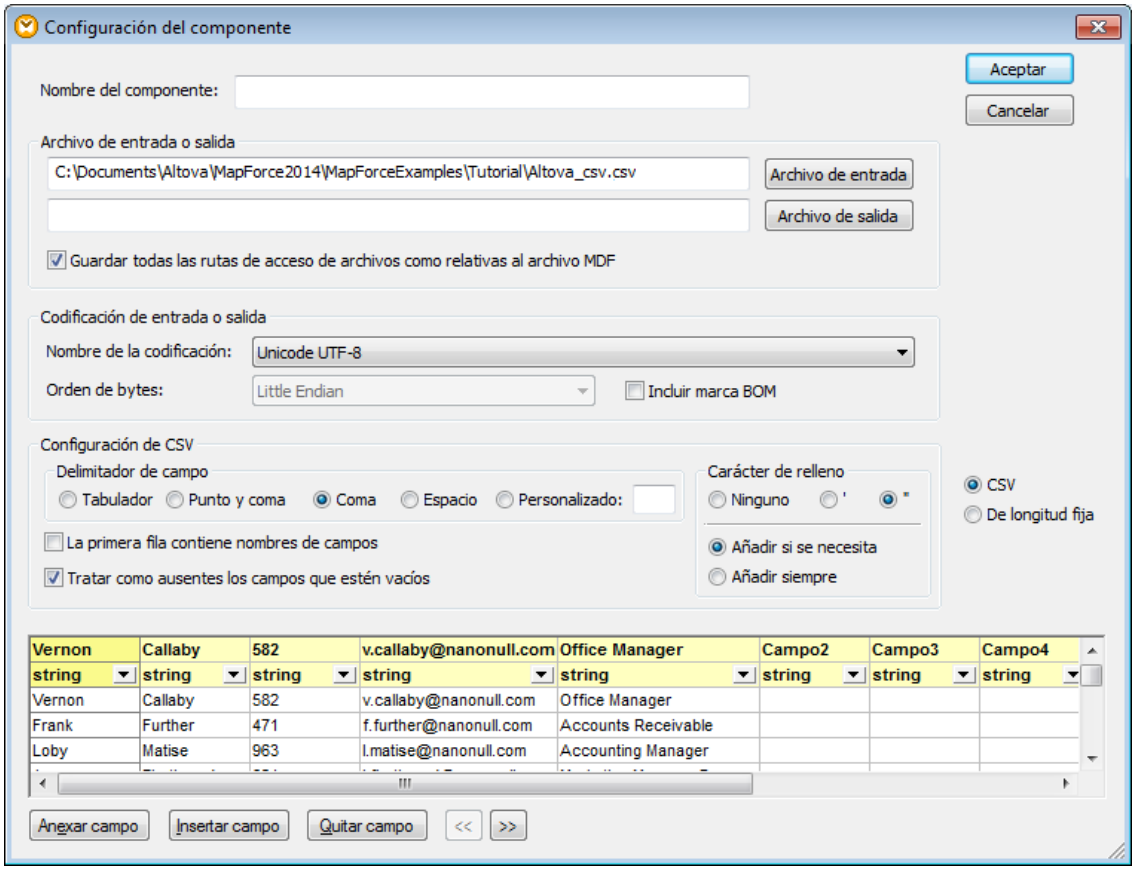
Antes de empezar compruebe que la opción de lenguaje seleccionada es **Java**, **C#**, **C++** o **BUILTIN**.

1. Seleccione la opción de menú **Insertar | Archivo de texto** o haga clic en el icono **Insertar archivo de texto** .
   Esto abre el cuadro de diálogo "Insertar componente de texto".

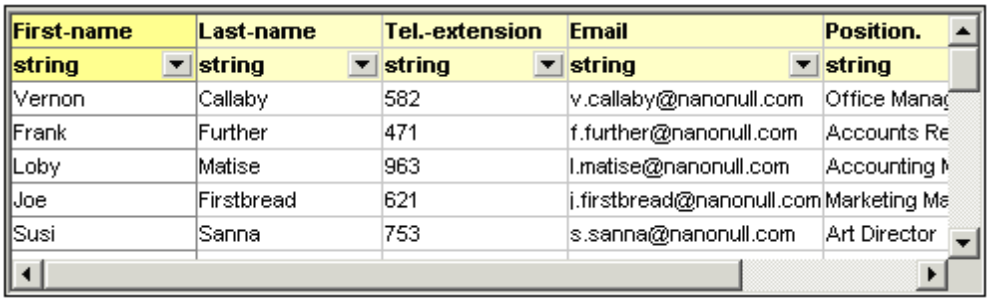


   Seleccione el segundo botón de opción (*Utilizar un procesamiento básico...*) y después haga clic en **Continuar**.
   Esto abre el cuadro de diálogo "Configuración del componente" donde puede seleccionar el tipo de archivo con el que desea trabajar (CSV o FLF). La selección predeterminada es CSV.

 © 2014 Altova GmbH



2. Haga clic en el botón **Archivo de entrada** y seleccione el archivo CSV `Altova_csv.csv`. El contenido del archivo aparece en la ventana de vista previa situada en la parte inferior del diálogo. Observe que la ventana de vista previa puede mostrar un máximo de 20 filas.
3. Haga clic en el título **Campo1** y cambie el texto por **First-name**. Haga lo mismo con los demás campos (cámbielos por **Last-name**, **Tel.-extension**, **Email** y **Position**)

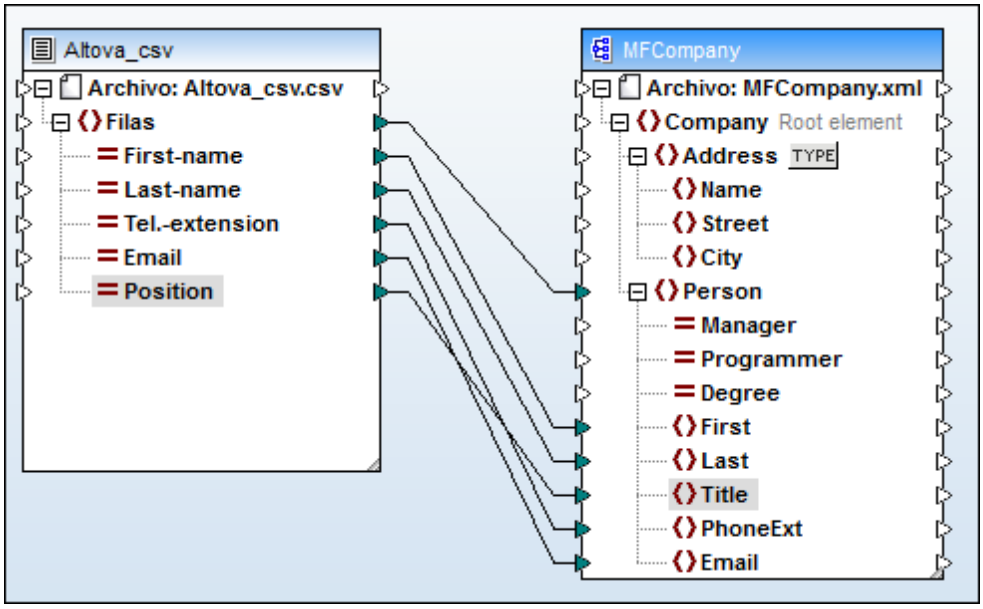


First-name	Last-name	Tel.-extension	Email	Position.
string	string	string	string	string
Vernon	Callaby	582	v.callaby@nanonull.com	Office Manag
Frank	Further	471	f.further@nanonull.com	Accounts Re
Loby	Matise	963	l.matise@nanonull.com	Accounting M
Joe	Firstbread	621	j.firstbread@nanonull.com	Marketing Ma
Susi	Sanna	753	s.sanna@nanonull.com	Art Director

Nota: pulse la tecla **Tabulador** para pasar de un campo a otro de la tabla: campo1, tipo 1, campo2, tipo 2, etc.

4. Cuando termine haga clic en **Aceptar**.
5. Aparece un aviso donde puede definir el nombre del componente. Haga clic en el botón **Cambiar el nombre del componente**.
   Ahora puede ver el componente CSV en la asignación.
6. Seleccione la opción de menú **Insertar | Archivo o esquema XML** y seleccione el esquema `MFCompany.xsd`.

7. En el aviso que aparece haga clic en **Omitir** y seleccione **Company** como elemento raíz.



8. Cree las asignaciones pertinentes entre los dos componentes (compruebe que **Rows** está conectado con **Person**) y abra el panel **Resultados**.

Los datos del archivo CSV se asignaron correctamente al archivo XML.

Nota: es fundamental conectar el elemento **Rows** del componente CSV con el elemento **Person** del esquema de destino. Esta conexión define en qué elementos se debe iterar. Es decir, por cada **Row** del archivo CSV se creará un elemento **Person** nuevo en el archivo XML de salida. Consulte los siguientes apartados para ver más ejemplos.

 © 2014 Altova GmbH

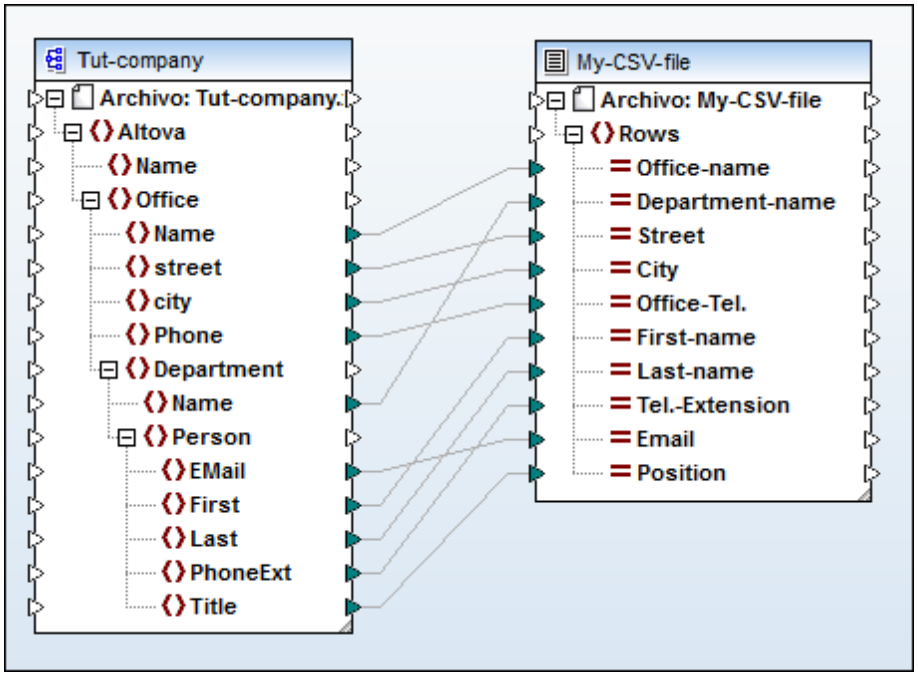
10.3.2 Asignar datos XML a archivos CSV iterando en los elementos

El ejemplo siguiente es el archivo `Tut-xml2csv.mfd` de la carpeta `...\MapForceExamples\Tutorial`.

- `Tut-company.xsd` y `Tut-company.xml` son el esquema de origen y el origen de datos XML respectivamente.
- **My-CSV-file** es el componente de archivo de texto. El nombre del componente se introduce en el campo *Archivo de entrada* del cuadro de diálogo "Configuración del componente".

  Esta asignación se presenta solamente a modo de ejemplo y no pretende ser un caso real de asignación.

Estas son las asignaciones que normalmente se crearían entre el archivo XML y el archivo CSV de nuestro ejemplo.

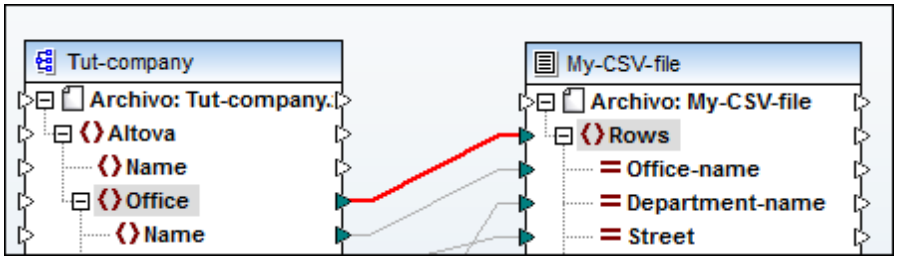


Al abrir el panel **Resultados** obtenemos este resultado. Como puede ver, el resultado incluye los datos de una oficina solamente.

```
1   "Microtech Partners, Inc.",Level 2 support,Perro Bvd 1324,Ottow
2
```

Para poder iterar en todas las oficinas y que sus datos aparezcan en el archivo CSV, es necesario conectar el elemento **`Office`** con **`Rows`**. Esto significa que por cada elemento **`Office`** del XML de origen, se crea una fila en el archivo CSV de destino. MapForce permite especificar el campo o elemento que debe servir de raíz/iterador para el elemento **`Rows`**.

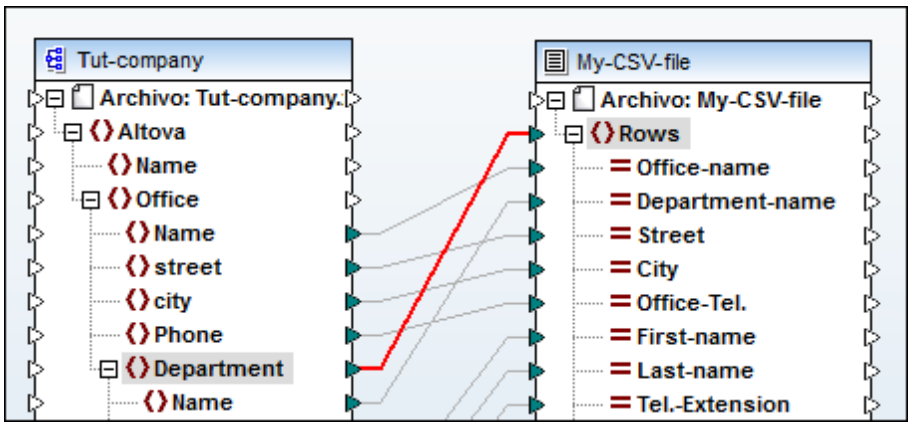
Si asignamos el elemento **`Office`** a **`Rows`**, se generan resultados para cada elemento **`Office`**.



Los elementos **Office** se crean en el orden en el que aparecen en el archivo 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
```

Si asignamos el elemento **Department** a **Rows**, se generan resultados para cada elemento **Department**.

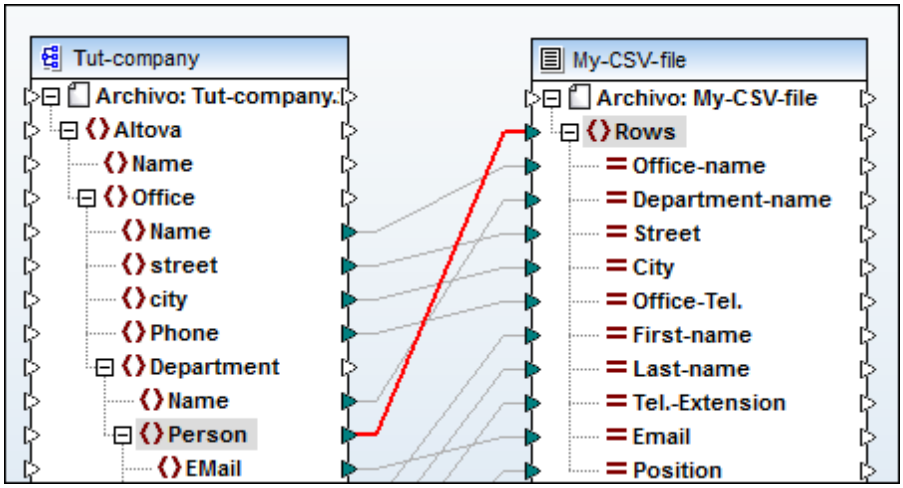


Los elementos **Department** se crean en el orden en el que aparecen en el archivo de origen.

```
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
```

Si asignamos el elemento **Person** a **Rows**, se generan resultados para cada elemento **Person**.

 © 2014 Altova GmbH



Los elementos **`Person`** se crean en el orden en el que aparecen en el archivo de origen (cada **`Person`**, dentro de cada **`Department`**, para cada **`Office`**)

```
1  "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Albert,Aldrich,582,A.Ald
2  "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Bert,Bander,471,b.bander
3  "Microtech, Inc.",Admin,Major Ave 1,Vancouver,5588339,Clive,Clovis,963,c.clovi
4  "Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,5588339,Dave,Durne
5  "Microtech, Inc.",Sales and Marketing,Major Ave 1,Vancouver,5588339,Eve,Ellas,
6  "Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,Fred,Fortunas,95
7  "Microtech, Inc.",Manufacturing,Major Ave 1,Vancouver,5588339,Gerry,Gundall,65
```

10.3.3 Crear jerarquías a partir de archivos CSV y FLF

El ejemplo siguiente es el archivo `Tut-headerDetail.mfd` de la carpeta `...\MapForceExamples\Tutorial`.

Este ejemplo usa un archivo CSV con campos que definen el tipo de registro y tiene este formato:

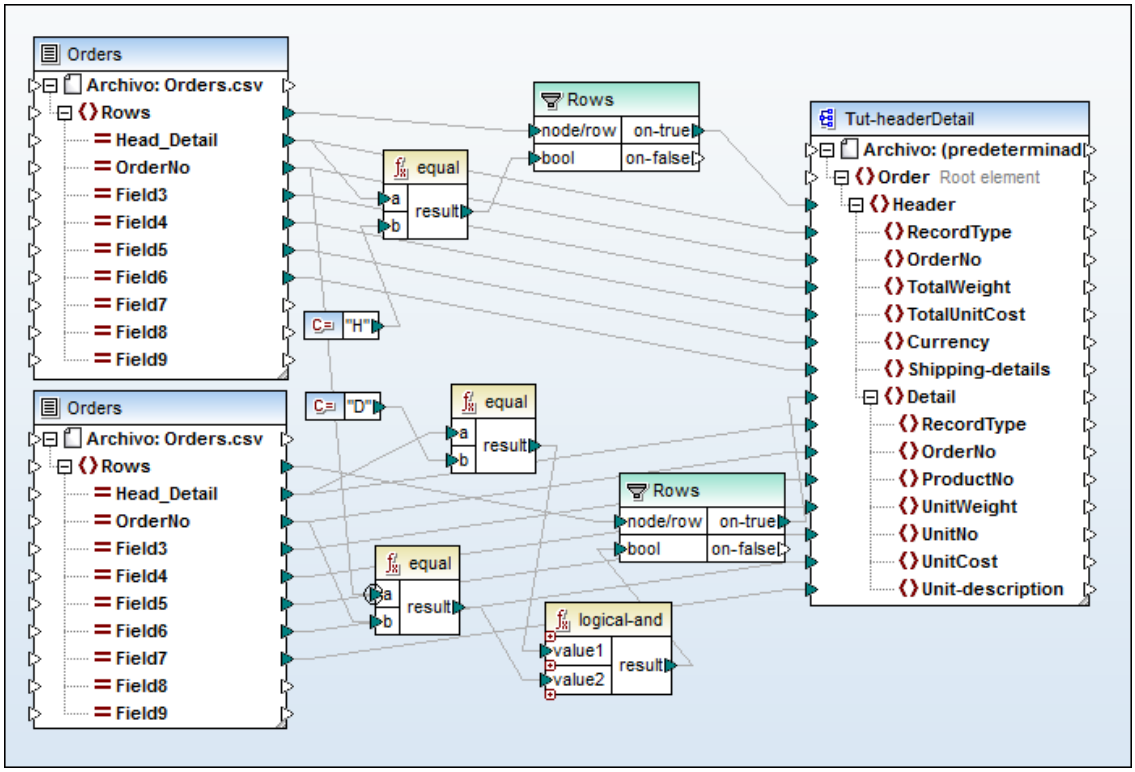
- **Campo 1**: H denota un registro **`Header`**, mientras que D denota un registro **`Detail`**.
- **Campo 2**: una clave común, tanto para los registros **`Header`** como los registros **`Detail`**.
- Un registro **`Header`**/**`Detail`** en cada línea.

Crear estructuras XML jerárquicas a partir de archivos planos usando campos clave
Aquí puede ver el contenido del archivo `Orders.csv`.

```
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,,
```

El objetivo de la asignación es:

- asignar el archivo plano CVS a un archivo XML jerárquico,
- filtrar los registros **`Header`**, designados con una **`H`**,
- y asociar los registros **`Detail`** (designados con una **`D`**) a los registros **`Header`**



Para conseguirlo hace falta que los registros **`Header`** y **`Detail`** tengan un campo en común. En este caso el campo común (también llamado *clave*) es el segundo campo del archivo CSV (es decir, **`OrderNo`**). En el archivo CSV el primer registro **`Header`** y los dos primeros registros **`Detail`**

 © 2014 Altova GmbH

tienen el valor común **111**.

Notas:
El archivo `Orders.csv` se insertó dos veces en la asignación para que el proceso sea 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 elemento secundario de **`Header`**.

El primer archivo de texto `Orders.csv` suministra los registros **`Header`** (y todos los campos asignados) al elemento **`Header`** del archivo de esquema de destino. El componente de filtrado **Rows** filtra todos los registros que empiecen con H y los envía al elemento **`Header`** del archivo de esquema.

El segundo archivo de texto `Orders.csv` suministra los registros **`Detail`** (y todos los campos asignados) filtrando los registros **`Detail`** que coinciden con la clave **`OrderNo`** del registro **`Header`**. Esto se consigue:

- comparando el campo **`OrderNo`** del registro **`Header`** con el mismo campo del registro **`Detail`** con la función **`equal`** (el contexto prioritario se establece en el parámetro **`a`**).
- usando la función **`logical-and`** para suministrar solamente los registros **`Detail`** que contengan el mismo campo **`OrderNo`** que el registro **`Header`**.

  El parámetro **`on-true`** del filtro **Rows** envía los registros filtrados a los elementos **`Header`** y **`Detail`** del esquema de destino.

En el panel **Resultados** puede ver el archivo XML generado por esta asignación. Cada registro **`Header`** contiene sus datos y todos los registros **`Detail`** que tienen el mismo **`OrderNo`**.


```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation
3    <Header>
4      <RecordType>H</RecordType>
5      <OrderNo>111</OrderNo>
6      <TotalWeight>332.1</TotalWeight>
7      <TotalUnitCost>22537.7</TotalUnitCost>
8      <Currency/>
9      <Shipping-details>Container ship</Shipping-details>
10     <Detail>
11       <RecordType>D</RecordType>
12       <OrderNo>111</OrderNo>
13       <ProductNo>A-1579-227</ProductNo>
14       <UnitWeight>10</UnitWeight>
15       <UnitNo>3</UnitNo>
16       <UnitCost>400</UnitCost>
17       <Unit-description>Microtome</Unit-description>
18     </Detail>
19     <Detail>
20       <RecordType>D</RecordType>
21       <OrderNo>111</OrderNo>
22       <ProductNo>B-152-427</ProductNo>
23       <UnitWeight>7</UnitWeight>
24       <UnitNo>6</UnitNo>
25       <UnitCost>1200</UnitCost>
26       <Unit-description>Miscellaneous</Unit-description>
27     </Detail>
28   </Header>
```


 © 2014 Altova GmbH

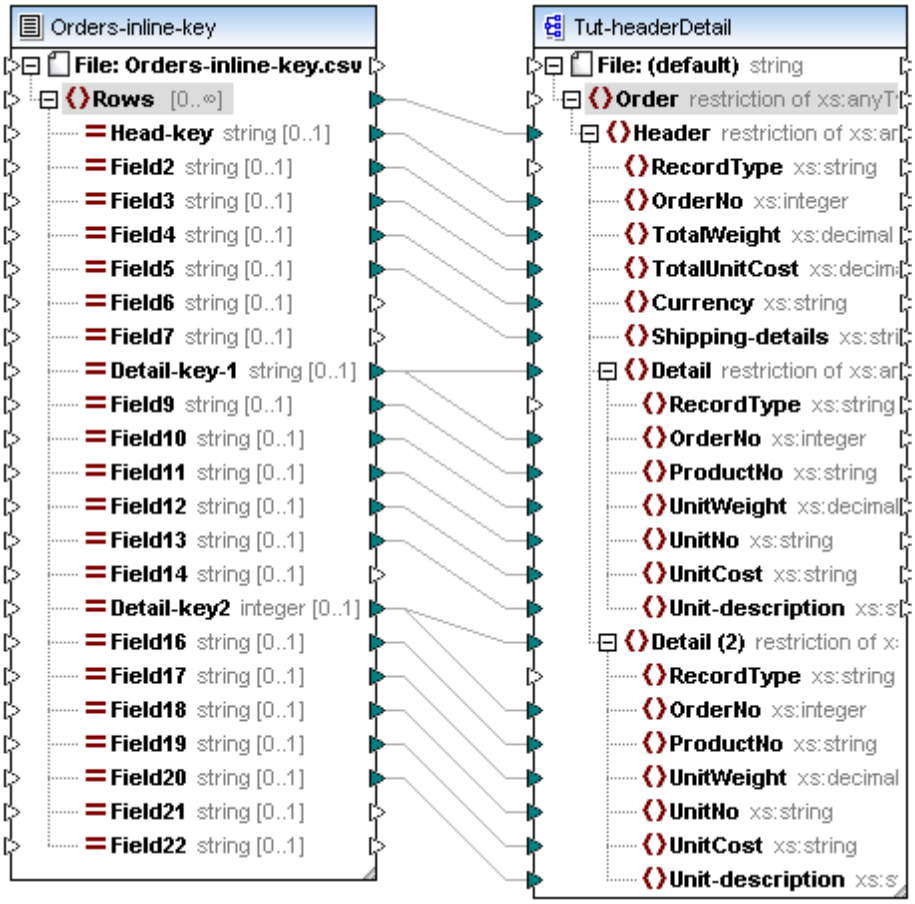
El segundo ejemplo, `Head-detail-inline.mfd`, usa un archivo CSV ligeramente diferente y está en la carpeta `...\MapForceExamples\Tutorial`.

- No hay disponible ningún designador de registro (ni H ni D)
- El primer campo del archivo CSV (`Head-key` o `Detail-keyes`) es el campo común o clave de los registros `Header` y `Detail`. El campo se asigna al elemento `OrderNo` del esquema de destino
- Todos los campos **`Header`** y **`Detail`** aparecen 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,,,,,
```

Nota importante:

- Los campos clave están asignados a los elementos **`OrderNo`** del esquema de destino.
- El elemento **`Detail`** del esquema de destino se duplicó (**`Detail (2)`**). Esto permite asignar el segundo conjunto de registros **`Detail`** al elemento adecuado.
- El resultado de esta asignación es el mismo archivo XML del primer ejemplo de este apartado.

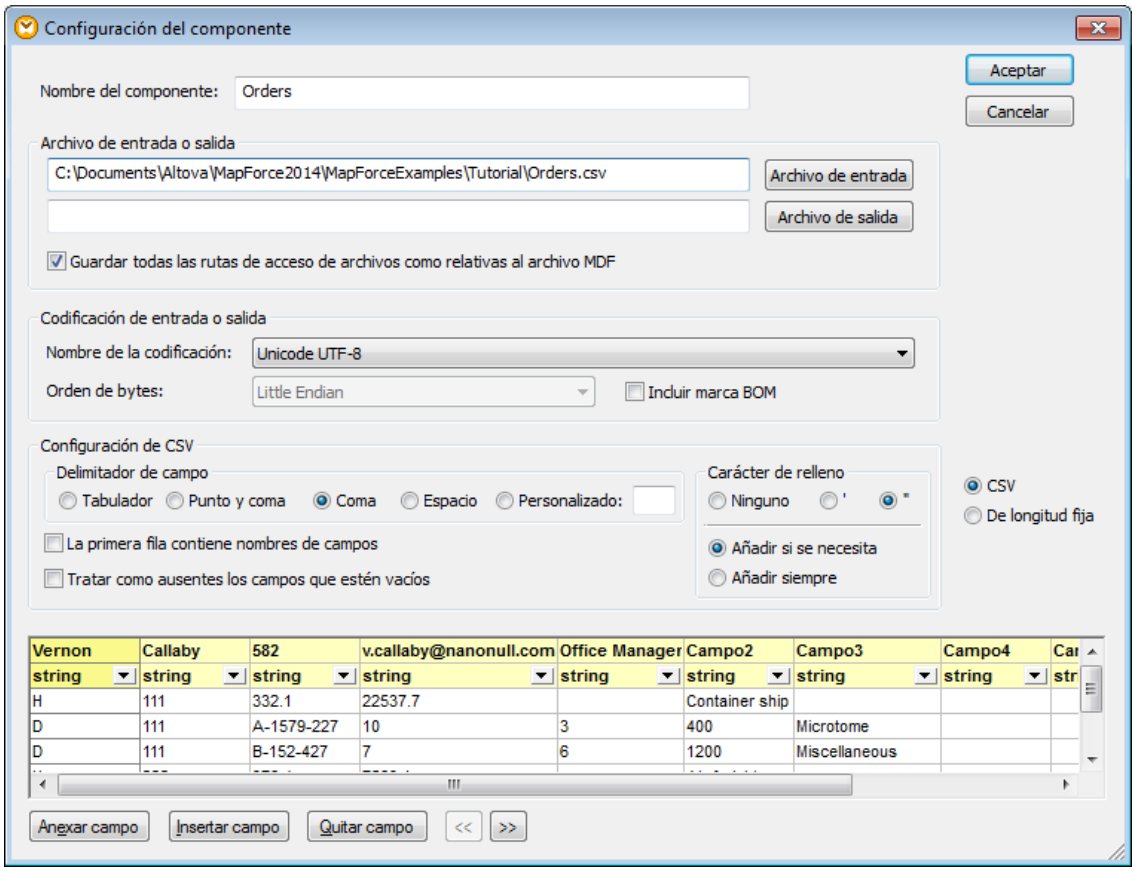


10.3.4 Opciones de los archivos CSV

Haga clic con el botón secundario en el componente **Orders_csv** del archivo `Tut-headerDetail.mfd` y seleccione **Propiedades** para abrir el cuadro de diálogo "Configuración del componente".

Cuadro de diálogo "Configuración del componente"
En este cuadro de diálogo, los tipos de los campos se comprueban automáticamente. Si los datos de entrada y el formato del campo definido en este cuadro de diálogo no coinciden, los datos se señalan en rojo. P. ej. si cambiamos el tipo del **campo1** de `string` a `integer`, todos los registros de la columna aparecen en rojo.

Nota: los tipos de campo de las columnas se basan en los tipos de datos XML Schema predeterminados. P. ej. el tipo `Date` tiene el formato AAAA-MM-DD.



Archivo de entrada:
Seleccione el archivo CVS que desea usar como archivo de origen para este componente. Este nombre de archivo se usará para leer los datos de ejemplo y la información sobre los campos, para la vista previa y para la generación de código.

Nota: si se trata de un componente de texto de **destino**, puede dejar vacío este campo. En este caso, la codificación de archivo predeterminada es UTF-8. Puede definir el tipo y el nombre de cada campo en el panel inferior y hacer clic en **Aceptar** para crear el archivo de texto de destino.

Cuando termine de diseñar la asignación, abra el panel **Resultados** para ver y **guardar** el archivo

 © 2014 Altova GmbH

de texto de destino (haga clic en el icono **Guardar resultado generado**).

Si escribe un nombre de archivo en este cuadro de texto (sin extensión de archivo), el nombre se le asigna al componente.

Archivo de salida:
Seleccione el archivo de destino en el que quiere guardar los datos de salida cuando genere código con MapForce. Cuando genere código, compruebe que los archivos de entrada y salida son diferentes. De lo contrario, el archivo de origen se sobrescribirá cuando se ejecute el código. Esto ocurre cuando se genera código para Java, C++ o C# o cuando se guarda el archivo de salida con BUILTIN (motor de ejecución integrado).

Codificación del archivo:
Permite definir/seleccionar la codificación de caracteres de los archivos de texto de entrada y salida. Si deja vacío el campo *Archivo de entrada*, la codificación de archivo predeterminada es UTF-8.

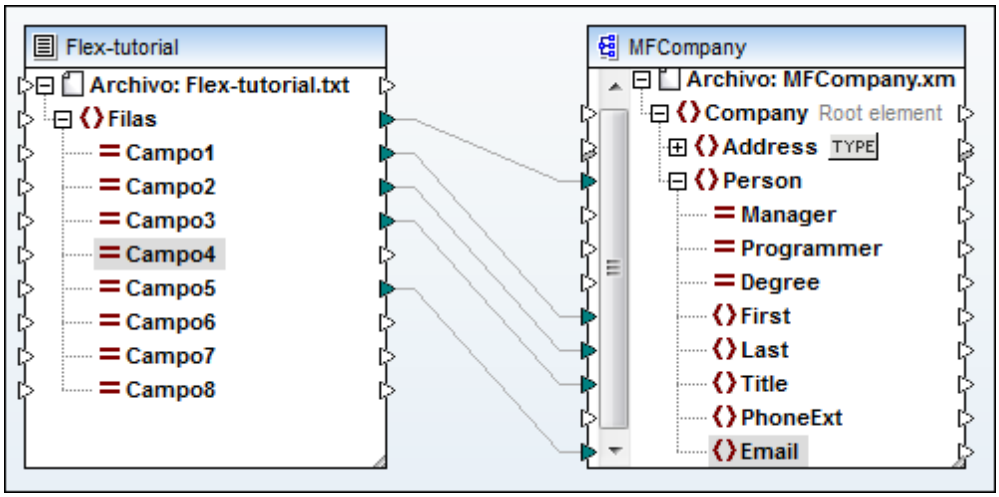
Configuración de CSV (Delimitador de campo):
Seleccione el tipo de delimitador para el archivo de texto (por defecto, los archivos CSV están delimitados por comas). También puede introducir un delimitador personalizado en el campo *Personalizado*.

Para ello, ponga el cursor dentro del campo *Personalizado* y:

- Pulse una tecla del teclado para introducir un valor nuevo o
- Haga doble clic en el campo *Personalizado* (para marcar el valor actual) y pulse una tecla diferente para cambiar el valor actual.

La primera fila contiene nombres de campo
Indica que los **valores** del primer registro del archivo de texto son encabezados de columna (visibles en la ventana de vista previa). Los encabezados de columna aparecen como nombres de elemento dentro del componente de texto en la ventana de asignación.

Tratar como ausentes los campos que estén vacíos
Permite definir que los campos vacíos del archivo de origen no produzcan un elemento vacío (elemento/atributo) en el archivo de destino.



Opción **activada**:
Si el archivo de origen solamente contiene datos en el **`Campo1`** (los campos **`Campo2`** a **`Campo5`** están vacíos), los elementos de destino que no reciben datos del archivo de origen no aparecen en el archivo de resultados (línea 39).

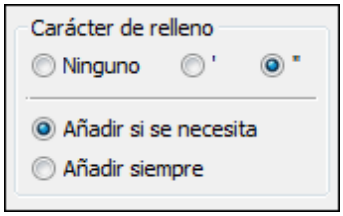
```
38  <Person>
39    <First>General outgassing pollutants</First>
40  </Person>
41  <Person>
42    <First>1100</First>
43    <Last>897</Last>
```

Opción **desactivada**:
Los campos vacíos del archivo de origen producen elementos de destino en el archivo de salida (es decir, los elementos **`Last`**, **`Title`** y **`Email`**).

```
33  <Person>
34    <First>General outgassing pollutants</First>
35    <Last/>
36    <Title/>
37    <Email/>
38  </Person>
39  <Person>
40    <First>1100</First>
41    <Last>897</Last>
```

Nota: los delimitadores para los campos vacíos del archivo de origen deben existir (p. ej. `General outgassing pollutants`**,,,,**).

Comillas:
Si el **archivo de entrada** contiene comillas alrededor de los valores de campo, seleccione el tipo de comillas que existe en el archivo de entrada. Para los archivos de salida se usará la misma configuraicón. Si quiere ver un ejemplo, consulte el apartado Asignar archivos CSV a archivos XML.



Para los **archivos de salida** debe especificar en qué casos se insertan las comillas:

Añadir si se necesita:
Añade el tipo de comillas seleccionado solamente en los campos donde el texto incluya el delimitador de campo o saltos de línea.

Añadir siempre:
Añade el tipo de comillas seleccionado en todos los campos del archivo CSV de salida.

Anexar campo, **Insertar campo**, **Quitar campo**
Permite anexar, insertar o quitar campos de la ventana de vista previa, que define la estructura del componente CSV.

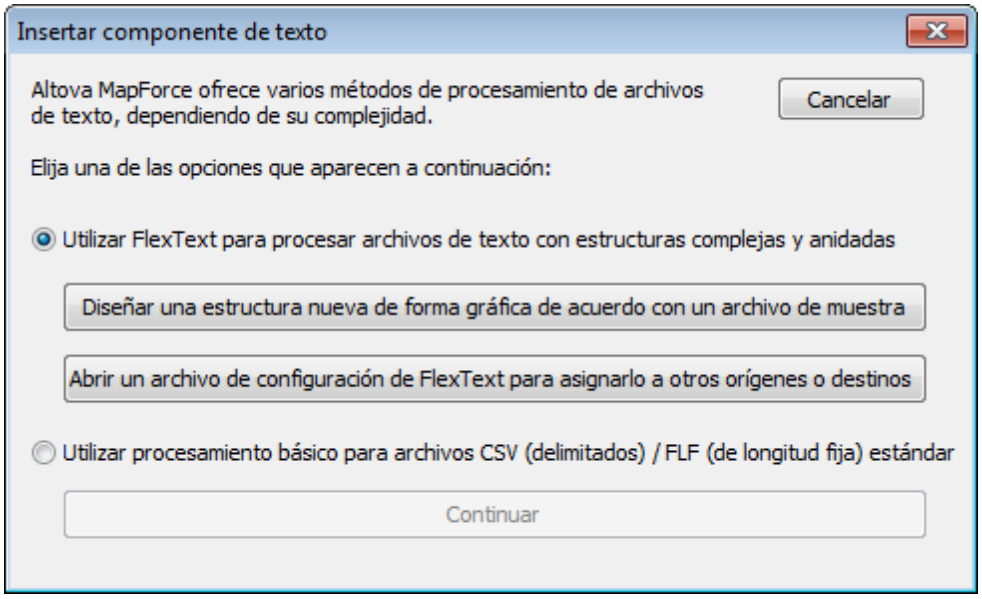
 © 2014 Altova GmbH

Botones Izquierda / **Derecha**
Haga clic en estos botones para mover a la izquierda/a la derecha la columna que está activa en la vista previa.

10.3.5 Asignar archivos de longitud fija (a una BD)

Este apartado explica cómo asignar un sencillo archivo de texto a una base de datos MS Access. El archivo de texto de origen es una cadena continua sin retornos de carro ni avances de línea. Todos los archivos utilizados en este apartado están disponibles en la carpeta `...\MapForceExamples\Tutorial`.

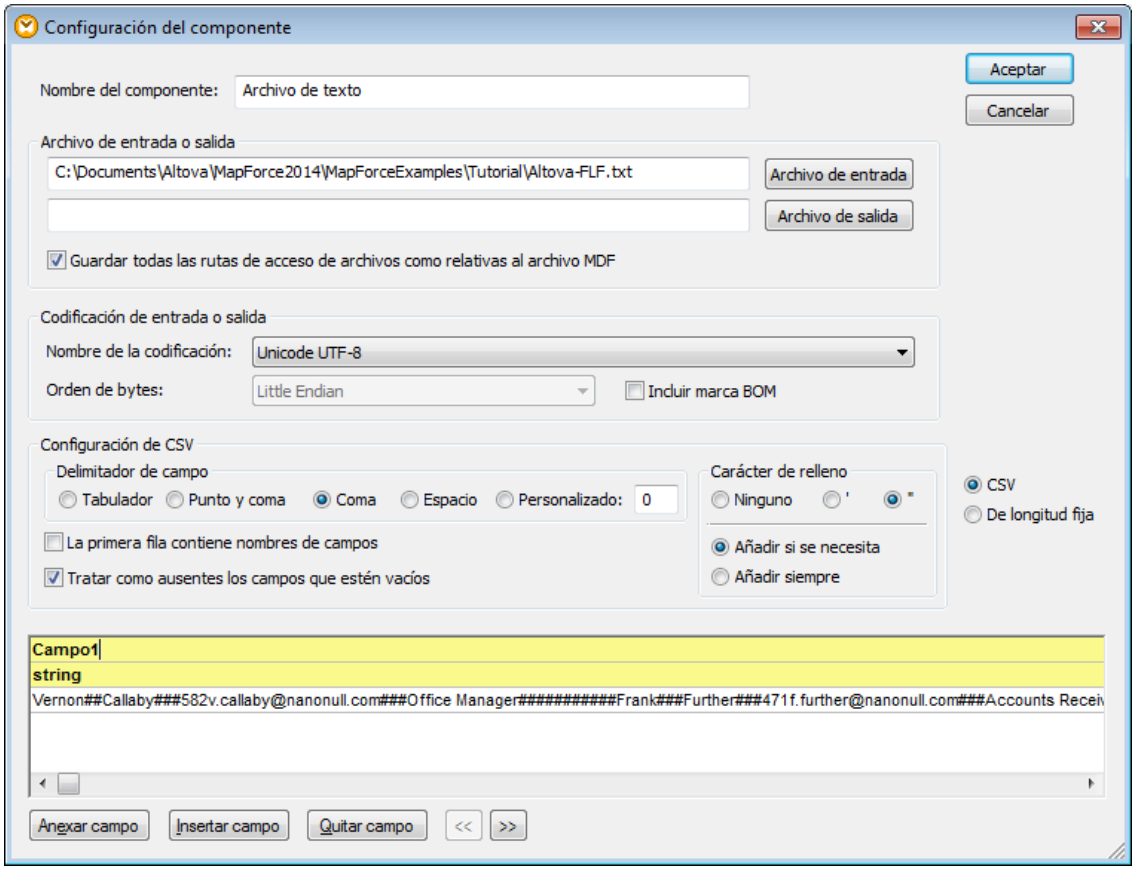
1. Seleccione la opción de menú **Insertar | Archivo de texto** o haga clic en el icono **Insertar archivo de texto** .
   Se abre el cuadro de diálogo "Insertar componente de texto".



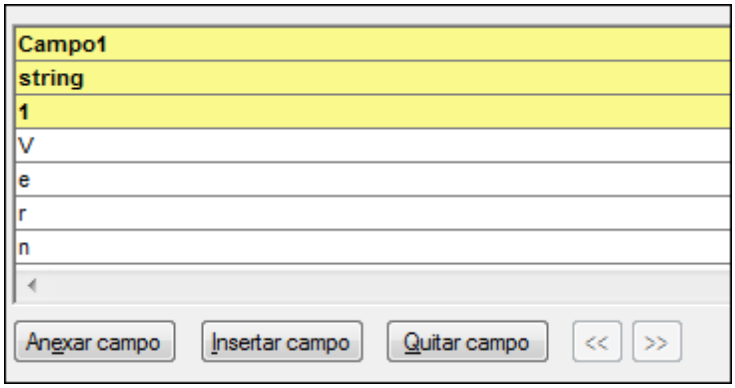
   Seleccione el botón de opción *Utilizar un procesamiento básico...* y haga clic en **Continuar**.
   Se abre el cuadro de diálogo "Configuración del componente" donde puede seleccionar el tipo de archivo con el que desea trabajar y configurarlo.

2. Haga clic en el botón **Archivo de entrada** y seleccione el archivo `Altova-FLF.txt`. Observe que el archivo incluye una sola cadena de texto que contiene los caracteres de relleno #.

 © 2014 Altova GmbH



3. Haga clic en el botón de opción *De longitud fija* (a la derecha del diálogo, bajo *CSV*).
4. Desactive la casilla *Asumir los delimitadores de registro presentes*.

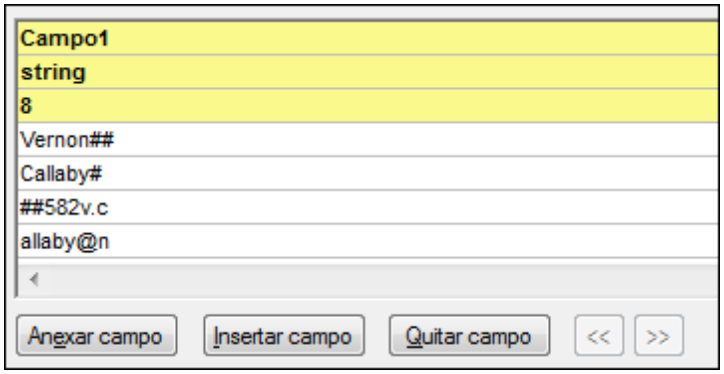


Al desactivar esta opción la vista previa cambia. Ahora tenemos un formato fijo compuesto por:

- un solo campo llamado **Campo1**
- cuyo formato es de tipo `string`
- cuya longitud fija es 1 carácter

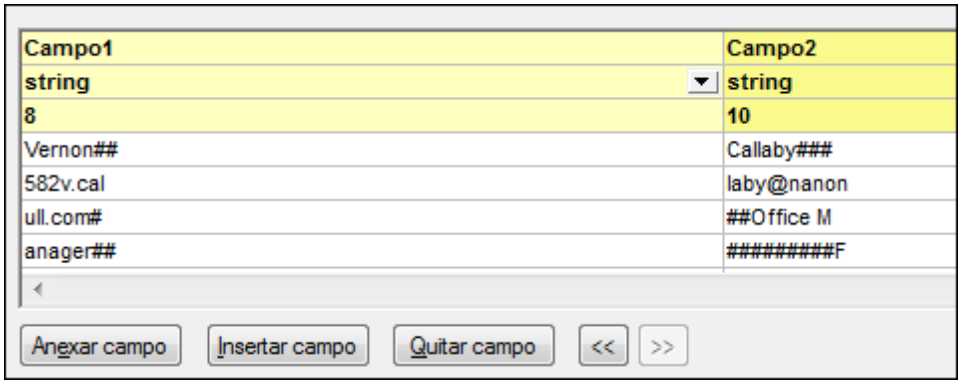
El **Campo1** ahora contiene más datos.

5. Haga clic en la fila que contiene el carácter `1`, cambie este valor por el número `8` y pulse **Entrar**.

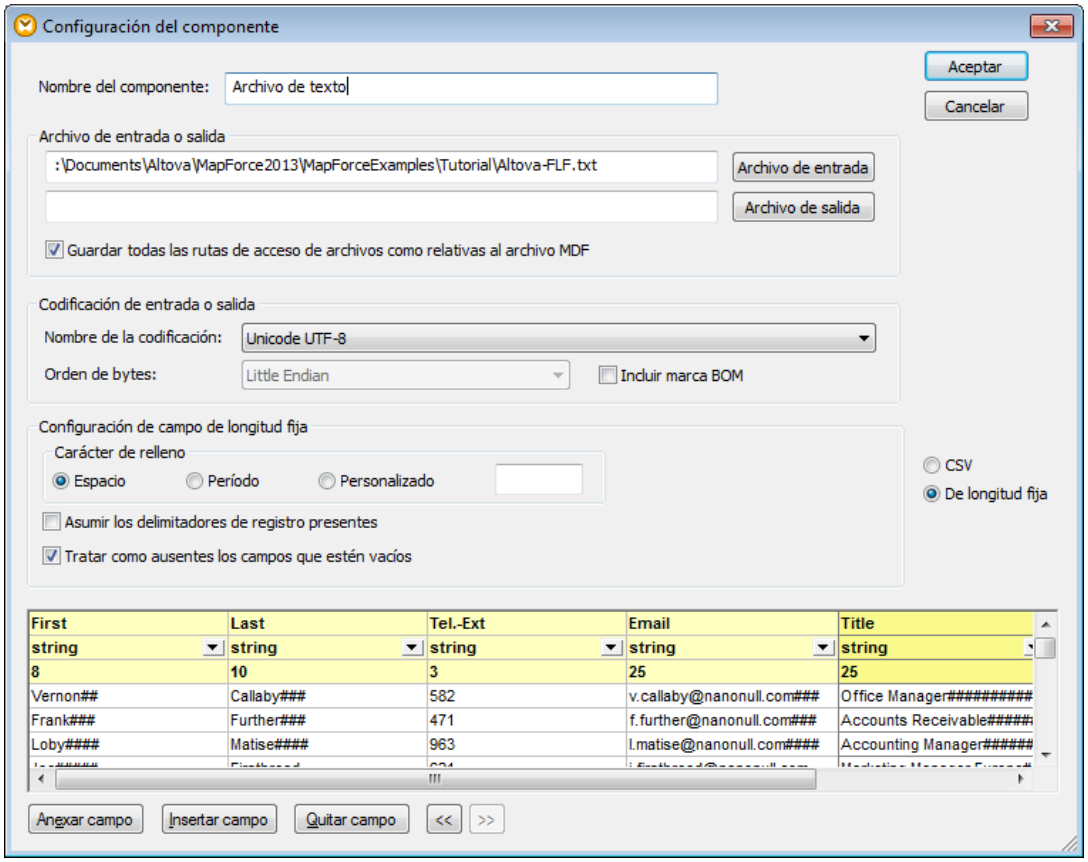


Ahora podemos ver más datos en la primera columna porque su nueva longitud fija es 8 caracteres.

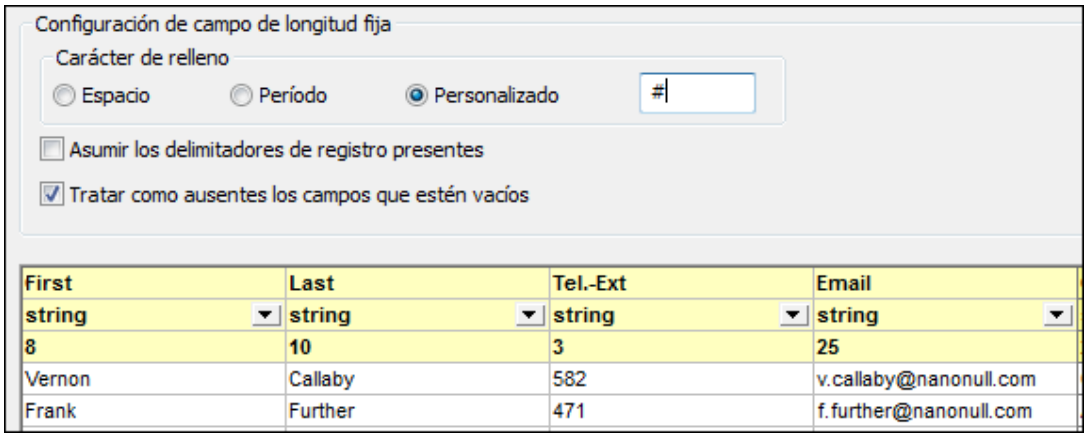
6. Haga clic en el botón **Anexar campo** y añada un campo nuevo (con longitud fija de 10 caracteres).



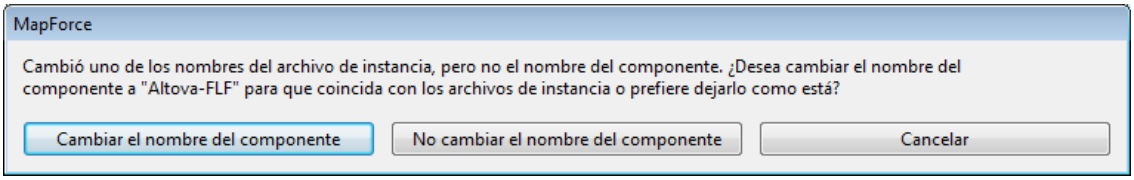
7. Siga los mismos pasos para crear tres campos más con longitud fija de 3, 25 y 25 caracteres respectivamente. Cambie el nombre de los campos por **`First`**, **`Last`**, **`Tel.-Ext`**, **`Email`**, **`Title`**. La vista previa tiene ahora este aspecto:



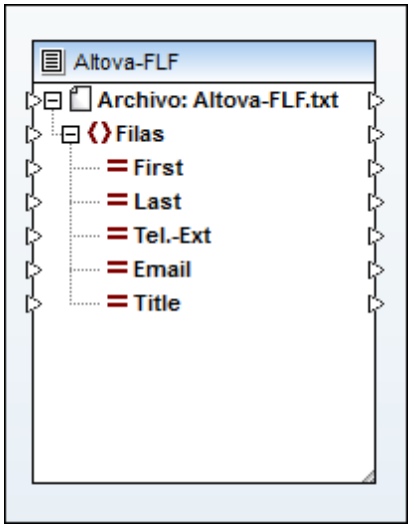
8. En el grupo de opciones *Configuración de campo de longitud fija*, seleccione el botón de opción *Personalizado* y escriba el carácter #. Como resultado se eliminan todos los caracteres de relleno # de la cadena de texto de entrada.



9. Haga clic en **Aceptar** para terminar de definir el componente de texto. Aparece el aviso de la imagen siguiente.
   Todos los componentes de una asignación tienen un nombre. El nombre predeterminado de un archivo de texto es **Archivo de texto**. Si selecciona un archivo de texto como entrada del componente, MapForce le pregunta si desea cambiar también el nombre del componente.



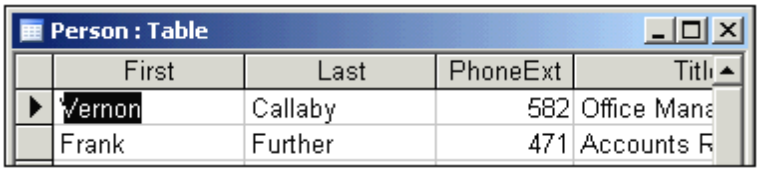
10. Haga clic en el botón **Cambiar el nombre del componente**.



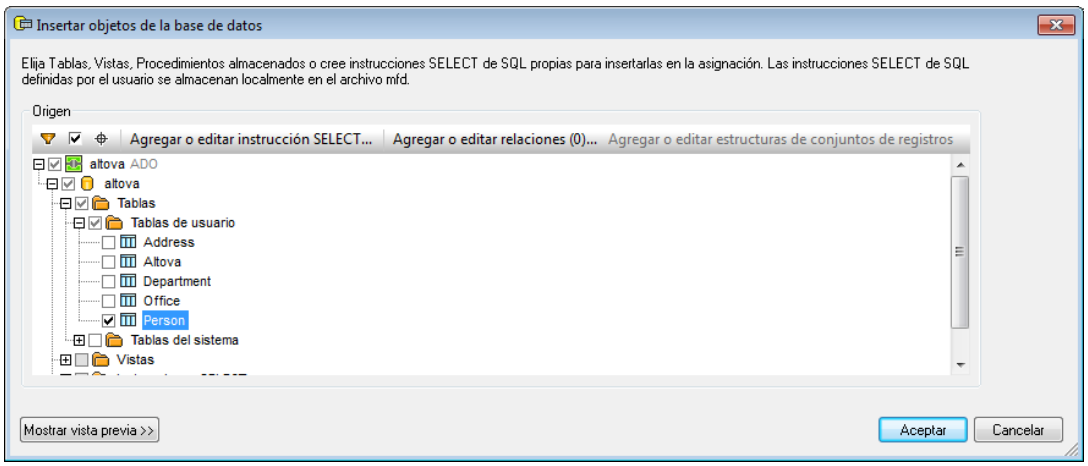
El componente de archivo de texto aparece en la ventana de asignación. Ahora puede asignar sus datos a otro componente de origen/destino.

 © 2014 Altova GmbH

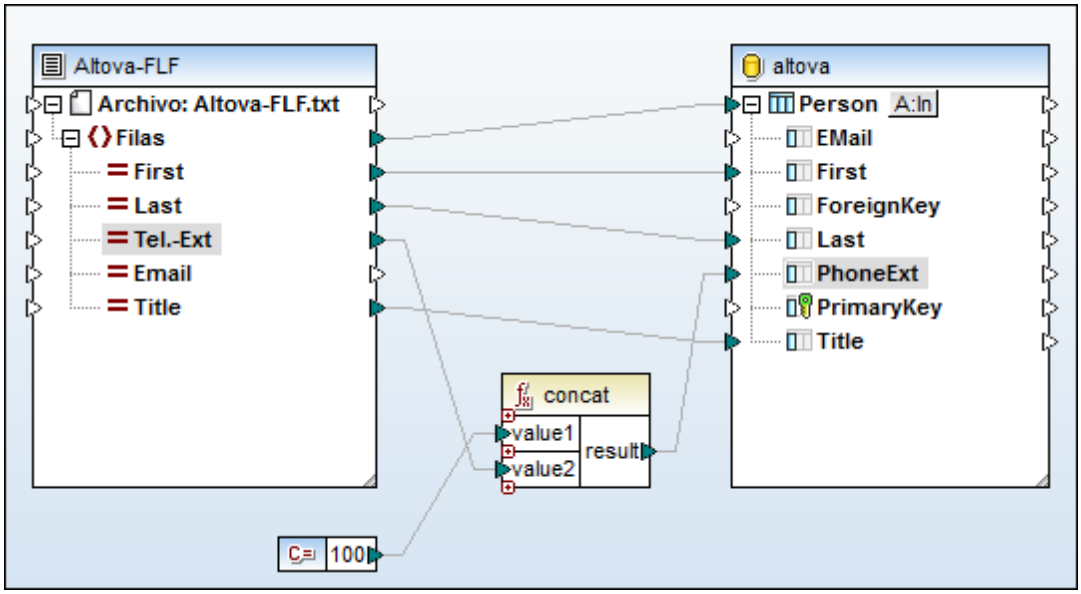
Asignar archivos de texto a una base de datos:
Ahora usaremos el archivo de longitud fija para actualizar las extensiones telefónicas guardadas en la base de datos `altova.mdb`.



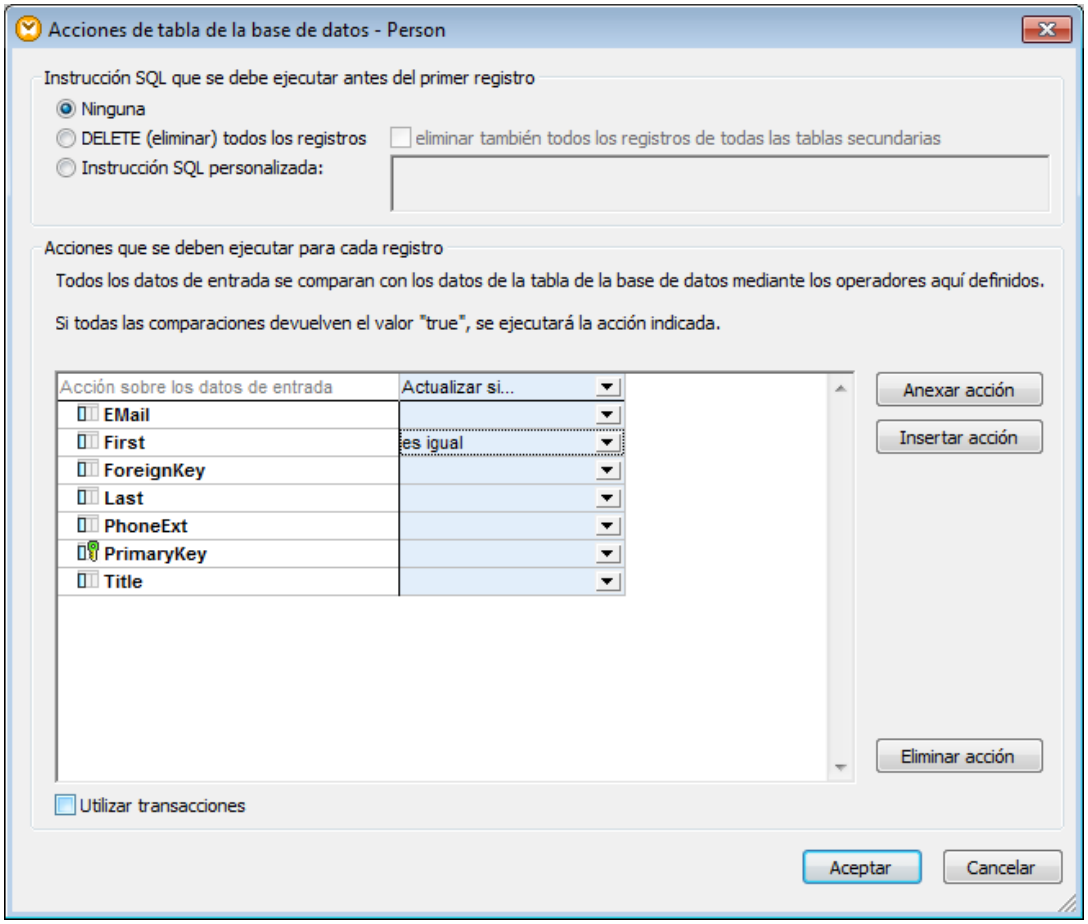
1. Seleccione la opción de menú **Insertar | Base de datos,** seleccione el botón de opción *Microsoft Access* y haga clic en **Siguiente**.
2. Seleccione la base de datos `altova.mdb` de la carpeta `...\MapForceExamples\Tutorial` y haga clic en **Siguiente**.
3. Marque la casilla de la tabla **`Person`**.



4. Haga clic en **Aceptar** para crear el componente de base de datos.
5. Haga clic en el botón de expansión para ver el contenido de la tabla de la BD.
6. Inserte la función **`concat`**.
7. Seleccione la opción de menú **Insertar | Constante**, haga clic en el botón de opción *Número* e introduzca el número `100` (el nuevo prefijo de extensión telefónica).
8. Cree las asignaciones que aparecen en esta imagen.

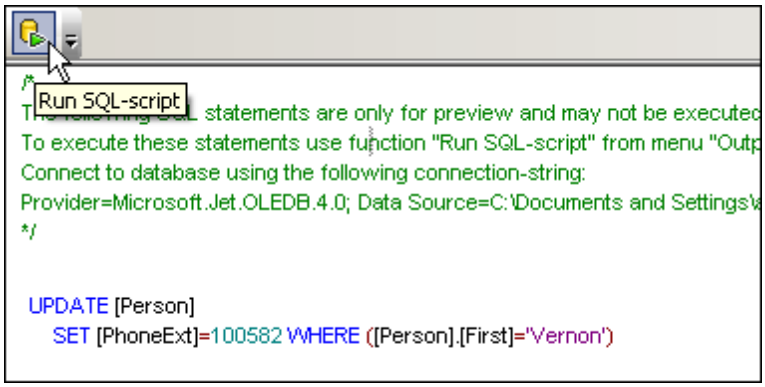


9. Haga clic en el icono de acción de tabla A:I de la entrada **Person**.

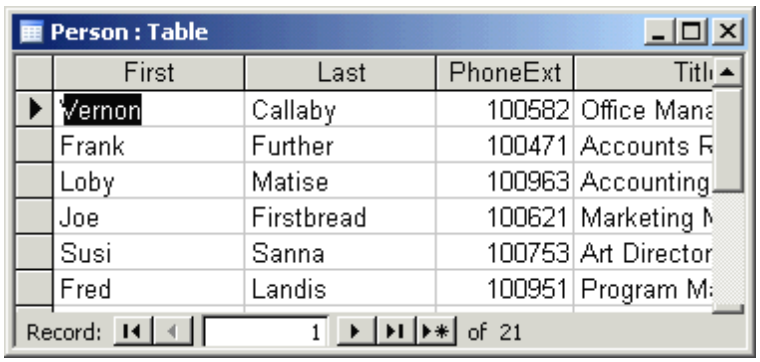


10. Haga clic en el cuadro combinado *Acción sobre los datos de entrada* y seleccione la opción **Actualizar si**.

11. Haga clic en el cuadro combinado de la fila **`First`**, seleccione la opción **es igual** y haga clic en **Aceptar**.
    Los datos de la tabla **`Person`** se actualizan solamente si el nombre de **`First`** es idéntico en el componente de origen y en la BD. Si esta condición se cumple, la acción que se lleva a cabo viene definida por la asignación. En este caso, a la extensión telefónica se le añade el prefijo 100 y la extensión se introduce en el campo **`PhoneExt`** de la tabla **`Person`**.
12. Abra el panel **Resultados** para generar una vista previa de la instrucción SQL. Para ejecutar las instrucciones SQL haga clic en el botón **Ejecutar script SQL**.



En la BD se actualiza el campo de extensión telefónica de todas las personas.

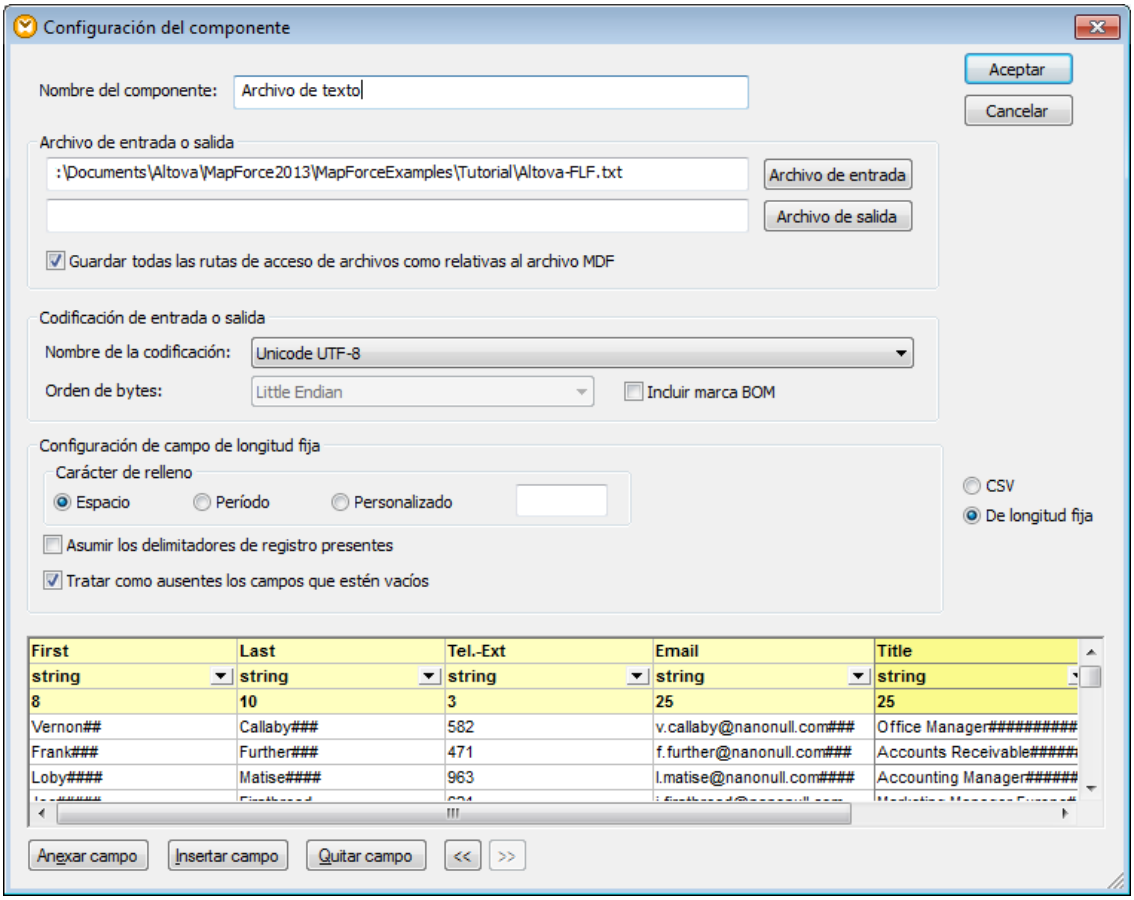


Opciones de los archivos de longitud fija

Haga clic con el botón secundario en el componente de archivo **Altova-FLF** y seleccione la opción **Propiedades**. Aparece el cuadro de diálogo "Configuración del componente".

Cuadro de diálogo "Configuración del componente"
En este cuadro de diálogo, los tipos de los campos se comprueban automáticamente. Si los datos de entrada y el formato del campo definido en este cuadro de diálogo no coinciden, los datos se señalan en rojo.

Nota: los tipos de campo de las columnas se basan en los tipos de datos XML Schema predeterminados. P. ej. el tipo `Date` tiene el formato AAAA-MM-DD.



Archivo de entrada
Seleccione el archivo de texto que desea usar como archivo de origen para este componente.

Nota: si se trata de un componente de texto de **destino**, puede dejar vacío este campo. En este caso, la codificación de archivo predeterminada es UTF-8. Puede definir el tipo y el nombre de cada campo en el panel inferior y hacer clic en **Aceptar** para crear el archivo de texto de destino.

Cuando termine de diseñar la asignación, abra el panel **Resultados** para ver y **guardar** el archivo de texto de destino (haga clic en el icono **Guardar resultado generado**).

Si escribe un nombre de archivo en este cuadro de texto (sin extensión de archivo), el nombre se le asigna al componente.

Archivo de salida
Seleccione el archivo de destino en el que quiere guardar los datos de salida cuando genere código con MapForce. Cuando genere código, compruebe que los archivos de entrada y salida son diferentes. De lo contrario, el archivo de origen se sobrescribirá cuando se ejecute el código. Esto ocurre cuando se genera código para Java, C++ o C# o cuando se guarda el archivo de salida con BUILTIN (motor de ejecución integrado).

Codificación del archivo
Permite definir/seleccionar la codificación de caracteres de los archivos de texto de entrada y salida. Si deja vacío el campo *Archivo de entrada*, la codificación de archivo predeterminada es UTF-8.

 © 2014 Altova GmbH

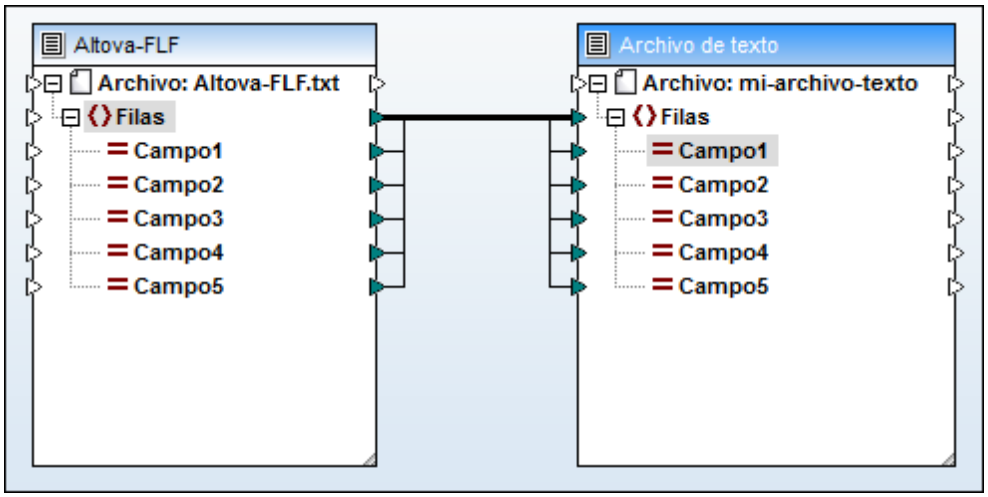
Carácter de relleno
Aquí puede definir los caracteres que se deben usar para completar/rellenar el resto del campo cuando la longitud de los datos entrantes es menor que la longitud definida para el campo. En el cuadro *Personalizado* puede definir un carácter de relleno personal.

Eliminar caracteres de relleno:
Si introduce en el cuadro *Personalizado* el mismo carácter de relleno que incluyen los datos entrantes, los caracteres de relleno se eliminan de los datos entrantes.

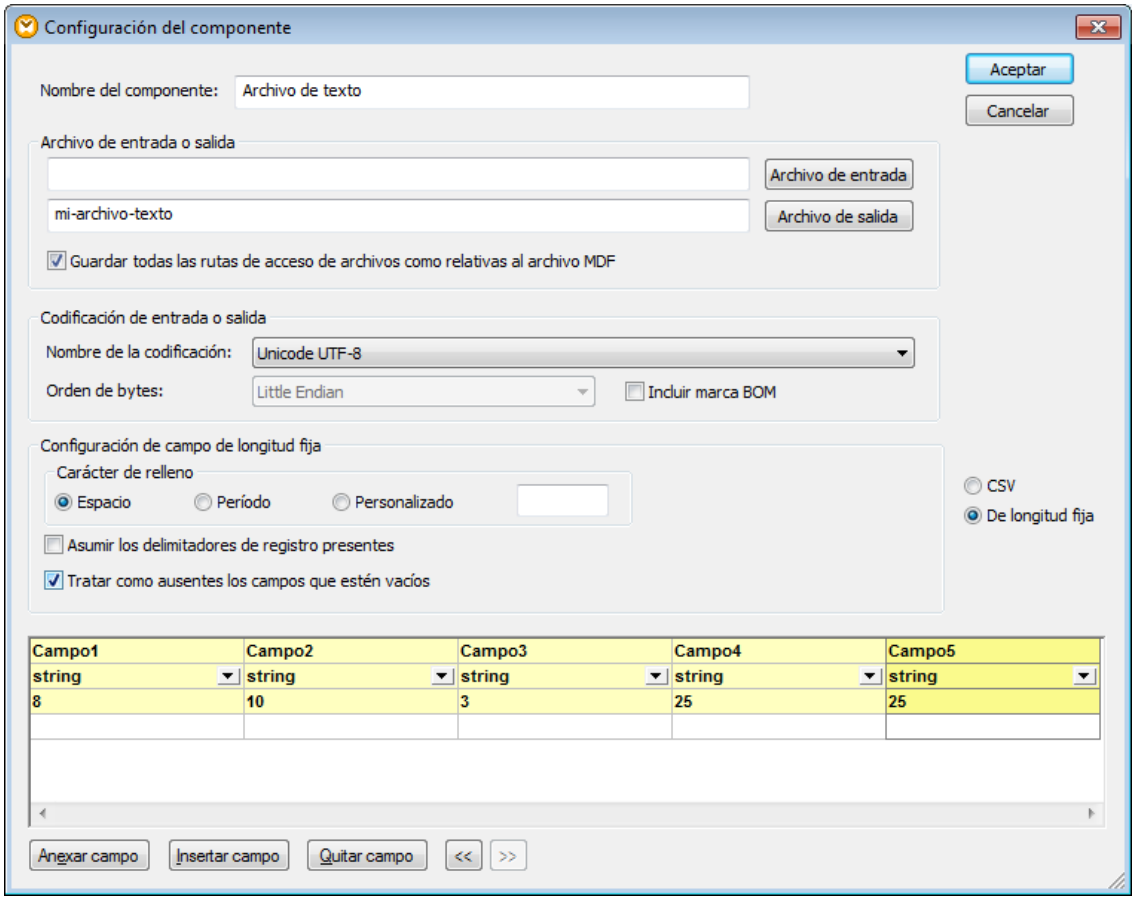
Para definir el carácter de relleno en el cuadro *Personalizado*, haga clic en el cuadro y:

- pulse una tecla del teclado para introducir un valor nuevo o
- haga doble clic en el cuadro *Personalizado* para marcar el valor actual y pulse otra tecla para cambiar el valor de la entrada.

Asumir los delimitadores de registro presentes
Si un archivo de texto de longitud fija (con una sola cadena) es el origen de datos de otro archivo de texto de longitud fija (en una asignación entre dos archivos de texto) y marca esta casilla en el archivo de **destino**, MapForce crea líneas nuevas después de rellenar la última columna del archivo de destino.



En el ejemplo anterior, el archivo de texto `Altova-FLF` se asigna a un archivo de texto de destino vacío (`mi-archivo-texto`).



Notas:

- La entrada *Archivo de entrada* está vacía. Esto significa que este componente de texto solamente recibe datos del componente de origen al que está asignado.
- Se definió una longitud de campo para cada campo.
- En la vista previa no se pueden ver datos porque el componente de destino no está basado en ningún archivo de texto.
- Abra el panel **Resultados** para ver los datos resultantes de la asignación.

Casilla *Asumir los delimitadores de registro presentes*

- casilla **activada**:
  se crea un registro nuevo después de la suma de las longitudes de campo definidas (en este caso, los campos suman 71 caracteres. Para el carácter 72 se creará un registro nuevo).

```
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#
```

- *casilla* ***desactivada***
  los datos asignados aparecen en una sola cadena, que incluye los caracteres de relleno definidos.

 © 2014 Altova GmbH

```
1    Vernon##Callaby###582v.callaby@nanonull.com###Office Manager###########Frank
```

Tratar como ausentes los campos que estén vacíos
Esta acción permite definir que los campos vacíos del archivo de origen no produzcan un elemento vacío (elemento/atributo) en el archivo de destino.

Si está **activada**, los elementos de destino que no reciben datos del archivo de origen no aparecen en el archivo de resultados.

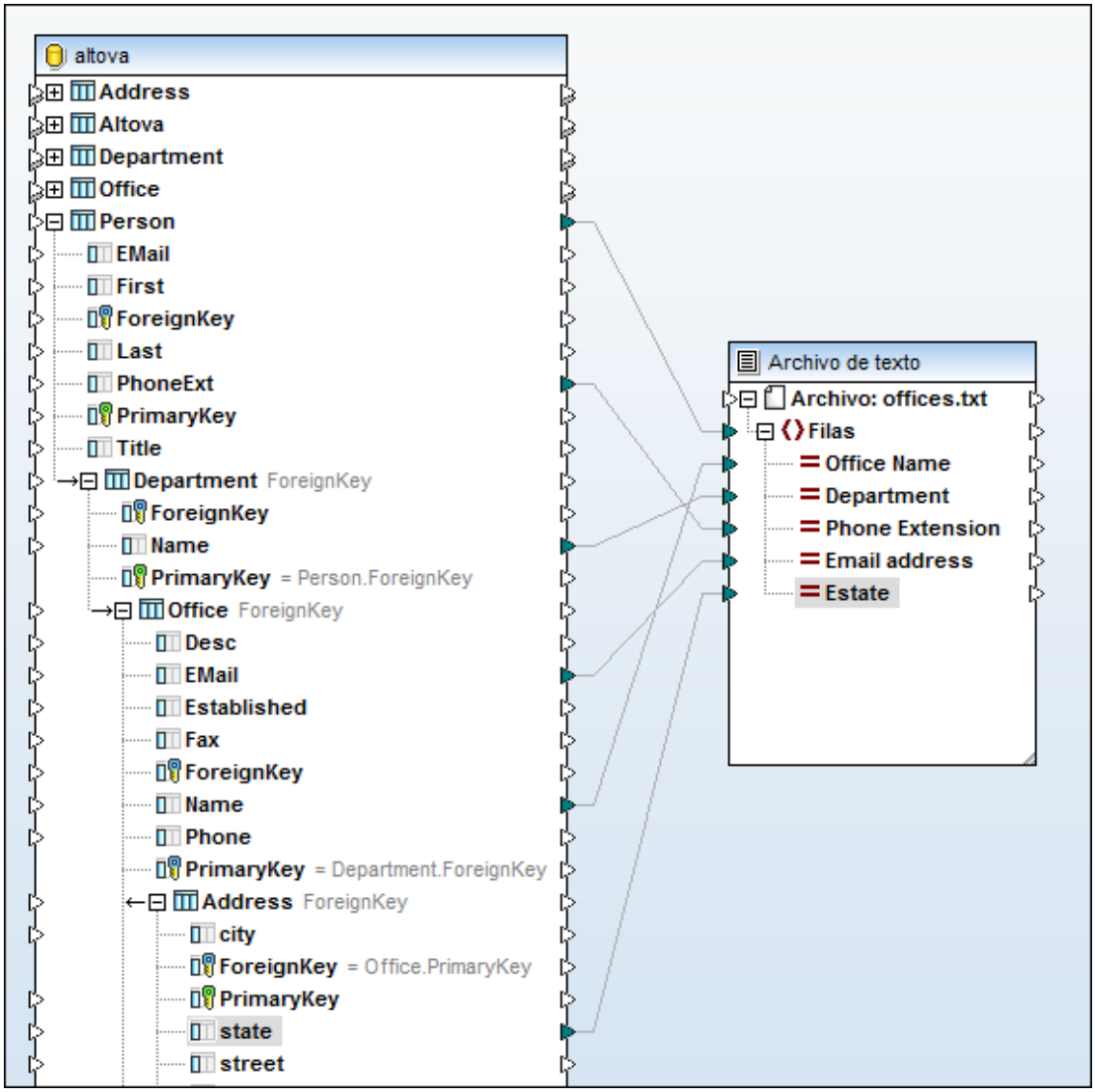
Si está **desactivada**, los campos vacíos del archivo de origen producen elementos de destino en el archivo de resultados.

Anexar campo, **Insertar campo**, **Quitar campo**
Permite anexar, insertar o quitar campos de la ventana de vista previa, que define la estructura del componente de longitud fija.

Botones Izquierda / **Derecha**
Haga clic en estos botones para mover a la izquierda/a la derecha la columna que está activa en la vista previa.

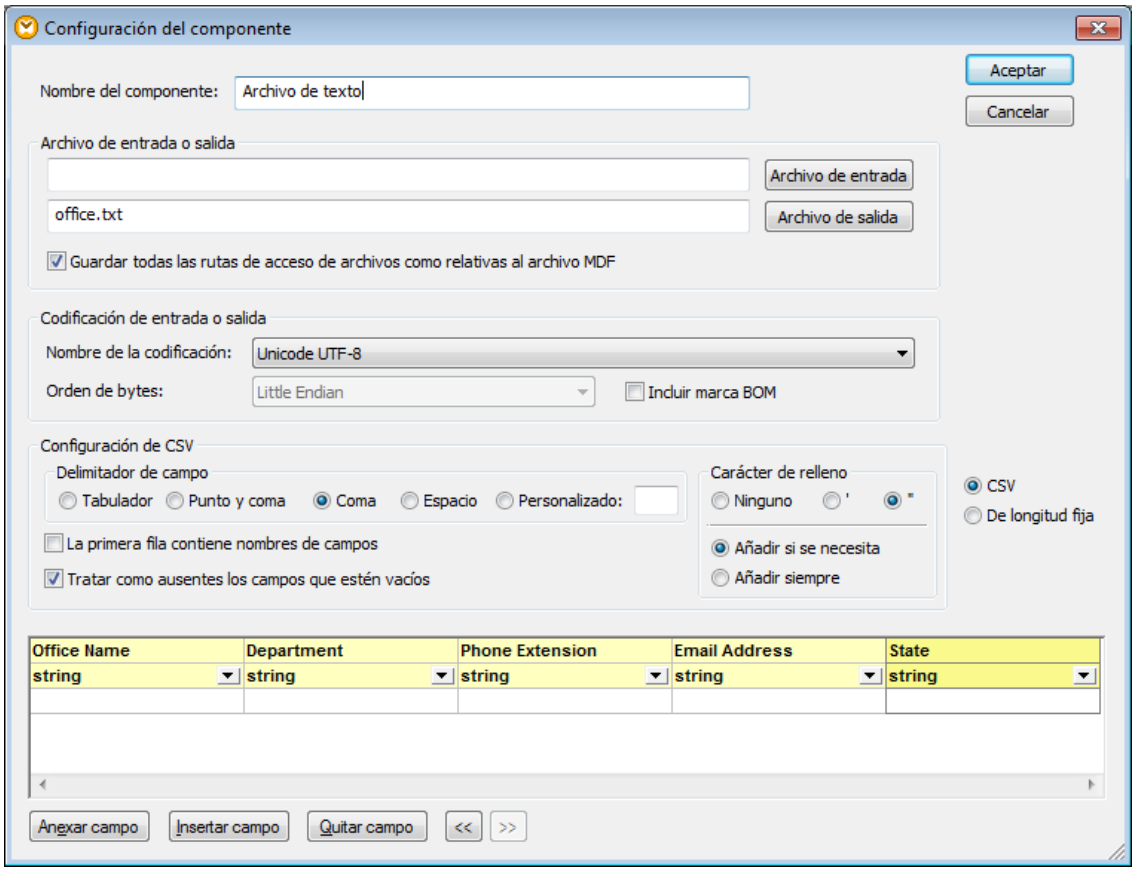
10.3.6 Asignar datos de BD a archivos de texto/CSV

Este apartado muestra una asignación entre una sencilla base de datos MS Access (`altova.mdb`) y un archivo CSV. La base de datos `altova.mdb` está disponible en la carpeta `...\MapForceExamples\Tutorial`.

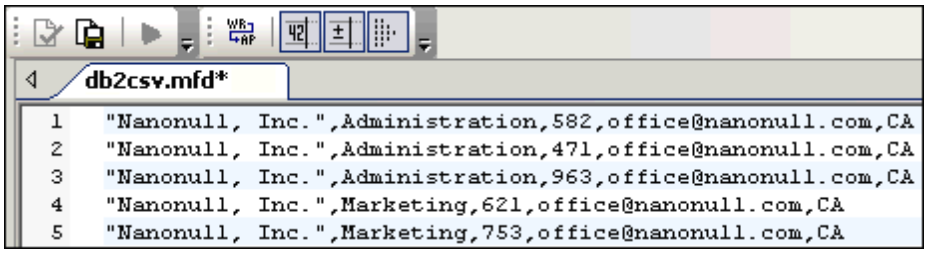


 © 2014 Altova GmbH

El archivo `Office.txt` (que se introdujo en el campo *Archivo de salida*) es el mismo que se suministra automáticamente al hacer clic en el icono **Guardar resultado generado** del panel **Resultados**.



Haga clic en el icono **Guardar resultado generado** para generar el archivo de texto de salida.

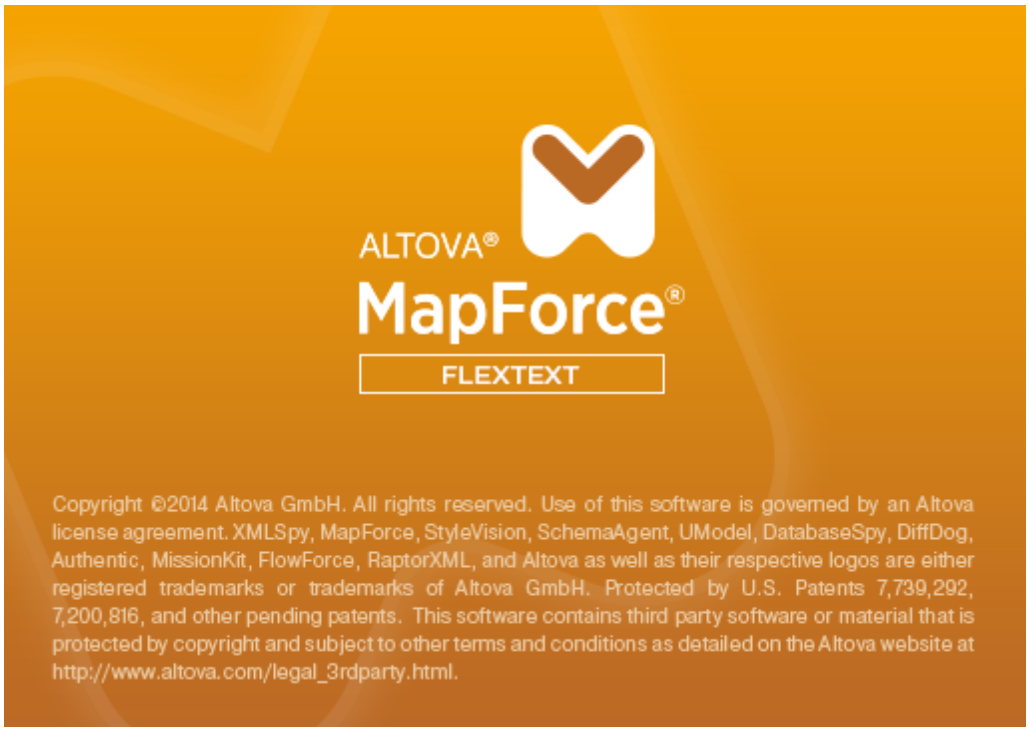


10.4 MapForce FlexText

Con el módulo FlexText de MapForce 2014 podrá procesar y crear asignaciones de datos para archivos de texto heredados.

Las plantillas de FlexText, que definen la estructura y el contenido de los archivos, se crean en el módulo FlexText y se insertan como componente dentro de la asignación de MapForce, donde puede decidir qué elementos/secciones se asignan a los componentes de destino.

Los componentes de destino pueden ser archivos de texto, XML, bases de datos o EDI.



Última actualización: 17/03/2014

10.4.1 Introducción

Sitio web de Altova: Asignación de archivos de texto complejos con FlexText

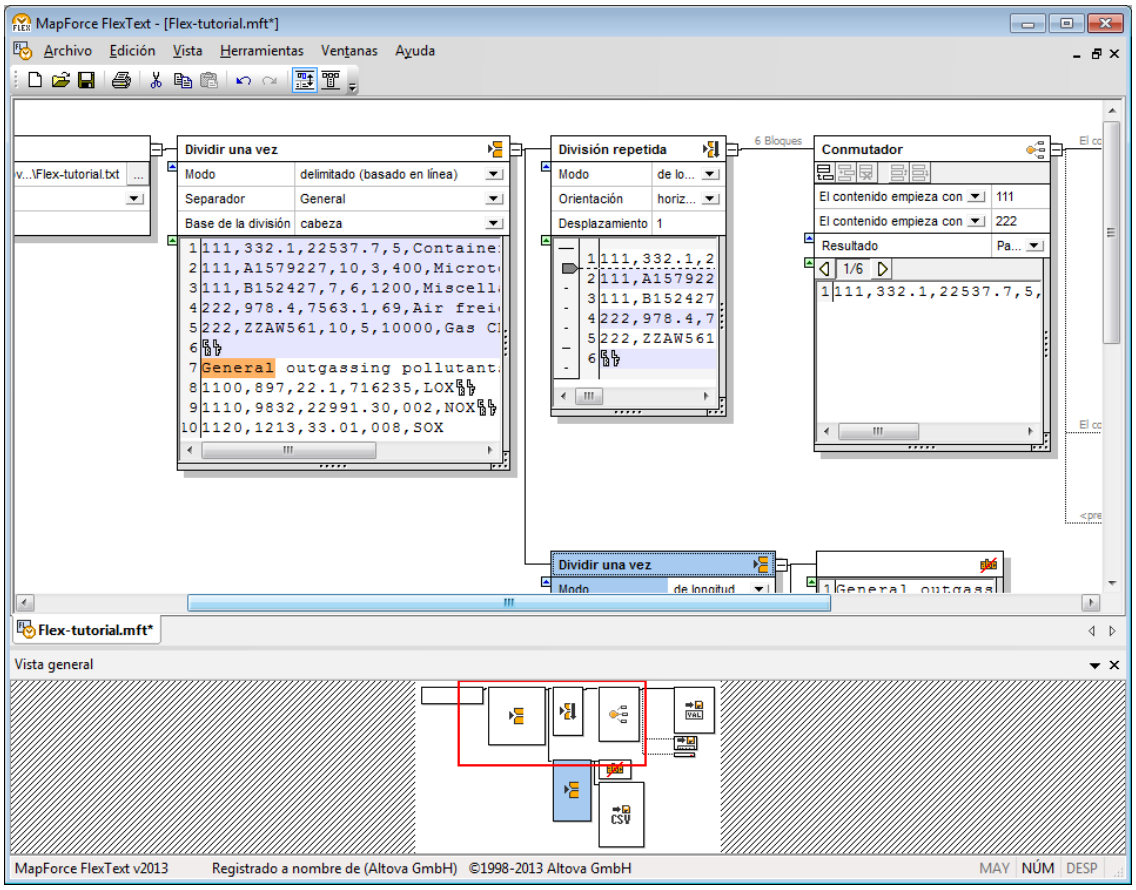
FlexText ofrece una interfaz gráfica para preparar archivos planos de gran complejidad (con varios tipos de delimitadores, estructuras anidadas, etc.) para poder usarlos en MapForce y generar código.

Las plantillas creadas en FlexText se cargan en MapForce, donde puede crear las asignaciones de datos. Las plantillas funcionan con archivos de texto suministrados/abiertos en MapForce y se pueden volver a utilizar una y otra vez con otros archivos de texto y en otras asignaciones.

En FlexText puede definir la estructura del archivo plano de forma interactiva y ver inmediatamente los resultados en un panel de texto de muestra.

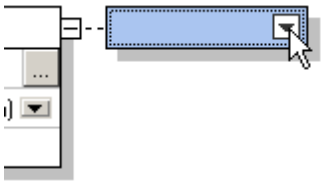
La interfaz gráfica de FlexText incluye tres paneles: **diseño**, **vista general** y **texto de muestra**.

- En el **panel de diseño** se pueden insertar contenedores de fragmentos de texto, cuyo nombre indica la función que desempeñan en la plantilla (p. ej. **División repetida**).
- Los **paneles de texto de muestra** ofrece una vista previa del contenido del contenedor activo.
- El **panel de vista general** muestra una visión de conjunto de todos los contenedores del panel de diseño. El rectángulo rojo indica la posición actual.

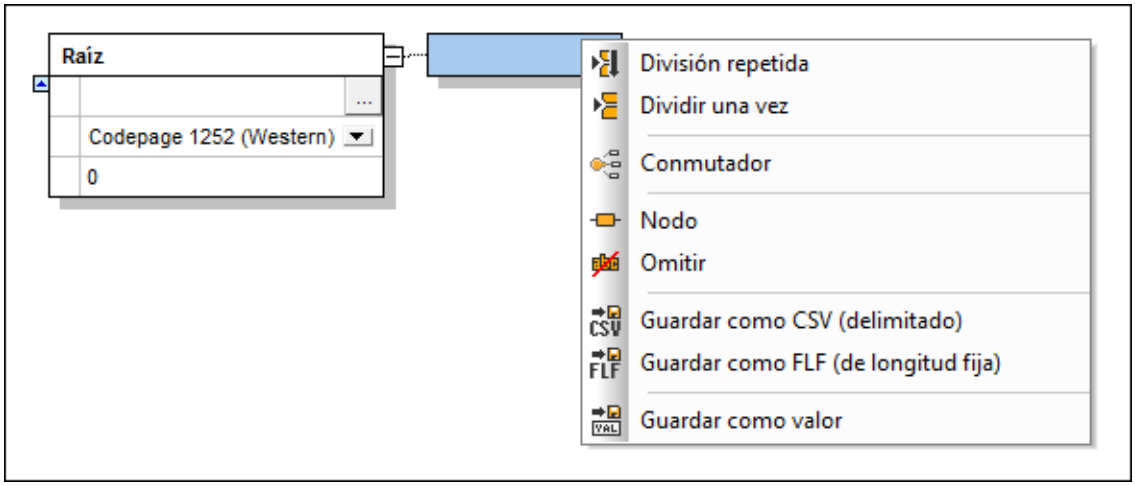


Cada contenedor incluye un icono interactivo que sirve para definir el tipo de contenedor y su

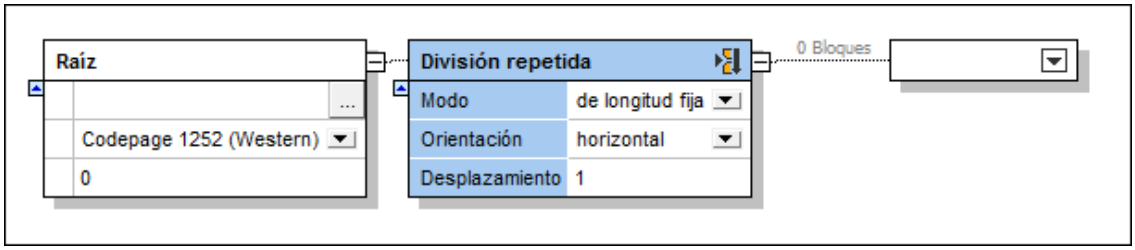
contenido.



Al hacer clic en el icono del contenedor aparece un menú contextual donde puede seleccionar el tipo de contenedor.



La opción seleccionada define la función y el contenido del contenedor y habilitan más funciones con las que podrá perfeccionar el contenido que se debe enviar después al componente de destino en MapForce.



Al seleccionar una opción del menú contextual:

- cambia el aspecto, el tipo y el icono del contenedor (p. ej. **División repetida** aparece en la barra de título del contenedor).
- se habilitan las opciones predeterminadas en el contenedor (p. ej. *modo*=`longitud fija`, *orientación*=`horizontal`, *desplazamiento*=`1`).
- se anexa automáticamente un contenedor nuevo al contenedor actual.
- los iconos ▣ / ▣ permiten ocultar/mostrar los compartimentos de los contenedores.
- la tecla **Mayús** permite contraer los contenedores en grupo. Al pulsar esta tecla aparece un icono con dos flechas (▣). Al hacer clic en el icono se contrae la sección del árbol de contenedores situada a la derecha del contenedor donde se hizo clic.

El icono **Texto del nodo en vista Diseño** activa el panel de texto de muestra, que incluye el contenido del contenedor activo.

El icono **Contraer automáticamente el texto de nodos inactivos** muestra el contenido

del contenedor activo y oculta el contenido de los demás contenedores.

10.4.2 Tutorial de MapForce FlexText

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.

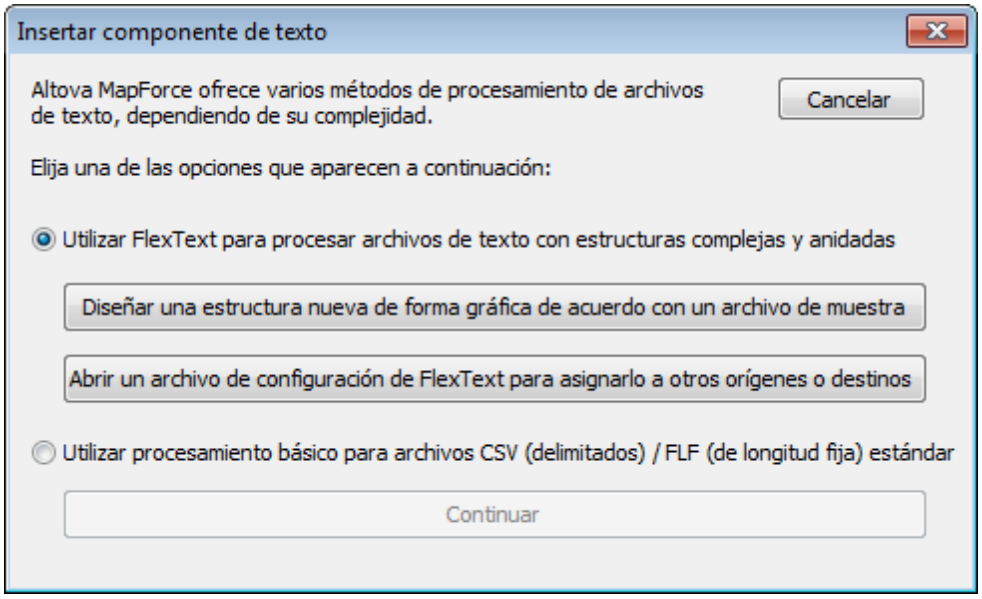
Para este ejemplo usamos el archivo `Flex-tutorial.txt`, disponible en la carpeta `...\MapForceExamples\Tutorial`, que tiene este formato:

```
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:

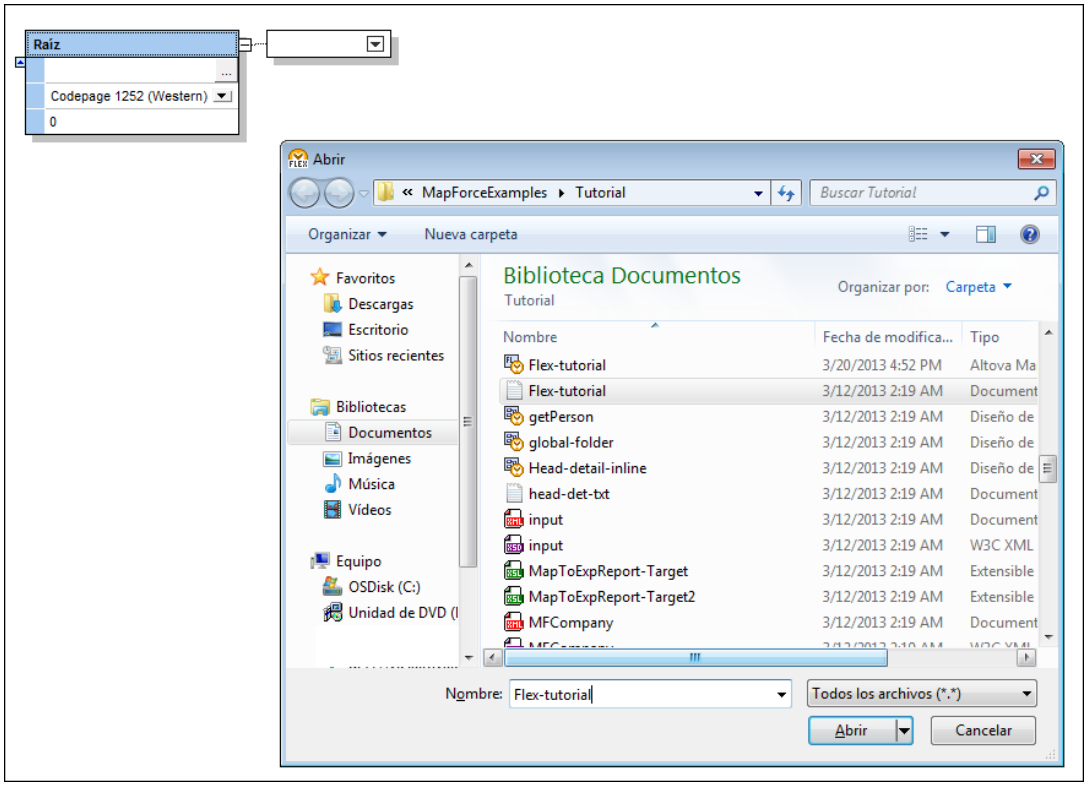
- separar los registros que contienen la clave 111 y 222 de los demás registros y guardarlos en elementos asignables.
- descartar los demás registros.
- crear un archivo CSV con los registros sobrantes.

1. Inicie MapForce y abra un archivo de asignación nuevo.
2. Seleccione la opción de menú **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...**.

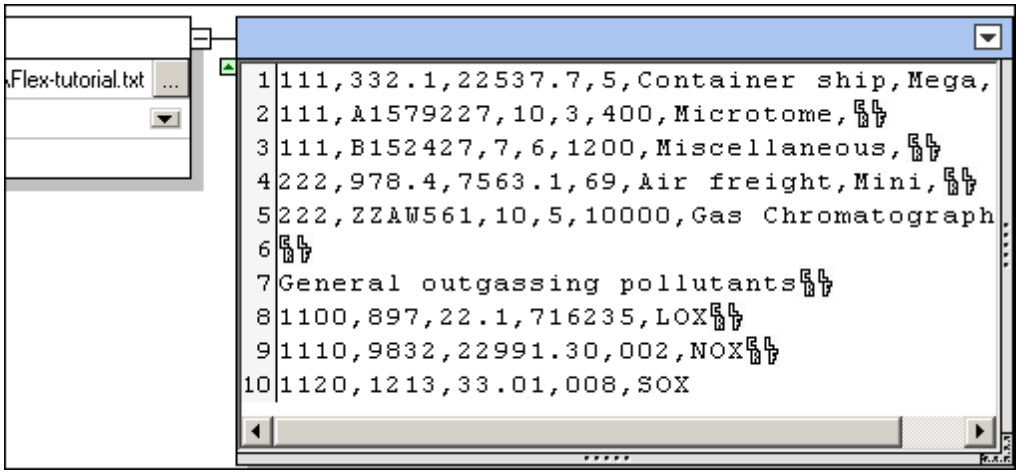


4. Introduzca un nombre para su plantilla FlexText (p.ej. `Flex-tutorial.mft`) y haga clic en **Guardar** para continuar.
   Aparece un diseño vacío en FlexText y el cuadro de diálogo "Abrir".

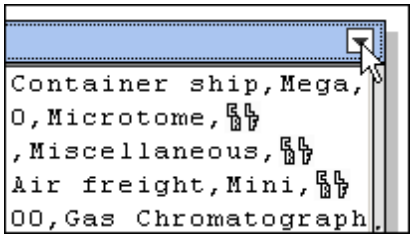
 © 2014 Altova GmbH



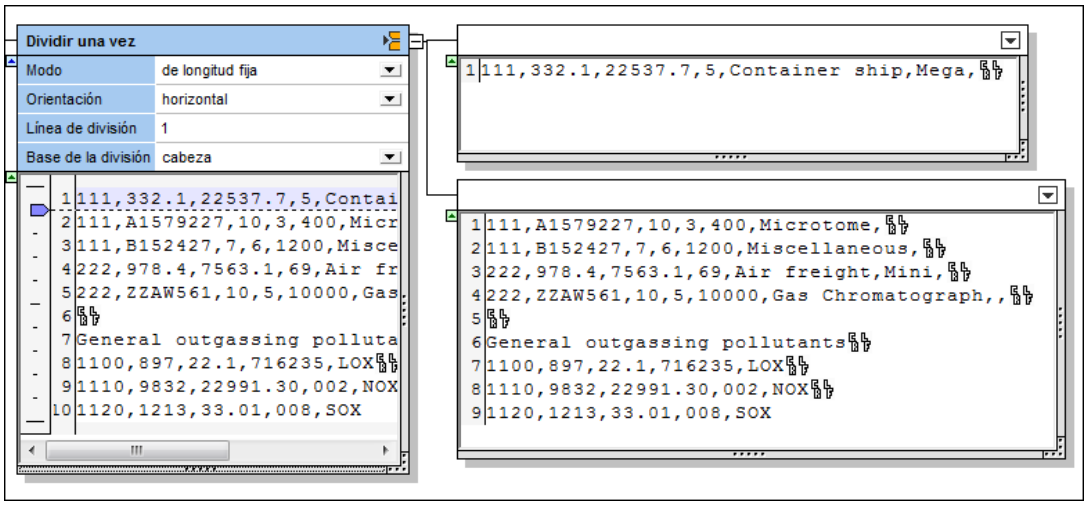
5. Seleccione el archivo de texto `Flex-tutorial.txt` de la carpeta `...\MapForceExamples\Tutorial` y haga clic en **Abrir**.



El panel de texto de muestra ofrece el contenido del archivo. Compruebe que está activo el icono **Texto del nodo en vista Diseño**.



6. Haga clic en el icono situado en la esquina superior derecha del contenedor. En el menú contextual seleccione **Dividir una vez**.
   Aparecen dos contenedores nuevos junto al contenedor llamado **Dividir una vez** (para más información consulte la sección Dividir una vez del tutorial).



   El contenedor **Dividir una vez** aparece con su configuración predeterminada: `longitud fija, orientación horizontal` y `línea de división=1`.

Como resultado de esta configuración:

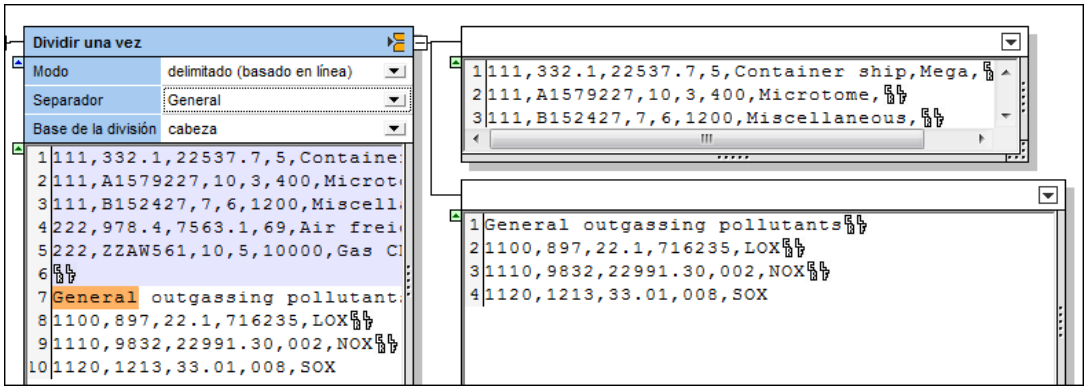
- el contenedor de la parte superior incluye la primera línea del archivo de texto (que está resaltada en el contenedor **Dividir una vez**).
- el contenedor de la parte inferior contiene el resto del archivo de texto.

 © 2014 Altova GmbH

10.4.3 Crear condiciones de división

En FlexText puede definir lo que llamamos *condiciones de división* para segmentar fragmentos de texto de varias formas.

1. Haga clic en el cuadro combinado *Modo* y seleccione la opción `delimitado (basado en línea)`.
2. Haga doble clic en el campo *Separador* y escriba `General`.

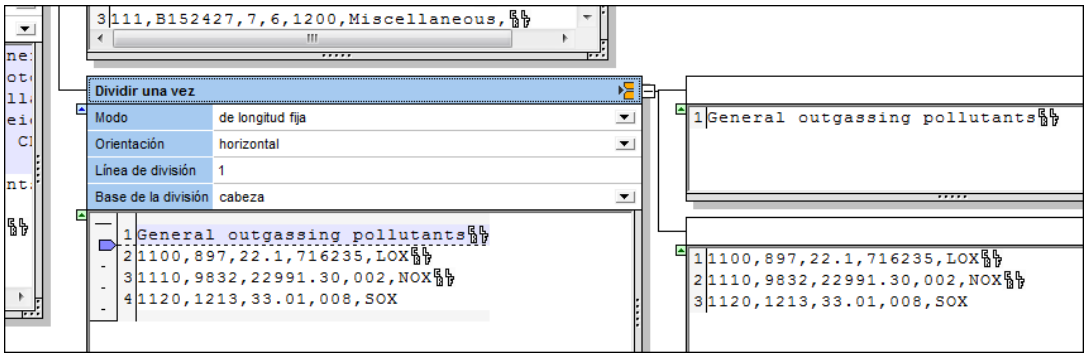


Observe cómo cambian los fragmentos de texto de los contenedores.

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.

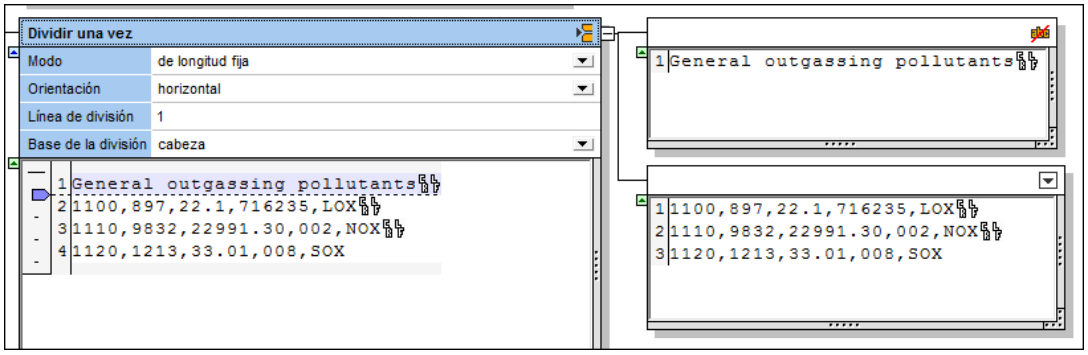
A continuación prepararemos el contenedor inferior para producir un archivo CSV que contenga los registros a partir de `1100`.

3. Haga clic en el icono interactivo del contenedor inferior y seleccione **Dividir una vez**.



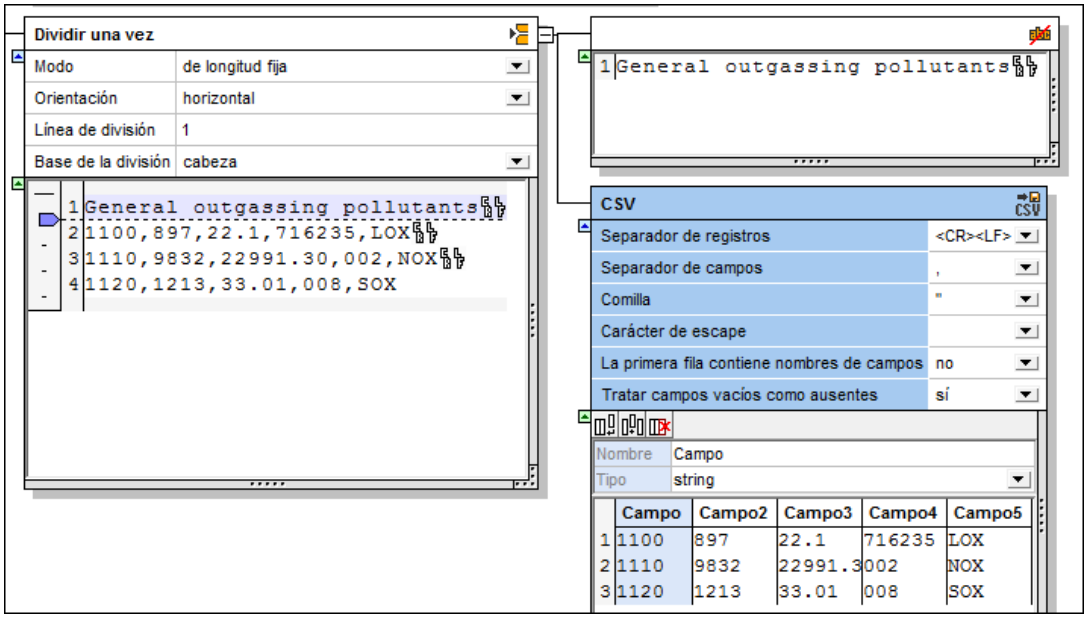
La configuración predeterminada de este nuevo contenedor **Dividir una vez** crea dos contenedores más a su derecha y separa la primera línea del fragmento de texto del resto. Nuestro objetivo es omitir esa primera línea y guardar el resto en un archivo CSV.

4. Haga clic en el 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 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.

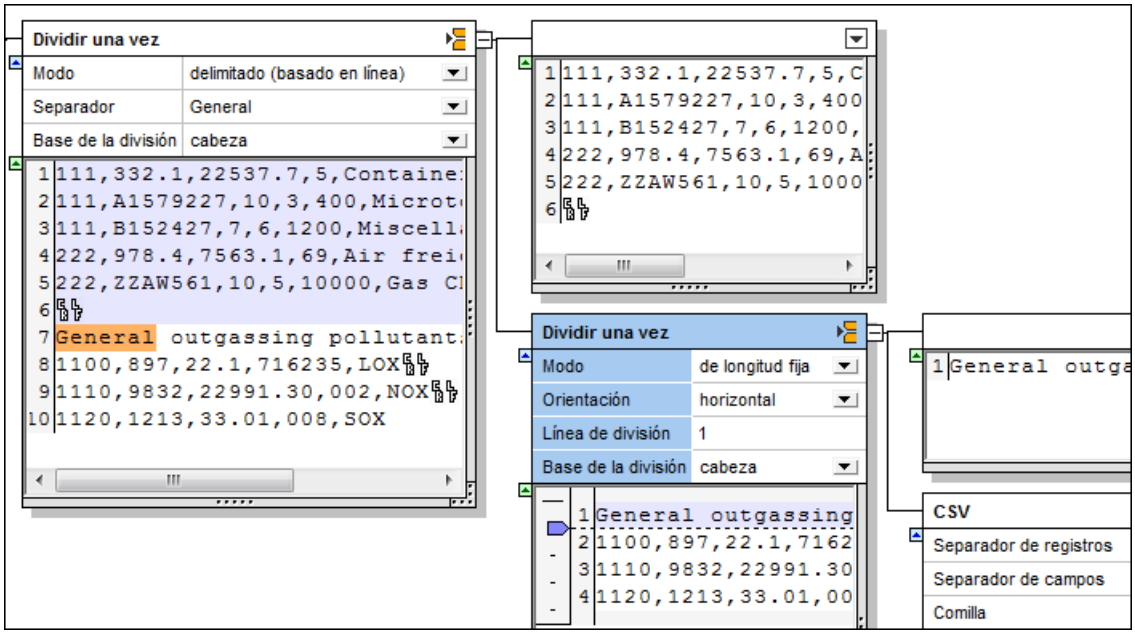
Ahora seguiremos definiendo el fragmento de texto sobrante.

10.4.4 Definir varias condiciones por contenedor/fragmento

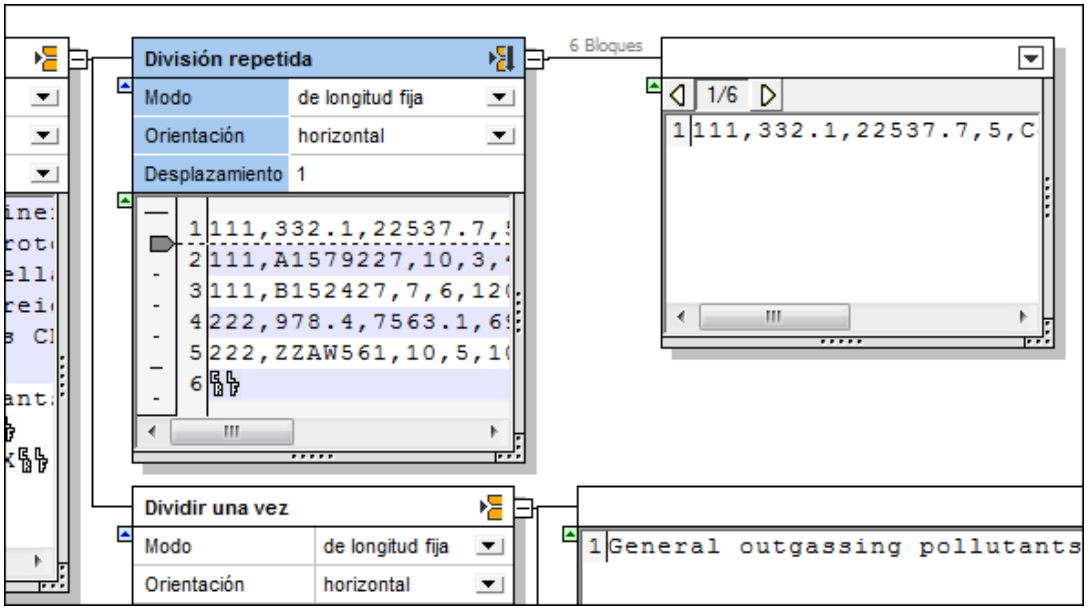
En FlexText puede definir varias condiciones por cada fragmento de texto, usando el contenedor **Conmutador**. A cada condición definida se le asigna automáticamente un contenedor asociado.

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**)

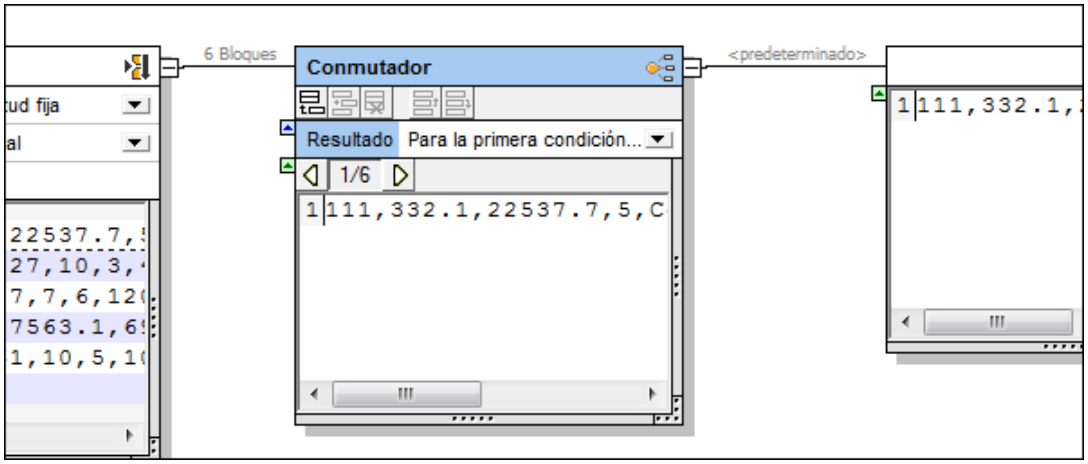


1. Haga clic en el icono del contenedor superior (el que todavía no tiene nombre) y seleccione la opción **División repetida**.



Para el tutorial 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 este nuevo contenedor y seleccione la opción **Conmutador**.

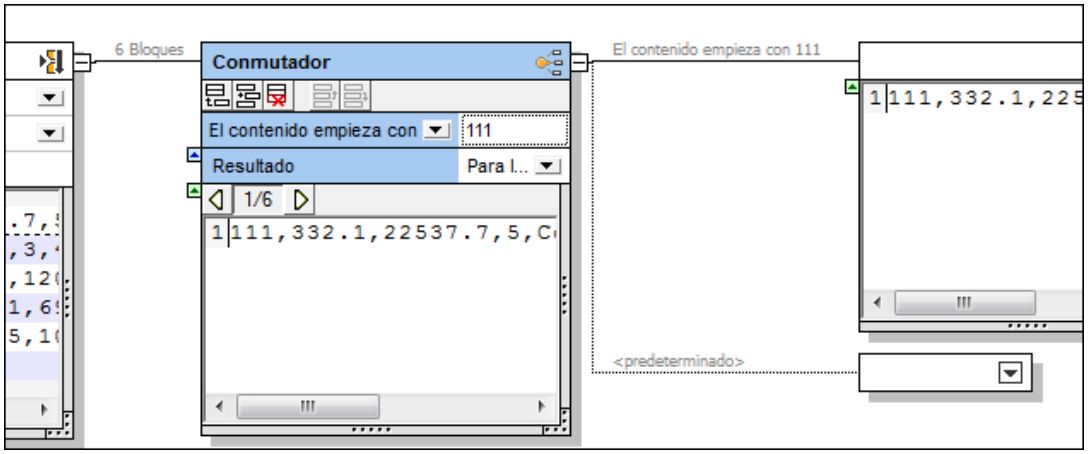


El estado inicial del contenedor **Conmutador** es:

- 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.

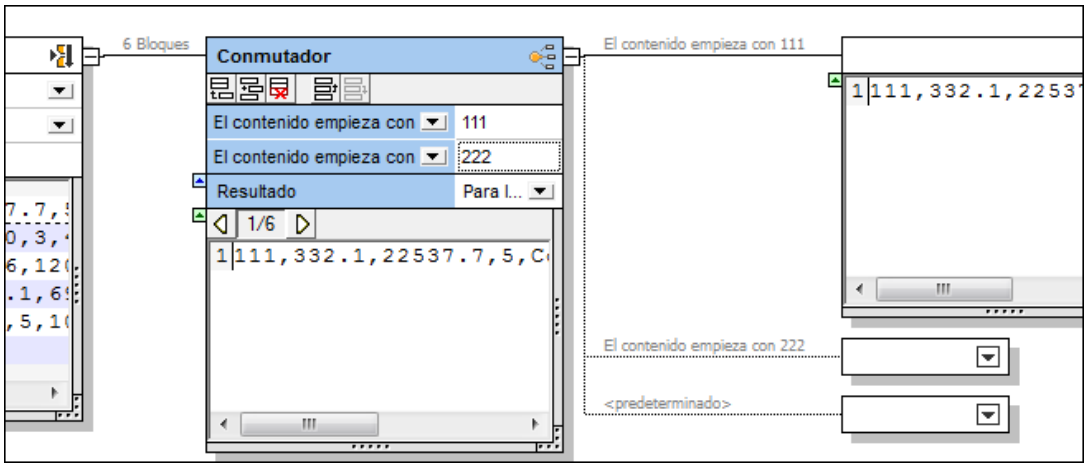
3. Haga clic en el icono **Anexar condición** de la barra de título del contenedor **Conmutador** para añadir una condición nueva.
4. Haga doble clic en el campo *El contenido empieza con* y escriba el número 111.

 © 2014 Altova GmbH



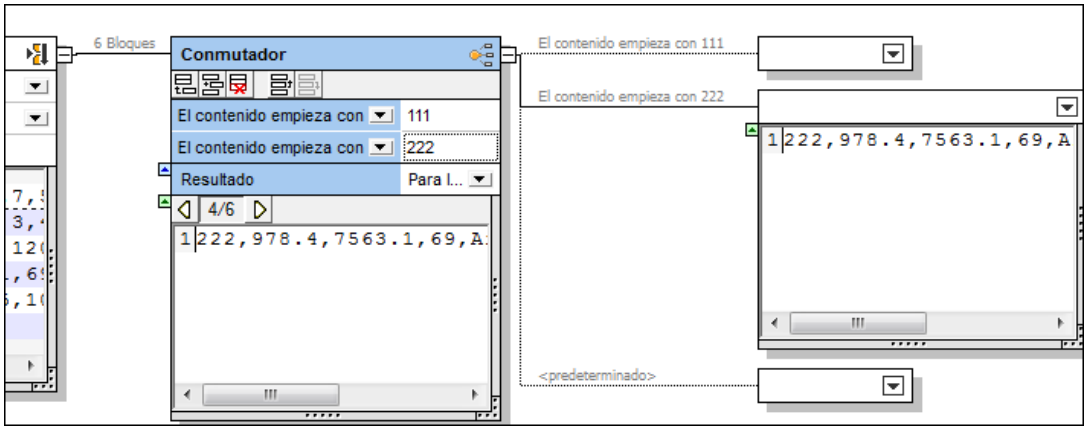
Esto define la primera condición y se añade un contenedor asociado (*El contenido empieza con 111*) encima del contenedor *predeterminado*.

5. Haga clic otra vez en el icono **Anexar condición** y escriba 222 en el campo *El contenido empieza con*.



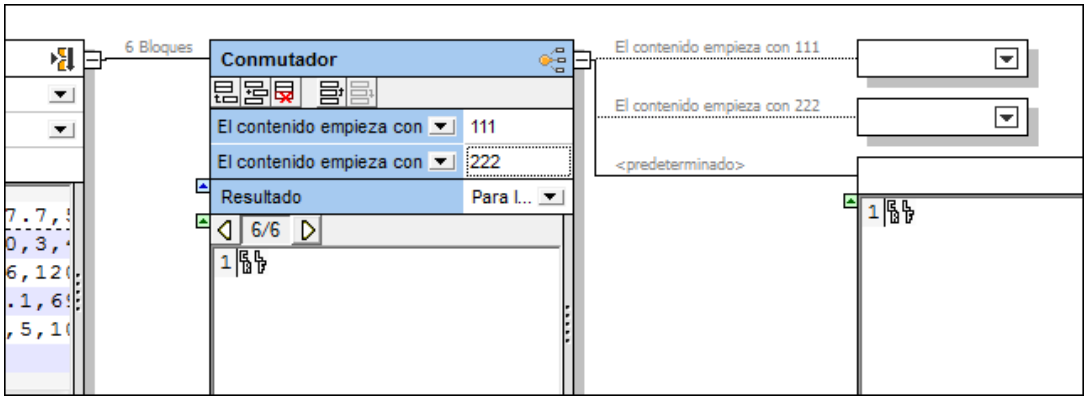
Se añade un tercer contenedor (*El contenido empieza con 222*).
Nota: si hace clic en el cuadro combinado *El contenido empieza con* también puede seleccionar la opción *Contiene* para especificar una cadena que puede aparecer en cualquier parte del fragmento de texto.

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*.

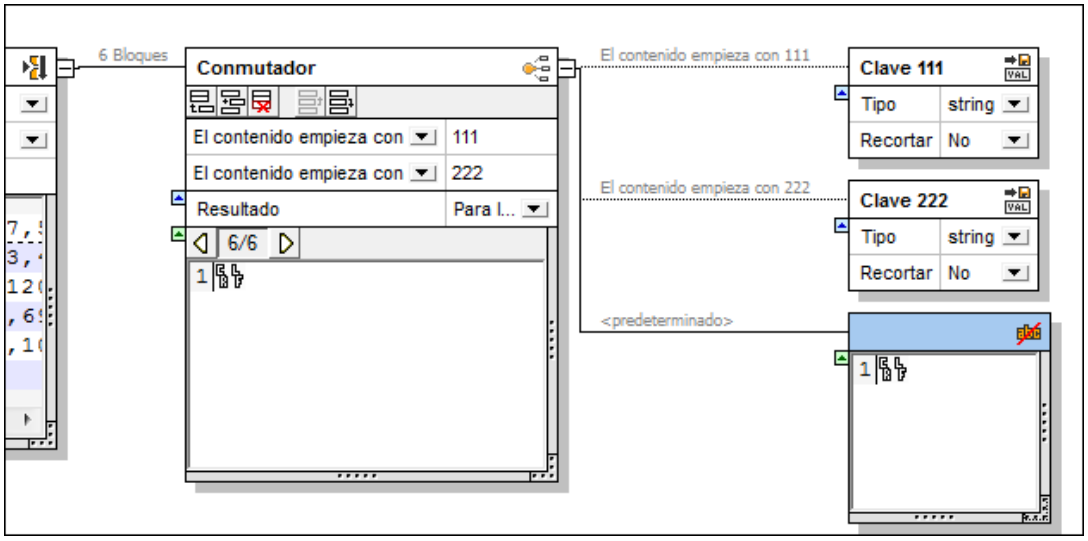
7. Siga haciendo clic hasta llegar al registro 6 de 6. En el contenedor *predeterminado* aparece solamente el carácter CR / LF.



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**. Haga doble clic en el título de estos contenedores y llámelos Clave 111 y Clave 222 (para reconocerlos más fácilmente).
9. Haga clic en el último contenedor (*predeterminado*) y cambie el tipo por **Omitir**.

 © 2014 Altova GmbH

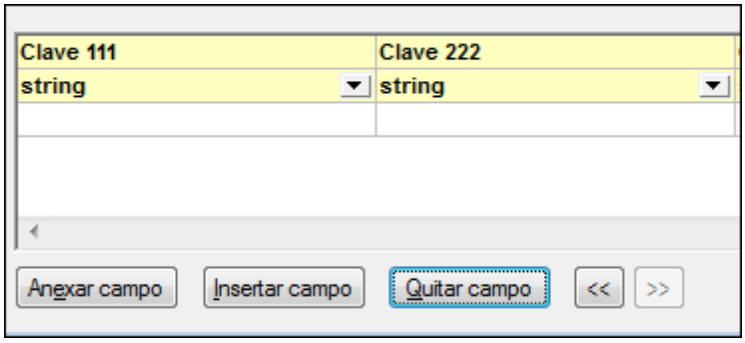


10. Guarde la plantilla de FlexText (p. ej. `Flex-Tutorial.mft`)

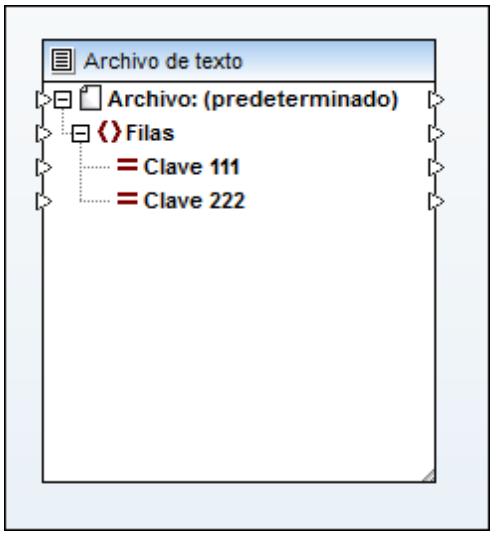
10.4.5 Usar plantillas de FlexText en MapForce

Crear los 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**.
   Se 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**.

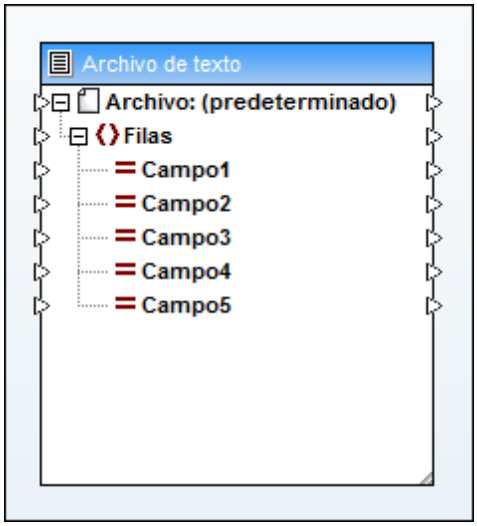


5. Haga lo mismo en el otro campo y póngale el nombre **Clave 222**. Para terminar haga clic en **Aceptar**.
   Se crea un componente de texto con dos campos.



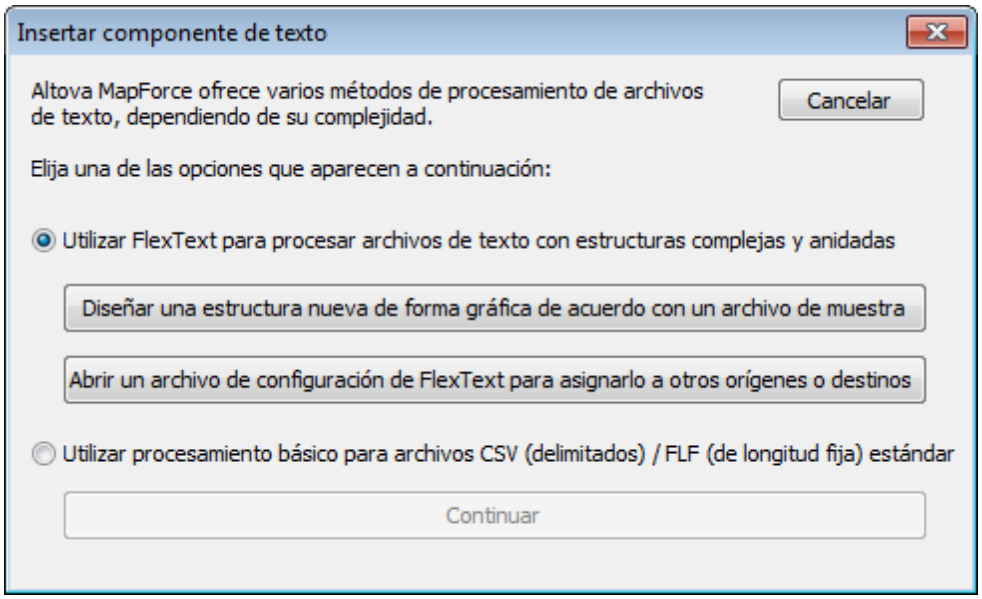
6. Utilice el mismo método para crear otro componente de texto que tenga cinco campos llamados **`Campo1`**, **`Campo2`**, etc.

 © 2014 Altova GmbH

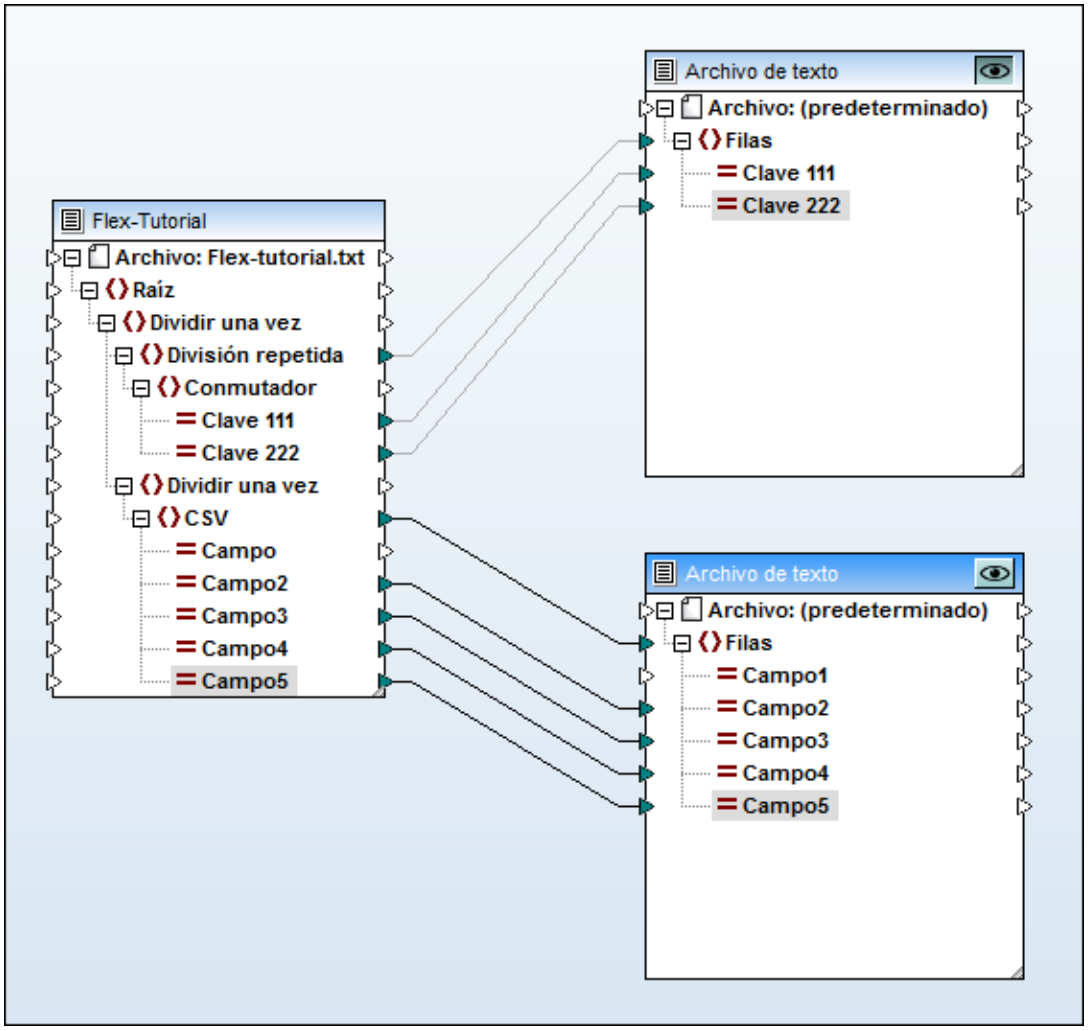


Usar la plantilla FlexText en MapForce:

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:

```
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
```

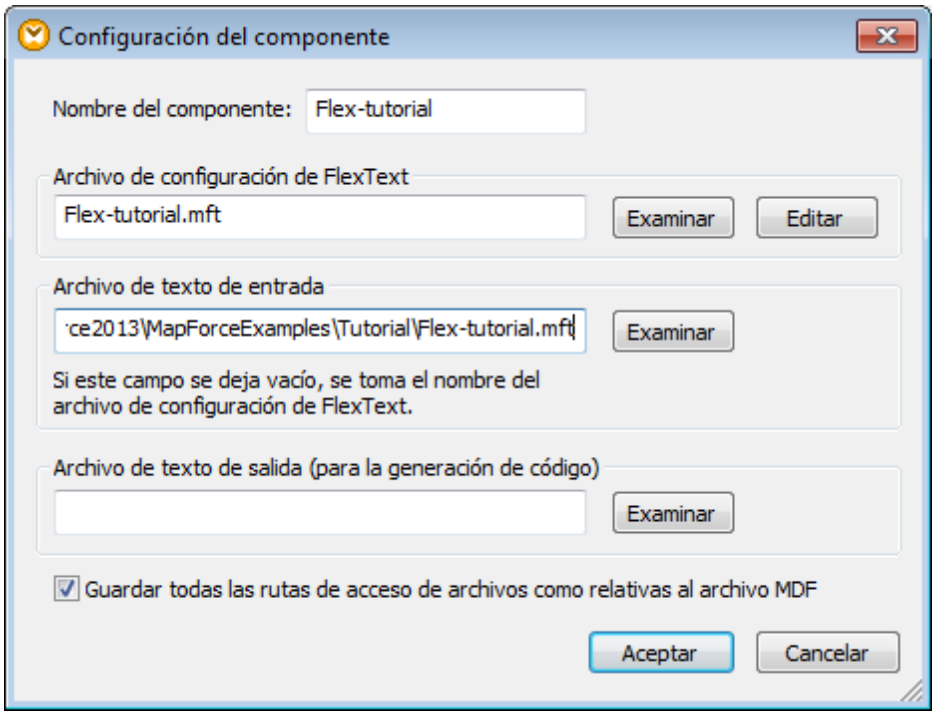
Vista previa del componente de texto inferior:

```
1  1100,897,22.1,716235,LOX
2  1110,9832,22991.30,002,NOX
3  1120,1213,33.01,008,SOX
4
```

Orígenes de datos en FlexText

Al hacer doble clic en el componente FlexText se abre el cuadro de diálogo "Configuración del componente", donde puede definir los archivos de entrada y de salida.

El campo *Archivo de configuración de FlexText* contiene el origen de datos definido cuando se creó el componente.



Si existe una entrada en el campo *Archivo de texto de entrada*, este es el origen de datos que se utiliza en lugar del archivo del campo *Archivo de configuración de FlexText*.

La entrada del campo *Archivo de texto de salida* se utiliza para el nombre de archivo que se crea durante la generación de código. También puede introducir una ruta de acceso completa para definir un directorio de destino concreto (p. ej. `C:\MisArchivos\sequence.txt`.)

10.4.6 Usar FlexText como componente de destino

La ventaja principal de los componentes FlexText es que dividen archivos de texto en fragmentos y permiten asignar los elementos del componente FlexText 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".

Componentes FlexText como componentes de destino
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.

Notas:

- 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

 © 2014 Altova GmbH

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.

10.4.7 Referencia del usuario de FlexText

En esta sección se describen con más detalle las características y funciones de FlexText, explicando cómo usarlas para conseguir determinados resultados.

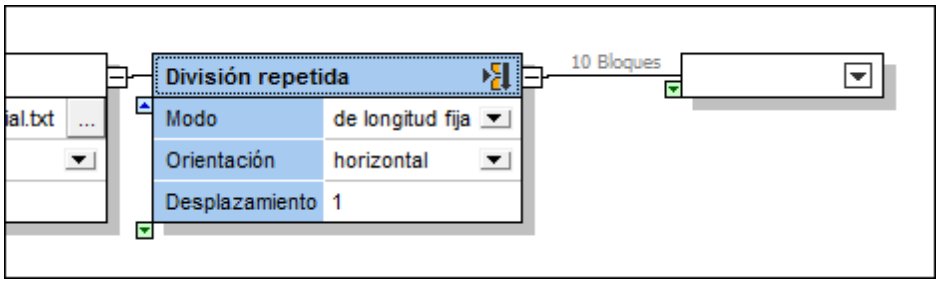
División repetida

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).

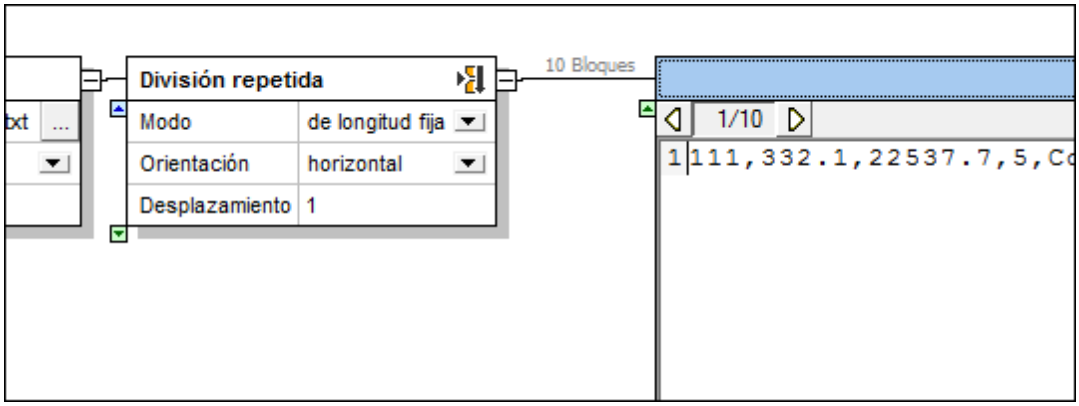
Cuando se selecciona la opción **División repetida** por primera vez, se definen automáticamente parámetros predeterminados y los fragmentos resultantes aparecen en el contenedor asociado. Observe que el contenedor **División repetida** está activo en ese momento y que el panel de texto de muestra ofrece 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 de texto de muestra puede recorrer los 11 bloques/ fragmentos generados con esta configuración.



 © 2014 Altova GmbH

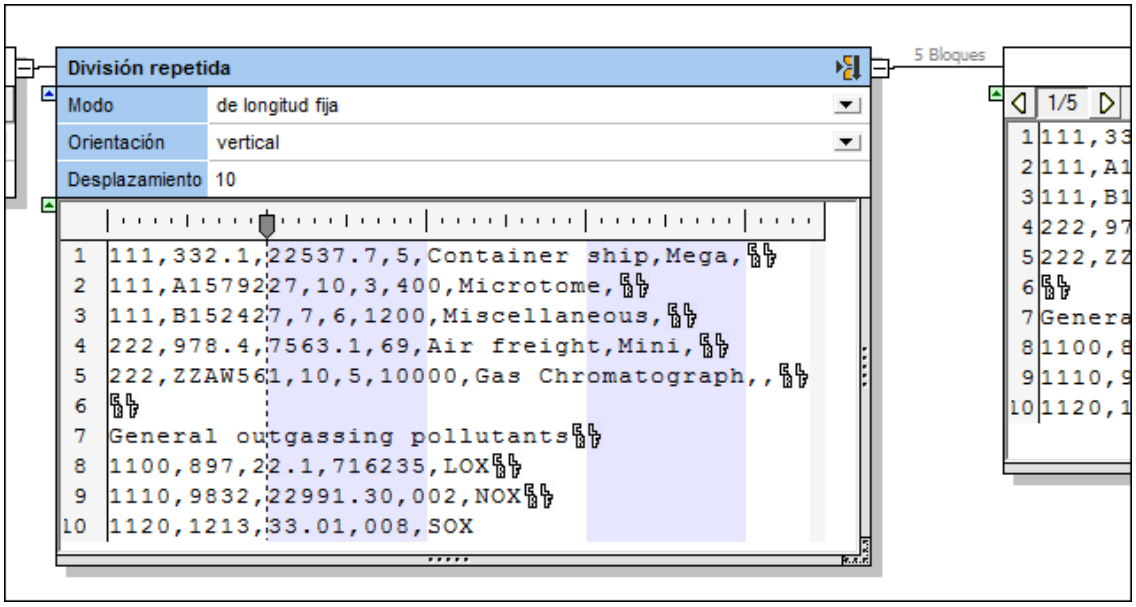
Modo: longitud fija

Este es el modo predeterminado de la opción **División repetida**.

Orientación:
La orientación permite definir cómo se divide el fragmento de texto: en líneas/registros o en columnas.

Seleccione la opción **`Horizontal`** para dividir el fragmento en **varias** secciones horizontales. Introduzca un valor en el campo *Desplazamiento* o arrastre la tabulación de la regla vertical.

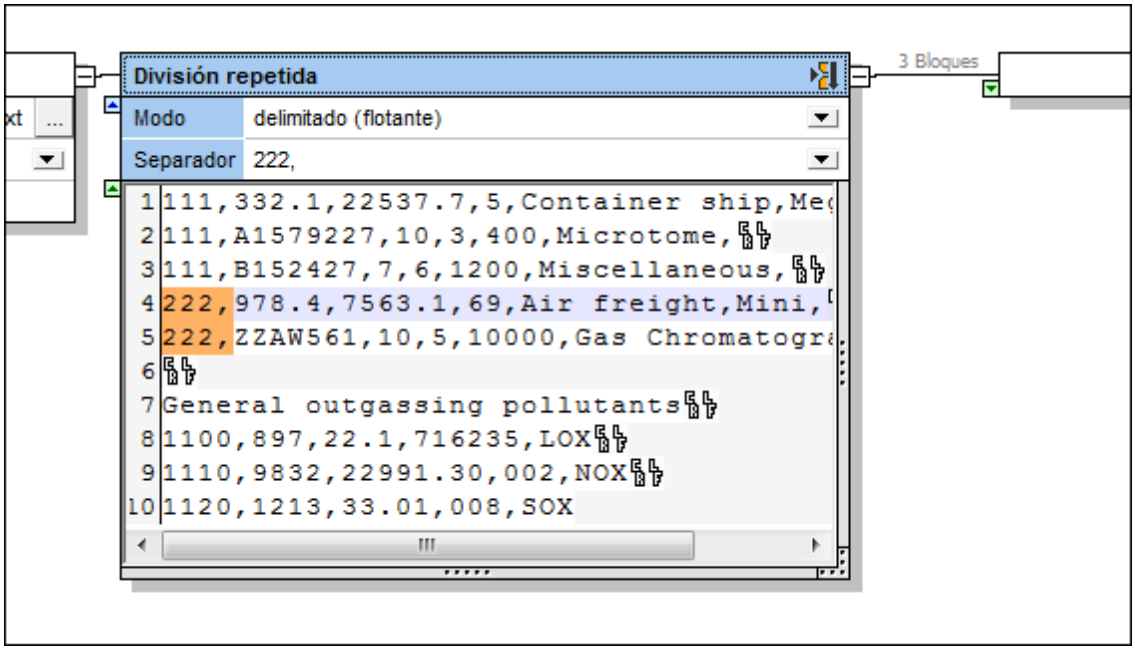
Seleccione la opción **`Vertical`** para dividir el fragmento en varias columnas verticales. Introduzca un valor en el campo *Desplazamiento* o arrastre la tabulación de la regla horizontal (*imagen siguiente*). Cada fragmento contiene los caracteres definidos por el ancho de desplazamiento (p. ej. 10).



Modo: delimitado (flotante)

Configuración predeterminada del modo **delimitado (flotante)**:

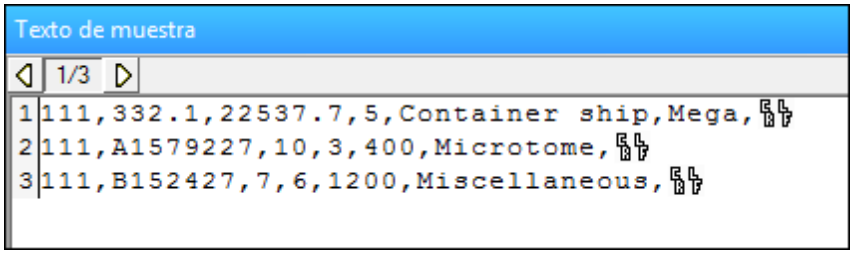
Separador	<vacío>



- Este modo de división repetida crea **varios** fragmentos, definidos por los caracteres separadores especificados en el campo *Separador*.
- Los caracteres separadores **no se incluyen** en el fragmento de texto.
- Se denomina *bloque/fragmento* el texto comprendido entre el primer carácter situado **después** del separador y el carácter situado **antes** de la siguiente aparación del mismo separador (*excepto en el caso del primer y el último fragmento, ver más abajo*).

Por ejemplo, si en el ejemplo de la imagen anterior usamos el separador **222**, se obtienen estos 3 fragmentos:

El **fragmento nº1** contiene todos los caracteres comprendidos entre el **principio** del fragmento y el principio del **primer separador 222**:



Si el separador no es el primer conjunto de caracteres de la primera línea del fragmento (como en este ejemplo), el primer fragmento incluye todo el texto **hasta** la primera aparición del separador.

Si en este ejemplo el primer separador fuera **111** en lugar de 222, el primer fragmento sería una cadena de longitud cero porque el separador aparece al principio de la primera línea del fragmento de origen.

 © 2014 Altova GmbH

El **fragmento nº2** contiene la **primera** línea que incluye el separador 222 (pero sin el separador).

Texto de muestra

2/3

```
1 978.4,7563.1,69,Air freight,Mini,
```

El **fragmento nº3** contiene la **siguiente** línea que incluye el separador 222 (pero sin el separador), hasta el final del archivo de texto.

Texto de muestra

3/3

```
1 ZZAW561,10,5,10000,Gas Chromatograph,,
2
3 General outgassing pollutants
4 1100,897,22.1,716235,LOX
5 1110,9832,22991.30,002,NOX
6 1120,1213,33.01,008,SOX
```

Utilice el modo **delimitado (flotante)** cuando necesite procesar:

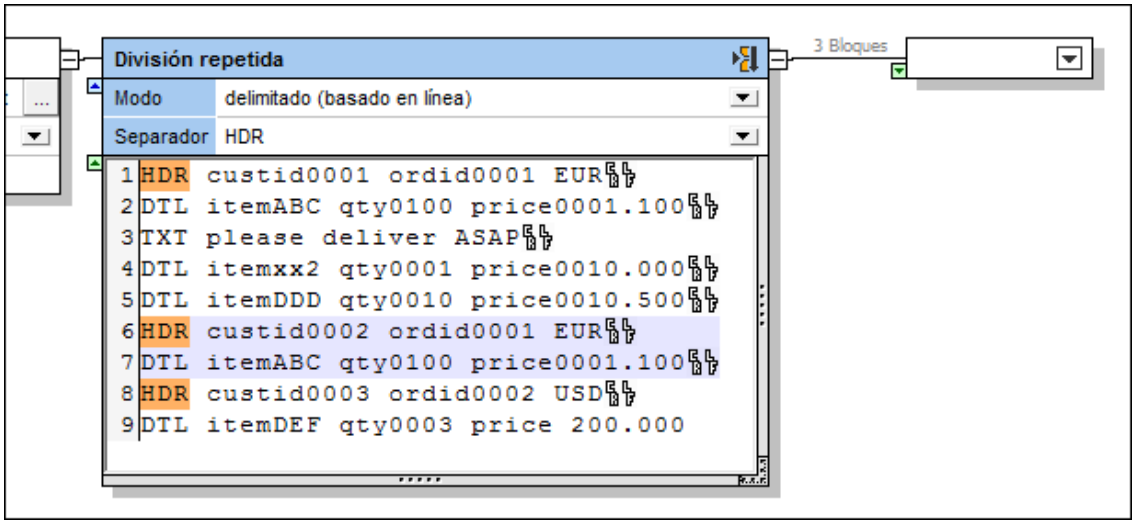
- fragmentos de texto que contienen separadores que desea eliminar
- fragmentos de texto que no tienen caracteres CR/LF y, por tanto, tienen separadores dentro de las líneas.

```
ous,222,978.4,7563.1,69,Air freight,Mini,222,ZZAW561,
```

Modo: delimitado (basado en línea)

Configuración predeterminada del modo **delimitado (basado en línea)**:

Separador	<vacío>



- Este modo de división repetida crea **varios** fragmentos, cuyo contenido viene definido por los caracteres separadores especificados en el campo *Separador*.
- Los caracteres separadores **se incluyen** dentro del fragmento.
- Se denomina *fragmento* la línea entera que contiene el separador, hasta la **siguiente** línea que contiene el mismo separador.
- Si el separador **no aparece en la primera línea**, entonces el primer fragmento contiene todas las líneas hasta la primera línea que contenga el separador.

Por ejemplo, si usamos el separador **HDR** (*imagen anterior*) obtenemos estos 3 fragmentos:

El **fragmento nº1** contiene todas las líneas comprendidas entre el principio del archivo/fragmento y la siguiente línea que contenga el mismo separador.

```
1 HDR custid0001 ordid0001 EUR
2 DTL itemABC qty0100 price0001.100
3 TXT please deliver ASAP
4 DTL itemxx2 qty0001 price0010.000
5 DTL itemDDD qty0010 price0010.500
```

Observe que con este modo de división puede acceder a todas las líneas comprendidas entre dos separadores (p. ej. un archivo donde las líneas DTL o TXT pueden ser opcionales o aparecer desordenadas).

El **fragmento nº2** contiene todas las líneas comprendidas entre la segunda aparición de HDR y la siguiente aparición de HDR.

```
1 HDR custid0002 ordid0001 EUR
2 DTL itemABC qty0100 price0001.100
```

El **fragmento nº3** contiene todas las líneas comprendidas entre la tercera aparición de HDR y el

 © 2014 Altova GmbH

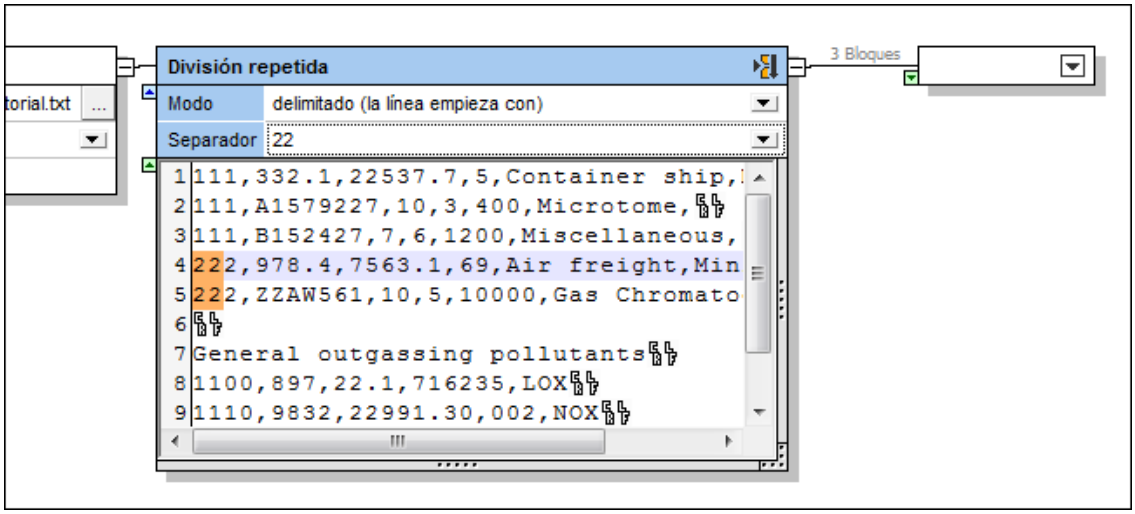
final del archivo.

```
1 HDR custid0003 ordid0002 USD
2 DTL itemDEF qty0003 price 200.000
```

Modo: delimitado (la línea empieza con)

Configuración predeterminada del modo **delimitado (la línea empieza con)**:

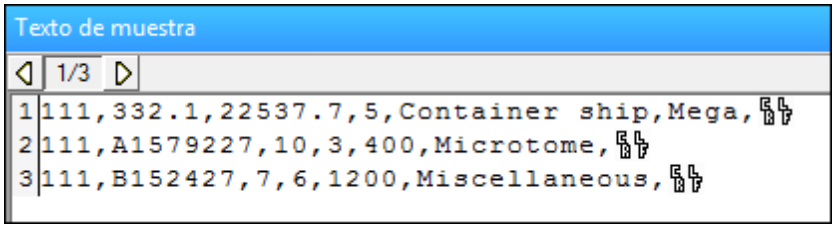
Separador	<vacío>



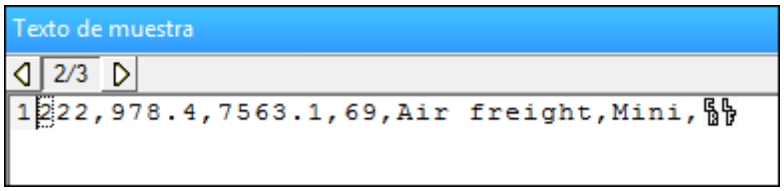
- Este modo de división repetida crea **varios** fragmentos, cuyo contenido viene definido por los caracteres separadores especificados en el campo *Separador*.
- Los caracteres separadores **se incluyen** en el fragmento.
- Se denomina *fragmento* la línea que **empieza con el separador** especificado, hasta la siguiente línea que empieza con el mismo separador.
- Si la primera línea no empieza con el separador, entonces el primer fragmento contiene todas las líneas que aparecen antes de la primera línea que empieza con el separador.

Por ejemplo, si usamos el separador 22 (*imagen anterior*) obtenemos estos 3 fragmentos:

El **fragmento nº1** contiene todos los caracteres comprendidos entre el principio del archivo/ fragmento y la primera línea que empieza con el separador 22.

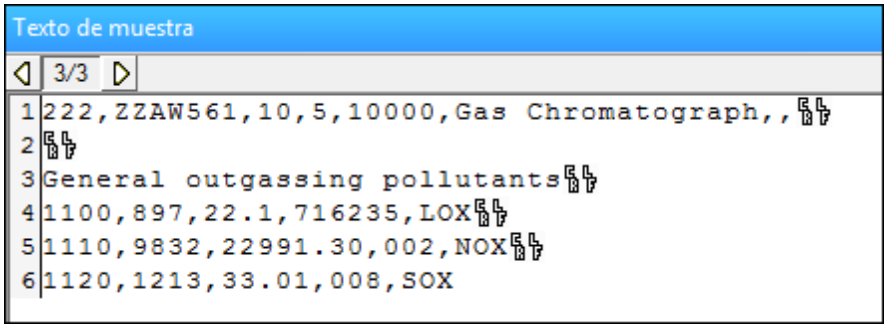


El **fragmento nº2** contiene todas las líneas del archivo comprendidas entre la primera aparición de 22 y la segunda aparición de 22 (en este caso, una sola línea).

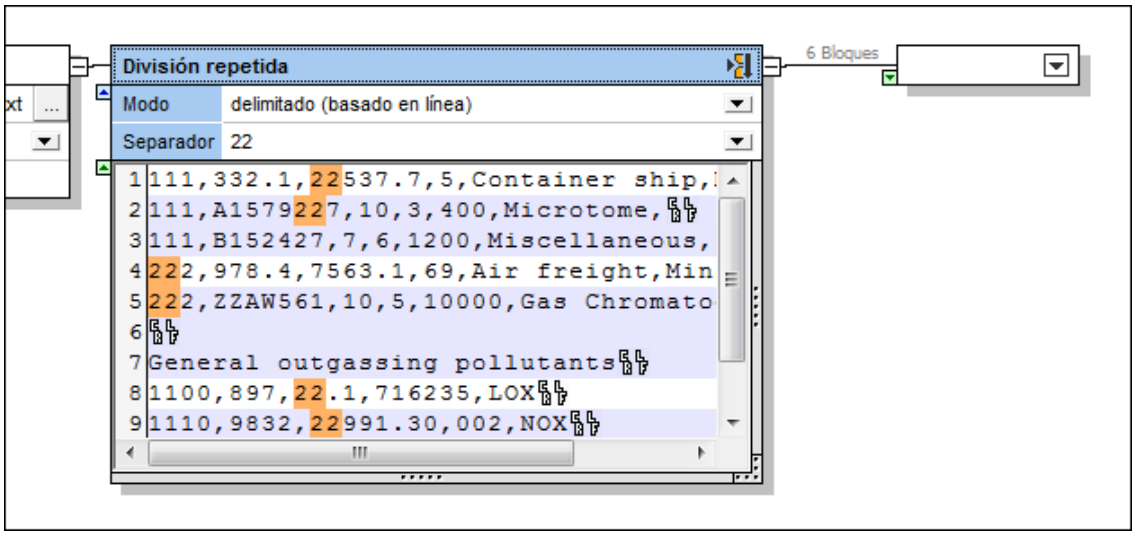


El **fragmento nº3** contiene todas las líneas del archivo comprendidas entre la segunda aparición de 22 y el final del archivo.

 © 2014 Altova GmbH



Compare el resultado de este ejemplo con lo que ocurriría si utilizara el modo **delimitado (basado en línea)** con el separador 22:



Con el modo **delimitado (basado en línea)** este mismo archivo da lugar a 6 fragmentos compuestos por las líneas que contienen el separador 22 en cualquier posición.

Dividir una vez

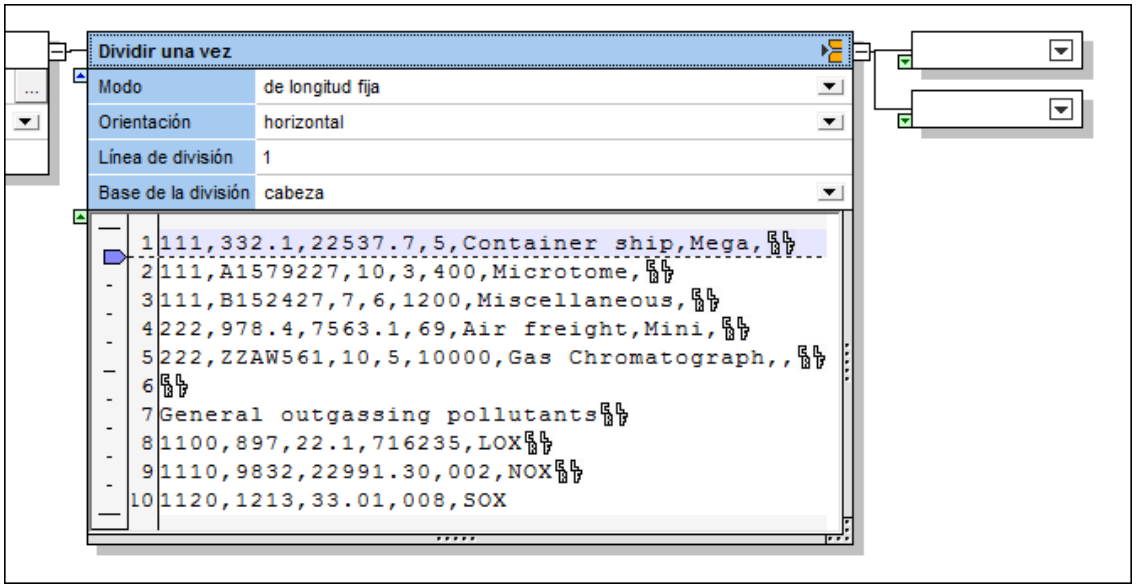
Con esta opción de división puede crear dos contenedores alineados verticalmente. El contenedor superior contiene el texto definido por la condición creada en el contenedor **Dividir una vez**. El contenedor inferior contiene el resto del archivo/fragmento de texto. La opción de división **Dividir una vez** tiene varios modos: longitud fija, delimitado (flotante) y delimitado (basado en línea).

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 de 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	cabeza



Resultado predeterminado:
La primera línea del texto aparece en el contenedor superior. El contenedor inferior incluye el resto del archivo/fragmento de texto.

 © 2014 Altova GmbH

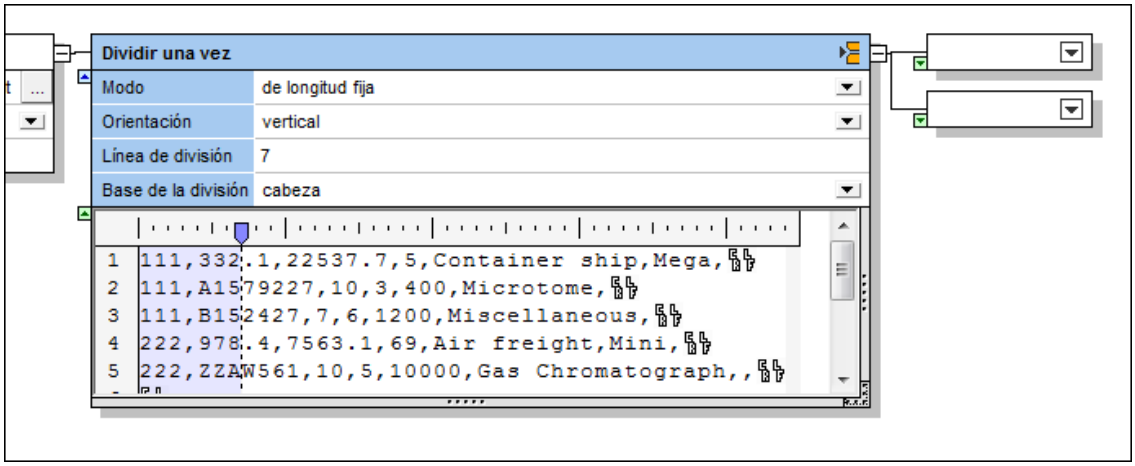
Modo: longitud fija

Este es el modo predeterminado de la opción **Dividir una vez**.

Orientación:
La orientación permite definir cómo se divide el fragmento de texto: en líneas/registros o en columnas.

Seleccione la opción **`Horizontal`** para dividir el fragmento en **dos** secciones horizontales. Introduzca un valor en el campo *Línea de división* o arrastre la tabulación de la regla vertical.

Seleccione la opción **`Vertical`** para dividir el fragmento en **dos** columnas verticales. Introduzca un valor en el campo *Línea de división* o arrastre la tabulación de la regla horizontal (*imagen siguiente*).



Línea de división:
El número de líneas después del cual se debe dividir el fragmento.

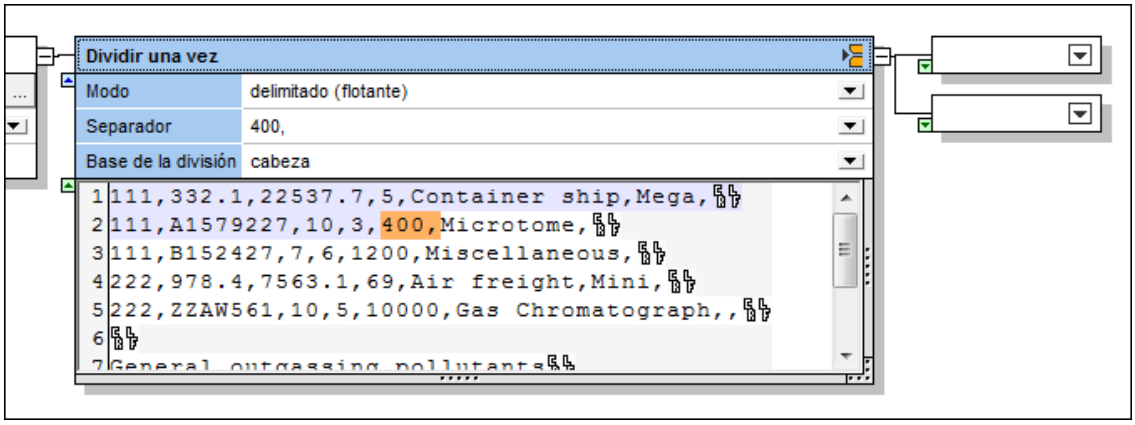
Base de la división:

`Cabeza`	divide el fragmento por el número de líneas de división empezando por arriba
`Cola`	divide el fragmento por el número de líneas de división empezando por abajo (utilice esta opción cuando no conozca el número de líneas/registros)

Modo: delimitado (flotante)

Configuración predeterminada del modo **delimitado (flotante)**:

Separador	`<vacío>`
Base de la división	`cabeza`



- Este modo de división crea **dos** fragmentos, cuyo contenido viene definido por los caracteres separadores especificados en el campo *Separador*.
- Los caracteres separadores **no se incluyen** dentro de los fragmentos.
- El **primer fragmento** es el texto comprendido entre el primer carácter del archivo/fragmento y el carácter que aparece antes del separador.
- El **segundo fragmento** es el texto comprendido entre el separador y el final del archivo/fragmento.
- Si el separador aparece en la primera/última **posición** del archivo/fragmento, el contenedor superior estará vacío.

Utilice este modo de división para dividir en dos **una sección** de un archivo (o fragmento) en la que el separador se encuentra en cualquier posición. Este modo de división es útil si trabaja con archivos que no contienen caracteres CR/LF y desea dividirlos en dos fragmentos, en una posición concreta.

El fragmento del contenedor **superior** contiene el texto comprendido entre el principio del archivo y el separador (es decir, `400`)

```
1 111,332.1,22537.7,5,Container ship,Mega,
2 111,A1579227,10,3,
```

El fragmento del contenedor **inferior** contiene el resto del archivo, desde el carácter situado tras el separador hasta el final del archivo.

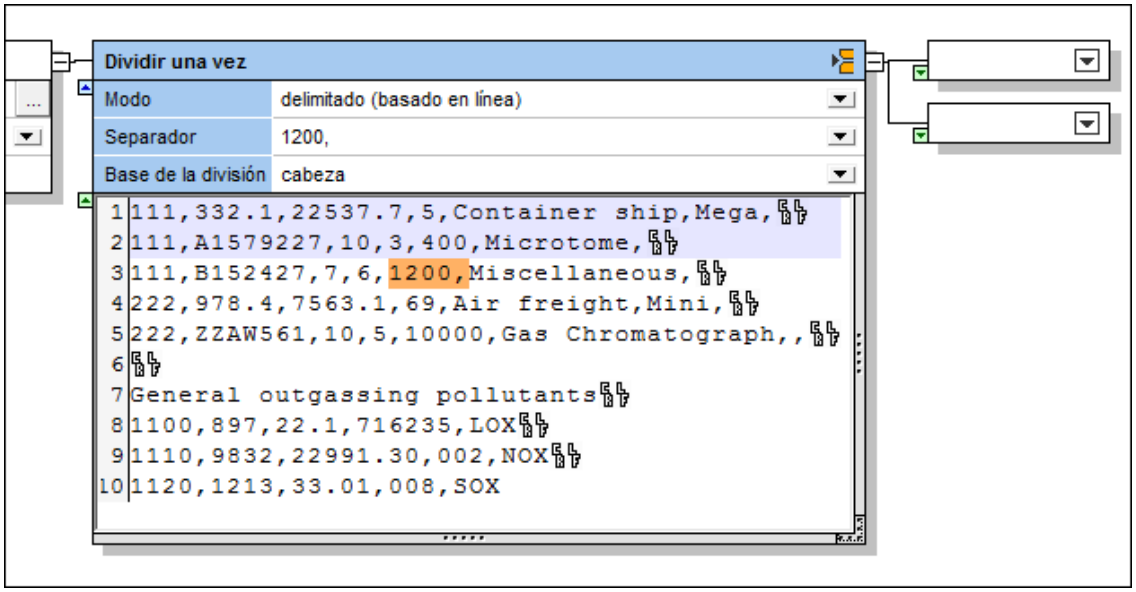
```
1 Microtome,
2 111,B152427,7,6,1200,Miscellaneous,
3 222,978.4,7563.1,69,Air freight,Mini,
4 222,ZZAW561,10,5,10000,Gas Chromatograph,,
```

 © 2014 Altova GmbH

Modo: delimitado (basado en línea)

Configuración predeterminada del modo **delimitado (flotante)**:

Separador	`<vacío>`
Base de la división	`cabeza`



- Este modo de división crea **dos** fragmentos, cuyo contenido viene definido por los caracteres separadores especificados en el campo *Separador*.
- Los separadores **se incluyen** dentro de los fragmentos.
- El **primer fragmento** es todo el texto del archivo, hasta llegar a la línea que contiene el separador.
- El **segundo fragmento** está compuesto por la línea que contiene el separador y el resto del archivo, hasta llegar al final.
- Si el separador aparece en la primera/última **línea** del archivo, el contenedor superior estará vacío.

Utilice este modo de división para dividir en **dos** un archivo (o un fragmento) en el que el separador se encuentra en una de las líneas. La línea que contiene el separador no se divide, si no que se conserva entera. Este modo de división suele ser útil si trabaja con archivos que contienen delimitadores de registro (CR/LF) y desea dividir el fragmento en dos.

El fragmento superior contiene todo el texto del archivo **hasta** llegar a la **línea** que contiene el separador.

```
1 111,332.1,22537.7,5,Container ship,Mega,
2 111,A1579227,10,3,400,Microtome,
```

El fragmento inferior contiene toda la línea que contiene el separador (p. ej. `1200`) y el resto del archivo hasta llegar al final.

```
1 111,B152427,7,6,1200,Miscellaneous,
2 222,978.4,7563.1,69,Air freight,Mini,
3 222,ZZAW561,10,5,10000,Gas Chromatograph,,
```

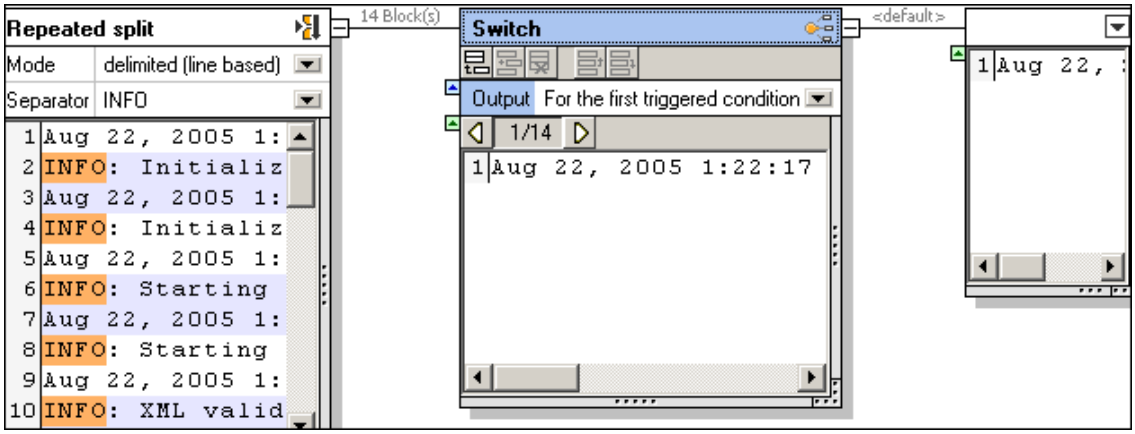
Conmutador

Con esta opción de división puede definir varias palabras clave o condiciones para un solo fragmento de texto. Cada palabra clave definida tiene asociado un contenedor que recibe datos solamente si la condición se cumple (es decir, si es verdadera). Si no se cumple ninguna condición, el fragmento se asigna a un contenedor *predeterminado*.

Esta es la **configuración predeterminada** del contenedor **Conmutador**:

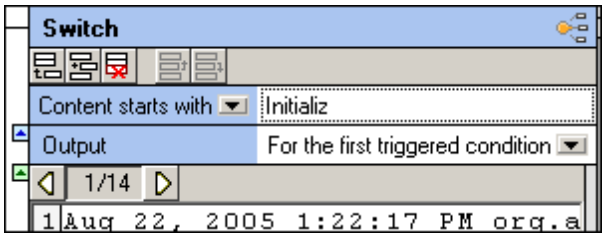
Salida	`Para la primera condición desencadenada`

El ejemplo de la siguiente imagen procesa un archivo de registro Tomcat. El objetivo es separar los distintos procesos para poder asignarlos por separado. Cuando se define un contenedor **Conmutador** por primera vez, a la derecha del **Conmutador** aparece solamente el contenedor *predeterminado*, al cual se pasan automáticamente todos los datos.



El contenedor **División repetida**, en el 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 añadir una condición nueva al contenedor **Conmutador**.
2. Haga doble clic en el campo *El contenido empieza con*, introduzca la cadena `Initializ` y pulse la tecla **Entrar**.

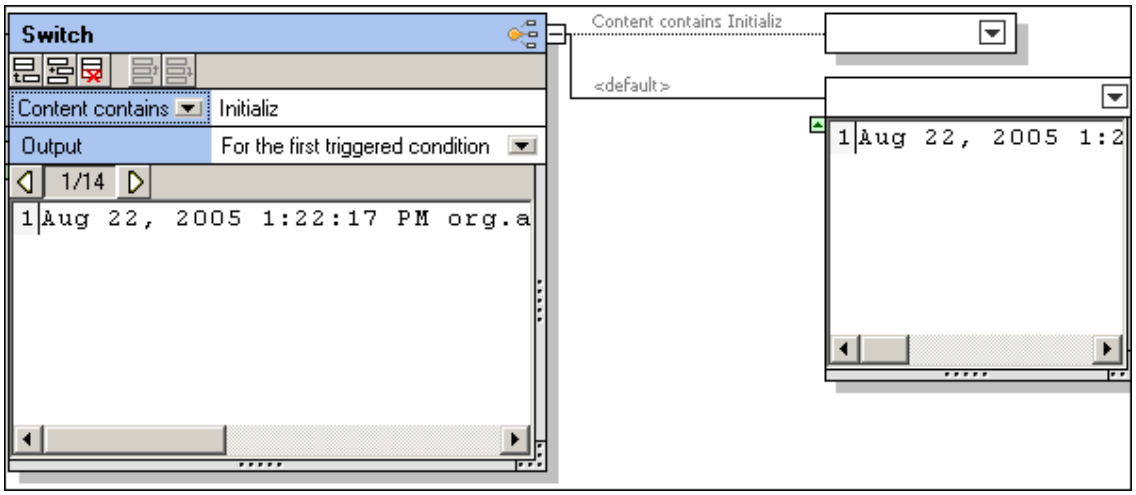


Se añade un contenedor nuevo. Los datos se enviarán a este contenedor si la condición se cumple. Si no se cumple, los datos se enviarán al contenedor *predeterminado*.
La condición recién definida no se cumple (el primer fragmento no empieza con la cadena `Initializ`) y su contenido se envía al contenedor *predeterminado*.

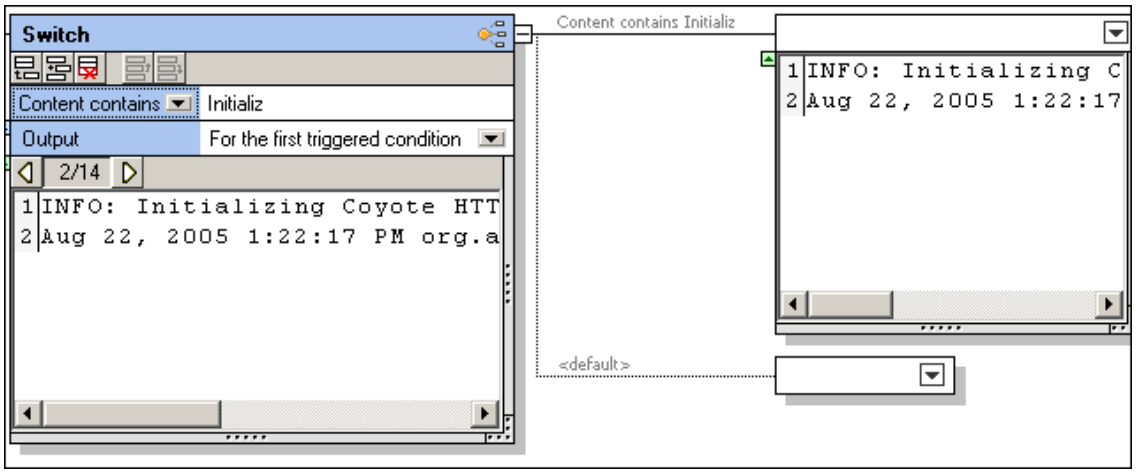
3. Haga clic en el cuadro combinado *El contenido empieza con* y en su lugar seleccione la

 © 2014 Altova GmbH

opción *El contenido contiene*.
Esta condición tampoco se cumple en el primer bloque/fragmento y su contenido se envía al contenedor *predeterminado*.

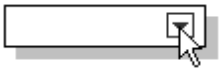


4. Haga clic en el icono **Mostrar siguiente bloque** para ver el siguiente fragmento de texto.

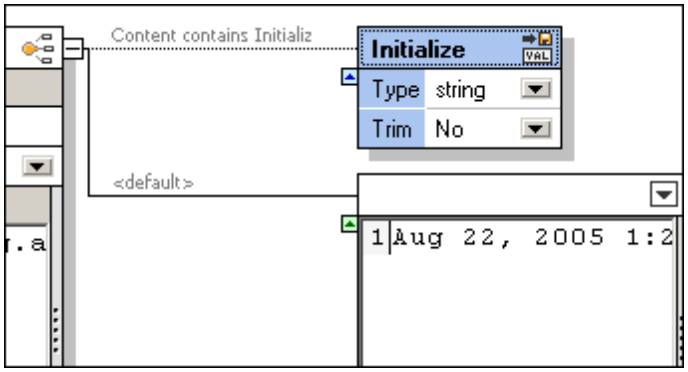


El contenido de este segundo fragmento se pasa al contenedor asociado a la condición *El contenido contiene...* y el contenedor *predeterminado* está vacío. Recorra los diferentes fragmentos (14 en total) con las flechas de desplazamiento para ver una vista previa del contenido de los fragmentos y en qué casos se cumple la condición.

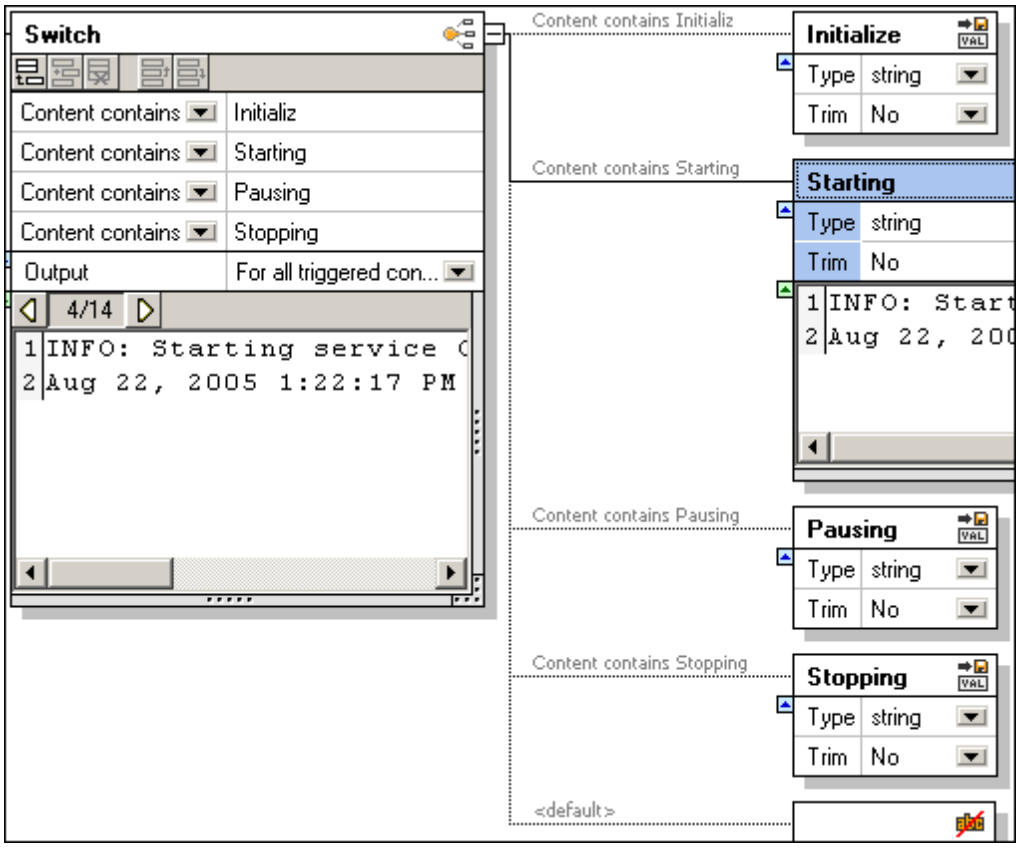
5. Haga clic en el botón interactivo del contenedor *El contenido contiene...* y seleccione la opción **Guardar como valor**.



6. Haga doble clic en el título **Guardar** del contenedor y escriba **Initialize**.



7. Haga clic en el icono **Anexar** para añadir una condición nueva al contenedor **Conmutador**.
8. Haga clic en el campo *El contenido empieza con* y seleccione *El contenido empieza con*.
   Puede incluir todas las condiciones que quiera.
   Ponga un nombre a cada contenedor asociado para que sea más fácil reconocerlos en la vista de asignación de MapForce.



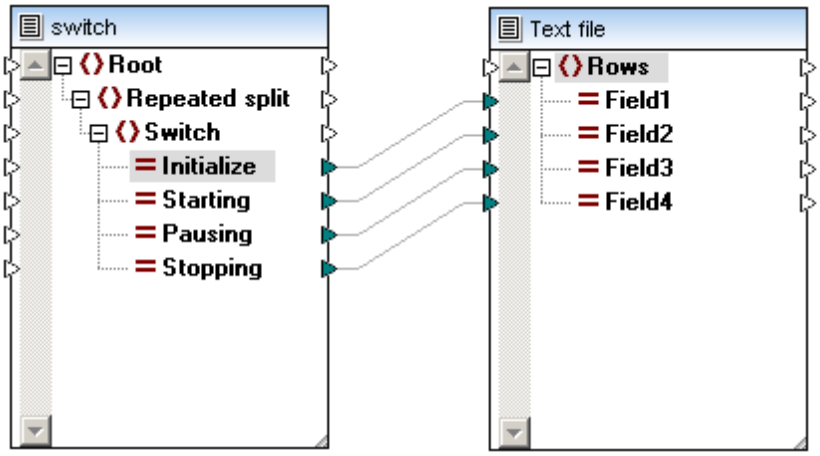
En la imagen anterior puede ver las cuatro condiciones definidas y el contenido del contenedor **Starting** para el bloque/fragmento nº4. Los contenedores asociados a las condiciones se renombraron para poder identificarlos más fácilmente en el componente de MapForce.

Recuerde que puede cambiar la posición de las condiciones dentro de la lista, usando

 © 2014 Altova GmbH

los botones **Subir** y **Bajar**.

9. Guarde la plantilla de FlexText e insértela en MapForce.



Nota: si un fragmento de texto del fragmento actual cumple una condición, **todos los datos** del fragmento se pasan al contenedor asociado. Los datos no se dividen, simplemente se envían a los contenedores asociados o, si no se cumple ninguna condición, al contenedor *predeterminado*. Los contenedores asociados producidos por un contenedor Conmutador se pueden seguir procesando. Por ejemplo, puede definirlos como contenedores **Dividir una vez**, **División repetida**, etc.

El contenido empieza con:
Los datos se pasan al contenedor asociado solamente si la cadena de condición aparece al principio del fragmento de texto.

El contenido contiene:
Los datos se pasan al contenedor asociado solamente si la cadena de condición aparece en cualquier posición en el fragmento de texto.

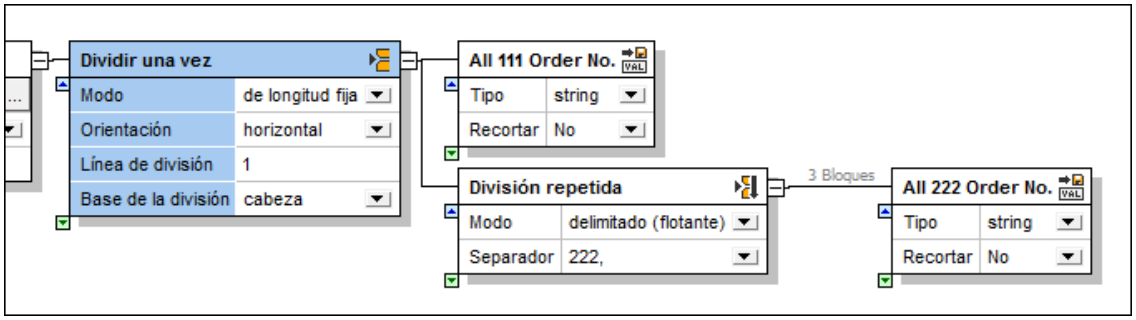
Para la primera condición desencadenada:
Los datos se pasan al contenedor asociado cuando **se cumple una de las condiciones** de la lista. Las demás condiciones se ignoran, aunque se cumplan, y no se pasan más datos a los demás contenedores asociados.

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.

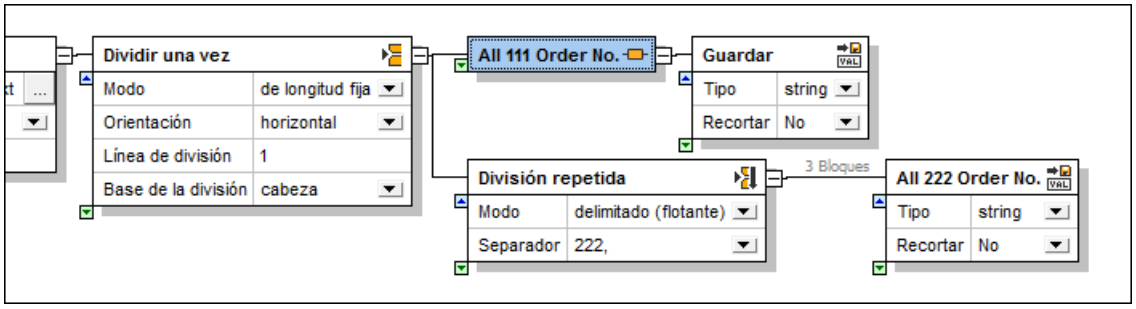
Nodo

Con esta opción puede añadir un nivel jerárquico nuevo a las estructuras en forma de árbol de FlexText. 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.



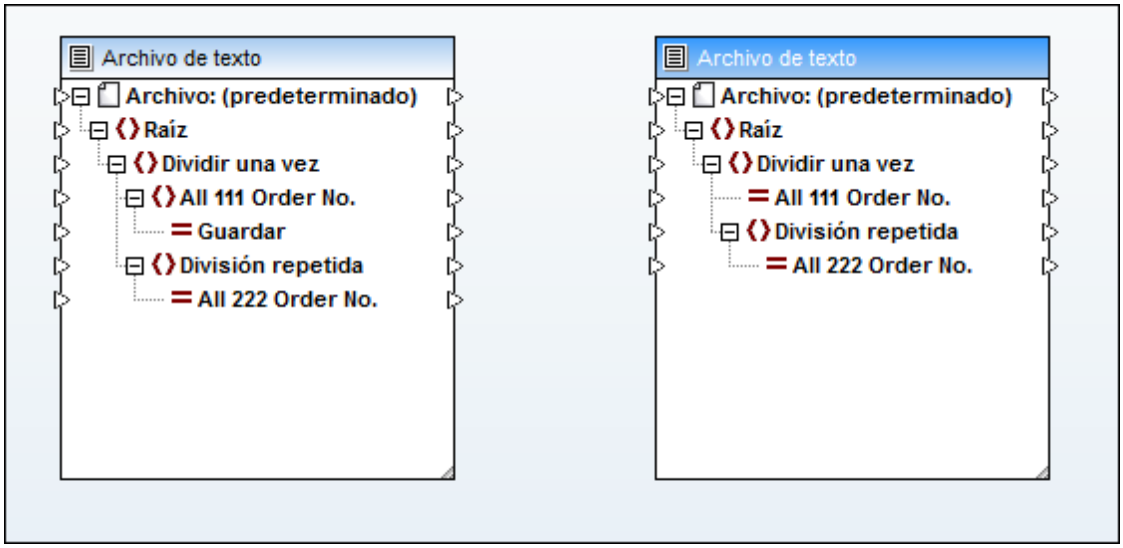
Haga clic en el icono interactivo del contenedor y seleccione la opción **Nodo**.



Se añade un contenedor nuevo a la derecha del contenedor actual.

Nota: el contenedor anexado automáticamente se definió como contenedor **Guardar valor**.

La siguiente imagen muestra el aspecto (la estructura) que la plantilla FlexText tiene en MapForce antes y después de crear el contenedor **Nodo**.



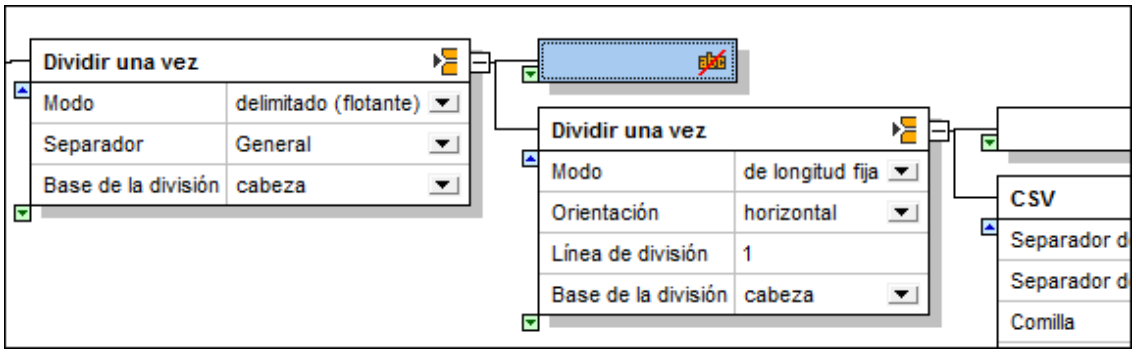
El componente de la derecha muestra la estructura inicial de la plantilla, antes de añadir el contenedor **Nodo** nuevo.

El componente de la izquierda muestra la nueva estructura. Ahora **`All 111 Order No.`** es el elemento primario y debajo de él se añadió el elemento secundario **`Guardar`**.

 © 2014 Altova GmbH

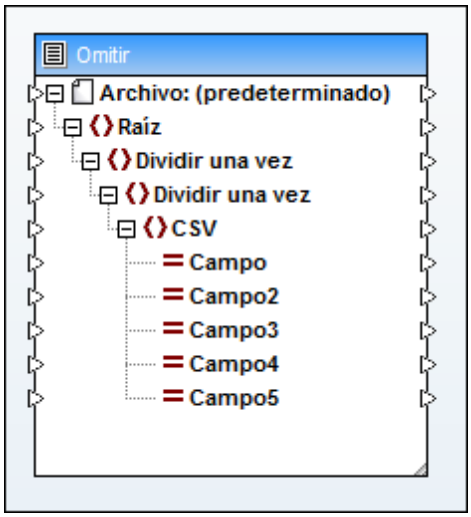
Omitir

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 el componente FlexText.

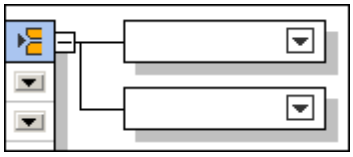


Por ejemplo, en la imagen anterior, el contenedor activo se definió como contenedor **Omitir**. Por tanto, el texto de este contenedor no aparecerá como elemento asignable en MapForce.

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 MapForce el tipo de contenedor: **Guardar valor**, **Guardar como CVS**, etc.

Guardar como CSV (delimitado)

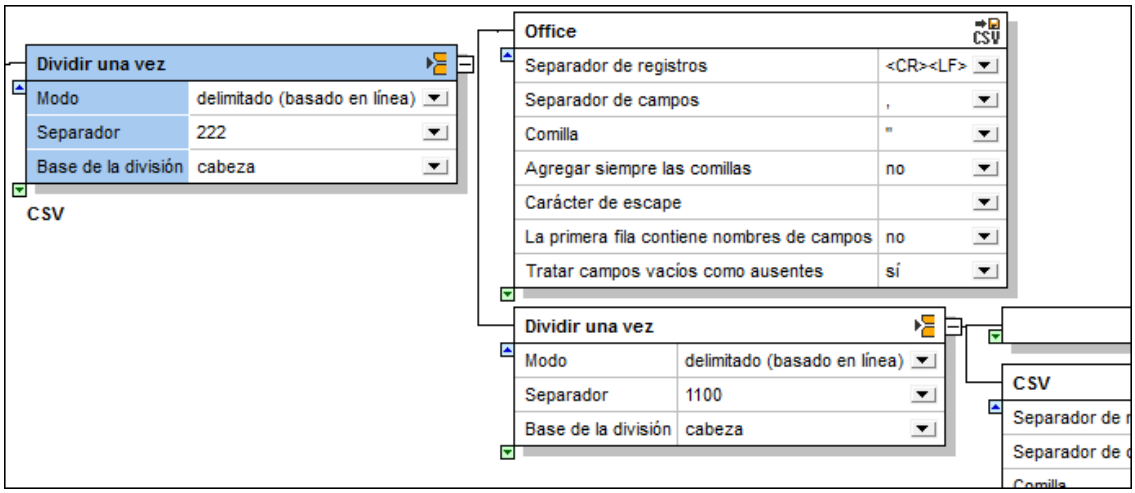
Con esta opción puede guardar fragmentos de texto como texto CSV y asignar sus columnas en MapForce. En FlexText puede crear un número ilimitado de contenedores/componentes CSV y cada contenedor CSV puede tener varios separadores.

El panel de 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**:

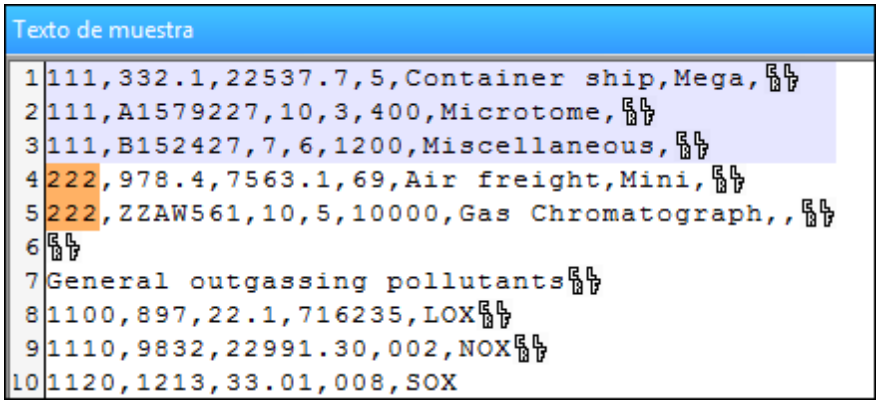
Separador de registros	`CR LF`
Separador de campos	`,`
Comilla	`"`
Agregar siempre las comillas	`no`
Carácter de escape	`<vacío>`
La primera fila contiene nombres de campos	`no`
Tratar campos vacíos como ausentes	`sí`

En el ejemplo de la siguiente imagen 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).

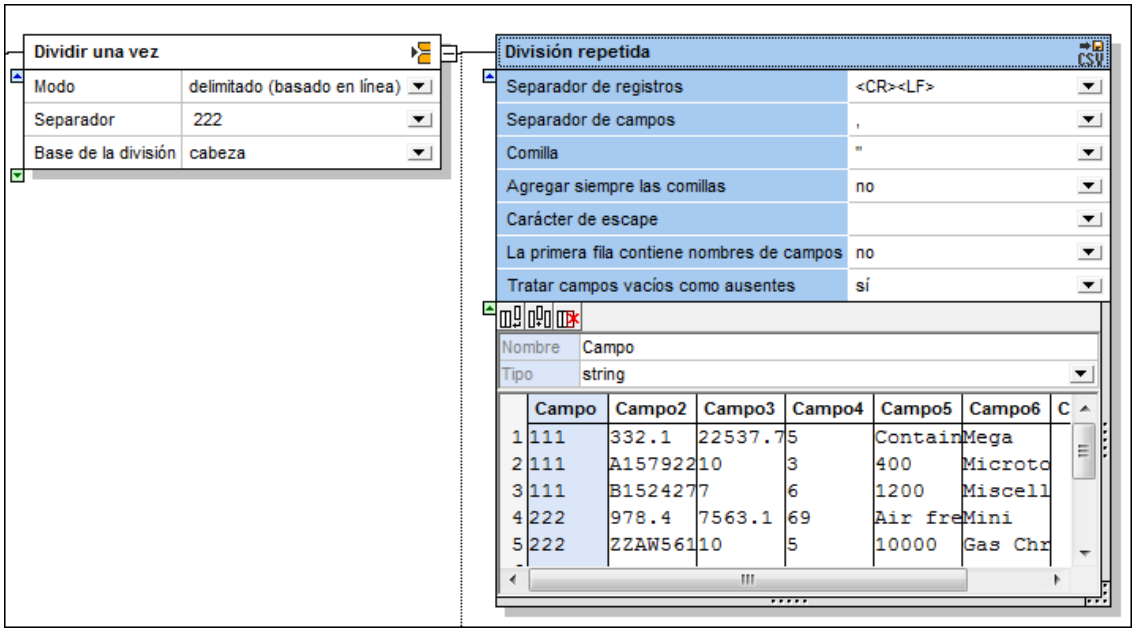


El contenedor **Dividir una vez** (*imagen anterior*) se utiliza para crear **dos** contenedores. Para ello se usa el modo **`delimitado (basado en línea)`** con el separador 222. Todos los registros que aparecen antes del primer 222 se pasan al contenedor **CSV**. El primer contenedor, formado por todos los registros que incluyen el texto 111, se define como contenedor **CSV**. El panel de texto de muestra ofrece una vista previa del contenido del contenedor activo, es decir, el contenedor **Dividir una vez**.

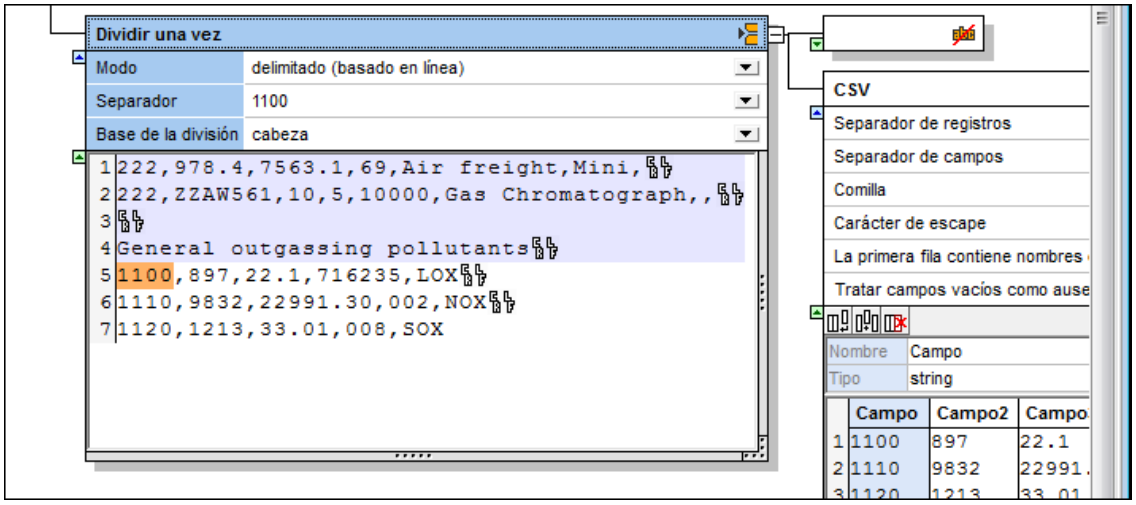
 © 2014 Altova GmbH



La configuración predeterminada del contenedor **CSV** no se modifica. Haga clic en el contenedor **CSV** para ver su contenido en forma de tabla.

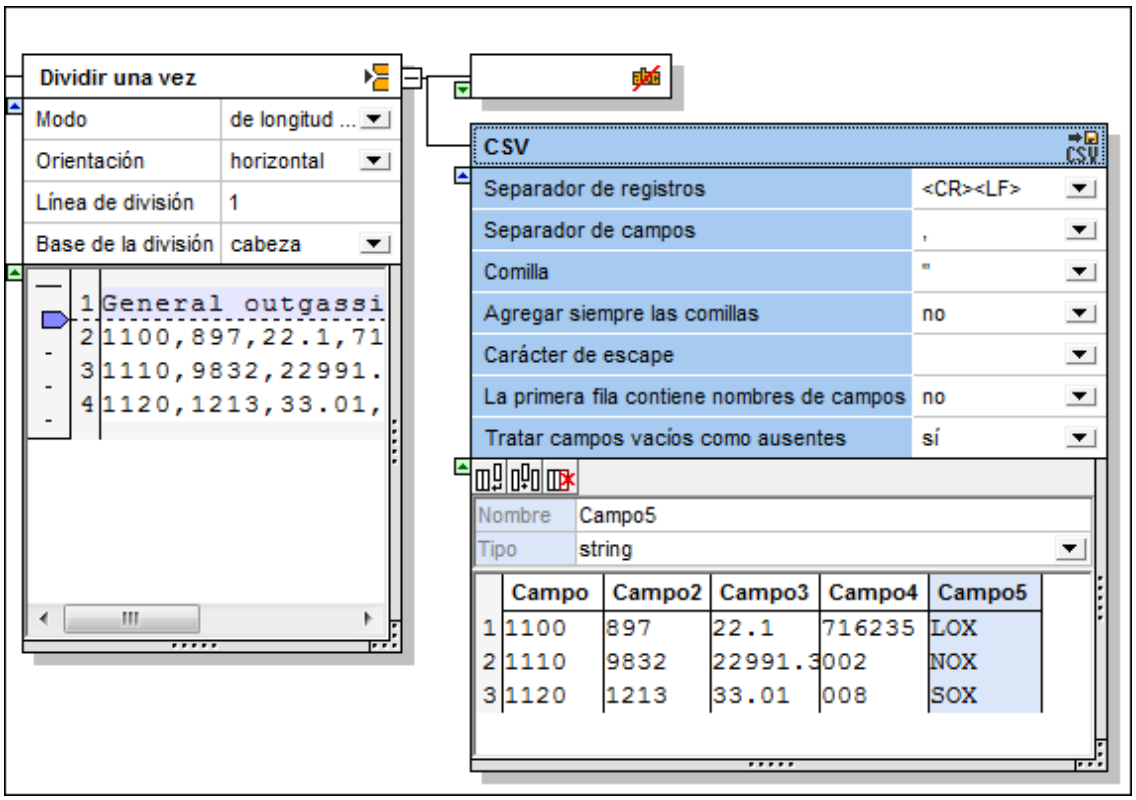


El segundo contenedor incluye los datos sobrantes y se define a su vez como contenedor **Dividir una vez**. Como resultado se crean dos contenedores más, uno de los cuales se define también como contenedor **CSV**. Haga clic en el contenedor **Dividir una vez** para ver su contenido actual.



El modo **`delimitado (basado en la línea)`** con el separador `1100` se utiliza para dividir los datos sobrantes en dos secciones.

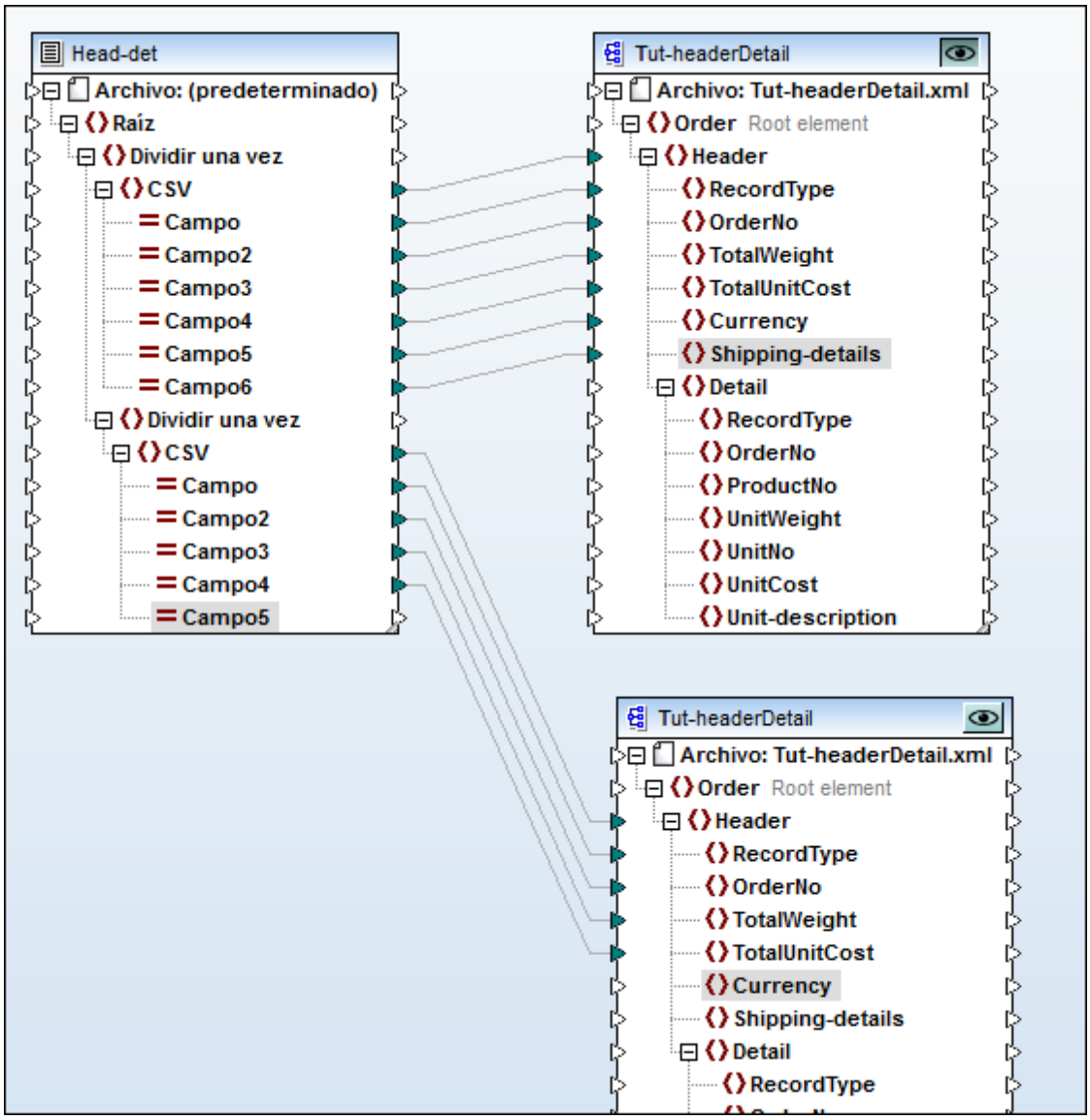
- Todos los registros que aparecen antes del primer 1100 se pasan al primer contenedor, que a su vez se define como contenedor **Omitir** , para que no se pueda asignar en MapForce.
- El segundo contenedor se define como contenedor **CSV**. La configuración predeterminada no se cambia. Haga clic en el 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

 © 2014 Altova GmbH

de la asignación. En este ejemplo, cada contenedor **CSV** se asigna a un archivo XML diferente.



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.

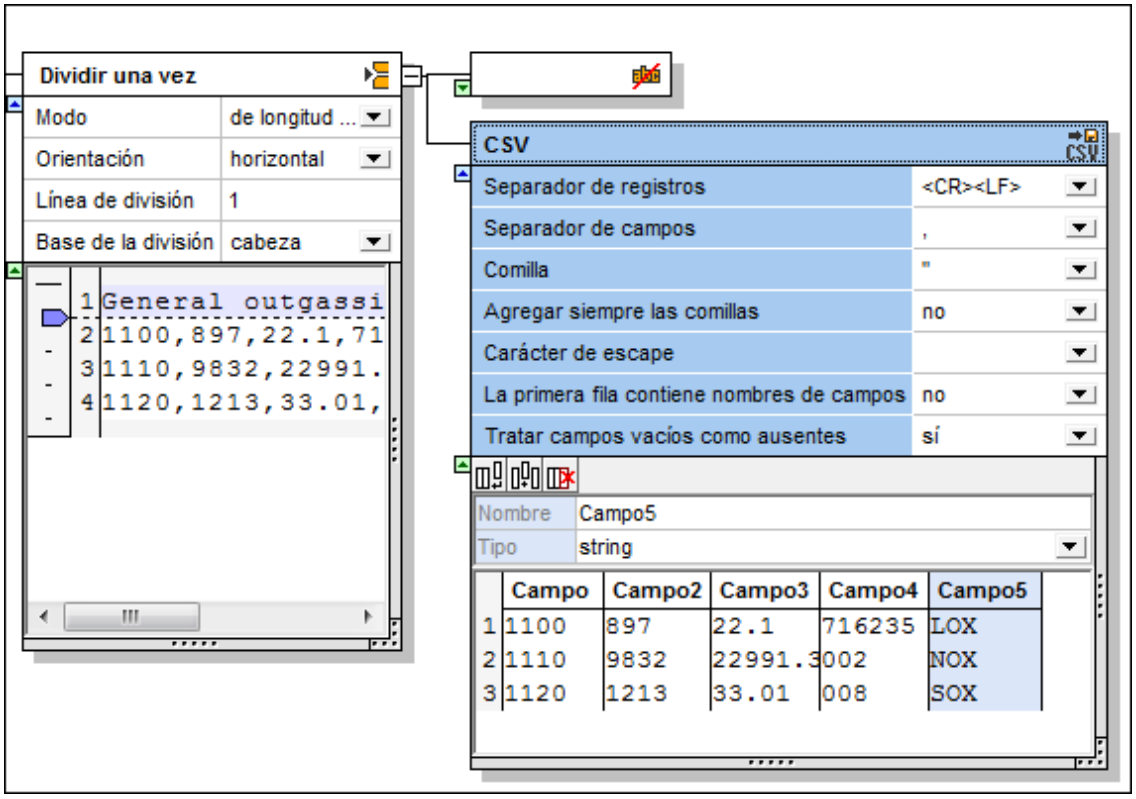
```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     <Header>
4       <RecordType>111</RecordType>
5       <OrderNo>332</OrderNo>
6       <TotalWeight>22537.7</TotalWeight>
7     </Header>
8     <Header>
9       <RecordType>111</RecordType>
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.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     <Header>
4       <RecordType>1100</RecordType>
5       <OrderNo>897</OrderNo>
6       <TotalWeight>22.1</TotalWeight>
7       <TotalUnitCost>716235</TotalUnitCost>
8     </Header>
9     <Header>
10      <RecordType>1110</RecordType>
11      <OrderNo>9832</OrderNo>
12      <TotalWeight>22991.3</TotalWeight>
13      <TotalUnitCost>2</TotalUnitCost>
14    </Header>
15    <Header>
16      <RecordType>1120</RecordType>
17      <OrderNo>1213</OrderNo>
18      <TotalWeight>33.01</TotalWeight>
19      <TotalUnitCost>8</TotalUnitCost>
20    </Header>
21  </Order>
22
```


© 2014 Altova GmbH

Configurar los datos/el contenedor CSV:



Al hacer clic en un campo del panel de 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.
- 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 mouse 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.

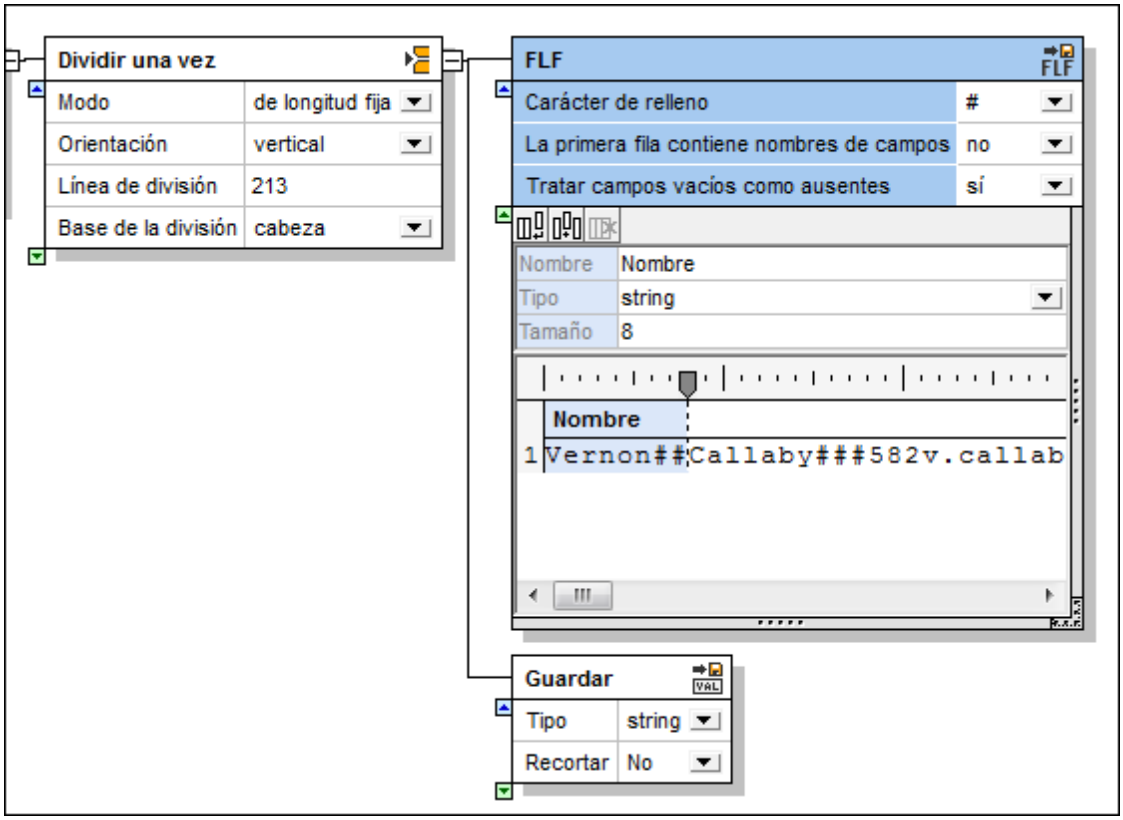
Guardar como FLF (longitud fija)

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 de 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`



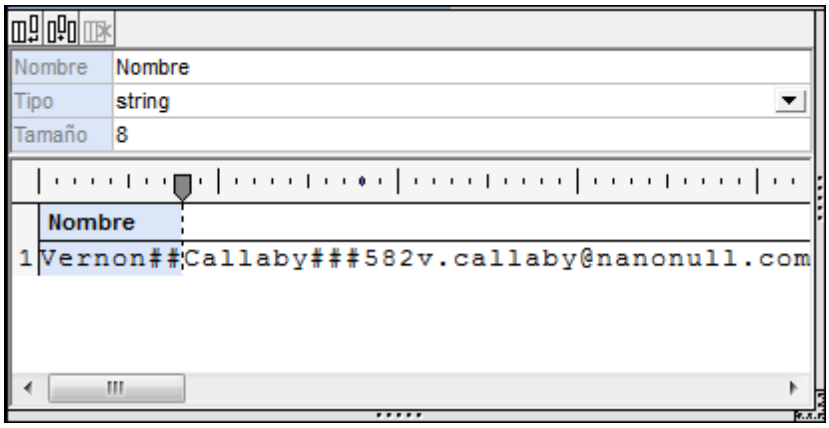
Configurar los datos/el contenedor FLF:
Tras definir un contenedor como **Guardar como FLF**, el panel de texto de muestra tiene el aspecto de la imagen anterior. Se inserta automáticamente un campo de 10 caracteres de ancho.

- Haga clic en la tabulación de la regla y arrástrela a la nueva posición. Aparece un cuadro

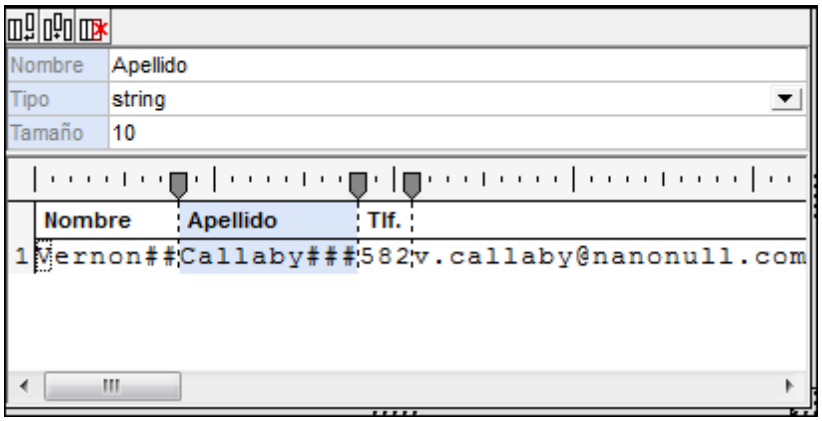
 © 2014 Altova GmbH

emergente que indica 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*.



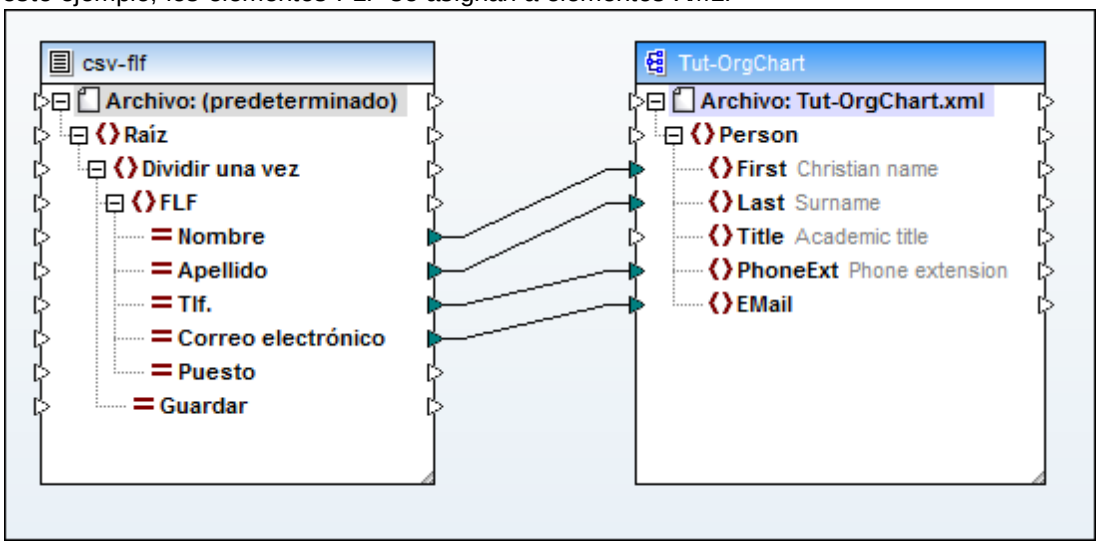
- Para quitar un campo, haga clic en la tabulación y arrástrela fuera de la regla. Cuando la tabulación aparezca con una cruz roja, suelte el botón del mouse y la tabulación desaparece.

  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 de 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.



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.

Se considera que un campo está ausente cuando no hay datos entre dos caracteres de relleno consecutivos.

Guardar valor

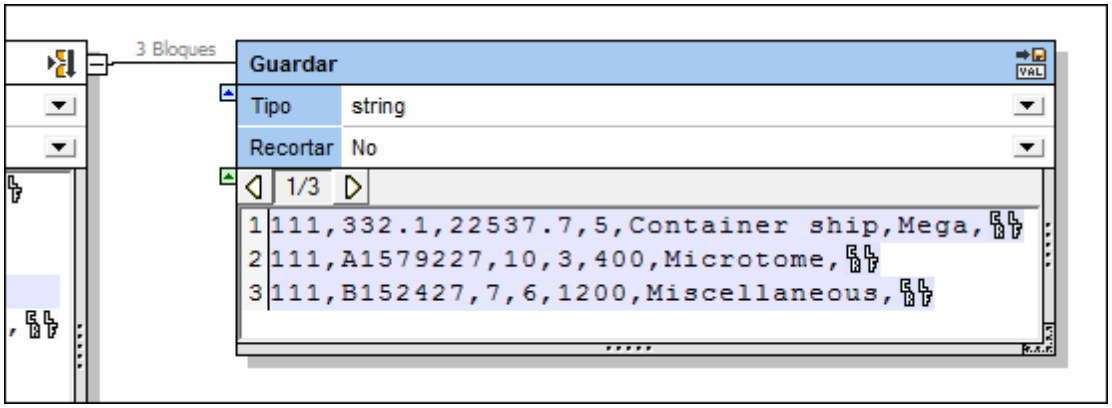
Con esta opción puede crear un contenedor cuyos datos estarán disponibles en MapForce en forma de elemento asignable. Si no cambia el nombre del contenedor en FlexText, el elemento asignable aparece en MapForce con el nombre **`Guardar`**.

Esta esta la **configuración predeterminada** del contenedor **Guardar**:

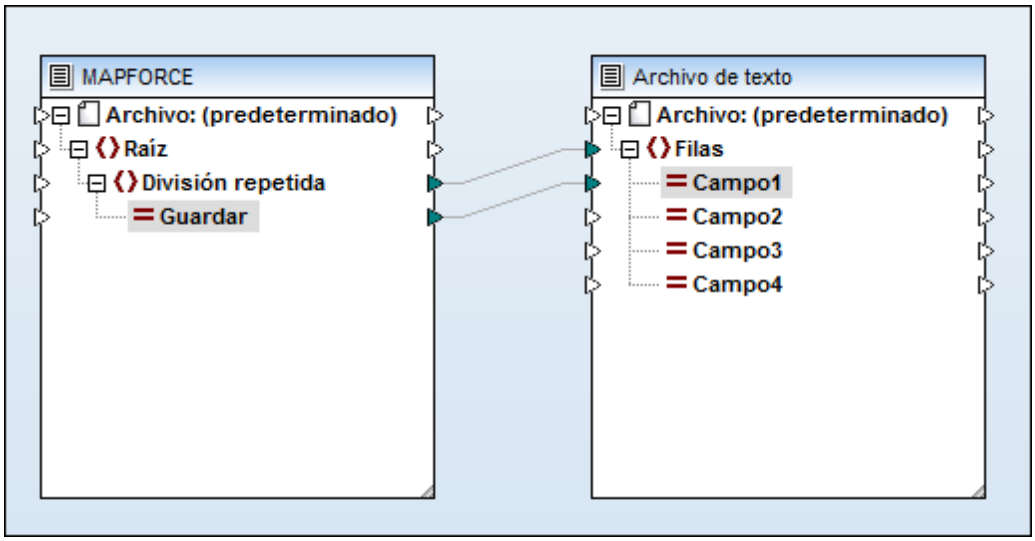
Tipo	`string`
Recortar	no

En el ejemplo siguiente puede ver un contenedor **Guardar** y su contenido en el panel de texto de muestra.

 © 2014 Altova GmbH



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`**.

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*.

10.5 EDI: UN/EDIFACT, ANSI X12, HL7, PADIS, SAP IDoc

Sitio web de Altova: Asignación de datos de archivos EDI / EDIFACT / HL7

MapForce es compatible con varios formatos EDI (intercambio electrónico de datos) y permite asignar datos EDI al resto de formatos compatibles. Para trabajar con archivos EDI es necesario seleccionar Java, C#, C++ o BUILTIN (motor de ejecución integrado) como lenguaje de salida.

Puede descargar los archivos de configuración de todos los formatos EDI compatibles desde el sitio web de Altova.

Los archivos de configuración que vienen con el instalador de MapForce se suelen actualizar en cada versión nueva de MapForce. Si quiere usar una versión concreta del estándar EDI, descargue los archivos de configuración desde el sitio web de Altova.

Los componentes EDI se pueden usar como componentes de origen y destino en las asignaciones de MapForce. Sus datos se pueden asignar a componentes de esquema XML, bases de datos, etc.

Además MapForce puede procesar varios tipos de mensajes en cada componente EDI (de la misma versión). Es decir, en MapForce podrá:

- procesar como instancias de entrada varios archivos que contengan mensajes diferentes
- procesar mensajes de tipo diferente dentro del mismo archivo (es decir, dentro del mismo intercambio)

UN/EDIFACT
es un estándar de la industria financiera para el intercambio de documentos y un estándar de la ONU. MapForce es compatible con los mensajes incluidos en los directorios **93A - 10A** y usa la versión **2010A** como versión predeterminada del estándar UN/EDIFACT. En este directorio hay unos 200 tipos de mensaje diferentes.

Para más información sobre los directorios, códigos de servico, etc. visite esta página http://www.unece.org/trade/untdid/welcome.htm.

ANSI X12
es un estándar industrial para el intercambio de documentos. MapForce es compatible con las versiones 3040, 3050, 3060, 3070, 4010, 4020, 4030, 4040, 4041, 4042, 4050, 4051, 4052, 4060, 5010, 5011, 5012, 5020, 5030, 5040, 5050, 6010 y 6020. Además utiliza la versión **6020** como versión predeterminada. Para más información visite este sitio web http://www.x12.org/.

Los componentes ANSI X12 tienen nodos *virtuales* en los que se escribe la información sobre errores de analizador EDI, según las opciones seleccionadas en el cuadro de diálogo Configuración de la validación de EDI. Además puede crear un mensaje de confirmación funcional X12 997 a partir de cualquier documento X12.

HL7 (Health Level Seven)
es un estándar industrial para el intercambio de datos entre aplicaciones médicas. MapForce es compatible con las versiones 2.2 - 2.6 y usa la versión **2.6** como versión predeterminada. La versión 3.x, basada en XML, se puede usar en las asignaciones de MapForce 2014 que utilicen componentes de esquema.

 © 2014 Altova GmbH

Desde el sitio web de Altova puede descargar los instaladores para los esquemas XML y archivos de configuración HL7 V2.2 - V2.5.1.

Además, la carpeta **`EDI and X12 mappings`** del proyecto **`MapForceExamples`** contiene el archivo de muestra `HL7V260_to_HL7V3.mfd` que asigna un archivo XML HL7 V2.6 a un archivo XML HL7 V3.

Para más información sobre este estándar visite este sitio web http://www.hl7.org/.

IATA PADIS

PADIS (*estándar de intercambio de datos sobre pasajeros y aeropuertos*) es un estándar industrial para el intercambio de información sobre pasajeros y aeropuertos mediante documentos EDI. MapForce es compatible con las versiones 00.1 - 08.1.

Para más información visite este sitio web http://www.iata.org/Pages/default.aspx.

SAP IDocs

Los documentos SAP IDocs (documentos intermedios) se utilizan para intercambiar datos comerciales entre aplicaciones SAP R/3 y aplicaciones no SAP. Los documentos son un tipo de almacenamiento de datos intermedio que se puede intercambiar entre sistemas distintos.

Para más información visite el sitio web http://sap.com.

HIPAA X12

HIPAA se basa en el estándar X12 EDI 5010 pero tiene sus propias versiones especiales, que son compatibles con MapForce 2011 versión 3 o superior. Altova también ofrece instaladores para las versiones HIPAA 5010A1 y 5010A2.

Descargar más bibliotecas:

Visite la página Bibliotecas de MapForce del sitio web de Altova para descargar instaladores para más archivos de configuración EDI.

10.5.1 Terminología EDI

A continuación se describen brevemente los términos principales del estándar UN/EDIFACT y los términos equivalentes del estándar ANSI X12, utilizado en EE UU.

Mensajes (llamados transacciones en ANSI X12)
Un mensaje (o conjunto de transacciones) es una secuencia específica de segmentos que representa un documento comercial. En MapForce puede usar componentes EDI que contengan varios tipos de mensajes diferentes.

Segmento
Uno de los registros que forman parte 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 ANSI X12). Los segmentos de una transacción pueden ser de tipo obligatorio o condicional (opcional).

Elemento
Cada uno de los campos de datos que forman parte de un segmento. Los elementos se pueden entender como campos, es decir, contienen un tipo de datos, un nombre o una dirección, por ejemplo. Los elementos se dividen en elementos compuestos, formados por componentes o subelementos.

Separadores
Los elementos se delimitan con caracteres separadores. En UN/EDIFACT estos separadores son caracteres predeterminados o definidos en un segmento de control UNA opcional.

Caracteres separadores predeterminados:

- dos puntos **:** separador de componentes de elemento
- más **+** separador de elementos de datos
- apóstrofe **'** terminador de segmento

En HL7 se puede usar un separador de sub-subcampo adicional (el separador predeterminado es &). Este separador se denomina *separador de subcomponentes* en el cuadro de diálogo "Configuración del componente".

Intercambio
Un conjunto de sobres de grupo y/o mensajes para el mismo destinatario.

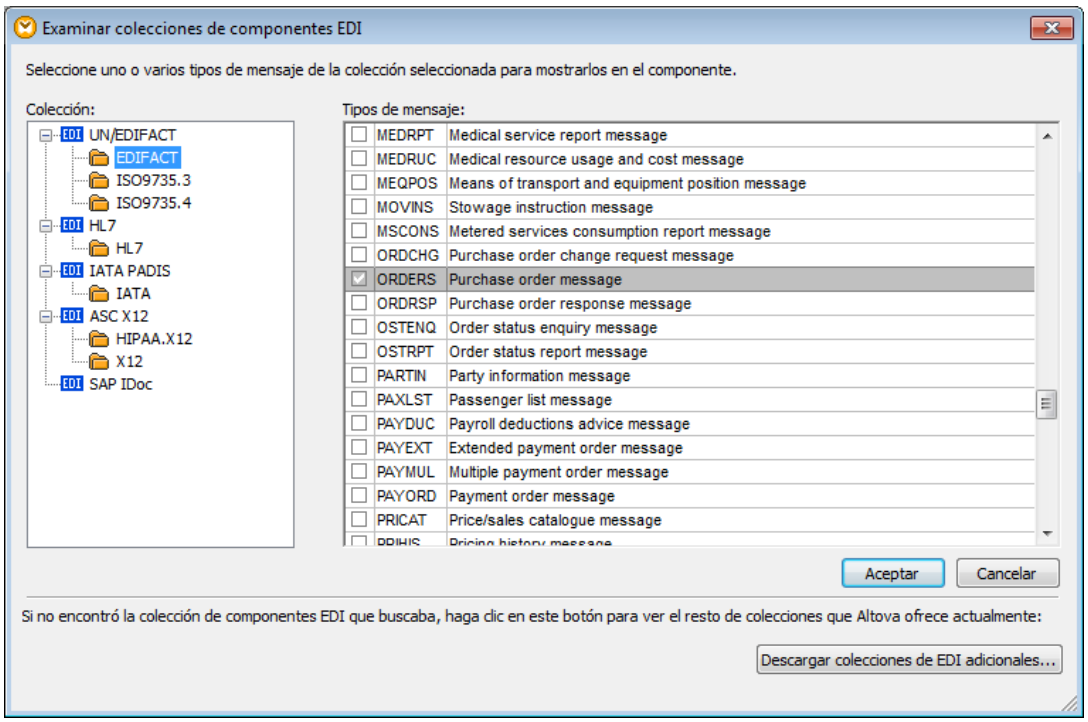
 © 2014 Altova GmbH

10.5.2 Asignaciones UN/EDIFACT - XML Schema

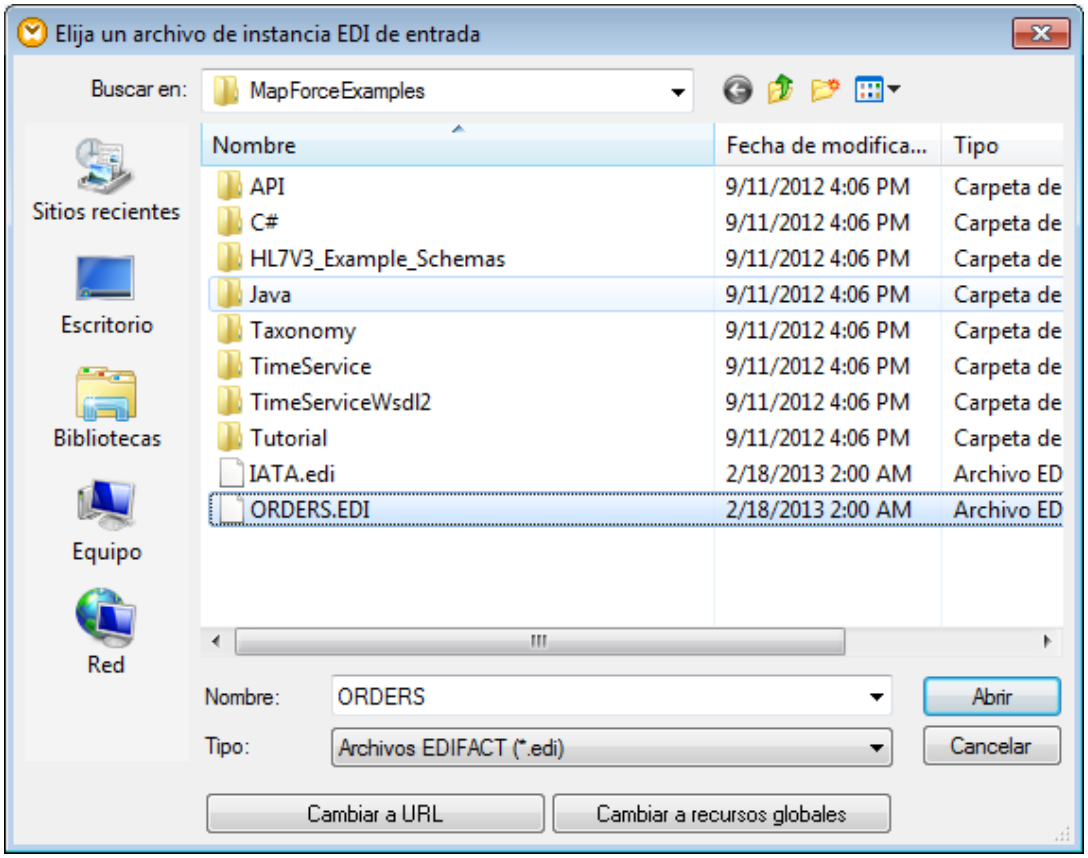
Este apartado explica cómo asignar datos de mensajes UN/EDIFACT a un esquema XML para generar un archivo XML (que después se puede procesar). La asignación de datos descrita en este apartado es el archivo `EDI_Order.mfd`, disponible en la carpeta `...\MapForceExamples`.

Crear el componente EDIFACT en MapForce:

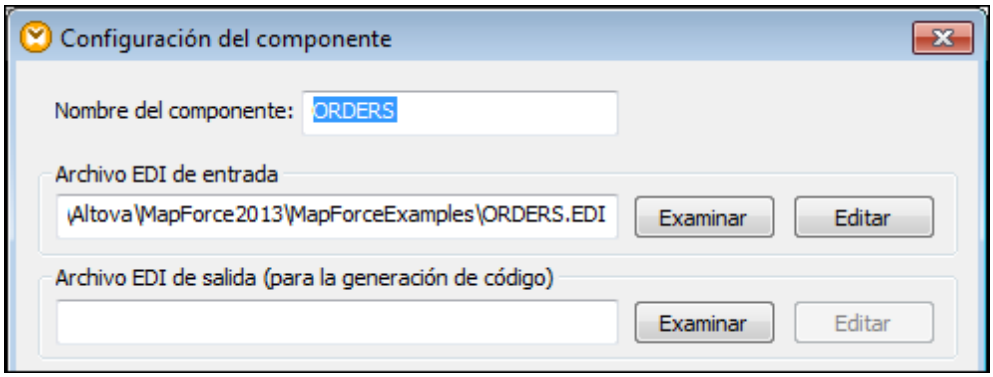
1. Cree un archivo de asignación nuevo y compruebe que la opción activa es **Java**, **C#**, **C++** o **BUILTIN**. MapForce no puede generar código XSLT 1.0/2.0 ni XQuery para asignaciones de datos EDI.
2. Seleccione la opción de menú **Insertar | EDI** o haga clic en el icono **EDI**. En la lista de colecciones EDI (*imagen siguiente*), seleccione **UN/EDIFACT | EDIFACT**. En el panel *Tipos de mensaje* marque la casilla **ORDERS** y haga clic en **Aceptar**.



3. En el aviso que aparece haga clic en el botón **Examinar**. Después seleccione `ORDERS.EDI` en el directorio `...\MapForceExamples` y haga clic en **Abrir**.

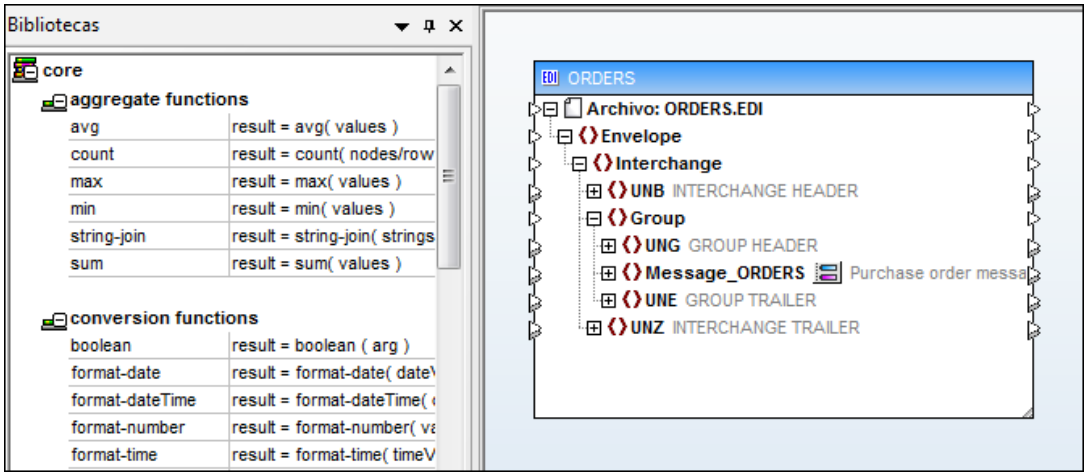


A continuación se abre el cuadro de diálogo "Configuración del componente" donde puede seleccionar el archivo EDI de entrada y de salida. Para continuar haga clic en **Aceptar**.



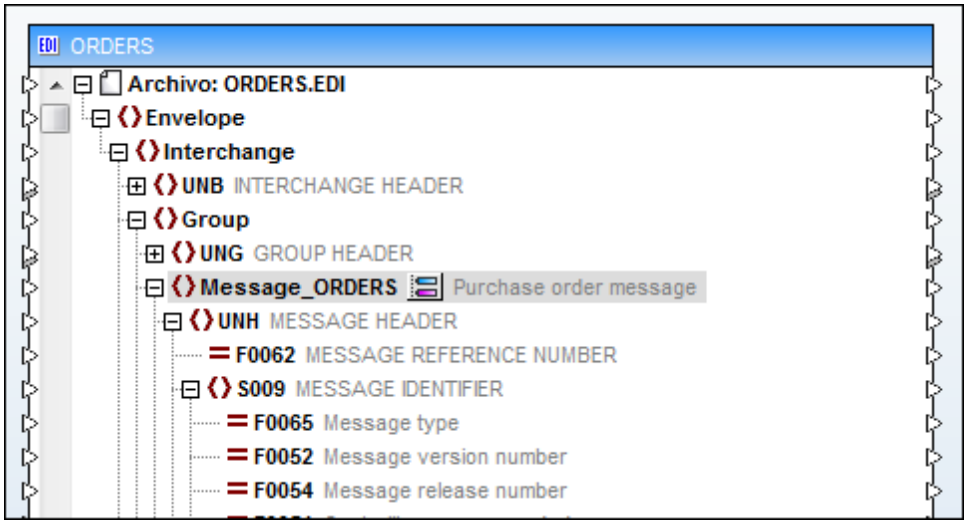
4. El componente EDI aparece ahora en el panel **Asignación**.

 © 2014 Altova GmbH



Con BUILTIN seleccionado

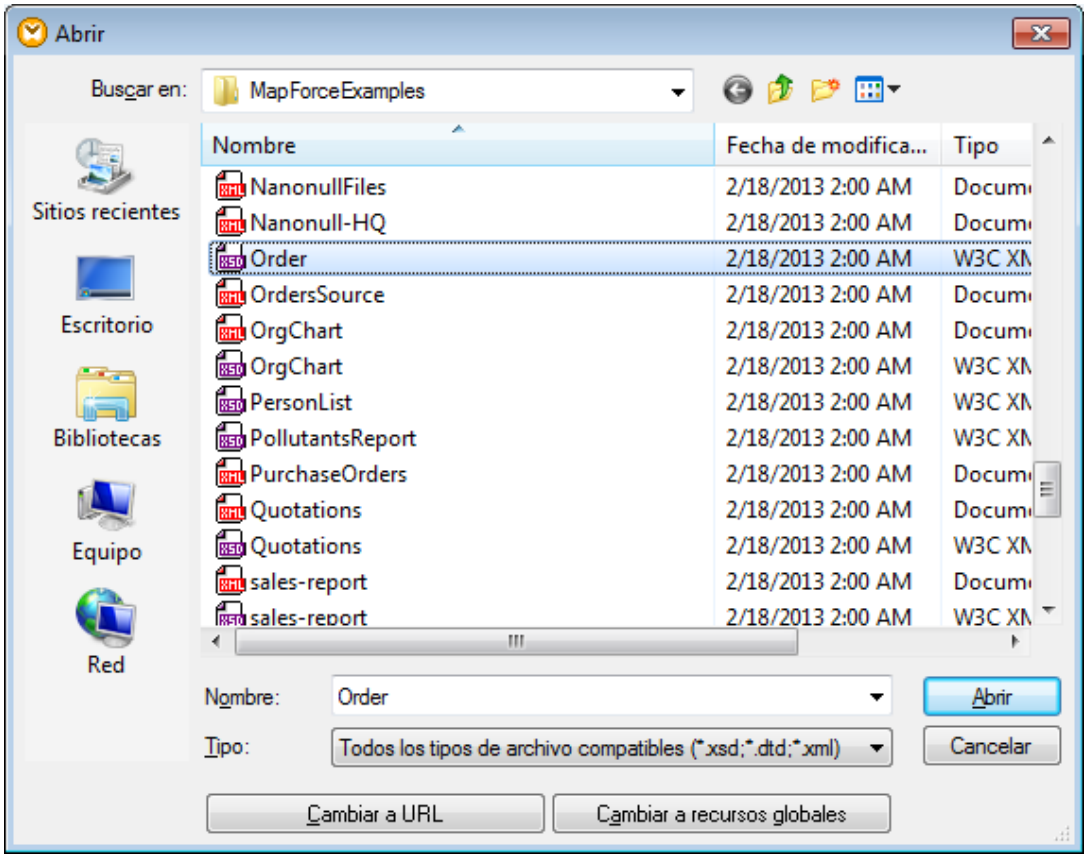
5. Haga clic en la entrada **`Message_ORDERS`** y pulse la tecla ***** del teclado numérico para ver sus elementos.
6. Haga clic en el icono de expansión situado en la esquina inferior derecha del componente para ajustar el tamaño de la ventana.



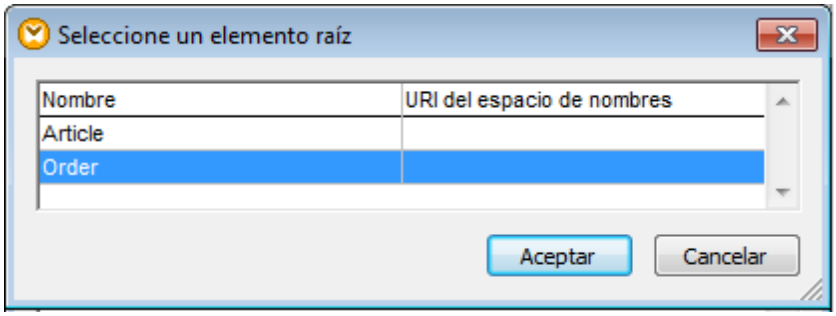
Con BUILTIN seleccionado

Crear el componente de esquema de destino:

1. Haga clic en el icono **Insertar archivo o esquema XML** y seleccione `Order.xsd` en el directorio **`...\MapForceExamples`**.

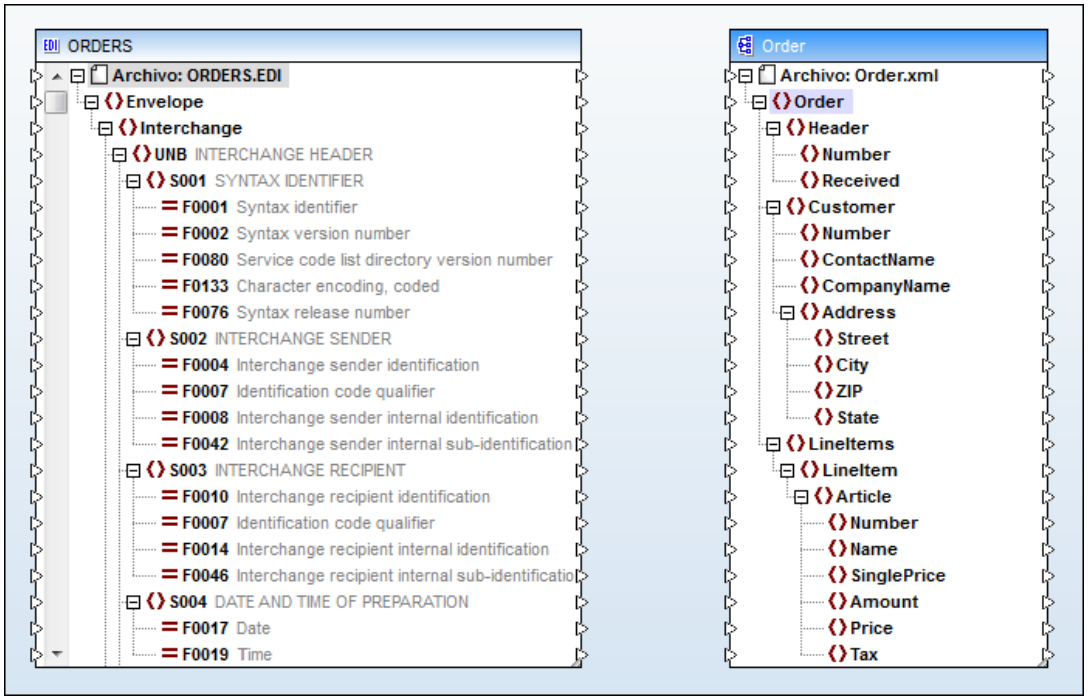


2. En el aviso que aparece haga clic en **Omitir** y seleccione **`Order`** como elemento raíz del documento de destino.



3. Haga clic en el elemento raíz **`Order`** del componente de esquema y pulse la tecla * para ver todos sus elementos.
4. Haga clic en el icono de expansión situado en la esquina inferior derecha del componente para ajustar el tamaño de la ventana.

 © 2014 Altova GmbH



Ahora podemos empezar a asignar los elementos EDI del componente EDI de origen al esquema de destino.

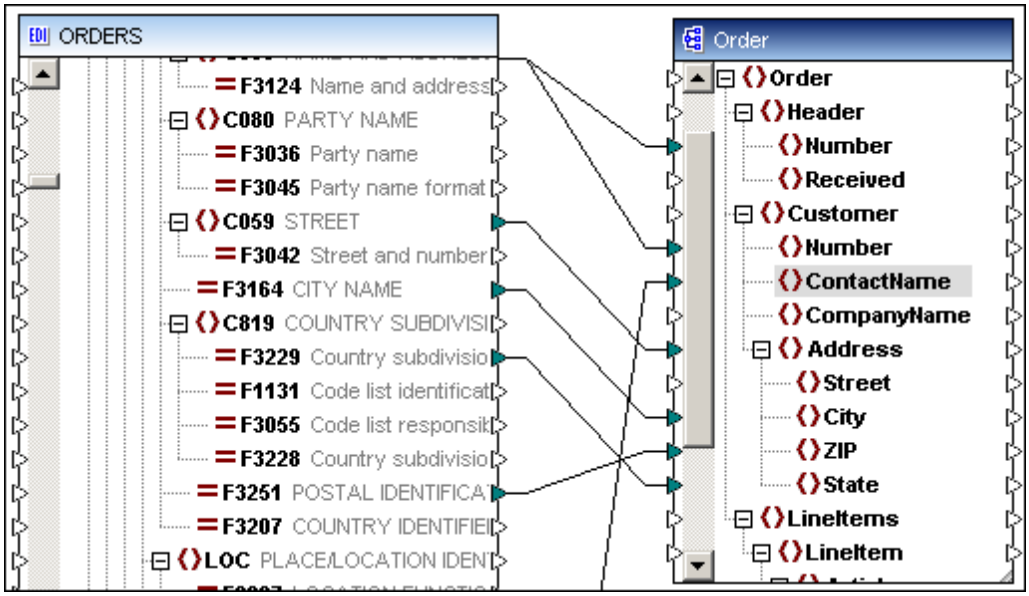
Asignar elementos EDI:

El componente EDI nos muestra la estructura de un mensaje, en base a la colección seleccionada (en este caso, **ORDERS**).

Por lo general, no todos los nodos contienen datos, por lo que el autor del proyecto debe conocer bastante bien los documentos EDI con los que trabaja para localizar los nodos que le interesan. En el caso que nos ocupa, debemos crear las asignaciones que aparecen a continuación:

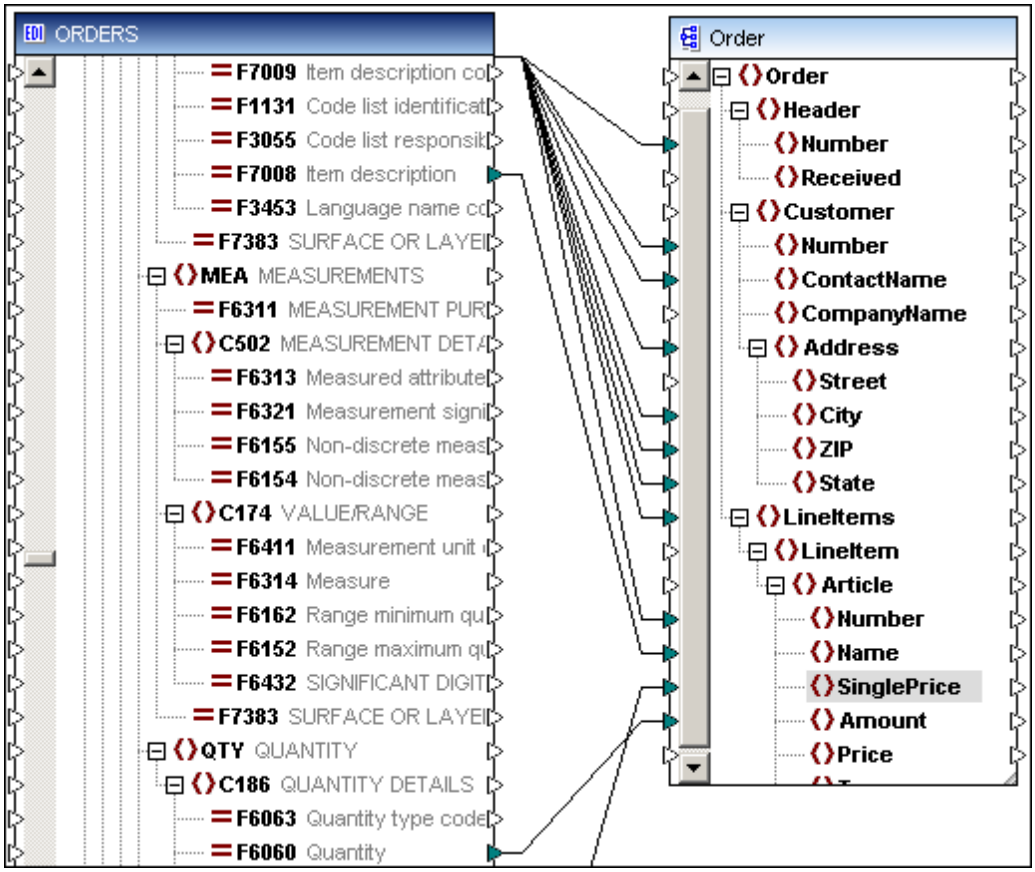
Nodo EDI		**Nodo del esquema XML**
BGM/ C106/ F1004	->	Order / Header / Number
SG2/ NAD/ C082/ F3055	->	Order / Customer / Number
SG2/ NAD/ C080/ F3036	->	Order / Customer / CompanyName
SG2/ NAD/ C059/ F3042	->	Order / Customer / Address/ Street
SG2/ NAD/ F3164	->	Order / Customer / Address/ City
SG2/ 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:



Continúe el proceso de asignación creando estas asignaciones:

SG28	-> Order/LineItems
SG28/LIN/C212/F7140	-> Order/LineItems/LineItem/Article/Number
SG28/IMD/C273/F7008	-> Order/LineItems/LineItem/Article/Name
SG28/QTY/C186/F6060	-> Order/LineItems/LineItem/Article/Amount
SG28/SG32/PRI/C509/F5118	-> Order/LineItems/LineItem/Article/SinglePrice

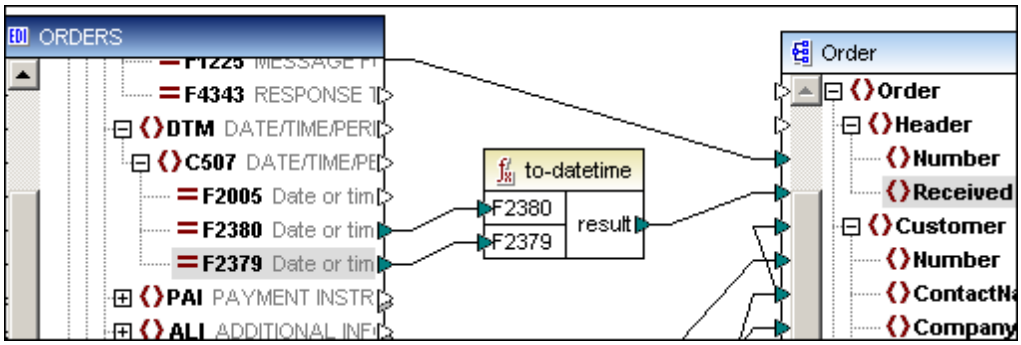


Arrastre una función **`to-datetime`** de la biblioteca **edifact** hasta el panel **Asignación**.

Si aplica los componentes F2380 y F2379 del elemento **DTM/C507**, podrá crear un elemento de tipo `datetime` llamado **`Received`** (recibido) con el formato adecuado.

Por tanto, creamos estas asignaciones:

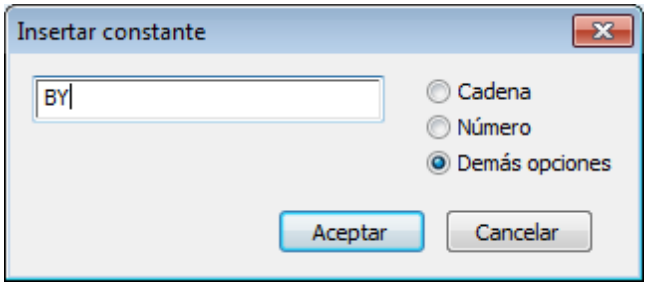
`DTM/C507/F2380`	->	la entrada **`F2380`** de la función **`to-datetime`**
`DTM/C507/F2379`	->	la entrada **`F2379`** de la función **`to-datetime`**
el parámetro **`result`** de la función **`to-datetime`**	->	`Order/Header/Received`



Filtrar las órdenes de compra del comprador (`Buyer`):

Llegados a este punto queremos filtrar las órdenes de compra del comprador (**Buyer**). Estas órdenes de compra se identifican con el calificador de código de función de la parte del segmento NAD (*nombre y dirección*). En este caso, el valor **BY** significa **BUYER** (comprador), es decir, la parte a quien se vende la mercancía o servicio.

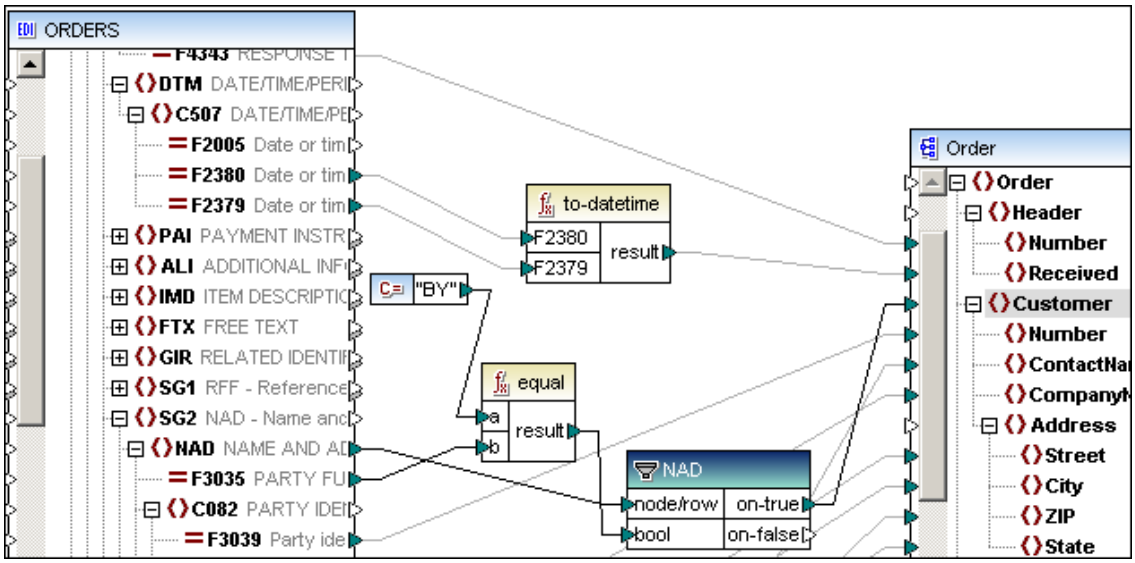
1. Arrastre la función **equal** de la biblioteca **core** hasta el panel **Asignación**.
2. Seleccione la opción de menú **Insertar | Filtro: nodos/filas**.
3. Inserte un componente de constante (**Insertar | Constante**), haga clic en clic en *Demás opciones* y escriba BY en el cuadro de texto:



Cree estas asignaciones en este orden:

SG2/NAD/F3035	->	el parámetro **b** de la función **equal**
la constante **BY**	->	el parámetro **a** de la función **equal**
el parámetro **result** de la función **equal**	->	el parámetro **bool** del **filtro**
SG2/NAD	->	el parámetro **node/row** del **filtro**
el parámetro **result** del filtro	->	**Order/Customer**

El objetivo es asignar los datos solamente si el nodo **NAD** hace referencia al comprador (**Buyer**), por medio del calificador de código de función de la parte **BY**.



El paso final consiste en calcular el precio y los costes fiscales.

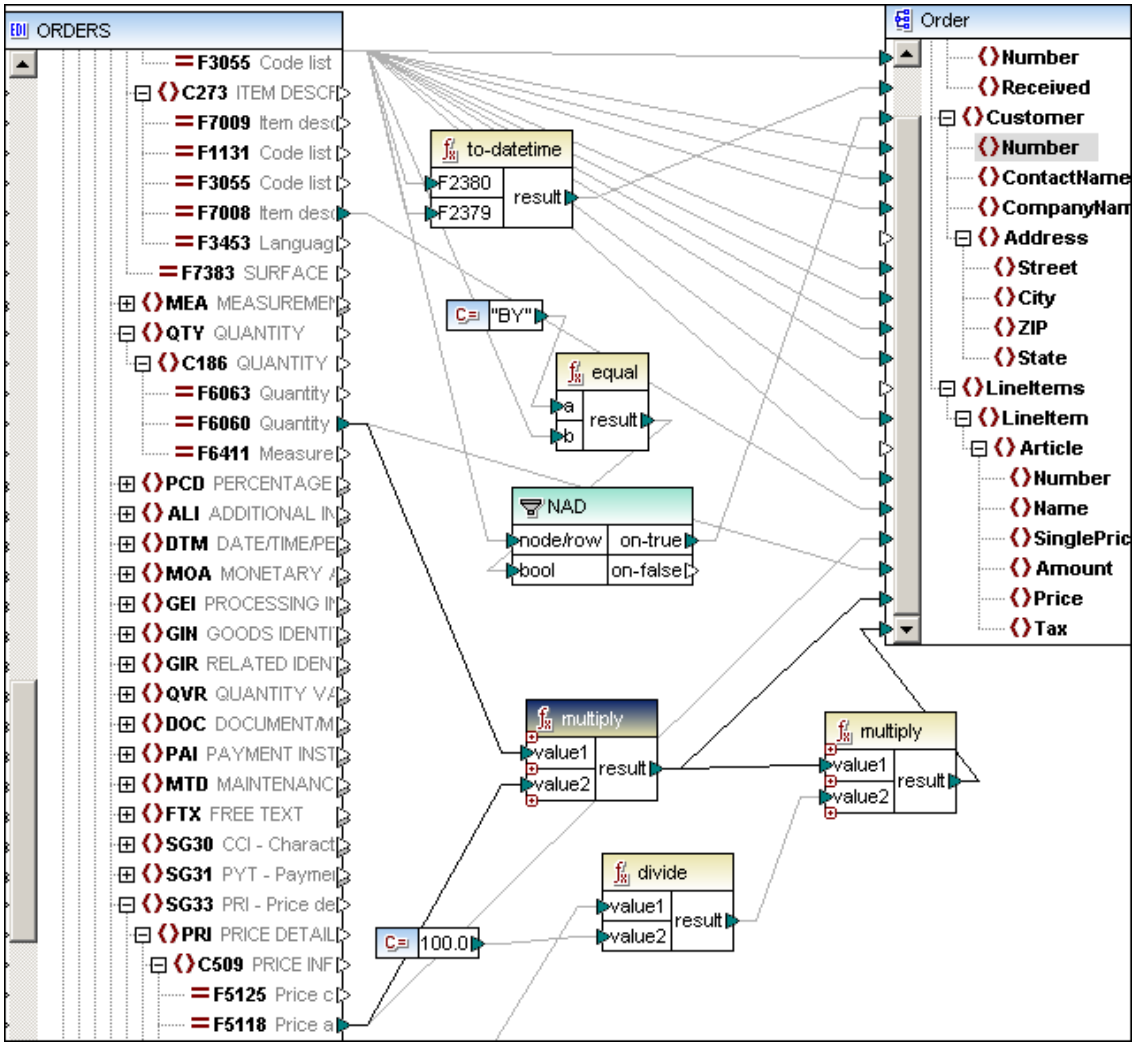
1. Arrastre dos funciones **multiply** y una función **divide** de la biblioteca **core** hasta el panel **Asignación**.

2. Inserte un componente de constante (**Insertar | Constante**), seleccione la opción *Número* y escriba `100.0` en el cuadro de texto.

Cree estas asignaciones en este orden:

`SG28/QTY/C186/F6060`	->	el parámetro **`value1`** de la primera función **`multiply`**
`SG28/SG32/PRI/C509/F5118`	->	el parámetro **`value2`** de la primera función **`multiply`**
el parámetro **`result`** de la primera función **`multiply`**	->	**`Order/LineItems/LineItem/Article/Price`**
`SG28/SG38/TAX/C243/F5278`	->	el parámetro **`value1`** de la función **`divide`**
la constante **`100.0`**	->	el parámetro **`value2`** de la función **`divide`**
el parámetro **`result`** de la primera función **`multiply`**	->	el parámetro **`value1`** de la segunda función **`multiply`**
el parámetro **`result`** de la función **`divide`**	->	el parámetro **`value2`** de la segunda función **`multiply`**
el parámetro **`result`** de la segunda función **`multiply`**	->	**`Order/LineItems/LineItem/Article/Tax`**

Llegados a este punto la asignación tiene este aspecto:



Al hacer clic en la pestaña **Resultados** MapForce lleva a cabo una transformación instantánea y muestra una vista previa del documento XML resultante:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3    <Header>
4      <Number>ABC123456XYZ</Number>
5      <Received>2004-04-30T17:42:00-09:00</Received>
6    </Header>
7    <Customer>
8      <Number>123</Number>
9      <ContactName>Michelle Butler</ContactName>
10     <CompanyName>Nanonull, Inc.</CompanyName>
11     <Address>
12       <Street>119 Oakstreet Suite 4876</Street>
13       <City>Vereno</City>
14       <ZIP>29213</ZIP>
15       <State>CA</State>
16     </Address>
17   </Customer>
18   <LineItems>
19     <LineItem>
20       <Article>
21         <Number>42</Number>
22         <Name>Pizza Pepperoni</Name>
23         <SinglePrice>7.2</SinglePrice>
24         <Amount>1</Amount>
25         <Price>7.2</Price>
26         <Tax>0.648</Tax>
27       </Article>
28     </LineItem>
```

10.5.3 Validación de componentes EDI

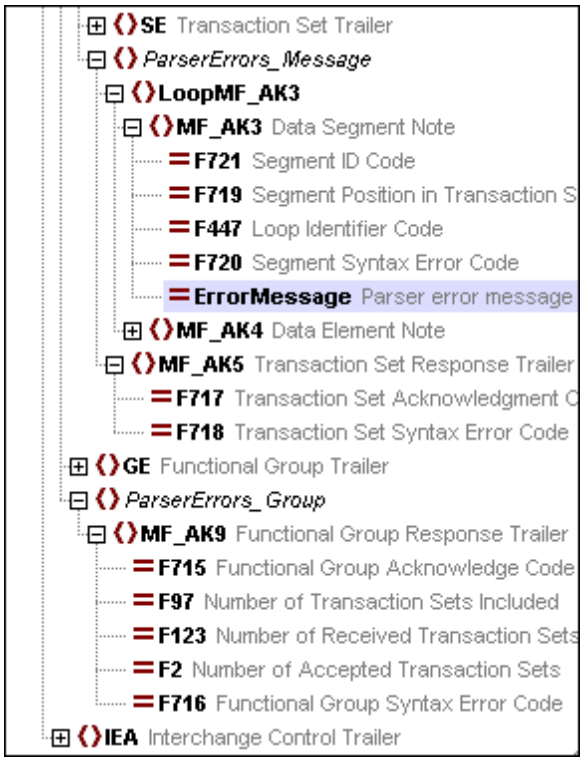
Cuando se ejecuta la asignación (al abrir el panel **Resultados**) MapForce puede validar todos los documentos EDI de **origen** y **destino** que estén en un formato EDI compatible. No obstante, este proceso no es un proceso de validación completo de la sintaxis ni de la semántica EDI. Como ejemplo puede ver el archivo `X12_To_XML_Order.mfd` , disponible en la carpeta `...\MapForceExamples`.

El usuario puede definir los parámetros de validación en función de sus requisitos. También puede elegir qué errores se ignoran, cuándo se detiene el proceso de validación, cuándo se aceptan/rechazan los errores, etc.

Cuando ejecute el código de programa generado, los errores de análisis inician una excepción que detiene la asignación y los errores de validación crean un mensaje en el resultado estándar.

Confirmación X12
Para los componentes EDI de origen, los resultados de la validación (informe) se colocan en elementos virtuales en la base del componente EDI (bajo los nodos ***ParserError_Message*** y ***ParserErrors_Group***). Puede usar estos elementos virtuales cuando genere una confirmación X12 997 o 999.

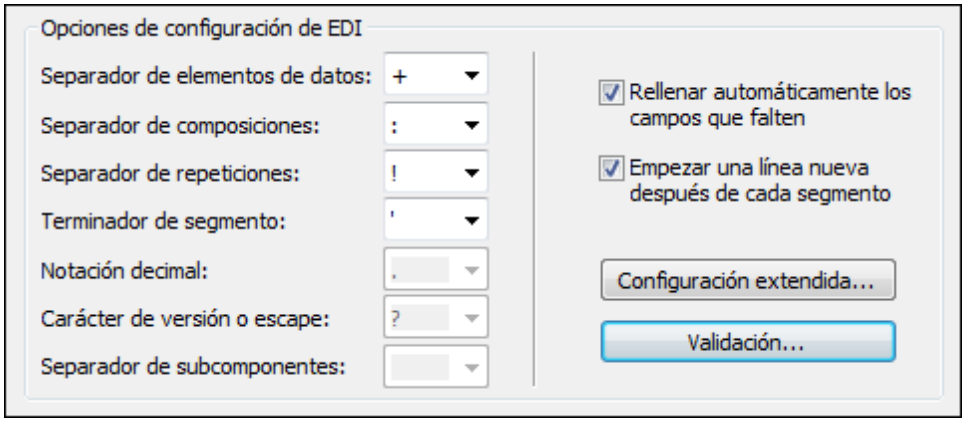
Para crear un archivo de confirmación X12 que informe del estado del intercambio consulte el apartado Confirmación funcional 997 de X12 o Confirmación de implementación 999 de X12.



Para personalizar la configuración de la validación:

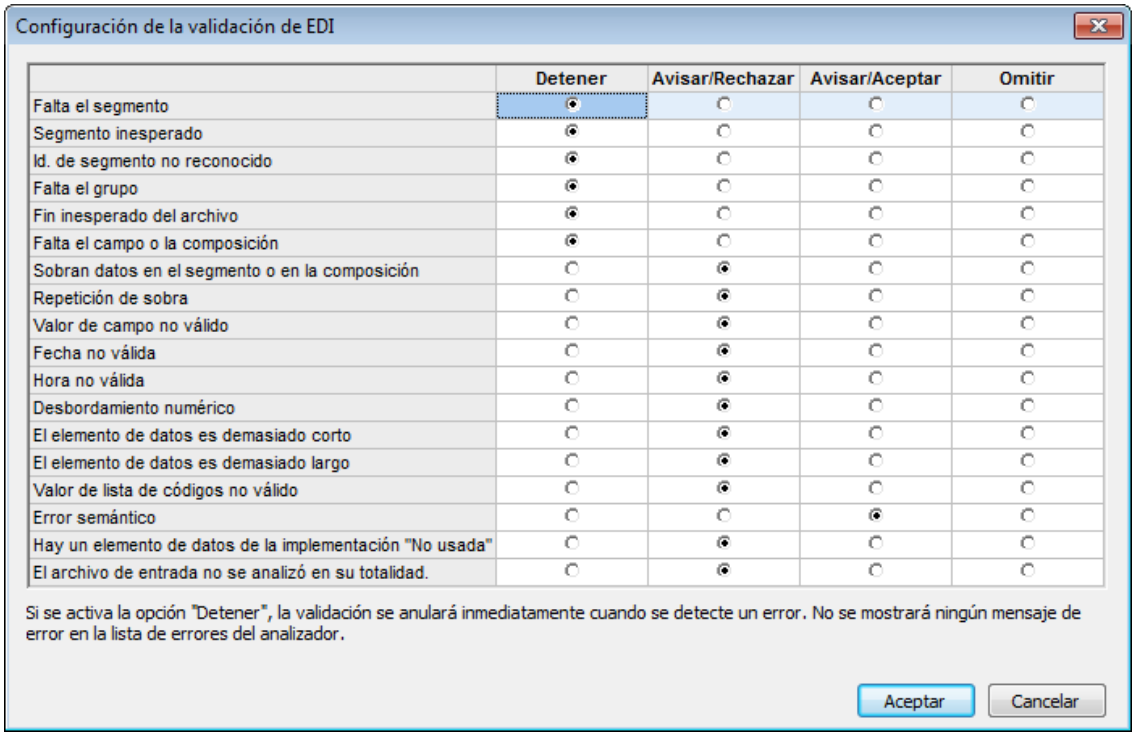
1. Haga doble clic en un componente EDI y después clic en el botón **Validación...** del cuadro de diálogo "Configuración del componente".

 © 2014 Altova GmbH



La configuración predeterminada de la validación aparece en la imagen siguiente.

2. Seleccione las opciones de validación que desea utilizar y haga clic en **Aceptar**.



Utilice la opción **Detener** para detectar errores graves y detener la ejecución de la asignación sin generar un mensaje.

Utilice la opción **Avisar/Rechazar** y **Avisar/Aceptar** para enviar información a los elementos **`Parser_Errors_Message`** y **`Parser_Errors_Group`** del componente EDI. Estos elementos se pueden asignar después a los demás componentes.

Dependiendo de la opción elegida, los elementos **`Functional Group Acknowledge Code F715`** y **`Transaction Set Acknowledgment Code F717`** contienen:

el valor **`R`** (de *Rechazar*) o

el valor **`E`** (de *Aceptar*, pero con errores). Los errores también aparecen en la ventana Mensajes.

Utilice la opción **Omitir** para pasar por alto el error. Los elementos **`Parser_Errors_Message`** y **`Parser_Errors_Group`** no incluyen información sobre el error.

Este es el significado de los mensajes de error:

Nombre	Descripción	El error puede ocurrir 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/ 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/ 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/ destino
El elemento de datos es demasiado largo	La longitud de un elemento de datos es mayor que el máximo especificado.	componente de origen/ destino
Valor de lista de códigos no válido	Un valor de lista de códigos no coincide con el conjunto de valores especificado.	componente de origen/ destino
Error semántico	Se produjo un error semántico.	componente de origen/ destino
Hay un elemento de datos de la implementación "No usada"		

 © 2014 Altova GmbH

El archivo de entrada no se analizó en su totalidad.		

Una vez creadas las asignaciones con el componente de destino, abra el panel **Resultados** para ver el resultado de la asignación.

Si existen errores en el documento EDI de origen o destino, cuando ejecute la asignación los errores se señalan en la ventana Mensajes. El hecho de que exista un error en el EDI de origen no significa necesariamente que el proceso de asignación falle. Puede que MapForce consiga generar resultados en el panel **Resultados**.

10.5.4 X12: Confirmación funcional 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.

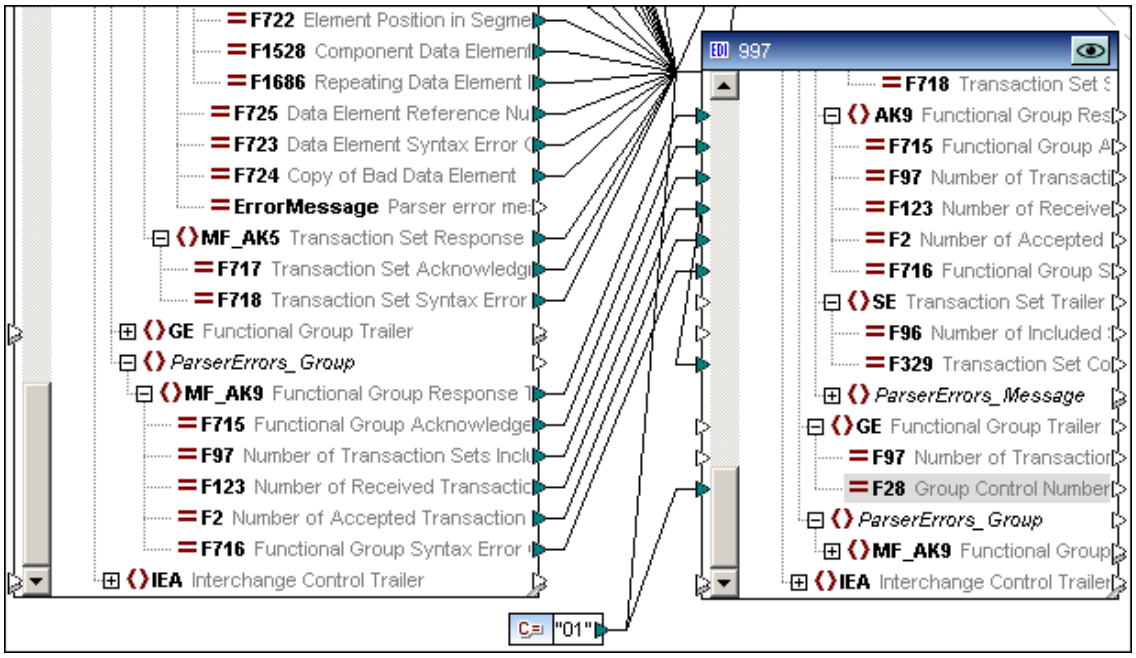
Como ejemplo de ello puede ver el archivo `X12_To_XML_Order.mfd` disponible en la carpeta `...\MapForceExamples`.

Consulte el tema Confirmación 997 de X12 para ver los segmentos de error de la confirmación y qué significan.

Recuerde que MapForce también puede generar confirmaciones de implementación 999 de X12.

Para generar la confirmación funcional 997 de X12:

1. Haga clic con el botón secundario en el componente EDI de origen y seleccione **Crear asignación a EDI x12 997**.



   MapForce crea un componente EDI 997 y crea automáticamente todas las conexiones necesarias para generar la confirmación 997 de X12.

2. Haga clic en el botón **Vista previa** del componente EDI y después abra el panel **Resultados** para ver el archivo de confirmación resultante.

```
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 997:

- Si está generando código, escriba un nombre de archivo en el campo *Archivo EDI de salida* del cuadro de diálogo "Configuración del componente"
- Haga clic en el icono **Guardar resultado generado** en el panel **Resultados** (introduzca la ruta de acceso y el nombre de archivo).

10.5.5 X12: Confirmación de implementación 999

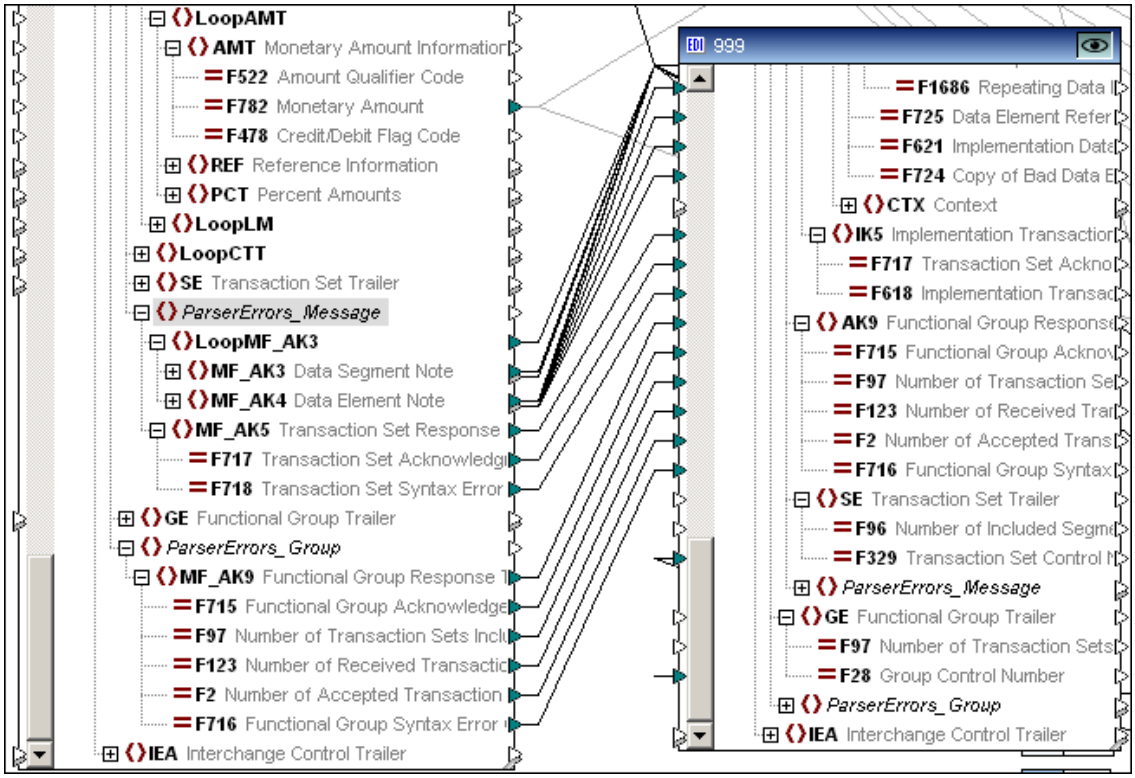
El conjunto de transacciones de confirmación de implementación 999 de X12 informa de errores de aplicación o errores de incumplimiento de la guía de implementación X12.

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 un componente 999 de X12 en el panel **Asignación** y crear las conexiones necesarias.

Para generar la confirmación de implementación EDI 999:

1. Haga clic con el botón secundario en el componente EDI y seleccione **Crear asignación a EDI X12 999**.



MapForce crea un componente EDI 999 y crea automáticamente las asignaciones de datos necesarias para generar la confirmación de implementación 999 de X12.

2. Haga clic en el botón **Vista previa** del componente EDI 999 y después abra el panel **Resultados** para ver el archivo de confirmación resultante.

 © 2014 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:

- Si está generando código, escriba un nombre de archivo en el campo *Archivo EDI de salida* del cuadro de diálogo "Configuración del componente"
- Haga clic en el icono **Guardar resultado generado** en el panel **Resultados** (introduzca la ruta de acceso y el nombre de archivo).

10.5.6 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.

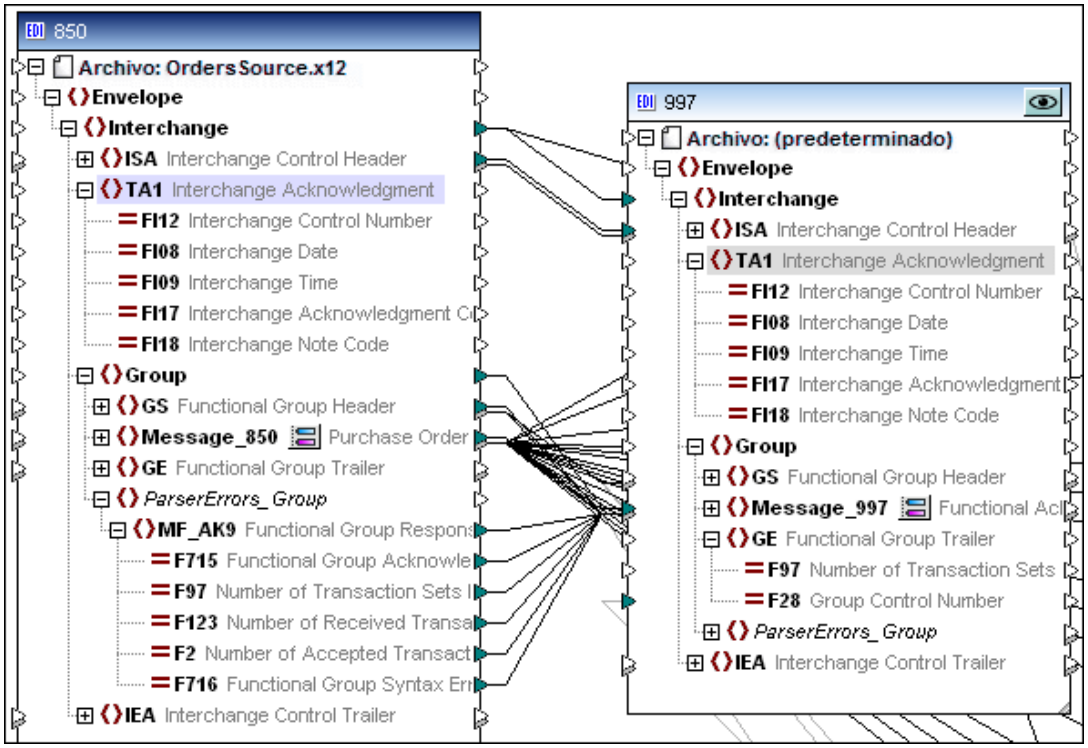
Para incluir el segmento TA1 en un componente EDI:

1. Busque la carpeta que contiene el archivo `Envelope.Config` (disponible en la carpeta `c:\Archivos de programa\Altova\MapForce2014\MapForceEDI\X12`) y edite el archivo.

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Config xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:nol
3     <Meta>
4       <Release>6020</Release>
5       <Agency>X12</Agency>
6     </Meta>
7     <Format standard="X12"/>
8     <Include href="X12.Segment"/>
9     <Include href="X12.Codelist"/>
10    <Include collection="EDI.Collection"/>
11    <Group name="Envelope">
12      <Group name="Interchange" maxOccurs="unbounded">
13        <Segment ref="ISA" minOccurs="0"/>
14        <Segment ref="TA1" minOccurs="0" maxOccurs="unbounded"/>
15        <Group name="Group" maxOccurs="unbounded">
```

2. Añada una referencia al segmento TA1 añadiendo la línea `<Segment ref="TA1" minOccurs="0" maxOccurs="unbounded"/>` entre los segmentos `ISA` y `Group` y guarde el archivo.
3. Abra el archivo `X12_To_XML_Order.mfd` de la carpeta `MapForceExamples` para ver que el segmento se añadió al componente de orden de compra **850** y al componente de confirmación funcional **997**. Ahora puede elegir los elementos de TA1 que desea asignar al componente de destino.

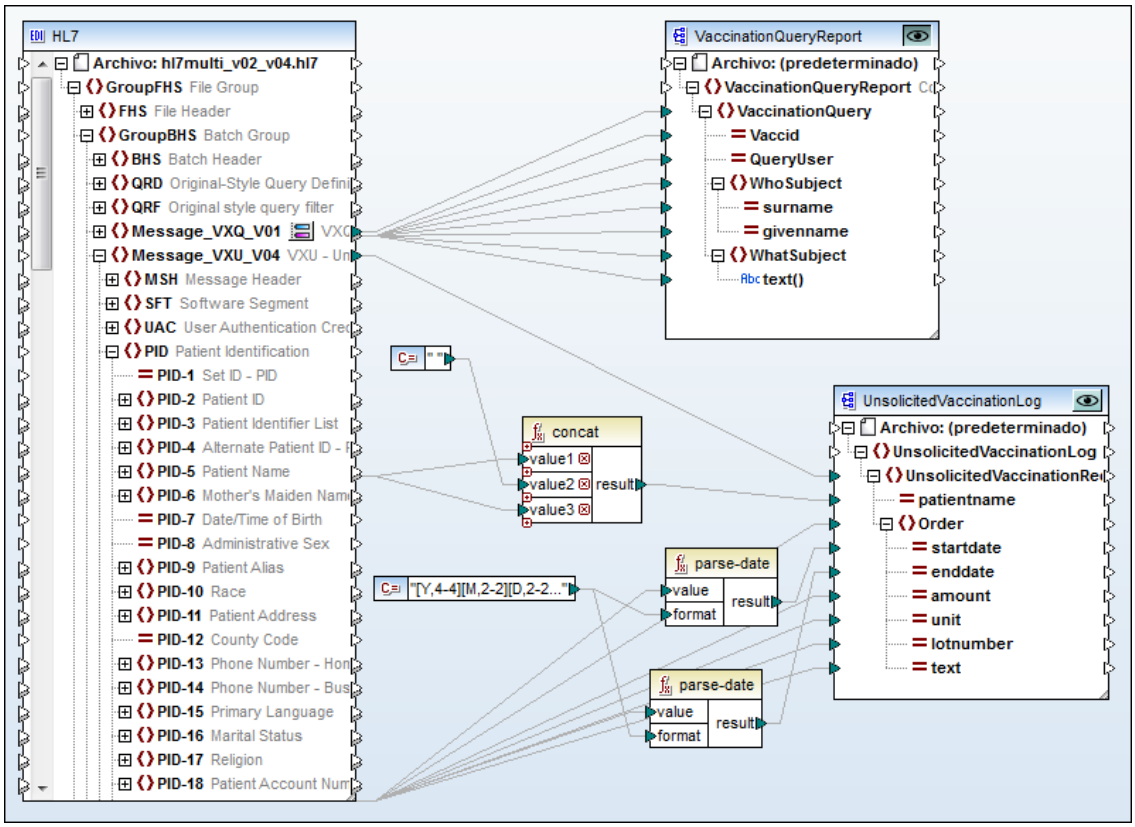
 © 2014 Altova GmbH


850
Archivo: OrdersSource.x12
Envelope
Interchange
ISA Interchange Control Header
TA1 Interchange Acknowledgment
FI12 Interchange Control Number
FI08 Interchange Date
FI09 Interchange Time
FI17 Interchange Acknowledgment Cl
FI18 Interchange Note Code
Group
GS Functional Group Header
Message_850 Purchase Order
GE Functional Group Trailer
ParserErrors_Group
MF_AK9 Functional Group Respons
F715 Functional Group Acknowle
F97 Number of Transaction Sets I
F123 Number of Received Transa
F2 Number of Accepted Transact
F716 Functional Group Syntax Er
IEA Interchange Control Trailer
997
Archivo: (predeterminado)
Envelope
Interchange
ISA Interchange Control Header
TA1 Interchange Acknowledgment
FI12 Interchange Control Number
FI08 Interchange Date
FI09 Interchange Time
FI17 Interchange Acknowledgment
FI18 Interchange Note Code
Group
GS Functional Group Header
Message_997 Functional Acl
GE Functional Group Trailer
F97 Number of Transaction Sets
F28 Group Control Number
ParserErrors_Group
IEA Interchange Control Trailer

10.5.7 Varios mensajes EDI por componente

MapForce ya puede procesar varios tipos de mensajes EDI por componente (de una misma versión). Esto significa que ahora puede:

- procesar como instancias de entrada varios archivos que contengan mensajes diferentes
- procesar mensajes de tipo diferente dentro del mismo archivo (es decir, dentro del intercambio)

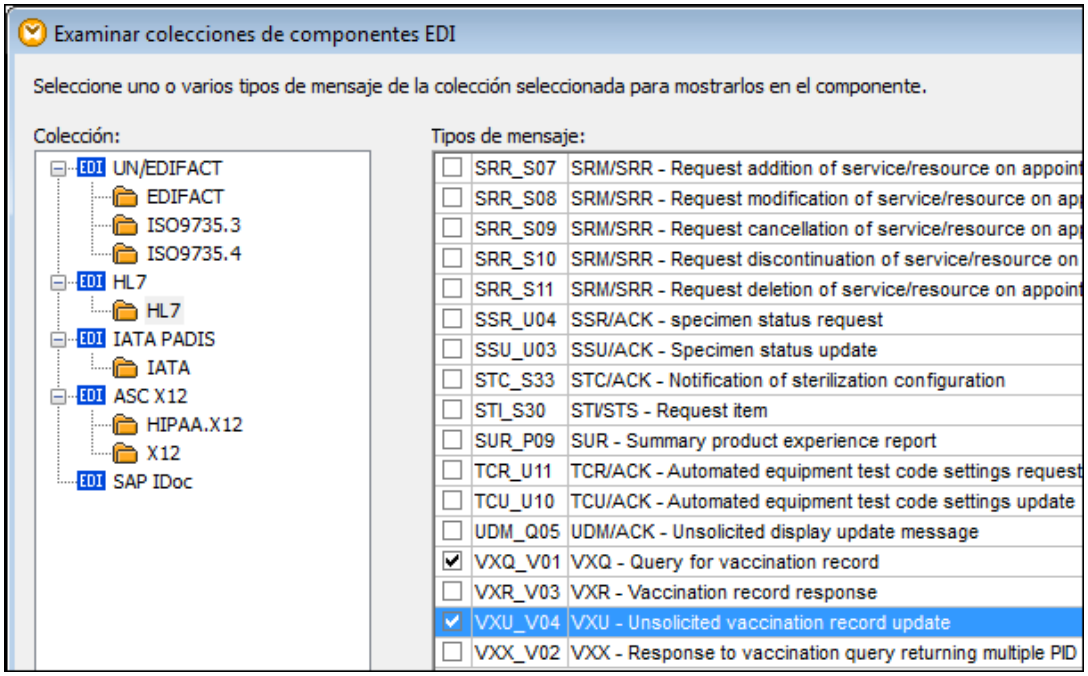


Nota: el icono **Seleccionar mensaje EDI** del componente EDI abre el cuadro de diálogo "Examinar colecciones de componentes EDI", donde puede seleccionar varios mensajes por cada intercambio (por cada componente EDI).

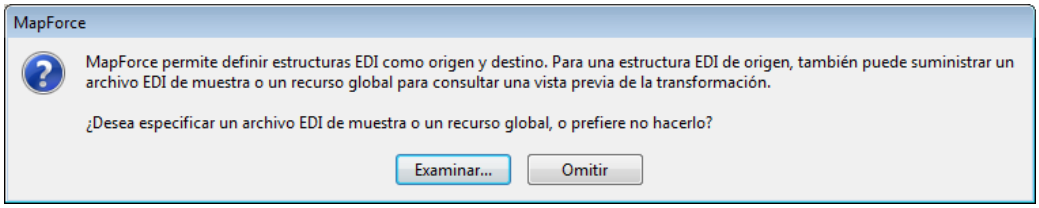
El ejemplo de la imagen anterior muestra la asignación `HL7_MultiMessageTypes.mfd` de la carpeta `...\MapForceExamples`.

Para crear un componente EDI que incluya varios tipos de mensaje diferentes:

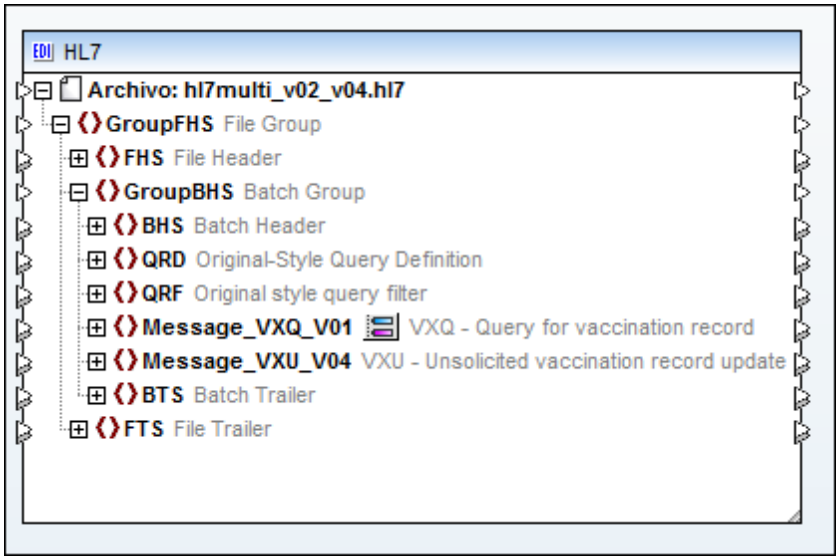
1. Cree una asignación nueva, comprobando que la opción activa es **Java**, **C#**, **C++** o **BUILTIN**. MapForce no puede generar código XSLT 1.0/2.0 ni XQuery para asignaciones de datos EDI.
2. Seleccione la opción de menú **Insertar | EDI** o haga clic en el icono **EDI**. Aparece el cuadro de diálogo "Examinar colecciones de componentes EDI". Seleccione la colección **HL7 | HL7**.



3. Haga clic en los tipos de mensaje que deben aparecer en el componente (p. ej. **`VXU_V01`** y **`VXU_V04`**) y después haga clic en **Aceptar**.
   Aparece un aviso pidiendo un archivo EDI de muestra.



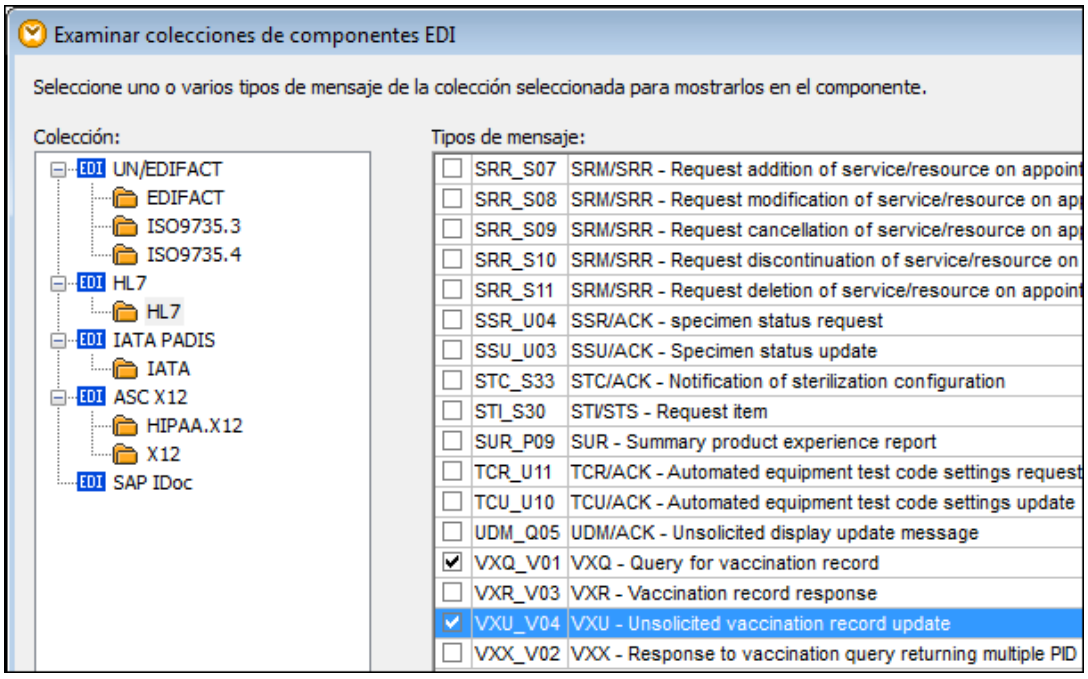
4. Haga clic en el botón **Examinar** y seleccione el archivo EDI que debe aportar los datos a la asignación (p. ej. `hl7multi_v02_v04.hl7`) y haga clic en **Abrir**.
   Se abre el cuadro de diálogo "Configuración del componente", donde puede cambiar la configuración del archivo o de la codificación.
5. Haga clic en **Aceptar** para usar la configuración predeterminada e insertar el componente.



Los dos mensajes que seleccionó en el cuadro de diálogo "Examinar colecciones de componentes EDI" aparecen ahora como elementos del grupo **`GroupBHS`** (es decir, **`Message_VXQ_V01`** y **`Message_VXQ_V04`**) del componente EDI.

Para cambiar el número de mensajes del componente:

1. Haga clic en el icono **Seleccionar mensaje EDI** que aparece junto al primer mensaje del componente.
   Se abre el cuadro de diálogo "Examinar colecciones de componentes EDI".
2. Active/desactive las casillas de los mensajes que desea insertar en el componente y después haga clic en **Aceptar**.



 © 2014 Altova GmbH

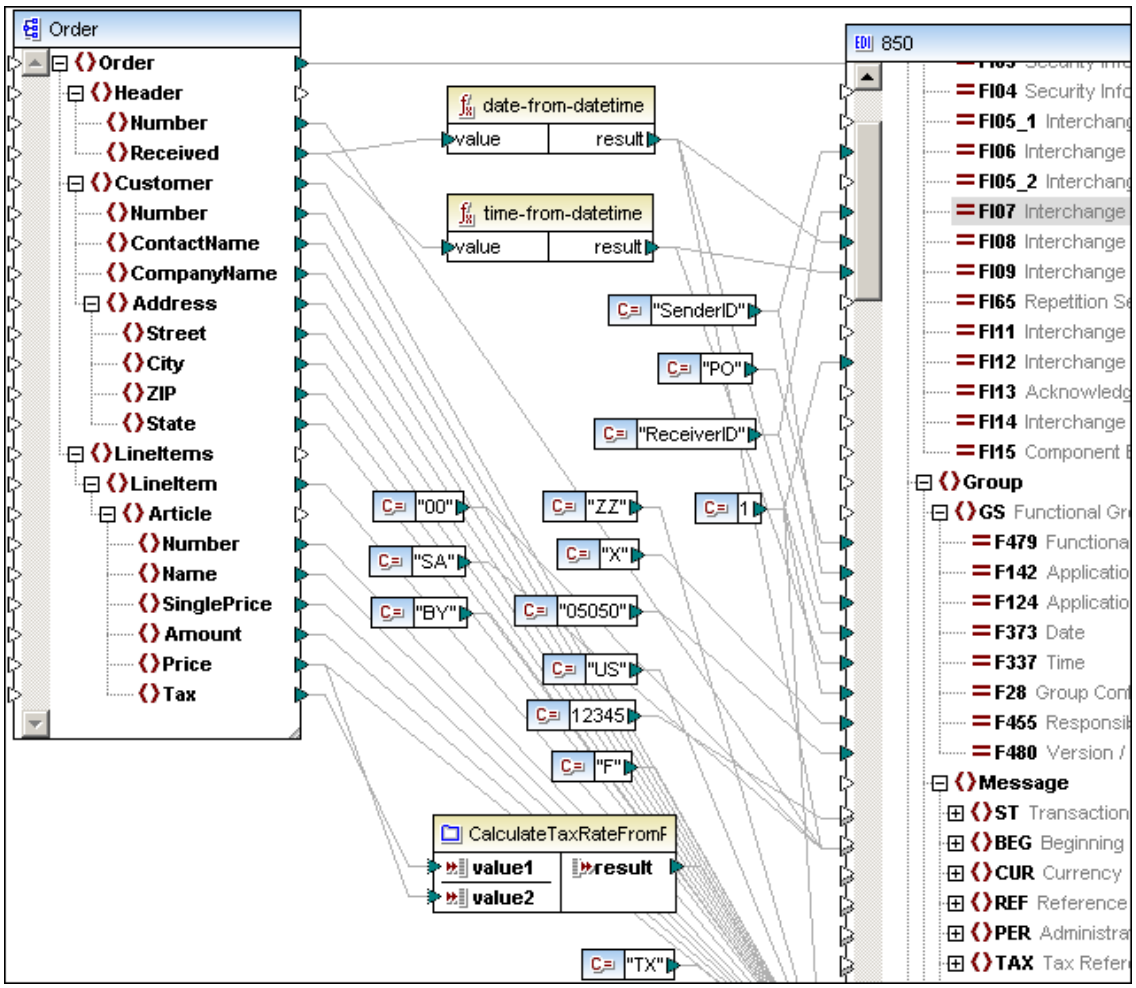
10.5.8 UN/EDIFACT y ANSI X12 como destino

En la asignación `XML_To_X12_Order.mfd` de la carpeta `...\MapForceExamples` se asignan datos de un archivo XML a un archivo ANSI X12 EDI. El archivo EDI resultante se puede validar con la especificación X12 haciendo clic en el icono **Validar archivo de salida**.

En los apartados siguientes describimos:

- las opciones de configuración más frecuentes en UN/EDIFACT y ANSI X12
- las reglas de validación y la configuración de finalización automática utilizadas para los formatos EDI.

Nota: es importante tener presente la secuencia en que aparecen los elementos en el componente de origen. Por ello, recuerde crear conectores usando la opción **Basada en el origen (contenido mixto)**. Para más información consulte el apartado Asignación basada en el origen (contenido mixto).



Finalización automática de datos

La finalización automática de datos no suele cambiar los datos ya existentes. Sin embargo, crea (determinados) nodos obligatorios e inserta datos allá donde sea necesario después del proceso de asignación (o de lo contrario el documento de salida no será válido).

Tenga en cuenta que en los apartados siguientes, los campos que no aparecen en la sección ***Finalización automática de datos*** ni se insertan ni se crean. Los valores correctos no se pueden determinar automáticamente.

Configuración genérica:
Puede definir varias opciones de configuración para todos los documentos EDI:

- *Rellenar automáticamente los campos que falten:*
  debe habilitarse o no la finalización automática. Predeterminado: **true**
- *Empezar una línea nueva después de cada segmento*
  debe anexarse una línea nueva o no a cada segmento para una mejor lectura de los datos. El estándar EDI omite estas líneas si están en un mensaje. Predeterminado: **true**
- *Separador de elementos de datos* (ver la especificación EDIFACT/X12).
  Predeterminado: **+**
- *Separador de composiciones* (ver la especificación EDIFACT/X12).
  Predeterminado **:**
- Terminador de segmento (ver la especificación EDIFACT/X12).
  Predeterminado **'**
- Notación decimal (ver la especificación EDIFACT/X12).
  Predeterminado **.**
- Carácter de versión (ver la especificación EDIFACT/X12).
  Predeterminado **?**

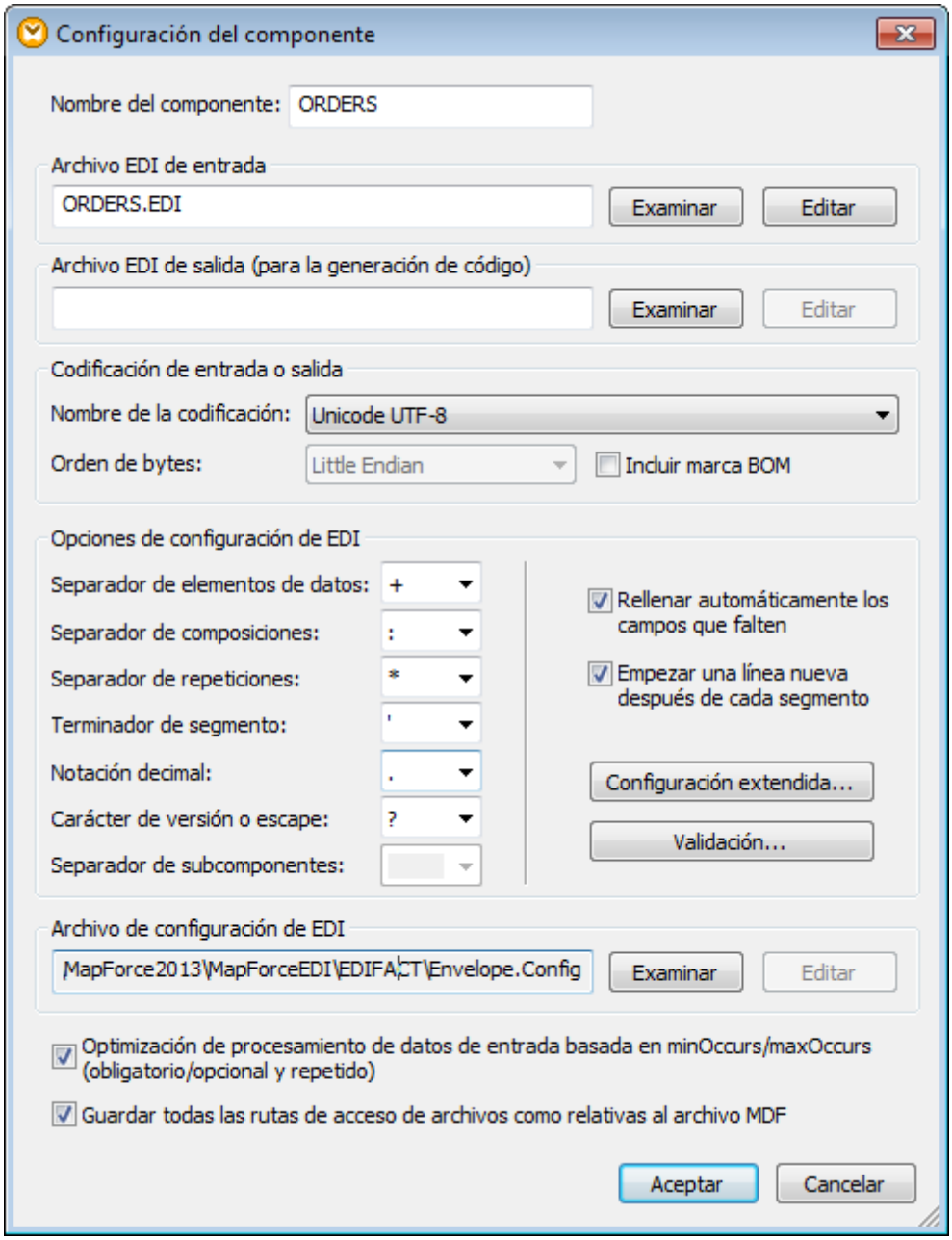
- Cuando el separador predeterminado es &, HL7 permite usar un separador de sub subcampos adicional. Este separador se denomina *Separador de subcomponentes* en el cuadro de diálogo "Configuración del componente".

  Haga clic con el botón secundario en el componente EDI y seleccione **Propiedades** para abrir el cuadro de diálogo "Configuración del componente". La configuración UN/EDIFACT se usa tanto para componentes UN/EDIFACT como para componentes ANSI X12.

Prioridad de los separadores al leer/escribir archivos EDI
Los separadores EDI introducidos en este cuadro de diálogo siempre surten efecto cuando se **escriben** los archivos EDI. Cuando se **leen** archivos EDI los separadores solamente surten efecto si el archivo de entrada no contiene sus propios separadores (p. ej. los archivos EDIFACT en los que falta el segmento UNA "service string advice").

Si un archivo/componente de entrada EDI contiene definiciones de separadores (p. ej. un archivo X12 que contiene un segmento ISA), los separadores ya existentes reemplazan a los separadores definidos en el cuadro de diálogo "Configuración del componente".

 © 2014 Altova GmbH



Haga clic en el botón **Configuración extendida...** para definir opciones de configuración avanzadas.

Definir caracteres no imprimibles:
Puede usar caracteres no imprimibles como separadores si en uno de los cuadros combinados escribe `x` seguido del código hexadecimal de carácter ASCII (p. ej. `x1e` para el carácter de control RS: separador de registros ASCII, código decimal 30).

Asignación de tipos de datos de fecha y hora
Antes de la versión 2006R3 de MapForce, la fecha y hora de los componentes EDIFACT y X12 eran de tipo `string`. A partir de la versión 2006R3 los elementos de fecha y hora se pueden asignar directamente a elementos **`xsd:time`** o **`xsd:date`** (o elementos `date/time` SQL).

Las funciones definidas por el usuario creadas en versiones anteriores a 2006R3 que convierten **`xsd:date`** o **`xsd:time`** en un formato EDI generan un mensaje de error y no se pueden usar sin antes modificarlas. Utilice el tipo de datos **`xsd:dateTime`**, que se puede extraer con las funciones integradas **`date-from-datetime`** y **`time-from-datetime`** de la biblioteca **`datetime`**.

Si necesita asignar fechas dentro de funciones definidas por el usuario, compruebe primero que se ajustan al formato de fecha y hora ISO 8601 (es decir, AAAA-MM-DD).

Validación
La función de validación EDI es compatible con el panel **Resultados** (BUILTIN) y en el código generado por MapForce. En el panel **Resultados** seleccione el comando **Resultados | Validar archivo de salida** para validar el archivo EDI de salida.

Cuando ejecute el código de programa generado, los errores de análisis inician una excepción que detiene la asignación y los errores de validación crean un mensaje en el resultado estándar.

Validación de componentes de destino UN/EDIFACT

Cuando se valida un documento UN/EDIFACT se comprueba:

- si existen o no los segmentos UNB y UNZ.
- si UNB/S004 contiene una especificación válida de fecha y hora.
- 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.

La **finalización automática de datos** para 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).

- 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 sección (***Finalización automática de datos)*** 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:

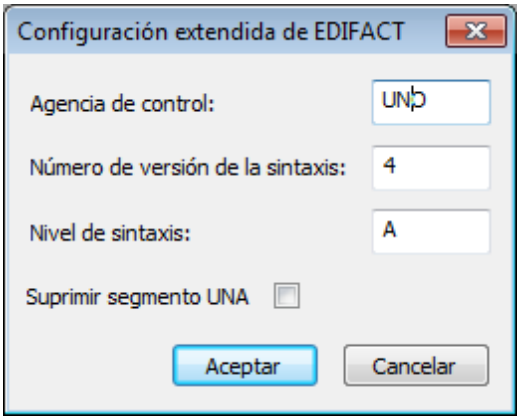
- si UNG/0038 existe. Si no contiene datos, se inserta el nombre del mensaje.
- si UNG/S006 y UNG/S007 existen.
- si UNG/S004 existe. Si no contiene datos, se inserta la fecha/hora actual en formato EDI.
- si UNG/0051 existe. Si no contiene datos, se insertan los dos primeros caracteres de la agencia de control definida por el usuario. (Botón **Configuración extendida | Agencia de control**).
- 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**).
- 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)

Configuración extendida

Haga clic en el botón **Configuración extendida...** para definir opciones avanzadas de configuración para el componente. Para más información consulte la especificación UN/EDIFACT.



- *Agencia de control:* UNO (predeterminado)
- *Número de versión de la sintaxis:* 4 (predeterminado)
- *Nivel de sintaxis:* 4 (predeterminado)
- *Suprimir segmento UNA:* opción desactivada por defecto

Validación de componentes de destino ANSI X12

Cuando se valida un documento ANSI X12 se 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.
- 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.

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**.)
- 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**.)
- 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 (**Configuración del componente | Separador de composiciones**.)
- 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 sección (***Finalización automática de datos***) 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:

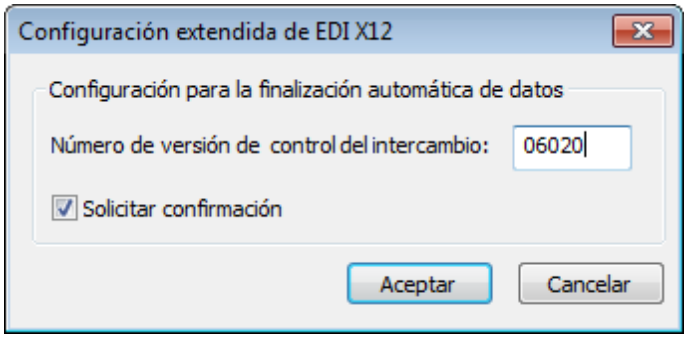
- si existe GS/373. Si no contiene datos, se inserta la fecha actual en formato EDI.
- si existe GS/337. Si no contiene datos, se inserta la hora actual en formato EDI.
- si existe GE/97. Si no contiene datos, se calcula e inserta el número de mensajes del grupo funcional.
- si existe GE/28. Si no contiene 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.

Configuración extendida
Haga clic en el botón **Configuración extendida...** para definir las opciones avanzadas de configuración para el componente. Para más información consulte la especificación EDI/X12.



- *Número de versión de control del intercambio:* 06020 (predeterminado)
- *Solicitar información:* opción activada por defecto

 © 2014 Altova GmbH

10.5.9 Personalizar un mensaje EDIFACT

MapForce permite personalizar mensajes EDI para que reconozcan formatos EDIFACT no estándar o modificados.

Para el ejemplo siguiente usamos el archivo `Orders-Custom-EDI.mfd` disponible en la carpeta `...\MapForceExamples\Tutorial`. En esta carpeta también está el archivo ZIP `EDIFACT.Nanonull.zip`, que contiene el resultado final del proceso de personalización.

El archivo `Orders-Custom.EDI` de la carpeta `...\MapForceExamples\Tutorial` se modificó para que incluya un elemento componente nuevo en el segmento CTA:

- La línea 9 contiene la entrada `Mr` y
- La línea 11 contiene la entrada `Mrs`.

```
 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'
```

El segmento CTA (información de contacto) del mensaje **`ORDERS`** debe ampliarse para que sea posible asignar los datos nuevos.

Configurar el ejemplo de personalización

Este apartado describe la personalización de archivos de configuración con el objetivo de poder asignar el mensaje EDIFACT modificado al esquema `ORDER-EDI.xsd`. Para ello se asume que el archivo ZIP que viene con MapForce no se está utilizando.

Configurar el ejemplo de personalización:

1. Bajo `Archivos de programa...\MapForceEDI\EDIFACT` cree la carpeta `EDIFACT.Nanonull`.
2. Copie estos archivos de la carpeta `...\MapForceEDI\EDIFACT` a la carpeta `EDIFACT.Nanonull` recién creada:

   ```
   Admin.Segment
   EDI.Collection
   EDSD.Segment
   Envelope.Config
   ORDERS.Config
   UNCL.Codelist
   ```

3. Cambie los atributos de los archivos a **lectura-escritura** para que se puedan editar.

Configurar el archivo `EDI.Collection`:

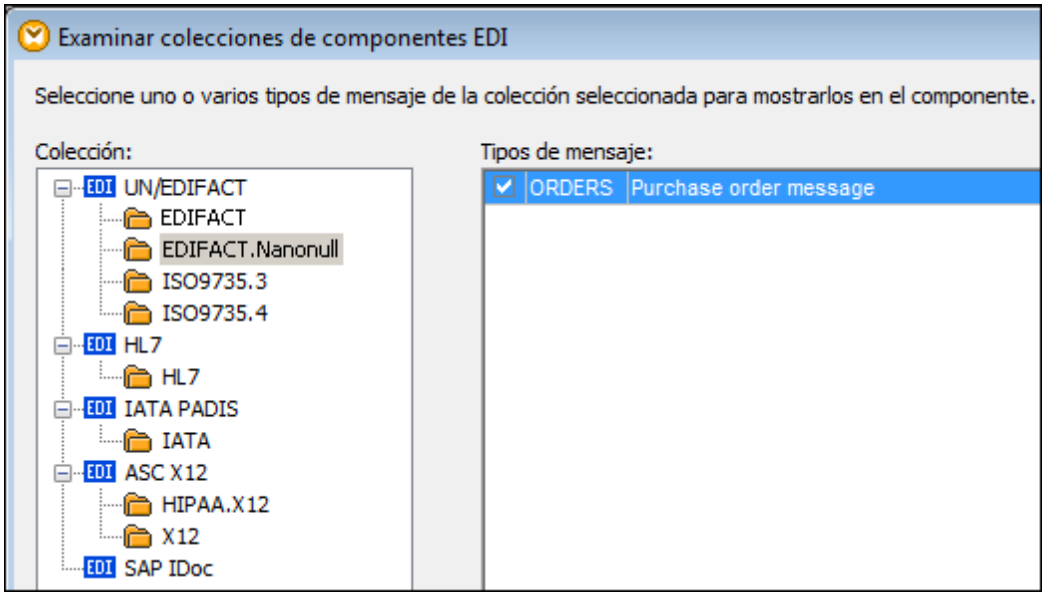
1. Abra el archivo `EDI.Collection` en XMLSpy o en otro editor.
2. Elimine todos los elementos `Message`, excepto el elemento `Message` llamado `ORDERS`. Compruebe que conserva las etiquetas `<Messages>`.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Messages Version="3">
3    <Meta>
4      <Version>D</Version>
5      <Release>10A</Release>
6      <Agency>UN</Agency>
7    </Meta>
8    <Root File="Envelope.Config"/>
9    <Message Type="ORDERS" File="ORDERS.Config" Description="Purchase order message"/>
10 </Messages>
```

3. Guarde el archivo.

Para ver el contenido del archivo de colecciones:

- Inicie MapForce y seleccione **Insertar | EDI** o haga clic en el icono **Insertar EDI**. 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 `EDIFACT.Nanonull`. Si hace clic en esta carpeta, observará que solamente incluye una entrada, el tipo de mensaje `ORDERS`.



Nota: 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 `ORDERS`.

Este apartado explica cómo redefinir el segmento CTA (información de contactos) añadiendo el campo X1000 para que esté a disposición de los distintos mensajes. CTA está formado por el campo F3139 y la composición C056.

Hay varias maneras de personalizar el segmento:

- **De forma global**, personalizando el archivo `EDSD.segment`. Todos los segmentos, de

todos los mensajes que usan la composición C056, contendrán/harán referencia al elemento nuevo.

- **De forma local**, personalizando el archivo `ORDERS.Config`. Solamente el segmento CTA personalizado del mensaje actual contendrá el elemento nuevo.

UNCL.Codelist
Este archivo define los códigos EDIFACT y los valores que pueden incluir y se utiliza para validar los archivos de entrada/salida de MapForce. Si su organización utiliza un código especial que no aparece en la lista de códigos EDIFACT, añádalo en este archivo.

EDI.Collection
Este archivo enumera todos los mensajes del directorio actual y ofrece una lista de los mensajes disponibles cuando se inserta un archivo EDIFACT en MapForce. Este ejemplo contiene solamente un mensaje, el mensaje `ORDERS`.

Edite el archivo y añada todos los mensajes que necesita.

ORDERS.Config
Este es el archivo de configuración para los archivos de mensaje `Purchase order`. Este archivo contiene todos los grupos y definiciones de segmentos utilizados en los mensajes `Orders`. Los cambios realizados en este archivo pueden definir personalizaciones locales o inline.

Admin.Segment
Este archivo describe los segmentos *Interchange level administrative*. Se utilizan para analizar el archivo EDIFACT.

EDSD.Segment (Electronic Data Segment Definition)
Este archivo define el nombre de los segmentos, de los campos y de las composiciones de los archivos EDIFACT y se utiliza para analizar el archivo EDIFACT. Los cambios realizados en este archivo son personalizaciones globales y afectan a todos los segmentos y mensajes.

Personalización global

- Para poder acceder al nuevo campo X1000 de forma global solamente es necesario realizar cambios en el archivo `EDSD.Segment`
- Todos los **segmentos** de todos los **mensajes** que usen la composición C056 contendrán/harán referencia al elemento nuevo.

Redefinición de composición en el archivo `EDSD.Segment`:
Abra el archivo `EDSD.Segment` en XMLSpy o en otro editor y navegue hasta **`Config | Elements | Composite | C056`**.

```
<Composite name="C056" info="DEPARTMENT OR EMPLOYEE DETAILS">
    <Data ref="F3413" minOccurs="0"/>
    <Data ref="F3412" minOccurs="0"/>
</Composite>
```

Inserte esta línea en el segmento C056, justo debajo de F3412:

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
```

La definición de la composición tiene este aspecto:

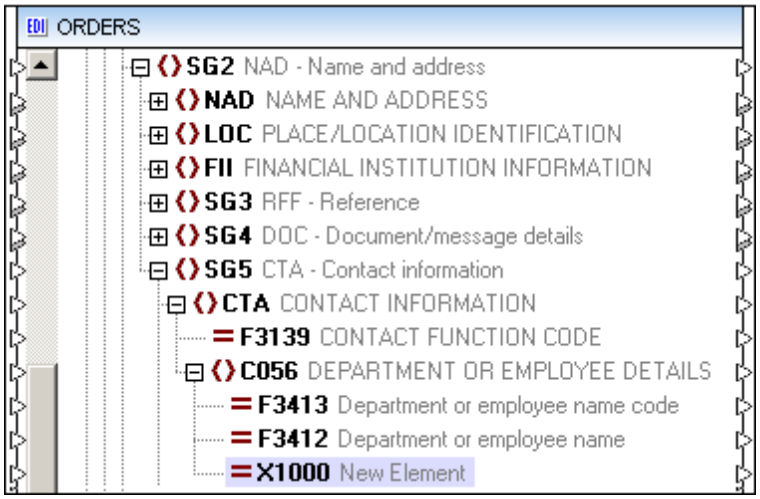
```
<Composite name="C056" info="DEPARTMENT OR EMPLOYEE DETAILS">
    <Data ref="F3413" minOccurs="0"/>
    <Data ref="F3412" minOccurs="0"/>
    <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
</Composite>
```

Nota: el campo X1000 nuevo se define usando el atributo `name`, al contrario de los otros campos del segmento, que usan el atributo `ref`. Los otros dos campos se definen al principio del archivo `EDSD-Segment`, fuera de la sección `Composite` (usando el elemento `Data name`) y solamente se les hace referencia aquí. Al campo nuevo se le puede hacer referencia desde diferentes elementos `Segment` o `Composite`.

Cuando personalice sus archivos EDI, los valores que podrá usar para, por ejemplo, el atributo `type`, serán por lo general todos los tipos de XML Schema que se utilizan en los archivos de configuración que vienen con MapForce. Algunos tipos simples se pueden utilizar, pero no en todos los casos. Además, los tipos simples `anyType`, `ENTITIES` y `QName` no son compatibles.

Vista previa del campo nuevo en MapForce:

1. Seleccione la opción de menú **Insertar | EDI** o haga clic en el icono **Insertar EDI** .
2. En el cuadro de diálogo "Examinar colecciones" seleccione la carpeta `EDIFACT.Nanonull` y el mensaje **`ORDERS`**.
3. En el aviso que aparece haga clic en el botón **Omitir**.
   El componente **`ORDERS`** aparece ahora en el panel **Asignación**.
4. Navegue hasta el elemento **`Envelope/Interchange/Group/Message_ORDERS/SG2/SG5/CTA/C056`** para ver el elemento X1000 nuevo.



Personalización local

- Para tener acceso local al campo X1000 nuevo solamente hace falta realizar cambios en el archivo `ORDERS.config`.
- Solamente el segmento CTA redefinido del mensaje **actual** contendrá/hará referencia al elemento nuevo.
- En otras palabras, el segmento CTA se redefine localmente para que contenga la composición C056 redefinida, con la definición local del campo X1000 nuevo.

Abra el archivo `ORDERS.Config` en XMLSpy o en otro editor y navegue hasta el elemento **`Message`**

| Envelope | Interchange | Group | Message | SG2 | SG5 | CTA (o busque **SG5**).

```
47    <Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact information">
48        <Segment ref="CTA"/>
49        <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
50    </Group>
```

Reemplace la línea
```
<Segment ref="CTA"/>
```

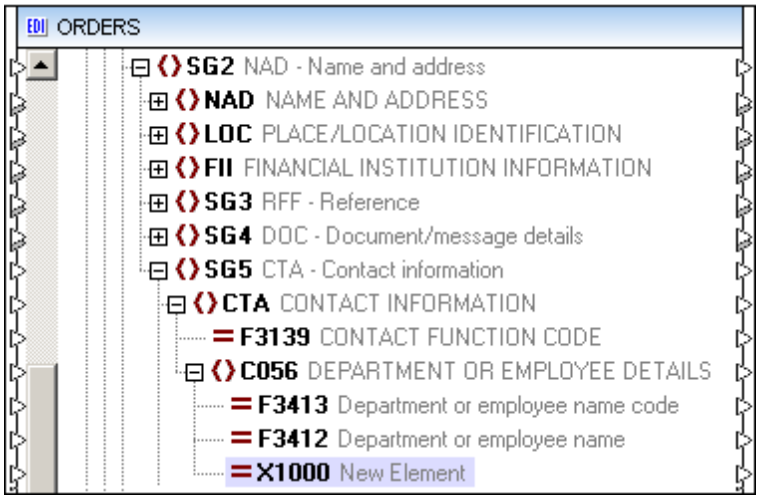
con estas líneas:

```
<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>
```

```
47  <Group name="SG5" minOccurs="0" maxOccurs="5" info="CTA - Contact information">
48      <Segment name="CTA" id="CTA_ORDERS_SG5" info="CONTACT INFORMATION">
49      <Data ref="F3139" minOccurs="0"/>
50          <Composite name="C056" minOccurs="0" info="DEPARTMENT OR EMPLOYEE DETAILS">
51          <Data ref="F3413" minOccurs="0"/>
52          <Data ref="F3412" minOccurs="0"/>
53          <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Ele
54      </Composite>
55  </Segment>
56  <Segment ref="COM" minOccurs="0" maxOccurs="5"/>
57  </Group>
```

Vista previa del campo nuevo en MapForce:

1. Seleccione la opción de menú **Insertar | EDI** o haga clic en el icono **Insertar EDI** .
2. En el cuadro de diálogo "Examinar colecciones" seleccione la carpeta `EDIFACT.Nanonull` y el mensaje **ORDERS**.
3. En el aviso que aparece haga clic en el botón **Omitir**.
   El componente **ORDERS** aparece ahora en el panel **Asignación**.
4. Navegue hasta el elemento **Envelope/Interchange/Group/Message_ORDERS/SG2/SG5/CTA/C056** para ver el elemento X1000 nuevo.



Ejemplo

La asignación del ejemplo que aparece a continuación (`Orders-Custom-EDI.mfd`) está disponible en el directorio `...\MapForceExamples\Tutorial`.

En este ejemplo se asignan datos del archivo `ORDERS-Custom.EDI` al esquema `Order-EDI`. El campo que se añadió a la estructura EDI, **`X1000`**, se asignó al elemento **`Salutation`**.

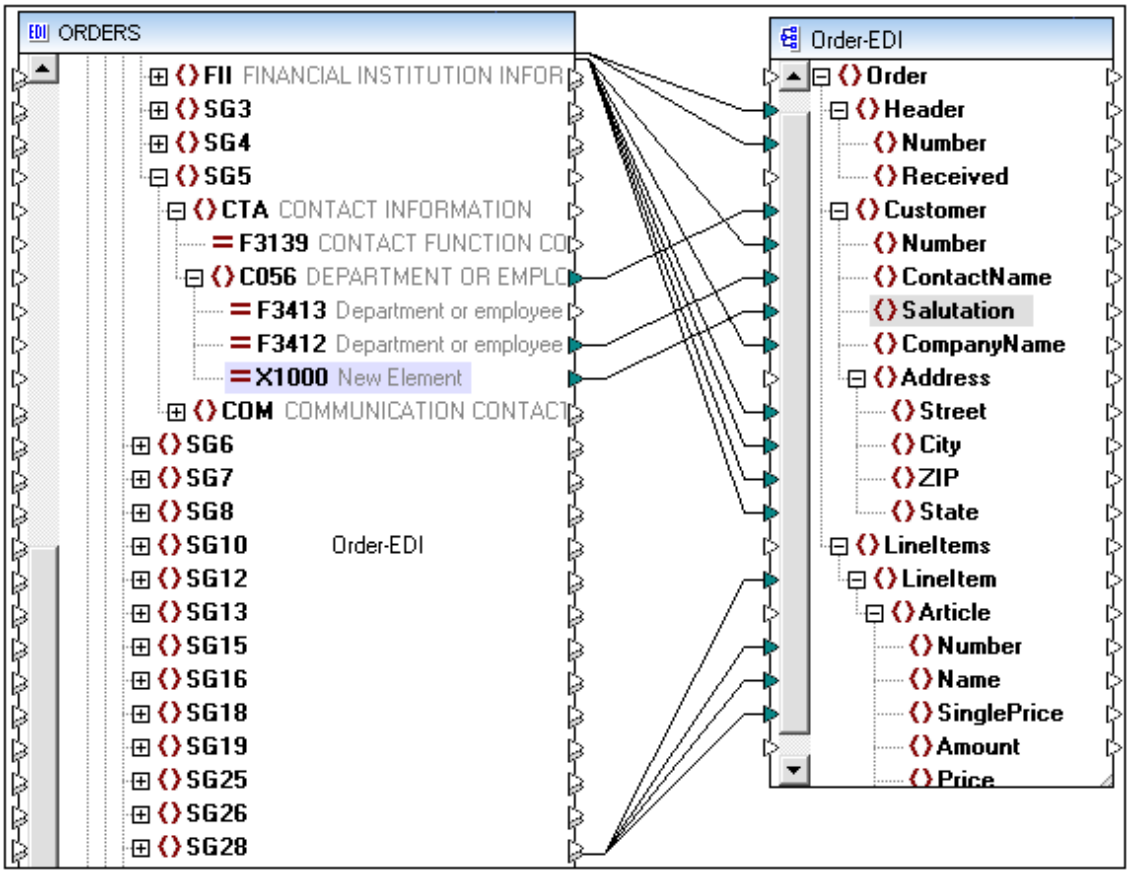
Para ver el resultado de la personalización:

1. Cree una carpeta nueva bajo el directorio `...\MapForceEDI` y llámela `EDIFACT.Nanonull`.
2. Descomprima el archivo ZIP `EDIFACT.Nanonull.zip` (disponible en la carpeta `...\Tutorial`) en la carpeta nueva. El archivo ZIP contiene estos archivos:

   ```
   Admin.Segment
   EDI.Collection
   EDSD.Segment
   Orders.config
   UNCL.Codelist
   ```

3. Abra el archivo `Orders-Custom-EDI.mfd` y abra el panel **Resultados** para ver el resultado de la asignación.

 © 2014 Altova GmbH



Este es el resultado de la asignación.

```
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>
```

Nota sobre la generación de código:
Cuando se genera código C++, se genera también una clase llamada `CX1000Type`, a la que se

puede acceder desde la clase `CC056Type`.

© 2014 Altova GmbH

10.5.10 Personalizar transacciones ANSI X12

MapForce permite personalizar transacciones X12 para que reconozcan formatos X12 no estándar o modificados.

Este apartado usa como ejemplo el archivo `Orders-Custom-X12.mfd` disponible en la carpeta `...\MapForceExamples\Tutorial`. El archivo ZIP contiene el resultado final del proceso de personalización.

El archivo de origen X12 (Orders-Custom.X12) está en la carpeta `...\Tutorial` y se modificó para que incluya un campo nuevo en el segmento N2:

- 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'
```

Nota: los datos de personalización para EDIFACT y ANSI X12 vienen dados por dos archivos de estructuras equivalentes que permiten utilizar varios elementos consecutivos del mismo nombre en cada formato EDI. EDIFACT utiliza el archivo **EDSD.segment** mientras que ANSI X12 utiliza el archivo **X12.Segment**. Para más información consulte el apartado Varios elementos consecutivos.

Archivos de origen X12 y varios campos idénticos

Cuando se personaliza una transacción X12 es importante tener en cuenta que los segmentos suelen permitir el uso de varios **elementos consecutivos** del mismo **nombre**.

Varios elementos consecutivos del mismo nombre

En este apartado asignaremos un archivo ANSI X12 que está personalizado a un archivo de esquema XML. Al segmento N2 se le añadió el campo nuevo **`Mrs`**. Este campo de datos se asignará al elemento **`Salutation`** del esquema XML.

LOOP ID - N1		
3100	N1	Party Identification
3200	N2	Additional Name Information
3250	IN2	Individual Name Structure Components
3300	N3	Party Location
3400	N4	Geographic Location
3450	NX2	Location ID Component
3500	REF	Reference Information
3600	PER	Administrative Communications Contact

Al añadir el campo nuevo al archivo X12, también es necesario añadir un carácter separador (el carácter +) seguido de los datos. La especificación del segmento N2 (*imagen siguiente*) permite el uso de dos campos consecutivos llamados `Name`.

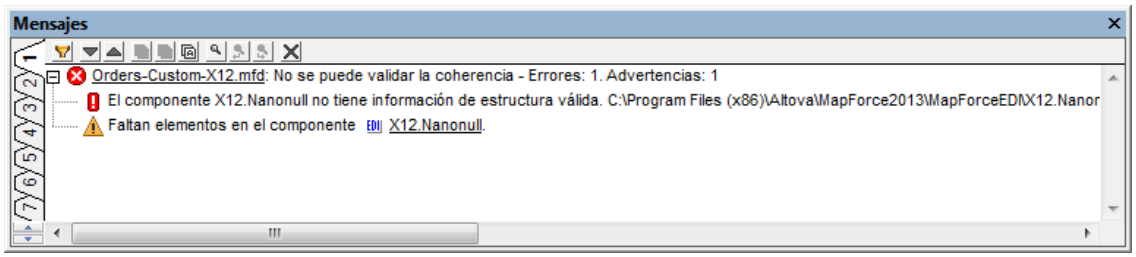
REF	ELE ID	NAME	RPT	ATTRIBUTES	
01	93	Name	M	AN	1/60
02	93	Name	O	AN	1/60

El archivo de origen X12, por tanto, debe tener esto en cuenta añadiendo un separador de campos vacío para la primera aparición (obligatoria) y otro separador para separar los datos reales. Es decir, para cumplir la especificación X12 se introduce **`++Mrs`**.

Puede buscar los campos que tienen varias entradas en el archivo **X12.Segment** que viene con MapForce y que está disponible en la carpeta `...MapForceEDI\X12`. En los segmentos que pueden tener varios campos idénticos (p. ej. el segmento 93) el atributo `mergedEntries` define cuántas veces puede aparecer el campo (en este caso 2 veces).

```
<Segment name="N2" info="Additional Name Information">
    <Data ref="F93" mergedEntries="2"/>
</Segment>
```

Si el archivo X12 tiene un solo separador + cuando se inserta el componente EDI con el archivo EDI asignado, aparece un mensaje de advertencia en la ventana Mensajes:



Configurar el ejemplo de personalización

En los apartados siguientes se describe cómo personalizar los archivos de configuración para permitir la asignación entre la transacción X12 modificada y el esquema `ORDER-x12.xsd`. En estos apartados se asume que el archivo ZIP que viene con MapForce no está en uso.

Configurar el ejemplo de personalización:

1. Bajo `Archivos de programa...\MapForceEDI\X12` cree la carpeta `X12.Nanonull`.
2. Copie estos archivos de la carpeta `...\MapForceEDI\X12` a la carpeta `X12.Nanonull` recién creada:

```
EDI.Collection
Envelope.Config
850.Config
X12.Segment
X12.Codelist
```

3. Cambie los atributos de los archivos a **lectura-escritura** para que se puedan editar.

Configurar el archivo `EDI.Collection`:

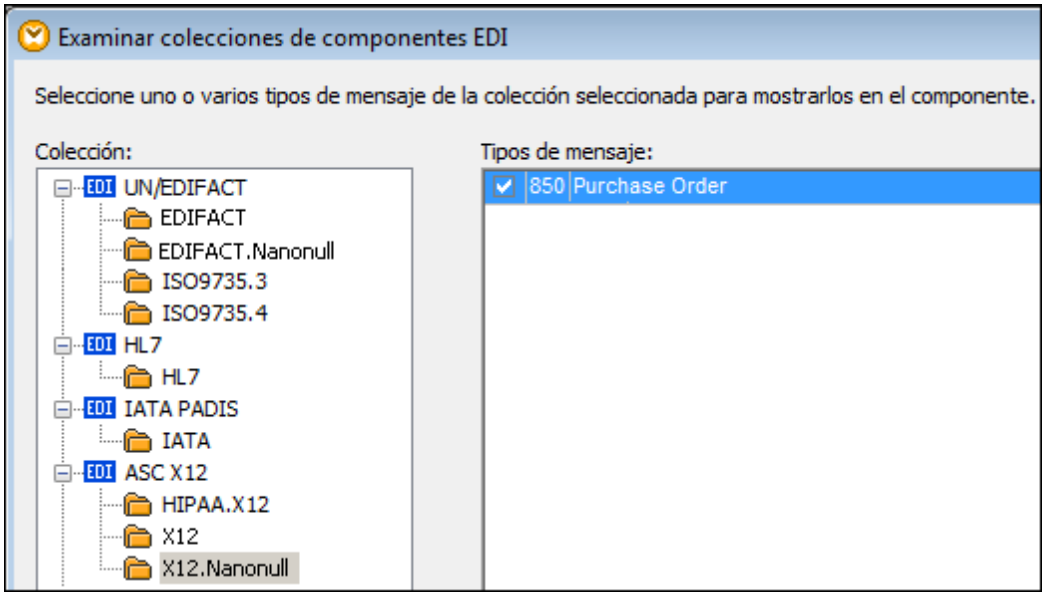
1. Abra el archivo `EDI.Collection` en XMLSpy o en otro editor.
2. Quite todos los mensajes **`Message`**, excepto el elemento **`Message`** llamado **`850 Purchase Orders`**. Compruebe que conserva las etiquetas `<Messages>`.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Messages Version="3">
3    <Meta>
4      <Release>6020</Release>
5      <Agency>X12</Agency>
6    </Meta>
7    <Root File="Envelope.Config"/>
8    <Message Type="850" File="850.Config" Description="Purchase Order"/>
9  </Messages>
```

3. Guarde el archivo.

Para ver el contenido del archivo de colecciones:

- Inicie MapForce y seleccione **Insertar | EDI** o haga clic en el icono **Insertar EDI**.
  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 `850 Purchase` .



Nota: 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`.

Este apartado explica cómo redefinir el segmento N2 (información de nombre) añadiendo el campo X1000 para que esté a disposición de las distintas transacciones. N2 está formado por el campo F93 Name.

Hay varias maneras de personalizar el segmento:

- **De forma global**, personalizando el archivo `X12.Segment`. Todos los segmentos, de todas las transacciones que usan N2, contendrán/harán referencia al elemento nuevo.
- **De forma local**, personalizando el archivo `850.Config`. Solamente el segmento N2 personalizado de la transacción actual contendrá el elemento nuevo.

EDI.Collection
Este archivo enumera todas las transacciones del directorio actual y ofrece una lista de las transacciones disponibles cuando se inserta un archivo X12 en MapForce. Este ejemplo contiene solamente una transacción, la transacción `Purchase Order`.

850.Config
Este es el archivo de configuración para los archivos de transacciones `Purchase order`. Este archivo contiene todos los grupos y definiciones de segmentos utilizados en las transacciones `Purchase Order`. Los cambios realizados en este archivo pueden definir personalizaciones locales o inline.

X12.Segment
Este archivo define el nombre de los segmentos, de los campos y de las composiciones de los archivos X12 y se utiliza para analizar el archivo. Los cambios realizados en este archivo son personalizaciones globales y afectan a todos los segmentos y todas las transacciones.

Personalización global

- Para poder acceder al campo X1000 nuevo de forma global solamente es necesario realizar cambios en el archivo `X12.Segment`.
- Todos los **segmentos** de todas las **transacciones** que usen **N2** contendrán/harán referencia al elemento nuevo.

Redefinición del archivo `X12.Segment`:
Abra el archivo `X12.Segment` en XMLSpy o en otro editor y navegue hasta **`Config | Elements | Segment name="N2" info="Additional Name Information"`**.

```
<Segment name="N2" info="Additional Name Information">
    <Data ref="F93" mergedEntries="2"/>
</Segment>
```

Inserte esta línea bajo F93 y guarde el archivo:

```
<Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
```

La definición tiene este aspecto:

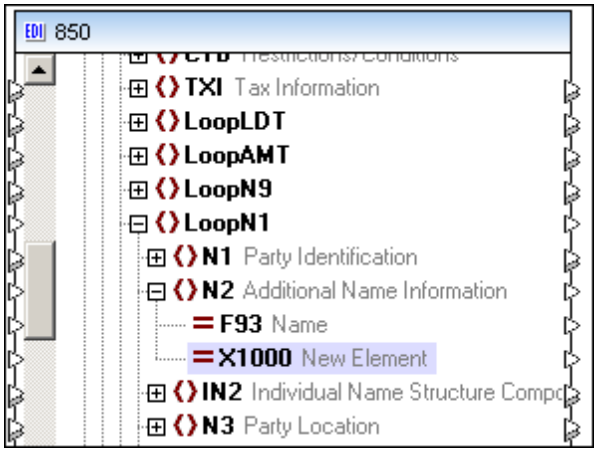
```
<Segment name="N2" info="Additional Name Information">
    <Data ref="F93" mergedEntries="2"/>
    <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New Element"/>
</Segment>
```

 © 2014 Altova GmbH

Nota: el campo X1000 nuevo se define usando el atributo `name`, al contrario de los otros campos del segmento, que usan el atributo `ref`. El campo F93 se define al principio del archivo `X12.Segment`, usando el elemento `Data name` y solamente se le hace referencia aquí. Al campo nuevo se le puede hacer referencia desde los diferentes elementos `Segment` o `Composite`.

Vista previa del campo nuevo en MapForce:

1. Seleccione la opción de menú **Insertar | EDI** o haga clic en el icono **Insertar EDI** .
2. En el cuadro de diálogo "Examinar colecciones" seleccione la carpeta `X12.Nanonull` y la transacción **`850 Purchase Order`**.
3. En el aviso que aparece haga clic en el botón **Omitir**.
   El componente **`850`** aparece ahora en el panel **Asignación**.
4. Navegue hasta el elemento **`Envelope/Interchange/Group/Message_850/LoopN1/N2`** para ver el elemento X1000 nuevo.



Personalización local

- Para tener acceso local al campo X1000 nuevo solamente hace falta realizar cambios en el archivo `850.config`.
- Solamente el segmento N2 redefinido de la transacción **actual** contendrá/hará referencia al campo X1000 nuevo.
- En otras palabras, el segmento se redefine localmente para que contenga el campo X1000 nuevo.

Personalización local: redefinición del segmento en el archivo `850.Config`:
Abra el archivo **`850.Config`** en XMLSpy o en otro editor y navegue hasta **`Config | Group | Message | Group name="LoopN1"`** (o busque **`LoopN1`**).

```
73  <Group name="LoopN1" maxOccurs="200" minOccurs="0">
74      <Segment ref="N1"/>
75      <Segment ref="N2" minOccurs="0" maxOccurs="2"/>
76      <Segment ref="IN2" minOccurs="0" maxOccurs="unbounded"/>
77      <Segment ref="N3" minOccurs="0" maxOccurs="2"/>
78      <Segment ref="N4" minOccurs="0" maxOccurs="unbounded"/>
```

Reemplace la línea del elemento **`Segment ref="N2"...`** con estas líneas:

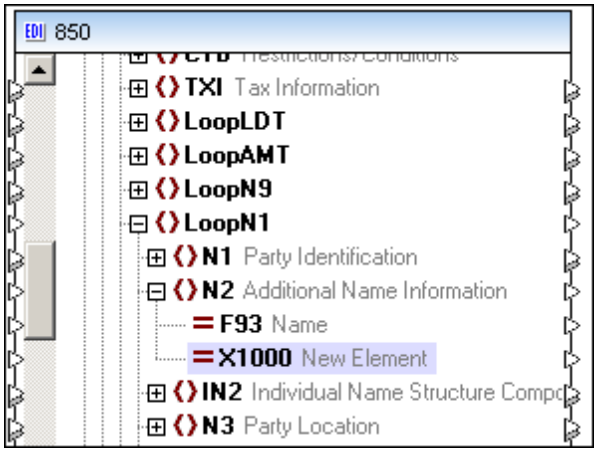
```
<Segment name="N2" info="Additional Name Information">
        <Data ref="F93" mergedEntries="2"/>
        <Data name="X1000" type="string" maxLength="35" minOccurs="0" info="New
```

```
Element"/>
</Segment>
```

```
73  <Group name="LoopN1" maxOccurs="200" minOccurs="0">
74      <Segment ref="N1"/>
75      <Segment name="N2" info="Additional Name Information">
76          <Data ref="F93" mergedEntries="2"/>
77          <Data name="X1000" type="string" maxLength="35" minOcc
78      </Segment>
79      <Segment ref="IN2" minOccurs="0" maxOccurs="unbounded"/>
```

Vista previa del campo nuevo en MapForce:

1. Seleccione la opción de menú **Insertar | EDI** o haga clic en el icono **Insertar EDI** .
2. En el cuadro de diálogo "Examinar colecciones" seleccione la carpeta `X12.Nanonull` y la transacción **`850 Purchase Order`**.
3. En el aviso que aparece haga clic en el botón **Omitir**.
   El componente **`850`** aparece ahora en el panel **Asignación**.
4. Navegue hasta el elemento **`Envelope/Interchange/Group/Message_850/LoopN1/N2`** para ver el elemento X1000 nuevo.



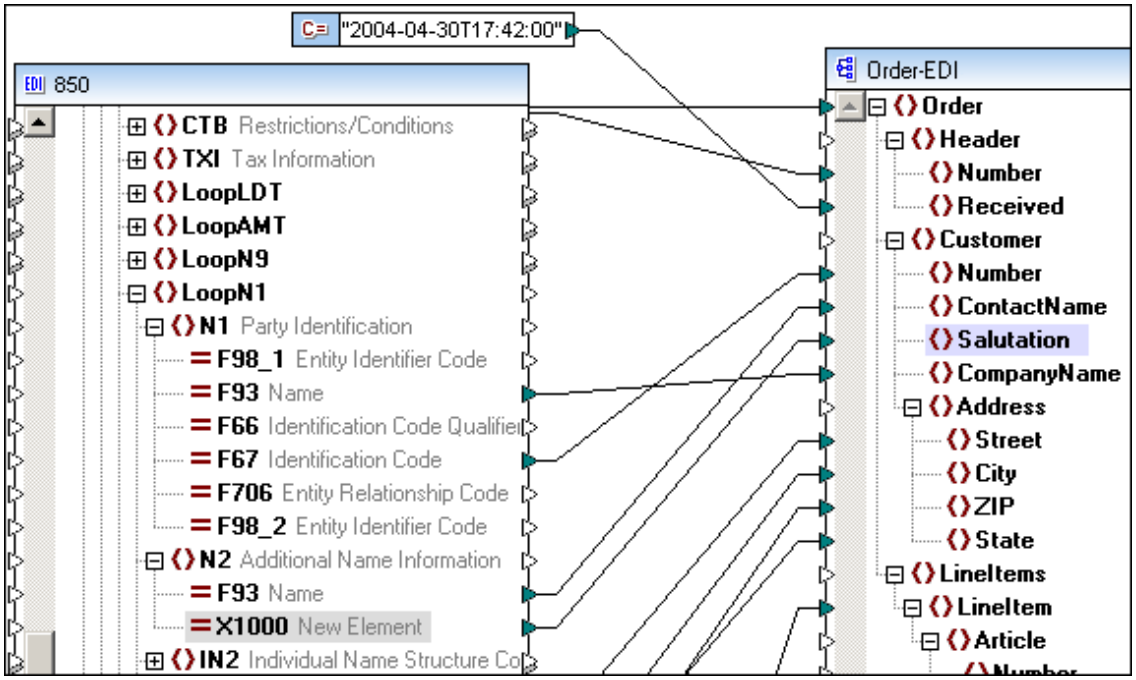
Ejemplo

La asignación del ejemplo que aparece a continuación (`Orders-Custom-X12.mfd`) está disponible en el directorio `...\MapForceExamples\Tutorial`.

Esta asignación asigna el archivo `ORDERS-Custom.x12` al esquema `Order-X12`. El campo que se añadió a la estructura EDI, **`X1000`**, se asignó al elemento **`Salutation`**.

1. Cree una carpeta nueva bajo el directorio `...\MapForceEDI` y llámela `X12.Nanonull`.
2. Descomprima el archivo ZIP `X12.Nanonull.zip` (disponible en la carpeta `...\Tutorial`) en la carpeta nueva.
3. Abra el archivo `Orders-Custom-X12.mfd`.

 © 2014 Altova GmbH



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

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Order xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespace
3    <Header>
4      <Number>ABC123456XYZ</Number>
5      <Received>2004-04-30T17:42:00</Received>
6    </Header>
7    <Customer>
8      <Number>123</Number>
9      <ContactName>Michelle Butler</ContactName>
10     <Salutation>Mrs</Salutation>
11     <CompanyName>Nanonull, Inc.</CompanyName>
12     <Address>
13       <Street>119 Oakstreet Suite 4876</Street>
14       <City>Vereno</City>
15       <ZIP>29213</ZIP>
16       <State>CA</State>
17     </Address>
```

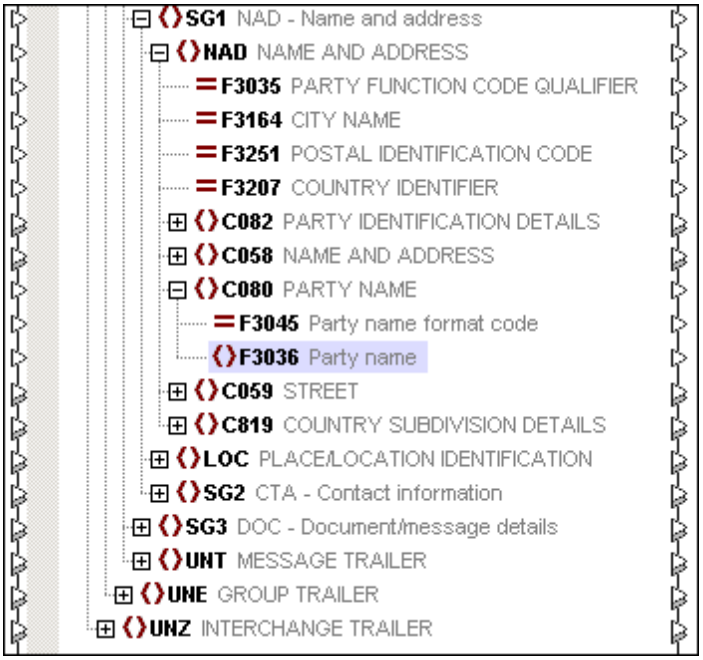
10.5.11 Nodos distintos combinando entradas

Cuando se crean asignaciones a componentes EDI, MapForce presenta automáticamente la entrada como un solo nodo asignable en el componente EDI, incluso si la entrada está formada por varias repeticiones consecutivas del mismo campo.

En EDIFACT y ANSI X12 hay dos archivos de estructuras equivalentes que permiten usar varios elementos consecutivos del mismo nombre en cada formato EDI. Se trata del archivo **`EDSD.Segment`** (para EDIFACT) y del archivo **`X12.Segment`** para ANSI X12.

Puede editar el archivo `EDSD.Segment` para dividir en varias partes (repeticiones) la secuencia combinada y crear nodos únicos/elementos asignables. Asegúrese de crear un directorio nuevo para el archivo `EDSD.Segment` y los demás archivos EDI que desea editar, tal y como se explica aquí.

El componente EDI **OSTRPT** (*imagen siguiente*) incluye el campo **`F3036 Party name`**, que aparece como un solo nodo asignable. Sin embargo, el campo **`Party name`** puede contener hasta cinco elementos **`Party Name`** por cada definición EDIFACT.



Abra el archivo `EDSD.Segment` para ver la definición de la composición C080 `Party Name`. En los segmentos que pueden tener varios campos idénticos (p. ej. F3036) el atributo `mergedEntries` define cuántas veces puede aparecer el campo (en este caso 5 veces).

```
<Composite name="C080" info="PARTY NAME">
    <Data ref="F3036" mergedEntries="5"/>
    <Data ref="F3045" minOccurs="0"/>
</Composite>
```

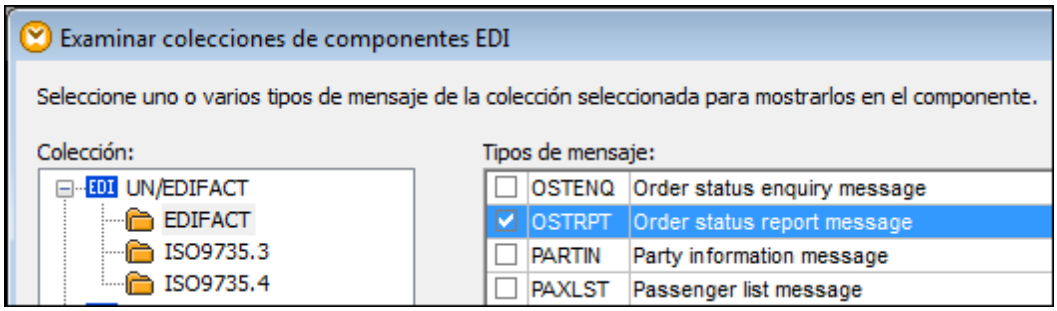
 © 2014 Altova GmbH

Para crear campos únicos a partir de una secuencia combinada:

1. Abra el archivo `EDSD.Segment` (o `X12.Segment`) situado en la carpeta `c:\Archivos de programa\Altova\MapForce2014\MapForceEDI\EDIFACT\` (o `c:\Archivos de programa\Altova\MapForce2014\MapForceEDI\X12\`).
2. Elimine el atributo `mergedEntries="5"`.
3. Copie el resto del elemento `<Data ref="F3036"/>` e insértelo tantas veces como indicara el atributo `mergedEntries`, es decir, 5 veces.
4. Añada el atributo `minOccurs="0"` a todos los campos **excepto al primero**.
5. Escriba un nombre de nodo único en cada uno de los campos, p. ej. `F3036_1`, `F3036_2` y así sucesivamente.

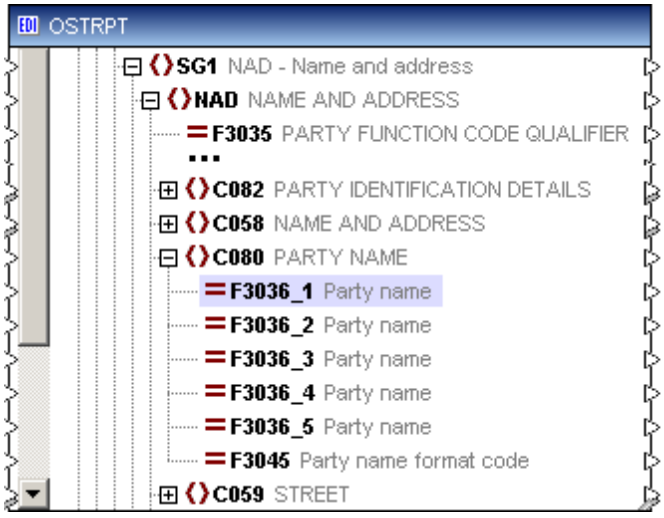
```
<Composite name="C080" info="PARTY NAME">
    <Data ref="F3036" nodeName="F3036_1"/>
    <Data ref="F3036" nodeName="F3036_2" minOccurs="0"/>
    <Data ref="F3036" nodeName="F3036_3" minOccurs="0"/>
    <Data ref="F3036" nodeName="F3036_4" minOccurs="0"/>
    <Data ref="F3036" nodeName="F3036_5" minOccurs="0"/>
    <Data ref="F3045" minOccurs="0"/>
</Composite>
```

6. Guarde el archivo.
7. Abra MapForce, haga clic en el icono **Insertar EDI** e inserte el tipo de mensaje **`Order Status Report message`** de la carpeta `EDIFACT`.



Se inserta el componente EDI **OSTRPT**.

8. Expanda los elementos del componente hasta encontrar el nodo **`Envelope | Interchange | Group | Message_OSTRPT | SG1 NAD`**.
9. Expanda el nodo **`C080`** para ver los nuevos campos únicos **`Party name`**.
   Ahora puede crear asignaciones de datos para estos campos.

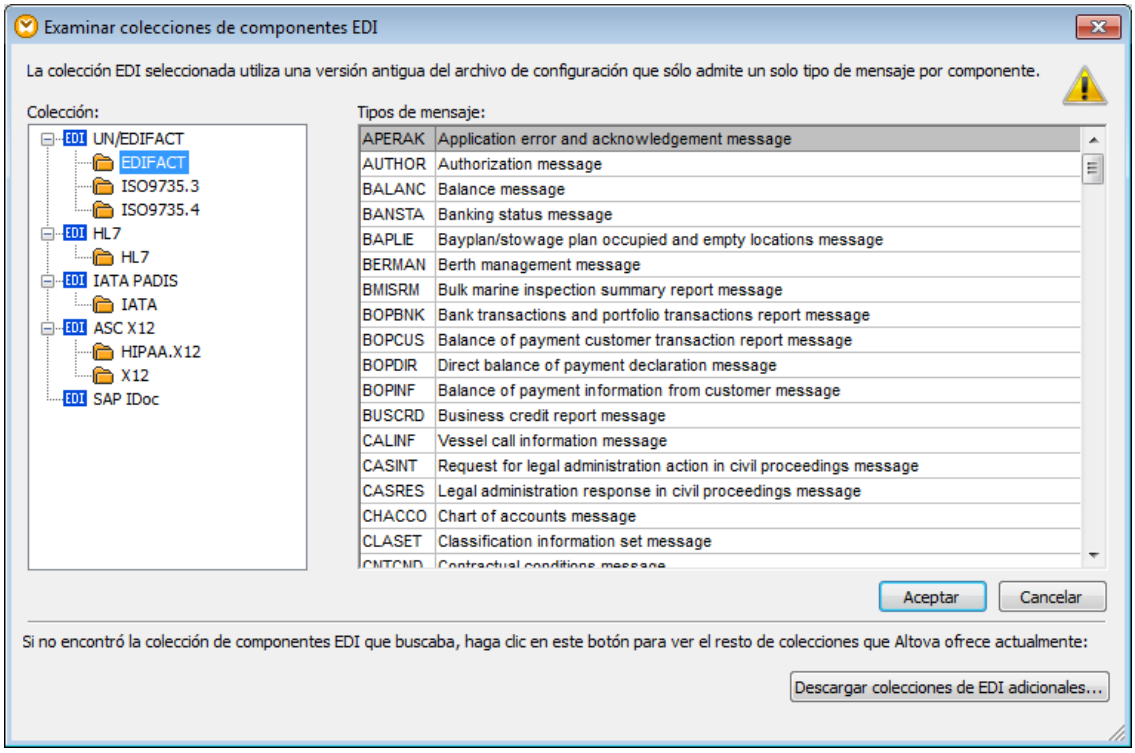


Para más información consulte el apartado Elementos consecutivos múltiples.

 © 2014 Altova GmbH

10.5.12 Actualizar archivos de configuración

Si usa archivos de configuración EDI anteriores a la versión 3, el cuadro de diálogo "Examinar colecciones de componentes EDI" incluye una advertencia para informarle de que está utilizando un archivo de configuración que solamente admite un tipo de mensaje por componente.



Actualice su archivo de configuración para poder trabajar con varios tipos de mensaje en cada componente:

1. Copie el archivo `Envelope.Config` de la carpeta de configuración original (p. ej. `EDIFACT`) en la carpeta que contiene los archivos de configuración personalizados (p. ej. `EDIFACT.Nanonull`).
2. Edite el archivo `EDI.Collection` y cambie el atributo `Version` del elemento raíz de `2` a `3`.
3. Después de la etiqueta `</Meta>` añada `<Root File="Envelope.Config"/>`:

```
<?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. Edite el archivo `ORDERS.Config` y cambie el atributo `Version` del elemento raíz de `2` a `3`.

5. Añada `<Format standard="EDIFACT"/>` (o X12, HL7, etc.).
6. Cambie `<Group name="Message"...>` por `<Group name="Message_ORDERS"...>` (o cualquier otro tipo de mensaje)
7. Elimine los niveles de grupo externos (`Envelope`, `Interchange` y sus segmentos):

```
<?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>
```

Si mientras edita el archivo de configuración tiene cargada una asignación que utiliza este archivo, deberá cargar la asignación. Las conexiones se actualizan automáticamente (el elemento conectado será `Message_ORDERS` en lugar de `Message`).

El mismo método se utiliza para actualizar los archivos de configuración EDI X12 o HL7.

 © 2014 Altova GmbH

10.5.13 SAP IDocs

Los documentos SAP IDocs (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.

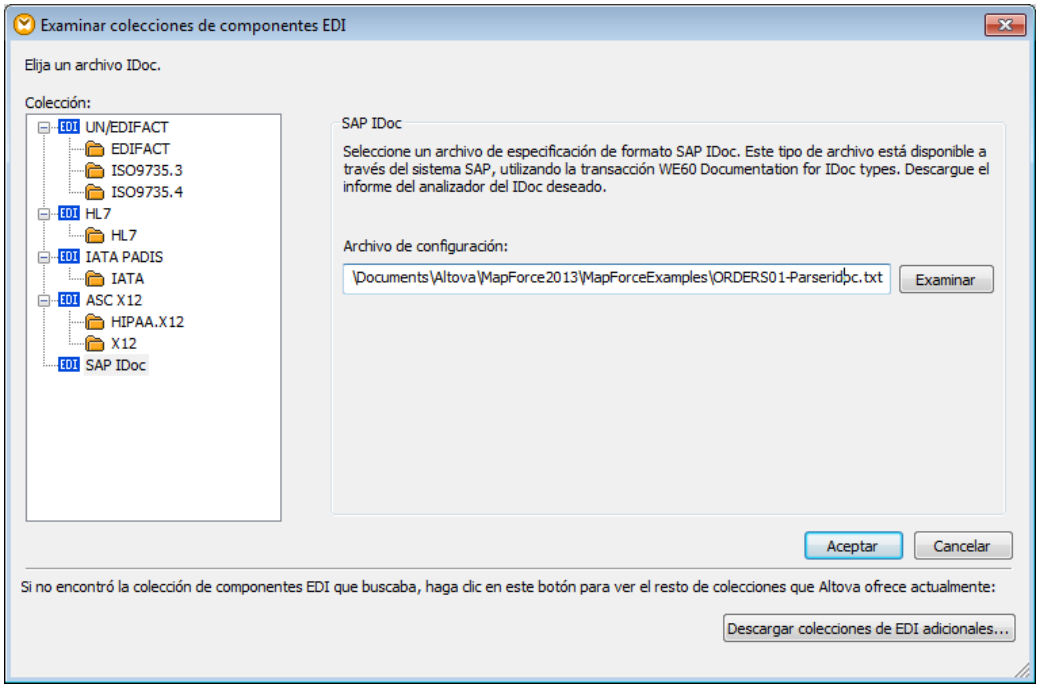
Versión de puerto 1: Versiones 2.1. y 2.2.
Versión de puerto 2: Versiones 3.0, 3.1 y sistemas R/2.
Versión de puerto 3: Versión 4.x (valor predeterminado)

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.

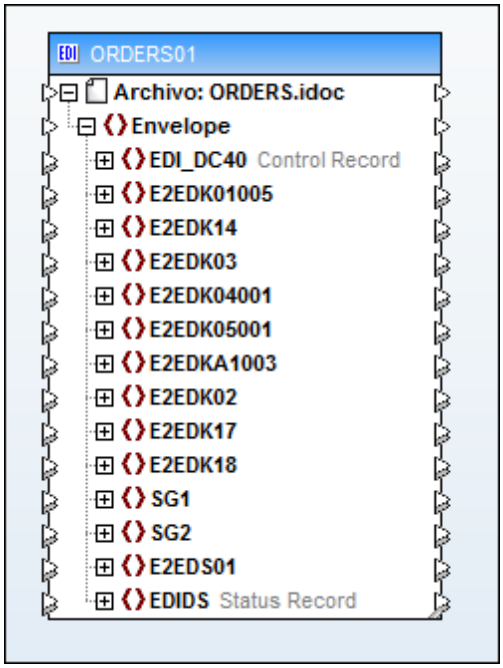
El archivo `IDoc_Order.mfd` disponible en `...\MapForceExamples` incluye una asignación de datos entre un documento SAP IDoc y un esquema XML de destino.

Para insertar un documento SAP IDoc:

1. Seleccione la opción de menú **Insertar | EDI**. Se abre el cuadro de diálogo "Examinar colecciones de componentes EDI".
2. En el cuadro de lista izquierdo seleccione `SAP IDoc`.
3. Haga clic en el botón **Examinar**, seleccione el archivo de configuración y haga clic en **Aceptar** para continuar.
   Este archivo de configuración puede venir con su sistema SAP (usando la transacción `WE60 Documentation for IDoc types`). En la carpeta `...\MapForceExamples` encontrará el archivo de configuración `ORDERS01-Parseridoc.txt`, que forma parte de la asignación `IDoc_Order.mfd`.



4. En el aviso que aparece haga clic en **Examinar** para seleccionar el archivo EDI de muestra (`*.idoc`) que suministra los datos si desea crear un componente de origen. Utilice, por ejemplo, el archivo `ORDERS.idoc` de la carpeta `...\MapForceExamples`. Aparece el cuadro de diálogo "Configuración del componente".
5. Haga clic en **Aceptar** para cerrar el cuadro de diálogo e insertar el componente IDoc.



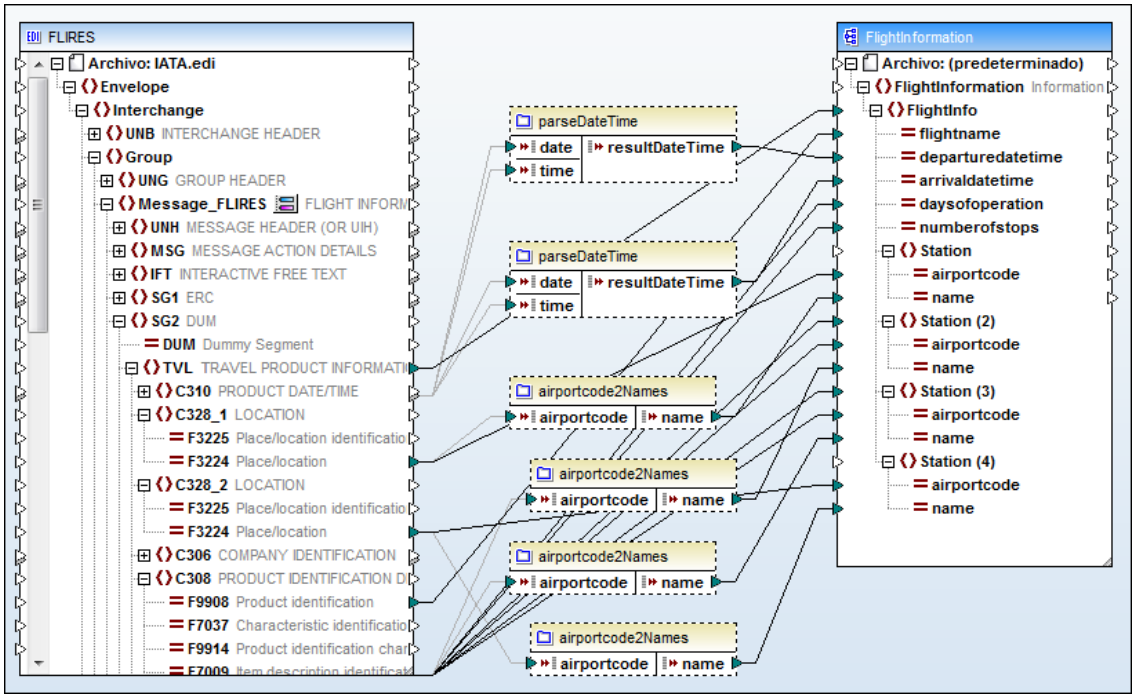
Para más información sobre cómo invocar la transacción WE60 y generar un informe de análisis consulte la documentación de SAP IDoc.

10.5.14 IATA PADIS

PADIS (*estándar de intercambio de datos sobre pasajeros y aeropuertos*) es un conjunto de mensajes que usa la sintaxis EDIFACT (ISO 9735).

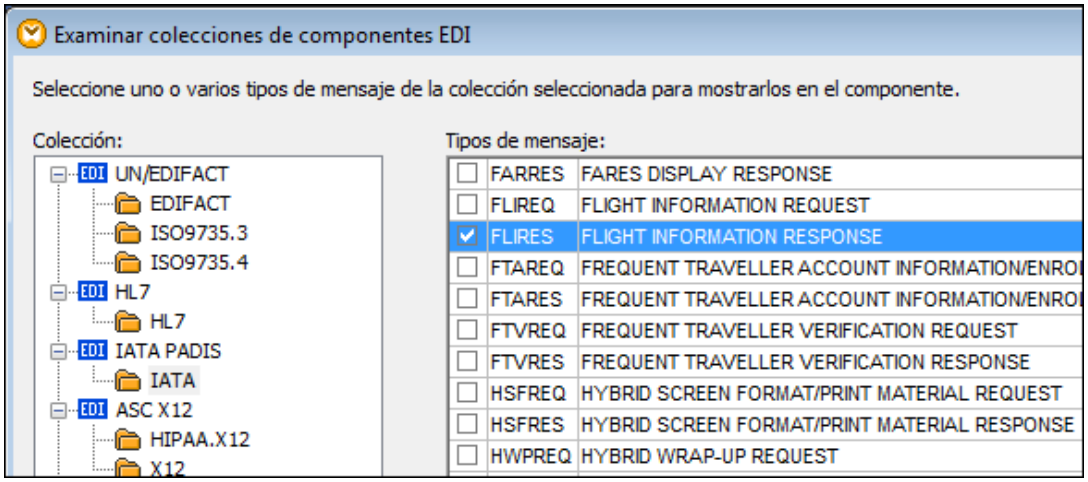
- MapForce es compatible con los mensajes que usan el encabezado y el finalizador de mensajes UNH/UNT.
- MapForce es compatible con las colecciones IATA.00.1 - IATA.08.1.

El archivo `IATA_FlightInformationReport.mfd` de la carpeta `...\MapForceExamples` es un ejemplo de asignación de datos entre un archivo IATA PADIS y un esquema XML de destino.

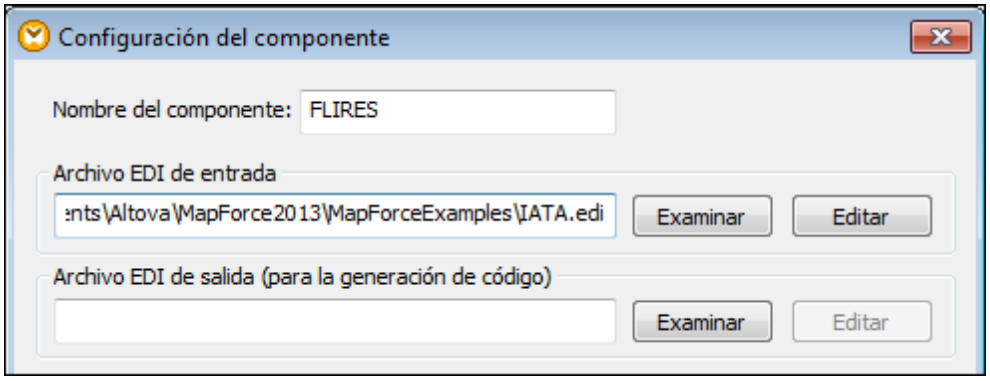


Para insertar un componente PADIS:

1. Seleccione la opción de menú **Insertar | EDI**. Se abre el cuadro de diálogo "Examinar colecciones de componentes EDI".
2. Haga clic en la entrada `IATA` en el cuadro de lista izquierdo.
3. Marque la casilla del tipo de mensaje que desea insertar (p. ej. **`FLIRES`**) y haga clic en **Aceptar**.



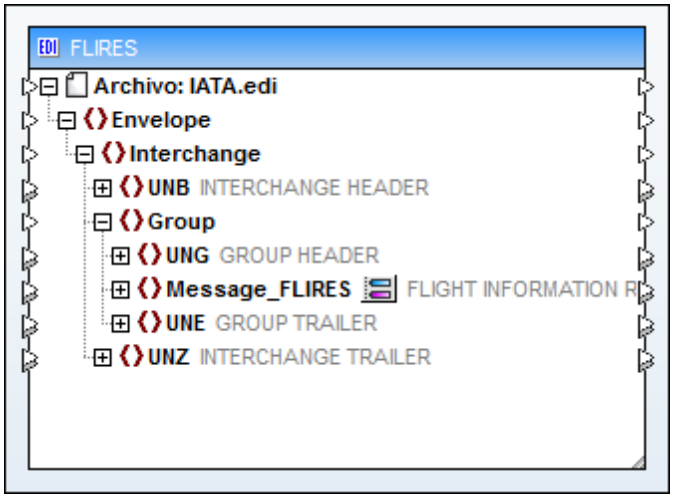
4. En el aviso que aparece haga clic en **Examinar**. Busque el archivo EDI de muestra (p. ej. `IATA.edi`) y haga clic en **Abrir**. Se abre el cuadro de diálogo "Configuración del componente".



El archivo `IATA.edi` suministra los datos para el componente de origen y está disponible en la carpeta `...\MapForceExamples`.

5. Haga clic en **Aceptar** para cerrar el cuadro de diálogo e insertar el componente PADIS.

 © 2014 Altova GmbH



Nota: el botón interactivo **Seleccionar mensaje EDI** abre el cuadro de diálogo "Examinar colecciones de componentes EDI", donde puede seleccionar varios tipos de mensajes para el componente.

10.5.15 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:

X12 name	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:

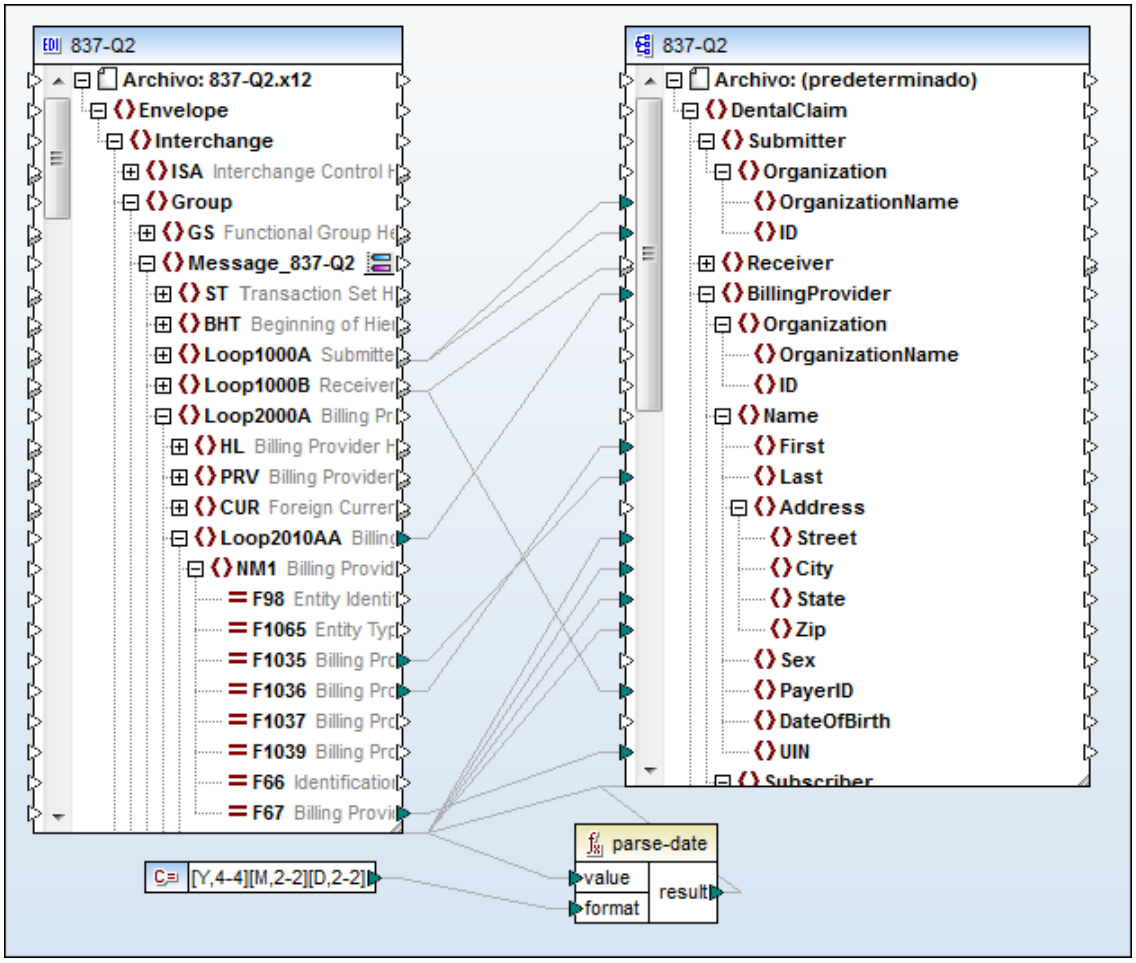
- conserva automáticamente la jerarquía de los segmentos HL
- es compatible con las llamadas estructuras flotantes (de 837 mensajes)
- finaliza automáticamente y valida más campos

MapForce también genera automáticamente la confirmación de implementación 999 de X12, similar a la confirmación funcional 999 y 997 del estándar X12.

En los componentes HIPAA también se pueden usar varios tipos de mensajes diferentes.

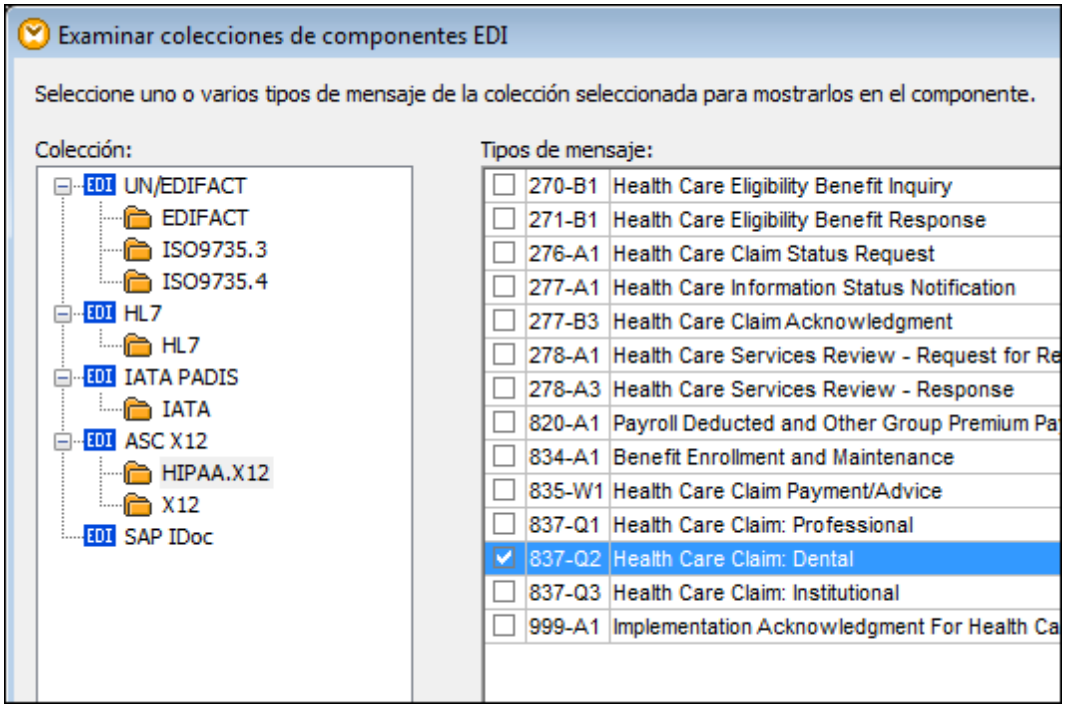
El archivo `HIPAA_837D.mfd` de la carpeta `...\MapForceExamples` asigna una transacción **837-Q2 Health Care Claim: Dental** a un esquema XML de destino.

 © 2014 Altova GmbH

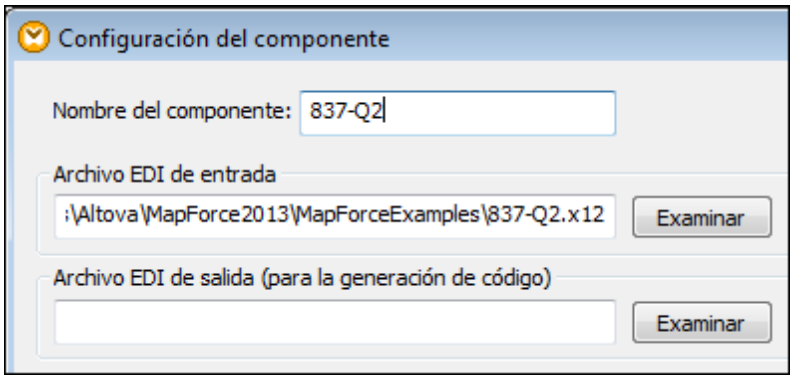


Para insertar un componente HIPAA X12:

1. Seleccione la opción de menú **Insertar | EDI**. Se abre el cuadro de diálogo "Examinar colecciones de componentes EDI".
2. Haga clic en la carpeta `HIPAA.X12` en el cuadro de lista izquierdo.
3. Marque la casilla del tipo de mensaje que desea insertar (p. ej. **`837-Q2: Health Care Claim: Dental`**) y después haga clic en **Aceptar**.

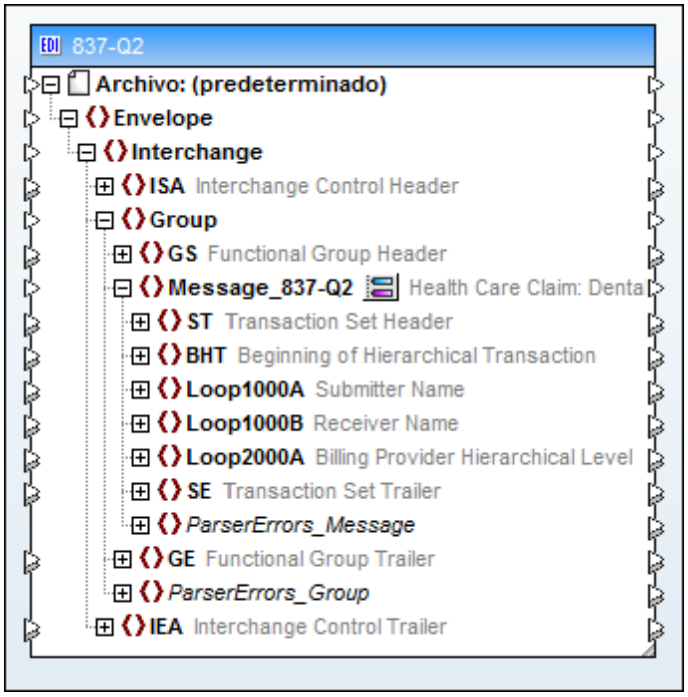


4. En el aviso que aparece haga clic en **Examinar** para buscar el archivo EDI de muestra (p. ej. `837-Q2.x12`)
   Aparece el cuadro de diálogo "Configuración del componente", donde puede configurar los separadores.



   Este archivo ofrece al componente de origen los datos de entrada. `837-Q2.x12` está disponible en la carpeta `...\MapForceExamples`.
5. Haga clic en **Aceptar** para cerrar el cuadro de diálogo "Configuración del componente" para insertar el componente HIPAA.

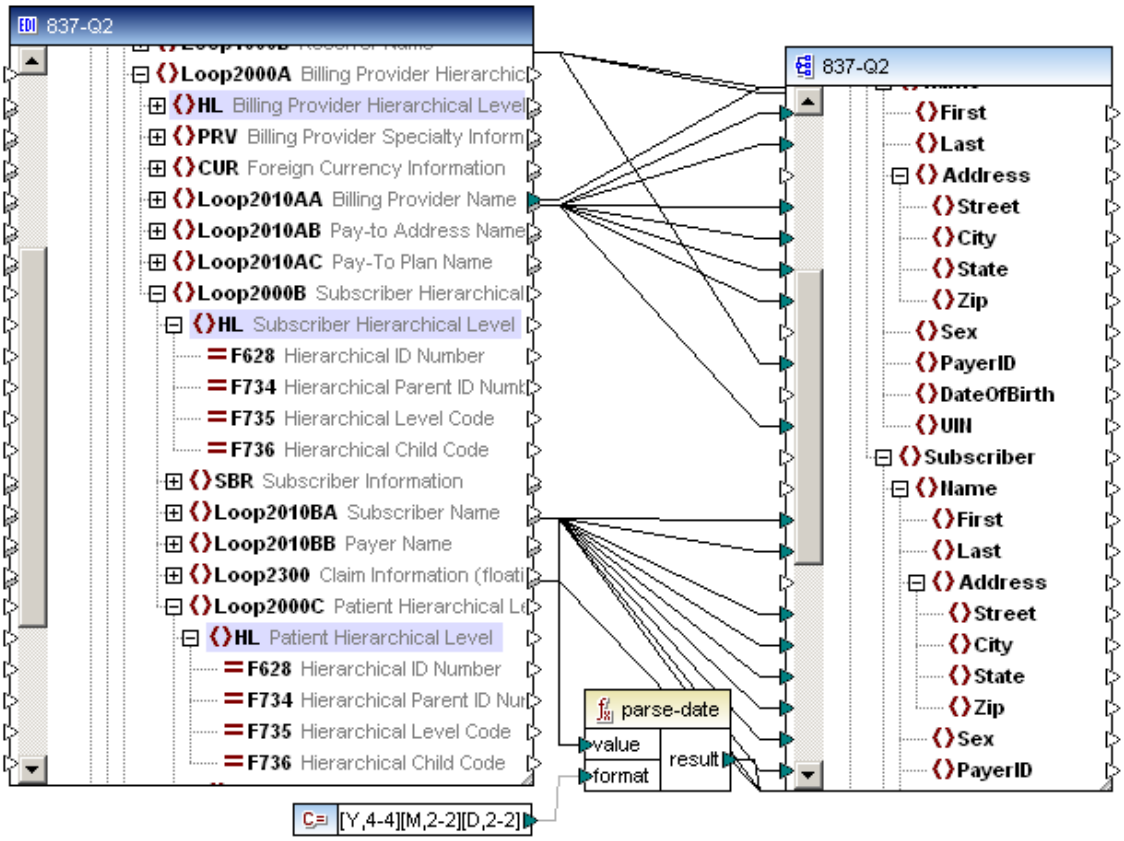
 © 2014 Altova GmbH



Nota: el botón interactivo **Seleccionar mensaje EDI** abre el cuadro de diálogo "Examinar colecciones de componentes EDI" donde puede seleccionar más mensajes para el componente.

Transacciones HIPAA

Transacción 837 Dental en MapForce



Jerarquía de los segmentos HL

Los segmentos HL se generan automáticamente y no es necesario crear asignaciones a mano. No obstante, si lo necesita puede asignar sus valores.

Finalización automática

Los archivos de configuración para las transacciones HIPAA contienen muchos códigos y se utilizan para finalizar automáticamente los campos. La finalización automática es contextual. Es decir, dependiendo del campo, la finalización automática utiliza unos valores u otros.

Para deshabilitar la finalización automática, desactive la casilla *Rellenar automáticamente los campos que falten* del cuadro de diálogo "Configuración del componente".

Validación

La función de validación es similar a la finalización automática porque también es contextual. Los errores de validación funcionan igual que en la validación de X12.

Edición de los archivos de configuración

Los archivos de configuración HIPAA son similares a los archivos de configuración X12, pero también tienen:

- valores (lista de código)
- condiciones
- marcas de finalización
- elementos de datos "no utilizados"

Los campos pueden tener una lista de códigos integrada que se utiliza para la validación y finalización automática. Los valores se pueden añadir a mano o se pueden quitar.

```
<Data ref="F365" info="Communication Number Qualifier" nodeName="F365_1">
<Values>
       <Value Code="EM" />
       <Value Code="FX" />
       <Value Code="TE" />
</Values>
</Data>
```

Condiciones

Las condiciones funcionan conjuntamente con los valores. Ya que un segmento o grupo (*loop*) puede aparecer varias veces, se identifican con una condición determinada. Para que la condición se cumpla, es necesario que esté presente un valor. De lo contrario, el grupo no se encuentra.

Por ejemplo, **`Loop1000`** se repite varias veces pero cada instancia tiene un significado semántico diferente.

Para distinguir uno de otro, MapForce usa una condición que especifica que **`Loop1000A`** debe tener el código de valor **`41`** en el campo **`F98`** del segmento **`NM1`**. Si no es así, entonces no se encuentra el nombre de grupo **`Loop1000A`**.

```
<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>
(…)
```

Los códigos de las condiciones se generan automáticamente en el componente de destino si contienen un valor único.

Los archivos de configuración se generan mediante un método que garantiza que los valores de la condición sean únicos en la secuencia de segmentos/grupos repetidos.

Si es necesario editar las condiciones (o sus valores), se debe tener en cuenta la restricción de unicidad.

Marcas de finalización

La finalización automática se puede ajustar al nivel de archivo de configuración. Para definirlo se introdujo un elemento nuevo llamado `Completion`, que tiene tres atributos:

- `singleConditions`: finalizar automáticamente condiciones únicas para todos los campos del componente de destino
- `singleValues`: finalizar automáticamente valores únicos para todos los campos del componente de destino
- `HL`: generar campos en el segmento HL del componente de destino (para todos los segmentos HL), donde `1` = verdadero.

```
<Message>
```

```
  <MessageType>837-Q2</MessageType>
  <Completion singleConditions="1" singleValues="1" HL="1" />
  <Description>Health Care Claim: Dental</Description>
(…)
```

Elementos de datos "no utilizados"
HIPAA omite varios elementos opcionales que están presentes en las transacciones X12 estándar. Estos elementos opcionales se ocultan en los componentes de MapForce.

El motor de validación comprueba si estos campos no utilizados están presentes o no en los archivos de origen. Estos campos no se pueden asignar al componente de destino (porque están ocultos). Por tanto, tampoco pueden aparecer en los archivos de resultados.

En los archivos de configuración se define que estos campos deben tener el atributo `maxOccurs` con el valor `0`. Si lo necesita, puede ocultar (o mostrar) estos campos cambiando el valor del atributo `maxOccurs`.

```
<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" />
```

Ámbito de la validación
La validación semántica no es compatible en MapForce. Esto significa que los campos circunstanciales se tratan como elementos opcionales.

 © 2014 Altova GmbH

10.6 Archivos MS OOXML Excel 2007+

Sitio web de Altova: Asignación de datos Excel

En MapForce puede crear asignaciones de datos de archivos Microsoft Excel 2007+ (Office Open XML) (*.xlsx), que se pueden usar como componentes de origen y destino.

Microsoft Excel 2007+ debe estar instalado en el equipo si quiere obtener una **vista previa** de las hojas de cálculo Excel 2007+ en el panel **Resultados**. Si usa una versión previa de Microsoft Excel o no tiene instalado Microsoft Excel, podrá realizar asignaciones de datos Excel en el panel **Asignación** y guardar los resultados en un archivo de salida, pero no podrá ver una vista previa en el panel **Resultados**.

El módulo de compatibilidad de Microsoft para Word, Excel y Powerpoint 2007+ no es suficiente para ver los resultados en el panel **Resultados**. Es necesario instalar Excel 2007+.

Para generar código para C# a partir de asignaciones de datos OOXML es necesario tener Visual Studio .NET 3.0 (o superior). La generación de código para Excel 2007+ es compatible con Java y C# para la lectura y escritura y con XSLT 2 para la lectura.

Hay dos maneras de generar/guardar datos generados:

- Abriendo el panel **Resultados** (o haciendo clic en el icono **BUILTIN**), lo cual genera una vista previa en MS Excel 2007+ con el motor de ejecución integrado y seleccionando la opción de menú **Resultados | Guardar el archivo de salida** o haciendo clic en .
- Seleccionado **Archivo | Generar código en | Java, C#** y compilando y ejecutando el código generado (no se pueden usar archivos Excel para generar código C++).

Los archivos Excel 2007+ se pueden insertar:

- como plantilla XLSX junto con un archivo XLSX de muestra que suministra los datos para la vista previa de la asignación (es decir, como componente de origen).
- como plantilla XLSX sin un archivo XSLX de muestra (es decir, como componente de destino).

En los apartados siguientes describimos:

- Cómo definir los elementos asignables de un archivo Excel
- Cómo usar rangos de nombres de Excel
- Cómo agregar, definir y mover rangos de filas
- Las asignaciones de datos entre archivos Excel 2007+ y archivos XML
- Las asignaciones de datos entre bases de datos y archivos Excel 2007+
- Cómo añadir datos a archivos Excel con formato previo

No olvide que ahora es posible asignar datos de archivos Excel 2007+ a archivos de taxonomía XBRL. Para más información consulte el apartado Entre Excel y XBRL.

10.6.1 Definir los elementos asignables del libro Excel

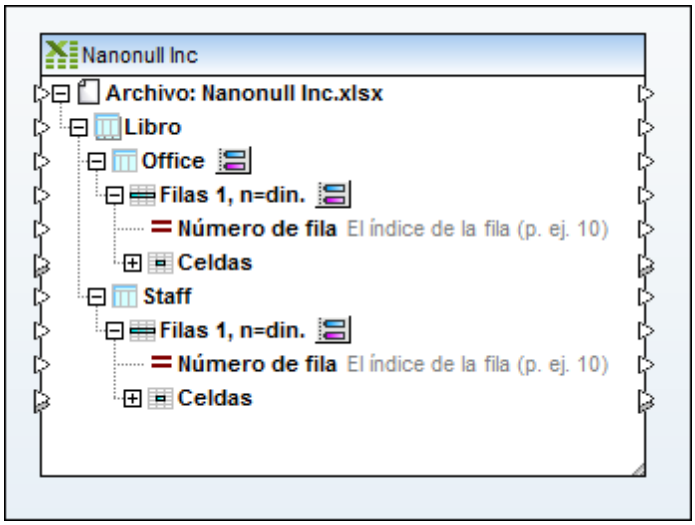
MapForce puede leer y escribir datos en forma de tabla en libros Excel. Esta es la estructura jerárquica básica de un libro Excel:

```
Libro
   Hojas de cálculo
      Filas
         Celdas (intersección de filas y columnas)
```

Al contrario que los archivos XML, los libros Excel no tienen esquemas y, por tanto, su estructura se define directamente en el componente de MapForce. Es decir, en el componente Excel de MapForce se puede definir los nombres de las hojas de cálculo y de las columnas.

Para insertar un archivo Excel *.xlsx en MapForce:

1. Haga clic en el icono **Insertar Excel 2007+** de la barra de herramientas. Aparece un aviso que solicita un archivo XLSX de muestra que aporte los datos para la vista previa. La información de este archivo se puede usar para personalizar la estructura jerárquica del componente Excel de MapForce.
2. Haga clic en **Examinar** y seleccione el archivo que desea usar para el componente Excel de origen (p. ej. `Nanonull.xlsx`).



Los rangos de celdas se pueden definir de forma manual tanto para los componentes de origen como para los de destino.

Para los componentes Excel de origen los rangos de celdas también se pueden definir mediante nombres de rango y tablas Excel de un archivo XSLS.

Al hacer clic en el botón interactivo de una fila aparece un cuadro de diálogo donde puede seleccionar/definir el rango. Las hojas de cálculo se importan y presentan automáticamente desde el archivo XLSX, pero esta opción predeterminada se puede cambiar.

Definir las celdas asignables:

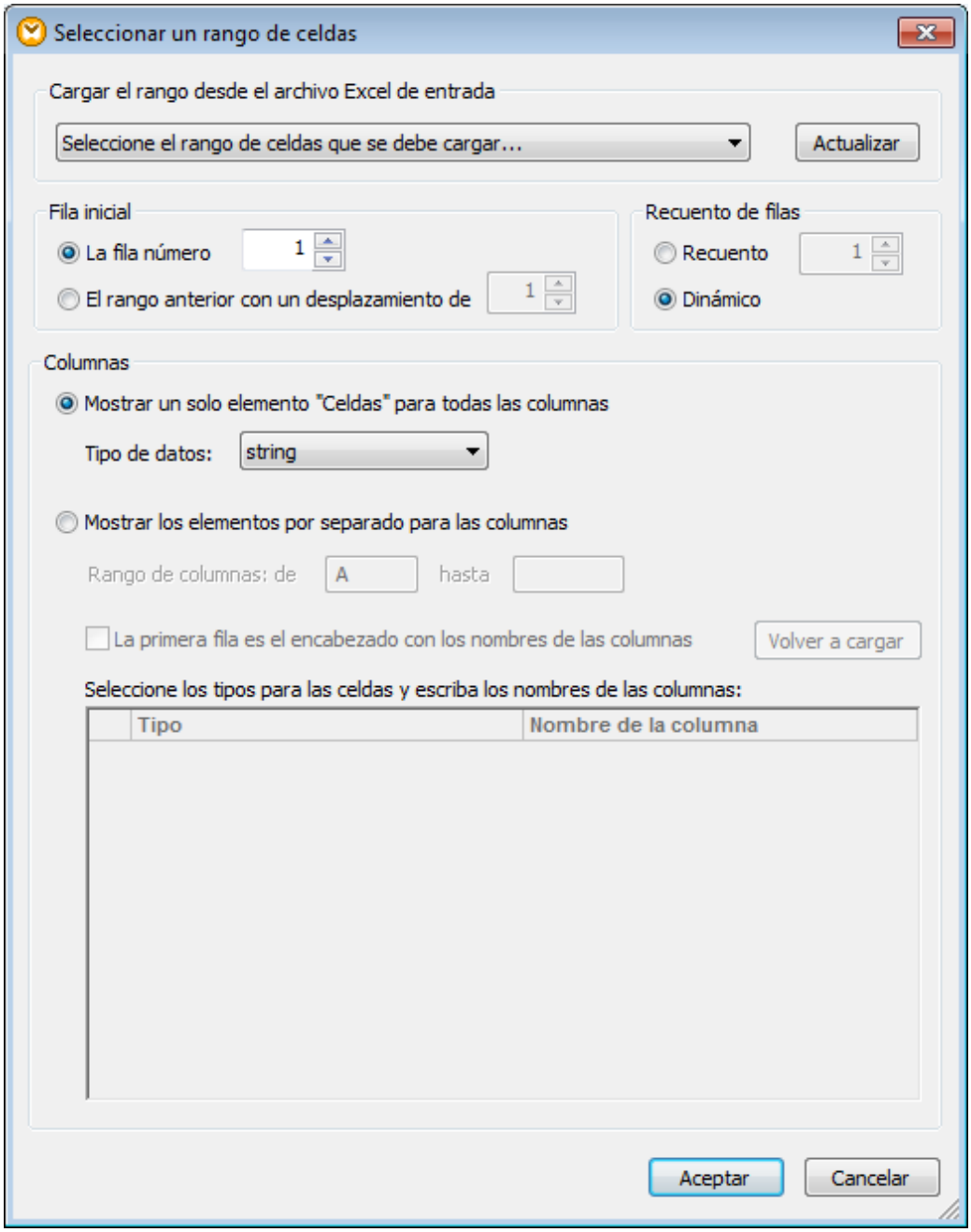
En MapForce las celdas de las diferentes filas de una hoja de cálculo se representan por defecto

 © 2014 Altova GmbH

con el elemento **Celdas**. Esto significa que todas las celdas de cada fila se procesan de forma secuencial. Esto es muy útil si todas las columnas de la hoja de cálculo tienen el mismo significado, es decir, si cada fila es una lista de valores.

Si tiene una estructura de tabla típica con varias columnas, en el árbol del componente Excel se puede presentar cada columna en un elemento diferente, lo cual facilita el acceso a las celdas:

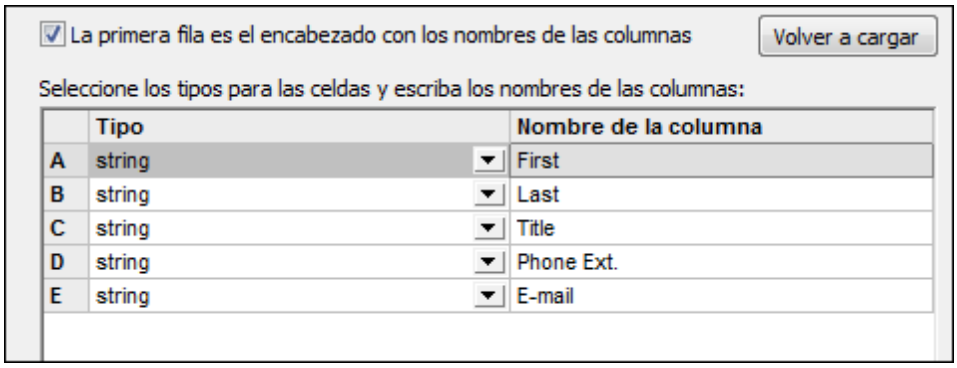
1. Haga clic en el botón interactivo del elemento **Filas 1, n=din** de la hoja de cálculo **Staff**. Se abre el cuadro de diálogo "Seleccionar un rango de celdas" (*imagen siguiente*).



2. Haga clic en el botón de opción *Mostrar los elementos por separado para las columnas* y

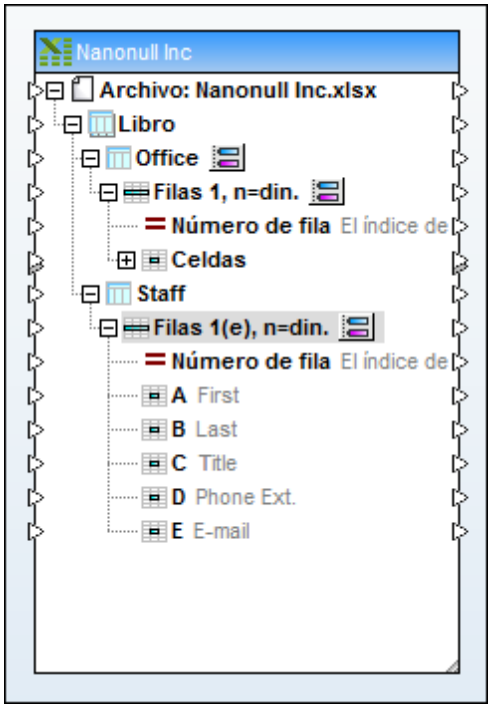
en el segundo campo escriba dónde termina el rango (p. ej. E).

3. Marque la casilla *La primera fila es el encabezado con los nombres de las columnas* para usar la primera fila como nombre de las columnas.
   En el panel inferior la columna **Nombre de la columna** se rellena con los nombres de campo de la hoja Excel.



   Si es necesario, puede cambiar el tipo de datos de los campos en el cuadro combinado **Tipo**.

4. Haga clic en **Aceptar** para confirmar los cambios. El componente Excel ahora incluye un elemento diferente por cada columna (los nodos situados bajo **`Staff | Filas`**):



Para redefinir las celdas asignables (para volver a abrir el cuadro de diálogo):

- Haga clic otra vez en el botón interactivo de **`Filas 1(e), n=din`**.

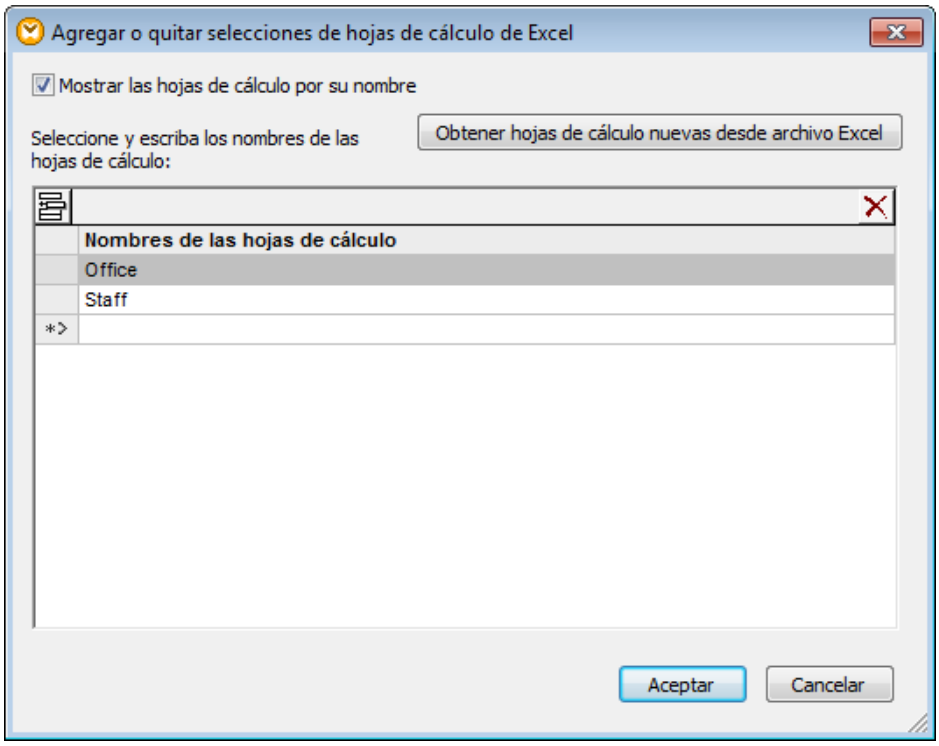
Definir las hojas de cálculo asignables:

Por lo general, cada hoja de cálculo de un libro puede tener un diseño distinto y, por tanto, cada hoja aparece como un elemento diferente en el componente Excel de MapForce. Si todas las hojas (o la mayoría) tienen la misma estructura, puede usar un solo elemento para representar

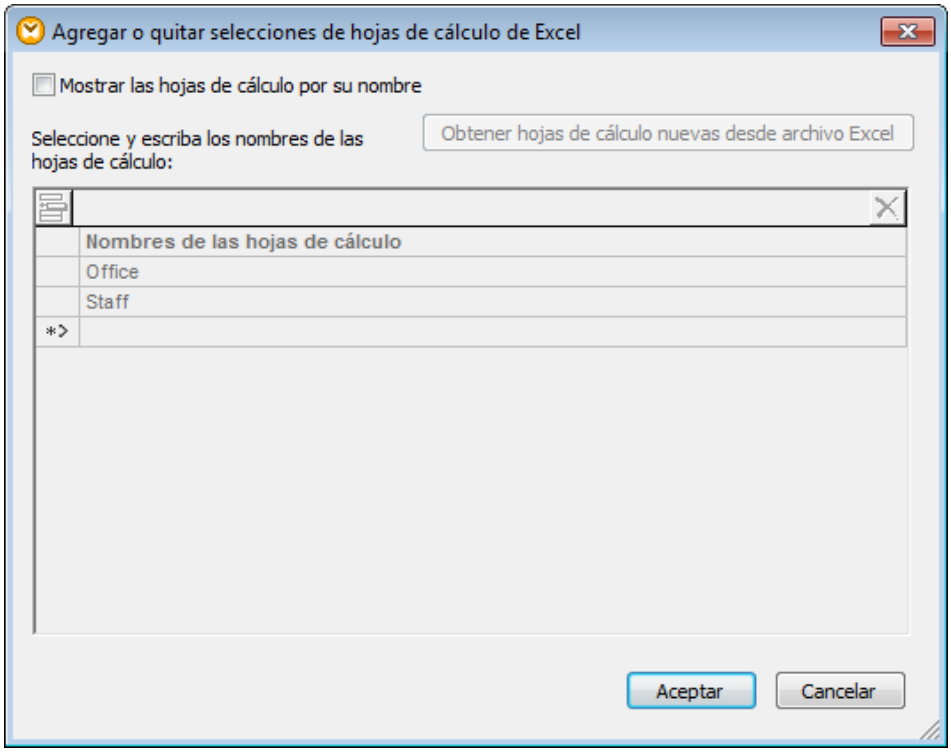
 © 2014 Altova GmbH

todas las hojas de cálculo:

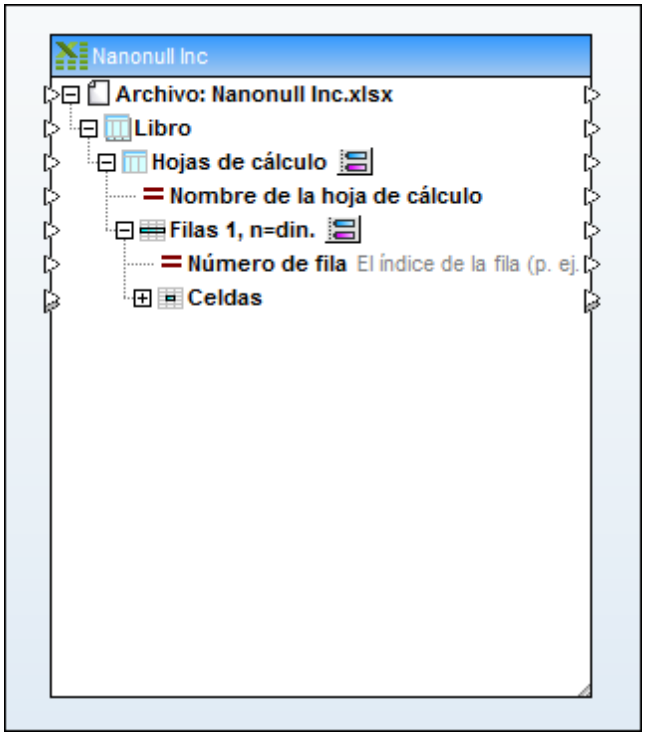
1. Haga clic en el botón interactivo del elemento **`Office`** del componente Excel. Se abre un cuadro de diálogo. Las hojas de cálculo definidas actualmente en el archivo Excel aparecen automáticamente en la lista **Nombres de las hojas de cálculo**.



2. **Desactive** la casilla *Mostrar las hojas de cálculo por su nombre*.



3. Haga clic en **Aceptar** para confirmar.
El componente Excel ahora incluye un elemento llamado **`Hojas de cálculo`** que representa todas las hojas de cálculo del libro Excel.



El libro está compuesto por los elementos **`Nombre de la hoja de cálculo`**, **`Filas 1, n=din`** y **`Celdas`**. En este caso el elemento **`Filas 1, n=din`** representa toda la hoja de

 © 2014 Altova GmbH

cálculo.

Nota: si un libro tiene varias hojas de cálculo pero está **desactivada** la opción *Mostrar las hojas de cálculo por su nombre*, entonces MapForce trata el libro como si fuera una sola hoja de cálculo. Esto permite a la asignación procesar múltiples hojas de cálculo de una sola vez, pero requiere que todas ellas tengan la misma estructura.

Agregar / eliminar hojas de cálculo

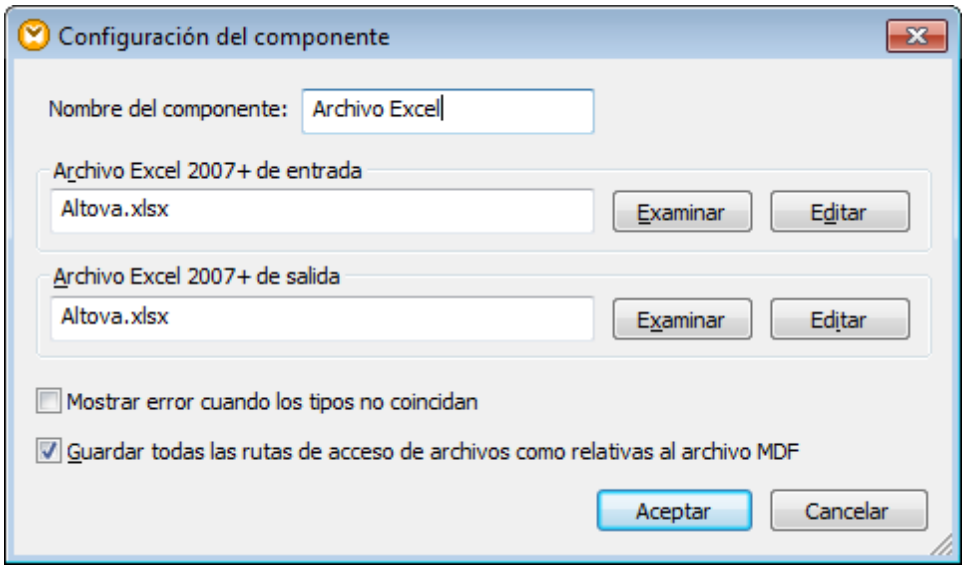
Inserta una hoja de cálculo nueva antes de la hoja seleccionada.

Anexa una hoja nueva. Escriba el nombre en el cuadro de texto situado a la derecha del icono.

Elimina la hoja de cálculo seleccionada.

Para anexar hojas nuevas que ya están en el archivo XLSX de entrada, haga clic en el botón **Obtener hojas de cálculo nuevas desde archivo Excel**.

Configuración de los componentes Excel 2007+
Los archivos de instancia de entrada y salida se pueden seleccionar como recurso global. Para más información consulte el apartado Recursos globales.

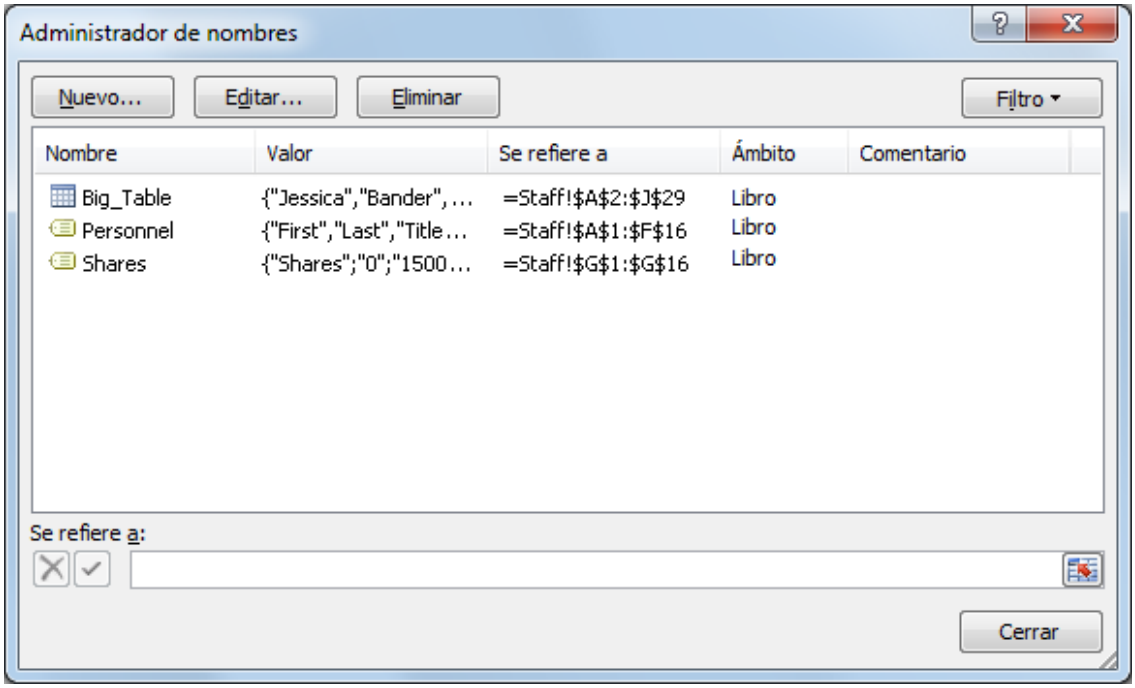


Si activa la opción *Mostrar error cuando los tipos no coincidan,* MapForce mostrará mensajes de error en la ventana Mensajes cuando el tipo de datos de la columna seleccionado por el usuario y el del archivo XLSX no coincidan (p. ej. si configura una columna como numérica, pero la columna tiene otro tipo de datos en el archivo XLSX, aparece un mensaje de error en la ventana Mensajes.)

10.6.2 Usar nombres Excel para rangos de celdas y tablas

MapForce puede leer nombres Excel para rangos y tablas y presentar los datos como elementos asignables en un componente de origen.

Tras definir los rangos y tablas y sus nombres en el libro `Nanonull.xlsx`, abra el "Administrador de nombres" de Excel para examinarlos. En el ejemplo de la imagen siguiente podemos ver que se definieron una tabla y dos rangos con nombre en la hoja de cálculo `Staff`.

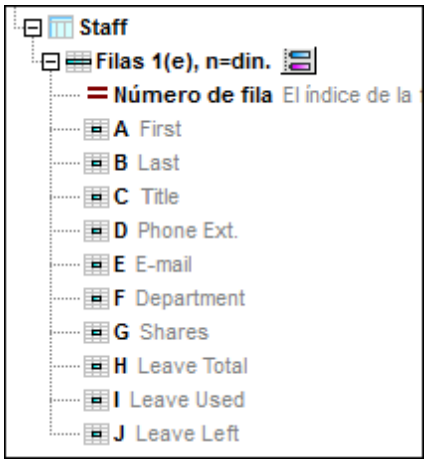


Nota: por el momento MapForce solamente es compatible con rangos rectangulares.

Seleccionar un rango con nombre de una hoja de cálculo Excel:

Abra el archivo de asignación `Excel_Company_to_XML.mfd` de la carpeta `...\MapForceExamples`:

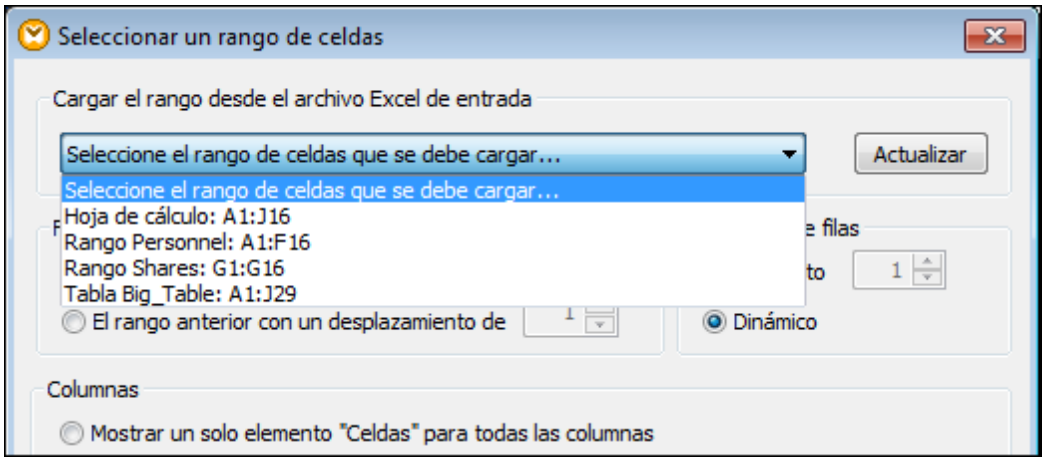
1. Haga clic en el botón interactivo **`Filas 1(e), n=din`** situado bajo la hoja de cálculo **`Staff`**.



 © 2014 Altova GmbH

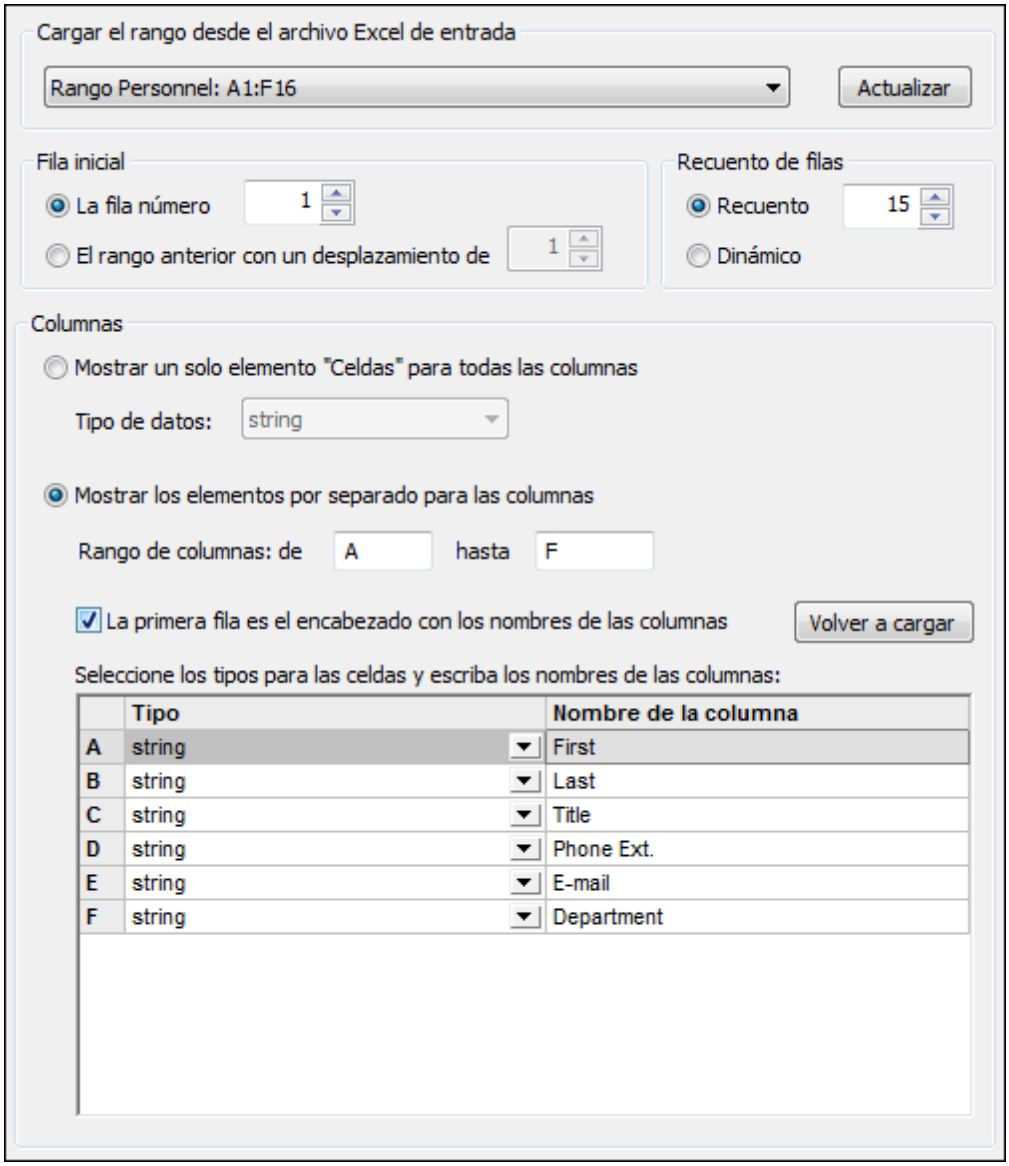
Se abre el cuadro de diálogo "Seleccionar un rango de celdas", donde puede definir los rangos de celdas del componente Excel.

2. Haga clic en el cuadro combinado para ver los rangos y las tablas que están definidos y el tamaño de la hoja de cálculo (referencias de celda).



Los nombres de los rangos y de las tablas (así como las referencias de celda) aparecen en el cuadro combinado desplegable.

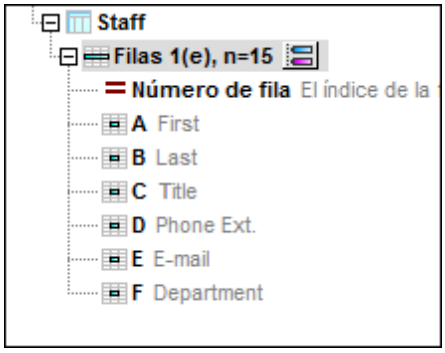
3. Seleccione el rango **Personnel: A1:F16** y observe cómo el resto del cuadro de diálogo se actualiza automáticamente con los datos/las coordenadas del rango.



- La fila inicial es la fila número 1
- El rango tiene un recuento de 16 filas
- El rango de columnas es de la columna A a la F
- La opción *Mostrar los elementos por separado para las columnas* está activada

4. Marque la casilla *La primera fila es el encabezado...* para usar los nombres de las columnas como encabezados.
5. En el mensaje que aparece haga clic en **Sí** para reducir el recuento de filas. No olvide que el recuento de filas no incluye la fila de encabezado. El rango **Personnel** incluye una fila de encabezado y 15 filas de datos.
6. Haga clic en **Aceptar** para cerrar el cuadro de diálogo.

 © 2014 Altova GmbH



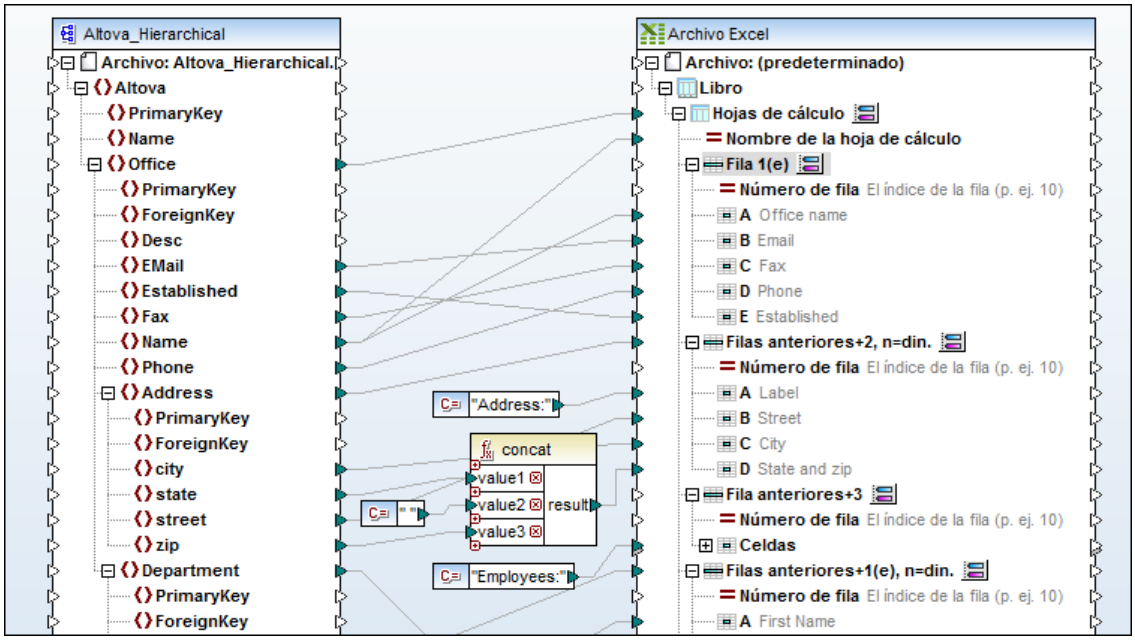
Ahora aparecen otros elementos bajo la hoja de cálculo **`Staff`** en el componente Excel. Junto al elemento **`Filas`** aparece un comentario con información sobre el rango.

Filas 1(e), n=15 significa que el rango empieza en la **fila 1**, que la primera fila es el **`encabezado`** y que el número de filas de datos es **15**.

10.6.3 Agregar, definir y mover rangos de filas

Puede definir rangos de celdas tanto para componentes Excel de origen como de destino, aunque el uso más corriente es definir rangos para los componentes de destino.

Este apartado explica cómo se diseñó la asignación del archivo de ejemplo `Altova_Hierarchical_Excel.mfd`, disponible en la carpeta `...\MapForceExamples`.

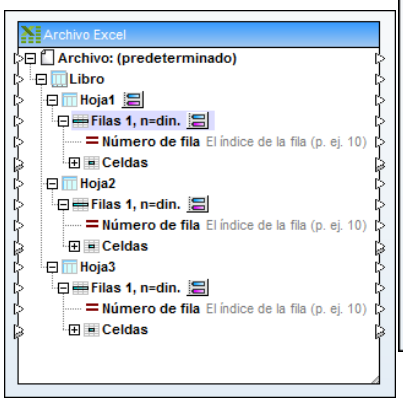


Cuando se inserta un componente de destino Excel sin aportar un archivo XLSX de muestra, MapForce define automáticamente un libro con tres hojas de cálculo.

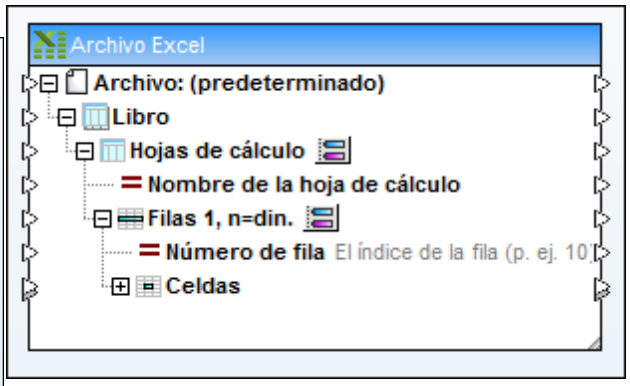
Si solamente quiere trabajar con hojas de cálculo, como en el ejemplo anterior, siga estos pasos:

- Haga clic en el botón interactivo de **`Hoja1`** y desactive la casilla *Mostrar las hojas de cálculo por su nombre* en el cuadro de diálogo.

Libro con hojas de cálculo activas

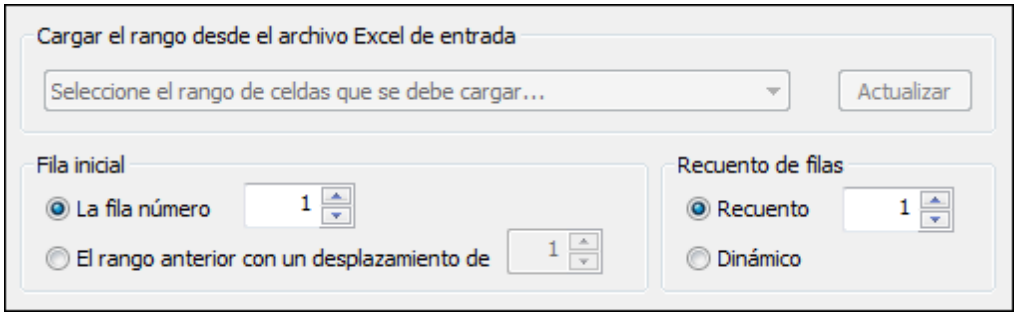


Libro con hojas de cálculo inactivas



Para agregar/crear el primer rango:

1. Haga clic en el botón interactivo del primer elemento **`Filas`** (**`Filas 1 n=din`**) para abrir el cuadro de diálogo "Seleccionar un rango de celdas".

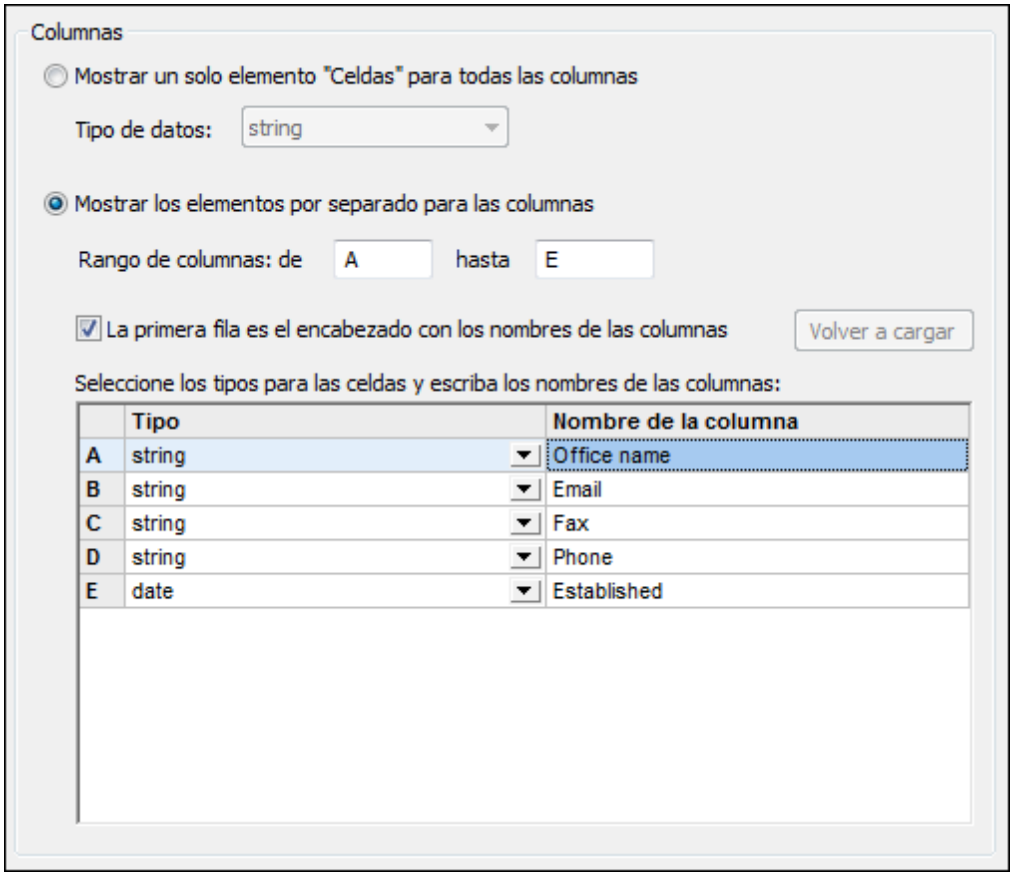


2. En este caso seleccione la *fila inicial* número 1 y un *recuento* de 1, porque el nombre de la oficina solamente debe ocupar una fila en la hoja de cálculo.

   Recuerde que el archivo XML de origen contiene datos de dos oficinas Nanonull: `Nanonull Inc` y `Nanonull Partners, Inc`. Es decir, el componente Excel genera dos hojas de cálculo en el libro Excel. El nombre de las hojas de cálculo viene dado por el nodo **`Office | Name`** del archivo XML de origen.

El **nombre de las columnas** para los elementos asignados de **`Office`** se definen:

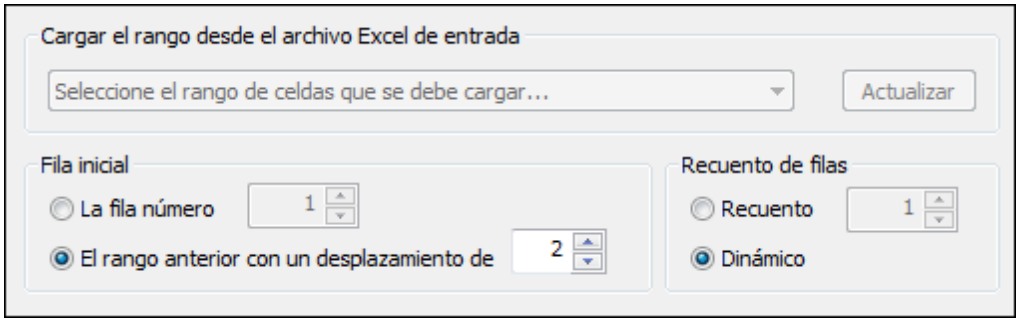
- haciendo clic en el botón de opción *Mostrar los elementos por separado para las columnas* e insertando la última columna del rango (p. ej. **E**).
- las columnas se generan en el cuadro de lista inferior (todas con el tipo de datos `string`).
- los nombres de columna se insertan directamente en la columna **Nombre de la columna** del cuadro de lista (*imagen siguiente*). En el cuadro combinado **Tipo** puede cambiar el tipo de datos de las celdas.



Rango nuevo: **`Fila1 (e)`**

Agregar un rango nuevo

1. Haga clic con el botón secundario en el primer elemento **`Filas`** (**`Filas 1, n=din`**) y seleccione **Agregar filas detrás** en el menú contextual.



El siguiente rango empieza en el rango anterior con un desplazamiento de 2 celdas, es decir, deja una celda vacía entre los dos rangos y el recuento de celdas es *Dinámico*.

Usamos este mismo método para crear los nombres de columna de las columnas A a la D.

Este rango no tiene una fila de encabezado y, por tanto, la casilla *La primera fila es el encabezado...* está desactivada.

Observe que el nombre de la primera columna (**`Label`**) viene dado por la constante

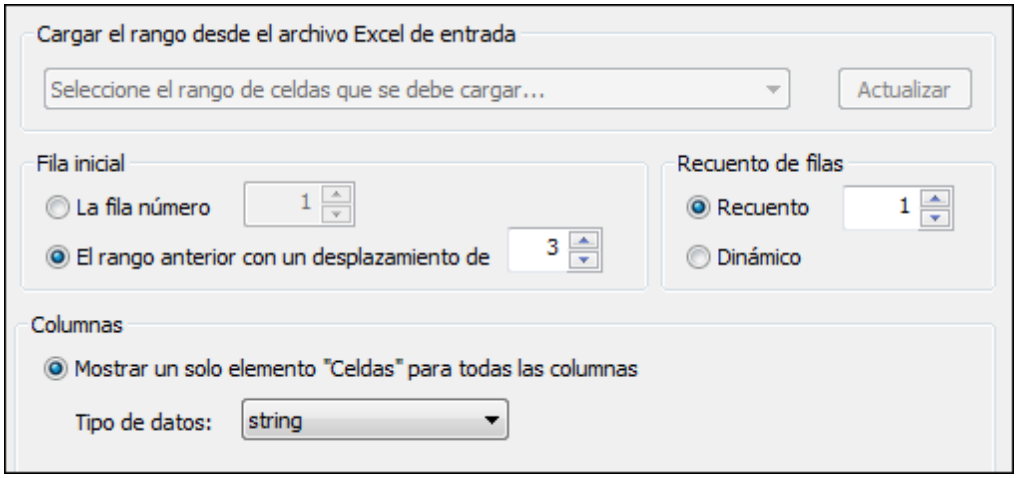
Address.

Rango nuevo: **Filas anteriores+2, n=din**

Agregar los rangos posteriores:

Utilice el mismo método para agregar rangos nuevos al componente Excel.

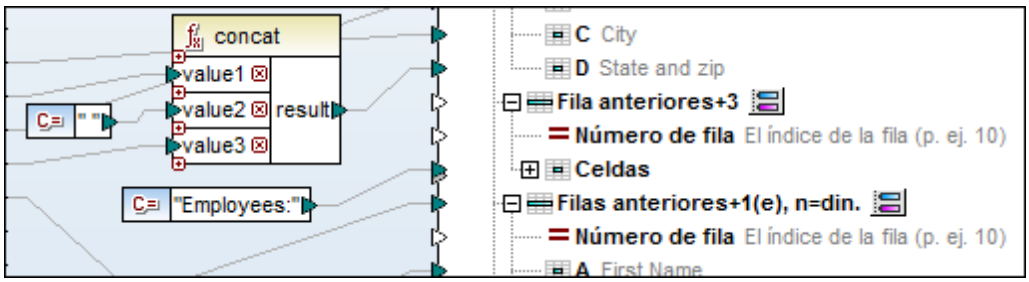
Rango Filas anteriores+3



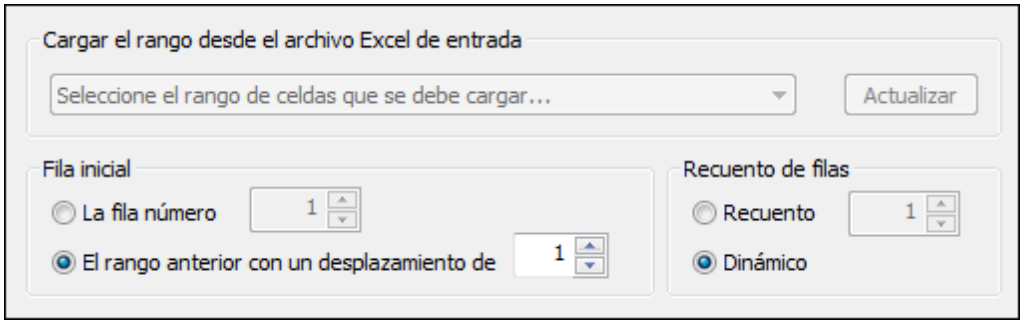
Este nuevo rango empieza en el rango anterior con un desplazamiento de 3 celdas y su recuento de filas es 1.

Observe que aquí está activa la casilla *Mostrar un solo elemento Celdas para todas las columnas.* No se definieron nombres de columna.

Esta definición de rango inserta dos filas vacías y suministra una fila que se debe rellenar con datos. Los datos de la tercera fila vienen de la constante **Employees** y está asignada al elemento **Celdas** del componente de destino.



Rango Filas anteriores+1(e), n=din



El siguiente rango empieza en el rango anterior con un desplazamiento de 1 celda, tiene un recuento de celdas dinámico y su primera fila es la fila de encabezado.

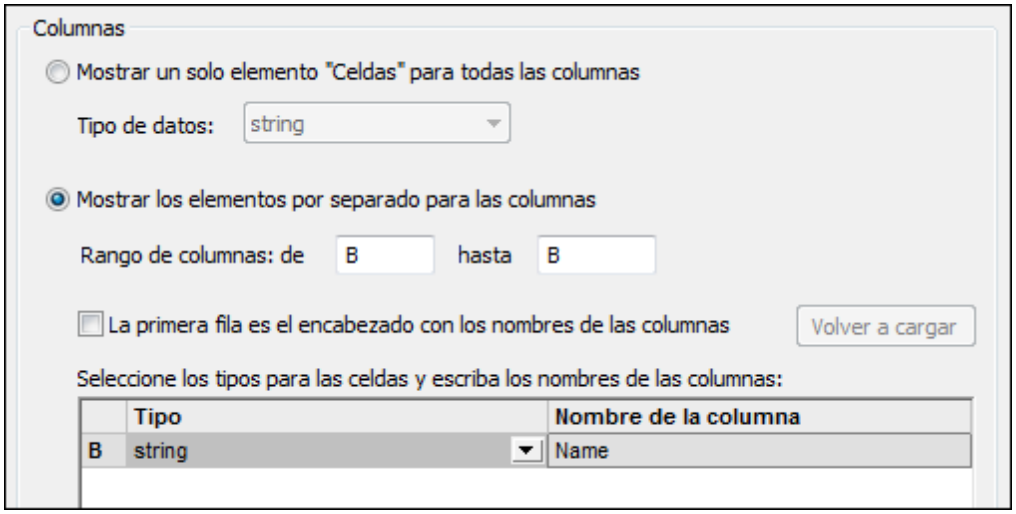
Esto significa que el número de filas que se crearán en el archivo Excel de destino es dinámico y depende de los datos suministrados por el elemento **`Person`** del componente XML de origen.

En este rango definimos los nombres de columna para las columnas A a la E siguiendo el método descrito anteriormente.

Rango `Filas anteriores+3`

Este rango se repite e inserta dos filas vacías y una tercera fila que contiene los datos de la constante **`List of departments`**.

Rango `Filas anteriores+1, n=din`



Este rango empieza en el rango anterior con un desplazamiento de 1 celda y tiene un recuento dinámico.

El rango de columnas es de la columna B a la B (es decir, solamente hay una columna y sus datos proceden del nodo **`Department | Name`** del XML de origen).

El nombre del departamento se inserta en la columna B.

Para agregar un rango en el componente Excel:

 © 2014 Altova GmbH

- Haga clic con el botón secundario en uno de los rangos ya existentes (p. ej. **`Filas 2, n=din`**) y en el menú contextual seleccione **Agregar filas delante** o **Agregar filas detrás**.

Para cambiar la posición de un rango del componente Excel:

- Haga clic con el botón secundario en el rango que desea mover y seleccione **Subir** o **Bajar** en el menú contextual.

Para quitar un rango del componente Excel:

- Haga clic en el rango que desea eliminar y seleccione la opción **Quitar filas** del menú contextual.

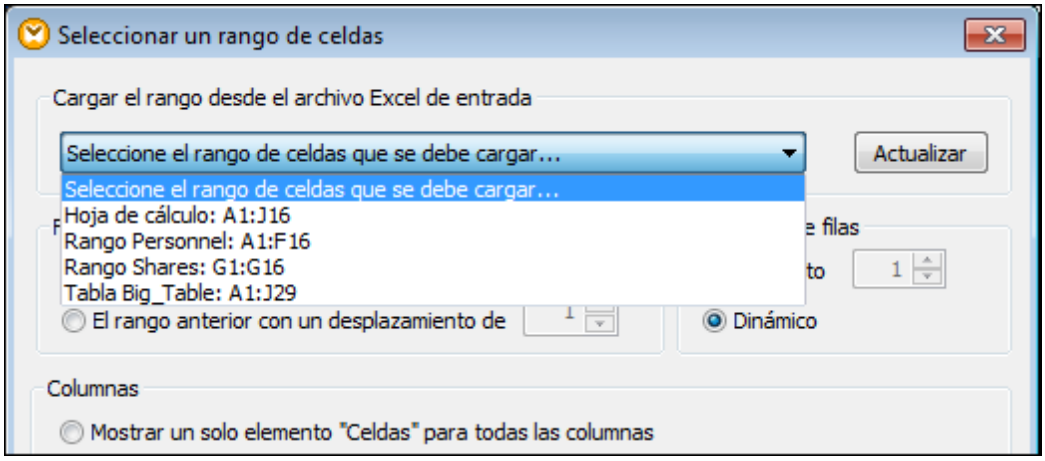
10.6.4 Seleccionar un rango de celdas (opciones)

Cargar el rango desde el archivo Excel de entrada
En este cuadro combinado puede seleccionar los nombres Excel para los rangos, las tablas Excel o la hoja de cálculo entera, si se definieron con anterioridad en el archivo Excel. Los nombres de rango/tabla, así como las referencias de celda, aparecen en el cuadro combinado desplegable.

No olvide que, por ahora, MapForce solamente es compatible con rangos rectangulares.

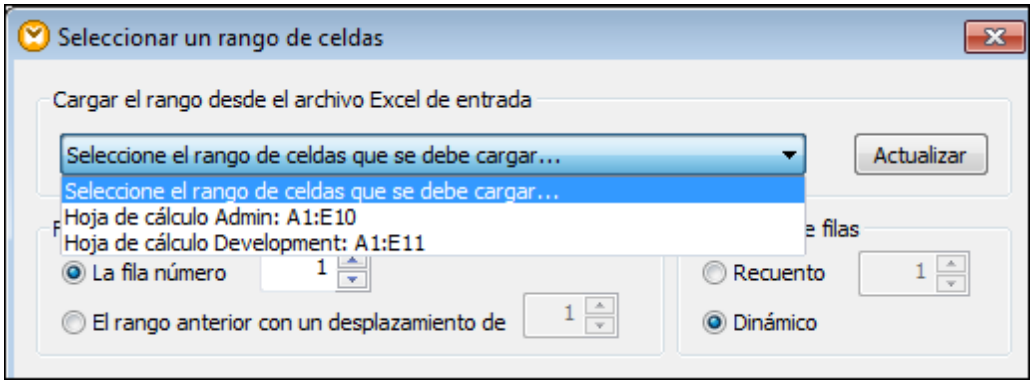
Tras seleccionar un rango y hacer clic en **Aceptar**, el rango nuevo aparece en el componente Excel. Si vuelve a abrir el cuadro de diálogo, en lugar del rango seleccionado aparece la entrada *Seleccionar el rango de celdas que se debe cargar*.

Si en el cuadro de diálogo "Agregar o quitar selecciones de hojas de cálculo de Excel" **está activa** la opción *Mostrar las hojas de cálculo por su nombre:*



El botón **Actualizar** vuelve a leer el archivo Excel y actualiza los rangos si se realizaron cambios en ellos.

Si en el cuadro de diálogo "Agregar o quitar selecciones de hojas de cálculo de Excel" **no está activa** la opción *Mostrar las hojas de cálculo por su nombre:*



La imagen anterior muestra los rangos disponibles cuando no está activa la opción *Mostrar las hojas de cálculo por su nombre.* En tal caso, las hojas de cálculo no aparecen en el componente Excel. Los rangos disponibles en el cuadro combinado *Cargar el rango...* son las hojas de cálculo

 © 2014 Altova GmbH

(**`Admin`** y **`Development`**) definidas en el archivo XLSX.

Fila inicial: la fila número...
Esta opción define la fila inicial de datos para el rango elegido. P. ej. si escribe 5 para el primer rango, significa que la fila número 5 será la primera fila.

Además si define un *Recuento* de 1 para un rango y en el siguiente rango elige la opción *El rango anterior con un desplazamiento de...*, la fila inicial de este segundo rango será la fila número 6. No obstante, el nuevo rango puede tener una definición absoluta (p. ej. Fila inicial= fila 1 y Recuento= 5 flas).

Además, si marca la casilla *La primera fila es el encabezado...*, los campos de la fila número 5 se usarán como encabezados de columna.

Fila inicial: el rango anterior con un desplazamiento de...
Esta opción define automáticamente la fila inicial del siguiente rango, es decir, se usa la última fila del rango anterior.

El desplazamiento define cuántas filas más se deben añadir al inicio de rango que se determinó automáticamente. El valor mínimo de desplazamiento es 1.

Recuento de filas: Recuento...
Cuando se definen rangos en un archivo Excel de **origen**, en el *Recuento de filas*:

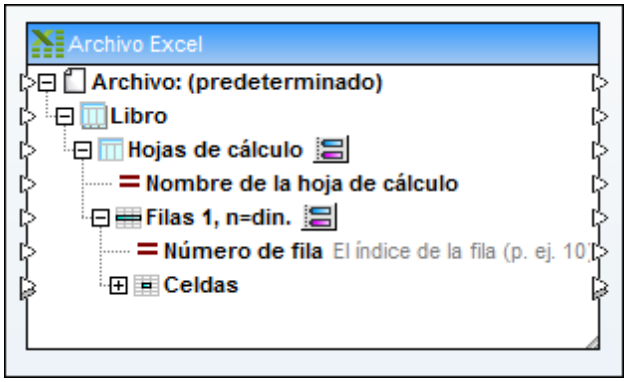
- puede insertar manualmente el número de filas que desea añadir a la *Fila inicial*
- o, si seleccionó un rango o una tabla Excel, el número de filas se determina automáticamente.

Nota: el recuento no tiene en cuenta la fila de encabezado.

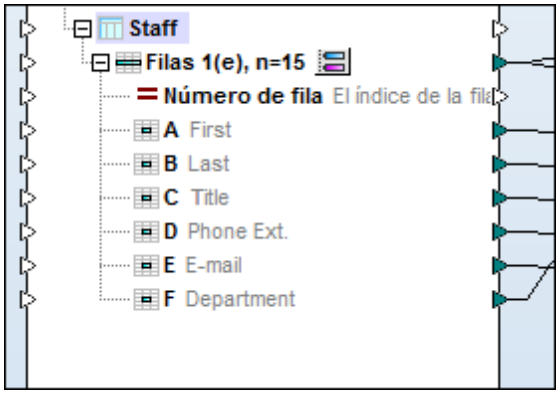
Recuento de filas: Dinámico
Cuando se definen rangos en un archivo Excel de **destino**, la opción *Dinámico* significa que el número de celdas viene dado por los datos de origen que se asignen al rango.

Cuando se definen rangos en un archivo Excel de **origen**, la opción *Dinámico* significa que el rango tiene un número ilimitado de filas. **Nota:** los demás rangos definidos en este caso no seleccionan nunca datos del archivo Excel de origen.

Columnas: *Mostrar un solo elemento Celdas para todas las columnas*
Esta opción reduce todos los elementos **`Celda`** a un solo elemento **`Celdas`** que se puede asignar. Para ver un ejemplo consulte el apartado Agregar, definir y mover rangos de filas.



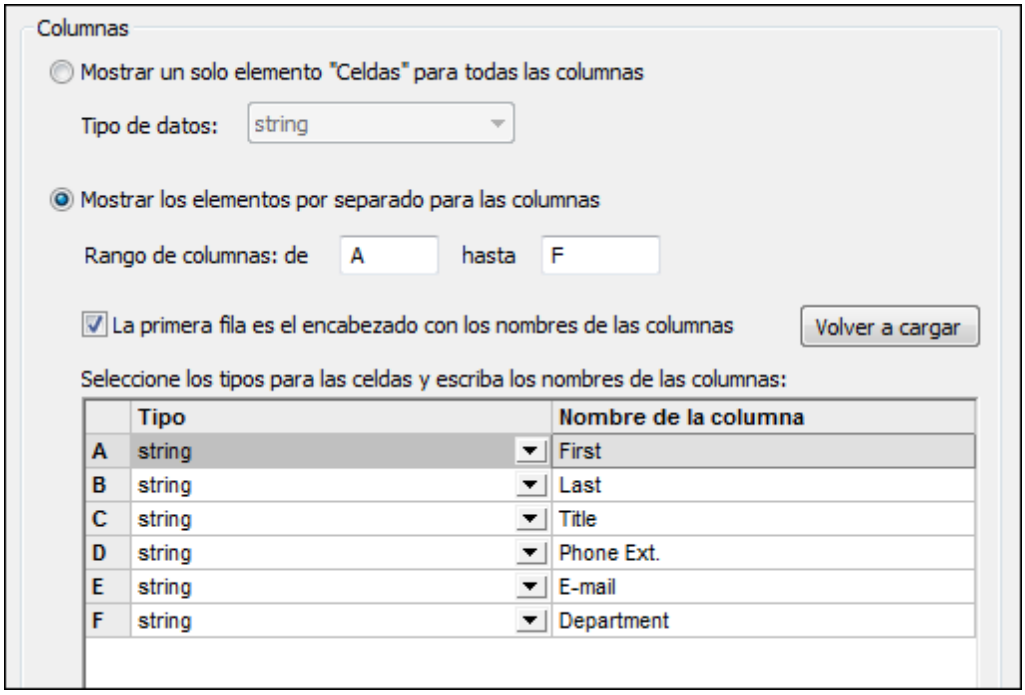
Columnas: ***Mostrar los elementos por separado para las columnas***
Esta opción muestra las hojas de cálculo del libro Excel y sirve para definir el número de columnas y sus nombres.



Si seleccionó una tabla Excel, los nombres de las columnas se insertan automáticamente. Los nombres de las columnas también se pueden definir a mano (basta con escribir los nombres en la columna **Nombre de la columna** del cuadro de lista).

Si la hoja de cálculo y la fila inicial están bien definidas y marca la casilla *La primera fila es el encabezado con los nombres de las columnas*, entonces los nombres de las columnas se definen automáticamente.

 © 2014 Altova GmbH



La primera fila es el encabezado con los nombres de las columnas
Esta opción define si la primera fila del rango se debe utilizar o no como encabezado de las columnas del rango.

Recuerde que la opción *Recuento de filas* no tiene en cuenta la fila de encabezado.

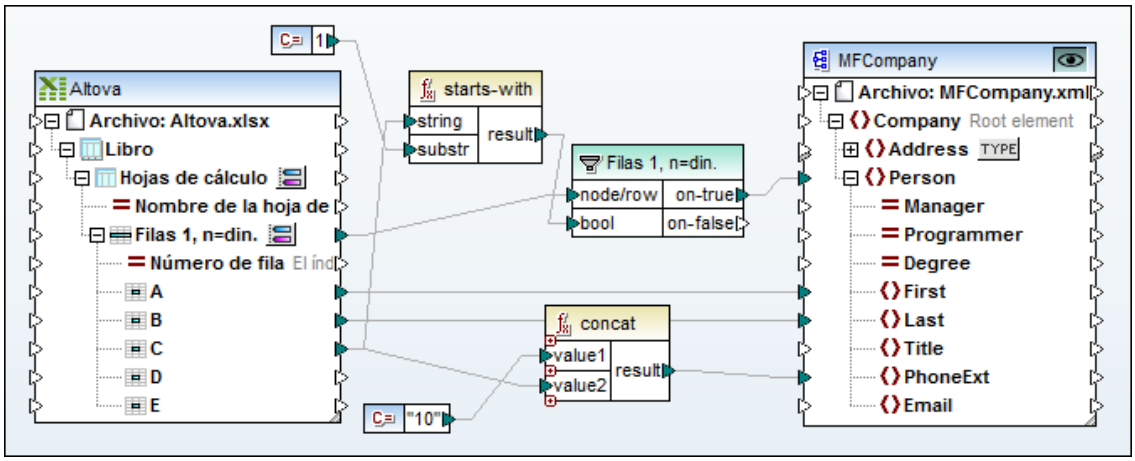
Si definió un recuento de filas, cuando active/desactive esta opción aparecerá un aviso preguntando si desea ajustar el valor del recuento de filas. Así se evita que el recuento de filas tenga una fila de más o una fila de menos.

10.6.5 Asignar archivos Excel a XML

El archivo de asignación utilizado como ejemplo en este apartado es el archivo `Excel-mapping.mfd` de la carpeta `...\MapForceExamples\Tutorial`. En este archivo encontrará 3 asignaciones de datos, pero este apartado describe solamente las dos primeras.

El **objetivo** de la primera asignación es:

- filtrar las personas del libro Excel cuyo campo **PhoneExt** empiece por "1" (en la columna C del libro Excel).
- agregar/concatenar el número "10" al número original y colocarlo en el archivo XML `MFCompany` junto con los datos de los campos **First** y **Last** (nombre y apellido) de la persona correspondiente.



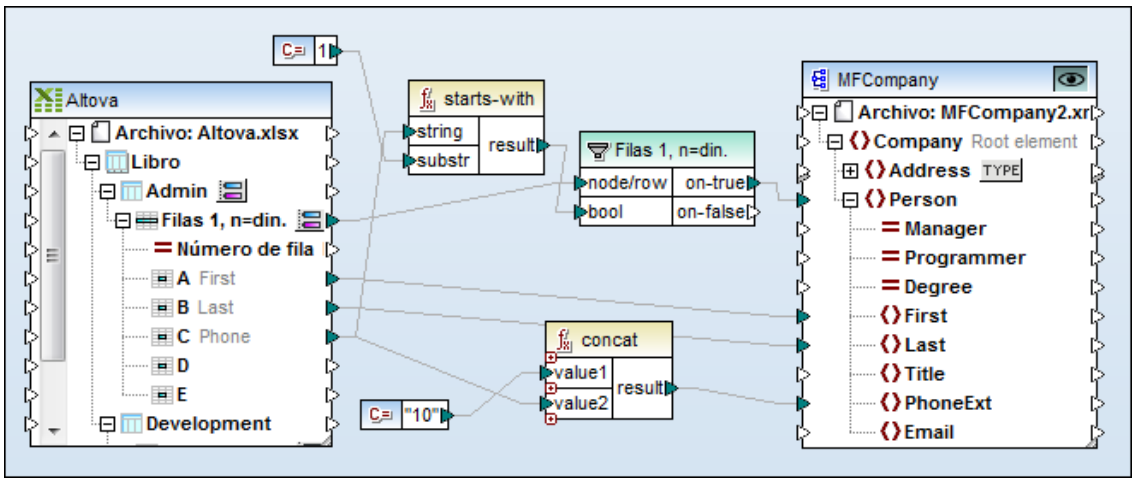
- **Altova.xlsx** es el componente de origen. Las columnas A y B ofrecen los campos **First** y **Last** respectivamente (nombre y apellido) de las personas. La columna C ofrece el número de extensión de teléfono (**PhoneExt**).
- Las hojas de cálculo del libro no aparecen por separado en el componente (se desactivó la opción *Mostrar las hojas de cálculo por su nombre*). Como puede ver, bajo el elemento **`Libro`** aparece el nombre genérico **`Hojas de cálculo`**.
- La función **`starts-with`** comprueba si la extensión telefónica (columna C) empieza por "1" y si el resultado es `true`, el componente de filtrado envía esos registros al componente de destino.
- La función **`concat`** añade el prefijo "10" a las extensiones telefónicas y coloca el número final en el elemento **`PhoneExt`** del componente de destino.
- El esquema `MFCompany.xsd` es el componente de destino y el archivo de salida incluirá los datos de las personas que pasen el filtro.

Resultado de la asignación:
Un total de 4 personas, con sus respectivos datos, se asignaron al archivo XML.

```
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 panel **Asignación** es igual a la anterior, con la única diferencia de que las hojas de cálculo se presentan una por una en el componente Excel (se activó la opción *Mostrar las hojas de cálculo por su nombre*)

- Las hojas de cálculo **`Admin`** y **`Development`** aparecen bajo el elemento **`Libro`**.
- Solamente se crearon conexiones entre la hoja de cálculo **`Admin`** y el componente de destino.



Resultado de la asignación:
Un total de 2 personas, con sus respectivos datos, se asignaron al archivo XML.

```
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>
```

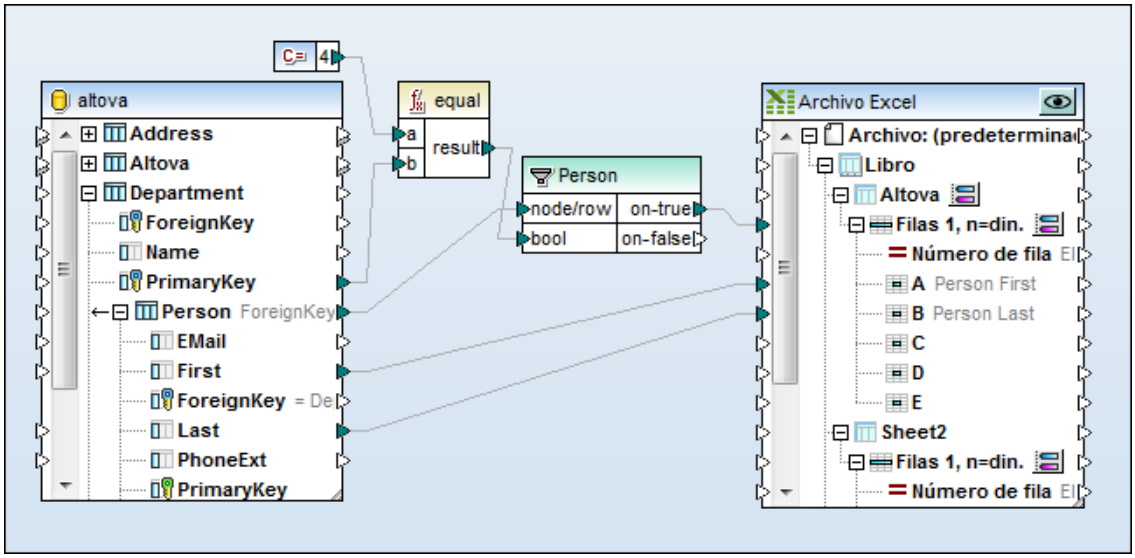

© 2014 Altova GmbH

10.6.6 Asignar datos de BD a Excel

El archivo de asignación utilizado como ejemplo en este apartado es el archivo `Excel-mapping.mfd` de la carpeta `...\MapForceExamples\Tutorial`. Este archivo contiene 3 asignaciones de datos, pero este apartado solamente describe la tercera y última asignación.

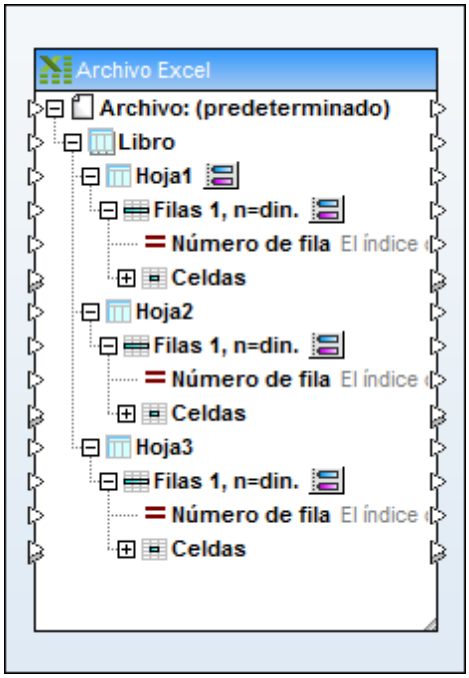
El **objetivo** de esta asignación es:

- filtrar las personas de la base de datos Altova cuya clave primaria de departamento sea 4, es decir, las personas del departamento de TI.
- insertar las personas del departamento de TI en la plantilla Excel de destino, que está vacía



Para crear un libro Excel vacío (una plantilla):

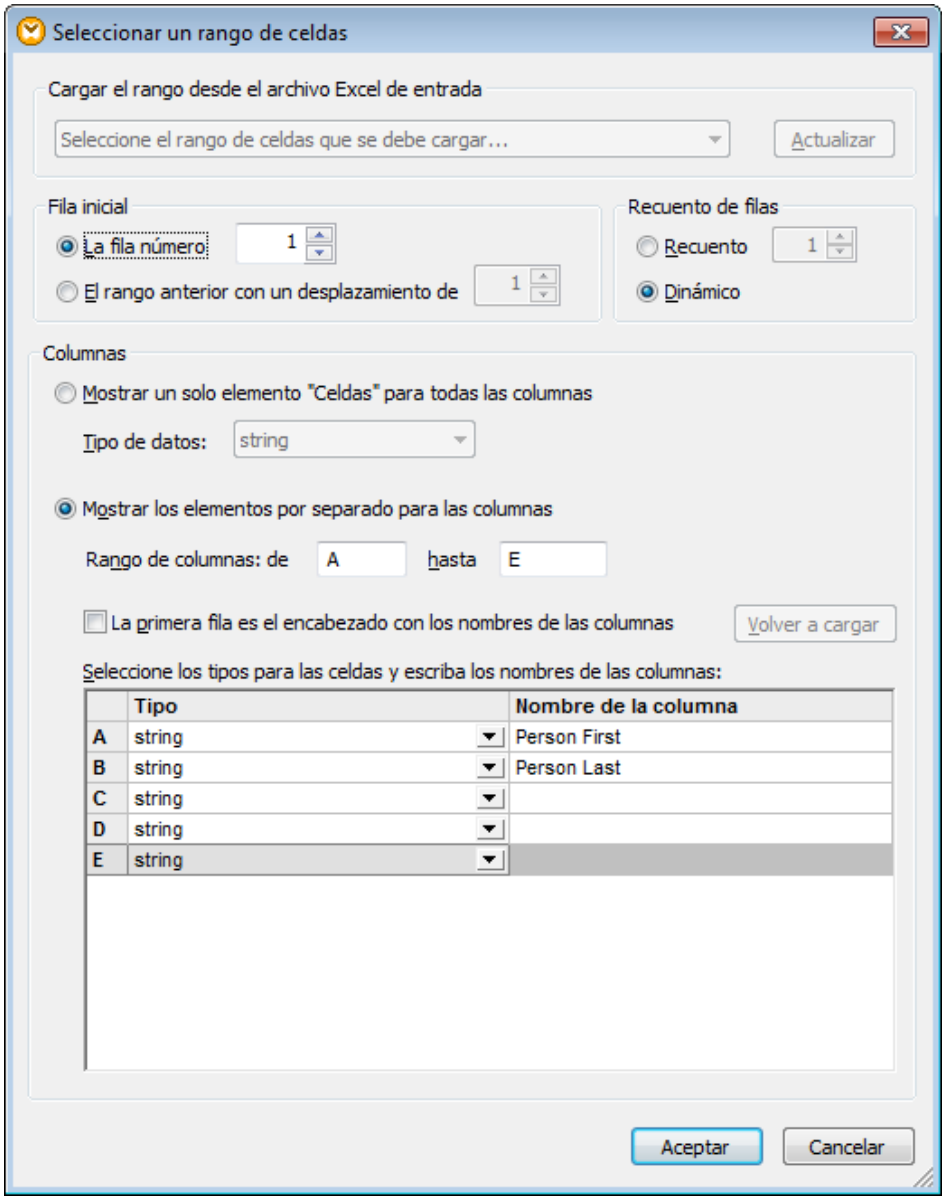
1. Haga clic en el icono **Insertar archivo Excel 2007+** de la barra de herramientas. Aparece un aviso solicitando un archivo XLSX de muestra que incluya los datos necesarios para la vista previa.
2. Haga clic en **Omitir**.



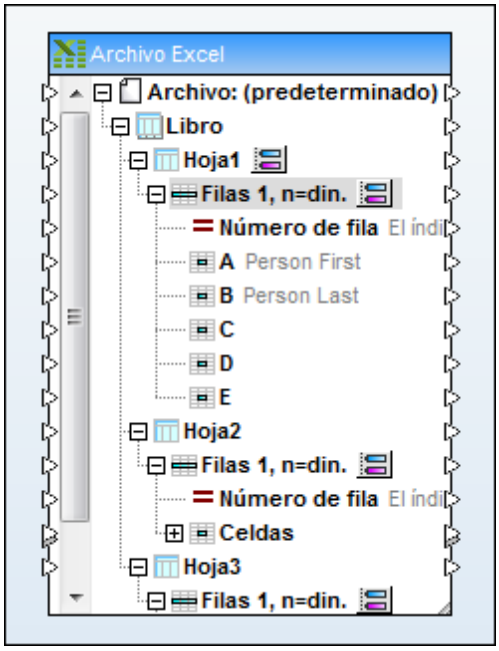
Se inserta un componente Excel vacío.

3. Haga clic en el botón interactivo de **`Filas 1, n=din`** situado bajo el elemento **`Hoja1`** para definir el número de columnas que debe tener la hoja de cálculo.
4. Seleccione el botón de opción *Mostrar los elementos por separado para las columnas* y en los cuadros de texto introduzca el rango **de A hasta E**.

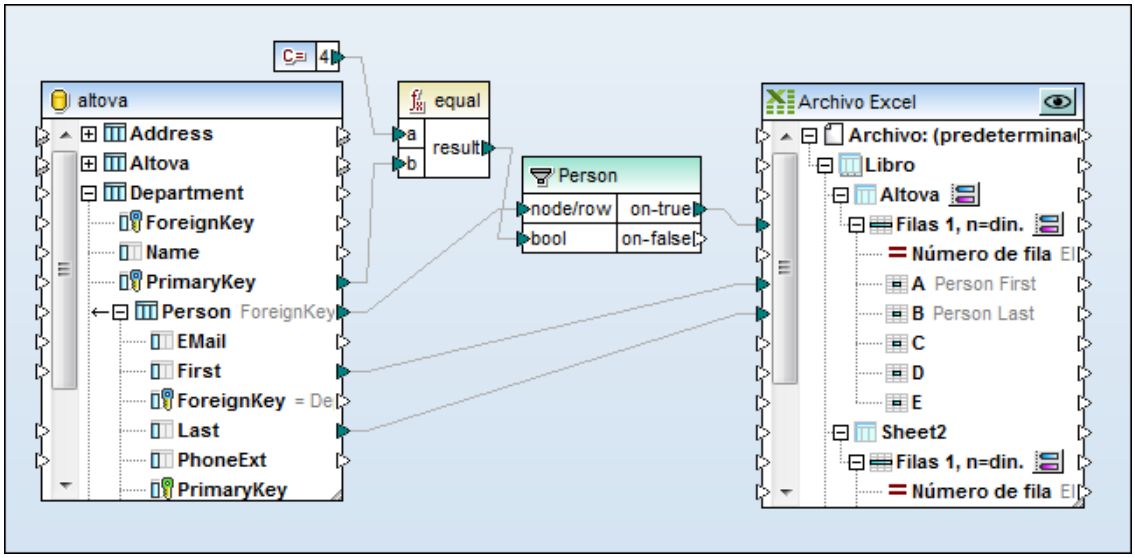
 © 2014 Altova GmbH



5. Haga clic en los campos **Nombre de la columna** y escriba un nombre que facilite la asignación de datos (p. ej. `Person First` y `Person Last`) y haga clic en **Aceptar**.



Ahora la hoja de cálculo contiene las columnas A - E, que se pueden asignar a otro componente.



- El valor de **`PrimaryKey`** se compara por medio de la función **`equal`** con el valor **`4`**, que viene dado por una constante.
- El componente de filtrado pasa los campos **`First`** y **`Last`** si la condición booleana (**`bool`**) da **`true`** como resultado. El contenido de los nodos secundarios conectados al parámetro **`node/row`** del componente de origen se envían al componente de destino.
- Observe que el elemento **`on-true`** del filtro está conectado al elemento **`Filas 1, n=din`** del archivo Excel.
- El nombre de la hoja de cálculo, **`Altova`**, se definió haciendo clic en el icono interactivo de **`Hoja1`** .

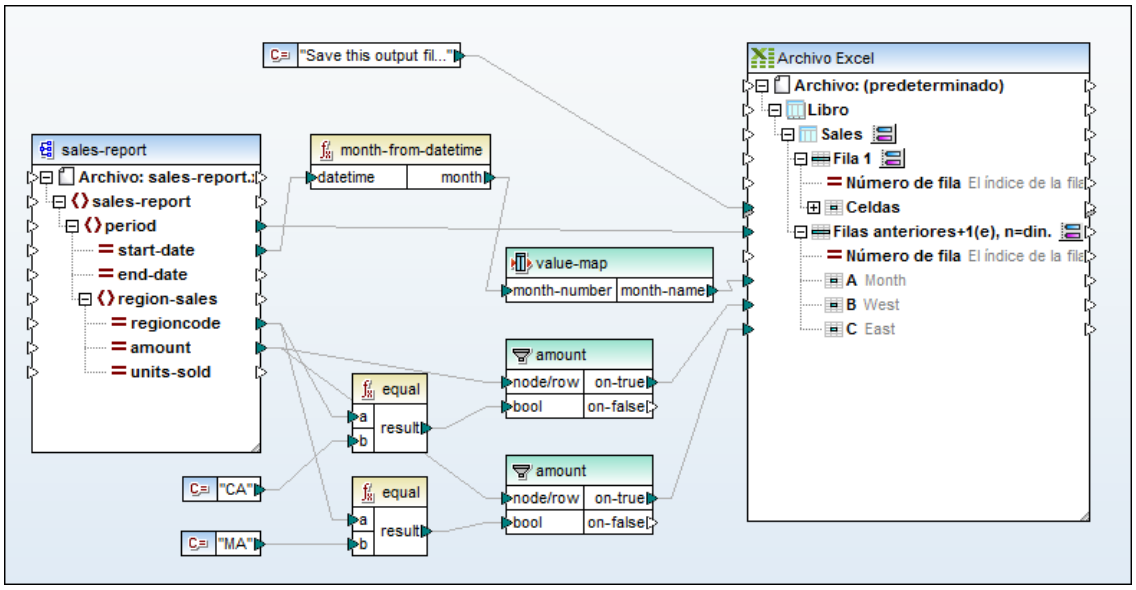
Resultado de la asignación:
Las 4 personas del departamento de TI se pasaron al libro Excel.

	A	B	C	D
1	Alex	Martin		
2	George	Hammer		
3	Jessica	Bander		
4	Lui	King		
5				
6				

Altova

10.6.7 Añadir datos a hojas Excel con formato previo

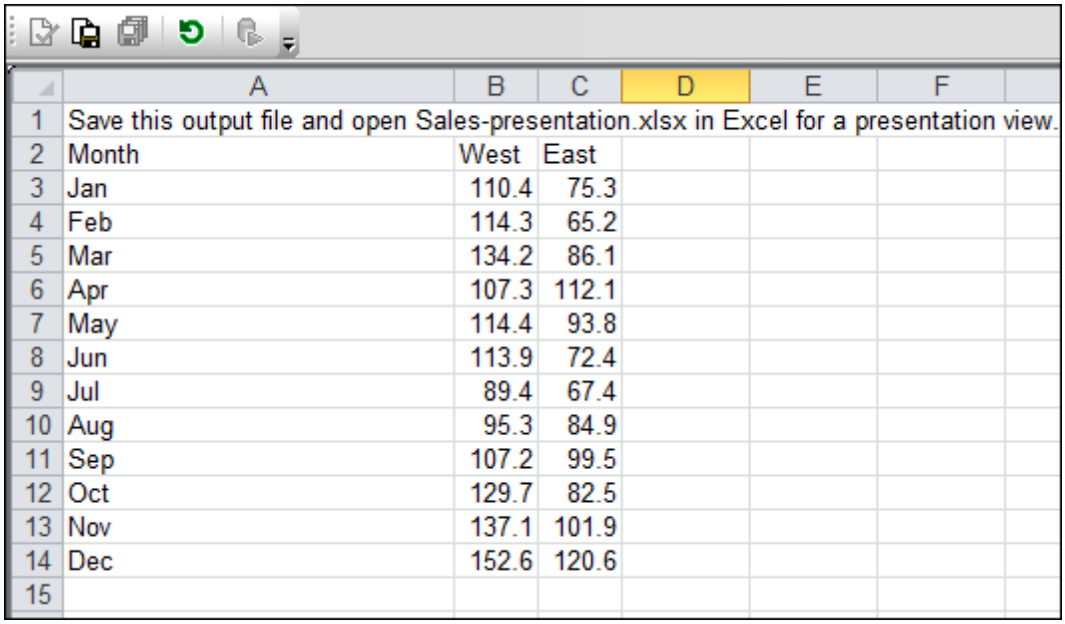
Los datos de MapForce que se envían a una hoja de cálculo Excel se pueden enviar como origen de datos para documentos Excel con formato previo. La imagen que aparece a continuación muestra la asignación de ejemplo `Sales_to_Excel.mfd` (disponible en la carpeta `...\MapForceExamples`).



Para guardar el resultado de la asignación como origen de datos Excel sin formato:

1. Haga clic en el icono **BUILTIN** para seleccionar el lenguaje de salida (también puede seleccionar un lenguaje de programación y compilar el código).
2. Haga clic en el botón **Resultados** para ver el resultado en forma de hoja Excel incrustada en MapForce.
3. Haga clic en el icono **Guardar resultado generado** y guarde el archivo con el nombre `xlsx-mapforce.xlsx`. Después haga clic en **Guardar** para guardar el libro en la carpeta `...\MapForceExamples`.

 © 2014 Altova GmbH

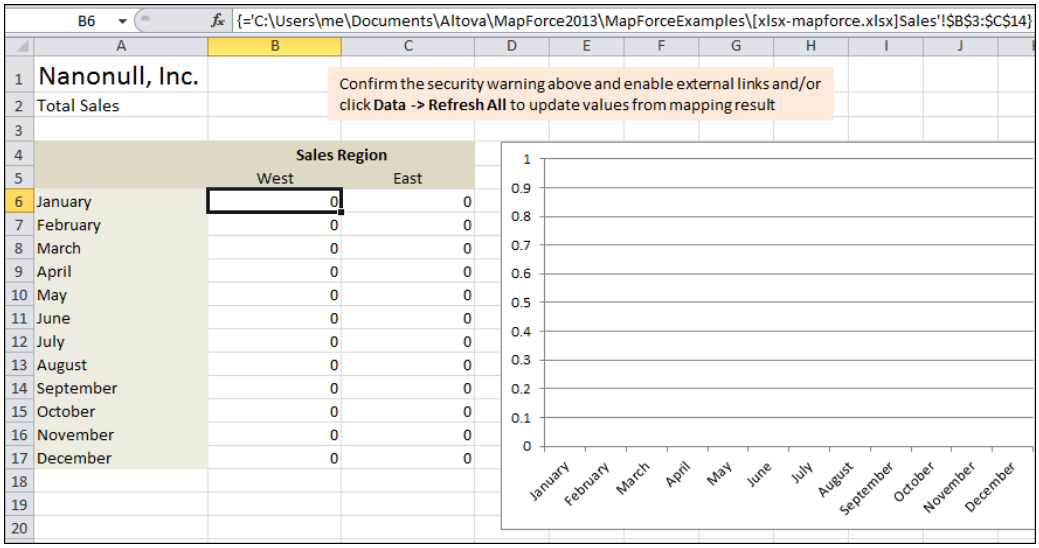


	A	B	C	D	E	F
1	Save this output file and open Sales-presentation.xlsx in Excel for a presentation view.					
2	Month	West	East			
3	Jan	110.4	75.3			
4	Feb	114.3	65.2			
5	Mar	134.2	86.1			
6	Apr	107.3	112.1			
7	May	114.4	93.8			
8	Jun	113.9	72.4			
9	Jul	89.4	67.4			
10	Aug	95.3	84.9			
11	Sep	107.2	99.5			
12	Oct	129.7	82.5			
13	Nov	137.1	101.9			
14	Dec	152.6	120.6			
15						

Dar formato al archivo de plantilla Excel:

Lo normal sería que, llegados a este punto, tuviéramos que crear un archivo Excel nuevo que serviría para mostrar los datos de MapForce. En él podríamos añadir fórmulas que hagan referencia a los datos del archivo de salida de MapForce, añadir un gráfico en la hoja, etc. Sin embargo, no hace falta hacer todo esto porque ya tenemos un archivo Excel con formato previo llamado `Sales-presentation.xlsx` que está en la carpeta `...\MapForceExamples`.

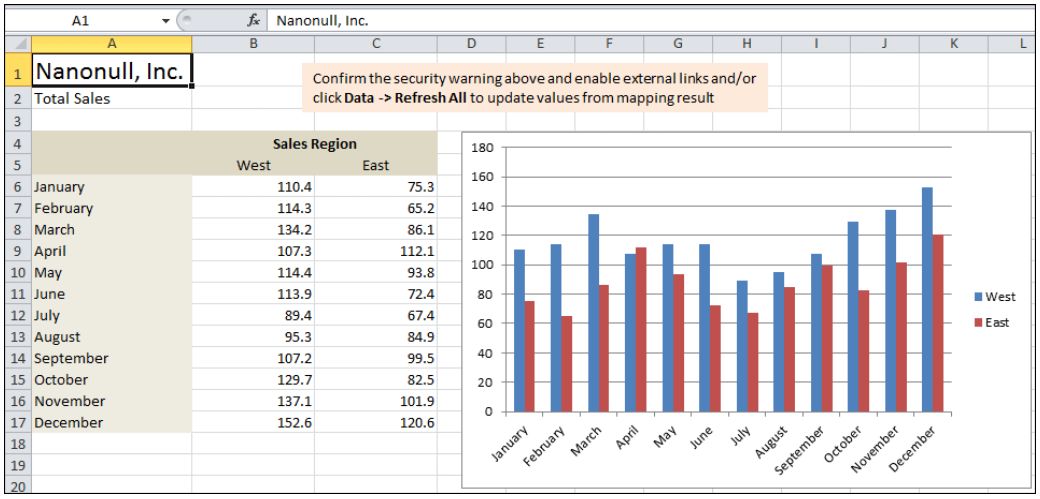
1. Haga doble clic en el archivo `Sales-presentation.xlsx`. Aparece una aviso advirtiendo de que se deshabilitaron los vínculos externos y que es necesario habilitarlos para crear enlaces con los datos de origen.



La posición actual del cursor, B6, muestra que el intervalo de los datos de origen es **Sales'!$B$3:$C$14** del archivo `xlsx-mapforce.xlsx`.

2. Haga clic en el botón **Habilitar** (o seleccione la opción de menú **Datos | Actualizar**

todo).
Los vínculos externos se habilitan y los datos de origen subyacentes aparecen en forma de tabla y de gráfico.



10.7 Asignación de datos XBRL

Sitio web de Altova: Asignación de datos XBRL

El conjunto de herramientas de Altova es compatible con las especificaciones XBRL 2.0 y XBRL Dimensions 1.0 y cada producto de Altova se ocupa de un aspecto del estándar XBRL.

XMLSpy 2014 crea/edita taxonomías nuevas (y genera informes XBRL a partir de ellas si hay disponible un archivo de transformación XSLT o si se creó un archivo XSLT en StyleVision).

StyleVision 2014 crea hojas de estilos/plantillas de taxonomías para generar informes XBRL.

MapForce 2014 crea asignaciones de datos entre taxonomías/archivos de instancia XBRL y archivos Excel 2007+, bases de datos y archivos CSV. Con el resultado de las asignaciones de datos se pueden crear informes o filtrar los datos de los documentos XBRL.

MapForce también permite crear asignaciones de datos entre documentos XBRL basados en taxonomías como:

- US-GAAP, incluida la versión US-GAAP 2012
- NIIF (IFRS)
- COREP / FINREP

En las asignaciones de datos de taxonomías/documentos XBRL se asume que el archivo de taxonomía XBRL subyacente está disponible. Los archivos de taxonomías de cada empresa se crean a partir de una taxonomía ya existente, como US-GAAP, por ejemplo. La taxonomía nueva amplía la taxonomía base y le añade un atributo y un prefijo de espacio de nombres nuevos.

Asignaciones de datos entre una BD y una taxonomía
El archivo de ejemplo `DB_to_XBRL.mfd` explica cómo se asignan datos de una base de datos MS Access a una taxonomía XBRL para generar un archivo XBRL de instancia válido.

Asignaciones de datos entre una hoja Excel y una taxonomía
El archivo de ejemplo `boa-balance-sheet.mfd` explica cómo asignar datos de una hoja de cálculo Excel a una taxonomía XBRL para generar un archivo XBRL de instancia válido.

10.7.1 Insertar documentos XBRL

En MapForce los documentos XBRL se pueden insertar en el panel **Asignación** de dos formas:

- como documento XML plano
- como documento XBRL jerárquico

Insertar un documento XBRL como documento XML
Seleccione la opción de menú **Insertar | Insertar archivo o esquema XML**.
El archivo XBRL se inserta como archivo XML sin conceptos de lógica de negocio ni estructura jerárquica XBRL. Esta opción es útil para asignaciones sencillas en las que no es necesario ocuparse de dimensiones ni de generar contextos XBRL automáticamente.

Insertar un documento XBRL como documento/taxonomía XBRL

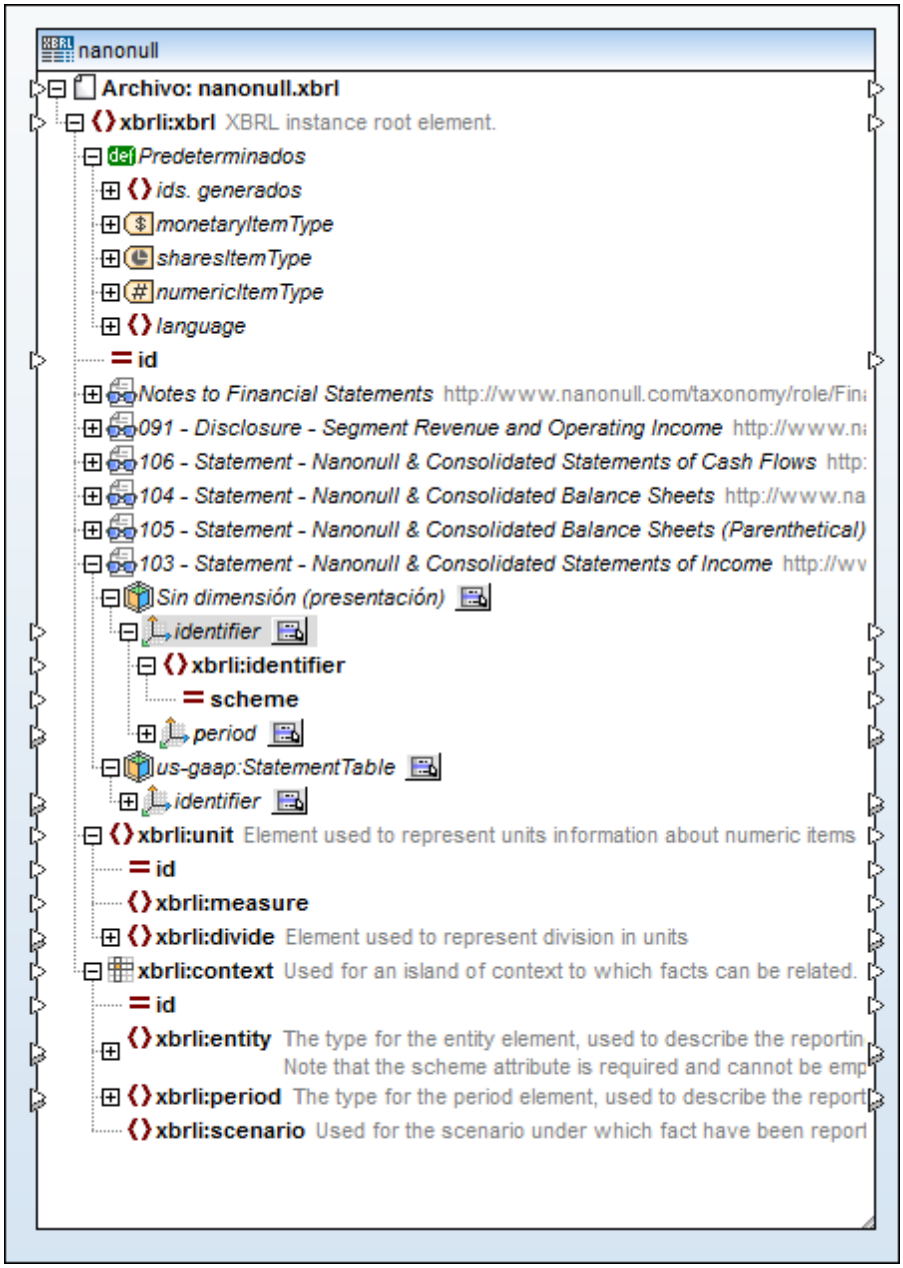
1. Seleccione la opción de menú **Insertar | Insertar documento XBRL**
2. Seleccione el archivo de taxonomía (*.xsd) o el archivo XBRL de instancia (*.xbrl o *.xml). Si selecciona una taxonomía aparece un cuadro de diálogo donde puede seleccionar un archivo de instancia válido.
3. Haga clic en **Examinar** si desea usar este componente XBRL como instancia de origen y seleccione el archivo XBRL de instancia.

Esto inserta un componente de taxonomía XBRL que presenta los contenidos de la taxonomía de forma jerárquica.

La estructura XBRL del documento muestra todas los extractos contables. Ofrece una gestión automática de los contextos a la hora de leer y escribir datos de dimensión de los hipercubos y a la hora de acceder a conceptos contables, definidos por la taxonomía XBRL. Los **valores** de estos elementos se denominan **hechos**.

La imagen siguiente muestra una jerarquía XBRL típica y ofrece un resumen de todos los elementos que suelen formar parte de un documento XBRL:

 © 2014 Altova GmbH



Elementos de un componente XBRL

Elemento raíz XBRL	El elemento **`xbrli:xbrl`** es el elemento raíz de instancia de todos los componentes XBRL. El elemento raíz contiene los extractos contables, las unidades **`xbrli:unit`**, los contextos **`xbrli:context`** y dos elementos especiales llamados **`Todos los conceptos`** y **`Todos los conceptos (sin procesar)`**. Estos dos últimos elementos se pueden activar/ocultar marcando la casilla correspondiente en el cuadro de diálogo "Configuración del componente" del componente XBRL.

Extracto contable	Ejemplo: *http://xbrl.boa.com/2006-12-31/ext/CashFlow* Los extractos contables representan funciones de vínculo extendido de las bases de enlaces de definición y de presentación de una taxonomía XBRL.
Hipercubo	Ejemplos: *Sin dimensión (presentación)* y *Statement [Table]* Los hipercubos usan información de las bases de enlace de definición y de presentación para dar una estructura jerárquica a las dimensiones, a los contextos y a otros conceptos XBRL relacionados. Hay varios tipos de hipercubos.
Dimensiones del hipercubo	Ejemplo: *Statement, Equity Components [Axis]* (segment) Las dimensiones XBRL se usan para estructurar la información contextual para *hechos* (datos) comerciales. En la taxonomía las dimensiones se definen dentro de un hipercubo. Además, MapForce muestra dos dimensiones predeterminadas, la dimensión *identifier* y la dimensión *period* . Hay dos tipos de dimensiones: explícitas o con tipo.
Valores de la dimensión	Ejemplo: **Retained Earnings [Member]** Se trata de los valores de miembro de dominio definidos de una dimensión.
Contexto	**xbrli:context** El nodo de contexto es el contenedor para todos los *hechos* (datos) comerciales. Los nodos de contexto situados dentro de un hipercubo se ocupa 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 tratamiento manual de las dimensiones.
Elemento abstracto	*Statement of Cash Flows [Abstract]* Los elementos abstractos se usan para organizar *hechos* relacionados. Los elementos abstractos se 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 elemento no existen realmente en el archivo XBRL de instancia, pero sirven para agrupar hechos de forma intuitiva.
Hechos	Los hechos son los valores de los elementos XBRL. Hay varias tipos de hechos, por ejemplo, el tipo monetario **Net Income (Loss)**

 © 2014 Altova GmbH

Tuplas	Ejemplo: **Accountants Report** Las tuplas son elementos complejos cuyos miembros son hechos y otras tuplas.
Unidad	**xbrli:unit** El elemento de unidad **`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`. Al definir este valor con una función de constante, se les aplica a todos los hechos del documento XBRL de instancia.
Nota al pie	**footnote** Las notas al pie permiten asignar información adicional a los conceptos.

Tipos de hechos y elementos XBRL

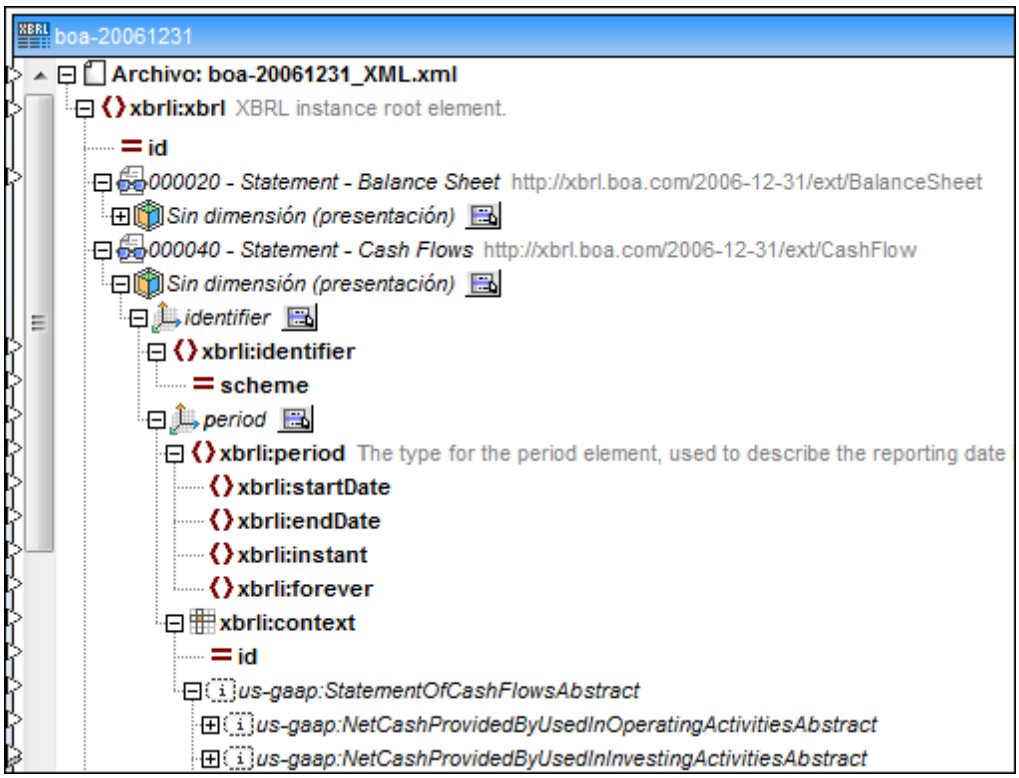
MapForce usa unos intuitivos iconos para identificar los diferentes tipos de hechos XBRL en la estructura jerárquica del componente.

Elementos de tipo monetario	**Debt Securities** xbrli:monetaryItemType
Elementos de tipo string	**Accrual for Environmental Loss Contingencies, Caption** xbrli:stringItemType **Accretion Expense [Member]** us-types:domainItemType **Accounts Payable and Accrued Liabilities Disclosure [Text Block]** us-types:text
Elementos de tipo numérico	**Debt Instrument, Convertible, Conversion Ratio** xbrli:decimalItemType **Bankruptcy Claims, Number Claims Filed** xbrli:integerItemType **Recorded Third-Party Environmental Recoveries, Discount Rate** us-types:perce
Elementos generales	**Adoption Date of SFAS 156** xbrli:dateItemType
Elementos de tipo shares	**Common Stock, Shares Outstanding** xbrli:sharesItemType

10.7.2 Componentes XBRL

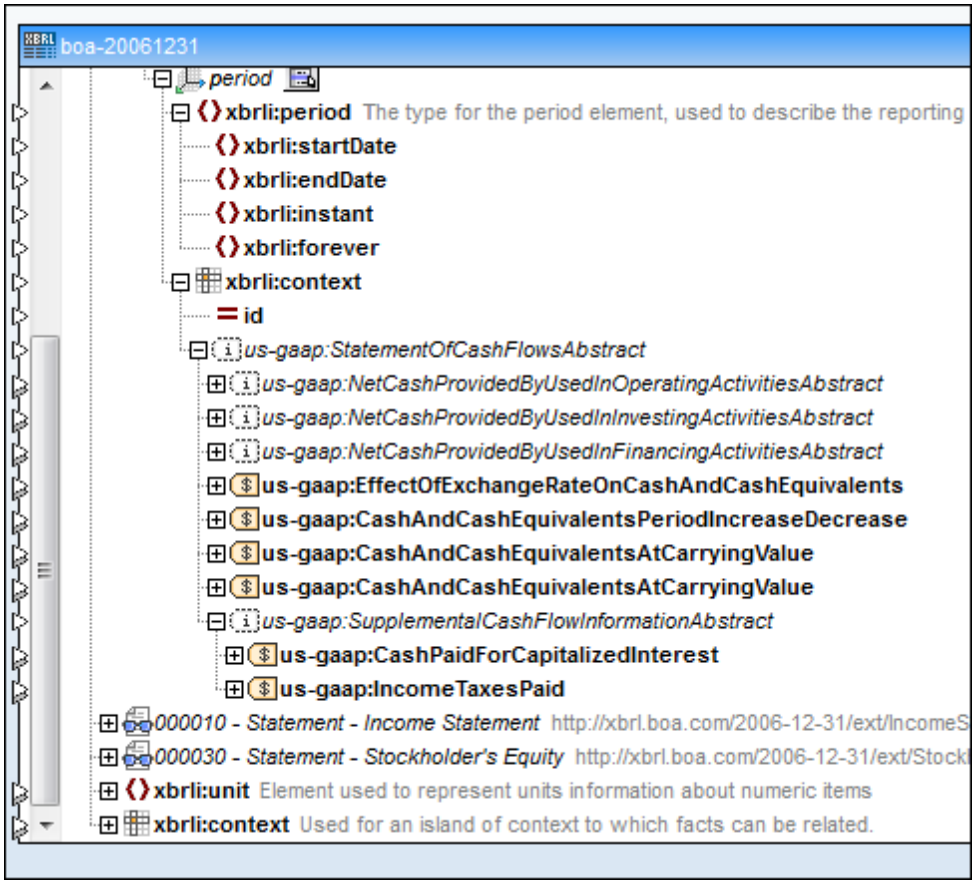
A continuación puede ver parte de un archivo XBRL en MapForce y algunos de sus elementos:

- El elemento raíz XBRL
- Extractos contables (p. ej. *00040 - Statement - Cash Flows*)
- Un hipercubo sin dimensión con las dimensiones predeterminadas **`identifier`** y **`period`**
- Elementos abstractos (p. ej. *StatementOfCashFlowsAbstract*)
- Conceptos/hechos XBRL (p. ej. *Effect of Exchange Rate Changes on Cash and Cash Equivalents*)



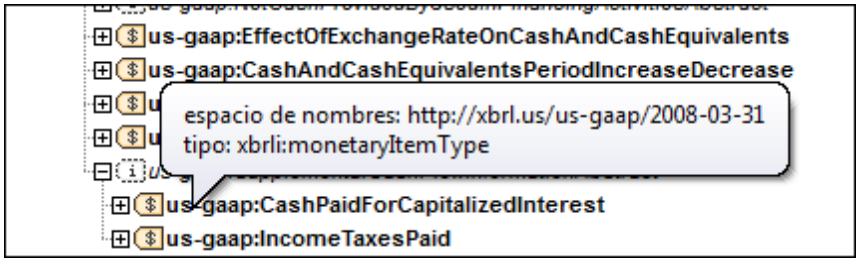
Si expandimos el elemento abstracto *Statement Of Cash Flows Abstact*, vemos que incluye más elementos abstractos (con un icono transparente) y otros elementos/hechos (con un icono color carne).

 © 2014 Altova GmbH

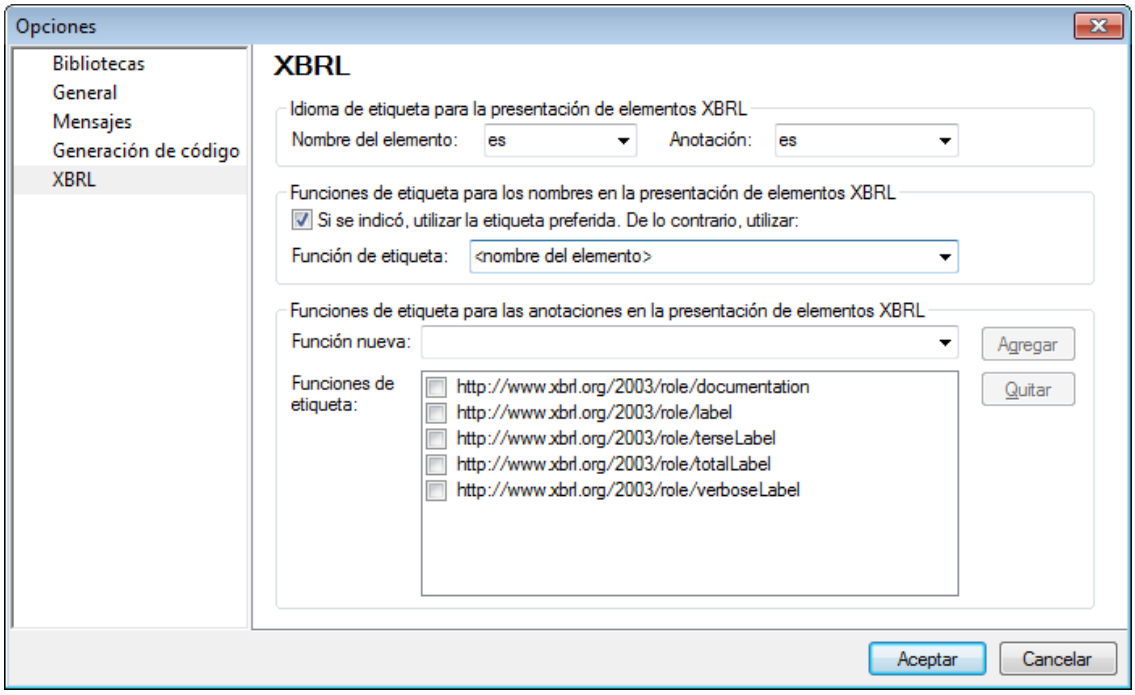


Cómo se presentan los elementos/hechos en el componente:
En la taxonomía los nombres de conceptos XBRL se definen en la base de enlaces de etiqueta. MapForce puede presentar en el componente cualquier función de etiqueta (configurando las opciones de visualización XBRL) o el nombre de elemento XML sin procesar.

Observe que, al pasar el puntero del mouse por un elemento abstracto o por un hecho, aparece información sobre el espacio de nombres, el nombre local o el tipo de datos (p. ej. `monetaryItemType`).



Para definir las opciones globales para las etiquetas XBRL haga clic en la opción de menú **Vista | Opciones de visualización XBRL** . Para más información consulte el apartado Vista de la *Referencia del usuario*.

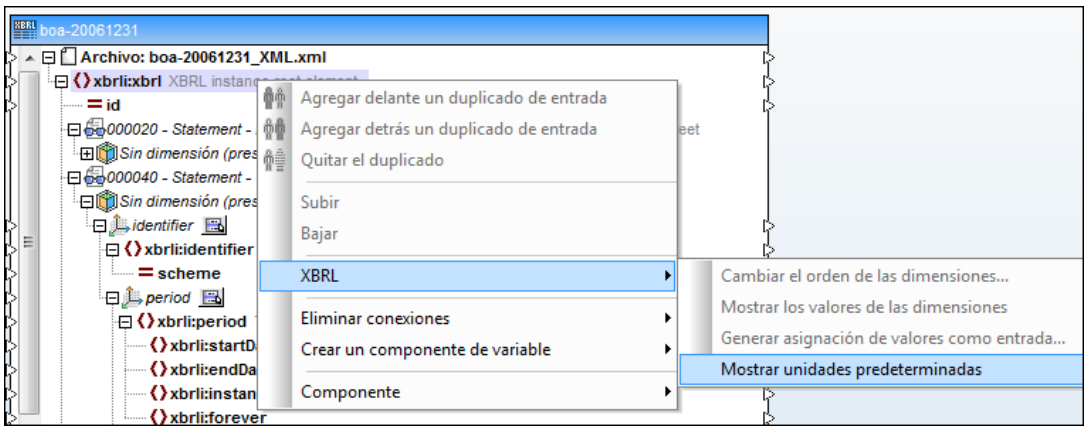


Unidades predeterminadas

Puede habilitar/definir unidades predeterminadas para todo el documento XBRL, para extractos contables, para hipercubos y para dimensiones de hipercubos.

Para ver las unidades predeterminadas:

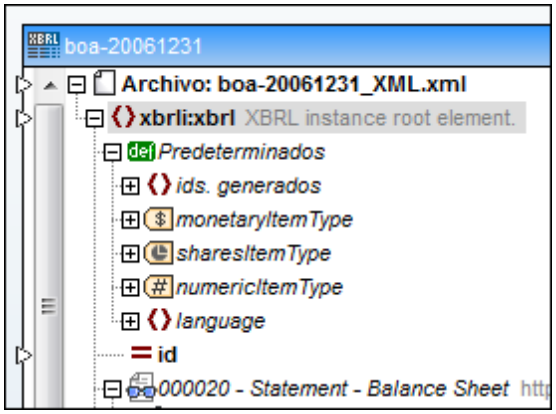
1. Haga clic con el botón secundario en el elemento para el que desea definir valores predeterminados (el elemento raíz en este caso) y seleccione **XBRL | Mostrar unidades predeterminadas**.



Si se trata de un hipercubo o de una dimensión, también puede hacer clic en su botón interactivo y seleccionar la misma opción en el menú emergente.

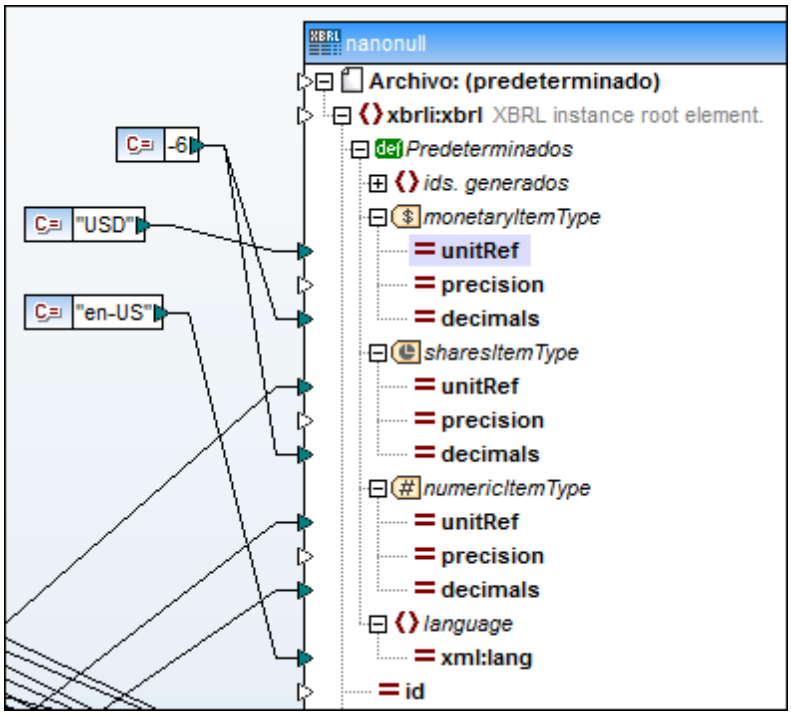
Como resultado se inserta el elemento `Predeterminados` que permite conectar valores predeterminados personales para los diferentes elementos.

 © 2014 Altova GmbH



Se trata de una eficaz manera de asignar valores a atributos presentes en el componente XBRL sin necesidad de crear conexiones explícitas con ellos. Los valores predeterminados se usan para todos los atributos relacionados presentes en el árbol jerárquico.

Por ejemplo, si al atributo **`unitRef`** del elemento *monetaryItemType* del árbol *Predeterminados* le asignamos una constante con un identificador de unidad, este valor predeterminado se asigna a todos los identificadores de unidad (**`unitRef`**) del elemento XBRL de tipo monetario (*monetaryItemType*) presente en el componente (excepto a aquellos que tengan otras asignaciones explícitas).



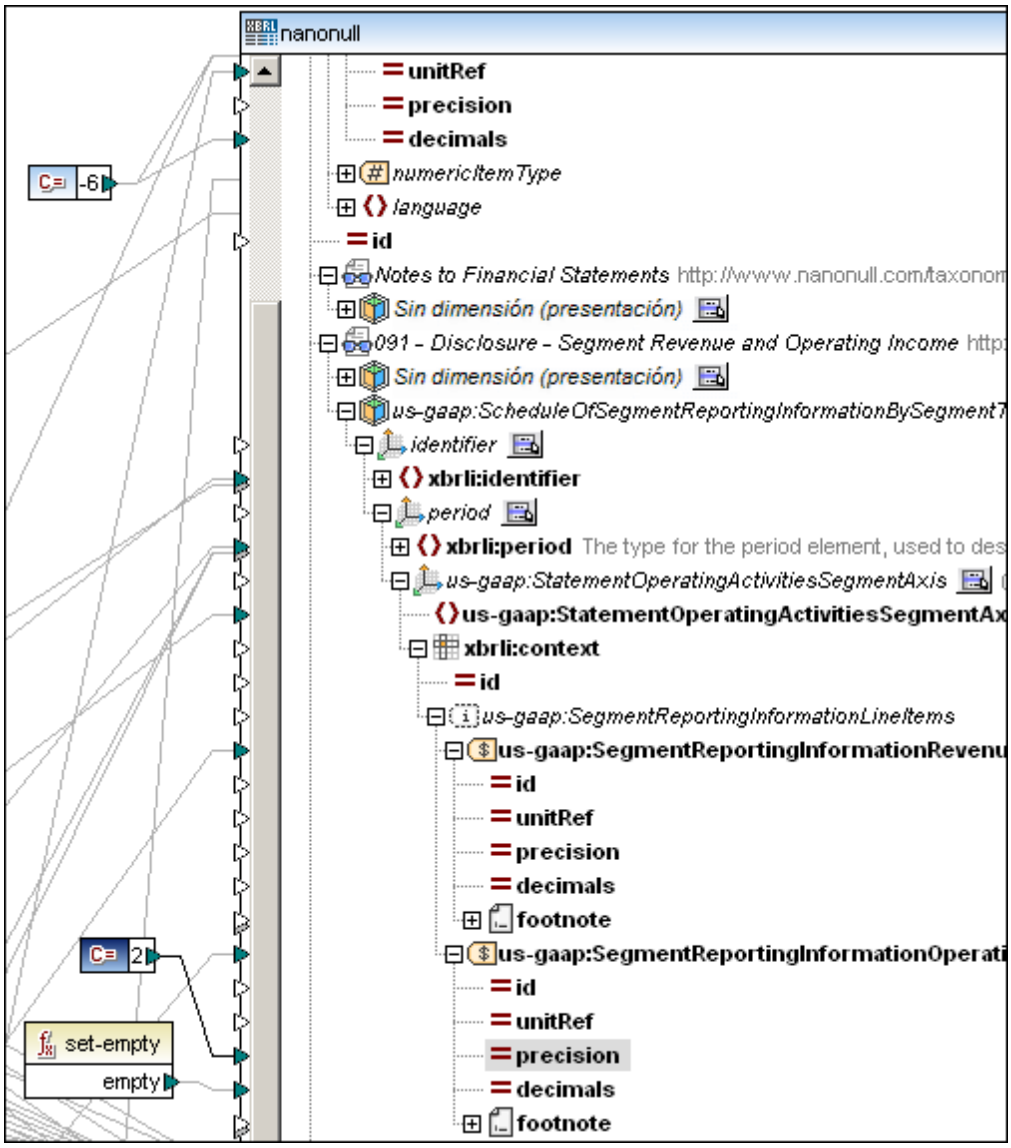
- Puesto que los valores *Predeterminados* se pueden definir en gran variedad de elementos de la estructura XBRL, cada subárbol puede tener valores predeterminados distintos.
- En la imagen anterior también puede comprobar el uso del valor predeterminado para el atributo **`xml:lang`**, que define el idioma de una nota al pie.

Para reemplazar/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 aparece el archivo de muestra DB_To_XBRL.mfd. En esta asignación 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.



Generación automática 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.

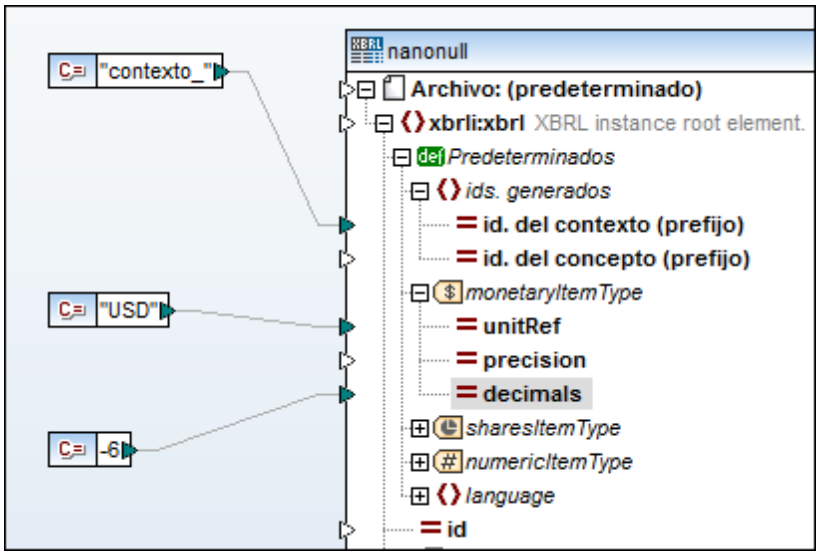
 © 2014 Altova GmbH

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.

Para generar un contexto personal

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`, etc. **Nota:** si no define un valor predeterminado, MapForce usa el prefijo `ctx_` para todos los ids. de contexto.

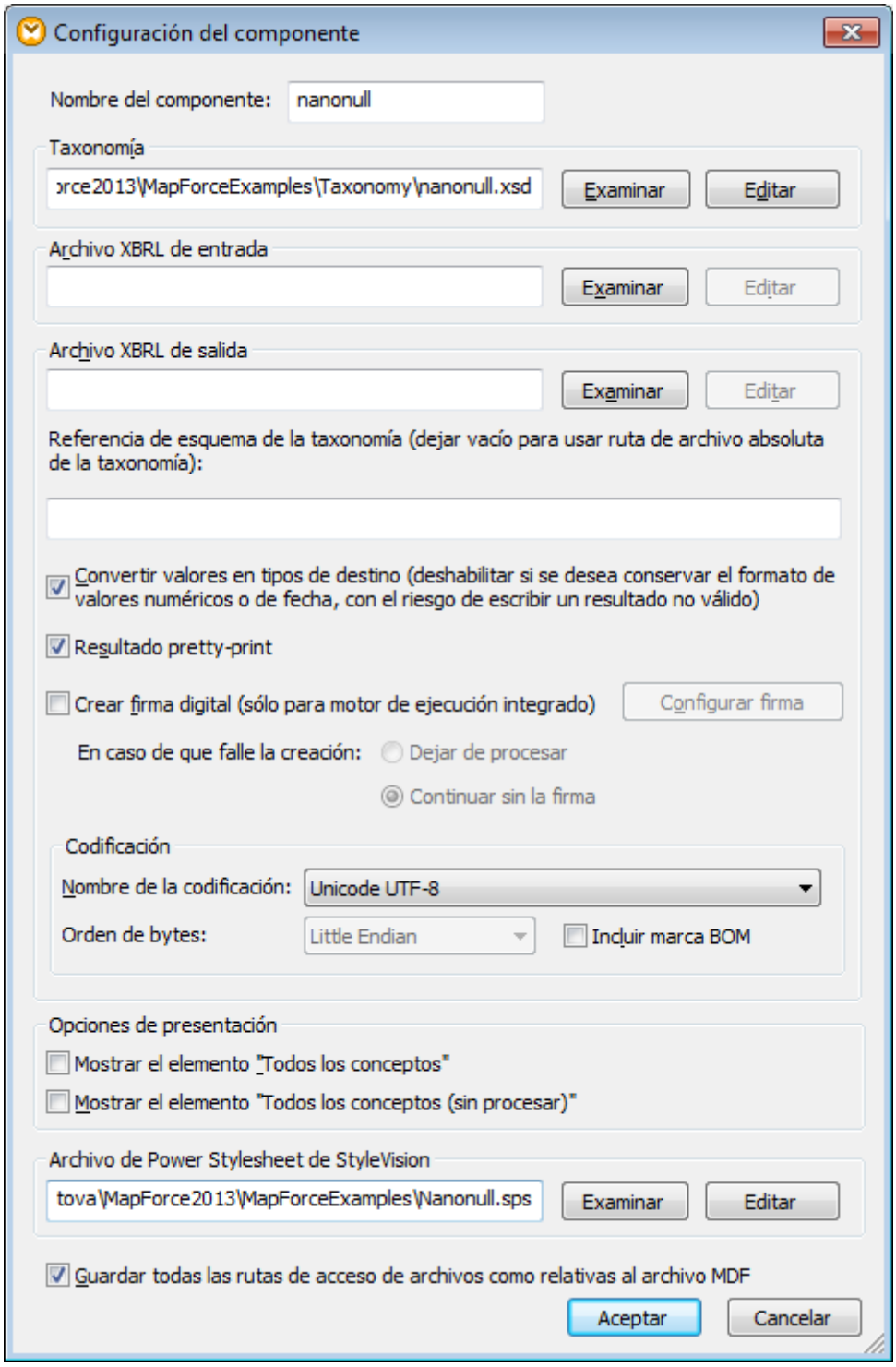


Si el archivo XBRL de salida contiene **notas al pie** (elementos `footnote`), los conceptos relacionados deben tener identificadores de concepto para poder crear enlaces a las notas al pie generadas automáticamente (elementos `footnoteLink`). Estos atributos identificadores se generan automáticamente y puede asignar un texto al nodo **id. de concepto (prefijo)**.

Nota: si no define un valor predeterminado, MapForce usa el prefijo `fact_` para todos los ids. de concepto.

Configuración y visualización del componente

Al hacer doble clic en un componente XBRL aparece el cuadro de diálogo "Configuración del componente", donde puede definir algunas opciones de visualización.



Taxonomía: nombre y ruta de acceso del archivo de taxonomía principal.

Archivo XBRL de entrada: seleccione o cambie el archivo XBRL de instancia para el componente XBRL. Este campo se rellena automáticamente al insertar el componente XBRL y asignarle un archivo XBRL de instancia.

 © 2014 Altova GmbH

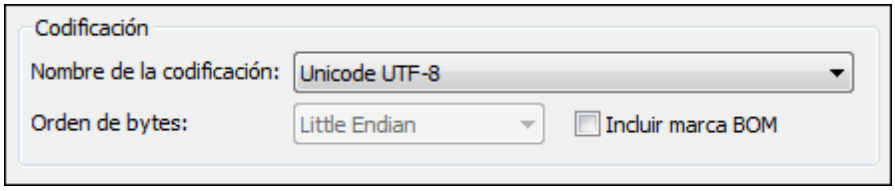
Archivo XBRL de salida: nombre y ruta de acceso del archivo XBRL de destino que se debe usar al generar y ejecutar el código de programa. Si al insertar el componente XBRL no le asigna un XBRL de instancia, este campo se rellena con el archivo de taxonomía y la extensión `xml`. Si asigna un XBRL de instancia al componente al crearlo, el archivo del campo *Archivo XBRL de entrada* se copia en este campo.

Referencia de esquema de la taxonomía: ruta de acceso del archivo de esquema referenciado/asociado y relativo al archivo MFD. Use este campo para especificar una taxonomía diferente para la validación. La referencia de la taxonomía se escribe en el atributo `href` del elemento `link:schemaRef`.
Por ejemplo `<link:schemaRef xlink:type="simple" xlink:href="..\..\nanonull.xsd"/>`.

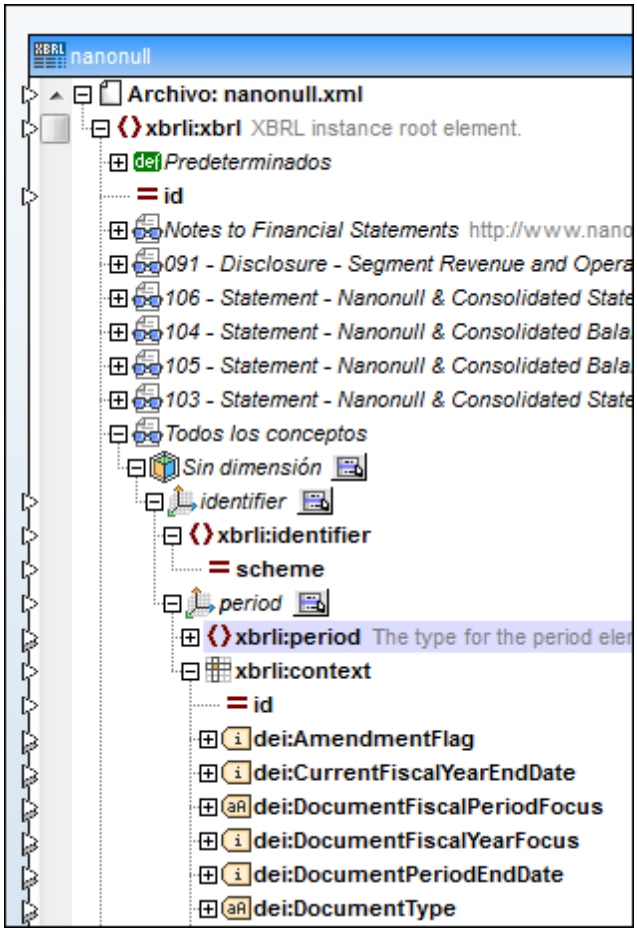
Convertir valores de destino en tipos de destino: defina si los tipos del esquema XML de destino se deben usar durante la asignación (opción predeterminada) o si todos los datos asignados al componente de destino se deben tratar como valores de **cadena**.

Resultado pretty-print: cambia el formato del documento XBRL en el panel **Resultados** para presentar el documento de forma estructurada. A cada nodo secundario se le aplica una sangría de 1 tabulación con respecto a su nodo primario.

Codificación

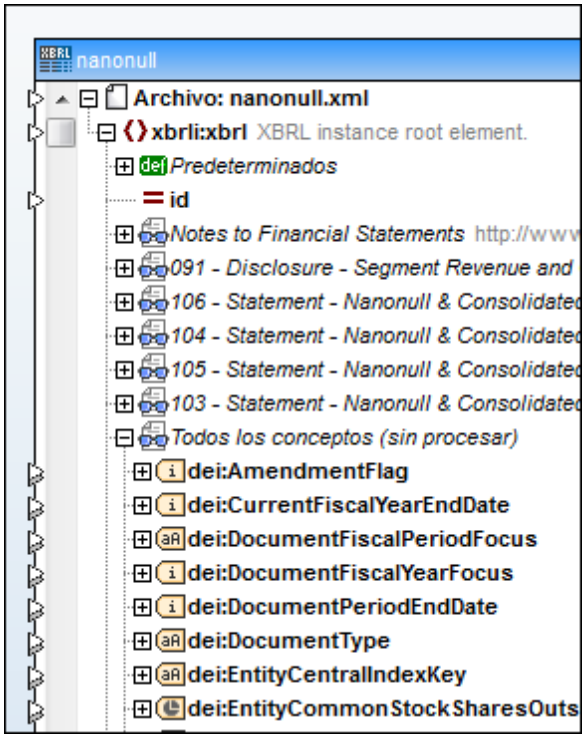


Mostrar el elemento Todos los conceptos: muestra el nodo adicional **`Todos los conceptos`** en la estructura jerárquica. Este nodo contiene el hipercubo **`Sin dimensión`**, que permite asignar todos los conceptos de la taxonomía independientemente de si se contabilizan o no dentro de los hipercubos mediante las dos dimensiones predeterminadas **`identifier`** y **`period`**. El nodo de **contexto** contiene todos los conceptos XBRL de la taxonomía como secundarios y permite generar contextos automáticamente. Los elementos abstractos se ocultan.



Mostrar el elemento Todos los conceptos (sin procesar): muestra el nodo adicional `Todos los conceptos (sin procesar)`, que ofrece acceso a todos los hechos del XBRL de instancia pero no permite la generación automática de contextos ni de dimensiones. Este elemento es una imagen de la estructura XML sin procesar de los conceptos del archivo de instancia. El atributo **`contextRef`** debe asignarse a mano.

 © 2014 Altova GmbH



Hipercubos XBRL

Los hipercubos permiten tratar los contextos automáticamente. Hay tres tipos de hipercubos:

- definidos por la taxonomía (p. ej *Statement [Table]*)
- generados por MapForce para simplificar las dimensiones estándar **`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`** *Dimensionless*

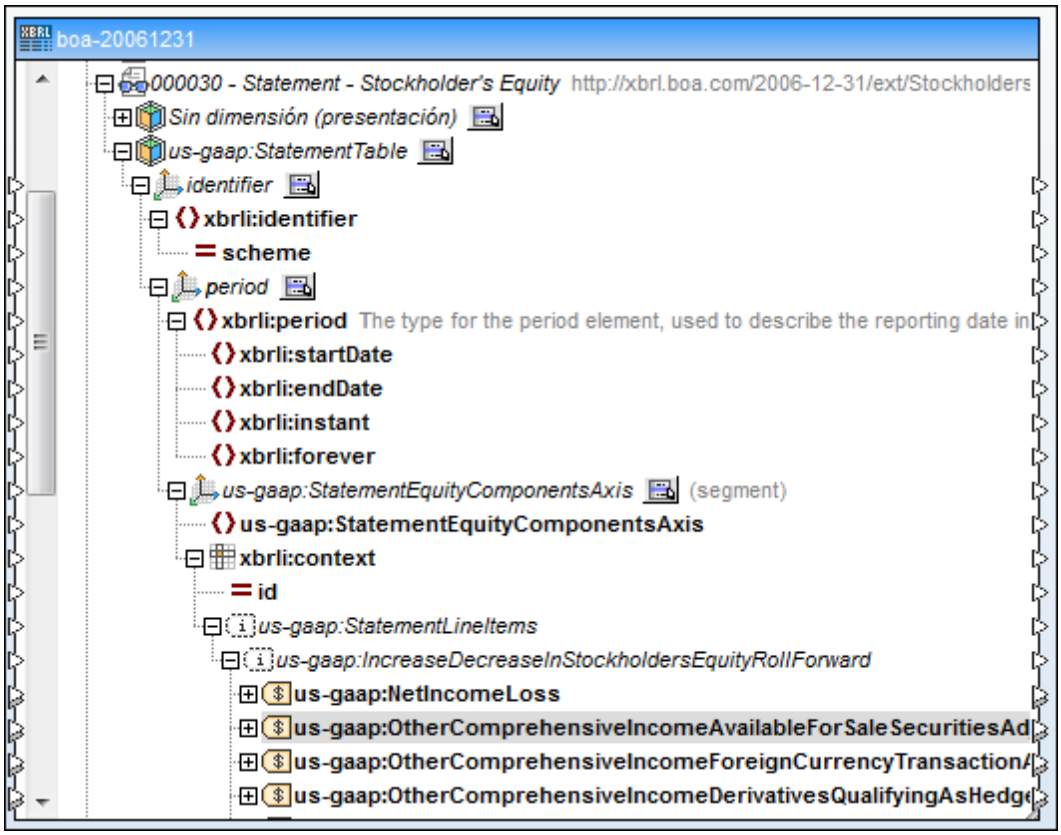
Todos los hipercubos contienen dos dimensiones predeterminadas (**`identifier`** y **`period`**) que permiten leer/escribir con facilidad estos elementos en cada contexto. Además las dimensiones definidas en la taxonomía se relacionan automáticamente a los elementos **`xbrli:segment`** y **`xbrli:scenario`** 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. Para más información consulte el apartado Configuración del componente.

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 la configuración predeterminada de una 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.

Dimensiones

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.

Representación de las dimensiones en un componente

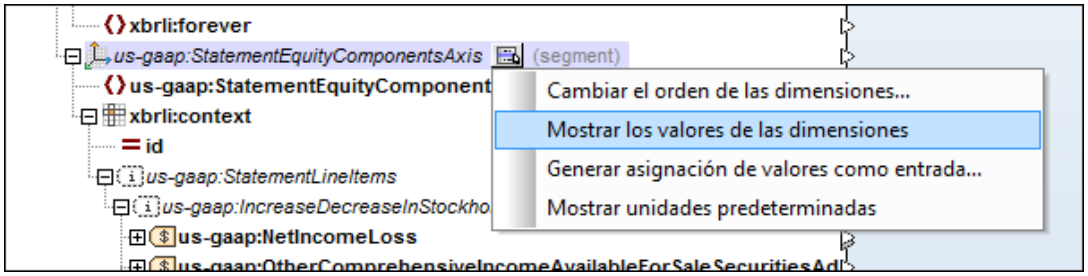
En un principio la dimensión explícita se presenta con un solo nodo secundario y se puede asignar directamente usando el secundario (p. ej. *Statement, Equity Components [Axis]*).

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.

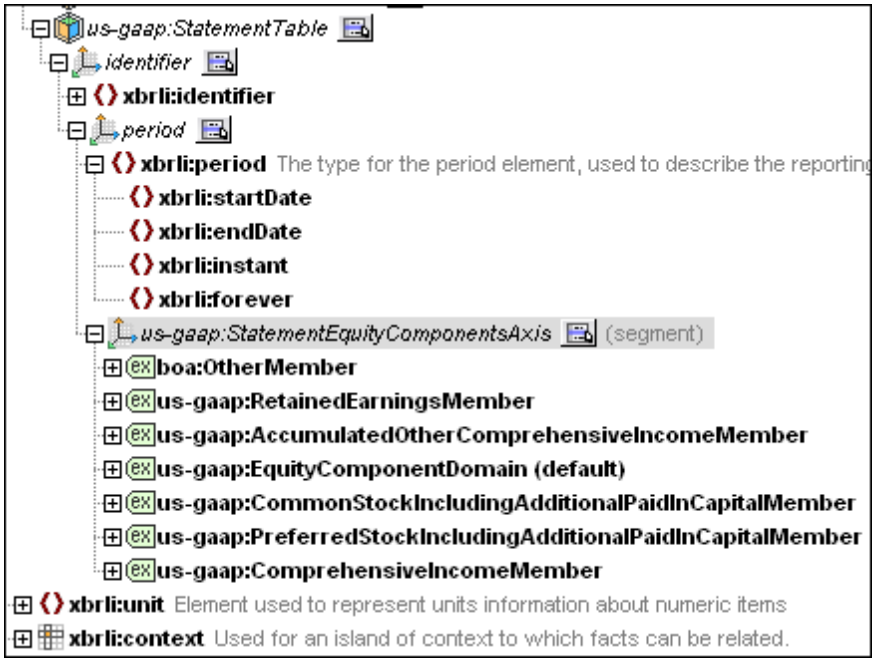
Si desea usar varias asignaciones diferentes para los hechos, MapForce puede representar cada valor del dominio de la dimensión en un nodo diferente.

Para mostrar los valores de la dimensión en el componente:

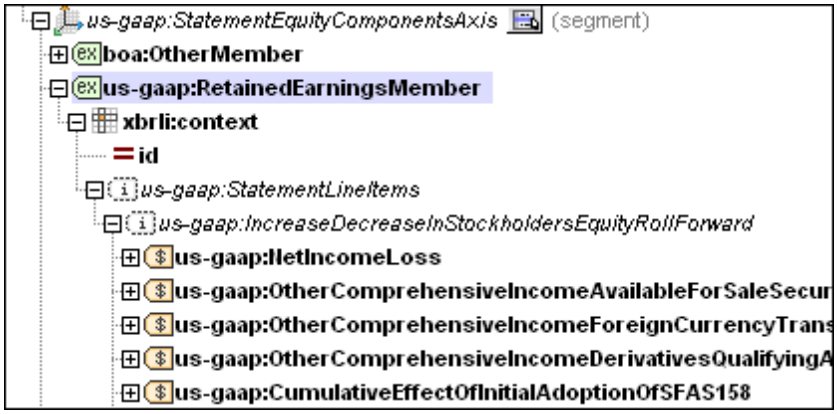
1. Haga clic en el botón interactivo de la dimensión cuyos valores desea ver y seleccione la opción **Mostrar los valores de las dimensiones**.



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.



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]*.

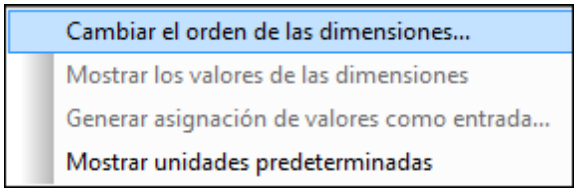
Orden de las dimensiones

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.

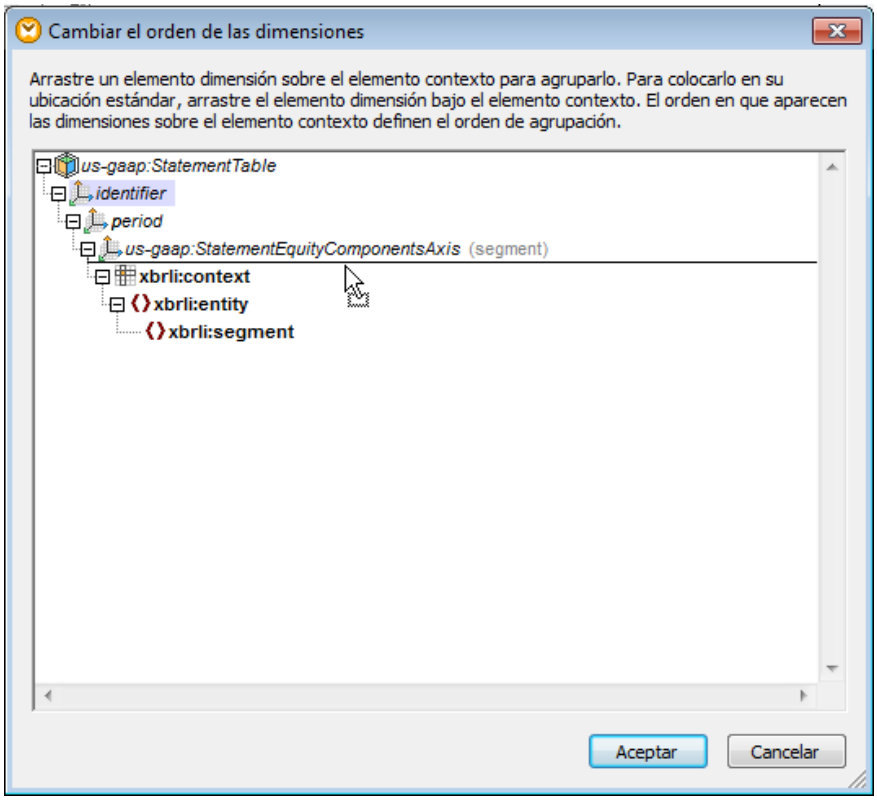
Para cambiar el orden de las dimensiones:

1. Haga clic en el botón interactivo de cualquier dimensión del hipercubo y seleccione la opción **Cambiar el orden de las dimensiones**.

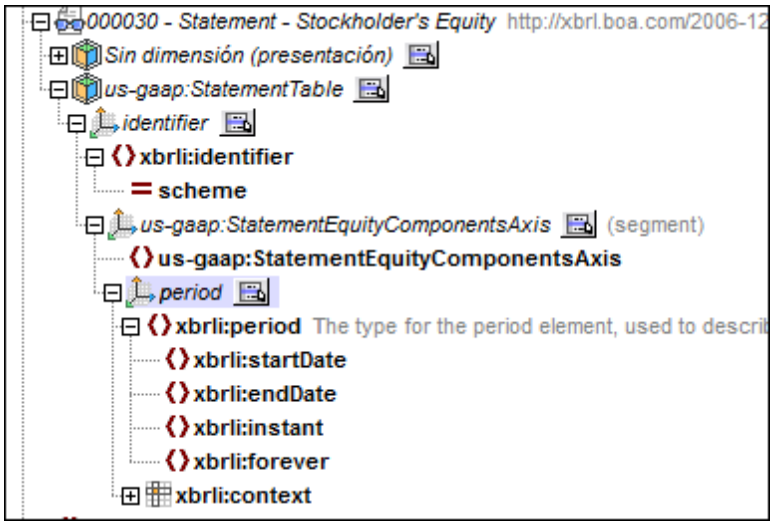


   A continuación se abre un cuadro de diálogo donde puede cambiar la posición de las dimensiones del hipercubo. Recuerde que los hipercubos tienen dos dimensiones predeterminadas (**`identifier`** y **`period`**) cuya posición se puede cambiar también.
2. Haga clic en la dimensión pertinente y arrástrela por el cuadro hasta su nueva posición.

 © 2014 Altova GmbH

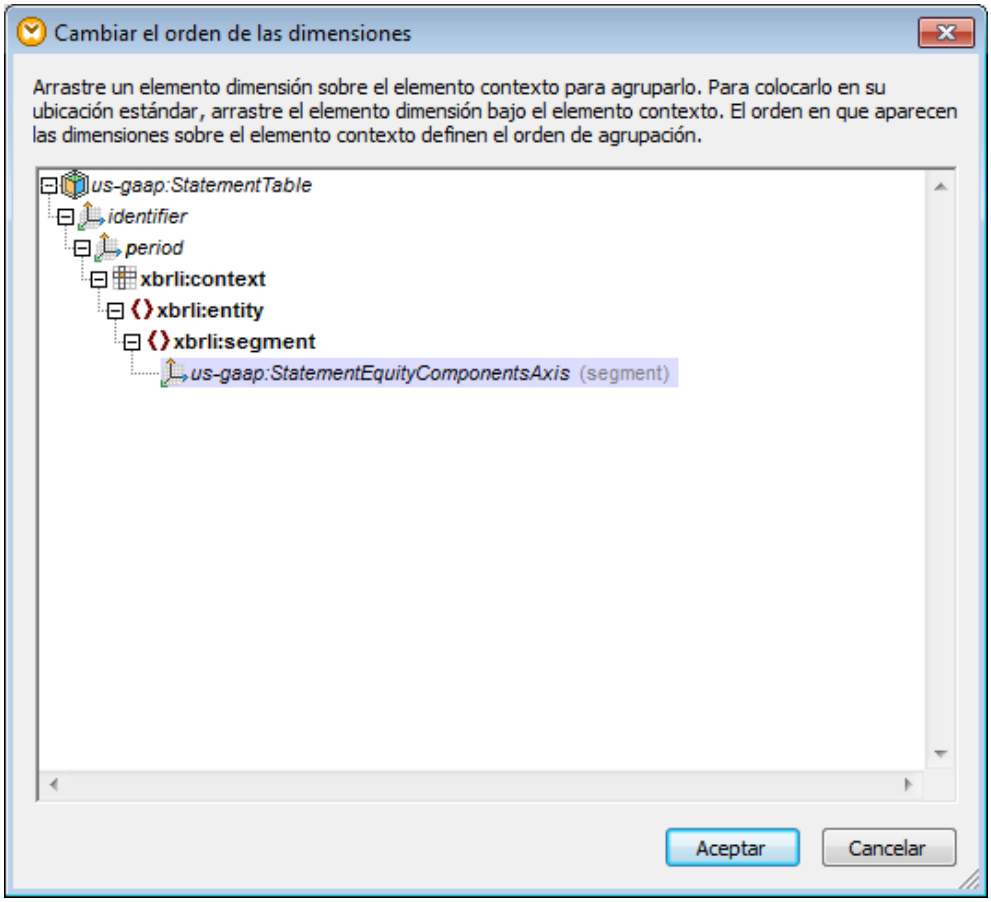


Al pasar el puntero por una posición donde se puede colocar la dimensión aparece una línea horizontal.



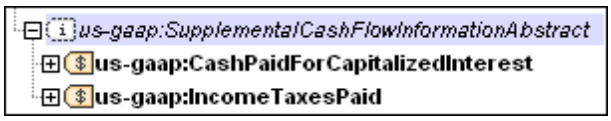
3. Haga clic en **Aceptar** para cerrar el cuadro de diálogo. La dimensión se coloca en su nueva posición dentro del componente.

Para **excluir** una dimensión de la jerarquía, arrastre la dimensión y colóquela debajo del elemento `xbrli:context`.



Cómo se deriva la estructura del componente de la taxonomía

La estructura jerárquica del componente XBRL en MapForce se crea a partir de los arcos de presentación **`xlink:from ... xlink:to`** de la base de enlaces de presentación.



Se trata de vínculos entre los conceptos y los hechos:

<link:**presentationArc** xlink:type="arc" xlink:arcrole="http://www.xbrl.org/2003/arcrole/parent-child"

xlink:from="**SupplementalCashFlowInformationAbstract"**

xlink:to="CashPaidForCapitalizedInterest"

xlink:from="**SupplementalCashFlowInformationAbstract**"

xlink:to="IncomeTaxesPaid"

 © 2014 Altova GmbH

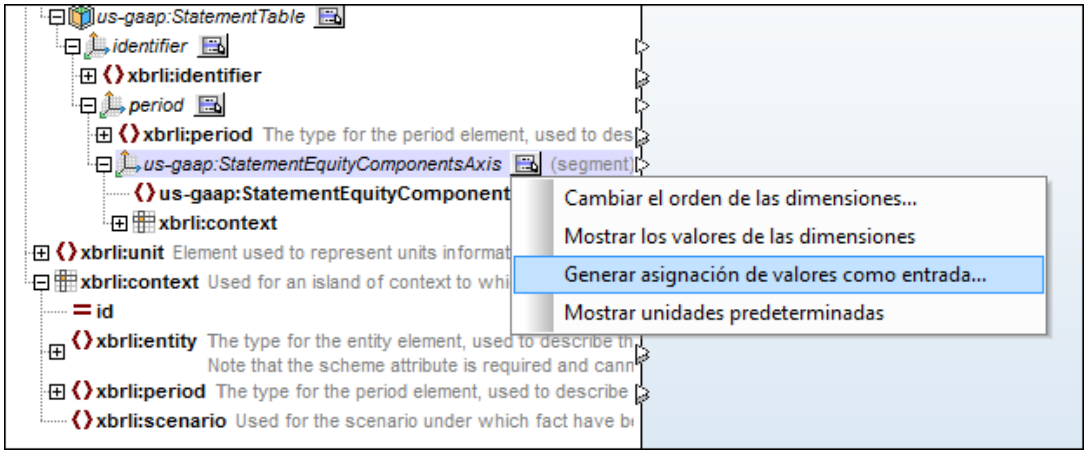
Función de asignación de valores XBRL

Esta opción transforma datos de entrada de cualquier tipo en un QName válido en el componente de destino. Es decir, la cadena de entrada se convierte en el nombre con prefijo (QName).

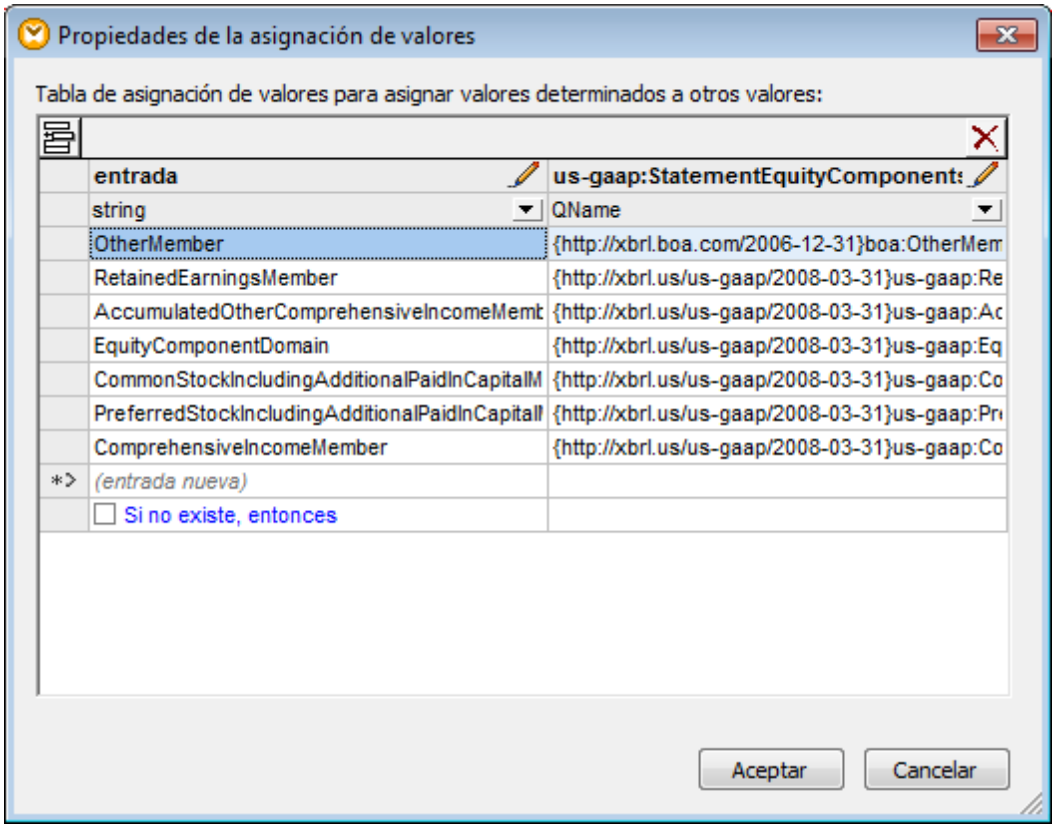
Para generar una tabla de búsqueda (asignación de valores) para la dimensión seleccionada:

Compruebe que los valores de la dimensión no están visibles en la dimensión elegida.

1. Haga clic en el botón interactivo de la dimensión y seleccione **Generar asignación de valores como entrada** en el menú contextual.

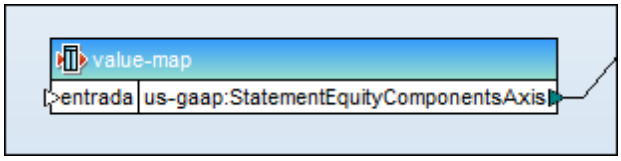


Esto abre el cuadro de diálogo "Propiedades de la asignación de valores", que contiene valores de entrada y salida generados automáticamente a partir de los miembros de dominio y del dominio predeterminado de la dimensión, tal y como se definieron en la taxonomía.



2. 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.



3. Conecte el elemento de entrada que contiene los datos de entrada que desea transformar al parámetro **`entrada`** de la función. La cadena de entrada `USA` se transforma en `nanonull:USA` para el miembro explícito de esa dimensión.

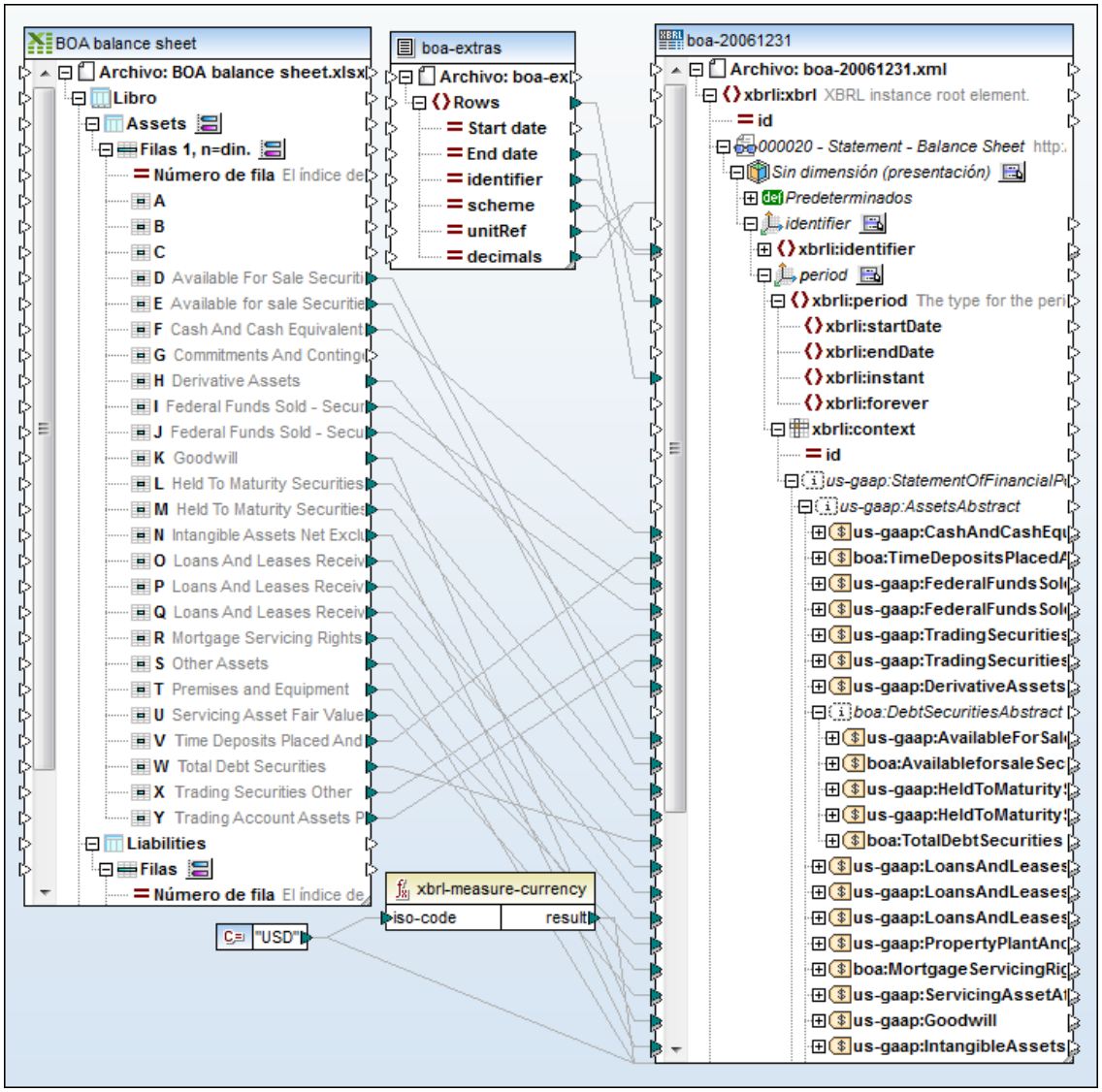
   Para ver un ejemplo de uso general de la función de asignación de valores, consulte el apartado Asignación de valores: transformar datos de entrada.

 © 2014 Altova GmbH

10.7.3 Ejemplo de asignación de datos Excel a XBRL

La imagen siguiente muestra el archivo de ejemplo `boa-balance-sheet.mfd` , disponible en la carpeta `...\MapForceExamples\Tutorial`. Los archivos de taxonomía utilizados en este archivo proceden del sitio web www.xbrl.us (haga clic en el enlace *Download all files as ZIP*) y también están disponibles en la carpeta `..\Tutorial`.

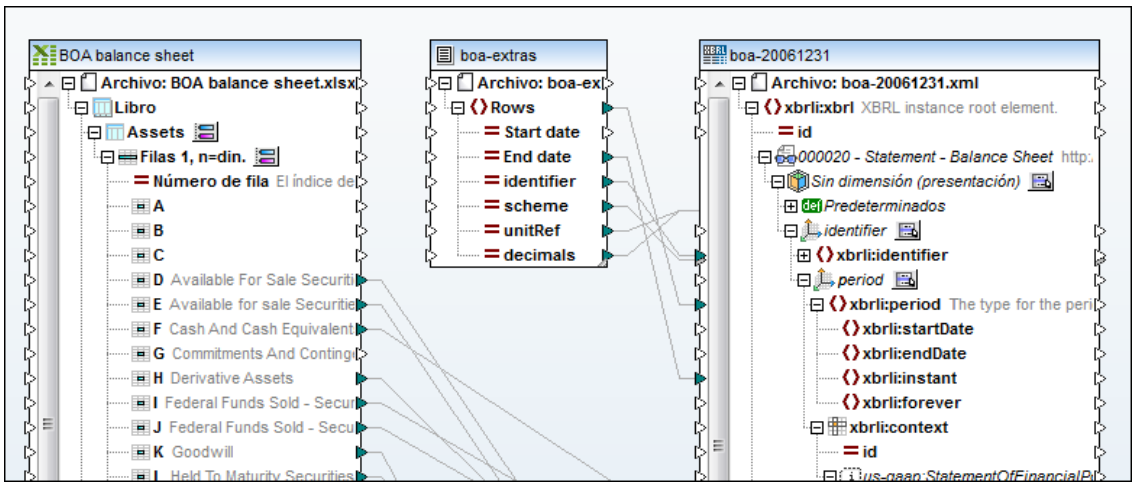
En este archivo de ejemplo puede ver cómo se asignan datos de una hoja Excel a una taxonomía XBRL para generar un archivo XBRL de instancia. Para no complicar demasiado las cosas, en el ejemplo solamente se crearon asignaciones con la hoja de cálculo **`Assets`**. El resultado de esta asignación es un documento XBRL válido que contiene los datos de la hoja **`Assets`** relacionados con un instante concreto, el día 31 de diciembre de 2006.



Cómo se hizo

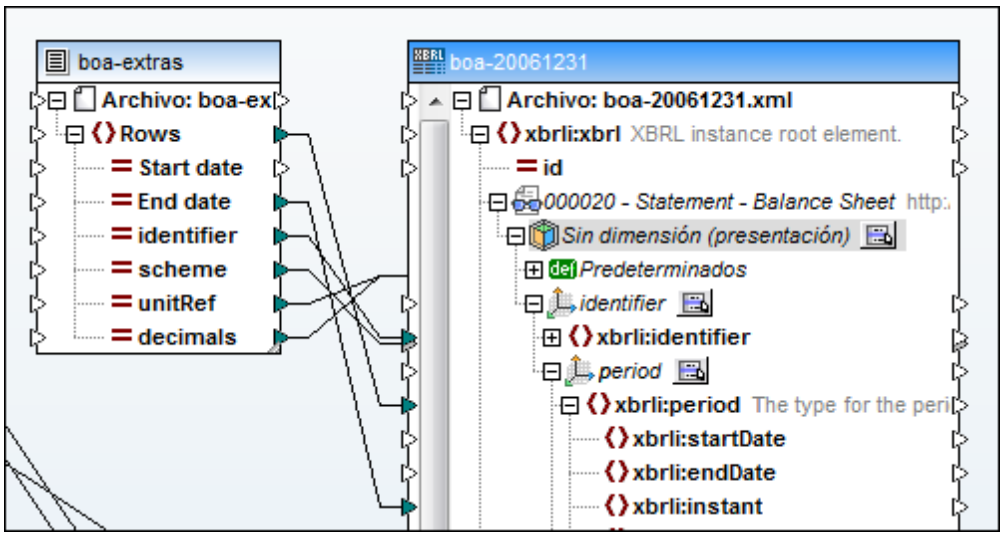
En este ejemplo las columnas Excel, que están en orden alfabético, se asignan directamente a los elementos/hechos XBRL de la taxonomía (p. ej. **`Cash and Cash Equivalents`**) .

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 utilizado en el documento XBRL de instancia de salida. Sin embargo, si el nombre de los elementos es el mismo, la ventaja es que la función de conexión automática de secundarios conecta los elementos secundarios que se llaman igual.



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 secundario en una dimensión de un hipercubo y seleccione la opción **XBRL | Mostrar unidades predeterminadas** o haga clic en el botón interactivo y seleccione la misma opción en el menú emergente.



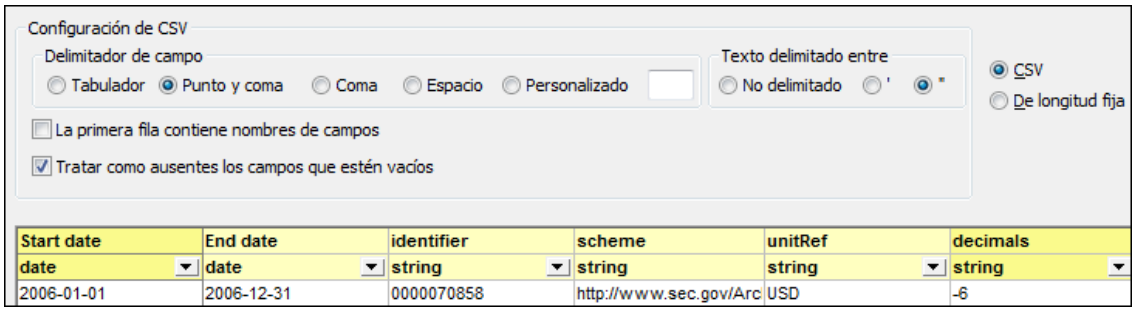
Elementos XBRL obligatorios necesarios en un archivo XBRL de instancia:

- 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

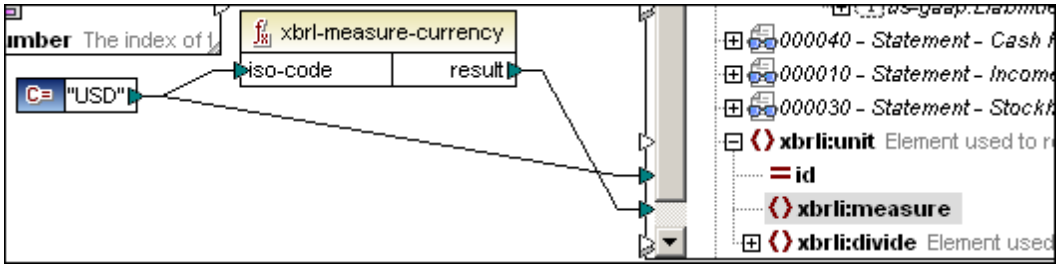
 © 2014 Altova GmbH

- 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.



Abra 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 del panel **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.

```
<?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:
  <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:ServicingAssetAtFa
  <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>
```


 © 2014 Altova GmbH

10.7.4 Ejemplo de asignación de datos de BD 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 (**`091 - Disclosure - Segment Revenue and Operating Income`**) y tres extractos (p. ej. **`106 - Statement - Nanonull and Consolidated Statement of Cash Flows`**).

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.

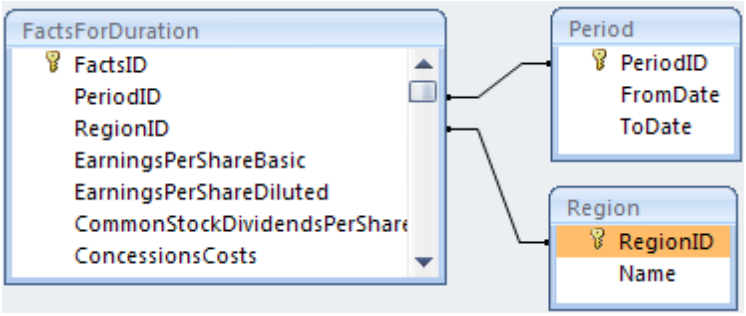
El informe contable **106 - Statement - Nanonull and Consolidated Statement of Cash Flows** muestra cómo escribir hechos en la instancia XBRL de salida relacionados tanto con períodos de tipo intervalo como de tipo instantáneo.

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.

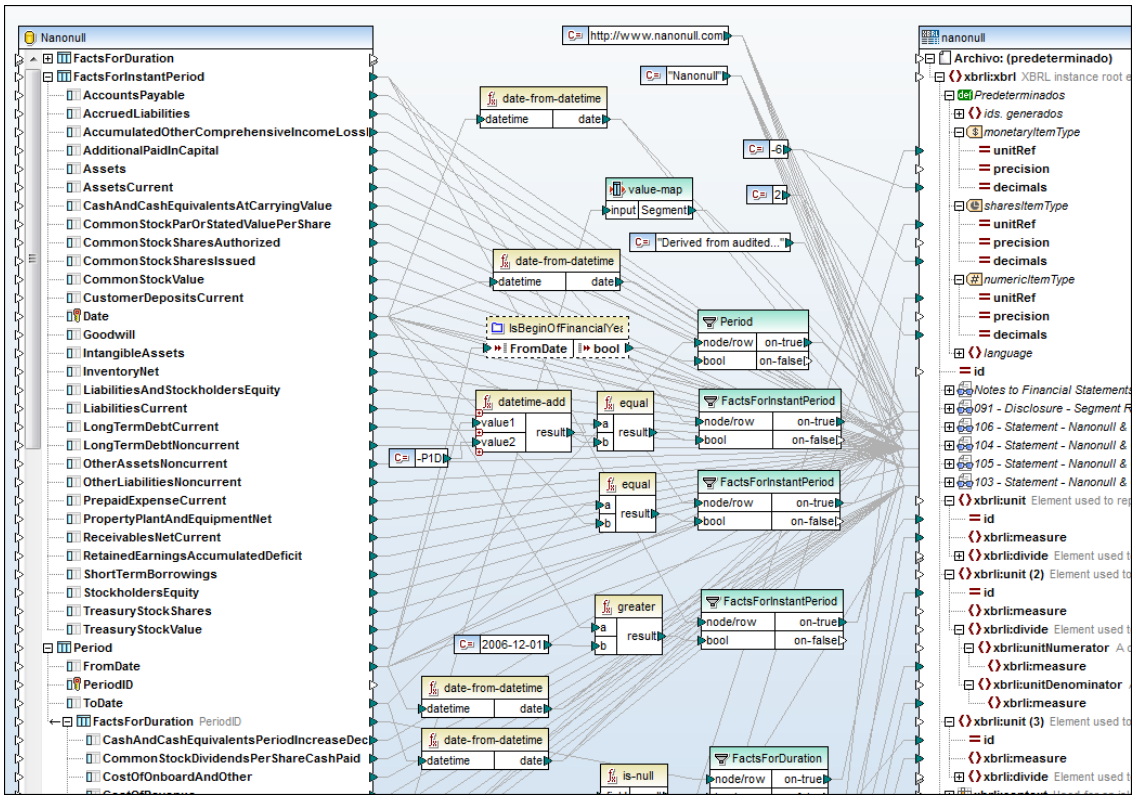
La asignación define dos unidades en la instancia de salida: **`USD`** y **`perShareItemType`**. Para ello hace falta duplicar el elemento XBRL **`xbrli:unit`**. Los elementos de medida relacionados se crean usando las funciones **`xbrl-measure-currency`** y **`xbrl-measure-shares`** de la biblioteca 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

Para más información consulte este ejemplo con elementos obligatorios.

10.8 Asignación de datos entre HL7 v3 y esquemas XML

MapForce 2014 es automáticamente compatible con HL7 versión **3.x** porque este estándar está basado en XML.

En la página Bibliotecas 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:

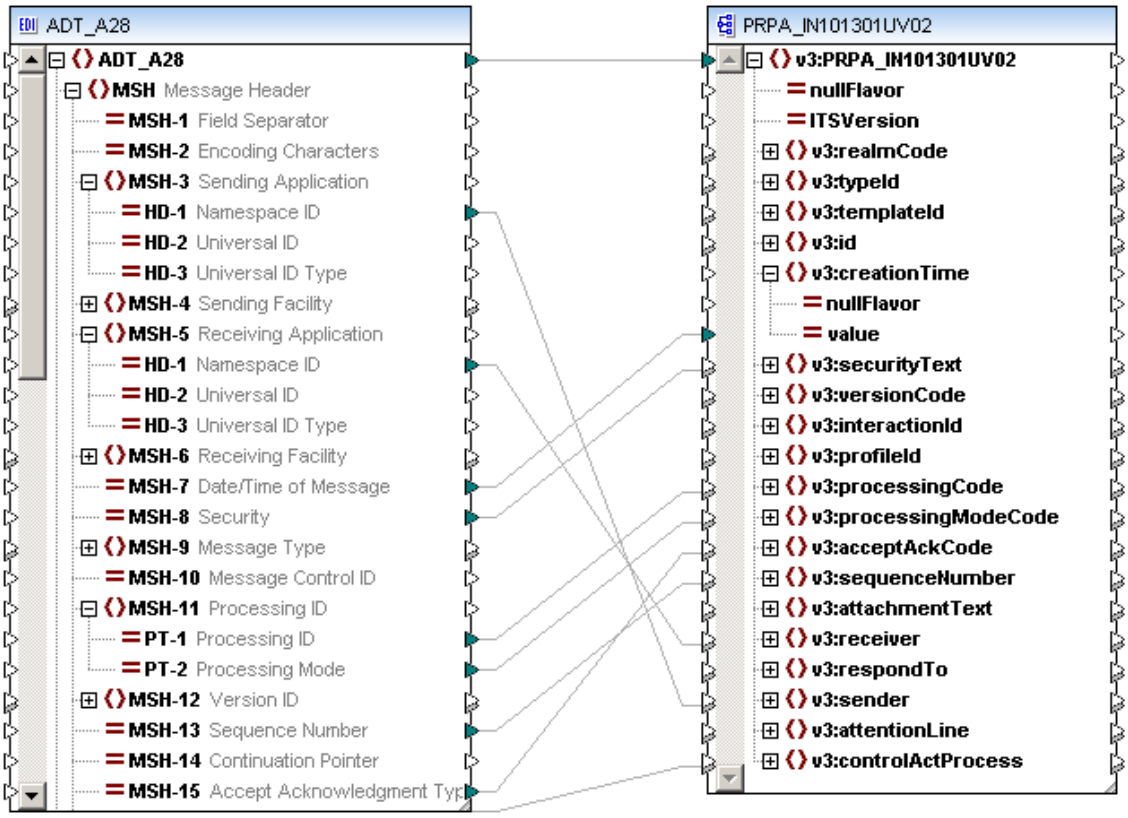
Equipos Windows XP	`C:\Archivos de programas\Altova\Common2014\Schemas\ hl7v3`
Equipos Windows	`C:\Archivos de programas\Altova\Common2014\Schemas\ hl7v3`
Equipos Windows7	`C:\Archivos de programas\Altova\Common2014\Schemas\ hl7v3`

Si utiliza una versión de 32 bits de MapForce en un sistema operativo de 64 bits, la ubicación es esta: `C:\Archivos de programas(x86)\Altova\Common2011\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`.



El archivo XML resultante aparece en la siguiente imagen.

 © 2014 Altova GmbH

```
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>
```

Altova MapForce 2014

Consejos prácticos

11 Consejos prácticos

En esta sección se describen tareas corrientes de la creación de asignaciones de datos.

General:

Cómo puedo...	Consulte este apartado
Implementar una asignación de MapForce en FlowForce Server	Implementar una asignación de MapForce
Compilar una asignación de MapForce en un archivo de ejecución de MapForce Server	Compilar una asignación de MapForce
Filtrar según ciertos criterios	Filtrar datos: recuperar datos dinámicos
Ordenar los datos de entrada	Componentes de ordenación: ordenar secuencias de entrada
Usar archivos de entrada y salida **dinámicos/múltiples** en la asignación	Varios archivos de entrada/salida dinámicos por componente
Asignar/usar un **tipo complejo derivado** (xsi:type)	Asignar tipos derivados de XML Schema
Crear una asignación definida por el usuario **recursiva**	Asignación definida por el usuario recursiva
Obtener una **vista previa** de la asignación (antes de generar código)	Motor de ejecución integrado BUILTIN (vista previa de la asignación)
Usar funciones de agregado **`min`**, **`max`**, **`sum`**, **`avg`** y **`count`**	Funciones de agregado
Usar un **componente de entrada** como **parámetro** durante la ejecución desde la línea de comandos	Línea de comandos: definir parámetros de entrada
Especificar un valor **alternativo** para un componente de entrada	Línea de comandos: invalidar parámetros de entrada
Definir y ejecutar una asignación con archivos de entrada/salida diferentes a los definidos cuando diseñé la asignación	Nombres de los componentes
Transformar un valor de entrada en uno de salida usando una **tabla de búsqueda**	Asignación de valores: transformar datos de entrada
Crear mis propios archivos de **catálogo**	Archivos de catálogo en MapForce
Unir varios archivos de origen en un solo archivo de destino	Unir varios archivos en un solo destino

Nodos

Cómo puedo...	Consulte este apartado

 © 2014 Altova GmbH

Probar nodos: nodos existentes / nodos no existentes	Comprobación de nodos: existe/no existe
Agrupar nodos según su contenido	Agrupamiento de nodos
Asignar datos en función de la **posición** de un nodo en una secuencia	Posición de los elementos de contexto en una secuencia
Comentar una asignación / determinados conectores o nodos	Configuración de la conexión

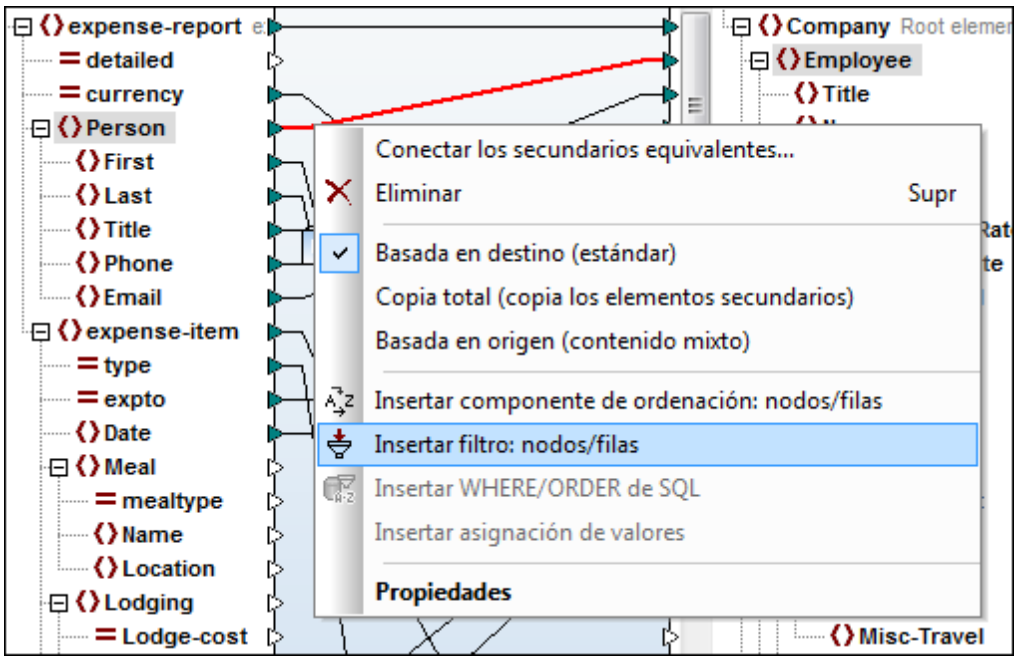
Cómo puedo...	**Consulte este apartado**
Crear un archivo XML de destino válido con un elemento raíz	Elemento raíz de los documentos de destino
Saber qué pasa con los **elementos secundarios** cuando aplico un **filtro** al elemento primario	Componentes de filtrado: consejos
Reemplazar los caracteres **hexadecimales especiales** de una BD con otros en el archivo de destino	Reemplazar caracteres especiales en datos de BD
Definir el nodo que debe actuar como **nodo de contexto** en un archivo de origen	Nodo de contexto prioritario (priority context)
Asignar varias **tablas jerárquicas** a un solo archivo XML de destino	Asignar varias tablas a un archivo XML
Ejecutar MapForce desde la **línea de comandos** (edición Enterprise/ Professional)	Parámetros de la línea de comandos
Definir **excepciones** en las asignaciones	Excepciones

11.1 Filtrar datos: recuperar datos dinámicos

Si quiere recuperar o filtrar ciertos datos y no otros, utilice un componente de **filtrado**.

Para insertar un componente de filtrado:

1. Haga clic con el botón secundario en el conector existente entre los nodos/elementos que desea filtrar.



2. En el menú contextual seleccione el comando **Insertar filtro: nodos/filas**.
   Esto inserta un componente de filtrado y lo conecta automáticamente a los componentes de origen y destino.
3. Defina la condición que desea filtrar (p. ej. PrimaryKey=4) y conecte el resultado con el parámetro **`bool`** del filtro.

Otra opción es:

- Haga clic en el icono de la barra de herramientas para insertar el filtro.
  El filtro se inserta sin conectar con los componentes de origen/destino y el título del componente es **filter**.

  Nada más conectar el filtro con el componente de origen, el filtro toma el nombre del elemento conectado al parámetro **`node/row`** del filtro (p.ej. **`Person`**).

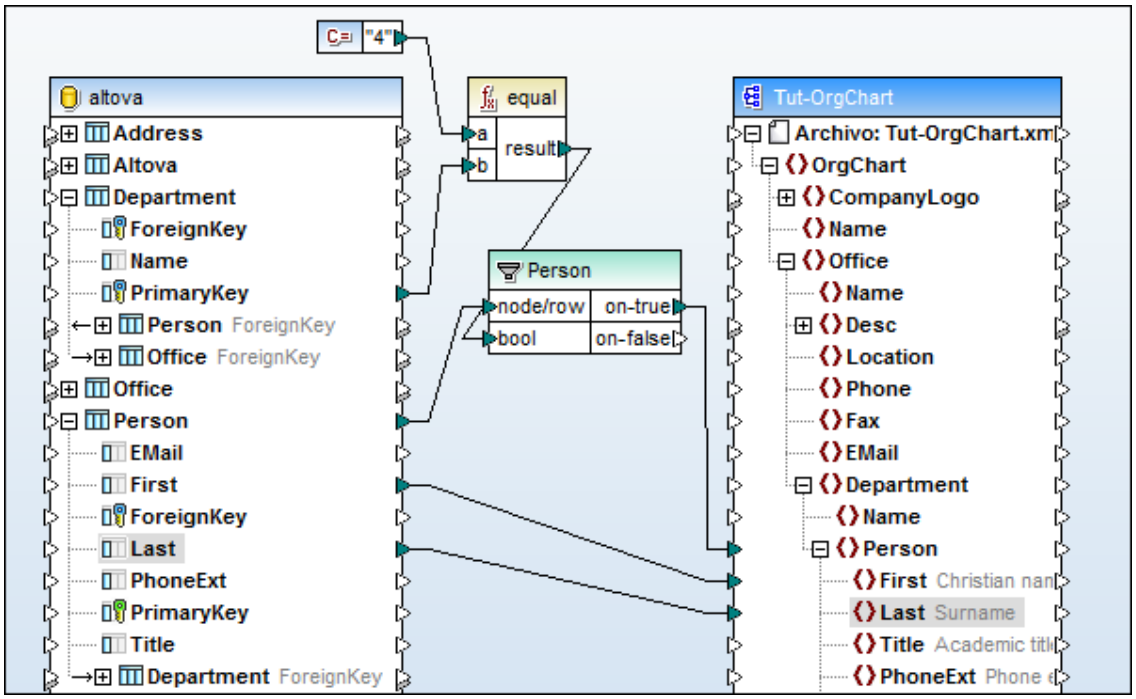
Objetivo:
Asignar todos los elementos **`First`**/**`Last`** de la tabla **`Person`**, donde la clave primaria de **`Department`** sea igual a 4.

- Por medio de la función **`equal`**, comparamos el valor de **`PrimaryKey`** con el valor 4,

 © 2014 Altova GmbH

suministrado por medio de una constante.

- **PrimaryKey** se asigna desde la tabla raíz **Department**.
- Los elementos **First** y **Last** se asignan desde la tabla **Person**, que es una **tabla secundaria** de la tabla raíz **Departamento**.
- Las asignaciones definidas bajo la **misma** tabla raíz, en este caso **Departamento**, conservan las relaciones entre sus tablas.



Un filtro tiene dos parámetros de entrada (**node/row** y **bool**) y dos de salida (**on-true** y **on-false**). Si la condición **bool** es **true**, entonces el contenido de los elementos/nodos secundarios del componente de origen conectados al parámetro **node/row** se reenvían al componente de destino. Si la condición **bool** es **false**, entonces los valores complemento se pueden usar creando una conexión con el parámetro **on-false**.

El resultado de la asignación muestra los nombres (**First**) y apellidos (**Last**) de las personas del departamento de TI (cuya clave primaria es 4).

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <OrgChart xsi:noNamespaceSchemaLocation="C:/DOCUME~1
    ="http://www.w3.org/2001/XMLSchema-instance">
3     <Office>
4       <Department>
5         <Person>
6           <First>Alex</First>
7           <Last>Martin</Last>
8         </Person>
9         <Person>
10          <First>George</First>
11          <Last>Hammer</Last>
12        </Person>
```

Si quiere ver más ejemplos de componentes de filtrado, consulte estos apartados:

- Filtrar datos XML: Filtrado de datos

- Filtrar archivos CSV según el encabezado y los detalles de los registros: Crear jerarquías a partir de archivos de texto CSV y FLF
- Relaciones de tablas y filtros: Relaciones de BD: cómo conservarlas/descartarlas
- Definir filtros Where de SQL: Componente / condición WHERE de SQL
- Componentes de filtrado: consejos

© 2014 Altova GmbH

11.2 Componentes de filtrado: consejos

Los componentes de filtrado son muy importantes para consultar datos de BD porque permiten trabajar con grandes cantidades de datos con eficacia. Si trabaja con tablas de BD que tienen miles de filas, puede usar filtros para limitar el acceso a las tablas y organizar con eficacia la manera en que se extraen los datos. El uso de filtros influye en la velocidad con que se genera la asignación.

En esta sección describimos varios métodos para optimizar el acceso a datos y acelerar el proceso de asignación en general.

Por lo general, recomendamos usar el menor número posible de filtros, pero además:

1. Evite concatenar componentes de filtrado.
2. Si puede, conecte los parámetros **`on-true/on-false`** del filtro con elementos primarios y no con elementos secundarios.
3. Conecte el parámetro **`on-false`** del filtro para asignar el conjunto de nodos complemento suministrado por el parámetro **`on-true`**.
4. Evite usar filtros para asignar datos a elementos secundarios si el elemento primario está conectado.
5. Use la característica de contexto prioritario para clasificar por orden de prioridad la ejecución de elementos no relacionados entre sí.

Evite concatenar componentes de filtrado
Cada componente de filtrado conduce a un bucle que recorre los datos de origen y, por tanto, accede al componente de origen **n** veces. Si concatenamos dos filtros, entonces el bucle recorre el componente de origen **n*n** veces.

Solución:
Utilice la función lógica **`logical-and`** para combinar las expresiones booleanas de los dos componentes de filtro. El resultado es un solo componente de filtro que recorre los datos **n** veces solamente.

Si puede, conecte los parámetros `on-true/on-false` del filtro con elementos primarios y no con elementos secundarios
Los componentes de filtrado funcionan mejor cuando están conectados a elementos primarios que tienen secundarios, en lugar de a los secundarios directamente.

La expresión **booleana** del filtro, por tanto, se evalúa con el primario **antes de** recorrer los elementos secundarios. Los filtros conectados a una tabla de BD de origen generan:

- "SELECT * FROM table WHERE <expresión>" si está conectado con el **elemento primario** o
- "SELECT * FROM table" y después se evalúa en cada fila si está conectado con **elementos secundarios**

Nota: cuando conecte un filtro a un elemento primario del componente de origen, también es necesario conectar el parámetro **`on-true/on-false`** con el elemento primario del componentes de destino. Si esto no es posible, entonces no aplique esta norma.

Conecte el parámetro `on-false` del filtro para asignar el conjunto de nodos

complemento suministrado por el parámetro `on-true`
Si conecta este parámetro podrá acceder rápidamente al conjunto de nodos complemento definido por la asignación.

Evite usar filtros para asignar datos a elementos secundarios si el elemento primario está conectado
Si usa un filtro para asignar datos desde un primario de origen a un primario de destino, se aplica automáticamente el **mismo filtro** a **cada secundario** del primario.

Si el primario se puede asignar, no es necesario usar componentes de filtrado para suministrar datos filtrados a elementos secundarios. Puede conectar los secundarios directamente.

Use la característica de contexto prioritario para clasificar por orden de prioridad la ejecución de elementos no relacionados entre sí
Las asignaciones siempre se ejecutan de arriba a abajo. Si recorre/busca en dos tablas, primero se procesa una tabla y luego otra. Si realiza asignaciones entre elementos no relacionados entre sí, sin establecer un contexto prioritario, MapForce no sabe cuál de los dos bucles debe ejecutar primero y, por tanto, seleccione automáticamente la primera tabla (u origen de datos).

Solución:
Decida cuál de las tablas/de los datos de origen se debe recorrer/buscar primero y después establezca el contexto prioritario en el conector correspondiente. Para ver un ejemplo más concreto, consulte el apartado Nodo de contexto prioritario (priority context).

Para definir un contexto prioritario:

- Haga clic con el botón secundario en el icono de entrada y seleccione **Contexto prioritario** en el menú contextual.
  Si este comando no está disponible en el menú contextual, conecte los demás iconos de entrada del componente.

Los filtros en las asignaciones basadas en el origen (de contenido mixto)
Las asignaciones basadas en el origen solamente funcionan si tienen conexiones directas entre componentes de origen y destino. Las conexiones existentes bajo una conexión basada en el origen no se consideran basadas en el origen y los elementos se manipulan en el orden de los elementos del componente de destino.

Si usa un solo filtro donde ambas salidas están conectadas a destinos iguales/diferentes, el filtro actúa como si hubiera dos componentes de filtrado distintos, uno de ellos con una condición negada.

Si un componente de excepción se conecta a una de las salidas del filtro, la condición de excepción se comprueba cuando se ejecutan las asignaciones a las otras salidas del filtro.

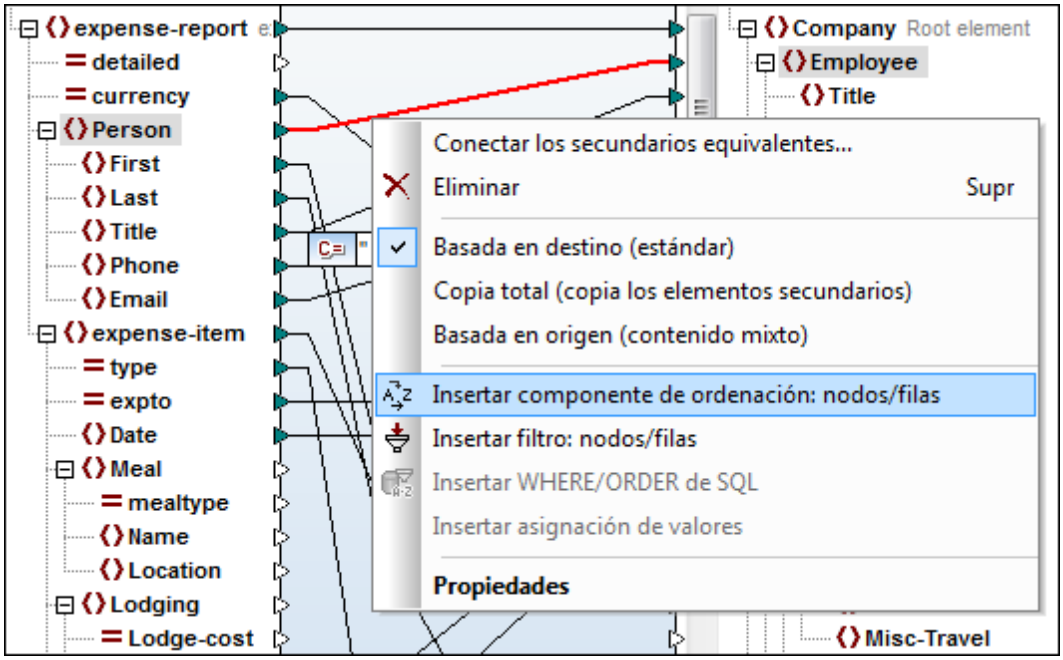
 © 2014 Altova GmbH

11.3 Componentes de ordenación: ordenar secuencias de entrada

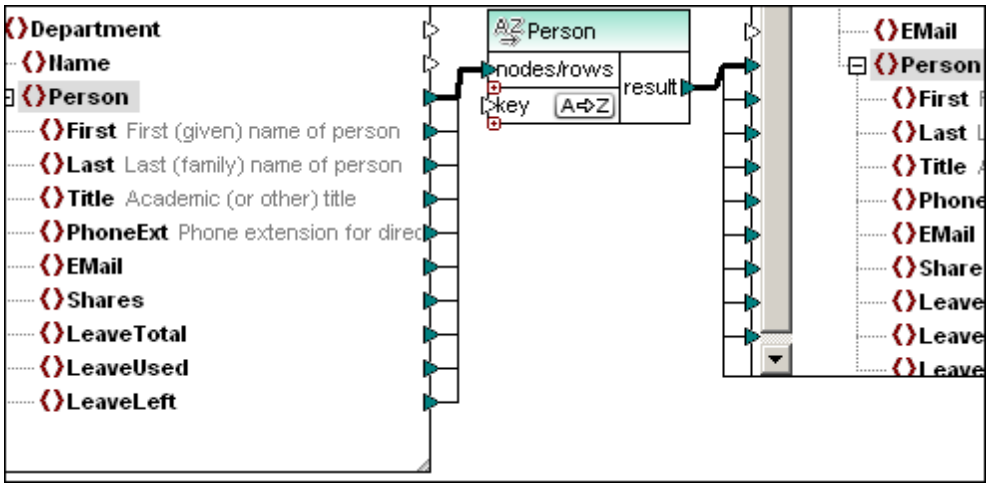
Para ordenar datos de entrada según un criterio de ordenación, use un componente de **ordenación**. Este componente se puede usar con XSLT2, XQuery y con el motor de ejecución integrado.

Para insertar un componente de ordenación:

1. Haga clic con el botón secundario en el conector existente entre los nodos que desea ordenar.



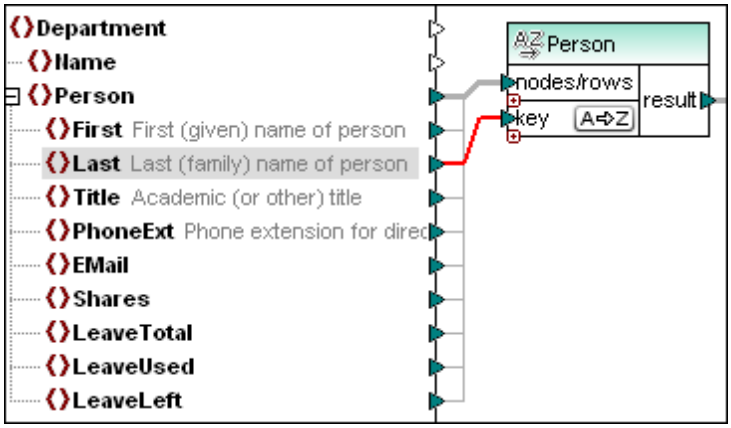
2. En el menú contextual seleccione **Insertar componente de ordenación: nodos/filas**.



Esto inserta un componente de ordenación, que se conecta automáticamente a los componentes de origen y destino.

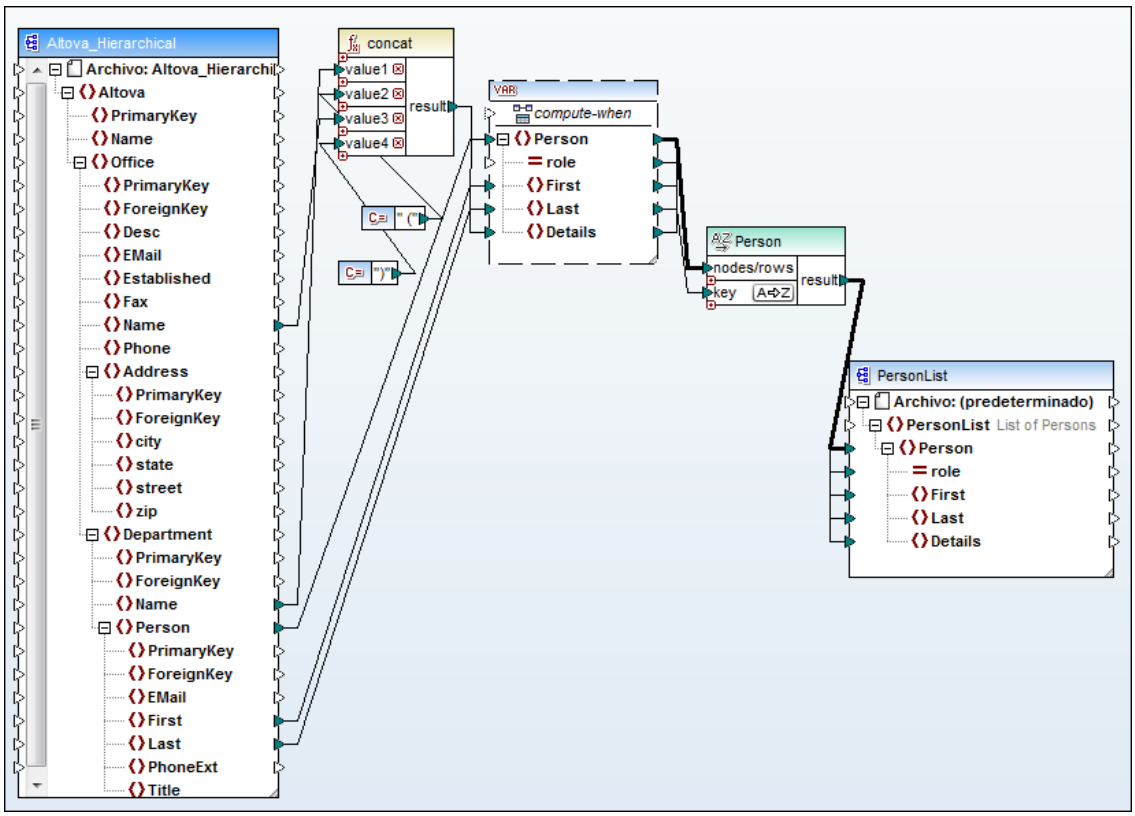
Para definir el criterio de ordenación:

- Conecte el elemento que desea usar como criterio de ordenación (p. ej. **Last**) con el parámetro **key** del componente de ordenación (que pasa a llamarse **Person**).



En los resultados de la asignación los elementos **Person** aparecerán ordenados según el elemento **Last**.

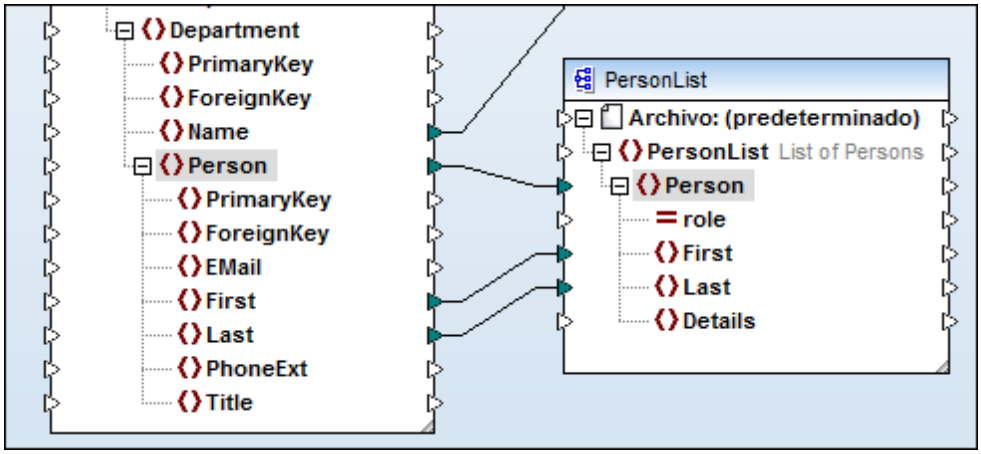
Tomemos como ejemplo el archivo `Altova_Hierarchical_Sort.mfd` de la carpeta `MapForceExamples`.



El objetivo es ordenar alfabéticamente las personas de cada oficina e incluir información detallada sobre los nombres de departamento y sobre la oficina. Para ello, este ejemplo usa una variable que permite acceder a los elementos primarios (primarios del nodo **Person**).

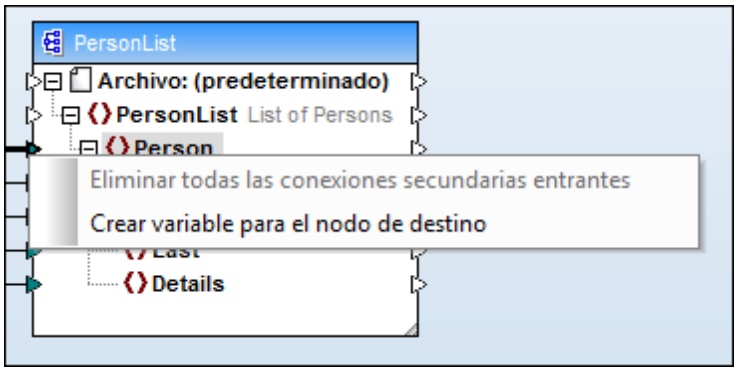
 © 2014 Altova GmbH

Este era el estado inicial de esta asignación:

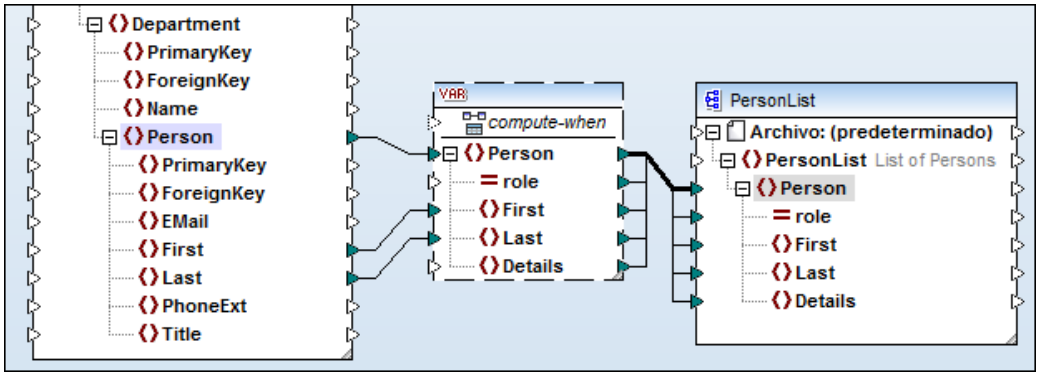


Puesto que deseamos ordenar los elementos **Person**, se conectan los elementos **Person** de ambos componentes (y también se conectan los secundarios que tienen el mismo nombre).

1. Haga clic con el botón secundario en el icono de entrada (en el triángulo) del elemento **Person** del componente de destino.
2. En el menú contextual seleccione **Crear variable para el nodo de destino**.

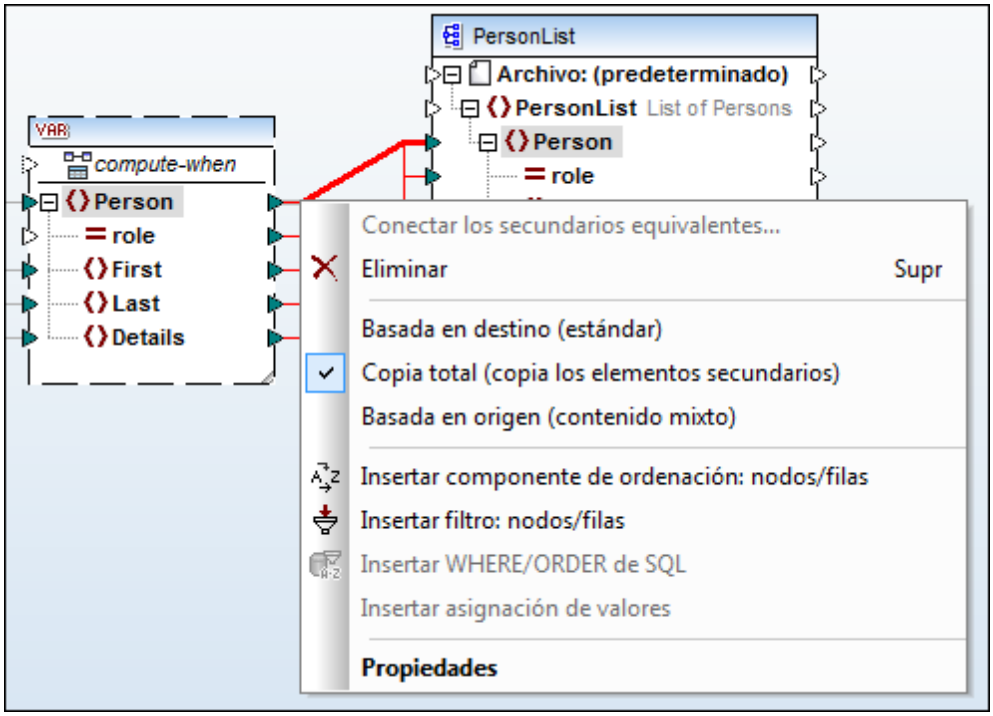


Esto inserta una variable intermedia compleja entre los componentes y se conecta a los elementos secundarios que tienen el mismo nombre. Para más información consulte el apartado Variables intermedias.

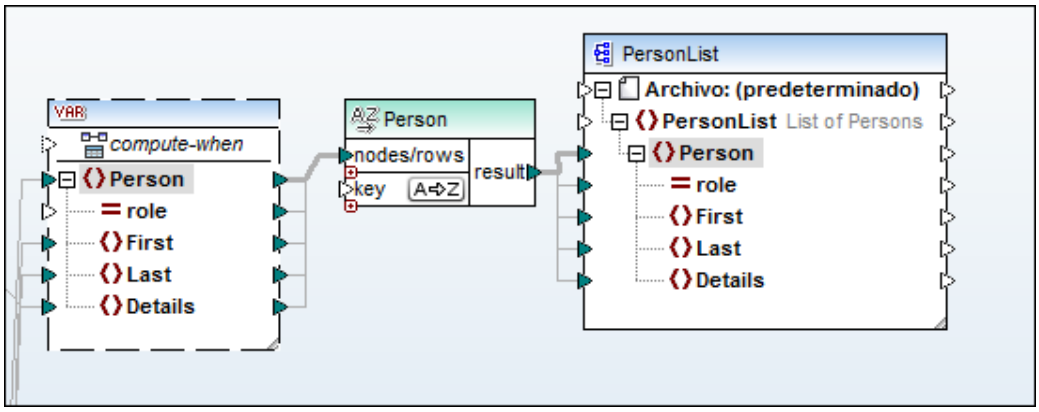


Puesto que queremos ordenar los elementos **Person** según su elemento **Last**, ahora insertamos el componente de ordenación.

3. Haga clic con el botón secundario en el conector de copia total y seleccione **Insertar componente de ordenación: nodos/filas**.



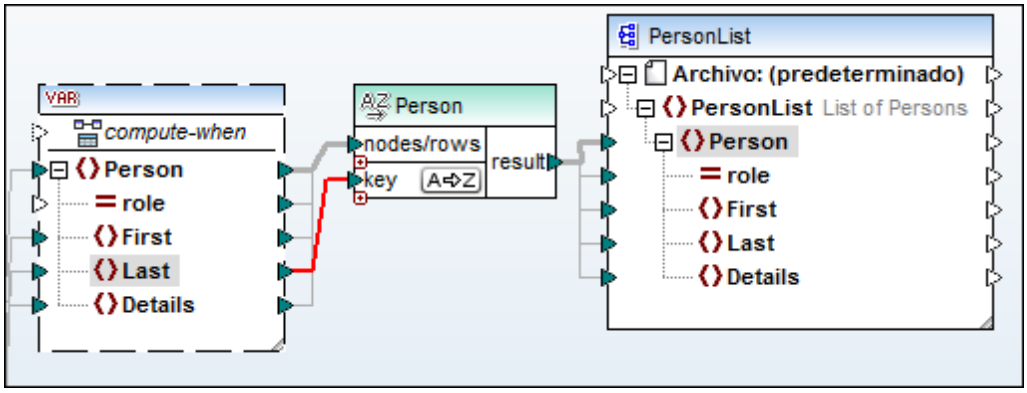
Esto inserta el componente de ordenación, que automáticamente se conecta a los elementos pertinentes.



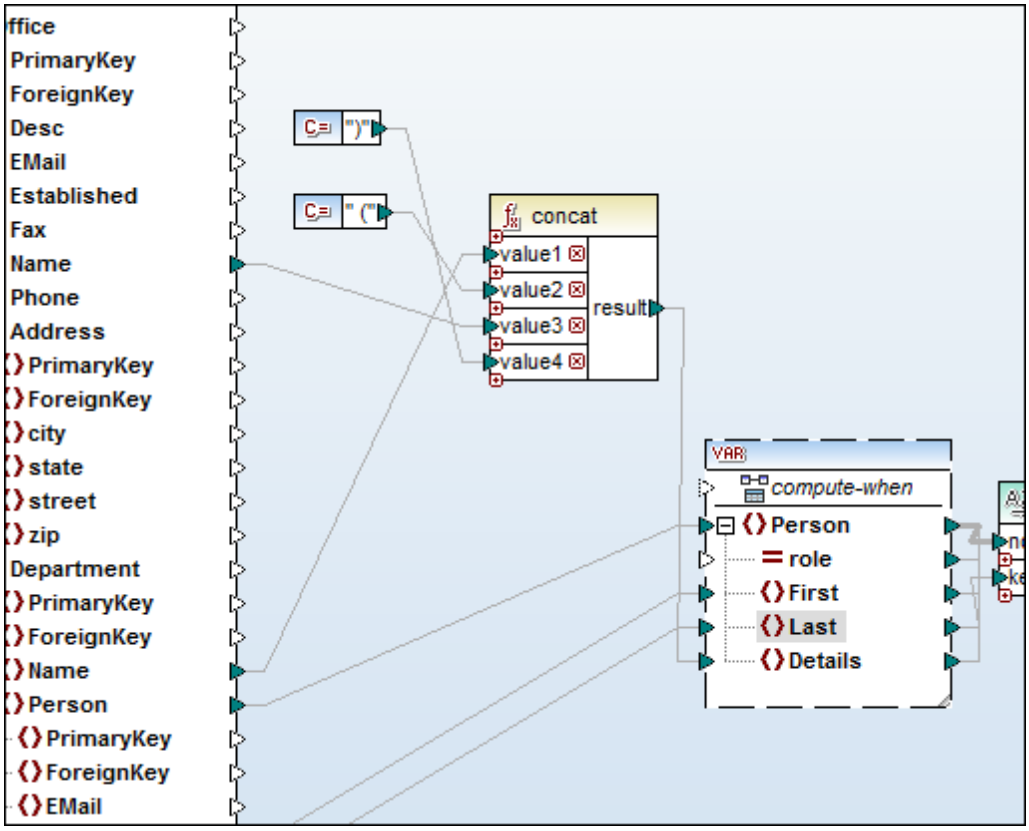
Lo única que falta es definir el criterio de ordenación.

4. Conecte el elemento **`Last`** de la variable intermedia al parámetro **`key`** del componente de ordenación.

 © 2014 Altova GmbH



5. Abra el panel **Resultados** para ver una vista previa del resultado.
   Todas las personas (**`Person`**) están ordenadas alfabéticamente por apellido (**`Last`**).
   Vuelve al panel **Asignación**.
6. Use la función **`concat`** y las constantes de la imagen siguiente para conectar los demás elementos.
7. Conecte el parámetro **`result`** de la función **`concat`** con el elemento **`Details`** de la variable intermedia.



8. Abra el panel **Resultados** para ver otra vez los resultados.

```
<PersonList xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamesp
C:/DOCUME~1/MYDOCU~1/Altova/MapForce2012/MapForceExamples/PersonList.xsd">
  <Person>
    <First>Jessica</First>
    <Last>Bander</Last>
    <Details>IT &amp; Technical Support (Nanonull, Inc.)</Details>
  </Person>
  <Person>
    <First>Valentin</First>
    <Last>Bass</Last>
    <Details>IT &amp; Technical Support (Nanonull Partners, Inc.)</Details>
  </Person>
  <Person>
    <First>Theo</First>
    <Last>Bone</Last>
    <Details>Administration (Nanonull Partners, Inc.)</Details>
  </Person>
```

Ahora las personas siguen estando ordenadas por apellido, pero se incluye información adicional (del elemento **`Details`**). El nombre de la oficina y del departamento aparece ahora debajo de cada persona porque la variable intermedia permite acceder a los datos primarios de cada nodo secundario. Esto solamente es posible gracias a la variable intermedia.

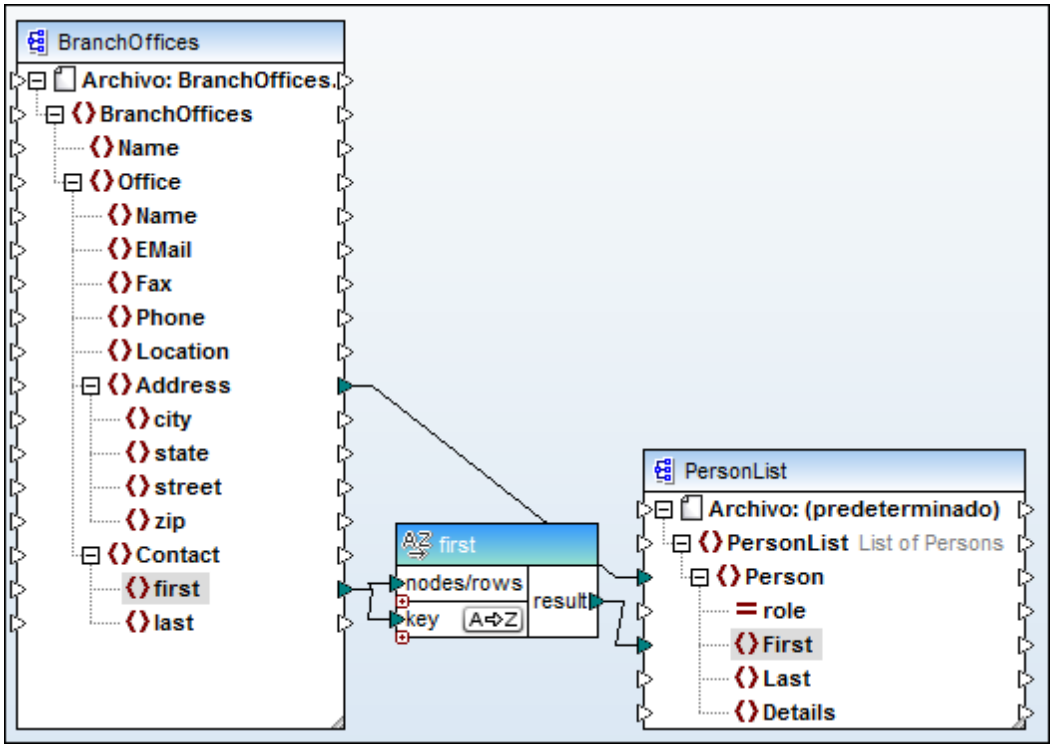
Para invertir la secuencia de clasificación:

- Haga clic en el icono A⇨Z del componente de ordenación. Ahora el icono es Z➔A, indicando que se invirtió la secuencia.

Para ordenar datos de entrada formados por elementos de tipo simple:

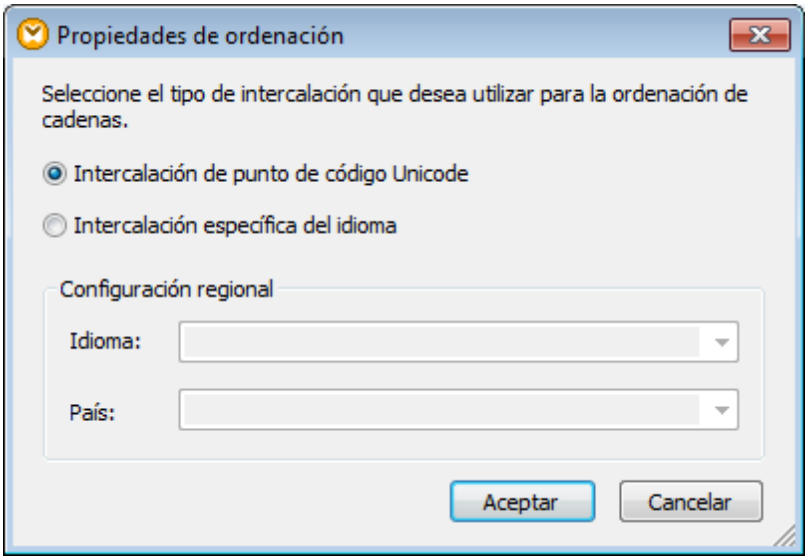
- Conecte el elemento de contenido simple (p. ej. **`First`**) con los dos parámetros (**`nodes/row`** y **`key`**) del componente de ordenación.

© 2014 Altova GmbH



Ordenar datos de base de datos (usando el motor de ejecución integrado BUILTIN)
También puede usar el componente de ordenación para ordenar los datos de una tabla. Sin embargo, para ello conseguirá un mayor rendimiento si utiliza el componente WHERE/ORDER de SQL.

Para ordenar cadenas de texto usando normas de un idioma concreto:
Haga doble clic en el título del componente de ordenación. Aparece este cuadro de diálogo:



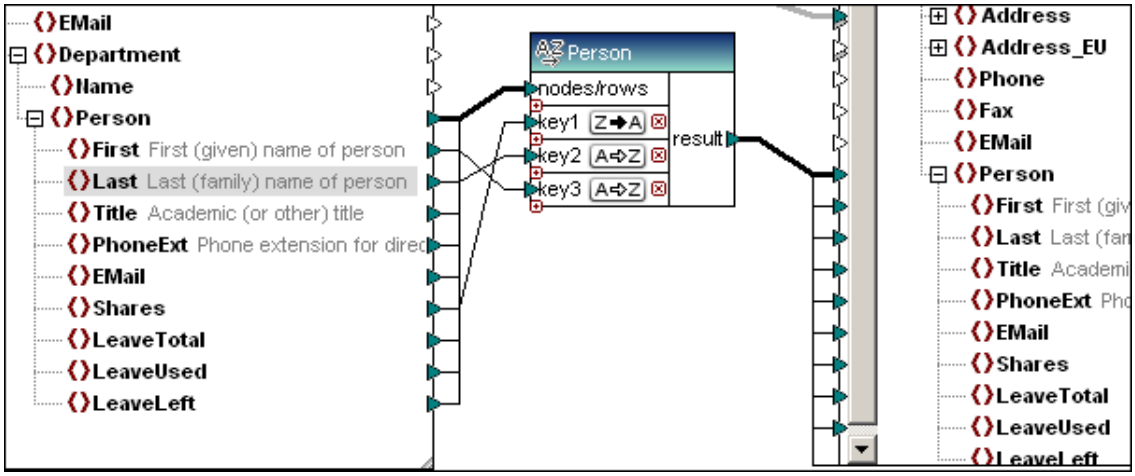
Intercalación de punto de código Unicode: esta opción (predeterminada) compara/ordena las cadenas de texto según los valores de punto de código. Estos valores son enteros que se asignaron a caracteres abstractos del conjunto de caracteres universal (UCS) adoptado por el Unicode Consortium. Esta opción permite ordenar muchos lenguajes y scripts.

Intercalación específica del idioma: esta opción permite definir el idioma y la variación del idioma que desea usar para la ordenación. Esta opción es compatible con el motor de ejecución integrado. También puede ser compatible con XSLT dependiendo del motor utilizado para ejecutar el código.

Para ordenar los datos según varios criterios diferentes:

Con el componente de ordenación también puede definir varios criterios de ordenación diferentes para los mismos datos.

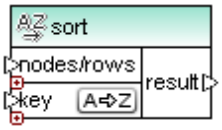
- Al hacer clic en el icono **+** se añade un criterio nuevo de ordenación (es decir, **key2**) en el componente.
- Al hacer clic en el icono **x** se elimina el criterio correspondiente.
- Al colocar un conector sobre un icono **+** se inserta/añade automáticamente un criterio nuevo conectado al elemento de origen.



Observe que **key1** tiene mayor prioridad que **key2** y **key2** tiene mayor prioridad que **key3**.

Para insertar un componente de ordenación de forma convencional:

- Haga clic en el icono de la barra de herramientas para insertar un componente de ordenación.
  Se inserta el componente de ordenación sin conectar y el nombre del componente (en la barra de título) es **sort**.

  Nada más conectar el componente de ordenación con el componente de origen, el componente de ordenación toma el nombre del elemento conectado al parámetro **`nodes/rows`**.

 © 2014 Altova GmbH

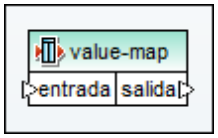
11.4 Asignación de valores: transformar datos de entrada

El componente de asignación de valores permite transformar un valor de entrada en otro valor de salida usando una tabla de búsqueda. Este componente es muy útil a la hora de convertir tipos de enumeración diferentes. La asignación de valores tiene un elemento de entrada y otro de salida.

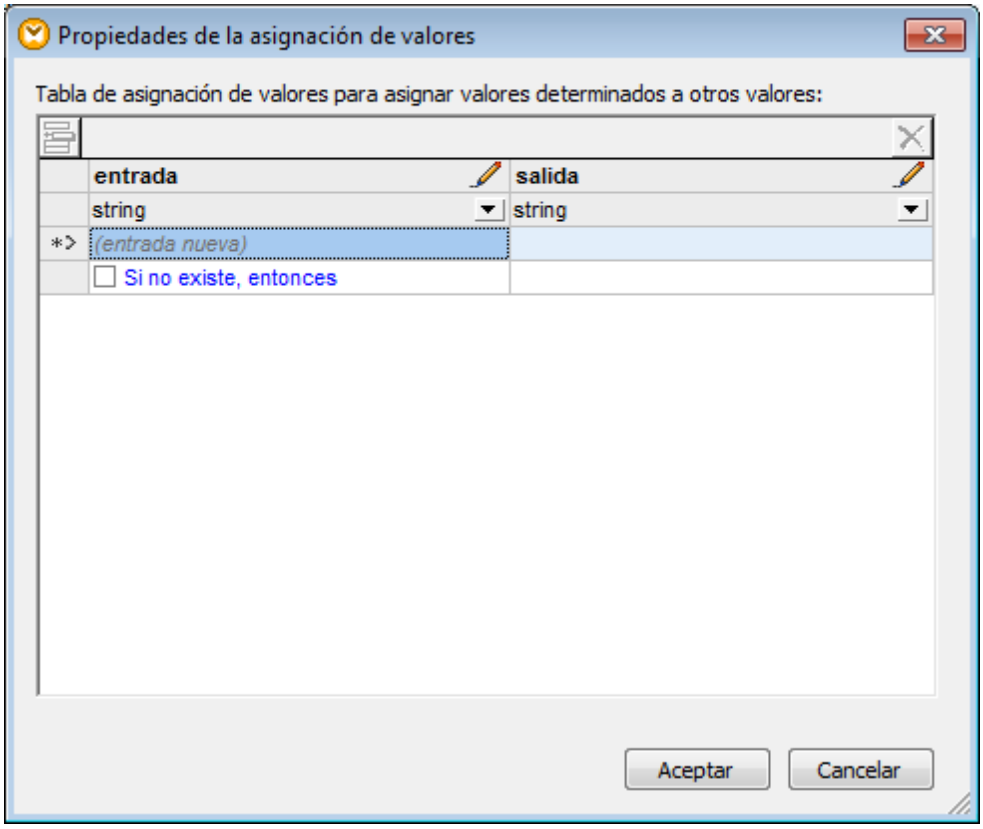
Nota: si quiere recuperar/filtrar solamente ciertos datos y no otros, use un componente de **filtrado**. Para más información consulte Tabla de búsqueda: filtrar datos dinámicos del apartado siguiente, Filtrado de datos, Crear jerarquías a partir de archivos CSV y FLF.

Para usar una asignación de valores:

1. Seleccione la opción de menú **Insertar | Asignación de valores** o haga clic en el icono .

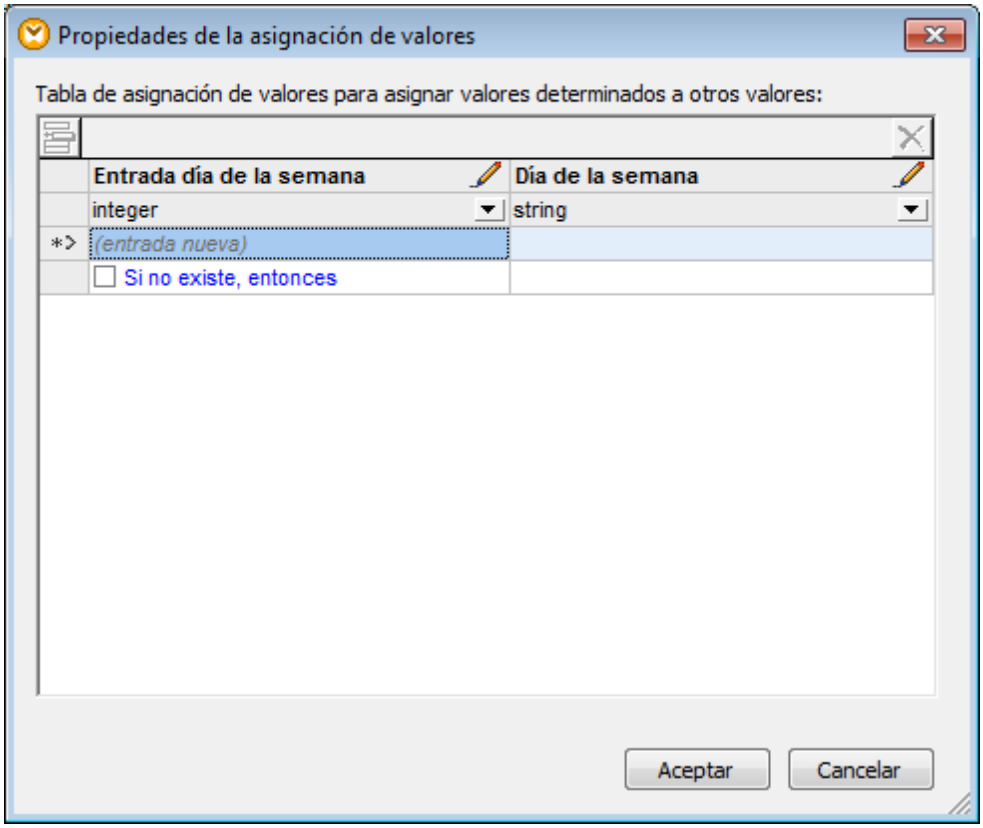


2. Haga doble clic en la asignación de valores (componente **`value-map`**) para abrir la tabla de la asignación de valores.

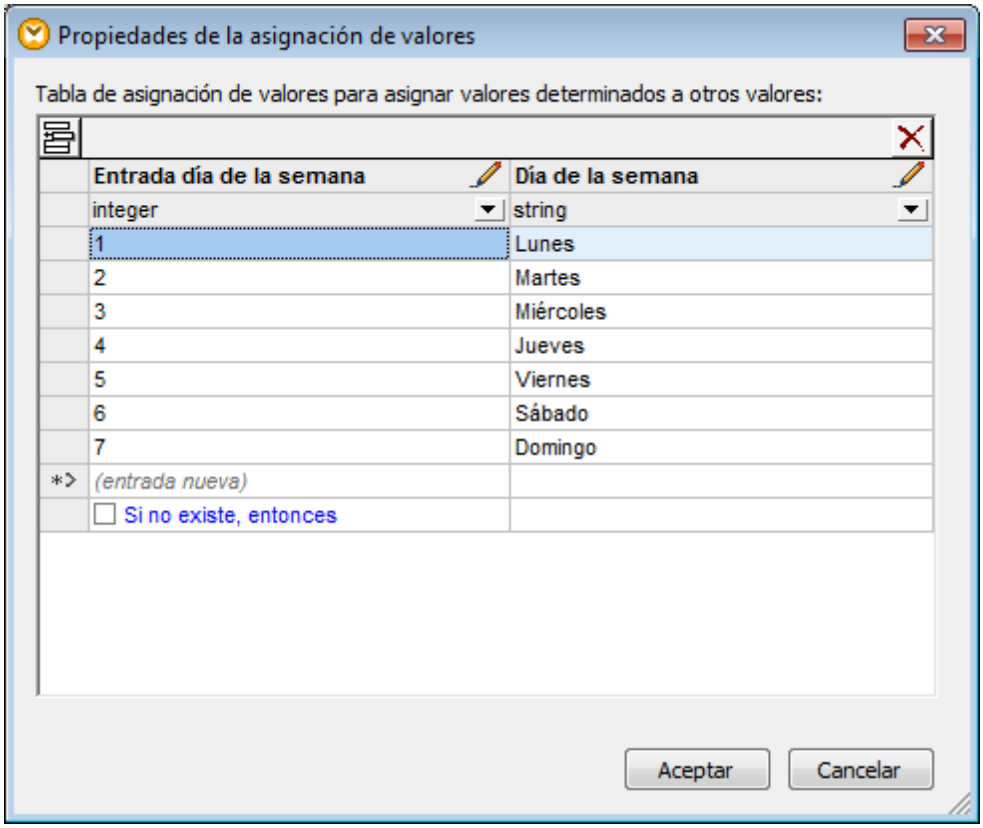


3. Haga clic en los encabezados de columna y escriba **`Entrada día de la semana`** en la primera columna y **`Día de la semana`** en la segunda.

 © 2014 Altova GmbH



4. Escriba el valor de entrada que desea transformar en la columna **`Entrada día de la semana`**.
5. En la columna **`Día de la semana`** escriba en qué valor quiere transformar el valor de entrada.
6. Para introducir un par de valores nuevo basta con escribir en el campo de entrada *`(entrada nueva)`*.
7. Haga clic en el cuadro combinado de tipo de datos, bajo el encabezado de las columnas para seleccionar el tipo de datos de entrada y el de salida (en este caso, **`integer`** y **`string`**).

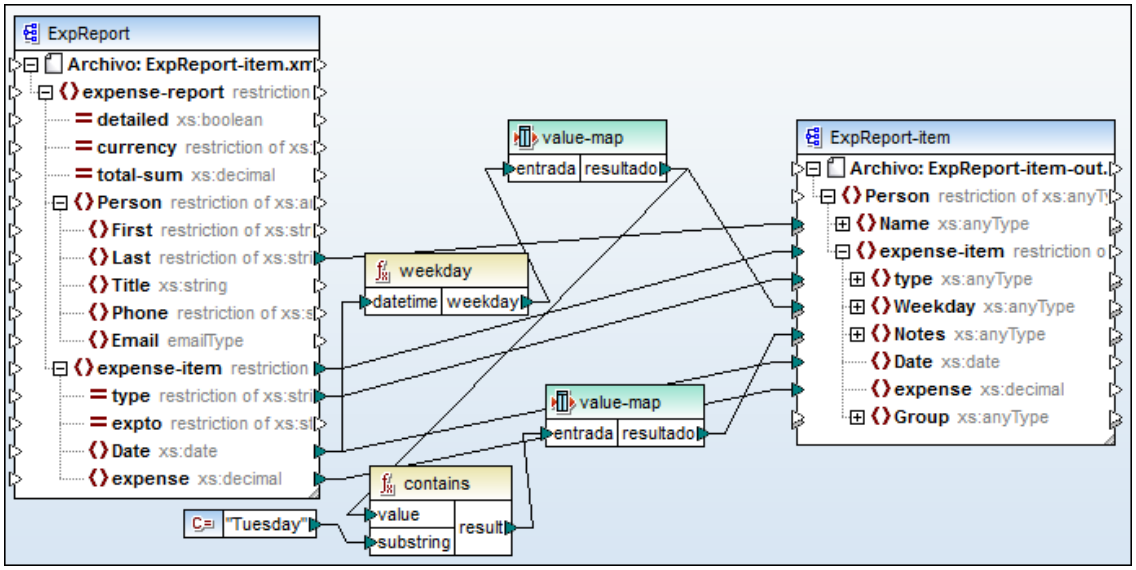


Nota: marque la casilla *Si no existe, entonces* y escriba un valor para definir un valor de salida alternativo en caso de que los valores suministrados no existan en la entrada. Para pasar a través de datos de origen sin cambiarlos, consulte el apartado Pasar datos sin cambiarlos a través de una asignación de valores.

8. Para cambiar el nombre de las columnas haga clic en el icono en forma de lápiz. El nombre de las columnas aparece en el componente de asignación de valores, para que pueda reconocer rápidamente qué función cumple el componente.

Como ejemplo podemos ver el archivo `Expense-valmap.mfd` guardado en la carpeta `...\MapForceExamples\Tutorial`, que incluye una asignación de valores.

 © 2014 Altova GmbH



Esta asignación extrae el día de la semana del elemento **`Date`** del componente de origen, convierte el valor numérico en texto y coloca el texto en el elemento **`Weekday`** del componente de destino (es decir, lunes, martes, etc.)

- La función **`weekday`** extrae el número del día de la semana del elemento **`Date`** del componente de origen **ExpReport**. El resultado de esta función son enteros comprendidos entre el 1 y el 7.
- El componente de asignación de valores transforma los enteros en los días de la semana (es decir, lunes, martes, etc.), tal y como mostraba la tabla que aparece al principio del apartado.
- Si el resultado de la asignación contiene `Tuesday` (martes), entonces se asigna el valor de salida correspondiente *Preparar informes financieros* al elemento **`Notes`** del componente de destino.
- Al abrir el panel **Resultados** podemos ver el archivo XML de destino con los datos 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 del mouse por la asignación de valores aparece un mensaje emergente que enumera los valores definidos en la tabla de la asignación de valores.

El resultado de varias funciones (como las funciones lógicas o de cadenas de texto)

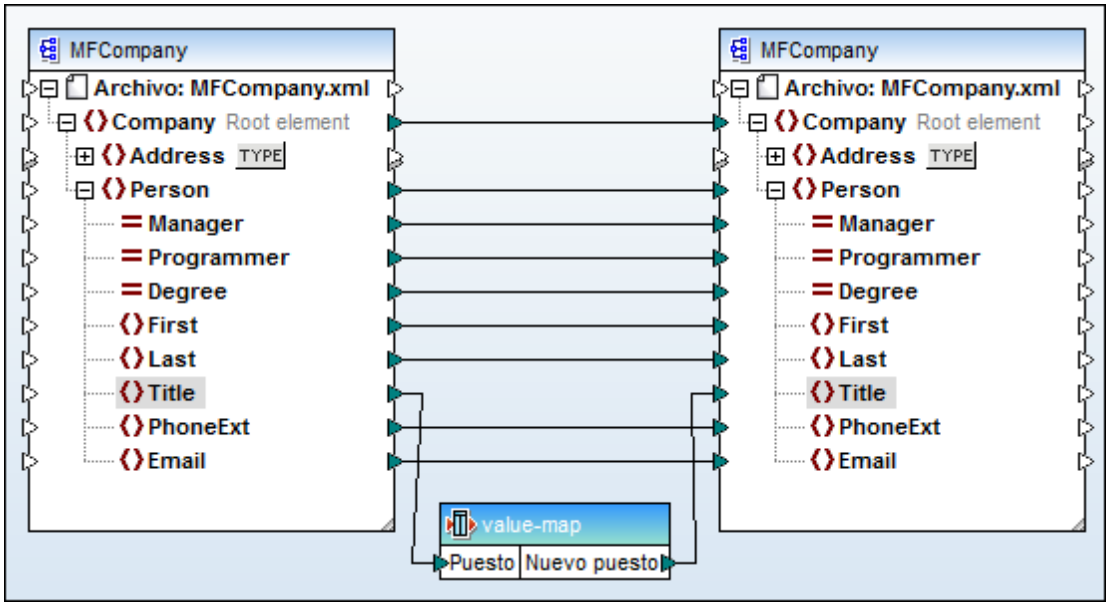
solamente puede ser el valor booleano **`true`** o **`false`**. Por tanto, en el campo **`entrada`** de la tabla de la asignación de valores debe introducir el valor que desea probar, p. ej. **`true`**.

© 2014 Altova GmbH

11.4.1 Pasar datos sin cambiarlos a través de una asignación de valores

En este apartado describimos una asignación en la que se transforman solamente ciertos nodos, mientras que el resto se pasan al componente de destino tal y como aparecen en el componente de origen.

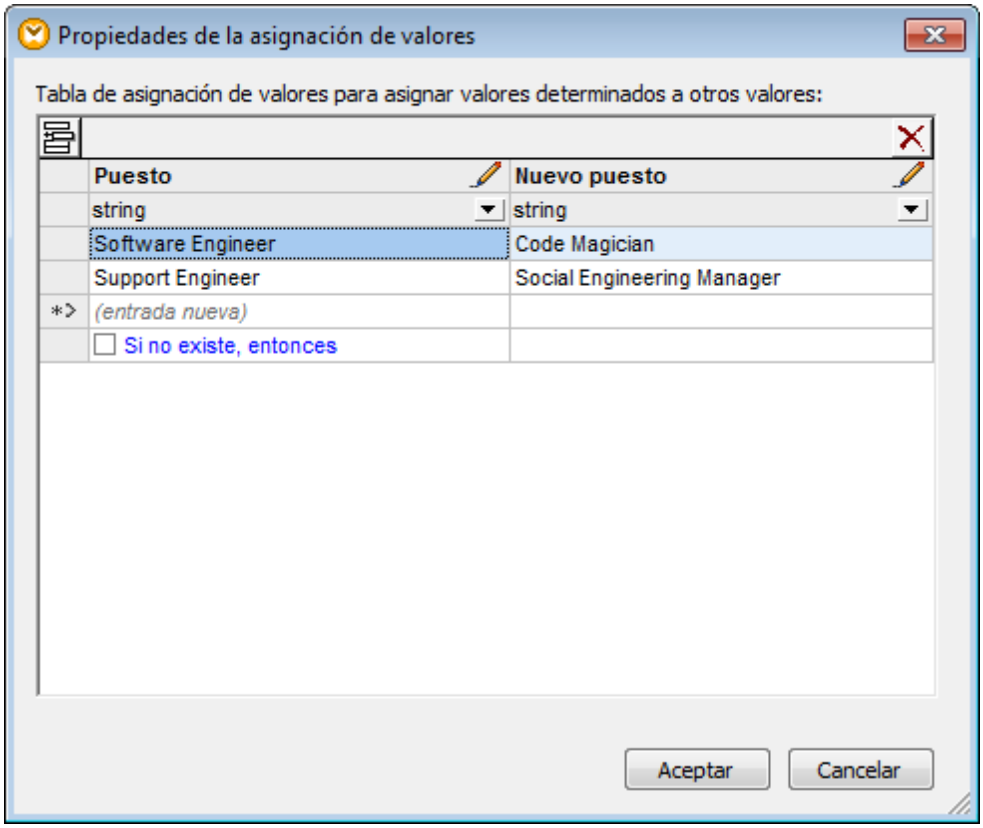
Por ejemplo, imaginemos que una compañía tiene una subsidiaria y necesita cambiar el nombre de dos puestos de trabajo de la subsidiaria y conservar el nombre de los demás puestos tal y como están.



La asignación obvia sería la de la imagen anterior, que usa la asignación de valores value-map para transformar el nombre de los puestos.
Si abrimos el panel **Resultados** podemos ver el resultado de esta 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>
```

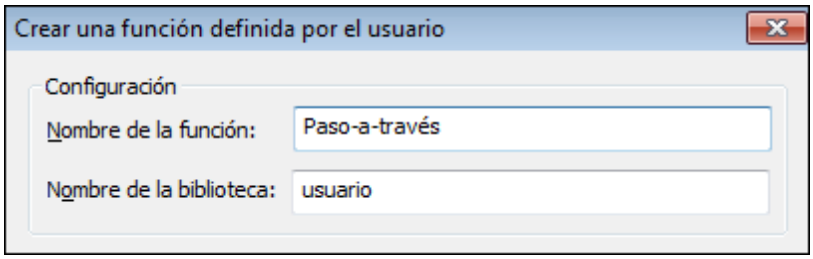
En el caso de empleados que no tengan ninguno de los dos puestos del componente de asignación de valores, el elemento **`Title`** se elimina en el archivo de salida.



Posible alternativa:
Si marcamos la casilla *Si no existe*, *entonces* y escribimos un término de sustitución, el nodo **`Title`** vuelve a aparecer en el archivo de salida, pero ahora todos estos empleados tienen el mismo **`Nuevo puesto`**.

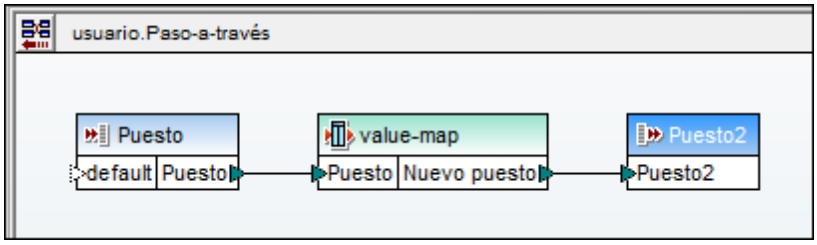
Solución:
Crear una función definida por el usuario que contenga el componente de asignación de valores y usar la función **`substitute-missing`** para suministrar los datos originales para los nodos vacíos.

1. Haga clic en el componente de asignación de valores y seleccione **Función | Crear una función definida por el usuario a partir de la selección**.

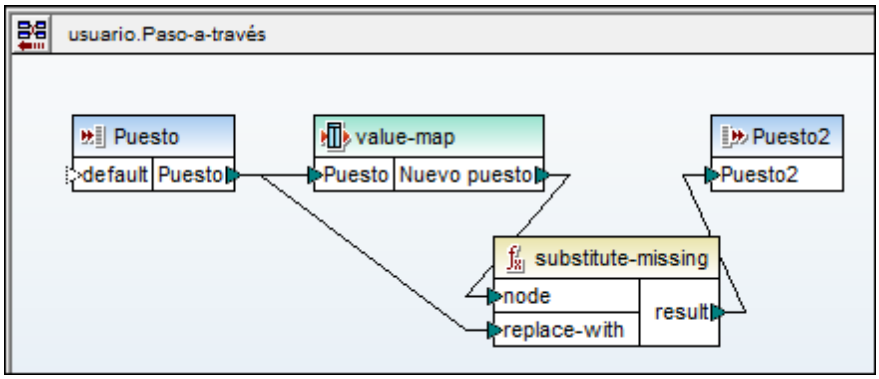


2. Escriba el nombre de la función, p. ej. Paso-a-través y haga clic en **Aceptar**.

 © 2014 Altova GmbH



3. Inserte una función **substitute-missing** desde la sección **core | node functions** de la ventana Bibliotecas y cree las conexiones que aparecen en esta imagen.



4. Abra el panel **Resultados** para ver el resultado:

- Las dos designaciones de **`Puesto`** de la asignación de valores se transforman en el **`Nuevo puesto`**.
- Los demás nodos **`Title`** del archivo de origen se pasan al archivo de destino tal y como aparecen en el componente de origen.

```
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 evalúa los datos de entrada.

- 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`**.

- Si los datos entrantes no coinciden con **ninguna entrada** de la primera columna, entonces no se pasa ningún valor de la asignación de valores al parámetro **`node`**, es decir, es un **nodo vacío**.

Cuando esto ocurre, la función **substitute-missing** recupera el nodo original y los datos originales del nodo **Title** y los pasa al parámetro **replace-with** y después a **Puesto2**.

© 2014 Altova GmbH

11.4.2 Propiedades de las asignaciones de valores

Iconos:

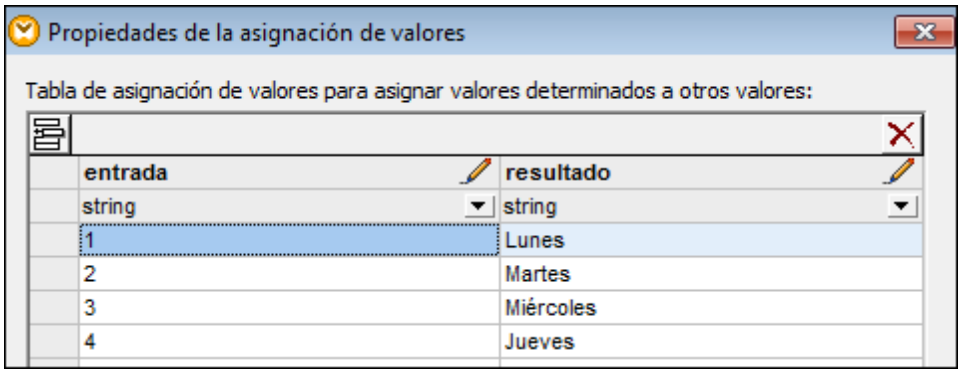
Inserta una fila nueva antes de la fila activa.

Elimina la fila activa.

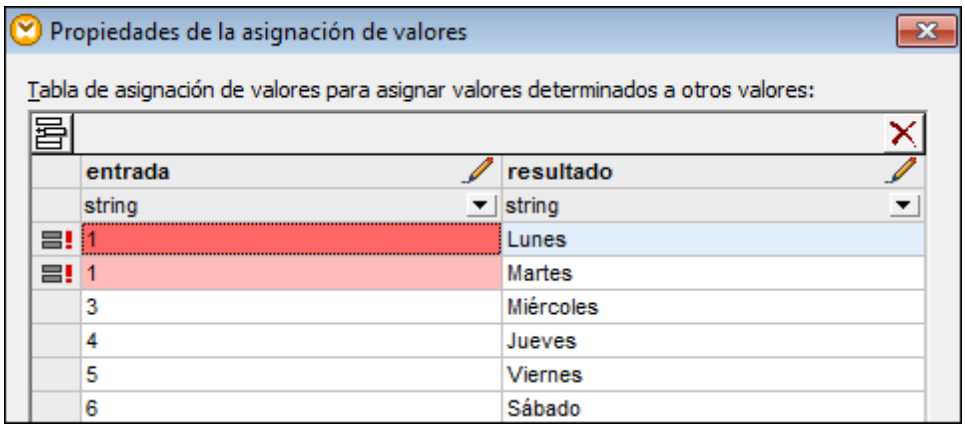
Permite **editar** el encabezado de la columna.

También puede reordenar las filas. Para ello arrastre la fila pertinente hasta su nueva posición.

Cambiar el encabezado de la columna:
Para editar el nombre (encabezado) de la columna, haga doble clic en el encabezado o haga clic en el icono en forma de lápiz. Recomendamos ponerle un nombre más significativo a estas dos columnas para que pueda identificar con mayor facilidad el componente (porque los nombres de las columnas aparecen en la asignación).



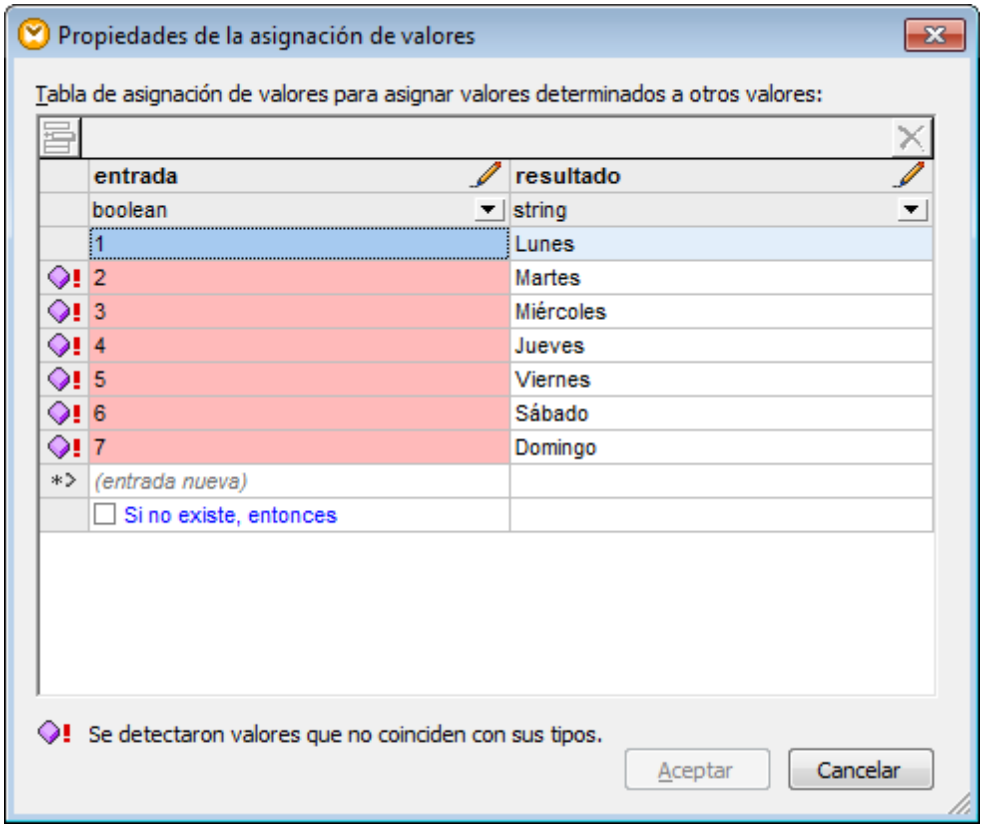
Usar valores de entrada únicos:
Los valores introducidos en la columna de entrada deben ser únicos. Si introduce dos veces el mismo valor, ambos se resaltan para que corrija el error.



Una vez corregido el error, se habilita el botón **Aceptar**.

Tipo de datos de entrada/salida
Los tipos de datos de entrada/salida se revisan automáticamente nada más seleccionar un tipo en el cuadro combinado. Si los tipos de datos no son compatibles, entonces se resaltan los campos afectados y se deshabilita el botón **Aceptar**. Elija un tipo de datos compatible.

En la imagen siguiente, por ejemplo, se eligieron los tipos de datos `boolean` y `string`.

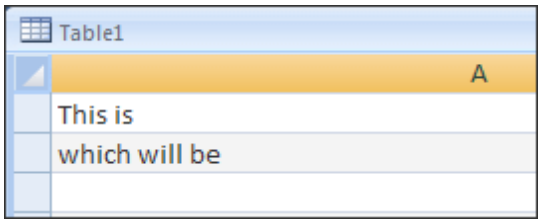


 © 2014 Altova GmbH

11.5 Reemplazar caracteres especiales en datos de BD

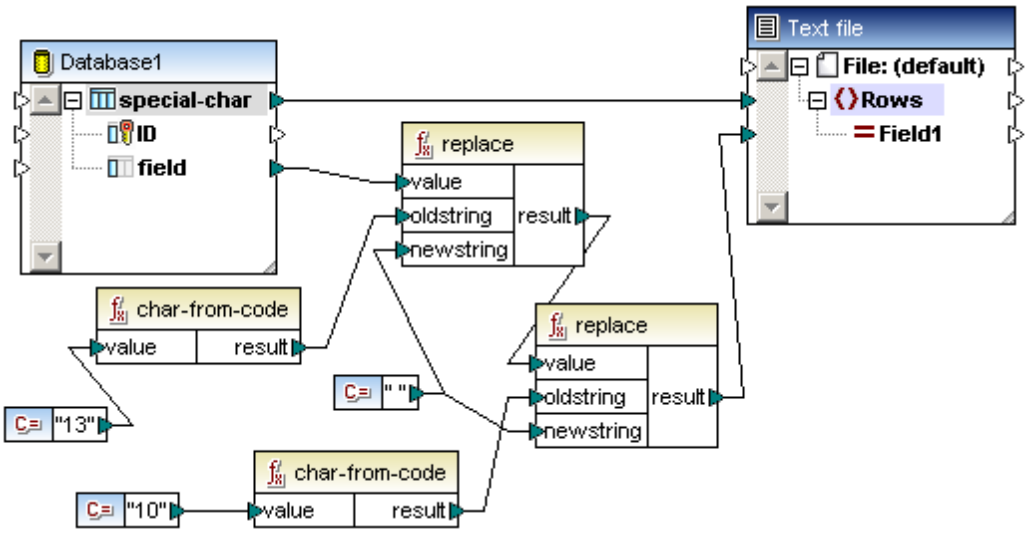
Cuando se transforman datos de BD a veces es necesario quitar algunos caracteres especiales (p. ej. newline o salto de línea, retornos de carro / saltos de línea o códigos hexa 0D/0A) del origen de datos. Esto se puede conseguir con una función **`char-from-code`** de la biblioteca de funciones **`string`**.

En la asignación que aparece más abajo el origen de datos es una base de datos MS Access formada por una sola tabla de dos filas. El texto de cada fila tiene varias líneas separadas por saltos de línea (LF), es decir, por el código hexa 0A, decimal 10.



En la asignación que aparece más abajo:

- Los códigos hexa 0D y 0A (decimal 13 y 10) se convierten en una cadena de texto, que puede procesarse después con la función **`replace`**
- La función **`replace`** reemplazar el parámetro **`oldstring`**, suministrado por la función **`char-from-code`** (0D/0A), y el carácter de espacio lo suministra una constante para que se utilice como parámetro **`newstring`**.
- Los datos convertidos se colocan en un componente de texto de destino.



Si abrimos el panel **Resultados** podemos ver los resultados de esta asignación. Los caracteres LF (CR) que estaban en el campo de la BD se reemplazaron con un carácter de espacio.

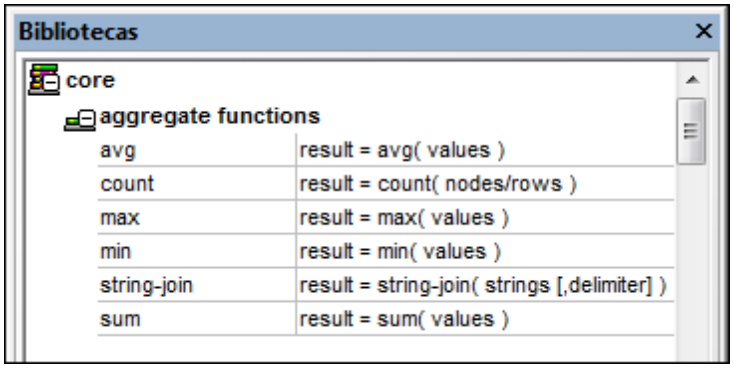
```
1   This is  our new  company policy
2   which will be  implemented immediately.
3
4
5
6
```


© 2014 Altova GmbH

11.6 Funciones de agregado: min, max, sum, count, avg

Las funciones de agregado realizan operaciones en un conjunto de valores de entrada (o en una secuencia de valores) en lugar de en un solo valor. Las funciones de agregado están disponibles en la biblioteca principal **core**. En este apartado usamos como ejemplo la asignación `Aggregates.mfd`, disponible en la carpeta `...\Tutorial`.

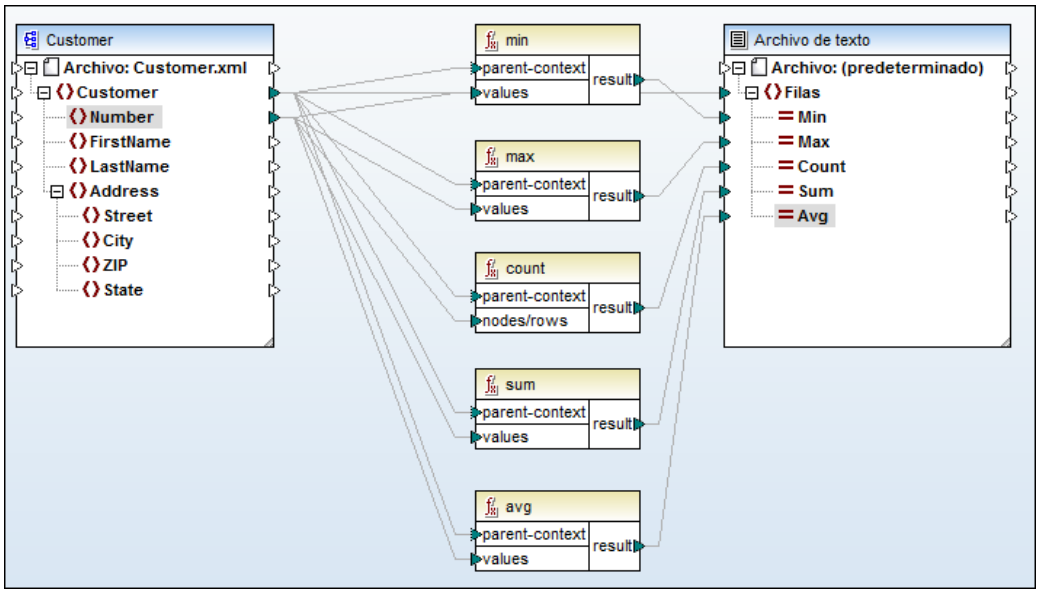
Consulte también el apartado Funciones de agregado: sumar nodos en XSLT 1.0/2.0 para aprender a crear funciones de agregado usando plantillas con nombre.



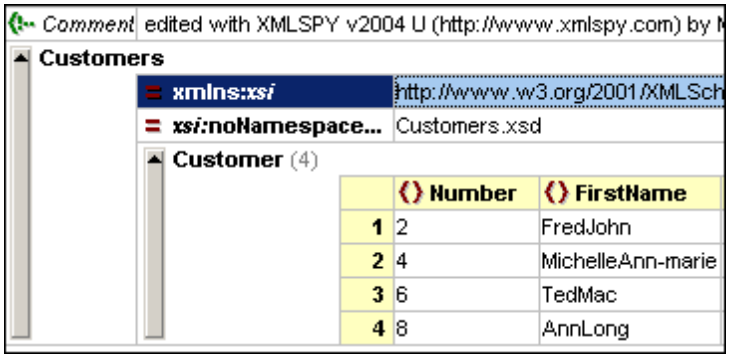
Con Java seleccionado

Las funciones de agregado tienen dos entradas:

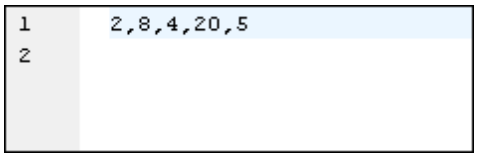
- la entrada **`values`** (**`nodes/rows`**) se conecta al elemento del componente de origen que suministra los datos, en este caso el elemento **`Number`**.
- la entrada **`parent-context`** se conecta al elemento en el que se desea iterar (es decir, el contexto. En este caso se quiere iterar en todos los elementos **`Customer`**). Este parámetro es opcional.



La instancia de entrada, en este caso, 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.

 © 2014 Altova GmbH

11.7 Asignar varias tablas a un archivo XML

Asignar varias tablas jerárquicas a un solo archivo XML de salida

- Imagine que tiene una base de datos y quiere extraer cierto número de tablas y asignarlas a un archivo XML.
- Entre estas tablas existen relaciones de clave principal y foránea.
- Las tablas relacionadas deben aparecer como elementos secundarios en el archivo XML resultante.

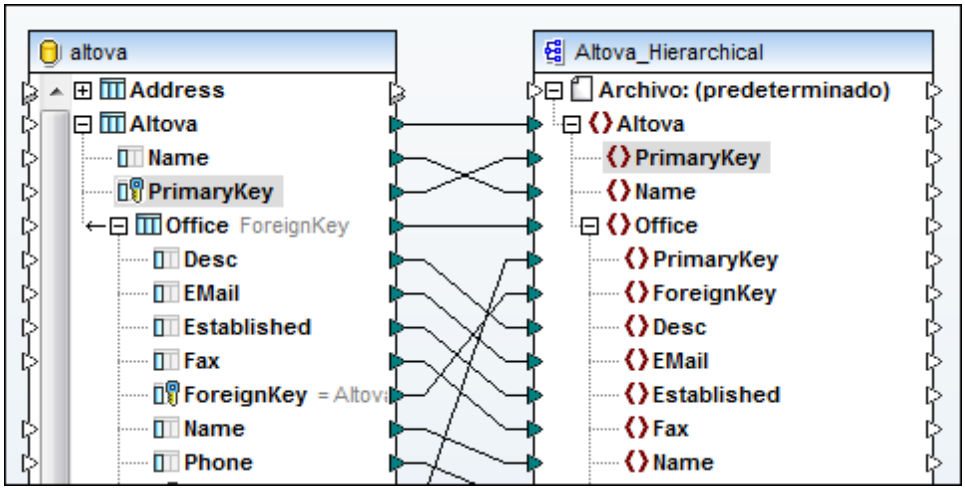
La asignación `DB_Altova_Hierachical.mfd`, disponible en la carpeta `...\MapForceExamples`, ilustra cómo hacer todo esto en una asignación de datos de BD. El esquema `Altova_Hierarchical.xsd` está en la misma carpeta. La estructura del esquema es prácticamente idéntica a la jerarquía de la base de datos Access. (El mismo método se puede usar para asignar bases de datos planas de formato XML/SQL.)

La base de datos MS Access `Altova.mdb` está en la carpeta `...\MapForceExamples\Tutorial`.

Requisitos del esquema:

- Todas las tablas relacionadas con Altova aparecen como elementos secundarios del elemento raíz de destino.
- Para conservar las relaciones de tabla todas las asignaciones se crearon bajo la tabla Altova del componente de BD de origen.

  La imagen siguiente muestra la asignación de datos desde la BD jerárquica Access hasta el esquema `Altova_Hierarchical.xsd`.



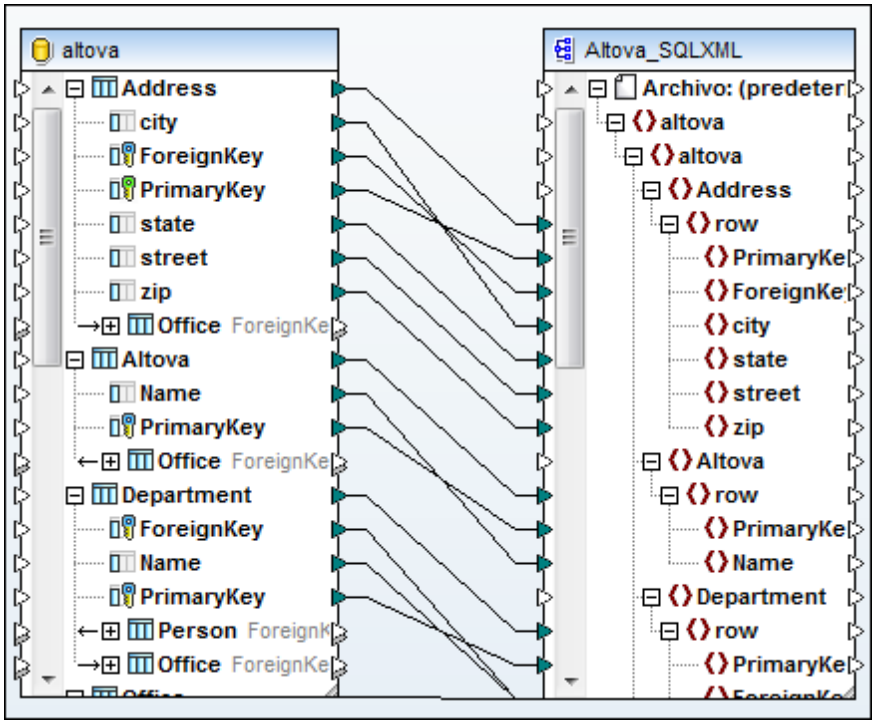
Asignar varias tablas de archivo plano a un solo archivo XML de salida

La imagen que aparece ahora muestra el mismo tipo de asignación a un esquema de BD SQL/XML de archivo plano.

Requisitos del esquema:

- La **estructura** del esquema debe seguir la especificación SQL/XML.
- XMLSpy tiene la capacidad de crear archivos SQL/XML de este tipo a partir de una BD SQL (comando **Conversión | Crear esquema de base de datos**). Después puede usar el **esquema** como componente de destino en MapForce.

- En este caso el nombre de las tablas se asigna al elemento secundario **row** de los elementos del esquema llamados como las tablas (p. ej. la tabla **Address** se asigna al elemento secundario **row** del elemento **Address**).



- Observe que la asignación anterior (`DB_Altova_SQLXML.mfd`) no conserva las relaciones de tabla porque se crearon asignaciones desde tablas raíz diferentes.

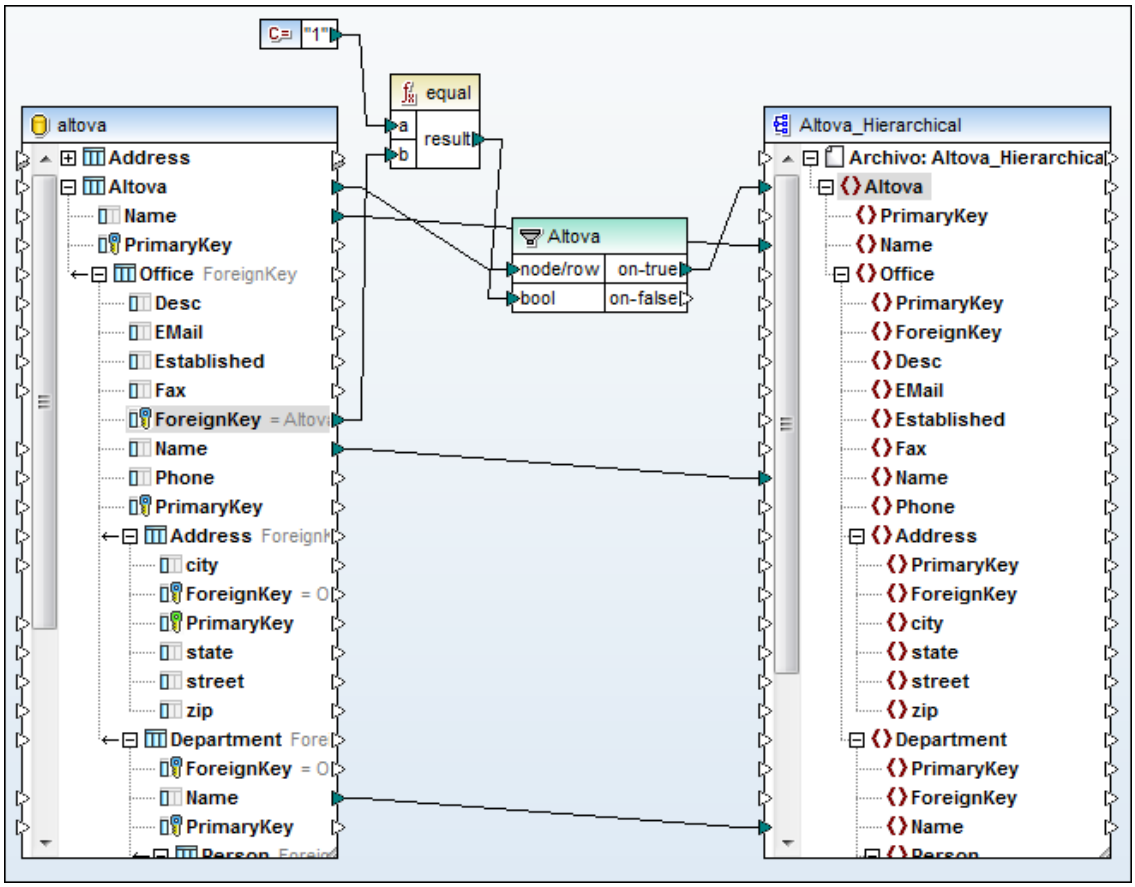
 © 2014 Altova GmbH

11.8 Elemento raíz de los documentos de destino

Elemento raíz de los archivos XML de destino
Cuando cree una asignación al elemento **raíz** del archivo/esquema XML de destino, compruebe que solamente se pasa un elemento o un registro al XML porque los documentos XML solamente pueden tener un elemento raíz.

En este caso, lo más recomendable es utilizar un componente de filtrado para asignar un solo elemento o registro.

- En el ejemplo siguiente, se comprueba si **`ForeignKey`** es igual a 1 y si lo es se pasa un elemento **`Altova`** al elemento raíz de destino.
- Si no existen asignaciones desde ningún elemento de origen hasta el elemento raíz de destino, entonces se inserta automáticamente el elemento raíz del esquema de destino.



Elemento raíz sin limitar:
Si no limita el elemento raíz del esquema de destino, entonces se insertan todos los elementos/registros de origen entre el primer elemento raíz. Esto genera archivos XML con formato XML correcto, pero no válidos.

11.9 Comparación booleana de los nodos de entrada

Tipos de datos en las funciones booleanas
Durante la evaluación de las funciones booleanas de la biblioteca principal **core** (**`equal`**, **`equal-or-greater`**, **`equal-or-less`**, **`greater`**, **`less`**, y **`not-equal`**), el resultado que se obtiene al evaluar los dos nodos de entrada depende tanto de los valores de entrada como de los tipos de datos utilizados para la comparación.

Ejemplo:
Imaginemos que usamos la función **`less`** (*menor que*) para comparar los valores enteros 4 y 12. Esta comparación produce el valor booleano **`true`** porque 4 es menor que 12. Por el contrario, si usamos la función **`less`** para comparar las cadenas de texto 4 y 12, el análisis léxico da como resultado el valor booleano **`false`** porque 4 es alfabéticamente mayor que el primer carácter (1) del segundo operando (12).

Si todos los datos de entrada son del mismo tipo (p. ej. todos los nodos de entrada son tipos numéricos o cadenas de texto), entonces la comparación se hace para el tipo común.

Nodos de entrada de tipo diferente
Si los nodos de entrada son de tipo diferente, p. ej. un entero y una cadena de texto o una cadena de texto y una fecha, entonces se aplica esta norma:

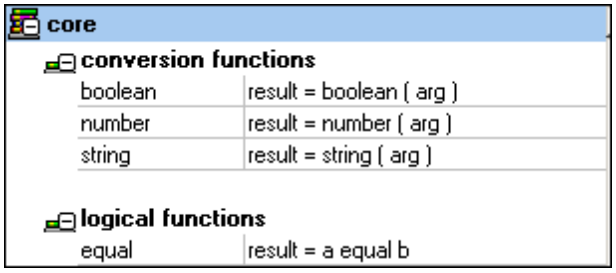
El tipo de datos utilizado para la comparación **siempre es el más general, es decir, el menos restrictivo**, de los dos tipos de entrada.

Antes de comparar dos valores, todos los valores de entrada se convierten a un tipo de datos común. Usando otra vez el ejemplo anterior, el tipo de datos `string` es menos restrictivo que el tipo de datos `integer`. Si comparamos el valor 4 de tipo `integer` y el valor 12 de tipo `string`, el valor 4 de tipo `integer` se convierte en el valor 4 de tipo `string` y después se compara con el valor 12 de tipo `string`.

Esta norma para comparar tipos mixtos sigue las directrices XSLT2 y evita estrategias de conversión de tipos basada en el contenido. La ventaja es que la lógica viene dada por la asignación y no cambia de forma dinámica.

Comprobaciones adicionales:
A partir de la versión 2006sp2 de MapForce, también se comprueba si la asignación tiene combinaciones de tipos incompatibles y se emiten errores de validación y mensajes de advertencia si es necesario. Podemos tomar como ejemplo la comparación de fechas con valores booleanos o valores datetime con valores numéricos.

MapForce es compatible con la conversión de tipos de datos explícita gracias a estas funciones de conversión de tipos de la biblioteca **core**: **`boolean`**, **`number`** y **`string`**. En el contexto anterior, estas tres funciones son aptas para determinar la interpretación de las comparaciones.



Con Java seleccionado

 © 2014 Altova GmbH

Puede añadir estas funciones de conversión a los nodos de entrada de funciones relacionadas, para cambiar el tipo de datos común y el resultado de la evaluación. Por ejemplo, si los nodos de cadena solamente almacenan valores numéricos, puede realizar una comparación numérica añadiendo a cada nodo de entrada la función de conversión **number** (en la sección **conversion** de la biblioteca principal **core**).

11.10 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 de elementos no relacionados entre sí, sin establecer el contexto prioritario, MapForce no sabe qué bucle debe ejecutarse primero. Cuando desconoce el contexto prioritario, MapForce selecciona automáticamente la primera tabla o nodo de origen.

Solución:
Decida en qué tabla debe realizarse la búsqueda primero o qué nodo de origen debe recorrerse primero y establezca el contexto prioritario en su conector.

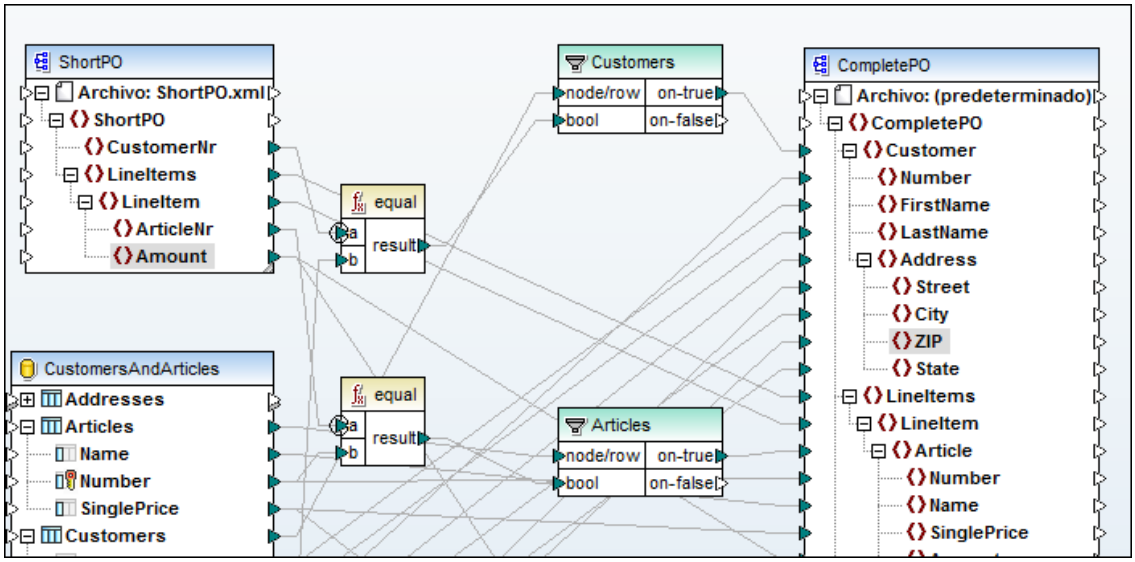
El ejemplo de la imagen siguiente es una versión simplificada del archivo `DB_CompletePO.mfd`, disponible en la carpeta `...\MapForceExamples`.

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.

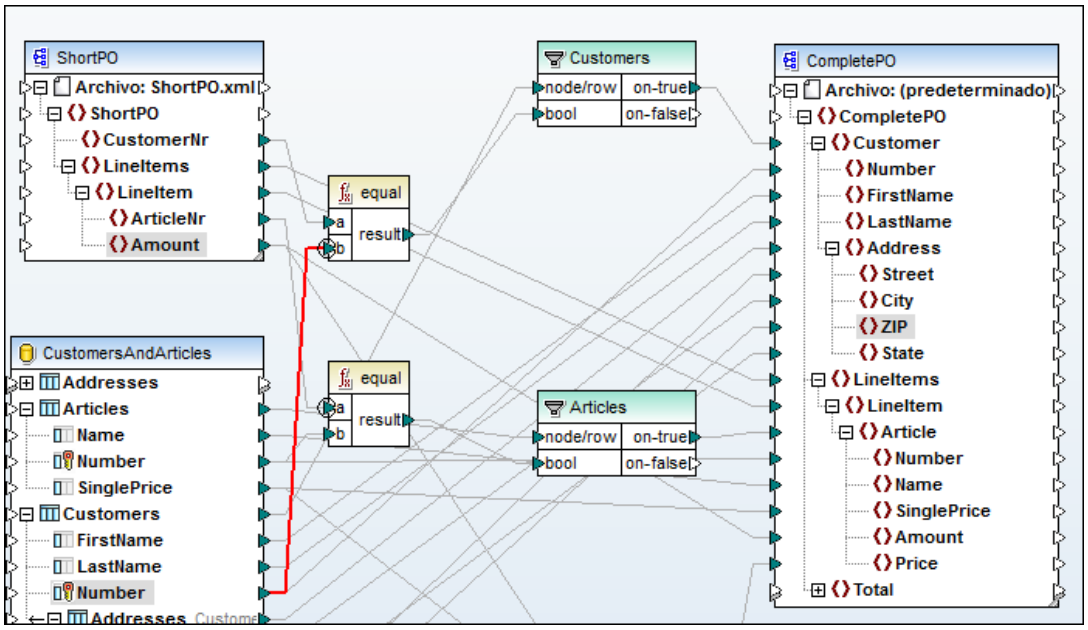
 © 2014 Altova GmbH



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**.



Esto significa que se genera una consulta de BD por cada **`Number`** y el resultado se compara después con cada elemento **`CustomerNr`** de **ShortPO**.

El contexto prioritario en las 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.

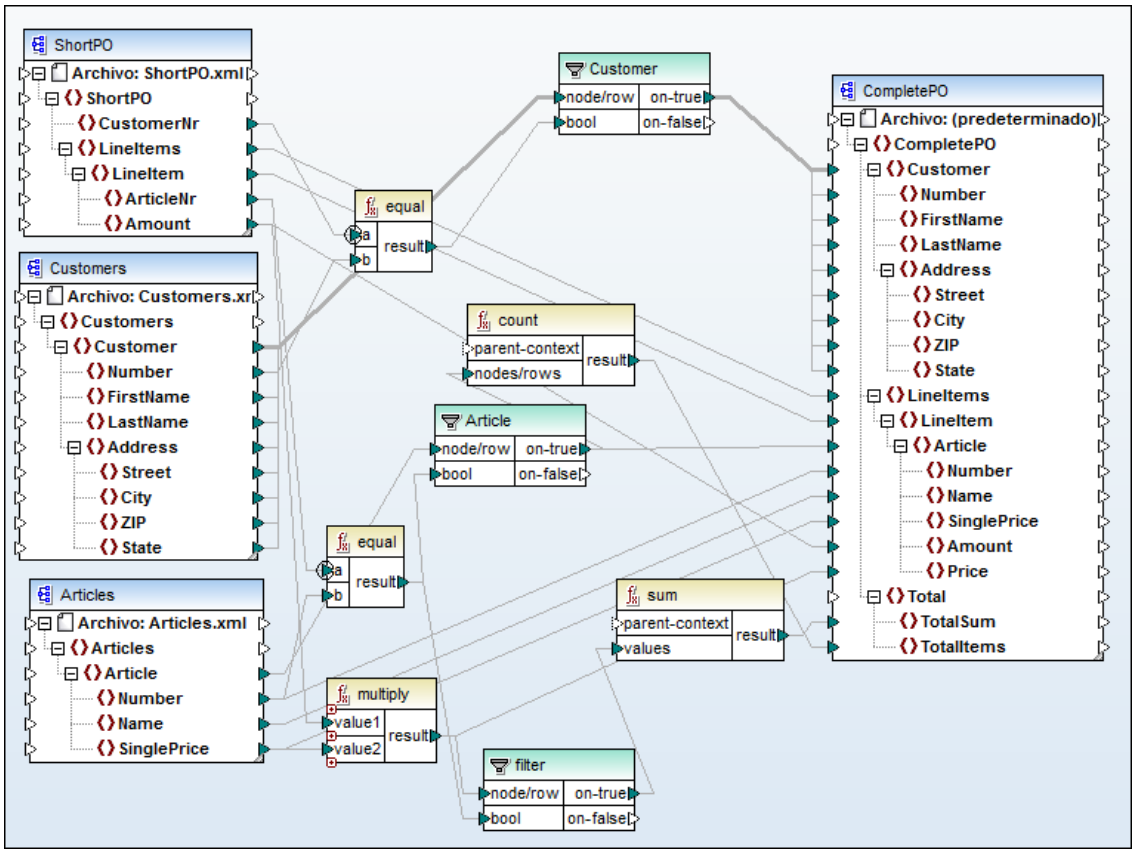
© 2014 Altova GmbH

11.11 Unir varios archivos en un solo archivo de destino

En MapForce puede consolidar varios archivos de origen en un solo archivo de destino.

En el ejemplo de la imagen siguiente se unen varios componentes de origen (basados en esquemas diferentes) en un solo esquema de destino. Para combinar un número aleatorio de archivos basados en el mismo esquema, consulte el apartado Nombres de archivo dinámicos: entrada/salida.

El archivo `CompletePO.mfd`, disponible en la carpeta `...\MapForceExamples`, enseña cómo se pueden unir tres archivos XML en un solo archivo XML de destino.



Observe que los datos de todos los componentes de origen se consolidan en un solo archivo XML de destino: el archivo **CompletePO**.

- El esquema de origen **ShortPO** tiene asociado un archivo XML de instancia y contiene solamente los números de cliente y los datos de los productos (es decir, partida, número y cantidad).
- El esquema de origen **Customers** tiene asociado un archivo XML de instancia y contiene los números de cliente y datos de contacto de los clientes (p. ej. el nombre y la dirección). **Nota:** en este archivo solamente hay un cliente, cuyo número de cliente es 3.
- El esquema de origen **Articles** tiene asociado un archivo XML de instancia y contiene los datos de los productos (es decir, el número, nombre y precio del producto).
- El esquema de destino **CompletePO** no tiene asociado ningún archivo de instancia, porque los datos vienen dados por los tres archivos XML de instancia de origen. La

estructura jerárquica de este esquema permite combinar todos los datos XML de origen y generar un archivo XML de salida válido.

El archivo de esquema debe crearse en un editor XML como XMLSpy, no es generado por MapForce (aunque si hubiera un archivo de instancia CompletePO.xml, MapForce podría crear el esquema).

La estructura de CompletePO es una mezcla de las estructuras de los archivos XML de origen.

El componente de **filtrado Customer** se utiliza para buscar/filtrar los datos de los archivos XML **ShortPO** y **Customers** donde los números de cliente sean idénticos y después pasar los datos asociados al componente de destino **CompletePO**.

- La función **`equal`** compara el elemento **`CustomerNr`** del componente de origen **ShortPO** con el elemento **`Number`** del componente de origen **Customers**.
- Puesto que el archivo de instancia XML de ShortPO solamente contiene un cliente (el cliente número 3), solamente se pueden pasar los datos de cliente y producto del cliente número 3 al componente de filtrado.
- El parámetro **`node/row`** del filtro pasa los datos de **Customer** al parámetro **`on-true`** cuando el parámetro **`bool`** es **`true`**, es decir, cuando se encuentra el mismo número (en este caso, el número de cliente 3).
- Los demás datos de cliente y producto se pasan al esquema de destino a través de dos filtros más.

 © 2014 Altova GmbH

11.12 Línea de comandos: definir parámetros de entrada

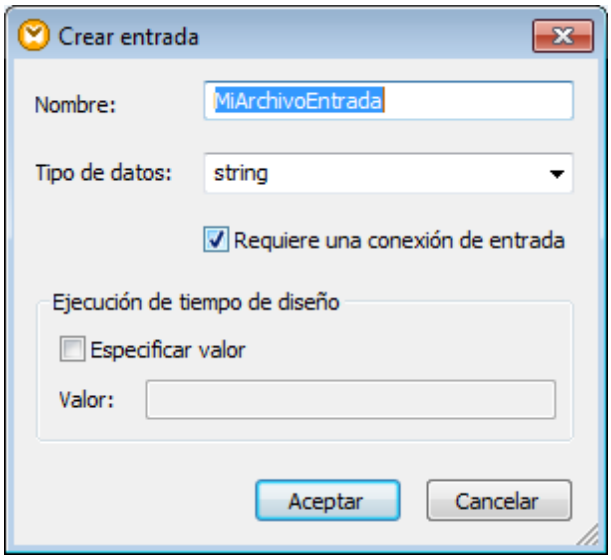
En MapForce puede crear componentes de entrada que funcionen como parámetros en la ejecución de la línea de comandos de una asignación. Estos componentes de entrada sirven para definir el contenido del parámetro cuando la asignación se ejecuta desde la línea de comandos y los valores que debe tomar cuando se obtenga una vista previa de los resultados en MapForce. El contenido del parámetro puede ser un nombre de archivo o cualquier otro valor.

Este tipo concreto de componente de entrada **no se puede usar dentro** de funciones definidas por el usuario. Solamente está disponible en la ventana principal de asignación.

Además de los componentes de entrada, también puede usar como parámetros de la línea de comandos todos aquellos componentes de origen/destino que tengan nombres únicos (consulte el apartado Nombres de los componentes). Los componentes de entrada se pueden usar para pasar nombres de archivo a la asignación, conectándolos con el nodo de entrada **`Archivo`** de los componentes (ver archivo de ejemplo `FileNamesAsParameters.mfd`), pero en muchos casos es más fácil usar directamente los nombres de componente.

Para definir un componente de entrada / parámetro de la línea de comandos:

1. Use la opción de menú **Función | Insertar componente de entrada**.
   Se abre el cuadro de diálogo "Crear entrada".
2. Escriba un nombre para la entrada (p. ej. MiArchivoEntrada) y seleccione el tipo de datos que desea usar (p. ej. string).



3. Para terminar haga clic en **Aceptar**.

   Los demás campos de este cuadro de diálogo sirven para especificar los valores que se deben usar cuando se ejecuta la asignación en el panel **Resultados** (vista previa) y no se usan en la ejecución desde la línea de comandos. Para más información consulte el apartado Parámetros de entrada: configuración predeterminada y vista previa.

Llamar a programas compilados desde la línea de comandos:

Si quiere suministrar nombres de archivo de entrada/salida a la asignación compilada `FileNameAsParameters.mfd`:

1. Genere código Java para la asignación en MapForce, **Archivo | Generar código en |**

Java.

2. Convierta su código Java generado en un archivo JAR ejecutable usando su entorno IDE, p. ej. `miAsignación.jar`.
3. Introduzca la línea de comandos:

   java -jar miAsignación.jar

Comodines en la ejecución de la línea de comandos (Java):
Cuando se llama desde la línea de comandos a un programa Java generado, compruebe que los parámetros comodín están entre comillas.

Uso correcto:

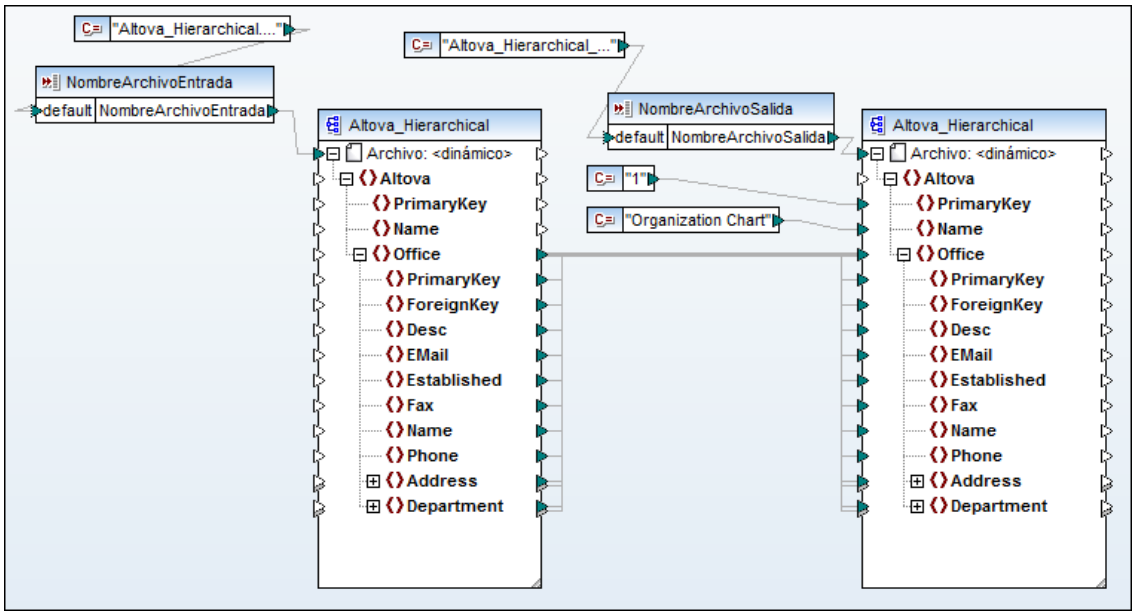
java -jar miAsignación.jar /InputFileName **"altova-*.xml"**

© 2014 Altova GmbH

11.13 Parámetros de entrada: configuración predeterminada y vista previa

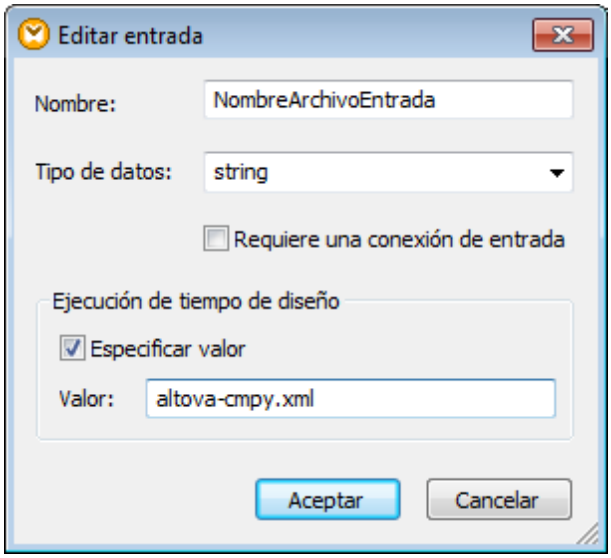
La asignación que aparece en la imagen siguiente (el archivo `FileNamesAsParameters.mfd` guardado en la carpeta `...\MapForceExamples`) usa dos componentes de entrada: uno suministra el nombre del archivo de origen y el otro suministra el del archivo de salida (a los elementos **`Archivo:<dinámico>`** de cada componente).

Los componentes **NombreArchivoEntrada** y **NombreArchivoSalida** son componentes de entrada de la asignación, que se pueden usar como parámetros en la ejecución de la línea de comandos.



Para definir un componente de entrada:

1. Use la opción de menú **Función | Insertar componente de entrada**.
   Se abre el cuadro de diálogo "Crear entrada".
2. Escriba el nombre de la función (p. ej. NombreArchivoEntrada) y seleccione el tipo de datos que desea usar (p. ej. string).



3. Marque la casilla *Especificar valor* e introduzca en el cuadro de texto el valor (o la cadena de texto) que desea usar para obtener la vista previa del resultado p. ej. altova-cmpy.xml.
4. Para terminar haga clic en **Aceptar** y después abra el panel **Resultados** para ver el resultado.
   El valor `altova-cmpy.xml` se usa como archivo de origen de la asignación y los datos asignados aparecen en el panel **Resultados**.

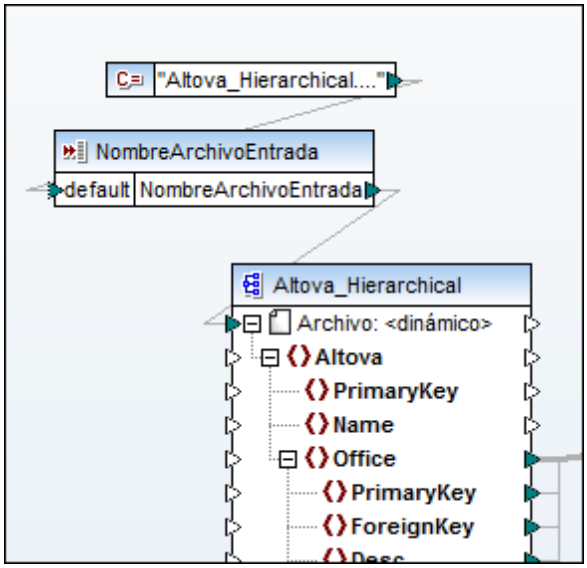
   El valor introducido en el campo *Valor:* del grupo *Ejecución de tiempo de diseño* solamente se aplica a la **vista previa** de resultados en el panel **Resultados**. Este valor no se usa para la generación de código ni para la ejecución de asignaciones desde la línea de comandos.

Para definir un valor predeterminado:

Tras definir el componente de entrada y hacer clic en **Aceptar**, el componente aparece en el área de asignación. Observe que el componente incluye un elemento **`default`**, mientras que el componente toma el nombre del otro elemento asignable (**`NombreArchivoEntrada`**).

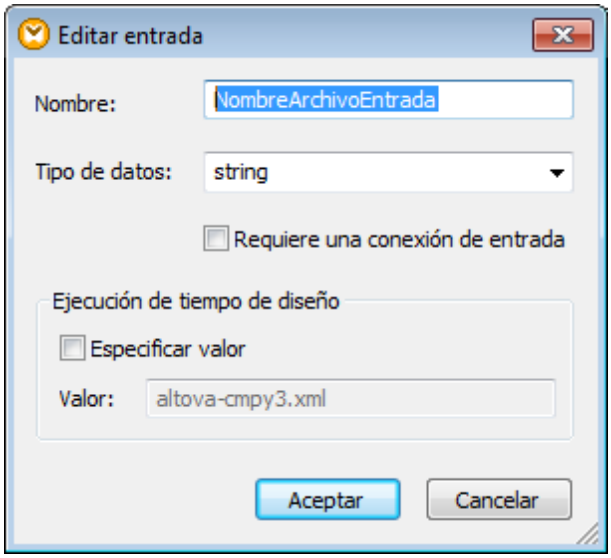
1. Ahora puede usar una **constante** para suministrar el valor predeterminado **`default`**, p. ej `Altova_Hierarchical.xml`.
2. Cree la constante y conéctela al elemento **`default`** del componente de entrada.

 © 2014 Altova GmbH



Para usar el valor predeterminado de un componente de entrada:

1. Haga doble clic en el componente de entrada y **desactive** las casillas *Requiere una conexión de entrada* y *Especificar valor*.



2. Haga clic en **Aceptar** y abre el panel **Resultados** para ver el nuevo resultado de la asignación. Ahora se usan los datos del archivo `Altova_Hierarchical.xml` para la vista previa.

Generar código XSLT 1.0, XSLT 2.0 o XSLT 3.0:
El valor del cuadro de texto *Valor:* se escribe en el archivo por lotes **DoTransform.bat**. Este archivo por lotes se genera automáticamente y se puede ejecutar después con el motor RaptorXML. Si desea usar otro archivo de entrada (o de salida) puede cambiar el nombre en el archivo por lotes.

Si no introduce ningún valor en el cuadro de texto *Valor:*, entonces se usa el valor predeterminado

presente en el código XSLT generado.

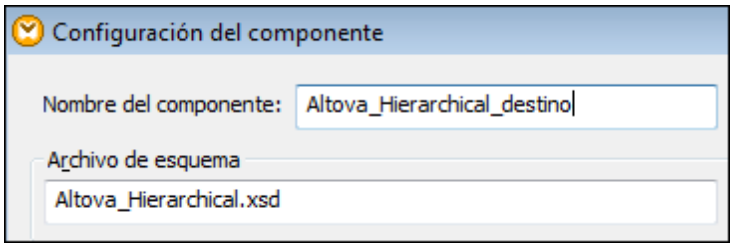
Usar valores/parámetros predeterminados `default` en la ejecución de la línea de comandos

La casilla *Requiere una conexión de entrada* debe estar **desactivada** (antes de generar código) si quiere usar el parámetro predeterminado **`default`** desde la línea de comandos. Introduzca `mapping.exe` para que el código generado use el parámetro predeterminado desde la línea de comandos.

© 2014 Altova GmbH

11.14 Nombres de los componentes

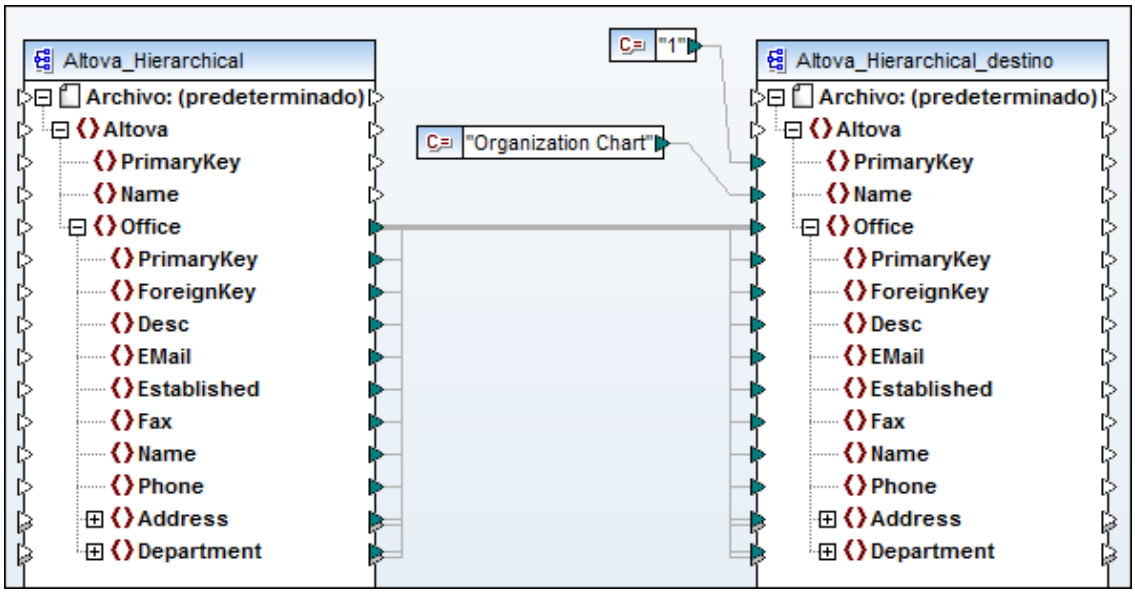
Todos los componentes de MapForce basados en archivos pueden tener un nombre de componente definido por el usuario. El nombre de componente aparece en el primer cuadro de texto del diálogo "Configuración del componente".



El nombre del componente se utiliza en estas situaciones:

- El archivo de entrada/salida utilizado por el componente se puede establecer en tiempo de ejecución en FlowForce, sin necesidad de usar componentes de entrada como se describe en el apartado Línea de comandos: definir parámetros de entrada.
- El nombre del script XSLT/XQuery generado se deriva del nombre del componente.
- El nombre del componente debe ser único si su intención es acceder al componente desde la línea de comandos o desde FlowForce. Para más información consulte el apartado Componente.

La asignación de la imagen siguiente es una versión simplificada del archivo de ejemplo `FileNamesAsParameters.mfd`, disponible en la carpeta `...\MapForceExamples`.



Este ejemplo simplificado no utiliza componentes de entrada distintos para definir archivos de instancia. Estos archivos/parámetros se definirán en la línea de comandos usando nombres de componente únicos:

- El componente de origen se llama **Altova_Hierarchical**

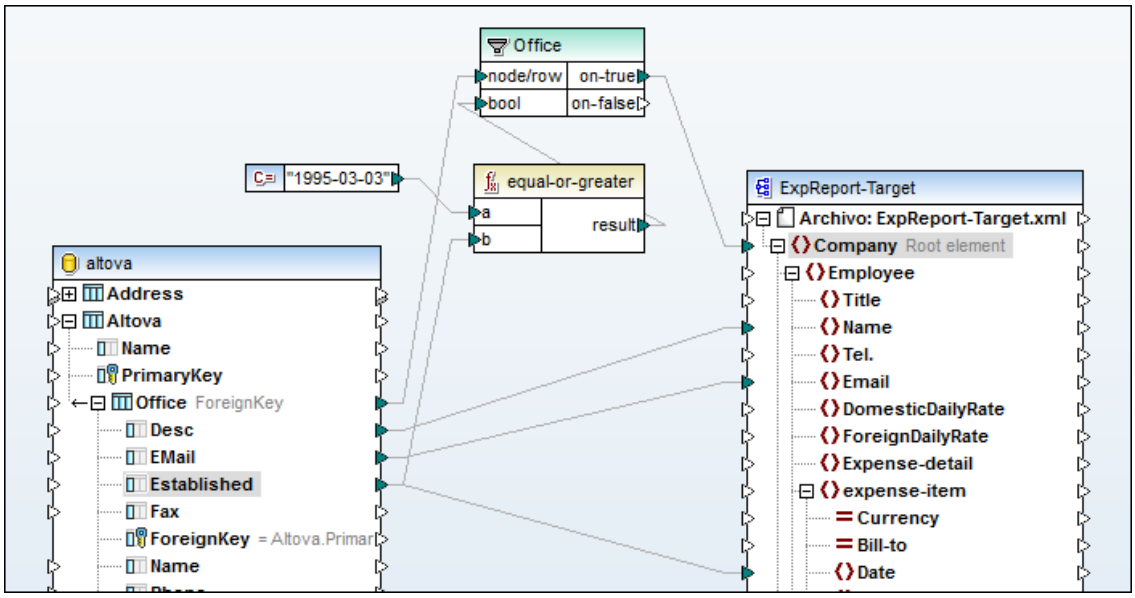
- El componente de destino se llama **Altova_Hierarchical_destino**

© 2014 Altova GmbH

11.15 Filtrar datos de base de datos por fecha

La asignación de la imagen que aparece más abajo enseña a usar el componente de filtrado para filtrar los registros de la BD que coincidan con una fecha determinada.

- El campo de la base de datos **`Established`** se define como un campo **`Date/Time`** en la base de datos.
- La fecha de comparación se introduce en una constante.
- Si el registro de fecha es mayor que **`1995-03-03`**, entonces el componente de filtrado pasa los datos de **`Office`** al archivo de destino. **Nota:** use el tipo de datos *Demás opciones* para el componente de constante.



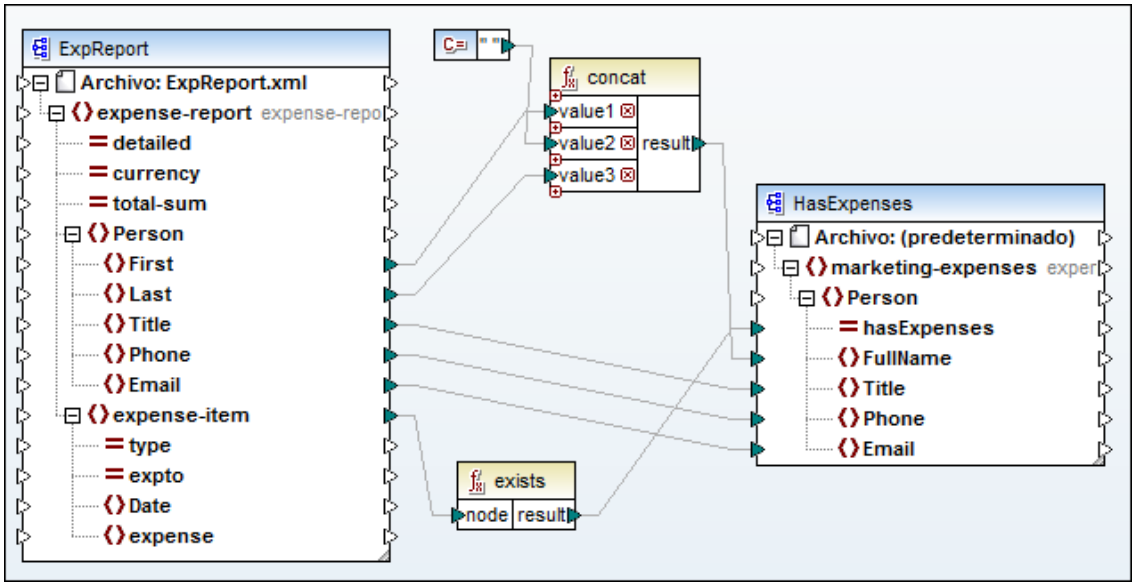
11.16 Comprobación, posición y agrupamiento de nodos

Las funciones de comprobación de nodos sirven para comprobar si existen determinados nodos en los archivos **XML de instancia**. Los elementos/atributos definidos como opcionales en el esquema XML pueden aparecer o no en el archivo XML de instancia. Utilice estas funciones para comprobar si existen determinados nodos y basar el resto del procesamiento de datos en el resultado obtenido.

Función `exists`

Devuelve **`true`** si el nodo existe, de lo contrario devuelve **`false`**.
Por ejemplo, la asignación `HasMarketingExpenses.mfd`, disponible en la carpeta ...\MapForceExamples contiene una función **`exists`**.

Si existe un elemento **`expense-item`** en el archivo XML, entonces al atributo **`hasExpenses`** se le da el valor **`true`** en el archivo/esquema XML de destino.



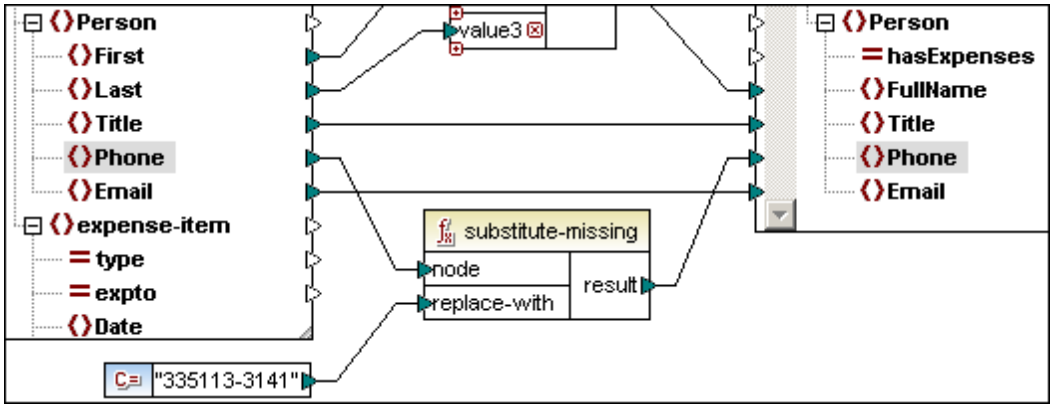
Función `not-exists`

Devuelve **`false`** si el nodo existe, de lo contrario devuelve **`true`**. Para más información consulte el apartado Asignar nodos que faltan con not-exists, que explica cómo asignar nodos que faltan.

Función `substitute-missing`

Esta función es una mezcla de la función **`exists`** y la condición **`if-else`**. Sirve para asignar el contenido actual del campo si existe el nodo en el archivo XML de origen. De lo contrario, utiliza el elemento asignado al parámetro **`replace-with`**.

 © 2014 Altova GmbH



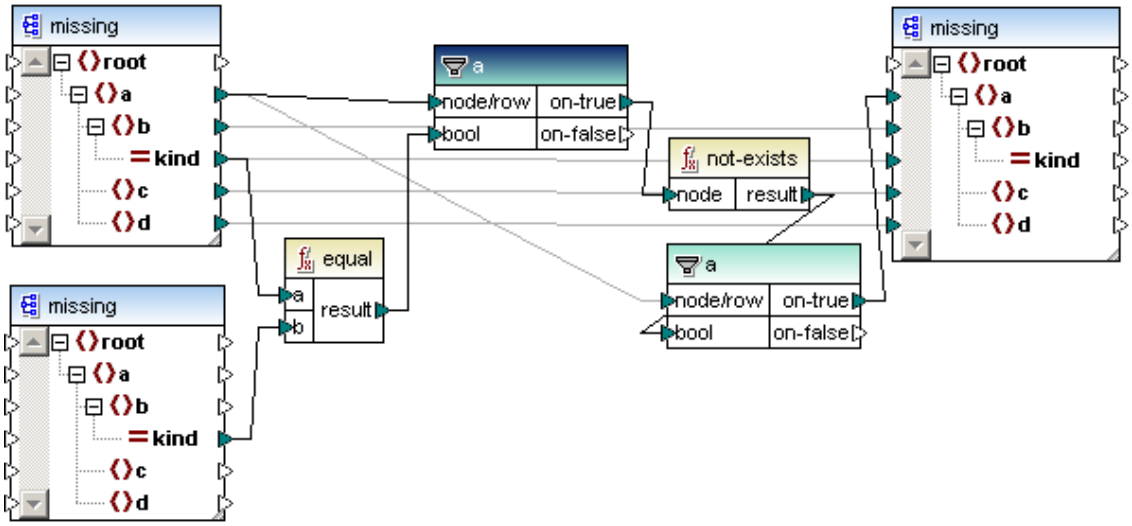
En el ejemplo de la imagen anterior, se comprueba si existe el nodo **Phone** en el archivo XML de instancia. Si el nodo no existe, se asigna el valor suministrado por la constante.

11.16.1 Asignar nodos que faltan con not-exists

El ejemplo que aparece a continuación explica cómo se puede usar la función **`not-exists`** para asignar nodos que no existen en uno de los dos archivos de origen.

La asignación del ejemplo siguiente:

- 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 sus contenidos al archivo de destino



A continuación aparecen los dos archivos XML de instancia. Las diferencias entre los dos archivos son:

- El archivo de la izquierda, **a.xml**, contiene el nodo **`<b kind="3">`**, que no está presente en el archivo **b.xml**.
- El archivo de la derecha, **b.xml**, contiene el nodo **`<b kind="4">`**, que no está presente en el archivo **a.xml**.

a.xml

```
<root xmlns:xsi="http://www.w3.
    <a>
        <b kind="1">b1</b>
        <c>c1</c>
        <d>d1</d>
    </a>
    <a>
        <b kind="2">b2</b>
        <c>c2</c>
        <d>d2</d>
    </a>
    <a>
        <b kind="3">b3</b>
        <c>c3</c>
        <d>d3</d>
    </a>
</root>
```

b.xml

```
<root xmlns:xsi="http://www.w3.
    <a>
        <b kind="1">foo</b>
        <c>foo</c>
        <d>foo</d>
    </a>
    <a>
        <b kind="2">foo</b>
        <c>foo</c>
        <d>foo</d>
    </a>
    <a>
        <b kind="4">foo</b>
        <c>foo</c>
        <d>foo</d>
    </a>
</root>
```

- La función **`equal`** compara los atributos **`kind`** de los archivos XML y pasa el resultado a un componente de filtrado.
- La función **`not-exists`** se coloca después del primer componente de filtrado y selecciona los nodos que faltan en cada archivo de origen.
- El segundo componente de filtrado pasa el nodo que falta y los demás datos desde el archivo **a.xml solamente** hasta el destino.

  Como resultado el nodo que falta en el archivo **b.xml** (**`<b kind="3">`**) se pasa al componente de destino.

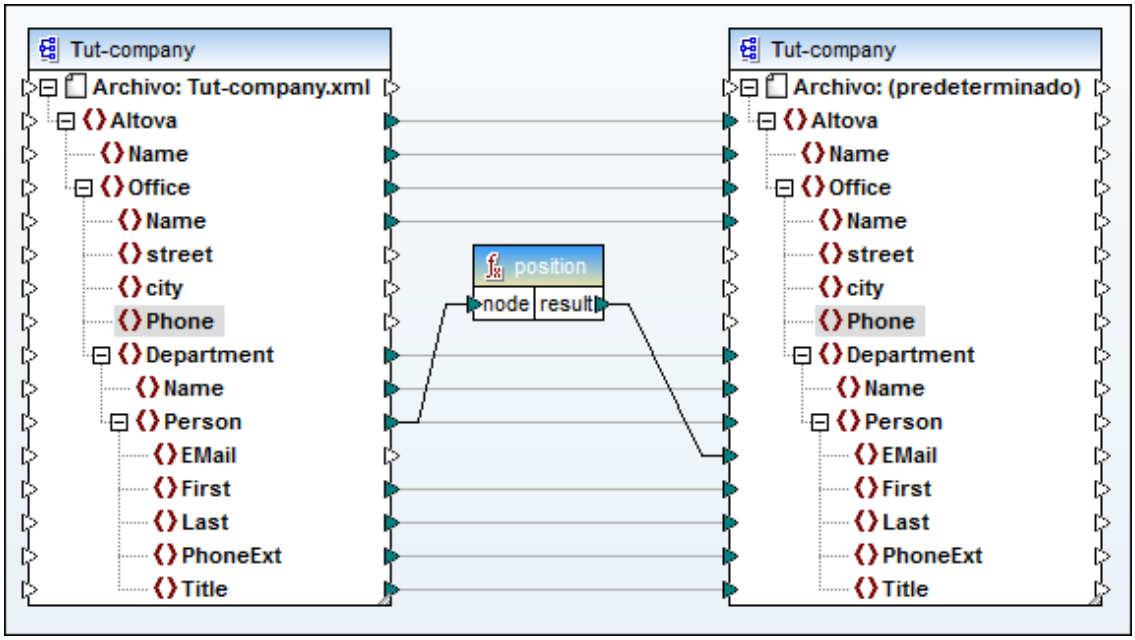
```
<root xsi:noNamespaceSchemaLocation="C:/DOCUME
    <a>
        <b kind="3">b3</b>
        <c>c3</c>
        <d>d3</d>
    </a>
</root>
```

11.16.2 Posición de los elementos de contexto en una secuencia

La función **position** sirve para determinar la posición de un nodo concreto dentro de una secuencia o para usar una posición determinada para filtrar los elementos.

El elemento de contexto viene definido por el elemento que está conectado al parámetro **node** de la función **position** (p. ej. **Person**, como muestra la asignación de la imagen).

Esta sencilla asignación añade un número de posición a cada elemento **Person** de cada elemento **Department**.



La secuencia se reinicia en 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>
```

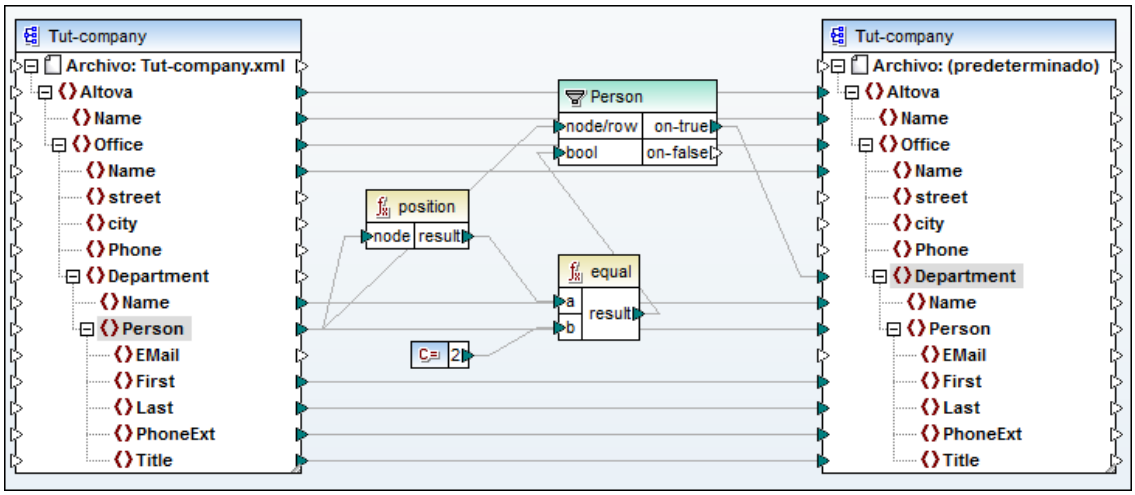
Filtrar determinados nodos por medio de la función position

Puede usar la función **position** junto con un filtro para asignar solamente los nodos que tengan

 © 2014 Altova GmbH

cierta posición en la secuencia del componente de origen.

Para filtrar una posición concreta de la secuencia, el parámetro **node/row** del filtro y el parámetro **node** de la función **position** deben estar conectados al mismo elemento del componente de origen.



La asignación de la imagen anterior da como resultado:

- el **segundo** elemento **Person** de cada elemento **Department**
- de cada elemento **Office** de Altova.

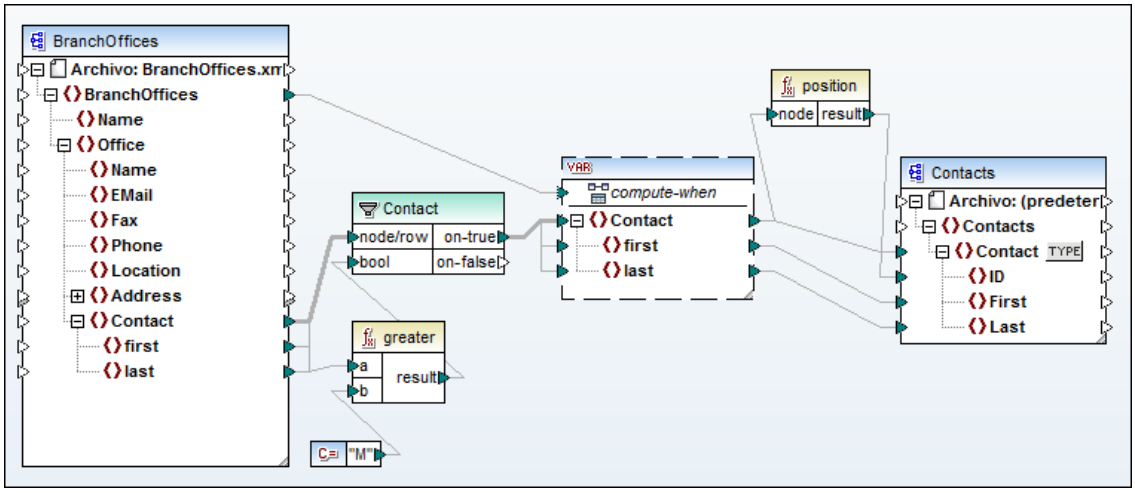
```
<Office>
  <Name>Microtech, Inc.</Name>
  <Department>
    <Name>Admin</Name>
    <Person>
      <EMail>b.bander@microtech.com</EMail>
      <First>Bert</First>
      <Last>Bander</Last>
      <Title>Accounts Receivable</Title>
    </Person>
  </Department>
  <Department>
    <Name>Sales and Marketing</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>
```

Averiguar la posición de un elemento en una secuencia filtrada

El componente de filtrado no es una función de secuencia y, por tanto, no se puede usar directamente con la función **position** para averiguar la posición de los elementos filtrados. Para esto es necesario usar una Variable.

El resultado de una variable siempre es una secuencia, es decir, una lista delimitada de valores, que también se puede usar para crear secuencias.

- La variable recopila los datos filtrados de **`Contact`** en los que el elemento **`Last`** empieza por una letra superior a la letra **`M`** (dada por la constante).
- Los datos de contacto se pasan (desde la variable) al componente de destino
- La función **`position`** numera estos datos de contacto de manera secuencial.



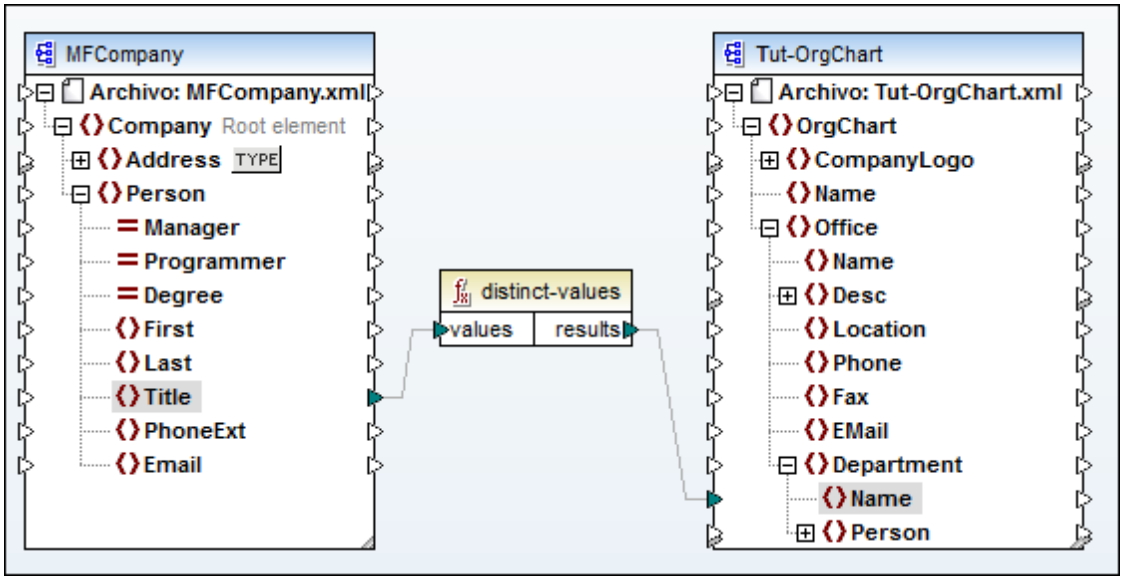
 © 2014 Altova GmbH

11.16.3 Agrupamiento de nodos / contenido de nodos

MapForce ofrece funciones para agrupar nodos y su contenido. Estas funciones están en la categoría de funciones **sequence** de la ventana Bibliotecas.

Función `distinct-values`
Sirve para quitar valores duplicados de un conjunto de resultados y asignar los elementos únicos al componente de destino.

En el ejemplo siguiente se examina el contenido de los elementos **`Title`** del componente de origen y cada elemento **`Title`** que tenga un valor único se asigna al elemento **`Department / Name`** del componente de destino.

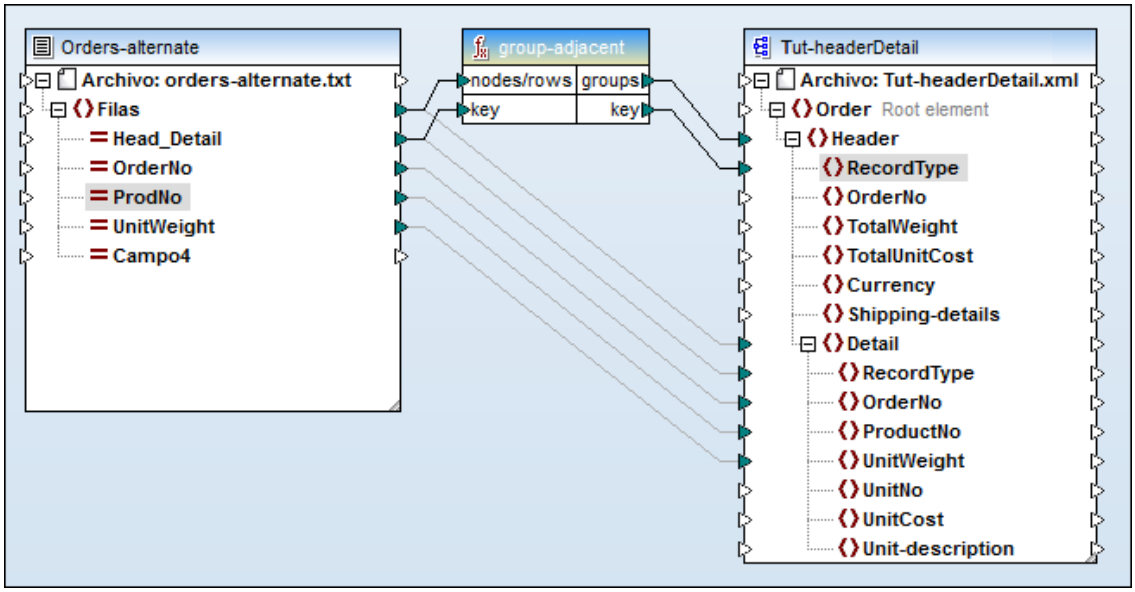


Observe que la secuencia que los elementos **`Title`** tenían en el componente de origen se conserva en el componente de destino.

```
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; Marketing Manager US</Name>
17     </Department>
18   </Office>
19 </OrgChart>
```

Función `group-adjacent`

Agrupa la secuencia de entrada en una serie de grupos, donde cada conjunto de elementos/nodos idénticamente adyacentes se colocan en un grupo nuevo distinto.



Si tenemos el archivo CSV que aparece a continuación, lo que queremos es que todos los registros **`Header`** y **`Detail`** aparezcan en grupos diferentes.

Head_Detail	OrderNo	ProdNo	UnitWeight	Campo4	Campo6	Campo7	Campo8	Campo9
string	string	string	string	string	string	string	string	string
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		

- Se crea un grupo nuevo con el primer elemento, en este caso H.
- Puesto que el elemento (o clave) siguiente de la secuencia es diferente (en este caso, D) se crea otro grupo llamado D.
- Los dos elementos D siguientes se añaden al mismo grupo D, porque son del mismo tipo.
- Con el elemento H se crea otro grupo H.
- Y después se crea otro grupo con dos elementos D.

 © 2014 Altova GmbH

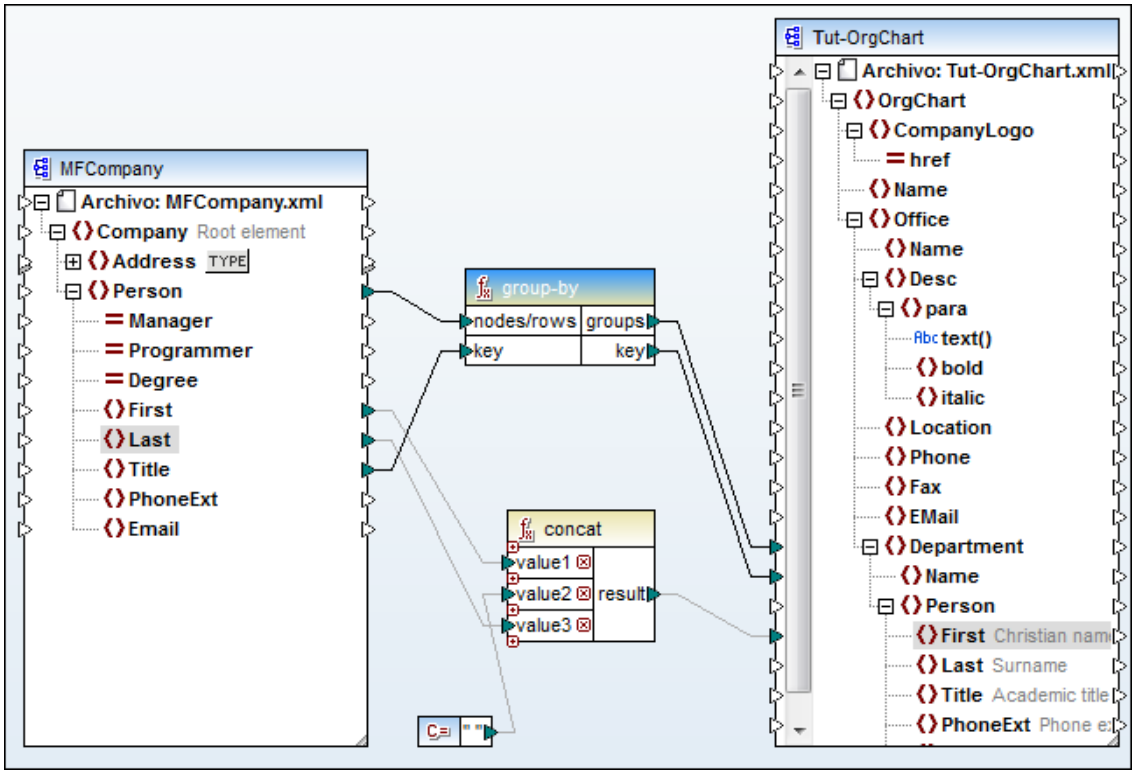
```
3   <Header>
4     <RecordType>H</RecordType>
5     <Detail>
6       <RecordType>H</RecordType>
7       <OrderNo>111</OrderNo>
8       <ProductNo>G-889-11</ProductNo>
9       <UnitWeight>9</UnitWeight>
10    </Detail>
11  </Header>
12  <Header>
13    <RecordType>D</RecordType>
14    <Detail>
15      <RecordType>D</RecordType>
16      <OrderNo>111</OrderNo>
17      <ProductNo>B-192-409</ProductNo>
18      <UnitWeight>1</UnitWeight>
19    </Detail>
20    <Detail>
21      <RecordType>D</RecordType>
22      <OrderNo>111</OrderNo>
23      <ProductNo>F-152-427</ProductNo>
24      <UnitWeight>3</UnitWeight>
25    </Detail>
26    <Detail>
27      <RecordType>D</RecordType>
28      <OrderNo>111</OrderNo>
29      <ProductNo>Q-821-207</ProductNo>
30      <UnitWeight>7</UnitWeight>
31    </Detail>
32  </Header>
```

Observe que la función **`group-adjacent`** utiliza el contenido del nodo/elemento como clave de agrupación. El **contenido** del campo **`Head_Detail`** se utiliza para agrupar los registros en el destino según el tipo de registro.

Función `group-by`

Agrupa la secuencia de entrada según distintas claves y genera la serie de grupos junto con sus claves. El ejemplo de la imagen siguiente muestra cómo funciona:

- La clave (**`key`**) que define los diferentes grupos del componente de origen es el elemento **`Title`**. Este elemento se utiliza para agrupar a las personas de la compañía.
- El nombre del grupo se coloca en el elemento **`Department / Name`** del componente de destino, mientras que los elementos **`First`** y **`Last`** se concatenan y se colocan en el elemento secundario **`Person / First`**.



Observe que la función **`group-by`** utiliza el **contenido** del elemento/nodo 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, fíjese en el **filtro implícito** de las filas existente entre el documento de origen y el documento de destino que puede verse en el ejemplo. En el documento de destino, cada elemento **`Department`** tiene solamente los elementos **`Person`** que **coinciden** con la clave de agrupación (**`key`**) porque la función **`group-by`** crea la jerarquía necesaria de forma instantánea.

Si tiene una jerarquía plana (CSV, FLF, etc.) con un nombre de archivo de salida dinámico, creada en parte con el valor de **`key`**, el filtro implícito sigue existiendo. Esto significa que quizás no haga falta conectar el resultado de los grupos al componente de destino.

En el panel **Resultados** podemos ver el resultado del agrupamiento de nodos.

 © 2014 Altova GmbH

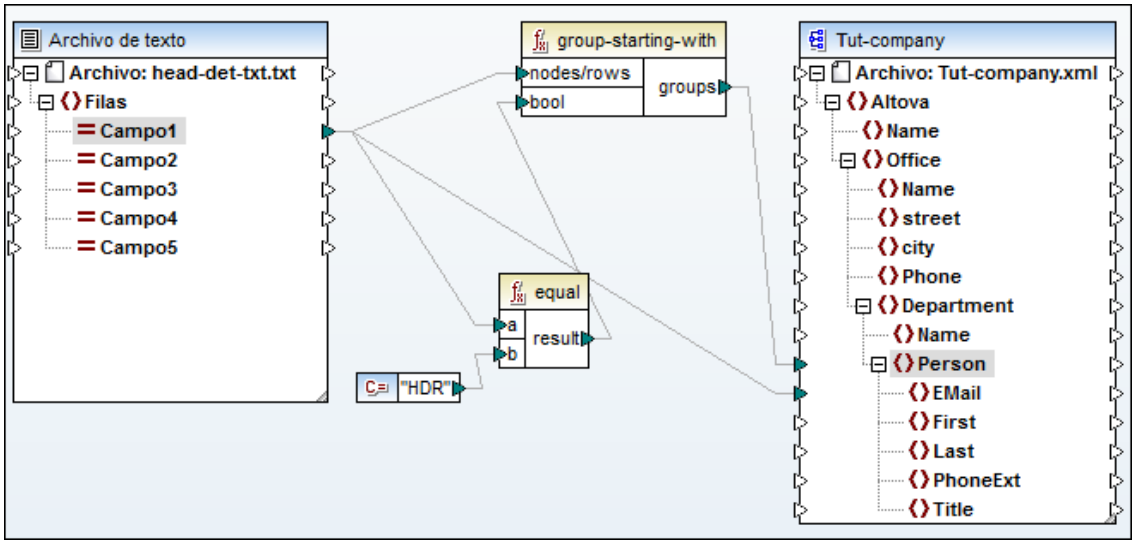
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <OrgChart xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart C:/DOCU
3    <Office>
4      <Department>
5        <Name>Office Manager</Name>
6        <Person>
7          <First>Vernon Callaby</First>
8          <First>Steve Meier</First>
9        </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>
```

Función `group-starting-with`

Agrupa la secuencia de entrada según el elemento suministrado, si este existe en los datos de origen. Para probar los datos de entrada se necesita una función booleana.

Campo1	Campo2	Campo3	Campo4	Campo5
string	string	string	string	string
HDR	custid0001	ordid	0001	EUR
DTL	itemABC	qty0100	price	0001.100
TXT	please	deliver	ASAP	
DTL	itemxx2	qty0001	price	0010.000
DTL	itemDDD	qty0010	price	0010.500
HDR	custid0002	ordid	0001	EUR
DTL	itemABC	qty0100	price	0001.100
HDR	custid0003	ordid	0002	USD
DTL	itemDEF	qty0003 price	200.000	

Esta función agrupa los elementos en base al **primer** elemento de un grupo, en este caso HDR.



El valor de los nodos/elementos no necesita ser idéntico y ni siquiera hace falta que exista. El nodo **`pattern`**, es decir, el nombre de los nodos/elementos, deben ser idéntico. De lo contrario

no se genera el grupo.

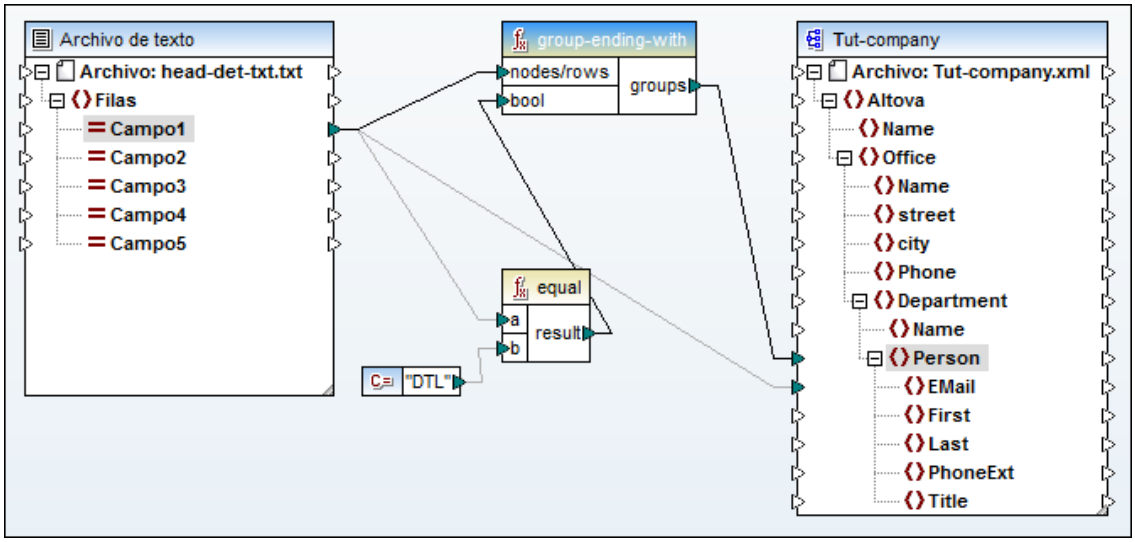
```
3   <Office>
4     <Department>
5       <Person>
6         <EMail>HDR</EMail>
7         <EMail>DTL</EMail>
8         <EMail>TXT</EMail>
9         <EMail>DTL</EMail>
10        <EMail>DTL</EMail>
11      </Person>
12      <Person>
13        <EMail>HDR</EMail>
14        <EMail>DTL</EMail>
15      </Person>
16      <Person>
17        <EMail>HDR</EMail>
18        <EMail>DTL</EMail>
19      </Person>
```

Este resultado muestra cómo se creó un grupo nuevo para cada elemento **HDR**.

Función `group-ending-with`

Esta función complementa a la función **`group-starting-with`** y termina cada grupo de la secuencia de entrada en base al elemento suministrado, si este existe en los datos de origen. Para probar los datos de entrada se necesita una función booleana.

Si usamos el mismo componente de origen que en el ejemplo anterior, este ejemplo muestra el resultado que se obtiene al usar DTL como elemento de la función **`group-ending-with`**.



En este caso el valor de los elementos/nodos no necesita ser idéntico y ni siquiera hace falta que exista. El nodo **`pattern`**, es decir, el nombre de los nodos/elementos, deben ser idéntico. De lo contrario no se genera el grupo.

 © 2014 Altova GmbH

```
<Office>
  <Department>
    <Person>
      <EMail>HDR</EMail>
      <EMail>DTL</EMail>
    </Person>
    <Person>
      <EMail>TXT</EMail>
      <EMail>DTL</EMail>
    </Person>
    <Person>
      <EMail>DTL</EMail>
    </Person>
    <Person>
      <EMail>HDR</EMail>
      <EMail>DTL</EMail>
    </Person>
    <Person>
      <EMail>HDR</EMail>
      <EMail>DTL</EMail>
```

Este resultado muestra cómo se creó un grupo nuevo cada vez que el elemento **DTL** puede ser el último elemento.

Función `set-empty`

Permite generar una secuencia vacía para el nodo especificado. Cuando se conecta a un nodo primario, todos los nodos secundarios se establecen como vacíos. En realidad, lo que hace es eliminar el nodo del componente de destino.

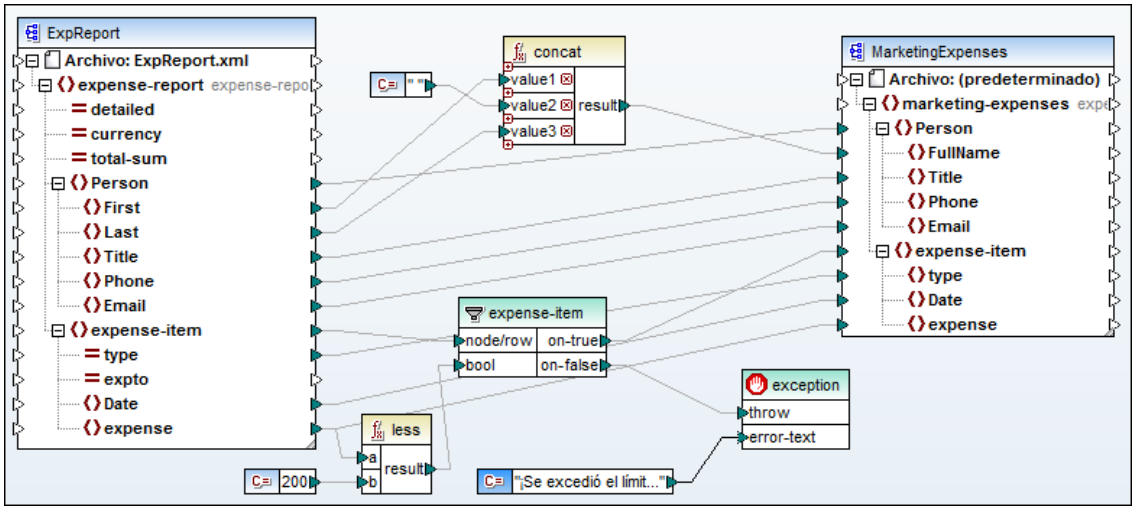
P. ej. puede invalidar los valores predeterminados de un documento XBRL que se definieron más arriba en el componente/en la taxonomía XBRL.

11.17 Excepciones

En MapForce puede definir excepciones, es decir, condiciones que inicien un mensaje de error. Cuando se satisface la condición, aparece un mensaje definido por el usuario y el proceso de asignación de datos se detiene. Un ejemplo es el archivo `ExpenseLimit.mfd`, disponible en la carpeta `...\MapForceExamples`, que contiene una función de excepción.

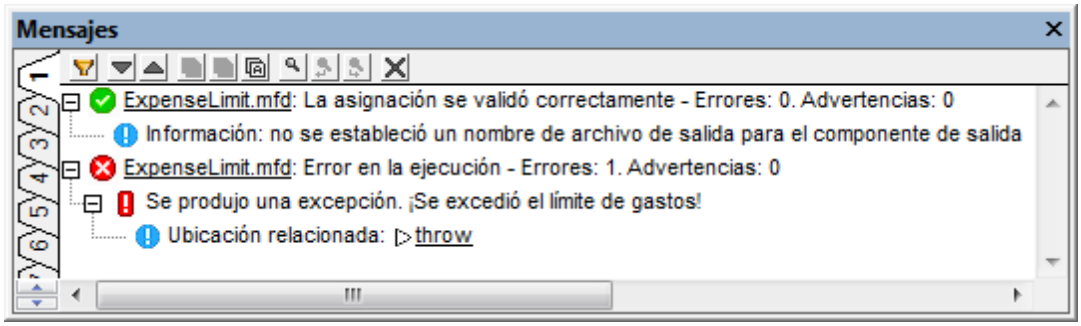
Para insertar un componente de excepción:

- Seleccione la opción de menú **Insertar | Excepción** o haga clic en el icono **Excepción** de la barra de herramientas.



En la asignación de la imagen anterior puede ver cómo se definen las excepciones. La excepción debe desencadenarse cuando se excede el límite de gastos, es decir, cuando el gasto tiene un importe superior a 200.

- La función **`less`** comprueba si el gasto es inferior a 200 y su resultado booleano se pasa al componente de **filtrado**.
- Si el resultado booleano de la función es **`false`** (es decir, si el gasto es igual o mayor que 200), el parámetro **`on-false`** del filtro activa la excepción y el proceso de asignación se detiene. (No olvide que también puede conectar la excepción al parámetro **`on-true`**.)
- El texto del error (*¡Se excedió el límite de gastos!*), suministrado por una **constante**, se asigna al parámetro **`error-text`** de la función de excepción.
- El texto del error aparece en el panel **Resultados** y cuando se ejecuta el código compilado.



 © 2014 Altova GmbH

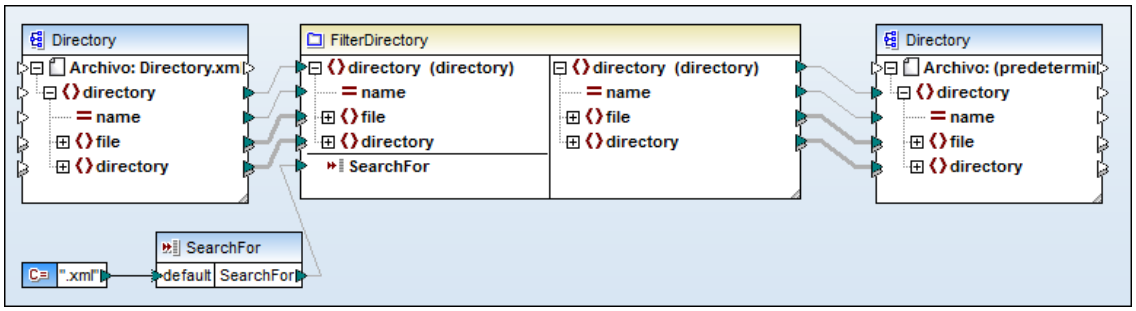
Nota:
Es importante fijarse en la posición del filtro en el ejemplo anterior:

- **Ambos parámetros del filtro** (`on-true` y `on-false`) deben tener conexiones. Uno de ellos debe estar conectado a la función de excepción y el otro al componente de destino que recibe los datos de origen filtrados. De lo contrario, la excepción no se puede iniciar.
- La **excepción** y el componente de **destino** deben estar conectados directamente con el **filtro**. Las funciones (o demás componentes) no pueden ponerse entre el filtro y la excepción ni entre el filtro y los componentes de destino.
- Al generar código **XSLT 2.0** y **XQuery**, MapForce emite el mensaje de error *Error en la ejecución* en la ventana Mensajes y se abre el panel correspondiente (XSLT2 o XQuery), donde aparece resaltada la línea que contiene el error.

11.18 Asignación recursiva definida por el usuario

Esta sección describe cómo se creó la asignación `RecursiveDirectoryFilter.mfd`, disponible en la carpeta `...\MapForceExamples`, y cómo se diseñan las asignaciones recursivas. La carpeta de proyecto MapForceExamples contiene más ejemplos de asignaciones recursivas.

La imagen siguiente muestra la asignación con la función recursiva definida por el usuario **`FilterDirectory`**, cuyo objetivo es filtrar una lista de archivos .xml presente en el archivo de origen.

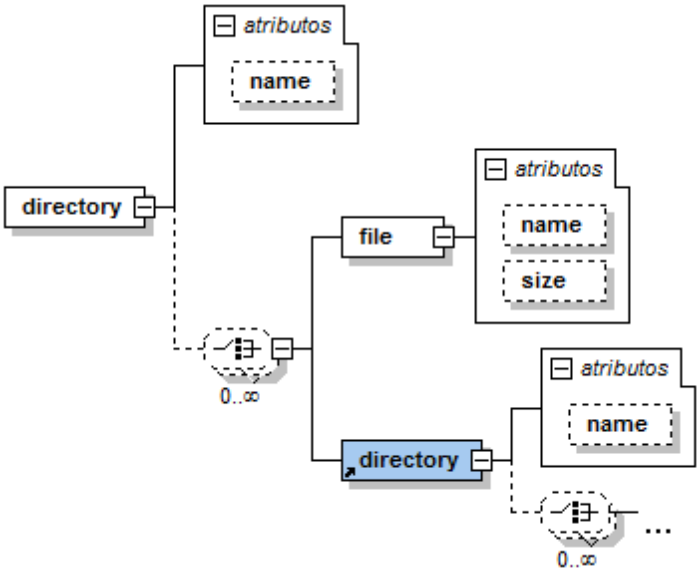


El **archivo de origen** que contiene los datos sobre archivos y directorios para esta asignación es el archivo `Directory.xml`. Este archivo XML suministra los datos sobre archivos y directorios en forma jerárquica (*imagen siguiente*).

```
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 `Directory.xml` tiene un elemento **recursivo** llamado **`directory`**, que permite que bajo el elemento **`directory`** aparezca un número ilimitado de subdirectorios.

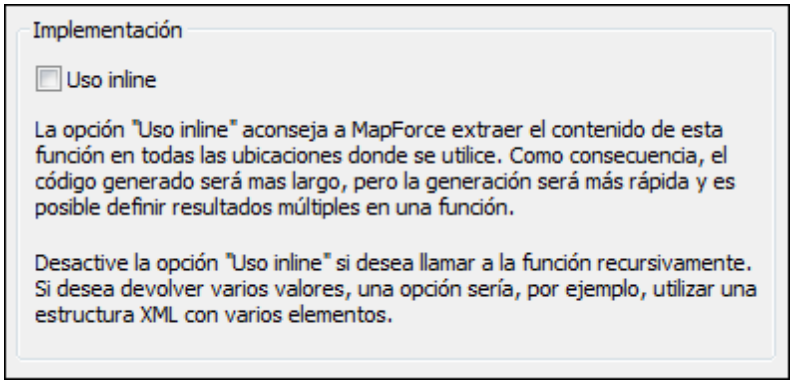
 © 2014 Altova GmbH


atributos
name
directory
file
atributos
name
size
0..∞
atributos
name
directory
0..∞
...

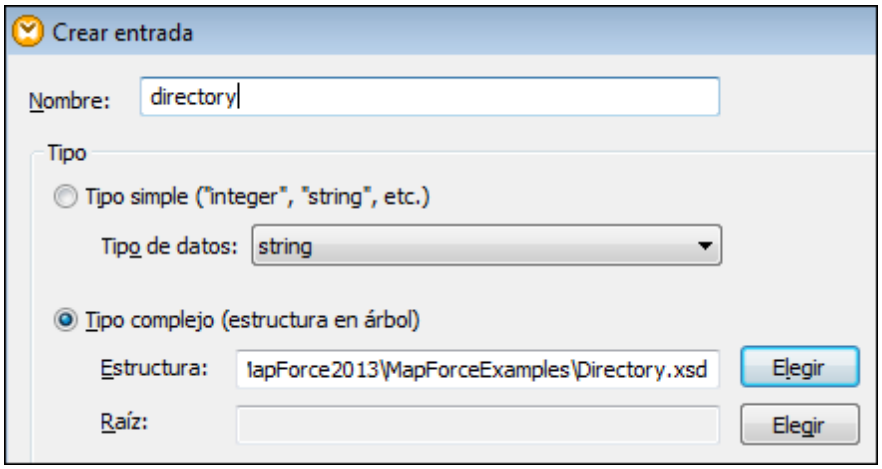
11.18.1 Definir una función recursiva definida por el usuario

Desde la ventana principal de asignación:

1. Seleccione la opción de menú **Función | Crear una función definida por el usuario** para empezar a diseñar la función. Escriba el nombre de la función en el cuadro de diálogo que aparece, p. ej. **`FilterDirectory`**.
2. **Desactive** la casilla *Uso inline* del grupo de opciones *Implementación* para que la función sea recursiva y haga clic en **Aceptar**.

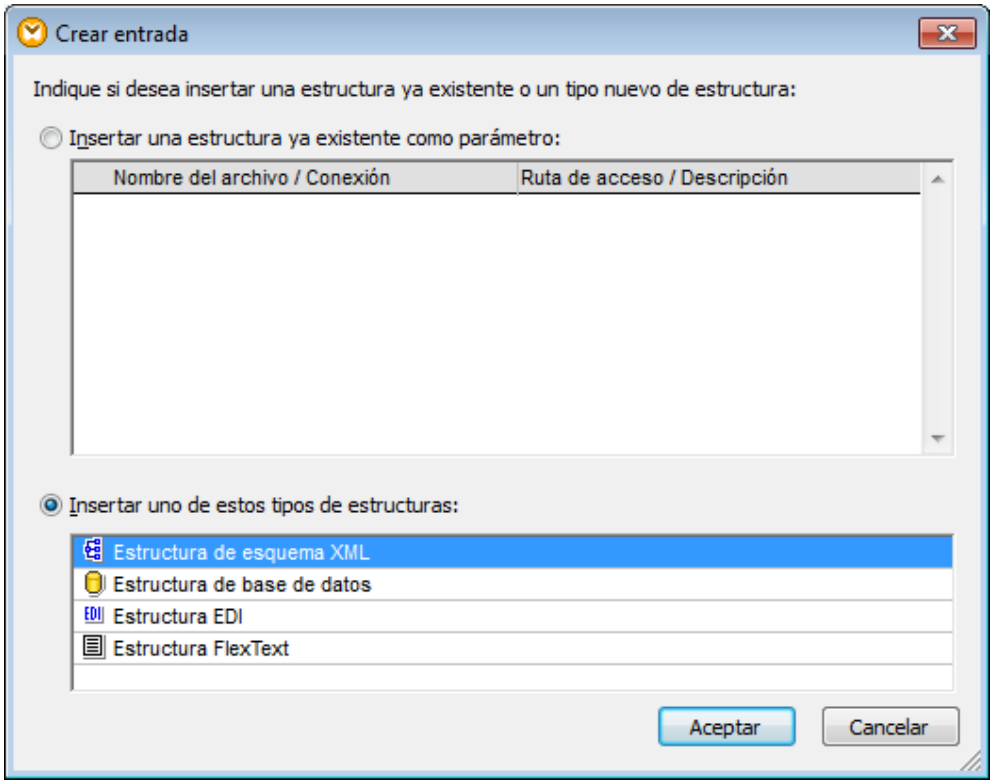


   Ahora está en el panel **FilterDirectory**, donde puede crear la función definida por el usuario.
3. Seleccione la opción de menú **Función | Insertar componente de entrada**.
4. En el cuadro de diálogo que aparece escriba el nombre del componente de entrada (p. ej. **`directory`**) y haga clic en el botón de opción *Tipo complejo (estructura en árbol)*.

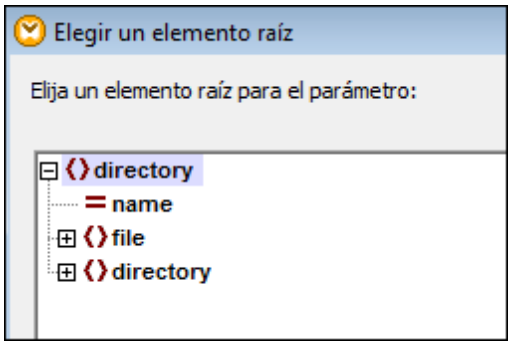


5. Haga clic en el botón **Elegir**. En el nuevo cuadro de diálogo haga clic en la entrada `Estructura de esquema XML` (en el panel inferior) y confirme la selección con **Aceptar**.

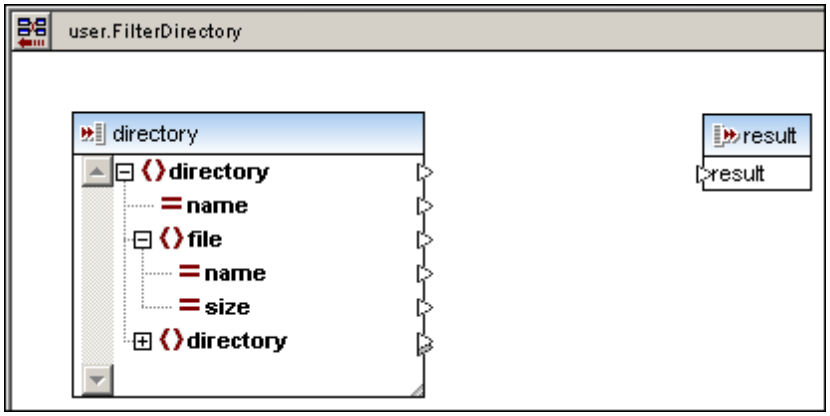
© 2014 Altova GmbH



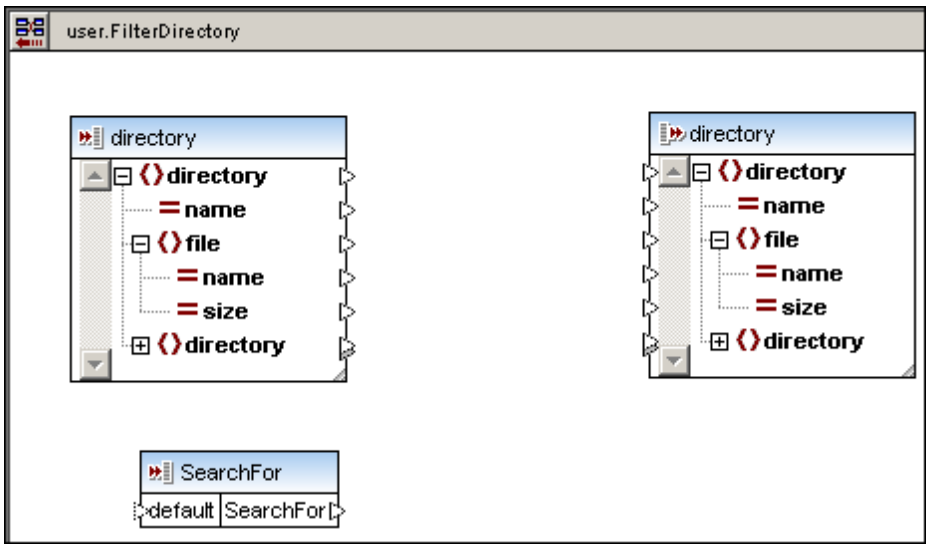
6. Seleccione el archivo `Directory.xsd` de la carpeta `...\MapForceExamples` y haga clic en **Abrir**.
7. Aparece el cuadro de diálogo "Elegir un elemento raíz". Seleccione la entrada `directory` (*imagen siguiente*) y haga clic en **Aceptar**.



8. Haga otra vez clic en **Aceptar** para insertar el parámetro de entrada complejo.
La función definida por el usuario tiene este aspecto en este momento.



9. Elimine el componente de salida simple **result** porque en su lugar vamos a usar un componente de salida complejo.
10. Seleccione la opción de menú **Función | Insertar componente de salida...** y siga los mismos pasos que con el componente de entrada (pasos 4-8).
    Ahora tiene dos componentes complejos en la función definida por el usuario, uno de entrada y otro de salida.
11. Seleccione la opción de menú **Función | Insertar componente de entrada...** e inserte un componente de tipo simple llamado **SearchFor**, por ejemplo. Desactive la casilla *Requiere una conexión* de entrada.

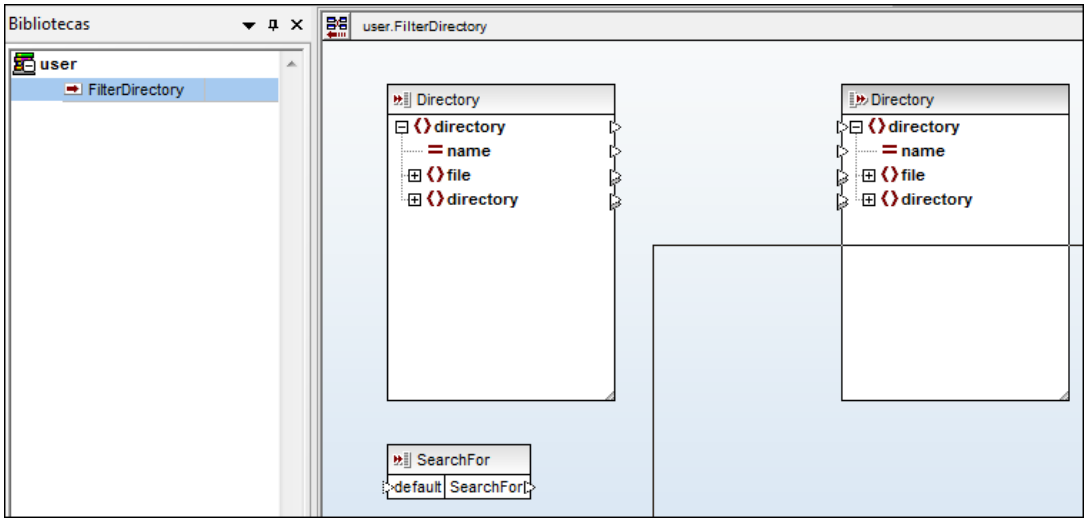


Insertar la función recursiva definida por el usuario

Ya están definidos todos los componentes de entrada y salida para la función definida por el usuario. Ahora sólo falta insertar la función *inacabada* en el panel de diseño de la función definida por el usuario.

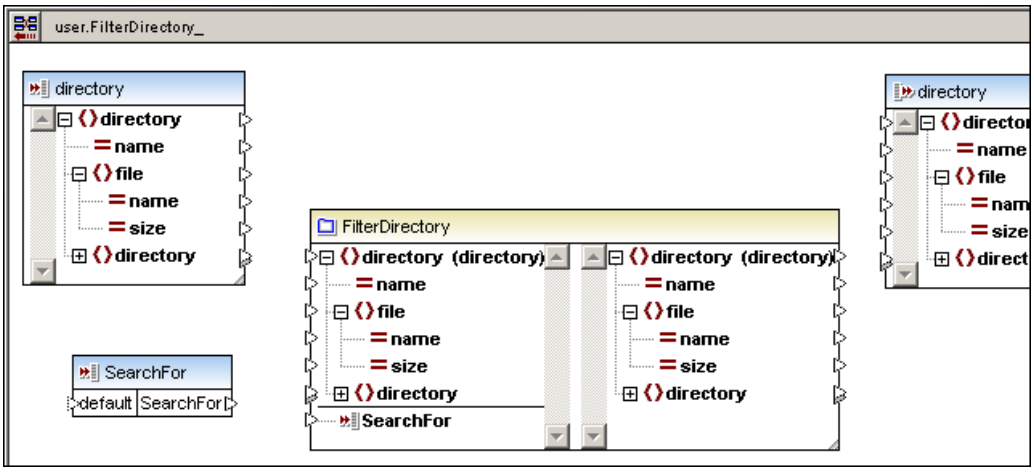
1. Busque la función **`FilterDirectory`** en la sección **`user`** de la ventana Bibliotecas.
2. Haga clic en **`FilterDirectory`** y arrastre la función hasta el panel de diseño **FilterDirectory** donde trabajó hasta ahora.

 © 2014 Altova GmbH

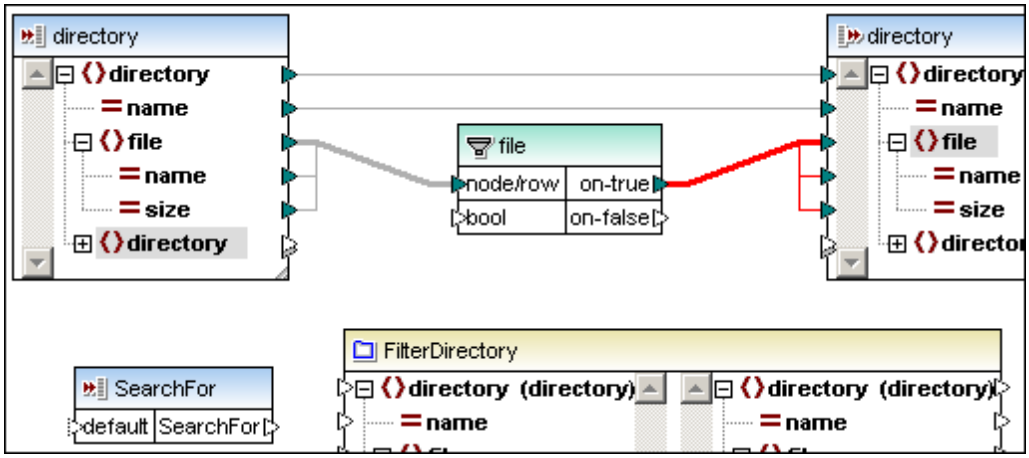


Con Java seleccionado

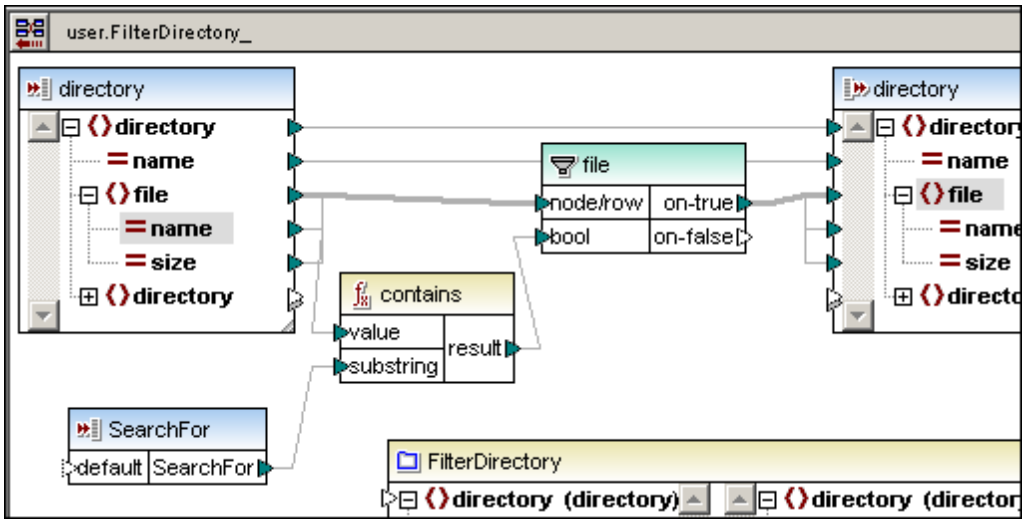
La función definida por el usuario aparece como componente recursivo en la ventana de la función definida por el usuario.



3. Conecte los elementos **`directory`**, **`name`** y **`file`** del componente de entrada con los elementos del mismo nombre del componente de salida.
4. Haga clic con el botón secundario en el conector de los elementos **`file`** y seleccione **Insertar filtro: nodos/filas**.
5. Ahora haga clic con el botón secundario en el conector del parámetro **`on-true`** y seleccione **Copia total** en el menú contextual.
   Los conectores se convierten en conectores de copia total.



6. Inserte la función **`contains`** de la sección **core | string functions** de la ventana Bibliotecas.
7. Conecte el elemento **`name`** con el parámetro **`value`** de la función **`contains`**. Conecte también el elemento **`SearchFor`** con el parámetro **`substring`** de la función **`contains`**. Por último, conecte el parámetro de salida **`result`** de la función **`contains`** con el parámetro **`bool`** del **filtro**.



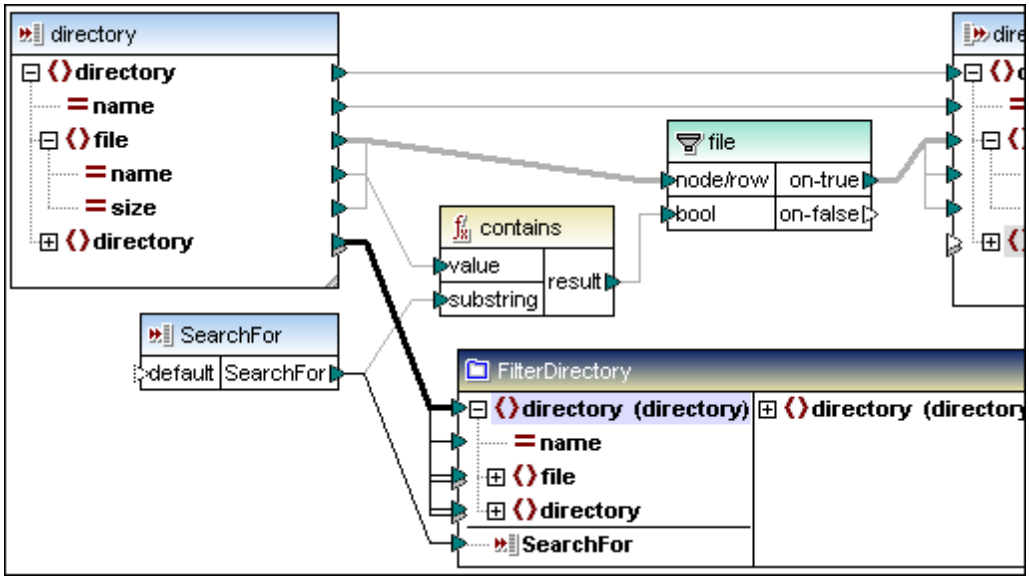
8. Para terminar conecte el elemento **`SearchFor`** del componente de entrada con el elemento **`SearchFor`** de la función definida por el usuario.

Definir la recursión

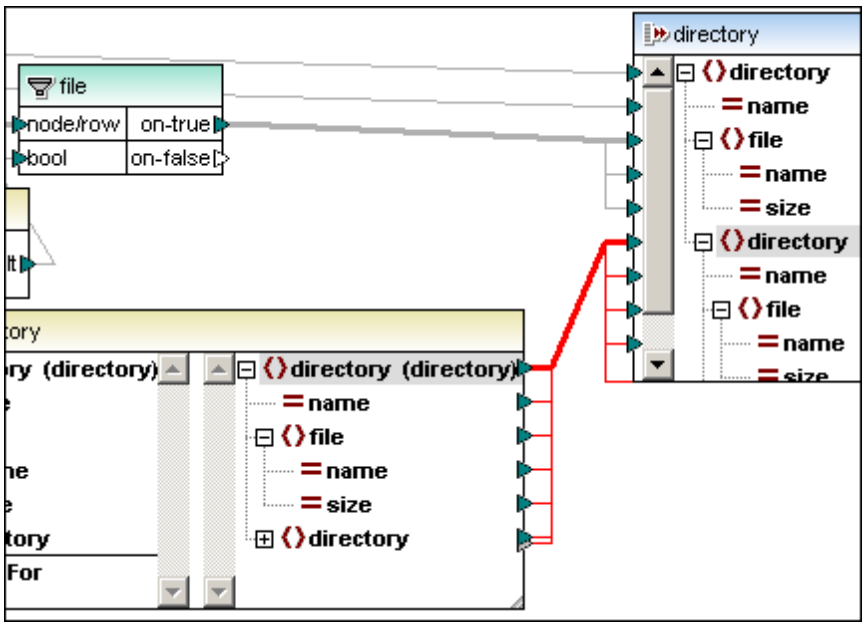
Ya hemos definido un nivel de recursividad del directorio. Ahora tenemos que definir cómo se procesa un subdirectorio.

Para empezar, compruebe que el icono **Conexión automática de secundarios** está activo en la barra de herramientas:

1. Conecte el elemento **`directory`** situado al final de la jerarquía del componente de entrada con el primer elemento **`directory`** de entrada de la función recursiva definida por el usuario.



2. Conecte el primer elemento **`directory`** de salida de la función recursiva definida por el usuario con el elemento **`directory`** situado al final de la jerarquía del componente de salida.
3. Haga clic con el botón secundario en el conector de esta última asignación, seleccione la opción **Copia total** y haga clic en **Aceptar**.



La función de finida por el usuario ya está terminada.

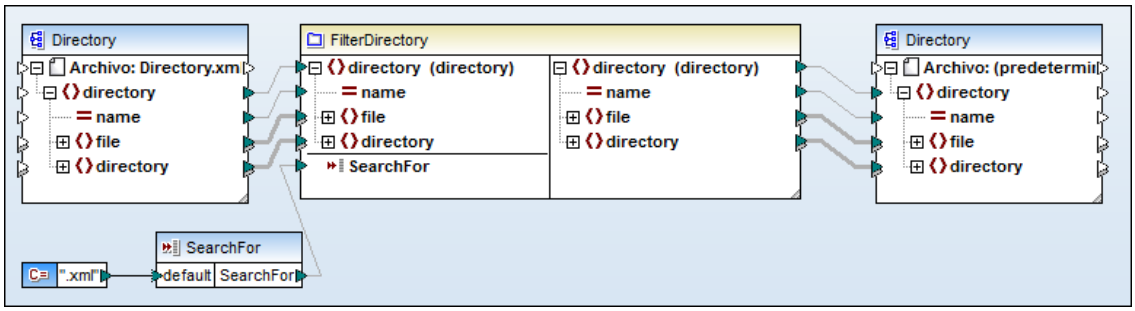
Haga clic en el icono situado en la parte superior del panel para volver al panel principal de asignación y seguir definiendo la asignación.

En el panel principal de asignación:

1. Inserte la función **`FilterDirectory`** de la sección **user** de la ventana Bibliotecas en el área de asignación.
2. Haga clic en **Insertar | Archivo o esquema XML** e inserte el esquema `Directory.xsd`

y el archivo de instancia `Directory.xml`.

3. Inserte un componente de salida basado en el esquema `Directory.xsd` y haga clic en **Omitir** cuando se le solicite un archivo de instancia.
4. Inserte la constante **`.xml`** y después un componente de entrada llamado **`SearchFor`**.
5. Cree las conexiones que aparecen en la imagen siguiente.
6. Cuando realice las conexiones de nivel superior (de **`directory`** a **`directory`** a ambos lados de la función definida por el usuario), haga clic con el botón secundario en el conector y seleccione **Copia total** en el menú contextual.



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

Nota: al hacer doble clic en el último elemento **`directory`** del componente **Directory** se abre un nivel de recursividad nuevo, es decir, verá un nuevo subnivel **directory | file | directory**. Con el conector de copia total se usan automáticamente todos los niveles de recursividad existentes en el archivo XML de instancia, sin necesidad de expandir los niveles de recursividad manualmente.

 © 2014 Altova GmbH

Altova MapForce 2014

Recursos globales

12 Recursos globales

Los recursos globales son una característica que mejora significativamente la interoperabilidad de los productos de Altova. Esta característica está disponible en estos productos: XMLSpy, MapForce, StyleVision y DatabaseSpy (también como parte del conjunto de aplicaciones Altova MissionKit).

Uso general de los recursos globales:

- El usuario puede definir flujos de trabajo que usan varias aplicaciones de Altova para procesar datos.
- Una aplicación puede invocar una aplicación de destino, iniciar en ella el procesamiento de datos y devolver los datos a la aplicación de origen.
- El usuario puede definir datos de entrada/salida, archivos y bases de datos como recursos globales.
- Durante el tiempo de ejecución se puede cambiar de recurso global para alternar recursos de desarrollo con recursos de implementación.
- Definición de casos hipotéticos para la realización de pruebas de calidad.

En todas las aplicaciones de Altova compatibles con la función de recursos globales, la ubicación predeterminada del archivo de definición de recursos globales, `GlobalResources.xml`, es la carpeta `c:\Documents and Settings\NombreUsuario\Mis Documentos\Altova\`. Gracias a este archivo de definición, los cambios realizados en los recursos globales están disponibles automáticamente en todas las aplicaciones. El usuario puede cambiar el nombre de este archivo de definición y también su ubicación. Para más información consulte el apartado Recursos globales: propiedades.

Mecanismo general:

- Los recursos globales se **definen** en una aplicación y automáticamente se guardan.
- Los recursos globales se **asignan** a los componentes cuyos datos queremos que sean variables.
- El recurso global se invoca / **activa** en una aplicación, lo cual permite cambiar de recursos en tiempo de ejecución.

En esta sección describimos cómo se definen y utilizan los recursos globales en MapForce, usando los archivos de ejemplo disponibles en la carpeta `...\MapForceExamples\Tutorial`.

Para activar la barra de herramientas Recursos globales:
Seleccione la opción de menú **Herramientas | Personalizar | Barras de herramientas** y marque la casilla *Recursos globales*.

El cuadro combinado de la barra de herramientas sirve para cambiar de un recurso a otro y siempre incluye la opción `Default`, que es la configuración predeterminada.
Al hacer clic en el icono **Recursos globales** se abre el cuadro de diálogo "Administrar recursos globales". Este cuadro de diálogo también se abre haciendo clic en **Herramientas | Recursos globales**.

 © 2014 Altova GmbH

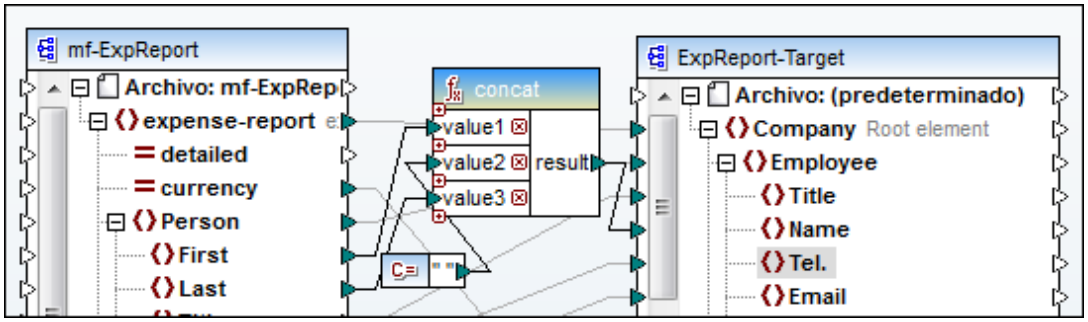
12.1 Recursos globales: archivos

En MapForce puede definir como recurso global cualquier archivo de entrada/salida de un componente, p. ej. archivos XML, esquemas XML, archivos de texto/CSV, bases de datos, archivos EDI y archivos Excel 2007 (o superior), etc.

Objetivo:

- Definir el archivo de entrada del componente de origen **mf-ExpReport** como recurso global.
- Alternar en tiempo de ejecución los dos archivos XML que suministran **datos de entrada** al componente y revisar el archivo XML resultante en el panel **Resultados**.

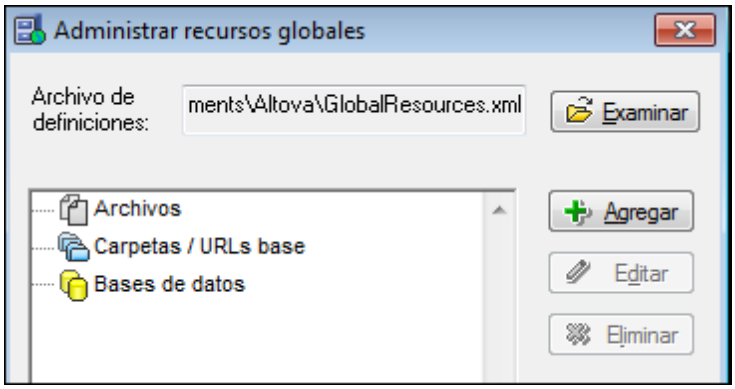
  Para este ejemplo utilizaremos el archivo `Tut-ExpReport.mfd` disponible en la carpeta `...\MapForceExamples\Tutorial`.



12.1.1 Definir y agregar recursos globales

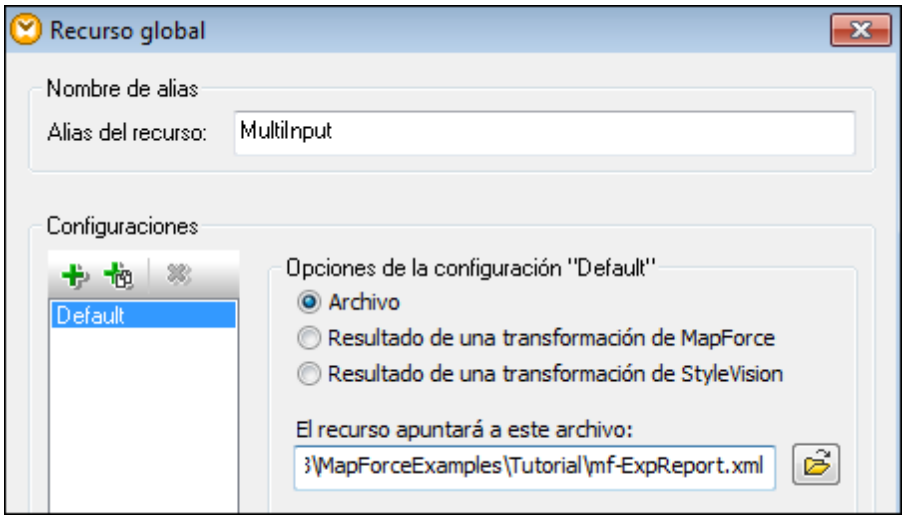
Definir/añadir recursos globales de archivo

1. Primero haga clic en el icono **Recursos globales** para abrir este cuadro de diálogo.



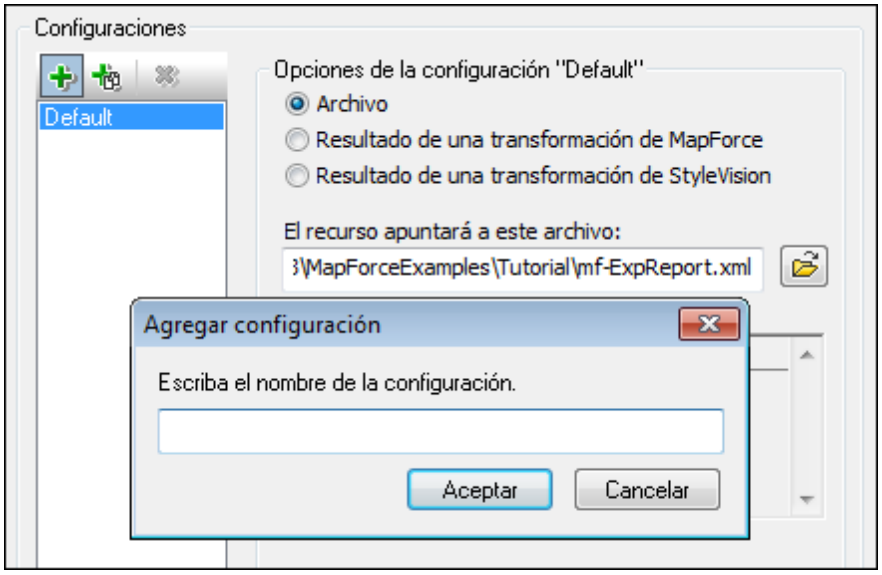
Así se representa un archivo de recursos globales vacío.

2. Haga clic en el botón **Agregar** y seleccione la opción **Archivo**. Se abre otro cuadro de diálogo.
3. En el campo *Alias del recurso* escriba el nombre de alias del recurso, p. ej. `MultiInput`.
4. Haga clic en el icono **Examinar** y seleccione el archivo que debe utilizarse como archivo de entrada predeterminado (`Default`), p. ej. `mf-ExpReport.xml`.

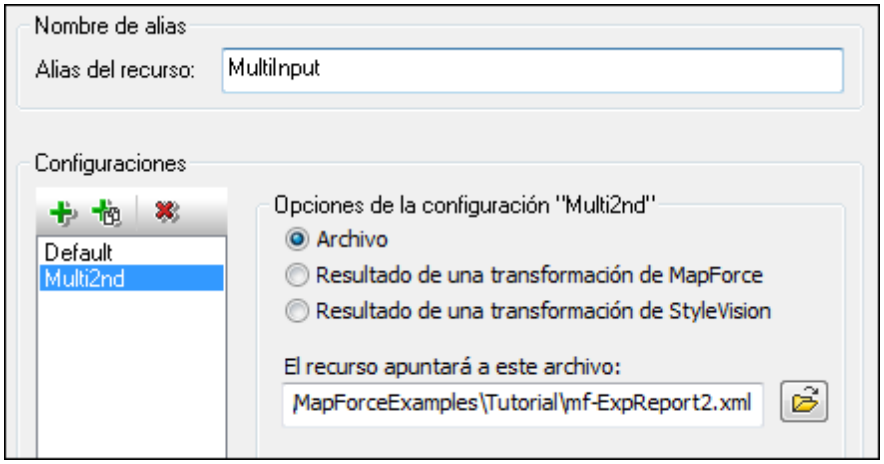


5. Haga clic en el icono **Agregar** del grupo de opciones *Configuraciones* para añadir una configuración nueva al *Alias* actual. Recuerde que con el icono **Copiar configuración** puede copiar la configuración seleccionada y guardarla con otro nombre.

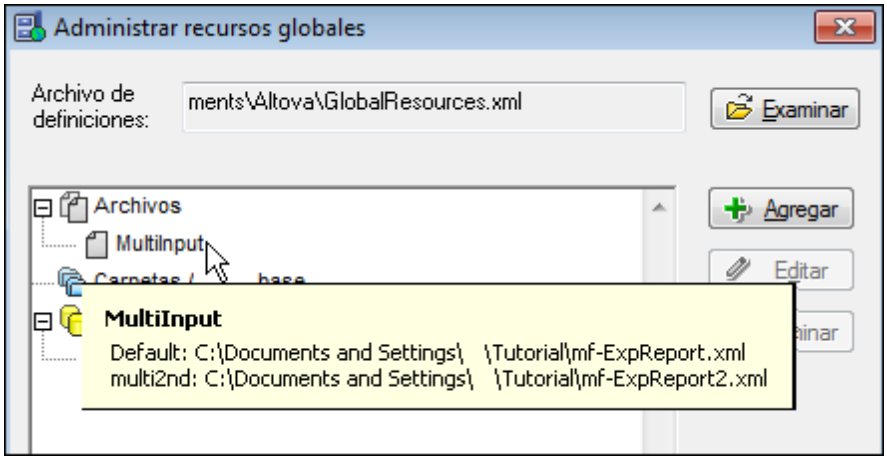
 © 2014 Altova GmbH



6. Escriba el nombre de la nueva configuración, p. ej. `Multi2nd`, y haga clic en **Aceptar**. `Multi2nd` se añade a la lista de configuraciones.
7. Haga clic otra vez en el icono **Examinar** y seleccione el archivo XML que debe utilizarse como archivo de entrada para la configuración `Multi2nd`, p. ej. `mf-ExpReport2.xml`.



8. Haga clic en **Aceptar** para terminar de definir el recurso global.
   El alias `MultiInput` se añadió al nodo *Archivos* del archivo de recursos globales.
   Si pasa el puntero del mouse sobre la entrada, aparece su definición.



9. Para finalizar haga clic en **Aceptar**.
   Ya hemos definido el recurso global. El siguiente paso es asignar los recursos globales a un componente.

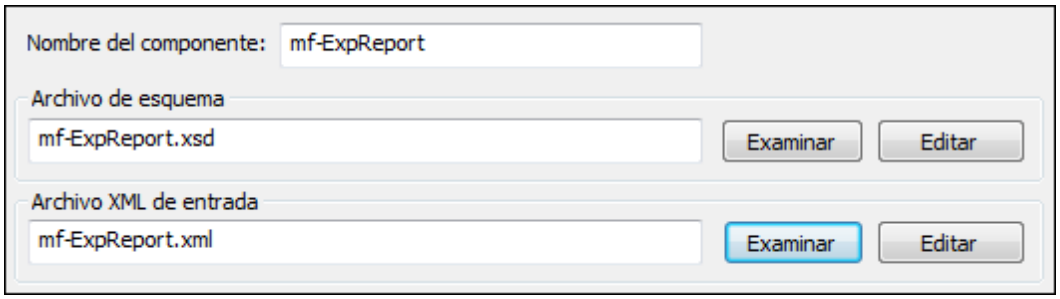
© 2014 Altova GmbH

12.1.2 Asignar un recurso global

Asignar recursos globales a un componente

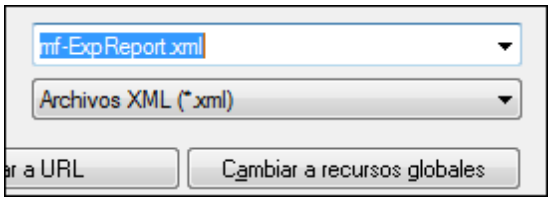
Ahora tenemos que asignar el recurso global al componente que debe utilizarlo, es decir, el archivo `mf-ExpReport.xml` que se utiliza como archivo de origen de la asignación.

1. Haga doble clic en el componente **mf-ExpReport** y después haga clic en el botón **Examinar** del campo *Archivo XML de entrada*.

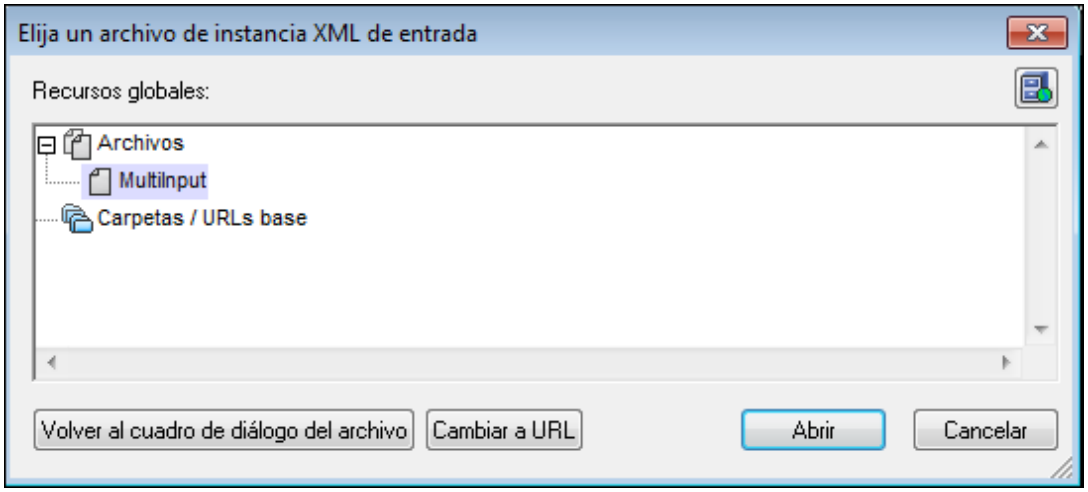


Se abre el cuadro de diálogo "Elija un archivo XML de instancia de entrada".

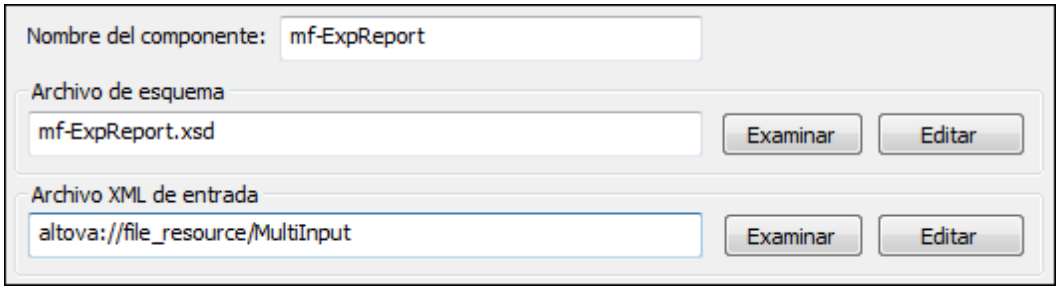
2. Haga clic en el botón **Cambiar a recursos globales**, situado en la parte inferior del cuadro de diálogo.



3. Haga clic en el recurso que desea asignar como archivo XML de entrada (en este caso `MultiInput`) y después haga clic en **Abrir**.



Nota: el campo *Archivo XML de entrada* del componente ahora contiene una referencia a un recurso: `altova://file_resource/MultiInput`.



4. Para terminar de asignar el recurso al componente haga clic en **Aceptar**.
   El siguiente paso es usar/activar el recurso global.

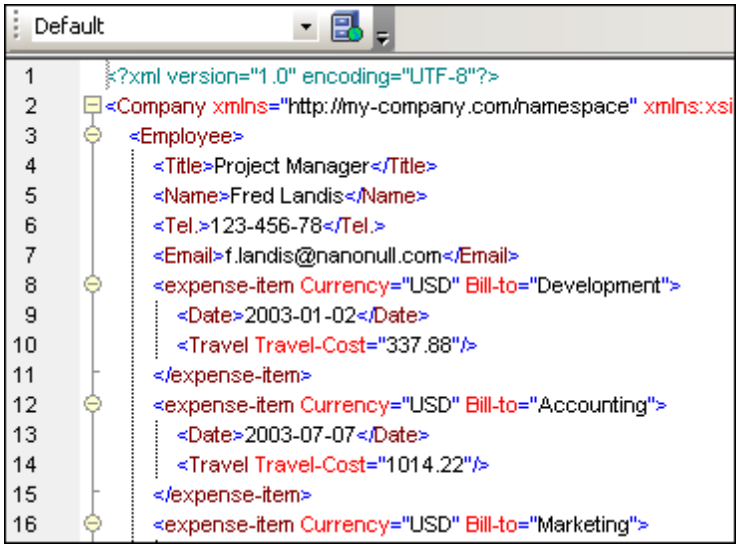
© 2014 Altova GmbH

12.1.3 Usar y activar un recurso global

Usar/activar un recurso global

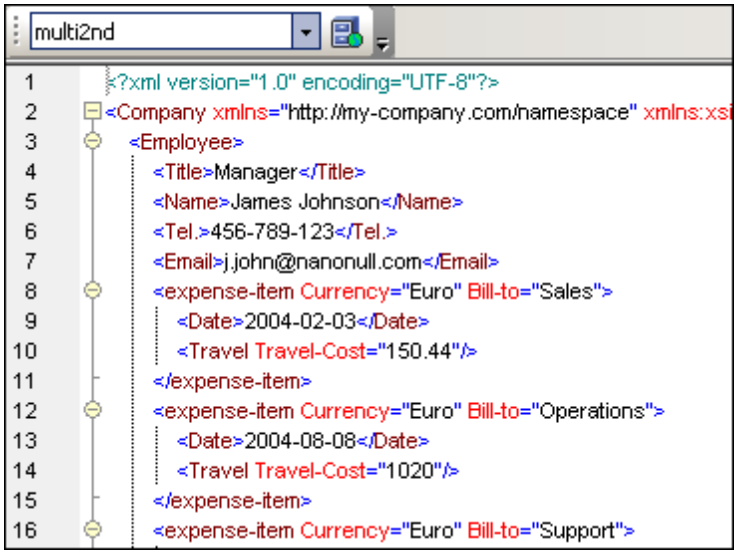
Llegados a este punto, la configuración activa es la configuración **`Default`**, que definimos en el primer paso para el alias `MultiInput`. Esto se puede ver en la barra de herramientas Recursos globales, que siempre indica la configuración activa.

1. Haga clic en la pestaña del panel **Resultados** para ver el resultado de la asignación.



2. Vuelva al panel **Asignación**.
3. Haga clic en el cuadro combinado de la barra de herramientas Recursos globales y seleccione `Multi2nd` de la lista desplegable.

4. Cambie al panel **Resultados** para ver el resultado nuevo.
   Ahora se utiliza el archivo `mf-ExpReport2.xml` como componente de origen de la asignación y produce resultados diferentes.



Note: el recurso global activo en cada momento (el indicado en la barra de herramientas Recursos globales) determina el resultado de la asignación. También determina la generación de código.

 © 2014 Altova GmbH

12.2 Recursos globales: carpetas

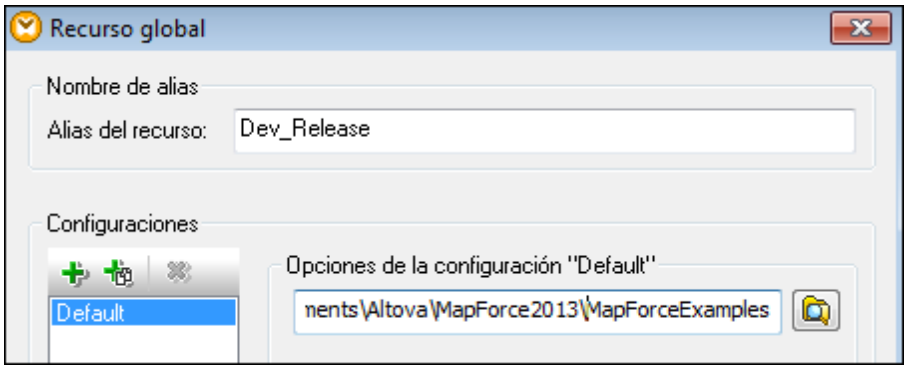
Las carpetas también se pueden definir como recurso global. Esto significa que los componentes de entrada pueden contener archivos que hacen referencia a carpetas diferentes, unas para la fase de desarrollo y otras para la fase de release, por ejemplo.

No obstante, en MapForce no es muy práctico definir carpetas como recursos globales porque durante la generación de código se le solicitará constantemente que confirme las carpetas de destino.

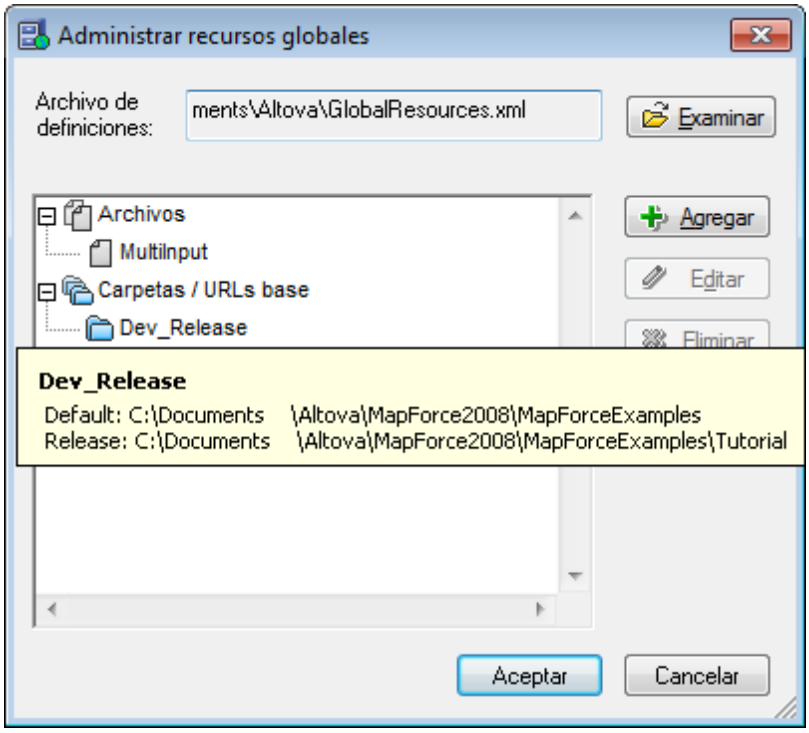
Para este ejemplo utlizamos el archivo de asignación `global-folder.mfd` guardado en la carpeta `...\MapForceExamples\Tutorial`.

Definir/añadir recursos globales de carpeta

1. Haga clic en el icono **Recursos globales** para abrir el cuadro de diálogo de recursos globales.
2. Haga clic en el botón **Agregar** y seleccione la opción **Carpeta**.
3. Escriba el nombre de alias del recurso, p. ej. `Dev_Release`.
4. Haga clic en el icono **Examinar** y seleccione la carpeta de entrada predeterminada (`Default`) `...\MapForceExamples`.



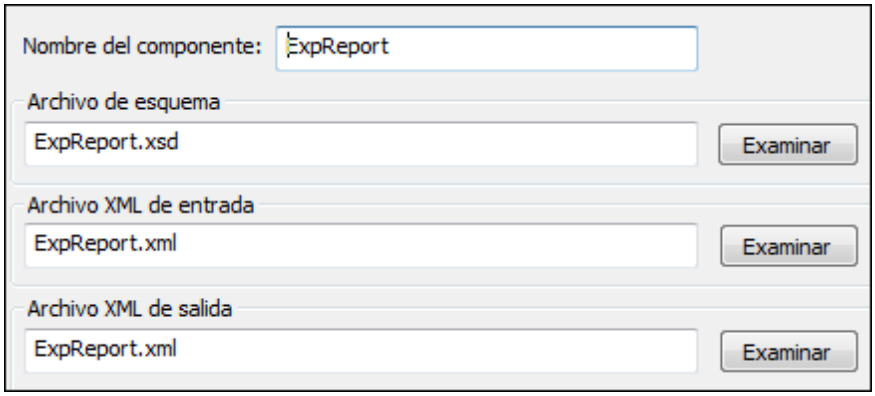
5. Haga clic en el icono **Agregar** del grupo de opciones *Configuraciones* para añadir una configuración nueva al *Alias* actual. Recuerde que con el icono **Copiar configuración** puede copiar la configuración seleccionada y guardarla con otro nombre.
6. Escriba el nombre de la nueva configuración, p. ej. `Release`, y haga clic en **Aceptar**. `Release` se añade a la lista de configuraciones.
7. Haga clic en el icono **Examinar** y seleccione la carpeta de entrada para el release, p. ej. `...\MapForceExamples\Tutorial`.



8. Para terminar de definir el recurso global de carpeta haga clic en **Aceptar**.

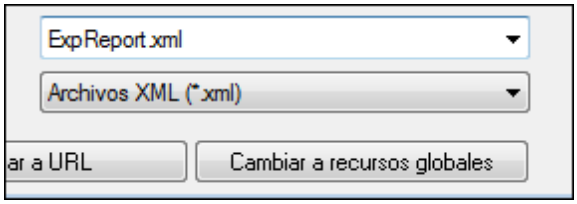
Asignar recursos globales de carpeta:

1. Haga doble clic en el componente **ExpReport** y después pulse el botón **Examinar** del campo *Archivo XML de entrada*.



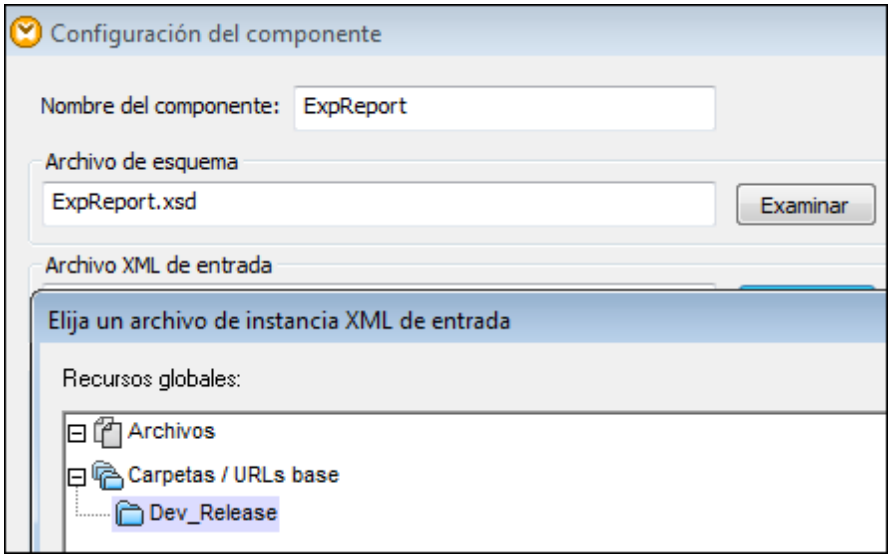
Aparece el cuadro de diálogo "Elija un archivo XML de instancia de entrada".

2. Haga clic en el botón **Cambiar a recursos globales**, situado en la parte inferior del cuadro de diálogo.



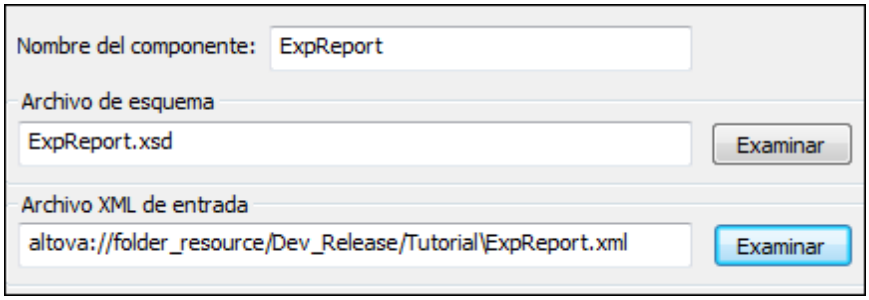
3. Seleccione el recurso que desea asignar como entrada para el componente (en este

caso `Dev_Release`) y haga clic en **Abrir**.



Aparece el cuadro de diálogo "Abrir".

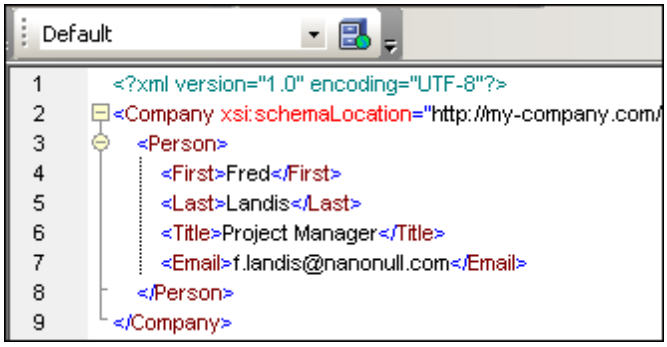
4. Seleccione el nombre del **archivo** que debe servir de **recurso** de archivo de desarrollo y para el release en ambas carpetas, p. ej. `ExpReport.xml`.
   Para terminar de asignar el recurso de carpeta haga clic en **Aceptar**.



No olvide que este archivo está disponible en ambas carpetas pero tiene diferente contenido.

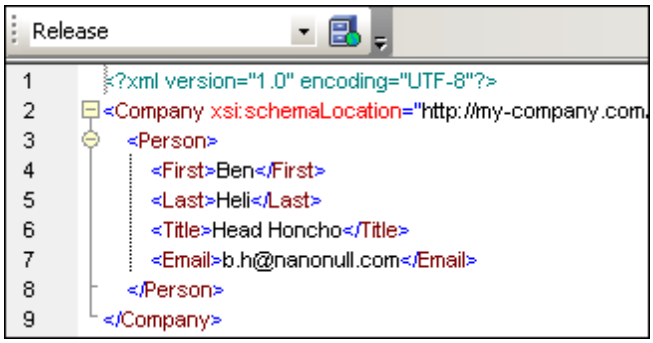
Cambiar de recurso de carpeta en tiempo de ejecución:

1. Cambie al panel **Resultados** para ver el resultado de la transformación.
   Observe que esta es la configuración/carpeta predeterminada (`Default`) `.../MapforceExamples`.



2. Vuelva al panel **Asignación**.

3. Haga clic en el cuadro combinado de la barra de herramientas Recursos globales y seleccione la opción `Release` en el cuadro combinado.
3. Cambie al panel **Resultados** para ver el resultado que se obtiene con el recurso global `Release`.



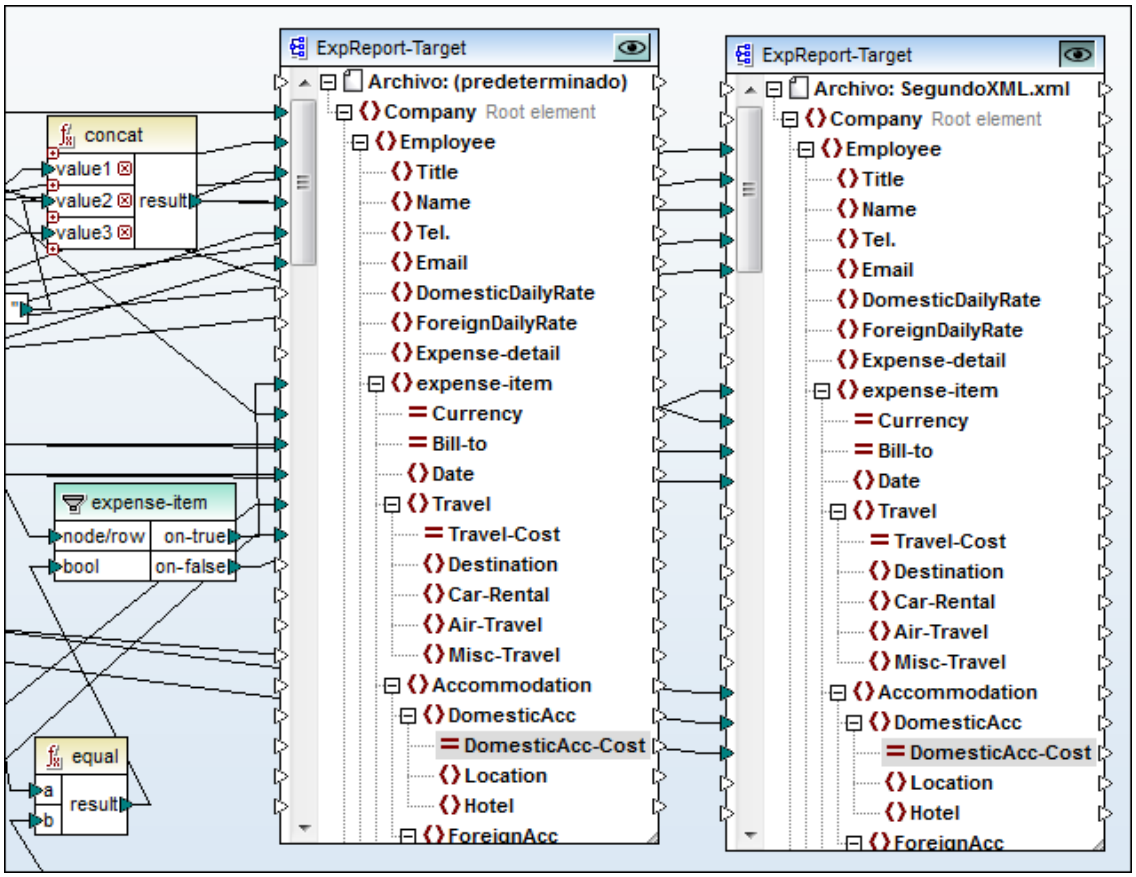
Ahora podemos ver el resultado obtenido usando la carpeta de release `.../MapforceExamples/Tutorial`.

 © 2014 Altova GmbH

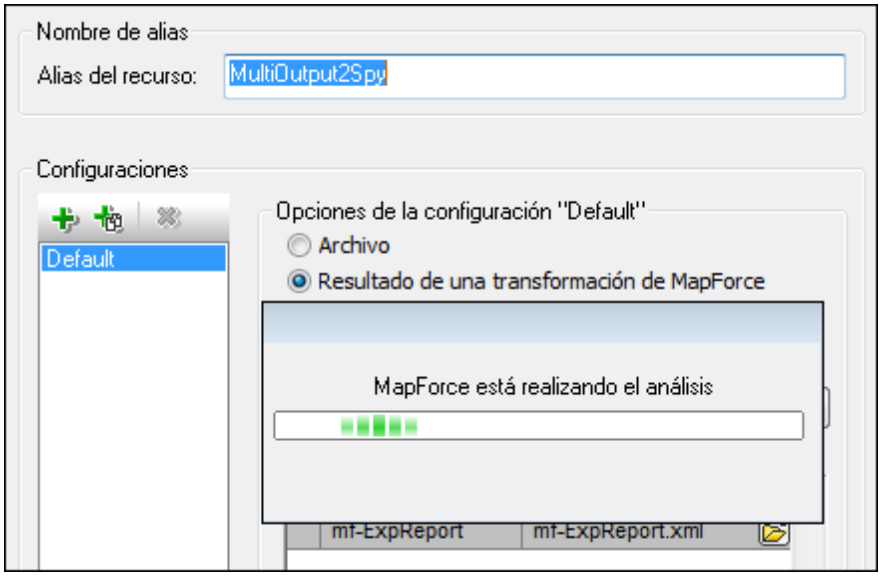
12.3 Recursos globales: flujo de trabajo en la aplicación

El objetivo de esta sección es crear un flujo de trabajo entre dos aplicaciones de Altova. El flujo de trabajo comienza en XMLSpy, que inicia MapForce para generar un archivo XML y MapForce devuelve el archivo XML generado a XMLSpy, donde se puede seguir procesando.

Esta asignación utiliza dos componentes de salida para producir dos tipos de resultados: un archivo con los gastos de viaje y otro con todos los gastos excepto los de viaje, creados a partir del archivo de entrada de gastos. Para este ejemplo usamos el archivo de asignación `Tut-ExpReport-multi.mfd` disponible en la carpeta `...\MapForceExamples\Tutorial`.

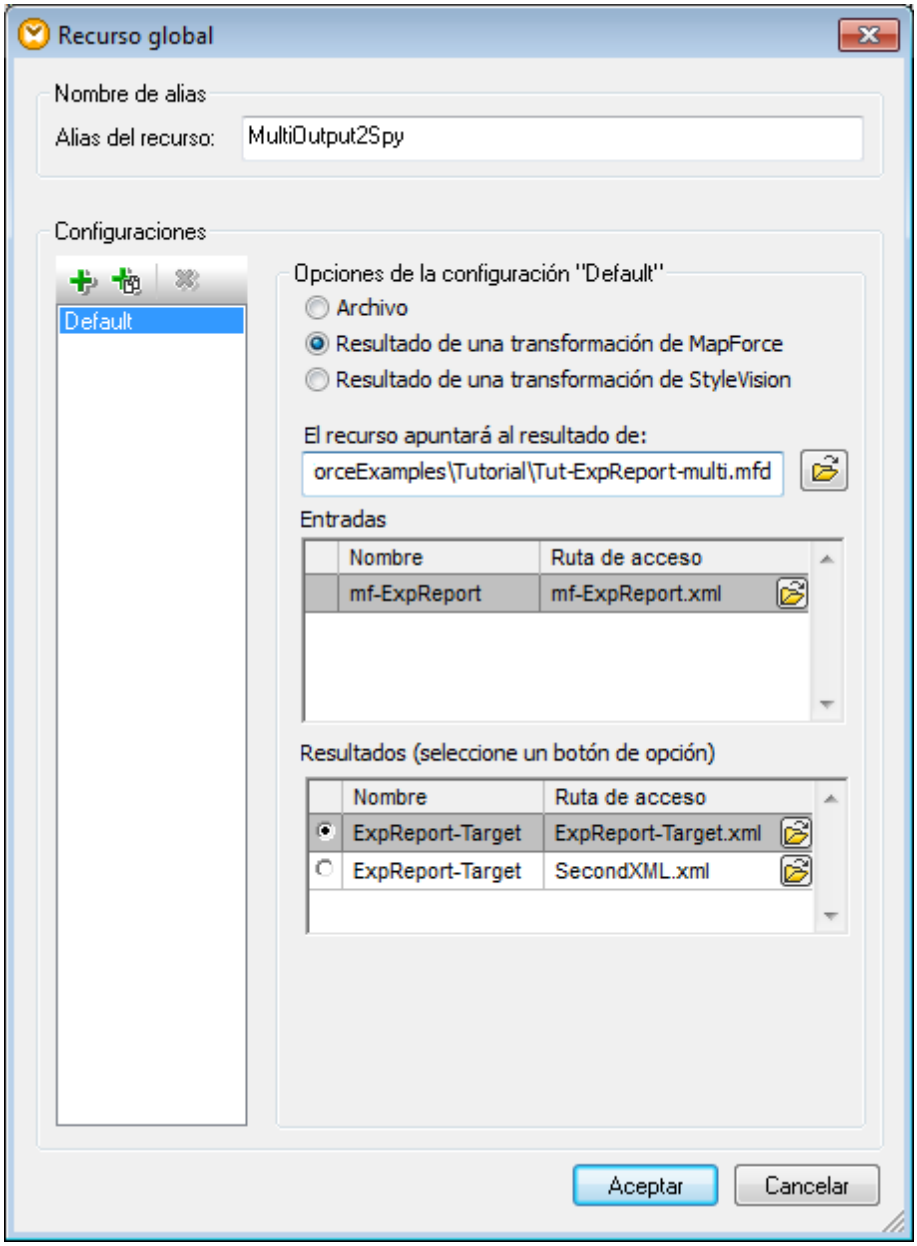


1. Haga clic en el icono **Recursos globales**.
2. Aparece el cuadro de diálogo de recursos globales. Haga clic en **Agregar** y seleccione la opción **Archivo**.
3. Escriba el nombre de alias del recurso, p. ej. `MultiOutput2Spy`
4. Seleccione el botón de opción *Resultado de una transformación de MapForce* y después haga clic en el icono **Examinar**.
5. Seleccione el archivo de asignación `Tut-ExpReport-multi.mfd`.

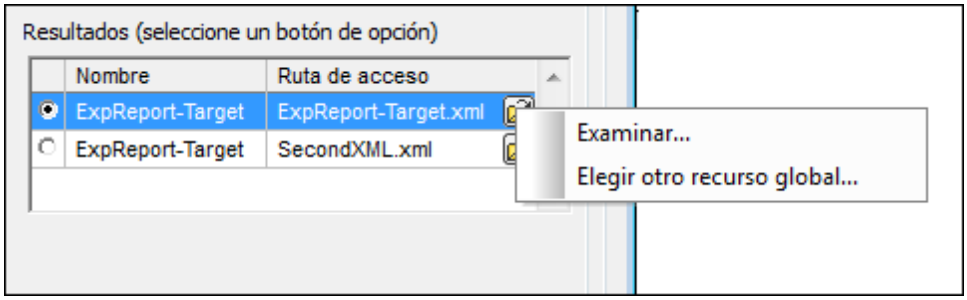


MapForce analiza la asignación y muestra los archivos de entrada y salida en recuadros diferentes.

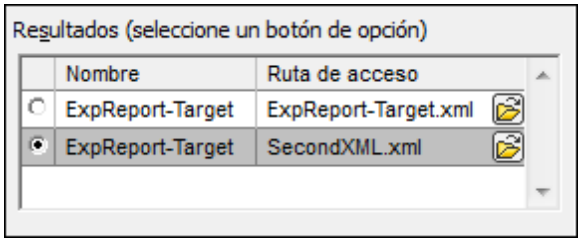
© 2014 Altova GmbH



6. En el panel *Resultados* seleccione el botón de opción superior.
   Observe que el nombre del archivo de salida es `ExpReport-Target.xml` y que ahora estamos definiendo la configuración `Default` (predeterminada).
7. Haga clic en el icono y seleccione la opción **Examinar** del menú emergente para definir la ubicación nueva del archivo de salida, p. ej. `C:\Temp`.



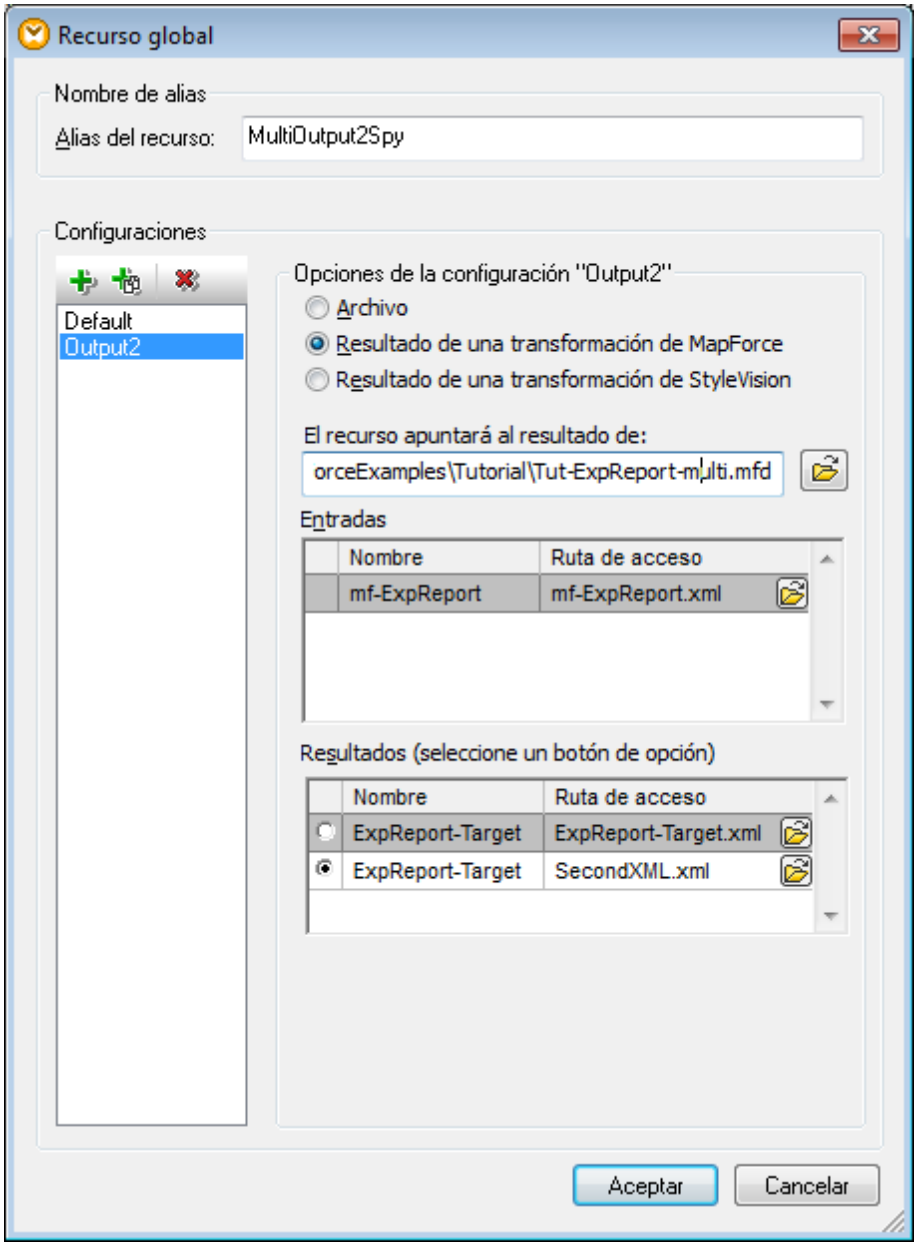
8. Escriba la ubicación nueva (p. ej. `C:\Temp`) y haga clic en **Guardar**. Esta ubicación puede ser diferente de la ubicación definida en la configuración del componente.
9. Haga clic en el icono del grupo *Configuraciones* (del cuadro de diálogo de recursos globales) para añadir una configuración nueva al alias del recurso.
11. Escriba el nombre de la configuración nueva, p. ej. `Output2`.
12. Seleccione el botón de opción *Resultado de una transformación de MapForce* y después haga clic en el icono **Examinar**.
    Seleccione el archivo `Tut-ExpReport-multi.mfd`.
13. Seleccione el botón de opción inferior del panel *Resultados*. Observe que el nombre del archivo de salida es `SecondXML.xml`.



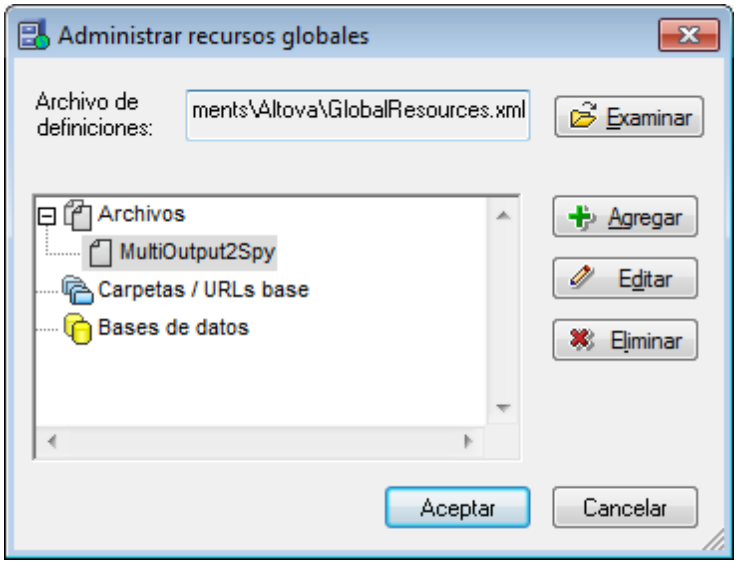
14. Haga clic en el icono y seleccione **Examinar** en el menú contextual para definir la ubicación nueva del archivo de salida, p. ej. `C:\Temp`.

    Nota: si en este menú contextual selecciona **Elegir otro recurso global** puede guardar el resultado de MapForce como recurso global, es decir, el resultado se almacena en un archivo al que apunta el recurso global.

 © 2014 Altova GmbH



15. Para guardar estos recursos globales nuevos haga clic en **Aceptar**.
    El alias del recurso nuevo `MultiOutput2Spy` se añadió al archivo de definición de recursos globales.



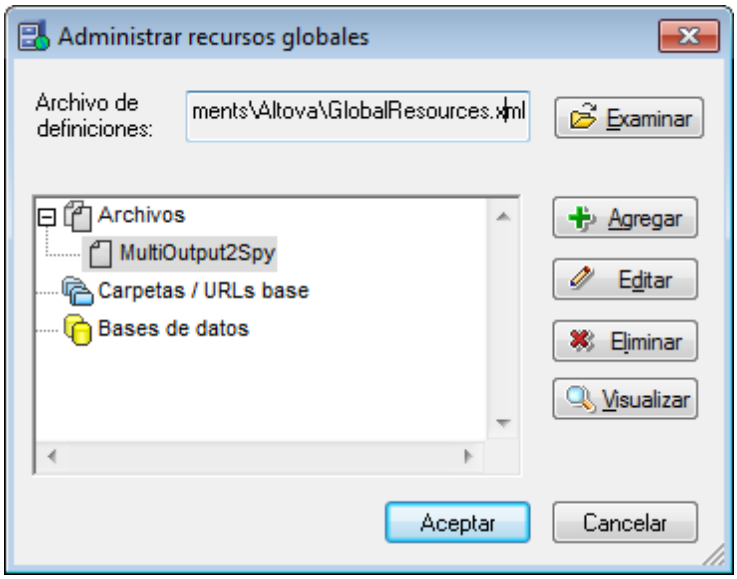
16. Haga clic en **Aceptar** para finalizar la fase de definición de recursos globales.

© 2014 Altova GmbH

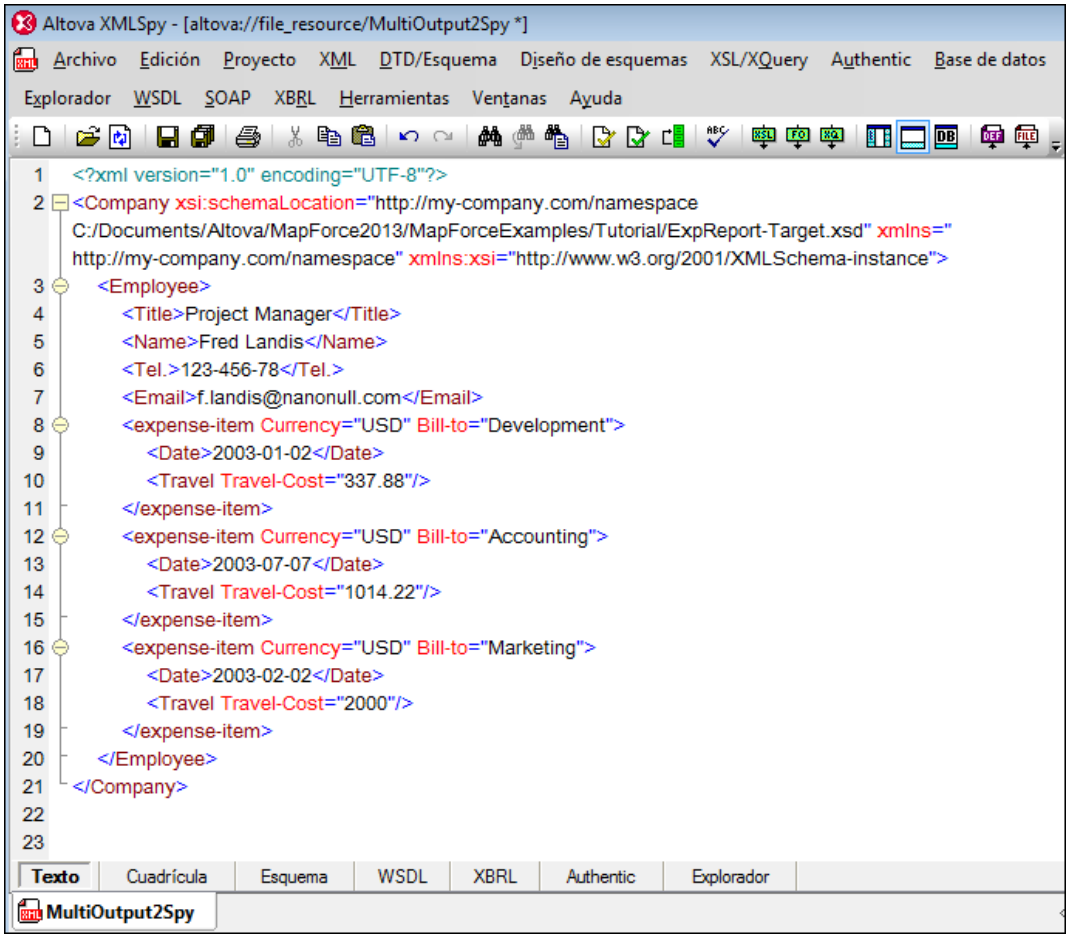
12.3.1 Iniciar el flujo de trabajo en la aplicación

En este apartado describimos cómo activar el recurso global en XMLSpy y cómo devolver a esta aplicación la transformación de MapForce resultante.

1. Inicie XMLSpy y cierre MapForce. Así podrá ver más claramente cómo interactúan las aplicaciones.
2. En XMLSpy seleccione **Herramientas | Recursos globales**.
3. Seleccione la entrada `MultiOutput2Spy` y haga clic en el botón **Visualizar**.



Aparece un cuadro que nos informa de que MapForce está realizando la transformación y el resultado de la transformación aparece en la vista Texto de XMLSpy.



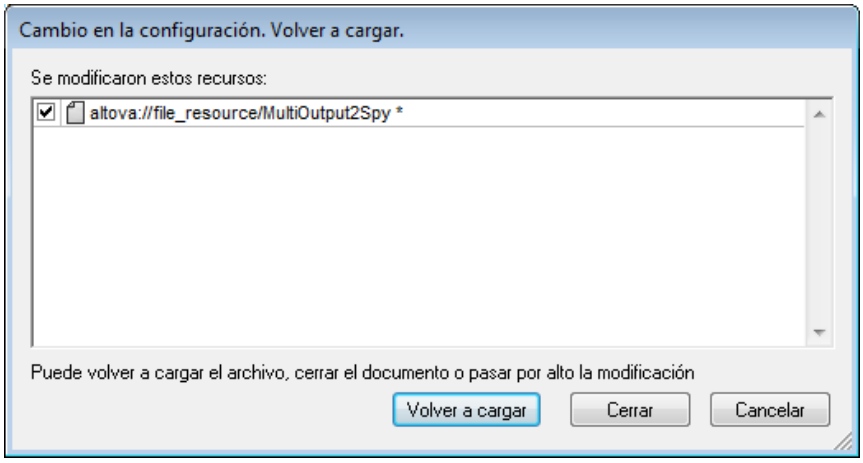
Nota:

- La configuración seleccionada en este momento es `Default`.
- El nombre de alias del recurso aparece en la barra de título de la aplicación, en este caso `altova://file_resource/MultiOutput2Spy`.
- El archivo de salida se abrió con el nombre `MultiOutput2Spy.xml`.
- El archivo `ExpReport-Target.xml` se copió en la carpeta `c:\Temp`.

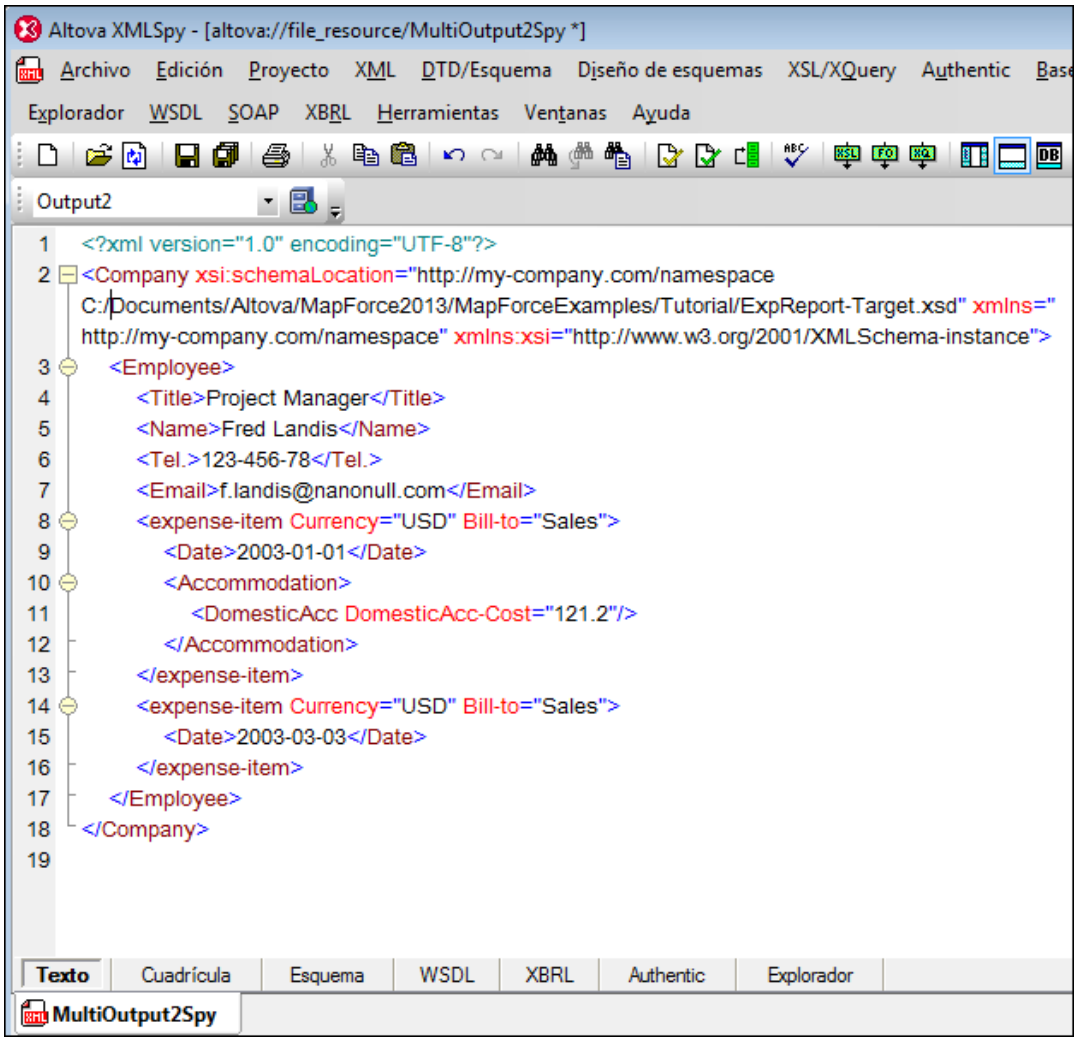
Para recuperar el archivo de salida que contiene todos los gastos menos los de viaje:

1. Haga clic en el cuadro combinado de la barra de herramientas Recursos globales y seleccione `Output2` en el cuadro combinado.
   Aparece este mensaje.

 © 2014 Altova GmbH



2. Haga clic en **Volver a cargar** para recuperar el segundo archivo de salida definido por el recurso.



El resultado de la transformación aparece en la vista Texto y sobrescribe el archivo anterior `MultiOutput2Spy.xml`.

Nota:

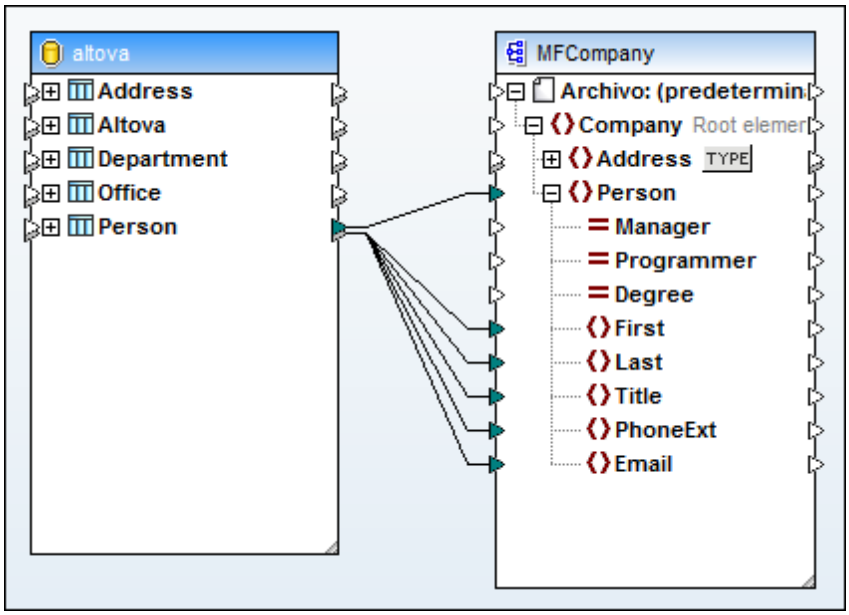
- La configuración seleccionada en este momento es `Output2`.
- El archivo de salida se abrió como MultiOutput2Spy.
- El archivo `SecondXML.xml` se copió a la carpeta `c:\Temp`.

© 2014 Altova GmbH

12.4 Recursos globales: bases de datos

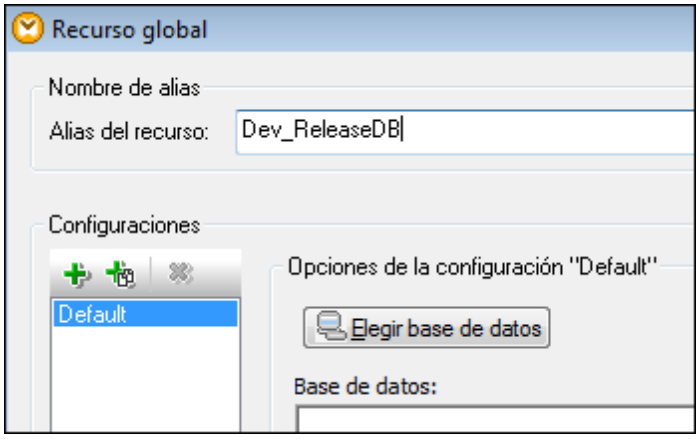
También puede definir como recursos globales los componentes de base de datos, lo cual permite hacer referencia a BD diferentes en la fase de desarrollo y en la fase del release. Los recursos de BD pueden ser tanto componentes de origen como de destino.

En esta sección utilizamos el archivo de asignación `PersonDB.mfd` disponible en la carpeta `...\MapForceExamples\Tutorial`.



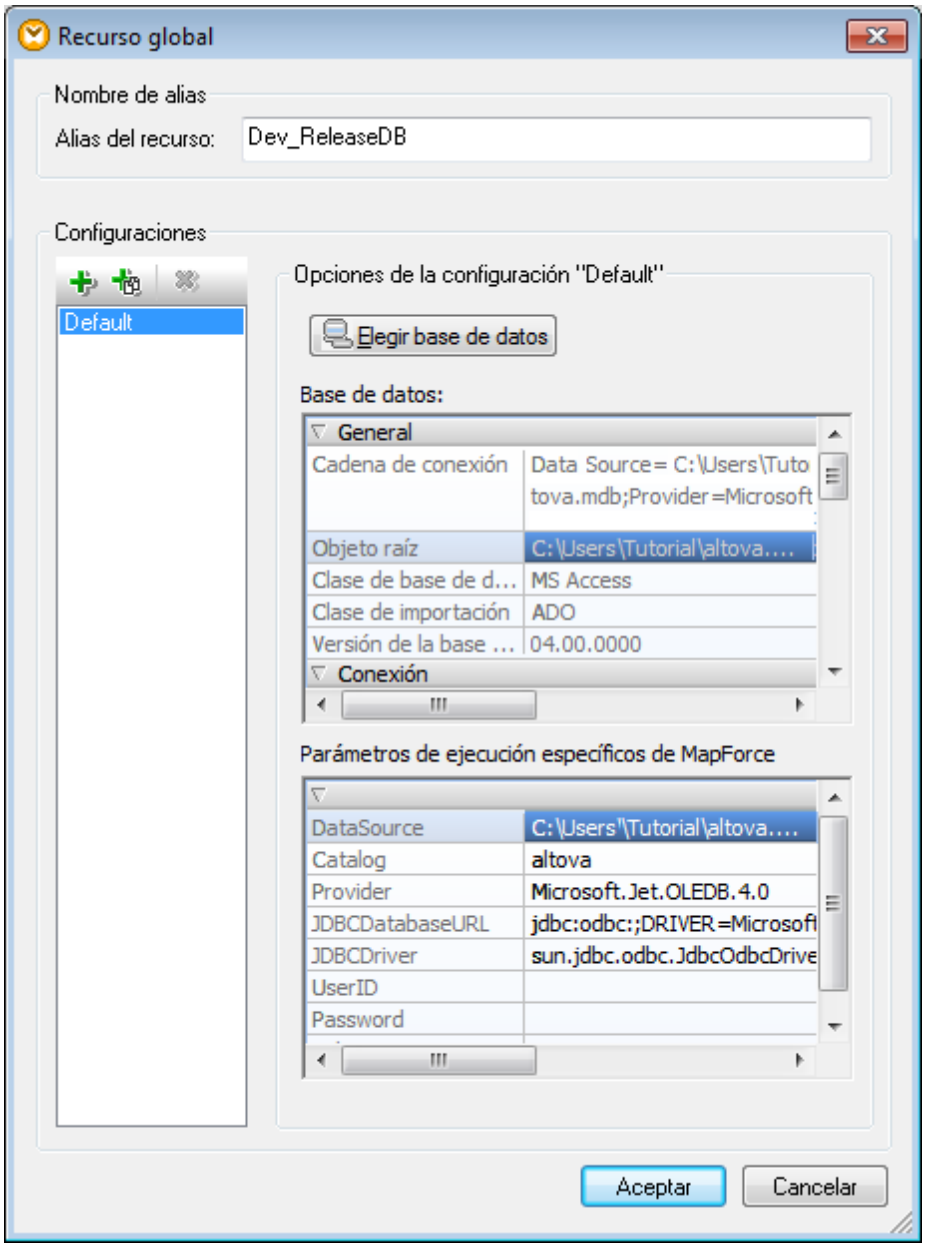
Definir/añadir un recurso global de base de datos

1. Haga clic en el icono de la barra de herramientas Recursos globales.
2. Aparece el cuadro de diálogo de recursos globales. Pulse el botón **Agregar** y seleccione la opción **Base de datos**.
3. Escriba el nombre de alias del recurso, p. ej. `Dev_ReleaseDB`.



4. Haga clic en el botón **Elegir base de datos**. Aparece otro cuadro de diálogo, seleccione el botón **Asistente para la conexión** y después seleccione la base de datos `Altova.mdb` situada en la carpeta `...\MapForceExamples`. Para terminar haga clic en **Conectarse**.

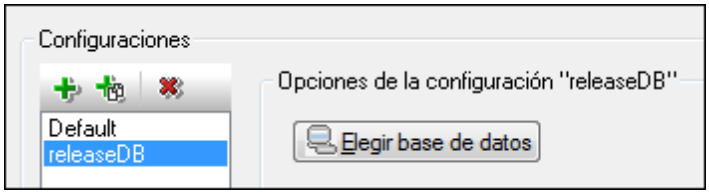
Esta será nuestra base de datos para la fase de **desarrollo**.



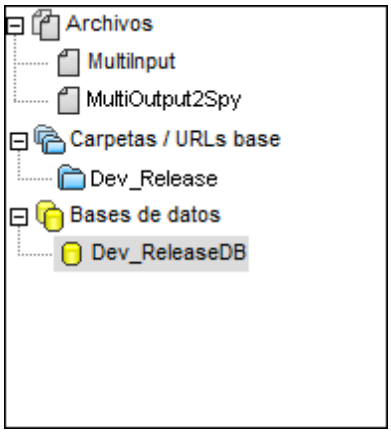
5. Haga clic en el botón **Agregar** del grupo *Configuraciones* para añadir una

 © 2014 Altova GmbH

configuración nueva al alias actual. Escriba el nombre de la nueva configuración, p. ej. `releaseDB`.



6. Haga clic otra vez en el botón **Elegir base de datos** y seleccione la base de datos de la fase de **release**, p. ej. `altova.mdb` de la carpeta `...\MapForceExamples\Tutorial`. Para terminar de definir los recursos globales de base de datos haga clic en **Aceptar**.



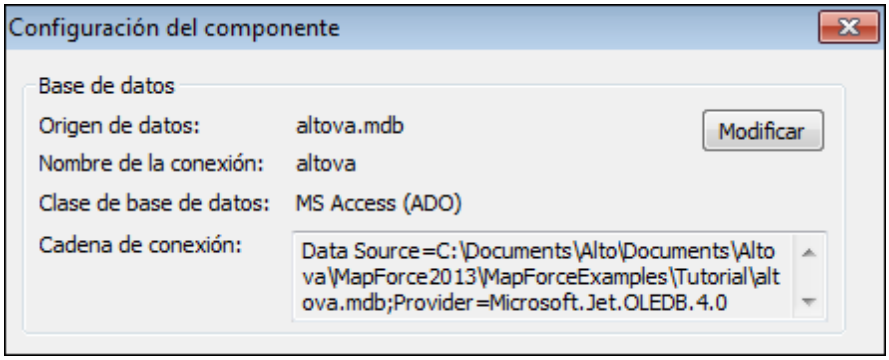
Nota:
En el cuadro de diálogo "Recursos globales" puede editar y cambiar los datos de los paneles *Base de datos* y *Parámetros de ejecución específicos de MapForce* después de seleccionar la BD. Por ejemplo, si hace doble clic en el campo *Contraseña*, podrá actualizar la contraseña actual de la BD.

El panel *Parámetros de ejecución específicos de MapForce* se rellena con los valores predeterminados de la BD una vez seleccionada la BD. Estos parámetros se utilizan a la hora de generar **código** y de generar la **vista previa** en el motor de ejecución integrado.

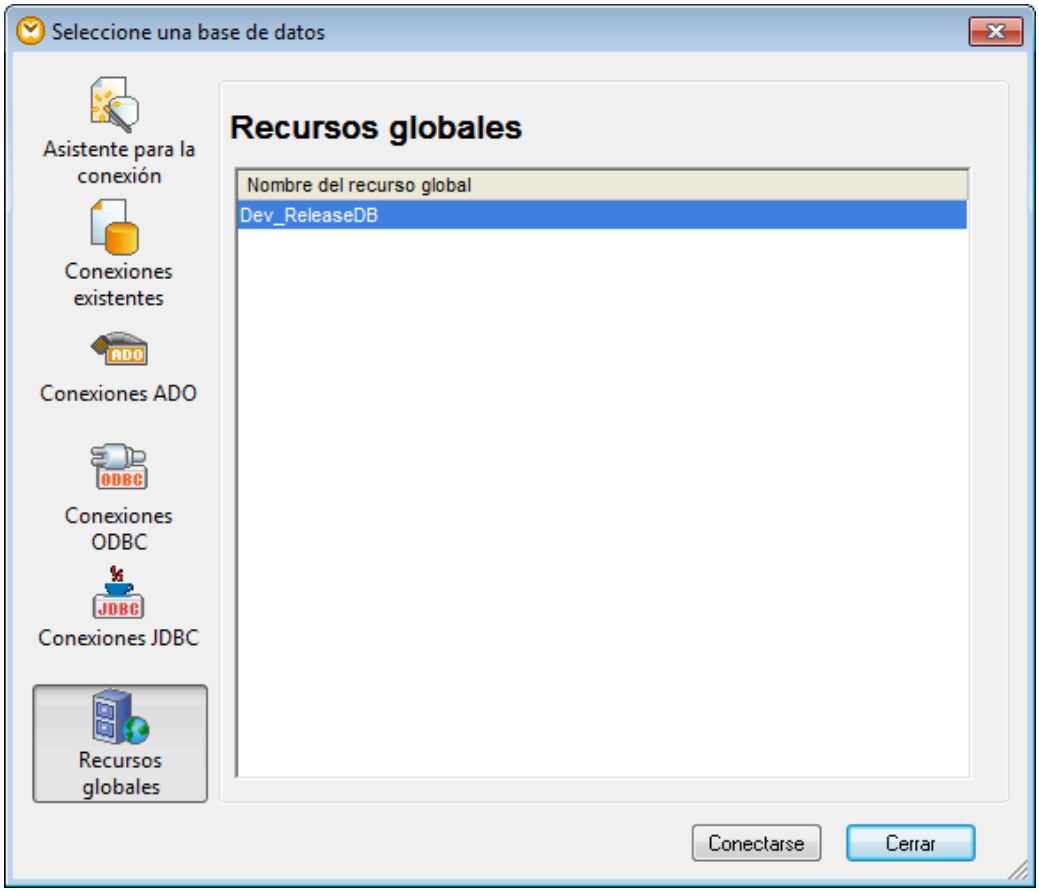
Dependiendo del tipo de BD utilizado, puede que aparezca el cuadro de diálogo "Elegir el objeto raíz". En este cuadro de diálogo puede seleccionar el objeto raíz (botón **Establecer objeto raíz**) o dejar la selección para otro momento (botón **Omitir**).

Asignar el recurso de base de datos:

1. Haga doble clic en el componente de BD **Altova**. Aparece el cuadro de diálogo de configuración del componente. Haga clic en el botón **Modificar**.

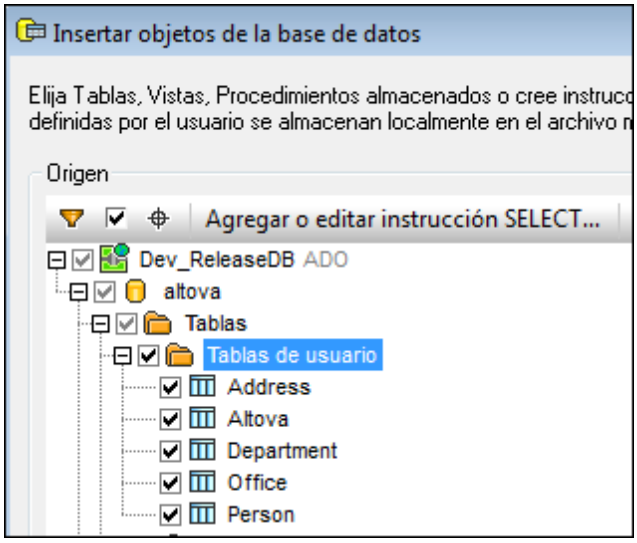


Aparece el cuadro de diálogo "Seleccione una base de datos". Pulse el botón **Recursos globales** situado en la parte inferior del panel izquierdo.

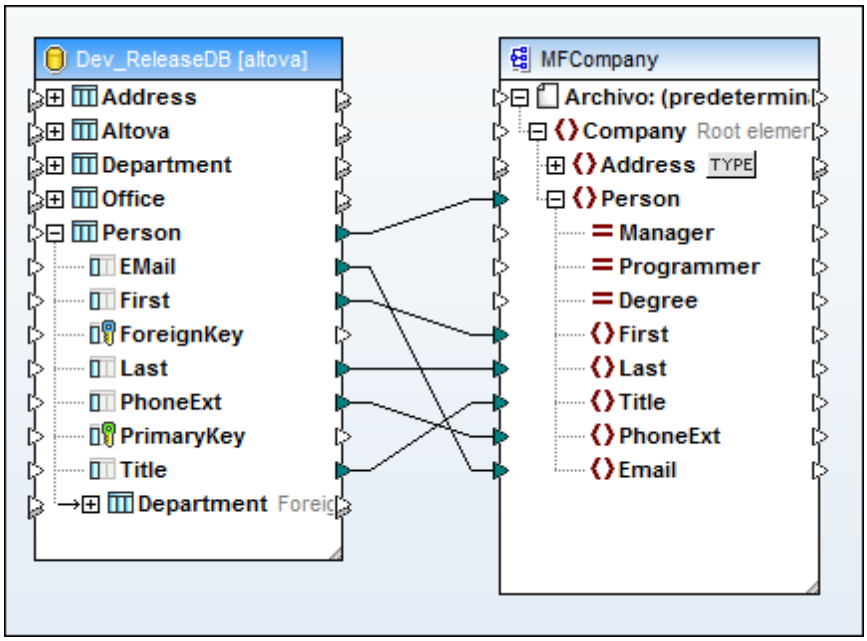


2. Seleccione `Dev_ReleaseDB` y haga clic en **Conectarse**.

 © 2014 Altova GmbH

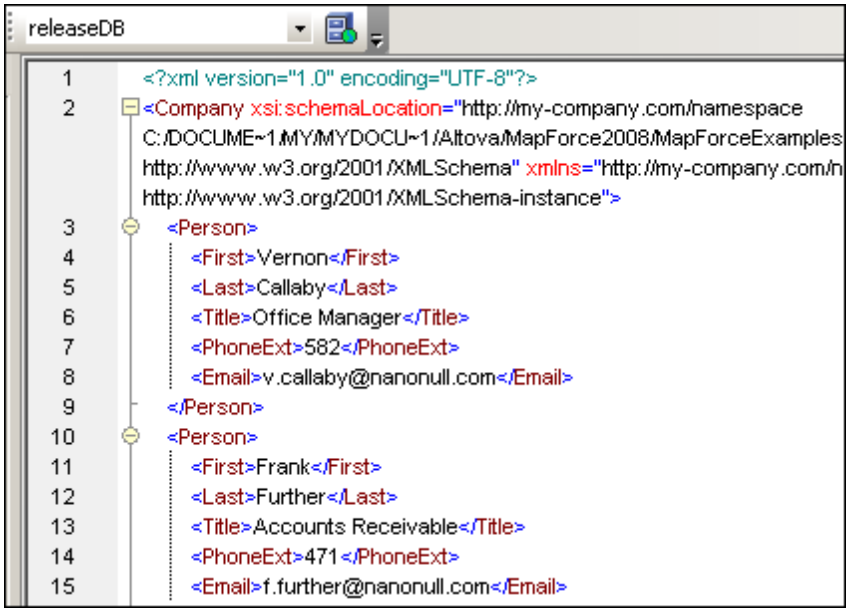


3. Seleccione las tablas que desea incorporar al componente y haga clic en **Aceptar**. El alias del recurso aparece ahora en el título del componente (**Dev_ReleaseDB [Altova]**)



Cambiar de recurso de BD en tiempo de ejecución:

1. Haga clic en la pestaña del panel **Resultados** para ver el resultado de la asignación. Tenga en cuenta que esta es la base de datos predeterminada (configuración `Default`) de la carpeta `.../MapforceExamples`.
2. Vuelva al panel **Asignación**. Haga clic en el cuadro combinado de la barra de herramientas Recursos globales y seleccione `releaseDB` en el cuadro combinado.
3. En el cuadro de diálogo que aparece haga clic en el botón **Volver a cargar**.
4. Haga clic otra vez en la pestaña del panel **Resultados** para ver el resultado de la asignación.



Ahora aparecen los datos de la BD de la fase de release. Puesto que ambas BD son idénticas, en este ejemplo no se aprecia ninguna diferencia entre los resultados.

 © 2014 Altova GmbH

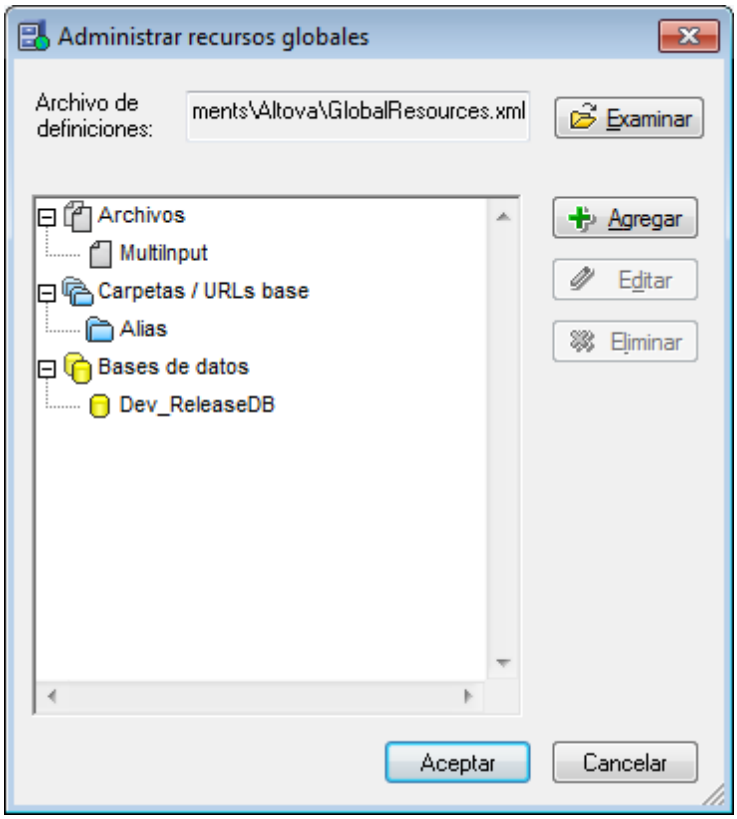
12.5 Recursos globales: propiedades

El archivo XML de recursos globales

Las definiciones de recursos globales se almacenan en un archivo XML, cuyo nombre predeterminado es `GlobalResources.xml`. Su ubicación predeterminada es `C:\Documents and Settings\<nombreUsuario>\Mis Documentos\Altova\`. Este archivo es el archivo XML de recursos globales predeterminado de todas las aplicaciones de Altova. Esto significa que los recursos globales definidos en una aplicación están disponibles en todas las demás aplicaciones de Altova automáticamente (siempre y cuando las demás aplicaciones usen también este archivo XML de recursos globales).

Si quiere puede cambiar el nombre de este archivo y guardarlo en otra ubicación. Por tanto, puede tener varios archivos XML de recursos globales. No obstante, en cada aplicación solamente puede haber un archivo XML activo y la aplicación tendrá acceso solamente a las definiciones del archivo que esté activo en ese momento.

El archivo XML de recursos globales se elige en el cuadro de diálogo "Administrar recursos globales". Haga clic en el botón **Examinar** del campo *Archivo de definiciones* y seleccione el archivo que desea usar.



Administrar recursos globales: añadir, editar, eliminar recursos

En el cuadro de diálogo "Administrar recursos globales" puede añadir un recurso global al archivo XML de recursos globales seleccionado. También puede editar los recursos o eliminarlos. El archivo XML de recursos globales organiza los alias añadidos en tres categorías: archivos/carpetas/bases de datos.

Para añadir un recurso global:

Pulse el botón **Agregar** y defina el recurso global en el cuadro de diálogo que aparece. Para terminar haga clic en **Aceptar** y el recurso global (o alias) se añade a la lista de definiciones del archivo XML de recursos globales seleccionado.

Para editar un recurso global:
Seleccione el recurso global y haga clic en **Editar**. Aparece un cuadro de diálogo donde puede realizar los cambios necesarios. Para terminar haga clic en **Aceptar**.

Para eliminar un recurso global:
Seleccione el recurso y haga clic en **Eliminar**.

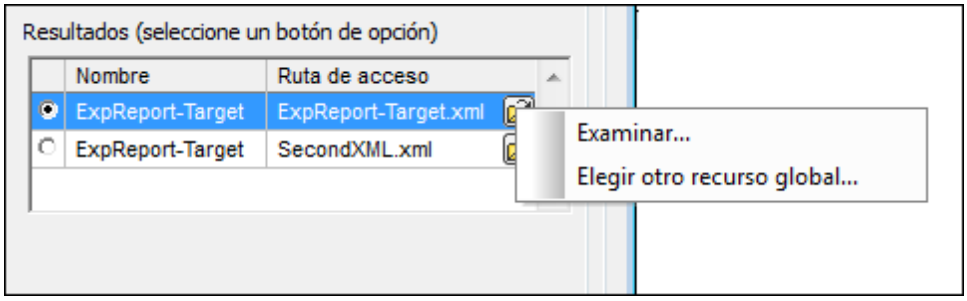
Para ver el resultado del flujo de trabajo de una aplicación:
Si una aplicación (p. ej. XMLSpy) llama a otra aplicación (p. ej. MapForce), entonces en el cuadro de diálogo "Administrar recursos globales" hay un botón más, el botón **Visualizar**.

Si hace clic en este botón **Visualizar** el efecto del recurso global seleccionado en ese momento se presenta en la aplicación que hace la llamada. Para más información, consulte el apartado Recursos globales: flujo de trabajo en la aplicación.

Para guardar los cambios realizados en el cuadro de diálogo "Administrar recursos globales":
Cuando termine de añadir, editar y eliminar recursos, asegúrese de que hace clic en el botón **Aceptar** para guardar todos los cambios en el archivo XML de recursos globales.

Nota: los nombres de alias de los recursos deben ser únicos dentro de la categoría a la que pertenece el recurso (es decir, archivos/carpetas/bases de datos). No obstante, puede definir un nombre de alias idéntico en dos categorías diferentes, p. ej. el alias `multiInput` puede existir en la categoría *Archivos* y en la categoría *Carpetas*.

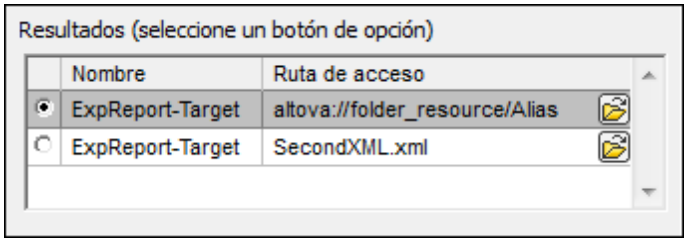
Seleccionar como recurso global los resultados de una transformación de MapForce
En una transformación de MapForce que tiene más de un archivo de resultados, puede definir uno de ellos como recurso global, seleccionando su botón de opción.



Los **resultados** generados por la asignación se pueden guardar:

- como recurso global (haciendo clic en el icono y eligiendo la opción **Elegir otro recurso global**). Los resultados se almacenan en un archivo al que apunta físicamente el recurso global.

 © 2014 Altova GmbH



- como archivo (haciendo clic en el icono y eligiendo la opción **Examinar**).

  Si no selecciona ninguna de estas dos opciones, se crea un archivo XML **temporal** cada vez que se utiliza el recurso global.

Qué recurso se utiliza en tiempo de ejecución

Hay dos factores que determinan qué recursos globales se pueden usar y qué recursos globales se utilizan en cada momento:

- *El archivo XML de recursos globales activo* (seleccionado en el cuadro de diálogo "Administrar recursos globales"). El archivo XML de recursos globales activo se puede cambiar en todo momento y la definiciones de recursos globales del nuevo archivo activo reemplazan inmediatamente las del archivo previo.

  Por tanto, del archivo XML de recursos globales activo depende: (i) qué recursos globales se pueden asignar y (ii) qué recursos globales se pueden consultar (qué recursos globales aparecen en el archivo XML de recursos globales).

- *La configuración activa* (**Herramientas | Configuración activa** o en la barra de herramientas Recursos globales). El comando **Herramientas | Configuración activa** y la barra de herramientas Recursos globales enumeran las configuraciones de todos los alias.

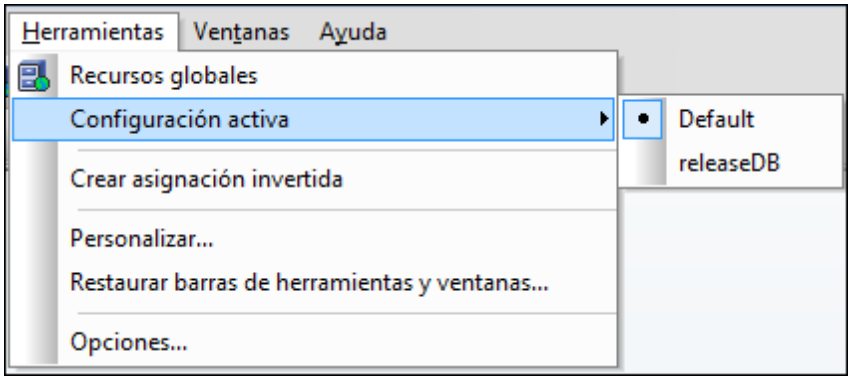
  La configuración seleccionada es la configuración activa en la aplicación. Es decir, allá donde se use un recurso global (o alias), se cargará el recurso asociado a la configuración activa de cada alias.

  La configuración activa se aplica a todos los alias que estén en uso. Si un alias no tiene una configuración que se llame como la configuración activa, entonces se usa la configuración predeterminada (`default`) de ese alias. La configuración activa no tiene relevancia a la hora de asignar recursos, solamente a la hora de utilizarlos.

Cambiar de recurso / configuración

Puede cambiar de recurso seleccionando otra configuración. Esto puede hacerse de dos maneras:

- Pase el cursor del mouse por el comando **Herramientas | Configuración activa** y aparece un submenú que enumera todas las configuraciones disponibles en el archivo XML de recursos globales activo. Haga clic en la configuración que desea utilizar.



- Haga clic en la flecha del cuadro combinado de la barra de herramientas Recursos globales. En la lista desplegable seleccione la configuración que desea usar. La barra de herramientas Recursos globales se puede activar/desactivar (mostrar/ocultar) en la pestaña *Barras de herramientas* del cuadro de diálogo Personalizar (**Herramientas | Personalizar**).

 © 2014 Altova GmbH

Altova MapForce 2014

Archivos de entrada/salida dinámicos por componente

13 Archivos de entrada/salida dinámicos por componente

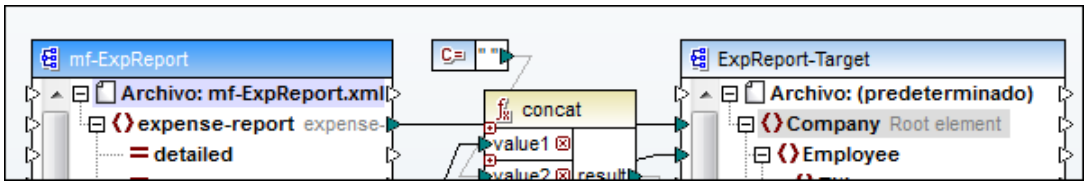
MapForce es capaz de procesar varios archivos de entrada/salida por componente y, por tanto, puede procesar todos los archivos de un directorio o de un grupo de archivos del directorio, usando caracteres de comodín en el componente de origen.

El ejemplo del tutorial explica cómo un componente de origen procesa dos archivos XML de entrada y cómo el componente de destino genera dos documentos XML diferentes.

Puede definir varios archivos de entrada/salida para este tipo de componentes:

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

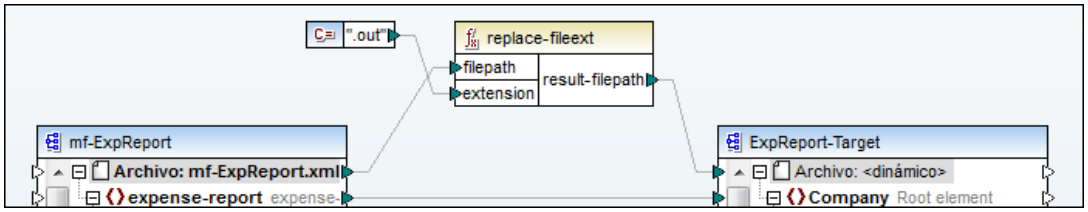
Observe el elemento **`Archivo:`** situado al principio de los componentes de esta asignación:



- El elemento **`Archivo:mf-ExpReport.xml`** del componente **mf-ExpReport** de la imagen anterior, por ejemplo, indica el archivo XML de entrada. Este elemento aparece automáticamente cuando se asigna un archivo XML de instancia a un esquema XML. (Si se definió también un archivo de salida, el nombre del archivo de salida aparece también en este elemento.)
- El elemento **`Archivo: (predeterminado)`** del componente **ExpReport-Target** indica que cuando se insertó el componente de esquema XML de destino, no se le asignó ningún archivo de instancia de salida. Es decir, el campo *Archivo XML de salida* está vacío. Por tanto, se usará un valor predeterminado cuando se ejecute la asignación.

La característica de nombres de archivo dinámicos se activa cuando al elemento **`Archivo:`** se asigna una cadena de texto que contiene un nombre de archivo. Si el componente se utiliza como componente de entrada, el nombre de archivo puede llevar comodines. Para más información, consulte el apartado sobre rutas de acceso relativas.

- El elemento **`Archivo: <dinámico>`** aparece cuando hay una conexión con el elemento **`Archivo:`** (es decir, ahora se admite el uso de varios archivos)
- La función **`replace-fileext`** convierte la extensión `.xml` en la extensión `.out` en los archivos de destino dinámicos.



 © 2014 Altova GmbH

Lenguajes de programación de MapForce compatibles con el uso de archivos dinámicos y comodines:

Lenguaje de destino	Nombre de archivo de entrada dinámico	Admite el uso de comodines en nombres de archivo de entrada	Nombre de archivo de salida dinámico
XSLT 1.0	*	no compatible con XSLT 1.0	no compatible con XSLT 1.0
XSLT 2.0	*	*(1)	*
XQuery	*	*(1)	no compatible con XQuery
C++	*	*	*
C#	*	*	*
Java	*	*	*
BUILTIN	*	*	*

* compatible

(1) Usa la función **`fn:collection`**. La implementación en los motores XSLT 2.0 y XQuery de Altova resuelve los comodines. Puede que otros motores se comporten de manera diferente.
Los comodines * y ? se resuelven cuando se introducen en el cuadro de diálogo "Configuración del componente" y también cuando se asigna una cadena al nodo **`Archivo:`**.

Para transformar código XSLT 1.0/2.0 así como código XQuery con el motor RaptorXML Server, consulte el apartado Generar código XSLT 1.0/2.0 y el apartado Generar código XQuery 1.0.

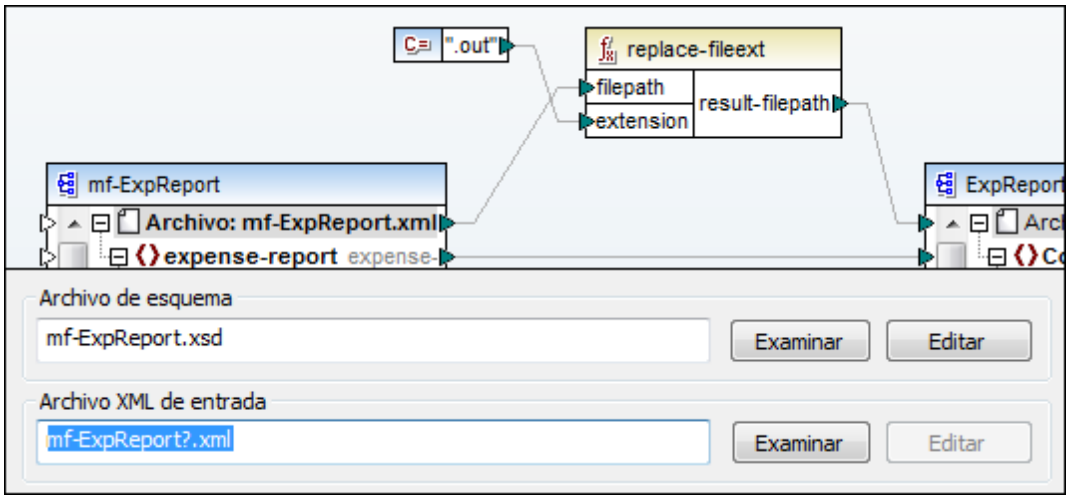
13.1 Nombres de archivo dinámicos: entrada/salida

Si asigna nombres de archivo de forma dinámica dentro de la asignación, podrá:

- generar una aplicación de asignación de datos donde los nombres de archivo de entrada/salida se definen en tiempo de ejecución
- pasar un grupo de archivos a otro formato
- dividir un archivo (o BD) de gran tamaño en varios archivos más pequeños
- unir varios archivos en un solo archivo de mayor tamaño (o cargarlos en una BD)

Para procesar varios archivos de entrada tiene dos opciones:

- Escriba una ruta de acceso de archivo con comodines (***** o **?**) en el campo *Archivo de entrada* en el cuadro de diálogo "Configuración del componente". Se procesarán todos los archivos que coincidan. El ejemplo de la imagen siguiente usa el carácter de comodín **?** en el campo *Archivo XML de entrada* para asignar todos los archivos que empiezan por `mf-ExpReport` seguidos de un carácter. En la carpeta `...\Tutorial` hay dos archivos que coinciden con este criterio.



- Asigne 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.

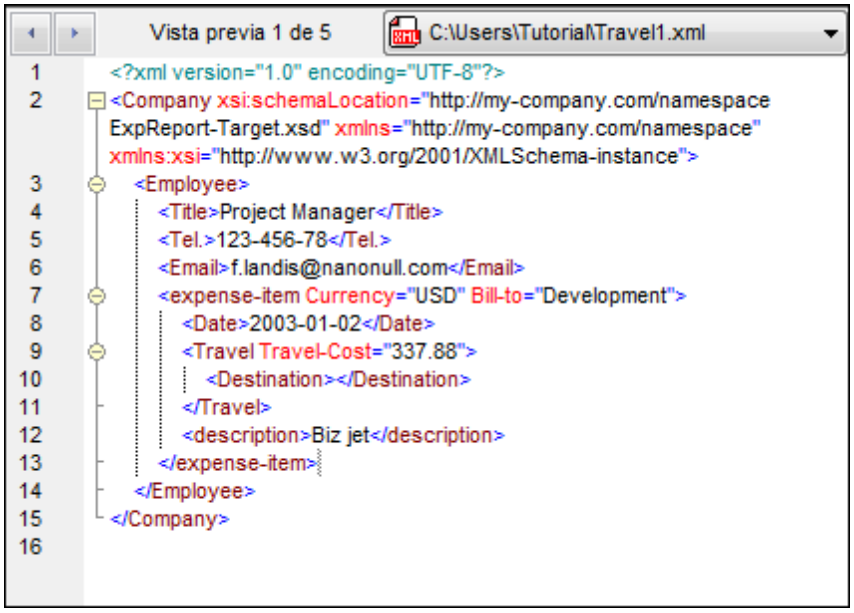
  Puede suministrar una secuencia de nombres de archivo mediante:

  - un archivo XML
  - un archivo de texto (CSV o FLF), etc.
  - campos de BD
  - una hoja de Excel, etc.

Vista previa de asignaciones con entradas/salidas dinámicas

Al pasar al panel **Resultados** aparece el resultado de la asignación en una ventana de vista previa. Si la asignación produce varios archivos de salida, como en el ejemplo siguiente (*Vista previa 1 de 5*), cada archivo tiene su propia pantalla de vista previa en el panel **Resultados**. Haga clic en los botones de flecha para ver todos los archivos de salida.

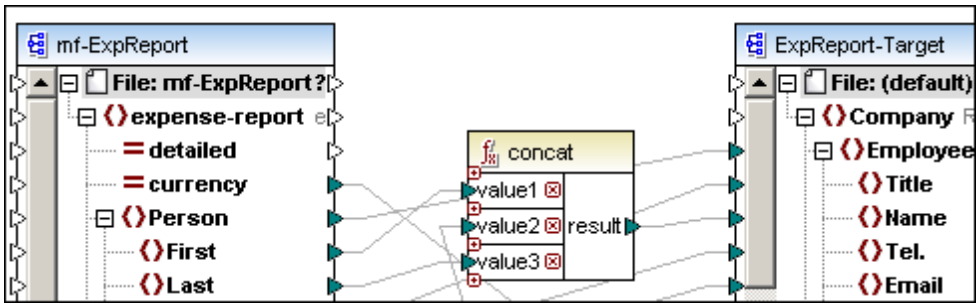
 © 2014 Altova GmbH



Haga clic en el icono **Guardar todos los resultados generados** para guardar todos los archivos de salida.

Varias entradas y una sola salida: combinación de archivos de entrada

Si la asignación tiene varios archivos de entrada estos se pueden combinar y unir en un **solo** archivo de salida. Esto ocurre si eliminamos la conexión **entre** los dos elementos **`Archivo:`** y el componente de origen sigue teniendo acceso a varios archivos, p. ej. por medio del comodín **?**. A diferencia del componente de origen, el componente de destino no puede admitir varios archivos de entrada. Por tanto, los archivos de entrada se anexan en el documento de destino.



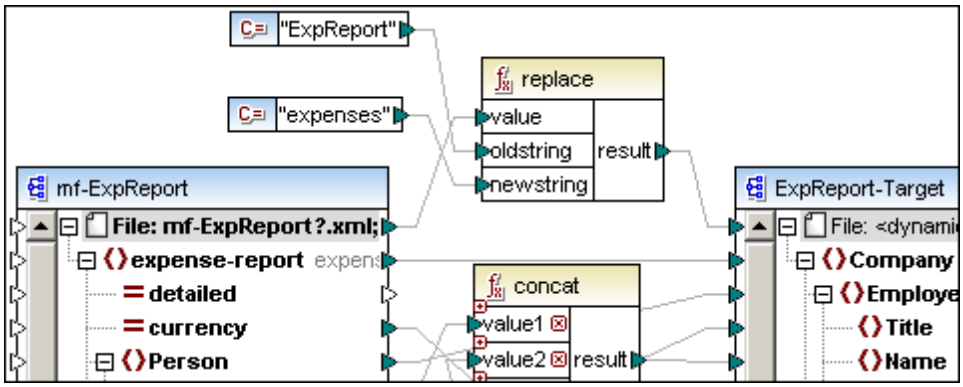
Varias entradas / varias salidas

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

El nombre de ruta de acceso completo del archivo que está en procesamiento se obtiene conectando el icono de salida del nodo **`Archivo:`**, p. ej. con la función **`concat`** o **`replace`** (con las que podemos añadir una extensión de archivo nueva).

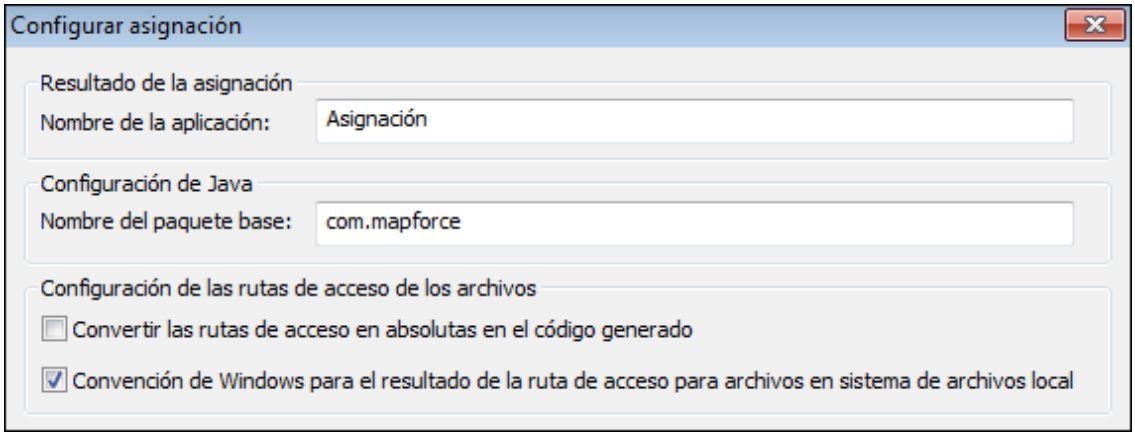
Nota: recomendamos no conectar los nodos **`Archivo:`** **directamente** sin usar funciones de procesamiento. Si lo hace, los archivos de entrada se sobrescribirán cuando se ejecute la

asignación. Hay varias funciones que le ayudarán a cambiar el nombre de los archivos de salida, p. ej. la función **`replace`** (*imagen siguiente*).



En el ejemplo de la imagen anterior, el nombre de los archivos de salida será `mf-expenses1.xml` y `mf-expenses2.xml`.

Además, con la opción de menú **Archivo | Configurar asignación** puede configurar las rutas de acceso de archivo utilizadas en todo el proyecto de asignación.



La casilla *Convención de Windows para el resultado de la....* garantiza el uso de la conveción de Windows para rutas de acceso. Si genera código XSLT2 (y XQuery), el nombre de archivo que está en procesamiento se recupera de forma interna usando la función **`document-uri`**, que devuelve una ruta de acceso de tipo `archivo://URI` para archivos locales.

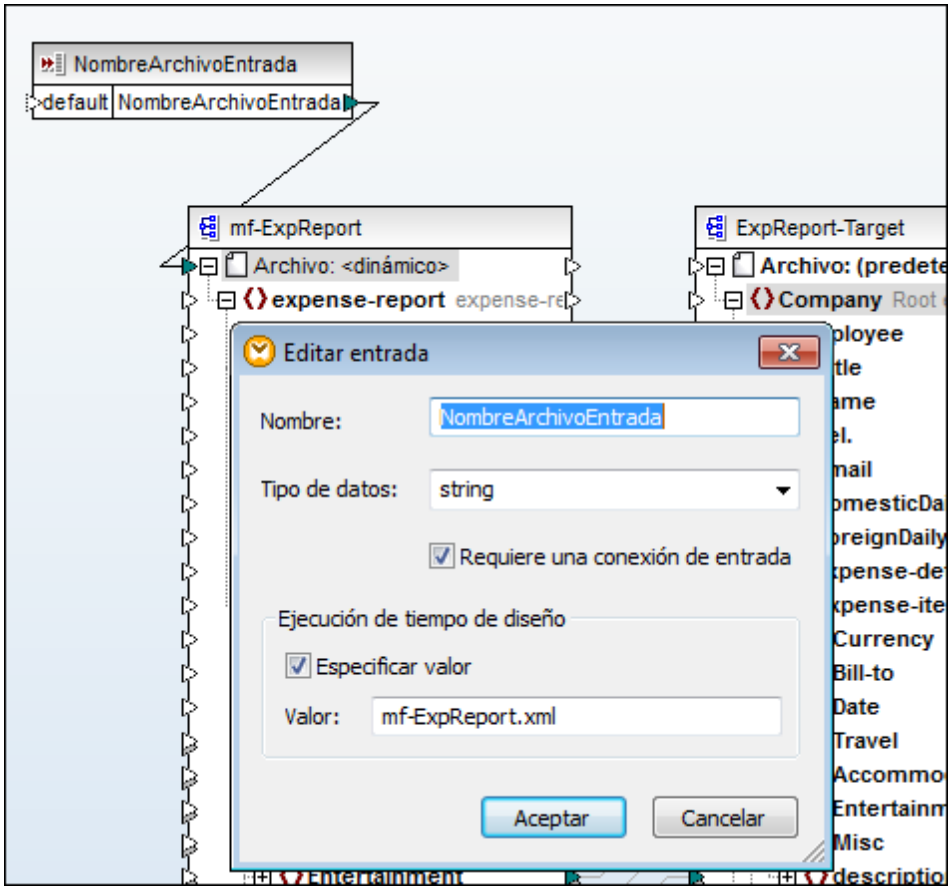
Si marca esta casilla, las rutas de acceso de tipo `archivo://URI` se convierten automáticamente en una ruta de archivo de Windows completa (p. ej. `C:\...`) para simplificar el resto del procesamiento.

 © 2014 Altova GmbH

13.2 Nombres de archivo dinámicos como parámetros de entrada

En MapForce puede crear componentes de entrada especiales que funcionen como **parámetros** en la ejecución de la línea de comandos de la asignación compilada. Este tipo concreto de componente de entrada no se puede usar **dentro** de una función definida por el usuario.

Para procesar un archivo usando un parámetro de entrada en tiempo de ejecución:
Para definir el nombre y la ruta de acceso del archivo en tiempo de ejecución, conecte un componente parámetro de entrada al icono de entrada del nodo **`Archivo:`** del componente de origen. Dependiendo de las demás conexiones a otros elementos, esta conexión definirá el nombre del archivo de entrada/salida.



La entrada `mf-ExpReport.xml` del campo *Valor:* solamente se usa para la vista previa del panel **Resultados**. No tiene efecto en los valores de parámetro utilizados durante la ejecución del código desde la línea de comandos. Para más información consulte el apartado Parámetros de entrada, invalidaciones y parámetros de la línea de comandos.

Después de generar y compilar el código, puede suministrar el nombre de archivo para la asignación usando esta línea de comandos:
mapping.exe /NombreArchivoEntrada Nombredelarchivo.xml.

Donde:

- **/NombreArchivoEntrada** es el nombre del primer parámetro
- **Nombredelarchivo.xml** es el segundo parámetro, es decir, el nombre de archivo

dinámico que debe utilizarse cuando la aplicación se ejecute desde la línea de comandos.

© 2014 Altova GmbH

13.3 Crear varios archivos XML a partir de un solo XML de origen

En la imagen siguiente aparece el contenido del archivo XML de origen `mf-ExpReport.xml`, disponible en la carpeta `...\MapForceExamples\Tutorial`. Se trata del informe de gastos del empleado Fred Landis y contiene cinco partidas de gastos de tipos diferentes. Para este ejemplo usamos el archivo `Tut-ExpReport-dyn.mfd`, disponible en la carpeta `...\Tutorial`.

Objetivo:
Generar un archivo XML diferente por cada partida de gasto (en total 5 archivos XML).

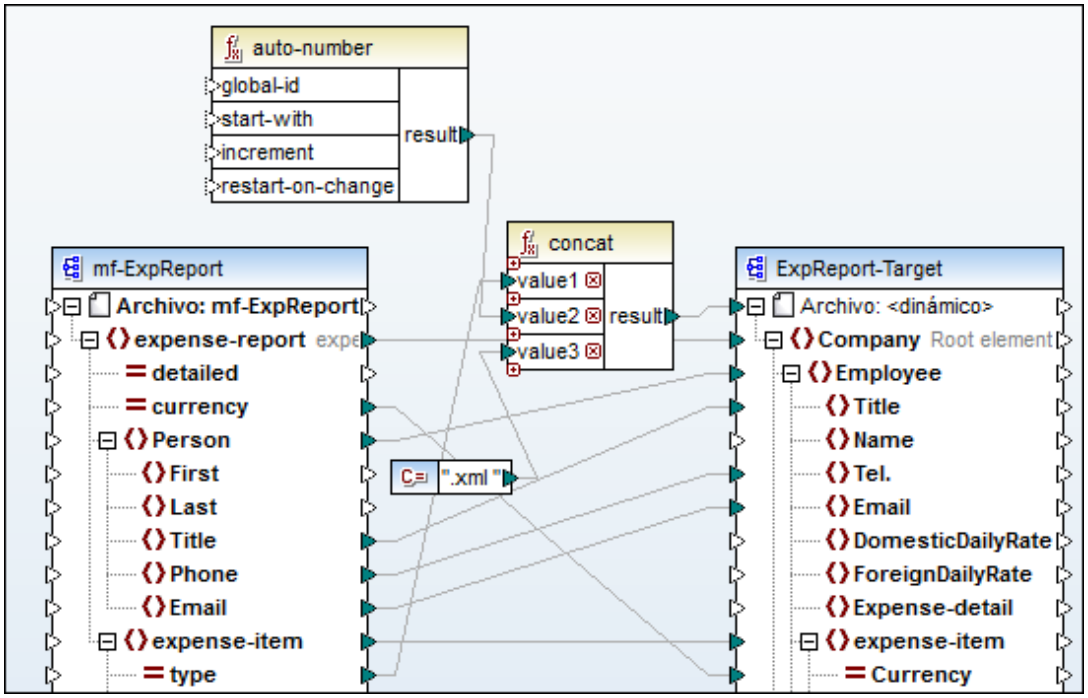
Person	
First	Fred
Last	Landis
Title	Project Manager
Phone	123-456-78
Email	f.landis@nanonull.com

expense-item (5)

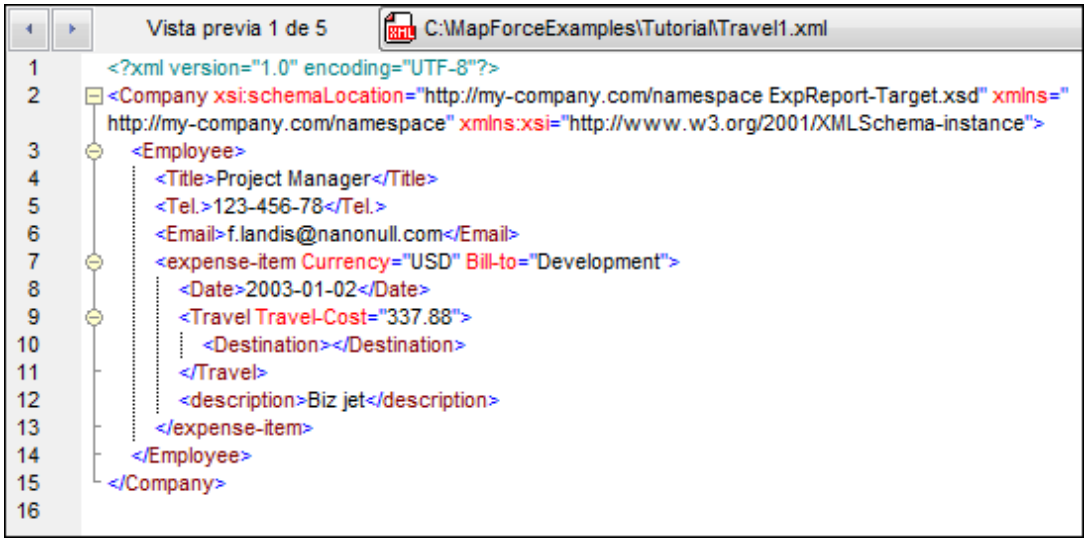
	type	expto	Date	Travel	Lodging
1	Travel	Development	2003-01-02	Travel Trav-cost=337.88	
2	Lodging	Sales	2003-01-01		Lodging
3	Travel	Accounting	2003-07-07	Travel Trav-cost=1014.22	
4	Travel	Marketing	2003-02-02	Travel Trav-cost=2000	
5	Meal	Sales	2003-03-03		

Puesto que el atributo **`type`** define el tipo de partida de gasto, este será el elemento que usaremos para separar los datos del archivo de origen:

1. Primero inserte la función **`concat`** y después cree una constante (haciendo clic en el icono correspondiente de la barra de herramientas).
2. Escriba `*.xml` como valor de cadena en el componente de constante.
3. Inserte la función **`auto-number`** del grupo de funciones **core | generator functions**.



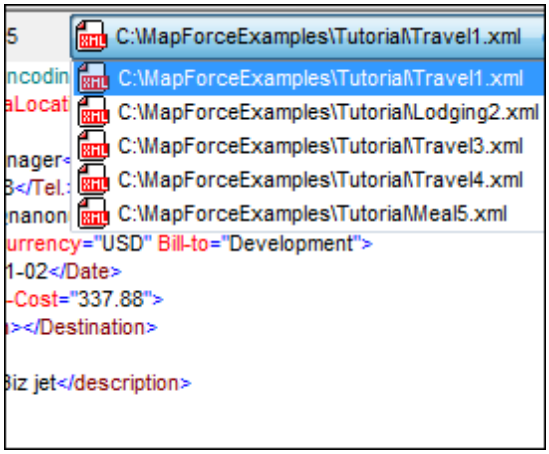
4. Cree las conexiones que aparecen en la imagen (entre **`type`** y **`value1`**, entre **`auto-number`** y **`value2`** y entre la constante y **`value3`**).
5. Conecte el parámetro de salida **`result`** de la función **`concat`** con el elemento **`Archivo:`** del componente de destino. Observe cómo el elemento pasa a llamarse **`Archivo: <dinámico>`**.
6. Defina el resto de conexiones.
7. Abra el panel **Resultados** para ver el resultado de la asignación.



Cada archivo XML de salida aparece en una pantalla diferente de la vista previa.

8. Para ver los demás archivos generados, haga clic en la flecha del cuadro combinado.

 © 2014 Altova GmbH



También puede pasar de un archivo a otro con los botones de flecha situados en la barra de herramientas de la vista previa.

Nota:

- El atributo **`type`** suministra la primera parte del nombre de archivo, p. ej. **`Travel`**.
- La función **`auto-number`** suministra el número de archivo en incrementos (la opción predeterminada es empezar por=1 y aumentar=1), con lo que obtenemos **`Travel1`**.
- El componente de **constante** suministra la extensión de archivo, es decir, `.xml`, con lo que para el primer archivo obtenemos `Travel1.xml`.
- Haga clic en el icono **Guardar todos** para guardar todos los archivos directamente desde el panel **Resultados**, sin necesidad de generar código.

13.4 Crear varios archivos XML a partir de una tabla

En la imagen siguiente aparece el contenido del archivo de base de datos `altova.mdb`, disponible en la carpeta `....\MapForceExamples\Tutorial`. La base de datos contiene cuatro tablas, una de las cuales (la tabla Person) contiene 21 registros personales. Para este ejemplo usamos el archivo `PersonDB-dyn.mfd`, disponible en la carpeta `...\Tutorial`.

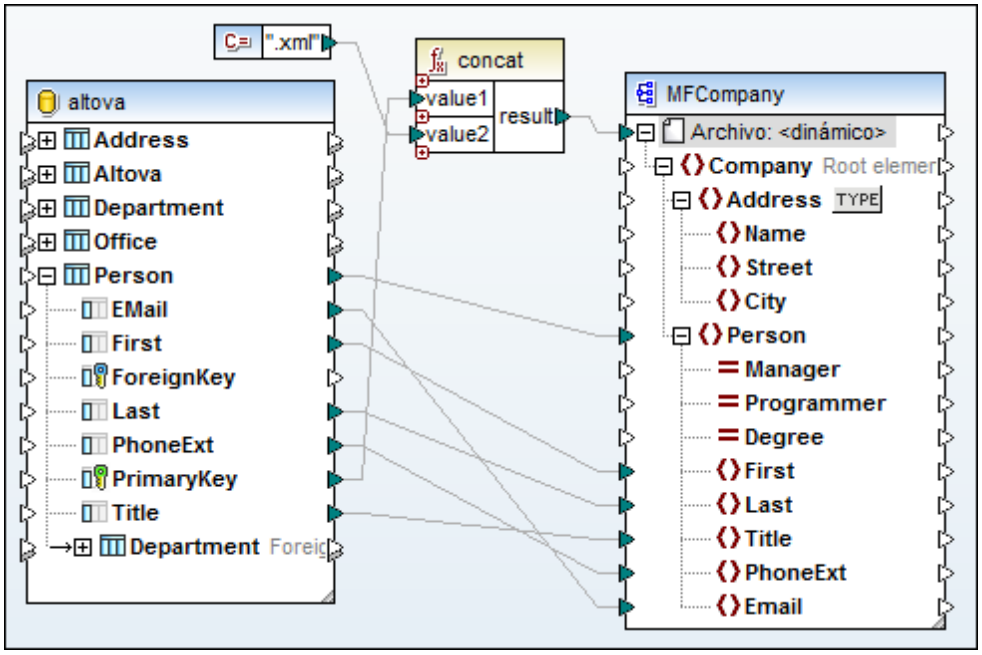
Objetivo:
Generar un archivo XML por cada registro de la tabla Person (*imagen siguiente*).

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

Puesto que el campo `PrimaryKey` define las personas de la tabla, este es el elemento que usaremos para separar los datos de la BD de origen en archivos diferentes.

1. Inserte una función **`concat`** y una constante.
2. Escriba `*.xml` como valor de cadena del componente de constante.

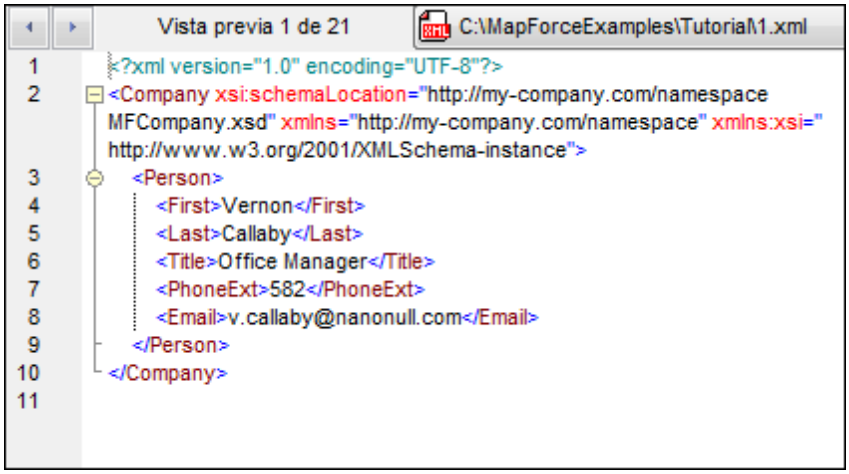


3. Cree las conexiones que aparecen en la imagen anterior (entre **`PrimaryKey`** y **`value1`** y entre la **constante** y **`value2`**).
4. Conecte el parámetro de salida **`result`** de la función **`concat`** con el elemento **`Archivo:`** del componente de destino. Observe cómo el elemento pasa a llamarse **`Archivo:`**

 © 2014 Altova GmbH

<dinámico>.

5. Defina el resto de conexiones.
6. Abra el panel **Resultados** para ver el resultado de la asignación.



Cada registro de la tabla generó un archivo XML y cada archivo XML de salida aparece en una pantalla diferente de la vista previa (*imagen anterior*).

También puede pasar de un archivo a otro con los botones de flecha situados en la barra de herramientas de la vista previa.

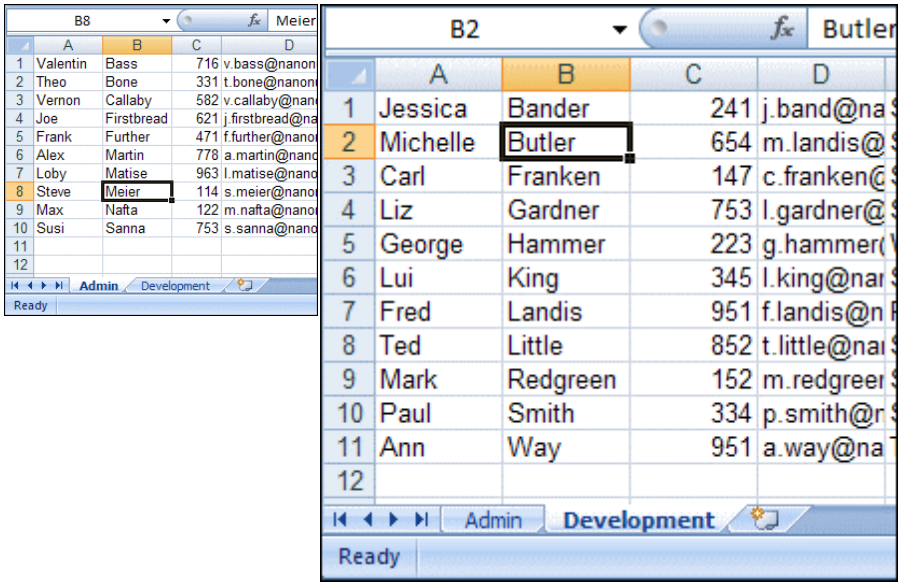
Nota:

- El campo **PrimaryKey** suministra la primera parte del nombre de archivo, p. ej. **1**.
- El componente de **constante** suministra la extensión del archivo (`.xml`), por eso el primer archivo se llama **1.xml**.
- Haga clic en el icono **Guardar todos** para guardar todos los archivos directamente desde el panel **Resultados**, sin necesidad de generar código.

13.5 Crear varios archivos XML a partir de Excel

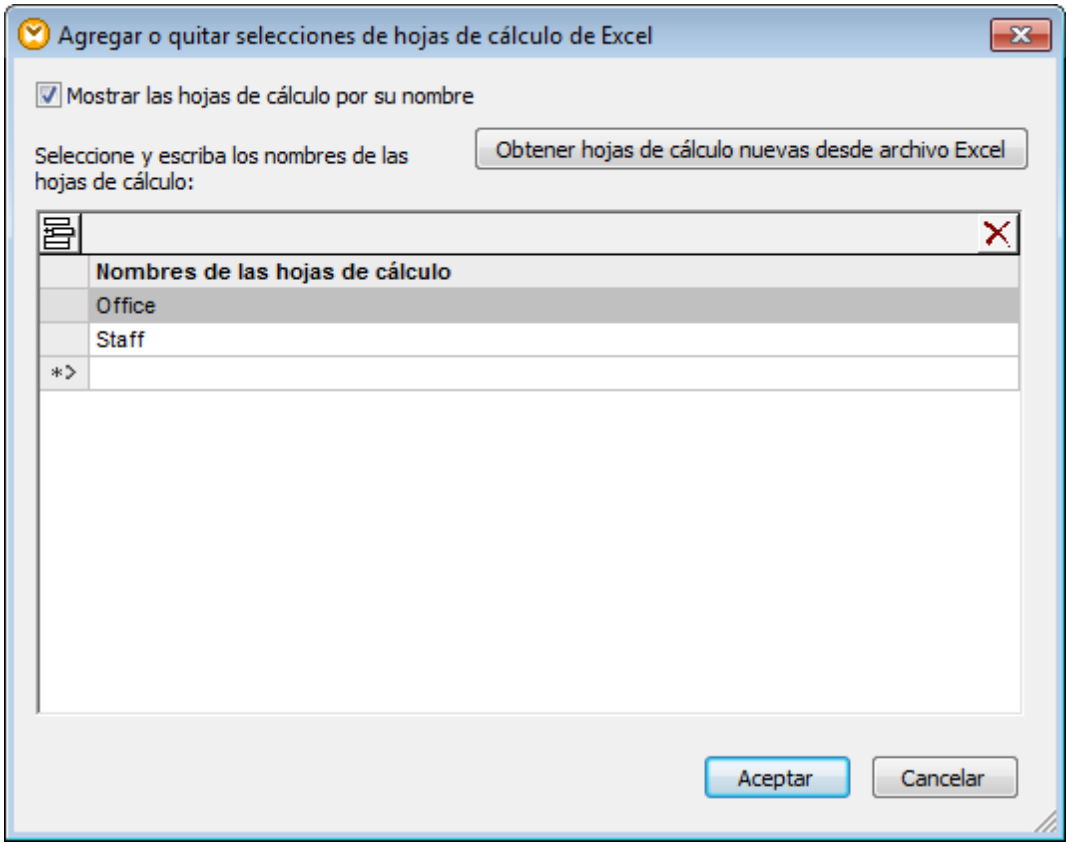
En la imagen siguiente aparece el contenido del archivo `altova.xlsx`, disponible en la carpeta `...\MapForceExamples\Tutorial`. Este archivo incluye dos hojas de cálculo: la hoja Admin, con 10 filas de datos, y la hoja Development, con 11 filas. Para este ejemplo usamos el archivo `Excel-Mapping-dyn.mfd`, disponible en la carpeta `...\Tutorial`.

Hoja de cálculo Admin Hoja de cálculo Development



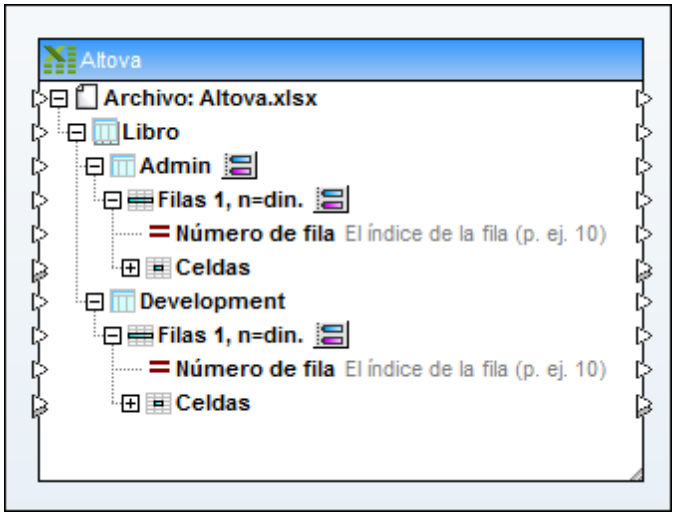
MapForce puede mostrar y asignar componentes Excel de dos formas diferentes, dependiendo de la configuración del componente. La configuración predeterminada aparece en la imagen siguiente. La casilla *Mostrar las hojas de cálculo ordenadas por nombre* se activa por defecto al insertar el componente Excel.

 © 2014 Altova GmbH

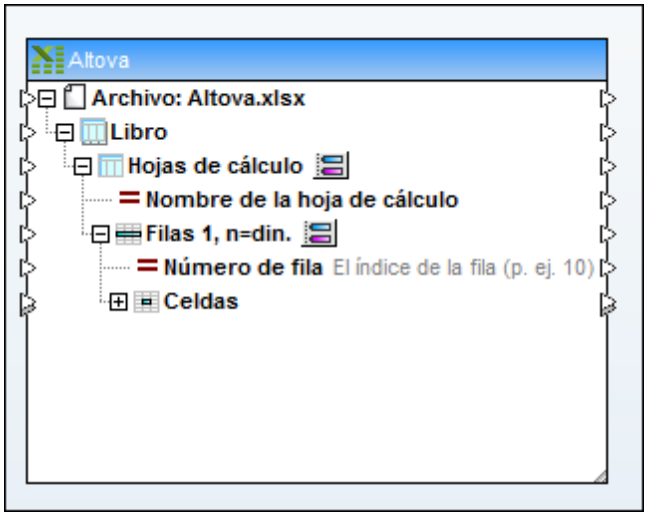


Para acceder al libro como si se tratase de una sola hoja de cálculo:

1. Haga clic en el icono situado junto al nodo **`Admin`** del componente Excel.



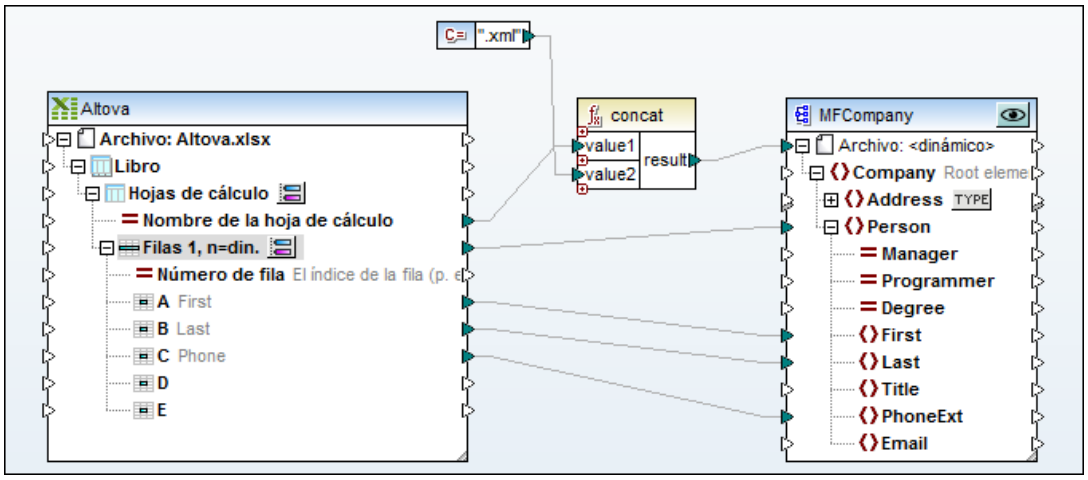
2. Desactive la casilla *Mostrar las hojas de cálculo ordenadas por nombre*.
   El nombre de las hojas de cálculo desaparece (*imagen siguiente*), pero en su lugar aparece un nodo llamado **`Nombre de la hoja de cálculo`**.



Objetivo nº1:
Generar un archivo XML por cada hoja de cálculo que contenga datos personales.

Puesto que el elemento **`Nombre de la hoja de cálculo`** define las diferentes hojas de cálculo del libro, este es el elemento que usaremos para separar los datos del libro de origen en varios archivos.

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



2. Cree las conexiones que aparecen en la imagen anterior (es decir, entre **`Nombre de la hoja de cálculo`** y **`value1`** y entre la **constante** y **`value2`**).
3. Conecte el parámetro **`result`** de la función **`concat`** con el elemento **`Archivo:`** del componente de destino. Observe cómo el elemento pasa a llamarse **`Archivo: <dinámico>`**.
4. Defina el resto de conexiones.
5. Abra el panel **Resultados** para ver el resultado de la asignación.

 © 2014 Altova GmbH

Vista previa 1 de 2 C:\Altova\MapForce2013\Admin.xml

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Company xsi:schemaLocation="http://my-company.com/namespace
   C:/Altova/MapForce2013/MapForceExamples/Tutorial/MFCompany.xsd"
   xmlns="http://my-company.com/namespace" xmlns:xsi="
   http://www.w3.org/2001/XMLSchema-instance">
3    <Person>
4      <First>Valentin</First>
5      <Last>Bass</Last>
6      <PhoneExt>716</PhoneExt>
7    </Person>
8    <Person>
9      <First>Theo</First>
10     <Last>Bone</Last>
11     <PhoneExt>331</PhoneExt>
12   </Person>
13   <Person>
14     <First>Vernon</First>
15     <Last>Callaby</Last>
16     <PhoneExt>582</PhoneExt>
17   </Person>
```

Cada hoja de cálculo generó un archivo XML y cada archivo XML de salida aparece en una pantalla diferente de la vista previa (*imagen anterior*).

También puede pasar de un archivo a otro con los botones de flecha situados en la barra de herramientas de la vista previa.

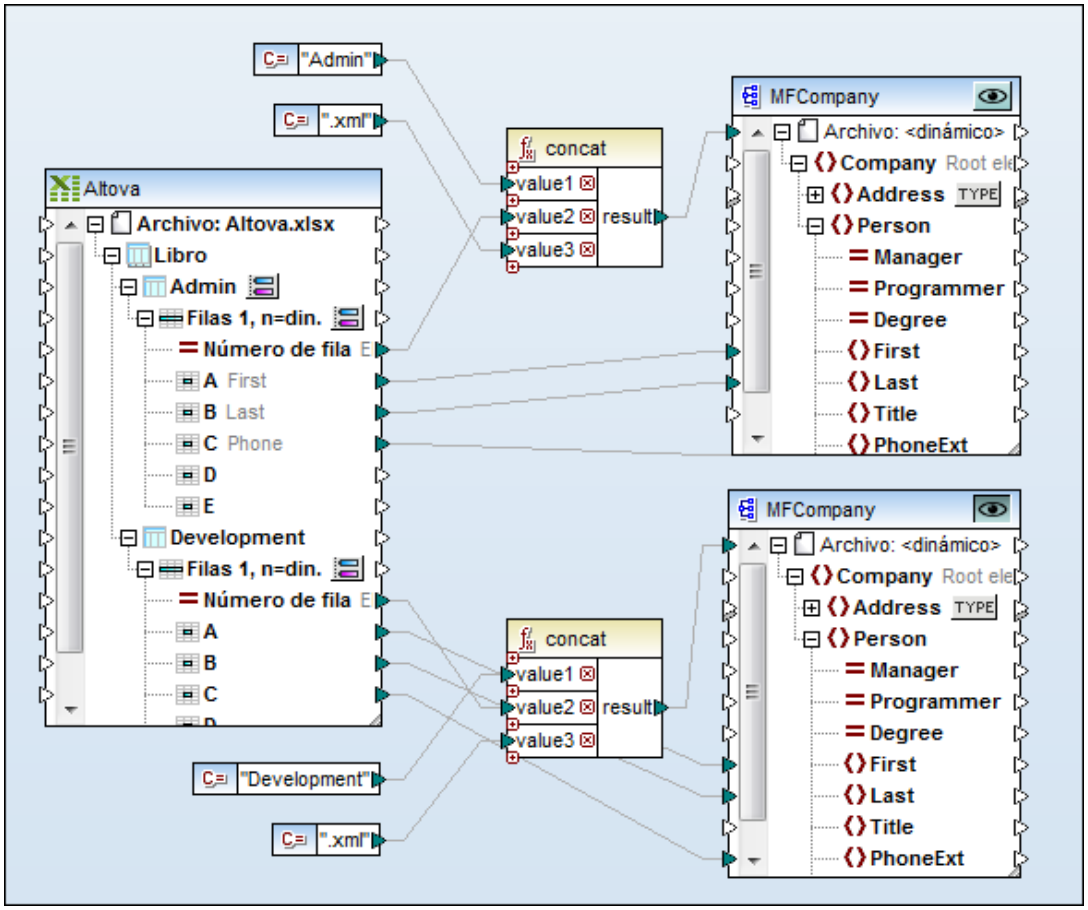
Nota:

- El nodo **`Nombre de la hoja de cálculo`** suministra la primera parte del nombre de archivo, p. ej. **`Admin`**.
- El componente de **constante** suministra la extensión del archivo (`.xml`), por eso el primer archivo se llama **`Admin.xml`**. Este primer archivo contiene todas las filas de esa hoja de cálculo.
- Haga clic en el icono **Guardar todos** para guardar todos los archivos directamente desde el panel **Resultados**, sin necesidad de generar código.

Objetivo nº2:
Generar un archivo XML por cada Persona que aparece en cada hoja de cálculo.

Para generar un archivo XML por cada persona de cada hoja de cálculo:

1. Haga clic en el icono situado junto al nodo **`Hojas de cálculo`** y en el cuadro de diálogo que parece marque la casilla *Mostrar las hojas de cálculo ordenadas por nombre.* Ahora pueden verse las dos hojas de cálculo del componente Excel: **`Admin`** y **`Development`**.



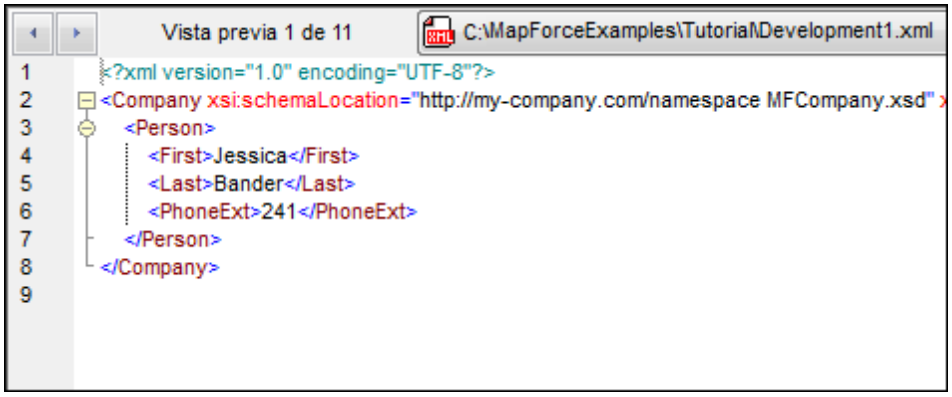
2. Inserte la función **`concat`** y una **constante** tal y como muestra la imagen anterior.
3. Inserte otro componente de esquema XML de destino (con el mismo nombre que el componente de destino actual) y cree las conexiones que muestra la imagen anterior.

   Puesto que el elemento **`Número de fila`** define las personas que aparecen en la hoja de cálculo (una persona en cada fila), este es el elemento que usaremos para separar los datos de cada hoja de cálculo en varios archivos.

Para el nodo **`Development`** del componente Excel **Altova**:

1. Cree las conexiones que aparecen en la imagen anterior (es decir, entre la **constante** `Development` y **`value1`**, entre **`Número de fila`** y **`value2`** y entre la **constante** `.xml` y **`value3`**).
2. Conecte el parámetro **`result`** de la función **`concat`** con el nodo **`Archivo:`** del componente de destino. Observe cómo el nodo pasa a llamarse **`Archivo: <dinámico>`**.
3. Defina el resto de conexiones.
4. Abra el panel **Resultados** para ver el resultado de la asignación.

 © 2014 Altova GmbH



Cada fila de la hoja de cálculo Development generó un archivo XML (es decir, 11 archivos XML en total) y cada archivo XML de salida aparece en una pantalla diferente de la vista previa (*imagen anterior*).

También puede pasar de un archivo a otro con los botones de flecha situados en la barra de herramientas de la vista previa.

Nota:

- La constante `Development` suministra la primera parte del nombre de archivo (`Development`).
- El nodo **`Número de fila`** suministra el número de la fila de la hoja de cálculo (p. ej. `1`).
- La constante `.xml` suministra la extensión de archivo (por eso obtenemos `Development1.xml`).
- Haga clic en el icono **Guardar todos** para guardar todos los archivos directamente desde el panel **Resultados**, sin necesidad de generar código.

Para ver el resultado de la hoja de cálculo Admin, haga clic en el botón de vista previa de ese componente de destino y después abra el panel **Resultados**. Las 10 filas de la hoja de cálculo Admin generaron 10 archivos XML de salida.

13.6 Rutas de acceso de archivo relativas y absolutas

Una ruta de acceso relativa es una ruta de acceso que no empieza con la letra de la unidad de disco, es decir, un nombre de archivo sin ruta de acceso. El contexto específico en el que se usa el nombre de archivo relativo define la ruta de acceso base. A partir de la versión 2010, MapForce se ocupa de los nombres de archivo relativos de otra manera pues, a partir de esa versión, MapForce es compatible con la asignación de nombres de archivo como datos dentro de una asignación.

Las versiones anteriores a MapForce 2010 guardaban las rutas de acceso de archivo como relativas al archivo MFD en caso de archivos situados en la misma carpeta que el archivo MFD o en carpetas descendientes y, cuando se abrían/cargaban estos archivos, las rutas de acceso se convertían en absolutas.

A partir de la versión 2010 de MapForce, todas las referencias a archivos externos, como a archivos XML de instancia o a esquemas XML, se almacenan tal y como se introducen en el cuadro de diálogo. De este modo, se pueden usar rutas de acceso relativas donde haga falta.

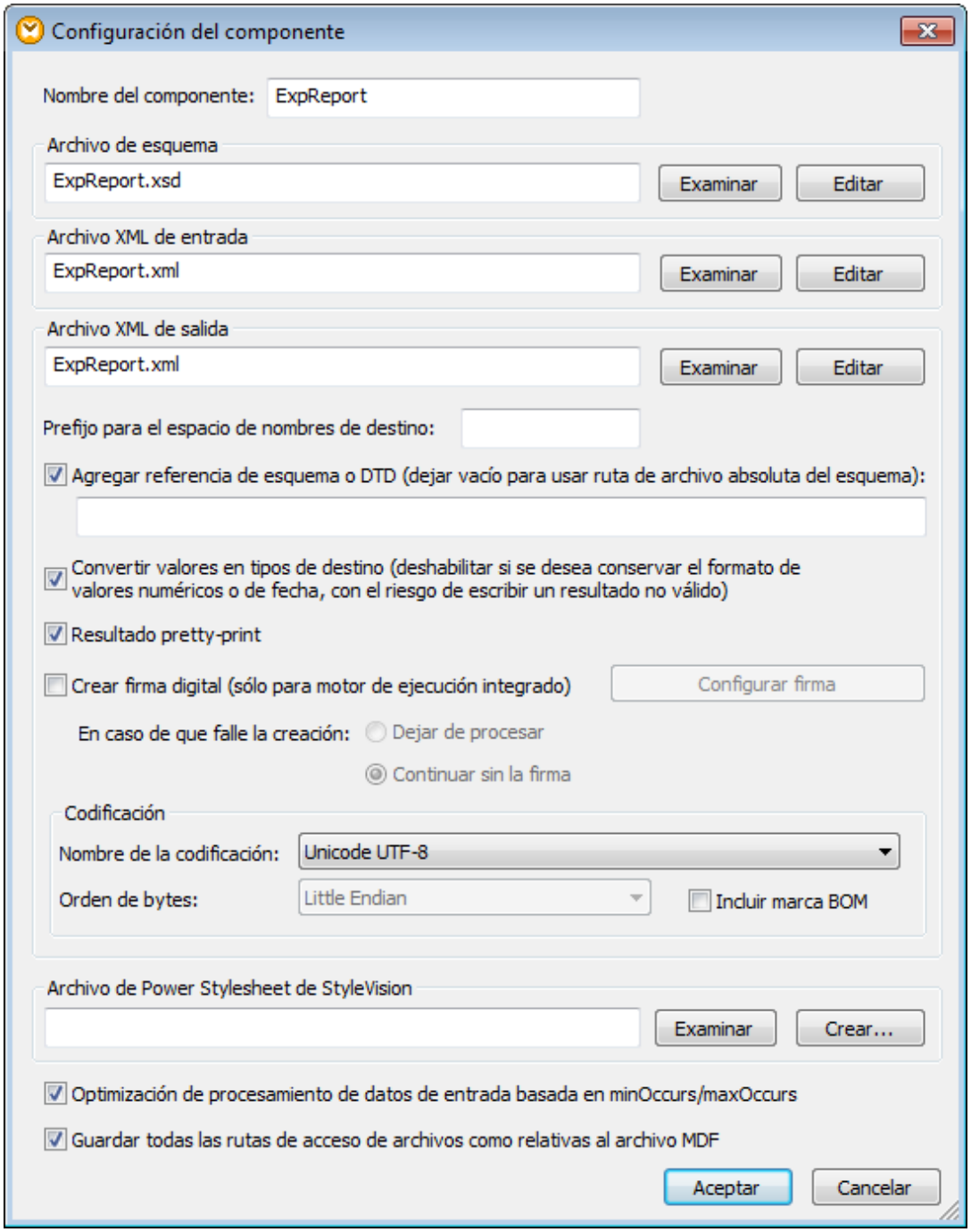
Guardar todas las rutas de acceso relativas en el archivo MFD
Esta nueva opción, disponible en los cuadros de diálogo de configuración de todos los componentes, sean del tipo que sean, guarda todas las rutas de acceso de archivo (del componente) como relativas a la ubicación del archivo MFD actual. En otras palabras, puede mover una asignación junto con todos los archivos relacionados a una ubicación nueva y las referencias a archivos se mantienen intactas.

Esto significa que:

- Las rutas de acceso absolutas se convertirán en relativas
- El directorio primario "**..**" también se insertará
- Al hacer clic en el comando **Guardar como...** las rutas de acceso de archivo (relativas al archivo MFD) se ajustarán a la nueva ubicación del archivo MFD

Nota: las rutas de acceso que hacen referencia a una unidad de disco no local o que usan una URL no se convertirán en rutas de acceso relativas.

 © 2014 Altova GmbH



Hay dos tipos diferentes de archivo a los que se hace referencia desde un archivo MFD:

- Los archivos de tipo esquema (esquemas XML, WSDL, archivos de configuración de FlexText, etc.) indicados en el campo *Archivo de esquema*.
- Los archivos de instancia indicados en los campo *Archivo xxx de entrada* o *Archivo xxx de salida*.

Archivos de tipo esquema

Los archivos de tipo esquema se usan para diseñar la asignación. Definen la **estructura** de los

archivos de instancia de entrada/salida asignados. Esta información se utiliza para presentar la jerarquía de elementos de los componentes de la asignación. En MapForce puede introducir y almacenar rutas de acceso relativas a los archivos de tipo esquema.

- Las rutas de acceso relativas a **archivos de tipo esquema** siempre **se resuelven como relativas al archivo MFD**.
- Si selecciona un esquema desde el cuadro de diálogo Abrir (p. ej. después de insertar un componente nuevo o al hacer clic en el botón **Examinar** del diálogo "Configuración del componente"), en el cuadro de texto siempre se inserta la ruta de acceso **absoluta**.
- Para establecer una ruta de acceso relativa, que también se almacenará en el archivo MFD, elimine la ruta de acceso del cuadro de texto o escriba el nombre de archivo o su ruta relativa. Esto ocurre de forma automática al guardar el archivo MFD si está activa la casilla *Guardar todas las rutas de acceso de archivos como relativas al archivo MDF*. También puede usar "**..**" para especificar la carpeta primaria del archivo MFD.
- Si guarda un archivo MFD que hace referencia a archivos del mismo directorio y después mueve todo el directorio a una ubicación nueva, las rutas de acceso **absolutas** almacenadas en el archivo MFD **no se actualizan**. Por tanto, los usuarios que utilicen sistemas de control de código fuente y directorios de trabajo diferentes deberían usar rutas de acceso relativas en estos casos.

Archivos de instancia y entorno de ejecución
El procesamiento de archivos de instancia se realiza en el código XSLT/XQuery generado o en la aplicación generada, así como en la vista previa de MapForce.

En la mayoría de los casos, no tiene sentido interpretar rutas de acceso relativas a archivos de instancia como relativas al archivo MFD porque puede que la ruta de acceso no exista en tiempo de ejecución (puesto que el código generado se puede implementar en otro equipo.) Los nombres de archivo **relativos** de archivos de instancia, por tanto, se resuelven en relación al **entorno de ejecución**:

Lenguaje de destino	Ruta de acceso base para el nombre de archivo de instancia relativo
XSLT/XSLT2	Ruta de acceso del archivo XSLT
XQuery	Ruta de acceso del archivo XQuery
C++, C#, Java	Directorio de trabajo cuando se ejecuta el código generado
Vista previa de MapForce Preview/ BUILTIN (o ejecución desde la API)	Ruta de acceso del archivo MFD
Ejecución de BUILTIN desde la línea de comandos	Indicada en la línea de comandos (directorio de destino)

El cuadro de diálogo "Configurar asignación" (**Archivo | Configurar asignación**) incluye una nueva casilla que garantiza la compatibilidad del código generado con archivos de asignación de MapForce (`.mfd`) creados en versiones anteriores a MapForce 2010: la casilla *Convertir las rutas de acceso en absolutas en el código generado*.

El estado de esta casilla se establece automáticamente dependiendo del tipo de archivo:

- la casilla se **activa** automáticamente cuando se abre o se crea un archivo de asignación

 © 2014 Altova GmbH

nuevo (en MapForce 2010 o superior).
Las rutas de acceso relativas para archivos de instancia de entrada/salida se escriben tal y como están en el código generado.

Esto permite implementar el código generado en un directorio diferente e incluso en un equipo diferente. Por eso, debe asegurarse de que los archivos a los que apuntan las rutas de acceso relativas estén disponibles en el entorno de tiempo de ejecución en la ubicación correcta.

- la casilla se **desactivada** automáticamente cuando se abre un archivo de asignación creado en una versión anterior a MapForce 2010.
  Las rutas de acceso relativas para los archivos de instancia de entrada/salida se convierten en absolutas (relativas al archivo MFD) antes de generarse el código. El resultado es el mismo que se obtendría al generar código con una versión previa de MapForce.

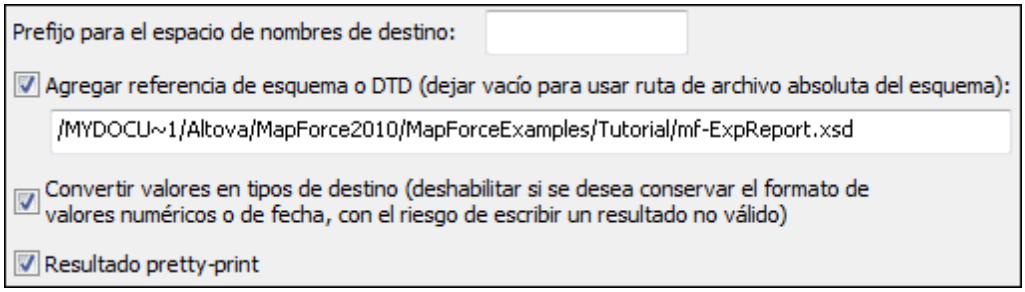
Recuerde que el nombre de archivo de instancia de origen también se utiliza para:

- Detectar el elemento raíz XML y el esquema referenciado
- Validar el archivo con el esquema seleccionado
- Leer nombres y columnas de hojas de cálculo de Excel
- Leer nombres de columnas y obtener una vista previa del contenido de archivos de texto (CSV o FLF)

Campo schemaLocation para archivos XML de destino

Las referencias a esquemas se pueden almacenar como relativas al archivo MFD. Asimismo, el archivo XML generado a partir del componente de destino suele estar en un directorio diferente. Por ello existe la posibilidad de introducir una ruta de acceso **schemaLocation** distinta para la **instancia** XML de destino, para que el archivo XML se pueda validar allí.

Esta ruta de acceso **schemaLocation** se define en el cuadro de texto situado bajo la casilla *Agregar referencia de esquema o DTD* del cuadro de diálogo "Configuración del componente". El mismo cuadro de texto está disponible para los componentes XBRL para hacer referencia al archivo de taxonomía.



La ruta del esquema asociado/referenciado introducida en este cuadro de texto se escribe en los archivos de instancia de destino generados, en su atributo `xsi:schemaLocation` o, si se usó una DTD, en la declaración DOCTYPE.

Para los archivos de destino XBRL, la referencia a la taxonomía se escribe en el atributo `href` del elemento `link:schemaRef`.

Nota: en este cuadro de texto también se puede introducir una URL, p. ej. http://mylocation.com/mf-expreport.xsd.

© 2014 Altova GmbH

Altova MapForce 2014

Variables intermedias

14 Variables intermedias

Las variables intermedias son un tipo de componente especial utilizado para solucionar problemas de asignación avanzada. Estas variables almacenan un resultado intermedio de la asignación, que se puede seguir procesando.

- Puede usar variables con todos los lenguajes excepto con XSLT 1.0
- Los resultados de las variables siempre son secuencias, es decir, una lista delimitada de valores, y también se pueden usar para crear secuencias.
- Las variables son componentes estructurales, con un nodo raíz, y no tienen asociadas instancias (archivos XML, etc.).
- Las variables permiten comparar los elementos de una secuencia con otros elementos de la misma secuencia.
- Las variables se pueden usar para generar secuencias intermedias. Los registros se pueden filtrar antes de ser transmitidos a un destino o después de la variable usando la función **`position`**, por ejemplo.

Diferencias entre variables y asignaciones en cadena

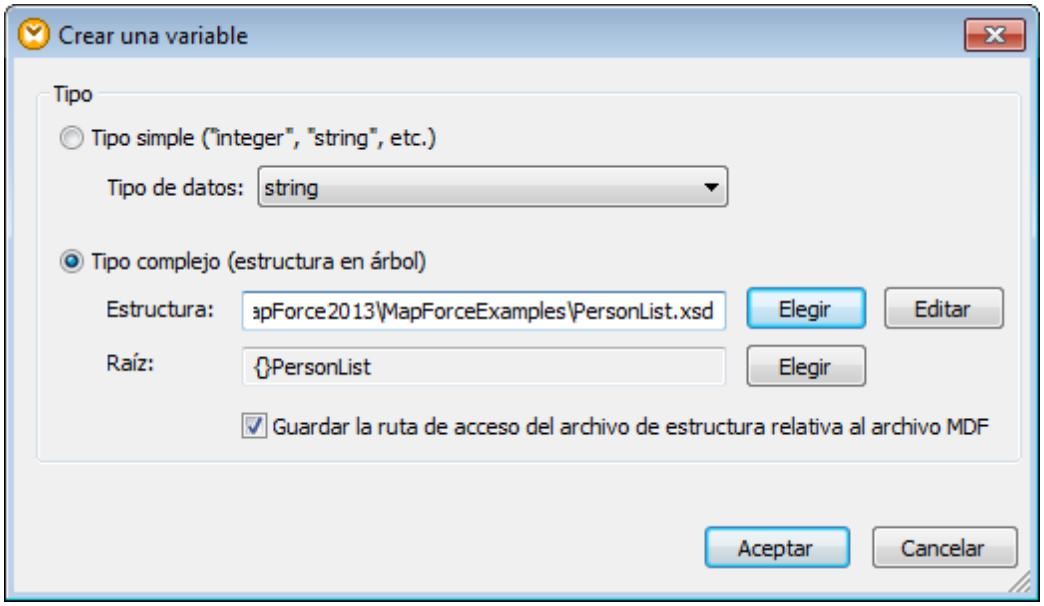
Asignaciones en cadena	Variables
Abarcan dos pasos totalmente independientes entre sí.	Se evalúan dependiendo del contexto/ámbito. Son controladas por una conexión **`compute-when`**
Cuando se ejecuta la asignación, los resultados intermedios se almacenan de forma externa en archivos XML	Cuando se ejecuta la asignación, los resultados intermedios se almacenan de forma interna (no se guardan en archivos físicos)
Se puede obtener una vista previa de los resultados intermedios en el panel Resultados	No es posible obtener una vista previa de los resultados de una variable

Para insertar variables intermedias:

Hay varias formas de insertar variables intermedias: desde el menú **Insertar**, haciendo clic en el icono o haciendo clic con el botón secundario en un icono de entrada/salida.

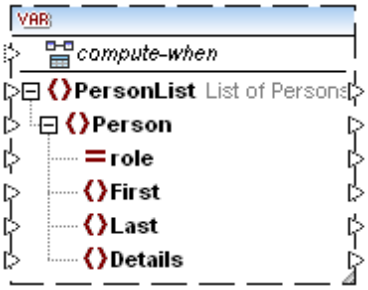
1. Seleccione **Insertar | Variable** o haga clic en el icono de la barra de herramientas. Aparece un cuadro de diálogo donde puede elegir si se inserta una variable de tipo simple o de tipo complejo.

 © 2014 Altova GmbH



2. Elija *Tipo complejo*.
3. Haga clic en el botón **Elegir** para seleccionar un esquema XML, por ejemplo, y seleccionar el elemento raíz.
4. Para terminar haga clic en **Aceptar** y se inserta la variable.

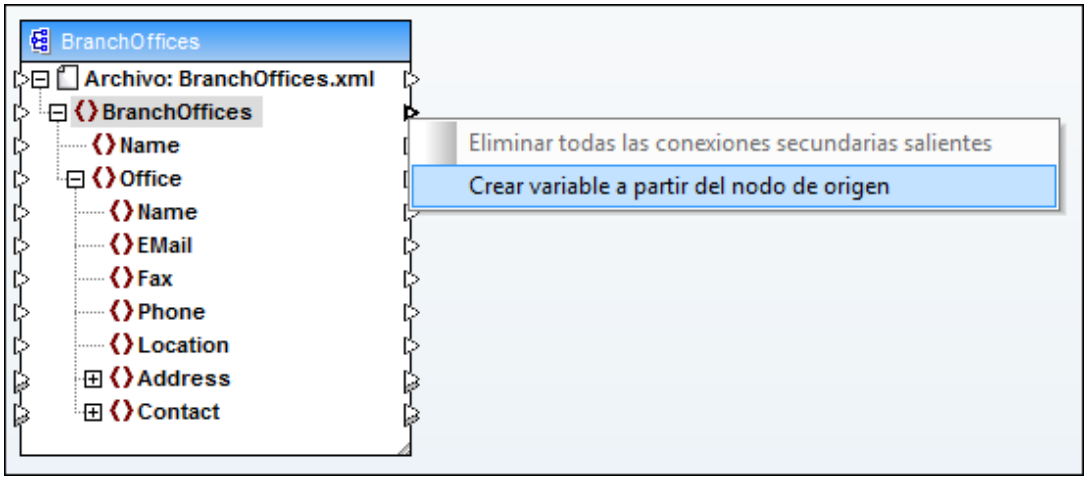
Ejemplo de variable compleja:



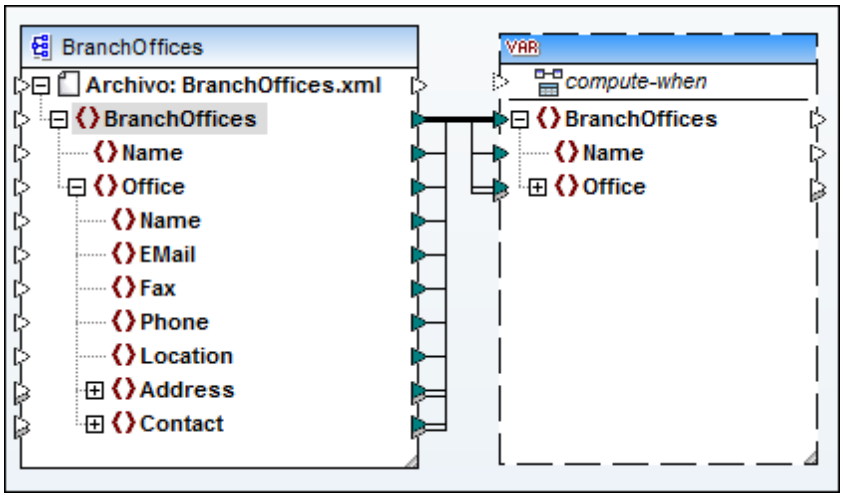
Ejemplo de variable simple (tienen un solo valor/nodo asignable, p. ej. una cadena de texto, un entero, etc. No obstante, el nodo **value** puede duplicarse.)

Más métodos para insertar variables:

- Haga clic con el botón secundario en el icono de **salida** de un componente y en el menú contextual seleccione **Crear variable a partir del nodo de origen**.

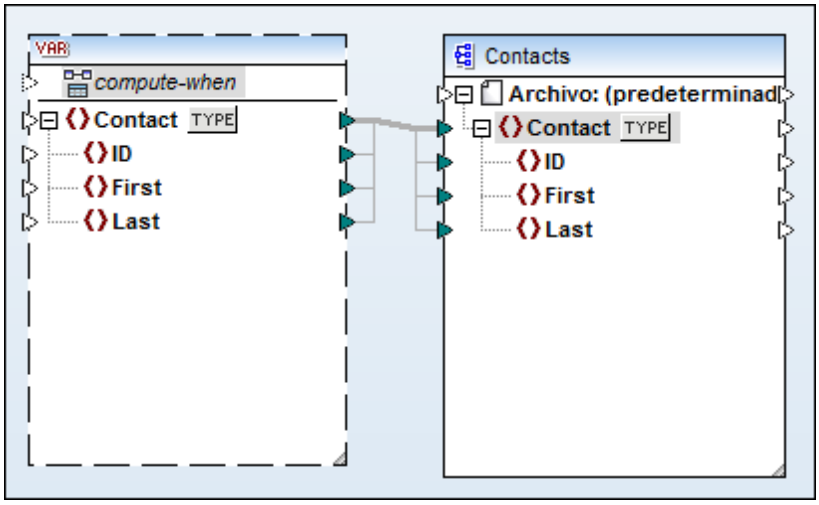


Esto crea una variable compleja usando el mismo esquema de origen y la variable se conecta automáticamente con todos los elementos del componente mediante una conexión de copia total.



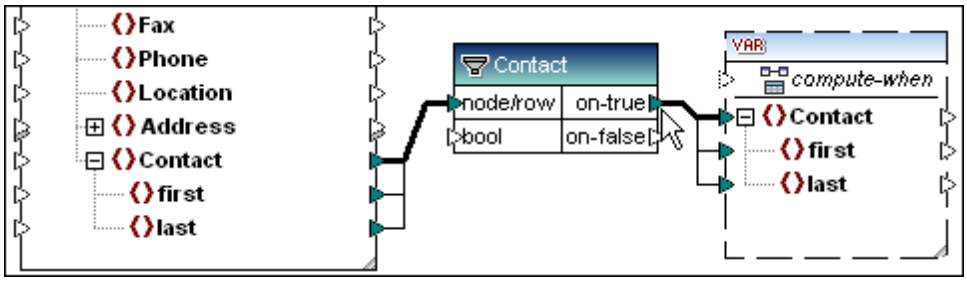
- Haga clic con el botón secundario en el icono de **entrada** de un componente de destino y en el menú contextual seleccione **Crear variable para el nodo de destino**.

 © 2014 Altova GmbH



Esto crea una variable compleja usando el mismo esquema como destino, con el elemento Contacto como nodo raíz, y la variable se conecta automáticamente con todos los elementos del componente mediante una conexión de copia total.

- Haga clic con el botón secundario en un icono de salida de un componente de filtrado (**`on-true`**/**`on-false`**) y en el menú contextual seleccione **Crear variable a partir del nodo de origen**.



Esto crea una variable compleja usando el esquema de origen y automáticamente se utiliza el elemento asignado al parámetro **`node/row`** del filtro como elemento raíz de la variable.

Compute-when

En las variables, el nodo de entrada **`compute-when`** sirve para controlar el ámbito de la variable. En otras palabras, define cuándo y con qué frecuencia se calcula el valor de la variable cuando se ejecuta la asignación. En muchos casos no hace falta conectar este nodo de entrada, pero puede ser esencial a la hora de sobrescribir el contexto predeterminado o para mejorar el rendimiento de la asignación.

Llegados a este punto es necesario mencionar el término ***subárbol***, que aquí utilizamos para denominar el conjunto formado por un nodo/elemento de un componente de destino y todos sus elementos descendientes (p. ej. un elemento `<Persona>` con sus secundarios `<Nombre>` y `<Apellido>`.)

Asimismo, el término ***valor de la variable*** hace referencia a los datos disponibles en el lado de

salida del componente de variable:

- En las variables simples, es una secuencia de valores atómicos cuyo tipo de datos se especifica en las propiedades del componente.
- En las variables complejas, es una secuencia de nodos raíz (del tipo especificado en las propiedades del componente), cada uno con todos sus nodos secundarios.

  La secuencia de valores atómicos (o nodos) también puede contener un solo elemento o incluso cero elementos. Esto depende de lo que esté conectado con el lado de entrada del componente variable y de si existen elementos primarios en los componentes de origen y destino.

Posibilidad nº1: `compute-when` sin conectar (configuración predeterminada)

Si **`compute-when` no está conectado** a ningún nodo de salida del componente de origen, el valor de la variable se calcula cada vez que se utilice **por primera vez** en un subárbol de **destino** (mediante una conexión directa entre la variable y un nodo del componente de destino o mediante una conexión indirecta a través de funciones). El mismo valor de variable se utiliza también para todos los nodos secundarios de destino que forman parte del subárbol.

El valor de la variable depende de las conexiones existentes entre los nodos primarios de los componentes de origen y destino (ver apartado Bucles, grupos y jerarquías).

Este comportamiento predeterminado es el mismo que presentan los resultados complejos de funciones definidas por el usuario y llamadas a funciones de servicio web.

Si la salida de la variable está conectada a varios nodos de destino **que no guardan relación entre sí**, entonces el valor de la variable se calcula por separado para cada uno de ellos. Esto puede dar lugar a resultados diferentes en cada caso porque el hecho de que haya conexiones primarias distintas influye al contexto en el que se evalúa el valor de la variable.

Posibilidad nº2: `compute-when` conectado

Si **`compute-when`** está conectado a un nodo de salida de un componente de origen, la variable se calcula cada vez que el **elemento de origen** se use **por primera vez** en un subárbol de destino.

En realidad, lo que ocurre es que la variable actúa como elemento secundario del elemento que está conectado a **`compute-when`**.

Esto hace posible el **enlace** entre la variable y un elemento de origen determinado, es decir, en tiempo de ejecución, la variable se vuelve a evaluar cada vez que se lee un elemento nuevo desde la secuencia del componente de origen.

Esto está relacionado con la regla general que rige las conexiones en MapForce (por cada elemento de origen, se crea un elemento de destino). En el caso de **`compute-when`**, por cada elemento de origen, se calcula el valor de la variable.

Posibilidad nº3: `compute-when=once` (calcular una vez)

 © 2014 Altova GmbH

Si hace clic con el botón secundario en el icono **`compute-when`** y selecciona **Calcular una vez** en el menú contextual, el icono pasa a **`compute-when=once`** y desaparece el icono de entrada.

Con esta configuración, el valor de la variable se calcula una vez **antes que** cualquier componente de destino, lo cual convierte a la variable en una constante global para el resto de la asignación.

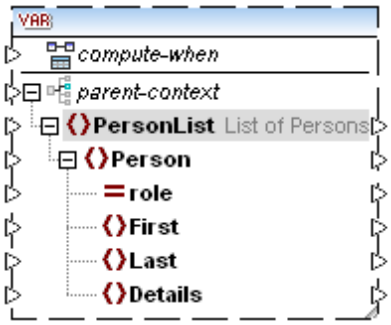
En una función definida por el usuario, la variable se evalúa cada vez que se llama a la función y antes de que se evalúe el resultado de la función propiamente dicha.

Contexto primario

Añadir un contexto primario es muy útil si se utilizan varios filtros y se necesita iterar en un nodo primario adicional.

Para añadir un contexto primario a una variable:

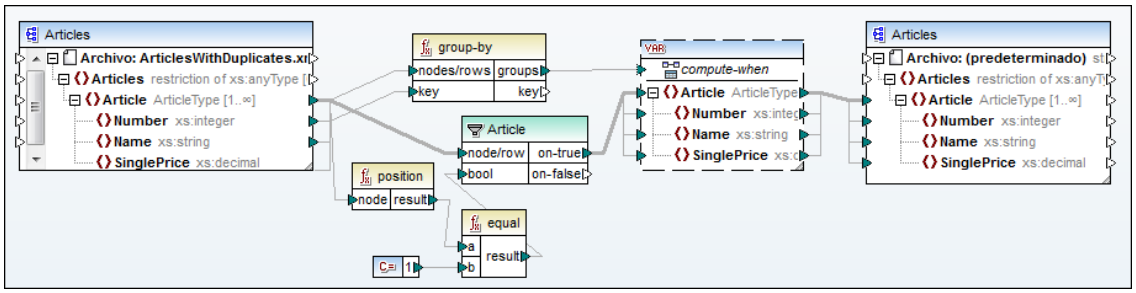
- Haga clic con el botón secundario en el nodo raíz de la variable y en el menú contextual seleccione **Agregar contexto primario**.
  Esto añade un nodo nuevo llamado **`parent-context`** a la jerarquía existente.



El contexto primario añade un nodo primario virtual a la jerarquía del componente. Esto ofrece un nodo más en el que se puede iterar, dentro del mismo componente de origen o de otro.

14.1 Variables: casos de uso

Filtrar instancias múltiples del mismo registro en una instancia de origen
A veces los datos de origen contienen varias instancias del mismo registro. En la mayoría de los casos, solamente es necesario asignar uno de estos registros al componente de destino. El ejemplo que usamos en este apartado es el archivo `DistinctArticles.mfd`, disponible en la carpeta `...\MapForceExamples`.



El archivo `ArticlesWithDuplicates.xml` contiene dos elementos **`Article`** que tienen el número de artículo **`1`** y otros dos elementos **`Article`** con el número de artículo **`3`**.

Articles

xmlns:*xsi*	http://www.w3.org/2001/XMLSchema-instance
xsi:noNamespac...	Articles.xsd

Article (6)

	Number	Name	SinglePrice
1	1	T-Shirt	25
2	1	T-Shirt Duplicate	25
3	2	Socks	2.30
4	3	Pants	34
5	4	Jacket	57.50
6	3	Pants Duplicate	34

El número de artículo (elemento **`Number`**) se utiliza como clave en la función **`group-by`** y, por tanto, crea un grupo por cada número de artículo único. Cada grupo contiene entonces un artículo y todos los duplicados innecesarios del artículo. El paso siguiente consiste en extraer el primer elemento de cada grupo y descartar el resto.

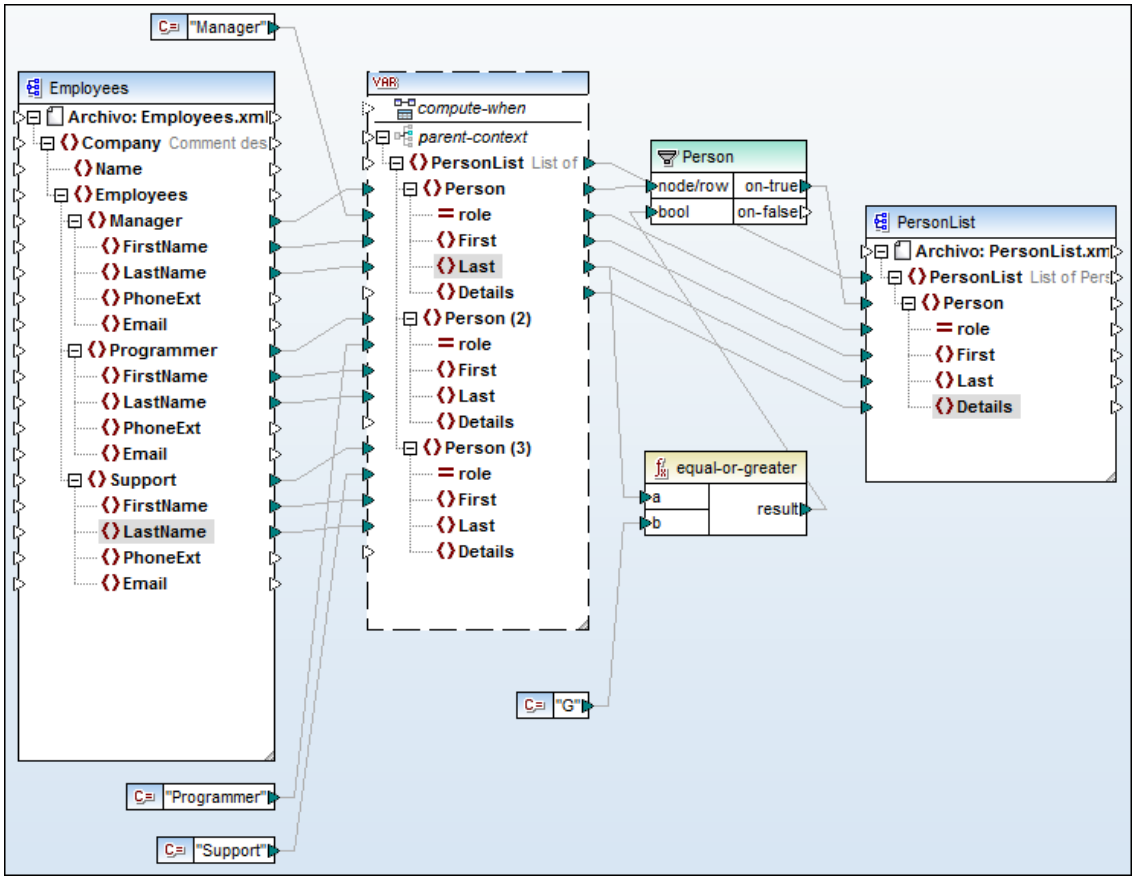
La salida del grupo (de la función **`group-by`**) está conectada al elemento **`compute-when`** de la variable. Esto significa que la variable se evalúa una vez por cada grupo, en el contexto del grupo. Esto establece un nivel de contexto adicional, como si hubiésemos conectado un elemento primario al elemento de destino **`Article`**.

Para seleccionar el primer elemento **`Article`** de cada grupo, usamos la función **`position`** y un componente de **filtrado** y los conectamos con la entrada de la **variable**.

Aplicar filtros a secuencias intermedias:
Los nodos de los componentes de variable se pueden duplicar igual que en cualquier otro tipo de componente. Esto permite generar secuencias a partir de varios orígenes diferentes y después procesar la secuencia.

La imagen siguiente ilustra cómo se podría modificar la asignación `PersonList.mfd` usando una

variable intermedia y cómo los componentes de constante pueden funcionar como elementos de entrada.

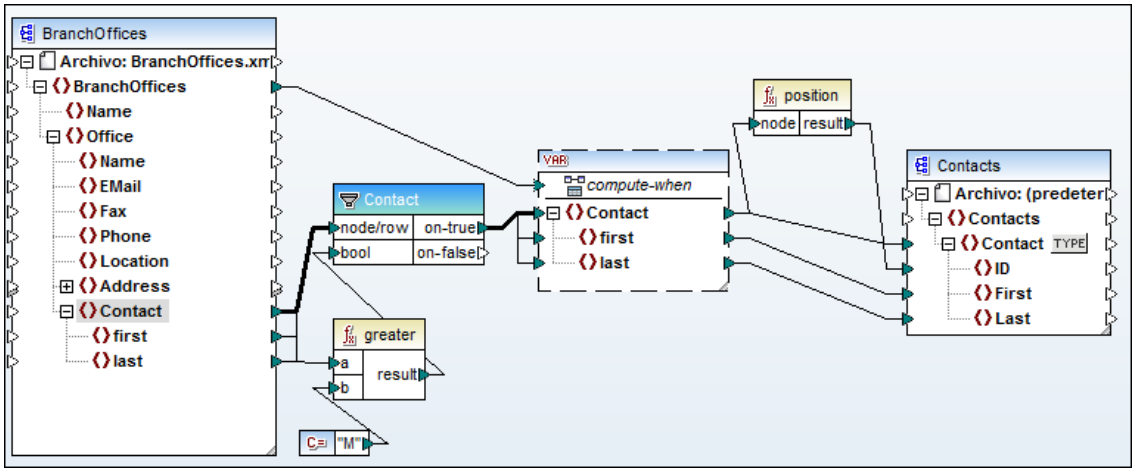


El nodo **`Person`** de la **variable** se duplicó dos veces y se añadió un filtro para seleccionar a las personas (nodos **`Person`**) cuyo apellido (**`Last`**) empieza con una letra superior a la G (G o H, I, etc.)

Numerar nodos de una secuencia filtrada

Para este ejemplo usamos el archivo `PositionInFilteredSequence.mfd` de la carpeta `...\MapForceExamples`, que recurre a una variable para recopilar los contactos filtrados cuyo apellido empieza con una letra superior a la M.

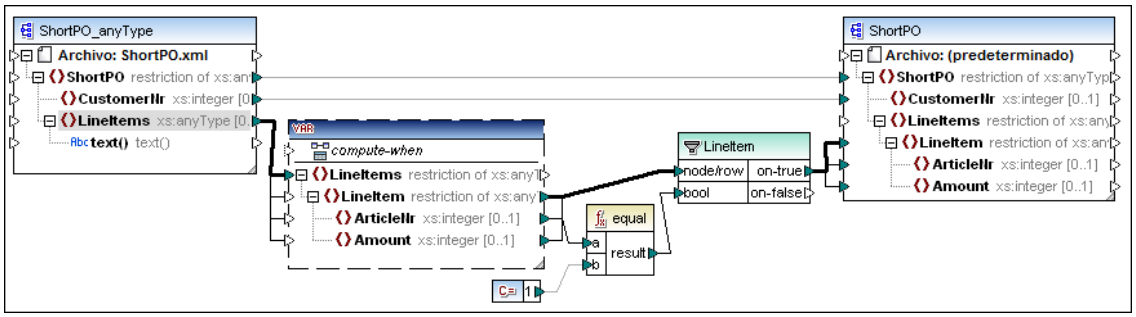
Después estos contactos se transmiten (desde la variable) hasta el componente de destino y la función **`position`** numera estos contactos de forma secuencial.



La variable actúa como un componente de origen más, permitiendo a la función **`position`** procesar la secuencia filtrada.

Extraer determinados datos de un nodo anyType del componente de origen

Este ejemplo está formado por un componente de origen que contiene elementos anyType de los que queremos extraer determinados datos.



La variable intermedia está basada en un esquema que tiene nodos del tipo de datos que deseamos asignar, es decir, los nodos **`ArticleNr`** y **`Amount`** tienen el tipo integer. Estos datos se filtran por medio de **`ArticleNr`** y se transmiten al componente de destino.

 © 2014 Altova GmbH

Altova MapForce 2014

Bibliotecas y funciones

15 Bibliotecas y funciones

Esta sección del manual explica cómo definir funciones definidas por el usuario y describe las bibliotecas de funciones disponibles para los distintos lenguajes de programación.

Definir funciones definidas por el usuario

Agregar funciones XSLT y XQuery personales

Agregar bibliotecas personales de funciones Java, C# y C++

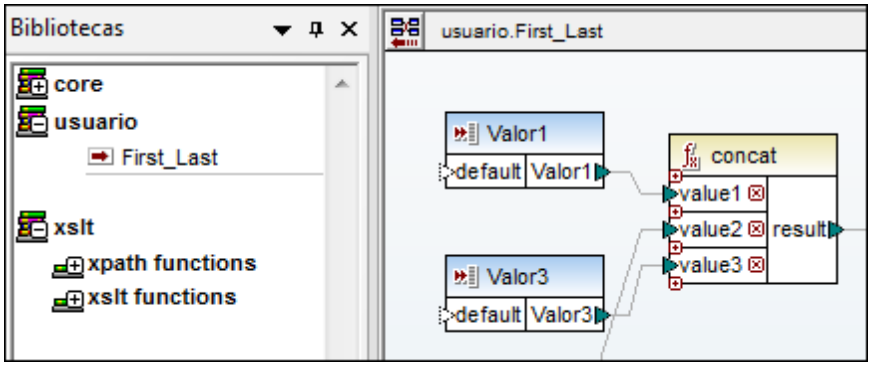
Referencia de funciones

 © 2014 Altova GmbH

15.1 Definir 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.



Con XSLT seleccionado

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:

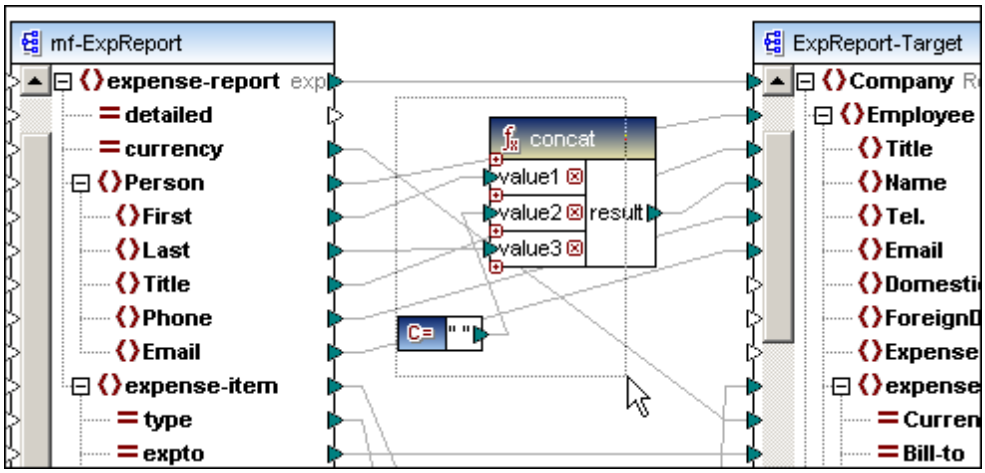
- reunir varias funciones de procesamiento en un solo componente (p. ej. para dar formato a un campo concreto o para buscar un valor)
- reutilizar estos componentes un número ilimitado de veces
- importar funciones definidas por el usuario en otras asignaciones (cargando el archivo de asignación como 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**.

Para el ejemplo siguiente utilizamos el archivo `Tut-ExpReport.mfd`, disponible en la carpeta `...\MapForceExamples\Tutorial`.

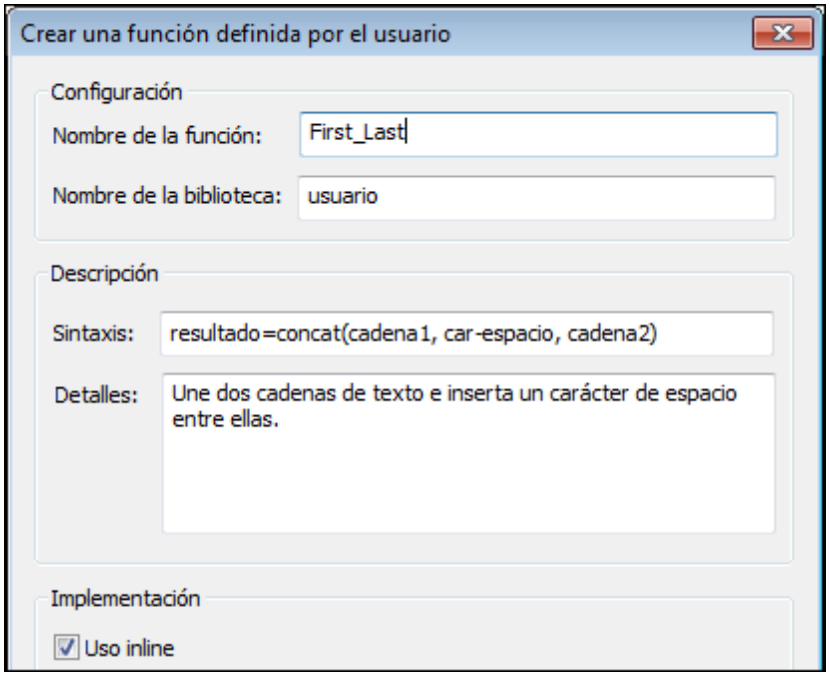
Para crear una función definida por el usuario a partir de componentes ya existentes:

1. Arrastre el puntero del mouse marcando la constante y la función **`concat`** (también puede hacer clic en estos componentes mientras pulsa la tecla **Ctrl**).

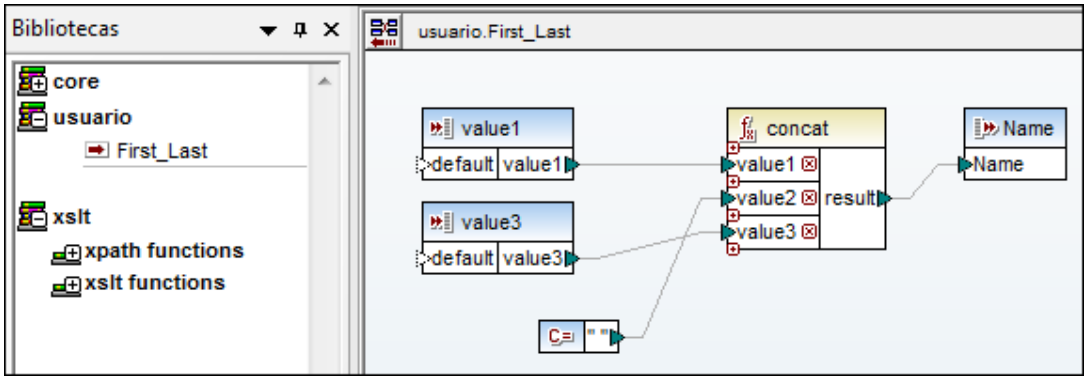


2. Seleccione la opción de menú **Función | Crear una función definida por el usuario a partir de la selección**.
3. Aparece un cuadro de diálogo donde puede escribir el nombre de la nueva función definida por el usuario (es decir, **`First_Last`**).
   Nota: los caracteres válidos para el nombre de 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*.

 © 2014 Altova GmbH

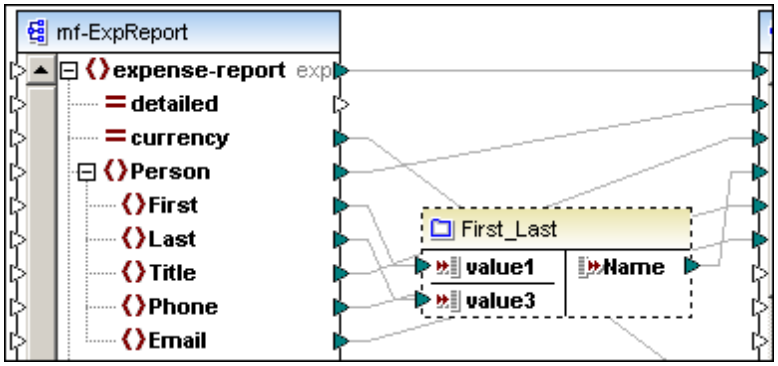


Los elementos que componen la función aparecen en un panel nuevo cuyo título es el nombre de la función. La biblioteca **usuario** aparece en el la ventana Bibliotecas y debajo el nombre de la función recién creada **`First_Last`**.



Con XSLT seleccionado

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.



Observe que las funciones definidas por el usuario tienen un contorno de guiones. Para más información consulte el apartado Funciones inline definidas por el usuario.

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 el apartado Reutilizar funciones definidas por el usuario

Para abrir una función definida por el usuario:

- haga doble clic en la barra de título de la función definida por el usuario en el panel **Asignación**.
- o haga 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 al panel de asignación principal.

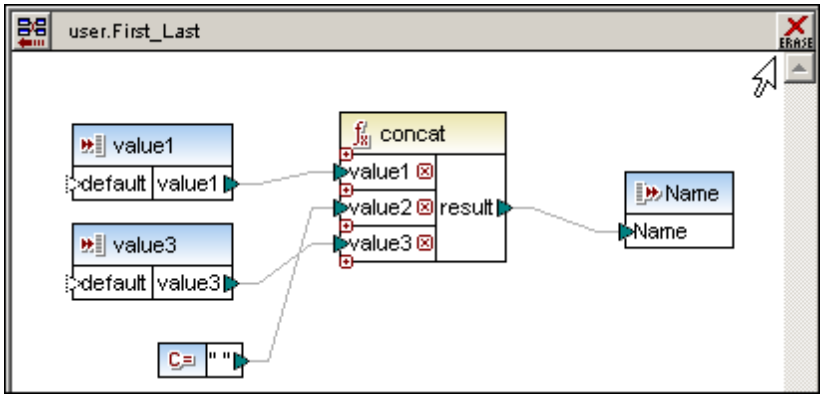
Pulse el botón **Atrás** para retroceder en el historial.

Pulse el botón **Adelante** para avanzar en el historial.

Para eliminar una función definida por el usuario de una biblioteca:

1. Haga doble clic en la función definida por el usuario en la ventana Bibliotecas. Se abre el panel de la función definida por el usuario.
2. Haga clic en el botón **Borrar** situado en la esquina superior derecha del panel.

 © 2014 Altova GmbH



Reutilizar y exportar/importar funciones definidas por el usuario:
Las funciones definidas por el usuario en una asignación se pueden importar en otra asignación diferente:

1. En la parte inferior de la ventana Bibliotecas haga clic en el botón **Agregar o quitar bibliotecas**. En el cuadro de diálogo que aparece haga clic en el botón **Agregar** y seleccione el archivo ***.mfd** cuyas funciones definidas por el usuario desea importar. Las funciones definidas por el usuario importadas aparecen en la ventana Bibliotecas.
2. Arrastre la función importada al panel **Asignación** para utilizarla de nuevo.

Nombre de las bibliotecas
Nota: puede usar el mismo nombre de biblioteca para todas las funciones definidas por el usuario de archivos *.mfd y bibliotecas personales diferentes (consulte el apartado Agregar bibliotecas personales para más información).

Las funciones de las fuentes disponibles aparecen bajo la misma biblioteca en la ventana Bibliotecas. No obstante, solamente se pueden editar (haciendo doble clic en ellas) las funciones del documento que esté activo en ese momento.

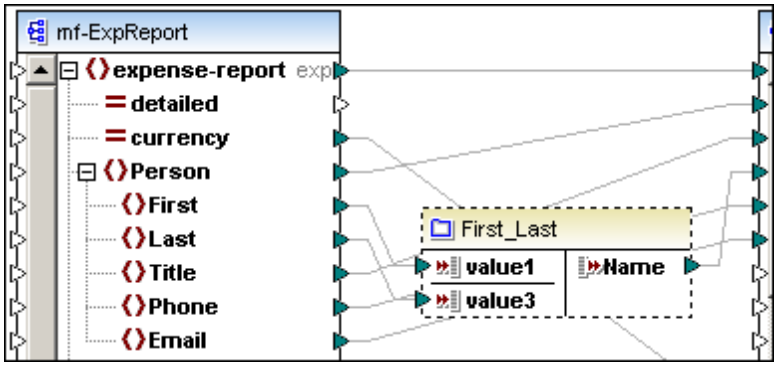
En el ejemplo siguiente:

- la función **`hello`** de la biblioteca **`helloworld`** se importó de una biblioteca personal *.mff
- 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

Con Java seleccionado

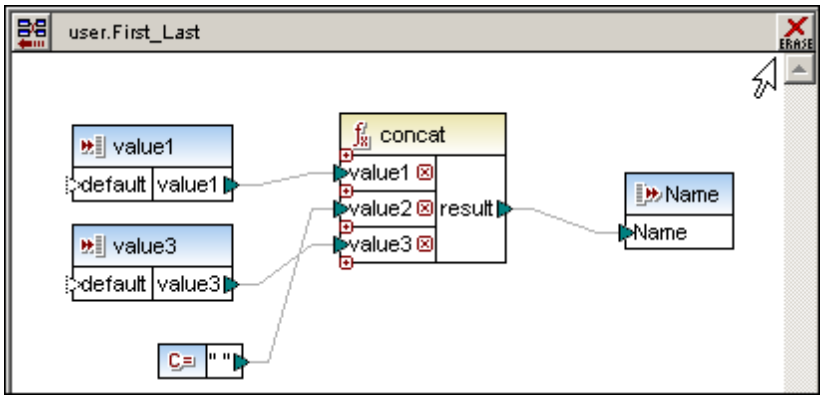
No olvide que, si realiza cambios en una función importada, cuando guarde el archivo ***.mfd** los cambios se aplicarán también a las asignaciones desde las que se importó la función.

Orden de los parámetros en las 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:

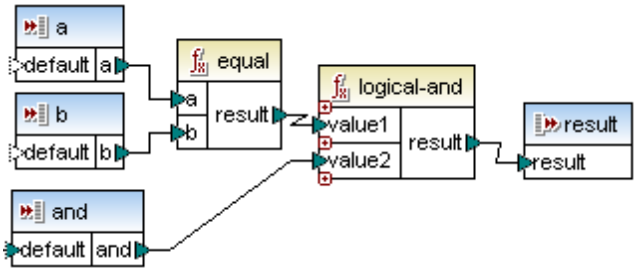
- la posición del componente y las acciones de ajuste de tamaño no se pueden deshacer.
- los componentes de entrada y salida recién añadidos se crean bajo la entrada y salida creada inmediatamente antes.
- puede mezclar parámetros complejos y simples. El orden de los parámetros se deduce de la posición del componente.

 © 2014 Altova GmbH

15.1.1 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**.

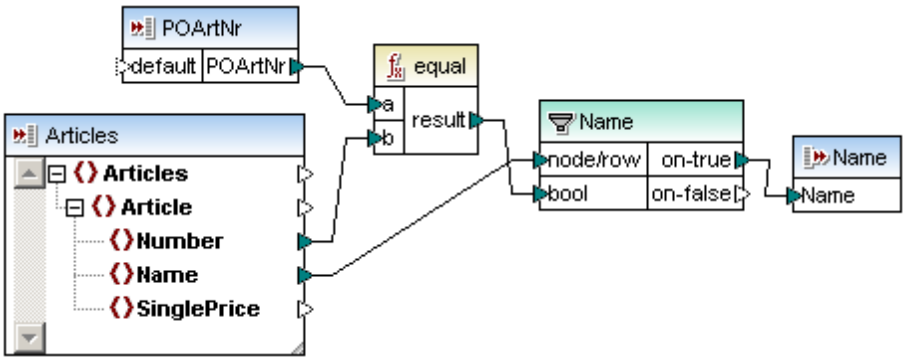
- Componentes/parámetros de entrada: **a**, **b**, **and**
- Componentes/parámetros de salida: **result**



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)



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`

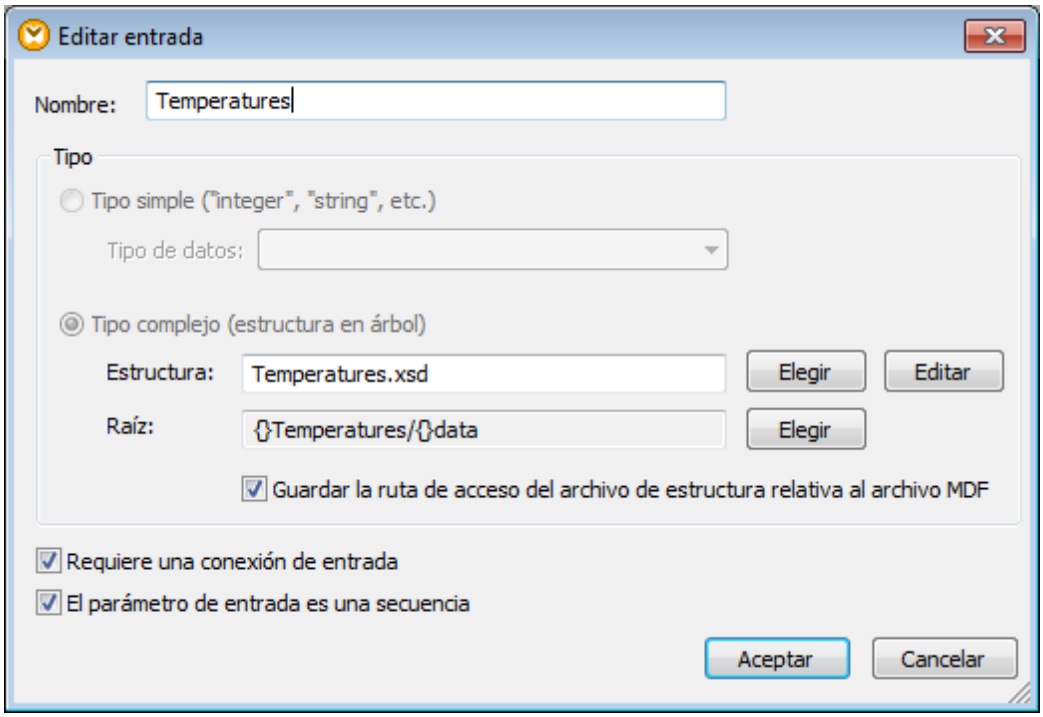
Nota: las funciones definidas por el usuario de las dos imágenes anteriores están disponibles en el archivo de ejemplo **`PersonListByBranchOffice.mfd`** , guardado en la carpeta `...\MapForceExamples`.

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. Para más información consulte el apartado Funciones de agregado.

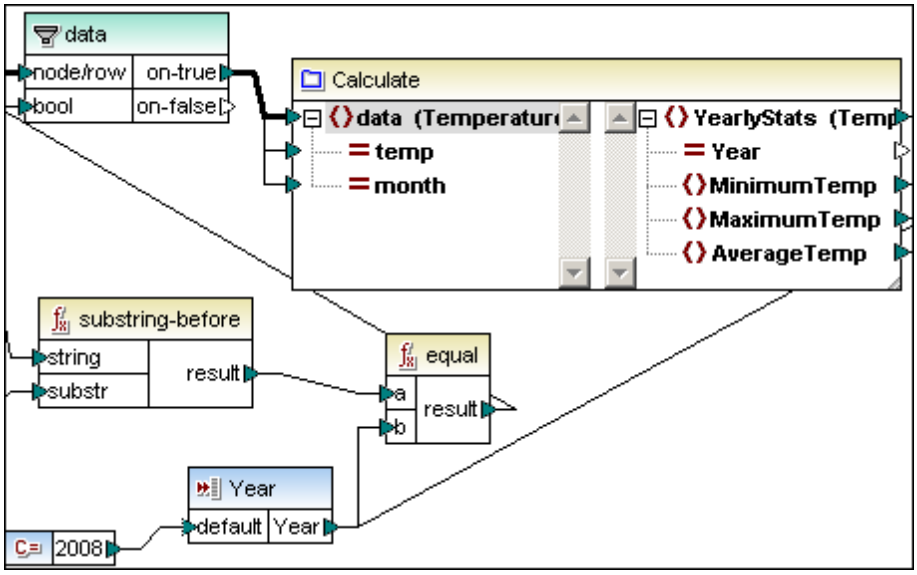
Si activa la casilla *El parámetro de entrada es una secuencia,* el componente trata la entrada como si fuera una secuencia de valores. Si desactiva la casilla, la entrada se trata como si fuera un valor único.



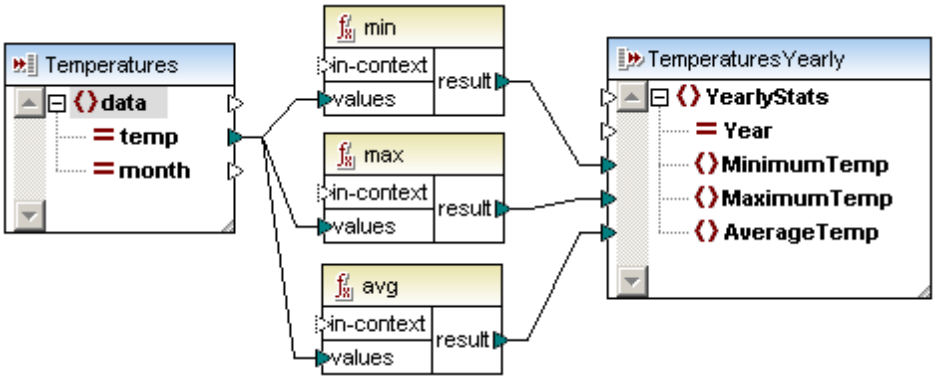
Este tipo de datos de entrada (de secuencia o no secuencia) determina **con qué frecuencia** se llama a la función.

- 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.

 © 2014 Altova GmbH



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.



- 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.

© 2014 Altova GmbH

15.1.2 Funciones definidas por el usuario inline y normales

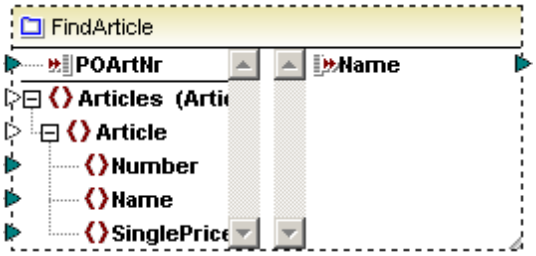
Las funciones inline son diferentes de las demás funciones por la manera en que se implementan cuando se genera código.

- 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 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 NORMALES** (es decir, no inline) tienen un contorno sólido:

Las funciones definidas por el usuario normales (no online) **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 normales 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 normales (no online) **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 normal 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 aparezcan los datos de entrada. Para más información consulte el apartado Parámetros de las funciones.

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.

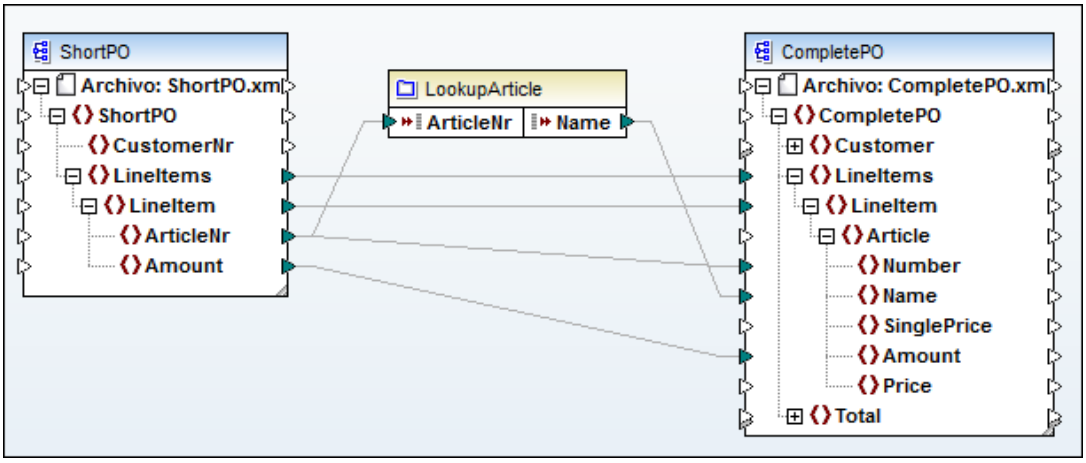
 © 2014 Altova GmbH

15.1.3 Crear una función de búsqueda sencilla

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**.

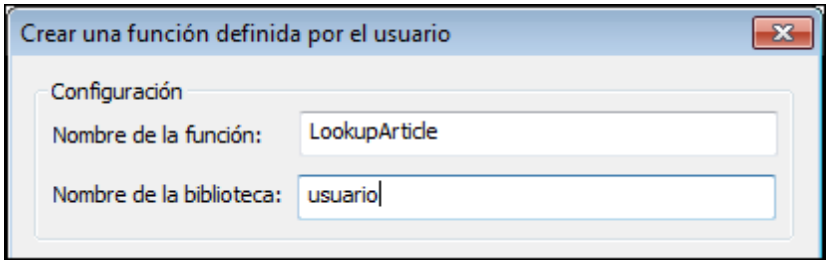


Instrucciones generales:

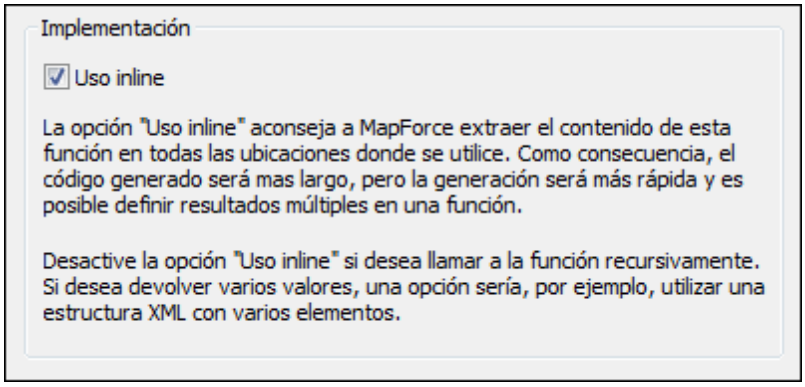
- Inserte el esquema `ShortPO.xsd` y asigne el archivo `ShortPO.xml` como archivo XML de entrada.
- Inserte el esquema `CompletePO.xsd` y seleccione el elemento raíz **`CompletePO`**.
- Inserte una función definida por el usuario nueva siguiendo los pasos descritos a continuación.

Para crear una función definida por el usuario desde cero:

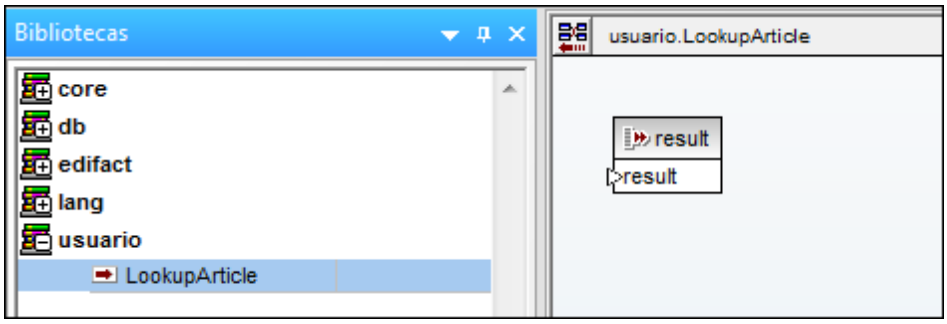
1. Seleccione la opción de menú **Función | Crear función definida por el usuario**.
2. En el cuadro de diálogo que aparece escriba el nombre de la función, p. ej. **`LookupArticle`**, y la biblioteca en la que desea almacenarla (p. ej. **`usuario`**).



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`**.



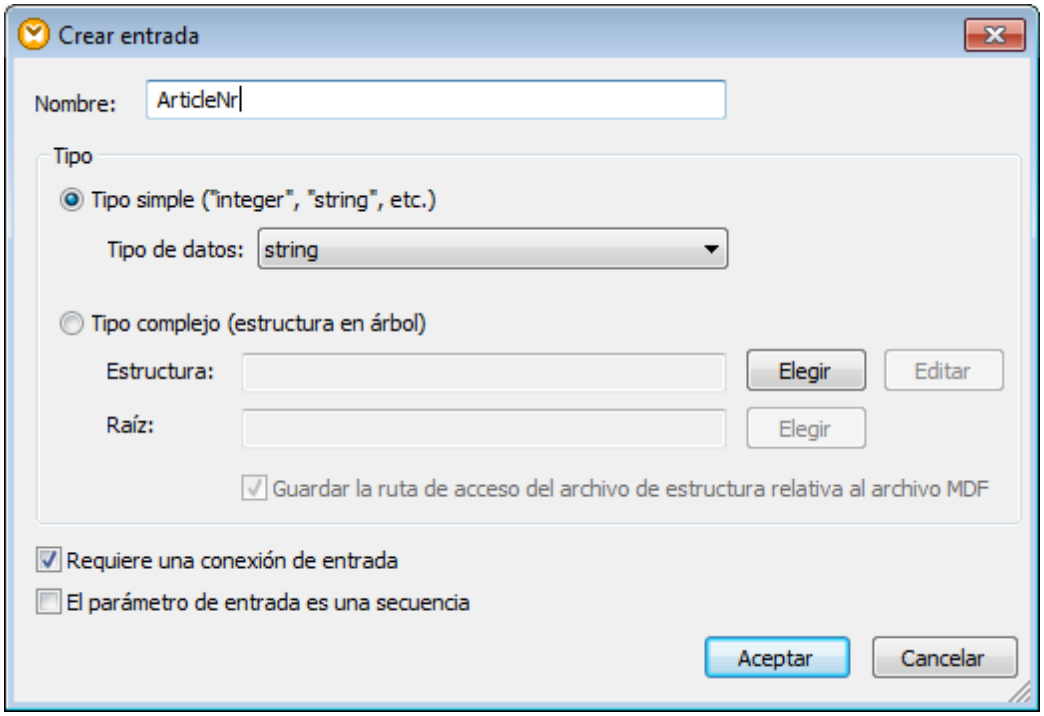
Con Java seleccionado

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`**.

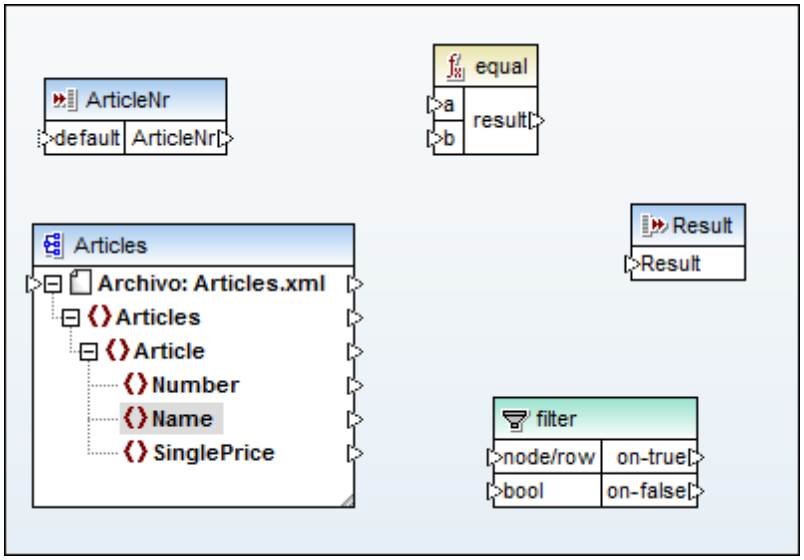
3. Haga clic en el icono **Insertar esquema o archivo XML** para insertar el esquema **`Articles`** y seleccionar el archivo XML **`Articles`** como origen de datos.
4. Haga clic en el icono **Insertar componente de entrada** para insertar un componente de entrada.
5. Escriba el nombre del parámetro de entrada (**`ArticleNr`**) y haga clic en **Aceptar**.

 © 2014 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.

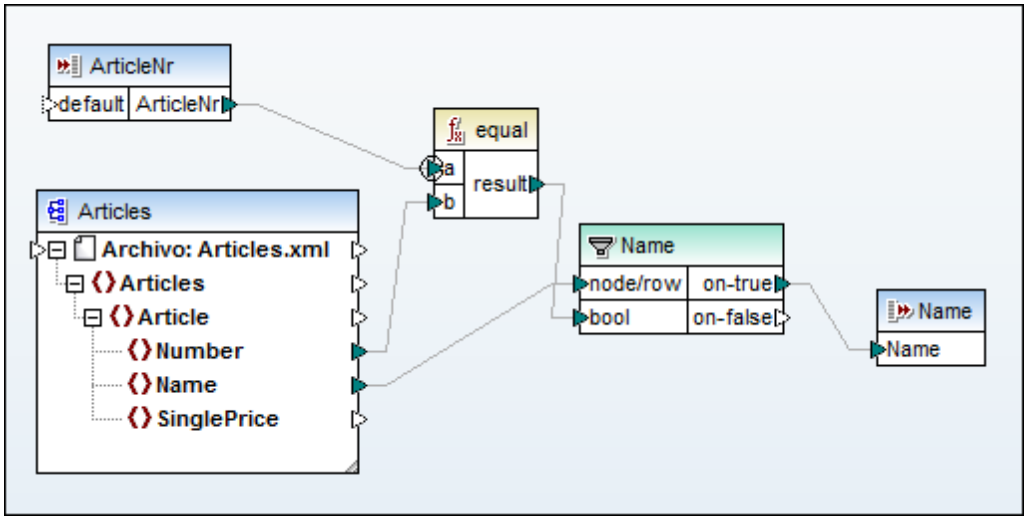
6. Inserte la función **`equal`** de la biblioteca **core | logical**.
7. Inserte un componente de filtrado haciendo clic en el icono **Insertar filtro** de la barra de herramientas.



Ahora cree las asignaciones que aparecen en la imagen siguiente entre los componentes de la función definida por el usuario.

8 Tras crear las asignaciones haga clic con el botón secundario en el parámetro **`a`** de la función **`equal`** y seleccione **Contexto prioritario** en el menú contextual.

9. Haga **doble clic** en la salida de la función y escriba el nombre **`Name`** para el parámetro de salida.



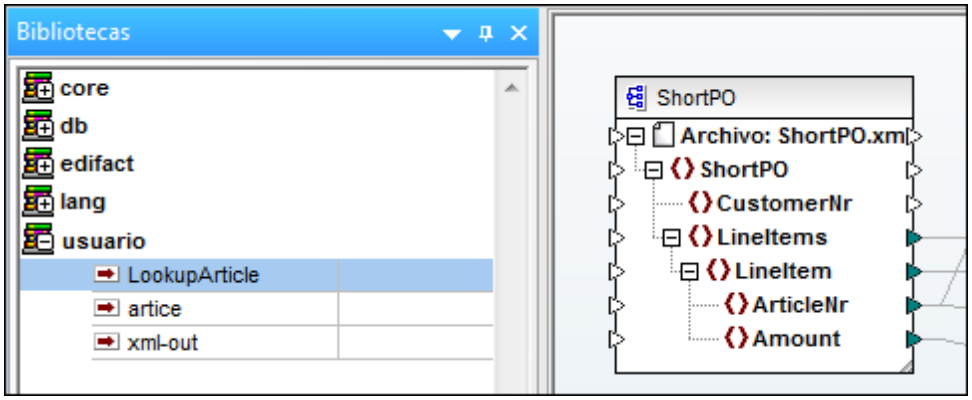
La función definida por el usuario ya está definida.

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**.

10. 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**.



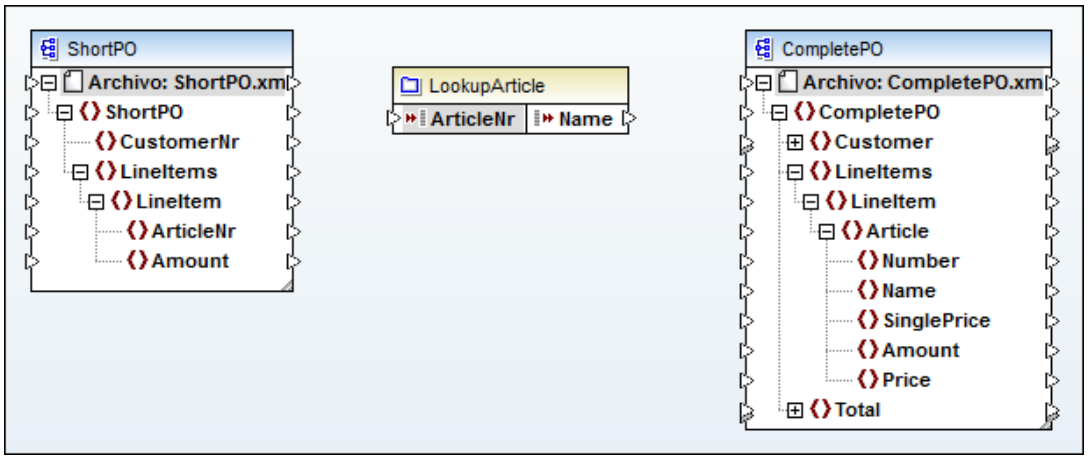
Con Java seleccionado

11. Arrastre la función **`LookupArticle`** hasta el panel **Asignación**.

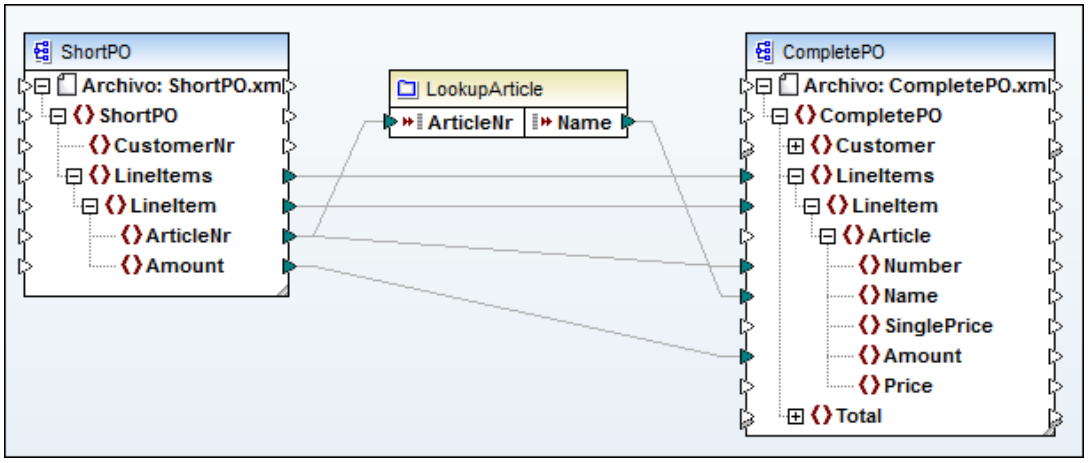
 © 2014 Altova GmbH

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.



10. Cree las conexiones que aparecen en la imagen siguiente y abra el panel **Resultados** para ver el resultado de la asignación.



15.1.4 Función definida por el usuario compleja: con un 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.

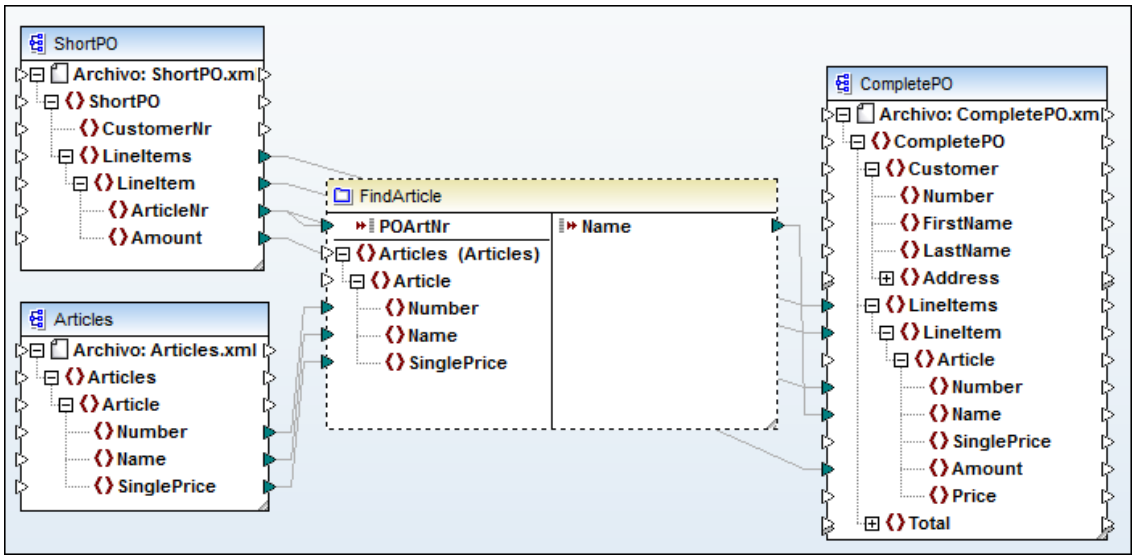
Tenga en cuenta que la función definida por el usuario **`FindArticle`** tiene dos partes.

La parte de la izquierda contiene los parámetros de entrada:

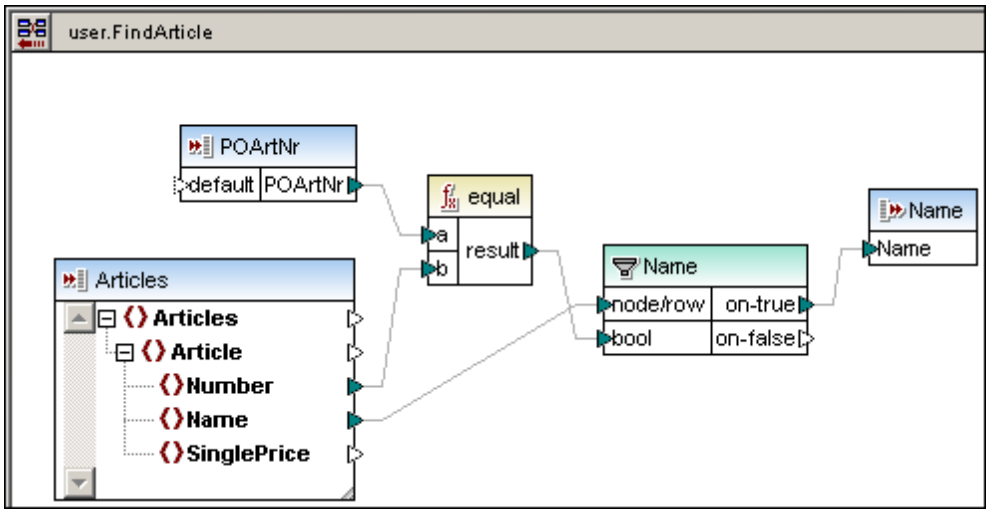
- un parámetro de entrada simple llamado **`POArtNr`**
- un parámetro de entrada complejo llamado **`Articles`**, cuyos nodos XML secundarios tienen asignaciones directas a otros componentes de la asignación

La parte de la derecha contiene:

- un parámetro de salida simple llamado **`Name`**.



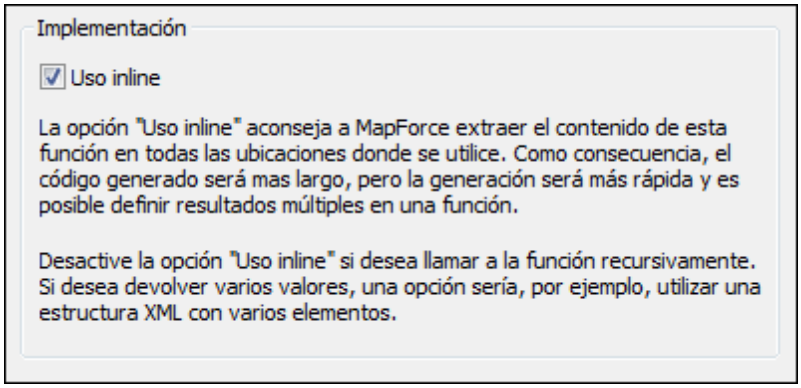
A continuación puede ver los componentes que constituyen la función definida por el usuario, con los dos componentes de entrada la izquierda y el de salida a la derecha.



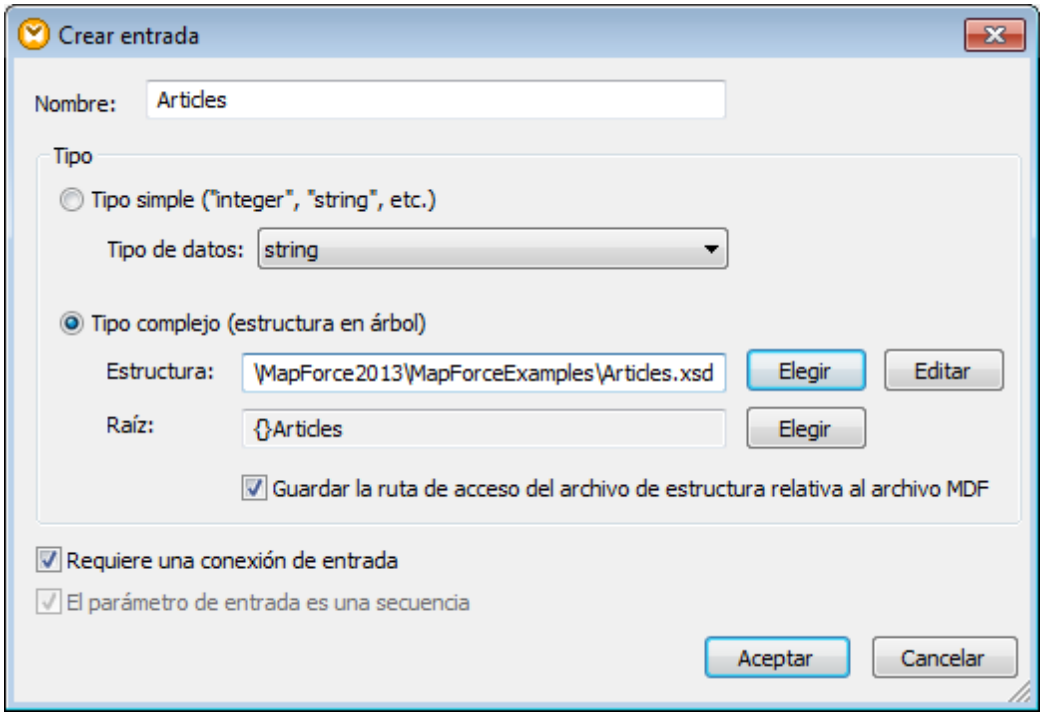
Definir componentes de entrada complejos

Para definir componentes de entrada complejos:

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.



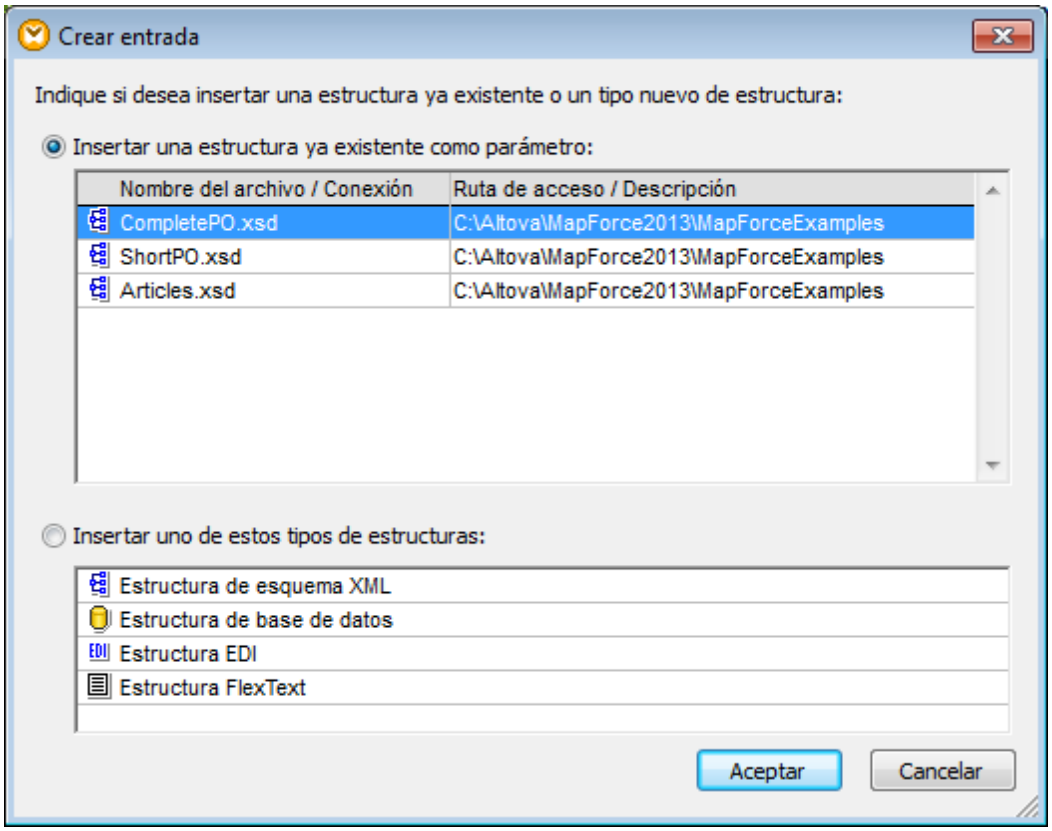
2. Haga clic en el icono **Insertar componente de entrada** de la barra de herramientas.
3. Escriba el nombre del componente de entrada en el campo *Nombre*.



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 listas.

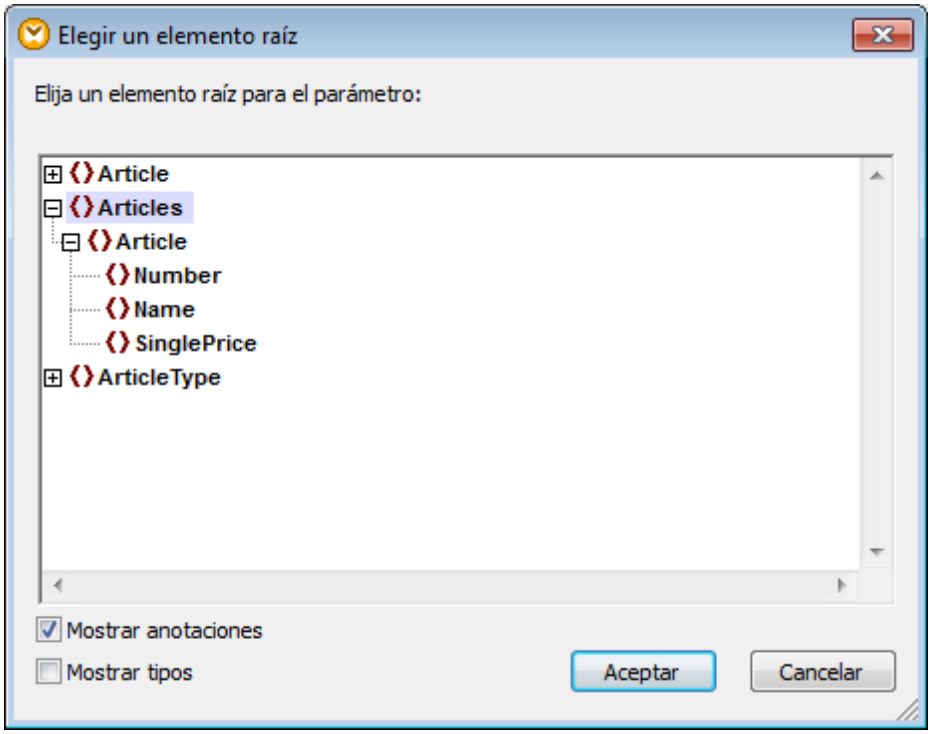
El primer cuadro de lista enumera los componentes ya **existentes** en la asignación (en este caso, tres esquemas). Es decir, la lista enumera todos los componentes que ya se insertaron en la asignación activa (p. ej. un esquema XML, una BD, un archivo EDI o una estructura FlexText).

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).

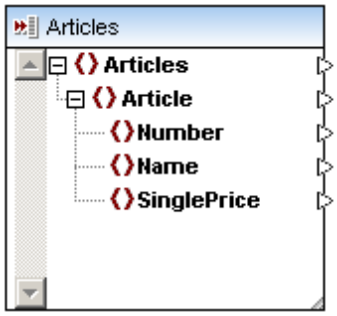


5. Seleccione el botón de opción *Insertar una estructura nueva*, después la opción **Estructura de esquema XML** y haga clic en **Aceptar**.
6. Seleccione **`Articles.xsd`** en el cuadro de diálogo Abrir.
7. Seleccione el elemento que desea usar como elemento raíz del componente (p. ej. **`Articles`**) y haga clic en **Aceptar**. Haga otra vez clic en **Aceptar** para cerrar el otro cuadro de diálogo.

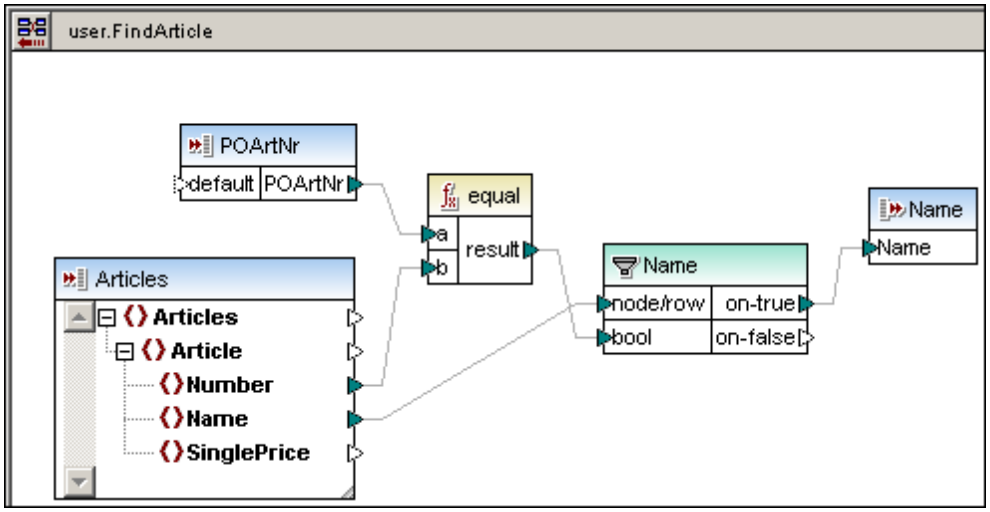
 © 2014 Altova GmbH



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.



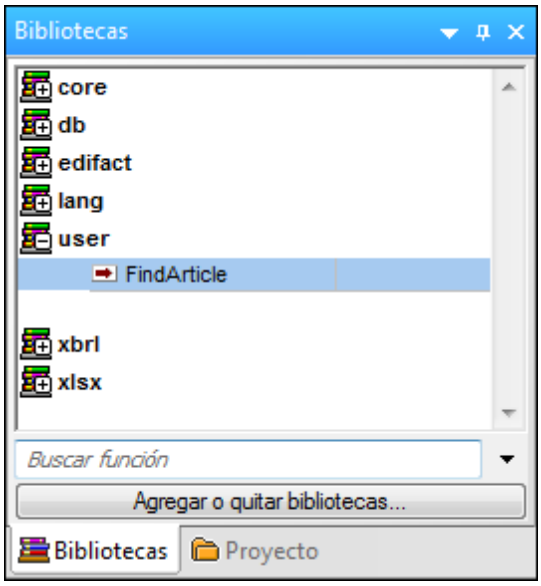
8. Inserte el resto de componentes que aparecen en la imagen siguiente, es decir, otro componente de entrada simple (**`POArtNr`**), un filtro, la función **`equal`** y un componente de salida (**`Name`**). Después conéctelos tal y como muestra la imagen.



Nota:

- 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.

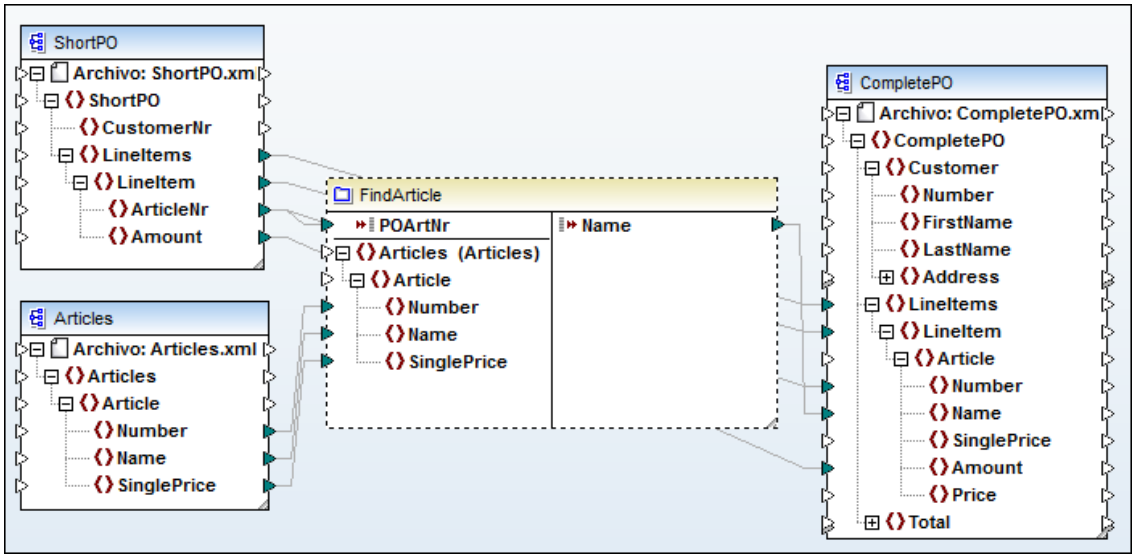
10. Haga clic en el icono **Volver** para volver a la asignación principal.
11. Inserte la función definida por el usuario nueva (arrastrándola desde la ventana Bibliotecas hasta el panel **Asignación**).



Con Java seleccionado

12. Cree las conexiones que aparecen en esta imagen.

 © 2014 Altova GmbH



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>
```

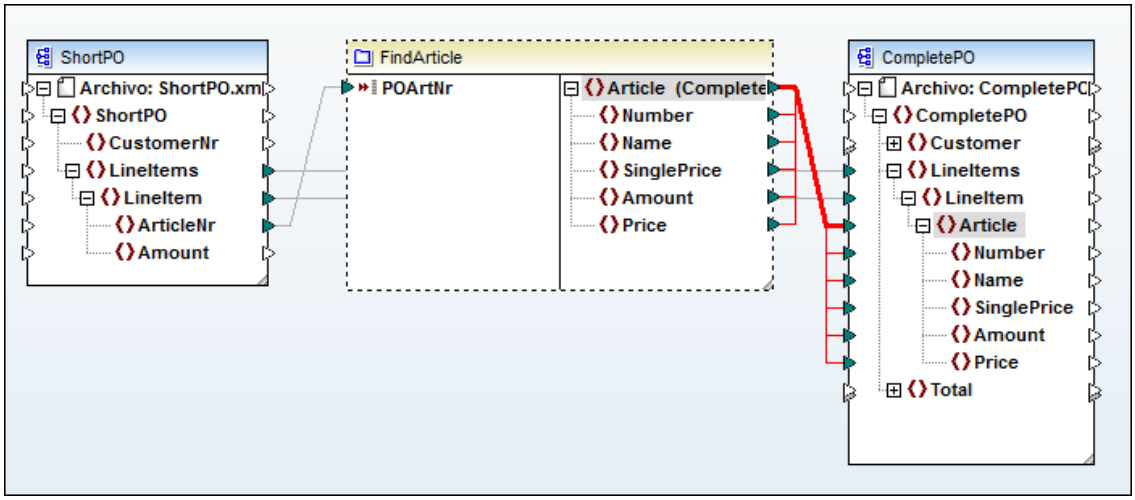
Nota: cuando cree conexiones de **copia total** entre un esquema y un parámetro de una función definida por el usuario, ambos componentes deben estar basados en el mismo esquema. Sin embargo, no es necesario que ambos tengan el mismo elemento raíz.

15.1.5 Función definida por el usuario compleja: con un nodo XML como salida

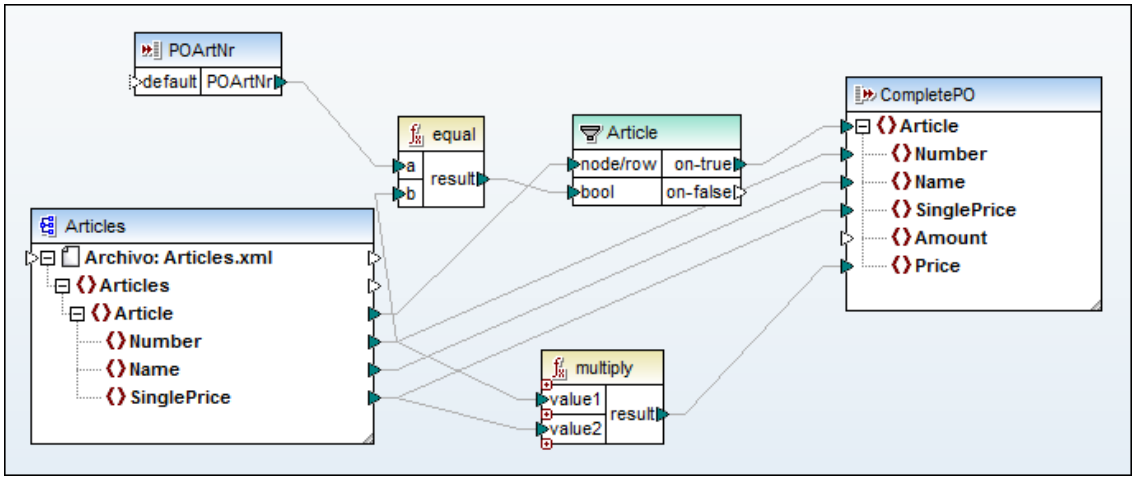
Este apartado utiliza el archivo de ejemplo `lookup-udf-out.mfd`, disponible en la carpeta `...\MapForceExamples`. A continuación explicamos cómo se define una función definida por el usuario de tipo inline con un componente de salida complejo.

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 la izquierda y el de salida a la derecha.

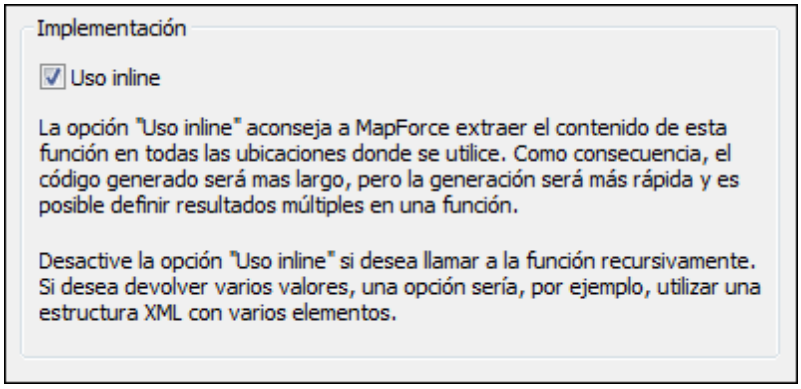


Definir componentes de salida complejos

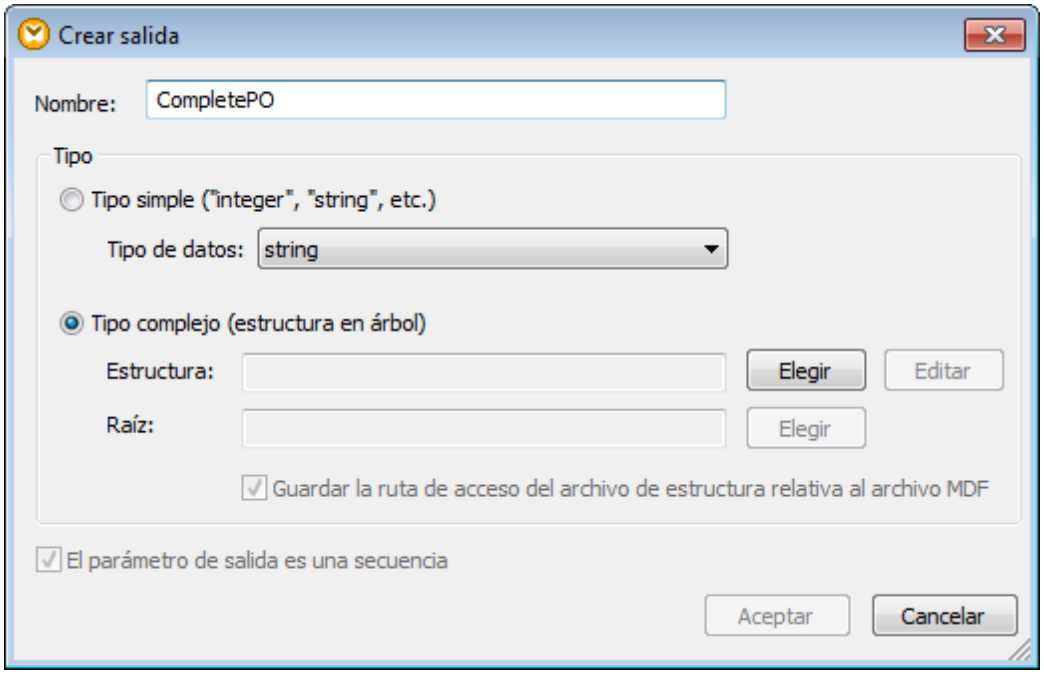
Para definir componentes de salida complejos:

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.

 © 2014 Altova GmbH



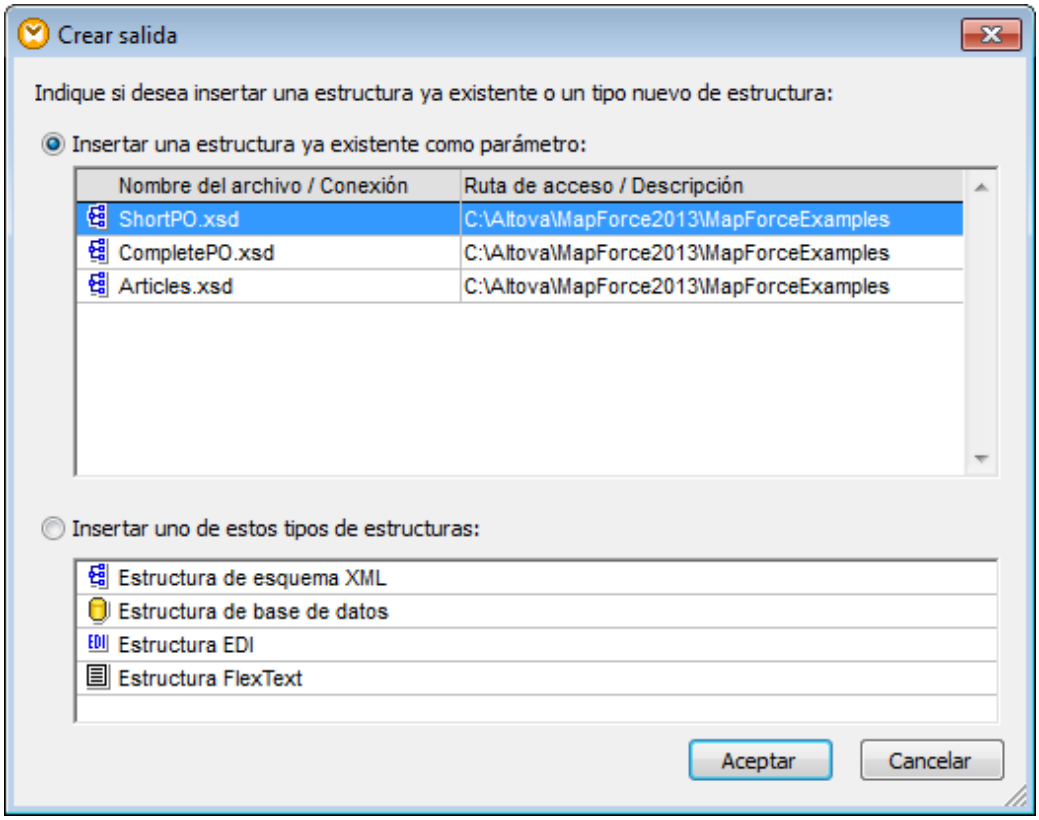
2. Haga clic en el icono **Insertar componente de salida** de la barra de herramientas. En el cuadro de diálogo escriba el nombre del componente de entrada en el campo *Nombre* (p. ej. CompletePO).



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.

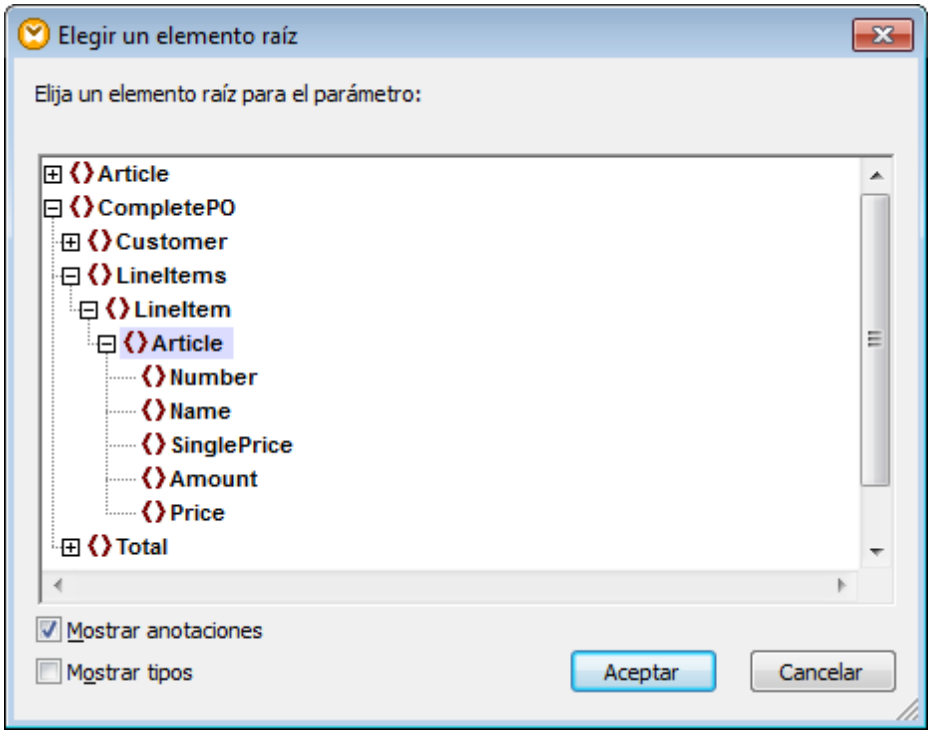
   El primer cuadro de lista enumera los componentes ya **existentes** en la asignación (en este caso, tres esquemas). Es decir, la lista enumera todos los componentes que ya se insertaron en la asignación activa (p. ej. un esquema XML, una BD, un archivo EDI o una estructura FlexText).

   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).

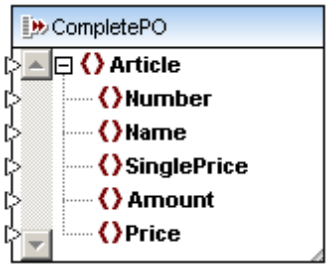


4. Seleccione el botón de opción *Insertar una estructura nueva*, seleccione la opción **Estructura de esquema XML** y haga clic en **Aceptar**.
5. Seleccione el esquema `CompletePO.xsd` en el cuadro de diálogo Abrir.
6. Haga clic en el elemento que desea usar como elemento raíz del componente (p. ej. **`Article`**) y haga clic en **Aceptar**. Haga otra vez clic en **Aceptar** para cerrar el otro cuadro de diálogo.

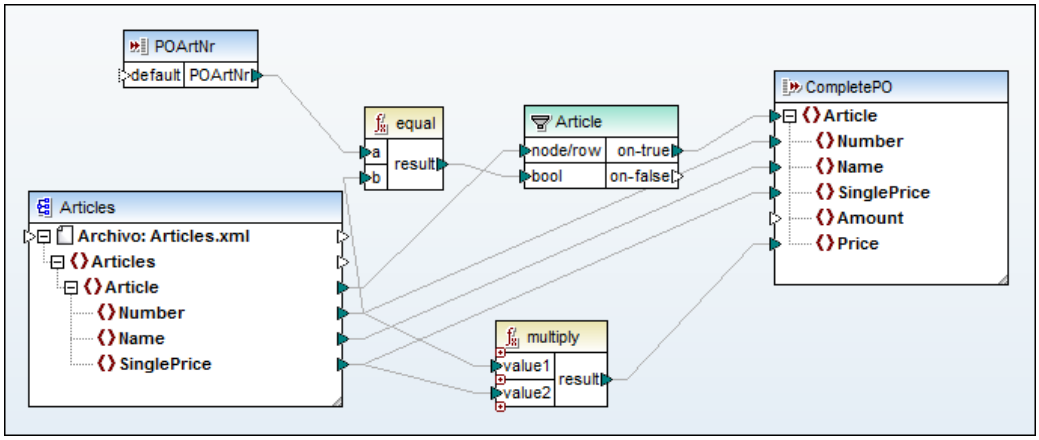
 © 2014 Altova GmbH



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.



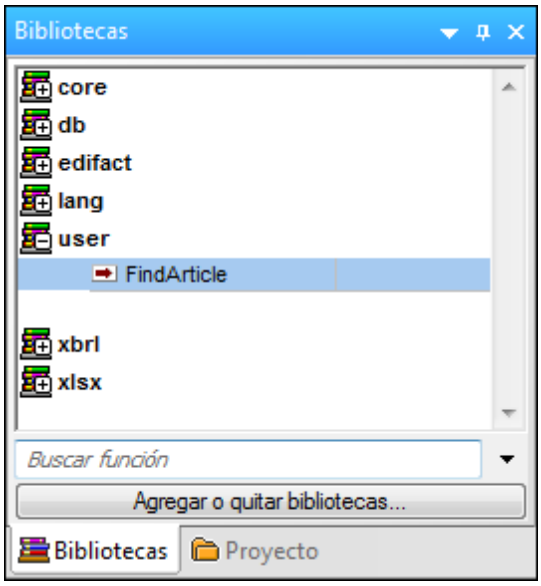
7. Inserte el esquema XML **Articles** en la función definida por el usuario y asigne el archivo `Articles.xml` como XML de instancia.
8. Inserte el resto de componentes que aparecen en la imagen siguiente, es decir: el componente de entrada simple **`POArtNr`**, un filtro, una función **`equal`** y una función **`multiply`**. Después conéctelas como muestra la imagen.



Nota:

- 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**.

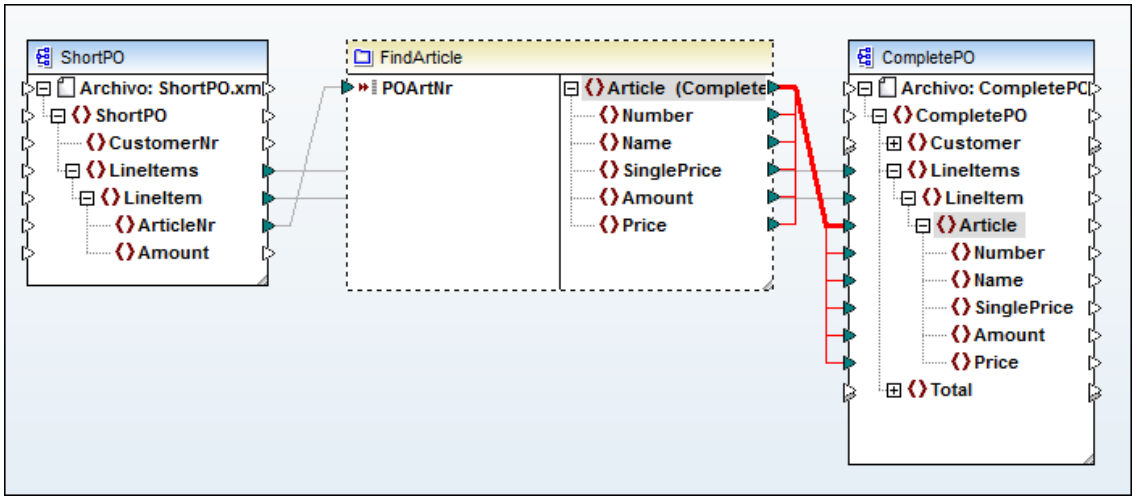
9. Haga clic en el icono **Volver** para volver a la asignación principal.
10. Inserte la función definida por el usuario nueva (arrastrándola desde la ventana Bibliotecas hasta el panel **Asignación**).



Con Java seleccionado

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 secundario en el conector y seleccione **Copia total** en el menú contextual. Los demás conectores se generan automáticamente y se resaltan en rojo (*imagen siguiente*).

 © 2014 Altova GmbH



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:

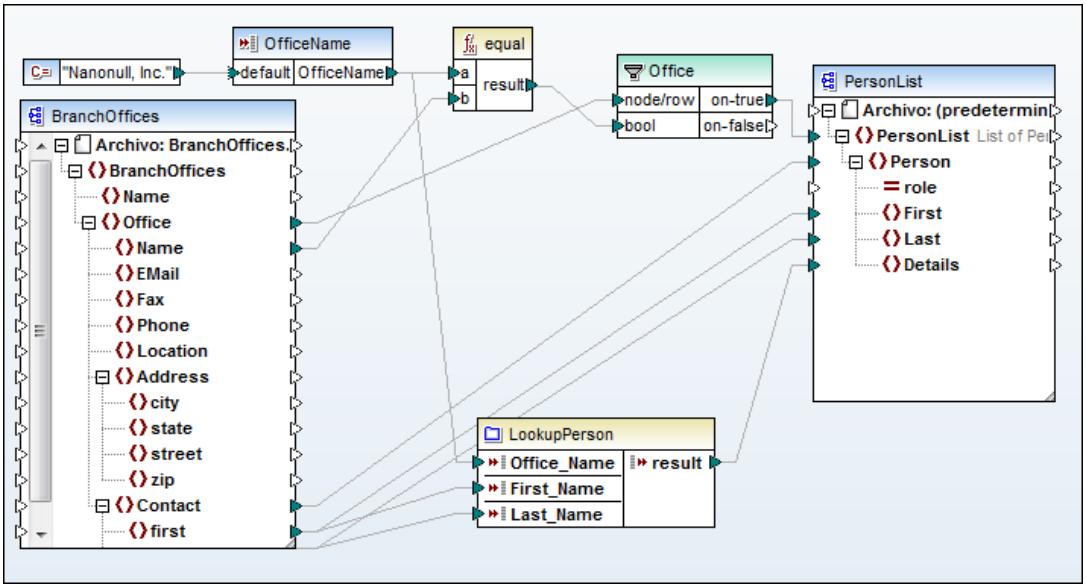
- un parámetro de salida complejo llamado **`Article (CompletePO)`** con nodos XML secundarios, que asigna los elementos filtrados (los que tienen el mismo número de artículo) al componente de destino **CompletePO**.

```
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>
```

15.1.6 Ejemplo de función definida por el usuario

El archivo `PersonListByBranchOffice.mfd`, de la carpeta `...\MapForceExamples`, ilustra en detalle estas características:

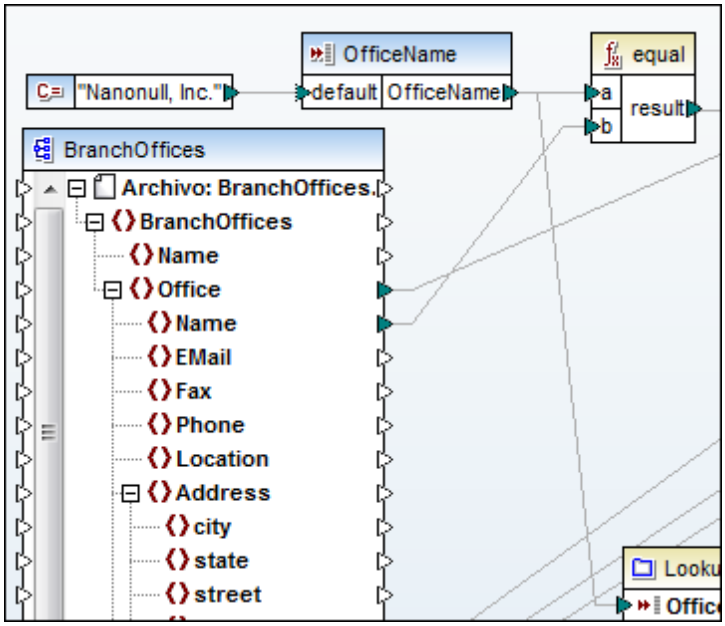
- funciones definidas por el usuario anidadas (p. ej. la función **`LookupPerson`**)
- funciones de búsqueda que generan una cadena de salida (p. ej. **`LookupPerson`**)
- parámetros de entrada **opcionales** que también pueden aportar un valor **predeterminado** (p. ej. el componente **`EqualAnd`** de la función **`LookupPerson`**)
- parámetros de entrada **configurables**, que pueden hacer de parámetros de la línea de comandos cuando se ejecuta el código de asignación generado.



Parámetros de entrada configurables

El componente de entrada (**`OfficeName`**) recibe datos aportados cuando se ejecuta una asignación. Esto se consigue 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 datos en el panel **Resultados**.

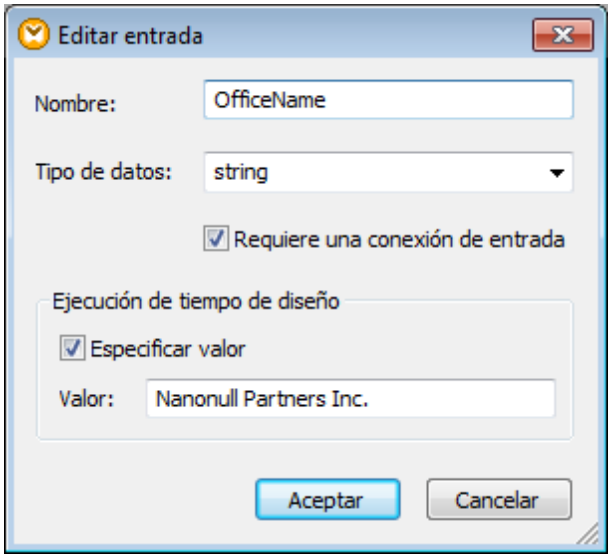


Para definir el valor de entrada:

1. Haga doble clic en el componente de entrada e introduzca un valor distinto en el cuadro de texto *Valor* del grupo de opciones *Ejecución de tiempo de diseño* (p. ej. `Nanonull Partners, Inc.`) y haga clic en **Aceptar**.
2. Abra el panel **Resultados** para ver el efecto. Ahora aparece un conjunto distinto de personas.

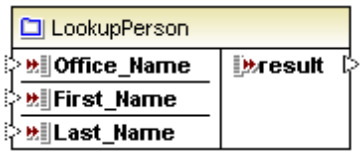
   Recuerde que los datos introducidos en este cuadro de diálogo se usan solamente en el modo de ***vista previa*** (es decir, al abrir el panel **Resultados**). Si no introduce un valor o desactiva la casilla *Especificar un valor*, se utilizan los datos asignados al icono de entrada **`default`**.

   Para más información consulte el apartado Línea de comandos: definir parámetros de entrada.



© 2014 Altova GmbH

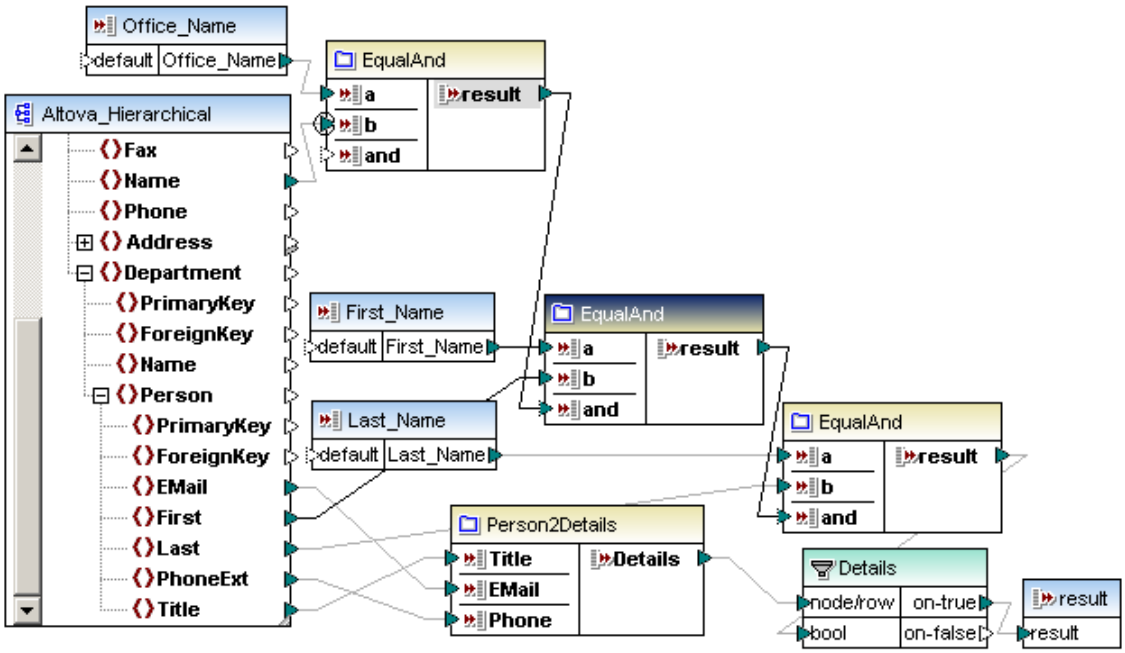
Componente LookupPerson



Al hacer doble clic en esta función definida por el usuario aparecen los componentes que la constituyen. Lo que hace este componente es:

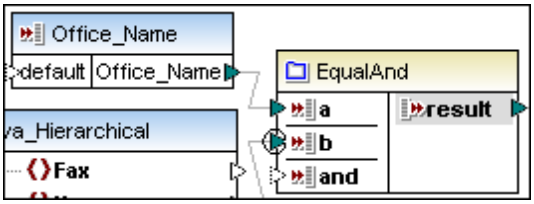
- **comparar** el contenido de **`Office`**, **`First`** y **`Last`** de `BranchOffices.xml` con los mismos campos del archivo `Altova_Hierarchical.xml`, usando los componentes de **entrada** y la función definida por el usuario **`EqualAnd`**.
- **reúne** los elementos **`Email`**, **`PhoneExt`** y **`Title`** usando la función definida por el usuario **`Person2Details`**
- **pasa** los datos de contacto del elemento **`Person`** al componente de **salida** si las comparaciones **`EqualAnd`** anteriores dan como resultado `true` (es decir, si enviaron el valor `true` al componente de filtrado).

Una función definida por el usuario siempre genera un valor, que también puede ser una cadena vacía. Este sería el caso si el valor **`bool`** del componente de filtrado fuera **`false`**. En lugar de los datos aportados por el componente **Person2Details**, la función generaría una cadena vacía.



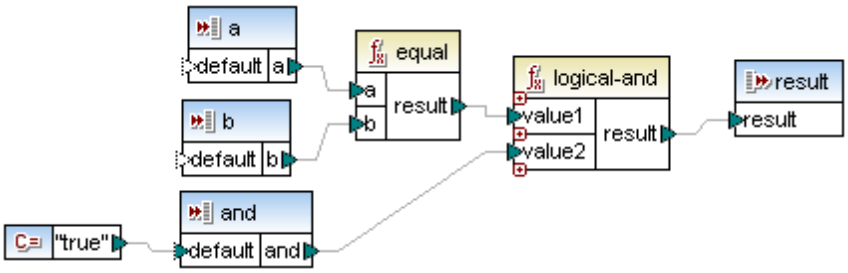
- Los tres componentes de entrada (**`Office_Name`**, **`First_Name`**, **`Last_Name`**) reciben sus datos del archivo XML `BranchOffices.xml`.
- El componente **`EqualAnd`** compara dos valores y ofrece un valor de comparación **opcional**, así como un valor predeterminado.
- **`Person2Details`** combina los tres campos de datos de **`Person`** y los pasa al componente de salida **`result`** a través de un filtro.

Componente EqualAnd



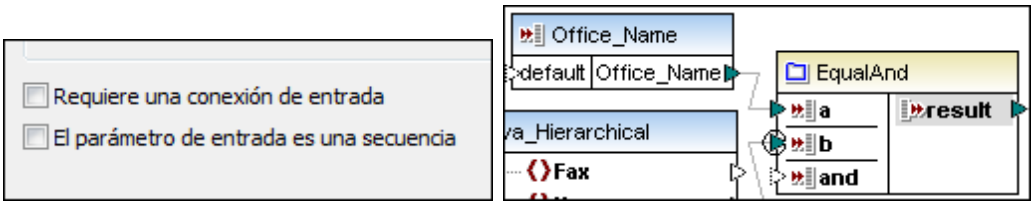
Al hacer doble clic en este componente definido por el usuario aparecen los componentes que lo constituyen. Lo que hace este componente es:

- compara dos parámetros de entrada **a** y **b** y pasa el resultado a la función **`logical-and`**. Observe que el parámetro **b** se definió como **contexto prioritario**. Esto garantiza que se procesen primero los datos de **`Person`** de esa oficina, suministrados por el parámetro **a**.
- compara, por medio de la función **`logical-and`**, el resultado de la primera comparación con el parámetro de entrada **opcional `and`**.
- pasa el valor booleano de esta segunda comparación al parámetro de salida **`result`**.



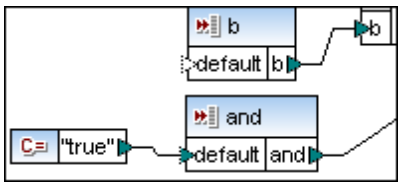
Parámetros opcionales

Al hacer doble clic en el parámetro **`and`** de la función **`EqualAnd`** (*imagen anterior*) aparece un cuadro de diálogo donde puede convertir el parámetro en opcional (desactivando la casilla *Requiere una conexión de entrada*).



Si **desactiva** la casilla *Requiere una conexión de entrada*:

- no es necesario conectar ningún elemento al icono de entrada de esta función definida por el usuario (p. ej. el parámetro **`and`** de la primera función **`EqualAnd`** no tiene un conector de entrada). Cuando el parámetro de entrada es opcional, el icono de entrada tiene un contorno de guiones.
- se puede suministrar un valor predeterminado (**`default`**) asignándole un componente de dentro de la función definida por el usuario (p. ej. la constante **`true`**).



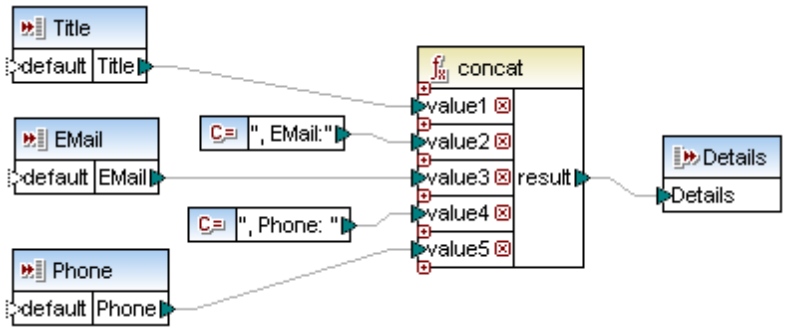
 © 2014 Altova GmbH

- una asignación desde otro elemento (asignada a la entrada opcional) tiene prioridad sobre el valor predeterminado **`default`**. P. ej. el parámetro **`and`** de la segunda función **`EqualAnd`** recibe datos de entrada del parámetro **`result`** de la primera función **`EqualAnd`**.

Componente Person2Details

Al hacer doble clic en este componente definido por el usuario aparecen los componentes que la constituyen. Lo que hace este componente es:

- concatenar tres entradas y pasar la cadena resultante al parámetro de salida.
- al hacer doble clic en un parámetro de salida aparece un cuadro de diálogo donde puede cambiar el nombre y el tipo de datos del parámetro.



 © 2014 Altova GmbH

15.2 Agregar funciones XSLT y XQuery personales

MapForce ofrece una función para agregar funciones personales a la biblioteca de funciones XLST integrada. Esta función se habilita haciendo clic en el icono **XSLT** o seleccionando **Resultados | XSLT 1.0**.

Los archivos XSLT aparecen como bibliotecas y todas sus **plantillas con nombre** aparecen como funciones bajo el nombre de la biblioteca.

- Las funciones deben declararse como plantillas con nombre en el archivo XSLT según las normas de la especificación XSLT 1.0.
- Si el archivo XSLT importado importa o incluye otros archivos XSLT, también se importan estos otros archivos y funciones XSLT.
- Cada plantilla con nombre aparece como una función bajo el nombre de la biblioteca.
- La cantidad de iconos de entrada asignables depende del número de parámetros utilizados en la llamada a plantillas. También se pueden usar parámetros opcionales.
- Las actualizaciones en los archivos XSLT importados se producen al iniciar el programa o cuando cambian los archivos.
- Los espacios de nombres son compatibles.

Nota: cuando escriba plantillas con nombre asegúrese de enlazar al espacio de nombres correcto todas las instrucciones XPath. Los enlaces de espacio de nombres de la asignación se pueden ver en el panel XSLT. Para más información consulte el anexo XSLT 1.0.

15.2.1 Agregar clases Java y funciones .NET DLL personales

Los archivos de clases Java compilados, así como los archivos de ensamblados .NET (incluidos los ensamblados .NET 4.0) se pueden agregar a la ventana Bibliotecas y utilizarse como cualquier otra función predeterminada de MapForce. La vista previa del resultado de las asignaciones que utilizan estas funciones Java y .NET se puede ver en el panel **Resultados** y las funciones están disponibles en el código generado.

MapForce **es compatible** con:

- los archivos de clases Java compilados (`.class`) cuando el lenguaje de salida seleccionado es **Java**.
- los archivos de ensamblados .NET cuando el lenguaje de salida seleccionado es C#. Los ensamblados .NET suelen ser compatibles independientemente del lenguaje original (C++ o VB.NET), siempre y cuando solamente utilicen tipos de datos básicos del ensamblado del sistema como parámetros y tipos devueltos.

MapForce **no es compatible** con:

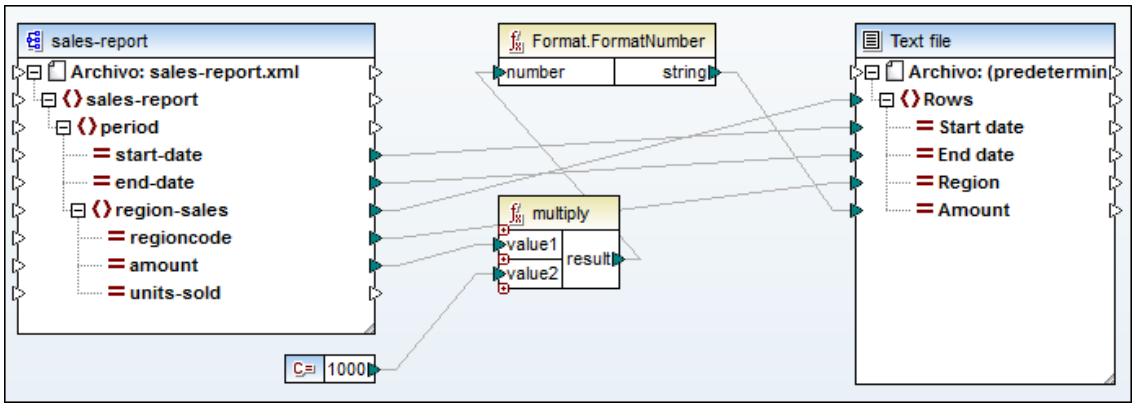
- los DLL de C++ nativos (ni otros DLL que no sean un ensamblado .NET)
- el uso de archivos o funciones `.class` o DLL mientras el lenguaje de salida seleccionado sea C++
- el uso de funciones Java o .NET directamente en XSLT (sería necesaria una función personal XSLT que haga de adaptador)

  Las funciones C++ no se pueden ver en la vista previa del panel **Resultados** pero están en el código generado.

Para agregar funciones Java o .NET personales necesita:

- las clases Java compiladas (el archivo `.class`) o
- los archivos de ensamblados .NET (archivos `.dll`).

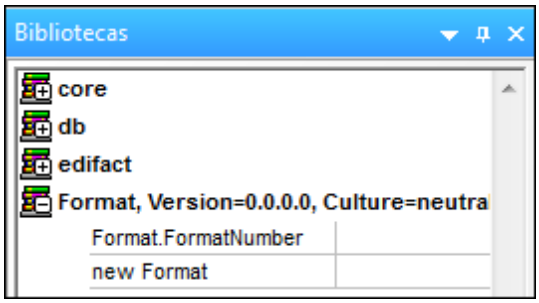
En las carpetas `...\MapForceExamples\`**`Java`** y `...\MapForceExamples\`**`C#`** encontrará archivos de ejemplo que ilustran el uso de estas funciones. Se trata de los archivos `FormatNumber.mfd`.



Para agregar el archivo de ensamblados .NET:

1. Haga clic en el botón **Agregar o quitar bibliotecas...** (en la parte inferior de la ventana

Bibliotecas) y seleccione el archivo `Format.dll` de la carpeta `...\MapForceExamples\C#\Format\bin\Debug`.
Aparece un mensaje confirmando que se agregó una función nueva. La función aparece bajo la biblioteca **format** de la ventana Bibliotecas.



Con C# seleccionado

2. Abra el archivo de asignación `FormatNumber.mfd` de la carpeta `...\MapForceExamples\C#` (*imagen anterior*).
3. 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 más información sobre la implementación de estas funciones consulte el apartado Detalles de las funciones Java y .NET.

15.2.2 Agregar funciones XSLT 1.0 personales

Los archivos necesarios para trabajar con el ejemplo de este apartado están en la carpeta `...\MapForceExamples`:

- `Name-splitter.xslt`
- `Name-splitter.xml` (el archivo XML de instancia para el esquema `Customers.xsd`)
- `Customers.xsd`
- `CompletePO.xsd`

Para ver cómo puede usar plantillas con nombre para sumar nodos consulte el apartado Funciones de agregado.

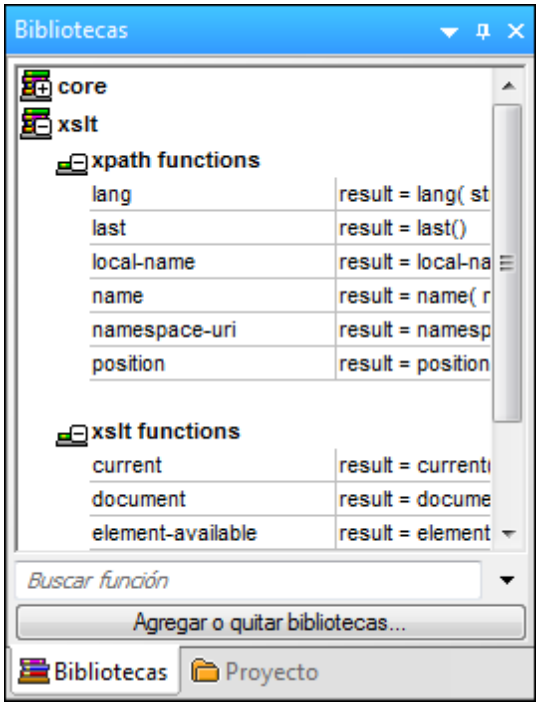
Para agregar una función XSLT personal:

1. Cree un archivo XSLT que consiga el resultado/la transformación deseada.
   Por ejemplo, el archivo `Name-splitter.xslt` (*imagen siguiente*) incluye una plantilla con nombre llamada **`tokenize`** con un solo parámetro **`string`**. Esta plantilla recorre una cadena de entrada y separa las letras mayúsculas insertando un espacio en blanco entre ellas.

```
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">
```

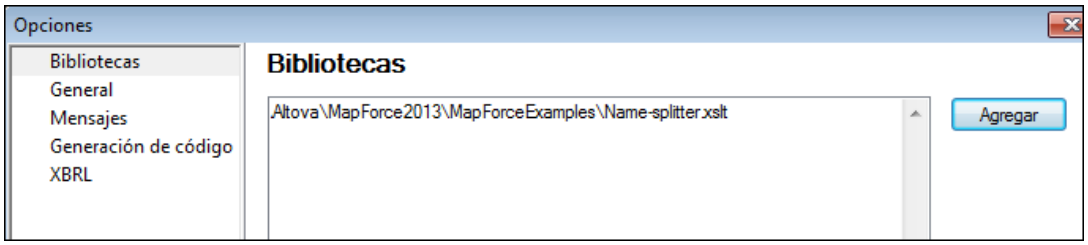
2. Haga clic en el botón **Agregar o quitar bibliotecas** y en el siguiente cuadro de diálogo haga clic en el botón **Agregar**.

 © 2014 Altova GmbH

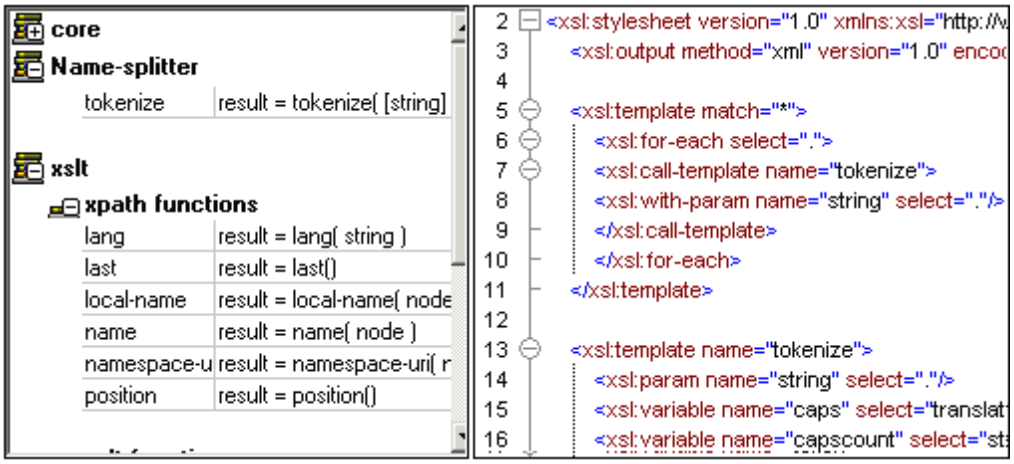


Con XSLT seleccionado

3. Seleccione el archivo XSL o XSLT que contiene la plantilla con nombre que desea usar como función (es decir, `Name-splitter.xslt`). El archivo XSLT aparece en la pestaña *Bibliotecas* del cuadro de diálogo Opciones.



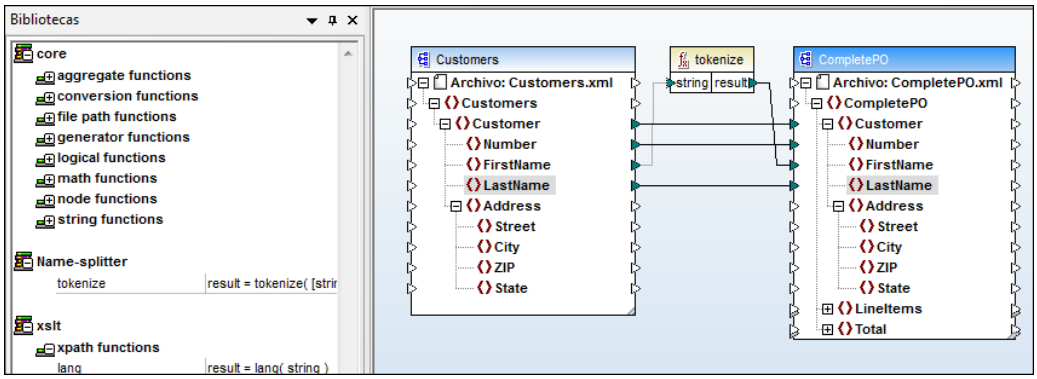
4. Haga clic en **Aceptar** para insertar la función nueva.



Con XSLT seleccionado

El nombre del archivo XSLT aparece en la ventana Bibliotecas junto con las funciones definidas como plantillas con nombre (en este caso, **Name-splitter** y la función **`tokenize`**).

5. Arrastre la función **`tokenize`** al panel **Asignación** para usarla en la asignación actual y cree las asignaciones necesarias tal y como muestra la imagen siguiente.



Con XSLT seleccionado

6. Abra el panel **XSLT** para ver el código XSLT generado.

```
-->
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform" xmlns:xsi="http://v
  <xsl:output method="xml" encoding="UTF-8"/>
  <xsl:include href="C:\Program Files\Altova\MAPFORCE2004\MapForceExamples\Name-splitter.xslt"/>
  <xsl:template match="/Customers">
    <CompletePO>
      <xsl:attribute name="xsi:noNamespaceSchemaLocation">C:/PROGRA~1/Altova/MAPFORCE200
      <xsl:for-each select="Customer">
        <Customer>
          <xsl:for-each select="Number">
            <Number>
              <xsl:value-of select="."/>
            </Number>
          </xsl:for-each>
          <xsl:for-each select="FirstName">
            <xsl:variable name="V47993824_47988944" select="."/>
            <xsl:variable name="V47993824_47939520">
              <xsl:call-template name="tokenize">
                <xsl:with-param name="string" select="$V47993824_47988944"/>
              </xsl:call-template>
            </xsl:variable>
```

Nota: en cuanto se utiliza la plantilla con nombre en la asignación, el archivo XSLT que contiene la plantilla se incluye en el código XSLT generado (`xsl:include href...`) y se le **llama** con el comando `xsl:call-template`.

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

 © 2014 Altova GmbH

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <CompletePO xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
3     <Customer>
4       <Number>1</Number>
5       <FirstName>Fred John</FirstName>
6       <LastName>Landis</LastName>
7     </Customer>
8     <Customer>
9       <Number>2</Number>
10      <FirstName>Michelle Ann-marie</FirstName>
11      <LastName>Butler</LastName>
12    </Customer>
13    <Customer>
14      <Number>3</Number>
15      <FirstName>Ted Mac</FirstName>
16      <LastName>Little</LastName>
```

Para eliminar funciones XSLT personales:

1. Haga clic en el botón **Agregar o quitar bibliotecas...**.
2. Haga clic en el nombre de biblioteca XSLT en la pestaña *Bibliotecas*.
3. Haga clic en el botón **Quitar** y después en **Aceptar** para confirmar.

15.2.3 Agregar funciones XSLT 2.0 personales

MapForce también permite importar funciones XSLT 2.0 de documentos XSLT 2.0 que tengan este formato:

<xsl:function name="MiFunción">

Para más información consulte el anexo XSLT 2.0 (información sobre implementación) y el apartado Funciones de agregado (que explica cómo usar plantillas con nombre para sumar nodos).

Tipos de datos en XPath 2.0
Si su documento XML hace referencia a un esquema XML y es válido con respecto a este esquema, debe construir explícitamente o convertir los tipos de datos que no se conviertan implícitamente al tipo de datos necesario por medio de una operación.

En el modelo de datos XPath 2.0 utilizado por el motor XSLT 2.0 de Altova, todos los valores de nodo atomizados del documento XML se asignan al tipo de datos `xs:untypedAtomic`. El tipo `xs:untypedAtomic` funciona muy bien con las conversiones implícitas de tipos.

Por ejemplo:

- la expresión `xs:untypedAtomic("1") + 1` da como resultado el valor `2` porque el operador de suma aumenta **implícitamente** el valor de `xdt:untypedAtomic` a `xs:double`.
- los operadores aritméticos aumentan los operandos a `xs:double` implícitamente.
- los operadores de comparación de valores aumentan los operandos a `xs:string` antes de la comparación.

 © 2014 Altova GmbH

15.2.4 Agregar funciones XQuery personales

MapForce permite importar módulos de bibliotecas XQuery.

Para más información consulte el anexo XQuery.

15.2.5 Funciones de agregado: sumar nodos en XSLT 1.0/2.0

Este apartado explica cómo se pueden procesar varios nodos de una instancia XML y asignar el resultado del procesamiento como valor único a un elemento de destino. Los archivos utilizados para ilustrar este ejemplo están en la carpeta `...\MapForceExamples\Tutorial`:

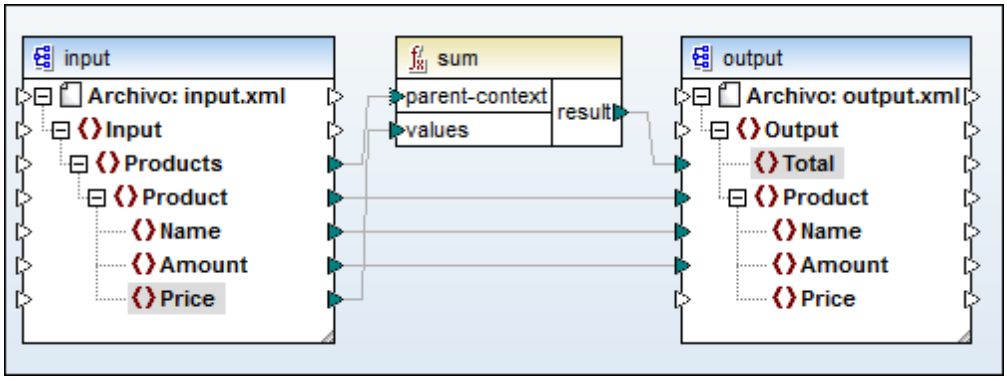
- `Summing-nodes.mfd`: el archivo de asignación
- `input.xml`: el archivo XML de entrada
- `input.xsd` y `output.xsd`: los esquemas de origen y destino respectivamente
- `Summing-nodes.xslt`: el archivo xslt que contiene una plantilla con nombre que suma los nodos

Las funciones de agregado se pueden crear y usar de dos formas diferentes:

- con las funciones de agregado de la biblioteca **core / aggregate** de la ventana Bibliotecas
- con una **plantilla con nombre**.

Biblioteca de funciones core / aggregate
Dependiendo de la biblioteca XSLT seleccionada (XSLT1 o XSLT2), la biblioteca core incluye unas funciones de agregado u otras. Mientras que XSLT 1 solamente ofrece las funciones de agregado **`count`** y **`sum`**, XSLT 2 ofrece **`avg`**, **`count`**, **`max`**, **`min`**, **`string-join`** y **`sum`**.

Arrastre la función de agregado que desea usar desde la ventana Bibliotecas hasta el panel **Asignación** y cree las conexiones que aparecen a continuación entre los componentes de origen y destino.



Para más información consulte el apartado Funciones de agregado.

Función de agregado: plantilla con nombre
La siguiente imagen muestra el archivo **XML de entrada**. El objetivo del ejemplo es sumar los campos **`Price`** de ciertos productos (en este caso de los productos A y B).

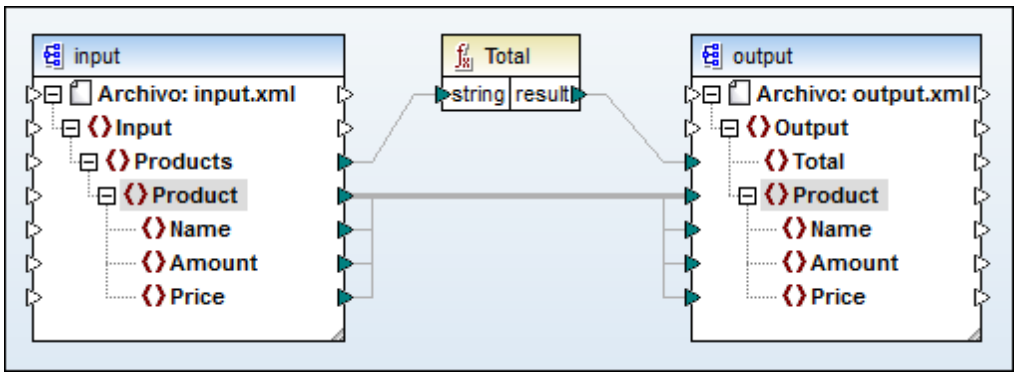
 © 2014 Altova GmbH

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <Input xmlns:xsi="http://www.w3.org/2001/XMLSchema
3       <Products>
4           <Product>
5               <Name>ProductA</Name>
6               <Amount>10</Amount>
7               <Price>5</Price>
8           </Product>
9           <Product>
10              <Name>ProductB</Name>
11              <Amount>5</Amount>
12              <Price>20</Price>
13          </Product>
14      </Products>
15  </Input>
```

La siguiente imagen muestra la hoja de estilos XSLT que utiliza la plantilla con nombre **`Total`** y un parámetro único **`string`**. La plantilla recorre el archivo XML de entrada y suma todos los valores obtenidos por la expresión XPath `/Product/Price`.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/19
    <xsl:output method="xml" version="1.0" encoding="UTF-8" i

    <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>
```

1. Haga clic en el botón **Agregar o quitar bibliotecas**. En el nuevo cuadro de diálogo haga clic en el botón **Agregar**.
2. Seleccione el archivo `Summing-nodes.xslt` de la carpeta `...\MapForceExamples\Tutorial` y haga clic en **Aceptar**.
3. Arrastre la función **`Total`** de la biblioteca recién creada (**Summing-nodes**) y cree las conexiones que aparecen en esta imagen.



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

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <Output xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNa
3    <Total>25</Total>
4    <Product>
5      <Name>ProductA</Name>
6      <Amount>10</Amount>
7      <Price>5</Price>
8    </Product>
9    <Product>
10     <Name>ProductB</Name>
11     <Amount>5</Amount>
12     <Price>20</Price>
13   </Product>
14 </Output>
15
```

Los campos **`Price`** de ambos productos se sumaron y el resultado se colocó en el campo **`Total`**.

Para sumar los nodos en XSLT 2.0:

- Cambie la declaración de la hoja de estilos en la plantilla por `... version="2.0"`.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <xsl:stylesheet version="2.0" xmlns:xs
3      <xsl:output method="xml" version="
```

 © 2014 Altova GmbH

15.3 Agregar bibliotecas personales de funciones Java, C# y C++

En MapForce puede crear y agregar bibliotecas de funciones personales para Java, C# y C++. Una vez creadas, estas funciones se pueden usar igual que las demás funciones integradas de MapForce.

Las bibliotecas se añaden con el botón **Agregar o quitar bibliotecas...** situado en la parte inferior de la ventana Bibliotecas o en el cuadro de diálogo "Opciones" (**Herramientas | Opciones**, pestaña *Bibliotecas*, botón **Agregar**). Las bibliotecas pueden ser archivos `.class` de Java, archivos `.DLL` de Visual Studio C# y archivos `.mff` (archivos de funciones de MapForce).

Nota sobre los archivos de funciones .MFF
Muchas asignaciones que usan este tipo de funciones personales (`.mff`) pueden verse en el panel Resultados (es decir, se puede obtener una vista previa de su resultado con el motor de ejecución integrado). Esta característica también funciona con las funciones C# y Java, pero solamente aquellas que usan tipos nativos, no las que usan clases generadas de Altova.

Todas estas funciones están disponibles cuando se genera código en MapForce.

Las funciones definidas por el usuario creadas de forma gráfica en una **asignación** no se pueden ni se deben guardar/asignar en un archivo `*.mff` porque se guardan en el archivo de asignación directamente. Para más información sobre la importación y gestión de funciones definidas por el usuario consulte este apartado.

Nota: el ejemplo que aparece a continuación utiliza Java SE Runtime Environment.

Para poder agregar funciones personales de archivos *.mff necesitará el archivo *.mff:

- que le dice a MapForce cuáles son las interfaces para las funciones
- que le dice a MapForce dónde se puede encontrar la implementación para el código generado. Esta implementación es una clase en el lenguaje de programación correspondiente que contiene los métodos estáticos definidos en el archivo `.mff`.

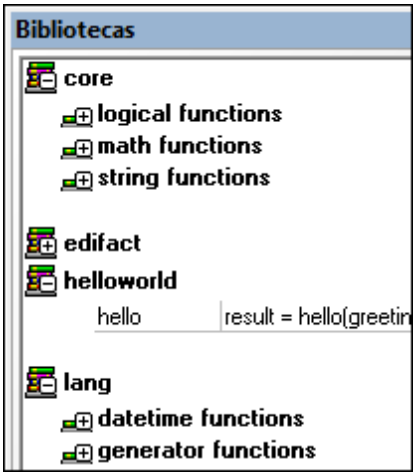
Por ejemplo, a continuación puede ver el aspecto que tiene un archivo `.mff` básico para C#:

```
<?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>
```

Nota: los archivos de bibliotecas `.mff` deben ser válidos con respecto al archivo de esquema `mff.xsd` de la carpeta `...\MapForceLibraries`. Este esquema define la configuración de la biblioteca y solamente es de uso interno. Altova GmbH se reserva el derecho de cambiar el formato de este archivo en las nuevas versiones del producto.

La siguiente imagen muestra el archivo `mff` anterior en MapForce. La biblioteca nueva **helloworld** aparece en la ventana Bibliotecas y contiene la función de cadena **`hello`**.



Con Java seleccionado

Los archivos `.mff` se pueden escribir para varios lenguajes de programación. Cada lenguaje adicional debe contener un elemento `<implementation>` adicional. Más adelante explicaremos en detalle el elemento `<implementation>`.

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 solamente utilice funciones que no tengan efectos secundarios.

 © 2014 Altova GmbH

15.3.1 Configurar el archivo MFF

A continuación describimos cómo puede adaptar un archivo `.mff` a sus requisitos.

El nombre de la biblioteca
En el archivo `.mff` el nombre de la biblioteca aparece en la siguiente línea:

```
<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">
```

Convencionalmente, el **nombre de la biblioteca** se escribe en **minúsculas**.

La entrada que aparecerá en la ventana Bibliotecas de MapForce se llamará **helloworld**. Recuerde que puede que la biblioteca no aparezca **nada más** hacer clic en el botón **Agregar** del cuadro de diálogo Opciones. Las bibliotecas aparecen en la ventana Bibliotecas solamente si uno de los componentes de la asignación contiene una implementación para el lenguaje de programación seleccionado actualmente.

Las bibliotecas y sus funciones se pueden activar/desactivar agregando o eliminando el archivo de biblioteca (`*.mff`) correspondiente.

Para agregar un archivo .mff a la ventana Bibliotecas:

1. Haga clic en el botón **Agregar o quitar bibliotecas...**.
2. En el cuadro de diálogo "Opciones" haga clic en el botón **Agregar**.
3. Seleccione la biblioteca `*.mff` que desea incluir y haga clic en **Abrir** para cargar el archivo en el cuadro de diálogo "Opciones".
4. Haga clic en **Aceptar** para confirmar.

Nota: si guarda el archivo `*.mff` en la carpeta `...\MapForceLibraries`, la biblioteca se cargará automáticamente cuando inicie MapForce y podrá usarla inmediatamente en todas las asignaciones.

Elementos `implementation` para la biblioteca helloworld:

```
...
<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>
...
```

Es necesario agregar un elemento `implementation` para cada lenguaje de programación con el que desea que sea compatible la biblioteca **helloworld**. La configuración de cada elemento `implementation` permite al código generado llamar a la función correspondiente definida en Java, C++ o C#.

A continuación describimos la configuración propia de 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 el archivo `Hello.class`. Para ello asegúrese de introducirlo en la variable CLASSPATH de Java. La variable classpath predeterminada está en las variables de entorno del sistema.

Recuerde que, cuando se trabaja con bibliotecas Java personales, solamente 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 código utilice el espacio de nombres definido aquí. C# también necesita conocer la ubicación del **`dll`** que se debe enlazar con el 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>
...
```

- **`namespace`** es el espacio de nombres en el que se definirá la clase **`Greetings`**. Debe ser igual al nombre de la biblioteca.
- **`path`** es la ruta de acceso en la que se encuentran los archivos de origen y de inclusión.
- cuando se genera código para una asignación, los archivos de origen y de inclusión se copian al directorio `dirdestino/`**`nombrebiblioteca`** (para definirlo haga clic en **Archivo | Generar código `xxx`** y seleccione el directorio de destino) y se incluyen en el archivo de proyecto.

Todos los archivos de inclusión suministrados se incluyen en el algoritmo generado.

Agregar un componente:
Cada componente que defina en la función se ubica dentro de un grupo de función. En el caso de la biblioteca **helloworld**:

```
...
<group name="string functions">
    <component name="hello">
        …
    </component>
</group>
...
```

15.3.2 Definir la interfaz de usuario del componente

El fragmento de código que aparece a continuación define cómo aparece un componente cuando se inserta en el panel **Asignación** de MapForce.

```
...
<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>
...
```

El nuevo componente de MapForce:

Puntos de datos
Los puntos de datos se pueden definir libremente como parámetros de entrada o salida de una función. El parámetro type de los puntos de datos especifica el tipo de parámetro/valor devuelto.

Nota: para cada función se permite un solo punto de datos de **destino** y varios puntos de datos de **origen**.

Los tipos de datos compatibles para los puntos de datos son:

- los tipos de XML Schema (p. ej. `xs:string`, `xs:integer`, etc.)

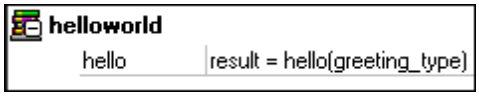
Estos tipos de datos deben corresponderse con los tipos de datos de los parámetros de la función definidos en la biblioteca Java, C++ o C#. Para la correspondencia entre los tipos de datos de XML Schema y los tipos del lenguaje, consulte las tablas del apartado Cómo escribir sus propias bibliotecas.

Altova ofrece compatibilidad con el uso de tipos simples de XML Schema (`date, time, duration, dateTime`) como clases en cada uno de los lenguajes compatibles. La integración de estos tipos simples en su biblioteca se explica más adelante.

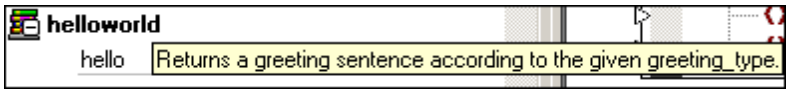
Descripción de las funciones
En la ventana Bibliotecas de MapForce las funciones aparecen acompañadas de una descripción breve y de otra más larga. La descripción breve aparece a la derecha del nombre de la función en la ventana Bibliotecas. Por su parte, la descripción más larga aparece al pasar el puntero del mouse por encima de la descripción breve.

Descripción breve:

 © 2014 Altova GmbH



Descripción larga:



15.3.3 Detalles de la implementación de la función

Llegados a este punto necesitamos crear una conexión entre la función en la ventana Bibliotecas y la función en las clases Java, C# o C++. Esto se consigue con el elemento `<implementation>`.

Como decíamos, una función puede tener varios elementos `<implementation>`, uno por cada lenguaje de programación compatible.

```
...
<component name="hello">
...
   <implementations>
      <implementation language="cs">
         <function name="HelloFunction"/>
      </implementation>
   </implementations>
...
</component>
...
```

La misma función puede llamarse `helloFunction` en Java y `HelloFunctionResponse` en C++. Por eso es necesario especificar un nombre de función distinto para cada lenguaje de programación.

En ese caso la función tendría este aspecto:

```
...
<component name="hello">
...
   <implementations>
      <implementation language="cs">
         <function name="HelloFunction"/>
      </implementation>
      <implementation language="java">
         <function name="helloFunction"/>
      </implementation>
      <implementation language="cpp">
         <function name="HelloFunctionResponse"/>
      </implementation>
   </implementations>
...
</component>
...
```

El valor dado como nombre de la función debe coincidir con el nombre del método de la clase Java, C# o C++.

 © 2014 Altova GmbH

15.3.4 Cómo escribir sus propias bibliotecas

La implementación de una biblioteca de funciones personal es una clase en el lenguaje de programación correspondiente que contiene los métodos estáticos para cada una de las funciones definidas en el archivo `.mff`.

Nota: si actualizó su versión de MapForce a la versión 2010, puede que necesite actualizar los tipos de datos utilizados en sus funciones personales. La correspondencia actual entre tipos de XML Schema y tipos de datos nativos aparece en los apartados siguientes.

En los siguientes apartados describimos cómo se crean bibliotecas para cada lenguaje de programación compatible:

Crear una biblioteca Java
Crear una biblioteca C#
Crear una biblioteca C++

Crear una biblioteca Java

Cómo se escribe una biblioteca Java:

1. Cree una clase Java nueva usando el ejemplo del apartado anterior y llámela **Hello**.
2. Añada el nombre del paquete bajo

```
...
<implementation language="java">
    <setting name="package" value="com.hello.functions"/>
    <setting name="class" value="Hello"/>
</implementation>
...
```

3. Si necesita tipos de XML Schema especiales (p. ej. `date`, `duration`, etc.), añada la línea `import com.altova.types.*;`

   La correspondencia exacta entre tipos de datos de XML Schema y tipos de datos Java aparece en una tabla al final de este apartado.

   Si tiene problemas para encontrar **com.altova.types** en su equipo, genere y compile el código Java sin las funciones personales y encontrará las clases en el directorio especificado.

4. Añada las funciones especificadas en el archivo `.mff` como método **public static**.

```
package com.hello.functions;

public class Hello {
    public static String HelloFunction ( boolean greetingType ) {
        if( greetingType )
            return "Hello World!";
        return "Hello User!";
    }
}
```

5. Compile el archivo Java en un archivo de clases y añádalo a su classpath. La biblioteca personal ya está lista.

Correspondencia de tipos de datos

Tipo de XML Schema	Tipo Java
anySimpleType	String
anyAtomicType	String
boolean	boolean
string	String
normalizedString	String
token	String
language	String
NMTOKEN	String
Name	String
NCName	String
ID	String
IDREF	String
ENTITY	String
untypedAtomic	String
dateTime	com.altova.types.DateTime
date	com.altova.types.DateTime
time	com.altova.types.DateTime
gYear	com.altova.types.DateTime
gYearMonth	com.altova.types.DateTime
gMonth	com.altova.types.DateTime
gMonthDay	com.altova.types.DateTime
gDay	com.altova.types.DateTime
duration	com.altova.types.Duration
base64Binary	byte[]
hexBinary	byte[]
anyURI	String
QName	javax.xml.namespace.QName
NOTATION	String
double	double
float	double
decimal	java.math.BigDecimal
integer	java.math.BigInteger
nonPositiveInteger	java.math.BigInteger
negativeInteger	java.math.BigInteger
long	long
int	int
short	int
byte	int
nonNegativeInteger	java.math.BigInteger
positiveInteger	java.math.BigInteger
unsignedLong	java.math.BigInteger
unsignedInt	long
unsignedShort	long
unsignedByte	long
dayTimeDuration	com.altova.types.Duration
yearMonthDuration	com.altova.types.Duration
NMTOKENS	String
IDREFS	String
ENTITIES	String

© 2014 Altova GmbH

Crear una biblioteca C#

Cómo se escribe una biblioteca C#:

1. Abra un proyecto nuevo en Visual Studio y cree una biblioteca de clases.
2. Haga clic en **Agregar referencia** y añada `Altova.dll`.

   Si tiene problemas para encontrar `Altova.dll` en su equipo, genere y compile el código C# sin las funciones personales y encontrará el DLL en el directorio especificado.

3. Si necesita tipos de XML Schema especiales (p. ej. `date`, `duration`, etc.), añada la línea
   `using Altova.Types;`

   La correspondencia exacta entre tipos de datos de XML Schema y tipos de datos C# aparece en una tabla al final de este apartado.

4. El nombre de la clase debe ser idéntico al especificado (es decir, **Greetings**)

```
   <implementation language="cs">
      <setting name="namespace" value="HelloWorldLibrary"/>
      <setting name="class" value="Greetings"/>
      <setting name="reference" value="C:\HelloWorldLibrary
\HelloWorldLibrary.dll"/>
   </implementation>
```

5. Añada el espacio de nombres usando el mismo valor que especificó en el `setting` de `implementation` en el archivo `.mff` (*fragmento de código anterior*).
6. Añada sus funciones como método **public static.**

   El código debería tener este aspecto:

```
using System;
using Altova.Types;

namespace HelloWorldLibrary
{
       public class Greetings
       {
              public static string HelloFunction(bool GreetingType)
              {
                     if( GreetingType )
                            return "Hello World!";
                     return "Hello User!";
              }
       }
}
```

7. El último paso es compilar el código. La ruta donde se ubica el DLL compilado debe ser el mismo que el `setting reference` del elemento `implementation`.

Correspondencia de tipos de datos

Tipo de XML Schema	Tipo C#
anySimpleType	string
anyAtomicType	string
boolean	bool
string	string
normalizedString	string
token	string
language	string
NMTOKEN	string
Name	string
NCName	string
ID	string
IDREF	string
ENTITY	string
untypedAtomic	string
dateTime	Altova.Types.DateTime
date	Altova.Types.DateTime
time	Altova.Types.DateTime
gYear	Altova.Types.DateTime
gYearMonth	Altova.Types.DateTime
gMonth	Altova.Types.DateTime
gMonthDay	Altova.Types.DateTime
gDay	Altova.Types.DateTime
duration	Altova.Types.Duration
base64Binary	byte[]
hexBinary	byte[]
anyURI	string
QName	Altova.Types.QName
NOTATION	string
double	double
float	double
decimal	decimal
integer	decimal
nonPositiveInteger	decimal
negativeInteger	decimal
long	long
int	int
short	int
byte	int
nonNegativeInteger	decimal
positiveInteger	decimal
unsignedLong	ulong
unsignedInt	ulong
unsignedShort	ulong
unsignedByte	ulong
dayTimeDuration	Altova.Types.Duration
yearMonthDuration	Altova.Types.Duration
NMTOKENS	string
IDREFS	string
ENTITIES	string

 © 2014 Altova GmbH

Crear una biblioteca C++

Cómo se escribe una biblioteca C++:

Cree los archivos **h** y **cpp** usando el mismo nombre y en la misma ubicación definida en el elemento `implementation` para toda la biblioteca.

Archivo de encabezado:

1. Escriba `using namespace altova;`
2. Añada el espacio de nombres especificado en el elemento `implementation`.
3. Añada la clase especificada en el elemento `implementation` en el archivo `.mff`, con las funciones estáticas especificadas en el archivo `.mff`.
4. No olvide escribir `ALTOVA_DECLSPECIFIER` delante del nombre de la clase para garantizar que las clases se compilan correctamente (independientemente de si usa una vinculación dinámica o estática en el código generado).
5. La correspondencia exacta entre tipos de datos de XML Schema y tipos de datos C++ aparece en una tabla al final de este apartado.

El **archivo de encabezado** resultante debería tener este aspecto:

```
#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
```

En el archivo **cpp**:

1. Las primeras líneas deben ser las inclusiones para **StdAfx.h** y las definiciones de la biblioteca base **Altova**. Copie estas líneas del código de muestra que aparece más abajo.
2. La ruta de acceso `../Altova` es correcta para sus archivos de origen porque se copiarán en un proyecto distinto en el código resultante, que estará en `dirdestino/nombrebibiloteca`.
3. La siguiente línea es la inclusión para el archivo de encabezado creado al principio.
4. Añada las implementaciones para sus funciones.
5. No olvide que las implementaciones deben estar en el espacio de nombres especificado en el archivo de encabezado y en el elemento `implementation` del archivo `.mff`.

El archivo **cpp** debería tener este aspecto:

```
#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!");
    }

}
```

A diferencia de Java o C#, en C++ no es necesario compilar los archivos de origen. Estos archivos se copiarán al código generado y se compilan con el resto de código de asignación.

Nota: si recibe un error de compilación en la línea que aparece a continuación, añada la ruta de acceso a **msado15.DLL**

```
#import "msado15.dll" rename("EOF", "EndOfFile")
```

Debe añadir donde se almacena **msado15.dll** la ruta de acceso a la sección de directorios de su entorno Visual Studio:

1. En Visual Studio seleccione **Herramientas | Opciones**
2. Haga clic en la pestaña *Directorios*.
3. En la lista desplegable *Mostrar directorios para* seleccione la opción **Archivos de inclusión**.
4. Añada una línea nueva con la ruta de acceso del archivo (por lo general `c:\Archivos de programa\Archivos comunes\System\ADO`)
5. Vuelva a generar el código.

 © 2014 Altova GmbH

Correspondencia de tipos de datos

Tipo de XML Schema	Tipo C++
anySimpleType	string_type
anyAtomicType	string_type
boolean	bool
string	string_type
normalizedString	string_type
token	string_type
language	string_type
NMTOKEN	string_type
Name	string_type
NCName	string_type
ID	string_type
IDREF	string_type
ENTITY	string_type
untypedAtomic	string_type
dateTime	altova::DateTime
date	altova::DateTime
time	altova::DateTime
gYear	altova::DateTime
gYearMonth	altova::DateTime
gMonth	altova::DateTime
gMonthDay	altova::DateTime
gDay	altova::DateTime
duration	altova::Duration
base64Binary	altova::mapforce::blob
hexBinary	altova::mapforce::blob
anyURI	string_type
QName	altova::QName
NOTATION	string_type
double	double
float	double
decimal	double
integer	__int64
nonPositiveInteger	__int64
negativeInteger	__int64
long	__int64
int	int
short	int
byte	int
nonNegativeInteger	unsigned __int64
positiveInteger	unsigned __int64
unsignedLong	unsigned __int64
unsignedInt	unsigned __int64
unsignedShort	unsigned __int64
unsignedByte	unsigned __int64
dayTimeDuration	altova::Duration
yearMonthDuration	altova::Duration
NMTOKENS	string_type
IDREFS	string_type
ENTITIES	string_type

15.4 Detalles de las funciones Java y .NET

Detalles de implementación

Agregar bibliotecas:

- puede añadir archivos `.class` de Java a MapForce (los archivos `.jar` no son compatibles)
- puede añadir archivos de ensamblados .NET `.dll` a MapForce

Advertencia: todas las funciones a las que se llame desde una asignación de MapForce deben ser ***idempotentes***. Esto significa que deberían devolver el mismo valor cada vez que se llama a la función con los mismos parámetros de entrada. MapForce llama a las funciones en un orden y frecuencia indeterminados.

Funciones Java: configuración
Si los archivos de clases Java importados dependen de otros archivos de clases, asegúrese de ajustar la variable de entorno CLASSPATH antes de iniciar MapForce. Los directorios primarios de todos los paquetes dependientes deberían añadirse a la variable CLASSPATH.

No hace falta añadir los archivos de clases importados y sus paquetes a la variable CLASSPATH porque el motor de ejecución integrado (así como el código Java generado) añade los paquetes importados automáticamente a la CLASSPATH o ANT del motor Java.

Funciones Java compatibles

Son compatibles las clases de nivel superior, los miembros estáticos y los miembros no estáticos:

- `new <nombreClase>(<arg1>, <arg2>, ...)`
- `<objeto>.new <clase-miembro>(<arg1>, <arg2>, ...)`

Son compatibles las funciones estáticas y las funciones de miembro:

- `<función>(<arg1>, <arg2>, ...)`
- `<objeto>.<método>(<arg1>, ...)`

Tipos
Correspondencias entre los tipos de XML Schema y Java:

Tipo de 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

 © 2014 Altova GmbH

xs:decimal	java.math.BigDecimal
xs:integer	java.math.BigInteger

Los demás tipos de Java no son compatibles, tampoco los tipos de matriz. Los métodos que usen dichos parámetros o valores devueltos se pasan por alto.

Los tipos de objeto son compatibles mediante la llamada a su constructor o como valor devuelto de un método. Pueden asignarse a otros métodos Java. No es posible manipular el objeto en MapForce.

Funciones .NET compatibles

Son compatibles las clases de nivel superior y las clases de miembros:

- `new <nombreClase>(<arg1>, <arg2>, ...)`

Son compatibles las funciones estáticas y las funciones de miembro:

- `<función>(<arg1>, <arg2>, ...)`
- `<objeto>.<método>(<arg1>, ...)`

Tipos:

Correspondencias entre los tipos de XML Schema y .NET/C#:

Tipo de XML Schema	Tipo .NET	Tipo C#
xs:string	System.String	string
xs:byte	System.SByte	sbyte
xs:short	System.Int16	short
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

Los demás tipos .NET/C# no son compatibles, tampoco los tipos de matriz. Los métodos que usen dichos parámetros o valores devueltos se pasan por alto.

Los tipos de objeto son compatibles mediante la llamada a su constructor o como valor devuelto de un método. Pueden asignarse a otros métodos Java. No es posible manipular el objeto en MapForce.

© 2014 Altova GmbH

15.5 Referencia de funciones

En esta sección se describen todas las funciones disponibles en la ventana Bibliotecas para cada uno de los lenguajes compatibles: XSLT1, XSLT2, XQuery, Java, C# y C++, así como el motor de ejecución integrado.

MapForce ofrece estas bibliotecas:

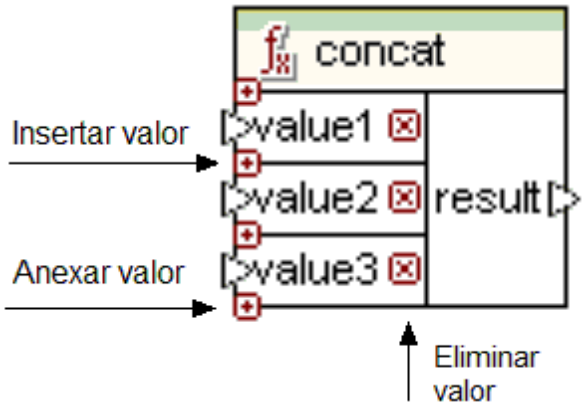
- **core**
- **db**
- **edifact**
- **lang**
- **xbrl**
- **xlsx**
- **xpath2**
- **xslt**

Funciones ampliables

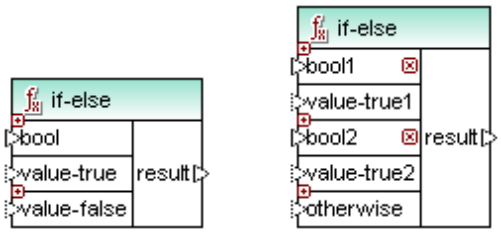
Hay varias funciones que se pueden ampliar: p. ej. las funciones **`concat`**, **`logical-and`**, **`logical-or`** e **`IF-ELSE`**. Esto significa que en este tipo de funciones puede insertar/anexar/eliminar parámetros dependiendo de sus requisitos.

Al hacer clic en el icono **+** se inserta/anexa el mismo tipo de parámetro. Asimismo, al hacer clic en la marca **x** se elimina el parámetro.

Nota: al soltar un conector en el icono **+** de la función, se inserta/anexa otro parámetro y la conexión se crea automáticamente.



Los parámetros de prueba IF de la función **`IF-Else`** también se pueden ampliar de esta forma.



Pase el puntero del mouse sobre la barra de título de la función para ver su descripción.

Pase el puntero del mouse sobre un parámetro de la función para ver el tipo de datos del argumento.

© 2014 Altova GmbH

15.5.1 Biblioteca core

La biblioteca **core** ofrece las funciones más útiles en todos los lenguajes. Las funciones de secuencia no están disponibles si el lenguaje seleccionado es XSLT 1.0.

Biblioteca core

- funciones de agregado
- funciones de conversión
- funciones de ruta de acceso
- funciones generador
- funciones lógicas
- funciones matemáticas
- funciones de nodo
- funciones de secuencia
- funciones de cadena

aggregate (funciones de agregado)

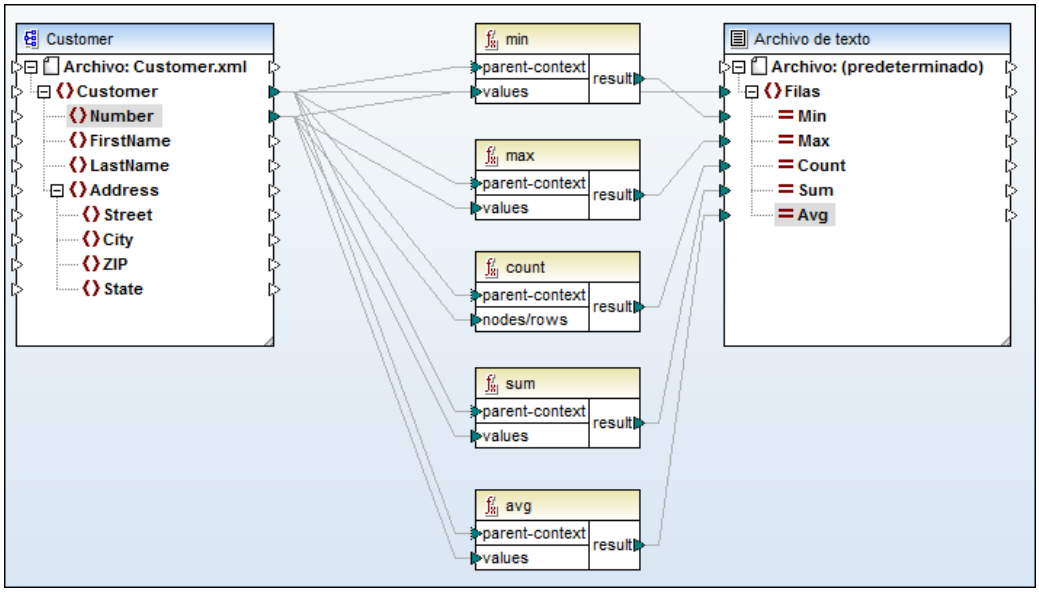
Las funciones de agregado de la biblioteca **core | aggregate** realizan operaciones en un conjunto o secuencia de valores de entrada. Los datos de entrada para las funciones **min**, **max**, **sum** y **avg** se convierten al tipo de datos `decimal` para poder procesarlos.

- 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. Esto también significa que el parámetro **parent-context** es un parámetro opcional.
- El parámetro **result** de la función debe conectarse al elemento de destino.

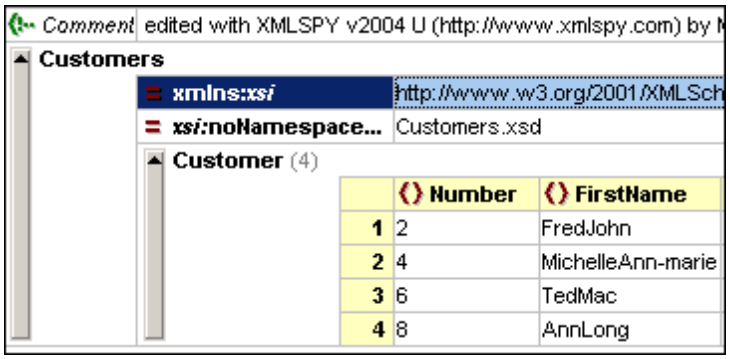
La asignación que aparece a continuación es la del archivo `Aggregates.mfd`, guardado en la carpeta `...\Tutorial` y explica cómo utilizar este tipo de funciones.

Las funciones de agregado tienen dos entradas:

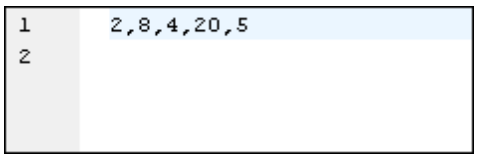
- la entrada **values** (**nodes/rows**) se conecta al elemento del componente de origen que suministra los datos, en este caso el elemento **Number**.
- la entrada **parent-context** se conecta al elemento en el que se desea iterar (es decir, el contexto. En este caso se quiere iterar en todos los elementos **Customer**). Este parámetro es opcional.



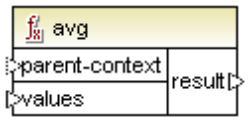
La instancia de entrada, en este caso, es un archivo XML que contiene estos datos:



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:

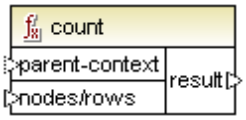


min=2, max=8, count=4, sum=20 y avg=5.



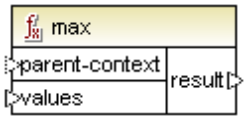
avg

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.



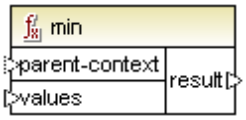
count

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.



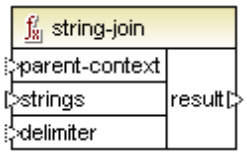
max

Devuelve el valor máximo de todos los valores de la secuencia de entrada. El máximo de un conjunto vacío es un conjunto vacío. Esta función no está disponible en XSLT1.



min

Devuelve el valor mínimo de todos los valores de la secuencia de entrada. El mínimo de un conjunto vacío es un conjunto vacío. Esta función no está disponible en XSLT1.

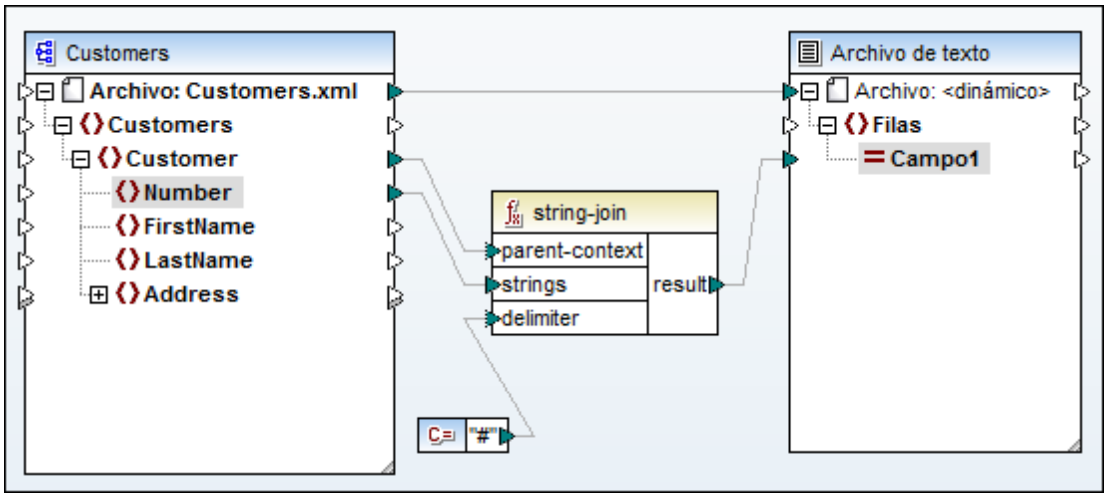


string-join

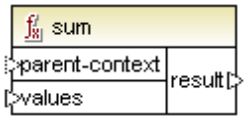
Enlaza todos los valores de la secuencia de entrada y forma una cadena delimitada por la cadena suministrada como delimitador al parámetro **`delimiter`**. Cuando se aplica a un conjunto vacío el resultado es una cadena vacía. Esta función no está disponible en XSLT1.

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 conecta una constante al parámetro **`delimiter`**, se usa una cadena vacía como delimitador (es decir, el resultado sería `2468`).



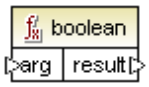
sum

Devuelve la suma aritmética de todos los valores de la secuencia de entrada. La suma de un conjunto vacío es 0. Esta función no está disponible en XSLT1.

conversion (funciones de conversión)

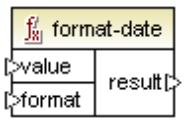
MapForce ofrece las funciones **`boolean`**, **`number`** y **`string`** de conversión de tipos de la biblioteca **conversion** para realizar conversiones explícitas de tipos de datos. Tenga en cuenta que, en la mayoría de las asignaciones, MapForce puede crear conversiones automáticamente y que solamente necesitará usar estas funciones en casos muy concretos.

parámetros de entrada = **`arg/value`**
parámetro de salida = **`result`**



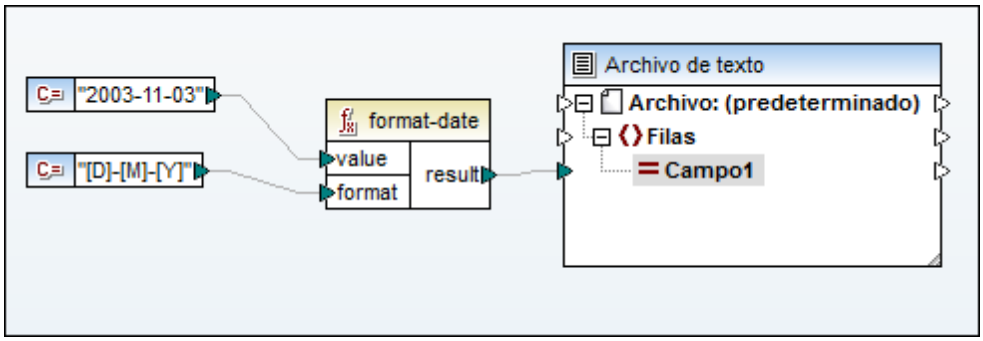
boolean

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`**.



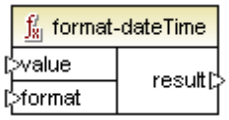
format-date

Convierte un valor de entrada `xs:date` en un valor de cadena.



Resultado: 3-11-2003

 © 2014 Altova GmbH



format-dateTime

Convierte un valor `dateTime` en un valor de cadena.

Argumento	Descripción
`value`	el valor `dateTime` al que se debe dar formato
`format`	una cadena de formato que indica de qué modo se debe dar formato al valor `dateTime`

Nota: si el resultado de la función (es decir, el parámetro **`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. Para ello desactive la casilla *Convertir valores en tipos de destino* en el cuadro de diálogo "Configuración" del componente de destino.

Cadena de formato

El argumento de formato 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 (dependiente 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 (dependiente del idioma)
m	minutos de una hora	00-59
s	segundos de un minuto	00-59
f	segundo fraccionario	numérica, con un decimal
Z	uso horario como diferencia horaria de UTC	+08:00 o PST con modificador alfabético
z	uso horario como diferencia horaria usando GMT	GMT+n

El modificador de formato:

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
n	nombre del componente en minúsculas	lunes, martes **(1)**
Nn	nombre del componente, primera letra en mayúsculas	Lunes, Martes **(1)**

(1) 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:

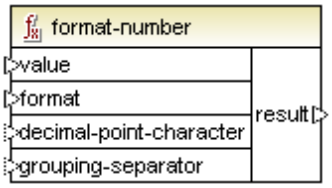
```
, min-width ("-" max-width)?
```

Ejemplos compatibles

DateTime	formato de cadena	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]	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

 © 2014 Altova GmbH

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



format-number

Disponible en XSLT 1.0, XSLT 2.0, Java, C#, C++ y el motor de ejecución integrado.

Argumento	Descripción
`value`	el número al que se debe aplicar formato
`format`	una cadena de formato que identifica cómo se debe dar formato al número
`decimal-point-format`	el carácter que se debe usar como carácter de punto decimal. El carácter predeterminado es "." (opcional)
`grouping-separator`	el separador/delimitador utilizado para separar los grupos de números. El carácter predeterminado es "," (opcional)

Nota: si el resultado de la función (es decir, el parámetro **`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. Para ello desactive la casilla *Convertir valores en tipos de destino* en el cuadro de diálogo "Configuración" del componente de destino.

format

una cadena de formato que identifica cómo se debe dar formato al número. Este es su formato:

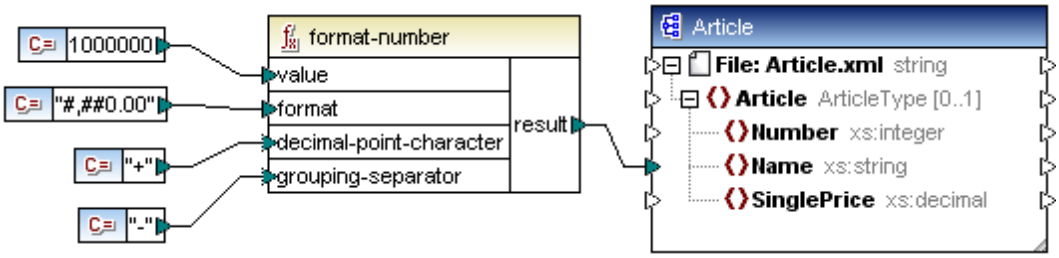
```
format := subformat (;subformat)?
  subformat := (prefix)? integer (.fraction)? (suffix)?
  prefix := cualquier carácter excepto un carácter especial
  suffix := cualquier carácter excepto un carácter especial
  integer := (#)* (0)* ( permitiendo aparecer a ',')
  fraction := (0)* (#)* ( permitiendo aparecer a ',')
```

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	Predetermi	Descripción

	nado	
dígito cero	0	en este punto del resultado siempre 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

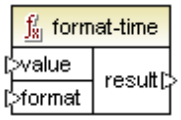
El método de redondeo utilizado por esta función es al alza. Es decir, redondea al alza si la fracción es mayor o igual a 0.5. Redondea a la baja si la fracción es menor que 0.5. Este método de redondeo solamente afecta al código generado y si la opción seleccionada es el motor de ejecución integrado.

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.

Número	formato de cadena	Resultado

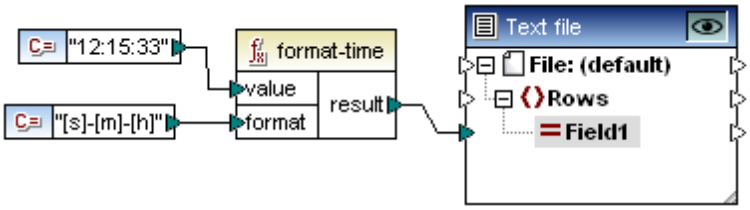
1234.5	#,##0.00	1,234.50
123.456	#,##0.00	123.46
1000000	#,##0.00	1,000,000.00
-59	#,##0.00	-59.00
1234	###0.0###	1234.0
1234.5	###0.0###	1234.5
.00025	###0.0###	0.0003
.00035	###0.0###	0.0004
0.25	#00%	25%
0.736	#00%	74%
1	#00%	100%
-42	#00%	-4200%
-3.12	#.00;(#.00)	(3.12)
-3.12	#.00;#.00CR	3.12CR



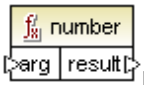
format-time

Convierte un valor de entrada `xs:time` en un valor de cadena

Por ejemplo

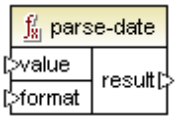


Resultado: 33-15-12



number

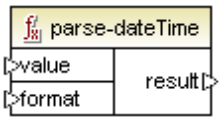
Convierte una cadena de entrada en un número. También convierte una entrada booleana en un número.



parse-date

Disponible para Java, C#, C++ y motor de ejecución integrado.

Convierte una cadena en una fecha, pero pasa por alto el componente de hora. Esta función se basa en la función **`parse-dateTime`**, pero pasa por alto el componente de hora. El resultado es de tipo `xs:date`.



parse-dateTime

Disponible en Java, C#, C++ y motor de ejecución integrado.

Convierte valores `string` en `dateTime`.

Argumento	Descripción
`value`	la cadena que contiene la fecha y hora
`format`	la cadena de imagen que define el formato actual del valor

Los componentes son:

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
H	hora (24 horas)	0-23
h	hora (12 horas)	1-12
P	A.M. o P.M.	alfabética (dependiente del idioma)
m	minutos de una hora	00-59
s	segundos de un minuto	00-59
f	segundo fraccionario	numérico, con un decimal
Z	uso horario como diferencia horaria de UTC	+08:00 o PST con modificador alfabético
z	uso horario como diferencia horaria de GMT	GMT+n

El modificador de formato:

Carácter	Descripción	Ejemplo
1	formato numérico decimal sin ceros iniciales: 1, 2, 3, ...	1, 2, 3
01	formato decimal, con dos dígitos: 01, 02, 03, ...	01, 02, 03

 © 2014 Altova GmbH

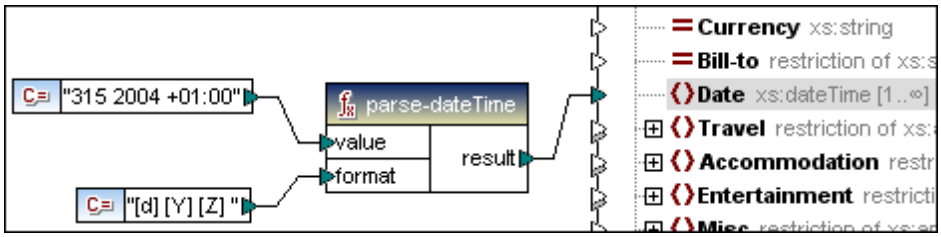
N	nombre del componente, todo en mayúsculas	LUNES, MARTES
n	nombre del componente, en minúsculas	lunes, martes
Nn	nombre del componente, la primera letra en mayúsculas	Lunes, Martes

1) Los modificadores **N, n** y **Nn** solamente son compatibles con el componente **M** (mes).

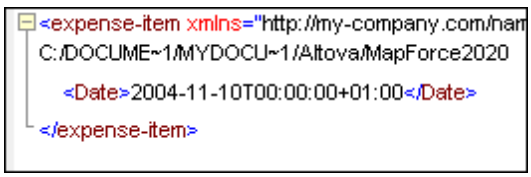
Ejemplos:

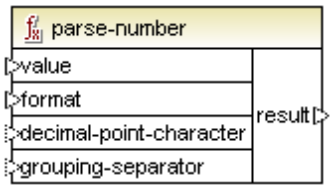
Cadena de fecha	Cadena de imagen	Resultado
21-03-2002 16:21:12.492 GMT+02:00	[D]-[M]-[Y] [H]:[m]:[s].[f] [z]	2002-03-21T16:21:12.492+02:00
315 2004 +01:00	[d] [Y] [Z]	2004-11-10T00:00:00+01:00
1.December.10 03:2:39 p.m. +01:00	[D].[MNn].[Y,2-2] [h]:[m]:[s] [P] [Z]	2010-12-01T15:02:39+01:00
20110620	[Y,4-4][M,2-2][D,2-2]	2011-06-20T00:00:00

Ejemplo en MapForce:



Resultado:





parse-number

Disponible para Java, C#, C++ y el motor de ejecución integrado.

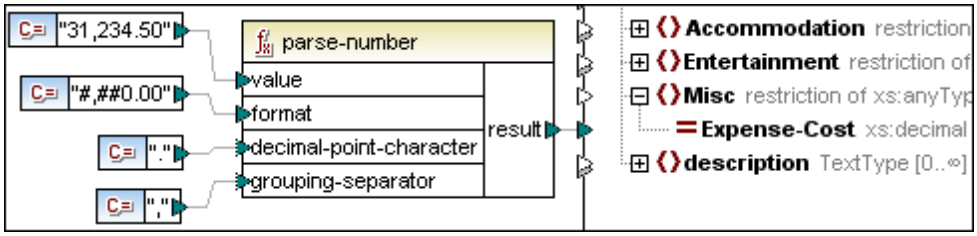
Convierte una cadena de entrada en un número decimal.

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)

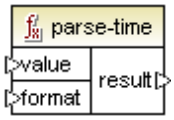
La cadena **format** utilizada en la función **parse-number** es la misma que se utiliza en la función **format-number**.

Ejemplo en MapForce:



Resultado:

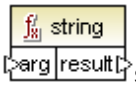
```
<expense-item xmlns="http://my-company.com/nam
C:/DOCUME~1/MYDOCU~1/Altova/MapForce2011
  <Misc MiscExpense-Cost="31234.5"/>
</expense-item>
```



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`.



string

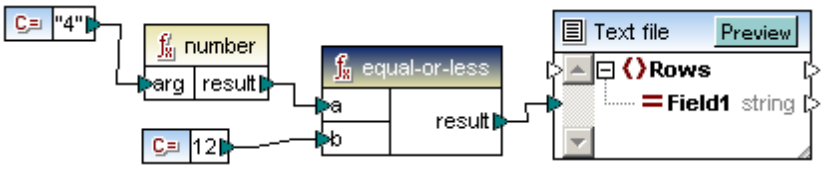
Convierte un valor de entrada en una cadena. La función también se puede usar para recuperar el contenido de texto de un nodo.

Si el nodo de entrada es un tipo complejo XML, todos los descendientes también se generan como una sola cadena.

Comparar tipos de nodos de entrada diferentes

Si los nodos de entrada son de tipos diferentes (p. ej. `integer` y `string`), puede usar las funciones de conversión para forzar una comparación de cadenas o numérica.

 © 2014 Altova GmbH



En el ejemplo anterior, la primera constante es de tipo `string` y contiene la cadena `4`.

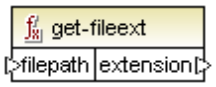
La segunda constante contiene el número 12. Para poder comparar los dos valores explícitamente, los tipos deben coincidir.

Si se añade una función **`number`** a la primera constante, la cadena es el valor numérico 4. El resultado de la comparación es "true".

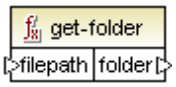
Recuerde que si la función **`number`** no se utiliza (si `4` no se conecta directamente al parámetro **`a`**), se realizaría una comparación de cadenas y el resultado sería "false".

file path (funciones de ruta de acceso)

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.



get-fileext
Devuelve la extensión de la ruta de archivo, incluido el carácter "."
P. ej. `c:\data\Sample.mfd` devuelve `".mfd"`

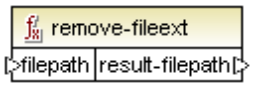


get-folder
Devuelve el nombre de la carpeta de la ruta de archivo, incluida la barra diagonal final o la barra diagonal inversa final.
P. ej. `c:/data/Sample.mfd` devuelve `c:/data/`

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.

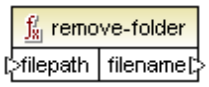
mfd-filepath
Si se le llama desde la asignación principal, la función devuelve el mismo resultado que la función **`main-mfd-filepath`**, es decir, la ruta completa del archivo `mfd` que contiene la asignación principal. Si el archivo `mfd` está sin guardar, la función devuelve una cadena vacía.

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.



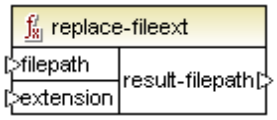
remove-fileext

Borra la extensión de la ruta de archivo, incluido el carácter "."
P. ej. `c:/data/Sample.mfd` devuelve `c:/data/Sample`.



remove-folder

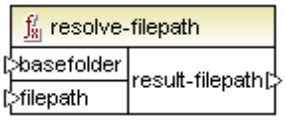
Borra el directorio de la ruta de acceso, incluida la barra diagonal final o la barra diagonal inversa final.
P. ej. `c:/data/Sample.mfd` devuelve `Sample.mfd`.



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`.

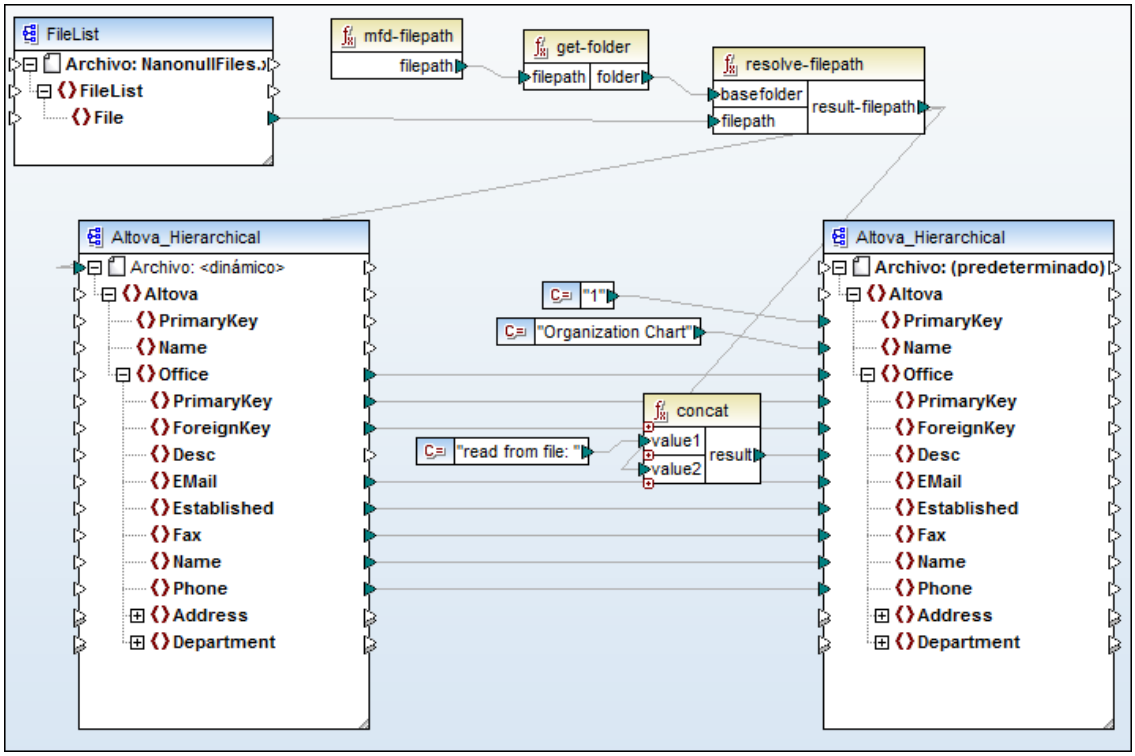


resolve-filepath

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`.

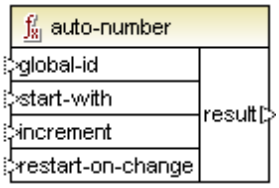
 © 2014 Altova GmbH



generator (funciones generador)

La función **`auto-number`** genera enteros en los nodos de destino de un componente, en base a los parámetros definidos.

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`**.



auto-number

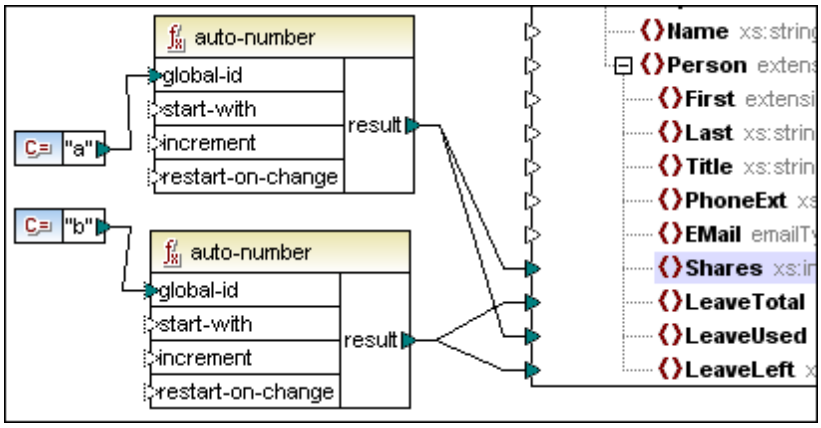
El resultado es un valor que empieza en el parámetro **`start_with`** y se incrementa con el valor de **`increment`**. Los valores predeterminados son **`start-with=1`** e **`increase=1`**. Ambos parámetros pueden ser negativos.

Parámetro `global-id`

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

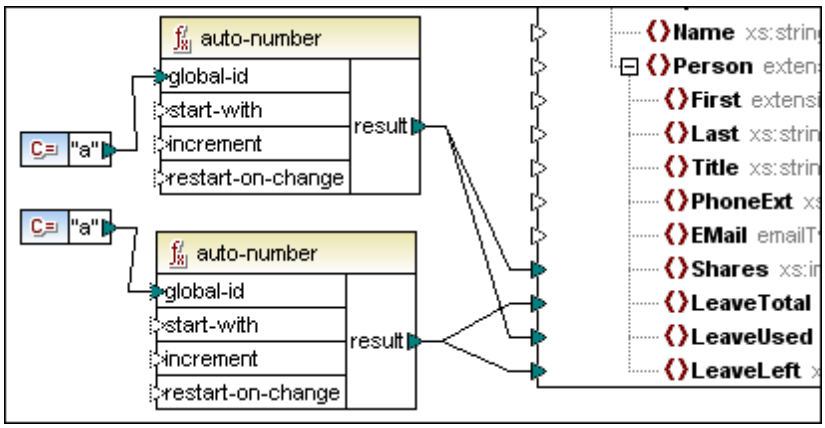
ejemplo siguiente, las funciones tienen parámetros **`global-id`** diferentes (**`a`** y **`b`**).

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.



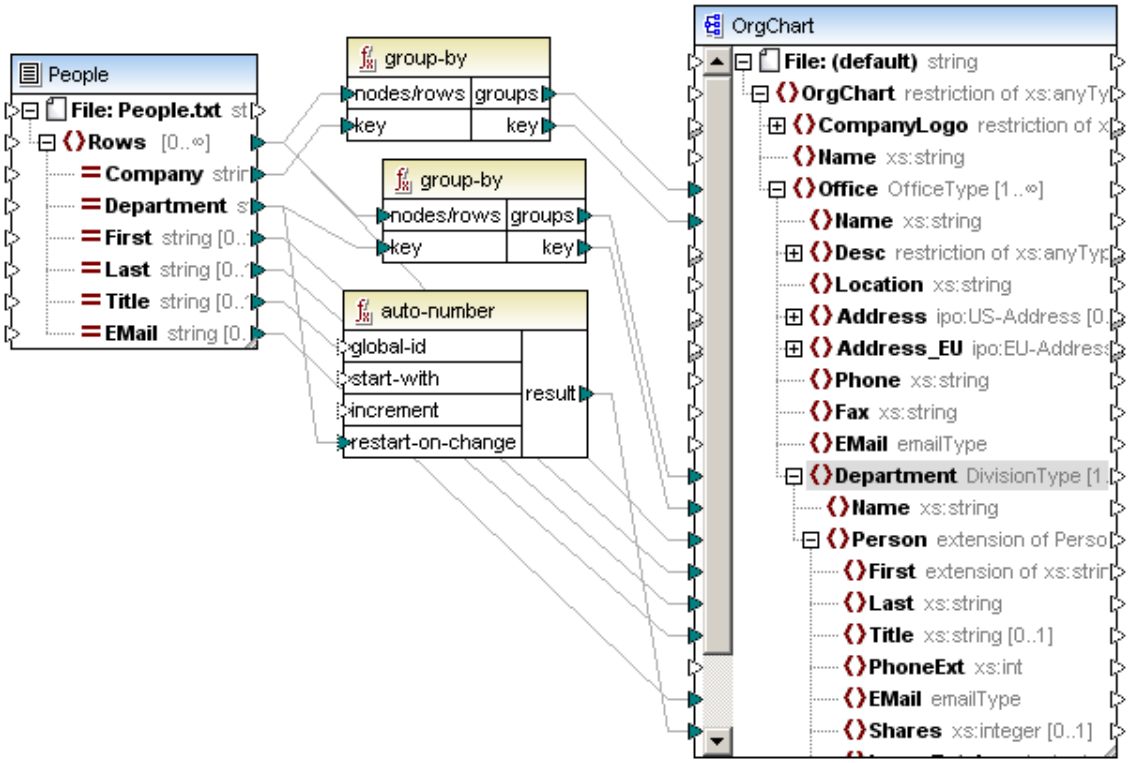
Parámetro `start-with`
El valor inicial utilizado para comenzar la secuencia de numeración automática. Valor predeterminado = 1.

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.



logical (funciones lógicas)

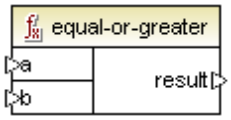
Las funciones lógicas de la biblioteca **core | logical** se usan por lo general para comparar datos de entrada y su resultado son los valores booleanos `true` o `false`. Estas funciones se suelen utilizar para probar datos antes de pasarlos, por medio de un filtro, a un subconjunto del componente de destino.

Cuando compara parámetros de entrada, MapForce selecciona el tipo común más específico para cada parámetro y después los compara. Si el tipo común es `anySimpleType`, ambos parámetros de entrada se comparan como datos `string`.

parámetros de entrada = **`a | b`** o **`value1 | value2`**
parámetro de salida = **`result`**

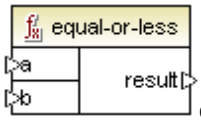
equal

El resultado es **`true`** si **`a=b`**. De lo contrario, el resultado es **`false`**.



equal-or-greater

El resultado es **`true`** si **`a`** es igual/mayor que **`b`**. De lo contrario, el resultado es **`false`**.



equal-or-less

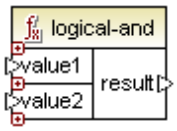
El resultado es **`true`** si **`a`** es igual o menor que **`b`**. De lo contrario, el resultado es **`false`**.

greater

El resultado es **`true`** si **`a`** es mayor que **`b`**. De lo contrario, el resultado es **`false`**.

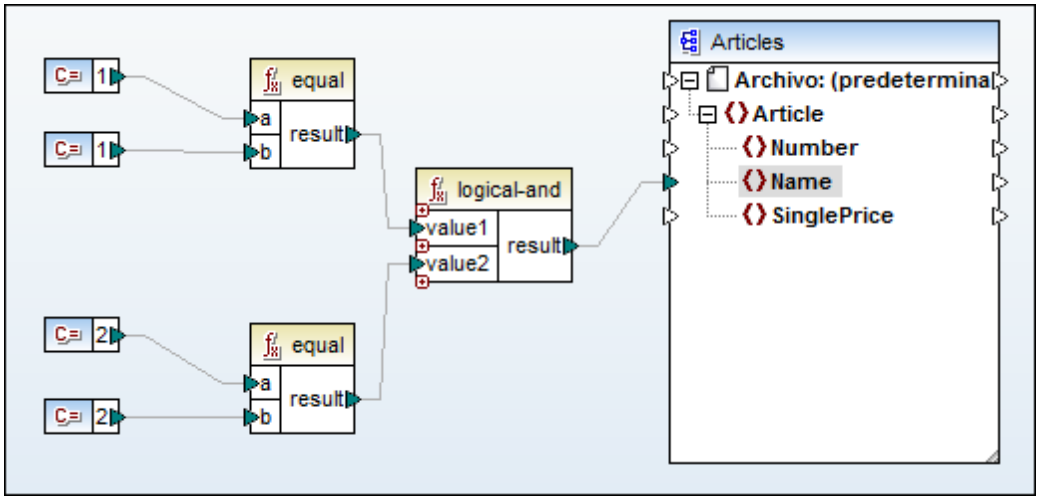
less

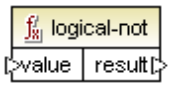
El resultado es **`true`** si **`a`** es menor que **`b`**. De lo contrario, el resultado es **`false`**.



logical-and

Si tanto **`value1`** como **`value2`** son **`true`**, el resultado es **`true`**. Si son diferentes, el resultado es **`false`**.

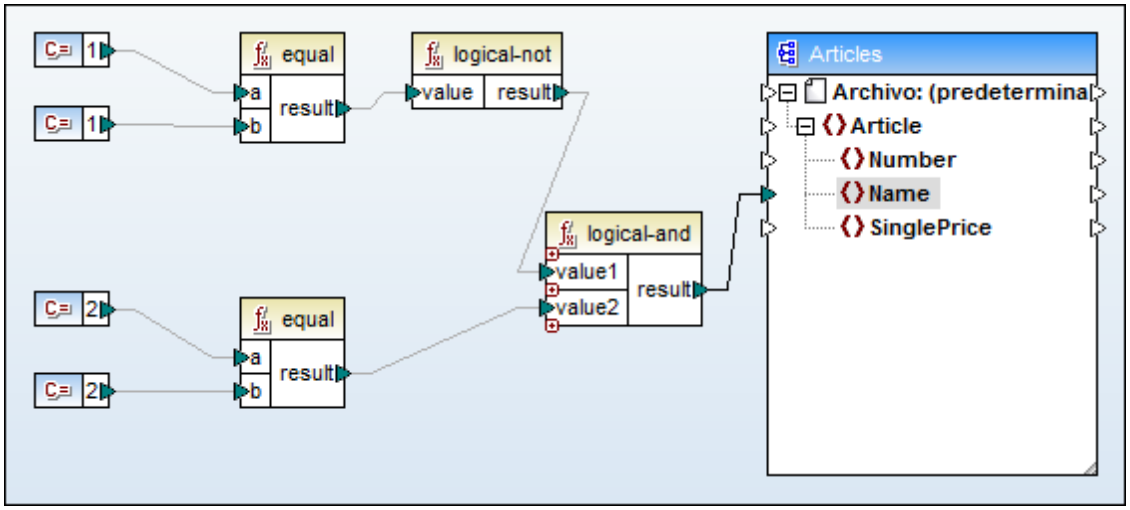


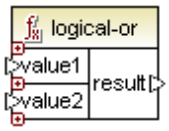


logical-not

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`**.

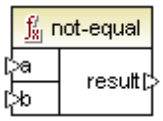
La función **`logical-not`** que aparece a continuación invierte el resultado de la función **`equal`**. La función **`logical-and`** ahora solamente devuelve el valor **`true`** si los valores booleanos de **`value1`** y **`value2`** son diferentes (es decir, **`true-false`** o **`false-true`**).





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`**.



not equal

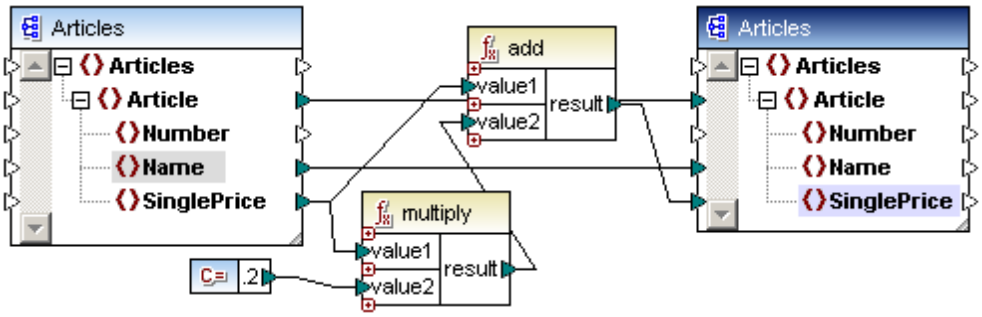
El resultado es **`true`** si **`a`** no es igual a **`b`**.

math (funciones matemáticas)

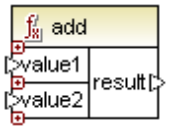
Las funciones matemáticas de la biblioteca **core | math** sirven para realizar operaciones matemáticas básicas con los datos. Tenga en cuenta que no se pueden usar para cálculos con datos `duration` ni `datetime`.

parámetros de entrada = **`value1 | value2`**
parámetro de salida = **`result`**

MapForce convierte automáticamente los valores de entrada en decimales para poder procesarlos.

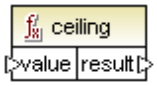


En la asignación de esta imagen, por ejemplo, se añade un 20% de impuestos sobre las ventas en los artículos asignados al componente de destino.



add

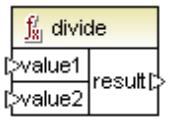
El resultado es el valor decimal resultante de sumar **`value1`** a **`value2`**.



ceiling

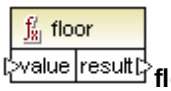
El resultado es el entero más pequeño que es mayor o igual que **`value`**, es decir, el valor entero mayor más próximo al valor decimal de entrada **`value`**.

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).



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.

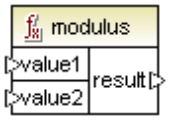


floor

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).

 © 2014 Altova GmbH

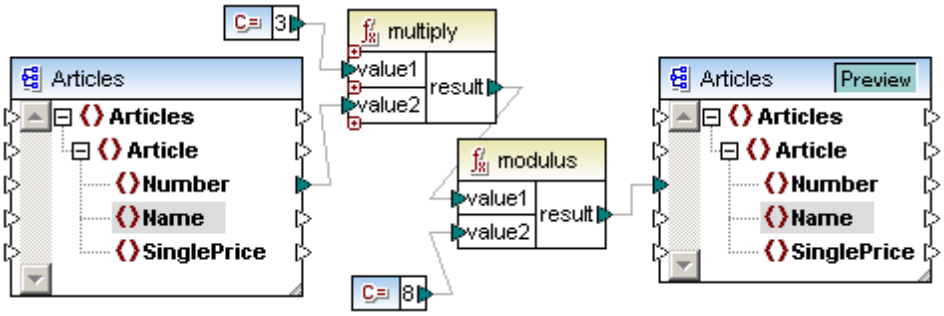


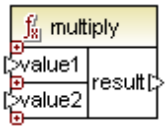
modulus

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.

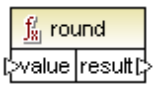
Cuando se aplica/utiliza la función **`modulus`** con **`value2`** = 8, los enteros restantes son 3, 6, 1 y 4.





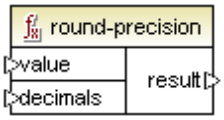
multiply

El resultado es el valor decimal resultante de multiplicar **`value1`** por **`value2`**.



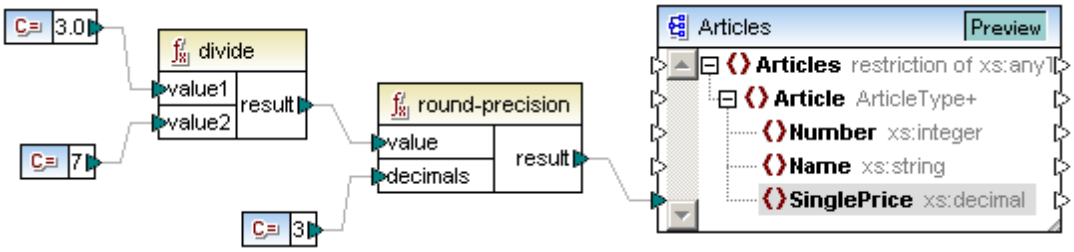
round

El resultado es el entero par más cercano al valor de **`value`**.

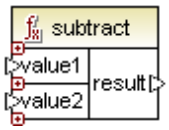


round-precision

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, comrpuebe que está asignado a un elemento de tipo `xs:decimal`.

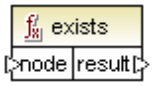


subtract

El resultado es el valor decimal resultante de restar **`value2`** a **`value1`**.

node (funciones de nodo)

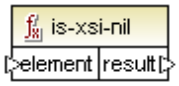
Las funciones de prueba de nodos de la biblioteca **core | node** sirven para comprobar si existen o no ciertos nodos en varios tipos de archivos de entrada (como XML, esquemas, texto, bases de datos, EDI, etc.) e incluso en los resultados de ciertas funciones. La función **`exists`** suele comprobar si hay una secuencia no vacía, es decir, si existe un nodo.



exists

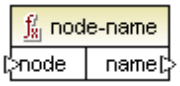
Devuelve el resultado **`true`** si el nodo existe. De lo contrario devuelve **`false`**.

Consulte este apartado para ver un ejemplo de uso.



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`.

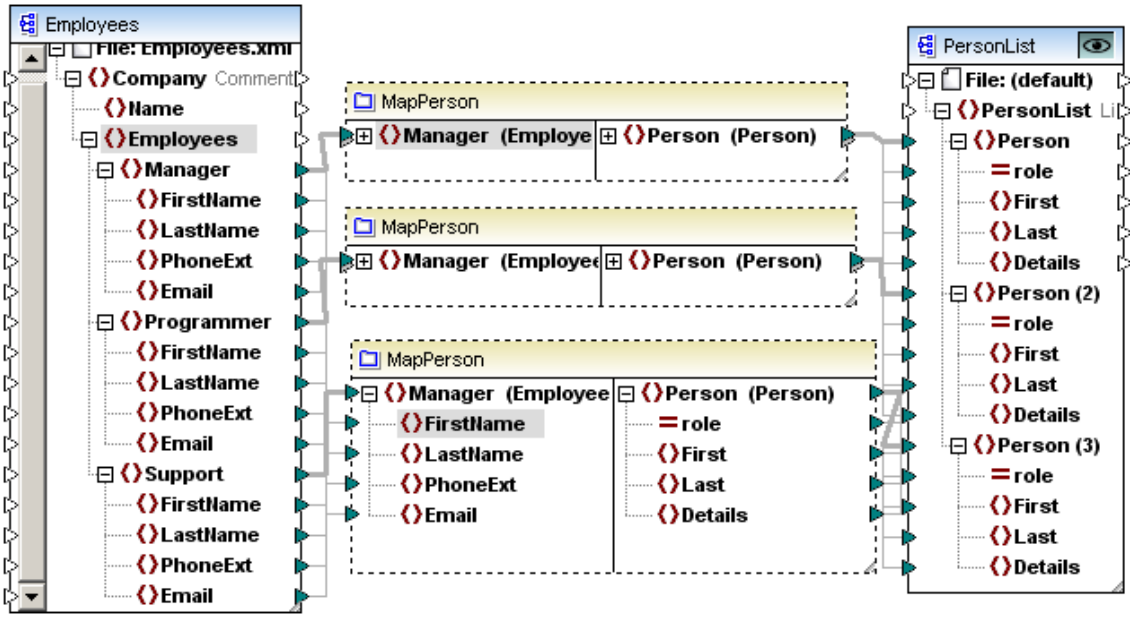


node-name

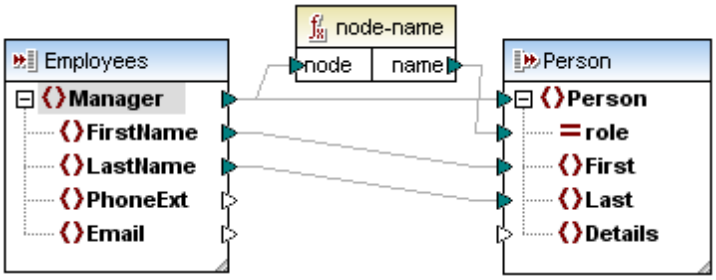
Devuelve el QName del nodo conectado, a no ser que sea un nodo XML `text()`. Si es así, devuelve un QName vacío. Esta función solamente funciona con los nodos que tienen nombre. Si XSLT es el lenguaje de destino (que llama a la función fn:node-name), devuelve una secuencia vacía para los nodos que no 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).

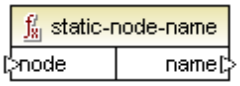
 © 2014 Altova GmbH



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`**.



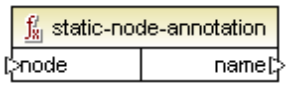
`Manager` recibe sus datos de **fuera** de la función definida por el usuario y su contenido puede ser `Manager`, `Programmer` o `Support`. Estos son los datos que se pasan al atributo `role` del componente de destino **PersonList**.



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.

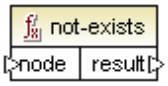
La conexión debe ser directa. No puede pasar por un filtro ni por una función definida por el usuario no inline. Se trata de una pseudo función, que en tiempo de generación se reemplaza con un texto adquirido del nodo conectado y, por tanto, está disponible en todos los lenguajes.



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.

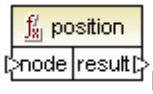
La conexión debe ser directa. No puede pasar por un filtro ni por una función definida por el usuario no inline. Se trata de una pseudo función, que en tiempo de generación se reemplaza con un texto adquirido del nodo conectado y, por tanto, está disponible en todos los lenguajes.



not-exists

Devuelve **`false`** si el nodo existe. De lo contrario, devuelve **`true`**.

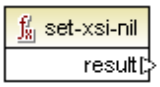
Consulte este apartado para ver un ejemplo de uso.



position

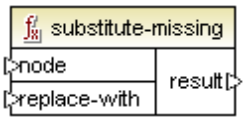
Devuelve la posición que tiene un nodo dentro de su secuencia.

Consulte este apartado para ver un ejemplo de uso.



set-xsi-nil

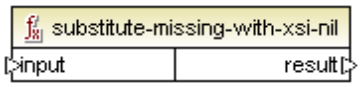
Define el nodo de destino como `xsi:nil`.



subsitute-missing

Esta función es una práctica mezcla de la función **`exists`** y la condición **`if-else`**. Utilice esta función para asignar el contenido del campo actual si el nodo existe en el archivo XML de origen. Si el nodo no existe, se utiliza el elemento asignado al parámetro **`replace-with`**.

Consulte este apartado para ver un ejemplo de uso.



substitute-missing-with-xsi-nil

Reemplaza los valores que faltan o los valores nulos del componente de origen con el atributo `xsi:nil` en el nodo de destino.

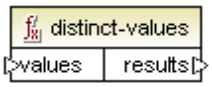
sequence (funciones de secuencia)

Las funciones de secuencia de la biblioteca **core | sequence** no están disponibles si el lenguaje seleccionado es XSLT 1.0.

MapForce ofrece funciones de secuencia 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.

 © 2014 Altova GmbH

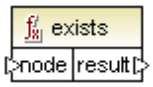
- El parámetro de entrada **key** tiene un tipo de datos arbitrario que se puede convertir en `string` en las funciones de secuencia **group-adjacent** y **group-by**.
- El parámetro de entrada **bool** tiene el tipo de datos **Boolean** en las funciones de secuencia **group-starting-with** y **group-ending-with**.
- El parámetro de salida **key** es la clave del grupo actual.



distinct-values

Permite quitar los valores duplicados de una secuencia y asignar los elementos únicos al componente de destino.

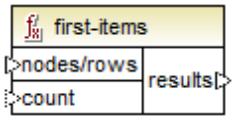
Consulte este apartado para ver un ejemplo de uso.



exists

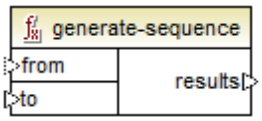
Devuelve **true** si el nodo existe. De lo contrario, devuelve **false**.

Consulte este apartado para ver un ejemplo de uso.



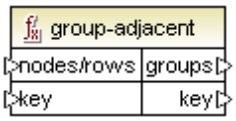
first-items

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.



generate-sequence

Crea una secuencia de enteros usando como límite los parámetros **from** y **to**.

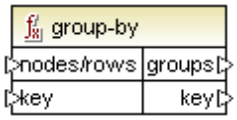


group-adjacent

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.

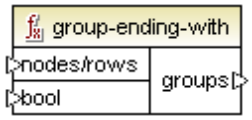
Consulte este apartado para ver un ejemplo de uso.



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.

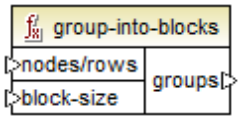
Consulte este apartado para ver un ejemplo de uso.



group-ending-with

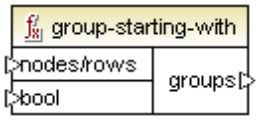
Agrupa la secuencia del parámetro de entrada **`nodes/rows`** en grupos, terminando con un grupo nuevo si el parámetro **`bool`** es `true`.

Consulte este apartado para ver un ejemplo de uso.



group-into-blocks

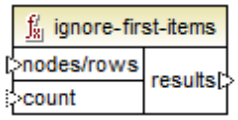
Agrupa la secuencia de entrada **`nodes/rows`** en bloques del mismo tamaño. El tamaño de los bloques viene dado por el parámetro **`block-size`**.



group-starting-with

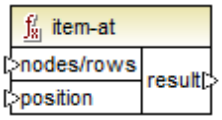
Agrupa la secuencia del parámetro de entrada **`nodes/rows`** en grupos, empezando con un grupo nuevo si el parámetro **`bool`** es **`true`**.

Consulte este apartado para ver un ejemplo de uso.



ignore-first-items

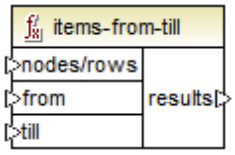
Pasa por alto los **`X`** primeros elementos de la secuencia de entrada, siendo **`X`** el número dado por el parámetro **`count`**. Por ejemplo, si se asigna el valor tres al parámetro **`count`** y un nodo primario al parámetro **`nodes/row`**, la función pasará por alto los tres primeros elementos del resultado.



item-at

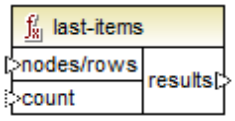
Devuelve los nodos de la secuencia **`nodes/rows`** situados en la posición dada por el parámetro **`position`**. El primer elemento está en la posición **`1`**.

 © 2014 Altova GmbH



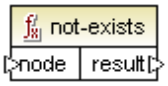
items-from-till

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`**.



last-items

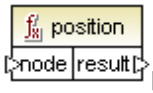
Devuelve los **`X`** últimos nodos de la secuencia **`nodes/rows`**, siendo X el número dado por el parámetro **`count`**.



not-exists

Si el nodo existe, la función devuelve **`false`**. De lo contrario, devuelve **`true`**.

Consulte este apartado para ver un ejemplo de uso.

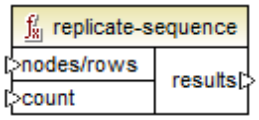


position

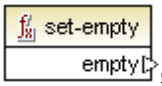
Devuelve la posición de un nodo dentro de la secuencia de la que forma parte.

Consulte este apartado para ver un ejemplo de uso.

replicate-sequence

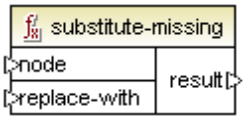


Copia **`X`** de la secuencia de entrada **`nodes/rows`** a la secuencia de salida, siendo **`X`** el número dado por el parámetro **`count`**.



set-empty

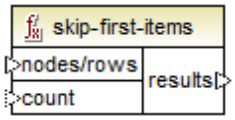
Sirve para cancelar los valores predeterminados de un documento XBRL definidos con anterioridad en el componente/la taxonomía XBRL.



subsitute-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`**.

Consulte este apartado para ver un ejemplo de uso.

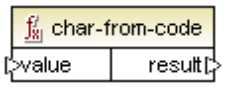


skip-first-items

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.

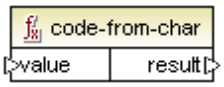
string (funciones de cadena)

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.



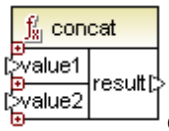
char-from-code

El resultado es el carácter que represente el valor Unicode decimal del parámetro **`value`**.



code-from-char

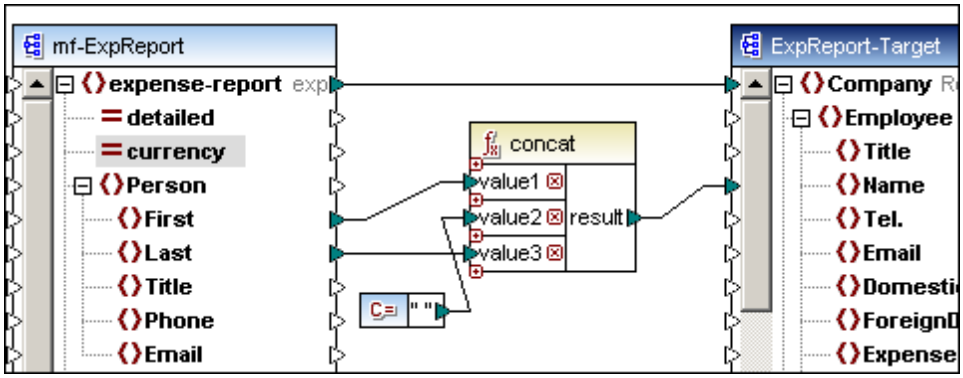
El resultado es el valor Unicode decimal del primer carácter del parámetro **`value`**.

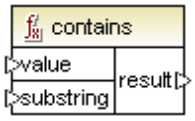


concat

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`.

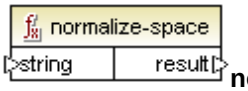
 © 2014 Altova GmbH





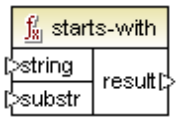
contains

El resultado es **`true`** si los datos asignados al parámetro de entrada **`value`** contienen la cadena asignada al parámetro de entrada **`substring`**.



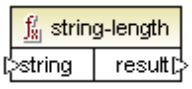
normalize-space

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).



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`**.



string-length

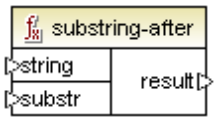
El resultado es el número de caracteres asignados al parámetro de entrada **`string`**.

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.

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.

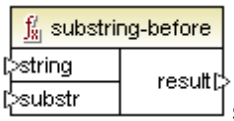
P. ej. la función **substring("56789",2,3)** da como resultado 678 (la cadena es 56789, la subcadena debe empezar en la posición 2 de la cadena y debe tener una longitud de 3 caracteres)



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.

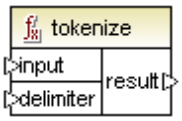
P. ej. la función **substring-after("2009/01/04","/")** da como resultado la subcadena 01/04.



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.

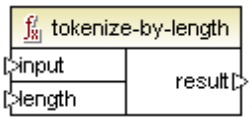


tokenize

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.

Para más información consulte el apartado Ejemplos de la función **tokenize**.



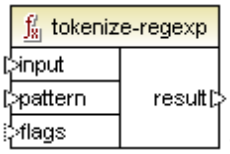
tokenize-by-length

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.

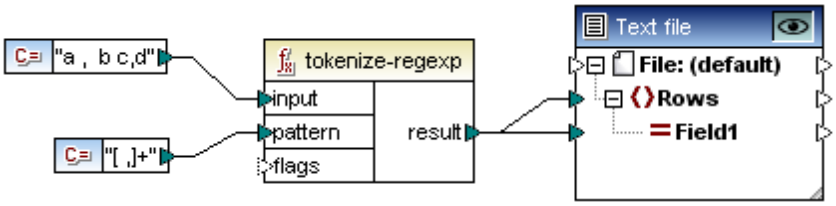
Para más información consulte el apartado Ejemplos de la función **tokenize**.

 © 2014 Altova GmbH



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 el espacio o la coma.

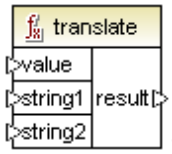
El cuantificador + especifica "una o más" apariciones de la clase de caracteres/cadena.

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).

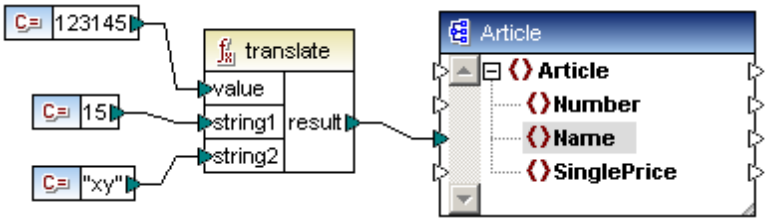
Consulte el apartado Expresiones regulares para más información.



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, se elimina el contenido de **`string1`**.



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)

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**"

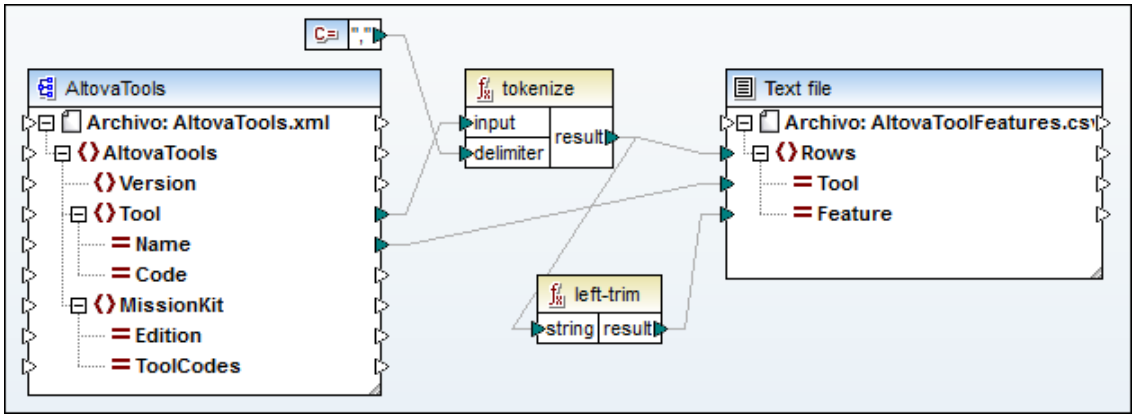
La cadena resultante es **ccbccabac**

Ejemplos de la función tokenize

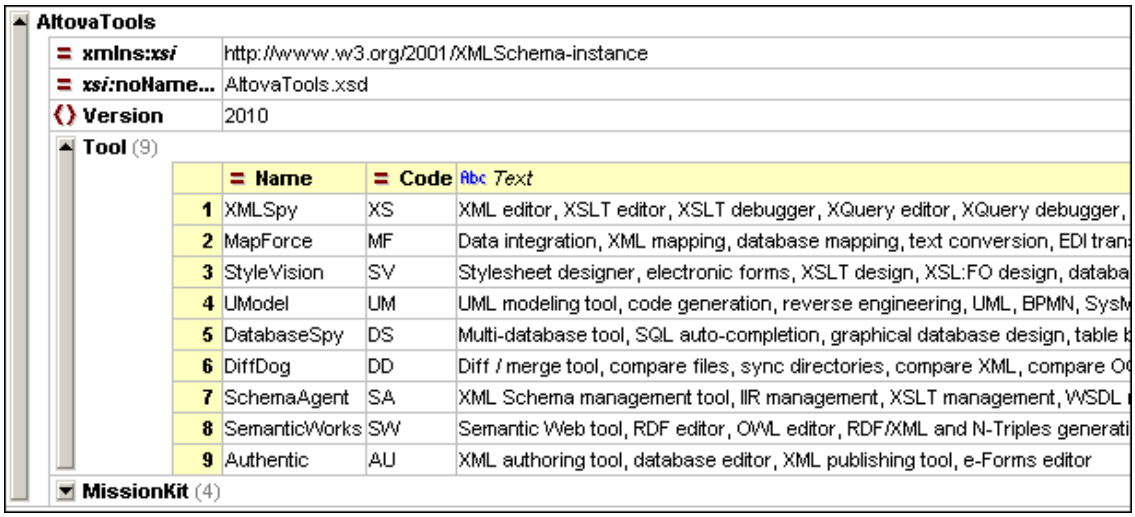
Ejemplo de uso de la función **tokenize**

El archivo `tokenizeString1.mfd` disponible en la carpeta `...\MapForceExamples` explica el uso de la función **tokenize**.

 © 2014 Altova GmbH



El archivo XML de origen aparece en la imagen siguiente. El elemento **`Tool`** tiene dos atributos (**`Name`** y **`Code`**) y su contenido 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:

- La función **`tokenize`** recibe datos del elemento **`Tool`** y utiliza el delimitador "**,**" para dividir los datos en trozos. El primer trozo es `XML editor`, por ejemplo.
- El parámetro **`result`** está asignado al elemento **`Filas`** del componente de destino. Por tanto, se genera una fila por cada trozo.
- El parámetro **`result`** también está asignado a la función **`left-trim`**, que elimina el espacio en blanco inicial de cada trozo.
- El parámetro **`result`** de la función **`left-trim`** está asignado al elemento **`Feature`** del componente de destino.
- El archivo de salida del componente de destino se definió como archivo CSV (`AltovaToolFeatures.csv`) y su delimitador de campo es un punto y coma (haga doble clic en el componente para ver su configuración).

Resultado de esta asignación

- El atributo **`Name`** de cada elemento **`Tool`** del archivo de origen se asigna al elemento **`Tool`** del componente de destino
- Cada trozo del contenido acortado (por medio de la función **`tokenize`**) de **`Tool`** se anexa al elemento **`Feature`** del componente de destino
- Por ejemplo, el primer **`Tool`** (XMLSpy) recibe el primer trozo: `XML editor`

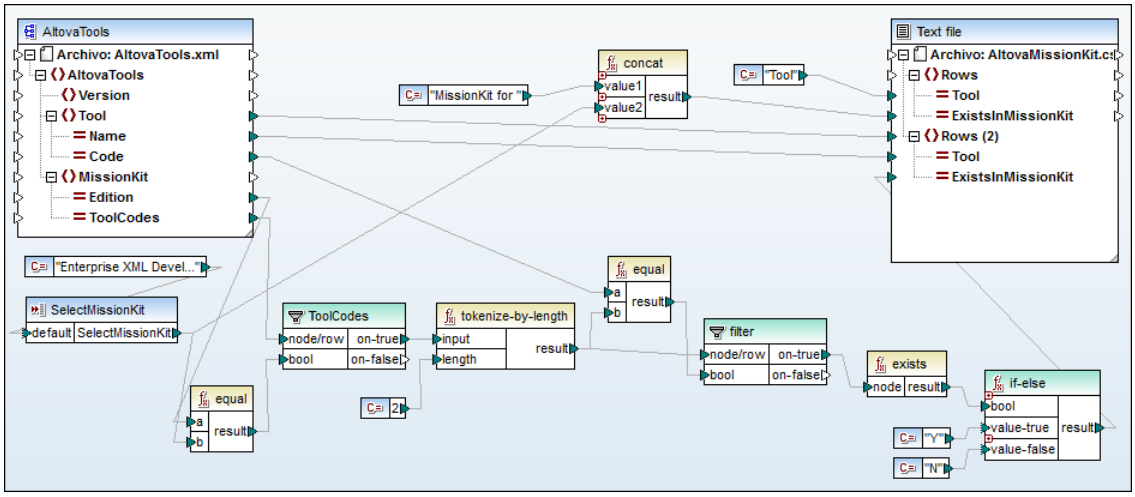
- Esto se repite con todos los trozos del elemento **`Tool`** actual y después con todos los demás elementos **`Tool`**.
- Abra el panel **Resultados** para ver el resultado de la asignación.

```
 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
```


© 2014 Altova GmbH

Ejemplo de la función **`tokenize-by-length`**

El archivo `tokenizeString2.mfd` disponible en la carpeta `...\MapForceExamples` explica el uso de la función **`tokenize-by-length`**.



El archivo XML de origen aparece en la siguiente imagen y es el mismo que el del primer ejemplo.
El elemento MissionKit también tiene dos atributos: **`Edition`** y **`ToolCodes`**, pero tiene contenido.

Tool (9)

	Name	Code	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 con todas las herramientas de Altova que forman parte de las diferentes ediciones de Altova MissionKit.

Características de la asignación:

- El componente de **entrada** **`SelectMissionKit`** recibe sus datos de entrada predeterminados de una constante, la 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** recibe datos del 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**.

Cómo funciona la función tokenize-by-length:

- El valor de **entrada** de **ToolCodes** XSMFSVDDSASW se divide en varios trozos, cada uno de dos caracteres (el parámetro **length** determina una longitud de 2 caracteres) y se obtiene un total de 6 trozos.
- Los trozos se pasan al parámetro **b** de la función **equal**. Cada trozo se compara con el valor de dos caracteres del atributo **Code** del archivo de origen (que tiene un total de 9 elementos con atributo **Code**)
- El resultado de la comparación (**true/false**) se pasa al parámetro **bool** del siguiente filtro.
- **Todos** los trozos resultantes de la función **tokenize-by-length** se pasan al parámetro **node/row** del filtro.
- La función **exists** ahora busca nodos existentes/no existentes en los datos recibidos del parámetro **on-true** del componente de filtrado.

  Los nodos existentes son los nodos cuyo valor **Code coincide** con el trozo de **ToolCodes**.

  Los nodos no existentes son los nodos cuyo valor **Code no coincide** con el trozo de **ToolCodes**.

- Los resultados booleanos de la función **exists** (**true/false**) se pasan a la condición **if-else**, que a su vez pasa **Y** al destino si existe el nodo y **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
```

Expresiones regulares

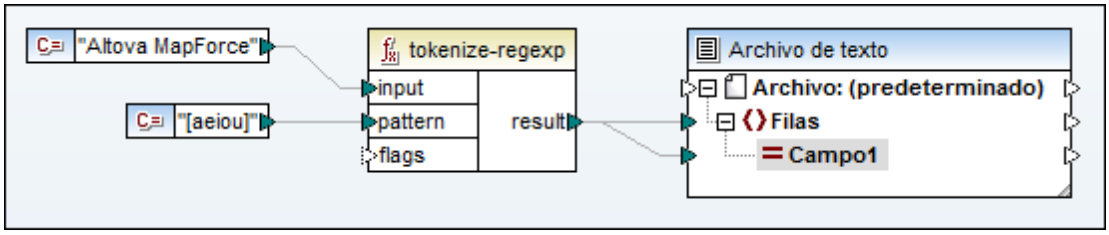
MapForce admite el uso de expresiones regulares en el parámetro **pattern** de las funciones **match-pattern** y **tokenize-regexp** para buscar cadenas concretas en el parámetro de entrada.

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:

 © 2014 Altova GmbH

parámetro **input** : la cadena en la que trabaja la expresión regular
parámetro **pattern** : la expresión regular
parámetro **flags** : parámetro opcional para definir cómo se debe interpretar la expresión regular
parámetro **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 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`**.

 © 2014 Altova GmbH

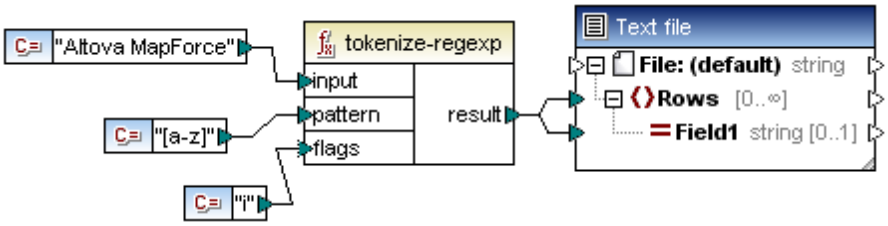
m
Si está presente, el proceso de búsqueda opera en el modo multilínea.

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).



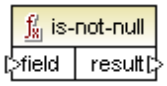
x
Si está presente, los caracteres de espacio en blanco se quitan de la expresión regular antes de iniciar el proceso de búsqueda. Los caracteres de espacio en blanco son `#x09`, `#x0A`, `#x0D` y `#x20`.

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.

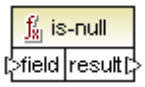
15.5.2 Biblioteca db

La biblioteca **db** contiene funciones que sirven para definir los resultados de la asignación cuando se encuentran campos NULL en bases de datos.



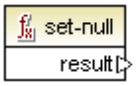
is-not-null

Si el campo es NULL, la función devuelve **`false`**. De lo contrario devuelve **`true`**.



is-null

Si el campo es NULL, la función devuelve **`true`**. De lo contrario devuelve **`false`**.

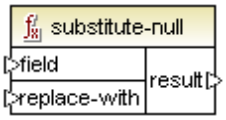


set-null

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.
- Si conecta **`set-null`** a un elemento **`simpleType`**, el elemento no se crea en el componente de destino.
- No está permitido conectar **`set-null`** a un elemento **`complexType`**, a una tabla ni a una fila. Cuando esto ocurre aparece un error de validación.

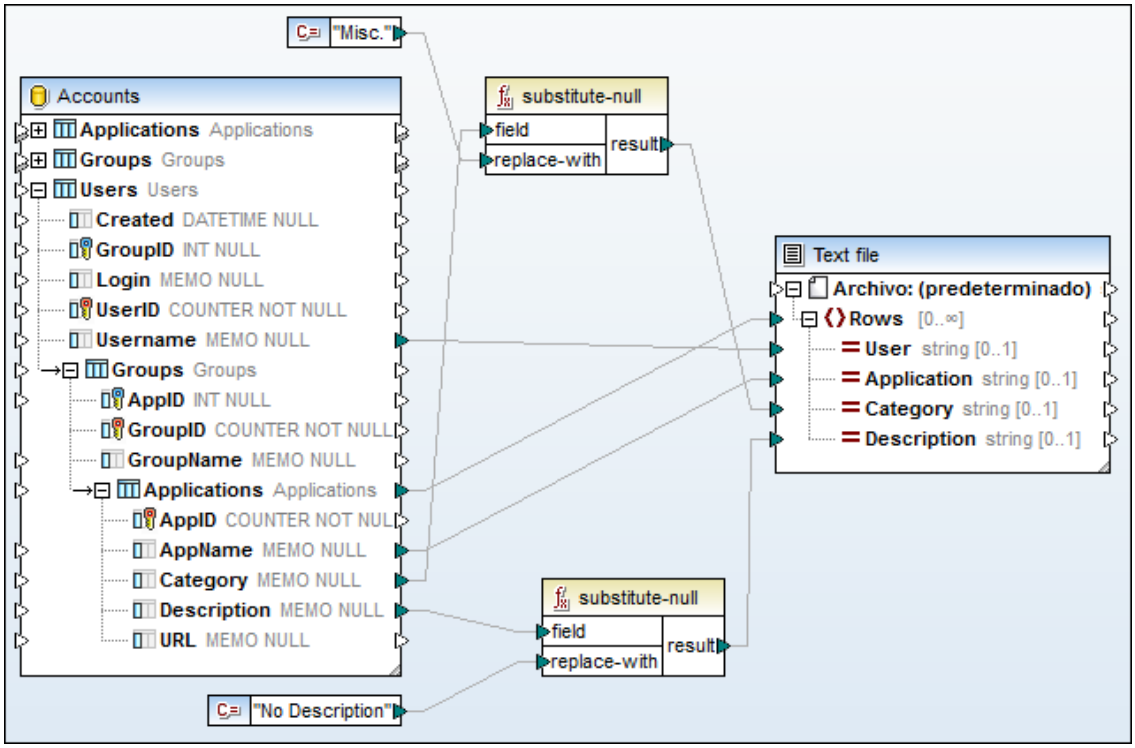


substitute-null

Se utiliza 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`.

 © 2014 Altova GmbH

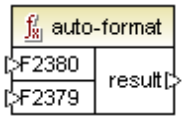


La primera función **`substitute-null`** (la que aparece en la parte superior) comprueba si existe la entrada **`Category`** en la tabla **`Applications`**. Puesto que **`Category`** no existe en la aplicación **`Notepad`**, se asigna la cadena de texto `Misc.` al elemento **`Category`** del archivo 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`.

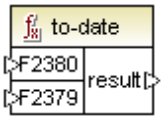
15.5.3 Biblioteca edifact

La biblioteca **edifact** ofrece funciones para convertir formatos `date`, `time` y `period` de EDI en datos `xs:date`.



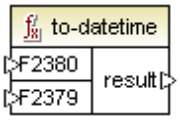
auto-format

El resultado es el valor `date`, `time` o `datetime` extraído de la fuente codificada (F2380) dado el código de formato (F2379).



to-date

El resultado es el valor `date` extraído de la fuente codificada (F2380) dado el código de formato (F2379).



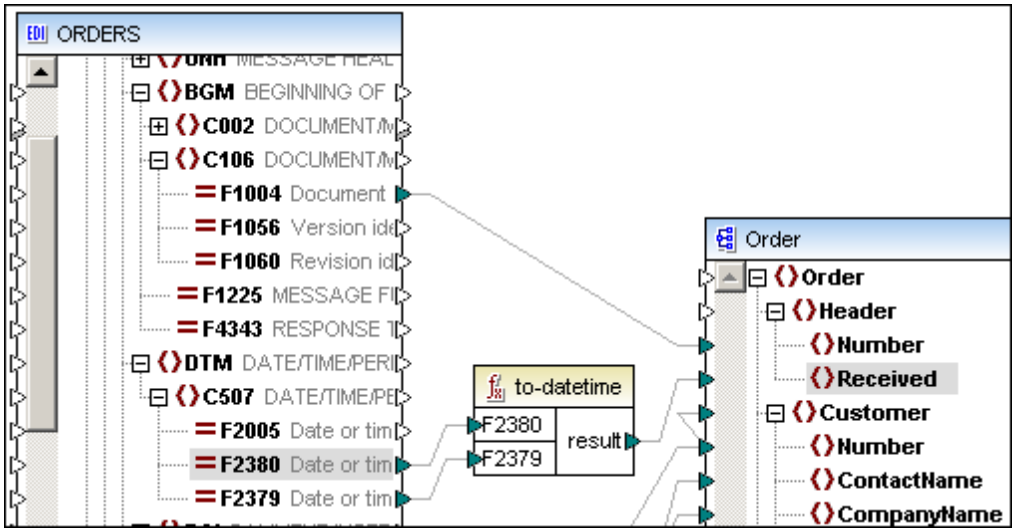
to-datetime

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`.

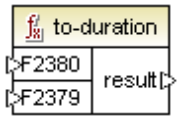
```
UNB+UNOB:1+003897733:01:MFGB+PARTNER ID:ZZ:ROUTING
ADDR+970101:1230+00000000000001++ORDERS++++1'
UNH+0001+ORDERS:D:08A:UN'
BGM+221+ABC123456XYZ+9'
DTM+4:20040430174 2PDT:303'
FTX+PUR+3++Pizza purchase order'
RFF+CT:123-456'
RFF+CR:1122'
NAD+SE+999::92++24h Pizza+Long Way+San-Francisco+C
CTA+SR+:Ted Little'
```

Este valor se convierte en `xs:date` usando la función **`to-datetime`**.



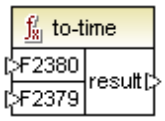
El resultado aparece a continuación. Este ejemplo está en el archivo `EDI-Order.mfd`, disponible en la carpeta `...\MapForceExamples`.

```
<?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>
```



to-duration

El resultado es el valor `Duration` extraído de la fuente codificada (F2380) dado el código de formato (F2379).



to-time

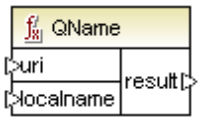
El resultado es el valor `Time` extraído de la fuente codificada (F2380) dado el código de formato (F2379).

15.5.4 Biblioteca lang

La biblioteca **lang** contiene funciones para trabajar con los lenguajes Java, C# o C++.

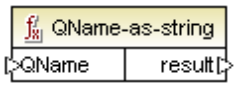
- funciones QName
- funciones de fecha y hora
- funciones generador
- funciones lógicas
- funciones matemáticas
- funciones de cadena

funciones QName



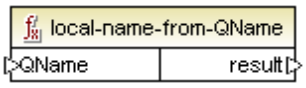
QName

El resultado es un QName construido a partir del URI de espacio de nombres y el nombre local.



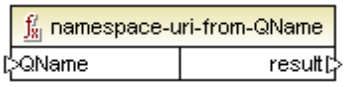
QName-as-string

El resultado es la representación de cadena única del QName.



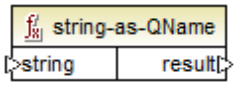
local-name-from-QName

El resultado es la parte de nombre local del QName.



namespace-uri-from-QName

El resultado es la parte de URI de espacio de nombres del QName.

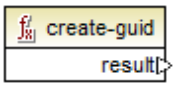


String-as-QName

Convierte la representación en forma de cadena de un QName en un QName.

generator (funciones generador)

Las funciones generador generan valores para campos de BD que no reciben datos de entrada del esquema, BD o EDI de origen.

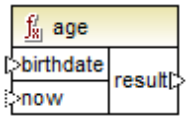


create-guid

El resultado es un identificador único global (como cadena con codificación hexadecimal) para un campo concreto.

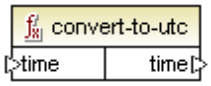
 © 2014 Altova GmbH

datetime (funciones de fecha y hora)



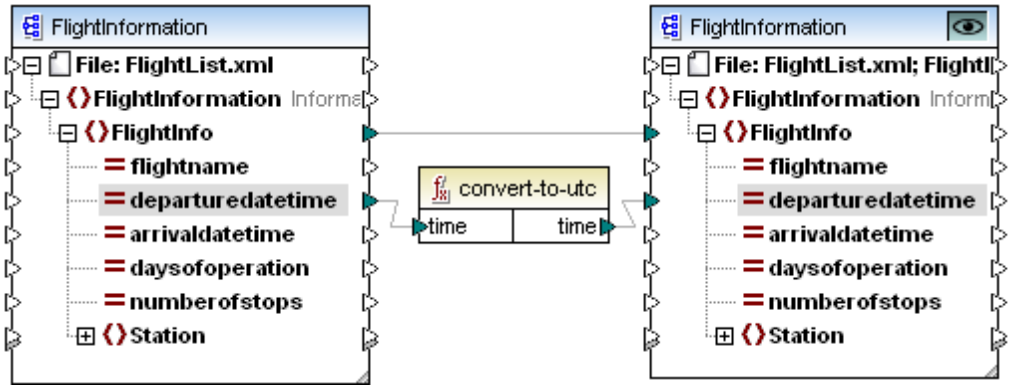
age

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.



convert-to-utc

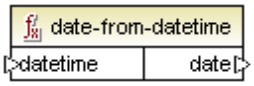
Convierte el parámetro de entrada local **`time`** en la hora universal coordinada o GMT. (La función tiene en cuenta el componente de uso horario, p. ej. `+5:00`).



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".`

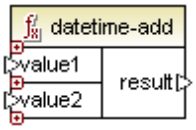


date-from-datetime

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"`



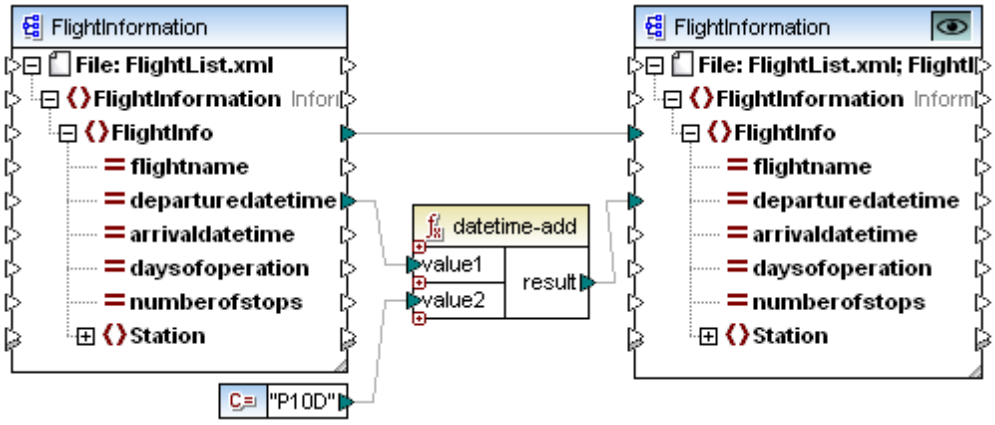
datetime-add

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).



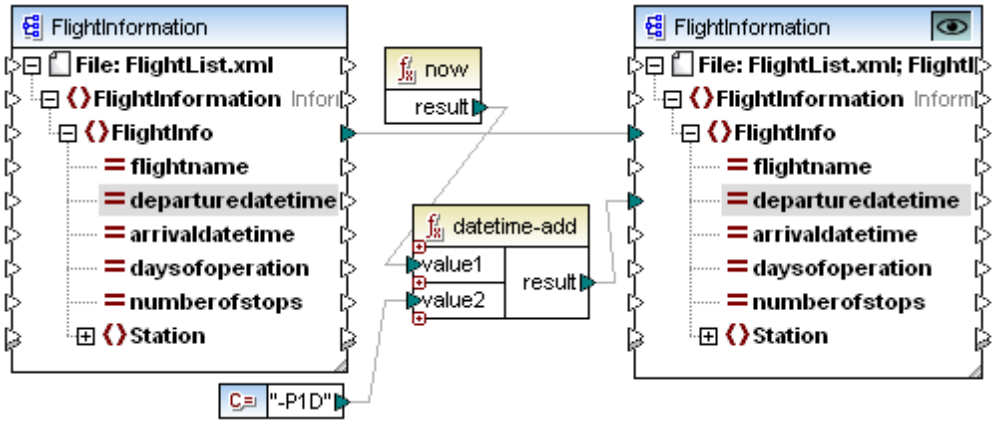
Por ejemplo, valor `datetime` del documento de instancia:
`departuredatetime="2001-12-17T09:30:02+05:00"`

Resultado:
`departuredatetime="2001-12-27T09:30:02+05:00"`

Para extraer la fecha de ayer de la entrada dateTime:

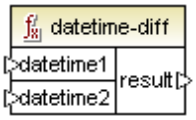
Use la función **now** para introducir la fecha/hora actual incluido el uso horario. El período se puede convertir en negativo usando el carácter - antes del designador P (por ejemplo -P1D, menos 1 día).

 © 2014 Altova GmbH



Por ejemplo, el valor `datetime` actual es 28 de febrero 2012, 17:19:54.748(miliseg)+01uso horario.
`now="2012-02-28T17:19:54.748+01:00"`

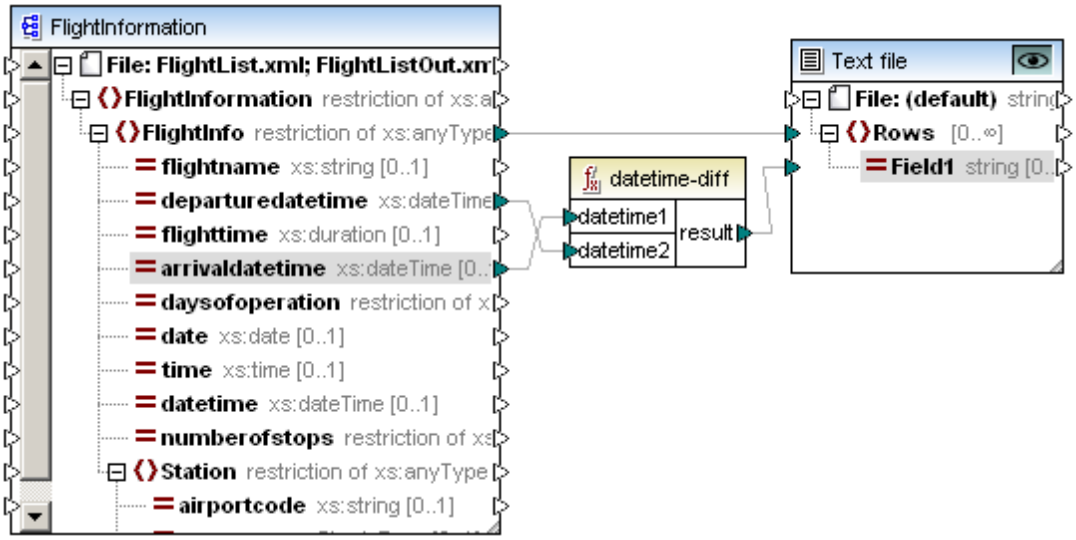
Resultado:
`departuredatetime="2012-02-27T17:19:54.748+01:00"`
es decir, 27 de febrero 2012, 17:19:54.748(miliseg)+01uso horario



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`**.

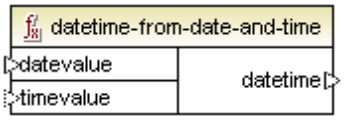


Por ejemplo, si queremos encontrar la diferencia en forma de duración entre las horas de salida (`departure`) y de llegada (`arrival`).

```
datetime1 arrivaldatetime="2001-12-17T19:30:02+05:00"
datetime2 departuredatetime="2001-12-17T09:30:02+05:00"
```

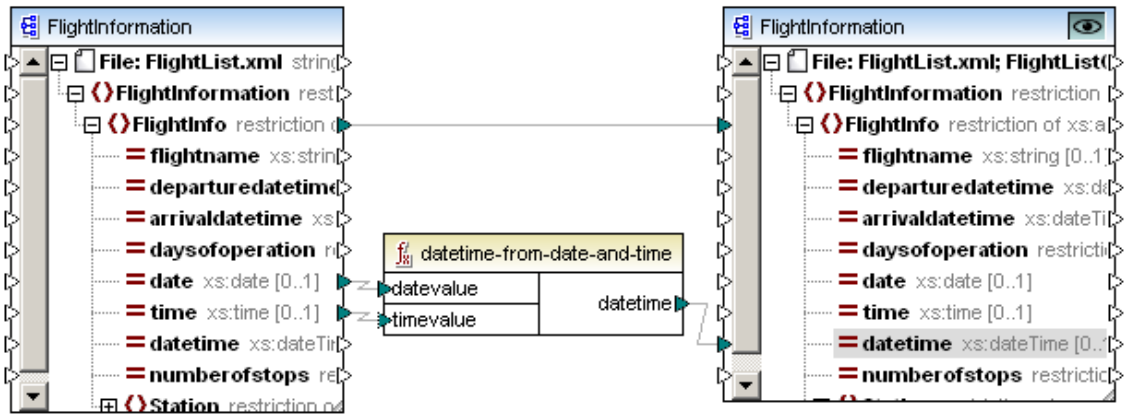
Resultado: la diferencia entre las dos es 10 horas.

```
result= PT10H
```



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`.

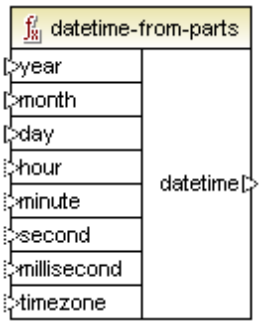


Ejemplo:

```
date="2012-06-29"
time="11:59:55"
```

Resultado:

```
dateTime="2012-06-29T11:59:55"
```



datetime-from-parts

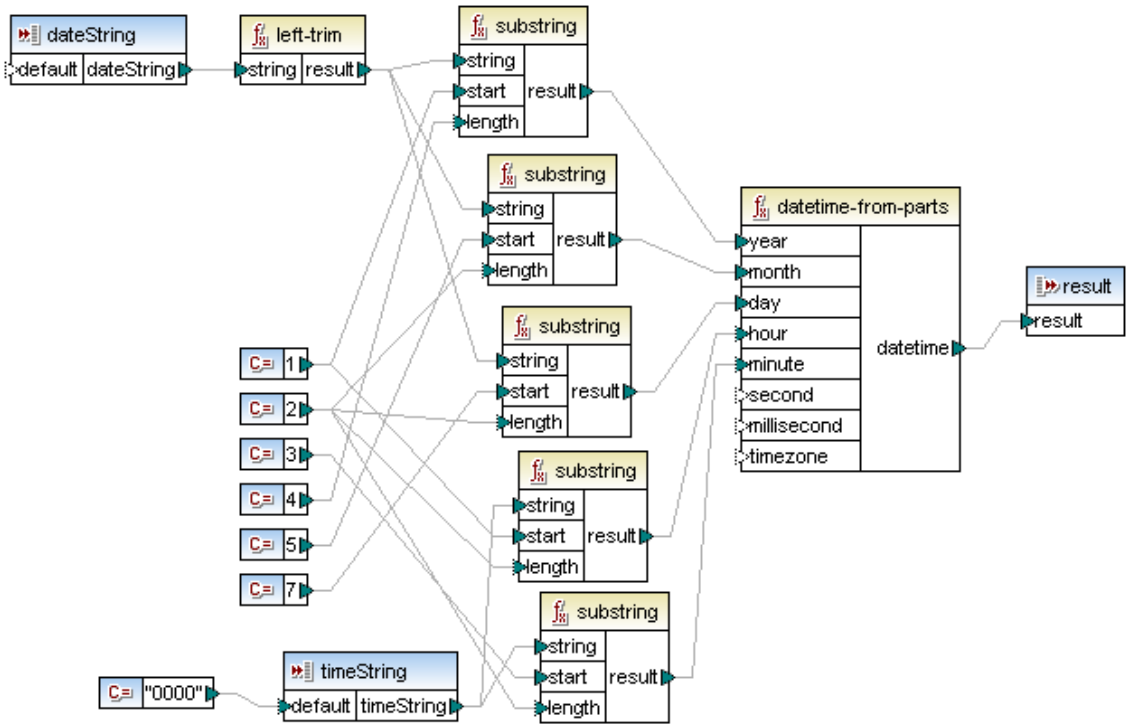
El resultado es un valor `datetime` creado uniendo cualquiera de estos argumentos: **`year, month, day, hour, minute, second, millisecond`** y **`timezone`**. Esta función normaliza automáticamente los parámetros suministrados. Por ejemplo, 32 de enero se convierte automáticamente en 1 de febrero.

Todos los argumentos son de tipo `xs:int`, excepto el argumento **`millisecond`**, que es de tipo

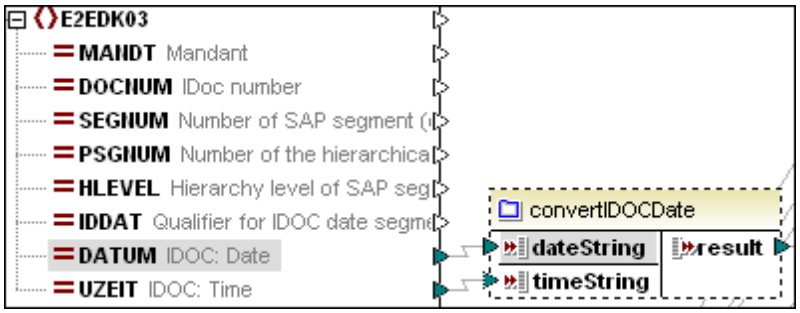
`xs:decimal`. El parámetro **`result`** es un valor de tipo `xs:dateTime`.

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`**.

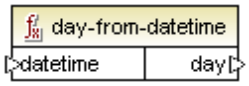
Observe que **`year, month`** y **`day`** son parámetros obligatorios y el resto son opcionales.



Los campos de fecha y hora vienen dados por el archivo de instancia IDOC:

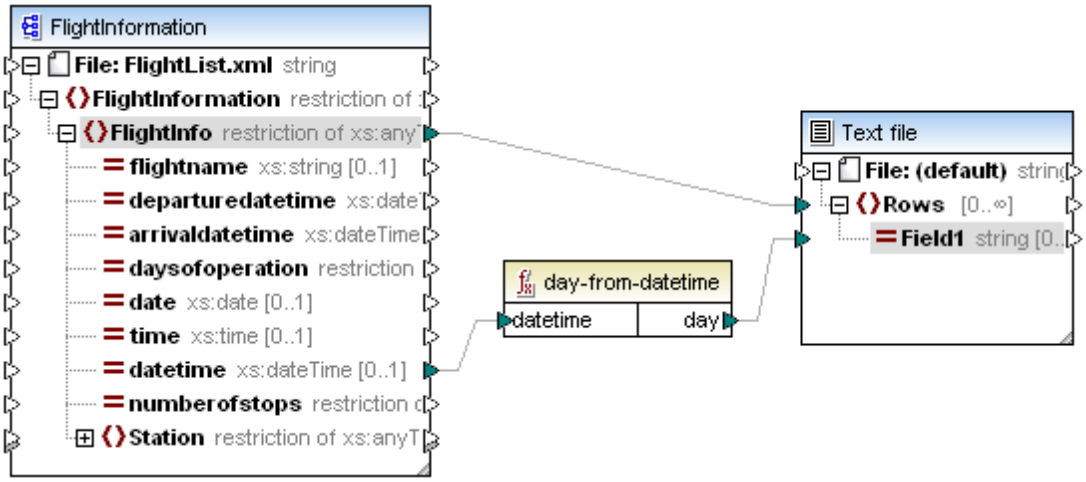



```
IDOC:Date    19990621
ICOC:Time    0930
Resultado       1999-06-21T09:30:00
```



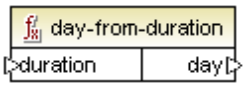
day-from-datetime

El resultado es el día del argumento `datetime`.



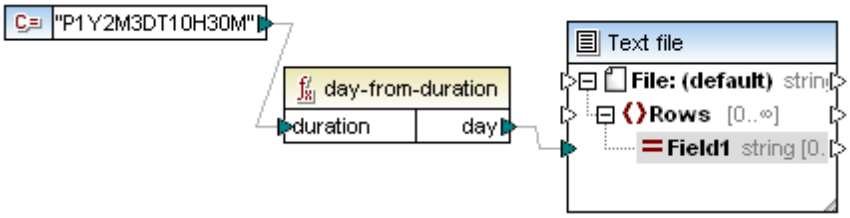
Ejemplo:
`datetime="2001-12-17T10:30:03+01:00"`

Resultado: `17`

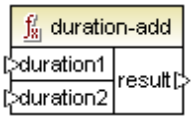


day-from-duration

El resultado es el día del argumento `duration`.



Ejemplo:
`duration="P1Y2M3DT10H30M"`
Resultado: `3`



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)`

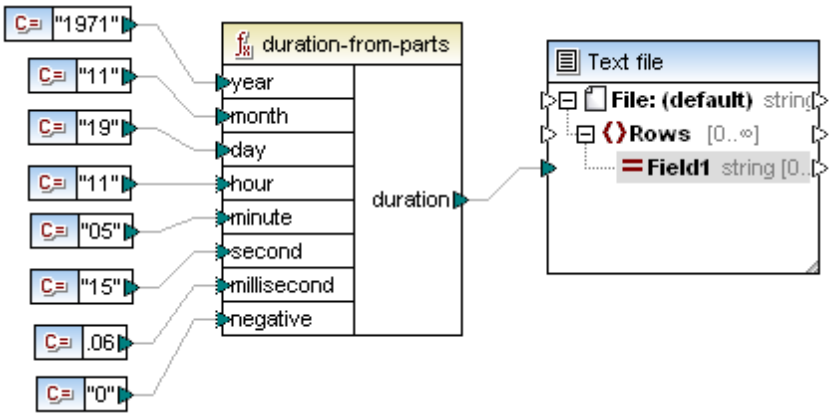
 © 2014 Altova GmbH

duration-from-parts

El resultado es una duración calculada al combinar estos argumentos: **`year, month, day, hour, minute, second, millisecond, negative.`**

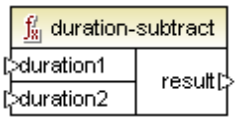
Las duraciones tienen este formato `P1Y2M3DT04H05M06.07S` es decir, P de período, 1 año (Y) 2 meses (M) 3 días (D) T de designador de hora, 04 horas (H), 05 minutos (M) 06.07 segundos.milisegundos.

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"`



duration-subtract

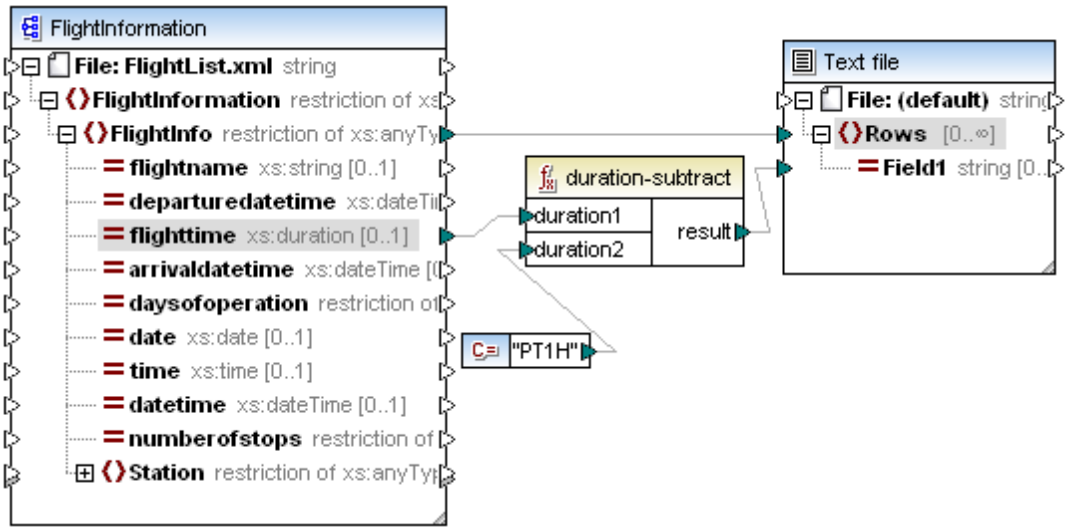
El resultado es la duración que se obtiene al sustraer **`duration2`** a **`duration1`**.

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:

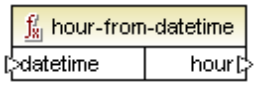
1 año (Y), 2 meses (M), 3 días (D), designador de hora T, 04 horas (H), 05 minutos (M).

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"`

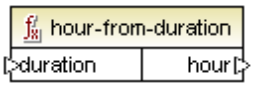
Resultado:
`PT4H7M`



hour-from-datetime

El resultado es el componente **`hour`** del argumento **`datetime`**.

Ejemplo:
`datetime="2001-12-17T09:30:02+05:00"`
`hour= 9`

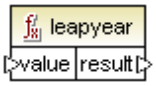


hour-from-duration

El resultado es el componente **`hour`** del argumento **`duration`**.

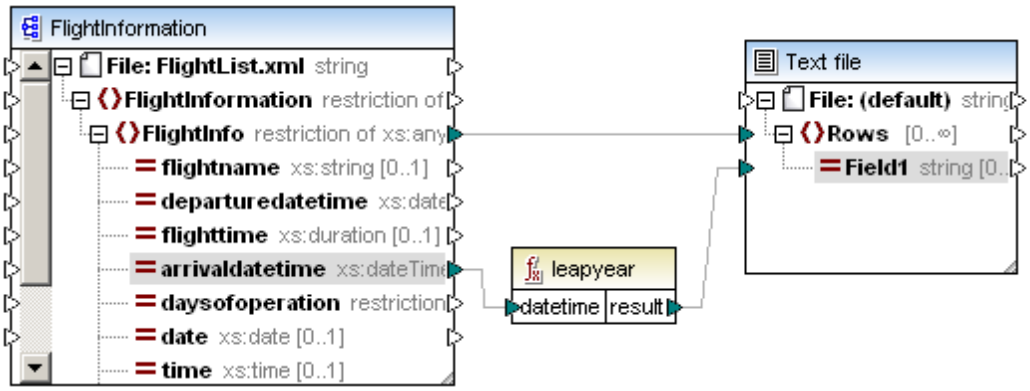
Ejemplo:
`duration="P0Y0M0DT05H07M"`
`hour= 5`

 © 2014 Altova GmbH



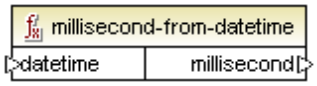
leapyear

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"
```

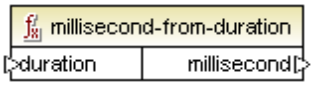


millisecond-from-datetime

El resultado es el componente **`millisecond`** del argumento **`datetime`**.

Ejemplo:

```
datetime="2001-12-17T09:30:02.544+05:00"
millisecond= 544
```

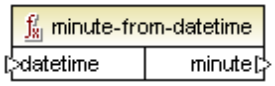


millisecond-from-duration

El resultado es el componente **`millisecond`** del argumento **`duration`**.

Ejemplo:

```
duration="P0Y0M0DT05H07M02.227S"
millisecond= 227
```

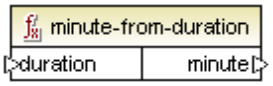


minute-from-datetime

El resultado es el componente **`minute`** del argumento **`datetime`**.

Ejemplo:

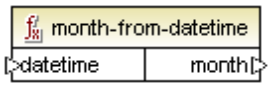
```
datetime="2001-12-17T09:30:02.544+05:00"
minute= 30
```



minute-from-duration

El resultado es el componente **minute** del argumento **duration**.

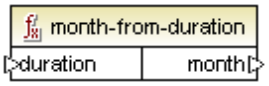
Ejemplo:
duration="P0Y0M0DT05H**07**M02.227S"
minute= 7



month-from-datetime

El resultado es el componente **month** del argumento **dateTime**.

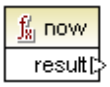
Ejemplo:
datetime="2001-**12**-17T09:30:02.544+05:00"
month= 12



month-from-duration

El resultado es el componente **month** del argumento **duration**.

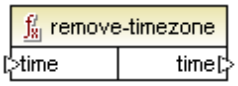
Ejemplo:
duration="P0Y**04**M0DT05H07M02.227S"
month= 4



now

El resultado es el valor de fecha y hora actual, incluido el uso horario.

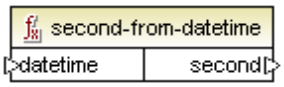
Ejemplo:
result= 2012-03-06T14:44:57.567+01:00



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

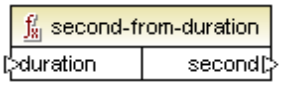


second-from-datetime

El resultado es el componente **second** del argumento **dateTime**.

 © 2014 Altova GmbH

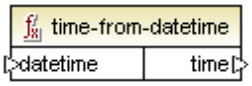
Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
second= 2
```



second-from-duration

El resultado es el componente **second** del argumento **duration**.

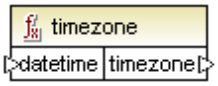
Ejemplo:
```
duration="P0Y04M0DT05H07M02.227S"
second= 2
```



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
```



timezone

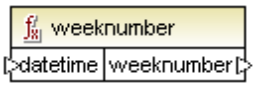
Devuelve el uso horario (p. ej. +05:00) en relación al UTC del valor **dateTime**. El uso horario se indica en minutos.

Ejemplo:
```
datetime="2001-12-17T09:30:02.544+05:00"
timezone= 300
```

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
```

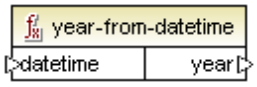


weeknumber

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
```

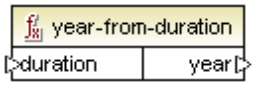


year-from-datetime

El resultado es el componente **`year`** del argumento **`dateTime`**.

Ejemplo:

```
datetime="2001-12-17T09:30:02.544+05:00"
year= 2001
```



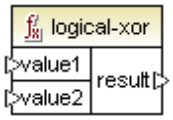
year-from-duration

El resultado es el componente **`year`** del argumento **`duration`**.

Ejemplo:

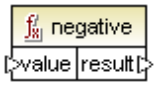
```
duration="P01Y04M0DT05H07M02.227S"
year= 1
```

logical (funciones lógicas)



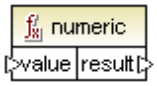
logical-xor

El resultado es `true` si **`value1`** es distinto de **`value2`**. De lo contrario, el resultado es `false`.



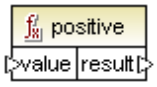
negative

El resultado es `true` si **`value`** es negativo (es decir, inferior a cero). De lo contrario, el resultado es `false`.



numeric

El resultado es `true` si **`value`** es un número. De lo contrario, el resultado es `false`. La entrada suele ser una cadena.

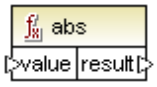


positive

El resultado es `true` si **`value`** es positivo (es decir, igual o mayor que cero). De lo contrario, el resultado es `false`.

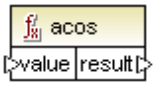
 © 2014 Altova GmbH

math (funciones matemáticas)



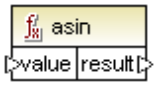
abs

El resultado es el valor absoluto del parámetro de entrada **`value`**.



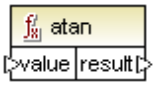
acos

El resultado es el arco coseno del parámetro de entrada **`value`**.



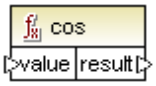
asin

El resultado es el arco seno del parámetro de entrada **`value`**.



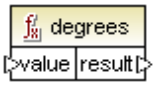
atan

El resultado es el arco tangente del parámetro de entrada **`value`**.



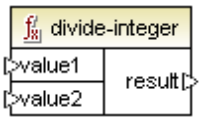
cos

El resultado es el coseno del parámetro de entrada **`value`**.



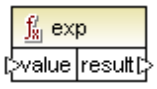
degrees

El resultado de convertir en grados el valor de **`value`** dado en radianes.



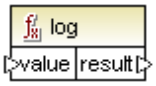
divide-integer

El resultado es el entero resultante de dividir **`value1`** por **`value2`**. Por ejemplo, **`15 divide-integer 2`**, el entero resultante es 7.



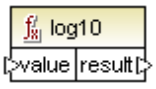
exp

El resultado es **`e`** (el logaritmo natural base) elevado a la potencia **`value`**.



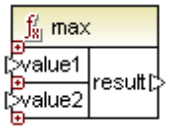
log

El resultado es el logaritmo natural de **`value`**.



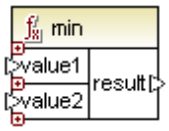
log10

El resultado es el logaritmo (base 10) de **`value`**.



max

El resultado es el valor numéricamente mayor de **`value1`** comparado con **`value2`**.



min

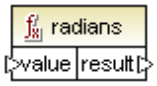
El resultado es el valor numéricamente menor de **`value1`** comparado con **`value2`**.

pi

El resultado es el valor de π.

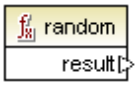
pow

El resultado es el valor de **`a`** elevado a la potencia **`b`**.



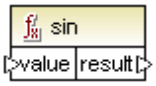
radians

El resultado de convertir en radianes el valor de **`value`** dado en grados.



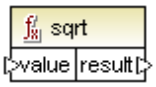
random

El resultado es un valor pseudoaleatorio comprendido entre 0.0 y 1.0



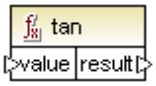
sin

El resultado es el seno de **`value`**.



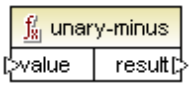
sqrt

El resultado es la raíz cuadrada de **`value`**.



tan

El resultado es la tangente de **`value`**.

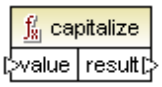


unary-minus

El resultado es la negación del valor con signo de **`value`**. Por ejemplo, +3 da -3 como resultado. -3 da 3 como resultado.

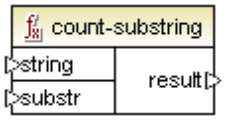
 © 2014 Altova GmbH

string (funciones de cadena)



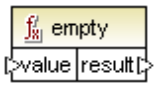
capitalize

El resultado es la cadena de entrada **`value`** después de poner en mayúsculas la primera letra de cada palabra.



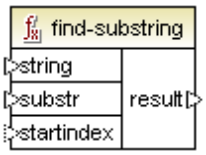
count-substring

El resultado es el número de veces que la cadena **`substr`** aparece en el parámetro **`string`**.



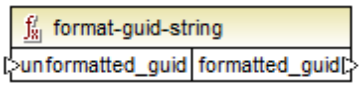
empty

El resultado es **`true`** si la cadena de entrada **`value`** está vacía. De lo contrario, el resultado es **`false`**.



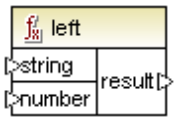
find-substring

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.



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 **generator**.

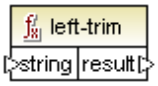


left

El resultado es una cadena que contiene los primeros **n** caracteres de **`string`**.

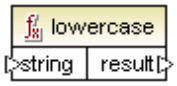
Ejemplo:

```
string="Esto es una frase"
number=4
result=Esto
```



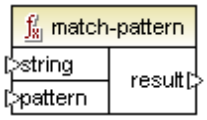
left-trim

El resultado es la cadena de entrada **`string`** después de quitarle todos los espacios en blanco iniciales.



lowercase

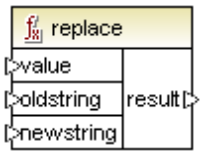
El resultado es la cadena de entrada **string** en minúsculas. Para los caracteres Unicode se usan los caracteres correspondientes en minúsculas.



match-pattern

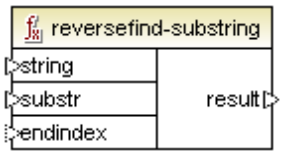
El resultado es **true** si la cadena de entrada **string** coincide con la expresión regular definida por el parámetro **pattern**. De lo contrario, el resultado es **false**. La sintaxis regex depende del lenguaje de destino seleccionado.

Para más información consulte el apartado Expresiones regulares.



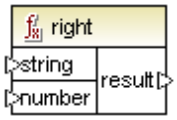
replace

El resultado es una cadena nueva formada por la cadena de entrada **value**, después de sustituir cada instancia de **oldstring** por **newstring**.



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.

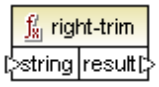


right

El resultado es una cadena que contiene los últimos **n** caracteres de la cadena **string**.

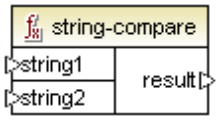
Ejemplo:

```
string="Esto es una frase"
number=5
result=frase
```



right-trim

El resultado es la cadena de entrada **string** después de eliminar todos los caracteres de espacio en blanco finales.



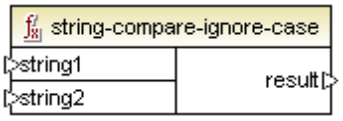
string-compare

 © 2014 Altova GmbH

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

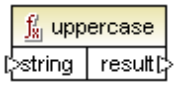


string-compare-ignore-case

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.

Si **`string1`** es menor que **`string2`**, entonces el resultado es < 0.

Si **`string1`** es mayor que **`string2`**, entonces el resultado es > 0



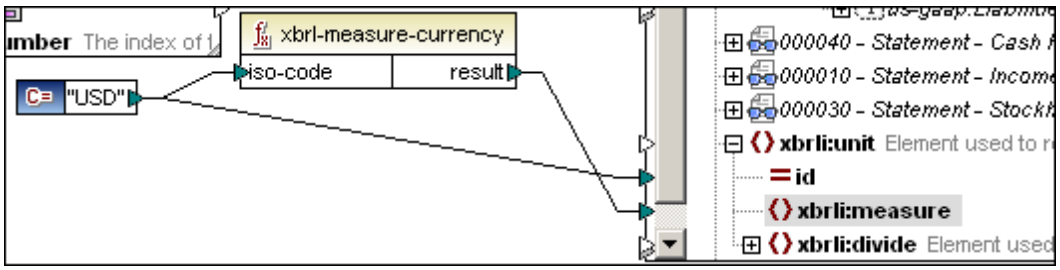
uppercase

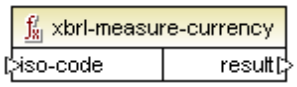
El resultado es la cadena de entrada **`string`** después de pasarla toda a mayúsculas. Para los caracteres Unicode se usan los caracteres correspondientes en mayúsculas.

15.5.5 Biblioteca xbrl

La biblioteca **xbrl** contiene funciones que convierten datos en el formato QName necesario para los archivos de instancia/taxonomía XBRL.

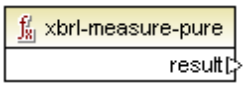
Los elementos secundarios `xbrl:ID` y `xbrl:measure` del elemento `xbrl:unit` son obligatorios en los archivos XBRL de instancia y, por tanto, deben tener asignaciones de datos. De lo contrario, la asignación no es válida.





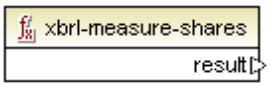
xbrl-measure-currency

El resultado es el QName para elementos de tipo `monetaryItemType` del código de divisas ISO 4217.



xbrl-measure-pure

El resultado es el QName para elementos de tasas, porcentajes y relaciones.



xbrl-measures-shares

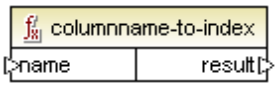
El resultado es el QName para elementos de tipo `sharesItemType`.

 © 2014 Altova GmbH

15.5.6 Biblioteca xlsx

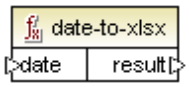
La biblioteca **xlsx** ofrece funciones para convertir datos de fecha y hora Excel.

Cuando genere código para C#, estas funciones dependen de la compatibilidad con componentes Excel, lo cual exige .NET 3.0 o superior. Por tanto, es necesario seleccionar *Visual Studio 2005 / 2008 o 2010* en la pestaña ***Generación*** de código del cuadro de diálogo "Opciones", bajo el grupo de opciones *Configuración de C#*.



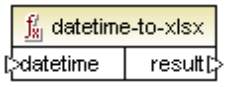
columnname-to-index

El resultado es el índice de la columna cuyo nombre coincide con el parámetro **`name`** (la columna 1 se llama A).



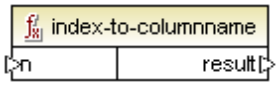
date-to-xlsx

El resultado es la representación Excel del valor de fecha/hora / fecha y hora del parámetro **`date`**.



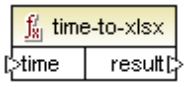
datetime-to-xlsx

El resultado es la representación Excel del valor de fecha y hora del parámetro **`datetime`**.



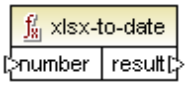
index-to-columnname

El resultado es el nombre de la columna número **n** (la columna A es la columna número 1).



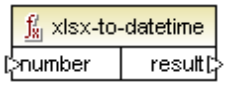
time-to-xlsx

El resultado es la representación Excel del valor de hora del parámetro **`time`**.



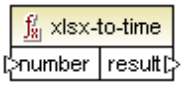
xslx-to-date

El resultado es el valor de fecha del campo `date` de Excel.



xlsx-to-datetime

El resultado es el valor de fecha y hora del campo `datetime` de Excel.



xlsx-to-time

El resultado es el valor de hora del campo `time` de Excel.

15.5.7 Biblioteca xpath2

La biblioteca **xpath2** contiene funciones para trabajar con los lenguajes XSLT2 y XQuery.

- funciones de descriptores de acceso
- funciones anyURI
- funciones booleanas
- constructores
- funciones de contexto
- funciones de duración, fecha y hora
- funciones de nodo
- funciones numéricas
- funciones para QName
- funciones de cadena

accessors (descriptores de acceso)

La biblioteca **xpath2 | accessors** ofrece estas funciones de descriptores de acceso:

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.

document-uri
No se implementa

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**).

string
La función **string** funciona igual que el constructor `xs:string`: convierte su argumento en `xs:string`.

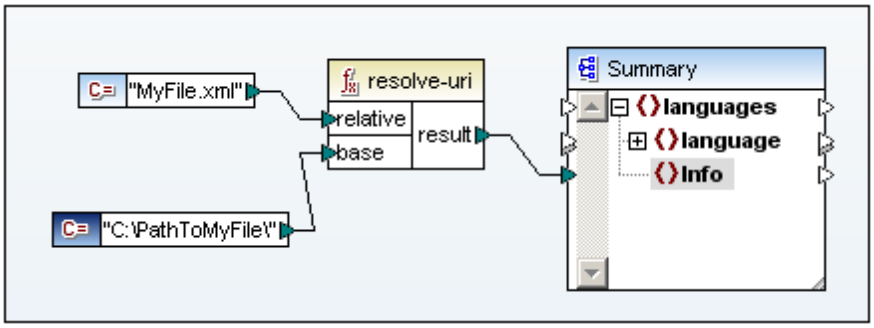
Cuando el argumento de entrada es un valor de un tipo atómico (por ejemplo `xs:decimal`), este valor atómico se convierte en un valor de tipo `xs:string`. Si el argumento de entrada es un nodo, se extrae el valor de cadena del nodo. (El valor de cadena de un nodo es una concatenación de los valores de los descendientes del nodo.)

anyURI

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`).

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.

 © 2014 Altova GmbH



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.

boolean (funciones booleanas)

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.

Función `true`

Inserta el valor booleano `true`.

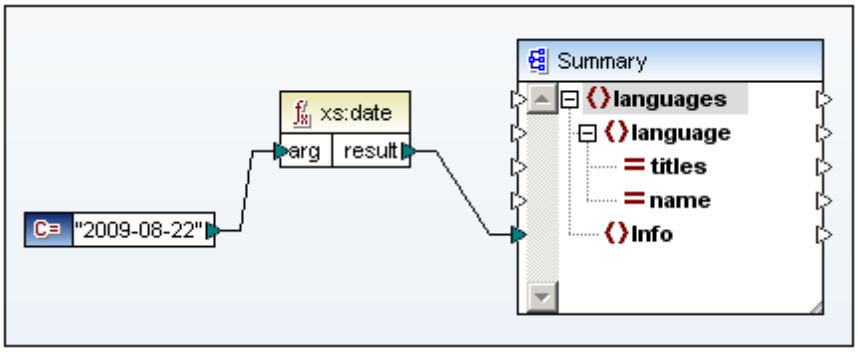
Función `false`

Inserta el valor booleano `false`.

constructores

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 mouse encima del argumento de entrada de la función aparece el tipo de datos esperado para el argumento.

context (funciones de contexto)

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.

Funciones de fecha y hora

Las funciones **`current-date`**, **`current-time`** y **`current-dateTime`** no tienen argumentos y devuelven la fecha/hora actual del reloj del sistema.

El tipo de datos del resultado depende de la función: **`current-date`** devuelve `xs:date`, **`current-time`** devuelve `xs:time` y **`current-dateTime`** devuelve `xs:dateTime`.

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.

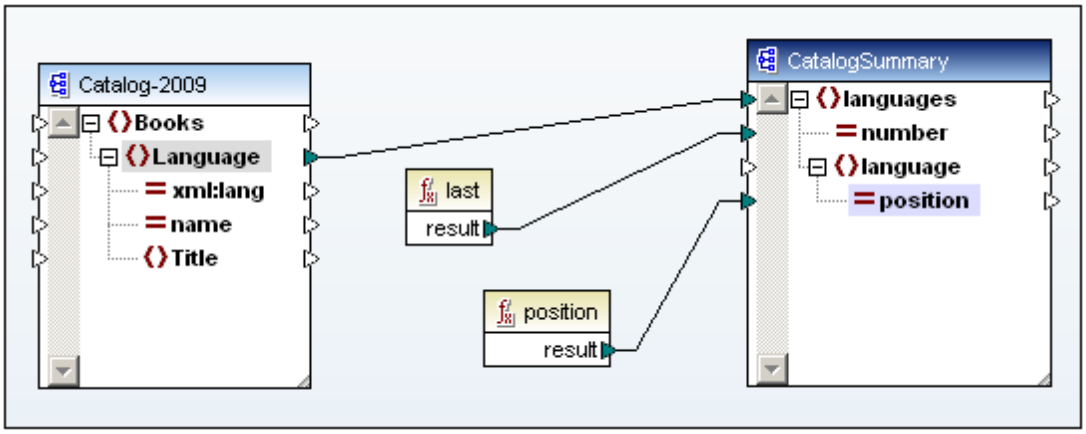
last, position

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.

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

 © 2014 Altova GmbH

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**.

duration, date and time (funciones de duración, fecha y hora)

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

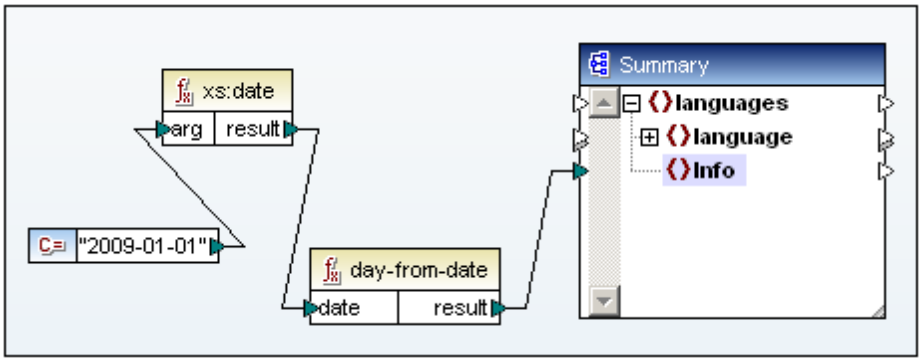
original se sustituye con el uso horario del segundo argumento.

- 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`

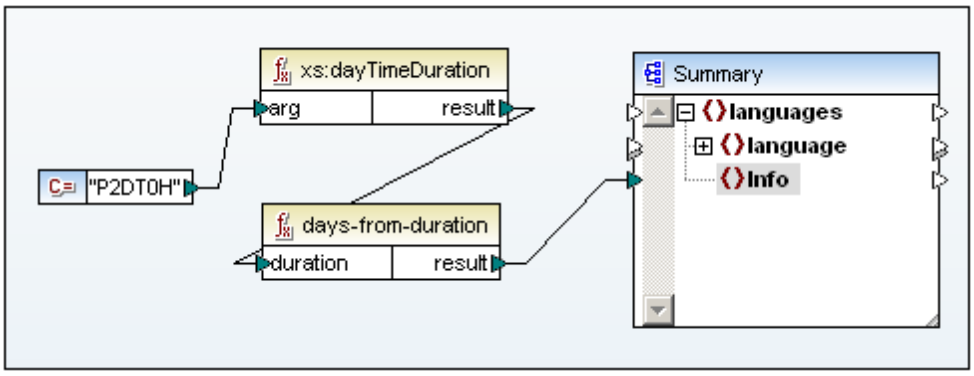
Cada función **`From`** extrae un componente de los (i) datos de fecha u hora o de los (ii) datos de duración. Los resultados son de tipo de datos `xs:decimal`.

Por ejemplo, si queremos extraer un componente de los datos de fecha u hora, podemos usar la función **`day-from-date`** (*imagen siguiente*).



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`.



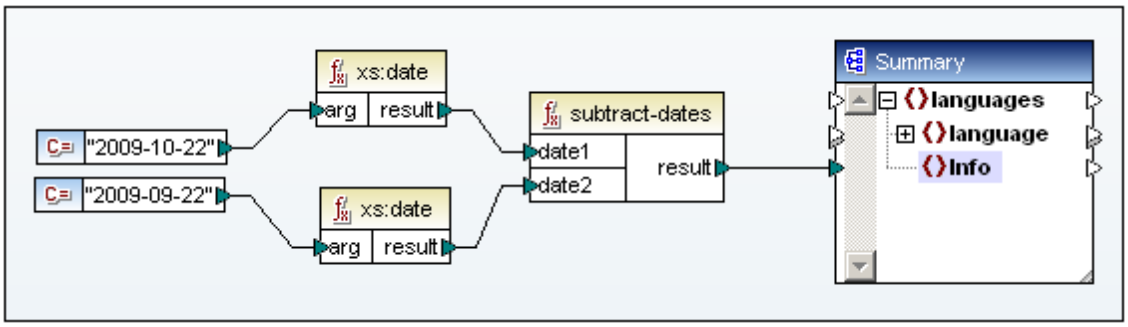
Funciones `Substract`

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-`**

 © 2014 Altova GmbH

`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`.



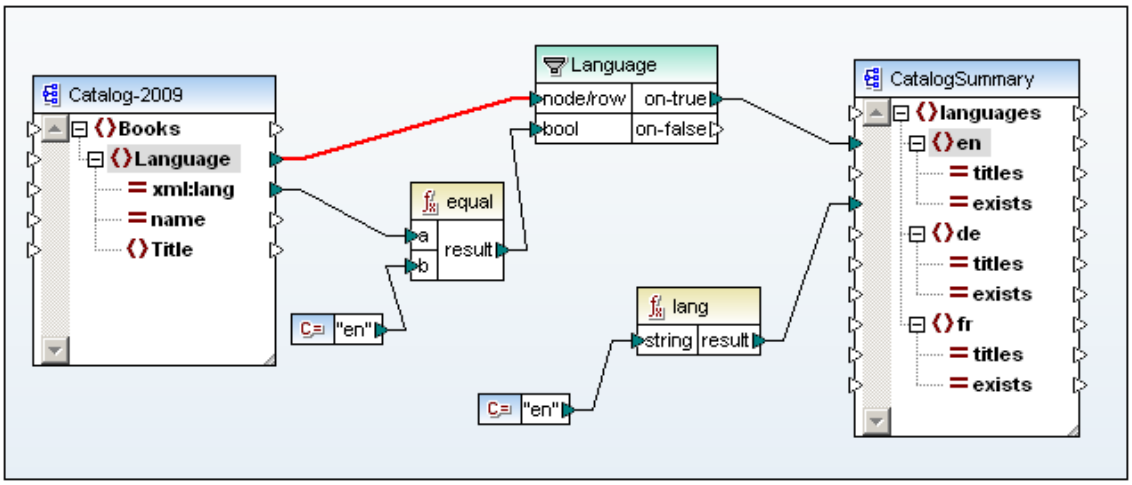
Nota: al pasar el puntero del mouse por el argumento de entrada de la función aparece el tipo de datos esperado del argumento.

node (funciones de nodo)

La biblioteca **xpath2** 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 `http://www.altova.com/examples`).

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.

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.

numeric (funciones numéricas)

La biblioteca de funciones **xpath2** 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.

 © 2014 Altova GmbH

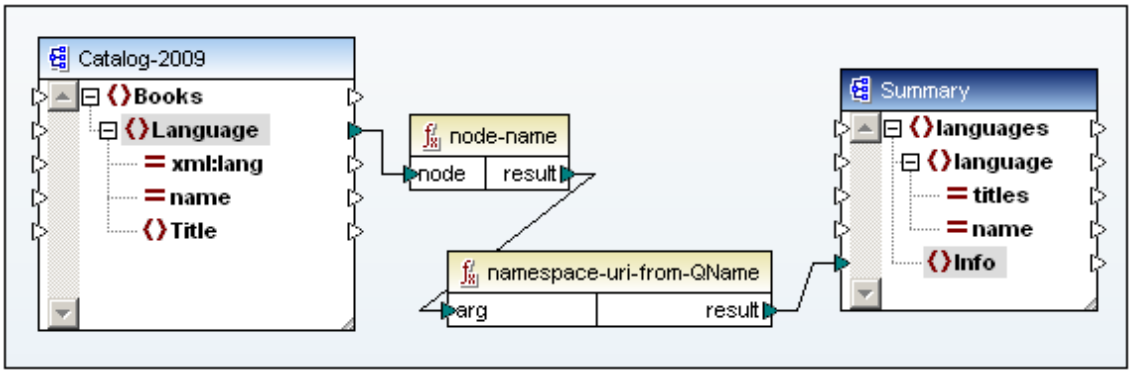
qname-related (funciones para QName)

En la biblioteca **xpath2 | qname-related** hay dos funciones muy similares: **`local-name-from-QName`** y **`namespace-uri-from-QName`**.

Ambas funciones toman un QName expandido (en forma de una cadena de texto) como argumentos de entrada y salida (el nombre local y el URI de espacio de nombres del QName expandido respectivamente).

Es importante tener en cuenta que, puesto que ambas funciones tienen cadenas de texto como entrada, no se puede conectar un nodo al argumento de entrada directamente.

El nodo debe enviarse primero a la función `node-name`, que devuelve el QName expandido. Este QName expandido se puede enviar como entrada de ambas funciones (*imagen siguiente*).



Ambas funciones dan como resultado una cadena de texto.

string (funciones de 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 (`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

 © 2014 Altova GmbH

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.

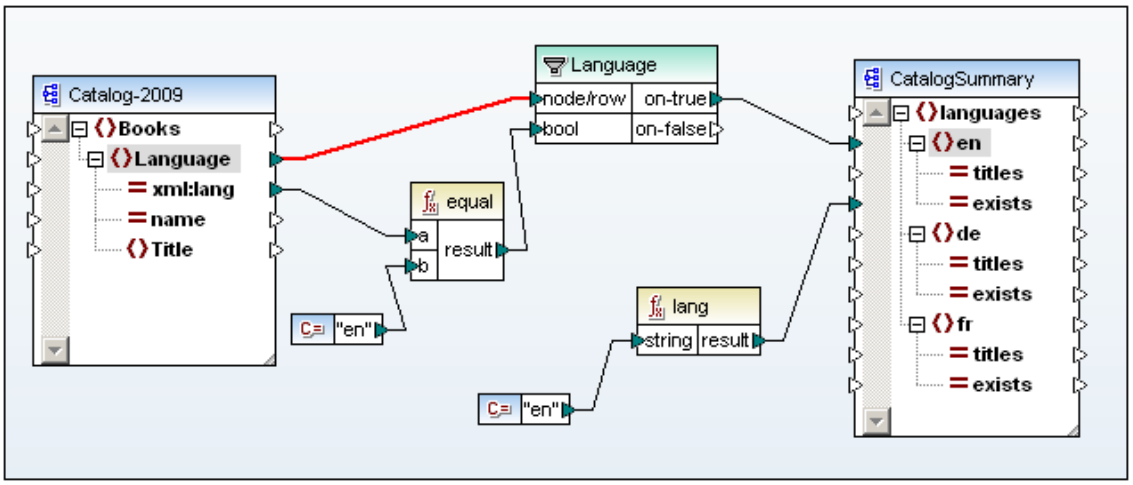
15.5.8 Biblioteca xslt

- funciones **xpath**
  Las funciones de la biblioteca **xpath** son funciones de conjuntos de nodos XPath 1.0.
- funciones **xslt**
  Las funciones de la biblioteca **xslt** son funciones XSLT 1.0.

xpath

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:

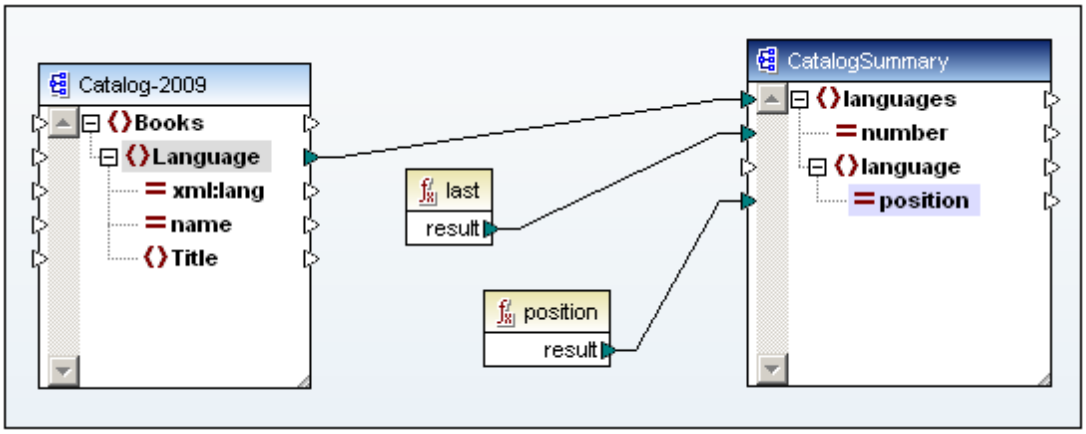
1. El elemento `Language` del esquema de origen tiene un atributo `xml:lang`.
2. Los nodos `Language` se filtran para que solamente se procesen aquellos cuyo atributo `xml:lang` tenga el valor `en` (la prueba de filtro se especifica en la función **`equal`**).
3. El nodo `Language` es el nodo de contexto 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

 © 2014 Altova GmbH

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`**.



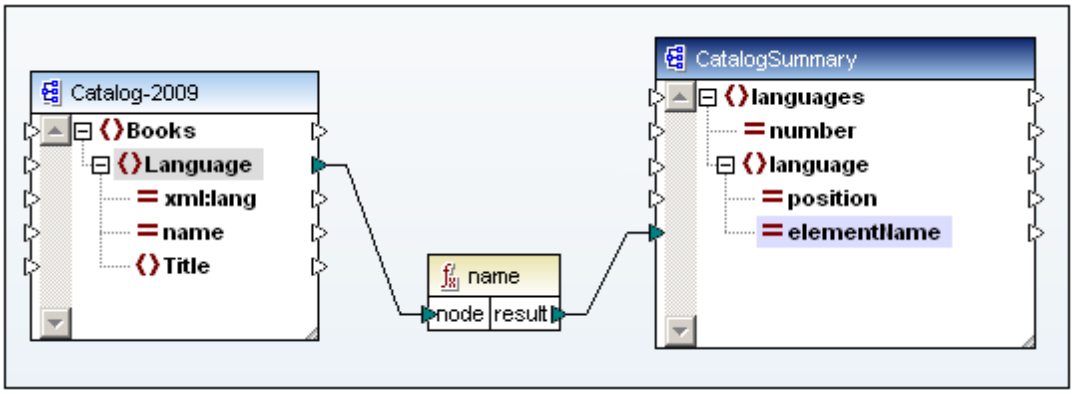
En el ejemplo, la función **`last`** devuelve la posición del último nodo del conjunto de nodos de contexto (el conjunto de nodos de los elementos `Language`) como valor del atributo `number`. Este valor también es el tamaño del conjunto de nodos, puesto que indica el número de nodos que contiene el 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`**.

name, local-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 la imagen siguiente 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="http://www.altova.com/namespaces"`.

Nota: en la biblioteca de funciones **core** puede encontrar más funciones XPath 1.0.

xslt

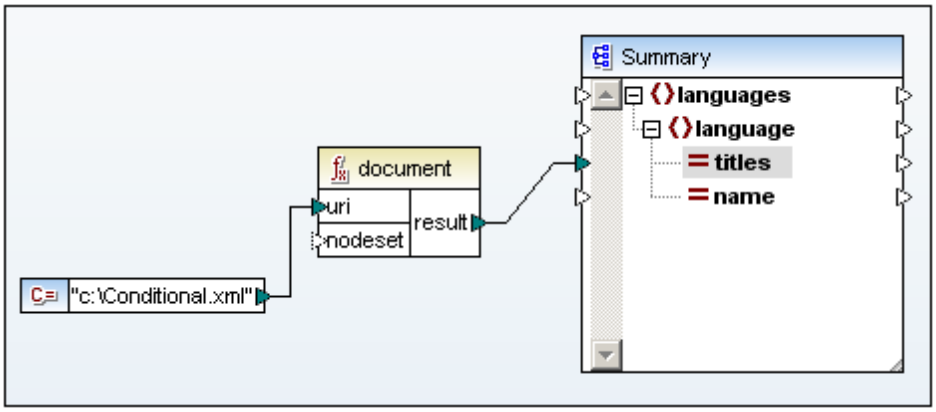
Las funciones de la biblioteca **xslt | xslt** son funciones XSLT 1.0. Pase el puntero del mouse por el argumento de entrada de la función para ver el tipo de datos esperado del argumento.

current

Esta función no toma argumentos y devuelve el nodo actual.

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.

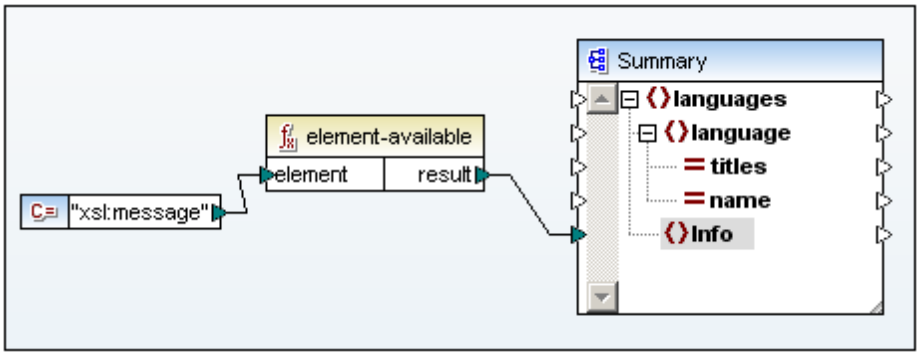


 © 2014 Altova GmbH

No olvide que el argumento **`uri`** es una cadena y debe ser la ruta de acceso absoluta de un archivo.

element-available

Esta función comprueba si un elemento es compatible con el procesador XSLT. La cadena del argumento se evalúa como QName. Por tanto, los elementos XSLT deben tener un 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.

function-available

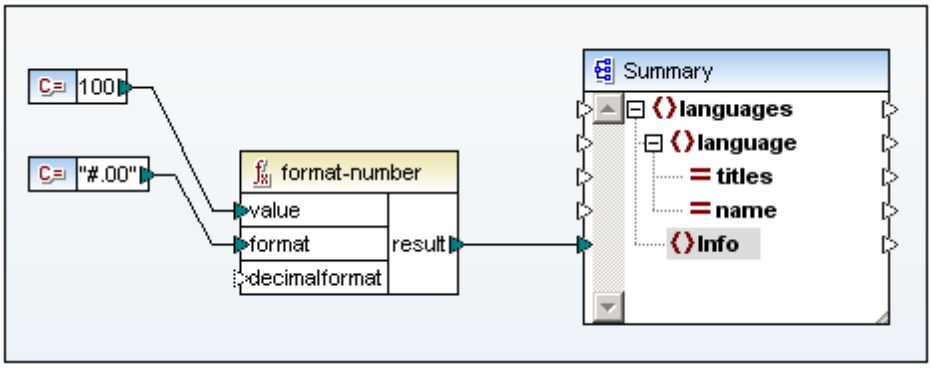
Esta función es similar a la función **`element-available`** y comprueba si el nombre de la función dado como argumento es compatible con el procesador XSLT.

La cadena de entrada se evalúa como QName. La función devuelve un valor booleano.

format-number

Esta función toma un entero como primer argumento (**`value`**) y una cadena de formato como segundo argumento (**`format`**). El tercer argumento es opcional y es una cadena que indica el formato decimal que se debe utilizar. Si falta este tercer argumento, se usa el formato decimal predeterminado.

Los formatos decimales los define el elemento XSLT 1.0 **`decimal-format`**: cada formato decimal definido puede tener un nombre y el nombre se puede usar como tercer argumento de la función **`format-number`**. Si se define un formato sin nombre, se usa el formato decimal predeterminado.



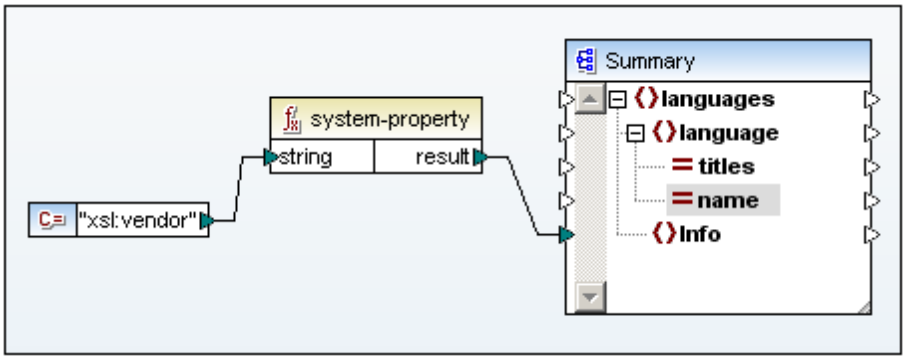
La función devuelve el número con formato de cadena.

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.

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.



unparsed-entity-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).

 © 2014 Altova GmbH

Altova MapForce 2014

Implementar servicios web

16 Implementar servicios web

En MapForce puede **crear servicios web SOAP** a partir de un archivo WSDL 1.1/2.0 diseñado en XMLSpy, por ejemplo, para asignar datos de operaciones WSDL y generar código de programa en Java o C# que implemente el servicio web. Puede ver una vista previa de los resultados en el panel **Resultados** (es decir, con el motor de ejecución integrado). Para terminar, basta con compilar el código generado e implementar el servicio web en el servidor web elegido.

MapForce admite el uso de proyectos WSDL compuestos por varias asignaciones, cada una de las cuales representa una operación de servicio web. Si genera código para todo el proyecto WSDL, se crea un servidor de servicio web entero. También puede generar código para cada una de las operaciones/asignaciones para la fase de pruebas.

Tenga en cuenta que este manual no describe la instalación ni la configuración del software de servidor web necesario. Consulte la documentación del paquete de software o pida a su departamento de TI que le instale el software necesario.

Requisitos para generar e implementar servicios web Java:

- MapForce Enterprise Edition
- Java 2 Software Development Kit (1.6 o superior): http://java.sun.com/j2se/
- Apache Tomcat: http://tomcat.apache.org
- Apache Axis2: http://ws.apache.org/axis2/, un marco SOAP que se ejecuta dentro de Tomcat
- Apache Ant: http://ant.apache.org/

Requisitos para generar servicios web C#:

- MapForce Enterprise Edition
- Microsoft Visual Studio 2005 o superior
- Microsoft Internet Information Services (IIS) versión 5.0 o superior.

El archivo de proyecto de MapForce y los demás archivos utilizados en esta sección del manual están disponibles en la carpeta `...\MapForceExamples\Tutorial`.

 © 2014 Altova GmbH

16.1 Información sobre WSDL: protocolos compatibles

El archivo WSDL necesario para producir un servicio web describe la interfaz del servicio web. El archivo WSDL debe crearse fuera de MapForce. Para ello puede usar XMLSpy, por ejemplo.

PortType (WSDL 1.1)
Un elemento `<portType>` define una interfaz de servicio web. Es decir:

- define las **operaciones** que se pueden llevar a cabo y
- define los **mensajes** que funcionan como entrada y salida de cada operación.

Tipos: (WSDL 1.1)
Un 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 corriente para archivos WSDL. MapForce representa estos elementos (tipos de datos) como elementos de un componente (mensaje) para que el usuario pueda asignarlos a otros elementos/construcciones.

Mensajes: (WSDL 1.1)
El elemento `<message>` define las **partes** de cada mensaje y los **elementos de datos** de los parámetros de entrada y salida de una operación. Se trata de los mensajes que intercambian el cliente y el servidor. Hay tres tipos de mensaje: entrada, salida y error.

En MapForce cada **mensaje** es un **componente** (por ejemplo, `getPersonSoapIn`) al que se le pueden asignar otros elementos. Los mensajes pueden incluir partes de mensaje.

Operaciones: (WSDL 1.1)
Las operaciones usan mensajes como parámetros de entrada y salida. Una operación debe tener:

- un mensaje de entrada
- cero o más mensajes de salida
- cero o más mensajes de error

Nota:

- Los mensajes de **entrada** solamente se pueden usar como componentes de **origen**
- Los mensajes de **salida y error** solamente se pueden usar como componentes de **destino**

En MapForce cada **operación** es un **archivo de asignación** (un archivo *.mfd). Todas las operaciones se agrupan en un archivo de proyecto WSDL de MapForce que describe el servicio web entero.

WSDL 2.0 es bastante diferente de WSDL. Las diferencias principales son:

- Los portType se llaman **interfaces** en WSDL 2.0.
- En WSDL 2.0 los **mensajes** y las **partes** se definen usando el sistema de tipos XML Schema en el elemento `types`.
- Los puertos se llaman **extremos** en WSDL 2.0.
- En WSDL 2.0 las entradas y salidas de las operaciones las define el esquema XML.

  En el cuadro de diálogo "Configuración del componente" de un componente WSDL de MapForce se usa el término *Extremo*, tanto para puertos WSDL 1.1 como para extremos WSDL 2.0.

Compatibilidad con versión 1.1, nota del W3C de http://www.w3.org/TR/wsdl

WSDL:	versión 2.0, recomendación del W3C de http://www.w3.org/TR/wsdl20/
Sistema de tipos WSDL:	XML Schema 2001
Compatibilidad con 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 sobre HTTP (los protocolos HTTP POST, HTTP GET son incompatibles).
C#	En C# SOAPAction debe ser diferente para cada operación
Enlaces:	no se pueden usar varias operaciones con el mismo nombre (WSDL 2.5).
estilo/uso:	• documento/literal: compatible. • RPC/literal: compatible en C# • RPC/encode: compatibilidad limitada • un estilo/uso por cada servicio web (Java) o por cada operación (C#)
Encabezados SOAP:	depende de la plataforma subyacente.
Atributo SOAP encodingStyle:	Si use="encoded", se asume el estilo de codificación "http://schemas.xmlsoap.org/soap/encoding/" para todo el soap:Body y ningún otro estilo de codificación es compatible.
SOAP:	el atributo encodingStyle se omite en los mensajes (SOAP 4.1.1).
Codificación SOAP:	• no son compatibles las direcciones href para recursos externos (SOAP 5.4.1). • solamente son compatibles las referencias a elementos independientes
SOAP-ENC:Array:	el acceso lineal es compatible. No son compatibles las matrices parciales ni dispersas.
Mejoras personales de SOAP:	incompatibles
Valores predeterminados o fijos de esquemas:	incompatibles
Validación de mensajes no SOAP	sin validar, se pasan al marco subyacente.
Espacios de nombres	las entradas que no sean de espacio de nombres no son válidas en WSDL y por tanto son incompatibles (en WSDL y XML 1.0)

Nota sobre WSDL 1.1:
Cuando use la combinación documento/literal en MapForce, es necesario que el elemento

 © 2014 Altova GmbH

`message / part` haga referencia a un elemento global, en lugar de a un tipo. Es decir:

El atributo `element` hace referencia a un **elemento global** definido en un esquema, el elemento `ns2:Vendor`:

```
<message name="processRequest">
        <part name="inputData" element="ns2:Vendor"/>
</message>
```

Mientras que el ejemplo siguiente hace referencia a un **tipo** del esquema:

```
<message name="processRequest">
        <part name="inputData" type="ns2:VendorType"/>
</message>
```

16.2 Crear proyectos de servicio web a partir de archivos WSDL

Objetivo del proyecto de servicio web

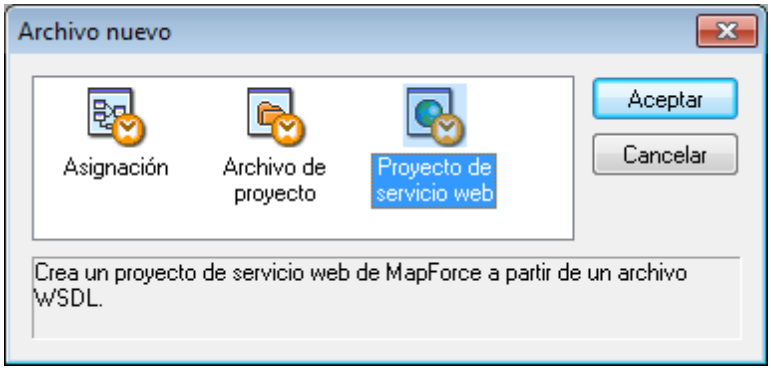
En este apartado crearemos un servicio web que permitirá al usuario:

- usar una solicitud SOAP para buscar un conjunto determinado de personas en una base de datos MS Access de un servidor. (La consulta se define/introduce en tiempo de ejecución en el cliente y después se envía al servidor.)
- recuperar los registros de la base de datos como una respuesta SOAP, que tome el formato de un documento XML que se devuelve al cliente y que contiene todas las personas encontradas por la consulta.

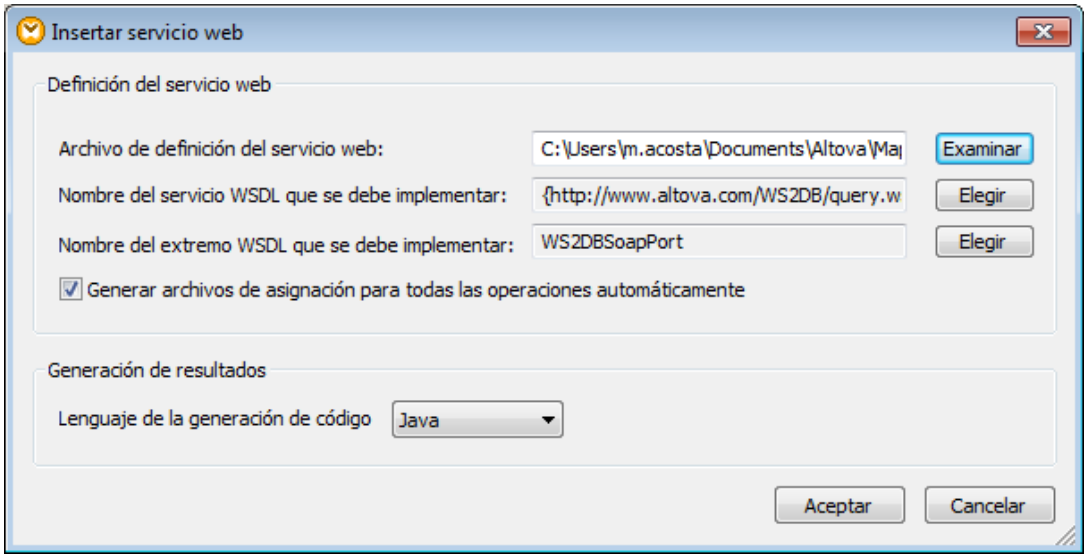
Nota: el proceso de asignación utilizado para generar el servicio web no depende del lenguaje de programación de destino. Se trata del mismo proceso, ya sean servicios web Java o C#. Las únicas diferencias existentes aparecen a la hora de compilar e implementar el servicio web en el servidor web.

Crear un proyecto de servicio web

1. Seleccione la opción de menú **Archivo | Nuevo**.
2. Seleccione el icono **Servicio de proyecto web** y haga clic en **Aceptar**.



3. Rellene el cuadro de diálogo "Insertar servicio web" que aparece.



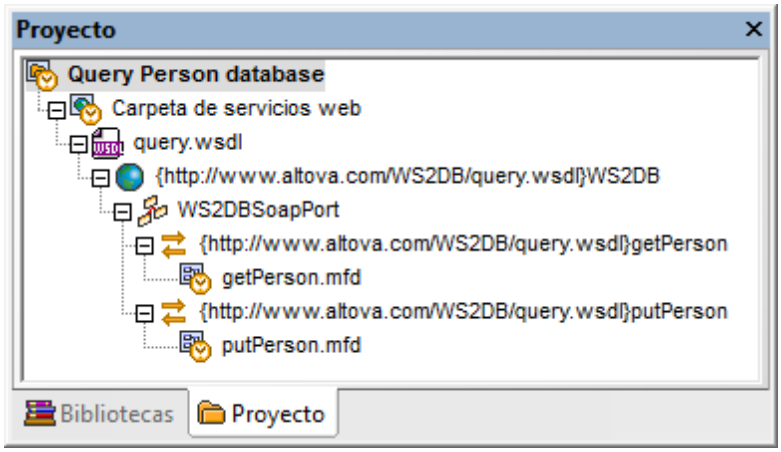
4. En el campo *Archivo de definición del servicio web* seleccione el archivo WSDL

 © 2014 Altova GmbH

`query.wsdl` de la carpeta `...\MapForceExamples\Tutorial`. Al seleccionar el archivo WSDL, MapForce rellena automáticamente los demás campos del cuadro de diálogo. Si su archivo WSDL contiene varias opciones, aparece otro cuadro de diálogo donde puede elegir el servicio o puerto WSDL que desea implementar.

5. Haga clic en **Aceptar**.
6. Escriba el nombre del proyecto WSDL en el cuadro de diálogo "Guardar el proyecto como".

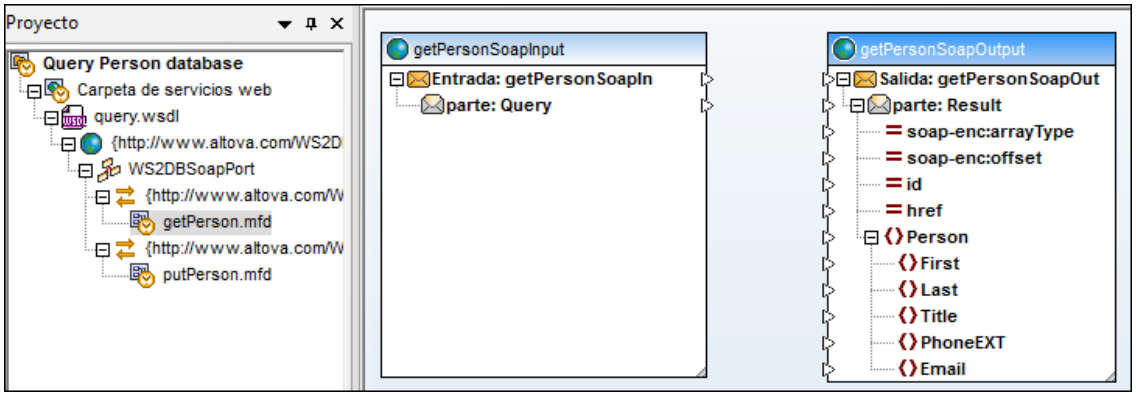
Haga clic en **Guardar** para confirmar la configuración y crear el archivo de proyecto WSDL.



En la ventana Proyecto aparece ahora el proyecto WSDL y todas las operaciones definidas en el archivo WSDL: las operaciones **getPerson** y **putPerson**.

Crear asignaciones de servicios web:

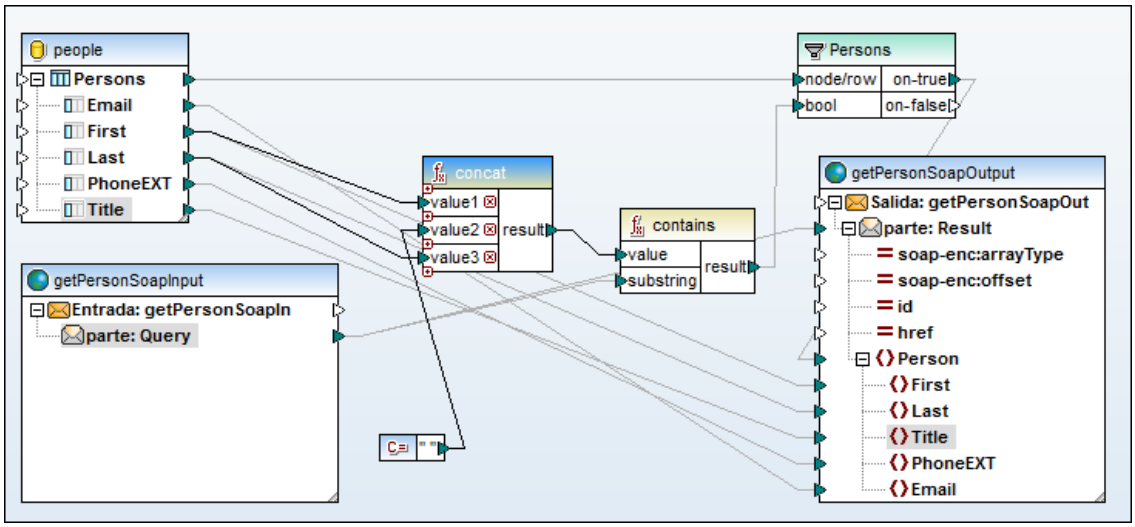
1. Haga doble clic en la operación **WSD2DB_getPerson** de la ventana Proyecto.
   Se crea una asignación nueva llamada `getPerson.mfd` que contiene dos componentes WSDL. El componente **getPersonSoapIn** contiene la consulta (**`query`**) que se usará para consultar la base de datos a través del servicio web. El componente **getPersonSoapOut** contiene los elementos **`Person`** definidos en el archivo WSDL.



Definir la asignación de la consulta:

1. Seleccione **Insertar | Base de datos** y utilice el asistente para insertar la base de datos `people.mdb` (disponible en la carpeta `...\Tutorial`).
   Los elementos de la BD coinciden con los del componente de destino **getPersonSoapOutput**.

2. Asigne los elementos **Person** de la BD a los elementos **Person** del componente **getPersonSoapOutput**.
3. Use la función **concat** para concatenar las columnas **First** y **Last** de la tabla **Persons**.
4. Inserte una función **contains** y conecte el parámetro **result** de la función **concat** con el parámetro **value** de la función **contains**.
5. Haga clic en el elemento **parte: Query** del componente **getPersonSoapInput** y conéctelo al parámetro **substring** de la función **contains**.
6. Haga clic otra vez en el elemento **parte: Query** y conéctelo al elemento **parte: Result** del componente de salida **getPersonSoapOutput**.
   El elemento **Query** del componente **getPersonSoapInput** es el marcador de posición de la consulta, es decir, donde se introduce la cadena de consulta en el cliente SOAP una vez generado, compilado e implementado el código.
7. Inserte un componente de filtrado y termine de crear las asignaciones entre los componentes tal y como muestra esta imagen.



Ahora puede generar código para crear un servicio web. Puede generar servicios web para Java o C#. Estas secciones describen la generación, compilación e implementación de servicios web para Java y C#.

 © 2014 Altova GmbH

16.3 Generar servicios web Java con MapForce

MapForce genera todo el código y los scripts necesarios para crear un servicio web. La única diferencia con el proceso de generación de código normal es que el código generado debe implementarse en el servidor Axis2 (Tomcat).

Tenga en cuenta que cuando el servidor web se ejecuta en un equipo remoto:

- el usuario debe tener derechos de administración remotos
- el marco Axis2 debe estar (parcialmente) instalado en el equipo local
- cuando se implementan servicios web en un equipo local (es decir, cuando el servidor es el equipo local) estos aspectos son irrelevantes.
- Axis2 es compatible tanto con SOAP 1.1 como con SOAP 1.2.

Generar código Java:
Tras crear (o abrir) el archivo de proyecto `Query Person database.mfp` del apartado anterior:

1. Seleccione la opción de menú **Proyecto | Generar código en | Java.**
2. Compile el código Java generado (usando Apache Ant, por ejemplo) para producir el archivo `*.aar` necesario para la implementación en Axis2

Estas carpetas y archivos se generan automáticamente en el directorio de destino:

- un directorio `com` con los subdirectorios `\altova` y `\MapForce`.

Implementación en Axis2
Tiene dos opciones (i) usar el servicio de carga de la página de administración para cargar el servicio web generado por MapForce como archivo `*.aar` o (ii) realizar una carga manual:

Si su versión de TomCat está instalada en la carpeta `C:\Archivos de programa\Apache Software Foundation\Tomcat 5.5` puede copiar el archivo `.aar` a mano al directorio `C:\Archivos de programa\Apache Software Foundation\Tomcat 5.5\webapps\axis2\WEB-INF\services`

Anulación de la implementación
Elimine el archivo `*.aar` del directorio `<ruta de tomcat>\webapps\axis2\WEB-INF\services.`

Limitaciones de Axis2
Axis2 no es totalmente compatible con el modo RPC/encode. Sin embargo, MapForce puede generar servicios web RPC/encode (tanto para SOAP 1.1 como para SOAP 1.2). La única restricción es que el archivo WSDL original no se recupera del servidor web.

Esto significa que, por ejemplo, `http://127.0.0.1/axis2/services/WS2DB?wsdl` **no devolvería** un archivo WSDL utilizable.

Para servicios web documento/literal la URL anterior ofrece un archivo WSDL correcto y utilizable. Sin embargo, será algo diferente del original: los comentarios se habrán eliminado y los espacios de nombres habrán cambiado. No obstante, tendrá la misma semántica que el archivo WSDL original con el que se generó el servicio.

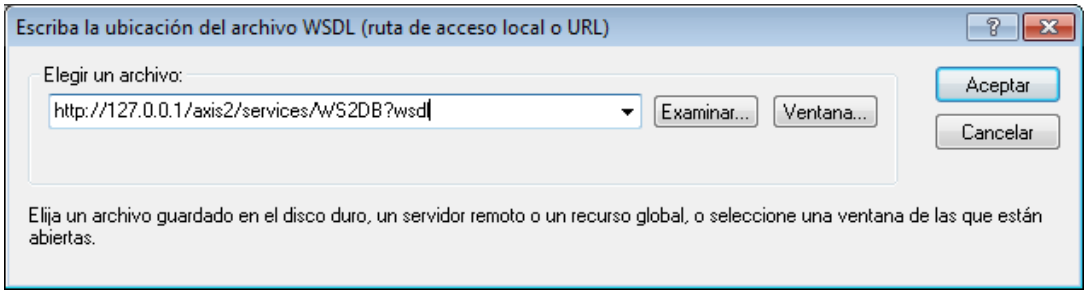
Aunque Axis2 no es compatible con el modo RPC/encoded, es capaz de generar WSDL a partir del código Java implementado (código compilado) y por tanto el código generado con MapForce puede procesar y procesa mensajes RPC/encode. Axis2 se usa solamente para el transporte.

16.3.1 Usar el servicio web: operación getPerson

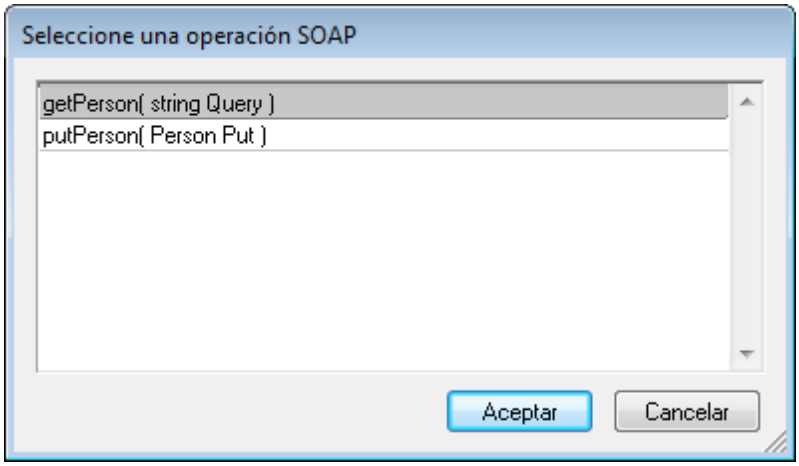
Usar el servicio web

Cree y envíe una solicitud SOAP con un cliente WSDL (p. ej. XMLSpy):

1. En XMLSpy seleccione la opción de menú **SOAP | Crear solicitud SOAP nueva**.
2. En el cuadro de diálogo que aparece escriba la ubicación del archivo WSDL (p. ej. `http://127.0.0.1/axis2/services/WS2DB?wsdl`) y haga clic en **Aceptar**.



3. Seleccione la operación SOAP que desea usar (**`getPerson( string Query)`**en este caso) y haga clic en **Aceptar** para crear el documento de solicitud SOAP.



La línea que contiene `<Query xsi:type="xsd:string">`**`String`**`</Query>` es el marcador de posición de la consulta, es decir, la cadena que se buscará en la base de datos.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/en
    <SOAP-ENV:Body>
        <m:getPerson xmlns:m="http://www.altova.com/WS2DB/query.wsdl"
            <Query xsi:type="xsd:string">String</Query>
        </m:getPerson>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

En el caso que nos ocupa queremos que el servicio web envíe los registros de todas las personas cuyo nombre o apellido contiene la cadena `Ro`.

4. Reemplace la cadena `String` por `Ro`.

```
<m:getPerson xmlns:m="http://www.altova.com/
    <Query xsi:type="xsd:string">Ro</Query>
```

 © 2014 Altova GmbH

5. Seleccione la opción de menú **SOAP | Enviar solicitud al servidor** para enviar la solicitud al servidor.

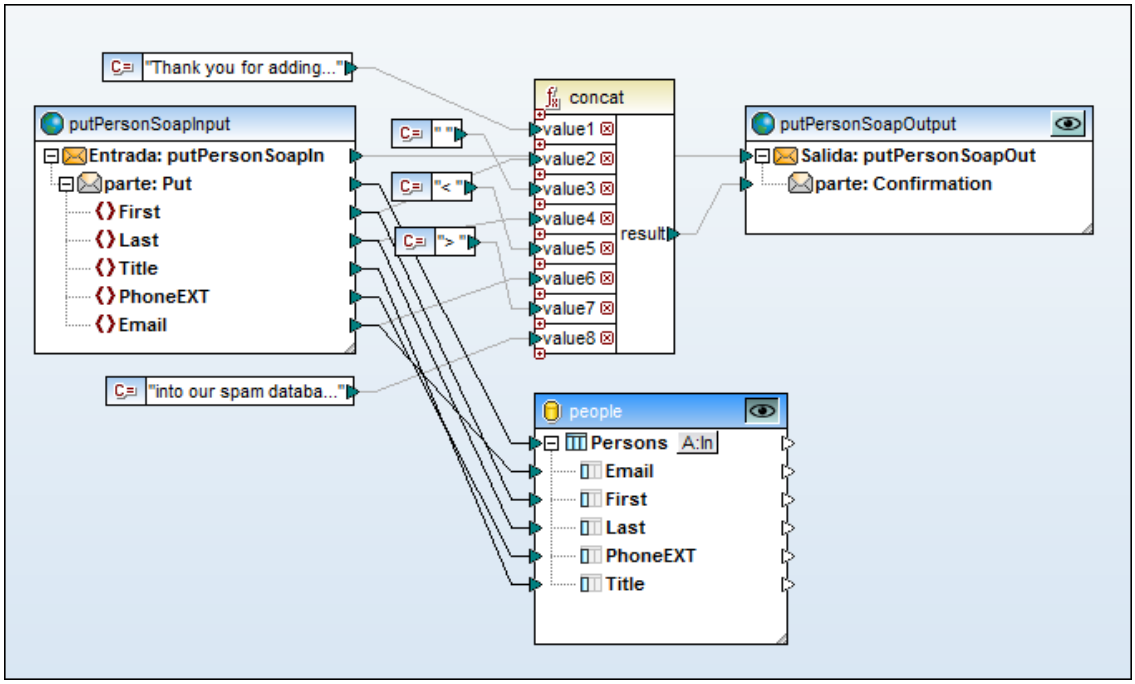

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/en
    <soapenv:Body>
        <m:getPersonResponse xmlns:n0="http://www.altova.com/WS2DB.
            <Result soapenc:arrayType="n0:Person[]" xsi:type="soape
                <Person xsi:type="n0:Person">
                    <First xsi:type="xsd:string">Martin</First>
                    <Last xsi:type="xsd:string">Rope</Last>
                    <Title xsi:type="xsd:string">Mr.</Title>
                    <PhoneEXT xsi:type="xsd:string">780</PhoneEXT>
                    <Email xsi:type="xsd:string">mr@strings.com</Em
                </Person>
                <Person xsi:type="n0:Person">
                    <First xsi:type="xsd:string">Ronald</First>
                    <Last xsi:type="xsd:string">Superstring</Last>
                    <Title xsi:type="xsd:string">Dr.</Title>
                    <PhoneEXT xsi:type="xsd:string">444</PhoneEXT>
                    <Email xsi:type="xsd:string">ros@strings.com</E
                </Person>
                <Person xsi:type="n0:Person">
                    <First xsi:type="xsd:string">Robert</First>
                    <Last xsi:type="xsd:string">Darkmatter</Last>
                    <Title xsi:type="xsd:string">Mathematician</Tit
                    <PhoneEXT xsi:type="xsd:string">299</PhoneEXT>
                    <Email xsi:type="xsd:string">rodark@strings.com
                </Person>
                <Person xsi:type="n0:Person">
                    <First xsi:type="xsd:string">Roger</First>
                    <Last xsi:type="xsd:string">Gravity</Last>
                    <Title xsi:type="xsd:string">Crisis manager</Ti
                    <PhoneEXT xsi:type="xsd:string">112</PhoneEXT>
                    <Email xsi:type="xsd:string">rog@strings.com</E
                </Person>
```


El servicio web devuelve el documento de respuesta SOAP y contiene los registros de 4 personas de la base de datos Access.

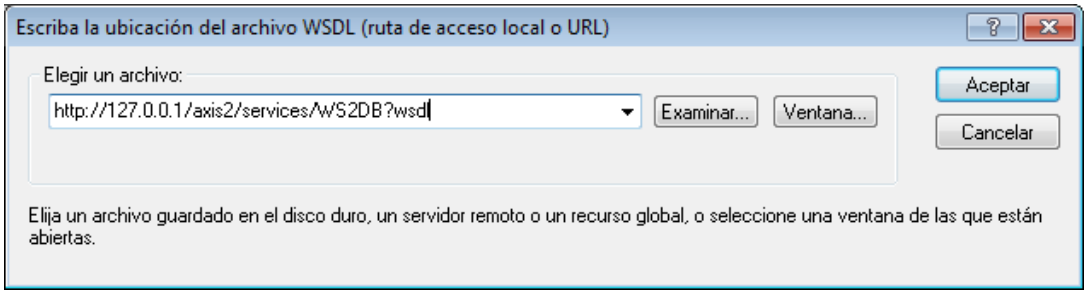
16.3.2 Usar el servicio web: operación putPerson

En la ventana Proyecto haga doble clic en el icono de la asignación **putPerson**.

El archivo `putPerson.mfd` se abre en una pestaña nueva.



1. En XMLSpy cree y envíe una solicitud SOAP.
2. Seleccione la opción **SOAP | Crear una solicitud SOAP nueva**.
3. Escriba la ubicación del archivo WSDL en el servidor (p. ej. `http://127.0.0.1/axis2/services/WS2DB?wsdl`) y haga clic en **Aceptar**.



4. Seleccione el nombre de la operación SOAP que desea usar (en este caso `putPerson(Person Put)`) y haga clic en **Aceptar** para crear el documento de solicitud SOAP. El documento de solicitud SOAP aporta los elementos de marcador de posición `Person` para que el usuario los rellene.

 © 2014 Altova GmbH

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/en
    <SOAP-ENV:Body>
        <m:putPerson xmlns:m="http://www.altova.com/WS2DB/query.wsdl"
            <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>
        </m:putPerson>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

5. Reemplace el texto del marcador de posición (`String`) con los datos reales.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/en
    <SOAP-ENV:Body>
        <m:putPerson xmlns:m="http://www.altova.com/WS2DB/query.wsdl
            <Put xsi:type="m0:Person">
                <First xsi:type="xsd:string">Fred</First>
                <Last xsi:type="xsd:string">Flagellator</Last>
                <Title xsi:type="xsd:string">Sir</Title>
                <PhoneEXT xsi:type="xsd:string">777</PhoneEXT>
                <Email xsi:type="xsd:string">ff@home.com</Email>
            </Put>
        </m:putPerson>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

6. Seleccione la opción de menú **SOAP | Enviar solicitud al servidor**.
   El comando **Enviar solicitud SOAP** envía el documento de solicitud SOAP editado al servidor.

   Los datos nuevos se añaden a la base de datos y se devuelve un documento de respuesta SOAP como confirmación. La asignación define el contenido de este mensaje de confirmación, es decir: *Gracias por añadir XYZ a nuestra base de datos de spam*.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd
http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instanc
    <soapenv:Body>
        <m:putPersonResponse xmlns:n0="http://www.altova.com/WS2DB.xsd" xmlns:xsi="
http://www.w3.org/2001/XMLSchema-instance" xmlns:m="http://www.altova.com/WS2DB/query
            <Confirmation xsi:type="xsd:string">Thank you for adding Fred Flagellator
&lt;ff@home.com&gt; into our spam database</Confirmation>
        </m:putPersonResponse>
    </soapenv:Body>
</soapenv:Envelope>
```

16.4 Generar servicios web C# con MapForce

En el archivo de consulta WSDL disponible en la carpeta `Tutorial` hay una sección que se convirtió en comentario. Se trata de la sección `<Service name="WS2DB">`, situada al final del archivo. Antes de empezar a trabajar con el ejemplo de este apartado, quite el comentario de esta sección y convierta en comentario la sección anterior (`<Service...>`). Para el ejemplo de este apartado asumimos que el servidor web está ubicado en el equipo local.

Nota: los esquemas URI para Axis2 (Java) e IIS (C#) son diferentes y es necesario realizar algunos cambios en el archivo WSDL. Las diferencias se concentran en el elemento `<service>`, que suele aparecer al final del archivo WSDL.

WSDL estilo Axis2

```
<service name="WS2DB">
        <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
                <soap:address location="http://localhost:8080/axis/services/WS2DB"/>
        </port>
</service>
```

WSDL estilo IIS

```
<service name="WS2DB">
        <port name="WS2DBSoapPort" binding="tns:WS2DBSoapBinding">
                <soap:address location="http://localhost/services/WS2DB.asmx"/>
        </port>
</service>
```

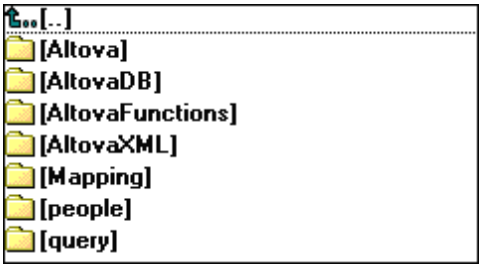
Los archivos WSDL de ejemplo que vienen con MapForce contienen ambas variantes, pero una de ellas es un comentario. Además, incluyen el parámetro **`soapAction`**, necesario para trabajar con IIS pero ignorado por Axis2.

Generar código C#:

Tras crear/abrir el archivo de proyecto `Query Person database.mfp` del apartado anterior:

1. Seleccione la opción de menú **Proyecto | Generar código en | C#**.
2. Seleccione el directorio de salida (`java-dev` en este caso) y haga clic en **Aceptar** para empezar a generar el código.
   Cuando el proceso se completa sin problemas, aparece el mensaje *El código se generó correctamente.* De lo contrario aparece un mensaje de error con todos los detalles.

Varias carpetas se generan automáticamente en el directorio de destino. Estas carpetas contienen el código C#, archivos de proyecto y archivos de solución.



Puede compilar el código generado en Visual Studio 2005/2008/2010.

Compilar el servicio web:

Nota: si usa Visual Studio 2005 o 2008 e IIS 7.x, quizás sea necesario instalar primero la característica de Windows *Compatibilidad con la configuración de IIS 6 y metabase de IIS.* También debe ejecutar Visual Studio como administrador.

Para las instrucciones que siguen a continuación asumimos que el servidor web está en el equipo local. Para compilar el servicio web para un equipo remoto, consulte la documentación de Microsoft (p. ej. http://learn.iis.net/page.aspx/387/using-visual-studio-2008-with-iis/).

1. Abra el archivo de solución generado `...\output\Query_Person_database\Query_Person_database_webservice.sln`.
2. Seleccione la opción de menú **Generar | Generar solución** para compilar el proyecto de servicio web.
3. Seleccione la opción de menú **Depurar | Iniciar depuración** para iniciar la aplicación.

Si usa un sistema operativo de 64 bits y el archivo de muestra no funciona porque falta el proveedor ADO (el proveedor ADO para Access solamente funciona con aplicaciones de 32 bits), puede hacer esto:

1. En Visual Studio seleccione la opción de menú **Generar | Administrador de configuración**, cree una plataforma de solución nueva para x86 y genere la solución.
2. Abra la consola de administración de IIS, busque el grupo de aplicaciones en el que se encuentra su aplicación, haga clic con el botón secundario en *Configuración avanzada* y en la tabla de propiedades que aparece cambie *Habilitar aplicaciones de 32 bits* a *True*.

Usar el servicio web

1. En XMLSpy cree y envíe una solicitud SOAP.
2. Seleccione la opción de menú **SOAP | Crear solicitud SOAP nueva**.
3. Escriba la ubicación del archivo WSDL en el servidor de IIS y haga clic en **Aceptar**.
4. Seleccione la operación SOAP que desea utilizar (`getPerson( string Query)`en este caso) y haga clic en **Aceptar** para crear el documento de solicitud SOAP.
   La línea que contiene `<Query xsi:type="xsd:string">String</Query>` es el marcador de posición para la consulta, es decir, la cadena que desea buscar en la base de datos.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/en
    <SOAP-ENV:Body>
        <m:getPerson xmlns:m="http://www.altova.com/WS2DB/query.wsdl"
            <Query xsi:type="xsd:string">String</Query>
        </m:getPerson>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

   En este caso queremos que el servicio web nos entregue los registros de todas aquellas personas cuyo nombre o apellido contenga la cadena `Ro`.
4. Reemplace la cadena `String` con la cadena de consulta `Ro`.

```
<m:getPerson xmlns:m="http://www.altova.com/
    <Query xsi:type="xsd:string">Ro</Query>
```

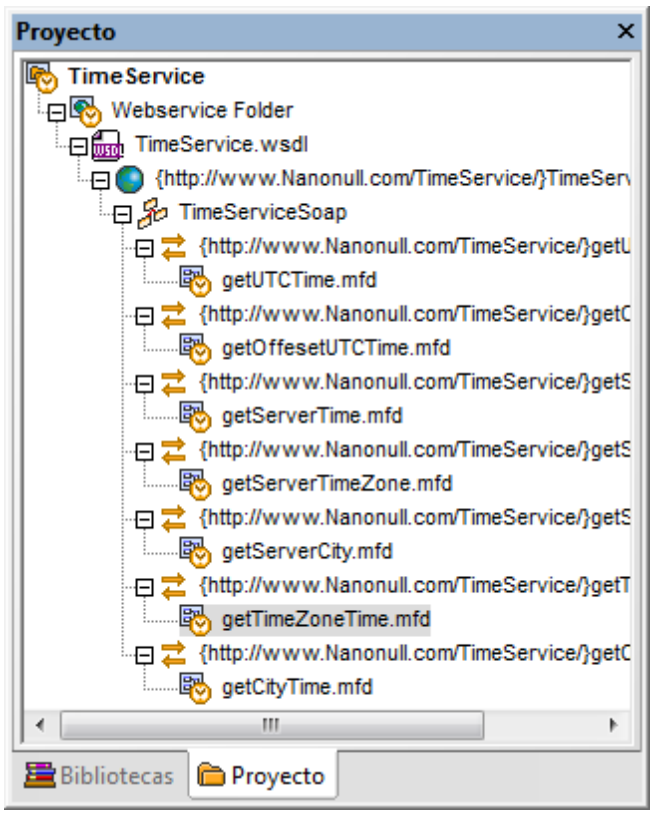
5. Seleccione la opción de menú **SOAP | Enviar solicitud al servidor** para enviar la solicitud al servidor.
   El servicio web devuelve el documento de respuesta SOAP, que contiene los registros de cuatro personas de la base de datos Access.

16.5 Elementos de error fault del servicio web

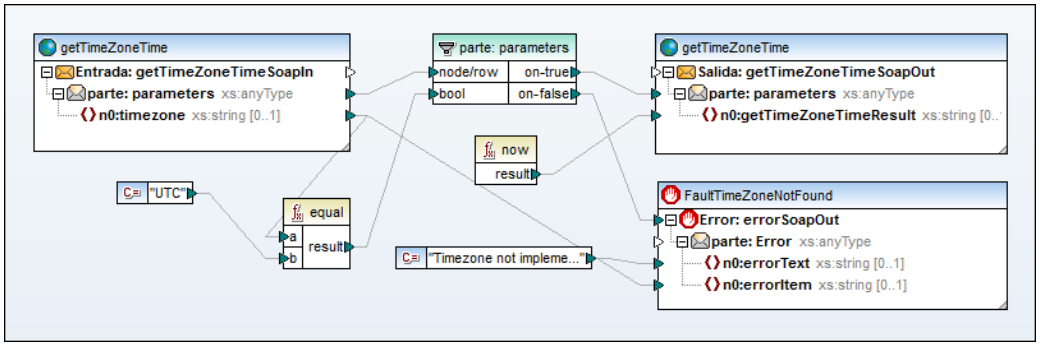
MapForce ofrece funciones para definir mensajes de error WSDL. Los archivos WSDL pueden contener elementos de error `fault` para cada operación y un atributo que contiene el mensaje de error. Para poder insertar el componente de error en la asignación el archivo WSDL debe incluir un elemento de error `fault`.

En MapForce puede definir la condición que generará un error. Cuando la condición se cumple, aparece un mensaje definido por el usuario en la ventana Mensajes y el proceso de asignación se detiene.

- Abra el archivo `Timeservice.mfp` de la carpeta `...\MapForceExamples\TimeService`
- En la carpeta Proyecto haga doble clic en la entrada del archivo `getTimeZoneTime.mfd` del proyecto `Timeservice.mfp` para ver un ejemplo de un componente de error.



Se abre una asignación que ilustra el uso de mensajes de error (WSDL 1.1 en este caso)



 © 2014 Altova GmbH

El ejemplo anterior muestra cómo se definen excepciones en una asignación. La excepción se desencadena cuando **`n0:Timezone`** no es igual a `UTC`.

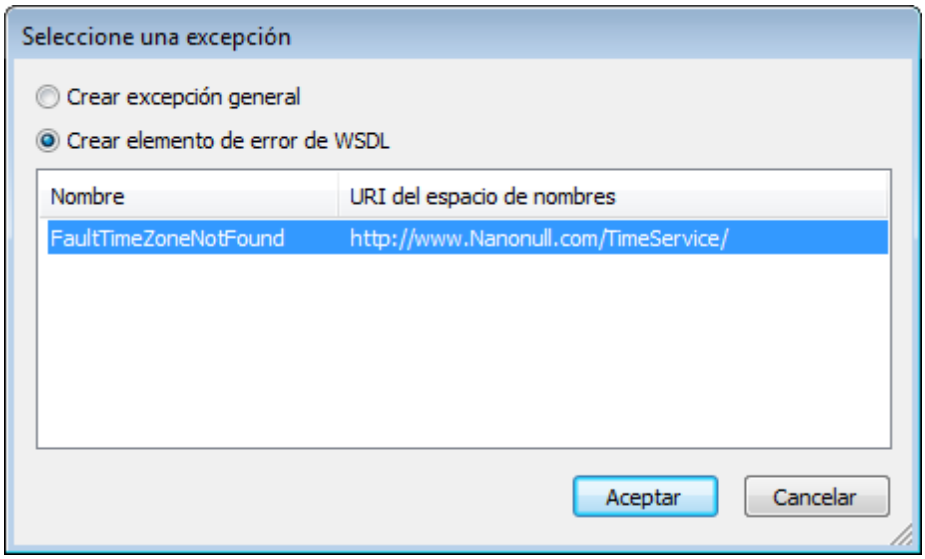
- La función **`equal`** comprueba si el uso horario es igual a `UTC` y su resultado booleano se envía al filtro **`parte: parameters`**.
- Si la condición es falsa (no igual a UTC) el parámetro **`on-false`** del filtro activa la excepción **`Fault:errorSoapOut`** y el proceso de asignación se detiene. No olvide que también puede conectar la excepción al parámetro **`on-true`** si lo necesita.)
- El mensaje `SoapFault` ofrece dos conjuntos de texto de error.

Nota: en el ejemplo anterior la posición del filtro tiene un papel importante:

- **Ambos parámetros** del filtro (**`on-true`** y **`on-false`**) deben estar conectados/asignados. Uno de ellos debe estar asignado al componente de error y el otro al componente de destino que recibe los datos de origen filtrados. De lo contrario, el componente de error no se desencadena.
- Los componentes de **excepción** y de **destino** deben estar **directamente conectados** al **filtro**. No se pueden poner funciones ni otros componentes entre el filtro y la excepción o entre el filtro y los componentes de destino.

Para insertar un error fault de servicio web:

1. Seleccione la opción de menú **Insertar | Excepción** o haga clic en el icono **Excepción** de la barra de herramientas.
2. Haga clic en el botón de opción *Crear elemento de error de WSDL*, seleccione el nombre del elemento de error en el cuadro de lista y después haga clic en **Aceptar**.



Altova MapForce 2014

Llamar a servicios web

17 Llamar a servicios web

MapForce permite **llamar a funciones de servicio web directamente** desde la asignación de datos. Esto significa que puede insertar una función de servicio web en una asignación, conectar componentes de origen y destino a la función y obtener una vista previa inmediata de la llamada a función de servicio web en el panel **Resultados**.

Recuerde que el servicio web debe estar conectado tanto al componente de origen como de destino, tal y como muestra el ejemplo del apartado Llamar al servicio web getPerson. Si falta uno de los componentes, la asignación de datos (y la validación) dará un error.

Las funciones de servicio web son compatibles con código Java y C# solamente, pero se puede obtener una vista previa de ellas con el motor de ejecución integrado.

En esta sección explicamos:

- cómo consultar un timeService usando como entrada una constante
- cómo consultar un servicio web que ofrece resultados de una base de datos

Actualmente las llamadas a servicios web desde MapForce son compatibles con estos protocolos:

SOAP 1.1 y 1.2

- RPC / encoded y Documento / Literal
- control de referencias (RPC/encoded)
- control de errores: detección de elementos de error wsdl o no wsdl

  los elementos de error wsdl se pueden asignar un componente de excepción, lo cual detiene la ejecución
  los elementos de error no wsdl aparecen en pantalla, lo cual detiene la ejecución

no SOAP

- HTTP GET: con codificación URL
- HTTP POST: con codificación URL y texto/xml

características corrientes

- autenticación (solamente autenticación básica)
- se controlan los errores devueltos por el servidor (aquellos que no son elementos de error fault, como una URL incorrecta, tiempo de espera, etc.)
- tiempo de espera

Los ejemplos de los apartados siguientes utilizan estos archivos de MapForce:

- El servicio web TimeService.wsdl de la carpeta ...\MapForceExamples\TimeService
- El servicio web TimeService2.0.wsdl de la carpeta ...\MapForceExamples \TimeserviceWsdl2
- El servicio web Query.wsdl de la carpeta ...\MapForceExamples\Tutorial

 © 2014 Altova GmbH

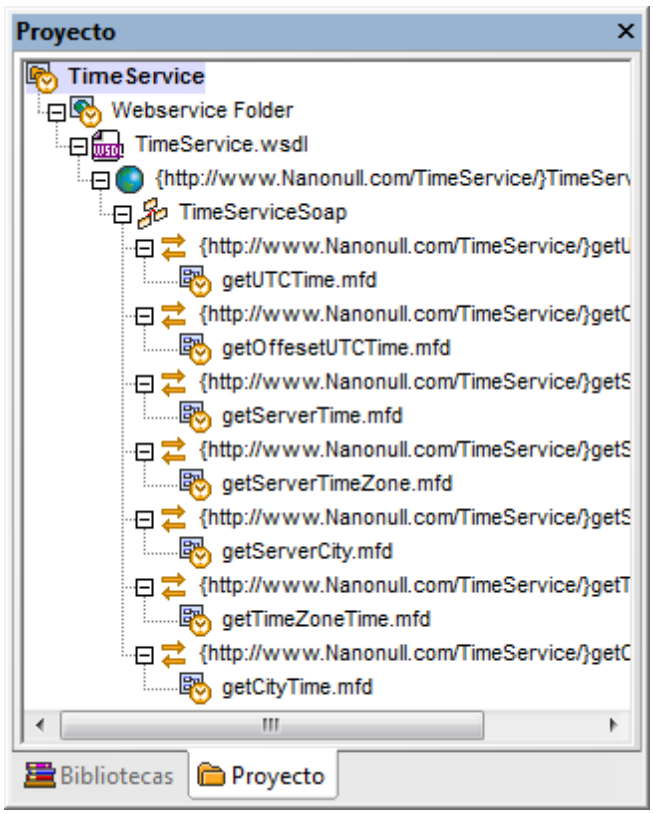
17.1 Llamar a la función de servicio web getCityTime

En este apartado explicamos cómo se llama a un servicio web implementado con MapForce, pero el servicio web se podría implementar con cualquier otra tecnología que admita un protocolo compatible.

La asignación que aparece a continuación es parte del proyecto de asignación `TimeService.mfp` de la carpeta `...\MapForceExamples\TimeService`. El archivo de proyecto `TimeService2.mfp` es compatible con WSDL 2.0.

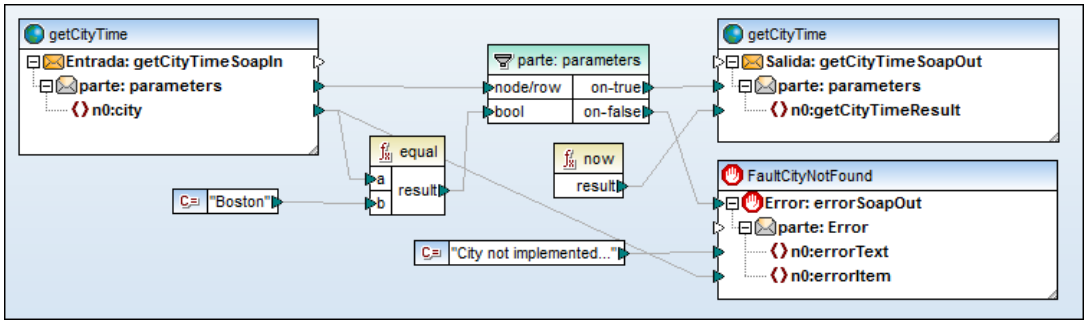
Ver las asignaciones/operaciones predefinidas en el proyecto de servicio web TimeService:

1. Seleccione la opción de menú **Archivo | Abrir** y seleccione el archivo `TimeService.mfp` de la carpeta `...\MapForceExamples\Timeservice`.



   Esto abre el archivo de proyecto **TimeService** y muestra las operaciones de servicio web y sus archivos de asignación `*.mfd` asociados. Tenga en cuenta que estos archivos de asignación se predefinieron a modo de ejemplo.

2. Haga doble clic en la entrada `getCityTime.mfd` de la ventana Proyecto.

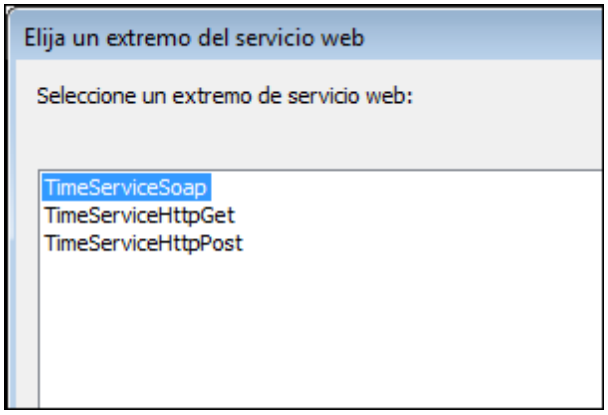


Esta asignación compara el valor dado por la constante `Boston` con el valor dado por el componente **getCityTimeSoapIn**, que usa el archivo `getCityTimeRequest.xml` como origen de datos para la vista previa.

La asignación de la imagen anterior se puede insertar como función de servicio web en un archivo de asignación estándar, sin necesidad de crear un proyecto WSDL.

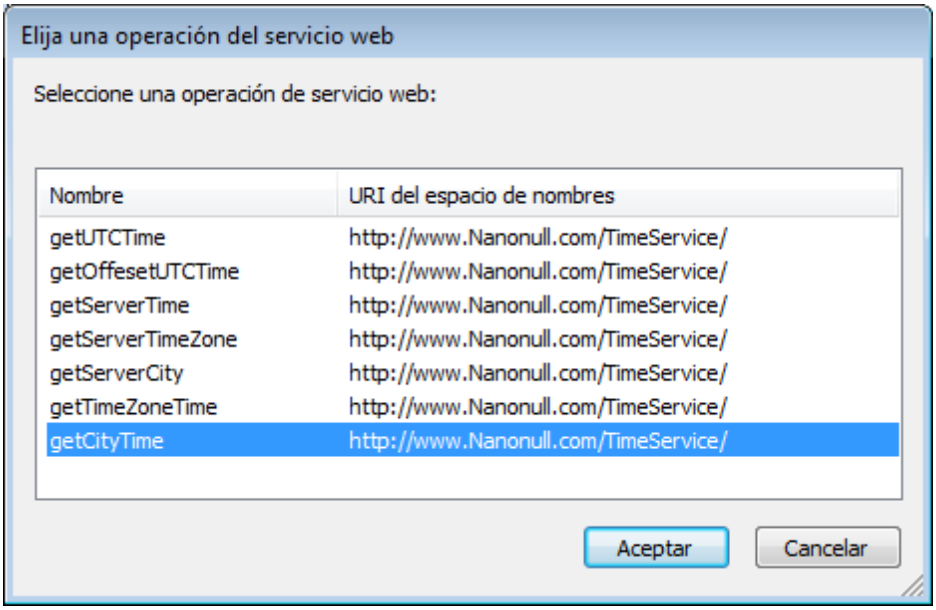
Para insertar getCityTime como función de servicio web:

1. Seleccione la opción **Archivo | Nuevo**, haga clic en la opción **Asignación** y después en **Aceptar**.
2. Seleccione la opción de menú **Insertar | Función de servicio web...** o haga clic en el icono de la barra de herramientas.
3. Haga clic en el botón **Examinar** para seleccionar el archivo de definición de servicio web WSDL. Seleccione el archivo `TimeService.wsdl` del directorio `TimeService` y haga clic en **Abrir**.
4. En el siguiente cuadro de diálogo elija un extremo del servicio web (p. ej. **`TimeServiceSoap`**) y después haga clic en **Aceptar**.

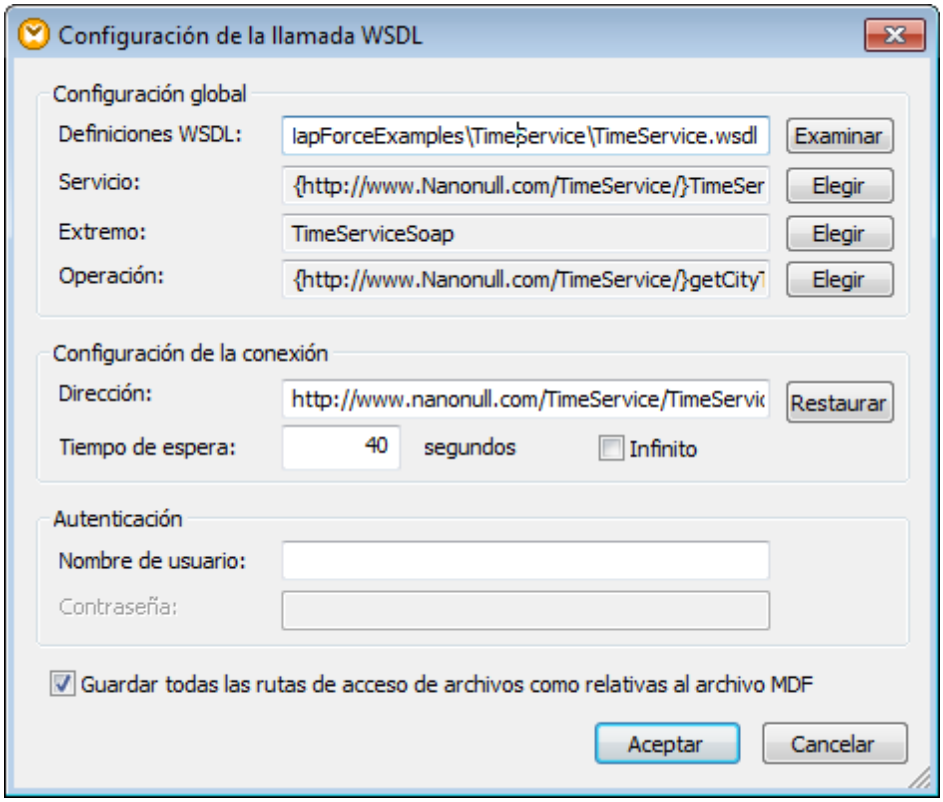


5. Elija la operación de servicio web **getCityTime** y haga clic en **Aceptar**.

 © 2014 Altova GmbH



Ahora aparece el cuadro de diálogo "Configuración de la llamada WSDL", donde puede seguir configurando la llamada.

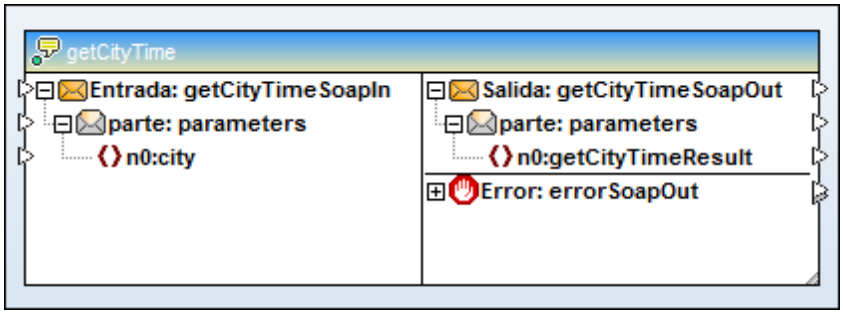


El grupo de opciones *Configuración de la conexión* muestra la URL de conexión y también permite definir el tiempo de espera de la conexión. El valor predeterminado es `40` segundos. La casilla *Infinito* envía a MapForce la instrucción de esperar un largo período de tiempo.

En el grupo de opciones *Autenticación* puede indicar el nombre de usuario y la

contraseña necesarios para acceder al servicio web.

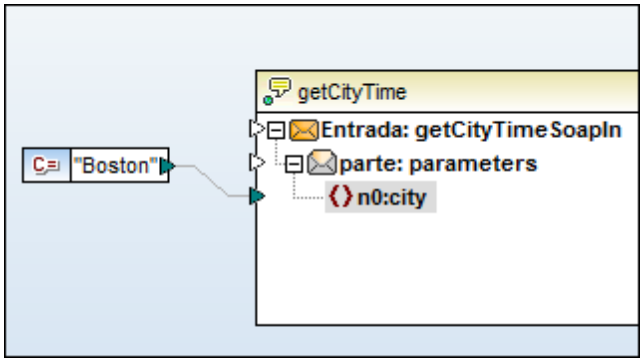
Se inserta un componente que representa a la función de servicio web **`getCityTime`**. Observe que de hecho representa a los ocho componentes presentes en el archivo `getCityTime.mfd`.



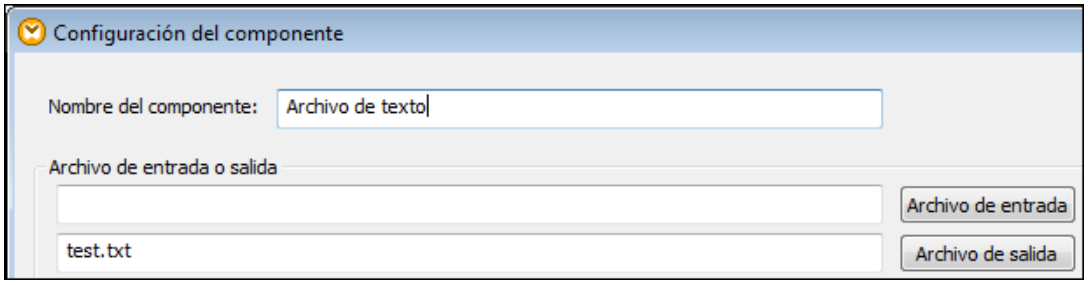
A la izquierda del componente aparecen los datos de entrada (**`SoapIn`**), mientras que a la derecha aparecen los datos de salida (**`SoapOut`**). A la derecha puede aparecer también una sección para elementos de error, si se definió algún error en el archivo WSDL.

Para definir los componentes de entrada/salida de la función WSDL:

1. Inserte el componente que aportará los datos de entrada (por ejemplo, una constante, un componente de texto o un esquema).
   En este caso inserte una constante y escriba la cadena de texto de entrada `Boston`.



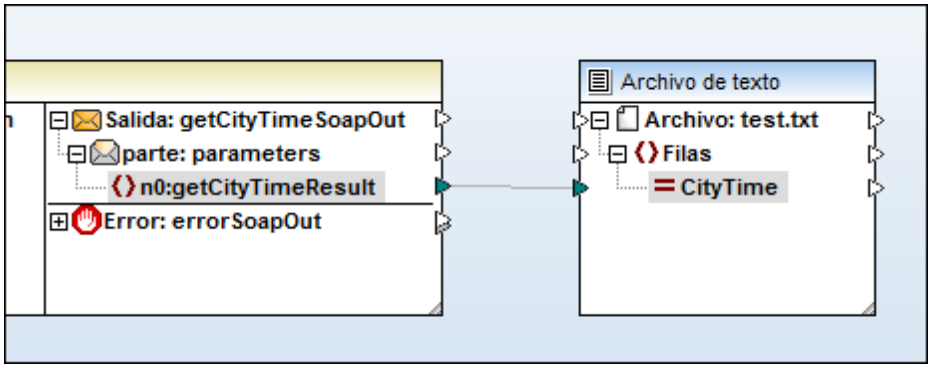
2. Conecte la constante al elemento **`n1:city`**.
3. Haga clic en **Insertar | Archivo de texto** para insertar un componente de texto. En el campo *Archivo de salida* del cuadro de diálogo "Configuración del componente" escriba el nombre del archivo de texto de salida y haga clic en **Aceptar**.



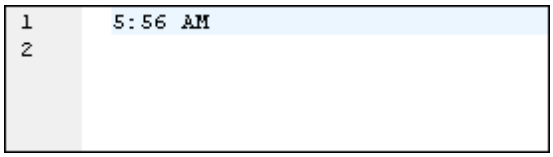
4. Conecte el elemento **`n0:getCityTimeResult`** de la función WSDL con el elemento

 © 2014 Altova GmbH

Campo1 del componente de texto.



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



En el panel **Resultados** aparece la hora actual en Boston.

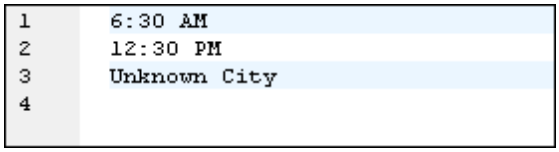
Nota: el valor de entrada de la función de servicio web siempre tiene prioridad sobre el origen de datos de la asignación original. Por ejemplo, el valor `Boston` de la constante tiene prioridad sobre el archivo de origen de datos `getCityTimeRequest.xml` de la asignación original.

Si el archivo de entrada incluye varias entradas, entonces el panel de **Resultados** mostrará los resultados de cada elemento en una línea distinta.

Por ejemplo, este archivo de texto contiene tres entradas:

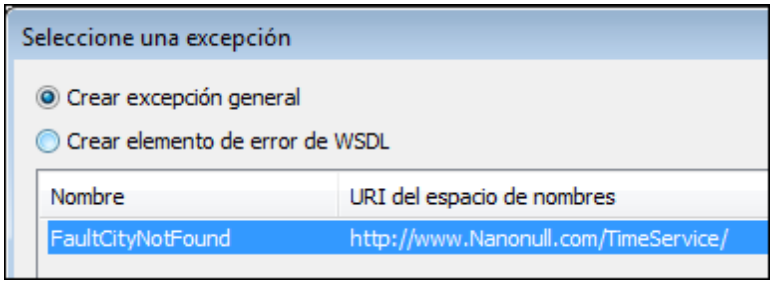
Cities list
string
Boston
Vienna
New York

Resultado en el panel **Resultados**:

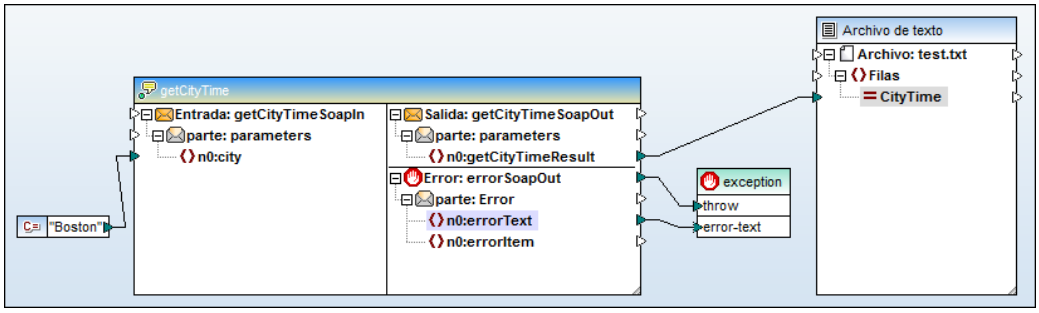


Para realizar asignaciones de datos a elementos de error fault de servicios web:

1. Seleccione **Insertar | Excepción** o haga clic en el icono **Excepción** .
2. Compruebe que está activo el botón de opción *Crear excepción general* y haga clic en **Aceptar** para continuar.



Se inserta un componente de excepción en la asignación.

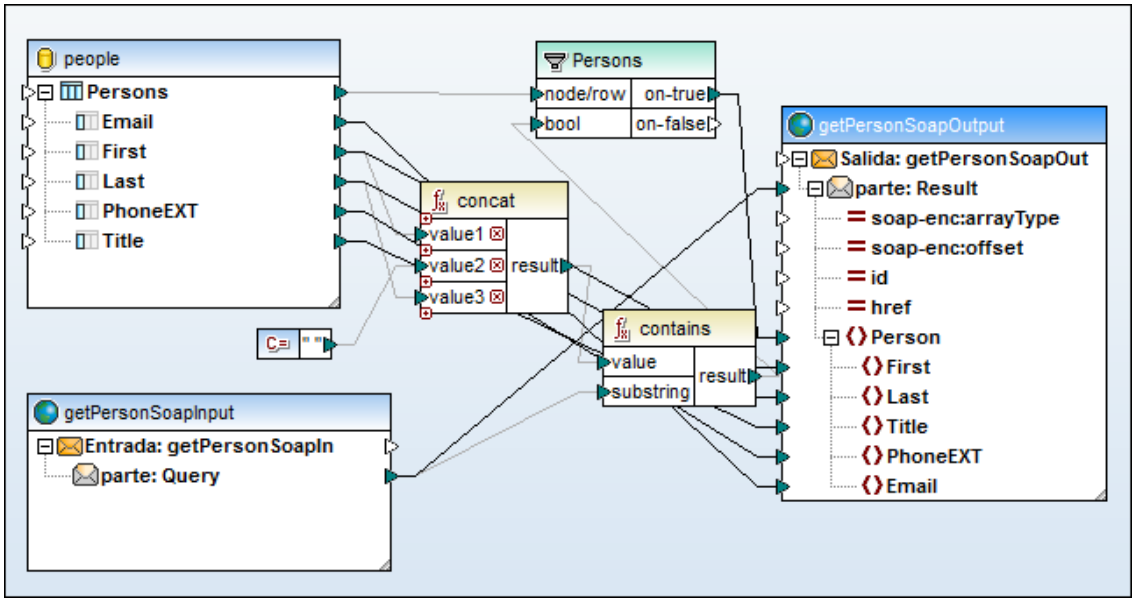


3. Asigne el elemento **`Error:`** al parámetro **`throw`** de la excepción.
4. Asigne el elemento **`n0:errorText`** al parámetro **`error-text`** de la excepción.

 © 2014 Altova GmbH

17.2 Llamar al servicio web getPerson

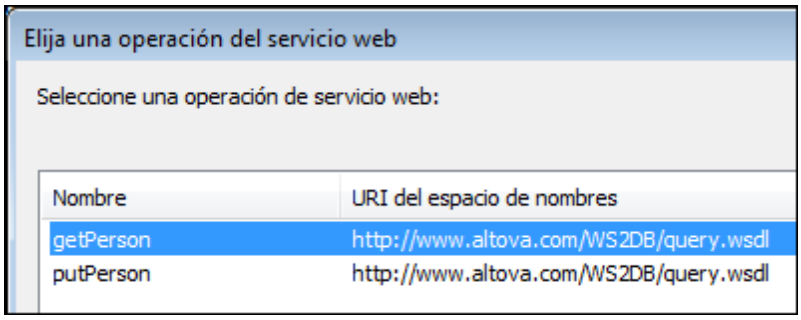
La asignación utilizada como ejemplo en este apartado forma parte del proyecto de asignación `Query Person Database.mfp`, disponible en la carpeta `...\Tutorial`. En el apartado Crear proyectos de servicio web a partir de archivos WSDL encontrará una descripción de este servicio web e instrucciones para generar código para él.



En este apartado explicamos cómo insertar una función de servicio web para conseguir los resultados descritos en el apartado Crear proyectos de servicio web a partir de archivos WSDL. Para este ejemplo asumimos que el servicio web se compiló e implementó en el servidor web.

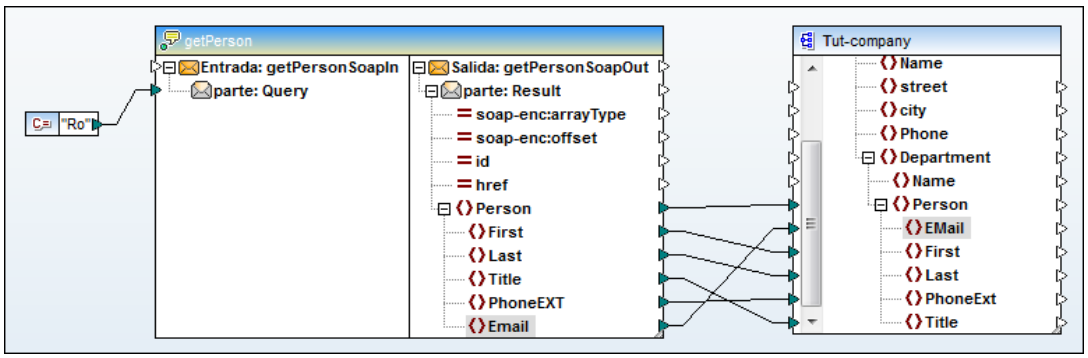
Para insertar la operación getPerson como función de servicio web:

1. Seleccione **Archivo | Nuevo**, haga clic en el icono **Asignación** y después en el botón **Aceptar**.
2. Seleccione la opción de menú **Insertar | Función de servicio web...** o haga clic en el icono de la barra de herramientas.
3. Haga clic en el botón **Examinar** y seleccione el archivo WSDL `query.wsdl` de la carpeta `...\Tutorial`. Después haga clic en el botón **Abrir**.
4. En el cuadro de diálogo "Elija una operación del servicio web" seleccione la operación **getPerson** y haga clic en **Aceptar.**



5. Inserte una constante cuyo valor sea la cadena de texto que desea usar para consultar la base de datos (p. ej. `Ro`).

6. Inserte el archivo de esquema XML que debe contener los datos resultantes (p. ej. el esquema `Tut-company.xsd` de la carpeta `...\Tutorial`) y cree las conexiones necesarias.



7. Abra el panel **Resultados** para ver los resultados de la asignación.

 © 2014 Altova GmbH

Altova MapForce 2014

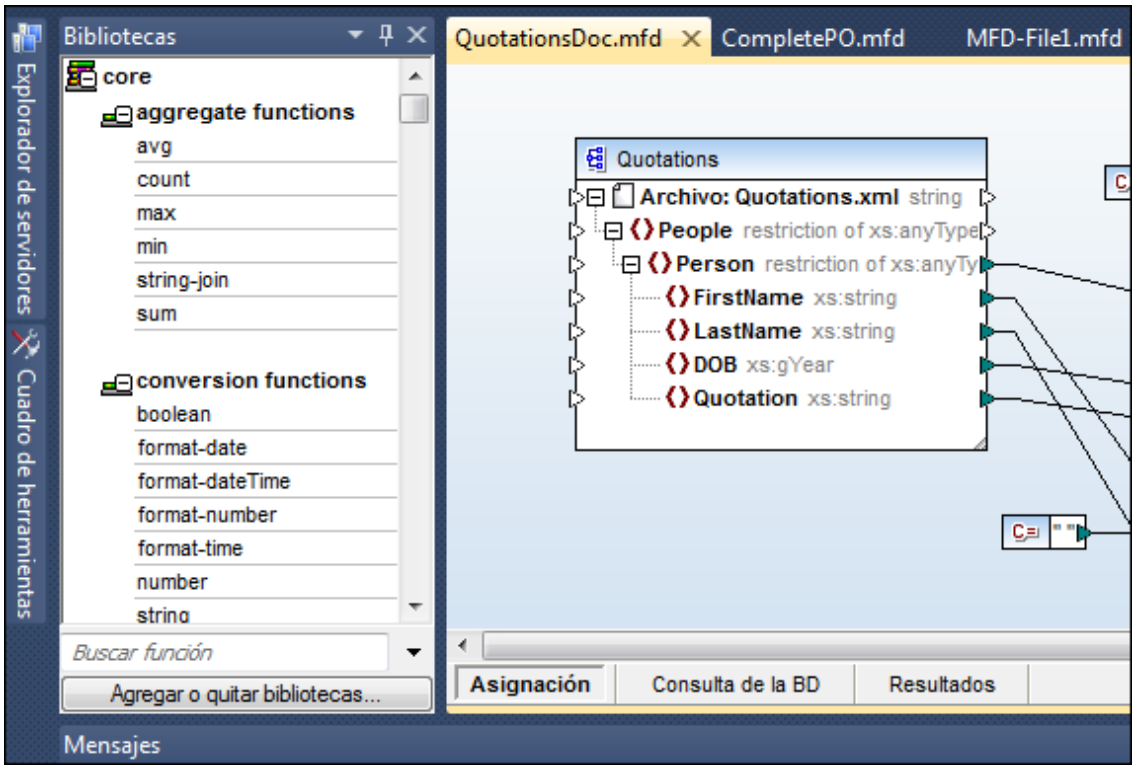
Complemento de MapForce para MS Visual Studio

18 Complemento de MapForce para MS Visual Studio

MapForce2014 se puede integrar en las versiones 2005/2008/2010/2012/2013 del entorno IDE Microsoft Visual Studio. Esto hace posible incorporar las avanzadas funciones de asignación e integración de datos de MapForce al avanzado entorno de desarrollo Visual Studio.

Para ello deberá:

- instalar Microsoft Visual Studio,
- instalar MapForce (Enterprise Edition o Professional Edition),
- descargar y ejecutar el paquete de integración de **MapForce**. Este paquete está disponible en la página de descargas de MapForce en www.altova.com.



Una vez instalado el paquete de integración podrá usar MapForce en Visual Studio.

Cómo habilitar el complemento

Puede que el complemento de MapForce no se habilitara automáticamente durante el proceso de instalación.

Para habilitar el complemento:

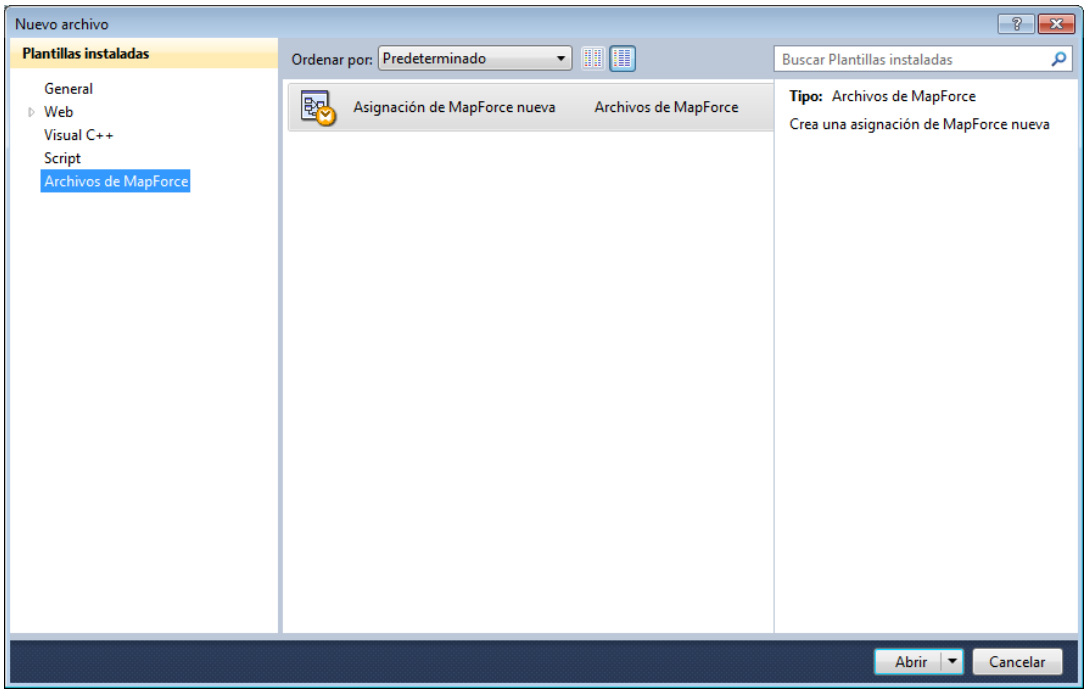
1. Navegue al directorio donde se instaló el ejecutable de Visual Studio (p. ej. `c:\Archivos de programa\Microsoft Visual Studio 8\Common7\IDE`)
2. Escriba este comando en la línea de comandos **devenv.exe /setup**.
3. Espere a que termine el proceso antes de iniciar la aplicación dentro de Visual Studio.

 © 2014 Altova GmbH

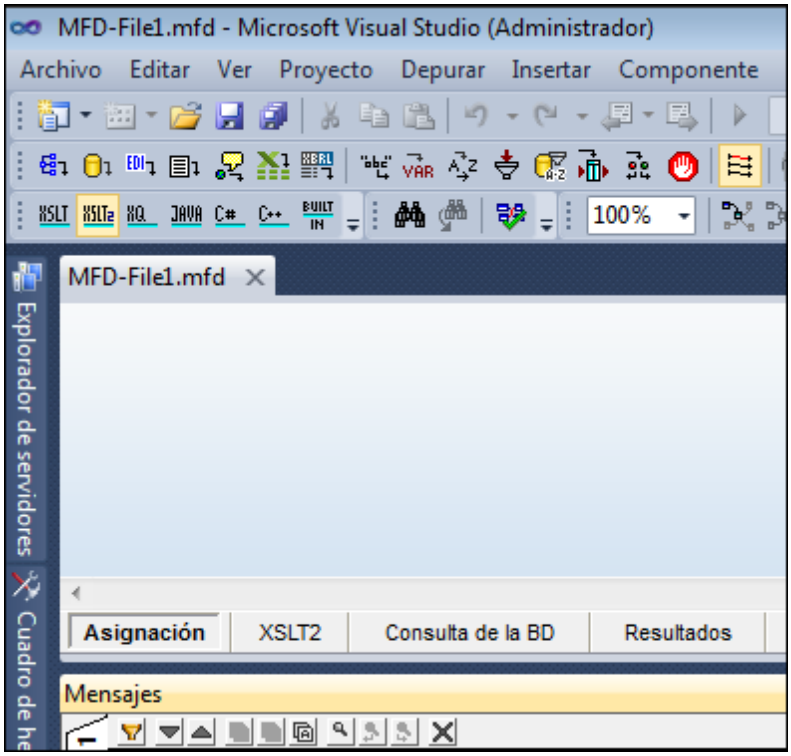
18.1 Abrir archivos de MapForce en Visual Studio

Para abrir un archivo de asignación nuevo de MapForce:

1. Seleccione la opción de menú **Archivo | Nuevo**.
2. En el cuadro *Plantillas instaladas* haga clic en la entrada **Archivos de MapForce**.

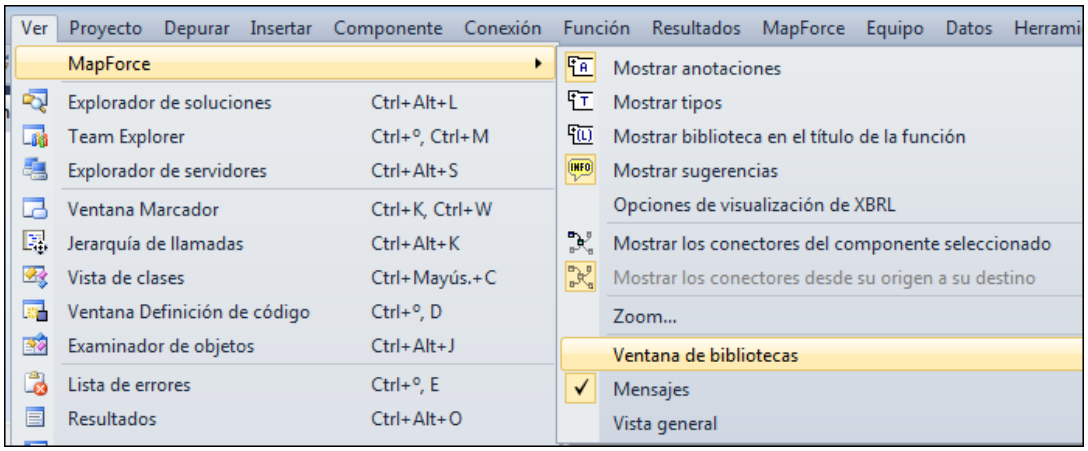


3. En el cuadro de la derecha haga doble clic en **Asignación de MapForce nueva**.
   Se abre un archivo de asignación en blanco.



Para habilitar la ventana Bibliotecas:

1. Seleccione la opción de menú **Ver | MapForce | Ventana de bibliotecas**.

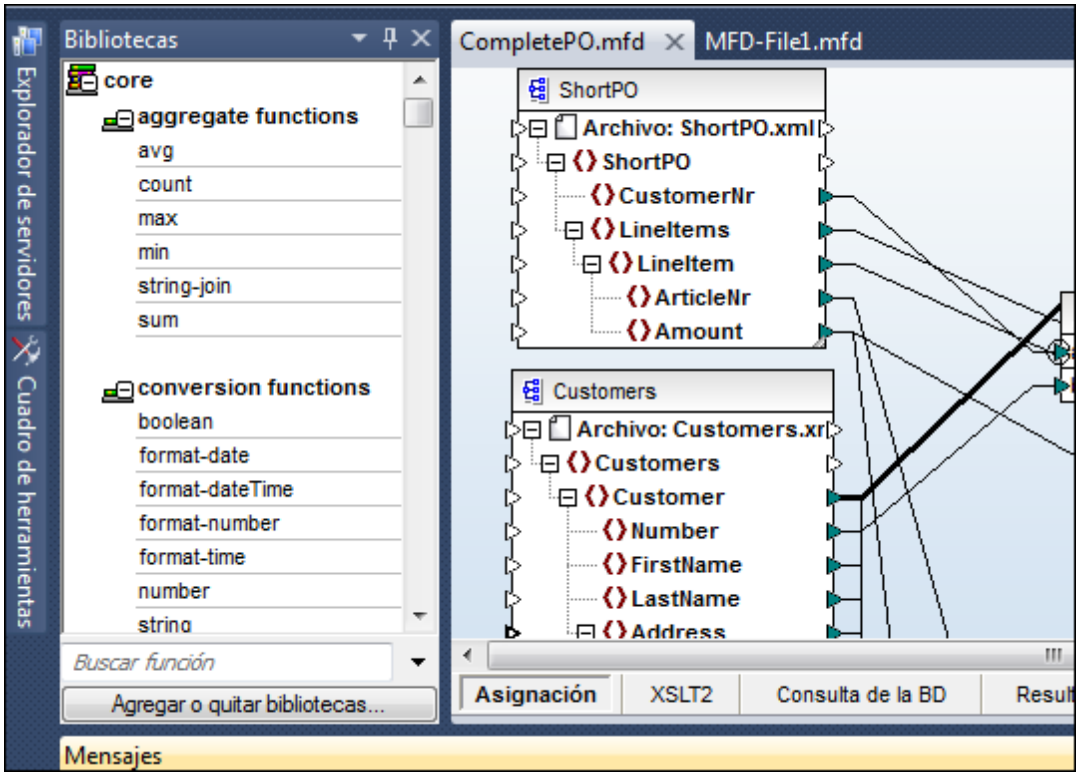


2. Acople la ventana flotante en la posición deseada (p. ej. en el lateral izquierdo).

Para abrir un archivo de ejemplo de MapForce:

1. Seleccione la opción de menú **Archivo | Abrir**, navegue hasta la carpeta **... \MapForceExamples** y abra un archivo de MapForce. Para la imagen siguiente se utilizó el archivo `CompletePO.mfd`.

 © 2014 Altova GmbH


Explorador de servidores
Cuadro de herramientas
Bibliotecas
core
aggregate functions
avg
count
max
min
string-join
sum
conversion functions
boolean
format-date
format-dateTime
format-number
format-time
number
string
Buscar función
Agregar o quitar bibliotecas...
CompletePO.mfd
MFD-File1.mfd
ShortPO
Archivo: ShortPO.xml
ShortPO
CustomerNr
LineItems
LineItem
ArticleNr
Amount
Customers
Archivo: Customers.xr
Customers
Customer
Number
FirstName
LastName
Address
Asignación
XSLT2
Consulta de la BD
Mensajes

18.2 Diferencias en MapForce para Visual Studio

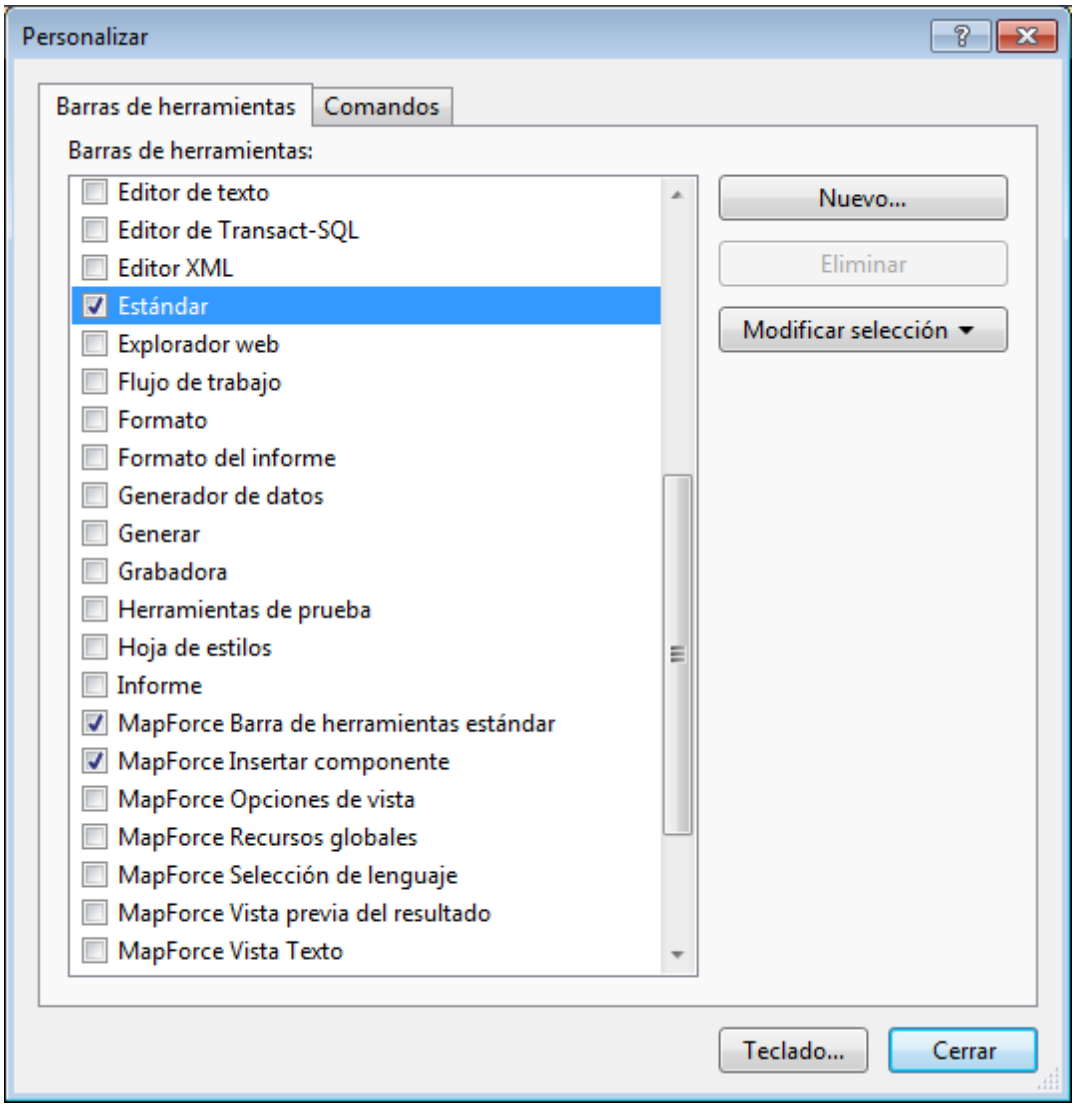
Los complementos de MapForce Enterprise y Professional Edition se integran en todas las versiones de MS Visual Studio de la misma manera. No obstante, Visual Studio no admite los archivos de proyecto de MapForce.

Diferencias en las funciones del complemento de MapForce para Visual Studio:

- Comandos **Deshacer** y **Rehacer** del menú **Edición**

Los comandos **Deshacer** y **Rehacer** afectan a todas las acciones (copiar, pegar, etc.) realizadas en el entorno de desarrollo, incluidas todas las acciones de MapForce.

- Pestañas *Barras de herramientas* y *Comandos* del cuadro de diálogo "Personalizar" (**Herramientas | Personalizar**).



Estas pestañas incluyen tanto comandos de Visual Studio como de MapForce.

- Menú **Ver**

Este menú contiene el submenú **MapForce**, con el que podrá habilitar o deshabilitar los paneles

 © 2014 Altova GmbH

de herramientas de MapForce. Además ofrece un comando para configurar la vista de MapForce.

- Menú **Ayuda**

Este menú contiene el submenú **Ayuda de MapForce**, con el que puede acceder a la ayuda en pantalla de MapForce. También contiene enlaces al centro de soporte técnico y al centro de descargas del sitio web de Altova.

Características de MapForce incompatibles con Visual Studio

En el complemento de MapForce para Visual Studio no está disponible la ventana Proyecto ni el menú **Proyecto**. Esto significa que los archivos de proyecto de MapForce (y los archivos de proyecto WSDL) no se pueden abrir en la edición para Visual Studio.

Altova MapForce 2014

Complemento de MapForce para Eclipse

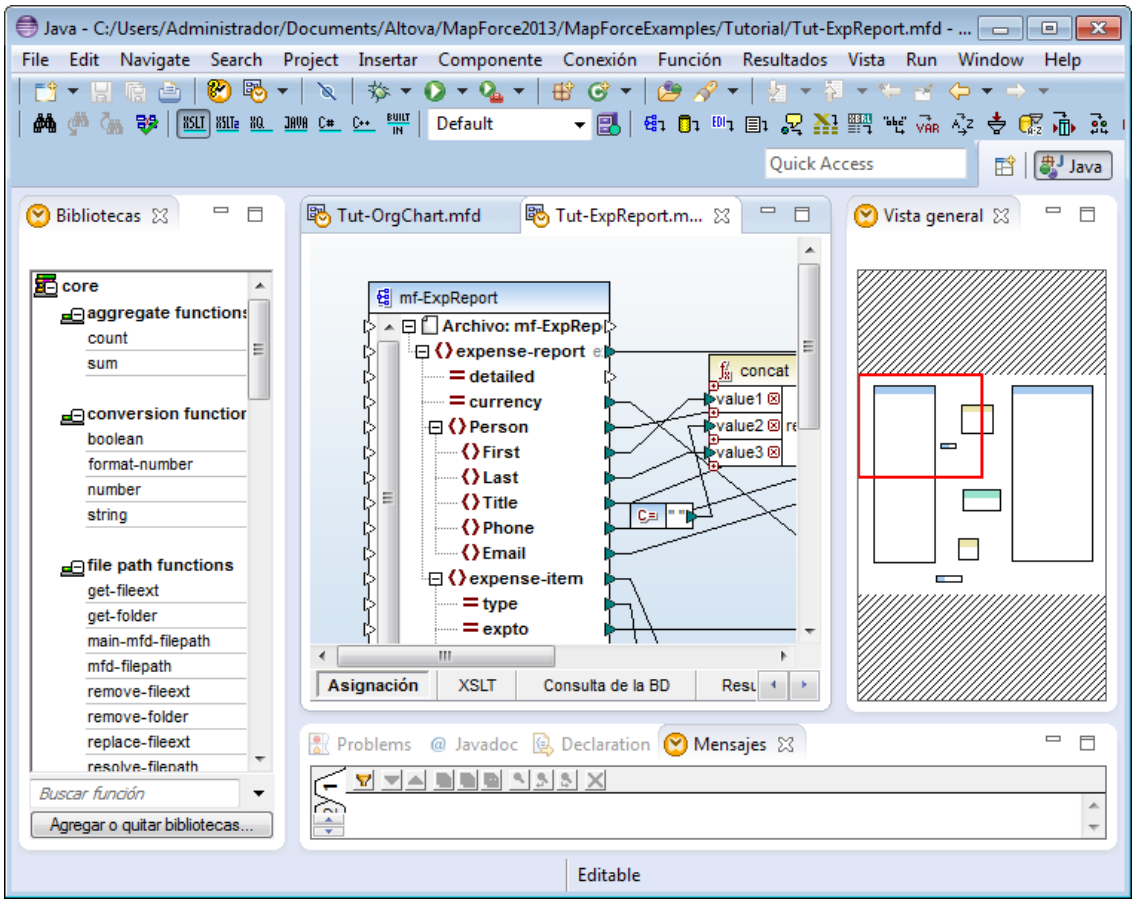
19 Complemento de MapForce para Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto que incorpora varios tipos de aplicaciones en forma de complementos.

Gracias al complemento de MapForce para Eclipse puede usar todas las funciones de MapForce Enterprise y Professional Edition desde la plataforma Eclipse 3.8 / 4.2 / 4.3.

El complemento de MapForce para Eclipse ofrece estas funciones y características:

- una herramienta de asignación visual de datos totalmente funcional para proyectos avanzados de integración de datos.
- generación de código en diferentes lenguajes de programación.
- sistema de ayuda en pantalla bajo el comando **Ayuda | MapForce| Contenido**.



 © 2014 Altova GmbH

19.1 Instalar el complemento de MapForce para Eclipse

Antes de instalar el complemento de MapForce para Eclipse, asegúrese de que estas aplicaciones están instaladas:

- MapForce Enterprise o Professional Edition.
- Java SE Runtime Environment 5.0 (JRE 5.0) o superior, que es necesario para que funcione Eclipse. Consulte el sitio web de Eclipse para obtener más información. Instale la versión de 32 o 64 bits dependiendo de la versión de MapForce que tenga instalada.
- Eclipse Platform 3.8 / 4.2 / 4.3. Instale la versión de 32 o 64 bits dependiendo de la versión de MapForce que tenga instalada.

Después de instalar todas estas aplicaciones puede instalar el complemento de MapForce para Eclipse que incluye el paquete de integración de MapForce (*ver más abajo*).

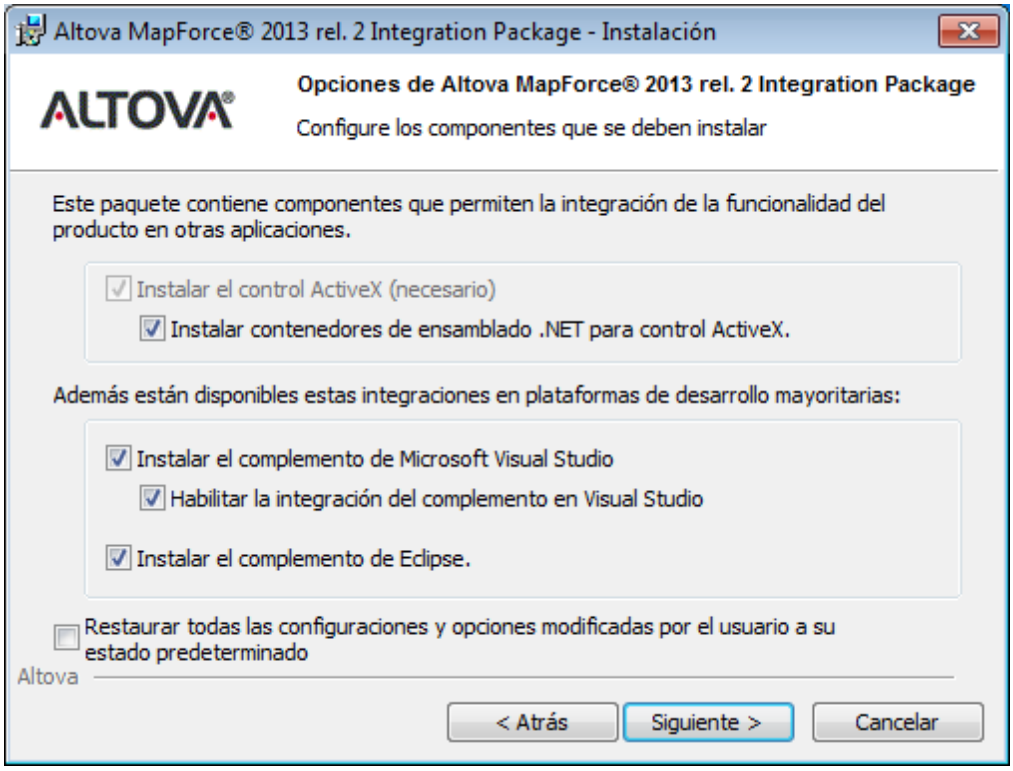
Nota sobre JRE
Si al abrir un documento en Eclipse recibe este mensaje de error:

```
java.lang.UnsupportedClassVersionError: com/altova/....
(Unsupported major.minor version 49.0)
```

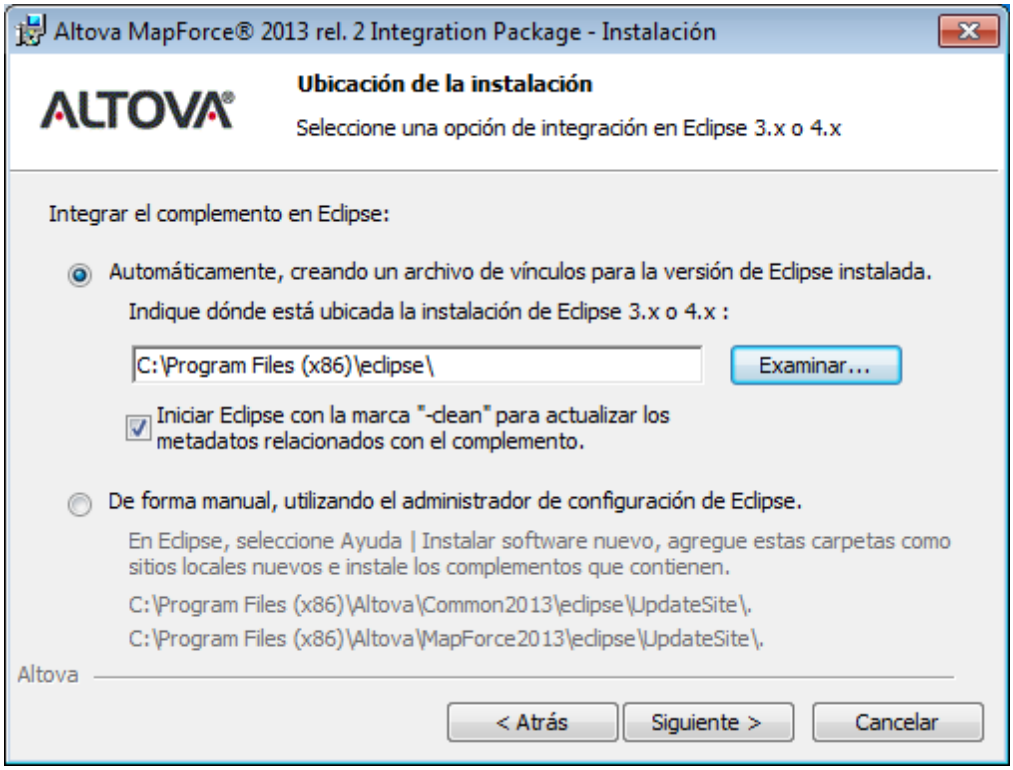
esto significa que Eclipse está usando una versión antigua de JRE presente en su equipo. Puesto que Eclipse usa la variable de entorno `PATH` para encontrar el ejecutable `javaw.exe`, el problema puede resolverse corrigiendo la variable de entorno PATH de modo que encuentre primero la versión más reciente de JRE. Otra opción es iniciar Eclipse con el parámetro de la línea de comandos `-vm`, que aporta la ruta de acceso a un ejecutable `javaw.exe` de la versión 5.0 o superior.

Paquete de integración de MapForce
El complemento de MapForce para Eclipse está en el paquete de integración de MapForce y se instala durante la instalación del paquete de integración de MapForce. Estas son las instrucciones de instalación:

1. Compruebe que tiene instalado MapForce (Enterprise o Professional Edition), JRE y Eclipse (*ver más arriba*).
2. En la página de descargas de componentes del sitio web de Altova descargue e instale el paquete de integración de MapForce. Durante la instalación hay dos pasos muy importantes que se describen en los puntos 3 y 4.
3. Durante la instalación del paquete de integración de MapForce aparece una pantalla que le pregunta si desea instalar el complemento de MapForce para Eclipse (*imagen siguiente*). Marque la casilla y haga clic en **Siguiente**.

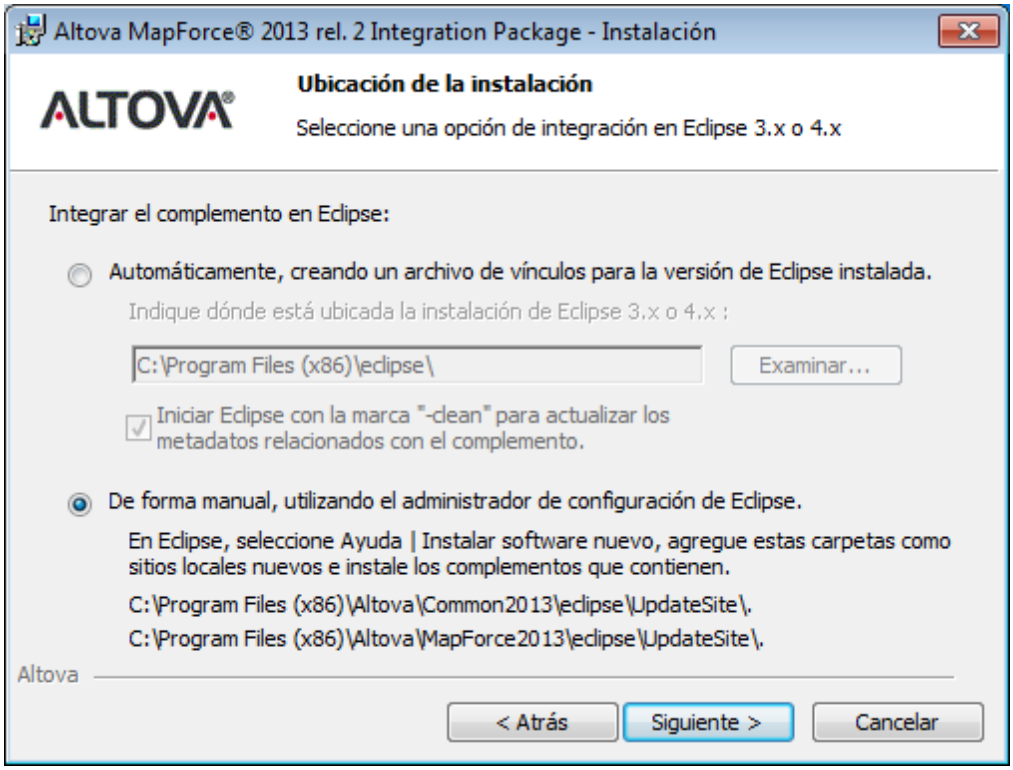


4. En la siguiente pantalla (*imagen siguiente*) puede indicar si el asistente para la instalación debe integrar el complemento de MapForce para Eclipse durante la instalación (opción *Automáticamente*) o si prefiere integrarlo de forma manual más adelante.



 © 2014 Altova GmbH

Recomendamos que elija la opción *Automáticamente* para que el asistente para la instalación se encargue de integrar el complemento. Seleccione la opción *Automáticamente* y después busque la carpeta donde se encuentra el ejecutable de Eclipse (`eclipse.exe`). Después haga clic en **Siguiente**. Si prefiere integrar el complemento de MapForce para Eclipse de forma manual, seleccione la opción *De forma manual* (*imagen siguiente*). Más abajo explicamos cómo se integra el complemento de forma manual.

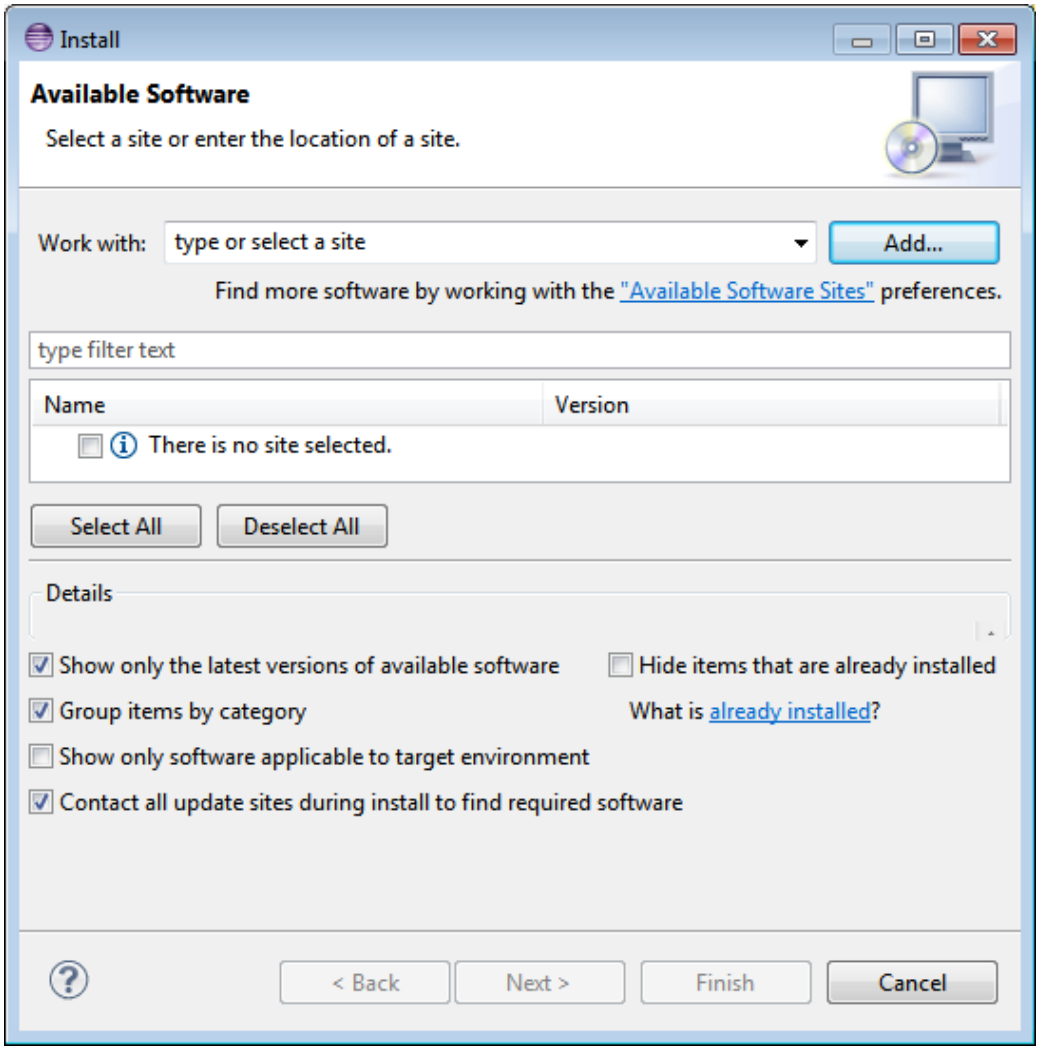


5. Finalice la instalación. Si seleccionó la integración automática, el complemento de MapForce para Eclipse se integrará en Eclipse y estará disponible la próxima vez que inicie Eclipse.

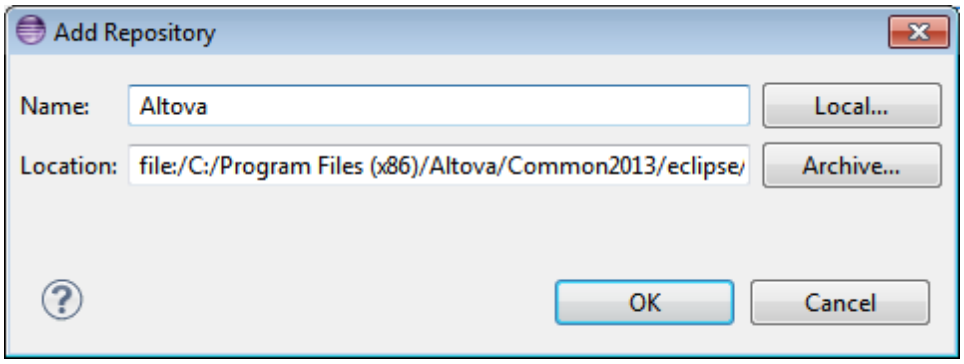
Integrar el complemento de MapForce en Eclipse de forma manual

Para integrar el complemento de MapForce en Eclipse de forma manual:

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).



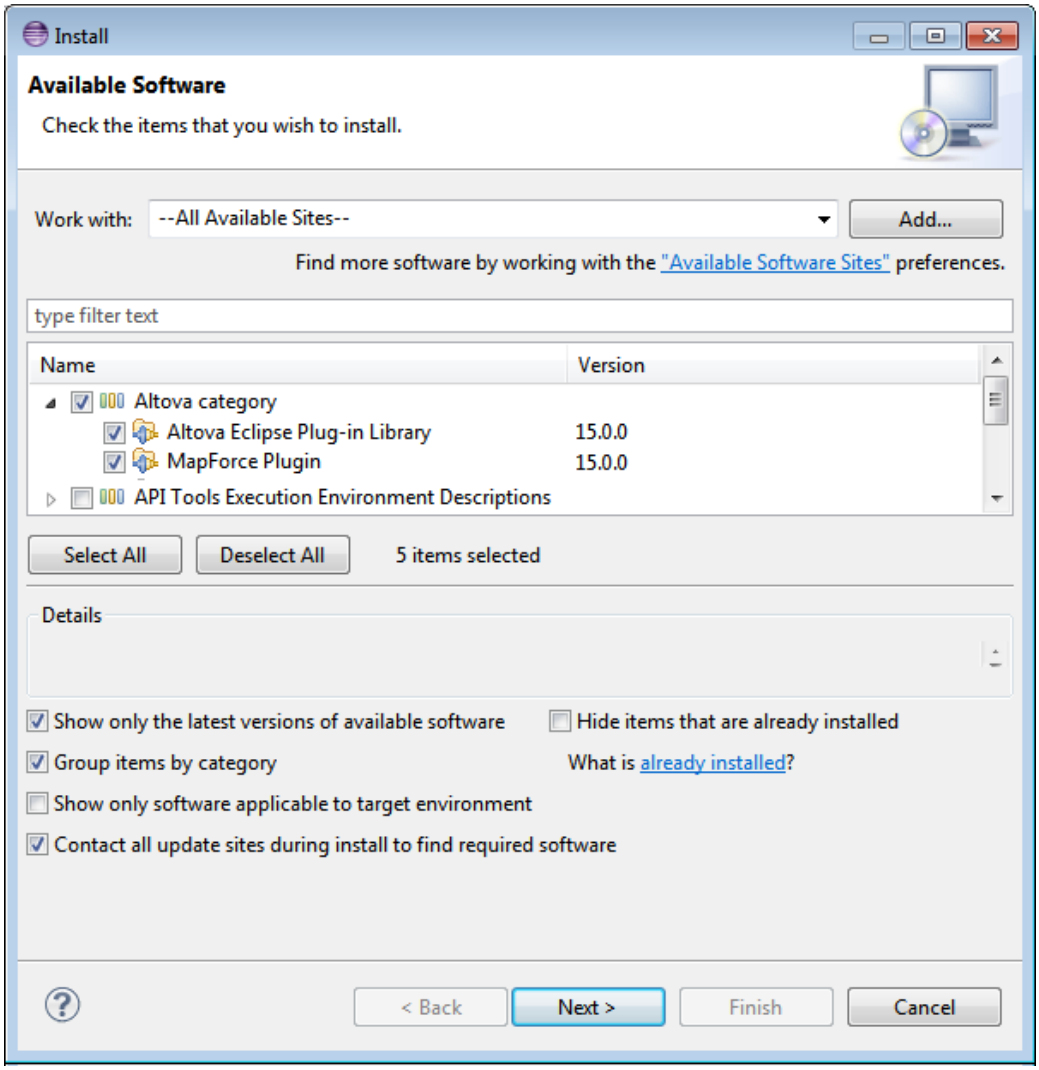
3. En el cuadro de diálogo que aparece, llamado "Add Repository" (Agregar repositorio, *imagen siguiente*) haga clic en el botón **Local**.
4. Busque la carpeta `c:\Archivos de programa\Altova\Common2014\eclipse\UpdateSite` y selecciónela. Escriba un nombre para el sitio (p. ej. Altova) y haga clic en **OK**.



5. Repita los pasos 2, 3 y 4 pero esta vez seleccione la carpeta `c:\Archivos de programa\Altova\XMLSpy2014\eclipse\UpdateSite` y escriba un nombre como "Altova MapForce".

 © 2014 Altova GmbH

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** (Siguiente).



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** (Siguiente) 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). Para terminar haga clic en el botón **Finish** (Finalizar).

Si tiene problemas con el complemento (por ejemplo, si faltan iconos), inicie Eclipse desde la línea de comandos con la marca `-clean`.

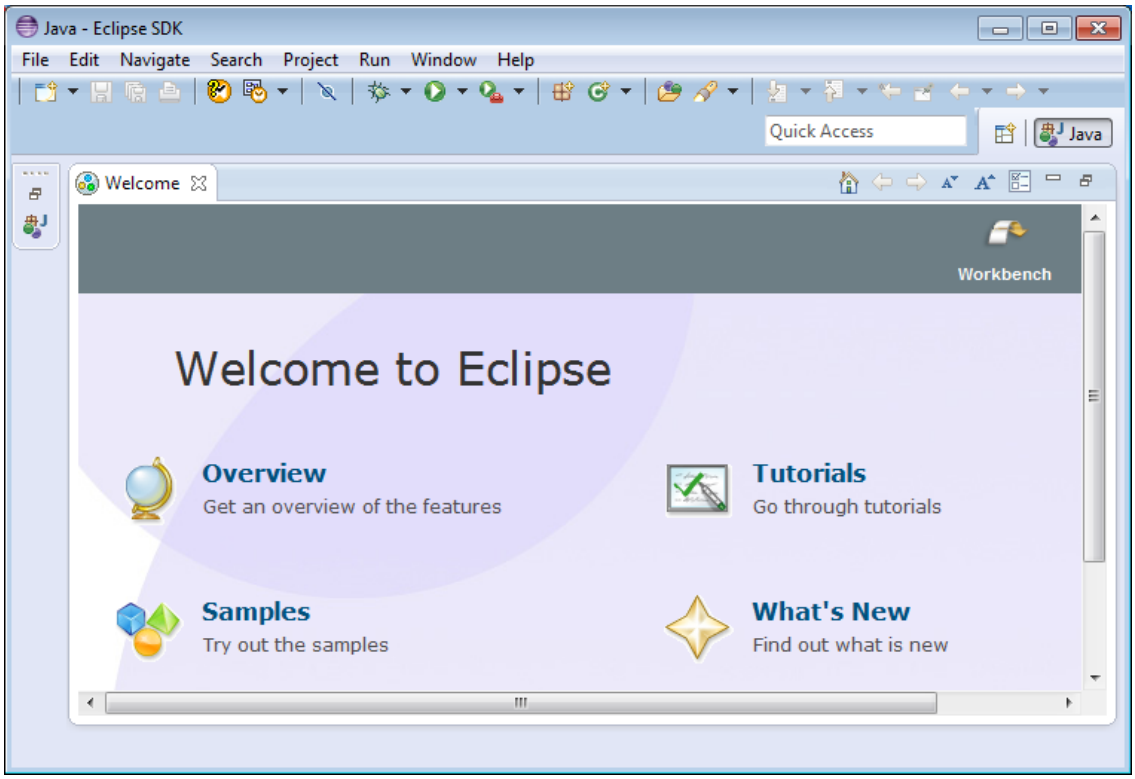
Versión instalada actualmente

Para comprobar qué versión tiene instalada del complemento de MapForce para Eclipse, en Eclipse seleccione la opción de menú **Help | About Eclipse**. Después seleccione el icono de MapForce.

© 2014 Altova GmbH

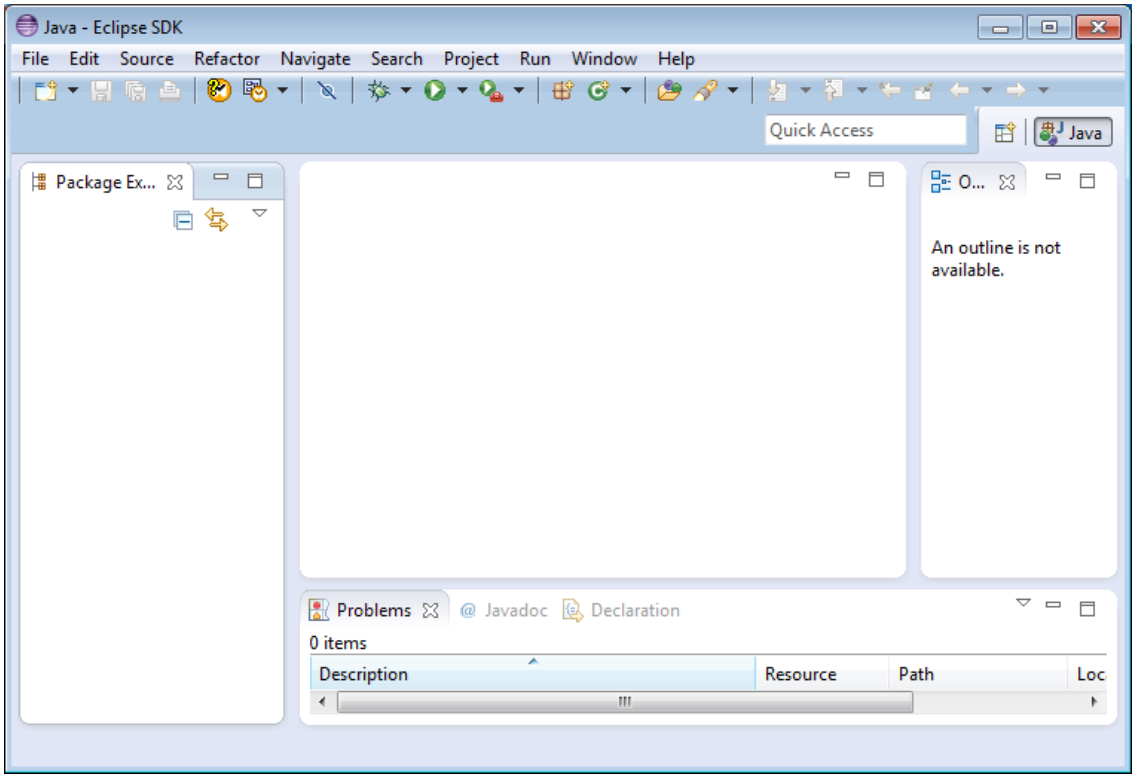
19.2 Iniciar Eclipse y usar el complemento de MapForce

En la pantalla de inicio de Eclipse haga clic en el icono **Workbench** (en la esquina superior derecha) para abrir una nueva área de trabajo. Aparece una ventana vacía de MapForce en Eclipse.



Si no puede ver esta pantalla inicial, seleccione **Help | Welcome** (*Ayuda* | *Pantalla de inicio*).

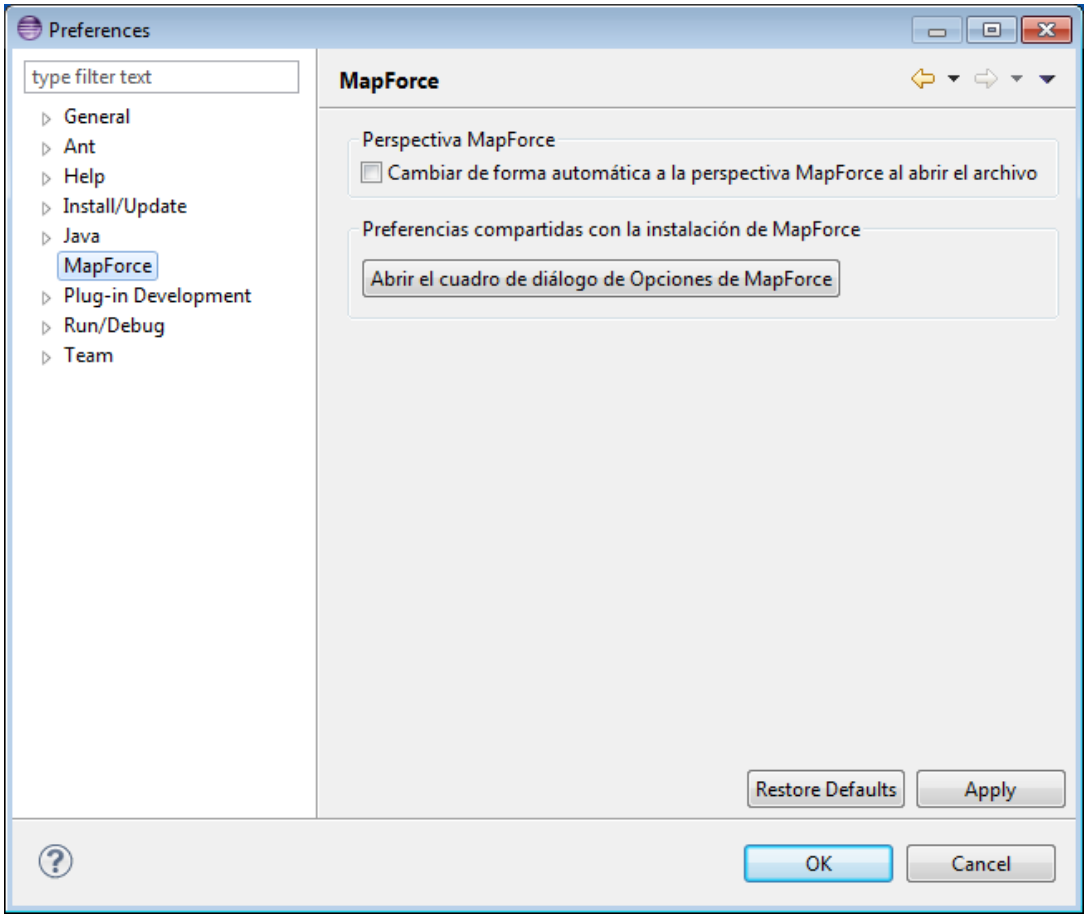
Iniciar Eclipse y usar el complemento de MapForce:
Tras hacer clic en el icono **Workbench** aparece la nueva área de trabajo de Eclipse en blanco.



Propiedades de MapForce:

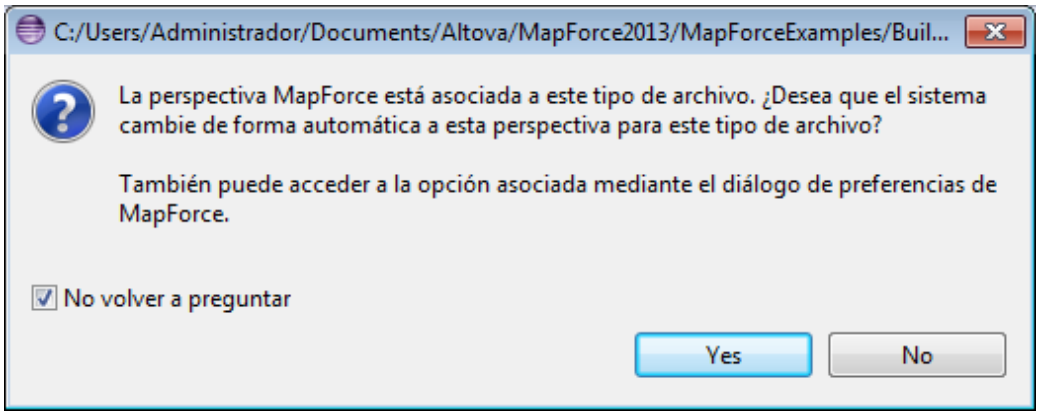
1. Seleccione la opción de menú **Window | Preferences** (*Ventana* | *Preferencias*) y haga clic en la entrada **MapForce**.
2. Marque la casilla *Cambiar de forma automática a la perspectiva MapForce al abrir el archivo*.

 © 2014 Altova GmbH



Haga clic en el botón **Abrir el cuadro de diálogo de Opciones de MapForce** para definir las opciones de configuración de MapForce (de sus bibliotecas, de la función de generación de código, etc.)

Al hacer doble clic en un archivo de asignación de MapForce (`*.mfd`) aparece un aviso informando de que la perspectiva MapForce está asociada a este tipo de archivo y le pregunta si desea que Eclipse cambie automáticamente a MapForce a partir de ahora. Esta configuración se puede cambiar en el cuadro de diálogo "Preferences" (*imagen anterior*).

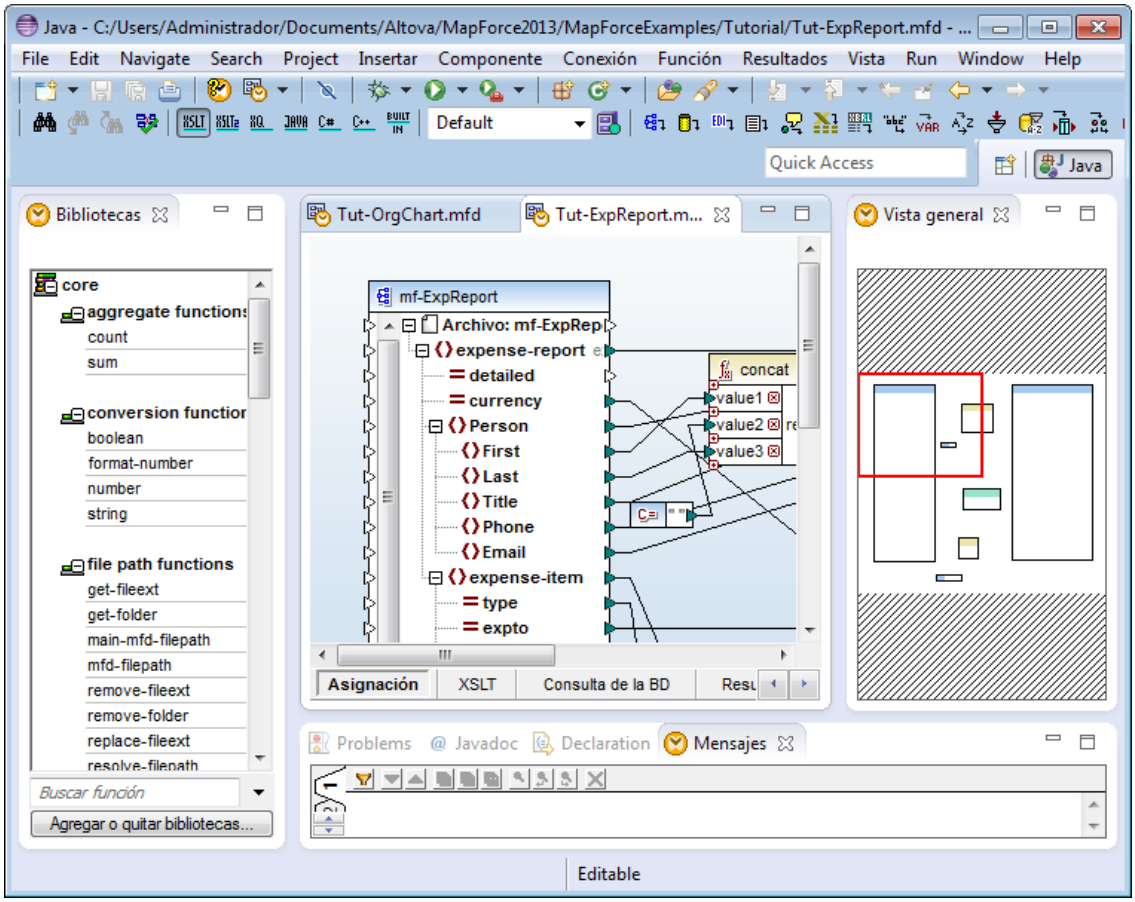


19.3 MapForce: editor, vista y perspectivas

Si activó la opción *Cambiar de forma automática a la perspectiva MapForce al abrir el archivo* en el cuadro de diálogo **Window | Preferences** (*Ventana* | *Preferencias*), la perspectiva MapForce se establece automáticamente. También puede habilitar la perspectiva MapForce siguiendo las instrucciones de este apartado.

Para habilitar la perspectiva MapForce en Eclipse:

- Seleccione la opción de menú **Window | Open perspective | Other | MapForce** (*Ventana* | *Abrir perspectiva* | *Otros*).



Ahora aparecen en el entorno Ecilpse todas las ventanas de MapForce:

- **Ventana Bibliotecas:** ofrece funciones de procesamiento de datos (predefinidas o definidas por el usuario).
- **Ventana Mensajes:** muestra mensajes de validación, error y advertencia.
- **Ventana Vista general:** muestra una visión de conjunto del diseño.

El panel del editor puede diseñar la asignación y ver una vista previa de sus resultados. Está formada por estos paneles:

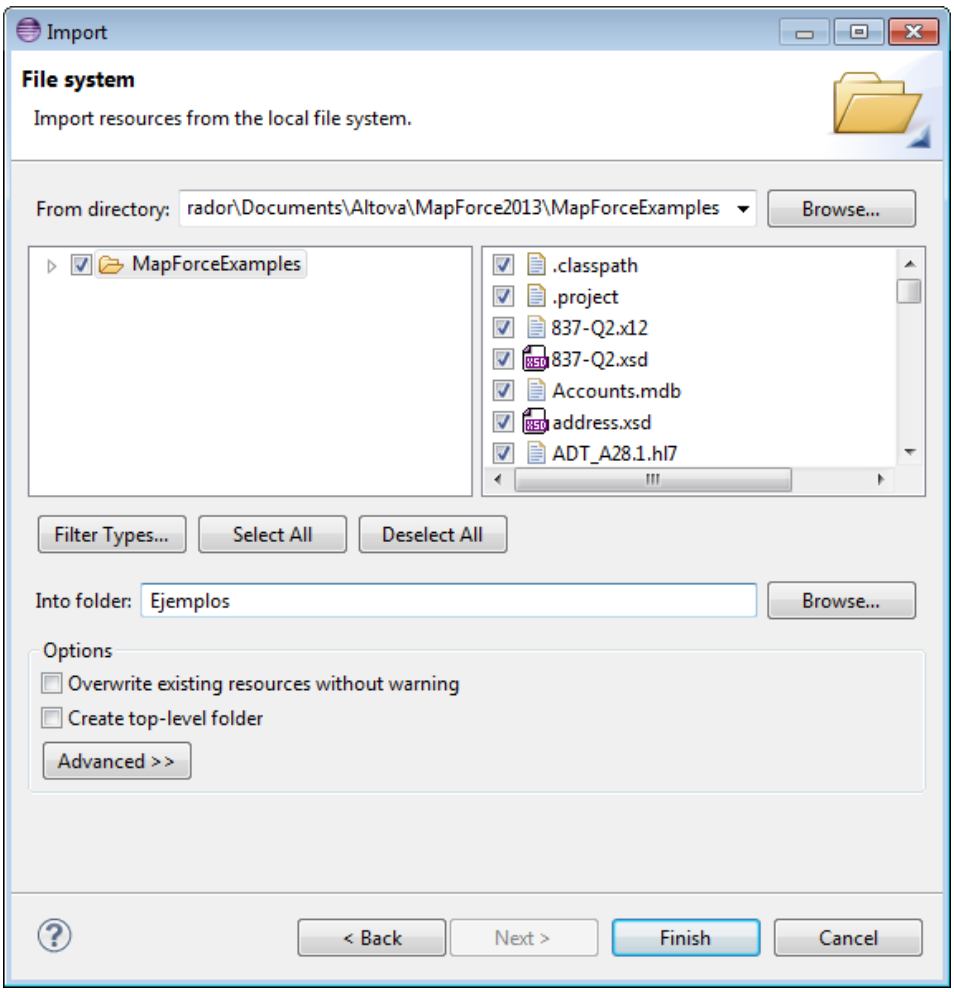
- **Asignación:** muestra el diseño gráfico de la asignación.
- **XSLT:** muestra el código XSLT generado. El nombre de este panel coincide con la opción seleccionada en el menú **Resultados | XSLT 1.0, XSLT2**, o **XQuery**.
- **Resultados:** muestra una vista previa de los resultados de la asignación.

 © 2014 Altova GmbH

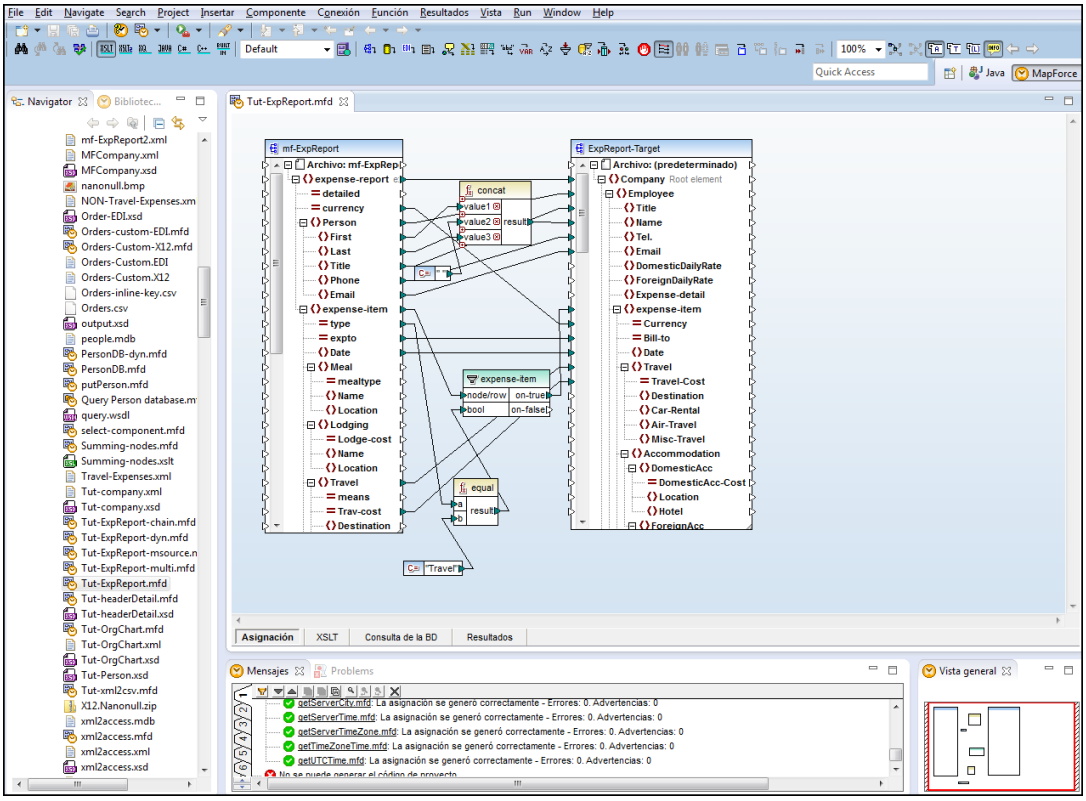
19.4 Importar la carpeta de ejemplos de MapForce

Para importar la carpeta de ejemplos de MapForce en la ventana Navigator:
Si no tiene ya un proyecto, cree uno con el comando **File | New | Project | Proyecto** de MapForce Eclipse.

1. Haga clic con el botón secundario en el nombre del proyecto en la pestaña *Navigator* y haga clic en **Import**.
2. Seleccione el comando de menú **General | File system** y después haga clic en **Next** para continuar.
3. Haga clic en el botón **Browse** situado a la derecha del cuadro de texto *From directory*. Seleccione la carpeta `Examples` de MapForce.
4. Marque la casilla **`MapForceExamples`**. Esto selecciona todos los archivos de los subdirectorios que aparecen en el cuadro de lista de la derecha.



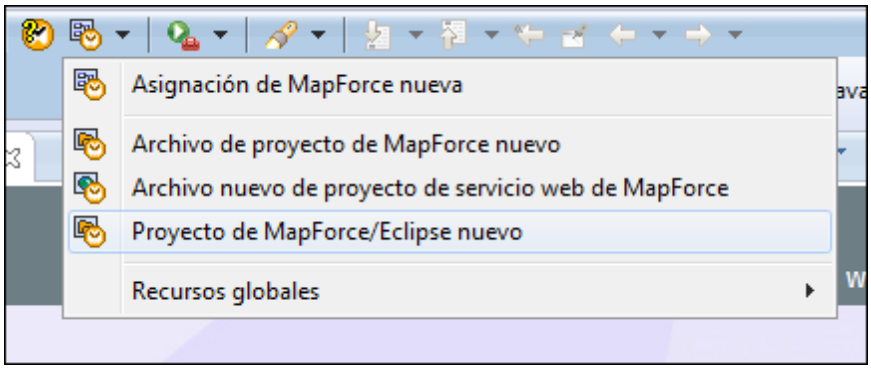
5. Si la carpeta de destino no aparece automáticamente, haga clic en el botón **Browse** junto al cuadro *Into folder*, seleccione la carpeta de destino y haga clic en **Finish** para finalizar.
   La estructura de la carpeta seleccionada y sus archivos se copian al área de trabajo de Eclipse.
6. Para abrir un archivo, haga doble clic en su nombre en la ventana Navigator.


File Edit Navigate Search Project Insertar Componente Conexión Función Resultados Vista Run Window Help
Navigator
Tut-ExpReport.mfd
mf-ExpReport
ExpReport-Target
concat
expense-item
equal
Asignación
XSLT
Consulta de la BD
Resultados
Mensajes
Problems
Vista general

19.5 Crear archivos de MapForce nuevos

Para crear archivos de asignación/proyecto de MapForce nuevos:

- Haga clic en el cuadro combinado **Archivo nuevo...** y seleccione una opción.



- **Asignación de MapForce nueva** crea un solo archivo de asignación.
- **Archivo de proyecto de MapForce nuevo** crea un proyecto de MapForce que puede reunir varios archivos de asignación en una sola unidad de generación de código.
- **Archivo nuevo de proyecto de servicio web de MapForce** crea un proyecto de servicio web.
- **Proyecto de MapForce/Eclipse nuevo** crea un proyecto de MapForce/Eclipse, añadiendo la carpeta a la ventana Navigator. Los proyectos MapForce/Eclipse son proyectos de Eclipse que tienen asignado un compilador de MapForce. Para más información consulte Usar proyectos de MapForce Eclipse para generación automática.

19.6 Generación de código con MapForce

Integración de compilación
Las asignaciones de MapForce se pueden incluir en cualquier proyecto de Eclipse. La generación de código de asignación se puede iniciar a mano (seleccionando una de las opciones de menú **Generar código...**, tanto para la asignación como para el archivo de proyecto de MapForce). La integración completa en el proceso de compilación automática de Eclipse se consigue asignando el compilador de MapForce a un proyecto de Eclipse.

Para la generación de código manual, consulte el apartado Generar código de asignación a mano.

Para la generación automática de código de asignación, consulte los apartados Usar proyectos de MapForce/Eclipse para generación automática y Agregar la naturaleza MapForce a un proyecto de Eclipse.

 © 2014 Altova GmbH

19.6.1 Generar código de asignación a mano

Para generar código de asignación a mano para un solo archivo de asignación:

1. Abra o seleccione la asignación y después seleccione el comando **Archivo | Generar código en** o **Archivo | Generar código en el lenguaje seleccionado.**
   MapForce Eclipse pregunta por una carpeta de destino para el código generado.
2. Seleccione la carpeta y haga clic en **Aceptar** para iniciar la generación de código.
   En la ventana Mensajes se muestran errores y advertencias durante la generación de código.

Para generar código de asignación a mano para varios archivos de asignación de un proyecto de MapForce:

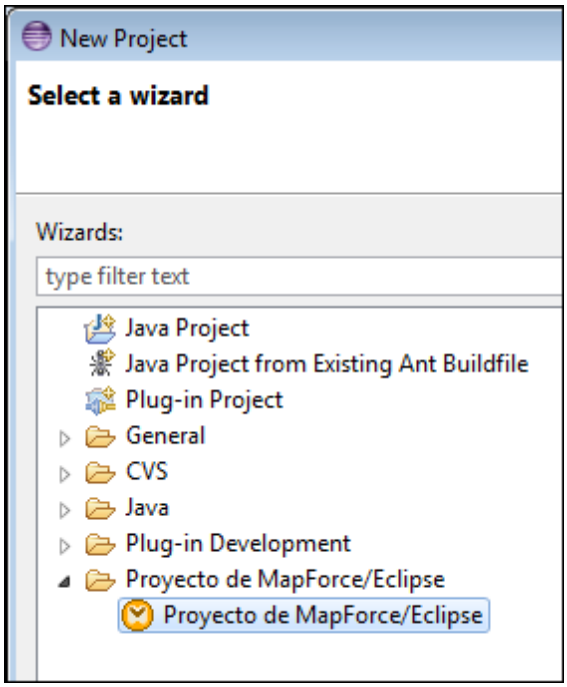
1. Abra o seleccione el archivo de proyecto de MapForce.
2. Seleccione el nodo raíz o cualquier otro nodo del archivo de proyecto.
3. Haga clic con el botón secundario y seleccione **Generar código** o **Generar código en...**.
   La carpeta de destino del código generado viene dada por las propiedades del nodo seleccionado o del nodo primario.
4. En la ventana Mensajes se muestran errores y advertencias durante la generación de código.

19.6.2 Usar proyectos de MapForce/Eclipse para generación automática

El complemento de MapForce tiene un compilador integrado. Este compilador tiene el Id. de naturaleza de proyecto `"com.altova.mapforceeclipseplugin.MapForceNature"`. Los proyectos de MapForce Eclipse tienen asignada esta naturaleza automáticamente. Si desea usar el compilador de proyectos de MapForce en otros proyectos de Eclipse, consulte el apartado Agregar la naturaleza MapForce Eclipse para generación automática.

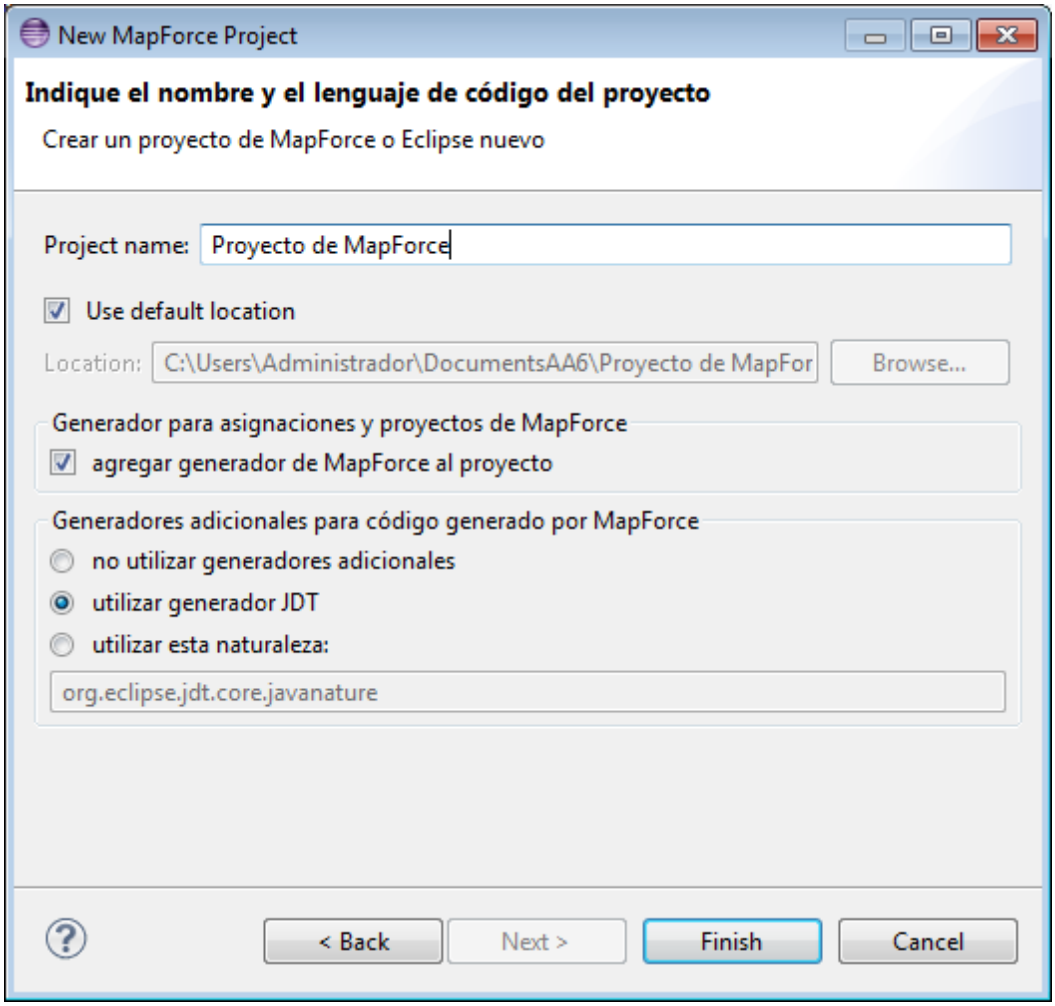
Para crear un proyecto MapForce Eclipse nuevo:

1. Haga clic en la ventana Navigator.
2. Ahora haga clic con el botón secundario y seleccione **New | Project.**
3. Expanda la entrada *Proyecto de MapForce/Eclipse*, seleccione *Proyecto de MapForce/Eclipse* y haga clic en **Next** para continuar.

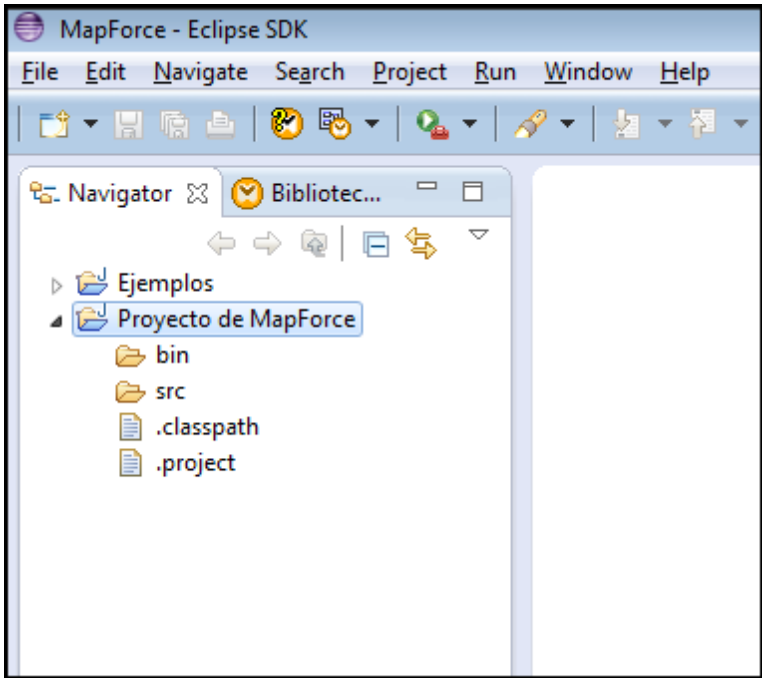


4. Escriba el nombre del proyecto (p. ej. Proyecto de MapForce) y ajuste la configuración según sus requisitos. Después haga clic en **Finish** para finalizar. Observe que la opción predeterminada en el grupo de opciones *Generadores adicionales....* es la opción *utilizar generador JDT*.

 © 2014 Altova GmbH



En esta carpeta se crea una carpeta de proyecto de Eclipse y algunas carpetas y archivos opcionales.



Ahora puede crear asignaciones y archivos de proyecto de MapForce dentro de este proyecto de Eclipse o copiar en él archivos ya existentes. Cada vez que se realicen cambios en un archivo de asignación o proyecto de MapForce, el código de asignación correspondiente se genera automáticamente. Los errores y advertencias sobre el proceso de generación aparecen en la ventana Mensajes y se añaden a la vista "Problems" de Eclipse.

Un proyecto de MapForce Eclipse es un proyecto de Eclipse que tiene asignada la naturaleza MapForce y que, por tanto, utiliza el compilador de MapForce.

Si en el proyecto de Eclipse hay archivos de proyecto de MapForce, el lenguaje de generación de código y las carpetas de destino de los resultados vienen dados por la configuración de estos archivos.

Si en el proyecto de Eclipse no hay archivos de proyecto de MapForce, pero al proyecto de Eclipse **se le asignó** la naturaleza JDT:

- la generación de código de asignación usa por defecto el lenguaje Java y el directorio de código fuente Java del proyecto se usa como directorio de salida del código de asignación.

  Al guardar la asignación se genera y compila automáticamente el código de asignación en Java. Use el comando **Run** o **Debug** de Java para probar la aplicación de asignación resultante.

Si en el proyecto de Eclipse no hay archivos de proyecto de MapForce, pero al proyecto de Eclipse **no se le asignó** la naturaleza JDT:

- la carpeta de destino de los resultados es la carpeta del proyecto y el lenguaje de generación de código es el indicado en el cuadro de diálogo "Opciones" de MapForce.

 © 2014 Altova GmbH

Para activar el proceso de generación automática:

1. Compruebe que está activada la opción de menú **Project | Build automatically**.

Para desactivar temporalmente la generación automática de código de asignación de MapForce:

Esta opción solamente está disponible en proyectos de Eclipse a los que se añadió la naturaleza MapForce.

1. Haga clic con el botón secundario en el proyecto de Eclipse en la ventana Navigator.
2. Seleccione la opción **Properties** del menú contextual.
3. En el panel izquierdo del cuadro de diálogo haga clic en **Builders** para ver una lista de compiladores.
4. Desactive la casilla *Generación de código de MapForce* en el panel derecho.
   A partir de ahora no se generará código de asignación automáticamente cada vez que se realicen cambios en los archivos de asignación o proyecto de MapForce de este proyecto de Eclipse.

19.6.3 Agregar la naturaleza MapForce a un proyecto de Eclipse

Aplicar la naturaleza MapForce a proyectos actuales:
Añada el texto que aparece a continuación a la sección `natures` del archivo `.project` en el proyecto de Eclipse (p. ej. en la carpeta `c:\eclipse33\workspace\MapForce\`):

```
<nature>com.altova.mapforceeclipseplugin.MapForceNature</nature>
```

```
<natures>
 <nature>org.eclipse.jdt.core.javanature</nature>
 <nature>com.altova.mapforceeclipseplugin.MapForceNature</nature>
</natures>
```

A partir de este momento los archivos de proyecto y asignaciones de MapForce que estén en este proyecto participarán en el proceso de generación automática. Para más información consulte Usar proyectos de MapForce/Eclipse para generación automática.

 © 2014 Altova GmbH

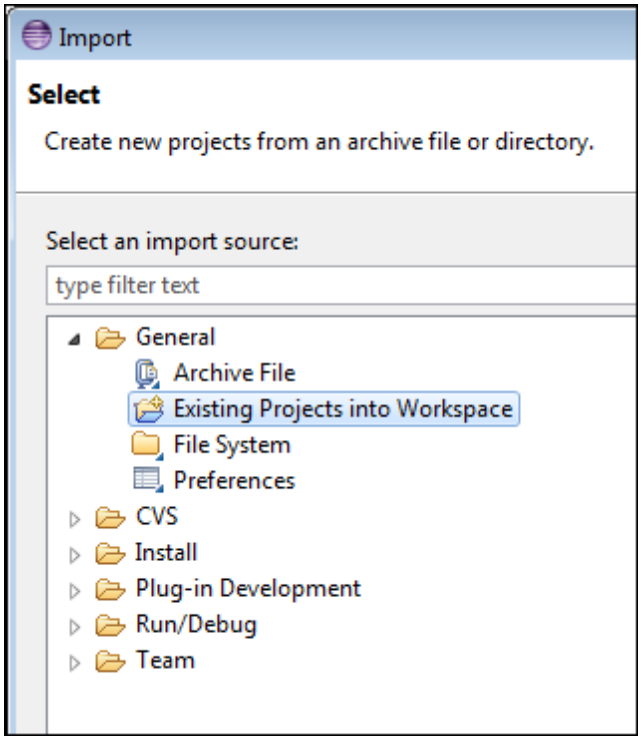
19.7 Ampliar el complemento de MapForce

El complemento de MapForce ofrece un punto de extensión de Eclipse con el Id. "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 COM del control de MapForce y de la API de MapForce.

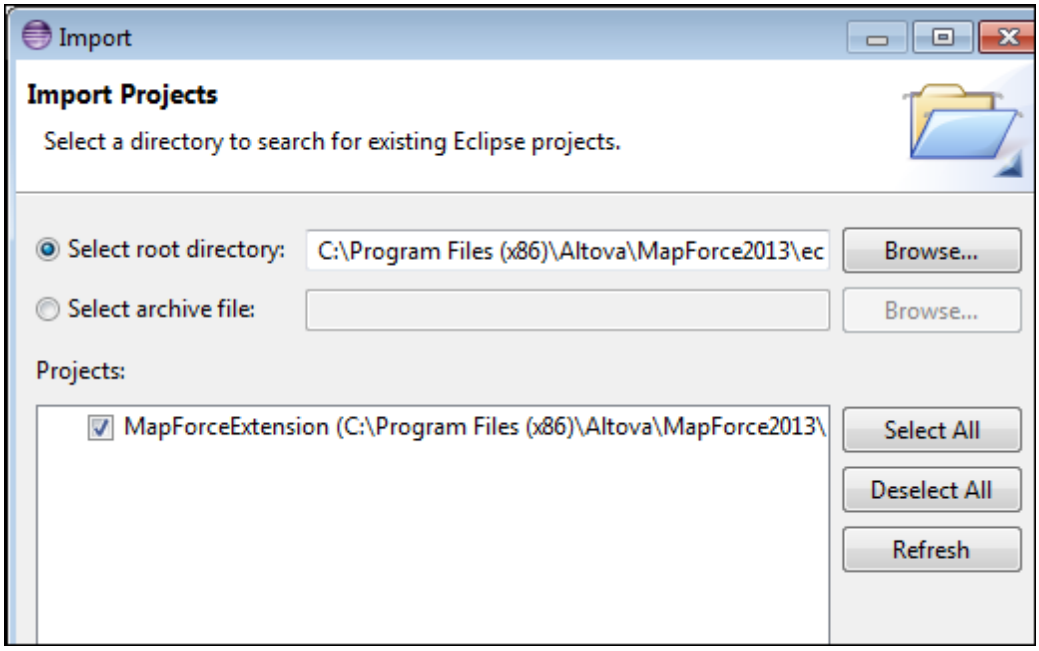
El paquete de instalación de MapForce Eclipse incluye un sencillo ejemplo de un complemento que usa este punto de extensión. Comprueba si las nuevas asignaciones de MapForce tienen eventos File open y establece el nivel de zoom de la vista en 70%.

Instalar el complemento de extensión de muestra:
El complemento de MapForce requiere tener instalado el complemento JDT (Java Development Tools).

1. Inicie Eclipse.
2. Haga clic con el botón secundario en la ventana Navigator o PackageExplorer y seleccione **Import** en el menú contextual.
3. Seleccione la entrada *Existing projects into Workspace* y haga clic en **Next** para pasar a la siguiente pantalla.



4. Haga clic en el botón **Browse...** del cuadro combinado *Select root directory* y seleccione el directorio del proyecto de muestra (`C:\Archivos de programa\Altova\MapForce2014\eclipse\workspace\MapForceExtension`).



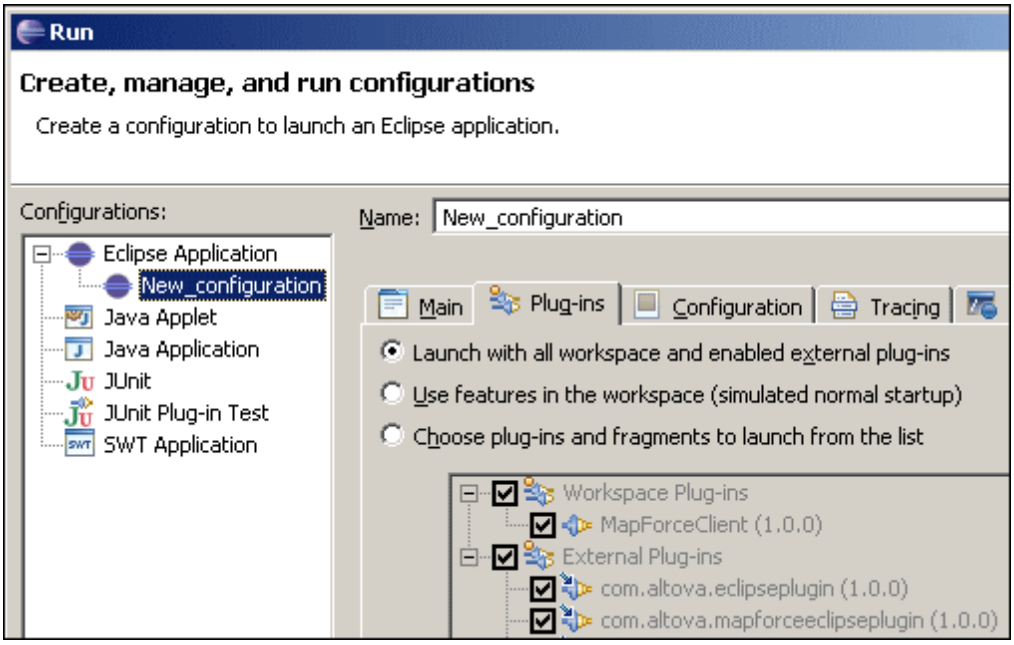
5. Para terminar haga clic en **Finish**.
   En el área de trabajo se crea un proyecto nuevo llamado **MapForceExtension**.

Acceder a javadoc para el punto de extensión del complemento de MapForce:

1. Abra el archivo `index.html` de la carpeta `docs` de la carpeta de instalación del complemento (`c:\Archivos de programa\Altova\MapForce2014\eclipse\docs\`)

Ejecutar el complemento de extensión de muestra:

1. Cambie a la perspectiva Java.
2. Seleccione la opción **Run | Run...**
3. Seleccione *Eclipse Application* y haga clic en *New_configuration*.



 © 2014 Altova GmbH

4. Compruebe si está seleccionado el proyecto MapForceClient en la pestaña *Plug-ins*.
5. Haga clic en el botón **Run**. Se abre una nueva área de trabajo de Eclipse.
6. Abra cualquier archivo de asignación de MapForce en la nueva área de trabajo. Se abrirá con un nivel de zoom del 70%.

Altova MapForce 2014

Referencia del usuario

20 Referencia del usuario

Este apartado enumera y describe todos los menús y opciones de menú de MapForce.

© 2014 Altova GmbH

20.1 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`).

Nota: no se pueden abrir asignaciones con características de una edición superior de MapForce. Por ejemplo, no se pueden abrir asignaciones con funciones de servicio web en la edición Professional Edition ni asignaciones de bases de datos en la edición Basic Edition.

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.

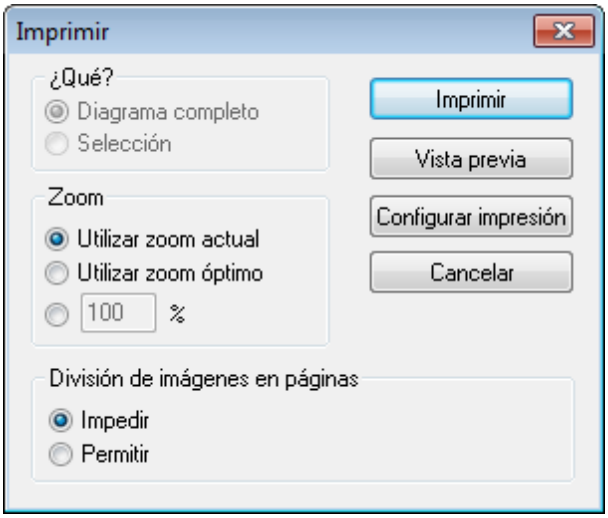
Guardar todos
Guarda todos los archivos que están abiertos.

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".



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.

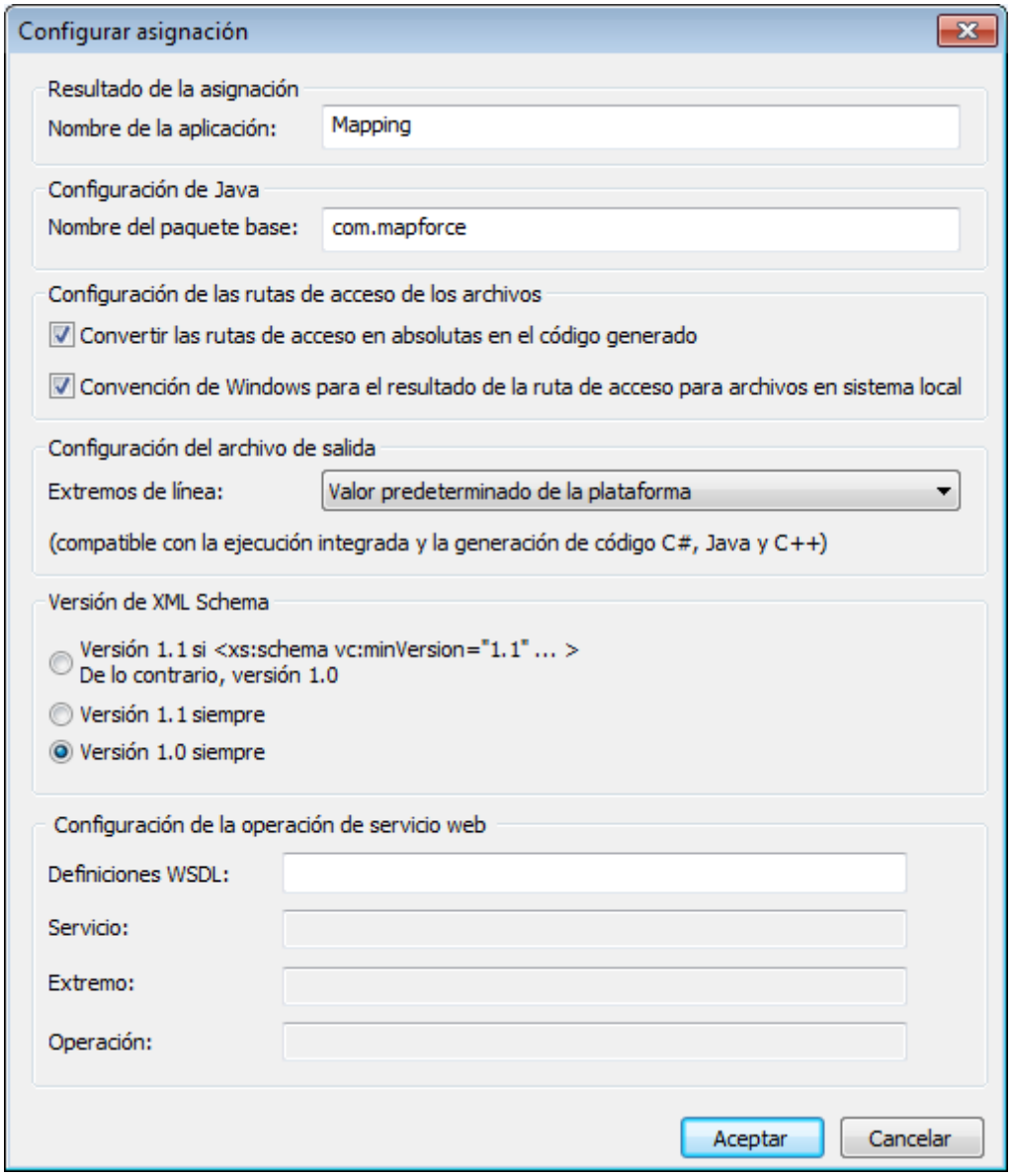
Vista previa de impresión
Abre el cuadro de diálogo "Imprimir" (ver comando **Imprimir**).

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 válidos y emite advertencias y errores de validación. Para más información consulte Validar asignaciones.

Configurar asignación

 © 2014 Altova GmbH



Aquí se define la configuración del documento de asignación y la configuración se guarda en el archivo `*.mfd`.

Resultado de la asignación

***Nombre de la aplicación*:** define el prefijo del nombre del archivo XSLT1.0/2.0 o el nombre de la aplicación Java, C# o C++ para los archivos de transformación generados.

Configuración de Java

***Nombre del paquete base*:** define el nombre del paquete base para los resultados Java.

Configuración de las rutas de acceso de los archivos

Convertir las rutas de acceso en absolutas en el código generado: garantiza la compatibilidad del código generado con archivos de asignación (`*.mfd`) creados en versiones de MapForce

anteriores a la versión 2010. Consulte el apartado Rutas de archivo relativas y absolutas para más información.

Convención de Windows para el resultado de la ruta de acceso...
Garantiza el cumplimiento de la convención de Windows para rutas de acceso. Cuando se generan archivos XSLT2 (y XQuery), el nombre de archivo actual se recupera de forma interna con la función document-uri, que devuelve una ruta de acceso en formato `archivo:// URI` para archivos locales.

Si marca esta casilla, la especificación de ruta de acceso `archivo:// URI` se convierte automáticamente en una ruta de acceso Windows (p. ej. `C:\...`) para simplificar el procesamiento.

Configuración del archivo de salida
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 la asignación en un servidor de FlowForce Server que se ejecute en diferentes sistemas operativos.

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 la versión actual.

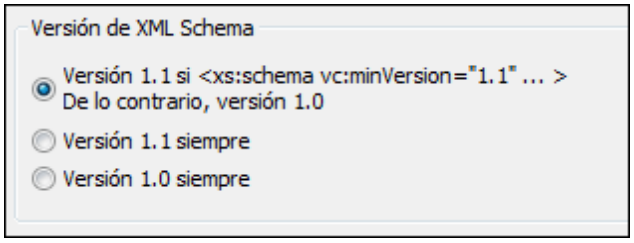
El **modo XSD** que se utiliza para abrir el archivo depende de dos opciones de configuración: una opción de configuración de la aplicación y otra del documento XSD.

¿Cómo seleccionar el modo XSD?
El modo XSD determina qué características de validación están disponibles para la asignación actual (XSD 1.0 o 1.1.).

- Puede ser un modo general para toda la aplicación, si selecciona la *Versión 1.1 siempre* o *Versión 1.0 siempre*. De ser así, todos los documentos XSD (de la asignación) se abrirán en el modo seleccionado.
- Puede ser un modo para la asignación activa solamente, si la declaración <xs:schema vc:minVersion="1.1"...> existe en el documento XSD. La aplicación selecciona automáticamente el modo XSD en función de los datos de esta declaración. 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.

  Nota: 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 es XSD 1.0.

 © 2014 Altova GmbH



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.

Configuración de la operación de servicio web:
Los campos *Definiciones WSDL*, *Servicio*, *Extremo* y *Operación* se rellenan automáticamente si el documento de asignación es parte de una implementación de servicio web.

Generar código en el lenguaje seleccionado
Genera código en el lenguaje seleccionado en la barra de herramientas: XSLT / XSLT 2, XQuery, Java, C#, o C++.

Generar código en | XSLT 1.0 / XSLT 2.0
Genera los archivos XSLT necesarios para la transformación de datos. Al hacer clic en este comando se abre el cuadro de diálogo "Buscar carpeta" donde puede seleccionar la ubicación del archivo XSLT.

Nota: el nombre del archivo XSLT generado se define en el campo *Nombre de la aplicación* del cuadro de diálogo "Configurar asignación" (**Archivo | Configurar asignación**).

Generar código en | XQuery
Genera los archivos XQuery necesarios para la transformación de datos. Al hacer clic en este comando se abre el cuadro de diálogo "Buscar carpeta" donde puede seleccionar la ubicación del archivo XQuery.

Nota: el nombre del archivo XQuery generado se define en el campo *Nombre de la aplicación* del cuadro de diálogo "Configurar asignación" (**Archivo | Configurar asignación**).

Generar código en | Java
Generar código en | C#
Generar código en | 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.

Nota: el nombre de los archivos de aplicación generados (así como el del archivo de proyecto C# `*.csproj`, del archivo de solución `*.sln` o del archivo de proyecto C++ `*.vcproj`) se define en el

campo *Nombre de la aplicación* del cuadro de diálogo "Configurar asignación" (**Archivo | Configurar 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
Implementa/genera un archivo de asignación de salida que se puede ejecutar directamente en MapForce Server sin necesidad de usar FlowForce Server.

Implementar en FlowForce Server
Implementa la asignación activa en FlowForce Server. Para implementar asignaciones en FlowForce y definir trabajos y desencadenadores es necesario tener instalado FlowForce Server y MapForce Enterprise Edition.

Puede descargar estos programas de la página http://www.altova.com/es/download/flowforce.html.

Para ver un ejemplo de implementación consulte el apartado Implementar una asignación en FlowForce.

Generar documentación
Genera documentación para los proyectos de asignación en varios formatos. Para más información consulte el apartado Documentar proyectos de asignación de datos.

Archivos recientes
Una lista con archivos abiertos recientemente.

Proyectos recientes
Una lista con proyectos abiertos recientemente.

Salir
Cierra la aplicación. Antes aparece un aviso preguntando si desea guardar los archivos que estén sin guardar.

 © 2014 Altova GmbH

20.2 Edición

La mayoría de los comandos del menú **Edición** se habilitan al abrir el panel **Resultados** o el panel **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 los paneles panel **XSLT**, **XSLT2**, **XQuery** y **Resultados**.

Buscar siguiente **F3**
Busca la siguiente aparición del término de búsqueda.

Buscar anterior **Mayús F3**
Busca la aparición 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/código de los paneles **XSLT**, **XSLT2**, **XQuery** y **Resultados**.

20.3 Insertar

Archivo o esquema XML
Inserta un archivo de esquema XML como componente de origen o destino. Puede seleccionar archivos XML con una referencia de esquema, en cuyo caso se inserta automáticamente el esquema referenciado. Si selecciona un archivo de esquema XML, aparece un aviso preguntando si desea incluir un archivo XML de instancia que aporte los datos para los paneles **XSLT**, **XSLT2**, **XQuery** y **Resultados**. Si selecciona un archivo XML sin referencia de esquema, aparece un aviso preguntando si desea generar automáticamente el esquema XML correspondiente.

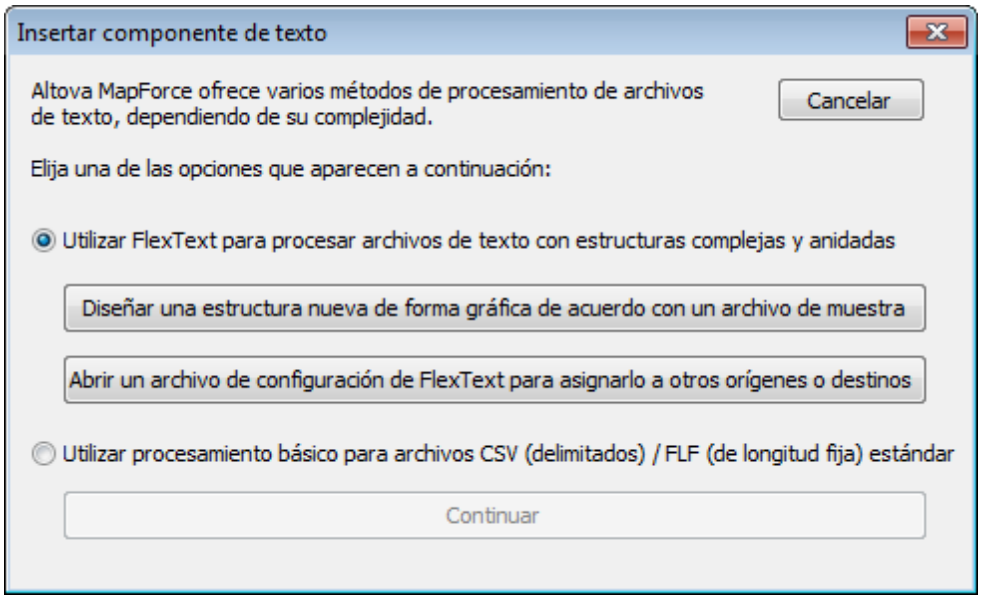
Base de datos
Inserta una base de datos como componente de origen o destino. La base de datos aporta la estructura del esquema y aparece representada en una vista en forma de árbol.

EDI
Inserta un documento EDI como componente de origen o destino. Para más información consulte la sección EDI de este manual.

Archivo de texto
Inserta un archivo plano (es decir, un archivo de texto CSV o de longitud fija). Ambos tipos de archivo pueden usarse como componente de origen o de destino.

Abre el cuadro de diálogo "Insertar componente de texto" donde puede elegir si:

- diseña una plantilla FlexText basada en un archivo de muestra.
- abre una plantilla FlexText ya existente para asignarla a otros componentes de destino.
- inserta un archivo de texto CSV/FLF estándar.



Función de servicio web
Inserta en la asignación una función de servicio web definida por un archivo WSDL. Conecte la función de servicio web a componentes de entrada y salida para obtener una vista previa

 © 2014 Altova GmbH

inmediata de los resultados de la función de servicio web en el panel **Resultados**. Consulte el apartado Llamar a servicios web para obtener más información.

Archivo Excel 2007+
Inserta un archivo Microsoft Excel 2007+ (Office Open XML) (`*.xlsx`) como componente de origen o destino. Para más información consulte la sección Asignación de datos de archivos MS OOXML Excel 2007+.

Para ver una vista previa de las Excel 2007+ en el panel **Resultados** es necesario tener Microsoft Excel 2007+ instalado en el equipo. En MapForce puede usar archivos Excel 2007+ en las asignaciones de datos aunque tenga una versión previa de Microsoft Excel o aunque no tenga Microsoft Excel instalado. Sin embargo, en estos casos no podrá ver la vista previa de los resultados en el panel **Resultados**. Solamente puede guardar los resultados haciendo clic en el icono **Guardar archivo de salida**.

Documento XBRL
Inserta un documento XBRL o de taxonomía como componente de origen o destino. Para más información consulte la sección Asignación de datos XBRL.

Constante
Inserta una constante que aporta datos fijos a una conexión de entrada. Los datos se introducen en un cuadro de diálogo que aparece nada más crear el componente de constante. Las constantes solamente tienen una conexión de salida y 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. Consulte el apartado Variables intermedias para obtener más información.

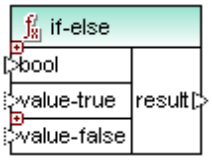
Filtro: nodos/filas
Inserta un componente con dos parámetros de entrada (**`node/row`** y **`bool`**) y dos de salida (**`on-true`** y **`on-false`**). Si el valor booleano es `true`, el valor del parámetro **`node/row`** se envía al parámetro **`on-true`**. Si el booleano es `false`, el valor del parámetro **`node/row`** se envía al parámetro **`on-false`**. Para ver un ejemplo de uso consulte el *Tutorial*.

WHERE/ORDER de SQL
Inserta un componente de filtrado especial para datos de BD que permite anexar cualquier cláusula WHERE de SQL a las consultas generadas por MapForce. Para más información consulte el apartado Componente WHERE/ORDER de SQL.

Asignación de valores
Inserta un componente que transforma un valor de entrada a un valor de salida usando una tabla de búsqueda. El componente solamente tiene una entrada y una salida. Para más información consulte el apartado Asignación de valores: transformar datos de entrada.

Condición IF-Else

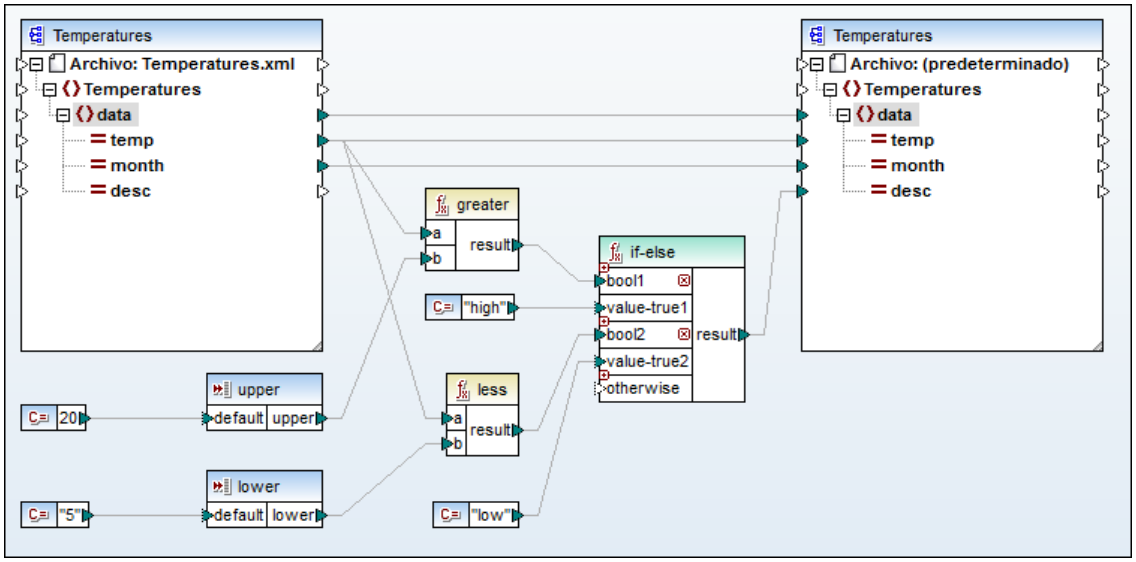
Este componente sirve para pasar diferentes conjuntos de datos dependiendo del resultado de una condición predefinida. El título de este componente es **`if-else`**.



- El primer parámetro de entrada es **`bool`**, que contiene valores booleanos.
- Si el valor de **`bool`** es `true`, los datos del parámetro de entrada **`value-true`** se pasan al parámetro de salida **`result`**.
- Si el valor de **`bool`** es `false`, los datos del parámetro de entrada **`value-false`** se pasan al parámetro de salida **`result`**.
- El parámetro de salida **`result`** incluye los datos que le pasan los parámetros de entrada **`value-true`**/**`value-false`**.

La condición **`IF-Else`** es una función **ampliable**. Esto significa que puede verificar varias condiciones distintas y usar el parámetro **`otherwise`** para enviar al componente de destino el valor de la condición **`Else`**.

Haga clic en el icono **+** para insertar/anexar un par **`if-else`** nuevo. Para borrar el par de parámetros haga clic en el icono **x**.



Por ejemplo, en la asignación de la imagen anterior se analizan los datos de temperatura:

- Si la temperatura (elemento **`temp`**) es superior a **20** (función **`greater`**), se pasa el valor booleano `true` al parámetro **`bool1`** y el resultado es la cadena **high** del parámetro **`value-true1`.**
- O si la temperatura (elemento **`temp`**) es menor que **5** (función **`less`**), se pasa el valor booleano `true` al parámetro **`bool2`** y el resultado es la cadena **low** del parámetro **`value-true2`**.

 © 2014 Altova GmbH

- De lo contrario, el resultado del componente es una cadena vacía (nada) porque el parámetro de entrada **`otherwise`** no está conectado.

Resultado de la asignación:

```
<?xml version="1.0" encoding="UTF-8"?>
<Temperatures xsi:noNamespaceSchemaLocation="c:/DOCUME~
http://www.w3.org/2001/XMLSchema-instance">
  <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"/>
```

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 Excepciones en MapForceo Elementos de error fault en servicios web para más información.

20.4 Proyecto

MapForce integra la interfaz de múltiples documentos y permite al usuario agrupar sus asignaciones en proyectos. Los archivos de proyecto tienen la extensión `*.mfp`.

En MapForce se pueden definir dos tipos de proyectos:

- Un conjunto de asignaciones independientes (un proyecto estándar)
- Un conjunto relacionado de asignaciones que componen un proyecto de asignación WSDL (para implementar un servicio web)
- En ambos tipos de proyectos MapForce puede generar código para todo el proyecto

Volver a cargar el proyecto
Vuelve a cargar el proyecto activo y cambia a la ventana Proyecto.

Cerrar el proyecto
Cierra el proyecto activo.

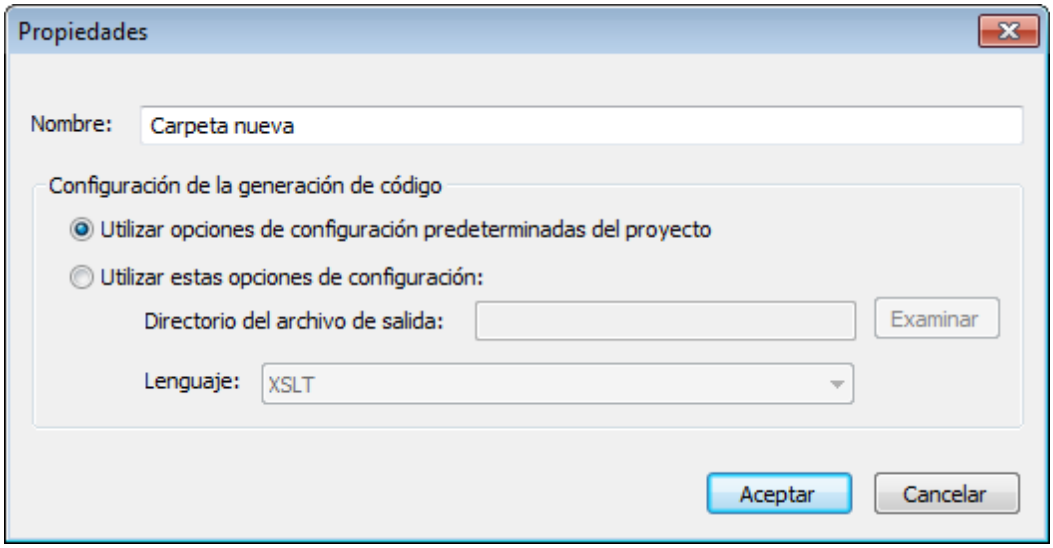
Guardar el proyecto
Guarda el proyecto activo.

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 activo.

Crear carpeta
Agrega una carpeta nueva a la estructura del proyecto activo. Este comando solamente está disponible cuando se puede crear una carpeta nueva. Al hacer clic en este comando aparece el cuadro de diálogo "Propiedades" donde puede utilizar la configuración predeterminada o definir una configuración personal, haciendo clic en el botón de opción *Utilizar estas opciones de configuración.*

 © 2014 Altova GmbH



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...
Inserta un servicio web basado en un archivo WSDL ya existente.

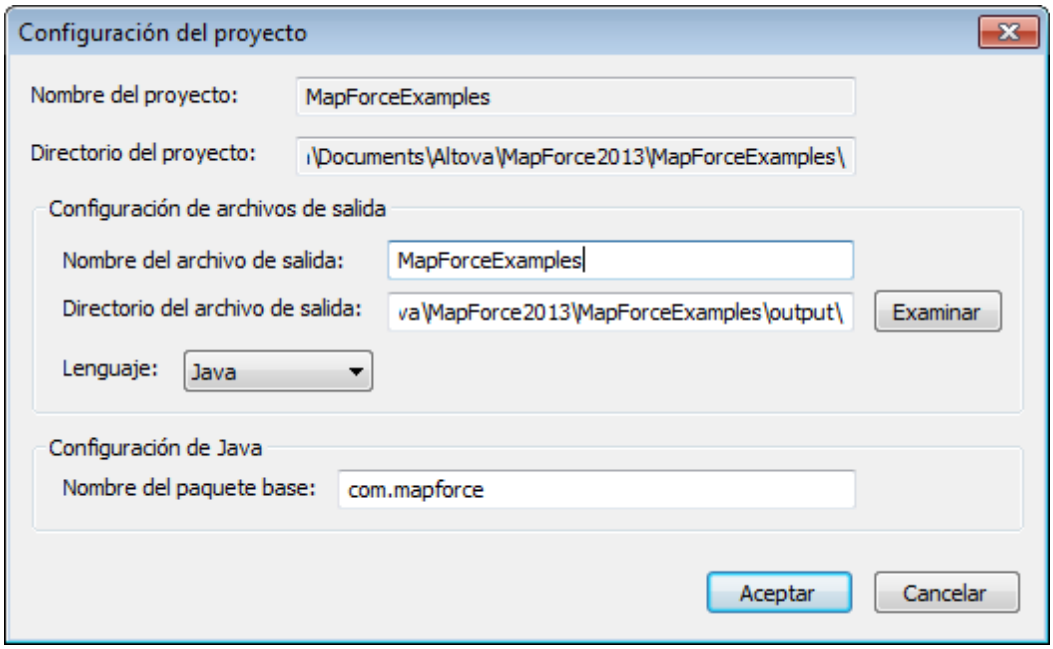
Abrir archivo en XMLSpy
Abre el archivo WSDL seleccionado (resaltado en la ventana Proyecto) en la herramienta XMLSpy.

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

Haga clic en el nombre del proyecto (p. ej. **`MapforceExamples`**) en la ventana Proyecto y seleccione el comando **Proyecto | Propiedades**. Aparece un cuadro de diálogo donde puede definir la configuración global del proyecto. Haga clic en una carpeta o nombre de archivo, en la ventana Proyecto, y seleccione este comando para configurar la carpeta o el archivo.



 © 2014 Altova GmbH

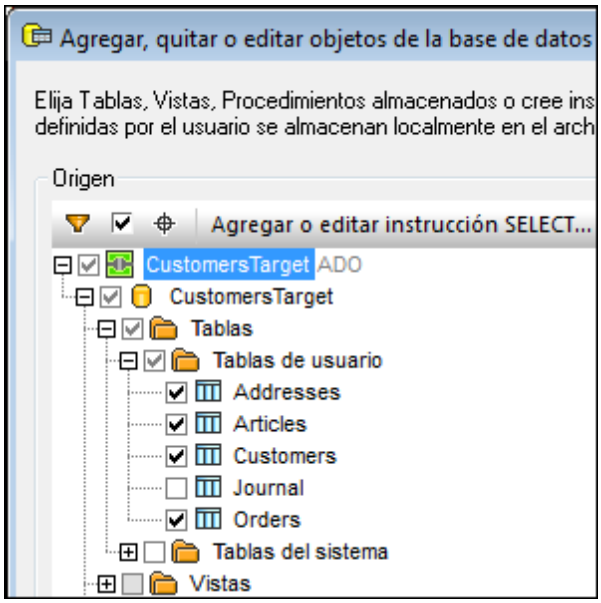
20.5 Componente

Cambiar de elemento raíz
Sirve para seleccionar un elemento raíz diferente del archivo XML de instancia.

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 cambiar el número de tablas del componente de base de datos activándolas/desactivándolas en este cuadro de diálogo:



Crear asignación a EDI X12 997
La confirmación funcional 997 de X12 informa del estado del intercambio EDI. Todos los errores encontrados durante el procesamiento del documento aparecen en esta confirmación.

MapForce puede generar documentos 997 de X12 automáticamente en el panel **Asignación**. Para más información sobre los segmentos de error y lo que significan consulte el apartado Confirmación 997 de X12.

Crear asignación a EDI X12 999
El conjunto de transacciones de la confirmación de implementación 999 de X12 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 X12: Confirmación de implementación 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 Duplicar elementos de entrada del *Tutorial*.

Haga clic con el botón secundario 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 Duplicar elementos de entrada del *Tutorial*.

Haga clic con el botón secundario 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 Duplicar elementos de entrada 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 BD de destino. Consulte la sección Acciones de tabla, configuración de clave y transacciones para más información.

Consultar la base de datos
Crea una instrucción SELECT basada en la tabla / el campo resaltado en el componente de base de datos. Este comando se habilita al hacer clic una vez en la tabla / 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 conexiones con un componente de destino.

Propiedades
Abre un cuadro de diálogo con las opciones de configuración actuales del componente.

 © 2014 Altova GmbH

Nombre del componente

Todos los componentes de una asignación que estén basados en un archivo (es decir, todos excepto los componentes de base de datos) tienen un nombre de componente, que se define automáticamente nada más crear el componente. Este nombre se puede cambiar en todo momento.

El nombre del componente se puede utilizar para acceder al componente desde FlowForce. El nombre de componente debe ser único si tiene pensado acceder al componente desde FlowForce.

El nombre predeterminado se genera de varias maneras, dependiendo del tipo de componente. El nombre puede estar basado en:

- El nombre del archivo XML de entrada/salida
- El nombre de la taxonomía
- El nombre del mensaje EDI
- El nombre del archivo de configuración de FlexText
- El tipo de componente (p. ej. `Archivo de texto` o `Archivo Excel`)

Si el nombre de componente se generó automáticamente y después selecciona un archivo de instancia, aparece un aviso preguntando si desea actualizar también el nombre del componente.

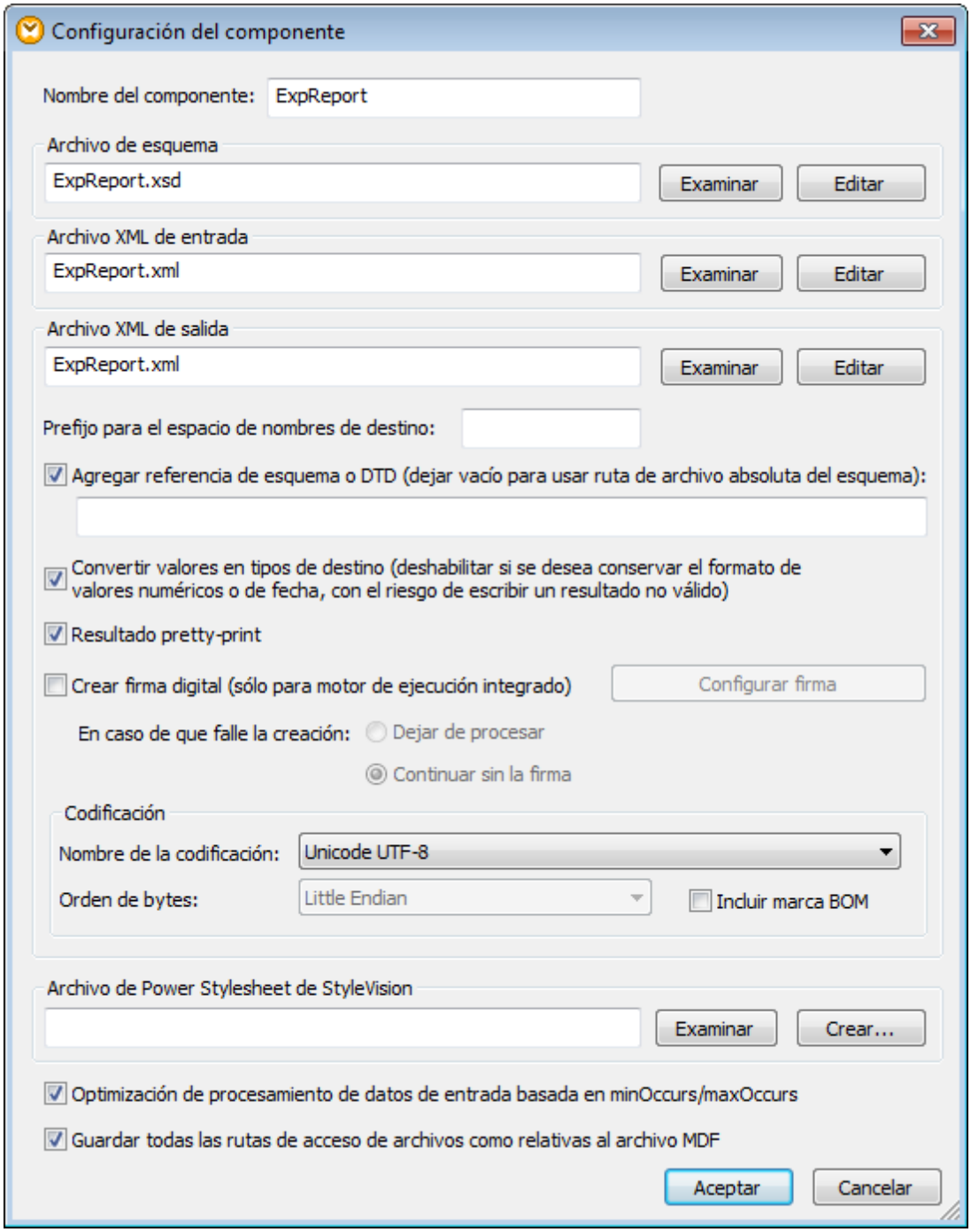
El nombre del componente puede contener:

- espacios (p. ej. `Archivo de texto` o `Archivo Excel`). No está permitido usar espacios al principio ni al final de la cadena.
- puntos (p. ej. `Orders.EDI`)

Tenga en cuenta que en la línea de comandos es difícil o a veces imposible introducir determinados caracteres y que los caracteres nacionales pueden tener codificación diferente en Windows y en la línea de comandos.

El nombre del componente no puede contener:

- barras diagonales, barras diagonales inversas ni dos puntos
- comillas (ni simples ni dobles)
- tabulaciones ni CR/LF como espacios en blanco



Archivo de esquema: el nombre y la ruta de acceso del archivo de esquema de destino.

Archivo XML de entrada: el archivo XML de instancia para el componente de esquema. Este campo se rellena automáticamente nada más insertar el componente de esquema en la asignación y asignarle un archivo XML de instancia.

Archivo XML de salida: el nombre y la ruta de acceso del archivo XML de salida para la generación y ejecución de código de programa. El nombre de archivo aparece como primer elemento del componente. El contenido del campo *Archivo XML de entrada* se copia

 © 2014 Altova GmbH

automáticamente en este campo cuando se asigna un archivo XML de instancia al componente de esquema. Si no asigna un archivo XML al componente, este campo contiene la entrada `NombreYRutaDelArchivoDeEsquema.xml`.

Prefijo para el espacio de nombres de destino: el prefijo para el espacio de nombres de destino si se trata de un documento / esquema XML. Para poder asignarle un prefjo, en el esquema de destino se debe definir el espacio de nombres de destino.

Agregar referencia de esquema o DTD: agrega al elemento raíz del XML de salida la ruta del archivo de esquema XML referenciado.

Introduzca aquí una ruta de acceso para definir la ubicación del archivo de esquema al que hace referencia el archivo XML de instancia. Esto permite validar el archivo de salida en la ubicación de la asignación cuando se ejecuta la asignación. Puede introducir una dirección http:// o una ruta de acceso absoluta o relativa.

Desactive esta opción para separar el archivo XML de instancia del esquema XML o DTD referenciado. Por ejemplo, si tiene pensado enviar el XML resultante a alguien que no tiene acceso al esquema XML subyacente.

Convertir valores en tipos de destino: elija si los tipos del esquema XML de destino se deben usar durante la asignación (opción activada, predeterminada) o si todo los datos asignados al componente de destino se deben tratar como valores de **cadena**.

Desactive esta opción para conservar el formato exacto de los valores. Por ejemplo, para cumplir con la faceta `pattern` de un esquema que exige un número determinado de dígitos decimales en un valor numérico.

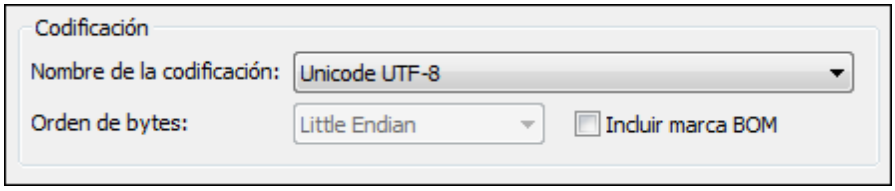
Puede usar funciones de asignación para aplicar el formato de cadena adecuado a un número y después asignar esta cadena al componente de destino.

Recuerde que si desactiva esta opción también se deshabilita la detección de valores no válidos (p. ej. escribir letras en campos numéricos).

Resultado pretty-print: reformatea su documento XML en el panel **Resultados** para presentarlo de manera estructurada y más fácil de leer. (A cada nodo secundario se le aplica una sangría.)

Crear firma digital (sólo para motor de ejecución integrado): las firmas digitales solamente se pueden generar si la opción seleccionada es **BUILTIN**. Para más información consulte el apartado Crear firmas digitales.

Codificación



A partir de la versión 2008 de MapForce cada componente, tanto de origen como de destino, tiene su propia codificación. Esto significa que los archivos de asignación `*.mfd` ya no tienen una codificación predeterminada, sino que cada componente de la asignación tiene una codificación propia. Cuando hablamos de componentes en sentido general nos referimos a componentes XML y de texto , así como componentes EDI y EDI.

No obstante, en la pestaña ***General*** del cuadro de diálogo "Opciones" (**Herramientas | Opciones**) hay un grupo de opciones llamado *Codificación predeterminada para componentes*

nuevos que se aplica cada vez que se crea/inserta un componente. Las opciones de codificación predeterminada para componentes nuevos son:

- *Nombre de la codificación*
- *Orden de bytes* (little endian o big endian)
- Casilla *Incluir marca BOM*

Las opciones activas por defecto son:

- UTF-8
- little endian (deshabilitada para UTF-8)
- sin marca BOM.

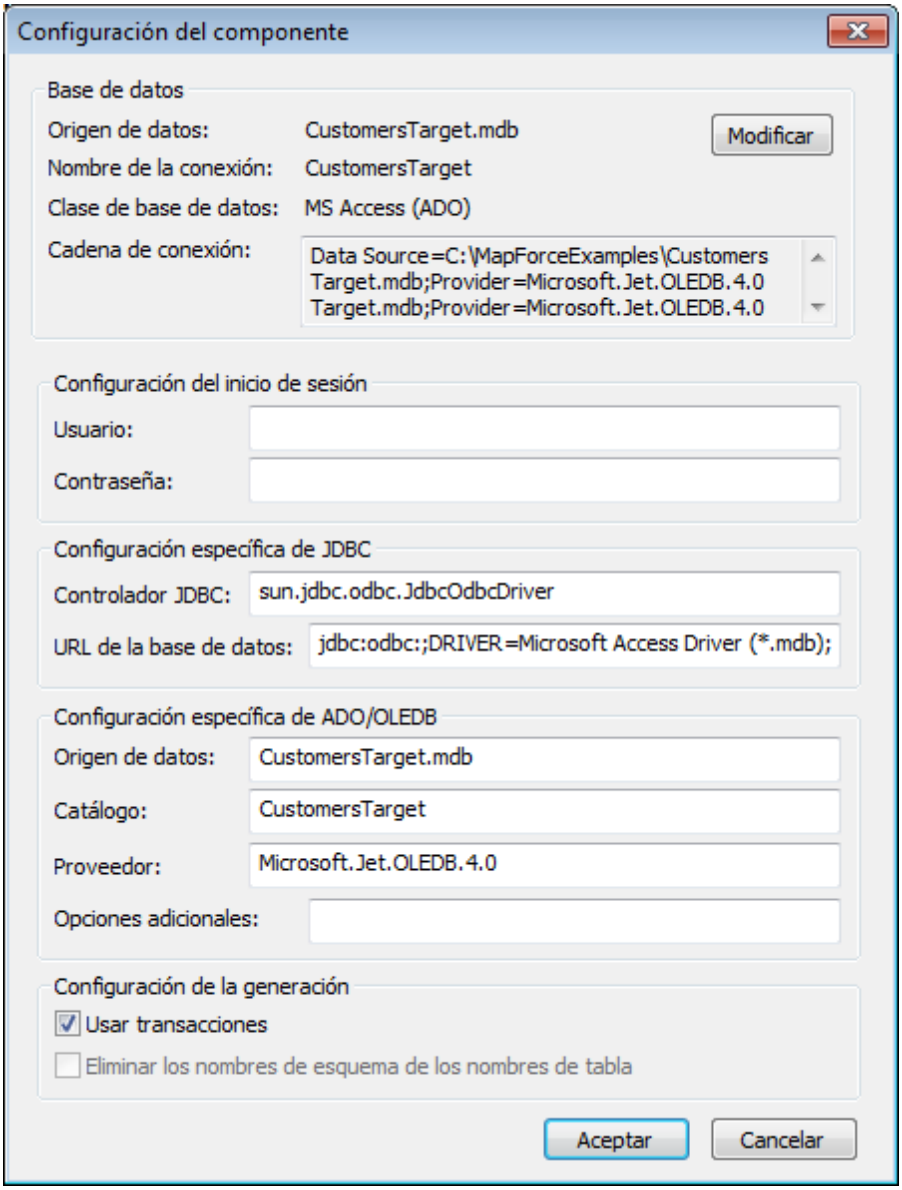
Nota: aunque la casilla *Incluir marca BOM* esté activa en el cuadro de diálogo "Configuración del componente", esto no tiene efecto alguno a la hora de generar código XSLT 1.0/2.0 porque estos lenguajes no admiten marcas BOM.

Optimización de procesamiento de datos de entrada basada en min/maxOccurs
A partir de la versión 2009 MapForce incluye una función de optimización para secuencias que contienen más de un elemento (p. ej. atributos necesarios o elementos secundarios con minOccurs/maxOccurs = 1. En casos como este, se extrae el primer elemento de la secuencia y se procesa directamente como valor atómico y no como secuencia.

Si los datos de entrada **no son válidos** con respecto al esquema, puede que se detecte una secuencia vacía, lo cual detiene la asignación y emite un mensaje de error. Para permitir el procesamiento de dichos datos de entrada no válidos, deshabilite esta función de optimización para los componentes EDI y XML (en el cuadro de diálogo "Configuración del componente".

El cuadro de diálogo "Configuración del componente" de los componentes de BD es diferente.

 © 2014 Altova GmbH



Base de datos

Origen de datos: el nombre del componente de base de datos. Haga clic en el botón **Modificar** para seleccionar otra BD o redefinir las tablas del componente. Se conservarán las conexiones a tablas que tengan el mismo nombre. También puede cambiar las tablas del componente haciendo clic con el botón secundario en el componente y seleccionando **Agregar, quitar o editar objetos...**.

Cadena de conexión: la cadena de conexión a la base de datos. Este campo no se puede editar.

Configuración del inicio de sesión

Usuario: el nombre de usuario necesario para acceder a la BD.

***Contraseña**:* la contraseña necesaria para acceder a la BD.

Estas opciones se utilizan para la generación de código en todos los lenguajes de destino y con el motor de ejecución integrado.

Configuración específica de JDBC:

Controlador JDBC: el controlador activo para el componente de BD. El controlador predeterminado se introduce automáticamente nada más definir el componente de BD. Si quiere, puede cambiar el controlador. No obstante, compruebe que la sintaxis utilizada en el campo *URL de la base de datos* es la adecuada para el controlador elegido. La configuración específica de JDBC solamente se utiliza para la generación de código Java.

URL de la base de datos: dirección URL de la BD. Compruebe que la URL concuerda con la sintaxis del controlador JDBC elegido en el campo *Controlador JDBC*.

Configuración específica de ADO/OLEDB:

Origen de datos: el nombre del origen de datos.

Proveedor: el proveedor activo para el componente de base de datos. El proveedor se introduce automáticamente nada más definir el componente de base de datos.

La configuración específica de ADO/OLEDB solamente se utiliza para la generación de código C++ y C#. Los datos de los campos *Origen de datos* y *Catálogo* no se usan con el motor de ejecución integrado.

Catálogo: el nombre de la BD.

Opciones adicionales: opciones adicionales de la BD.

Configuración de la generación:

Usar transacciones: permite el procesamiento de transacciones cuando se usa una BD como componente de destino. Cuando se detecta un error aparece un cuadro de diálogo donde puede elegir el siguiente paso. Active esta opción para habilitar el procesamiento de transacciones en todas las tablas del componente de BD. La configuración de la generación se aplica al código generado en todos los lenguajes y al motor de ejecución integrado.

Eliminar los nombres de esquema de los nombres de tabla: permite omitir los nombres de esquema de BD en el código generado y conservar los nombres de tabla solamente. Esta opción solamente funciona con las instrucciones SELECT de SQL generadas con MapForce. Las instrucciones SQL definidas por el usuario, durante la creación de tablas virtuales, no se modifican.

 © 2014 Altova GmbH

20.6 Conexión

Conectar automáticamente los secundarios equivalentes
Activa/desactiva la función "Conexión automática de secundarios equivalentes".

Configurar la conexión de secundarios equivalentes
Abre el cuadro de diálogo "Configurar la conexión de secundarios equivalentes" donde puede configurar la conexión.

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 (si está activo el icono **Conexión automática de elementos secundarios** de la barra de herramientas). Para más información consulte el apartado Propiedades de los conectores.

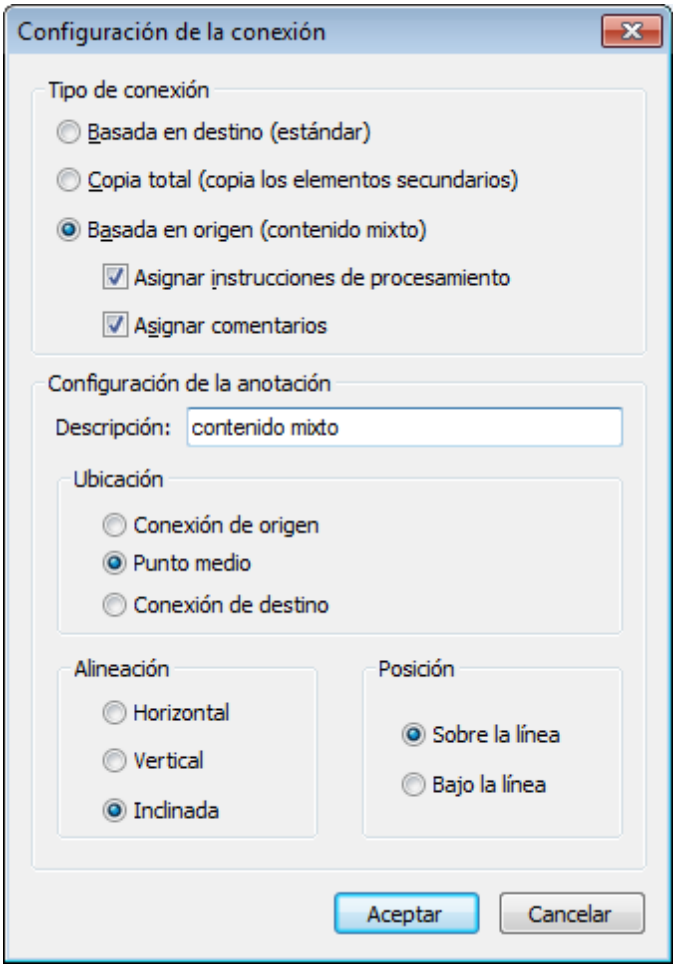
Basada en el destino (estándar)
Cambia el tipo de conector a una asignación estándar. Consulte el apartado Usar asignaciones estándar en elementos de contenido mixto para más información.

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. Para más información consulte el apartado Conexiones de copia total.

Basada en origen (contenido mixto)
Cambia el tipo de conector a una asignación basada en origen / de contenido mixto y permite seleccionar otros elementos que se deben asignar. Para ello los elementos adicionales deben ser elementos secundarios del elemento asignado en el archivo XML de origen. Para más información consulte el apartado Configuración predeterminada: asignar contenido mixto.

Propiedades:
Abre un cuadro de diálogo donde puede definir la configuración específica del conector actual. Las opciones no 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.



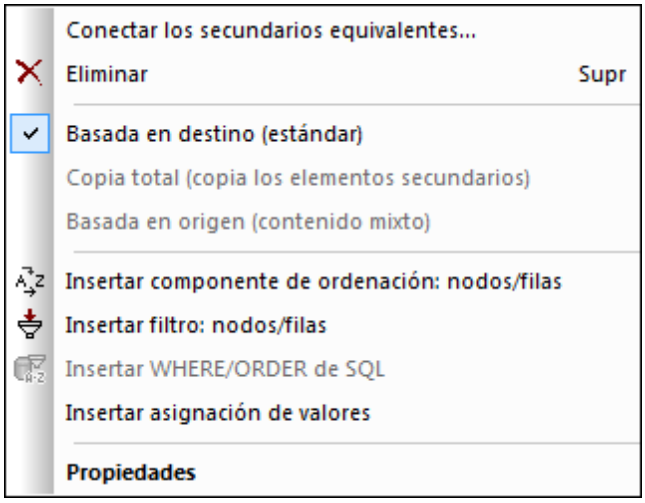
Configuración de la anotación:

Puede añadir una **etiqueta/anotación** a cada conector para una mayor claridad.

1. Haga doble clic en el conector y escriba el nombre del conector en el campo *Descripción*.
   Inmediatamente después se habilitan todas las opciones del grupo de opciones *Configuración de la anotación*.
2. Defina la posición y la alineación de la etiqueta.

Menú contextual de los conectores:

 © 2014 Altova GmbH



Conectar los secundarios equivalentes...
Abre el cuadro de diálogo "Conectar secundarios equivalentes", donde puede cambiar la configuración de la conexión automática de secundarios.

Eliminar
Elimina el conector seleccionado.

Basada en destino (estándar)
Cambia el tipo de conector a una asignación estándar. Consulte el apartado Usar asignaciones estándar en elementos de contenido mixto para más información.

Copia total (copia los elementos secundarios)
Cambia el tipo de conector a una conexión de copia total y conecta todos los secundarios que tengan el mismo nombre. Para más información consulte el apartado Conexiones de copia total.

Basada en origen (contenido mixto)
Cambia el tipo de conector a una asignación basada en origen / de contenido mixto. Para más información consulte el apartado Asignación basada en origen y de contenido mixto.

Insertar filtro: nodos/filas
Inserta un componente de filtrado en el conector. El conector de origen se conecta al parámetro **`nodes/row`** del filtro y el conector de destino se conecta al parámetro **`on-true`**. Para más información consulte el apartado Filtrar datos: recuperar datos dinámicos.

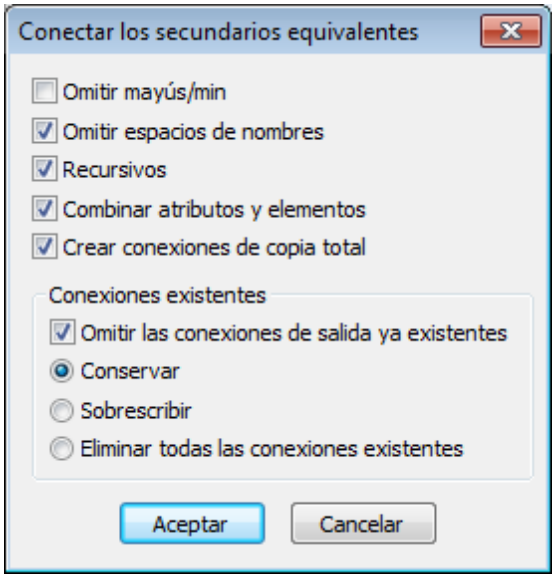
Insertar WHERE/ORDER de SQL
Inserta un componente WHERE de SQL en el conector. El conector de origen se conecta al parámetro de la tabla y el conector de destino se conecta al parámetro **`result`**. Para más información consulte Componente WHERE/ORDER de SQL.

Propiedades:
Abre el cuadro de diálogo "Configuración de la conexión", donde puede definir la configuración específica del conector actual y la anotación del conector (ver más arriba para más información).

Cuadro de diálogo "Conectar los secundarios equivalentes"
La conexión automática de secundarios equivalentes permite crear varios conectores entre los elementos que tienen el mismo nombre en el componente de origen y en el de destino. Las

conexiones de copia total se crean por defecto.

1. Conecte dos elementos (primarios) que tengan elementos secundarios con exactamente el mismo nombre.
2. Haga clic con el botón secundario en el conector de copia total y seleccione el comando **Conectar los secundarios equivalentes.**



3. En el cuadro de diálogo seleccione las opciones de configuración y haga clic en **Aceptar**.

   Se crean conectores para todos los secundarios que tengan exactamente el mismo nombre y que cumplan con los requisitos seleccionados en el cuadro de diálogo.

Nota: las opciones seleccionadas en este cuadro de diálogo se conservan y se aplican a todas las conexiones de copia total si está activo el icono **Conexión automática de secundarios** en la barra de herramientas. Para activar/desactivar el icono, simplemente haga clic en él.

Omitir mayús/min
No tiene en cuenta el uso de mayúsculas/minúsculas en el nombre de los secundarios.

Omitir espacios de nombres
No tiene en cuenta los espacios de nombres de los secundarios.

Recursivos
Tras crear el primer conjunto de conectores, se busca también en los elementos inferiores. Si existen elementos con el mismo nombre en niveles inferiores, también se crean conexiones entre ellos. Después se comparan los elementos secundarios de estos elementos inferiores y así sucesivamente.

Combinar atributos y elementos
Permite crear conectores entre los elementos que tienen el mismo nombre incluso si son de tipo diferente (p. ej. existen dos nodos llamados `Nombre`, pero uno es un elemento y el otro es un atributo). Marque esta casilla para crear conectores entre estos elementos.

Crear conexiones de copia total
Activada por defecto. Si es posible, se crean conexiones de copia total entre los elementos de

 © 2014 Altova GmbH

origen y destino.

Conexiones existentes:

Omitir las conexiones de salida ya existentes
Crea conexiones **adicionales** con otros componentes, incluso si los iconos de salida ya están conectados a otros elementos.

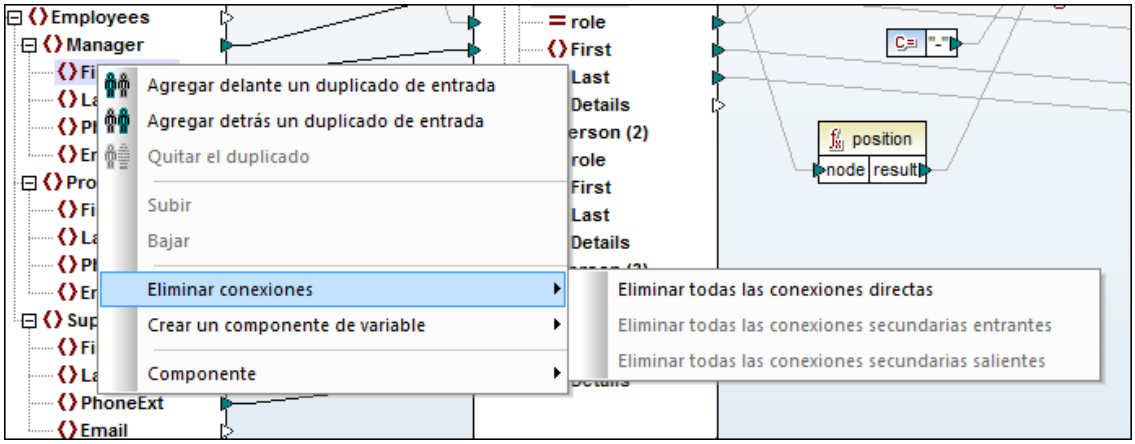
Conservar
Conserva las conexiones ya existentes.

Sobrescribir
Vuelve a crear los conectores en función de las opciones seleccionadas. Los conectores ya existentes se descartan.

Eliminar todas las conexiones existentes
Elimina todos los conectores ya existentes antes de crear conectores nuevos.

Cómo eliminar conexiones

Los conectores creados usando el cuadro de diálogo "Conectar los secundarios equivalentes" o durante el proceso de asignación se pueden eliminar todos de una vez.



Haga clic con el botón secundario en el nombre del elemento (no en el conector).
Seleccione **Eliminar conexiones | Eliminar todas las conexiones directas**

Eliminar todas las conexiones directas
Elimina todos los conectores con conexiones directas del componente actual con otros componentes de origen y destino.

Eliminar todas las conexiones secundarias entrantes
Solamente está disponible si hace clic con el botón secundario en un elemento del componente de destino. Elimina todos los conectores secundarios entrantes.

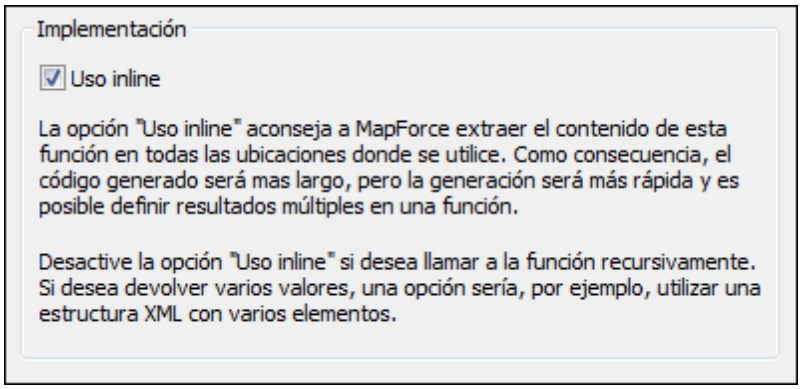
Eliminar todas las conexiones secundarias salientes
Solamente está disponible si hace clic con el botón secundario en un elemento del componente de origen. Elimina todos los conectores secundarios salientes.

20.7 Función

Crear una función definida por el usuario...
Crea una función definida por el usuario nueva donde puede insertar los componentes que necesite. Tras definir la función, en el panel de diseño de la función se inserta automáticamente un solo componente de salida. En una función definida por el usuario solamente puede haber un componente de salida, a no ser que la función se defina como inline. Para más información consulte Crear una función definida por el usuario desde cero.

Crear una función definida por el usuario a partir de la selección...
Crea una función definida por el usuario nueva a partir de los elementos seleccionados en el panel **Asignación**. Para más información consulte Agregar funciones definidas por el usuario.

Configuración de la función
Abre el cuadro de diálogo de configuración de la función definida por el usuario seleccionada. Utilice este cuadro de diálogo para cambiar el tipo de la función. Para abrir el cuadro de diálogo haga doble clic en la barra de título de la función definida por el usuario y después seleccione **Función | Configuración de la función**.



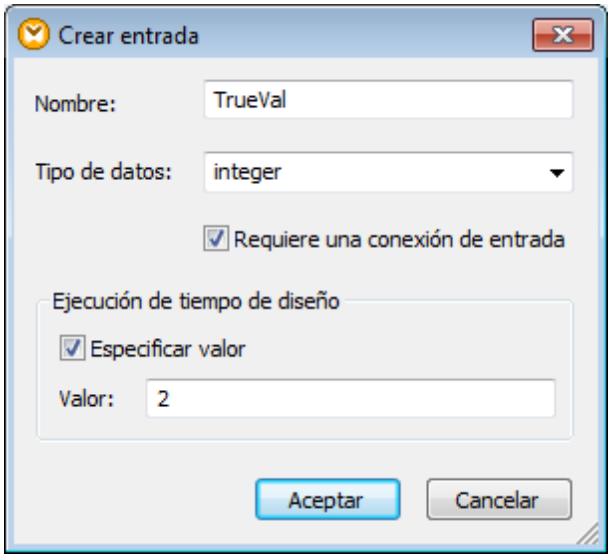
Quitar función
Elimina la función definida por el usuario activa. Haga doble clic en la barra de título de la función definida por el usuario y después seleccione **Función | Quitar función**. Es decir, este comando solamente está disponible mientras se edita la función definida por el usuario.

Antes de borrar la función, MapForce emite un aviso recordándole en qué biblioteca está la función e informándole de la posibilidad de que algunas instancias dejen de ser válidas.

Insertar componente de entrada
Inserta un componente de entrada en la asignación o en la función definida por el usuario.

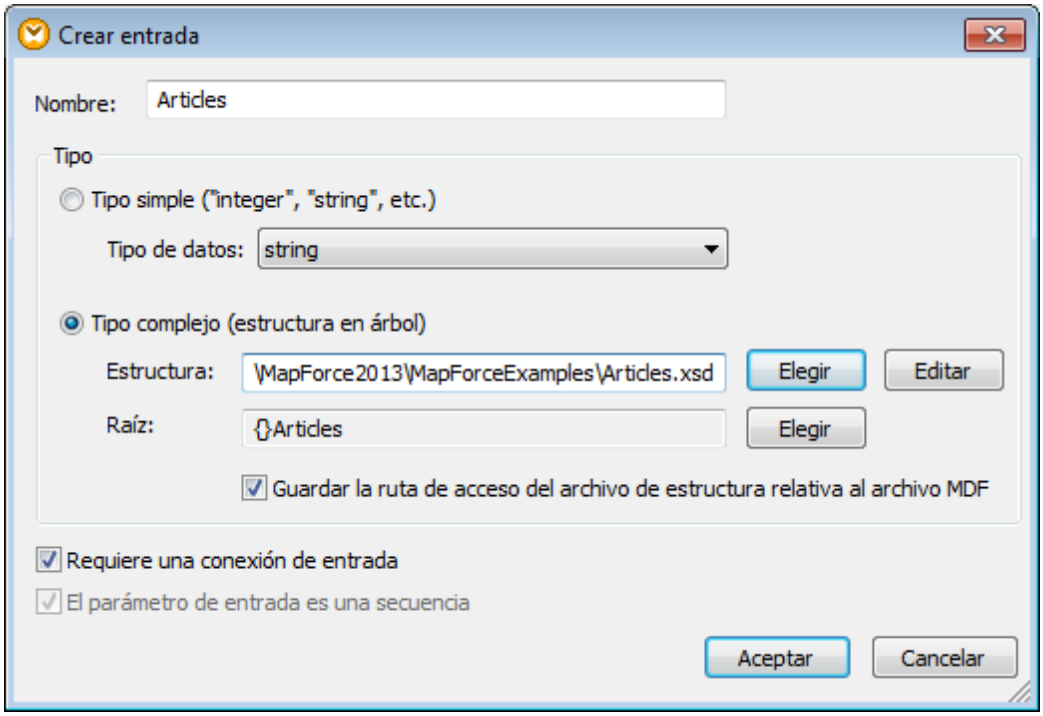
Si está trabajando en el panel **Asignación**, aparece el cuadro de diálogo de la siguiente imagen. Este tipo de componente de entrada sirve para definir un parámetro en la ejecución de la línea de comandos de la asignación compilada. Consulte Línea de comandos: definir parámetros de entrada para más información.

 © 2014 Altova GmbH



Si está trabajando en el panel de la función definida por el usuario, aparece el cuadro de diálogo de la siguiente imagen. Este tipo de componente sirve para definir:

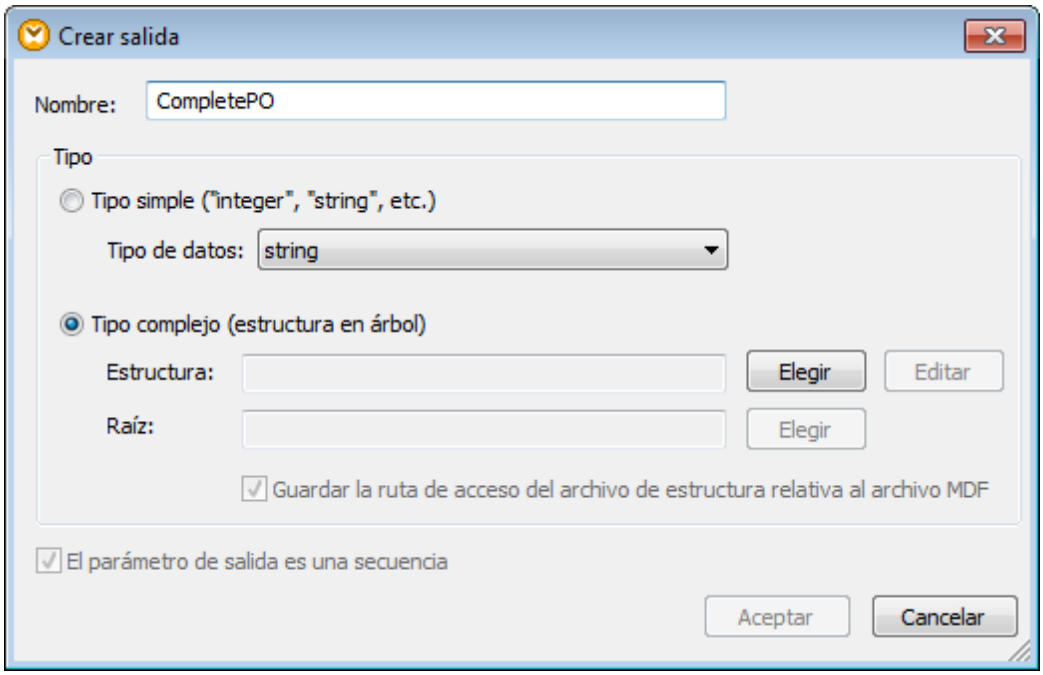
- entradas de tipo simple
- entradas de tipo complejo (p. ej. estructuras basadas en esquemas)



Insertar componente de salida
Inserta un componente de salida en una función definida por el usuario. Al hacer clic en este comando aparece el cuadro de diálogo de la siguiente imagen. Este tipo de componente sirve para definir:

- salidas de tipo simple

- salidas de tipo complejo (p. ej. estructuras basadas en esquemas)



 © 2014 Altova GmbH

20.8 Resultados

El primer grupo de opciones (**XSLT 1.0, XSLT 2.0, etc.**) sirve para definir el lenguaje de destino del código.

Recuerde que la opción **Motor de ejecución integrado** sirve para ver una vista previa de los resultados de la asignación en el panel **Resultados**, pero que no sirve para generar código de programa. Para más información consulte la sección Motor de ejecución integrado.

Validar archivo de salida
Valida el archivo XML de salida con el esquema al que se hace referencia.

Guardar el archivo de salida
Guarda los datos del panel **Resultados** en un archivo.

Guardar todos los archivos de salida
Guarda todos los archivos de salida generados para asignaciones dinámicas. Consulte el apartado Nombres de archivo dinámicos: entrada/salida para más información.

Volver a generar archivo de salida
Vuelve a generar la asignación actual desde el panel **Resultados**. Si en el panel **Resultados** está visible un script SQL, el script ejecuta la asignación a la BD de destino, teniendo en cuenta las acciones de tabla definidas.

Ejecutar script SQL
Ejecuta la asignación a la BD 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**.

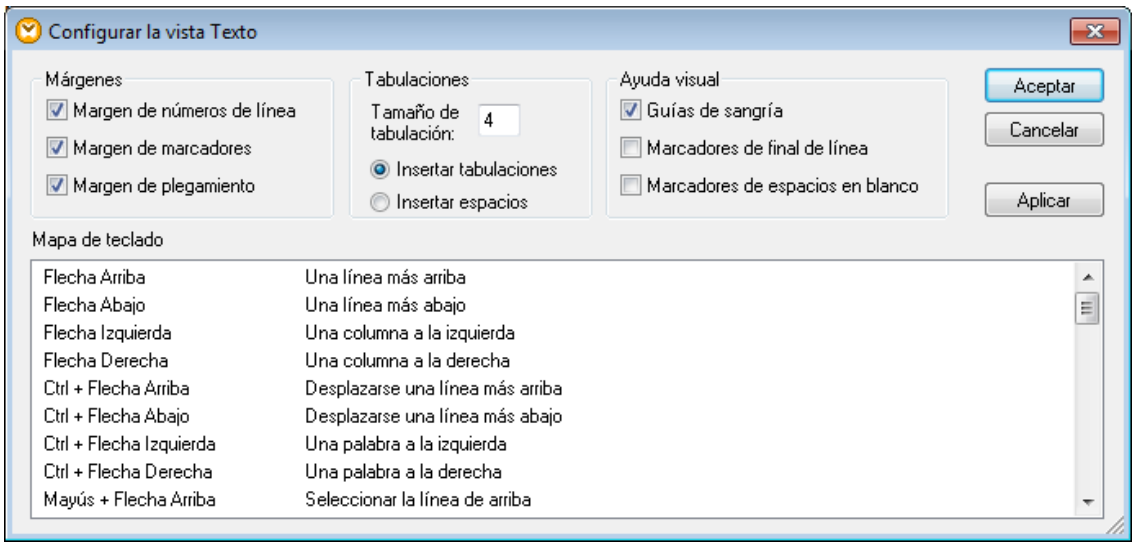
Marcador anterior
Navega hasta el marcador anterior en el panel **Resultados**.

Quitar todos los marcadores
Quita todos los marcadores 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* (*imagen siguiente*).

Configurar la vista Texto
Permite configurar el formato del texto en el panel **Resultados** y muestra las teclas de acceso rápido definidas para la vista.



Nota: las opciones *Insertar tabulaciones* e *Insertar espacios* no afectan inmediatamente al texto visible en el panel **Resultados**, sino que únicamente se aplican si activa la opción **Resultados | Texto XML pretty-print**.

 © 2014 Altova GmbH

20.9 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.

F7383	
tipo	string [0..1]
anot.	SURFACE OR LAYER CODE

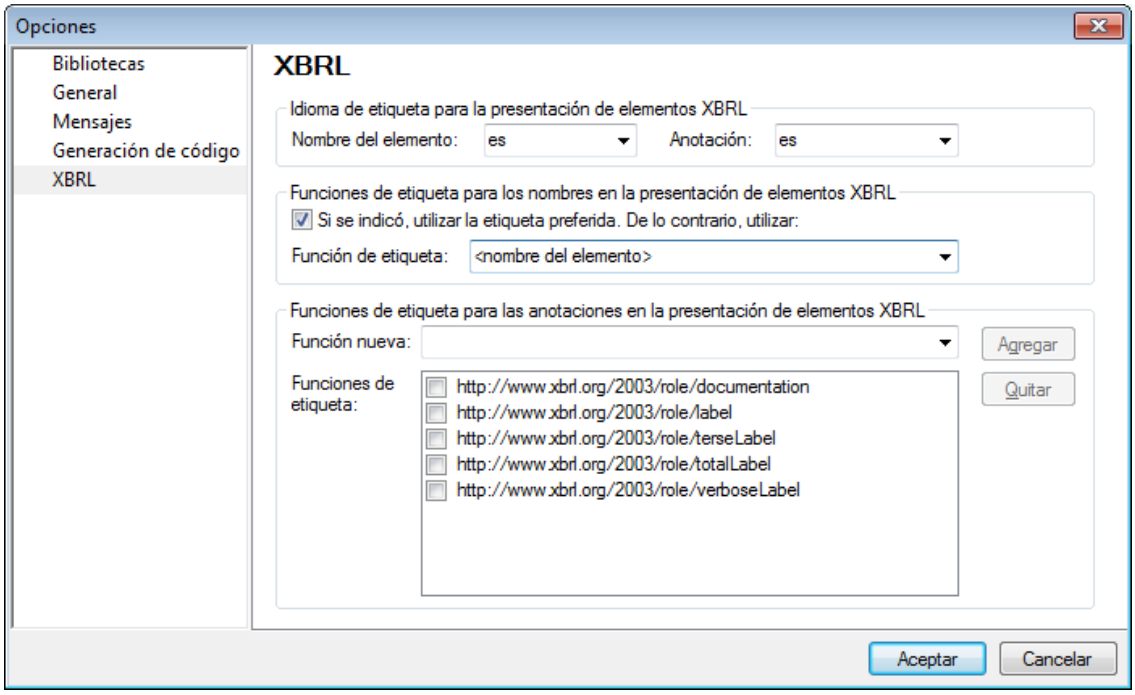
Mostrar tipos
Muestra el tipo de datos de cada elemento o atributo.
Si también se activa el icono **Mostrar anotaciones**, el tipo y la anotación aparecen en una tabla.

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 mouse por encima de la función.

Opciones de visualización de XBRL
Permite configurar la visualización de componentes XBRL:

- El idioma de la etiqueta para elementos XBRL y sus anotaciones
- Las funciones de etiqueta preferidas para los nombres de elemento XBRL
- El tipo de función de etiqueta de las anotaciones para elementos XBRL



Mostrar los conectores del componente seleccionado
Permite alternar entre una vista:

- con **todos los conectores** de la asignación
- y los **conectores relacionados** con los componentes seleccionados.

Mostrar los conectores desde su origen a su destino
Permite alternar entre una vista:

- con todos los conectores **directamente unidos** al componente seleccionado
- y 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

 © 2014 Altova GmbH

Muestra/oculta la barra de estado que está visible bajo la ventana Mensajes.

Ventana de 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.

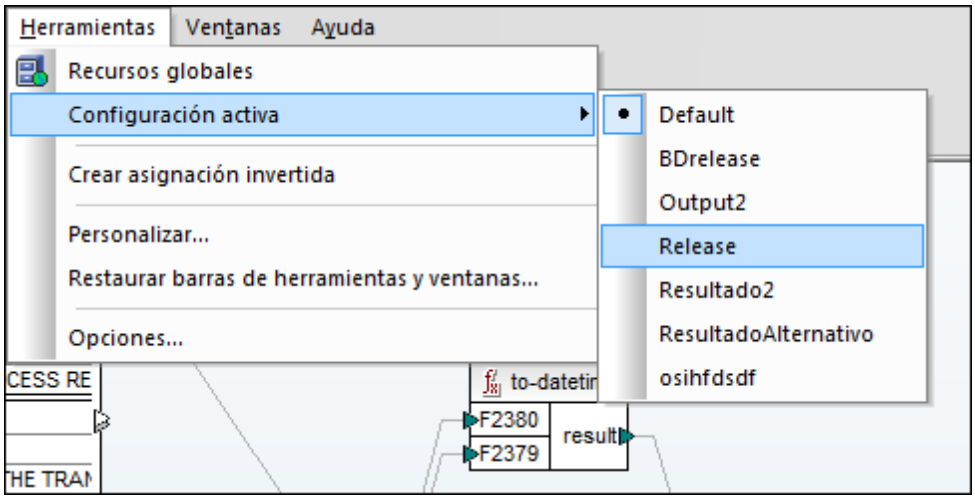
Vista general
Muestra/oculta la ventana Vista general. Arrastre el rectángulo rojo por la ventana de vista general para navegar por el panel **Asignación**.

Ventana de proyecto
Muestra/oculta la ventana Proyecto.

20.10 Herramientas

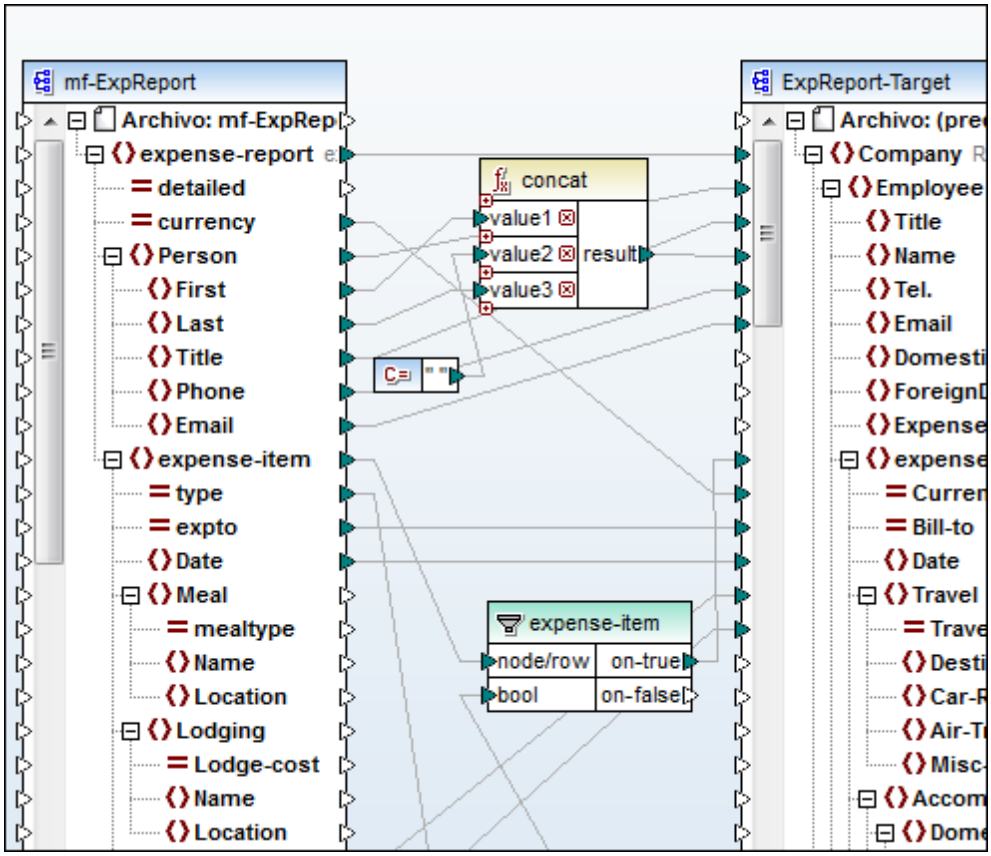
Recursos globales
Abre el cuadro de diálogo "Administrar recursos globales" donde puede agregar, editar o eliminar recursos globales en el archivo XML de recursos globales. Consulte el apartado Recursos globales: propiedades para más información.

Configuración activa
Permite seleccionar un recurso global o cambiar el recurso global activo desde una lista que enumera todos los recursos y configuraciones del archivo XML de recursos globales. Seleccione la configuración deseada en el submenú.

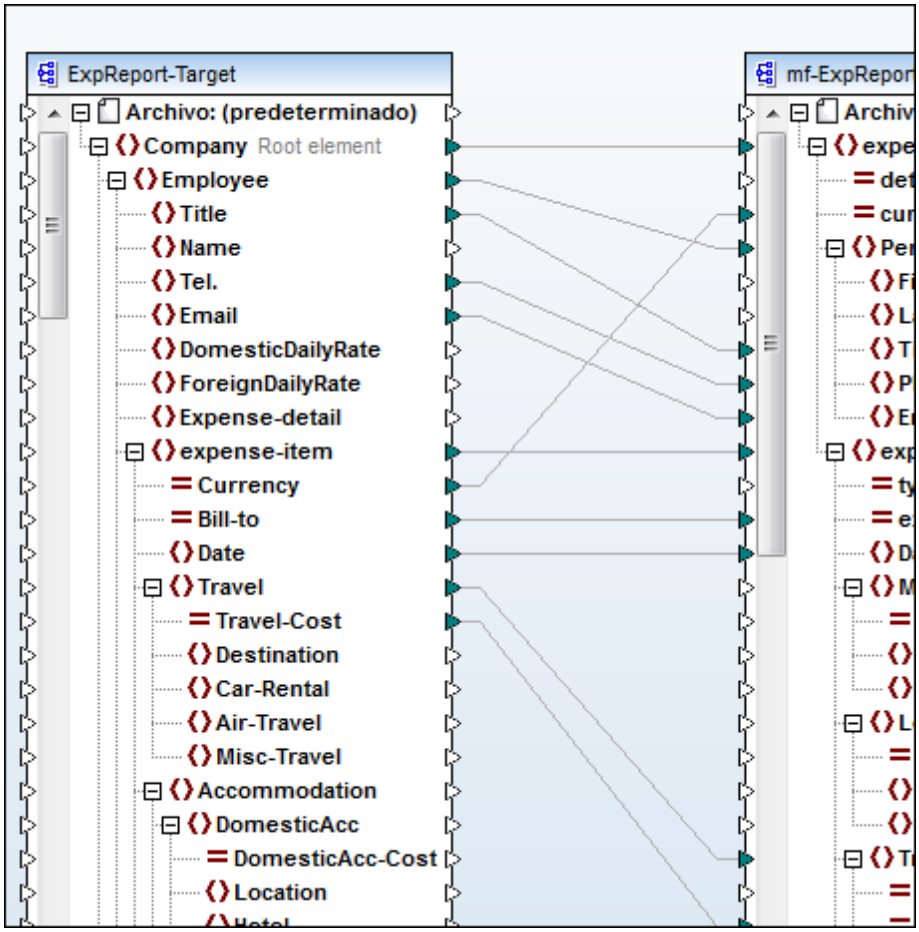


Crear asignación invertida
Crea una asignación invertida a partir de la asignación activa. Tenga en cuenta que el resultado no es una asignación completa porque en la asignación invertida solamente se conservan las conexiones directas entre los componentes de la asignación original. Es muy posible que la asignación resultante no sea válido o que no se pueda ejecutar (al abrir el panel **Resultados**) sin antes realizar ciertos cambios a manos.

P. ej. la asignación `Tut-ExpReport.mfd` de la carpeta `...\MapForceExamples\Tutorial`:



Asignación invertida:



Aspectos generales:

- el componente de origen se convierte en el componente de destino y viceversa.
- si se asignó un archivo XML de entrada y salida a uno de los componentes, se permutan.

Conexiones 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 no se cambian

Conexiones que se **eliminan**

- 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. Para ello es necesario reiniciar MapForce.

Personalizar...

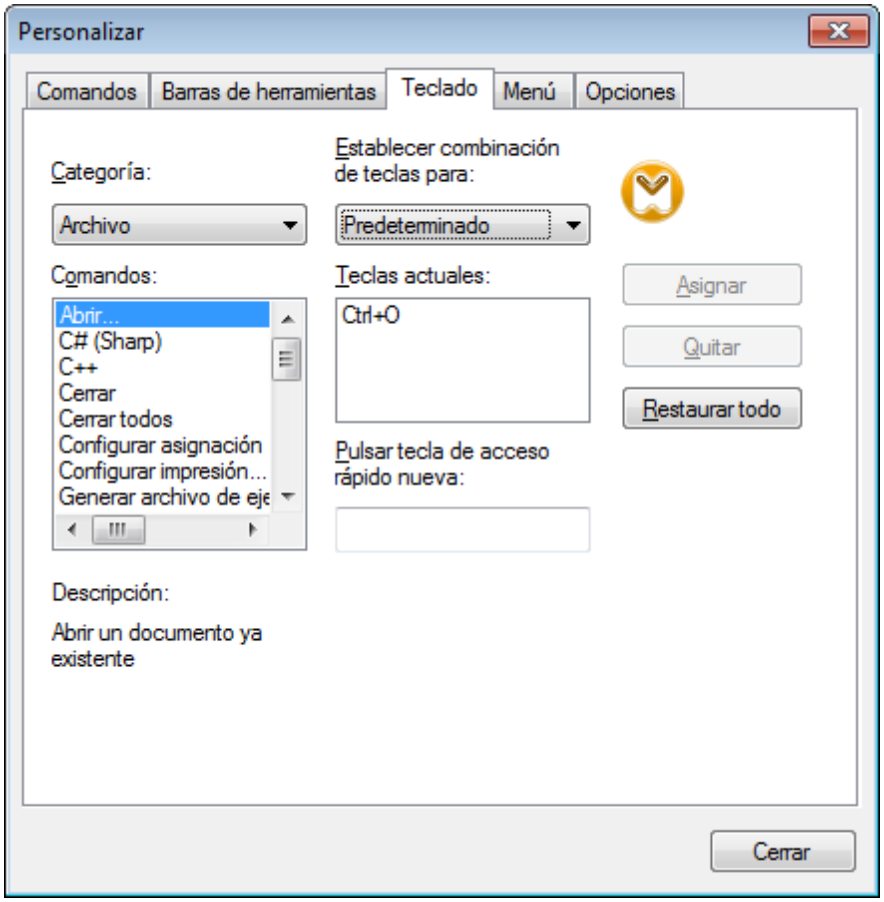
 © 2014 Altova GmbH

Este comando permite personalizar MapForce. Al hacer clic en el comando aparece el cuadro de diálogo "Personalizar", compuesto por varias pestañas.

La pestaña *Teclado* sirve para definir (o cambiar) teclas de acceso rápido para cualquier comando de MapForce.

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 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.

Establecer combinación de teclas para:
Este cuadro combinado no tiene por ahora ninguna función.

Teclas de acceso rápido asignadas actualmente:

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

 © 2014 Altova GmbH

Mayús + F2	Marcador anterior
CTRL + Mayús + F2	Quitar todos los marcadores

Teclas de acceso rápido de zoom

CTRL + rueda del mouse hacia adelante	Acercarse
CTRL + rueda del mouse hacia detrás	Alejarse
CTRL + 0 (cero)	Restablecer el nivel de zoom

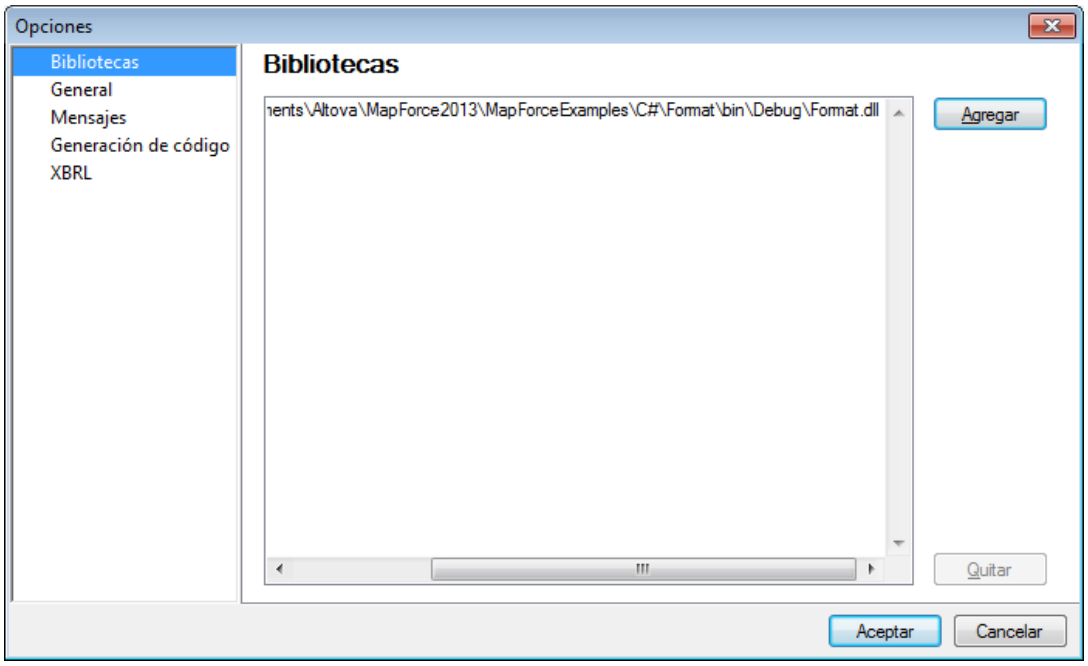
Opciones

Abre el cuadro de diálogo "Opciones" donde puede:

- Agregar o eliminar funciones XSLT o bibliotecas personales definidas por el usuario.
- Definir opciones de configuración generales, como la codificación predeterminada de los componentes nuevos (pestaña *General*).
- Definir qué tipo de mensajes desea recibir.
- Definir la configuración del compilador y de IDE.
- Definir la presentación de los componentes XBRL.

Pestaña *Bibliotecas*:

- Desde aquí puede agregar o eliminar funciones/bibliotecas XSLT definidas por el usuario o de otro lenguaje de programación.



Pestaña *General*:

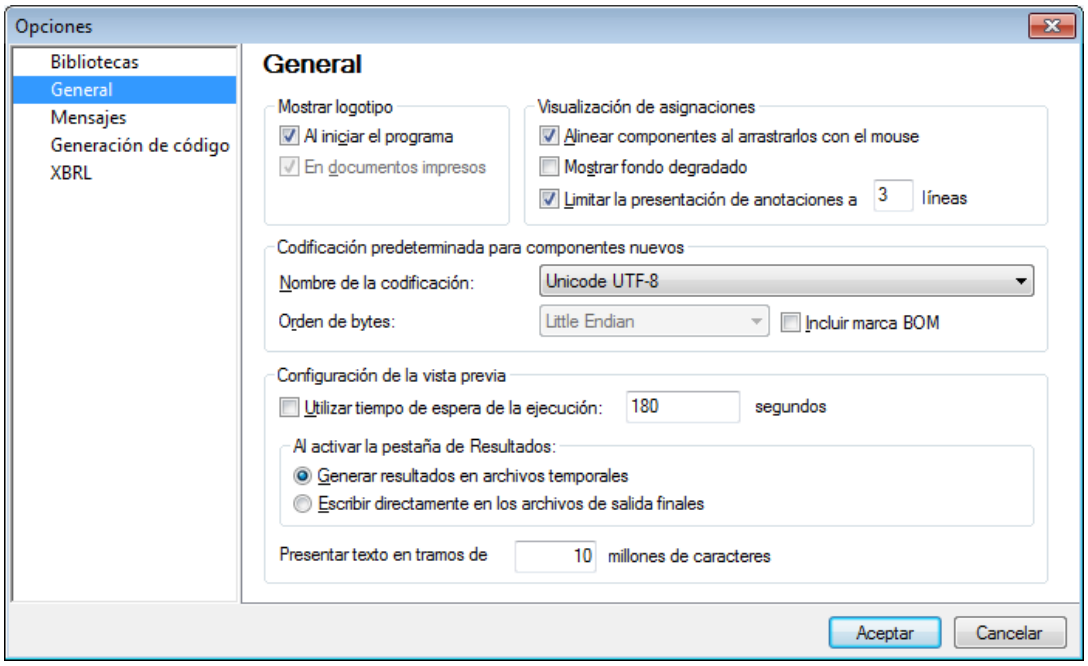
- Indique si quiere que aparezca el logo, el copyright, etc. al iniciar MapForce o al imprimir asignaciones.
- Indique si desea alinear los componentes y las funciones con los demás componentes

mientras los arrastra con el mouse.

- Habilite/deshabilite el color de fondo degradado de MapForce.
- Limite el texto de anotación de los componentes a un máximo de x líneas. También se aplica a las instrucciones SELECT visibles en el componente.
- Defina la codificación de caracteres predeterminada para componentes nuevos.
- Defina el tiempo de espera de la ejecución del panel **Resultados** cuando genere vistas previas de resultados.
- Indique si desea generar archivos **temporales** (opción activa por defecto) o escribir archivos de salida en el disco directamente al hacer clic en el botón/en la pestaña **Resultados**.

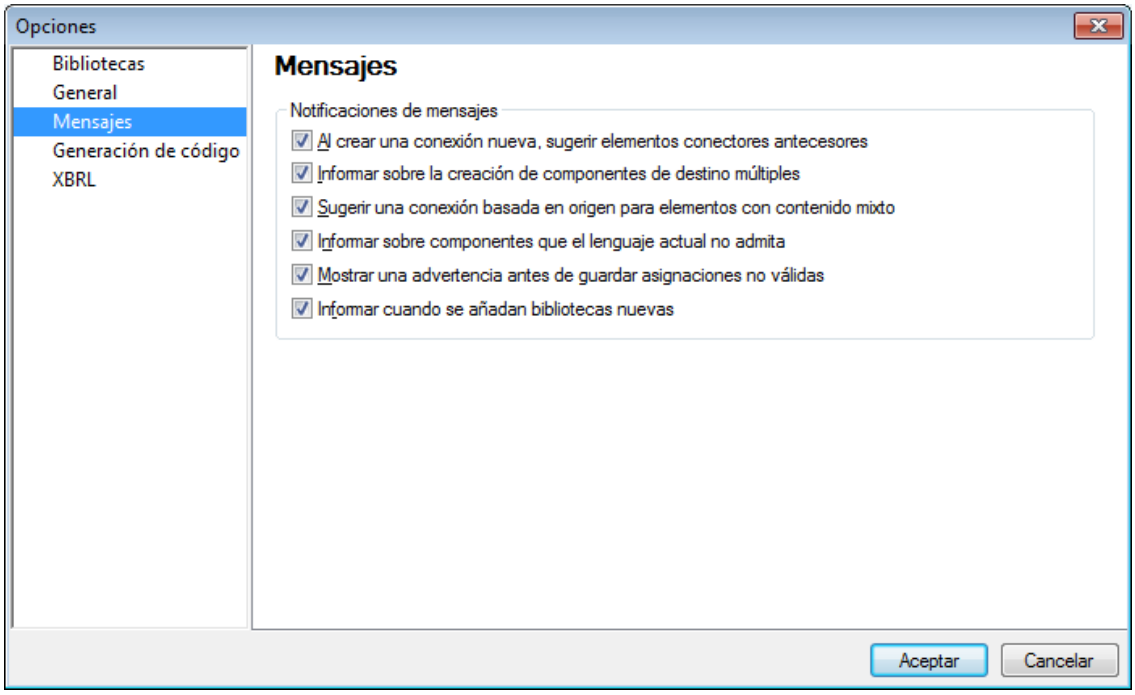
  Advertencia: habilite la opción *Escribir directamente en los archivos de salida finales* para sobrescribir los archivos de resultados sin solicitar confirmación previa.

- Limite el resultado a un máximo de x millones de caracteres cuando se creen resultados con el motor de ejecución integrado (el único lenguaje de salida compatible con secuencias de datos XML, CSV y FLF).



Pestaña *Mensajes*:
Permite restaurar los avisos y mensajes deshabilitados previamente al marcar la casilla "No volver a mostrar este mensaje" o "No volver a preguntarme".

 © 2014 Altova GmbH



Pestaña *Generación de código*:
En esta pestaña puede definir las opciones de configuración de su entorno IDE y de su compilador.

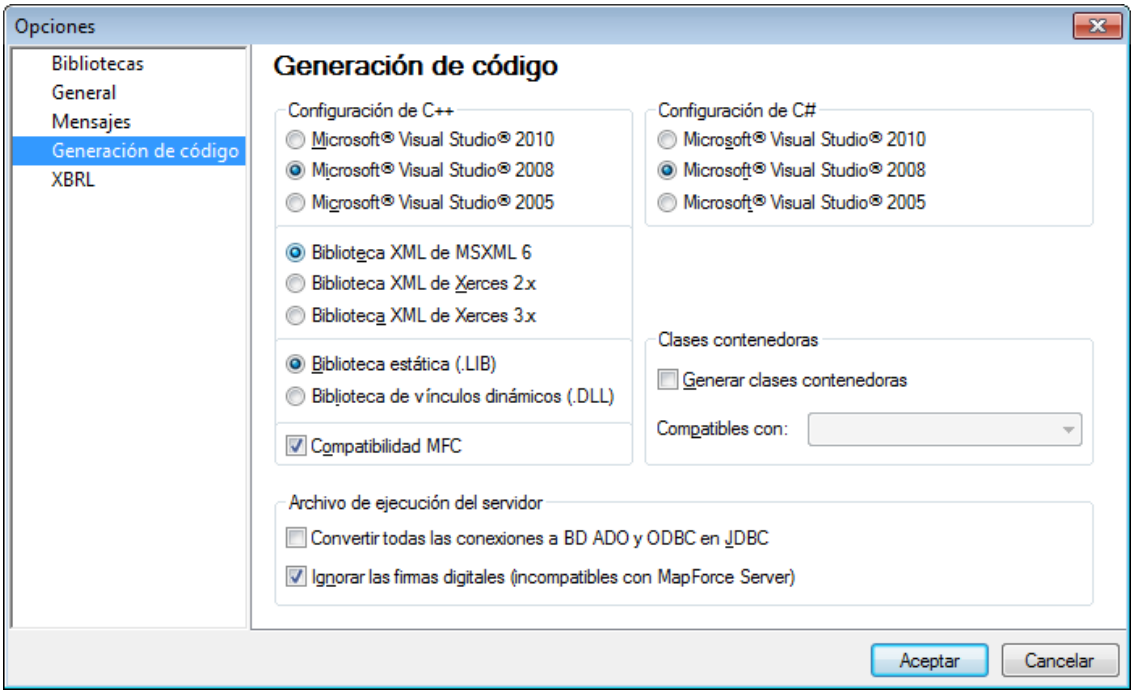
Configuración de C++:
En este grupo de opciones puede definir la configuración del compilador para el entorno C++.

Configuración de C#:
En este grupo de opciones puede definir la configuración del compilador para el entorno C#.

Clases contenedoras:
Este grupo de opciones sirve para generar clases contenedoras para esquemas XML. Las clases contenedoras generadas se pueden usar en el código que incluya el código generado con MapForce.

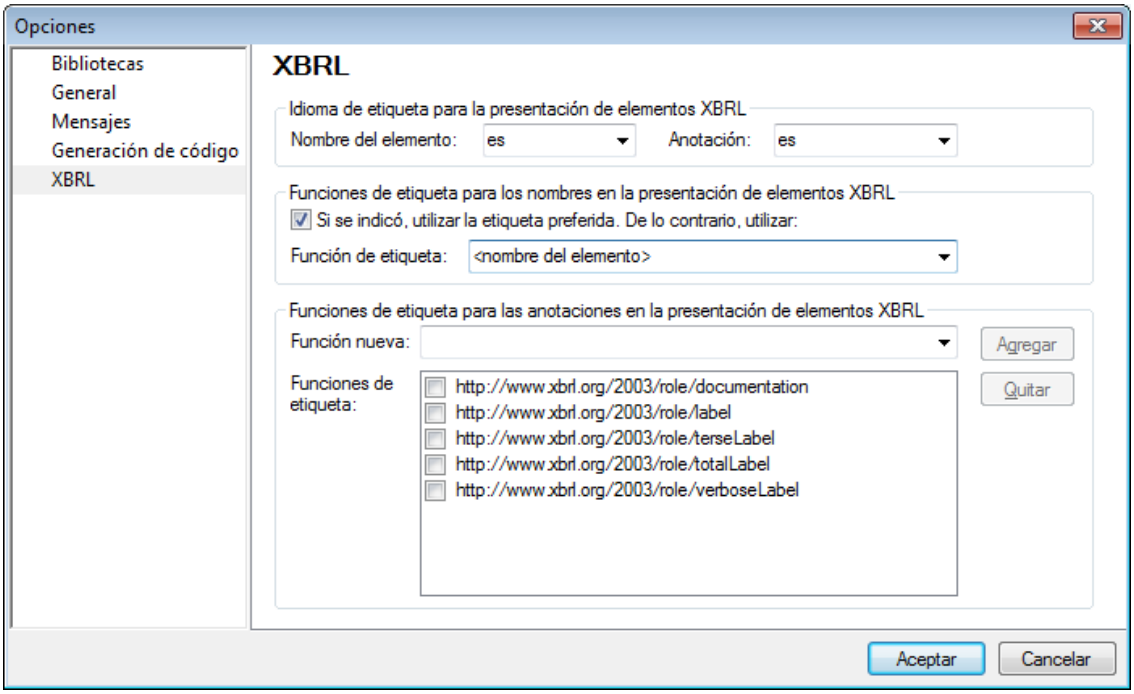
Archivo de ejecución del servidor:

- *Convertir todas las conexiones a BD ADO y ODBC en JDBC:* si marca esta casilla, durante la compilación de una asignación en un archivo de ejecución de MapForce Server, todas las conexiones a BD ADO y ODBC se transformarán en JDBC usando el controlador JDBC y la URL de BD definida en el cuadro de diálogo "Configuración del componente" de BD.
- *Ignorar las firmas digitales (incompatibles con MapForce Server)*: esta casilla está marcada por defecto para pasar por alto los datos de firmas digitales porque ninguna edición de MapForce Server es compatible con el uso de firmas digitales. Si desactiva esta casilla, los datos de firmas digitales se añadirán al archivo de ejecución de MapForce Server.



Pestaña *XBRL*:
En esta pestaña puede definir la presentación y visualización de los componentes XBRL:

- El idioma de la etiqueta para elementos XBRL y sus anotaciones
- Las funciones de etiqueta preferidas para los nombres de elemento XBRL
- El tipo de función de etiqueta de las anotaciones para elementos XBRL



 © 2014 Altova GmbH

20.11 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.

© 2014 Altova GmbH

20.12 Ayuda

El menú **Ayuda** contiene comandos con los que puede acceder a la ayuda en pantalla de MapForce, a la página de soporte técnico del sitio web de Altova, etc. El menú **Ayuda** también permite acceder al cuadro de diálogo de activación del software, donde puede introducir su código clave de licencia después de comprar el producto.

Los tres apartados siguientes describen todos los comandos del menú **Ayuda**:

- Contenido, Índice, Buscar
- Registro, Formulario de pedido
- Otros comandos

20.12.1 Contenido, índice, buscar

El comando **Ayuda | Contenido** abre la ayuda en pantalla de MapForce con la tabla de contenido en el panel izquierdo. Esta tabla de contenido ofrece una vista general de todo el documento de ayuda. Al hacer clic en una entrada del contenido se abre el tema correspondiente de la ayuda.

El comando **Ayuda | Índice** abre la ayuda en pantalla de MapForce con el índice de palabras clave en el panel izquierdo. Este índice temático enumera todas las palabras claves de la ayuda y permite navegar a un tema con solo hacer doble clic en la palabra clave correspondiente. Una palabra clave puede estar asociada a varios temas.

El comando **Ayuda | Buscar** abre la ayuda en pantalla de MapForce con la función de búsqueda en el panel izquierdo. Siga estos pasos para buscar un término en el documento de ayuda:

1. Escriba el término de búsqueda en el campo de consulta y pulse la tecla **Entrar**.
   La ayuda en pantalla muestra una lista de temas que contienen el término de búsqueda elegido.
2. Haga doble clic en uno de los temas de la lista para ver su contenido.

© 2014 Altova GmbH

20.12.2 Activación, formulario de pedido, registro, actualizaciones

Activación del software
Después de descargar el software de Altova puede activarlo con un código clave de prueba gratis o con un código clave de licencia permanente.

- ***Código de prueba gratis:*** Cuando inicie el software por primera vez, aparece el cuadro de diálogo "Activación del software". En este diálogo hay un botón para solicitar un código clave de prueba gratis. Escriba su nombre, el nombre de su compañía y su dirección de correo electrónico en el diálogo y después pulse el botón **Enviar solicitud**. Nuestro servidor envía un código clave de prueba a la dirección de correo electrónico proporcionada (que suele llegar en solo unos minutos). Copie y pegue el código clave de prueba en el campo *Código clave* y pulse el botón **Aceptar** para empezar a trabajar con el software de Altova. El software permanecerá desbloqueado 30 días.
- ***Código clave permanente:*** El cuadro de diálogo "Activación del software" incluye un botón para comprar un código clave permanente. Al hacer clic en este botón se abre la página de la tienda en línea de Altova. Hay dos tipos de licencias permanentes: licencias de un solo usuario o de varios usuarios. Nuestro servidor envía la licencia a su correo electrónico. La *licencia de un solo usuario* contiene los datos de la licencia e incluye su nombre, el de su compañía, su correo electrónico y el código clave. La *licencia de varios usuarios* contiene los datos de la licencia e incluye el nombre de su compañía y el código clave. Recuerde que, según lo dispuesto en el contrato de licencia, no puede instalar más del número máximo de copias del software de Altova en equipos de su organización (licencia por número de usuarios). Asegúrese de que los datos introducidos en el cuadro de diálogo de activación del software coinciden con los datos del correo electrónico que incluye la licencia.

  Nota: cuando introduzca los datos de la licencia en el cuadro de diálogo "Activación del software", compruebe que los datos introducidos coinciden con los datos del correo electrónico que incluye la licencia. En caso de usar una licencia de varios usuarios, cada usuario debe introducir su nombre en el campo *Nombre*.

Para abrir el cuadro de diálogo "Activación del software" haga clic en el comando **Ayuda | Activación del software**.

Formulario de pedido
Cuando desee comprar un código clave permanente, haga clic en el botón **Comprar un código clave permanente** del cuadro de diálogo "Activación del software" (*ver más arriba*) o seleccione el comando **Ayuda | Formulario de pedido**, que le llevará a la tienda en línea de Altova.

Registro del software
Al hacer clic en este comando se abre la página de registro del software en una pestaña del explorador web. Registro su software de Altova para recibir información sobre novedades del producto.

Buscar actualizaciones
Este comando busca si en el servidor de Altova hay una versión más reciente del producto.

20.12.3 Otros comandos

Centro de soporte técnico
Este comando 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
Este comando 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
Este comando 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.

MapForce en Internet
Este comando es un enlace al sitio web de Altova, donde encontrará más información sobre MapForce, otros productos de Altova y tecnologías relacionadas.

Cursos de MapForce
Este comando es un enlace a la página de cursos en línea del sitio web de Altova.

Acerca de MapForce
Este comando abre la pantalla de presentación de MapForce, que incluye el número de versión del producto e información sobre copyright.

© 2014 Altova GmbH

Altova MapForce 2014

Generador de código

21 Generador de código

MapForce incluye un generador de código integrado que puede generar archivos de clases Java, C++ y C# automáticamente a partir de definiciones de XML Schema, archivos de texo, bases de datos y archivos UN/EDIFACT y ASC X12.

En las asignaciones de datos de MapForce puede usar varias fuentes de datos y varios componentes de destino y puede utilizar todo tipo de de datos.

El resultado de la generación de código es una aplicación completa y totalmente funcional que realiza la asignación para usted. Puede ejecutar la aplicación tal y como se generó, insertar el código generado en su propia aplicación o ampliar la aplicación generada con funciones nuevas.

© 2014 Altova GmbH

21.1 Introducción al generador de código

Cuando se genera código a partir de esquemas XML, las plantillas predeterminadas del generador de código de MapForce generan automáticamente definiciones de clase que se corresponden con todos los elementos o tipos complejos declarados y que redefinen los tipos complejos de su esquema XML, conservando la derivación de clases tal y como se define en el esquema XML mediante extensiones de tipos complejos, así como todas las clases necesarias que ejecutan la asignación.

Si se trata de un esquema XML complejo que importa componentes de esquema de varios espacios de nombres, MapForce conserva esta información generando los espacios de nombre C# o C++ correspondientes o paquetes Java.

También se implementa código adicional, como funciones que leen archivos XML en una representación DOM (Document Object Model) en memoria, que escriben archivos XML de una representación DOM a un archivo de sistema o que convierten cadenas en árboles XML DOM y viceversa.

El código de programa de salida se expresa en los lenguajes de programación C++, C# o Java.

Lenguaje de destino	**C++**	**C#**	**Java**
Entornos de desarrollo	Microsoft Visual C++ 2010 Microsoft Visual C++ 2008 Microsoft Visual C++ 2005	Microsoft Visual Studio 2010 Microsoft Visual Studio 2008 Microsoft Visual Studio 2005	Apache Ant (archivo build.xml) Eclipse 3.4+ Borland JBuilder
Implementaciones XML DOM	MSXML 6.0 o Apache Xerces 2.6 o superior	System.Xml	JAXP
API de bases de datos (sólo en MapForce)	ADO	ADO.NET	JDBC

C++
El código C++ generado utiliza MSXML 6.0 o Apache Xerces 2.6 (o superior). Tanto MapForce como XMLSpy generan directamente archivos de proyecto y de solución/área de trabajo para Visual C++ o Visual Studio 2005-2010. Para Visual Studio se generan archivos **.sln** y **.vcproj**. El código generado también puede ser compatible con el conjunto de clases MFC, por ejemplo mediante métodos de conversión para las CString generadas.

Nota importante:
Cuando genere código C++ para Visual Studio 2005/2008/2010 y use una biblioteca Xerces compilada previamente para Visual C++, es necesario cambiar una opción de configuración del compilador en todos los proyectos de la aplicación:

1. Seleccione todos los proyectos en el explorador de la aplicación.
2. [Project] | [Properties] | [Configuration Properties] | [C/C++] | [Language]
3. Seleccione *All Configurations*

4. Cambie **Treat wchar_t** como **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 (p. ej. VB.NET, Managed C++, J# o cualquier otro lenguaje dirigido a la plataforma .NET). Se pueden generar archivos de proyecto para Visual Studio 2005, 2008 y 2010.

Java
El código Java generado se escribe de acuerdo con la API de Java estándar para XML Parsing (JAXP) e incluye un archivo de proyecto JBuilder, un archivo de compilación Ant y archivos de proyecto para Eclipse. La generación de código es compatible con Java 6 o superior.

Código de salida generado con MapForce:

Código de salida generado	Ubicación	MapForce
Bibliotecas estándar	Carpeta `Altova`	☑
Bibliotecas de contenedoras de esquema	Carpeta `nombreDeEsquema`	☑
Bibliotecas de información sobre el tipo de base de datos	Carpeta `nombreDeBD`	☑
Bibliotecas de información sobre el tipo de EDI	Carpeta `nombreDeMensajeEDI`	☑
Aplicación		
Aplicación de asignación (aplicación completa)		☑
Compilación y ejecución, ejecuta la asignación definida.		☑
La aplicación de asginación se puede:	ampliar según los requisitos del usuario	
	importar en una aplicación del usuario	☑

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).

 © 2014 Altova GmbH

21.2 Novedades

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

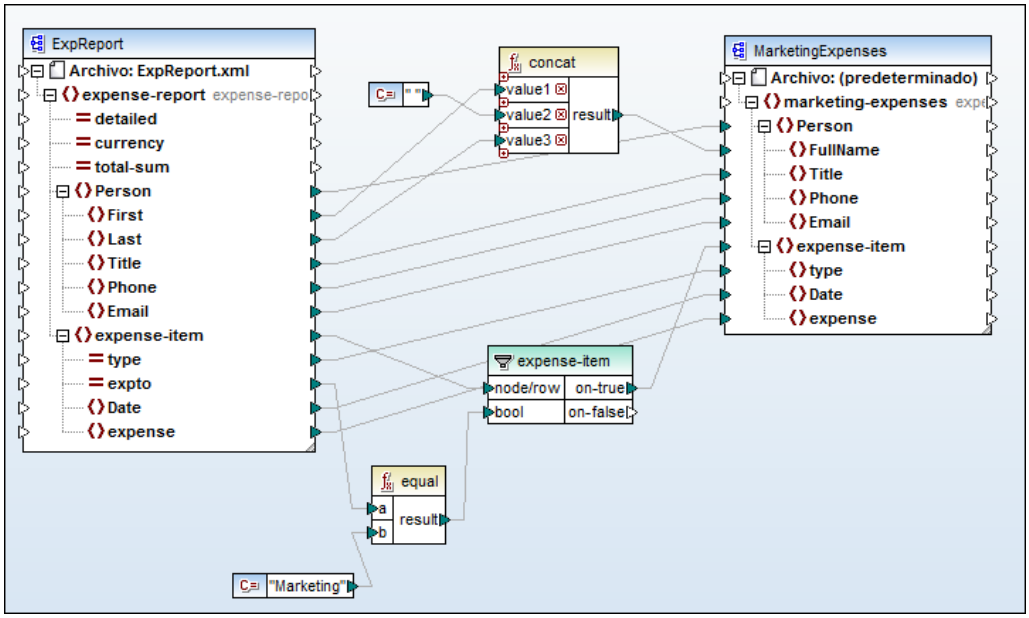
- Sintaxis nueva para evitar colisiones de nombres
- Tipos de datos nuevos para un uso más sencillo y un mayor rendimiento (uso de tipos nativos cuando sea posible, control de valores null...)
- Los atributos ya no se generan como colecciones
- El contenido de elementos simples ahora también se considera 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

© 2014 Altova GmbH

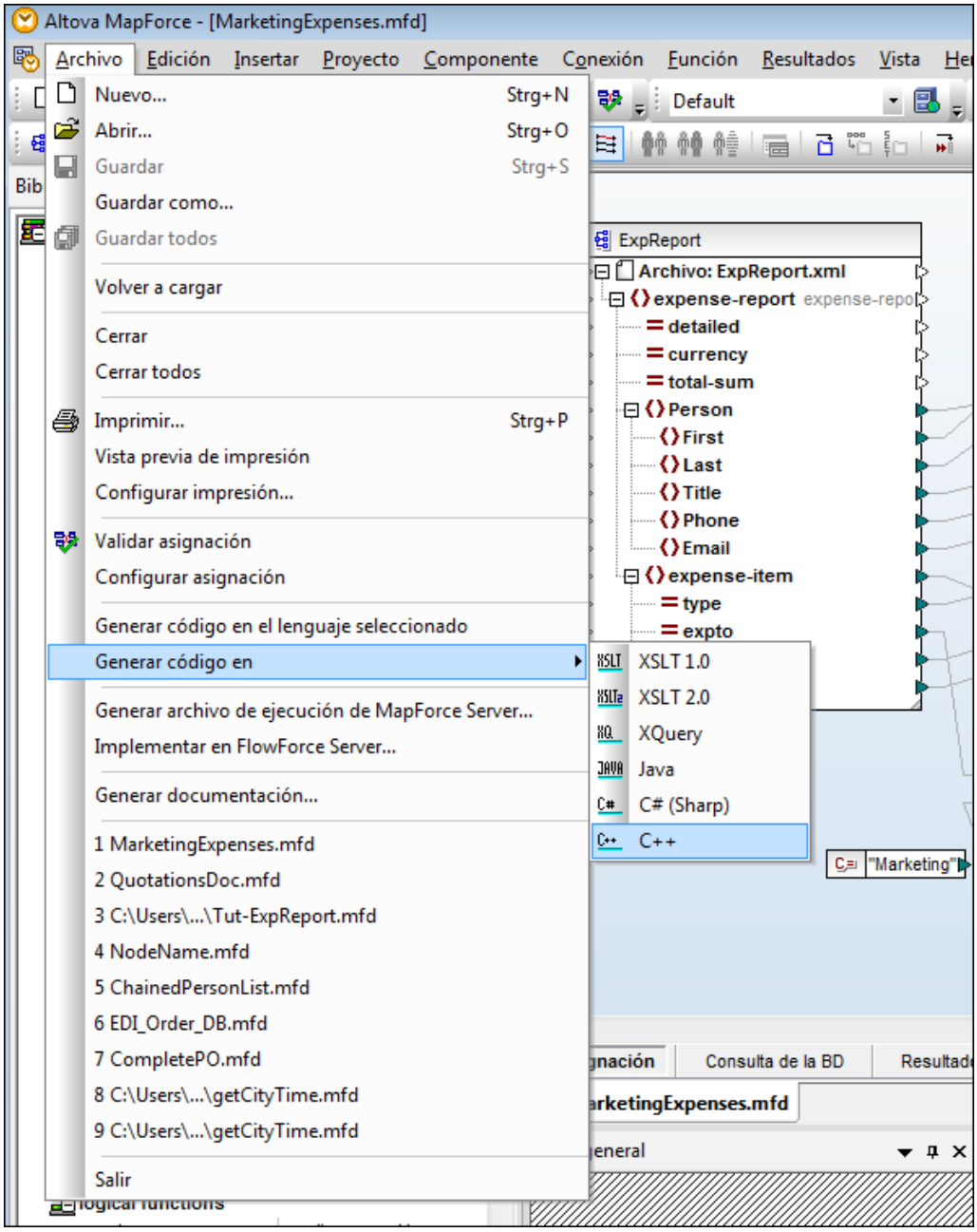
21.3 Generar código de programa

Este ejemplo muestra el orden general que se debe seguir para generar código de programa desde MapForce. Para este ejemplo utilizamos el archivo de asignación `MarketingExpenses.mfd` disponible en la carpeta `...\MapForceExamples`. También puede consultar el apartado Integrar código de MapForce en su aplicación para obtener más información.

1. Abra el archivo de asignación `MarketingExpenses.mfd`.



2. Seleccione la opción de menú **Archivo | Generar código en | C++**.



Aparece el cuadro de diálogo "Buscar carpeta".
3. Navegue hasta la carpeta donde desea guardar el código y haga clic en **Aceptar** para confirmar. Aparece el mensaje *El código se generó correctamente*.
4. El código generado se coloca en subcarpetas dentro del directorio especificado y contiene todas las bibliotecas necesarias para compilar y ejecutar el código de asignación.

Las secciones siguientes ofrecen más información sobre el proceso de generación y compilación de código en cada lenguaje de programación:

Generar código Java
Generar código C#

Generar código C++

21.3.1 Generar código Java

Requisitos y configuración predeterminada:

La aplicación generada por MapForce es compatible con Java 6 (o superior).

Para asignar datos de BD y compilar código Java es necesario tener instalados **controladores JDBC**. Para más información consulte el apartado Conexiones JDBC.

Haga clic en **Archivo | Configurar asignación** para definir la configuración del proyecto de asignación. La configuración predeterminada es:

- Nombre de la aplicación=Mapping
- Nombre del paquete base=com.mapforce

Para ver la configuración específica de JDBC, haga clic en el componente de base de datos y seleccione la opción de menú **Componente | Propiedades**.

En esta tabla puede ver las aplicaciones que se pueden compilar en cada entorno:

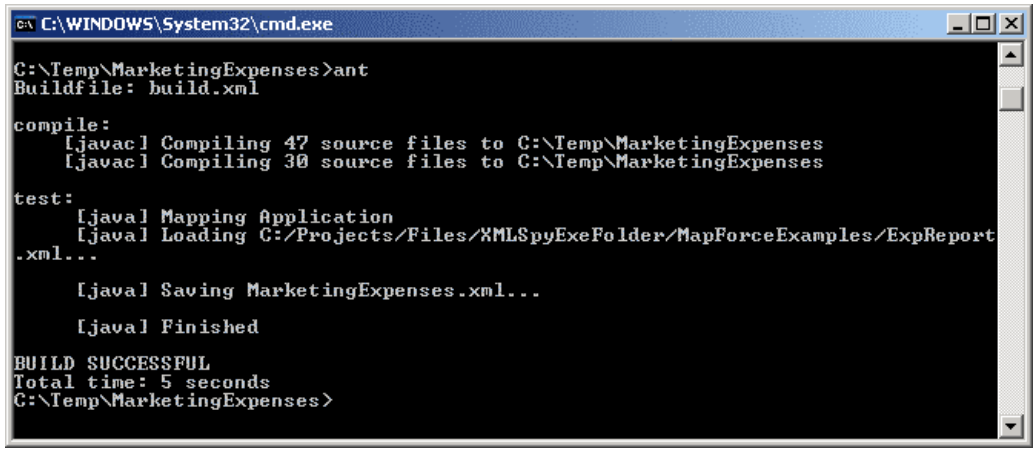
Nombre del archivo (con el nombre de aplicación predeterminado)	Nota
MappingConsole.java	Aplicación de consola
MappingApplication.java	Aplicación de diálogo

Para generar código Java en MapForce:

1. Seleccione la opción de menú **Archivo | Generar código en | Java**.
   MapForce solicita el directorio de destino para los archivos generados.
2. Seleccione el directorio donde desea guardar los archivos (p. ej. `c:\codegen\Java`) y haga clic en **Aceptar**.
   Si el proceso finaliza correctamente, aparece el mensaje *El código se generó correctamente*. De lo contrario aparece un mensaje de error que describe el problema en detalle.

Para generar código con un script de generación Ant:

- Navegue hasta el subdirectorio Java y ejecute el script `ant` (que abre automáticamente el archivo `build.xml`)
- Esto compila y ejecuta el código Java. El archivo XML de destino se genera automáticamente al final de esta secuencia.



Nota: si ejecuta el script `ant` sin ningún parámetro (como en las instrucciones), el código se

 © 2014 Altova GmbH

compila pero no se crea ningún archivo JAR. Para generar un archivo JAR con un script `ant`, utilice el comando `ant jar`.

Cuando genere archivos JAR a partir de código Java generador previamente, tenga en cuenta que las entradas de classpath para el controlador de BD no se añaden automáticamente al archivo de manifiesto. Primero es necesario editar el archivo `build.xml` (que se encarga de generar el archivo de manifiesto. Por ejemplo:

Para editar el archivo **`build.xml`**:

1. El primer paso es añadir al archivo `build.xml` esta línea (que contiene el nombre de atributo `="Class-Path"`):

   Por ejemplo, para MySQL usaríamos:

   ```
   <attribute name="Class-Path" value="mysql-connector-java-5.1.16-bin.jar"/>
   ```

   El elemento `manifest` del archivo `build.xml` sería este:

   ```
   <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. Después debe copiar el archivo JAR del **controlador** JDBC a la carpeta que contiene el archivo JAR de la aplicación generada.

Para generar código a mano:

Debe editar el archivo `MANIFEST.MF` manualmente, añadiendo el archivo JAR externo del controlador de BD a classpath. Por ejemplo, anexe esta línea al archivo de manifiesto:

```
Class-Path: mysql-connector-java-5.1.16-bin.jar
```

21.3.2 Generar código C#

Requisitos y configuración predeterminada:

Haga clic en **Archivo | Configurar asignación** para definir las opciones de configuración de la asignación. Las opciones predeterminadas son:

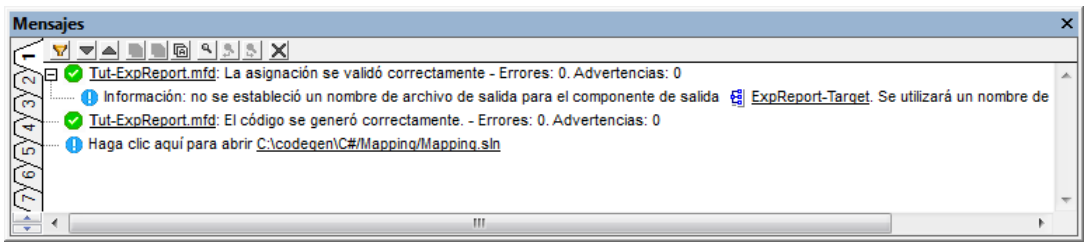
- Nombre de la aplicación=Mapping

Para ver la configuración específica para bases de datos, haga clic en el componente de base de datos y seleccione la opción de menú **Componente | Propiedades**.

MapForce genera archivos **.sln** y **csproj** para Microsoft Visual Studio. La pestaña *Generación de código* del cuadro de diálogo "Opciones" (**Herramientas | Opciones)** permite seleccionar la versión del IDE de destino. Los usuarios de Mono pueden usar archivos de solución de Visual Studio con xbuild.

Para generar código C# en MapForce:

1. Seleccione la opción de menú **Archivo | Generar código en | C# (Sharp)**.
   MapForce solicita el directorio de destino para los archivos generados.
2. Seleccione el directorio donde desea guardar los archivos (p. ej. `c:\codegen\C#`) y haga clic en **Aceptar**.
   Si el proceso finaliza correctamente, aparece el mensaje *El código se generó correctamente*. De lo contrario aparece un mensaje de error que describe el problema en detalle.

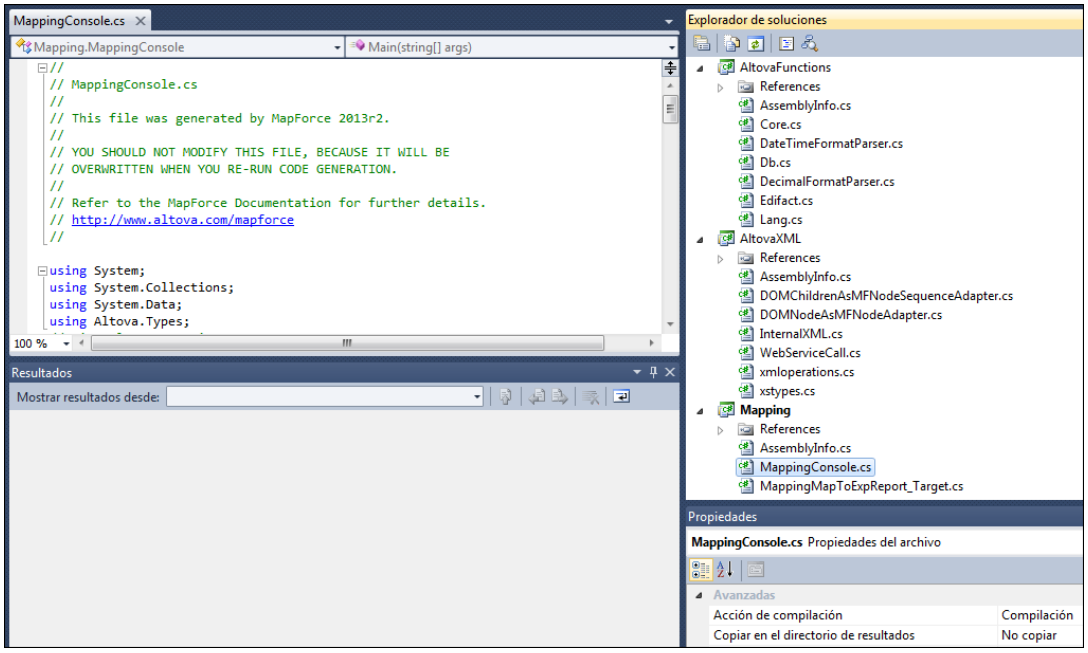


Carpeta c:\codegen\C#\mapping
Ahora puede compilar el proyecto en Microsoft Visual Studio.

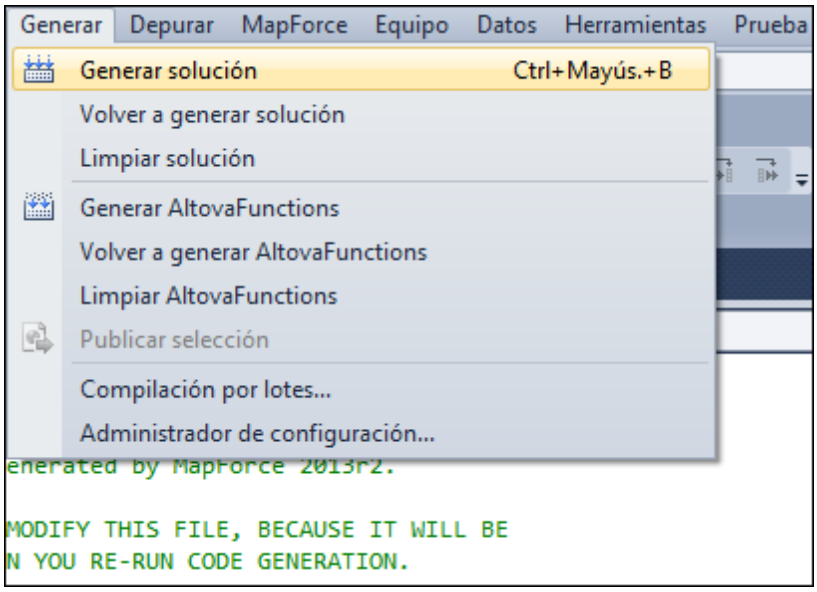
- `Mapping.sln` (para Visual Studio)

1. En la ventana Mensajes de MapForce haga doble clic en el hipervínculo *Haga clic aquí para abrir* `C:\codegen\c#/Mapping/Mapping.sln` para abrir el archivo de solución. Este hipervínculo aparece en la ventana Mensajes cuando se genera código C# y C++. Para el código Java deberá navegar manualmente hasta la carpeta de destino.

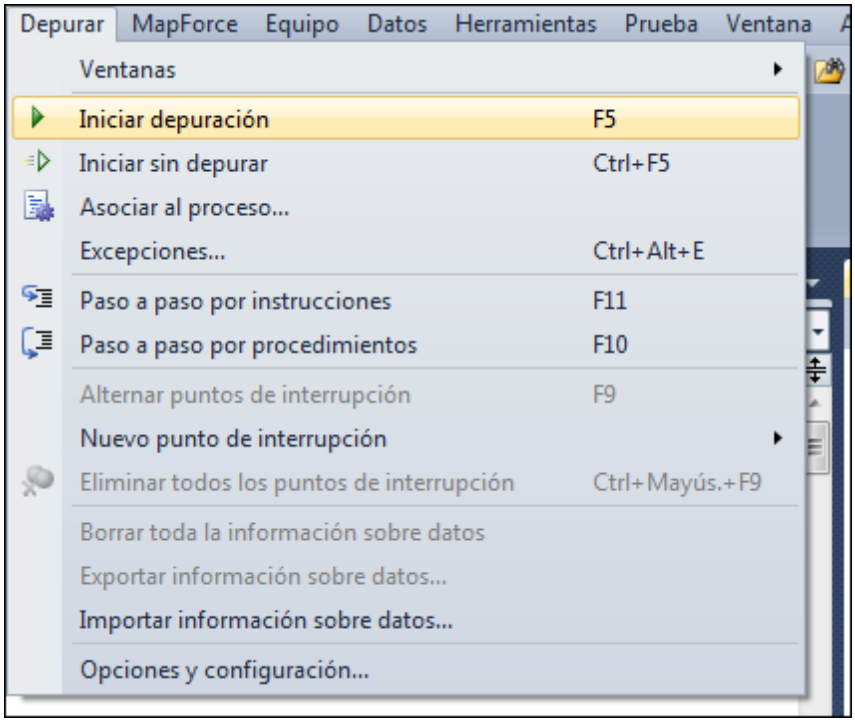
 © 2014 Altova GmbH



2. Seleccione la opción de menú **Generar | Generar solución** para compilar el proyecto de asignación.



3. Seleccione la opción de menú **Depurar | Iniciar depuración** para iniciar la aplicación.



La aplicación de asignación se inicia y se crea el archivo XML de destino.

 © 2014 Altova GmbH

21.3.3 Generar código C++

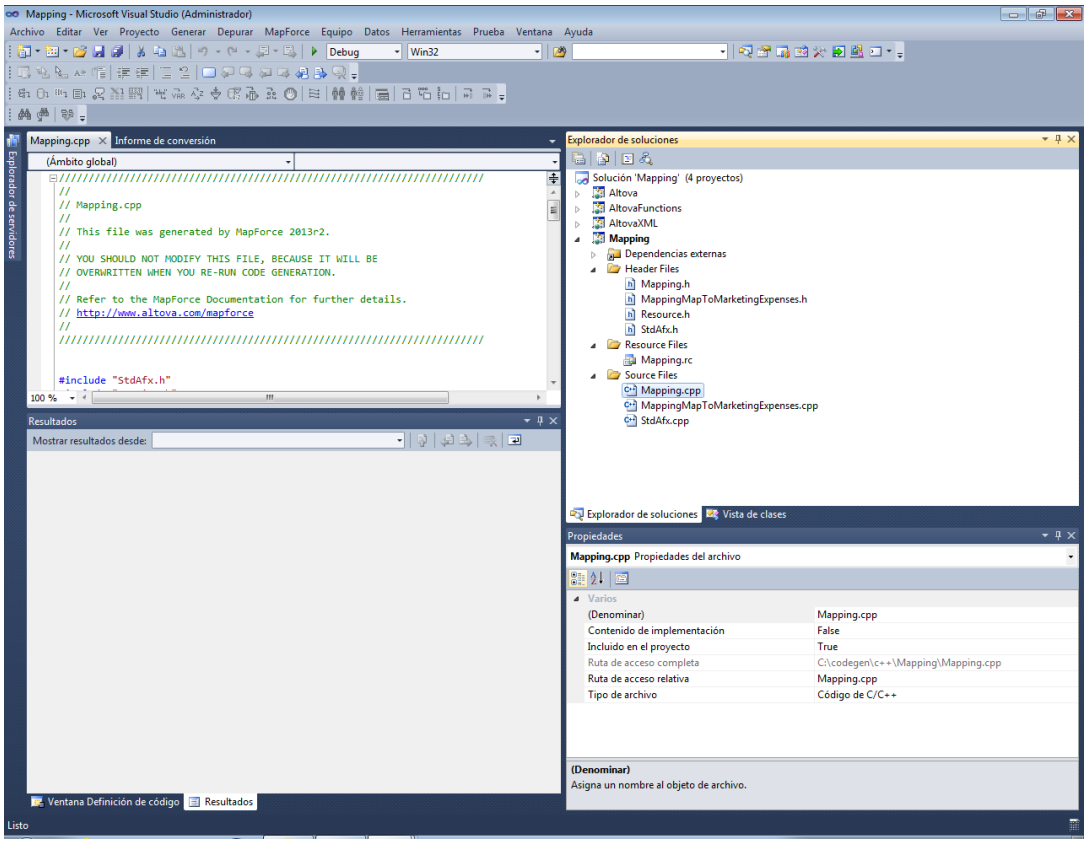
Requisitos y configuración predeterminada:

- La generación de código C++ de MapForce es compatible con Visual Studio 2005, 2008 y 2010 directamente.
- Para Visual Studio se generan archivos **.sln** y **.vcproj**. En la pestaña *Generación de código* del cuadro de diálogo "Opciones" (**Herramientas | Opciones**) puede elegir el IDE de destino y las opciones de generación de código.
- Haga clic en **Archivo | Configurar asignación** para definir las opciones de configuración de la asignación. Las opciones predeterminadas son: *Nombre de la aplicación=Mapping*.

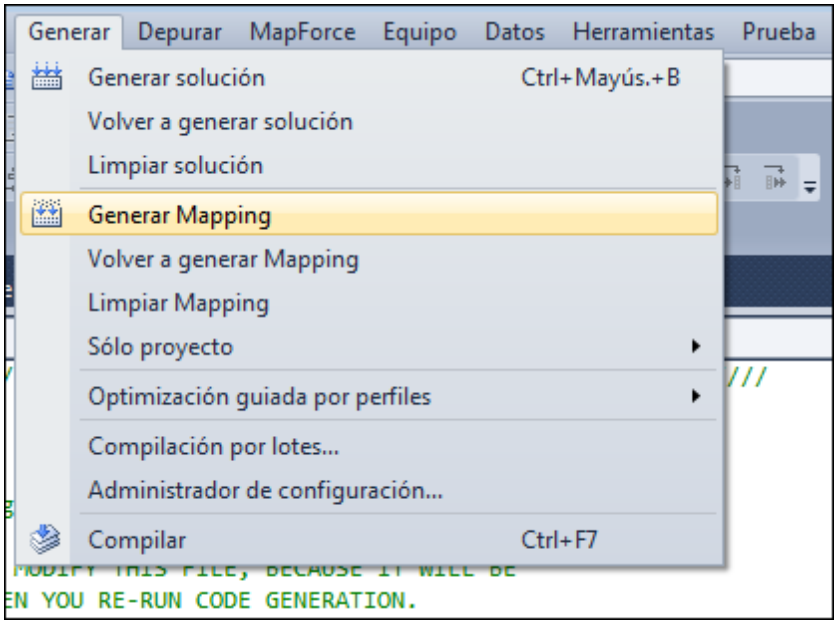
Para ver la configuración específica para bases de datos, haga clic en el componente de base de datos y seleccione la opción de menú **Componente | Propiedades**.

Para generar código C++ en MapForce:

1. Seleccione la opción de menú **Archivo | Generar código en | C++**.
   MapForce solicita el directorio de destino para los archivos generados.
2. Seleccione el directorio donde desea guardar los archivos (p. ej. `c:\codegen\C++`) y haga clic en **Aceptar**.
   Si el proceso finaliza correctamente, aparece el mensaje *El código se generó correctamente*. De lo contrario aparece un mensaje de error que describe el problema en detalle.
3. Abra el archivo `mapping.sln` del subdirectorio `Mapping` (`c:\codegen\C++\mapping`) en Microsoft Visual C++.



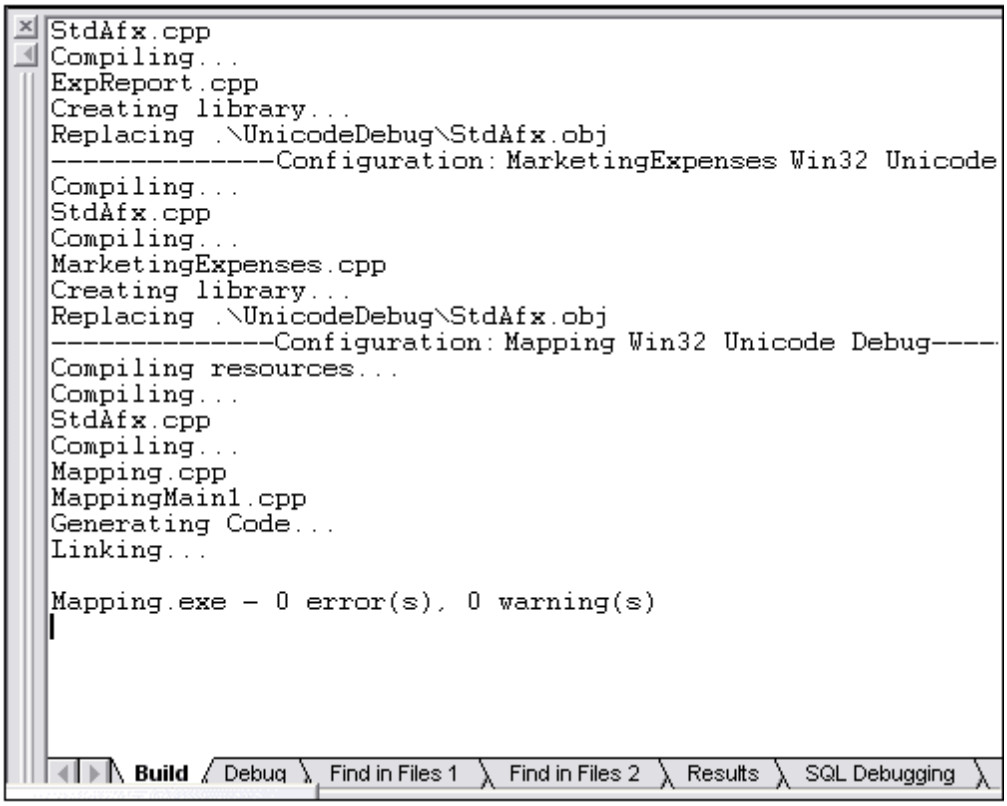
4. Seleccione la opción de menú **Generar | Generar Mapping.**



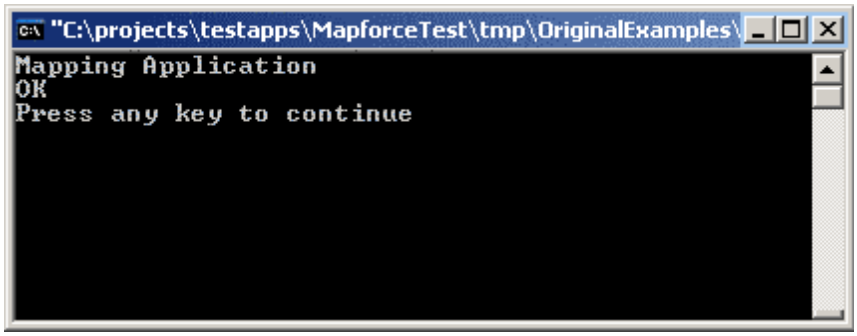
Puede elegir entre cuatro configuraciones de generación diferentes. Recuerde que Unicode es la única opción compatible con todo el juego de caracteres Unicode en archivos XML y otro tipo de archivos. Si selecciona una opción no Unicode, la aplicación funcionará con la página de códigos local de su instalación de Windows.

Depurar:	Depurar Unicode Depurar NonUnicode
Versión:	Versión Unicode Versión Non Unicode

 © 2014 Altova GmbH

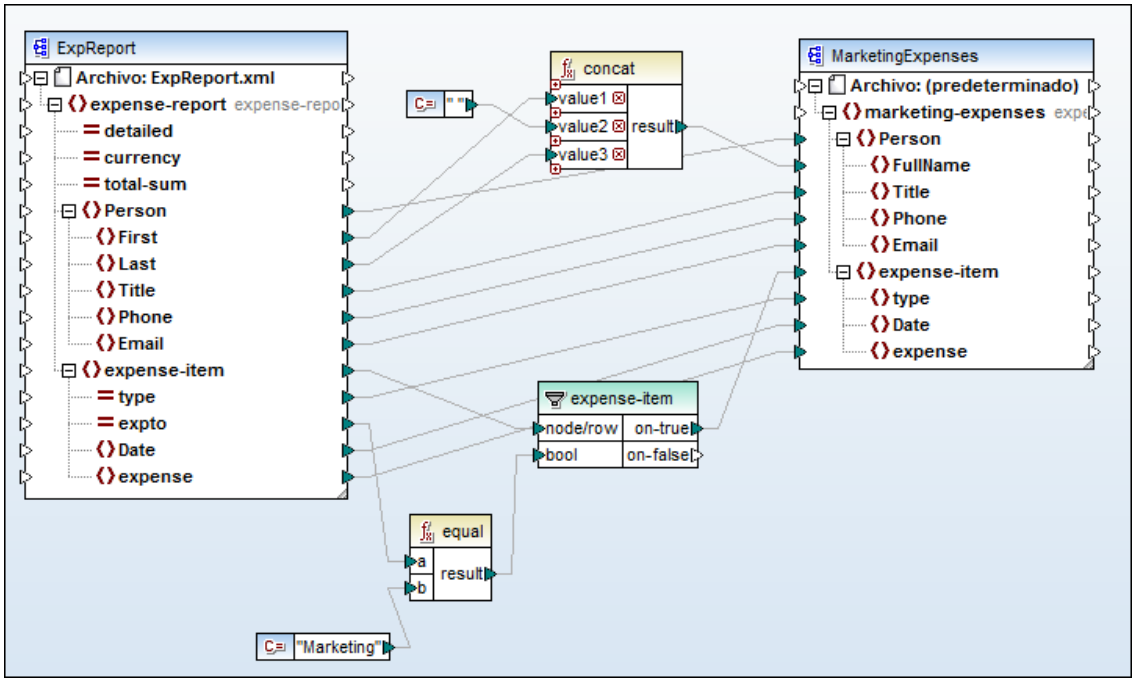


5. Una vez generado el código, ejecute el programa **Mapping.exe** para asignar los datos.



21.4 Ejemplo de asignación para generación de código

El ejemplo de asignación que aparece a continuación es el archivo de asignación `MarketingExpenses.mfd` de la carpeta `...\MapForceExamples`. Para más información consulte el apartado Integrar código de MapForce en su aplicación.



- Los parámetros para las funciones **Run** del código que aparece a continuación muestran la ruta del archivo de origen y el archivo XML de destino que produce la asignación. Pueden aparecer varios archivos de origen, pero solamente se puede asociar un archivo de destino.

```
MappingMapToMarketingExpensesObject.run(
        "C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml",
        "MarketingExpenses.xml" );
```

- Si existen **varios archivos de destino**, la sección `MappingMapToXXX` del código **se repite** por cada archivo de destino.

```
MappingMapToMarketingExpenses MappingMapToMarketingExpensesObject = new
MappingMapToMarketingExpenses();
MappingMapToMarketingExpensesObject.registerTraceTarget(ttc);
MappingMapToMarketingExpensesObject.run(
    "C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml",
    "MarketingExpenses.xml"
);
```

- Bajo la sección `Exception` puede introducir otro código de control de errores:

```
catch (Exception e), or
catch (CAltovaException& e)
```

Los fragmentos de código que aparecen a continuación son parte del código de asignación

 © 2014 Altova GmbH

generado para cada lenguaje de programación. Este código se puede personalizar con facilidad (para aceptar nombres de archivo desde la línea de comandos, por ejemplo) o se pueden integrar en otros programas.

Java (com/mapforce/MappingConsole.java)

```
public static void main(String[] args) {
    System.out.println("Mapping Application");
    try { // Mapping
        TraceTargetConsole ttc = new TraceTargetConsole();
        MappingMapToMarketingExpenses MappingMapToMarketingExpensesObject = new
MappingMapToMarketingExpenses();
        MappingMapToMarketingExpensesObject.registerTraceTarget(ttc);
        MappingMapToMarketingExpensesObject.run(
            "C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml",
            "MarketingExpenses.xml"
        );
        System.out.println("Finished");
    }
    catch (com.altova.UserException ue) {
        System.out.print("USER EXCEPTION:");
        System.out.println( ue.getMessage() );
        System.exit(1);
    }
    catch (Exception e) {
        System.out.print("ERROR:");
        System.out.println( e.getMessage() );
        e.printStackTrace();
        System.exit(1);
    }
}
```

C# (Mapping/MappingConsole.cs)

```
public static void Main(string[] args)
{
    try
    {
        TraceTargetConsole ttc = new TraceTargetConsole();
        MappingMapToMarketingExpenses MappingMapToMarketingExpensesObject = new
MappingMapToMarketingExpenses();
        MappingMapToMarketingExpensesObject.RegisterTraceTarget(ttc);
        MappingMapToMarketingExpensesObject.Run(
            "C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/ExpReport.xml",
            "MarketingExpenses.xml");
        Console.Out.WriteLine("Finished");
    }
    catch (Altova.UserException ue)
    {
        Console.Out.Write("USER EXCEPTION: ");
        Console.Out.WriteLine( ue.Message );
        System.Environment.Exit(1);
    }
    catch (Exception e)
    {
        Console.Out.Write("ERROR: ");
        Console.Out.WriteLine( e.Message );
```

```
        Console.Out.WriteLine( e.StackTrace );
        System.Environment.Exit(1);
    }
}
```

C++ (Mapping/Mapping.cpp)

```
int _tmain(int argc, TCHAR* argv[], TCHAR* envp[])
{
    tcout << _T("Mapping Application") << endl;

    try
    {
        CoInitialize(NULL);
        {
            MappingMapToMarketingExpenses MappingMapToMarketingExpensesObject;
            MappingMapToMarketingExpensesObject.Run(
                _T("C:/Projects/Files/XMLSpyExeFolder/MapForceExamples/
ExpReport.xml"),
                _T("MarketingExpenses.xml"));
        }
        CoUninitialize();

        tcout << _T("OK") << endl;
        return 0;
    }
    catch (CAltovaException& e)
    {
        if (e.IsUserException())
            tcerr << _T("User Exception: ");
        else
            tcerr << _T("Error: ");
        tcerr << e.GetInfo().c_str() << endl;
        return 1;
    }
    catch (_com_error& e)
    {
        tcerr << _T("COM-Error from ") << (TCHAR*)e.Source() << _T(":") << endl;
        tcerr << (TCHAR*)e.Description() << endl;
        return 1;
    }
    catch (std::exception& e)
    {
        cerr << "Exception: " << e.what() << endl;
        return 1;
    }
    catch (...)
    {
        tcerr << _T("Unknown error") << endl;
        return 1;
    }
}
```

 © 2014 Altova GmbH

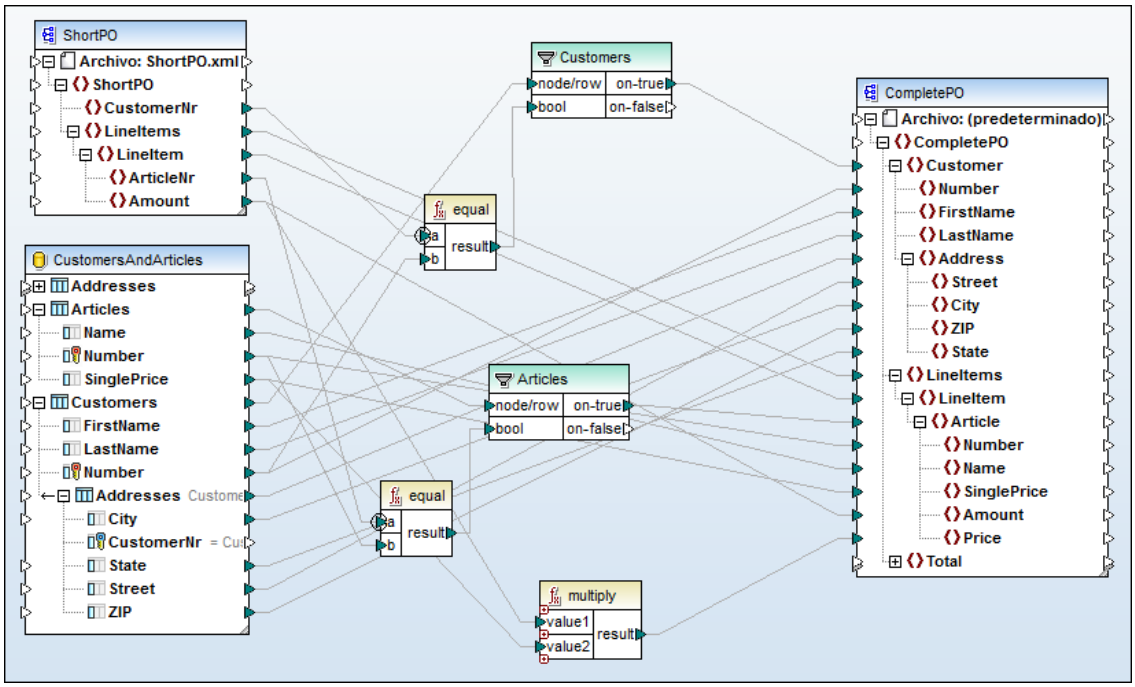
21.5 Integrar código de MapForce en su aplicación

Puede integrar el código generado por MapForce en su propia aplicación o adaptarlo a ella, incluso si el resultado de la generación de código es una aplicación totalmente funcional. El archivo de asignación de datos que aparece en la siguiente imagen es `DB_CompletePO.mfd`, disponible en la carpeta `...\MapForceExamples`.

Para más información consulte el apartado Generar código de programa.

En este apartado se explica:

- cómo **se modifican** los archivos de origen y destino de un archivo de asignación
- cómo utilizar una secuencia XML de entrada como origen de datos
- y dónde puede añadir su propio código de **control de errores**



Este ejemplo incluye dos orígenes de datos y un destino:

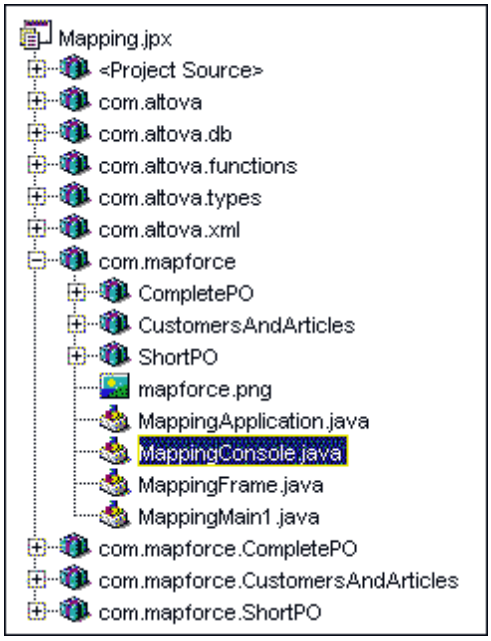
- ShortPO.xml es el archivo **XML de origen**
- CustomersAndArticles.mdb es la **base de datos de origen**
- CompletePO.xml es el archivo **XML de destino**

21.5.1 Código de MapForce en aplicaciones Java

Para este ejemplo asumimos que utiliza **Borland JBuilder** como entorno Java.

Tras generar el código Java en MapForce y definir una carpeta llamada `output` como carpeta de destino:

1. Navegue hasta la carpeta de destino `output` y abra el archivo de proyecto `Mapping.jpx`.
2. Haga doble clic en el archivo `MappingConsole.java`.



A continuación puede ver un fragmento del código.

```
        {
            com.altova.io.Input ShortPO2Source = new
com.altova.io.FileInput("ShortPO.xml");
            com.altova.io.Output CompletePO2Target = new
com.altova.io.FileOutput("CompletePO.xml");

            MappingMapToCompletePOObject.run(
                  java.sql.DriverManager.getConnection(
                     "jdbc:odbc:;DRIVER=Microsoft Access Driver
(*.mdb);DBQ=CustomersAndArticles.mdb",
                     "",
                     ""),
                  ShortPO2Source,
                  CompletePO2Target);
        }
```

Observe que, para una mayor claridad, se eliminaron los nombres de ruta del código fuente generado.

En `MappingMapToCompletePO.run` todos los parámetros que se pasan al método **`run`** son archivos de origen, **`excepto`** el último (`CompletePO.xml`). Compruebe si el último parámetro es el archivo de destino.

 © 2014 Altova GmbH

Los archivos de **origen** de este ejemplo son:

- Archivo XML de origen: `ShortPO.xml`
- Base de datos de origen: `CustomerAndArticles.mdb`, incluida la cadena de conexión. Los dos **parámetros vacíos** **""** que siguen al parámetro de base de datos inicial son para el **nombre de usuario** y la **contraseña** (en texto no cifrado), en caso de que la BD requiera un nombre de usuario y una contraseña.

El archivo de **destino** de este ejemplo es:

- `CompletePO.xml`

Para definir archivos de origen y destino propios:

- Edite los parámetros que se pasan a los constructores `FileInput` y `FileOutput`.

Para usar una secuencia XML de entrada como origen de datos XML:

- Sustituya `FileInput` por `StreamInput` y pase **`java.io.InputStream`** en lugar de un nombre de archivo.

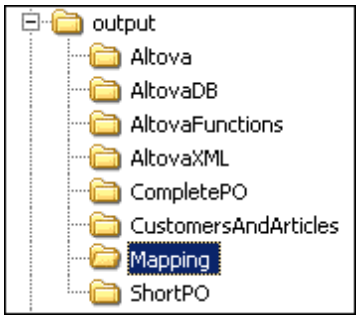
Para agregar un código de control de errores:

- Edite el código que aparece bajo el código **`catch`** `(Exception ex)`.

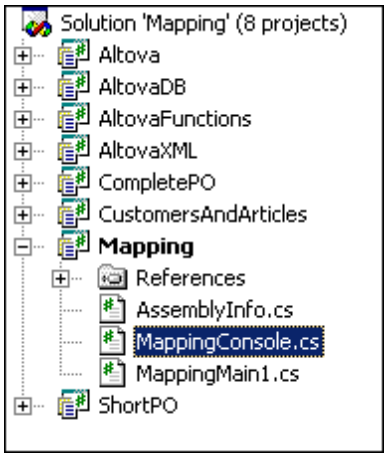
21.5.2 Código de MapForce en aplicaciones C#

Tras generar el código C# en MapForce y definir una carpeta llamada `output` como carpeta de destino:

1. Navegue hasta la carpeta de destino `output\Mapping` y abra el archivo `Mapping.sln` en Visual Studio.



2. Haga doble clic en el archivo `MappingConsole.cs`.



A continuación puede ver un fragmento del código.

```
            {
               Altova.IO.Input ShortPO2Source = new
Altova.IO.FileInput("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);
            }
```

Observe que, para una mayor claridad, se eliminaron los nombres de ruta del código fuente generado.

En `MappingMapToCompletePO.run` todos los parámetros que se pasan al método **`run`** son archivos de origen, **`excepto`** el último (`CompletePO.xml`). Compruebe si el último parámetro es el archivo de destino.

 © 2014 Altova GmbH

Los archivos de **origen** de este ejemplo son:

- Archivo XML de origen: `ShortPO.xml`
- Base de datos de origen: `CustomerAndArticles.mdb`, incluida la cadena de conexión.

El archivo de **destino** de este ejemplo es:

- `CompletePO.xml`

Para definir archivos de origen y destino propios:

- Edite los parámetros que se pasan a los constructores `FileInput` y `FileOutput`.

Para usar una secuencia XML de entrada como origen de datos XML:

- Sustituya `FileInput` por `StreamInput` y pase **`System.IO.Stream`** en lugar de un nombre de archivo.

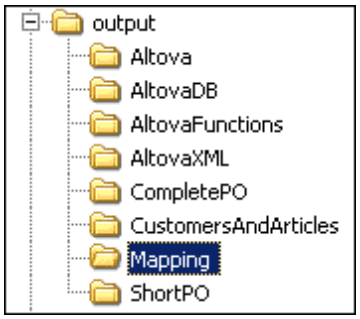
Para agregar un código de control de errores:

- Edite el código que aparece bajo el código **`catch`** `(Exception e)`.

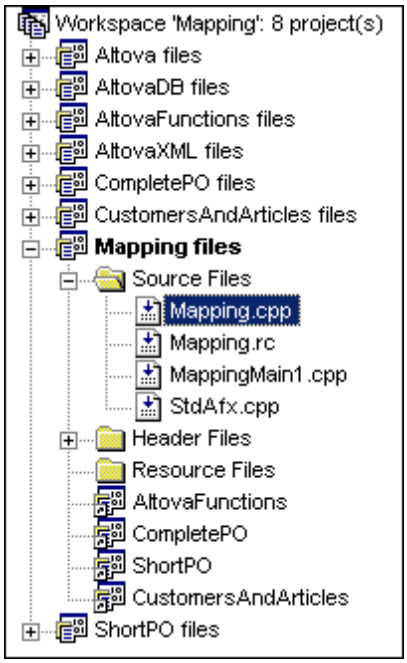
21.5.3 Código de MapForce en aplicaciones C++

Tras generar el código C++ en MapForce y definir una carpeta llamada `output` como carpeta de destino:

1. Navegue hasta la carpeta de destino `output\Mapping` y abra el archivo `Mapping.sln` en MS Visual C++.



2. Haga doble clic en el archivo `Mapping.cpp` para abrir el archivo de proyecto de asignación.



A continuación puede ver un fragmento del código.

```
CoInitialize(NULL);
{
   MappingMapToCompletePO MappingMapToCompletePOObject;
   MappingMapToCompletePOObject.Run(
      _T("Provider=Microsoft.Jet.OLEDB.4.0; Data
Source=CustomersAndArticles.mdb; "),
      _T("ShortPO.xml"),
      _T("CompletePO.xml"));
}
CoUninitialize();
```

 © 2014 Altova GmbH

```
tcout << _T("Finshed") << endl;
return 0;
```

Observe que, para una mayor claridad, se eliminaron los nombres de ruta del código fuente generado.

En `MappingMapToCompletePO.run` todos los parámetros que se pasan al método **`run`** son archivos de origen, **excepto** el último (`CompletePO.xml`). Compruebe si el último parámetro es el archivo de destino.

Los archivos de **origen** de este ejemplo son:

- Archivo XML de origen: `ShortPO.xml`
- Base de datos de origen: `CustomerAndArticles.mdb`, incluida la cadena de conexión.

El archivo de **destino** de este ejemplo es:

- `CompletePO.xml`

Para definir archivos de origen y destino propios:

- Edite directamente los parámetros que se pasan al método **`run`** de `MappingMapToCompletePOObject`.

Para usar una secuencia XML de entrada como origen de datos XML:

- Navegue hasta la declaración del método **`run`** y configure allí los parámetros necesarios

Para agregar un código de control de errores:

- Edite el código que aparece bajo el código **`catch`** `(CAltovaException& e)`.

21.5.4 Compatibilidad con secuencias de datos

A partir de la versión 2008 Release 2, el código C# y Java generado con MapForce también admite el uso de secuencias como entrada/salida de la asignación.

La función más importante de las clases generadas de la asignación es el método **run**. Tiene un parámetro por cada componente de origen o entrada estático de la asignación y un parámetro final para el componente de salida. Los componentes que procesan varios archivos no aparecen como parámetros en el método **run** porque en este caso los nombres de archivo se procesan de forma dinámica dentro de la asignación.

Antes de la versión 2008 Release 2 de MapForce, el método **run** exigía usar como argumentos los nombres de archivo y los objetos de conexión de BD. Ahora también puede usar objetos de entrada o salida en lugar de nombres de archivo.

Es decir, puede usar como origen o destino:

- archivos (identificados por el nombre de archivo)
- secuencias binarias
- secuencias de caracteres (escritores/lectores): se omitirá el juego de codificación de caracteres del componente de MapForce
- cadenas de texto: se omitirá el juego de codificación de caracteres del componente de MapForce
- documentos DOM (para XML solamente)

Estos objetos están disponibles en el paquete **com.altova.io** (Java) y en los espacios de nombres **Altova.IO** (C#) y se generan siempre. Los prefijos de espacio de nombres / de paquete se omiten en esta sección para mayor claridad.

Los objetos de entrada y salida tienen clases base llamadas **Input** y **Output** (es decir, **com.altova.io.Input** y **com.altova.io.Output** para Java y **Altova.IO.Input** y **Altova.IO.Output** para C#).

En los siguientes ejemplos, cuando decimos *texto* nos referimos a cualquier archivo EDI / X12 / HL7, FlexText, CSV o FLF.

Recuerde que los archivos Excel solamente se pueden asignar desde archivos o secuencias binarias y que solamente son compatibles con MapForce Enterprise Edition.

Los objetos contenedores que aparecen a continuación se usan como parámetros del método **run**:

Java:

Archivos/nombres de archivo (para XML, texto y Excel):

```
com.altova.io.FileInput(String filename)
com.altova.io.FileOutput(String filename)
```

Secuencias (para XML, texto y Excel):

```
com.altova.io.StreamInput(java.io.InputStream stream)
com.altova.io.StreamOutput(java.io.OutputStream stream)
```

Cadenas (para XML y texto):

```
com.altova.io.StringInput(String xmlcontent)
```

 © 2014 Altova GmbH

```
com.altova.io.StringOutput()
```

Java IO reader/writer (para XML y texto):

```
com.altova.io.ReaderInput(java.io.Reader reader)
com.altova.io.WriterOutput(java.io.Writer writer)
```

Documentos DOM (para XML solamente):

```
com.altova.io.DocumentInput(org.w3c.dom.Document document)
com.altova.io.DocumentOutput(org.w3c.dom.Document document)
```

C#:

Archivos/nombres de archivo (para XML, texto y Excel):

```
Altova.IO.FileInput(string filename)
Altova.IO.FileOutput(string filename)
```

Secuencias (para XML, texto y Excel):

```
Altova.IO.StreamInput(System.IO.Stream stream)
Altova.IO.StreamOutput(System.IO.Stream stream)
```

Cadenas (para XML y texto):

```
Altova.IO.StringInput(string content)
Altova.IO.StringOutput(StringBuilder content)
```

Java IO reader/writer (para XML y texto):

```
Altova.IO.ReaderInput(System.IO.TextReader reader)
Altova.IO.WriterOutput(System.IO.TextWriter writer)
```

Documentos DOM (para XML solamente):

```
Altova.IO.DocumentInput(System.Xml.XmlDocument document)
Altova.IO.DocumentOutput(System.Xml.XmlDocument document)
```

MapForce genera la función **`run`**, que toma varios **Input** y un **Output**.

Ejemplo

Usamos una aplicación Java (o C#) y usaremos MapForce para generar código de asignación que integraremos en nuestra aplicación. Por ejemplo, la asignación tiene dos archivos XML de origen y un archivo de texto de destino. MapForce genera esta función **`run`**:

```
void run(Input in1. Input in2, Output out1);
```

Imaginemos que nuestra aplicación exige que asignemos datos desde un archivo local y una secuencia binaria a una secuencia de caracteres. Los datos vienen dados por otras fuentes y queremos integrar el código generado por MapForce en nuestra aplicación. La aplicación ofrece los orígenes y destinos como:

```
String filename;
Java.io.InputStream stream;
Java.io.Writer writer;
```

Es necesario construir estos contenedores para la función **`run`** generada por MapForce:

```
// aquí se considera que com.altova.io está importado:
Input input1 = new FileInput(filename);
Input input2 = new StreamInput(stream);
Output output1 = new WriterOutput(writer);
```

Llegados a este punto es necesario llamar a la función **run** generada por MapForce:

```
MappingMapToSomething.run(input1, input2, output1);
```

Y ya está.

El comportamiento del código **C#** es casi idéntico al de Java, excepto que **run** se llama **Run** y que las clases `reader/writer` y la secuencia de .NET tienen otro nombre en C#.

Los demás tipos de entrada y salida, como las cadenas de texto o los documentos DOM, se usan de la misma manera, pero hay algunas diferencias que se indican a continuación.

Comportamiento de las secuencias

Es necesario señalar las normas que rigen las secuencias de caracteres y las secuencias binarias:

- Las secuencias y los readers/writers deben estar abiertos y listos para usar antes de llamar a **run**.
- La función **run** generada con MapForce cierra por defecto las secuencias y los readers/writers cuando termina. Si no quiere que esto pase, antes de llamar a la función **run** inserte esta línea:

  ```
  MappingMapToSomething.setCloseObjectsAfterRun(true);  // Java
  ```

  o

  ```
  MappingMapToSomething.CloseObjectsAfterRun = false;   // C#
  ```

- los orígenes y destinos Excel no se pueden leer ni escribir en secuencias de caracteres.

Comportamiento de StringOutput

Hay una pequeña diferencia entre el comportamiento de StringOutput en C# y en Java.

En **Java** StringOutput no toma un argumento. Al contenido se puede acceder con:

```
// asignación desde String hasta (otro) String

String blah = "<here>is some xml text</here>";

Input input = new StringInput(blah);
Output output = new StringOutput();

MappingMapToBlah.run(input, output);

String myTargetData = output.getString().toString();
```

 © 2014 Altova GmbH

El método **getString()** devuelve un *StringBuffer*, de ahí que sea necesario usar **toString()**.

En **C#**, sin embargo, StringOutput toma un argumento (StringBuilder) que debe suministrarse de antemano.
El argumento StringBuilder puede contener datos y el resultado de la asignación se anexa.

Recuerde que los orígenes y destinos Excel no se pueden asignar a cadenas de texto.

Comportamiento de los documentos DOM

Si lo necesita para su aplicación, puede pasar DocumentInput / DocumentOutput como argumento para **`run`**.

No olvide que en el modo de destino el documento que se pasa al constructor DocumentOutput debe estar vacío.

Tras llamar a **`run`**, el documento DOM dado/generado por el constructor de DocumentOutput ya contiene datos asignados, por lo que no es necesario "guardar en el documento". Una vez finalizado el proceso de asignación, puede manipular el documento todo lo que quiera.

A los documentos DOM solamente se puede asignar contenido XML.

Bases de datos

Si una asignación incluye componentes de base de datos, la función **`run`** incluirá el objeto de conexión de base de datos en la posición correspondiente.

Por ejemplo, si la asignación asigna contenido de texto, XML y otros datos de una base de datos a un archivo de destino, MapForce genera esta función **`run`**:

```
void run(Input in1, Input in2, java.sql.Connection dbConn, Output
out1);
```

El orden de los argumentos es importante. Puede modificar los parámetros `dbConn` o usar los parámetros predeterminados generados por MapForce cuando integre el código en su aplicación.

21.6 Usar la biblioteca de código generado

Resumen

La función de generación de código de MapForce genera una aplicación completa que ejecuta automáticamente todos los pasos de la asignación. Si desea integrar el código de asignación dentro de su propia aplicación, puede generar bibliotecas para todos los esquemas XML usados en la asignación. Estas bibliotecas permiten al código crear o leer instancias XML que se usan o crean con el código de la asignación.

Si en el cuadro de diálogo Opciones selecciona la opción *Generar clases contenedoras*, MapForce no sólo crea automáticamente la aplicación de la asignación, sino que además genera clases contenedoras para todos los esquemas usados en la asignación. Estas se pueden usar en el código igual que las bibliotecas generadas por el generador de código de XMLSpy.

La biblioteca generada con MapForce es un contenedor DOM (Document Object Model) que permite leer, modificar y crear documentos XML de forma sencilla y segura. Todos los datos se guardan en el DOM y hay métodos para extraer datos del DOM y para actualizar y crear datos en el DOM.

Bibliotecas generadas

Cuando MapForce genera código a partir de un esquema XML o de una DTD, se generan estas bibliotecas:

Nombre	Finalidad
Altova	Biblioteca base que ofrece compatibilidad general con el tiempo de ejecución, idéntica para todos los esquemas
AltovaXML	Biblioteca base que ofrece compatibilidad con el tiempo de ejecución para XML, idéntica para todos los esquemas
SuEsquema	Biblioteca que contiene declaraciones generadas a partir del esquema de entrada. Recibe el nombre del archivo de esquema o DTD
*SuEsquema*Test	Aplicación base de prueba (sólo en XMLSpy)

Espacios de nombres y generación de nombres

Por lo general, el generador de código intenta conservar los nombres que los espacios de nombres, clases y miembros tenían en el esquema XML original. Los caracteres que no sean válidos en los identificadores del lenguaje de destino se sustituyen con el carácter "_". A los nombres que puedan entrar en conflicto con otros nombres o palabras reservadas se les anexa un número. La generación de nombres puede personalizarse con la declaración default en la plantilla SPL.

Los espacios de nombres del esquema XML se convierten en paquetes en Java o en espacios de nombres en C# o C++, usando el prefijo de espacio de nombres del esquema como espacio de nombres del código. La biblioteca completa se incluye en un paquete o espacio de nombres derivado del nombre de archivo del esquema para que pueda usar varias bibliotecas generadas en un programa sin que se produzcan conflictos de nombres.

Tipos de datos

XML Schema tiene un modelo de tipos de datos mucho más complicado que Java, C# y C++. El

 © 2014 Altova GmbH

generador de código convierte los 44 tipos de XML Schema en un par de tipos primitivos del lenguaje de destino o en clases derivadas de la biblioteca Altova. La asignación de tipos simples se puede configurar en la plantilla SPL.

Los tipos complejos y los tipos derivados definidos en el esquema se convierten en clases en la biblioteca generada.

Las facetas de enumeración de tipos simples se convierten en constantes simbólicas.

Clases generadas

Por lo general, MapForce genera una clase por cada tipo encontrado en el esquema XML o por cada elemento de la DTD. También se genera una clase adicional por biblioteca para representar el documento propiamente dicho y que se puede usar para cargar y guardar el documento.

Administración de memoria

Un árbol DOM está compuesto de nodos, que siempre son propiedad de un documento DOM concreto, incluso si el nodo no forma parte del contenido del documento. Las clases generadas no son valores, sino referencias a los nodos DOM que representan. Esto significa que al asignar una instancia de una clase generada no se copia el valor, sino que sólo se crea una referencia adicional a los mismos datos.

21.6.1 Esquema de ejemplo

Usar las clases generadas

Para no complicar las cosas demasiado vamos a trabajar con un archivo xsd pequeño. El código fuente aparece a continuación. Guarde este código como `Library.xsd` si desea conseguir los mismos resultados que en este ejemplo:

```
<?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: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="ISBN" type="xs:string" use="required"/>
		<xs:attribute name="Variant" type="BookVariant"/>
	</xs:complexType>
	<xs:simpleType name="BookVariant">
		<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>
```

En nuestros ejemplos utilizamos siempre este archivo de esquema para que pueda comprender las diferencias que hay entre los lenguajes de programación gracias a los ejemplos.

Recuerde que las clases generadas recibirán nombres diferentes dependiendo del archivo xsd. Cuando trabaje con sus propios esquemas y otros archivos de muestra, asegúrese de adaptar los nombres de las funciones y las variables.

 © 2014 Altova GmbH

21.6.2 Usar la biblioteca de código Java generado

Paquetes generados

Nombres	Finalidad
com.altova	Paquete base que ofrece compatibilidad con tiempo de ejecución general, idéntico para todos los esquemas
com.altova.xml	Paquete base que ofrece compatibilidad con tiempo de ejecución para XML, idéntico para todos los esquemas
com.SuEsquema	Paquete que ofrece declaraciones generadas a partir del esquema de entrada, que recibe el nombre del archivo de esquema o DTD
*com.SuEsquema*Test	Aplicación base de prueba (sólo en XMLSpy)

Implementación de DOM
El código Java generado usa JAXP (API de Java para procesamiento XML) como interfaz DOM subyacente. Recuerde que la implementación predeterminada de DOM que viene con Java 1.4 no crea atributos de espacio de nombres automáticamente al serializar documentos XML. Si tiene problemas, instale Java 6 o superior.

Tipos de datos
Esta es la asignación predeterminada de tipos de XML Schema a los tipos de datos Java:

XML Schema	Java	Comentarios
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 unsigned
xs:int	int	
xs:unsignedInt	long	Java no tiene tipos unsigned
xs:dateTime, date, time, gYearMonth, gYear, gMonthDay, gDay, gMonth	com.altova.types.DateTime	
xs:duration	com.altova.types.Duration	
xs:hexBinary and xs:base64Binary	byte[]	La codificación y decodificación de datos binarios no se hace automáticamente
xs:anySimpleType	string	

Los tipos de XML Schema que no aparecen en esta lista son tipos derivados y se asignan al tipo de Java equivalente a su tipo base correspondiente.

Clase de documento generada
Además de clases para los tipos declarados en el esquema XML, también se genera una clase de documento. Esta clase contiene todos los elementos raíz posibles como miembros de clase e

incluye los métodos que aparecen a continuación ("Doc" es el nombre de la clase de documento generada):

Método	Finalidad
static *Doc* **loadFromFile**(String fileName)	Carga un documento XML a partir de un archivo
static *Doc* **loadFromString**(String xml)	Carga un documento XML a partir de una cadena
static *Doc* **loadFromBinary**(byte[] xml)	Carga un documento XML a partir de una matriz de bytes
void **saveToFile**(String fileName, boolean prettyPrint, String encoding = "UTF-8")	Guarda un documento XML en un archivo con la codificación indicada. Si no se indica ninguna codificación, entonces se usa UTF-8
void **SaveToFile**(String fileName, boolean prettyPrint, String encoding, boolean bigEndian, boolean writeBOM)	Guarda un documento XML en un archivo con la codificación indicada. Para las codificaciones Unicode puede indicar el orden de bytes y la marca de orden de bytes Unicode
String **saveToString**(boolean prettyPrint)	Guarda un documento XML en una cadena
byte[] **saveToBinary**(boolean prettyPrint, String encoding = "UTF-8")	Guarda un documento XML en una matriz de bytes con la codificación indicada. Si no se indica ninguna codificación, entonces se usa UTF-8
byte[] **saveToBinary**(boolean prettyPrint, String encoding, boolean bigEndian, boolean writeBOM)	Guarda un documento XML en una matriz de bytes con la codificación indicada. Para las codificaciones Unicode puede indicar el orden de bytes y la marca de orden de bytes Unicode
static *Doc* **createDocument**()	Crea un documento XML nuevo vacío
void **setSchemaLocation**(String schemaLocation)	Agrega el atributo xsi:schemaLocation o xsi:noNamespaceSchemaLocation al elemento raíz. Para ello es necesario que exista un elemento raíz

Clases de tipo generadas

Por cada tipo presente en el esquema se genera una clase que contiene un miembro por cada atributo y elemento del tipo. Los miembros reciben el mismo nombre que los atributos y elementos del esquema original (en case de conflicto, se anexa un número al nombre). Para los tipos simples y mixtos, se generan métodos **getValue**() y **setValue**(string). Para los tipos simples con facetas de enumeración, se pueden usar **getEnumerationValue**() y **setEnumerationValue**(int) junto con las constantes generadas por cada valor de enumeración. Además, el método **getStaticInfo**() permite acceder a información del esquema como **com.altova.xml.meta.SimpleType** o **com.altova.xml.meta.ComplexType** (véase más adelante).

La derivación por extensión de complexType de XML Schema da lugar a clases derivadas en la biblioteca generada.

Clases generadas de miembros atributo

Por cada miembro atributo de un tipo se crea una clase con estos métodos.

Método	Finalidad
boolean **exists**()	Devuelve el valor true si existe el atributo

 © 2014 Altova GmbH

void **remove**()	Quita el atributo de su elemento primario
AttributeType **getValue**()	Obtiene el valor del atributo
void **setValue**(*AttributeType* value)	Establece el valor del atributo
int **getEnumerationValue**()	Se genera solo para tipos enumeration. Devuelve una de las constantes generadas para los valores posibles o **Invalid** si el valor no coincide con ningún valor enumerado en el esquema
void **setEnumerationValue**(int)	Se genera solo para tipos enumeration. Pasa a este método una de las constantes generadas para los valores posibles para establecer el valor
com.altova.xml.meta.Attribute **getInfo**()	Devuelve un objeto para consultar información del esquema (véase más adelante)

Clases generadas de miembros elemento

Por cada miembro elemento de un tipo se crea una clase con estos métodos.

Método	**Finalidad**
MemberType **first**()	Devuelve la primera instancia del miembro elemento
MemberType **at**(int index)	Devuelve la instancia del miembro elemento
MemberType **last**()	Devuelve la última instancia del miembro elemento
MemberType **append**()	Crea un elemento nuevo y lo anexa a su elemento primario
boolean **exists**()	Devuelve true si existe al menos un elemento
int **count**()	Devuelve el recuento de elementos
void **remove**()	Elimina todas las apariciones del elemento en su elemento primario
void **removeAt**(int index)	Elimina la aparición del elemento indicado por el índice
java.util.Iterator **iterator**()	Devuelve un objeto para las instancias repetidas del miembro elemento
MemberType **getValue**()	Obtiene el contenido del elemento (sólo se genera si el elemento puede tener contenido simple o mixto)
void **setValue**(*MemberType* value)	Establece el contenido del elemento (sólo se genera si el elemento puede tener contenido simple o mixto)
int **getEnumerationValue**()	Sólo se genera para tipos enumeration. Devuelve una de las constantes generadas para los valores posibles o **Invalid** si el valor no coincide con ninguno de los valores enumerados en el esquema
void **setEnumerationValue**(int)	Sólo se genera para tipos enumeration. Pasa a este método una de las constantes generadas para los valores posibles para establecer el valor
com.altova.xml.meta.Element **getInfo**()	Devuelve un objeto para consultar información del esquema (véase más adelante)

Crear documentos XML

La biblioteca de código generada facilita la creación de archivos XML. Empecemos con un archivo XML de ejemplo para el esquema de ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<Library xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd"
xmlns="http://www.nanonull.com/LibrarySample" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
	<Book ISBN="0764549642" Variant="Paperback">
		<Title>The XML Spy Handbook</Title>
		<Author>Altova</Author>
	</Book>
</Library>
```

El código Java que aparece a continuación se usó para crear este archivo. Las clases Library2, LibraryType y BookType se generan a partir del esquema y representan todo el documento, el tipo del elemento <Library> y el tipo complejo BookType, que es el tipo del elemento <Book>. Observe que la clase Library se renombró automáticamente y ahora se llama Library2 para evitar posibles conflictos con el nombre de espacio de nombres de la biblioteca.

```
	protected static void example() throws Exception {
		// create a new, empty XML document
		Library2 libDoc = Library2.createDocument();

		// create the root element <Library> and add it to the document
		LibraryType lib = libDoc.Library3.append();

		// create a new <Book> and add it to the library
		BookType book = lib.Book.append();

		// set the ISBN attribute of the book
		book.ISBN.setValue("0764549642");

		// set the Variant attribute of the book using an enumeration
	constant
		book.Variant.setEnumerationValue( BookVariant.EPAPERBACK );

		// add the <Title> and <Author> elements, and set values
		book.Title.append().setValue("The XML Spy Handbook");
		book.Author.append().setValue("Altova");

		// set the schema location (this is optional)
		libDoc.setSchemaLocation("Library.xsd");

		// save the XML document to a file with default encoding (UTF-8).
"true" causes the file to be pretty-printed.
		libDoc.saveToFile("Library1.xml", true);
	}
```

Observe que todos los elementos se crean llamando a append() y que a los atributos (como el atributo ISBN del ejemplo) se puede acceder como miembros de la estructura.

Leer documentos XML

 © 2014 Altova GmbH

Este ejemplo de código lee el archivo que acabamos de crear y muestra los diferentes métodos que hay para leer archivos XML:

```
	protected static void example() throws Exception {
		// load XML document
		Library2 libDoc = Library2.loadFromFile("Library1.xml");

		// get the first (and only) root element <Library>
		LibraryType lib = libDoc.Library3.first();

		// it is possible to check whether an element exists:
		if (!lib.Book.exists())
		{
			System.out.println("This library is empty.");
			return;
		}

		// iteration: for each <Book>...
		for (java.util.Iterator itBook = lib.Book.iterator();
itBook.hasNext(); )
		{
			BookType book = (BookType) itBook.next();

			// output values of ISBN attribute and (first and only)
title element
			System.out.println("ISBN: " + book.ISBN.getValue());
			System.out.println("Title: " +
book.Title.first().getValue());

			// read and compare an enumeration value
			if (book.Variant.getEnumerationValue() ==
BookVariant.EPAPERBACK)
				System.out.println("This is a paperback book.");

			// for each <Author>...
			for (java.util.Iterator itAuthor = book.Author.iterator();
itAuthor.hasNext(); )
				System.out.println("Author: " +
((com.Library.xs.stringType)itAuthor.next()).getValue());

			// alternative: use count and index
			for (int j = 0; j < book.Author.count(); ++j)
				System.out.println("Author: " +
book.Author.at(j).getValue());
		}
	}
```

Observe que el tipo xs.stringType del miembro book.Author es una clase generada para todos los elementos que tengan contenido de cadena simple. Para los demás tipos de XML Schema también se generan clases similares.

Si lo deesa, también puede cargar un documento, modificarlo añadiéndole elementos y atributos y después guardar el archivo en el disco.

Control de errores
Los errores se notifican mediante excepciones. Estas clases de excepción se definen en el espacio de nombres com.altova:

Clase	Clase base	Descripción
SourceInstanceUnvailableException	Exception	Se produjo un problema al cargar una instancia XML
TargetInstanceUnavailableException	Exception	Se produjo un problema al guardar una instancia XML

Además se suelen usar estas excepciones de Java:

Clase	Descripción
java.lang.Error	Error interno de lógica de programa (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 se produjo un error de conversión de tipos
java.lang.ArithmeticException	Se produjo un error de conversión de tipos numéricos

Acceder a metadatos
La biblioteca generada permite acceder a información estática del esquema a través de estas clases. Los métodos que devuelven una de las clases de metadatos devuelven null si la propiedad correspondiente no existe.

com.altova.xml.meta.SimpleType

Método	Finalidad
SimpleType **getBaseType**()	Devuelve el tipo base de este tipo
String **getLocalName**()	Devuelve el nombre local del tipo
String **getNamespaceURI**()	Devuelve el URI de espacio de nombres del tipo
int **getMinLength**()	Devuelve el valor de esta faceta
int **getMaxLength**()	Devuelve el valor de esta faceta
int **getLength**()	Devuelve el valor de esta faceta
int **getTotalDigits**()	Devuelve el valor de esta faceta
int **getFractionDigits**()	Devuelve el valor de esta faceta
String **getMinInclusive**()	Devuelve el valor de esta faceta
String **getMinExclusive**()	Devuelve el valor de esta faceta
String **getMaxInclusive**()	Devuelve el valor de esta faceta
String **getMaxExclusive**()	Devuelve el valor de esta faceta
String[] **getEnumerations**()	Devuelve una matriz de todas las facetas enumeration
String[] **getPatterns**()	Devuelve una matriz de todas las facetas pattern
int **getWhitespace**()	Devuelve el valor de la faceta whitespace, que es uno de estos: com.altova.typeinfo.WhitespaceType.Whitespace_Unknown com.altova.typeinfo.WhitespaceType.Whitespace_Preserve com.altova.typeinfo.WhitespaceType.Whitespac

 © 2014 Altova GmbH

	e_Replace com.altova.typeinfo.WhitespaceType.Whitespace_Collapse

com.altova.xml.meta.ComplexType

Método	**Finalidad**
Attribute[] **GetAttributes**()	Devuelve una lista con todos los atributos
Element[] **GetElements**()	Devuelve una lista con todos los elementos
ComplexType **getBaseType**()	Devuelve el tipo base de este tipo
String **getLocalName**()	Devuelve el nombre local del tipo
String **getNamespaceURI**()	Devuelve el URI de espacio de nombres del tipo
SimpleType **getContentType**()	Devuelve el tipo simple del contenido
Element **findElement**(String localName, String namespaceURI)	Encuentra el elemento con el nombre local y el URI de espacio de nombres indicados
Attribute **findAttribute**(String localName, String namespaceURI)	Encuentra el atributo con el nombre local y el URI de espacio de nombres indicados

com.altova.xml.meta.Element

Método	**Finalidad**
ComplexType **getDataType**()	Devuelve el tipo del elemento. Recuerde que es siempre 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
int **getMinOccurs**()	Devuelve el valor de minOccurs definido en el esquema
int **getMaxOccurs**()	Devuelve el valor de maxOccurs definido en el esquema
String **getLocalName**()	Devuelve el nombre local del elemento
String **getNamespaceURI**()	Devuelve el URI de espacio de nombres del elemento

com.altova.xml.meta.Attribute

Método	**Finalidad**
SimpleType **getDataType**()	Devuelve el tipo del contenido del atributo
boolean **isRequired**()	Devuelve true si el atributo es obligatorio
String **getLocalName**()	Devuelve el nombre local del atributo
String **getNamespaceURI**()	Devuelve el URI de espacio de nombres del atributo

21.6.3 Usar la biblioteca de código C++ generado

Bibliotecas generadas

Nombre	Finalidad
Altova	Biblioteca base que ofrece compatibilidad con tiempo de ejecución general, idéntico para todos los esquemas
AltovaXML	Biblioteca base que ofrece compatibilidad con tiempo de ejecución para XML (MSXML o Xerces), idéntico para todos los esquemas
SuEsquema	Biblioteca que contiene declaraciones generadas a partir del esquema de entrada, que recibe el nombre del archivo de esquema o DTD
*SuEsquema*Test	Aplicación base de prueba (sólo en XMLSpy)

Implementaciones de DOM

El código C++ generado es compatible con Microsoft MSXML o con Apache Xerces 2.6 o superior, dependiendo de la opción de generación de código elegida. La sintaxis para usar el código generado es idéntica en ambos casos.

Tipos de caracteres

El código C++ generado se puede compilar con o sin compatibilidad con Unicode. Dependiendo la opción elegida, los tipos **string_type** y **tstring** se definirán como **std::string** or **std::wstring** y se componen de caracteres anchos o estrechos. Para usar caracteres Unicode en su archivo XML que no se puedan representar con el conjunto de caracteres de 8 bits actual, debe habilitar la compatibilidad con Unicode. Preste especial atención a las macros **_T()**. Esta macro permite almacenar correctamente las constantes de cadena, tanto si compila para programas Unicode como para programas no Unicode.

Tipos de datos

Esta es la asignación predeterminada de tipos de XML Schema a tipos de datos C++.

XML Schema	C++	Comentarios
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, así que se usa double
xs:float, xs:double	double	
xs:integer	__int64	xs:integer tiene un intervalo ilimitado y se asignado a __int64 por razones de eficacia
xs:nonNegativeInteger	unsigned __int64	véase el punto 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 descodificación de datos binarios se hace de forma automática

 © 2014 Altova GmbH

xs:anySimpleType	string_type	

Los tipos de XML Schema que no aparecen en esta lista son tipos derivados y se asignan al tipo de C++ equivalente a su tipo base.

Clase de documento generada

Además de clases para los tipos declarados en el esquema XML, también se genera una clase de documento. Esta clase contiene todos los elementos raíz posibles como miembros y los métodos que aparecen a continuación ("*CDoc"* es el nombre de la clase de documento propiamente dicha):

Método	**Finalidad**
static *CDoc* **LoadFromFile**(const string_type& fileName)	Carga un documento XML a partir de un archivo
static *CDoc* **LoadFromString**(const string_type& xml)	Carga un documento XML a partir de una cadena
static *CDoc* **LoadFromBinary**(const std:vector<unsigned char>& xml)	Carga un documento XML a partir de una matriz de bytes
void **SaveToFile**(const string_type& fileName, bool prettyPrint, const string_type& encoding = "UTF-8")	Guarda un documento en un archivo con la codificación indicada. Si no se indica ninguna codificación, entonces se usa UTF-8
void **SaveToFile**(const string_type& fileName, bool prettyPrint, const string_type& encoding, bool bigEndian, bool writeBOM)	Guarda un documento XML en un archivo con la codificación indicada. Para las codificaciones Unicode puede indicar el orden de bytes y la marca de orden de bytes Unicode
void **SaveToFile**(const string_type& fileName, bool prettyPrint, const string_type& encoding, const string_type& lineend) *Nota:* solo disponible en Xerces3.	Guarda un documento XML en un archivo con la codificación y el fin de línea especificados.
void **SaveToFile**(const string_type& fileName, bool prettyPrint, const string_type& encoding, bool bigEndian, bool writeBOM, const string_type& lineend) *Nota:* solo disponible en Xerces3.	Guarda un documento XML en un archivo con la codificación y el fin de línea especificados. Para las codificaciones Unicode se puede especificar la marca de orden de bytes Unicode y el orden de bytes.
string_type **SaveToString**(bool prettyPrint)	Guarda un documento XML en una cadena
std::vector<unsigned char> **SaveToBinary**(bool prettyPrint, const string_type& encoding = "UTF-8")	Guarda un documento XML en una matriz de bytes con la codificación indicada. Si no se indica ninguna codificación, entonces se usa UTF-8
std::vector<unsigned char> **SaveToBinary**(bool prettyPrint, const string_type& encoding, bool bigEndian, bool writeBOM)	Guarda un documento XML en una matriz de bytes con la codificación indicada. Para las codificaciones Unicode puede indicar el orden de bytes y la marca de orden de bytes Unicode
static *CDoc* **CreateDocument**()	Crea un documento XML nuevo vacío. Debe liberarse mediante DestroyDocument().
void **DestroyDocument**()	Destruye un documento. Se invalidan todas las referencias al documento y a sus nodos. Es necesario llamar a este método al terminar de trabajar con el documento
void **SetSchemaLocation**(const string_type& schemaLocation)	Agrega un atributo xsi:schemaLocation o xsi:noNamespaceSchemaLocation al elemento raíz. Para ello es necesario que exista un

	elemento raíz
void **SetDTDLocation**(const string_type& dtdLocation)	Agrega una declaración DOCTYPE con el ID de sistema indicado. Es necesario que exista un elemento raíz. Este método no es compatible con MSXML porque no se puede agregar una declaración DOCTYPE a un documento en memoria

Clases de tipo 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 colisiones, se anexa un número al nombre). Para los tipos simples se generan operadores de conversión y asignación. Para los tipos simples con facetas de enumeración se pueden usar los métodos **GetEnumerationValue**() y **SetEnumerationValue**(int) junto con las constantes generadas por cada valor de enumeración. Además el método **StaticInfo**() permite acceder a la información de esquema como **altova::meta::SimpleType** or **altova::meta::ComplexType** (véase más adelante).

La derivación por extensión de complexType de XML Schema da lugar a clases derivadas en la biblioteca generada.

Clases generadas de miembros atributo
Por cada miembro atributo de un tipo se genera una clase con estos métodos.

Método	Finalidad
bool **exists**()	Devuelve el valor true si existe el atributo
void **remove**()	Quita el atributo de su elemento primario
int **GetEnumerationValue**()	Se genera solo para tipos enumeration. Devuelve una de las constantes generadas para los valores posibles o **Invalid** si el valor no coincide con ningún valor enumerado en el esquema
void **SetEnumerationValue**(int)	Se genera solo para tipos enumeration. Pasa a este método una de las constantes generadas para los valores posibles para establecer el valor
altova::meta::Attribute **info**()	Devuelve un objeto para consultar información del esquema (véase más adelante)

Además se crean operadores de asignación y conversión con el tipo nativo del atributo como origen/destino para que este se pueda usar en asignaciones directamente.

Clases generadas de miembros elemento
Por cada miembro elemento de un itpo se genera una clase con estos métodos.

Método	Finalidad
MemberType **first**()	Devuelve la primera instancia del miembro elemento
MemberType operator**[]**(unsigned int index)	Devuelve el miembro elemento indicado por el índice

 © 2014 Altova GmbH

MemberType **last**()	Devuelve la última instancia del miembro elemento
MemberType **append**()	Crea un elemento nuevo y lo anexa a su elemento primario
bool **exists**()	Devuelve true si existe como mínimo un elemento
unsigned int **count**()	Devuelve el recuento de elementos
void **remove**()	Elimina todas las apariciones del elemento de su elemento primario
void **remove**(unsigned int index)	Elimina la aparición del elemento indicada por el índice
Iterator<*MemberType*> **all**()	Devuelve un objeto para las instancias repetidas del miembro elemento
int **GetEnumerationValue**()	Se genera solo para tipos enumeration. Devuelve una de las constantes generadas para los valores posibles o **Invalid** si el valor no coincide con ningún valor enumerado en el esquema
void **SetEnumerationValue**(int)	Se genera solo para tipos enumeration. Pasa a este método una de las constantes generadas para los valores posibles para establecer el valor
altova::meta::Element **info**()	Devuelve un objeto para consultar la información del esquema (véase más adelante)

Para elementos de contenido mixto o simple se crean operadores de asignación y conversión con el tipo nativo del elemento como origen/destino para que este se pueda usar en asignaciones directamente.

Crear documentos XML

La biblioteca de código generada facilita en gran medida la creación de archivos XML. Empecemos con un archivo XML de ejemplo para el esquema de ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<Library xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd"
xmlns="http://www.nanonull.com/LibrarySample" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
        <Book ISBN="0764549642" Variant="Paperback">
                <Title>The XML Spy Handbook</Title>
                <Author>Altova</Author>
        </Book>
</Library>
```

El código C++ que aparece a continuación se usó para crear este archivo. Las clases CLibrary, CLibraryType y CBookType se generan a partir del esquema y representan todo el documento, el tipo del elemento <Library> y el tipo complejo BookType, que es el tipo del elemento <Book>.

```
using namespace Library;
void Example()
{
    // create a new, empty XML document
    CLibrary libDoc = CLibrary::CreateDocument();
```

```
    // create the root element <Library> and add it to the document
    CLibraryType lib = libDoc.Library2.append();

    // create a new <Book> and add it to the library
    CBookType book = lib.Book.append();

    // set the ISBN attribute of the book
    book.ISBN = _T("0764549642");

    // set the Variant attribute of the book using an enumeration constant
    book.Variant.SetEnumerationValue( CBookVariant::k_Paperback );

    // add the <Title> and <Author> elements, and set values
    book.Title.append() = _T("The XML Spy Handbook");
    book.Author.append() = _T("Altova");

    // set the schema location (this is optional)
    libDoc.SetSchemaLocation(_T("Library.xsd"));

    // save the XML document to a file with default encoding (UTF-8). "true"
causes the file to be pretty-printed.
    libDoc.SaveToFile(_T("Library1.xml"), true);

    // destroy the document - all references to the document and any of its
nodes become invalid
    libDoc.DestroyDocument();
}
```

Observe que todos los elementos se crean llamando a append() y que a los atributos (como el atributo ISBN del ejemplo) se puede acceder como miembros de la estructura. Recuerde que debe destruir el elemento cuando termine de trabajar con él.

Leer documentos XML

El ejemplo que aparece a continuación lee el archivo que acabamos de crear y muestra los diferentes métodos que hay para leer archivos XML:

```
using namespace Library;
void Example()
{
    // load XML document
    CLibrary libDoc = CLibrary::LoadFromFile(_T("Library1.xml"));

    // get the first (and only) root element <Library>
    CLibraryType lib = libDoc.Library2.first();

    // it is possible to check whether an element exists:
    if (!lib.Book.exists())
    {
            tcout << "This library is empty." << std::endl;
            return;
    }
```

 © 2014 Altova GmbH

```
        // iteration: for each <Book>...
        for (Iterator<CBookType> itBook = lib.Book.all(); itBook; ++itBook)
        {
                // output values of ISBN attribute and (first and only) title
    element
                tcout << "ISBN: " << tstring(itBook->ISBN) << std::endl;
                tcout << "Title: " << tstring(itBook->Title.first()) << std::endl;

                // read and compare an enumeration value
                if (itBook->Variant.GetEnumerationValue() ==
CBookVariant::k_Paperback)
                        tcout << "This is a paperback book." << std::endl;

                // for each <Author>...
                for (CBookType::Author::iterator itAuthor = itBook->Author.all();
    itAuthor; ++itAuthor)
                        tcout << "Author: " << tstring(itAuthor) << std::endl;

                // alternative: use count and index
                for (unsigned int j = 0; j < itBook->Author.count(); ++j)
                        tcout << "Author: " << tstring(itBook->Author[j]) <<
    std::endl;
        }

        // destroy the document - all references to the document and any of its
    nodes become invalid
        libDoc.DestroyDocument();
    }
```

Para acceder al contenido de atributos y elementos como cadenas para el operador << cout, es necesario una conversión explícita en tstring.
También puede cargar un documento, modificarlo añadiéndole o eliminando elementos y atributos y guardarlo en el disco.

Control de errores

Los errores se notifican mediante excepciones. Estas clases de excepción se definen en el espacio de nombres altova:

Clase	Clase base	Descripción
Error	std::logic_error	Error interno de lógica de programa (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	Se produjo un error de conversión de tipos
StringParseException	ConversionException	No se puede convertir un valor del espacio léxico en espacio de valores
ValueNotRepresentableException	ConversionException	No se puede convertir un valor del espacio de valores en espacio léxico
OutOfRangeException	ConversionException	No se puede representar un valor de origen en el dominio de destino

InvalidOperationException	Exception	Se intentó una operación que no es válida en el contexto dado
DataSourceUnavailableException	Exception	Se produjo un error al cargar una instancia XML
DataTargetUnavailableException	Exception	Se produjo un error al guardar una instancia XML

Todas las clases de excepción contienen un texto de mensaje y un puntero que indica 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 está disponible) o NULL

Acceder a metadatos

La biblioteca generada permite acceder a información estática del esquema a través de estas clases. Todos los métodos se declaran como const. Los métodos que devuelven una de las clases de metadatos devuelven un objeto NULL si la propiedad correspondiente no existe.

altova::meta::SimpleType

Método	Finalidad
operador bool()	Devuelve true si este no es el simpleType NULL
operador !()	Devuelve true si este es el simpleType NULL
SimpleType **GetBaseType**()	Devuelve el tipo base de este tipo
string_type **GetLocalName**()	Devuelve el nombre local del tipo
string_type **GetNamespaceURI**()	Devuelve el URI de espacio de nombres del tipo
unsigned int **GetMinLength**()	Devuelve el valor de esta faceta
unsigned int **GetMaxLength**()	Devuelve el valor de esta faceta
unsigned int **GetLength**()	Devuelve el valor de esta faceta
unsigned int **GetTotalDigits**()	Devuelve el valor de esta faceta
unsigned int **GetFractionDigits**()	Devuelve el valor de esta faceta
string_type **GetMinInclusive**()	Devuelve el valor de esta faceta
string_type **GetMinExclusive**()	Devuelve el valor de esta faceta
string_type **GetMaxInclusive**()	Devuelve el valor de esta faceta
string_type **GetMaxExclusive**()	Devuelve el valor de esta faceta
std::vector<string_type> **GetEnumerations**()	Devuelve una lista con todas las facetas enumeration
std::vector<string_type> **GetPatterns**()	Devuelve una lista con todas las facetas pattern
WhitespaceType **GetWhitespace**()	Devuelve el valor de la faceta whitespace, que es uno de estos: Whitespace_Unknown Whitespace_Preserve Whitespace_Replace Whitespace_Collapse

altova::meta::ComplexType

Método	Finalidad
operator bool()	Devuelve true si este no es el ComplexType NULL

 © 2014 Altova GmbH

operator !()	Devuelve true si este es el ComplexType NULL
std::vector<Attribute> **GetAttributes**()	Devuelve una lista con todos los atributos
std::vector<Element> **GetElements**()	Devuelve una lista con todos los elementos
ComplexType **GetBaseType**()	Devuelve el tipo base de este tipo
string_type **GetLocalName**()	Devuelve el nombre local del tipo
string_type **GetNamespaceURI**()	Devuelve el URI de espacio de nombres del tipo
SimpleType **GetContentType**()	Devuelve el tipo simple del contenido
Element **FindElement**(const char_type* localName, const char_type* namespaceURI)	Encuentra el elemento con el nombre local y el URI de espacio de nombres indicados
Attribute **FindAttribute**(const char_type* localName, const char_type* namespaceURI)	Encuentra el atributo con el nombre local y el URI de espacio de nombres indicados

altova::meta::Element

Método	**Finalidad**
operator bool()	Devuelve true si este no es el Element NULL
operator !()	Devuelve true si este es el Element NULL
ComplexType **GetDataType**()	Devuelve el tipo del elemento. Recuerde que siempre es un tipo complejo incluso si se declaró como simple en el esquema original. Use el GetContentType() del objeto devuelto para obtener el tipo de contenido simple
unsigned int **GetMinOccurs**()	Devuelve el valor de minOccurs definido en el esquema
unsigned int **GetMaxOccurs**()	Devuelve el valor de maxOccurs definido en el esquema
string_type **GetLocalName**()	Devuelve el nombre local del elemento
string_type **GetNamespaceURI**()	Devuelve el URI de espacio de nombres del elemento

altova::meta::Attribute

Método	**Finalidad**
operator bool()	Devuelve true si este no es el Attribute NULL
operator !()	Devuelve true si este es el Attribute NULL
SimpleType **GetDataType**()	Devuelve el tipo del contenido del atributo
bool **IsRequired**()	Devuelve true si el atributo es obligatorio
string_type **GetLocalName**()	Devuelve el nombre local del atributo
string_type **GetNamespaceURI**()	Devuelve el URI de espacio de nombres del atributo

21.6.4 Usar la biblioteca de código C# generado

Bibliotecas generadas

Nombre	Finalidad
Altova	Biblioteca base que ofrece compatibilidad con tiempo de ejecución general, idéntico para todos los esquemas
AltovaXML	Biblioteca base que ofrece compatibilidad con tiempo de ejecución para XML (MSXML o Xerces), idéntico para todos los esquemas
SuEsquema	Biblioteca que ofrece declaraciones generadas a partir del esquema de entrada, que recibe el nombre del archivo de esquema o DTD
*SuEsquema*Test	Aplicación base de prueba (sólo en XMLSpy)

Implementación de DOM
El código C# generado usa la biblioteca estándar System.XML de .NET como implementación de DOM subyacente.

Tipos de datos
Esta es la asignación predeterminada de tipos de XML Schema a tipos de datos C#.

XML Schema	C#	Comentarios
xs:string	string	
xs:boolean	bool	
xs:decimal	decimal	xs:decimal tiene intervalo y precisión ilimitados. Se asigna a decimal por razones de 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 descodificación de datos binarios se hace de forma automática
xs:anySimpleType	string	

Los tipos de XML Schema que no aparecen en esta lista son tipos derivados y se asignan al tipo de C# equivalente a su tipo base.

Clase de documento generada
Además de clases para los tipos declarados en el esquema XML, también se genera la clase de documento. Contiene todos los elementos raíz posibles como miembros y los métodos que aparecen a continuación ("Doc" es el nombre de la clase de documento generada):

 © 2014 Altova GmbH

Método	Finalidad
static *Doc* **LoadFromFile**(string fileName)	Carga un documento XML a partir de un archivo
static *Doc* **LoadFromString**(string xml)	Carga un documento XML a partir de una cadena
static *Doc* **LoadFromBinary**(byte[] xml)	Carga un documento XML a partir de una matriz de bytes
void **SaveToFile**(string fileName, bool prettyPrint, string encoding = "UTF-8")	Guarda un documento en un archivo con la codificación indicada. Si no se indica ninguna codificación, entonces se usa UTF-8
void **SaveToFile**(string fileName, bool prettyPrint, string encoding, string lineend)	Guarda un documento XML en un archivo con la codificación y el fin de línea indicados.
string **SaveToString**(bool prettyPrint)	Guarda un documento XML en una cadena
byte[] **SaveToBinary**(bool prettyPrint, string encoding = "UTF-8")	Guarda un documento XML en una matriz de bytes con la codificación indicada. Si no se indica ninguna codificación, entonces se usa UTF-8
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"
void **SetSchemaLocation**(string schemaLocation)	Agrega un atributo xsi:schemaLocation o xsi:noNamespaceSchemaLocation al elemento raíz. Para ello es necesario que exista un elemento raíz
void **SetDTDLocation**(string dtdLocation)	Agrega una declaración DOCTYPE con el ID de sistema indicado. Para ello es necesario que exista un elemento raíz

Clases de tipo 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 y nombres del esquema original (en case de colisiones, se anexa un número al nombre). Para los tipos simples y complejos, se genera una propiedad **Value** para acceder al contenido textual. Para tipos simples con facetas enumeration, la propiedad **EnumerationValue** se puede usar junto con las constantes generadas por cada valor de enumeración. Además la propiedad **StaticInfo**() permite acceder a la información de esquema como **Altova.Xml.Meta.SimpleType** or **Altova.Xml.Meta.ComplexType** (véase más adelante).

La derivación por extensión de complexType de XML Schema da lugar a clases derivadas en la biblioteca generada.

Clases generadas de miembros atributo

Por cada miembro atributo de un tipo se crea una clase con estos métodos o propiedades.

Método/Propiedad	Finalidad
bool **Exists**()	Devuelve true si existe el atributo
void **Remove**()	Quita el atributo de su elemento primario
AttributeType **Value**	Establece u obtiene el valor del atributo
int **EnumerationValue**	Se genera solo para tipos enumeration. Establece u obtiene el valor del atributo usando una de las constantes generadas para los valores posibles. La lectura devuelve **Invalid** si

	el valor no coincide con ningún valor enumerado en el esquema
Altova.Xml.Meta.Attribute **Info**	Devuelve un objeto para consultar información del esquema (véase más adelante)

Clases generadas de miembros elemento

Por cada miembro elemento de un tipo se crea una clase con estos métodos o propiedades. La clase implementa la interfaz estándar System.Collections.IEnumerable, para que se pueda usar con la instrucción *foreach*.

Método/Propiedad	**Finalidad**
MemberType **First**	Devuelve la primera instancia del miembro elemento
MemberType this**[**int index**]**	Devuelve el miembro elemento indicado por el índice
MemberType **At**(int index)	Devuelve el miembro elemento indicado por el índice
MemberType **Last**	Devuelve la última instancia del miembro elemento
MemberType **Append**()	Crea un elemento nuevo y lo anexa a su elemento primario
bool **Exists**	Devuelve true si existe como mínimo un elemento
int **Count**	Devuelve el recuento de elementos
void **Remove**()	Devuelve todas las apariciones del elemento desde su elemento primario
void **RemoveAt**(int index)	Elimina todas las apariciones del elemento indicado por el índice
System.Collections.IEnumerator **GetEnumerator**()	Devuelve un objeto para las instancias repetidas del miembro elemento
MemberType **Value**	Establece u obtiene el contenido del elemento (se genera solo si el elemento puede tener contenido simple o mixto)
int **EnumerationValue**	Se genera solo para tipos enumeration. Establece u obtiene el valor del elemento usando una de las constantes generadas para los valores posibles. La lectura devuelve **Invalid** si el valor no coincide con ningún valor enumerado en el esquema
Altova.Xml.Meta.Element **Info**	Devuelve un objeto para consultar la información del esquema (véase más adelante)

Crear documentos XML

La biblioteca de código generada facilita la creación de archivos XML. Empecemos con un archivo XML de ejemplo para el esquema de ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<Library xsi:schemaLocation="http://www.nanonull.com/LibrarySample Library.xsd"
xmlns="http://www.nanonull.com/LibrarySample" xmlns:xsi="http://www.w3.org/2001/
XMLSchema-instance">
        <Book ISBN="0764549642" Variant="Paperback">
```

 © 2014 Altova GmbH

```
            <Title>The XML Spy Handbook</Title>
            <Author>Altova</Author>
        </Book>
</Library>
```

El código C# que aparece a continuación se usó para crear este archivo. Las clases Library2, LibraryType y BookType se generan a partir del schema y representan todo el documento, el tipo del elemento <Library> y el tipo complejo BookType, que es el tipo del elemento <Book>. Observe que la clase Library se renombró automáticamente y ahora se llama Library2 para evitar posibles conflictos con el nombre de espacio de nombres de la biblioteca.

```
    using Library;
    ...
        protected static void Example()
        {
                // create a new, empty XML document
                Library2 libDoc = Library2.CreateDocument();

                // create the root element <Library> and add it to the document
                LibraryType lib = libDoc.Library3.Append();

                // create a new <Book> and add it to the library
                BookType book = lib.Book.Append();

                // set the ISBN attribute of the book
                book.ISBN.Value = "0764549642";

                // set the Variant attribute of the book using an enumeration
constant
                book.Variant.EnumerationValue = BookVariant.EnumValues.ePaperback;

                // add the <Title> and <Author> elements, and set values
                book.Title.Append().Value = "The XML Spy Handbook";
                book.Author.Append().Value = "Altova";

                // set the schema location (this is optional)
                libDoc.SetSchemaLocation("Library.xsd");

                // save the XML document to a file with default encoding (UTF-8).
"true" causes the file to be pretty-printed.
                libDoc.SaveToFile("Library1.xml", true);
        }
```

Observe que todos los elementos se crean llamando a append() y que a los atributos (como el atributo ISBN del ejemplo) se puede acceder como miembros de la estructura.

Leer documentos XML

El ejemplo que aparece a continuación lee el archivo que acabamos de crear y muestra los diferentes métodos que hay para leer archivos XML:

```
    using Library;
    ...
```

```
        protected static void Example()
        {
                // load XML document
                Library2 libDoc = Library2.LoadFromFile("Library1.xml");

                // get the first (and only) root element <Library>
                LibraryType lib = libDoc.Library3.First;

                // it is possible to check whether an element exists:
                if (!lib.Book.Exists)
                {
                        Console.WriteLine("This library is empty.");
                        return;
                }

                // iteration: for each <Book>...
                foreach (BookType book in lib.Book)
                {
                        // output values of ISBN attribute and (first and only)
title element
                        Console.WriteLine("ISBN: " + book.ISBN.Value);
                        Console.WriteLine("Title: " + book.Title.First.Value);

                        // read and compare an enumeration value
                        if (book.Variant.EnumerationValue ==
BookVariant.EnumValues.ePaperback)
                                Console.WriteLine("This is a paperback book.");

                        // for each <Author>...
                        foreach (xs.stringType author in book.Author)
                                Console.WriteLine("Author: " + author.Value);

                        // alternative: use count and index
                        for (int j = 0; j < book.Author.Count; ++j)
                                Console.WriteLine("Author: " + book.Author[j].Value);
                }
        }
```

Observe que el tipo xs.stringType del miembro book.Author es una clase generada para todos los elementos que tengan contenido de cadena simple. Para los demás tipos de XML Schema se generan clases similares.

Si lo deesa, también puede cargar un documento, modificarlo añadiéndole elementos y atributos y después guardar el archivo en el disco.

Control de errores

Los errores se notifican mediante excepciones. Estas clases de excepción se definen en el espacio de nombres Altova:

Clase	Clase base	Descripción
ConversionException	Exception	Error de conversión de tipos
StringParseException	ConversionExcepti	No se puede convertir un valor del espacio léxico

 © 2014 Altova GmbH

	on	en espacio de valores
DataSourceUnavailableException	System.Exception	Se produjo un error al cargar una instancia XML
DataTargetUnavailableException	System.Exception	Se produjo un error al guardar una instancia XML

También se suelen usar estas excepciones de .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 válidos o se produjo un error de conversión de tipos
System.FormatException	No se puede convertir un valor del espacio léxico en espacio de valores
System.InvalidCastException	No se puede convertir un valor en otro tipo
System.OverflowException	Un valor de origen no se puede representar en el dominio de destino

Acceder a metadatos

La biblioteca generada permite acceder a información estática del esquema a través de las clases que aparecen a continuación. Las propiedades que devuelven una de las clases de metadatos devuelven null si la propiedad correspondiente no existe.

Altova.Xml.Meta.SimpleType

Método/Propiedad	Finalidad
SimpleType **BaseType**	Devuelve el tipo base de este tipo
string **LocalName**	Devuelve el nombre local del tipo
string **NamespaceURI**	Devuelve el URI de espacio de nombres del tipo
XmlQualifiedName **QualifiedName**	Devuelve el nombre completo de este tipo
int **MinLength**	Devuelve el valor de esta faceta
int **MaxLength**	Devuelve el valor de esta faceta
int **Length**	Devuelve el valor de esta faceta
int **TotalDigits**	Devuelve el valor de esta faceta
int **FractionDigits**	Devuelve el valor de esta faceta
string **MinInclusive**	Devuelve el valor de esta faceta
string **MinExclusive**	Devuelve el valor de esta faceta
string **MaxInclusive**	Devuelve el valor de esta faceta
string **MaxExclusive**	Devuelve el valor de esta faceta
string[] **Enumerations**	Devuelve una lista con todas las facetas enumeration
string[] **Patterns**	Devuelve una lista con todas las facetas pattern
WhitespaceType **Whitespace**	Devuelve el valor de la faceta whitespace, que es uno de estos: Unknown Preserve Replace Collapse

Altova.Xml.Meta.ComplexType

Método/Propiedad	Finalidad

Attribute[] **Attributes**	Devuelve una lista con todos los atributos
Element[] **Elements**	Devuelve una lista con todos los elementos
ComplexType **BaseType**	Devuelve el tipo base de este tipo
string **LocalName**	Devuelve el nombre local del tipo
string **NamespaceURI**	Devuelve el URI de espacio de nombres del tipo
XmlQualifiedName **QualifiedName**	Devuelve el nombre completo de este tipo
SimpleType **ContentType**	Devuelve el tipo simple del contenido
Element **FindElement**(string localName, string namespaceURI)	Encuentra el elemento con el nombre local y el URI de espacio de nombres indicados
Attribute **FindAttribute**(string localName, string namespaceURI)	Encuentra el atributo con el nombre local y el URI de espacio de nombres indicados

Altova.Xml.Meta.Element

Método/Propiedad	**Finalidad**
ComplexType **DataType**	Devuelve el tipo del elemento. Recuerde que es siempre un tipo complejo incluso si se declaró como tipo simple en el esquema original. Use la propiedad getContentType del objeto devuelto para obtener el tipo de contenido simple.
int **MinOccurs**	Devuelve el valor de minOccurs definido en el esquema
int **MaxOccurs**	Devuelve el valor de maxOccurs definido en el esquema
string **LocalName**	Returns the local name of the element
string **NamespaceURI**	Devuelve el nombre local del elemento
XmlQualifiedName **QualifiedName**	Devuelve el nombre completo del elemento

Altova.Xml.Meta.Attribute

Método/Propiedad	**Finalidad**
SimpleType **DataType**	Devuelve el tipo del contenido del atributo
bool **Required**()	Devuelve true si el atributo es obligatorio
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

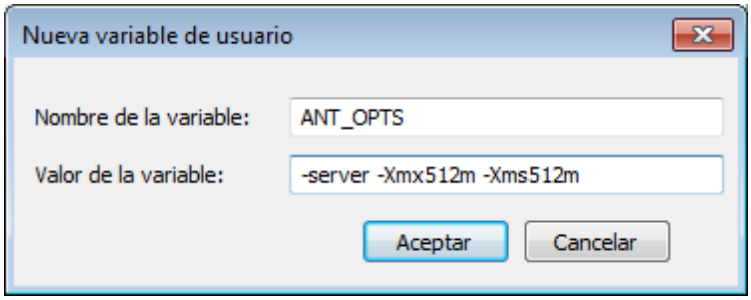
© 2014 Altova GmbH

21.7 Consejos para la generación de código

Excepciones de memoria insuficiente durante la compilación de Java y cómo resolverlas

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. Añada estos valores a la variables.



- 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.

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\MapForce2014\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.

Compatibilidad con XML Schema

Estas construcciones de XML Schema se traducen en código:

- Espacios de nombres XML: se asignan a espacios de nombres en el lenguaje de programación de destino

Tipos simples

- Tipos de XML Schema integrados: se asignan a tipos o clases primitivos nativos de la biblioteca de tipos de Altova
- Derivados por extensión
- Derivados por restricción

- Facetas
- Enumeraciones
- Patrones

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 avanzadas de XML Schema **no son compatibles** (o no totalmente) con MapForce cuando se generan clases **contenedoras** opcionales:

- Comodines: xs:any y xs:anyAttribute
- Modelos de contenido (sequence, choice, all): el compositor de nivel superior está disponible en SPL, pero las clases generadas no lo imponen
- valores default y fixed para atributos: están disponibles en SP, pero las clases generadas no los establecen ni los imponen
- los atributos xsi:type, tipos abstractos: se generan métodos SetXsiType y el usuario debe llamarlos
- Tipos union: no todas las combinaciones son compatibles
- Grupos de sustitución: algunos son compatibles (se resuelven como "choice")
- el atributo nillable="true" y xsi:nil
- restricciones unique
- key y keyref

 © 2014 Altova GmbH

21.8 Opciones del generador de código

La opción **Herramientas | Opciones** permite definir diferentes opciones de configuración de MapForce.

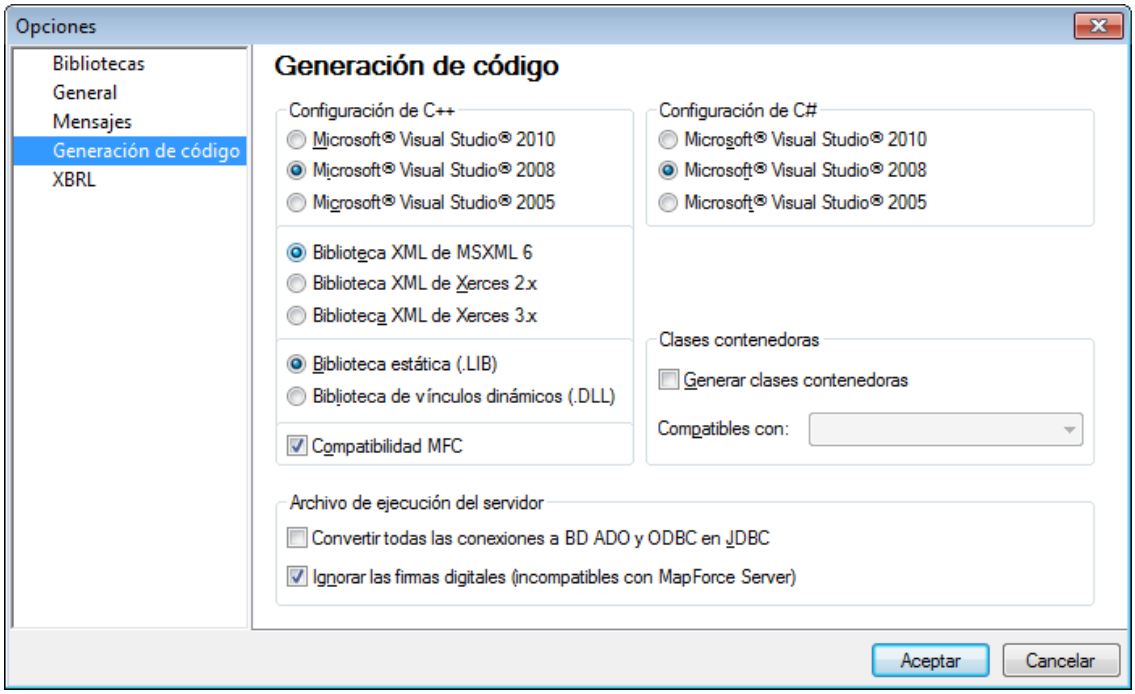
En la pestaña *Generación de código* puede definir opciones de:

Configuración de C++

- Archivo de proyecto Microsoft Visual Studio 2010, 2008, 2005.
- Generar código usando la biblioteca XML de MSXML 6.0 o Apache Xerces
- Generar bibliotecas estáticas o bibliotecas de vínculos dinámicas
- Generar código con o sin compatibilidad MFC

Configuración de C#
Seleccione el tipo de archivo de proyecto que desea generar:

- Archivo de proyecto Microsoft Visual Studio 2010, 2008 o 2005.



Configuración de Java
Las implementaciones de servicio web generadas por MapForce admiten el protocolo SOAP 1.1 o SOAP 1.2, dependiendo del enlace seleccionado del archivo WSDL. Apache Axis2 admite tanto SOAP 1.1 como SOAP 1.2.

Clases contenedoras:
Si desea integrar la aplicación de MapForce generada en su propio programa y acceder a las instancias XML de origen/destino usando clases contenedoras como las generadas por XMLSpy, entonces puede habilitar aquí la generación de estas clases.

Nota: no se admiten las clases contenedoras para estructuras no XML (como EDI, HL7, etc.).

21.9 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 **...\MapForce2014\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.

 © 2014 Altova GmbH

21.9.1 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
$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: **]**.

21.9.2 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

 © 2014 Altova GmbH

	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.

21.9.3 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 **$** se usa cuando se **declara** o **usa** una variable, un nombre de variable siempre tiene el prefijo **$**.
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**.

 © 2014 Altova GmbH

21.9.4 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	integer	1 para la DTD, 2 para el esquema XML
$TheLibrary	Library	La biblioteca derivada del esquema XML o de la DTD
$module	string	Nombre del esquema de origen sin extensión
$outputpath	string	La ruta de acceso de salida indicada por el usuario o la ruta de acceso de salida predeterminada

Para generación de código **C++** solamente:

Nombre	Tipo	Descripción
$domtype	integer	1 para MSXML, 2 para Xerces
$XercesVersion	integer	2 para Xerces 2.x, 3 para Xerces 3.x
$libtype	integer	1 para biblioteca estática LIB, 2 para DLL
$mfc	boolean	True si se habilita la compatibilidad con MFC
$vc6project	boolean	True si se deben generar archivos de proyecto de Visual C++
$VS2005Project	boolean	True si se deben generar archivos de proyectos de Visual C++ 2005
$VS2008Project	boolean	True si se deben generar archivos de proyecto de Visual C++ 2008
$VS2010Project	boolean	True si se deben generar archivos de proyecto de Visual C++ 2010

Para generación de código **C#** solamente:

Nombre	Tipo	Descripción
$CreateVS2005Project	boolean	True si se deben generar archivos de proyecto de Visual Studio 2005
$CreateVS2008Project	boolean	True si se deben generar archivos de proyecto de Visual Studio 2008
$CreateVS2010Project	boolean	True si se deben generar archivos de proyecto de Visual Studio 2010

21.9.5 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"
```

 © 2014 Altova GmbH

21.9.6 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

21.9.7 Condiciones

En SPL puede usar instrucciones `if` estándar. Esta es la sintaxis para estas instrucciones:

```
if condición
   instrucciones
else
   instrucciones
endif
```

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`.

 © 2014 Altova GmbH

21.9.8 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.

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).

 © 2014 Altova GmbH

21.9.9 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)

Declaración de subrutina

Subrutinas

Ejemplo de sintaxis:

```
Sub SimpleSub()
... líneas de código
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 comienza inmediatamente después del paréntesis de cierre.
- **EndSub** denota el final del bloque de código.

Nota importante: no está permitida la **declaración** recurrente o en cascada de subrutinas. Es decir, una subrutina no pude contener otra subrutina.

Parámetros

Puede pasar parámetros por medio de procedimientos usando esta sintaxis:

- Todos los parámetros deben ser variables.
- Las variables deben llevar el prefijo **$**.
- Las variables locales están definidas en una subrutina.
- Las variables locales se declaran de forma explícita, fuera de las subrutinas.
- Use la coma (**,**) para separar varios parámetros dentro de los paréntesis.
- Los parámetros pueden pasar valores.

Parámetros: cómo pasar valores

Los parámetros se pueden pasar de dos formas distintas (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** indica que el parámetro se pasa por valor. Recuerde que la mayoría de los objetos se pueden pasar sólo por referencia.

- **ByRef** indica que el parámetro se pasa por referencia. Este es el método predeterminado si no se indica la palabra clave ByVal ni ByRef.

Valores devueltos de una función

Para devolver un valor desde una subrutina, use la instrucción **return**. A esta función se le 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
]
```

Invocación de subrutina

Para invocar a una subrutina utilice **call** seguido del nombre de procedimiento y los parámetros.

```
Call SimpleSub()
```

o

```
Call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
```

Invocación de funciones

Para invocar una función (es decir, cualquier subrutina que contenga una instrucción **return**) basta con usar su nombre dentro de una expresión. No use la instrucción **call** para llamar a funciones.

Ejemplo:

```
$QName = MakeQualifiedName($namespace, "entry")
```

Ejemplo de subrutina

En la imagen siguiente la declaración e invocación de subrutinas aparece resaltada en color amarillo.

 © 2014 Altova GmbH

A sample SPL file:

```
[create $outputpath & $module & "output.txt"

' define sub SimpleSub()
sub SimpleSub()
]SimpleSub() called
[endsub

' execute sub SimpleSub()
call SimpleSub()

$ParamByValue    = "Original Value"
]ParamByValue    = [=$ParamByValue]
[$ParamByRef     = "Original Value"
]ParamByRef      = [=$ParamByRef]

' define 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

' run sub CompleteSub()
call CompleteSub( "FirstParameter", $ParamByValue, $ParamByRef )
]
ParamByValue=[=$ParamByValue]
ParamByRef=[=$ParamByRef]
[
close
]
```

El mismo código de muestra:

```
[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
]
```


© 2014 Altova GmbH

21.9.10 Tipos integrados

Esta sección describe las propiedades de los tipos integrados usados en las variables predefinidas que describen el esquema analizado.

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)

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

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
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.

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/

 © 2014 Altova GmbH

		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

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

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
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

 © 2014 Altova GmbH

Altova MapForce 2014

The MapForce API

22 The MapForce API

The COM-based API of MapForce enables clients to easily access the functionality of MapForce. As a result, it is now possible to automate a wide range of tasks.

MapForce follows the common specifications for automation servers set out by Microsoft. It is possible to access the methods and properties of the API de MapForce from common development environments, such as those using .NET, C++ and VisualBasic, and with scripting languages like JavaScript and VBScript.

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.
- See Error handling for details of how to avoid annoying error messages.
- Free references explicitly, if using languages such as C++.

To integrate your version of MapForce2014 into the Microsoft Visual Studio versions 2005, 2008 and 2010 you need to do the following:

- Install Microsoft Visual Studio
- Install MapForce (Enterprise or Professional Edition)
- Download and run the MapForce Visual Studio .NET Edition integration for Microsoft Visual Studio .NET package. This package is available on the MapForce (Enterprise and Professional Editions) download page at www.altova.com.

- Example files for the integration package are installed into the respective language folders below the **c:\Program Files\Altova\MapForce2014\Examples\ActiveX** installation folder.

Please note:
The manager control (e.g. MapForceControl) of ActiveX Controls can be instantiated only once in a process (even if it is destroyed, you cannot re-create it. To avoid this create the MapForce Control and its parent window only once and then show/hide the window as needed.

Mapforce integration and deployment on client computers:
If you create a .NET application and intend to distribute it to other clients, you will have to install the following on the client computer(s):

- The MapForce application
- The MapForce integration package.
- The custom integration code.

 © 2014 Altova GmbH

22.1 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:

- The object model
- Example: Code-Generation
- Example: Mapping Execution
- Example: Project Support
- Error handling

22.1.1 Object model

The starting point for every application which uses the API de MapForce is the `Application` object.

To create an instance of the `Application` object, call `CreateObject("MapForce.Application")` from VisualBasic, or a similar function from your preferred development environment, to create a COM object. There is no need to create any other objects to use the complete API de MapForce. All other interfaces are accessed through other objects, with the `Application` object as the starting point.

The application object consists of the following parts (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
```

Once you have created an `Application` object, you can start using the functionality of MapForce. You will generally either open an existing `Document`, create a new one, or generate code for, or from, this document.

 © 2014 Altova GmbH

22.1.2 Example: Code-Generation

See also
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";
       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 ---------------------
```

 © 2014 Altova GmbH

22.1.3 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 )
      {
         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 )
```

 © 2014 Altova GmbH

```
   {
      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\\MapForce2014\\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" );
    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 );
}
```

 © 2014 Altova GmbH

22.1.4 Example: Project Support

See also
Code Generation

The following JScript example shows how you can use the MapForce project and project-item objects of the API de MapForce to automate complex tasks. Depending on your installation you might need to change the value of the variable `strSamplePath` to the example 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, you should comment out the lines that insert the WebService project. Users of the Basic edition will not have access to project-related functions at all.

```
// ///////////// global variables /////////////////
var objMapForce = null;
var objWshShell = null;
var objFSO = null;

// !!! adapt the following path to your needs. !!!
var strSamplePath = "C:\\Documents and Settings\\<username>\\My Documents\
\Altova\\MapForce2014\\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
					strGlobalText += strTab + "  code generation
language is " +
                                                objItem.CodeGenSettings_Langu
age +
                                                " 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";
			}
```

 © 2014 Altova GmbH

```
                        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);
        objFolder1.GenerateCode();
}

// ////////////////////// MAIN ////////////////////////////

CreateGlobalObjects();
objMapForce.Visible = true;

LoadSampleProject();
CreateNewProject();
```

```
GenerateCodeForNewProject();

// uncomment to shut down application when script ends
// objMapForce.Visible = false;
```

22.1.5 Error handling

The API de MapForce 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.

VisualBasic, 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 API de MapForce, 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 API de MapForce in different development environments.

VisualBasic
A common way to handle errors in VisualBasic 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. VisualBasic fills its own Err object with the information from the IErrorInfo interface.

Example:

```
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.

Example:

```
 function Generate()
 {
```

 © 2014 Altova GmbH

```
  // 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();
  }
 }
```

22.1.6 Programming Languages

Programming languages differ in the way they support COM access. A few examples for the most frequently used languages (*links below*) will help you get started. The code listings in this section are available in files in the API Examples folder and can be tested straight away. The path to the API Examples folder is given below:

Windows XP	`C:/Documents and Settings/<username>/My Documents/ Altova/MapForce2014/MapForceExamples/API/`
Windows Vista, Windows 7	`C:/Users/<username>/Documents/ Altova/MapForce2014/MapForceExamples/API/`

C#
C# can be used to access the Application API functionality. The code listings show how to access the API for certain basic functionality.

Java
The MapForce API can be accessed from Java code. This section explains how some basic MapForce functionality can be accessed from Java code. It is organized into the following sub-sections.

JScript
The JScript listings demonstrate the following basic functionality.

C#

The C# programming language can be used to access the Application API functionality. You could use Visual Studio 2008 or Visual Studio 2010 to create the C# code, saving it in a Visual Studio project. Create the project as follows:

1. In Microsoft Visual Studio, add a new project using **File | New | Project**.
2. Add a reference to the MapForce Type Library by clicking **Project | Add Reference**. The Add Reference dialog pops up, displaying a list of installed COM components. Select the MapForce Type Library component from the list to add it.
3. Enter the code you want.
4. Compile the code and run it.

Example C# project
Your MapForce package contains an example C# project, which is located in the C# folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/ Altova/MapForce2014/MapForceExamples/API/`
Windows Vista, Windows 7	`C:/Users/<username>/Documents/ Altova/MapForce2014/MapForceExamples/API/`

You can compile and run the project from within Visual Studio 2008 or Visual Studio 2010.

 © 2014 Altova GmbH

Platform configuration
If you have a 64-bit operating system and are using a 32-bit installation of MapForce, you must 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…>**)

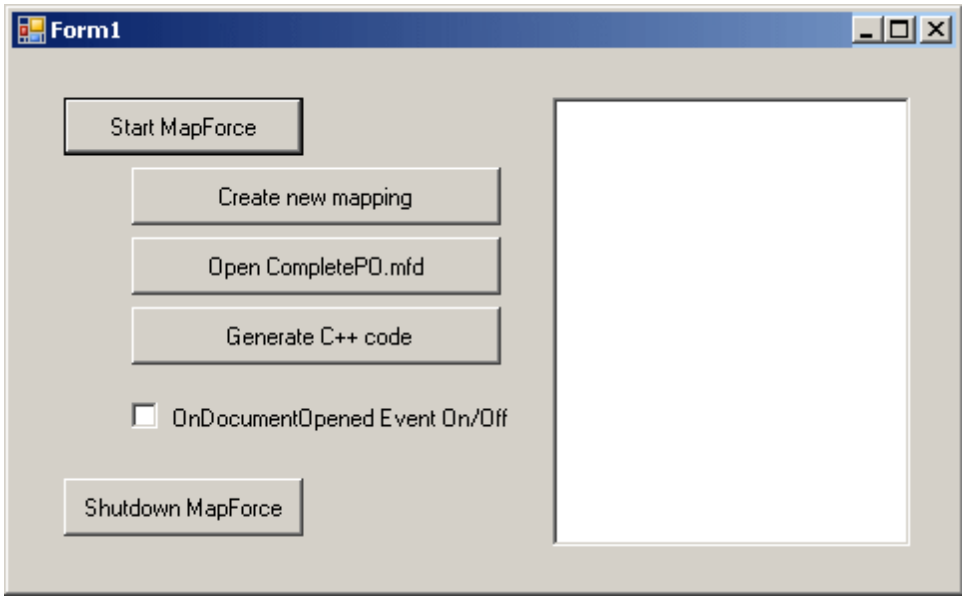
File List:

Readme.txt	This file
AutomateMapForce.csproj	Visual Studio 2008 and 2010 project file
AutomateMapForce_VS2008.sln	Visual Studio 2008 solution file
AutomateMapForce_VS2010.sln	Visual Studio 2010 solution file
Program.cs	C# example source code

Properties:
Form1.cs
Form1.Designer.cs
Form1.resx

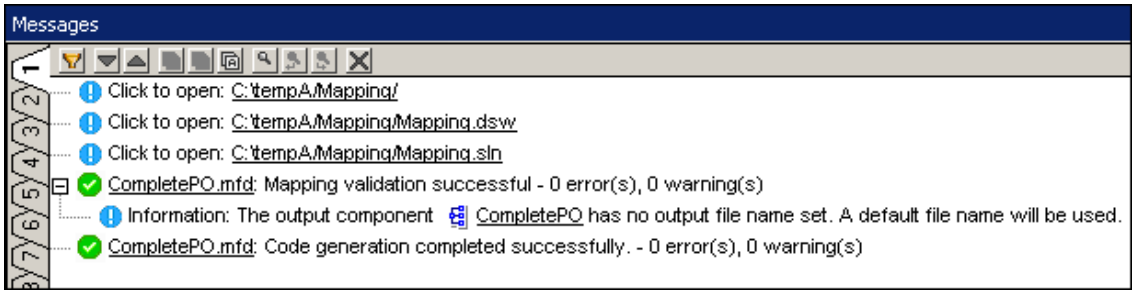
What the example does
The example shows a simple user interface (*screenshot below*) with buttons that invoke basic MapForce operations:

- Starts MapForce
- Creates a new mapping design
- Opens the CompletePO.mfd file from the ...\MapForceExamples folder
- Generates C++ code in a temp directory
- Shuts down MapForce



Compiling and running the example
Double-click the file `AutomateMapForce_VS2008.sln` (for Visual Studio 2008) or the file `AutomateMapForce_VS2010.sln` (for Visual Studio 2010). Alternatively the file can be opened from within Visual Studio (with **File | Open | Project/Solution**). To compile and run the example, select **Debug | Start Debugging** or **Debug | Start Without Debugging**.

Clicking the GenerateC++ code button generates code for the previously opened CompletePO.mfd mapping.



Code listing of Form1.cs
The listing is commented for ease of understanding. Parts of the code are also presented separately in the sub-sections of this section, according to the Application API functionality they access.

The code essentially consists of a series of handlers for the buttons in the user interface shown in the screenshot above.

```
using System;
using System.Collections.Generic;
```

 © 2014 Altova GmbH

```
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)
        {
            // locate examples installed with MapForce.
            // REMARK: You might need to adapt this if you have a different
major version of the product.
            strExamplesFolder =
Environment.GetEnvironmentVariable("USERPROFILE") + "\\My Documents\\Altova\
\MapForce2012\\MapForceExamples\\";
        }

        // 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;

                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
            {
                String strError = "";
                MapForceLib.Document doc = MapForce.ActiveDocument;

                listBoxMessages.Items.Add("Active document " + doc.Name);
                MapForceLib.ErrorMarkers errMarkers =
```

 © 2014 Altova GmbH

```
doc.GenerateCodeEx(MapForceLib.ENUMProgrammingLanguage.eCpp);

                System.Collections.IEnumerator errMarkersCollection =
errMarkers.GetEnumerator();

                bool bEmpty = true;
                while (errMarkersCollection.MoveNext())
                {
                    bEmpty = false;
                    Object obj = errMarkersCollection.Current;

                    if (obj is MapForceLib.ErrorMarker)
                        strError = ((MapForceLib.ErrorMarker)obj).Text;
                    listBoxMessages.Items.Add("Error text: " + strError);
                }

                if (bEmpty)
                    listBoxMessages.Items.Add("Code generation completed
successfully.");
            }
            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);
        // 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(handleOnDocumentOpe
ned);
                else
                    MapForce.OnDocumentOpened -= new
MapForceLib._IApplicationEvents_OnDocumentOpenedEventHandler(handleOnDocumentOpe
ned);
            }
        }
    }
}
```

Java

The Application API can be accessed from Java code. To allow accessing the MapForce automation server directly from Java code, the libraries listed below must reside in the `classpath`. They are installed in the folder: `JavaAPI` in the MapForce application folder.

- `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

Note: In order to use the Java API, the DLL and jar files must be in the Java Classpath.

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 of the example Java project, see the section, Example Java Project.

Rules for mapping the Application API names to Java

 © 2014 Altova GmbH

The rules for mapping between the Application API and the Java wrapper are 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`

The following section explains how some basic MapForce functionality can be accessed from Java code.

- Example Java Project

Example Java Project

The MapForce installation package contains an example Java project, located in the Java folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/ Altova/MapForce2014/MapForceExamples/API/`
Windows Vista, Windows 7	`C:/Users/<username>/Documents/ Altova/MapForce2014/MapForceExamples/API/`

This folder contains Java examples for the MapForce API. 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:

`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 JDK 1.5 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.

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 F1to 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 MapForce Java-COM bridge
```

 © 2014 Altova GmbH

```
import com.altova.automation.MapForce.*;
import com.altova.automation.MapForce.Enums.ENUMProgrammingLanguage;

/**
 * A simple example that starts MapForce 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.
         String strExamplesFolder = System.getenv("USERPROFILE") + "\\My
Documents\\Altova\\MapForce2012\\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();
          if (element instanceof ErrorMarker)
             sError = ((ErrorMarker)element).getText();
          System.out.print("Error text: " + sError + "\n");
      }
   }
}
```

JScript

This section contains listings of JScript code that demonstrate the following basic functionality:

Example files

The code is available in example files that you can test as is or modify to suit your needs. The JScript example files are located in the `JScript` folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/ Altova/MapForce2014/MapForceExamples/API/`
Windows Vista, Windows 7	`C:/Users/<username>/Documents/ Altova/MapForce2014/MapForceExamples/API/`

 © 2014 Altova GmbH

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 newer versions of Windows (XP, Vista, 7) 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 newer versions of Windows (XP, Vista, 7) will execute the script. After the script is executed, the command console gets closed automatically.

`DocumentAccess.js`	Shows how to open, iterate and close documents
`GenerateCode.js`	Shows how to invoke code generation using JScript
`Start.js`	Start Mapforce registered as an automation server or connect to a running instance.
`Readme.txt`	This file.

Start application

The JScript below starts the application and shuts it down. If an instance of the application is already running, the running instance will be returned.

Script listing
The JScript listing below is explained with comments in the 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)
{
   Exit("Can't access or create MapForce.Application");
}

// 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
```

Running the script
The JScript code listed above is available in the file `Start.js` located in the `JScript` folder of the API Examples folder:

Windows XP	C:/Documents and Settings/<username>/My Documents/ Altova/MapForce2014/MapForceExamples/API/
Windows Vista, Windows 7	C:/Users/<username>/Documents/ Altova/MapForce2014/MapForceExamples/API/

To run the script, start it from a command prompt window or from Windows Explorer.

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.

Script listing
The JScript listing below is explained with comments in the 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)
{
   Exit("Can't access or create MapForce.Application");
}

// 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");
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\MapForce2012\\MapForceExamples\\";

objMapForce.Documents.OpenDocument(strExampleFolder + "CompletePO.mfd");
objMapForce.Documents.OpenDocument(strExampleFolder +
"Altova_Hierarchical_DB.mfd");
```

 © 2014 Altova GmbH

```
// ***************************** 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
```

Running the script

The JScript code listed below is available in the file `DocumentAccess.js` located in the `JScript` folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/ Altova/MapForce2014/MapForceExamples/API/`
Windows Vista, Windows 7	`C:/Users/<username>/Documents/ Altova/MapForce2014/MapForceExamples/API/`

To run the script, start it from a command prompt window or from Windows Explorer.

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.

Script listing

The JScript listing below is explained with comments in the 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)
{
   Exit("Can't access or create MapForce.Application");
}

// 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");
strExampleFolder = objWshShell.ExpandEnvironmentStrings("%USERPROFILE%") + "\\My
Documents\\Altova\\MapForce2012\\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"
****************************************

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";
```

 © 2014 Altova GmbH

```
   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
```

Running the script

The JScript code listed below is available in the file `GenerateCode.js` located in the `JScript` folder of the API Examples folder:

Windows XP	`C:/Documents and Settings/<username>/My Documents/ Altova/MapForce2014/MapForceExamples/API/`
Windows Vista, Windows 7	`C:/Users/<username>/Documents/ Altova/MapForce2014/MapForceExamples/API/`

To run the script, start it from a command prompt window or from Windows Explorer.

22.2 Object Reference

Object Hierarchy

```
Application
   Options
   Project
      ProjectItem
   Documents
      Document
         MapForceView
         Mapping
            Component
               Datapoint
            Components
            Connection
         Mappings
   ErrorMarkers
      ErrorMarker
AppOutputLines
   AppOutputLine
      AppOutputLines
         ...
      AppOutputLineSymbol

Enumerations
```

Description
This section contains the reference of the API de MapForce 3.0 Type Library.

© 2014 Altova GmbH

22.2.1 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.

Events
Events

Properties and Methods
Properties to navigate the object model:
Application
Parent
Options
Project
Documents

Application status:
Visible
Name
Quit
Status
WindowHandle

MapForce designs:
NewDocument
OpenDocument
OpenURL
ActiveDocument

MapForce projects:
NewProject (Enterprise or Professional edition is required)
OpenProject (Enterprise or Professional edition is required)
ActiveProject (Enterprise or Professional edition is required)

MapForce code generation:
HighlightSerializedMarker

Global resources:
GlobalResourceConfig
GlobalResourceFile

Version information:
Edition
IsAPISupported
MajorVersion
MinorVersion

Examples
The following examples show how the automation interface of MapForce can be accessed from different programming environments in different languages.

```
' ------------------- begin VBA example ---------------------
' create a new instance of <SPY-MAP>.
Dim objMapForce As Application
Set objMapForce = CreateObject("MapForce.Application")
' -------------------- end example ---------------------

' ------------------- begin VBScript example ---------------------
' access a running, or create a new instance of MapForce.
' works with scripts running in the Windows scripting host.
Set objMapForce = GetObject("MapForce.Application");
' -------------------- end example ---------------------

// ------------------- begin JScript example ---------------------
// Access a running, or create a new instance of <MapForce
// works with scripts executed in the Windows scripting host
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"); }
// -------------------- end example ---------------------
```

Events

This object supports the following events:

OnDocumentOpened
OnProjectOpened
OnShutdown

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.

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.

OnShutdown

Event: OnShutdown ()

Description

 © 2014 Altova GmbH

This event is triggered when the application is shutting down.

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.

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.

Application

Property: `Application` as `Application` (read-only)

Description
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.

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.

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.

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.

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.

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.

 © 2014 Altova GmbH

IsAPISupported

Property: `IsAPISupported` as `Boolean` (read-only)

Description
Returns whether the API is supported in this version of MapForce.

Errors
1001 Invalid address for the return parameter was specified.

MajorVersion

Property: `MajorVersion` as `Long` (read-only)

Description
The major version number of the product, e.g. 2006 for 2006 R2 SP1.

Errors
1000 The application object is no longer valid.
1001 Invalid address for the return parameter was specified.

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.

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.

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.

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.

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.

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.

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.

 © 2014 Altova GmbH

1001 Invalid address for the return parameter was specified.

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.

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.

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.

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.

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.

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).

Fal se 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.

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.

© 2014 Altova GmbH

22.2.2 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`

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.

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.

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.

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.

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.
4101 Invalid address for the return parameter was specified.

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.

 © 2014 Altova GmbH

4101 Invalid address for the return parameter was specified.

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.

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.

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.

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.

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.

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.

GetLineTextEx

Method: `GetLineTextEx` (*`psTextPartSeperator`* as `String`, *`psLineSeperator`* as `String`) as `String`

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.

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.

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

 © 2014 Altova GmbH

part, line, tab and item separators.

Errors

4100 The object is no longer valid.
4101 Invalid address for the return parameter was specified.

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.

22.2.3 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`

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.

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.

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.
4001	Invalid address for the return parameter was specified.

Parent

Property: `Parent` as `AppOutputLine` (read-only)

Description
The parent object according to the object model.

 © 2014 Altova GmbH

Errors

4000	The object is no longer valid.
4001	Invalid address for the return parameter was specified.

22.2.4 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`

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.

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.

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.

© 2014 Altova GmbH

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.

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.

22.2.5 Component

A Component represents a MapForce component.

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`

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.

CanChangeInputInstanceFile

Property: CanChangeInputInstanceFile as Boolean (read-only)

Description
Indicates if the input instance file name can be changed.

 © 2014 Altova GmbH

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.

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.

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.

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 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.

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.

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.

 © 2014 Altova GmbH

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.

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.

InputInstanceFile

Property: `InputInstanceFile` as `String`

Description
Gets or sets the component's input instance file.

Errors
1200 The object is no longer valid.
1201 Invalid address for the return parameter was specified.

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.

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.

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.

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.

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.

Preview

Property: Preview as `Boolean`

Description
Gets or sets, if the component is the current preview component.

 © 2014 Altova GmbH

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 component as preview component instead.

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.

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.

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.

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.

© 2014 Altova GmbH

22.2.6 Components

Represents a collection of Component objects.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Iterating through the collection:
`Count`
`Item`

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.

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.

Item

Property: `Item` (*`nIndex`* as `Integer`) as `Component` (read-only)

Description
Retrieves the component at `nIndex` from the collection. Indices start with 1.

Errors

1200	The object is no longer valid.
1201	Invalid address for the return parameter was specified.

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.

© 2014 Altova GmbH

22.2.7 Connection

A Connection object represents a connector between two components.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Properties
`ConnectionType`

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.

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.

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.

22.2.8 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`

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.

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.

Parent

Property: `Parent` as `Component` (read-only)

Description
The parent object according to the object model.

Errors

 © 2014 Altova GmbH

2000 The object is no longer valid.
2001 Invalid address for the return parameter was specified.

22.2.9 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

 © 2014 Altova GmbH

Events

This object supports the following events:

OnDocumentClosed
OnModifiedFlagChanged

OnDocumentClosed

Event: OnDocumentClosed (*i_objDocument* as Document)

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.

OnModifiedFlagChanged

Event: OnModifiedFlagChanged (*i_bIsModified* as Boolean)

Description
This event is triggered when a document's modification status changes.

Activate

Method: Activate ()

Description
Makes this document the active document.

Errors
1200 The application object is no longer valid.

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.

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.

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.

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.

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.

GenerateCHashCode

Method: `GenerateCHashCode` ()

Description
Generate 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.

 © 2014 Altova GmbH

1205 Error during code generation.

See also
Code Generation

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.

See also
Code Generation

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.

See also
Code Generation

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.

See also
`Code Generation`

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.

See also
`Code Generation`

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.

See also
`Code Generation`

 © 2014 Altova GmbH

GenerateXQuery

Method: `GenerateXQuery` ()

Description
Generates mapping code as XQuery. 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.

See also
`Code Generation`

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.

See also
`Code Generation`

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.

See also
`Code Generation`

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.

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.

See also

`Code Generation`

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.

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.

Mappings

Property: `Mappings` as `Mappings` (read-only)

Description

 © 2014 Altova GmbH

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.

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.

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.

See also

`Code Generation`

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.

See also

`Code Generation`

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.

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.

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.

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.

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.

© 2014 Altova GmbH

22.2.10 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`

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.

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.

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.

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.

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.

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.

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.

© 2014 Altova GmbH

22.2.11 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`

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.

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.

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.

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.

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.
1901 Invalid address for the return parameter was specified.

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.

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.

© 2014 Altova GmbH

22.2.12 ErrorMarkers

Represents a collection of ErrorMarker objects.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Iterating through the collection:
`Count`
`Item`

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.

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.

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.

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.

© 2014 Altova GmbH

22.2.13 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`

View activation and view properties:
`Active`
`ShowItemTypes`
`ShowLibraryInFunctionHeader`
`HighlightMyConnections`
`HighlightMyConnectionsRecursivly`

Mapping related properties:
`ActiveMapping`
`ActiveMappingName`

Adding items:
`InsertWSDLCall`
`InsertXMLFile`
`InsertXMLSchema`
`InsertXMLSchemaWithSample`

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.

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.

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.

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.

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.

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.

InsertWSDLCall

Method: `InsertWSDLCall` (*`i_strWSDLFileName`* as `String`)

Description
Adds a new WSDL call component to the mapping.

 © 2014 Altova GmbH

Errors

1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

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.

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.

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.

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.

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.

ShowLibraryInFunctionHeader

Property: `ShowLibraryInFunctionHeader` as `Boolean`

Description
This property defines whether the name of the function library should be part of function names.

 © 2014 Altova GmbH

Errors

1300 The application object is no longer valid.
1301 Invalid address for the return parameter was specified.

22.2.14 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`

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.

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.

CreateConnection

Method: CreateConnection(DatapointFrom as Datapoint, DatapointTo as Datapoint) as Connection

 © 2014 Altova GmbH

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.

InsertFunctionCall

Method: InsertFunctionCall(strFunctionName as String, strLibraryName as String) as Component

Description
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.

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.

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.

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.

InsertXMLSchemaOutputParameter

Method: InsertXMLSchemaOutputParameter(strParamName as String, strSchemaFileName as String, strXMLRootElementName as String) as Component

 © 2014 Altova GmbH

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.

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.

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.

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.

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.

© 2014 Altova GmbH

22.2.15 Mappings

Represents a collection of Mapping objects.

Properties and Methods
Properties to navigate the object model:
`Application`
`Parent`

Iterating through the collection:
`Count`
`Item`

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.

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.

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.
1201	Invalid address for the return parameter was specified.

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.

© 2014 Altova GmbH

22.2.16 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`

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.

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.

See also
`Code Generation`

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

See also
`Code Generation`

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.

See also
`Code Generation`

CppSettings_GenerateVSProjectFile

Property: `CppSettings_GenerateVSProjectFile` as `ENUMProjectType`

Description
Specifies which version of VisualStudio (2005 / 2008 / 2010) 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

See also
`Code Generation`

© 2014 Altova GmbH

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.

See also
`Code Generation`

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.

See also
`Code Generation`

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

See also
`Code Generation`

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.

See also

`Code Generation`

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.

See also

`Code Generation`

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.

See also

`Code Generation`

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.

See also

`Code Generation`

© 2014 Altova GmbH

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.

See also
`Code Generation`

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.

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.

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.

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.

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.

See also
`Code Generation`

© 2014 Altova GmbH

22.2.17 Project (Enterprise or Professional Edition)

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

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. Note: in order to use these properties and methods, you will need to have the Enterprise or Professional edition of MapForce installed on your computer.

For operations with Web services, the Enterprise edition is required.

Events

This object supports the following events:

OnProjectClosed

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`.

_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;
```

 © 2014 Altova GmbH

```
    if ( objProject != null )
    {
        IterateProjectItemsRec( new Enumerator(objProject) );
    }
}
```

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.
1705	Mapping could not be assigned to project. Maybe it is already contained in the target folder.

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.

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.

Close

Method: `Close` ()

Description
Closes the project without saving.

Errors
1500 The object is no longer valid.

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`.

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.

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.

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

 © 2014 Altova GmbH

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

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

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

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.

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
}
```

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

1500 The object is no longer valid.
1501 Invalid package name specified.
Invalid address for the return parameter was specified.

Name

Property: `Name` as `String` (read-only)

Description
Name of the project file without file path.

Errors

1500 The object is no longer valid.

 © 2014 Altova GmbH

1501 Invalid address for the return parameter was specified.

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.

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.

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.

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.

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.

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.

Saved

Property: `Saved` as Boolean (read-only)

Description
`True` if the project was not modified since the last Save operation, `False` otherwise.

Errors
1500 The object is no longer valid.
1501 Invalid address for the return parameter was specified.

© 2014 Altova GmbH

22.2.18 ProjectItem (Enterprise or Professional Edition)

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. Note that, in order to use these properties and methods, you will need to have the Enterprise or Professional edition of MapForce installed on your computer. For operations with Web services, the Enterprise edition is required.

_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`.

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.

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
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.

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.

 © 2014 Altova GmbH

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.

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.

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.

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.

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.

CreateMappingForProject

Method: CreateMappingForProject (*i_strFileName* as String) as ProjectItem

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.

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.

GenerateCodeEx

Method: GenerateCode () as ErrorMarkers

Description

 © 2014 Altova GmbH

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.

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.

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.

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.

Examples
See `Project.Item` or `Project._NewEnum`.

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.

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.

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.

 © 2014 Altova GmbH

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.

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.

Remove

Method: `Remove` ()

Description
Remove this project item and all its children from the project tree.

Errors
1700 The object is no longer valid.

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.
1701 Invalid address for the return parameter was specified.
1702 The project item is not a part of a Web service.

22.3 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.

© 2014 Altova GmbH

22.3.1 ENUMApacheAxisVersion (obsolete)

Description
Enumeration values to select the Apache Axis version.

Possible values:

eApacheAxisVersion_Axis	= 1
eApacheAxisVersion_Axis2	= 2

See also
`Code Generation`

22.3.2 ENUMApplicationStatus

Description

Enumeration values to indicate the status of the application.

Possible values:

eApplicationRunning	= 0
eApplicationAfterLicenseCheck	= 1
eApplicationBeforeLicenseCheck	= 2
eApplicationConcurrentLicenseCheckFailed	= 3

© 2014 Altova GmbH

22.3.3 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
eSeverity_Autoinsertion	= 15
eSeverity_GlobalResources_DefaultWarning	= 16

22.3.4 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

© 2014 Altova GmbH

22.3.5 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

22.3.6 ENUMComponentDatapointSide

Description
Enumeration values to indicate the side of a datapoint on its component.

Possible values:

eDatapointSideInput	= 0
eDatapointSideOutput	= 1

See also
GetRootDatapoint

© 2014 Altova GmbH

22.3.7 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

22.3.8 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

© 2014 Altova GmbH

22.3.9 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

22.3.10 ENUMConnectionType

Description
Enumeration values to indicate the type of a connection.

Possible values:

eConnectionTypeTargetDriven	= 0
eConnectionTypeSourceDriven	= 1
eConnectionTypeCopyAll	= 2

See also
ConnectionType

© 2014 Altova GmbH

22.3.11 ENUMDOMType

Description
Enumeration values to specify the DOM type used by generated C++ mapping code.

Possible values:

eDOMType_xerces	= 1
eDOMType_xerces3	= 2
eDOMType_msxml6	= 3

Obsolete values

eDOMType_msxml4	= 0

Obsolete in this context means that this value is not supported and should not be used.

`eDOMType_xerces` indicates Xerces 2.x usage; `eDOMType_xerces3` indicates Xerces 3.x usage.

See also
`Code Generation`

22.3.12 ENUMLibType

Description

Enumeration values to specify the library type used by the generated C++ mapping code.

Possible values:

eLibType_static	= 0
eLibType_dll	= 1

See also

`Code Generation`

© 2014 Altova GmbH

22.3.13 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

See also

`Code Generation`

22.3.14 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`

© 2014 Altova GmbH

22.3.15 ENUMProjectType

Description
Enumeration values to select a project type for generated C# mapping code.

Possible values:

	eVisualStudio2005Project	= 4	Also for C++ code
	eVisualStudio2008Project	= 5	Also for C++ code
	eVisualStudio2010Project	= 6	Also for C++ code
Obsolete values	eVisualStudioProject	= 0	
	eVisualStudio2003Project	= 1	
	eBorlandProject	= 2	

Obsolete in this context means that this value is not supported and should not be used.

See also
`Code Generation`

22.3.16 ENUMSearchDatapointFlags

Description
Enumeration values used as bit-flags; to be used as combination of flags when searching for a datapoint.

Possible values:

eSearchDatapointElement	= 1
eSearchDatapointAttribute	= 2

See also
GetChild

© 2014 Altova GmbH

22.3.17 ENUMViewMode

Description
Enumeration values to select a MapForce view.

Possible values:

eMapForceView	= 0
eXSLView	= 1
eOutputView	= 2

Altova MapForce 2014

ActiveX Integration

23 ActiveX Integration

MapForceControl is a control that provides a means of integration of the MapForce user interface and the functionality described in this section into most kinds of applications. ActiveX technology was chosen so as to allow integration using any of a wide variety of languages; this enables C++, C#, VisualBasic, or HTML to be used for integration (ActiveX components integrated in HTML officially only work with Microsoft Internet Explorer). The attached Java wrapper library allows integration into Java. All components are full OLE Controls, which makes integration as simple as possible. Two different levels of integration are provided, thus enabling the integration to be adapted to a wide range of needs.

To integrate MapForce you must install the MapForce Integration Package. Ensure that you install MapForce first, and then the MapForce Integration Package.

For a successful integration you have to consider the following main design factors:

- What technology or programming language can the hosting application use to integrate the MapForceControl?
- Should the integrated UI look exactly like MapForce with all its menus, toolbars, and windows, or will a subset of these elements—like allowing only one document and a restricted set of commands—be more effective?
- How deep will the integration be? Should the MapForce user interface be used as is? Are user interface extensions and/or restrictions required? Can some frequently used tasks be automated?

The sections, Integration at the Application Level and Integration at Document Level describe the key steps at these respective levels. The Programming Languages section provides examples in C#, HTML, Visual Basic, and Java. Looking through these examples will help you to make the right decisions quickly. The section, Object Reference, describes all COM objects that can be used for integration, together with their properties and methods.

For automation tasks, the MapForce Automation Interface is accessible from the MapForceControl as well.

For information about how to integrate MapForce into Microsoft Visual Studio see the section, MapForce in Visual Studio.

MapForce integration and deployment on client computers:
If you create a .NET application and intend to distribute it to other clients, you will have to install the following on the client computer(s):

- The MapForce application
- The MapForce integration package.
- The custom integration code.

 © 2014 Altova GmbH

23.1 Integration at Application Level

Integration at application level is simple and straightforward. It allows you to embed the complete interface of MapForce into a window of your application. Since you get the whole user interface of MapForce, you get 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`. Its property `IntegrationLevel` defaults to application-level. You may use `Appearance` and `BorderStyle` to configure the appearance of the control's wrapper window. 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.

In the Programming Languages | HTML section is an example (Integration at Application Level) that shows how the MapForce application can be embedded in an HTML page. For usage with other programming languages, or more sophisticated access, see the Examples of integration at document-level (in C#, HTML, and Java).

23.2 Integration at Document Level

Integration at document level gives you freedom over instantiation and placement of the following parts of the MapForce user interface:

- Editing windows for MapForce mappings
- MapForce overview window
- MapForce library window
- MapForce validation window
- MapForce project window

If necessary, a replacement for the menus and toolbars of MapForce must be provided by your application.

You will need to instantiate and access multiple ActiveX controls, depending on which user interface parts you want to re-use. All these controls are contained in the MapForceControl OCX.

- Use MapForceControl to set the integration level and access application wide functionality.
- Use MapForceControlDocument to create any number of editor windows. It may be sufficient to create only one window and re-use it, depending on your needs.
- Optionally Use MapForceControlPlaceholder to embed MapForce overview, library, validation and project windows.
- Access run-time information about commands, menus, and toolbars available in MapForceControl to seamlessly integrate these commands into your application's menus and toolbars. See Query MapForce Commands for more information.

If you want to automate some behaviour of MapForce use the properties, methods, and events described for the MapForceControl, MapForceControlDocument and MapForceControlPlaceHolder. Consider using MapForceControl.Application, MapForceControlDocument.Document and MapForceControlPlaceHolder.Project for more complex access to MapForce functionality. However, 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.

See Examples on how to instantiate and access the necessary controls in different programming environments.

MapForce integration and deployment on client computers:
If you create a .NET application and intend to distribute it to other clients, you will have to install the following on the client computer(s):

- The MapForce application
- The MapForce integration package.
- The custom integration code.

 © 2014 Altova GmbH

23.2.1 Use MapForceControl

To integrate at document level, instantiate a `MapForceControl` first. Set the property `IntegrationLevel` to `ICActiveXIntegrationOnDocumentLevel (= 1)`. Set the window size of the embedding window to `0x0` to hide any user interface behind the control. You may use `Appearance` and `BorderStyle` to configure the appearance of the control's wrapper window.

Avoid using the method `Open` since this might lead to unexpected results. Use the corresponding open methods of `MapForceControlDocument` and `MapForceControlPlaceHolder`, instead.

See Query MapForce Commands for a description of how to integrate MapForce commands into your application. Send commands to MapForce via the method `Exec`. Query if a command is currently enabled or disabled using the method `QueryStatus`.

23.2.2 Use MapForceControlDocument

An instance of the `MapForceControlDocument` ActiveX control allows you to embed one MapForce document editing window into your application. You can use any number of instances you need.

Use the method `Open` to load any other existing file.

The control does not support a read-only mode. The value of the property `ReadOnly` is ignored.

Use `Path` and `Save` or methods and properties accessible via the property `Document` to access document functionality.

© 2014 Altova GmbH

23.2.3 Use MapForceControlPlaceHolder

Instances of MapForceControlPlaceHolder ActiveX controls allow you to selectively embed the additional helper windows of MapForce into your application. The property `PlaceholderWindowID` selects the MapForce helper window to be embedded. Use only one MapForceControlPlaceHolder for each window identifier. See for valid window identifiers.

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.

23.2.4 Query MapForce Commands

When integrating at document-level, no menu or toolbar from MapForce is available to your application. Instead, you can query all the commands and the structure of the application menu at runtime. Professional applications will need to integrate this menu in a sophisticated manner into their own menu structure. Your installation of MapForce even provides you with command label images used within MapForce. See the folder `MapForce2014\Examples\ActiveX\Images` of your MapForce installation for icons in GIF format. The file names correspond to the `labels` of commands.

© 2014 Altova GmbH

23.3 Programming Languages

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.

23.3.1 C#

The C# example shows how to integrate the MapForceControl in a common desktop application created with C# using Visual Studio 2008.The following topics are covered:

- Building a dynamic menu bar based on information the MapForceControl API provides.
- Usage of MapForce Placeholder controls in a standard frame window.
- Usage of a MapForce Placeholder control in a sizeable Tool Window.
- How to handle an event raised by the MapForceControl API.

Source code for all examples is available in the folder `<ApplicationFolder>\Examples\ActiveX\C#` of your MapForce installation. Please note that the example application is already complete. There is no need to change anything if you want to run it and see it working.

Introduction

Adding the MapForce components to the Toolbox
Before you take a look at the sample project please add the assemblies to the .NET IDE Toolbox. The MapForce Installer will have already installed the assemblies in the .NET Global Assembly Cache (GAC). If you open the Toolbox dialog under **Tools | Add/Remove Toolbox Items** the controls will appear as `AxMapForceControl`, `AxMapForceControlDocument` and `AxMapForceControlPlaceholder` on the .NET Framework Components tab. Check all to make them available to the IDE.

Now you can open the `MapForceApplication.sln` file in the `ActiveX\C#\MapForceApplication` folder to load the project.

Placing the MapForceControl

It is necessary to have one MapForceControl instance to set the integration level and to manage the Document and Placeholder controls of the MapForce library. The control is accessible via the General section of the Toolbox helper window in the IDE. To add it you need to select the component in the Toolbox window and drag a rectangle wherever you want to have it in the destination window. If you have an application which does not open a window on startup you can use a simple invisible Form with the control on it which is created manually in the code.

The example project adds this instance to the main MdiContainer MDIMain. If you open MDIMain in the Design View from the Solution Explorer you will see a light blue rectangle at the top-left side in the client area of the Frame window. Selecting this rectangle will show you the properties of the MapForceControl. It is important to set the `IntegrationLevel` property to **`ICActiveXIntegrationOnDocumentLevel`** in order to turn on the Document and Placeholder support of the MapForce library. Properties of the <%MAPCTRL%> component placed in the MDIFrame Window of the example application are shown below:

 © 2014 Altova GmbH

(DataBindings)	
(DynamicProperties)	
(Name)	**axMapForceControl**
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
CausesValidation	True
Dock	None
ImeMode	NoControl
IntegrationLevel	**ICActiveXIntegrationOnDocumentLevel**
Location	**280; 8**
Locked	False
Modifiers	Private
ReadOnly	**True**
Size	**224; 112**
TabIndex	**1**
TabStop	**False**
Tag	
Visible	**False**

Set the Visible flag to False to avoid any confusion about the control for the user.

Adding the Placeholder Control

Placeholders on the MDI Frame

The example project has to place Placeholder controls on the main MDI Frame. They are also added via the Toolbox window by dragging a rectangle on the destination Form. To set the type of the Placeholder which should be displayed one has to set the `PlaceholderWindowID` property. This property can also be changed during runtime in the code of the application. The Placeholder control would change its content immediately.

Properties of the Library window on the left side of the MDIMain Frame window are shown below:

(DataBindings)	
(DynamicProperties)	
(Name)	**axMapForceControlLibrary**
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
CausesValidation	True
Dock	**Left**
ImeMode	NoControl
Location	**0; 0**
Locked	False
Modifiers	Private
PlaceholderWindowID	**MapForceXLibraryWindow**
Size	**272; 625**
TabIndex	**2**
TabStop	True
Tag	
Visible	True

Properties of the Output window at the bottom:

(DataBindings)	
(DynamicProperties)	
(Name)	**axMapForceControlOutput**
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
CausesValidation	True
Dock	**Bottom**
ImeMode	NoControl
Location	**272; 473**
Locked	False
Modifiers	Private
PlaceholderWindowID	**MapForceXValidationWindow**
Size	**620; 152**
TabIndex	**4**
TabStop	True
Tag	
Visible	True

The Placeholders also have the Anchor and Dock properties set in order to react on resizing of the Frame window.

Placeholder on a separate Toolwindow

It is also possible to place a Placeholder control on a separate floating Toolwindow. To do this,

 © 2014 Altova GmbH

create a new Form as a Toolwindow and add the control as shown above. The MapForceOverviewWnd in the sample project contains the Overview window of MapForce.

Properties of the Overview Toolwindow:

(DataBindings)	
(DynamicProperties)	
(Name)	**axMapForceControlOverview**
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	**Top, Bottom, Left, Right**
CausesValidation	True
Dock	None
ImeMode	NoControl
Location	**0; 0**
Locked	False
Modifiers	**Public**
PlaceholderWindowID	**MapForceXOverviewWindow**
Size	**292; 266**
TabIndex	**0**
TabStop	True
Tag	
Visible	True

However, all Placeholder controls need a connection to the main MapForceControl. Normally this connection can be established automatically and there is nothing more to do. The two placeholders on the MDI Frame work like this. In the case of the Placeholder control in the Toolwindow, we need to add some code to the `public MDIMain()` method in `MDIMain.cs`:

```
m_MapForceOverview = new MapForceOverviewWnd();

MapForceControlLib.MapForceControlPlaceHolderClass type =
(MapForceControlLib.MapForceControlPlaceHolderClass)
m_MapForceOverview.axMapForceControlOverview.GetOcx();
type.AssignMultiDocCtrl((MapForceControlLib.MapForceControlClass)
axMapForceControl.GetOcx());

m_MapForceOverview.Show();
```

The MapForceOverviewWnd is created and shown here. In addition, a special method of the Placeholder control is called in order to connect the MapForcecontrol to it. `AssignMultiDocCtrl()` takes the MapForceControl as parameter and registers a reference to it in the Placeholder control.

Retrieving Command Information

The MapForceControl gives access to all commands of MapForce through its `CommandsStructure` property. The example project uses the `MapForce Commands` and `MapForce Command` interfaces to dynamically build a menu in the MDI Frame window.

The code to add the commands will be placed in the `MDIMain` method of the `MapForceApplication` class in the file `MDIMain.cs`:

```
public MDIMain()
{
    .
    .
    .
    MapForceControlLib.MapForceCommands objCommands;
    objCommands = axMapForceControl.CommandsStructure;

    long nCount = objCommands.Count;

    for(long idx = 0;idx < nCount;idx++)
    {
        MapForceControlLib.MapForceCommand     objCommand;
        objCommand = objCommands[(int)idx];

        // We are looking for the Menu with the name IDR_MAPFORCE. This menu
        contains
        // the complete main menu of MapForce.

        if(objCommand.Label == "IDR_MAPFORCE")
        {
            InsertMenuStructure(mainMenu.MenuItems, 1, objCommand, 0, 0,
            false);
        }
    }
    .
    .
    .
}
```

`mainMenu` is the name of the menu object of the MDI Frame window created in the Visual Studio IDE. `InsertMenuStructure` takes the MapForce menu from the IDR_`MAPFORCE` command object and adds the MapForce menu structure to the already existing menu of the sample project. No commands from the **File**, **Project**, or **Window** menu are added.

The new commands are instances of the class `CustomMenuItem`, which is defined in `CustomMenuItem.cs`. This class has an additional member to save the MapForce command ID, which is taken to execute the command using `Exec` on selecting the menu item. This code from `InsertMenuStructure` creates the new command:

```
CustomMenuItem   newMenuItem = new CustomMenuItem();

if(objCommand.IsSeparator)
    newMenuItem.Text = "-";
else
{
    newMenuItem.Text = strLabel;
    newMenuItem.m_MapForceCmdID = (int)objCommand.ID;
    newMenuItem.Click += new EventHandler(AltovaMenuItem_Click);
}
```

You can see that all commands get the same event handler `AltovaMenuItem_Click` which does

 © 2014 Altova GmbH

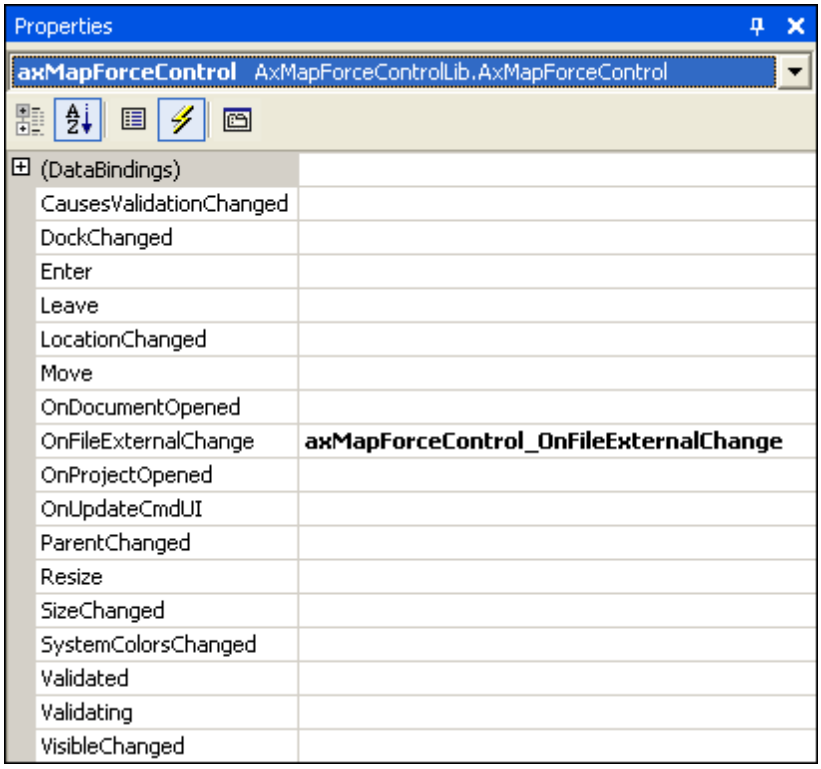
the processing of the command:

```
private void AltovaMenuItem_Click(object sender, EventArgs e)
{
    if(sender.GetType() ==
    System.Type.GetType("MapForceApplication.CustomMenuItem"))
    {
        CustomMenuItemcustomItem = (CustomMenuItem)sender;

        ProcessCommand(customItem.m_MapForceCmdID);
    }
}

private void ProcessCommand(int nID)
{
    MapForceDoc  docMapForce = GetCurrentMapForceDoc();

    if(docMapForce != null)
        docMapForce.axMapForceControlDoc.Exec(nID);
    else
        axMapForceControl.Exec(nID);
}
```

`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.

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 picture below shows the events of the main MapForceControl:



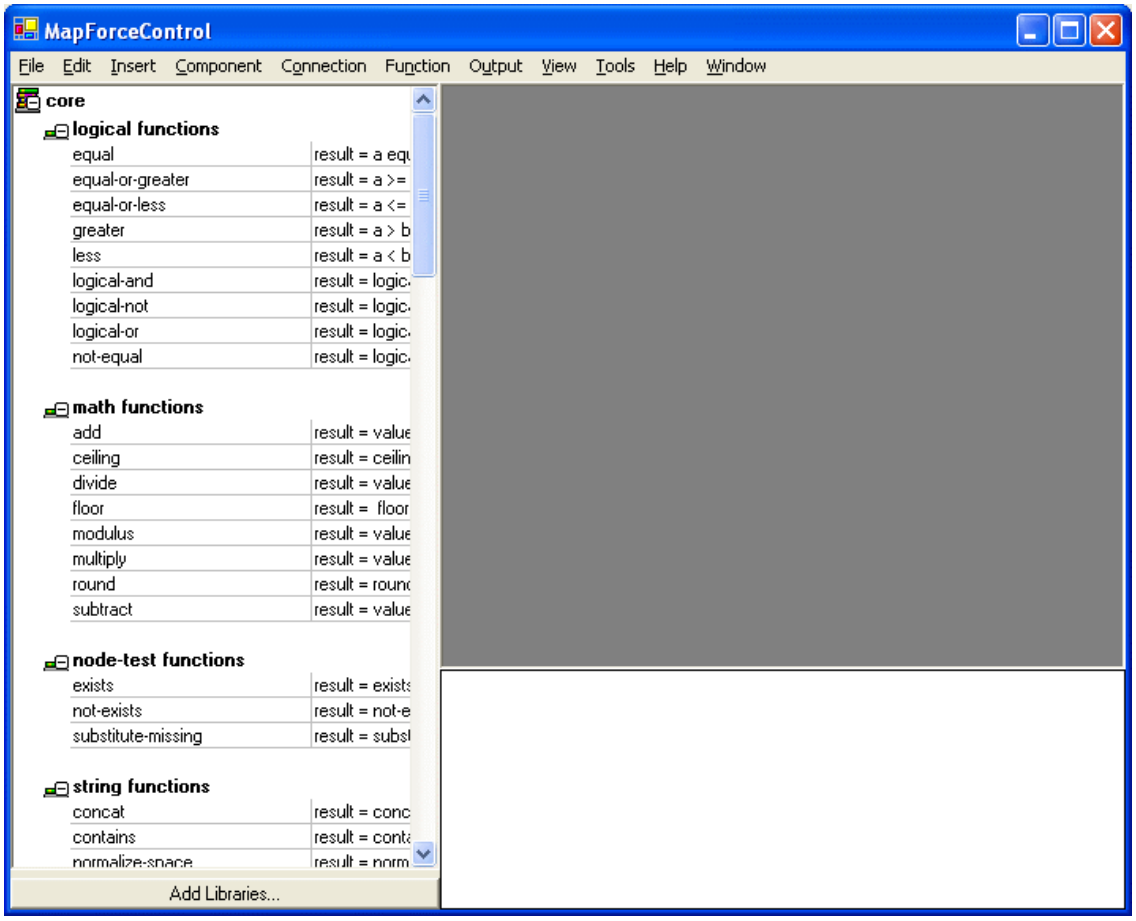
As you can see, the example project only overrides the `OnFileExternalChange` event. The creation of the C# delegate is done for you by the C# Framework. All you need to do is to fill the empty event handler. The handler implementation turns off any file reloading and displays a message box to inform 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;
}
```

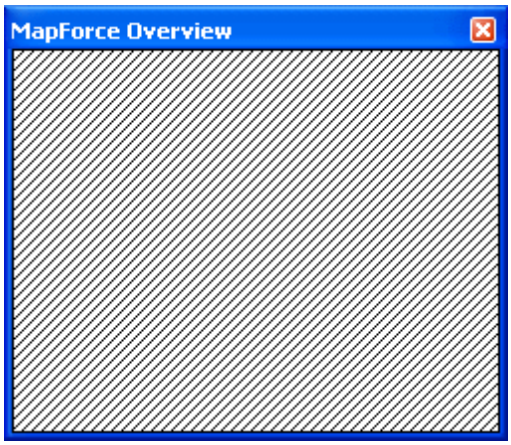
Testing the Example

After adding the assemblies to the Toolbox (see Introduction), you can run the sample project with F5 without the need to change anything in the code. The main MDI Frame window is created together with a floating Toolwindow containing the Overview window of MapForce. The application looks something like the screenshot below:

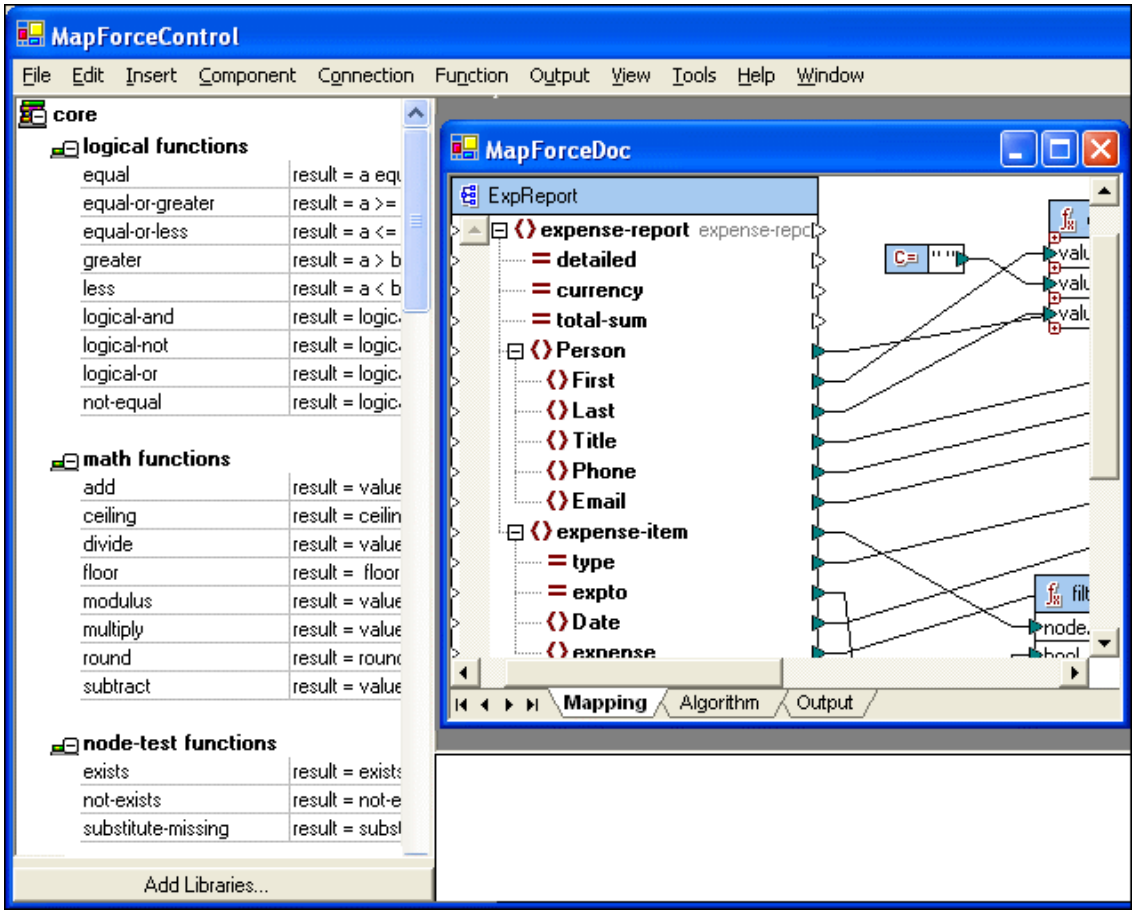
 © 2014 Altova GmbH



The floating Overview Toolwindow is also created:



Use **File | Open** to open the file `MarketingExpenses.mfd`, which is in the MapForce examples folder. The file is loaded and displayed in an own document child window:



After you load the document, you can try using menu commands. Note that context menus are also available. If you like, you can also load additional documents. Save any modifications using the **File | Save** command.

23.3.2 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.

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: `MapForce2014\Examples\ActiveX\HTML\MapForceActiveX_ApplicationLevel.htm`.

Note: This example works only in Internet Explorer.

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>
```

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. We use a method to locate the file relative to the MapForceControl so the example can run on different installations.

```
<SCRIPT ID=Javahandlers LANGUAGE=javascript>
// ---------------------------------
// open a pre-defined document
function BtnOpenMEFile()
{

       objMapForceControl.Open("C:\Documents and Settings\username\My Documents
\Altova\XMLSpy2014\Examples/MarketingExpenses.mfd");

}
</SCRIPT>
```

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.

```
<SCRIPT ID=Javahandlers LANGUAGE=javascript>
// ---------------------------------------------------------------------------
// generate code for active document into language-specific sub folders of
// the current default output directory. No user interaction necessary.
function BtnGenerate(languageID)
{
    // get top-level object of automation interface
    var objApp = objMapForceControl.Application;

    // get the active document
    var objDocument = objApp.ActiveDocument;

    // retrieve object to set the generation output path
    var objOptions = objApp.Options;

    if ( objDocument == null )
        alert( "no active document found" );
    else
    {
        if (languageID == 0)
        {
            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";
```

 © 2014 Altova GmbH

```
            objDocument .GenerateCHashCode();
        }
    }
}
</SCRIPT>
```

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 LANGUAGE="javascript">
    function objMapForceControl::OnDocumentOpened( objDocument )
    {
    // alert("Document '" + objDocument.Name + "' opened!");
    }
</SCRIPT>

<!-- ------------------------------------------------------------ -->
<!--  custom event 'OnDocumentClosed" of MapForceControl object  -->
<SCRIPT LANGUAGE="javascript">
    function objMapForceControl::OnDocumentClosed( objDocument )
    {
    // alert("Document '" + objDocument.Name + "' closed!");
    }
</SCRIPT>
```

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 customer 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 `C:\Program Files\Altova\MapForce2014\Examples\ActiveX\HTML\` folder of your MapForce installation.

Note: This example works only in Internet Explorer.

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="objMapForceXMapForceControl"
        Classid="clsid:A38637E9-5759-4456-A167-F01160CC22C1"
        width="0"
        height="0"
        VIEWASTEXT>
   <PARAM NAME="IntegrationLevel" VALUE="1">
</OBJECT>
```

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 mapping.

```
<OBJECT id="objDoc1"
        Classid="clsid:DFBB0871-DAFE-4502-BB66-08CEB7DF5255"
        width="600"
        height="500"
        VIEWASTEXT>
    <PARAM NAME="NewDocument">
</OBJECT>
```

Create Project Window

The HTML OBJECT tag is used to create a MapForceControlPlaceHolder window. The first additional custom parameter defines the placeholder to show the MapForce project window. The second parameter loads one of the example projects delivered with your MapForce installation (located in the <yourusername>/MyDocuments folder).

```
<OBJECT id="objProjectWindow"
        Classid="clsid:FDEC3B04-05F2-427d-988C-F03A85DE53C2"
        width="200"
        height="200"
        VIEWASTEXT>
    <PARAM name="PlaceholderWindowID" value="3">
      <PARAM name="FileName" value="MapForceExamples/MapForceExamples.mfp">
</OBJECT>
```

Create Placeholder for Helper Windows

The HTML OBJECT tag is used to instantiate a MapForceControlPlaceHolder ActiveX control that can host the different MapForce helper windows. Initially, no helper window is shown. See the example file.

```
<OBJECT id="objEHWindow"
        Classid="clsid:FDEC3B04-05F2-427d-988C-F03A85DE53C2"
        width="200"
        height="200"
        VIEWASTEXT>
   <PARAM name="PlaceholderWindowID" value="0">
</OBJECT>
```

 © 2014 Altova GmbH

Three buttons allow us to switch the actual window that will be shown. The JavaScript execute on-button-click sets the property `PlaceHolderWindowID` to the corresponding value defined in

```
<input type="button" value="Library Window" onclick="BtnHelperWindow(0)">
<input type="button" value="Overview Window" onclick="BtnHelperWindow(1)">
<input type="button" value="Validation Window" onclick="BtnHelperWindow(2)">

<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
//
-------------------------------------------------------------------------------
// specify which of the windows shall be shown in the placeholder control.
function BtnHelperWindow(i_ePlaceholderWindowID)
{
   objEHWindow.PlaceholderWindowID = i_ePlaceholderWindowID;
}
</SCRIPT>
```

Create a Custom Toolbar

The custom toolbar consists of buttons with images of MapForce commands. The command ID numbers can be found in the button elements shown below

```
<button id="btnInsertXML" title="Insert XML Schema/File"
onclick="BtnDoCommand(13635)">
      <img src="..\Images\ID_INSERT_XSD.gif" width="16" height="16" />
</button>
<button id="btnInsertDB" title="Insert Database" onclick="BtnDoCommand(13590)">
      <img src="..\Images\ID_INSERT_DATABASE.gif" width="16" height="16" />
</button>
<button id="btnInsertEDI" title="Insert EDI" onclick="BtnDoCommand(13591)">
      <img src="..\Images\ID_INSERT_EDI.gif" width="16" height="16" />
</button>
```

On clicking one of these buttons the corresponding command ID is sent to the manager control.

```
<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// ---------------------------------------------------------------
// 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.";
}
</SCRIPT>
```

Create More Buttons

In the example, we add some more buttons to show some automation code.

```
<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)">
</p>
<p>
    <input type="button" id="GenerateXSLT"   value="Generate XSLT"
onclick="BtnGenerate( objDoc1, 0 )">
    <input type="button" id="GenerateJava"   value="Generate Java"
onclick="BtnGenerate( objDoc1, 1 )">
    <input type="button" id="GenerateCpp"    value="Generate C++"
onclick="BtnGenerate( objDoc1, 2 )">
    <input type="button" id="GenerateCSharp" value="Generate C#"
onclick="BtnGenerate( objDoc1, 3 )">
</p>
```

The corresponding JavaScript looks like this:

```
<SCRIPT ID="Javahandlers" LANGUAGE="javascript">
// ------------------------------------------------------------
// 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:

       objDocCtrl.OpenDocument("C:\Documents and Settings\username\My Documents\
              Altova\XMLSpy2014\Examples/MarketingExpenses.mfd");
       objDocCtrl.setActive();

}

// ------------------------------------------------------------------------
// 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)
              {
                     objDocCtrl.Path = strPath.value;
                     objDocCtrl.SaveDocument();
              }
              else
```

 © 2014 Altova GmbH

```
                {
                        alert("Please set path for the document first!");
                        strPath.focus();
                }
        }

        objDocCtrl.setActive();
}
</SCRIPT>
```

Create Event Handler to Update Button Status

Availability of a command may vary with every mouseclick 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.

```
<SCRIPT LANGUAGE="javascript">

        function objMapForceX::OnUpdateCmdUI()
        {
        if ( document.readyState == "complete" )        // 'complete'
        {
                // update status of buttons
                GenerateXSLT.disabled =  ! (objDoc1.QueryStatus(13617) & 0x02);
        // not enabled
                GenerateJava.disabled =  ! (objDoc1.QueryStatus(13587) & 0x02);
        // not enabled
                GenerateCpp.disabled =   ! (objDoc1.QueryStatus(13589) & 0x02);
        // not enabled
                GenerateCSharp.disabled = ! (objDoc1.QueryStatus(13588) & 0x02);
        // not enabled

                btnFuncUserDef.disabled = ! (objDoc1.QueryStatus(13633) & 0x02);
                btnFuncUserDefSel.disabled = ! (objDoc1.QueryStatus(13634) & 0x02);
                btnFuncSettings.disabled = ! (objDoc1.QueryStatus(13632) & 0x02);
                btnInsertInput.disabled = ! (objDoc1.QueryStatus(13491) & 0x02);

                btnGenXSLT.disabled = ! (objDoc1.QueryStatus(13617) & 0x02);
                btnGenXSLT2.disabled = ! (objDoc1.QueryStatus(13618) & 0x02);
                btnGenXQuery.disabled = ! (objDoc1.QueryStatus(13586) & 0x02);
                btnGenCPP.disabled = ! (objDoc1.QueryStatus(13589) & 0x02);
                btnGenCSharp.disabled = ! (objDoc1.QueryStatus(13588) & 0x02);
                btnGenJava.disabled = ! (objDoc1.QueryStatus(13587) & 0x02);
        }
        }

        // set activity status of simulated toolbar
</SCRIPT>
```

23.3.3 Java

MapForce ActiveX components can be accessed from Java code. To allow this, the libraries listed below must reside in the classpath. These libraries are partly delivered with the MapForce Integration Package and are placed in the folder: `JavaAPI` in the MapForce application folder.

- `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 on the Java Classpath.

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 of the example Java project, see the section, Example Java Project.

Rules for mapping the ActiveX Control names to Java
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

 © 2014 Altova GmbH

`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

The `MapForceActiveX.jar` library

The classes and interfaces contained in the jar file are part of the `com.altova.automation.MapForce` package. They are described in the `MapForceActiveX_JavaDoc.zip` file from the folder `Examples\ActiveX\Java` in the MapForce application folder.

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

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:

`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
`MapForceActiveX.jar`	Java classes of the MapForce ActiveX control
`MapForceContainer.java`	Java example source code
`MapForceContainerEventHandler.java`	Java example source code
`XMLTreeDialog.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
`MapForceActiveX_JavaDoc.zip`	`Javadoc` file containing help documentation for the Java API

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 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 mappig table and prepare it for modal activation.

Running the example from the command line

Open a command prompt window and type:

```
buildAndRun.bat "C:\Program Files (x86)\Java\jdk1.6.0_04"
```

You may need to adapt the path to point to the binary folder of a 32-bit JDK 1.5 or later installation

 © 2014 Altova GmbH

on your computer. If you want to use the 64-bit version of the ActiveX control you need to a 64-bit JDK 1.6 or later. (Note that the AWT of JDK 1.5 is not fully compatible with the color schemes in Windows Vista / Windows 7.)

Press **Return**. The Java source in `MapForceContainer.java` will be compiled and then executed.

Loading the example in Eclipse

Open Eclipse and use the **Import | Existing Projects into Workspace** command to add the Eclipse project file located in the same folder as this Readme file to your workspace. Since you may not have write-access in this folder it is recommended to tell Eclipse to copy the project files into its workspace. The project `MapForceContainer` will then appear in your Package Explorer or Navigator.

If you want to use the 64-bit version of the MapForce ActiveX control you need to use a 64-bit version of Eclipse.

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 `MapForceContainer.java` is listed below with comments.

```
001 // access general JAVA-COM bridge classes
002 import com.altova.automation.libs.*;
003
004 //access MapForce Java-COM bridge
005 import com.altova.automation.MapForce.*;
006 import
com.altova.automation.MapForce.Enums.MapForceControlPlaceholderWindow;
007 import com.altova.automation.MapForce.Enums.ICActiveXIntegrationLevel;
008
009 //access AWT and Swing components
010 import java.awt.*;
011 import java.awt.event.*;
012
013 import javax.swing.*;
014
015 /**
016  * A simple example of a container for MapForce document-level integration
using Java AWT/Swing.
017  * The application's GUI shows a single document editing window and 3
different tool windows:
018  * The project window, the validation tool window and the library window.
019  *
020  * The application's menu gets created by reading it out from the MapForce
control.
021  *
```

```
022  * Communication between the project window and the document editing window
gets established
023  * by the event handler for the onOpenedOrFocused event. See
MapForceControlEventsDefaultHandler
024  * for further events.
025  *
026  * Feel free to modify and extend this sample.
027  *
028  * @author Altova GmbH
029  */
030 public class MapForceContainer
031 {
032   /**
033    * MapForce manager control - always needed
034    */
035   public static MapForceControl       mapForceControl = null;
036   /**
037    * MapForceDocument editing control
038    */
039   public static MapForceControlDocument   mapForceDocument = null;
040   /**
041    * Tool windows - MapForce place-holder controls
042    */
043   private static MapForceControlPlaceHolder mapForceProjectToolWindow;
044   private static MapForceControlPlaceHolder mapForceValidationToolWindow;
045   private static MapForceControlPlaceHolder mapForceLibraryToolWindow;
046
047   /**
048    * The hosting frame
049    */
050   private static Frame            frame;
051
052
053   /**
054    * Helper function that initializes the Document control
055    */
056   public static void initMapForceDocument()
057   {
058     try
059     {
060       if ( mapForceDocument != null )
061         frame.remove( mapForceDocument );
062       mapForceDocument = new MapForceControlDocument();
063       frame.add( mapForceDocument, BorderLayout.CENTER );
064       frame.validate();
065       // move the focus to the document control - used when querying for the
command status while enabling/disabling menu items
066       mapForceDocument.requestFocusInWindow();
067     }
068     catch ( Exception ex )
069     {
070       ex.printStackTrace();
071     }
072   }
073
074   /**
075    * Creation of the tree dialog
076    */
077   private static void loadTable(Mapping rootElement)
```

```
078   {
079     MapForceTable dlg = new MapForceTable( rootElement, frame );
080     dlg.pack();
081     dlg.setBounds( 0, 0, 800, 300 );
082     dlg.setLocationRelativeTo(frame);
083     dlg.setVisible( true );
084   }
085
086   /**
087    * The main entry point.
088    * Creates the application's frame and the MapForce windows.
089    * MapForceControl, although not visible, is important for the
coordination between the different ActiveX controls of MapForce.
090    */
091   public static void main( String[] args )
092   {
093     // in case of severe errors somewhere in the ActiveX controls an
AutomationException gets thrown
094     try
095     {
096       // Create the main frame of the application
097       frame = new Frame( "Java ActiveX host window" );
098       frame.setLayout( new BorderLayout() );
099
100       // Create the set of buttons and arrange them in a panel
101       Dimension btnDim = new Dimension ( 145, 25 );
102       JPanel westPanel = new JPanel();
103       westPanel.setPreferredSize( new Dimension( 155, 400 ) );
104       westPanel.setMaximumSize( new Dimension( 155, 800 ) );
105       westPanel.add( new Label( "Open documents") ) ;
106       JButton btnMarkExp = new JButton("MarketingExpenses");
westPanel.add( btnMarkExp );  btnMarkExp.setPreferredSize( btnDim );
107       JButton btnComplPO = new JButton("CompletePO");
westPanel.add( btnComplPO );  btnComplPO.setPreferredSize( btnDim );
108       westPanel.add( new Label( "Create/destroy") );
109       JButton btnProject = new JButton("Project window");
westPanel.add( btnProject );  btnProject.setPreferredSize( btnDim );
110       JButton btnValid = new JButton("Validation window");
westPanel.add( btnValid );    btnValid.setPreferredSize( btnDim );
111       JButton btnLibrary = new JButton("Library window");
westPanel.add( btnLibrary );  btnLibrary.setPreferredSize( btnDim );
112       westPanel.add( new Label( "Create menu") );
113       JButton btnMenu = new JButton("Load file menu");
westPanel.add( btnMenu );   btnMenu.setPreferredSize( btnDim );
114       westPanel.add( new Label( "Component structure") );
115       JButton btnTree = new JButton("Open table");
westPanel.add( btnTree );   btnTree.setPreferredSize( btnDim );
116
117       // Create the MapForce ActiveX control; the parameter determines that
we want to place document controls and place-holder
118       // controls individually. It gives us full control over the menu, as
well.
119       mapForceControl = new MapForceControl
( ICActiveXIntegrationLevel.ICActiveXIntegrationOnDocumentLevel.getValue(),
false );
120
121
122       // Panel that will hold the Project/Xpath/Attributes windows
123       final JPanel southPanel = new JPanel();
124
```

```
125       frame.add( southPanel,    BorderLayout.SOUTH );
126       frame.add( mapForceControl, BorderLayout.NORTH );
127       frame.add( westPanel,   BorderLayout.WEST );
128       initMapForceDocument();
129
130       // Listen in this class to communication events ( e.g.
onOpenedOrFocused is handled )
131       final MapForceContainerEventHandler handlerObject = new
MapForceContainerEventHandler();
132       mapForceControl.addListener( handlerObject );
133
134       // Prepare a shutdown mechanism
135       frame.addWindowListener( new WindowAdapter()
136       {
137         public void windowClosing( WindowEvent ev )
138         {
139           frame.dispose();
140           System.exit(0); }
141       } );
142       frame.setVisible( true );
143
144       // Locate samples installed with the product.
145       final String strExamplesFolder = System.getenv( "USERPROFILE" ) + "\
\My Documents\\Altova\\MapForce2013\\MapForceExamples\\";
146
147       // Create a project window and open the sample project in it
148       mapForceProjectToolWindow = new
MapForceControlPlaceHolder( MapForceControlPlaceholderWindow.MapForceXProjectWin
dow.getValue(), strExamplesFolder + "MapForceExamples.mfp" ) ;
149       mapForceProjectToolWindow.setPreferredSize( new Dimension( 200,
200 ) );
150       // For the beginning hide the project window
151       mapForceProjectToolWindow.setVisible( false ) ;
152       frame.add( mapForceProjectToolWindow, BorderLayout.NORTH ) ;
153
154       // Open the Marketing file when button is pressed
155       btnMarkExp.addActionListener( new ActionListener() {
156         public void actionPerformed(ActionEvent e) {
157           try {
158             // Instruct the Document control to open the file - avoid
calling the open method of MapForceControl (see help)
159             mapForceDocument.open( strExamplesFolder +
"MarketingExpenses.mfd" );
160             mapForceDocument.requestFocusInWindow();
161           } catch (AutomationException e1) {
162             e1.printStackTrace();
163           }
164         }
165       } );
166
167       // Open the Complete file when button is pressed
168       btnComplPO.addActionListener( new ActionListener() {
169         public void actionPerformed(ActionEvent e) {
170           try {
171             // Instruct the Document control to open the file - avoid
calling the open method of MapForceControl (see help)
172             mapForceDocument.open( strExamplesFolder + "CompletePO.mfd" );
173             mapForceDocument.requestFocusInWindow();
174           } catch (AutomationException e1) {
175             e1.printStackTrace();
```

 © 2014 Altova GmbH

```
176           }
177         }
178       } );
179
180       // Show/hide the project window when button is pressed
181       btnProject.addActionListener( new ActionListener() {
182         public void actionPerformed(ActionEvent e) {
183           if ( !mapForceProjectToolWindow.isVisible() ) {
184             // remove the hidden window from the frame (which acted like a
temporary parent)
185             frame.remove( mapForceProjectToolWindow ) ;
186             southPanel.add( mapForceProjectToolWindow );
187             mapForceProjectToolWindow.setVisible( true ) ;
188           } else {
189             mapForceProjectToolWindow.setVisible( false ) ;
190             southPanel.remove( mapForceProjectToolWindow );
191             // Add the hidden window to the frame (temporary parent)
192             frame.add( mapForceProjectToolWindow, BorderLayout.NORTH ) ;
193           }
194           frame.validate();
195         }
196       } ) ;
197
198       // Create/destroy the Validation window when button is pressed
199       btnValid.addActionListener( new ActionListener() {
200         public void actionPerformed(ActionEvent e) {
201           if ( mapForceValidationToolWindow == null ) {
202             try {
203               // Create a new window and add it to the south panel
204               mapForceValidationToolWindow = new MapForceControlPlaceHolder(
MapForceControlPlaceholderWindow.MapForceXValidationWindow.getValue(), "" );
205               mapForceValidationToolWindow.setPreferredSize( new
Dimension( 200, 200 ) );
206               southPanel.add( mapForceValidationToolWindow );
207             } catch (AutomationException e1) {
208               e1.printStackTrace();
209             }
210           } else {
211             // Remove the window and the reference
212             southPanel.remove( mapForceValidationToolWindow );
213             mapForceValidationToolWindow = null;
214           }
215           frame.validate();
216         }
217       } ) ;
218
219       // Create/destroy the Library window when button is pressed
220       btnLibrary.addActionListener( new ActionListener() {
221         public void actionPerformed(ActionEvent e) {
222           if ( mapForceLibraryToolWindow == null ) {
223             try {
224               // Create a new window and add it to the south panel
225               mapForceLibraryToolWindow = new
MapForceControlPlaceHolder( MapForceControlPlaceholderWindow.MapForceXLibraryWin
dow.getValue(), "" );
226               mapForceLibraryToolWindow.setPreferredSize( new
Dimension( 200, 200 ) );
227               southPanel.add( mapForceLibraryToolWindow );
228             } catch (AutomationException e1) {
229               e1.printStackTrace();
```

```
230             }
231           } else {
232             southPanel.remove( mapForceLibraryToolWindow );
233             mapForceLibraryToolWindow = null;
234           }
235           frame.validate();
236         }
237       } ) ;
238
239       // Load the file menu when the button is pressed
240       btnMenu.addActionListener( new ActionListener() {
241         public void actionPerformed(ActionEvent e) {
242           try {
243             // Create the menubar that will be attached to the frame
244             MenuBar mb = new MenuBar();
245             // Load the main menu's first item - the File menu
246             MapForceCommand xmlSpyMenu =
mapForceControl.getMainMenu().getSubCommands().getItem( 0 );
247             // Create Java menu items from the Commands objects
248             Menu fileMenu = new Menu();
249             handlerObject.fillMenu( fileMenu, xmlSpyMenu.getSubCommands() );
250             fileMenu.setLabel( xmlSpyMenu.getLabel().replace( "&", "" ) );
251             mb.add( fileMenu );
252             frame.setMenuBar( mb );
253             frame.validate();
254           } catch (AutomationException e1) {
255             e1.printStackTrace();
256           }
257           // Disable the button when the action has been performed
258           ((AbstractButton) e.getSource()).setEnabled( false );
259         }
260       } ) ;
261
262       // Load a table when the button is pushed
263       btnTree.addActionListener( new ActionListener() {
264         public void actionPerformed(ActionEvent e) {
265           try {
266             loadTable( mapForceDocument.getDocument().getMainMapping() );
267           } catch (AutomationException e1) {
268             e1.printStackTrace();
269           }
270         }
271       } ) ;
272
273       // Show the main window
274       frame.setBounds( 0, 0, 800, 600 );
275       frame.validate();
276
277       // feel free to extend.
278     }
279     catch ( AutomationException e )
280     {
281       e.printStackTrace();
282     }
283   }
284
285 }
```

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 ) );
```

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" ) +
        "\\My Documents\\Altova\\MapForce2014\\MapForceExamples\\";
3       mapForceProjectToolWindow = new
MapForceControlPlaceHolder( MapForceControlPlaceholderWindow.MapForceXProjectWin
dow.getValue(), strExamplesFolder + "MapForceExamples.mfp" ) ;
```

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    }
```

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 in in the menuMap
42            MenuItem mi = new
MenuItem( mapForceCommand.getLabel().replace( "&", "" ) );
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   }
```

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;
```

 © 2014 Altova GmbH

```
23      }
24    }
```

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
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 }
```


© 2014 Altova GmbH

23.3.4 Visual Basic

Source code for an integration of MapForceControl into a VisualBasic program can be found in the folder `MapForce2014\Examples\ActiveX\VisualBasic6` relative to your MapForce installation.

23.4 Command Table for MapForce

Tables in this section list the names and identifiers of all commands that are available within MapForce. Every sub-section lists the commands from the corresponding top-level menu of MapForce. The left-most column shows the command's menu text to make it easier for you to identify the functionality behind the command. The last sub-section is a collection of those commands that are not accessible via the main menu.

Depending on the edition of MapForce you have installed, some of these commands might not be supported. See Query MapForce Commands on how to query the current resource structure and command availability.

Use the command identifiers with MapForceControl.QueryStatus or MapForceControlDocument.QueryStatus to check the current status of a command. Use MapForceControl.Exec or MapForceControlDocument.Exec to execute a command.

File Menu
Edit Menu
Insert Menu
Project Menu
Component Menu
Connection Menu
Function Menu
Output Menu
View Menu
Tools Menu
Window Menu
Help Menu

Commands Not in Main Menu

 © 2014 Altova GmbH

23.4.1 File Menu

Commands from the File menu:

Menu Text	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
Save Project	ID_FILE_SAVEPROJECT	32378
Close Project	ID_FILE_CLOSEPROJECT	32355
Print...	IDC_FILE_PRINT	57607
Print Preview	IDC_FILE_PRINT_PREVIEW	57609
Print Setup...	ID_FILE_PRINT_SETUP	57606
Validate Mapping	ID_MAPPING_VALIDATE	32347
Generate code in selected language	ID_FILE_GENERATE_SELECTED_CODE	32362
Generate code in/XSLT 1.0	ID_FILE_GENERATEXSLT	32360
Generate code in/XSLT 2.0	ID_FILE_GENERATEXSLT2	32361
Generate code in/XQuery	ID_FILE_GENERATEXQUERY	32359
Generate code in/Java	ID_FILE_GENERATEJAVACODE	32358
Generate code in/C# (Sharp)	ID_FILE_GENERATECSCODE	32357
Generate code in/C++	ID_FILE_GENERATECPPCODE	32356
Generate documentation...	ID_FILE_GENERATE_DOCUMENTATION	32468
Mapping Settings...	ID_MAPPING_SETTINGS	32396
Recent Files/Recent File	ID_FILE_MRU_FILE1	57616
Recent Projects/Recent Project	ID_FILE_MRU_PROJECT1	32364
Exit	ID_APP_EXIT	57665

23.4.2 Edit Menu

Commands from the Edit menu:

Menu Text	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

© 2014 Altova GmbH

23.4.3 Insert Menu

Commands from the Insert menu:

Menu Text	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
Constant	ID_INSERT_CONSTANT	32388
Filter: Nodes/Rows	ID_INSERT_FILTER	32391
SQL-WHERE Condition	ID_INSERT_SQLWHERE_CONDITION	32351
Value-Map	ID_INSERT_VALUEMAP	32354
IF-Else Condition	ID_INSRT_CONDITION	32394
Exception	ID_INSERT_EXCEPTION	32311

23.4.4 Project Menu

Commands from the Project menu:

Menu Text	Command Name	ID
Add Files to Project...	ID_PROJECT_ADDFILESTOPROJECT	32420
Add Active File to Project...	ID_PROJECT_ADDACTIVEFILETOPROJECT	32419
Create Folder	ID_POPUP_PROJECT_CREATE_FOLDER	32310
Open Mapping for Operation	ID_POPUP_OPENOPERATIONSMAPPING	13692
Create Mapping for Operation...	ID_POPUP_CREATEMAPPINGFOROPERATION	32399
Add Mapping File for Operation...	ID_POPUP_PROJECT_ADD_MAPPING	32309
Reload Project	ID_PROJECT_RELOAD	32374
Insert Web Service...	ID_POPUP_PROJECT_INSERT_WEBSERVICE	32306
Open File In XMLSpy	ID_POPUP_PROJECT_OPENINXMLSPY	32305
Generate Code for Entire Project	ID_POPUP_PROJECT_GENERATE_PROJECT	32304
Generate code in/XSLT 1.0	ID_PROJECT_GENERATEXSLT	32425
Generate code in/XSLT 2.0	ID_PROJECT_GENERATEXSLT2	32426
Generate code in/XQuery	ID_PROJECT_GENERATEXQUERY	32424
Generate code in/Java	ID_PROJECT_GENERATEJAVACODE	32423
Generate code in/C# (Sharp)	ID_PROJECT_GENERATECSCODE	32422
Generate code in/C++	ID_PROJECT_GENERATECPPCODE	32421
Project Properties...	ID_PROJECT_PROPERTIES	32404

 © 2014 Altova GmbH

23.4.5 Component Menu

Commands from the Component menu:

Menu Text	Command Name	ID
Align Tree Left	ID_COMPONENT_LEFTALIGNTREE	32338
Align Tree Right	ID_COMPONENT_RIGHTALIGNTREE	32340
Change Root Element	ID_COMPONENT_CHANGEROOTELEMENT	32334
Edit Schema Definition in XMLSpy	ID_COMPONENT_EDIT_SCHEMA	32337
Edit FlexText Configuration	ID_COMPONENT_EDIT_MFT	32301
Add/Remove Tables...	ID_COMPONENT_SELECTTABLES	32346
Refresh	IDC_COMMAND_REFRESH_COMPONENT	32373
Duplicate Input	ID_COMPONENT_CREATE_DUPLICATE_ICON	32335
Remove Duplicate	ID_COMPONENT_REMOVE_DUPLICATE_ICON	32339
Database Table Actions	ID_POPUP_DATABASETABLEACTIONS	32400
Database Key Settings	ID_POPUP_VALUEKEYSETTINGS	32417
Query Database	ID_QUERY_DATABASE	32341
Properties	ID_COMPONENT_PROPERTIES	32336

23.4.6 Connection Menu

Commands from the Connection menu:

Menu Text	Menu Text	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 Mapping (mixed-content)	ID_POPUP_ORDERBYSOURCECONNECTION	32403
Properties	ID_POPUP_CONNECTION_SETTINGS	32398

© 2014 Altova GmbH

23.4.7 Function Menu

Commands from the Function menu:

Menu Text	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

23.4.8 Output Menu

Commands from the Output menu:

Menu Text	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
Validate output file	ID_XML_VALIDATE	32458
Save Output File...	IDC_FILE_SAVEGENERATEDOUTPUT	32321
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
Save All Output Files	IDC_FILE_SAVEALLGENERATEDOUTPUT	32374
Regenerate Output	ID_REGENERATE_PREVIEW_OUTPUT	32480
Text View Settings	ID_TEXTVIEWSETTINGSDIALOG	32472

© 2014 Altova GmbH

23.4.9 View Menu

Commands from the View menu:

Menu Text	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
Show selected component connectors	ID_VIEW_AUTOHIGHLIGHTCOMPONENTCONNECTIONS	32443
Show connectors from source to target	ID_VIEW_RECURSIVEAUTOHIGHLIGHT	32447
Zoom...	ID_VIEW_ZOOM	32451
Status Bar	ID_VIEW_STATUS_BAR	32449
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
XBRL Display Options	ID_VIEW_XBRL_DISPLAY_OPTIONS	32473
Back	ID_CMD_BACK	32479
Forward	ID_CMD_FORWARD	32478

23.4.10 Tools Menu

Commands from the Tools menu:

Menu Text	Command Name	ID
Global Resources	IDC_GLOBALRESOURCES	37401
Active Configuration	IDC_GLOBALRESOURCES_SUBMENUENTRY1	37408
Customize...	ID_VIEW_CUSTOMIZE	32444
Options...	ID_TOOLS_OPTIONS	32441
Restore Toolbars and Windows	ID_APP_RESET_TOOLBARS_AND_WINDOWS	32956

© 2014 Altova GmbH

23.4.11 Window Menu

Commands from the Window menu:

Menu Text	Command Name	ID
Cascade	ID_WINDOW_CASCADE	57650
Tile Horizontal	ID_WINDOW_TILE_HORZ	57651
Tile Vertical	ID_WINDOW_TILE_VERT	57652

23.4.12 Help Menu

Commands from the Help menu:

Menu Text	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
Components Download...	IDC_OPEN_COMPONENTS_PAGE	32325
MapForce on the Internet..	IDC_OPEN_XML_SPY_HOME	32328
MapForce Training...	IDC_OPEN_MAPFORCE_TRAINING_PAGE	32300
About MapForce...	ID_APP_ABOUT	57664

© 2014 Altova GmbH

23.4.13 Commands Not in Main Menu

Commands not in the main menu:

Menu Text	Command Name	ID
	IDC_QUICK_HELP	32329
Edit FlexText Configuration	ID_COMPONENT_EDIT_MFT	32301
Priority Context	ID_COMPONENT_PRIORITYCONTEXT	32318
	ID_EDIT_FINDPREV	32350
	ID_FUNCTION_GOTO_MAIN	32382
Insert Input	ID_FUNCTION_INSERT_INPUT_AT_POINT	32384
	ID_FUNCTION_REMOVE	32385
Replace component with internal function structure	ID_FUNCTION_REPLACE_WITH_COMPONENTS	32386
	ID_MAPFORCEVIEW_ZOOM	32395
	ID_NEXT_PANE	32397
Add Active File to Project	ID_POPUP_PROJECT_ADDACTIVEFILETOPROJECT	32405
Add Files to Project...	ID_POPUP_PROJECT_ADDFILESTOPROJECT	32406
C++	ID_POPUP_PROJECT_GENERATECPPCODE	32408
C# (Sharp)	ID_POPUP_PROJECT_GENERATECSCODE	32409
Java	ID_POPUP_PROJECT_GENERATEJAVACODE	32410
XQuery	ID_POPUP_PROJECT_GENERATEXQUERY	32411
XSLT 1.0	ID_POPUP_PROJECT_GENERATEXSLT	32412
XSLT 2.0	ID_POPUP_PROJECT_GENERATEXSLT2	32413
Generate All	ID_POPUP_PROJECT_GENERATE_ALL	32303
Generate code in default language	ID_POPUP_PROJECT_GENERATE_CODE	32414
Open	ID_POPUP_PROJECT_OPEN_MAPPING	32307
Properties...	ID_POPUP_PROJECT_PROJECTPROPERTIES	32428
Remove	ID_POPUP_PROJECT_REMOVE	32308
Add/Remove Selections...	ID_POPUP_STRUCTURENODE_SELECTIONS	32491
	ID_PREV_PANE	32418
	ID_TOGGLE_FOLDINGMARGIN	32438
	ID_TOGGLE_INDENTGUIDES	32439
	ID_TOGGLE_NUMLINEMARGIN	32440
	ID_WORD_WRAP	32457

23.5 Accessing MapForceAPI

The focus of this documentation is the ActiveX controls and interfaces required to integrate the MapForce user interface into your application. To allow you to automate or control the functionality of the integrated components, the following properties give you access to the MapForce automation interface (MapForceAPI):

`MapForceControl.Application`
`MapForceControlDocument.Document`
`MapForceControlPlaceHolder.Project`

Some restrictions apply to the usage of the MapForce automation interface when integrating MapForceControl at document-level. See Integration at document level for details.

© 2014 Altova GmbH

23.6 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.

23.6.1 MapForceCommand

Properties:
ID
Label
IsSeparator
ToolTip
StatusText
Accelerator
SubCommands

Description:
Each Command object can be one of three possible types:

- **Command**: ID is set to a value greater 0 and Label is set to the command name. IsSeparator is false and the SubCommands collection is empty.
- **Separator**: IsSeparator is true. ID is 0 and Label is not set. The SubCommands collection is empty.
- (**Sub**) **Menu**: The SubCommands collection contains Command objects and Label is the name of the menu. ID is set to 0 and IsSeparator is false.

Accelerator

Property: Label as string

Description:
For command objects that are children of the ALL_COMMANDS collection, this is 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.

ID

Property: ID as long

Description:
ID is 0 for separators and menus.
For commands, this is the ID which can be used with Exec and QueryStatus.

IsSeparator

Property: IsSeparator as boolean

Description:
True if the command is a separator.

 © 2014 Altova GmbH

Label

Property: `Label` as string

Description:
Label is empty for separators.
For command objects that are children of the ALL_COMMANDS collection, this is a unique name. Command icons are stored in files with this name. See Query Commands for more information.

For command objects that are children of menus, the label property holds the command's menu text.
For sub-menus, this property holds the menu text.

StatusText

Property: `Label` as string

Description:
For command objects that are children of the ALL_COMMANDS collection, this is the text shown in the status bar when the command is selected.

SubCommands

Property: `SubCommands` as `Commands`

Description:
The `SubCommands` collection holds any sub-commands if this command is actually a menu or submenu.

ToolTip

Property: `ToolTip` as string

Description:
For command objects that are children of the ALL_COMMANDS collection, this is the text shown as tool-tip.

23.6.2 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.

Count

Property: Count as long

Description:
Number of Command objects on this level of the collection.

Item

Property: Item (n as long) as Command

Description:
Gets the command with the index n in this collection. Index is 1-based.

© 2014 Altova GmbH

23.6.3 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
```

Properties

The following properties are defined:

IntegrationLevel
EnableUserPrompts
Appearance
BorderStyle

Command related properties:
CommandsList
MainMenu
Toolbars
CommandsStructure (deprecated)

Access to MapForceAPI:
Application

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`.

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.

BorderStyle

Property: `BorderStyle` as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

CommandsList

Property: `CommandList` as `Commands` (read-only)

Dispatch Id: *1004*

Description:
This property returns a flat list of all commands defined available with MapForceControl.

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*.

IntegrationLevel

Property: `IntegrationLevel` as `ICActiveXIntegrationLevel`

Dispatch Id: *1000*

 © 2014 Altova GmbH

Description:
The `IntegrationLevel` property determines the operation mode of the control. See also Integration at the 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.

MainMenu

Property: `MainMenu as` Command (read-only)

Dispatch Id: *1003*

Description:
This property gives access to the description of the MapForceControl main menu.

Toolbars

Property: `Toolbars` as Commands (read-only)

Dispatch Id: *1005*

Description:
This property returns a list of all toolbar descriptions that describe all toolbars available with MapForceControl.

Methods

The following methods are defined:

`Open`
`Exec`
`QueryStatus`

Exec

Method: `Exec` (`nCmdID` as long) as boolean

Dispatch Id: *6*

Description:
`Exec` calls the `MapForce` command with the ID `nCmdID`. If the command can be executed, the method returns `true`. See also `CommandsStructure` to get a list of all available commands and `QueryStatus` to retrieve the status of any command.

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`.

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.

Events

The MapForceControl ActiveX control provides the following connection point events:

`OnUpdateCmdUI`
`OnOpenedOrFocused`
`OnCloseEditingWindow`
`OnFileChangedAlert`
`OnContextChanged`

`OnDocumentOpened`
`OnValidationWindowUpdated`

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.

 © 2014 Altova GmbH

OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description:
This event is not used in MapForce

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.

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 harddisk 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.

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).

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.

OnToolWindowUpdated

Event: `OnToolWindowUpdated(pToolWnd` as long `)`

Dispatch Id: *1006*

Description:
This event is triggered when the tool window is updated.

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.

OnValidationWindowUpdated

Event: `OnValidationWindowUpdated` ()

Dispatch Id: *3*

Description:
This event is triggered whenever the validation output window, is updated with new information.

 © 2014 Altova GmbH

23.6.4 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
```

Properties

The following properties are defined:

ReadOnly
IsModified
Path
Appearance
BorderStyle

Access to MapForceAPI:
Document

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`.

BorderStyle

Property: `BorderStyle` as short

Dispatch Id: *-504*

Description:
A value of `1` displays the control with a thin border. Default value is `0`.

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 functionalities which can be used with the document loaded in the control. The property is read-only.

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.

Path

Property: `Path` as string

Dispatch Id: *1005*

Description:
Sets or gets the full path name of the document loaded into the control.

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.

 © 2014 Altova GmbH

Methods

The following methods are defined:

Document handling:
New
Open
Reload
Save
SaveAs

Command Handling:
Exec
QueryStatus

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`. The client should call the `Exec` method of the document control if there is currently an active document available in the application.

See also `CommandsStructure` to get a list of all available commands and `QueryStatus` to retrieve the status of any command.

New

Method: `New` () as boolean

Dispatch Id: *1000*

Description:
This method initializes a new document inside the control..

Open

Method: `Open` (`strFileName` as string) as boolean

Dispatch Id: *1001*

Description:
`Open` loads the file `strFileName` as the new document into the control.

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.

Reload

Method: `Reload` () as boolean

Dispatch Id: *1002*

Description:
`Reload` updates the document content from the file system.

Save

Method: `Save` () as boolean

Dispatch Id: *1003*

Description:
`Save` saves the current document at the location `Path`.

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.

Events

The MapForceControlDocument ActiveX control provides following connection point events:

`OnDocumentOpened`
`OnDocumentClosed`
`OnModifiedFlagChanged`
`OnContextChanged`
`OnFileChangedAlert`

 © 2014 Altova GmbH

`OnActivate`
`OnSetEditorTitle`

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.

OnContextChanged

Event: `OnContextChanged` (`i_strContextName` as String, `i_bActive` as bool) as bool

Dispatch Id: *1004*

Description: None

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.

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.

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.

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 harddisk 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.

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.

OnSetEditorTitle

Event: OnSetEditorTitle ()

Dispatch Id: *1006*

Description:
This event is being raised when the contained document is being internally renamed.

 © 2014 Altova GmbH

23.6.5 MapForceControlPlaceHolder

Properties available for all kinds of placeholder windows:
PlaceholderWindowID

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
```

Properties

The following properties are defined:

PlaceholderWindowID

Access to MapForceAPI:
Project

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.

PlaceholderWindowID

Property: PlaceholderWindowID as

Dispatch Id: *1*

Description:
Using this property the object knows 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 enumeration. The control changes its state immediately and shows the new MapForce window.

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 functionalities 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.

Methods

The following method is defined:

`OpenProject`
`CloseProject`

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)`.

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)`.

Events

The MapForceControlPlaceholder ActiveX control provides following connection point events:

`OnModifiedFlagChanged`

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.

 © 2014 Altova GmbH

OnSetLabel

Event: `OnSetLabel(i_strNewLabel as string)`

Dispatch Id: *1000*

Description:
Raised when the title of the placeholder window is changed.

23.6.6 Enumerations

The following enumerations are defined:

ICActiveXIntegrationLevel
MapForceControlPlaceholderWindow

ICActiveXIntegrationLevel

Possible values for the IntegrationLevel property of the MapForceControl.

```
ICActiveXIntegrationOnApplicationLevel  = 0
ICActiveXIntegrationOnDocumentLevel     = 1
```

MapForceControlPlaceholderWindow

This enumeration contains the list of the supported additional MapForce windows.

```
MapForceXNoWindow           = -1
MapForceXLibraryWindow      =  0
MapForceXOverviewWindow     =  1
MapForceXValidationWindow   =  2
MapForceXProjectWindow      =  3
```


© 2014 Altova GmbH

Altova MapForce 2014

Anexos

24 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
- Validador XML de Altova
- Motores XSLT y XQuery de Altova
- Compatibilidad con Unicode
- Uso de Internet

Información de la licencia

- Distribución electrónica de software
- Activación del software y medición de licencias
- Derechos de propiedad intelectual
- Contrato de licencia para el usuario final de Altova

© 2014 Altova GmbH

24.1 Información sobre motores de Altova

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.

24.1.1 Motor XSLT 1.0: información de implementación

El motor XSLT 1.0 de Altova está integrado en las herramientas XMLSpy, StyleVision, Authentic y MapForce de Altova. El motor XSLT 1.0 de Altova implementa la recomendación del 16 de noviembre de 1999 sobre XSLT 1.0 y la recomendación del 16 de noviembre de 1999 sobre XPath 1.0 del W3C. A continuación describimos sus limitaciones y su comportamiento durante la implementación.

Limitaciones

- Los elementos **`xsl:preserve-space`** y **`xsl:strip-space`** no son compatibles.
- Cuando el atributo `method` de `xsl:output` tiene el valor `HTML` o si se selecciona el resultado HTML como valor predeterminado, los caracteres especiales del archivo XML o XSLT se insertan en el documento HTML como caracteres especiales directamente. Es decir, no se insertan como referencias de caracteres HTML en el documento de salida. Por ejemplo, el carácter ` ` (la referencia de carácter decimal para un espacio de no separación) no se inserta como ` ` en el código HTML, sino que se inserta un espacio de no separación directamente.

Cómo se gestionan los nodos del documento XML de origen que contienen espacios en blanco

En los datos XML (y, por tanto, el conjunto de datos XML) que se pasan al motor XSLT 1.0 de Altova se eliminan los nodos de texto denominados *boundary-whitespace-only*. Un nodo de texto *boundary-whitespace-only* es un nodo de texto que contiene espacios en blanco entre dos elementos dentro de un elemento de contenido mixto. La eliminación de estos nodos puede afectar al valor que devuelven las funciones `fn:position()`, `fn:last()` y `fn:count()`.

Por lo general, por cada selección de nodo que selecciona también nodos de texto, los nodos de texto *boundary-whitespace-only* se incluyen en la selección. Sin embargo, puesto que en el conjunto de datos XML utilizado por los motores de Altova se eliminan los nodos de texto *boundary-whitespace-only*, estos nodos no están presentes en el conjunto de datos XML. Como resultado, el tamaño de la selección y la numeración de los nodos de la selección será diferente que en la selección que incluya dichos nodos de texto. Es decir, las funciones `fn:position()`, `fn:last()` y `fn:count()` pueden producir resultados diferentes en el motor XSLT 1.0 de Altova.

Los nodos de texto *boundary-whitespace-only* se evalúan como elementos del mismo nivel de otros elementos, sobre todo cuando se usa `xsl:apply-templates` para aplicar elementos. Cuando las funciones `fn:position()`, `fn:last()` y `fn:count()` se usan en patrones con una prueba de nombre (como `para[3]`, que es la abreviatura de `para[position()=3]`), los nodos de texto *boundary-whitespace-only* son irrelevantes porque solamente se seleccionan los elementos con nombre (como `para`). (No obstante, recuerde que los nodos *boundary-whitespace-only* sí son relevantes en los patrones que usan el comodín: como en `*[10]`, por ejemplo.)

Nota: si un nodo de texto *boundary-whitespace-only* es necesario para el documento de salida, entonces inserte el espacio en blanco necesario dentro de uno de los dos elementos secundarios adyacentes. Por ejemplo, cuando este fragmento de código XML

```
<para>This is <b>bold</b> <i>italic</>.</para>
```

se procesa con esta plantilla XSLT

 © 2014 Altova GmbH

```
<xsl:template match="para">
   <xsl:apply-templates/>
</xsl:template>
```

se genera este resultado:

```
This is bolditalic.
```

Para introducir un espacio en blanco entre `bold` y `italic` en el documento de salida, inserte un carácter de espacio dentro del elemento `<b>` o `<i>` en el XML de origen. Por ejemplo:

```
<para>This is <b>bold</b> <i> italic</i>.</para> o
<para>This is <b>bold&#x20;</b> <i>italic</i>.</para> o
<para>This is <b>bold</b><i>&#x20;italic</i>.</para>
```

Al procesar estos elementos `para` con la misma plantilla XSLT de antes, el motor genera este resultado:

```
This is bold italic.
```

24.1.2 Motor XSLT 2.0: información de implementación

El motor XSLT 2.0 de Altova está integrado en las herramientas XMLSpy, StyleVision, Authentic y MapForce de Altova. En esta sección ofrecemos información general sobre el motor y describimos el comportamiento de las funciones XSLT 2.0 durante la implementación.

Si busca información sobre el comportamiento de las funciones XPath 2.0, consulte la sección Funciones XPath 2.0 y XQuery 1.0.

Información general

El motor XSLT 2.0 de Altova cumple con la recomendación sobre XSLT 2.0 del W3C del 23 de enero de 2007.

Compatibilidad con versiones anteriores de XSLT

El motor XSLT 2.0 de Altova es compatible con versiones anteriores. No obstante, tenga en cuenta que puede haber diferencias entre el resultado generado por el motor XSLT 1.0 y el motor XSLT 2.0.

En los demás productos de Altova, no hay problemas de compatibilidad con versiones anteriores. Esto se debe a que estos productos seleccionan automáticamente el motor adecuado para la transformación. Por ejemplo, imagine que en XMLSpy quiere procesar un documento XML con una hoja de estilos XSLT 1.0. Cuando invoque el comando de transformación, XMLSpy selecciona automáticamente el motor XSLT 1.0 de XMLSpy para llevar a cabo la transformación.

Nota: la versión de la hoja de estilos se especifica en el atributo `version` del elemento `stylesheet` o `transform` de la hoja de estilos.

Espacios de nombres

La hoja de estilos XSLT 2.0 debe declarar los espacios de nombres que aparecen a continuación si desea usar las funciones y los constructores de tipos que ofrece XSLT 2.0. Los prefijos que aparecen a continuación son los que se usan normalmente, pero puede usar otros si lo desea.

Nombre del 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`

Por lo general, estos espacios de nombres se declaran en el elemento `xsl:stylesheet` o `xsl:transform`, tal y como muestra este fragmento de código:

```
<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"
   ...
```

 © 2014 Altova GmbH

```
</xsl:stylesheet>
```

Es necesario tener en cuenta algunos aspectos.

- El motor XSLT 2.0 de Altova usa el espacio de nombres Funciones XPath 2.0 y Funciones XQuery 1.0 (*ver la tabla anterior*) como **espacio de nombres predeterminado para las funciones**. Es decir, puede usar las funciones XPath 2.0 y XQuery 1.0 sin prefijo en la hoja de estilos. Si declara el espacio de nombres Funciones XPath 2.0 en la hoja de estilos con un prefijo, entonces puede usar también el prefijo asignado en la declaración.
- Cuando use tipos y constructores de tipo del espacio de nombres XML Schema, el prefijo usado en la declaración de espacio de nombres se debe usar al llamar al constructor de tipo (por ejemplo, `xs:date`).
- A partir de las recomendaciones CR del 23 de enero de 2007, el tipo de datos `untypedAtomic` y los tipos de datos de duración (`dayTimeDuration` y `yearMonthDuration`), que antes estaban en el espacio de nombres Tipos de datos XPath (cuyo prefijo suele ser `xdt:`) ahora están en el espacio de nombres XML Schema.
- Algunas funciones XPath 2.0 tienen el mismo nombre que los tipos de datos XML Schema. Por ejemplo, para las funciones XPath `fn:string` y `fn:boolean` existen tipos de datos XML Schema con los mismos nombres locales: `xs:string` y `xs:boolean`. De modo que, si usa la expresión XPath `string('Hello')`, la expresión se evalúa como `fn:string('Hello')`y no como `xs:string('Hello')`.

Compatibilidad con esquemas XML
El motor XSLT 2.0 de Altova está preparado para esquemas XML.

Espacios en blanco en el documento XML
El motor XSLT 2.0 de Altova elimina por defecto todos los espacios en blanco de los nodos *boundary-whitespace-only* del documento XML de origen. Esto afecta a los valores que devuelven las funciones `fn:position()`, `fn:last()`, `fn:count()` y `fn:deep-equal()`. Para más información, consulte el apartado Nodos del documento XML que contienen espacios en blanco de la sección *Funciones XPath 2.0 y XQuery 1.0*.

Nota: si un nodo de texto *boundary-whitespace-only* es necesario en el documento de salida, inserte el espacio en blanco necesario dentro de uno de los dos elementos secundarios adyacentes. Por ejemplo, cuando se procesa este fragmento XML

```
<para>This is <b>bold</b> <i>italic</>.</para>
```

con esta plantilla XSLT

```
<xsl:template match="para">
   <xsl:apply-templates/>
</xsl:template>
```

el motor produce este resultado:

```
This is bolditalic.
```

Para que haya un espacio en blanco entre `bold` y `italic` en el documento de salida, inserte un carácter de espacio dentro del elemento `<b>` o `<i>` en el XML de origen. Por ejemplo:

```
<para>This is <b>bold</b> <i> italic</>.</para> or
```

```
<para>This is <b>bold&#x20;</b> <i>italic</>.</para> or
<para>This is <b>bold</b><i>&#x20;italic</>.</para>
```

Si procesamos este fragmento de código XML con la misma plantilla XSLT, obtenemos este resultado:

```
This is bold italic.
```

Elementos y funciones XSLT 2.0

Las limitaciones de los elementos y funciones XSLT 2.0 y su comportamiento durante la implementación se describen en el apartado Elementos y funciones XSLT 2.0.

Funciones XPath 2.0

El comportamiento de las funciones XPath 2.0 durante la implementación se describe en la sección Funciones XPath 2.0 y XQuery 1.0.

Elementos y funciones XSLT 2.0

Limitaciones

Los elementos **`xsl:preserve-space`** y **`xsl:strip-space`** no son compatibles con el motor XSLT 2.0 de Altova.

Comportamiento durante la implementación

A continuación describimos cómo gestiona el motor XSLT 2.0 de Altova algunos aspectos del comportamiento de ciertas funciones XSLT 2.0 relacionados con la implementación.

`xsl:result-document`

También son compatibles las codificaciones `x-base16tobinary` y `x-base64tobinary`.

`function-available`

Prueba la disponibilidad de funciones de ámbito (funciones XSLT 2.0, XPath 2.0 y de extensión).

`unparsed-text`

El atributo `href` acepta (i) rutas de acceso relativas para los archivos de la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles las codificaciones `x-binarytobase16` y `x-binarytobase64`.

`unparsed-text-available`

El atributo `href` acepta (i) rutas de acceso relativas para los archivos de la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles las codificaciones `x-binarytobase16` y `x-binarytobase64`.

Nota: los valores de codificación `base16tobinary`, `base64tobinary`, `binarytobase16` y `binarytobase64` ya están en desuso y se implementaron en versiones anteriores del motor AltovaXML. Altova ya no publica el motor de validación XML y transformación XSLT/XQuery AltovaXML.

 © 2014 Altova GmbH

24.1.3 Motor XQuery 1.0: información de implementación

El motor XQuery 1.0 de Altova está integrado en las herramientas XMLSpy y MapForce de Altova. Esta sección incluye información sobre este motor y describe su comportamiento durante la implementación.

Cumplimiento de estándares
El motor XQuery 1.0 de Altova cumple con la recomendación sobre XQuery 1.0 del W3C del 23 de enero de 2007. El estándar XQuery ofrece cierta libertad a la hora de implementar muchas de sus características. A continuación explicamos cómo se implementan estas características en el motor XQuery 1.0 de Altova.

Compatibilidad con esquemas XML
El motor XQuery 1.0 de Altova está preparado para esquemas XML.

Codificación
Son compatibles las codificaciones de caracteres UTF-8 y UTF-16.

Espacios de nombres
Los URI del espacio de nombres y sus enlaces asociados están predefinidos.

Nombres del espacio de nombres	Prefijo	URI del espacio de nombres
Tipos de XML Schema	`xs:`	`http://www.w3.org/2001/XMLSchema`
Instancia de esquema	`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 necesario tener en cuenta algunos aspectos.

- El motor XQuery 1.0 de Altova reconoce los prefijos de la tabla anterior como prefijos enlazados a sus respectivos espacios de nombres.
- Puesto que el espacio de nombres de funciones integradas (*ver la tabla anterior*) es el espacio de nombres predeterminado para las funciones en XQuery, no es necesario usar el prefijo `fn:` cuando se invocan funciones integradas (por ejemplo, `string("Hello")` llamará a la función `fn:string`). Sin embargo, 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 predeterminado para las funciones declarando la expresión `default function namespace` en el prólogo de la consulta.
- Cuando use tipos del espacio de nombres de XML Schema, puede usar el prefijo `xs:` sin necesidad de declarar explícitamente los espacios de nombres ni enlazar estos prefijos a los espacios de nombres en el prólogo de la consulta. (Por ejemplo: `xs:date` y `xs:yearMonthDuration`.) Si desea usar otro prefijo para el espacio de nombres de XML Schema, este se debe declarar explícitamente en el prólogo de la consulta. (Por

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` ya no están en el espacio de nombres Tipos de datos XPath. Ahora están en el espacio de nombres XML Schema (es decir, `xs:yearMonthDuration`.)

Si los espacios de nombres de funciones, constructores de tipos, pruebas de nodos, etc. no se asignan correctamente, entonces se emite un mensaje de error. No obstante, recuerde que algunas funciones tienen el mismo nombre que los tipos de datos del esquema, como por ejemplo `fn:string` y `fn:boolean`. (Se definen tanto `xs:string` como `xs:boolean`.) El prefijo del espacio de nombres determina si se utiliza la función o el constructor de tipos.

Documento XML de origen y validación

Los documentos XML utilizados en la ejecución de un documento XQuery con el motor XQuery 1.0 de Altova deben tener un formato XML correcto. Sin embargo, no tienen que ser obligatoriamente válidos con respecto a un esquema XML. Si el archivo no es válido, se carga el archivo no válido sin la información del esquema. Si el archivo XML está asociado a un esquema externo y es válido con respecto a este esquema, se genera información de validación para los datos XML y esta información se usa para la evaluación de la consulta.

Comprobación de tipos estática y dinámica

La fase de análisis estático revisa aspectos de la consulta como la sintaxis, si existen referencias externas (para módulos, p. ej.), si las funciones y variables invocadas están definidas, etc. En la fase de análisis estático no se comprueban los tipos. Si se detecta un error en la fase de análisis estático, se emite un mensaje y se detiene la ejecución.

La fase de comprobación de tipos dinámica se lleva a cabo en tiempo real, mientras se ejecuta la consulta. Si un tipo no es compatible con los requisitos de una operación, se emite un mensaje de 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 cuyo tipo es `xs:string`.

Módulos de bibliotecas

Los módulos de bibliotecas almacenan funciones y variables para que puedan utilizarse otra vez. El motor XQuery 1.0 de Altova es compatible con módulos que se almacenan **en un solo archivo XQuery externo**. Dicho archivo de módulo debe contener la declaración `module` en el prólogo, que asocia un espacio de nombres de destino. A continuación mostramos un módulo de 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. Para utilizar el módulo, impórtelo al archivo XQuery con la instrucción `import module` en el prólogo de la consulta. La instrucción `import module` solamente importa las funciones y variables declaradas directamente en el archivo de módulo de bibliotecas. Ejemplo:

```
import module namespace modlib = "urn:module-library" at
   "modulefilename.xq";
if    ($modlib:company = "Altova")
```

 © 2014 Altova GmbH

```
then   modlib:webaddress()
else   error("No match found.")
```

Funciones externas

Las funciones externas no son compatibles. Es decir, aquellas expresiones que usan la palabra clave `external`, como por ejemplo:

```
declare function hoo($param as xs:integer) as xs:string external;
```

Intercalaciones

La intercalación predeterminada es la intercalación de punto de código Unicode. Por ahora, esta es la única intercalación compatible. Las comparaciones, incluida la función `fn:max`, se basan en esta intercalación.

Normalización de caracteres

El motor XQuery 1.0 de Altova no es compatible con ninguna forma de normalización de caracteres.

Precisión de los tipos numéricos

- El tipo de datos `xs:integer` tiene 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 máxima de 15 dígitos.

Instrucciones XQuery compatibles

La instrucción `Pragma` no es compatible con el motor XQuery 1.0 de Altova. Si está presente, esta instrucción se ignora y se evalúa la expresión de reserva.

Funciones XQuery compatibles

Para más información sobre el comportamiento de las funciones XQuery 1.0 consulte la sección Funciones XPath 2.0 y XQuery 1.0.

24.1.4 Funciones XPath 2.0 y XQuery 1.0

Las funciones XPath 2.0 y XQuery 1.0 se evalúan con:

- el **motor XPath 2.0 de Altova**, que es un componente del motor XSLT 2.0 de Altova y que se utiliza en el evaluador de XPath de Altova XMLSpy para evaluar expresiones XPath en el documento XML activo en la interfaz de XMLSpy.
- el **motor XQuery 1.0 de Altova**.

Esta sección describe cómo se gestionan funciones XPath 2.0 y XQuery 1.0 en los motores XPath 2.0 y XQuery 1.0. En ella se enumeran las funciones cuyo comportamiento depende de la implementación y aquellas cuyo comportamiento difiere en alguno de los tres entornos en los que se utilizan estas funciones (es decir, en el motor XSLT 2.0, en el motor XQuery 1.0 o en el evaluador de XPath de XMLSpy). Es necesario puntualizar que esta sección no describe el uso de estas funciones. Para más información sobre el uso de estas funciones, consulte la recomendación Funciones y operadores XQuery 1.0 y XPath 2.0 del W3C del 23 de enero de 2007.

Información general

Cumplimiento de estándares

- El motor XPath 2.0 de Altova implementa la recomendación XPath 2.0 del W3C del 23 de enero de 2007. El motor XQuery 1.0 implementa la recomendación XQuery 1.0 del W3C del 23 January 2007. Las funciones XPath 2.0 y XQuery 1.0 compatibles con estos motores cumplen la recomendación sobre funciones y operadores XQuery 1.0 y XPath 2.0 del W3C del 23 de enero de 2007.
- El motor XPath 2.0 de Altova también cumple con las normas de las especificaciones XML 1.0 (4ª Edición) y XML Namespaces (1.0).

Espacio de nombres predeterminado para las funciones

El espacio de nombres predeterminado para las funciones cumple con lo estipulado por el estándar. Por tanto, se puede llamar a las funciones sin el prefijo.

Nodos *boundary-whitespace-only* del documento XML de origen

En los datos XML (y, por tanto, en el conjunto de datos XML) que se pasan a los motores XPath 2.0 y XQuery 1.0 de Altova se eliminan los nodos de texto denominados *boundary-whitespace-only*. Un nodo de texto *boundary-whitespace-only* es un nodo de texto secundario que contiene espacios en blanco y está situado entre dos elementos dentro de un elemento de contenido mixto. La eliminación de estos nodos puede afectar al valor que devuelven las funciones `fn:position()`, `fn:last()`, `fn:count()` y `fn:deep-equal()`.

Por lo general, por cada selección de nodo que selecciona también nodos de texto, los nodos de texto *boundary-whitespace-only* se incluyen en la selección. Sin embargo, puesto que en el conjunto de datos XML utilizado por los motores de Altova se eliminan los nodos de texto *boundary-whitespace-only*, estos nodos no están presentes en el conjunto de datos XML. Como resultado, el tamaño de la selección y la numeración de los nodos de la selección será diferente que en la selección que incluye dichos nodos de texto. Es decir, las funciones `fn:position()`, `fn:last()`, `fn:count()` y `fn:deep-equal()` pueden producir resultados diferentes en los motores XPath 2.0 y XQuery 1.0 de Altova.

Los nodos de texto *boundary-whitespace-only* se evalúan como elementos del mismo nivel de

 © 2014 Altova GmbH

otros elementos sobre todo cuando se usa `xsl:apply-templates` para aplicar elementos. Cuando las funciones `fn:position()`, `fn:last()` y `fn:count()` se usan en patrones con una prueba de nombre (como `para[3]`, que es la abreviatura de `para[position()=3]`), los nodos de texto *boundary-whitespace-only* son irrelevantes porque solamente se seleccionan los elementos con nombre (como `para`). (No obstante, recuerde que los nodos *boundary-whitespace-only* sí son relevantes en los patrones que usan el comodín: como en `*[10]`, por ejemplo.)

Notación numérica

Al generarse el documento de salida, cuando se convierte un `xs:double` en una cadena de texto, se usa la notación científica (p. ej. `1.0E12`) si el valor absoluto es menor que 0,000001 o mayor que 1.000.000. De lo contrario se usa la notación decimal o de enteros.

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 producen un resultado cuyo tipo es `xs:decimal`, la precisión incluye 19 dígitos detrás del punto decimal, sin redondear.

Zona horaria implícita

Cuando es necesario comparar dos valores `date`, `time` o `dateTime`, es necesario conocer la zona horaria de los valores que se deben comparar. Cuando la zona horaria no se ofrece explícitamente en dicho valor, se usa la zona horaria implícita. La zona horaria implícita se toma del reloj del sistema y su valor se puede comprobar con la función `fn:implicit-timezone()`.

Intercalaciones

La única intercalación es la intercalación de punto de código Unicode. Por ahora, no se pueden usar otras intercalaciones. Las comparaciones, incluidas las funciones `fn:max` y `fn:min`, se basan en esta intercalación.

Eje del espacio de nombres

El eje del espacio de nombres se dejó de usar en XPath 2.0. Sin embargo, los motores XPath 2.0 y XQuery 1.0 admiten su uso. Para tener acceso a la información de espacio de nombres con mecanismos XPath 2.0, use las funciones `fn:in-scope-prefixes()`, `fn:namespace-uri()` y `fn:namespace-uri-for-prefix()`.

Comprobación de tipos estática

No es compatible la característica opcional de comprobación de tipos estática.

Funciones compatibles

La tabla que aparece a continuación enumera (por orden alfabético) ciertas funciones y su comportamiento durante la implementación. Antes de consultar la tabla, tenga en cuenta estos aspectos:

- Por lo general, cuando una función espera una secuencia de un componente como argumento y se presenta una secuencia con más de un componente, entonces la función devuelve un error.
- Todas las comparaciones de cadenas se hacen usando la intercalación de punto de

código Unicode.
- Los resultados que son QName se serializan en la forma `[prefix:]localname`.

Nombre de la función	Notas
`base-uri`	• Si se usan entidades externas en el XML de origen y si un nodo de la entidad externa se definió como argumento del nodo de entrada de la función `base-uri()`, se usa de todas maneras el URI base del documento XML que lo incluye (en lugar del URI base de la entidad externa). • El URI base de un nodo del archivo XML se puede modificar con el atributo `xml:base`.

`collection`	• El argumento es un URI relativo que se resuelve con respecto al URI base actual.
	• Si el URI resuelto identifica un archivo XML, entonces el archivo XML se trata como si fuese un catálogo que hace referencia a una colección de archivos. Este archivo debe tener este formato:
	`<collection>`
	`<doc href="uri-1" />`
	`<doc href="uri-2" />`
	`<doc href="uri-3" />`
	`</collection>`
	Los archivos a los que hacen referencia los atributos `href` se cargan y sus nodos de documento se devuelven en forma de secuencia.
	• Si el URI resuelto no identifica ningún archivo XML con la estructura de catálogo anterior, entonces la cadena de argumentos (donde están permitidos comodines como `?` y `*`) se utiliza como cadena de búsqueda. Se cargan los archivos XML cuyo nombre coincide con la expresión de búsqueda y sus nodos de documento se devuelven en forma de secuencia. Ver los ejemplos siguientes.
	• Ejemplo XSLT: la expresión `collection("c:\MyDocs\*.xml")//Title` devuelve una secuencia con todos los elementos `DocTitle` de los archivos `.xml` de la carpeta `MyDocs`.
	• Ejemplo XQuery: la expresión `{for $i in collection(c:\MyDocs\*.xml) return element doc{base-uri($i)}}` devuelve los URI base de todos los archivos `.xml` de la carpeta `MyDocs`, estando cada URI dentro de un elemento `doc`.
	• La intercalación predeterminada está vacía.

cont./

Nombre de la función	Notas

count	• Ver nota sobre espacios en blanco en el apartado Información general.
current-date, current-dateTime, current-time	• La fecha y la hora actuales son las del reloj del sistema. • La zona horaria es la zona horaria ofrecida por el contexto de evaluación. La zona horaria implícita es la del reloj del sistema. • La zona horaria siempre se especifica en el resultado.
deep-equal	• Ver nota sobre espacios en blanco en el apartado Información general.
doc	• Se emite un error solamente si no hay ningún archivo XML disponible en la ubicación especificada o si el archivo XML no tiene un formato XML correcto. El archivo se valida si hay un esquema asociado. Si no es válido, el archivo se carga sin la información de esquema.
id	• En un documento con formato XML correcto pero no válido que contiene un mínimo de dos elementos con el mismo valor para ID, se devuelve el primer elemento.
in-scope-prefixes	• Solamente se pueden dejar de declarar en el documento XML los espacios de nombres predeterminados. Sin embargo, aunque se deje de declarar un espacio de nombres predeterminado en un nodo de elemento, se devuelve el prefijo para el espacio de nombres predeterminado (la cadena de longitud cero) para ese nodo.
last	• Ver nota sobre espacios en blanco en el apartado Información general.
lower-case	• El conjunto de caracteres Unicode es compatible.
normalize-unicode	• Las formas de normalización NFC, NFD, NFKC y NFKD son compatibles.

cont./

Nombre de la función	Notas
position	• Ver nota sobre espacios en blanco en el apartado Información general.
resolve-uri	• Si se omite el segundo argumento opcional, el URI que se

	debe resolver (el primer argumento) se resuelve con respecto al URI base del contexto estático, que es el URI de la hoja de estilos XSLT o el URI base dado en el prólogo del documento XQuery. • El URI relativo (el primer argumento) se anexa después del último `"/"` de la notación de ruta de acceso de la notación del URI base. • Si el valor del primer argumento es la cadena de longitud cero, se devuelve el URI base del contexto estático y este URI incluye el nombre de archivo del documento del que se deriva el URI base del contexto estático (p. ej. el archivo XSLT o XML).
`static-base-uri`	• El URI base del contexto estático es el URI base de la hoja de estilos XSLT o el URI base especificado en el prólogo del documento XQuery. • Cuando se utiliza en el evaluador de XPath del entorno XMLSpy, el URI base del contexto estático es el URI del documento XML activo.
`upper-case`	• El conjunto de caracteres Unicode es compatible.

© 2014 Altova GmbH

24.1.5 Funciones de extensión XSLT y XQuery

En los lenguajes de programación, como Java y C#, hay varias funciones predefinidas que no están disponibles como funciones XQuery/XPath ni como funciones XSLT. Un buen ejemplo son las funciones matemáticas disponibles en Java, como `sin()` y `cos()`. Si estas funciones estuvieran a disposición de los diseñadores de hojas de estilos XSLT y consultas XQuery, el ámbito de aplicación de las hojas de estilos y de las consultas sería mayor y sería más fácil crearlas.

Los motores XSLT y XQuery utilizados en los productos de Altova admiten el uso de funciones de extensión en Java y .NET. También admiten el uso de scripts MSXSL para XSLT y las funciones de extensión de Altova.

Tenga en cuenta que, excepto algunas funciones de extensión de Altova para XSLT, a casi todas las funciones de extensión que aparecen en este anexo se les llama desde expresiones XPath. Este anexo describe cómo usar funciones de extensión y scripts MSXSL en sus hojas de estilos XSLT y en sus documentos XQuery. El anexo se divide en estos apartados:

- Funciones de extensión de Altova
- Funciones de extensión Java
- Funciones de extensión .NET
- Scripts MSXSL para XSLT

Los dos principales aspectos que se tienen en cuenta en estos apartados son: (i) cómo se llama a las funciones de cada biblioteca y (ii) qué reglas se siguen para convertir los argumentos de una llamada a función en el formato de entrada de la función y qué reglas se siguen para la conversión de retorno (el resultado de la función se convierte en objeto de datos XSLT/XQuery).

Requisitos
Para poder trabajar con funciones de extensión, en el equipo que ejecuta la transformación XSLT o la ejecución XQuery debe tener instalado (o debe tener acceso a) un entorno de ejecución Java (para acceder a funciones Java) y .NET Framework 2.0 (mínimo, para acceder a funciones .NET).

Funciones de extensión de Altova

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

```
altova:
```

que se supone estará enlazado al espacio de nombres señalado.

Las funciones de extensión de Altova se pueden usar tanto en un contexto XPath como en un contexto XSLT (es decir, como funciones XPath y como funciones XSLT respectivamente). Asegúrese de utilizar la función de extensión en el contexto adecuado (XPath o XSLT).

Las funciones de extensión descritas en esta sección son compatibles con la versión actual de su producto 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.

Funciones XPath
Puede usar estas funciones en contextos XPath:

- `altova:generate-auto-number()`
- `altova:reset-auto-number()`
- `altova:get-temp-folder()`

Funciones XSLT
Puede usar estas funciones en un contexto XSLT, igual que las funciones XSLT 2.0 `current-group()` o `key()`:

- **`altova:evaluate()`**
- **`altova:distinct-nodes()`**
- **`altova:encode-for-rtf()`**
- **`altova:xbrl-labels()`**
- **`altova:xbrl-footnotes()`**

Nota: las funciones XBRL de Altova solamente están disponibles en los productos de Altova compatibles con XBRL.

Funciones de gráficos (solamente en las ediciones Enterprise y Server)
Las funciones de extensión de Altova para gráficos solamente son compatibles con las ediciones Enterprise y Server de los productos de Altova y permiten generar gráficos a partir de datos XML. Hay dos tipos de funciones para gráficos:

- Funciones para generar y guardar gráficos
- Funciones para crear gráficos

En este apartado también incluimos un fragmento de código de la estructura XML de los datos de gráfico, a partir de la cual se pueden generar gráficos. Para terminar, ofrecemos un documento XSLT de muestra que ilustra cómo usar funciones para gráficos para generar gráficos a partir de datos XML.

Funciones XSLT generales

Las funciones XSLT descritas en este apartado son compatibles con la versión actual de su producto 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.

Recuerde que las funciones de extensión de Altova están en el **espacio de nombres de funciones de extensión de Altova**

 © 2014 Altova GmbH

http://www.altova.com/xslt-extensions

y en esta sección se presentan con el prefijo

altova:

que se supone estará enlazado al espacio de nombres señalado.

Estas funciones generales están disponibles en un contexto XSLT, más o menos como las funciones XSLT 2.0 `current-group()` o `key()`:

- **altova:evaluate()**
- **altova:distinct-nodes()**
- **altova:encode-for-rtf()**
- **altova:xbrl-labels()**
- **altova:xbrl-footnotes()**

Nota: las funciones XBRL de Altova solamente están disponibles en los productos de Altova compatibles con XBRL.

Funciones para contextos XSLT

Estas funciones se pueden usar en contextos XSLT.

altova:evaluate()

Toma una expresión XPath, pasada en forma de cadena, como argumento obligatorio. Devuelve el resultado de la expresión evaluada.

altova:evaluate(XPathExp as xs:string)

Por ejemplo:

altova:evaluate('//Name[1]')

Aquí, la expresión `//Name[1]` se pasa como cadena si se escribe entre comillas simples. La función `altova:evaluate` devuelve el contenido del primer elemento `Name` del documento.

La función `altova:evaluate` puede tomar otros argumentos (opcionales). Estos argumentos son los valores de las variables de nombre `p1`, `p2`, `p3`... `pN` que se pueden usar en la expresión XPath.

altova:evaluate(`XPathExp as xs:string` **[, p1value ... pNvalue])**

donde

- los nombres de variable deben tener el formato `pX`, siendo `X` un número entero
- la secuencia de los argumentos de la función, a partir del segundo argumento, corresponde a la secuencia de las variables `p1` a `pN`. De modo que el segundo argumento será el valor de la variable `p1`, el tercero será el valor de la variable `p2` y así sucesivamente.

- los valores de variable deben ser de tipo `item*`

Por ejemplo:

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate( $xpath, 10, 20, 'hi' )" />
```
Resultado: "`hi 20 10`"

Observe que en el ejemplo anterior:

- 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.
- el cuarto argumento de la función es un valor de cadena, al estar escrito entre comillas simples.
- el atributo `select` del elemento `xs:variable` suministra la expresión XPath. Puesto que la expresión debe ser de tipo `xs:string`, se escribe entre comillas simples.

Aquí tiene más ejemplos:

```
<xsl:variable name="xpath" select="'$p1'" />
<xsl:value-of select="altova:evaluate( $xpath, //Name[1] )" />
```
Resultado: el valor del primer elemento `Name`.

```
<xsl:variable name="xpath" select="'$p1'" />
<xsl:value-of select="altova:evaluate( $xpath, '//Name[1]' )" />
```
Resultado: "`//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. Digamos 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`).

En la función de extensión `altova:evaluate()`puede usar estas variables:

- Variables estáticas: `<xsl:value-of select="$i3, $i2, $i1" />`
  Resultado: el valor de las tres variables.

- Expresiones XPath dinámicas con variables dinámicas:

 © 2014 Altova GmbH

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate( $xpath, 10, 20, 30 )" />
```
Resultado: "30 20 10"

- Expresiones XPath dinámicas sin variables dinámicas:
```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate( $xpath )" />
```
Resultado: "No variable defined for $p3."

Nota: el contexto estático incluye espacios de nombres, tipos y funciones (pero no variables) del entorno desde donde se llama a la función. El URI base y el espacio de nombres predeterminado se heredan.

altova:distinct-nodes()

La función `altova:distinct-nodes()` toma un conjunto de nodos como entrada y devuelve el mismo conjunto menos los nodos que tienen el mismo valor. La comparación se realiza por medio de la función XPath/XQuery `fn:deep-equal`.

```
altova:distinct-nodes( $arg as node()* )  as node()*
```

altova:encode-for-rtf()

La función `altova:encode-for-rtf()` convierte la cadena de entrada en código para RTF.

```
altova:encode-for-rtf( $inputstr as xs:string?,
$preserveallwhitespace as xs:boolean,
$preservenewlines as xs:boolean) as xs:string
```

Los espacios en blanco y las líneas nuevas se conservan dependiendo del valor booleano especificado en sus respectivos parámetros.

altova:xbrl-labels()

La función `altova:xbrl-labels()` toma dos argumentos de entrada: un nombre de nodo y la ubicación del archivo de taxonomía que incluye el nodo. La función devuelve las etiquetas XBRL asociadas al nodo de entrada.

```
altova:xbrl-labels( $name as xs:QName, $file as xs:string )  as node()*
```

altova:xbrl-footnotes()

La función `altova:footnotes()` toma un nodo como argumento de entrada y devuelve el conjunto de los nodos de nota al pie XBRL a los que hace referencia el nodo de entrada.

```
altova:footnotes( $arg as node() )  as node()*
```

Funciones XPath generales

Las funciones de uso general que aparecen en este apartado se pueden usar en contextos XPath y son compatibles con la versión actual de su producto 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.

Recuerde que las funciones de extensión de Altova están en el **espacio de nombres de funciones de extensión de Altova**

`http://www.altova.com/xslt-extensions`

y en esta sección se presentan con el prefijo

`altova:`

que se supone estará enlazado al espacio de nombres señalado.

Estas funciones de uso general están disponibles en contextos XPath:

- **`altova:generate-auto-number()`**
- **`altova:reset-auto-number()`**
- **`altova:get-temp-folder()`**

Funciones para contextos XPath
Estas funciones se pueden usar en contextos XPath.

`altova:generate-auto-number(id as xs:string, start-with as xs:double, increment as xs:double, reset-on-change as xs:string)`
Genera una serie de números que tienen el ID especificado. Se especifica el entero inicial y el incremento.

`altova:reset-auto-number(id as xs:string)`
Esta función restablece la numeración automática de la serie de numeración automática especificada con el argumento de Id. La serie vuelve a empezar por el entero inicial de la serie (ver la función `altova:generate-auto-number`).

`altova:get-temp-folder as xs:string`

 © 2014 Altova GmbH

Obtiene la carpeta temporal.

`altova:repeat-string(xs:string, xs:integer) as xs:string`
Genera un `xs:string` que es la cadena de entrada (primer parámetro) repetido `x` veces, siendo `x` el segundo parámetro.

Funciones para gráficos (XPath)

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.

Hay dos tipos de funciones para gráficos:

- 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 los 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 los 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

`altova:generate-chart-image` `($chart, $width, $height, $encoding) as atomic`

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `altova:create-chart`
- `$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.

`altova:generate-chart-image` `($chart, $width, $height, $encoding, $imagetype) as atomic`

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `altova:create-chart`
- `$width` y `$height` deben especificarse con una unidad de longitud
- `$encoding` puede ser `binarytobase64` o `binarytobase16`
- `$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.

`altova:save-chart-image` `($chart, $filename, $width, $height) as empty()` *(solo en Windows)*

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `altova:create-chart`
- `$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`.

`altova:save-chart-image` `($chart, $filename, $width, $height, $imagetype) as empty()` *(solo en Windows)*

donde

- `$chart` es el componente de extensión de gráficos obtenido con la función `altova:create-chart`
- `$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.

`altova:create-chart``($chart-config, $chart-data-series*)` *como componente de extensión de gráficos*

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `altova:create-chart-config` o con la función `altova:create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series obtenido con la función `altova:create-chart-data-series` o con la función `altova: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.

`altova:create-chart-config``($type-name, $title)` *como componente de extensión* `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.

`altova:create-chart-config-from-xml``($xml-struct)` *como componente de extensión* `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.

`altova:create-chart-data-series``($series-name?, $x-values*, $y-values*)` *como componente de extensión* `chart-data-series`

donde

- `$series-name` especifica el nombre de la serie
- `$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.

`altova:create-chart-data-row``(x, y1, y2, y3, ...)` *como componente de extensión* `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.

`altova:create-chart-data-series-from-rows``($series-names as xs:string*, $row*)` *como componente de extensión* `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.

`altova:create-chart-layer``($chart-config, $chart-data-series*)` *como componente de extensión* `chart-layer`

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `altova:create-chart-config` o con la función `altova:create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series que se obtiene con la función `altova:create-chart-data-series` o con la función `altova:create-chart-data-series-from-rows`

La función devuelve un componente de extensión chart-layer, que contiene los datos de la capa de gráfico.

`altova:create-multi-layer-chart``($chart-config, $chart-data-series*, $chart-layer*)`

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `altova:create-chart-config` o con la función `altova:create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series obtenido con la función `altova:create-chart-data-series` o con la función `altova:create-chart-data-series-from-rows`
- `$chart-layer` es el componente de extensión chart-layer obtenido con la función `altova:create-chart-layer`

La función devuelve un componente de extensión multi-layer-chart.

`altova:create-multi-layer-chart``($chart-config, $chart-data-series*, $chart-layer*, xs:boolean $mergecategoryvalues)`

 © 2014 Altova GmbH

donde

- `$chart-config` es el componente de extensión chart-config obtenido con la función `altova:create-chart-config` o con la función `altova:create-chart-config-from-xml`
- `$chart-data-series` es el componente de extensión chart-data-series obtenido con la función `altova:create-chart-data-series` o con la función `altova:create-chart-data-series-from-rows`
- `$chart-layer` es el componente de extensión chart-layer obtenido con la función `altova:create-chart-layer`

La función devuelve un componente de extensión multi-layer-chart.

Estructura XML de los datos de gráfico

A continuación puede ver una estructura XML de los 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*
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.%"/>
</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>
```

Ejemplo: funciones de 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 o 3.0.

Nota: las funciones para gráficos son compatibles solamente con las ediciones **Enterprise** y **Server Editions** 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.

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

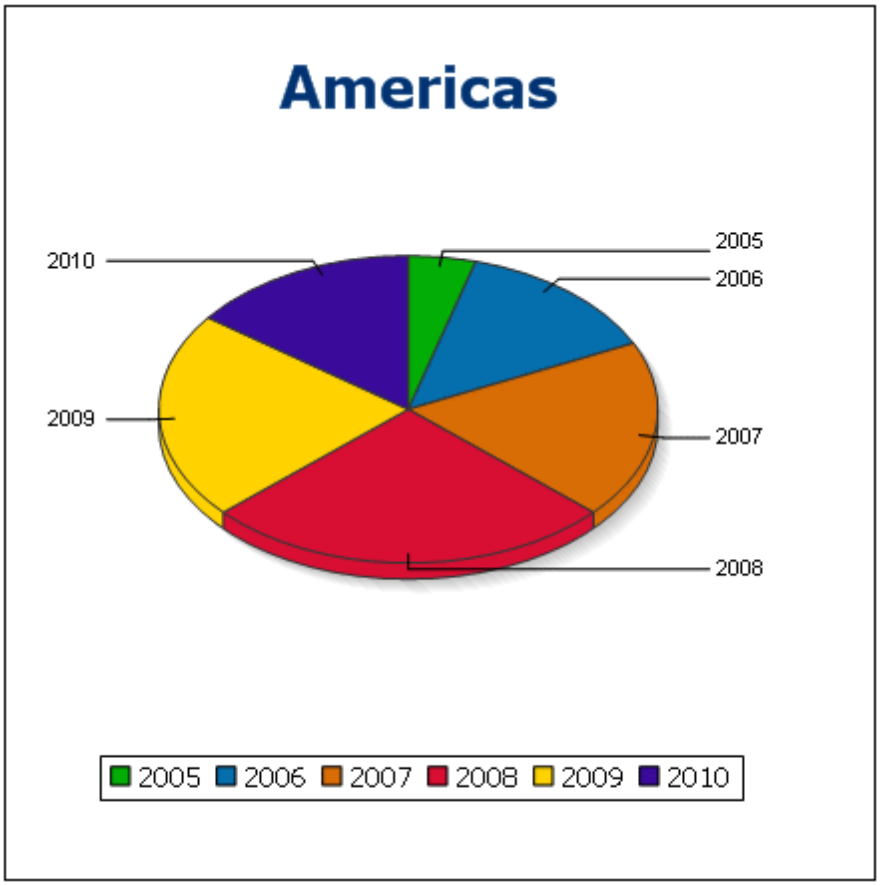
Este documento XML se puede procesar con el XSLT que aparece arriba. Los datos del documento XML se usan para generar el gráfico circular de la imagen que aparece más abajo.

```
<?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.

 © 2014 Altova GmbH



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:

- Java: constructores
- Java: métodos estáticos y campos estáticos
- Java: métodos de instancia y campos de instancia
- Tipos de datos: XSLT/XQuery a Java
- Tipos de datos: Java a XSLT/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.

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.

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 más abajo*.)
- 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 más abajo*.)

- El archivo de clases está en un paquete. El archivo XSLT/XQuery está en una carpeta cualquiera. (*Ver ejemplo más abajo*.)
- El archivo de clases no está en un paquete. El archivo XSLT/XQuery está una carpeta cualquiera. (*Ver ejemplo más abajo*.)

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`.

```
<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>
```

 © 2014 Altova GmbH

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.

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!/"
```

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.

Java: 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`

 © 2014 Altova GmbH

(`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" />
```

Java: 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()`).

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>
```

Java: 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`.

Tipos de datos: XPath/XQuery a 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

 © 2014 Altova GmbH

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 (`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.

Tipos de datos: Java a 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).

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:

- .NET: constructores
- .NET: métodos estáticos y campos estáticos
- .NET: métodos de instancia y campos de instancia
- Tipos de datos: XSLT/XQuery en .NET
- Tipos de datos: .NET en XSLT/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

 © 2014 Altova GmbH

	`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`:

```
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.

.NET: 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:

 © 2014 Altova GmbH

```
<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" />
```

.NET: métodos 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>
```

.NET: 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.

Tipos de datos: XPath/XQuery a .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.

 © 2014 Altova GmbH

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.

Tipos de datos: .NET a 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).

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>`.

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.

 © 2014 Altova GmbH

24.2 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

24.2.1 Requisitos de SO y memoria

Sistema operativo
Las aplicaciones de software de Altova están disponibles en estas plataformas:

- Aplicaciones de 32 bits para Windows XP, Windows Vista, Windows 7, Windows 8, Windows Server 2003 y 2008
- Aplicaciones de 64 bits para Windows Vista, Windows 7, Windows 8 y Windows Server 2012

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.

 © 2014 Altova GmbH

24.2.2 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.

24.2.3 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.*

© 2014 Altova GmbH

24.2.4 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.

24.2.5 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 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 pestaña *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.

 © 2014 Altova GmbH

24.3 Información sobre licencias

Esta sección incluye:

- información sobre la distribución de este producto de software.
- información sobre la activación del software y modelo de licencias.
- información sobre los derechos de propiedad intelectual relacionados con 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.

24.3.1 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 al iniciarse la aplicación. Si desea usar 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 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.

 © 2014 Altova GmbH

24.3.2 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.

24.3.3 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.

© 2014 Altova GmbH

24.3.4 Contrato de licencia para el usuario final de Altova

THIS IS A LEGAL DOCUMENT -- RETAIN FOR YOUR RECORDS

ALTOVA® END USER LICENSE AGREEMENT

Licensor:
Altova GmbH
Rudolfsplatz 13a/9
A-1010 Wien
Austria

Important - Read Carefully. Notice to User:

This End User License Agreement ("Agreement") is a legal document between you and Altova GmbH ("Altova"). It is important that you read this document before using the Altova-provided software ("Software") and any accompanying documentation, including, without limitation printed materials, 'online' files, or electronic documentation ("Documentation"). By clicking the "I accept" and "Next" buttons below, or by installing, or otherwise using the Software, you agree to be bound by the terms of this Agreement as well as the Altova Privacy Policy ("Privacy Policy") including, without limitation, the warranty disclaimers, limitation of liability, data use and termination provisions below, whether or not you decide to purchase the Software. You agree that this agreement is enforceable like any written agreement negotiated and signed by you. If you do not agree, you are not licensed to use the Software, and you must destroy any downloaded copies of the Software in your possession or control. You may print a copy of this Agreement as part of the installation process at the time of acceptance. Alternatively, a copy of this Agreement may be found at http://www.altova.com/eula and a copy of the Privacy Policy may be found at http://www.altova.com/privacy.

1. SOFTWARE LICENSE

(a) License Grant.

(i) Upon your acceptance of this Agreement Altova grants you a non-exclusive, non-transferable (except as provided below), limited license, without the right to grant sublicenses, to install and use a copy of the Software on one compatible personal computer or workstation up to the Permitted Number of computers. Subject to the limitations set forth in Section 1(c), you may install and use a copy of the Software on more than one of your compatible personal computers or workstations if you have purchased a Named-User license. Subject to the limitations set forth in Sections 1(d) and 1(e), users may use the software concurrently on a network. The Permitted Number of computers and/or users and the type of license, e.g. Installed, Named-Users, and Concurrent-User, shall be determined and specified at such time as you elect to purchase the Software. Installed user licenses are intended to be fixed and not concurrent. In other words, you cannot uninstall the Software on one machine in order to reinstall that license to a different machine and then uninstall and reinstall back to the original machine. Installations should be static. Notwithstanding the foregoing, permanent uninstallations and redeployments are acceptable in limited circumstances such as if an employee leaves the company or the machine is permanently decommissioned. During the evaluation period, hereinafter defined, only a single user may install and use the software on one (1) personal computer or workstation. If you have licensed the Software as part of a suite of Altova software products (collectively, the "Suite") and have not installed each product individually, then the Agreement governs your use of all of the software included in the Suite.

(ii) If you have licensed SchemaAgent, then the terms and conditions of this Agreement apply to your use of the SchemaAgent server software ("SchemaAgent Server") included therein, as applicable, and you are licensed to use SchemaAgent Server solely in connection with your use of Altova Software and solely for the purposes described in the accompanying documentation.

(iii) If you have licensed Software that enables users to generate source code, your license to install and use a copy of the Software as provided herein permits you to generate source code based on (i) Altova Library modules that are included in the Software (such generated code hereinafter referred to as the "Restricted Source Code") and (ii) schemas or mappings that you create or provide (such code as may be generated from your schema or mapping source materials hereinafter referred to as the "Unrestricted Source Code"). In addition to the rights granted herein, Altova grants you a non-exclusive, non-transferable, limited license to compile the complete generated code (comprised of the combination of the Restricted Source Code and the Unrestricted Source Code) into executable object code form, and to use, copy, distribute or license that executable. You may not distribute or redistribute, sublicense, sell, or transfer the Restricted Source Code to a third-party in the un-compiled form unless said third-party already has a license to the Restricted Source Code through their separate agreement with Altova. Notwithstanding anything to the contrary herein, you may not distribute, incorporate or combine with other software, or otherwise use the Altova Library modules or Restricted Source Code, or any Altova intellectual property embodied in or associated with the Altova Library modules or Restricted Source Code, in any manner that would subject the Restricted Source Code to the terms of a copyleft, free software or open source license that would require the Restricted Source Code or Altova Library modules source code to be disclosed in source code form. Notwithstanding anything to the contrary herein, you may not use the Software to develop and distribute other software programs that directly compete with any Altova software or service without prior written permission. Altova reserves all other rights in and to the Software. With respect to the feature(s) of UModel that permit reverse-engineering of your own source code or other source code that you have lawfully obtained, such use by you does not constitute a violation of this Agreement. Except as otherwise expressly permitted in Section 1(j) reverse engineering of the Software is strictly prohibited as further detailed therein.

(iv) In the event Restricted Source Code is incorporated into executable object code form, you will include the following statement in (1) introductory splash screens, or if none, within one or more screens readily accessible by the end-user, and (2) in the electronic and/or hard copy documentation: "Portions of this program were developed using Altova® [name of Altova Software, e.g. MapForce® 2011] and includes libraries owned by Altova GmbH, Copyright © 2007-2011 Altova GmbH (www.altova.com)."

(b) Server Use for Installation and Use of SchemaAgent. You may install one (1) copy of the Software on a computer file server within your internal network solely for the purpose of downloading and installing the Software onto other computers within your internal network up to the Permitted Number of computers in a commercial environment only. If you have licensed SchemaAgent, then you may install SchemaAgent Server on any server computer or workstation and use it in connection with your Software. No other network use is permitted, including without limitation using the Software either directly or through commands, data or instructions from or to a computer not part of your internal network, for Internet or Web-hosting services or by any user not licensed to use this copy of the Software through a valid license from Altova.

(c) Named-Use. If you have licensed the "Named-User" version of the software, you may install the Software on up to five (5) compatible personal computers or workstations of which you are the primary user thereby allowing you to switch from one computer to the other as necessary provided that only one (1) instance of the Software will be used by you as the Named-User at any

given time. If you have purchased multiple Named-User licenses, each individual Named-User will receive a separate license key code.

(d) Concurrent Use in Same Physical Network or Office Location. If you have licensed a "Concurrent-User" version of the Software, you may install the Software on any compatible computers in a commercial environment only, up to ten (10) times the Permitted Number of users, provided that only the Permitted Number of users actually use the Software at the same time and further provided that the computers on which the Software is installed are on the same physical computer network. The Permitted Number of concurrent users shall be delineated at such time as you elect to purchase the Software licenses. Each separate physical network or office location requires its own set of separate Concurrent User Licenses for those wishing to use the Concurrent User versions of the Software in more than one location or on more than one network, all subject to the above Permitted Number limitations and based on the number of users using the Software. If a computer is not on the same physical network, then a locally installed user license or a license dedicated to concurrent use in a virtual environment is required. Home User restrictions and limitations with respect to the Concurrent User licenses used on home computers are set forth in Section 1(g).

(e) Concurrent Use in Virtual Environment. If you have purchased Concurrent-User Licenses, you may install a copy of the Software on a terminal server (Microsoft Terminal Server or Citrix Metaframe), application virtualization server (Microsoft App-V, Citrix XenApp, or VMWare ThinApp) or virtual machine environment within your internal network for the sole and exclusive purpose of permitting individual users within your organization to access and use the Software through a terminal server, application virtualization session, or virtual machine environment from another computer provided that the total number of users that access or use the Software concurrently at any given point in time on such network, virtual machine or terminal server does not exceed the Permitted Number; and provided that the total number of users authorized to use the Software through the terminal server, application virtualization session, or virtual machine environment does not exceed ten (10) times the Permitted Number of users. In a virtual environment, you must deploy a reliable and accurate means of preventing users from exceeding the Permitted Number of concurrent users. Altova makes no warranties or representations about the performance of Altova software in a terminal server, application virtualization session, or virtual machine environment and the foregoing are expressly excluded from the limited warranty in Section 5 hereof. Technical support is not available with respect to issues arising from use in such environments.

(f) Backup and Archival Copies. You may make one (1) backup and one (1) archival copy of the Software, provided your backup and archival copies are not installed or used on any computer and further provided that all such copies shall bear the original and unmodified copyright, patent and other intellectual property markings that appear on or in the Software. You may not transfer the rights to a backup or archival copy unless you transfer all rights in the Software as provided under Section 3.

(g) Home Use (Personal and Non-Commercial). In order to further familiarize yourself with the Software and allow you to explore its features and functions, you, as the primary user of the computer on which the Software is installed for commercial purposes, may also install one copy of the Software on only one (1) home personal computer (such as your laptop or desktop) solely for your personal and non-commercial ("HPNC") use. This HPNC copy may not be used in any commercial or revenue-generating business activities, including without limitation, work-from-home, teleworking, telecommuting, or other work-related use of the Software. The HPNC copy of the Software may not be used at the same time on a home personal computer as the Software is being used on the primary computer.

(h) Key Codes, Upgrades and Updates. Prior to your purchase and as part of the registration for the thirty (30) day evaluation period, as applicable, you will receive an evaluation key code. You will receive a purchase key code when you elect to purchase the Software from either Altova GmbH or an authorized reseller. The purchase key code will enable you to activate the Software beyond the initial evaluation period. You may not re-license, reproduce or distribute any key code except with the express written permission of Altova. If the Software that you have licensed is an upgrade or an update, then the latest update or upgrade that you download and install replaces all or part of the Software previously licensed. The update or upgrade and the associated license keys does not constitute the granting of a second license to the Software in that you may not use the upgrade or updated copy in addition to the copy of the Software that it is replacing and whose license has terminated.

(i) Title. Title to the Software is not transferred to you. Ownership of all copies of the Software and of copies made by you is vested in Altova, subject to the rights of use granted to you in this Agreement. As between you and Altova, documents, files, stylesheets, generated program code (including the Unrestricted Source Code) and schemas that are authored or created by you via your utilization of the Software, in accordance with its Documentation and the terms of this Agreement, are your property unless they are created using Evaluation Software, as defined in Section 4 of this Agreement, in which case you have only a limited license to use any output that contains generated program code (including Unrestricted Source Code) such as Java, C++, C#, VB.NET or XSLT and associated project files and build scripts, as well as generated XML, XML Schemas, documentation, UML diagrams, and database structures only for the thirty (30) day evaluation period.

(j) Reverse Engineering. Except and to the limited extent as may be otherwise specifically provided by applicable law in the European Union, you may not reverse engineer, decompile, disassemble or otherwise attempt to discover the source code, underlying ideas, underlying user interface techniques or algorithms of the Software by any means whatsoever, directly or indirectly, or disclose any of the foregoing, except to the extent you may be expressly permitted to decompile under applicable law in the European Union, if it is essential to do so in order to achieve operability of the Software with another software program, and you have first requested Altova to provide the information necessary to achieve such operability and Altova has not made such information available. Altova has the right to impose reasonable conditions and to request a reasonable fee before providing such information. Any information supplied by Altova or obtained by you, as permitted hereunder, may only be used by you for the purpose described herein and may not be disclosed to any third party or used to create any software which is substantially similar to the expression of the Software. Requests for information from users in the European Union with respect to the above should be directed to the Altova Customer Support Department.

(k) Other Restrictions. You may not loan, rent, lease, sublicense, distribute or otherwise transfer all or any portion of the Software to third parties except to the limited extent set forth in Section 3 or as otherwise expressly provided. You may not copy the Software except as expressly set forth above, and any copies that you are permitted to make pursuant to this Agreement must contain the same copyright, patent and other intellectual property markings that appear on or in the Software. You may not modify, adapt or translate the Software. You may not, directly or indirectly, encumber or suffer to exist any lien or security interest on the Software; knowingly take any action that would cause the Software to be placed in the public domain; or use the Software in any computer environment not specified in this Agreement. You may not permit any use of or access to the Software by any third party in connection with a commercial service offering, such as for a cloud-based or web-based SaaS offering.

You will comply with applicable law and Altova's instructions regarding the use of the

Software. You agree to notify your employees and agents who may have access to the Software of the restrictions contained in this Agreement and to ensure their compliance with these restrictions.

(I) NO GUARANTEE. THE SOFTWARE IS NEITHER GUARANTEED NOR WARRANTED TO BE ERROR-FREE NOR SHALL ANY LIABILITY BE ASSUMED BY ALTOVA IN THIS RESPECT. NOTWITHSTANDING ANY SUPPORT FOR ANY TECHNICAL STANDARD, THE SOFTWARE IS NOT INTENDED FOR USE IN OR IN CONNECTION WITH, WITHOUT LIMITATION, THE OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION, COMMUNICATION SYSTEMS, AIR TRAFFIC CONTROL EQUIPMENT, MEDICAL DEVICES OR LIFE SUPPORT SYSTEMS, MEDICAL OR HEALTH CARE APPLICATIONS, OR OTHER APPLICATIONS WHERE THE FAILURE OF THE SOFTWARE OR ERRORS IN DATA PROCESSING COULD LEAD TO DEATH, PERSONAL INJURY OR SEVERE PHYSICAL OR ENVIRONMENTAL DAMAGE. YOU AGREE THAT YOU ARE SOLELY RESPONSIBLE FOR THE ACCURACY AND ADEQUACY OF THE SOFTWARE AND ANY DATA GENERATED OR PROCESSED BY THE SOFTWARE FOR YOUR INTENDED USE AND YOU WILL DEFEND, INDEMNIFY AND HOLD ALTOVA, ITS OFFICERS AND EMPLOYEES HARMLESS FROM ANY THIRD PARTY CLAIMS, DEMANDS, OR SUITS THAT ARE BASED UPON THE ACCURACY AND ADEQUACY OF THE SOFTWARE IN YOUR USE OR ANY DATA GENERATED BY THE SOFTWARE IN YOUR USE.

2. INTELLECTUAL PROPERTY RIGHTS

You acknowledge that the Software and any copies that you are authorized by Altova to make are the intellectual property of and are owned by Altova and its suppliers. The structure, organization and code of the Software are the valuable trade secrets and confidential information of Altova and its suppliers. The Software is protected by copyright, including without limitation by United States Copyright Law, international treaty provisions and applicable laws in the country in which it is being used. You acknowledge that Altova retains the ownership of all patents, copyrights, trade secrets, trademarks and other intellectual property rights pertaining to the Software, and that Altova's ownership rights extend to any images, photographs, animations, videos, audio, music, text and "applets" incorporated into the Software and all accompanying printed materials. You will take no actions which adversely affect Altova's intellectual property rights in the Software. Trademarks shall be used in accordance with accepted trademark practice, including identification of trademark owners' names. Trademarks may only be used to identify printed output produced by the Software, and such use of any trademark does not give you any right of ownership in that trademark. Altova®, XMLSpy®, Authentic®, StyleVision®, MapForce®, UModel®, DatabaseSpy®, DiffDog®, SchemaAgent®, SemanticWorks®, MissionKit®, Markup Your Mind®, Nanonull™, RaptorXML™, RaptorXML Server™, RaptorXML +XBRL Server™, Powered By RaptorXML™, FlowForce Server™, StyleVision Server™, and MapForce Server™ are trademarks of Altova GmbH. (pending or registered in numerous countries). Unicode and the Unicode Logo are trademarks of Unicode, Inc. Windows, Windows XP, Windows Vista, Windows 7, and Windows 8 are trademarks of Microsoft. W3C, CSS, DOM, MathML, RDF, XHTML, XML and XSL are trademarks (registered in numerous countries) of the World Wide Web Consortium (W3C); marks of the W3C are registered and held by its host institutions, MIT, INRIA and Keio. Except as expressly stated above, this Agreement does not grant you any intellectual property rights in the Software. Notifications of claimed copyright infringement should be sent to Altova's copyright agent as further provided on the Altova Web Site.

3. LIMITED TRANSFER RIGHTS

Notwithstanding the foregoing, you may transfer all your rights to use the Software to another person or legal entity provided that: (a) you also transfer this Agreement, the Software and all

other software or hardware bundled or pre-installed with the Software, including all copies, updates and prior versions, and all copies of font software converted into other formats, to such person or entity; (b) you retain no copies, including backups and copies stored on a computer; (c) the receiving party secures a personalized key code from Altova; and (d) the receiving party accepts the terms and conditions of this Agreement and any other terms and conditions upon which you legally purchased a license to the Software. Notwithstanding the foregoing, you may not transfer education, pre-release, or not-for-resale copies of the Software.

4. PRE-RELEASE AND EVALUATION PRODUCT ADDITIONAL TERMS

If the product you have received with this license is pre-commercial release or beta Software ("Pre-release Software"), then this Section applies. In addition, this section applies to all evaluation and/or demonstration copies of Altova software ("Evaluation Software") and continues in effect until you purchase a license. To the extent that any provision in this section is in conflict with any other term or condition in this Agreement, this section shall supersede such other term(s) and condition(s) with respect to the Pre-release and/or Evaluation Software, but only to the extent necessary to resolve the conflict. You acknowledge that the Pre-release Software is a pre-release version, does not represent final product from Altova, and may contain bugs, errors and other problems that could cause system or other failures and data loss. CONSEQUENTLY, THE PRE-RELEASE AND/OR EVALUATION SOFTWARE IS PROVIDED TO YOU **"AS-IS" WITH NO WARRANTIES FOR USE OR PERFORMANCE,** AND ALTOVA DISCLAIMS ANY WARRANTY OR LIABILITY OBLIGATIONS TO YOU OF ANY KIND, WHETHER EXPRESS OR IMPLIED. WHERE LEGALLY LIABILITY CANNOT BE EXCLUDED FOR PRE-RELEASE AND/OR EVALUATION SOFTWARE, BUT IT MAY BE LIMITED, ALTOVA'S LIABILITY AND THAT OF ITS SUPPLIERS SHALL BE LIMITED TO THE SUM OF FIFTY DOLLARS (USD $50) IN TOTAL. If the Evaluation Software has a time-out feature, then the software will cease operation after the conclusion of the designated evaluation period. Upon such expiration date, your license will expire unless otherwise extended. Your license to use any output created with the Evaluation Software that contains generated program code (including Unrestricted Source Code) such as Java, C++, C, VB.NET or XSLT and associated project files and build scripts as well as generated XML, XML Schemas, documentation, UML diagrams, and database structures terminates automatically upon the expiration of the designated evaluation period but the license to use such output is revived upon your purchase of a license for the Software that you evaluated and used to create such output. Access to any files created with the Evaluation Software is entirely at your risk. You acknowledge that Altova has not promised or guaranteed to you that Pre-release Software will be announced or made available to anyone in the future, that Altova has no express or implied obligation to you to announce or introduce the Pre-release Software, and that Altova may not introduce a product similar to or compatible with the Pre-release Software. Accordingly, you acknowledge that any research or development that you perform regarding the Pre-release Software or any product associated with the Pre-release Software is done entirely at your own risk. During the term of this Agreement, if requested by Altova, you will provide feedback to Altova regarding testing and use of the Pre-release Software, including error or bug reports. If you have been provided the Pre-release Software pursuant to a separate written agreement, your use of the Software is governed by such agreement. You may not sublicense, lease, loan, rent, distribute or otherwise transfer the Pre-release Software. Upon receipt of a later unreleased version of the Pre-release Software or release by Altova of a publicly released commercial version of the Software, whether as a stand-alone product or as part of a larger product, you agree to return or destroy all earlier Pre-release Software received from Altova and to abide by the terms of the license agreement for any such later versions of the Pre-release Software.

5. LIMITED WARRANTY AND LIMITATION OF LIABILITY

(a) Limited Warranty and Customer Remedies. Altova warrants to the person or entity

that first purchases a license for use of the Software pursuant to the terms of this Agreement that (i) the Software will perform substantially in accordance with any accompanying Documentation for a period of ninety (90) days from the date of receipt, and (ii) any support services provided by Altova shall be substantially as described in Section 6 of this agreement. Some states and jurisdictions do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you. To the extent allowed by applicable law, implied warranties on the Software, if any, are limited to ninety (90) days. Altova's and its suppliers' entire liability and your exclusive remedy shall be, at Altova's option, either (i) return of the price paid, if any, or (ii) repair or replacement of the Software that does not meet Altova's Limited Warranty and which is returned to Altova with a copy of your receipt. This Limited Warranty is void if failure of the Software has resulted from accident, abuse, misapplication, abnormal use, Trojan horse, virus, or any other malicious external code. Any replacement Software will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer. This limited warranty does not apply to Evaluation and/or Pre-release Software.

(b) No Other Warranties and Disclaimer. THE FOREGOING LIMITED WARRANTY AND REMEDIES STATE THE SOLE AND EXCLUSIVE REMEDIES FOR ALTOVA OR ITS SUPPLIER'S BREACH OF WARRANTY. ALTOVA AND ITS SUPPLIERS DO NOT AND CANNOT WARRANT THE PERFORMANCE OR RESULTS YOU MAY OBTAIN BY USING THE SOFTWARE. EXCEPT FOR THE FOREGOING LIMITED WARRANTY, AND FOR ANY WARRANTY, CONDITION, REPRESENTATION OR TERM TO THE EXTENT WHICH THE SAME CANNOT OR MAY NOT BE EXCLUDED OR LIMITED BY LAW APPLICABLE TO YOU IN YOUR JURISDICTION, ALTOVA AND ITS SUPPLIERS MAKE NO WARRANTIES, CONDITIONS, REPRESENTATIONS OR TERMS, EXPRESS OR IMPLIED, WHETHER BY STATUTE, COMMON LAW, CUSTOM, USAGE OR OTHERWISE AS TO ANY OTHER MATTERS. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, ALTOVA AND ITS SUPPLIERS DISCLAIM ALL OTHER WARRANTIES AND CONDITIONS, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, SATISFACTORY QUALITY, INFORMATIONAL CONTENT OR ACCURACY, QUIET ENJOYMENT, TITLE AND NON-INFRINGEMENT, WITH REGARD TO THE SOFTWARE, AND THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES. THIS LIMITED WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHERS, WHICH VARY FROM STATE/JURISDICTION TO STATE/JURISDICTION.

(c) Limitation of Liability. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW EVEN IF A REMEDY FAILS ITS ESSENTIAL PURPOSE, IN NO EVENT SHALL ALTOVA OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, DIRECT, INDIRECT OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR ANY OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THE SOFTWARE OR THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, EVEN IF ALTOVA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN ANY CASE, ALTOVA'S ENTIRE LIABILITY UNDER ANY PROVISION OF THIS AGREEMENT SHALL BE LIMITED TO THE AMOUNT ACTUALLY PAID BY YOU FOR THE SOFTWARE PRODUCT. Because some states and jurisdictions do not allow the exclusion or limitation of liability, the above limitation may not apply to you. In such states and jurisdictions, Altova's liability shall be limited to the greatest extent permitted by law and the limitations or exclusions of warranties and liability contained herein do not prejudice applicable statutory consumer rights of person acquiring goods otherwise than in the course of business. The disclaimer and limited liability above are fundamental to this Agreement between Altova and you.

(d) Infringement Claims. Altova will indemnify and hold you harmless and will defend or settle any claim, suit or proceeding brought against you by a third party that is based upon a

claim that the content contained in the Software infringes a copyright or violates an intellectual or proprietary right protected by United States or European Union law ("Claim"), but only to the extent the Claim arises directly out of the use of the Software and subject to the limitations set forth in Section 5 of this Agreement except as otherwise expressly provided. You must notify Altova in writing of any Claim within ten (10) business days after you first receive notice of the Claim, and you shall provide to Altova at no cost such assistance and cooperation as Altova may reasonably request from time to time in connection with the defense of the Claim. Altova shall have sole control over any Claim (including, without limitation, the selection of counsel and the right to settle on your behalf on any terms Altova deems desirable in the sole exercise of its discretion). You may, at your sole cost, retain separate counsel and participate in the defense or settlement negotiations. Altova shall pay actual damages, costs, and attorney fees awarded against you (or payable by you pursuant to a settlement agreement) in connection with a Claim to the extent such direct damages and costs are not reimbursed to you by insurance or a third party, to an aggregate maximum equal to the purchase price of the Software. If the Software or its use becomes the subject of a Claim or its use is enjoined, or if in the opinion of Altova's legal counsel the Software is likely to become the subject of a Claim, Altova shall attempt to resolve the Claim by using commercially reasonable efforts to modify the Software or obtain a license to continue using the Software. If in the opinion of Altova's legal counsel the Claim, the injunction or potential Claim cannot be resolved through reasonable modification or licensing, Altova, at its own election, may terminate this Agreement without penalty, and will refund to you on a pro rata basis any fees paid in advance by you to Altova. THE FOREGOING CONSTITUTES ALTOVA'S SOLE AND EXCLUSIVE LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT. This indemnity does not apply to situations where the alleged infringement, whether patent or otherwise, is the result of a combination of the Altova software and additional elements supplied by you.

6. SUPPORT AND MAINTENANCE

Altova offers multiple optional "Support & Maintenance Package(s)" ("SMP") for the version of Software product edition that you have licensed, which you may elect to purchase in addition to your Software license. The Support Period, hereinafter defined, covered by such SMP shall be delineated at such time as you elect to purchase a SMP. Your rights with respect to support and maintenance as well as your upgrade eligibility depend on your decision to purchase SMP and the level of SMP that you have purchased:

(a) If you have not purchased SMP, you will receive the Software AS IS and will not receive any maintenance releases or updates. However, Altova, at its option and in its sole discretion on a case by case basis, may decide to offer maintenance releases to you as a courtesy, but these maintenance releases will not include any new features in excess of the feature set at the time of your purchase of the Software. In addition, Altova will provide free technical support to you for thirty (30) days after the date of your purchase (the "Support Period" for the purposes of this paragraph 6(a), and Altova, in its sole discretion on a case by case basis, may also provide free courtesy technical support during your thirty (30) day evaluation period. Technical support is provided via a Web-based support form only, and there is no guaranteed response time.

(b) If you have purchased SMP, then solely for the duration of its delineated Support Period, **you are eligible to receive the version of the Software edition** that you have licensed and all maintenance releases and updates for that edition that are released during your Support Period. For the duration of your SMP's Support Period, you will also be eligible to receive upgrades to the comparable edition of the next version of the Software that succeeds the Software edition that you have licensed for applicable upgrades released during your Support Period. The specific upgrade edition that you are eligible to receive based on your Support Period is further detailed in the SMP that you have purchased. Software that is introduced as separate product is not included in SMP. Maintenance releases, updates and upgrades may or may not include additional features. In

 © 2014 Altova GmbH

addition, Altova will provide Priority Technical Support to you for the duration of the Support Period. Priority Technical Support is provided via a Web-based support form only and Altova will make commercially reasonable efforts to respond via e-mail to all requests within forty-eight (48) hours during Altova's business hours (MO-FR, 8am UTC – 10pm UTC, Austrian and US holidays excluded) and to make reasonable efforts to provide work-arounds to errors reported in the Software.

During the Support Period you may also report any Software problem or error to Altova. If Altova determines that a reported reproducible material error in the Software exists and significantly impairs the usability and utility of the Software, Altova agrees to use reasonable commercial efforts to correct or provide a usable work-around solution in an upcoming maintenance release or update, which is made available at certain times at Altova's sole discretion.

If Altova, in its discretion, requests written verification of an error or malfunction discovered by you or requests supporting example files that exhibit the Software problem, you shall promptly provide such verification or files, by email, telecopy, or overnight mail, setting forth in reasonable detail the respects in which the Software fails to perform. You shall use reasonable efforts to cooperate in diagnosis or study of errors. Altova may include error corrections in maintenance releases, updates, or new major releases of the Software. Altova is not obligated to fix errors that are immaterial. Immaterial errors are those that do not significantly impact use of the Software as determined by Altova in its sole discretion. Whether or not you have purchased the Support & Maintenance Package, technical support only covers issues or questions resulting directly out of the operation of the Software and Altova will not provide you with generic consultation, assistance, or advice under any circumstances.

Updating Software may require the updating of software not covered by this Agreement before installation. Updates of the operating system and application software not specifically covered by this Agreement are your responsibility and will not be provided by Altova under this Agreement. Altova's obligations under this Section 6 are contingent upon your proper use of the Software and your compliance with the terms and conditions of this Agreement at all times. Altova shall be under no obligation to provide the above technical support if, in Altova's opinion, the Software has failed due to the following conditions: (i) damage caused by the relocation of the Software to another location or CPU; (ii) alterations, modifications or attempts to change the Software without Altova's written approval; (iii) causes external to the Software, such as natural disasters, the failure or fluctuation of electrical power, or computer equipment failure; (iv) your failure to maintain the Software at Altova's specified release level; or (v) use of the Software with other software without Altova's prior written approval. It will be your sole responsibility to: (i) comply with all Altova-specified operating and troubleshooting procedures and then notify Altova immediately of Software malfunction and provide Altova with complete information thereof; (ii) provide for the security of your confidential information; (iii) establish and maintain backup systems and procedures necessary to reconstruct lost or altered files, data or programs.

7. SOFTWARE ACTIVATION, UPDATES AND LICENSE METERING

(a) License Metering. The Software includes a built-in license metering module that is designed to assist you with monitoring license compliance in small local networks. The metering module attempts to communicate with other machines on your local area network. You permit Altova to use your internal network for license monitoring for this purpose. This license metering module may be used to assist with your license compliance but should not be the sole method. Should your firewall settings block said communications, you must deploy an accurate means of monitoring usage by the end user and preventing users from using the Software more than the Permitted Number.

(b) License Compliance Monitoring. You are required to utilize a process or tool to ensure that the Permitted Number is not exceeded. Without prejudice or waiver of any potential violations of the Agreement, Altova may provide you with additional compliance tools should you be unable to accurately account for license usage within your organization. If provided with such a tool by Altova, you (a) are required to use it in order to comply with the terms of this Agreement and (b) permit Altova to use your internal network for license monitoring and metering and to generate compliance reports that are communicated to Altova from time to time.

(c) Software Activation. **The Software may use your internal network and Internet connection for the purpose of transmitting license-related data at the time of installation, registration, use, or update to an Altova Master License Server and validating the authenticity of the license-related data in order to protect Altova against unlicensed or illegal use of the Software and to improve customer service. Activation is based on the exchange of license related data between your computer and the Altova Master License Server. You agree that Altova may use these measures and you agree to follow any applicable requirements. You further agree that use of license key codes that are not or were not generated by Altova and lawfully obtained from Altova, or an authorized reseller as part of an effort to activate or use the Software violates Altova's intellectual property rights as well as the terms of this Agreement. You agree that efforts to circumvent or disable Altova's copyright protection mechanisms, the license management mechanism, or the Altova Master License Server violate Altova's intellectual property rights as well as the terms of this Agreement. Altova expressly reserves the rights to seek all available legal and equitable remedies to prevent such actions and to recover lost profits, damages and costs.**

(d) LiveUpdate. Altova provides a new LiveUpdate notification service to you, which is free of charge. Altova may use your internal network and Internet connection for the purpose of transmitting license-related data to an Altova-operated LiveUpdate server to validate your license at appropriate intervals and determine if there is any update available for you.

(e) Use of Data. The terms and conditions of the Privacy Policy are set out in full at http://www.altova.com/privacy and are incorporated by reference into this Agreement. By your acceptance of the terms of this Agreement and/or use of the Software, you authorize the collection, use and disclosure of information collected by Altova for the purposes provided for in this Agreement and/or the Privacy Policy. Altova has the right in its sole discretion to amend this provision of the Agreement and/or Privacy Policy at any time. You are encouraged to review the terms of the Privacy Policy as posted on the Altova Web site from time to time.

(f) Audit Rights. You agree that Altova may audit your use of the Software for compliance with the terms of this Agreement at any time, upon reasonable notice. In the event that such audit reveals any use of the Software by you other than in full compliance with the terms of this Agreement, you shall reimburse Altova for all reasonable expenses related to such audit in addition to any other liabilities you may incur as a result of such non-compliance.

(g) Notice to European Users. Please note that the information as described in paragraph 7(d) above may be transferred outside of the European Economic Area, for purposes of processing, analysis, and review, by Altova, Inc., a company located in Beverly, Massachusetts, U.S.A., or its subsidiaries or Altova's subsidiaries or divisions, or authorized partners, located worldwide. You are advised that the United States uses a sectoral model of privacy protection that relies on a mix of legislation, governmental regulation, and self-regulation. You are further advised that the Council of the European Union has found that this model does not provide "adequate" privacy protections as contemplated by Article 25 of the European Union's Data Directive. (Directive 95/46/EC, 1995 O.J. (L 281) 31). Article 26 of the European Union's Data Directive

allows for transfer of personal data from the European Union to a third country if the individual has unambiguously given his consent to the transfer of personal information, regardless of the third country's level of protection. By agreeing to this Agreement, you consent to the transfer of all such information to the United States and the processing of that information as described in this Agreement and the Privacy Policy.

8. TERM AND TERMINATION

This Agreement may be terminated (a) by your giving Altova written notice of termination; (b) by Altova, at its option, giving you written notice of termination if you commit a breach of this Agreement and fail to cure such breach within ten (10) days after notice from Altova; or (c) at the request of an authorized Altova reseller in the event that you fail to make your license payment or other monies due and payable. In addition the Agreement governing your use of a previous version of the Software that you have upgraded or updated is terminated upon your acceptance of the terms and conditions of the Agreement accompanying such upgrade or update. Upon any termination of the Agreement, you must cease all use of the Software that this Agreement governs, destroy all copies then in your possession or control and take such other actions as Altova may reasonably request to ensure that no copies of the Software remain in your possession or control. The terms and conditions set forth in Sections 1(h), 1(i), 1(j), 1(k), 1(l), 2, 5, 7, 9, 10, 11, and 11 survive termination as applicable.

9. RESTRICTED RIGHTS NOTICE AND EXPORT RESTRICTIONS

The Software was developed entirely at private expense and is commercial computer software provided with **RESTRICTED RIGHTS**. Use, duplication or disclosure by the U.S. Government or a U.S. Government contractor or subcontractor is subject to the restrictions set forth in this Agreement and as provided in FAR 12.211 and 12.212 (48 C.F.R. §12.211 and 12.212) or DFARS 227. 7202 (48 C.F.R. §227-7202) as applicable. Consistent with the above as applicable, Commercial Computer Software and Commercial Computer Documentation licensed to U.S. government end users only as commercial items and only with those rights as are granted to all other end users under the terms and conditions set forth in this Agreement. Manufacturer is Altova GmbH, Rudolfsplatz 13a/9, A-1010 Vienna, Austria/EU. You may not use or otherwise export or re-export the Software or Documentation except as authorized by United States law and the laws of the jurisdiction in which the Software was obtained. In particular, but without limitation, the Software or Documentation may not be exported or re-exported (i) into (or to a national or resident of) any U.S. embargoed country or (ii) to anyone on the U.S. Treasury Department's list of Specially Designated Nationals or the U.S. Department of Commerce's Table of Denial Orders. By using the Software, you represent and warrant that you are not located in, under control of, or a national or resident of any such country or on any such list.

10. U.S. GOVERNMENT ENTITIES

Notwithstanding the foregoing, if you are an agency, instrumentality or department of the federal government of the United States, then this Agreement shall be governed in accordance with the laws of the United States of America, and in the absence of applicable federal law, the laws of the Commonwealth of Massachusetts will apply. Further, and notwithstanding anything to the contrary in this Agreement (including but not limited to Section 5 (Indemnification)), all claims, demands, complaints and disputes will be subject to the Contract Disputes Act (41 U.S.C. §§7101 *et seq.*), the Tucker Act (28 U.S.C. §1346(a) and §1491), or the Federal Tort Claims Act (28 U.S.C. §§1346(b), 2401-2402, 2671-2672, 2674-2680), FAR 1.601(a) and 43.102 (Contract Modifications); FAR 12.302(b), as applicable, or other applicable governing authority. For the avoidance of doubt, if you are an agency, instrumentality, or department of the federal, state or local government of the U.S. or a U.S. public and accredited educational institution, then your

indemnification obligations are only applicable to the extent they would not cause you to violate any applicable law (e.g., the Anti-Deficiency Act), and you have any legally required authorization or authorizing statute.

11. THIRD PARTY SOFTWARE

The Software may contain third party software which requires notices and/or additional terms and conditions. Such required third party software notices and/or additional terms and conditions are located at our Website at http://www.altova.com/legal_3rdparty.html and are made a part of and incorporated by reference into this Agreement. By accepting this Agreement, you are also accepting the additional terms and conditions, if any, set forth therein.

12. JURISDICTION, CHOICE OF LAW, AND VENUE

If you are located in the European Union and are using the Software in the European Union and not in the United States, then this Agreement will be governed by and construed in accordance with the laws of the Republic of Austria (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Software resides in the Handelsgericht, Wien (Commercial Court, Vienna) and you further agree and expressly consent to the exercise of personal jurisdiction in the Handelsgericht, Wien (Commercial Court, Vienna) in connection with any such dispute or claim.

If you are located in the United States or are using the Software in the United States then this Agreement will be governed by and construed in accordance with the laws of the Commonwealth of Massachusetts, USA (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Software resides in the federal or state courts of the Commonwealth of Massachusetts and you further agree and expressly consent to the exercise of personal jurisdiction in the federal or state courts of the Commonwealth of Massachusetts in connection with any such dispute or claim.

If you are located outside of the European Union or the United States and are not using the Software in the United States, then this Agreement will be governed by and construed in accordance with the laws of the Republic of Austria (excluding its conflict of laws principles and the U.N. Convention on Contracts for the International Sale of Goods) and you expressly agree that exclusive jurisdiction for any claim or dispute with Altova or relating in any way to your use of the Software resides in the Handelsgericht, Wien (Commercial Court, Vienna) and you further agree and expressly consent to the exercise of personal jurisdiction in the Handelsgericht Wien (Commercial Court, Vienna) in connection with any such dispute or claim. This Agreement will not be governed by the conflict of law rules of any jurisdiction or the United Nations Convention on Contracts for the International Sale of Goods, the application of which is expressly excluded.

13. TRANSLATIONS

Where Altova has provided you with a foreign translation of the English language version, you agree that the translation is provided for your convenience only and that the English language version will control. If there is any contradiction between the English language version and a translation, then the English language version shall take precedence.

14. GENERAL PROVISIONS

This Agreement contains the entire agreement and understanding of the parties with respect to

 © 2014 Altova GmbH

the subject matter hereof, and supersedes all prior written and oral understandings of the parties with respect to the subject matter hereof. Any notice or other communication given under this Agreement shall be in writing and shall have been properly given by either of us to the other if sent by certified or registered mail, return receipt requested, or by overnight courier to the address shown on Altova's Web site for Altova and the address shown in Altova's records for you, or such other address as the parties may designate by notice given in the manner set forth above. This Agreement will bind and inure to the benefit of the parties and our respective heirs, personal and legal representatives, affiliates, successors and permitted assigns. The failure of either of us at any time to require performance of any provision hereof shall in no manner affect such party's right at a later time to enforce the same or any other term of this Agreement. This Agreement may be amended only by a document in writing signed by both of us. In the event of a breach or threatened breach of this Agreement by either party, the other shall have all applicable equitable as well as legal remedies. Each party is duly authorized and empowered to enter into and perform this Agreement. If, for any reason, any provision of this Agreement is held invalid or otherwise unenforceable, such invalidity or unenforceability shall not affect the remainder of this Agreement, and this Agreement shall continue in full force and effect to the fullest extent allowed by law. The parties knowingly and expressly consent to the foregoing terms and conditions.

Last updated: 2013-10-17

Índice

(

(default),
multi input / output components, 63

.

.NET,
diferencias entre MapForce y MapForce para Visual Studio, 936
.NET 2002/2003, 1028

/

/-,
parámetros en tiempo de ejecución (línea de comandos), 659

<

<dynamic>,
file input / output, 63

0

0D y 0A,
reemplazar caracteres especiales, 639

2

2005R3,
modo de compatibilidad, 1075
2014, 1101

8

8601,
ISO (formato fecha y hora), 507

9

997,
confirmación funcional de X12, 498
999,
confirmación de implementación de X12, 500

A

Abrir,
archivos de MapForce en VS .NET, 933
script SQL, 353
abs, 881, 894
Absolutas,
rutas de acceso (ventajas y desventajas), 740
Access,
asignar datos MS Access a IBM DB2, 333
compatibilidad con MS Access, 230
Acción, 281
acciones de tabla, 281
acciones de tabla y transacciones, 282
eliminar datos de tabla antes de la acción, 281
icono de acciones de tabla, 254
omisión (acción de tabla), 267
Acción de base de datos,
actualización, 254
actualización y eliminación de secundarios, 261
eliminación, 264
inserción, 246
omisión, 267
Acción de tabla,
insertar restantes, 254
Acciones de tabla,
comparar campos, 282
Acerca de MapForce, 1016
acos, 881
Activar el software, 1015

Active, 1173
ActiveDocument, 1131, 1167
Actualización,
acción de base de datos, 254
Actualizar,
base de datos, 349
y eliminar secundario, 261
add, 843
Adjust-to-Timezone, 891
Advertencia,
validación, 114
Agregado,
función de agregado (usar plantillas con nombre), 802
funciones, 641
Agregar, 981
archivo de recursos globales, 690
biblioteca personal, 805
duplicado delante/detrás, 981
funciones definidas por el usuario, 757
relaciones a tablas, 303
ubicación del esquema, 981
Agregar/quitar tablas,
en el componente de base de datos, 238
Agrupar,
bucles en iteración, 407
Ajustar tamaño,
componente "ajuste óptimo", 28
Ajuste óptimo,
doble clic en el icono de ajuste del tamaño, 28
Alineación automática,
líneas de guía, 29
Alinear,
componentes en la ventana de asignación, 29
Alinear automáticamente,
componentes de la asignación, 86
Altova XML,
DoTransform.bat, 45
Ampliar,
parámetros de las funciones, 189
Análisis de impacto,
hoja de estilos SPS, 138
Analizador,
integrado en los productos de Altova, 1355
Analizador XML,
información sobre, 1355
Analizador XML de Altova,
información sobre, 1355
Anidadas,
funciones definidas por el usuario, 786
Anotación,
conector, 989
ANSI X12,
como destino de la asignación, 507
reglas de finalización automática, 512
reglas de validación de destino, 512
ANT,
compilar servicio web Java, 911
script, 1026
Any,
xs:any, 216
anyURI,
funciones, 888
API,
accessing, 1280
conectarse a través de la API ODBC, 270
documentation, 1096
overview, 1097
API de MapForce, 1096
overview, 1097
API de MapForce Type Library, 1128
API ODBC,
conectarse a través de, 270
Application, 1129
for Documents, 1167
Application object, 1098, 1130
Application-level,
integration of MapForce, 1227
AppOutputLine, 1137
AppOutputLines, 1142
AppOutputLineSymbol, 1144
Archivo, 967
agregar configuración de recurso, 690
archivo AAR Axis2, 911
catálogo, 127
codificación, 414, 425
definir recurso global, 690
Archivo AAR,
archivo axis para Axis2, 911
Archivo de biblioteca,
MFF, 805
Archivo de definición,
globalresource.xml, 688
Archivo de destino,
varios archivos de destino a partir de un solo archivo de origen, 729
Archivo de origen,

© 2014 Altova GmbH

Archivo de origen,
dividir entre varios archivos de destino, 729
Archivo de solución, 1028
Archivo MFF,
configurar, 807
Archivos,
nombres de archivo dinámicos como archivos de entrada, 727
varios archivos a partir de un archivo Excel, 734
varios archivos a partir de una BD, 729, 732
Archivos delimitados,
CSV, 404
Archivos planos,
asignación, 403
Archivos XML,
a partir de un solo XML de origen, 729
generar a partir de archivos Excel, 734
generar a partir de una BD, 732
Arrastrar y colocar,
generar instrucción SQL, 351
Asignación,
archivos CSV, 404
archivos de formato plano, 403
archivos de texto, 418
basada en el destino, 163
basada en el origen, contenido mixto, 157
basada en el tipo, 165
compilar para MapForce, 117
conector, 150
definir elementos asignables Excel 2007, 548
documentar, 132
elementos del esquema, 34
elementos secundarios, 103
estándar, 163
hojas de estilos SPS predefinidas para la documentación, 138
implementar en FlowForce Server, 118
libros, 548
nombre del esquema de destino, 45
número de conectors, 150
propiedades, 103
secuencia de procesamiento, 171
tutorial, 26
validación, 114
validar estructura, 114
ventana, alineación automática, 29
Asignación automática,
elementos secundarios, 103
Asignación basada en el destino, 156
Asignación de valores,
dimensiones de hipercubos, 599
pasar datos sin cambiarlos, 633
tabla de búsqueda, 628
tabla de búsqueda, propiedades, 637
XBRL, 599
Asignación en cadena,
mostrar componente final usando StyleVision, 176, 181
Asignaciones,
procesamiento automatizado, 22
Asignar,
a elemento Filas, 407
archivo SPS a componentes, 146
datos a bases de datos, 238
datos de BD - Java, 72
datos de secuencias de datos, 1044
datos de una BD de gran tamaño, 307
datos de una BD de gran tamaño (parte de la BD), 309
datos de una BD de gran tamaño (toda la BD), 308
datos XML (método genérico), 319
recurso global a componente, 693
XBRL, 579
y consultar datos XML, 329
asin, 881
Asistente para la conexión,
crear conexiones, 270
Asumir,
presencia de delimitadores, 425
atan, 881
atomización de nodos,
en la evaluación de XPath 2.0 y XQuery 1.0, 1312
ATTLIST,
URI de espacio de nombres del documento DTD, 202
Ausentes,
campos vacíos como, 425
tratar campos vacíos como, 414, 470, 476
auto-format, 866
Automática,
carga automática de bibliotecas, 807
Automatizar,
procesamiento, 22
auto-number, 269, 839
avg, 825
Ayuda, 1013
ver Ayuda en pantalla, 1014
Ayuda en pantalla,
búsqueda, 1014
contenido, 1014

© 2014 Altova GmbH

Ayuda en pantalla,
índice, 1014
A-Z,
componente de ordenación, 619

B

Background,
with gradient, 1189
Bajar,
elemento en el componente, 981
Barra de herramientas,
recurso global, 689
Basada en el destino,
asignación, 163
Basada en el origen,
asignación de contenido mixto, 157
frente a asignación estándar, 163
Basadas en el tipo,
conexiones, 165
Basado en línea,
dividir una vez, 463
división repetida, 456
Base,
tipo base (tipos derivados), 203
Base de datos,
acciones de tabla, 281
actualizar, 349
asignación completa, 308
asignación de datos desde un archivo XML, 238
asignación de datos entre BD y XBRL, 605
asignación parcial, 309
asignar datos - Java, 72
asignar datos de BD de gran tamaño, 307
asignar datos XML (método genérico), 319
asignar esquema, 319
asistente para la conexión, 342
cambiar de base de datos, 238
cambiar de BD, 981
como recurso global, 711
comparación de características, 291
compatibilidad, 230
conexiones JDBC, 277
crear relación, 303
deshacer, 349
generar varios archivos XML a partir de, 732
IBM DB2, 297
insertar, 72
MS Access, 291
MS SQL Server, 292
MySQL, 295
Oracle, 294
panel Consulta de la BD, 342
quitar nombre del esquema del código, 981
reemplazar caracteres especiales, 639
Sybase, 296
tiempo de espera de ejecución, 285
vista previa de las tablas, 324, 337
y varios componentes de origen, 981
Base de datos de gran tamaño,
importar, 308
Base package name,
for Java, 1164
Bases de datos,
asignación, 238
compatibles, 67, 230
XQuery, 230
base-uri, 888
BD,
filtrar registros de fecha, 661
ORDER BY, 313
BETWEEN,
SQL WHERE, 316
Biblioteca, 1091
agregar biblioteca personal, 805
agregar funciones XQuery, 801
agregar funciones XSLT, 796
agregar funciones XSLT 2.0, 800
buscar función en, 189
carga automática, 807
definir, 756
definir la interfaz del usuario del componente, 810
función, 189
generador de funciones, 269
importar funciones definidas por el usuario, 757
nueva C# 2007r3, 813
nueva C++ 2007r3, 813
nueva Java 2007r3, 813
personal, 189
referencia de funciones, 823
XPath 2, 189
Biblioteca personal,
agregar, 805
Bibliotecas,

Bibliotecas,
y funciones definidas por el usuario, 757
BOM,
marca de orden de bytes, 981
Bool,
salida si bool = false, 786
boolean, 828
Booleano/a,
comparar nodos de entrada, 646
Borrar,
funciones definidas por el usuario, 757
Bucle,
iterar en, 407
Bucles,
grupos y jerarquías, 192
Build.xml, 1026
BUILTIN,
nombre de componente, 659
Buscar,
función en biblioteca, 189
XSLT: panel Resultados, 973

C

C#, 1028
code generation, 1160
código, 1018
compilar código, 1028
configuración, 1075
crear servicio web C#, 916
enumeration, 1221
error handling, 1110
generar código, 1028
integrar código generado, 1040
integration of MapForce, 1234
options, 1185, 1187
C++, 1031
code generation, 1161
código, 1018
compilar código, 1031
configuración, 1075
configuraciones de compilación, 1031
enumeration, 1217, 1218
error handling, 1110
generación de EDIFACT / X12, 520
generar código, 1031
integrar código generado, 1042
options, 1185, 1186, 1187
Calcular una vez,
variable, 752
Call,
template, 796
Cambiar,
componente de base de datos, 238
configuración, recurso global, 695
Cambiar de,
configuración, recurso global, 695
Campo,
claves en archivos de texto, 410
delimitador, 414
longitud fija, 418
personalizado (CSV), 414
personalizado CSV, 425
Campos,
comparación en acciones de tabla, 282
Campos NULL,
campos vacíos, 414
Campos que faltan,
tratar como ausentes, 470, 476
Campos vacíos,
tratar como ausentes, 414, 425
capitalize, 883
Carácter,
de relleno, 418
Carácter de relleno,
eliminar, 418, 425
Caracteres especiales,
reemplazar, 639
Carpetas,
como recurso global, 697
disposición en carpetas (Consulta de la BD), 357
Catálogo,
archivo, 127
CDATA, 214
ceiling, 843
Centro de descarga de componentes,
del sitio web de Altova, 1016
Certificado,
firmas digitales, 225
Clases contenedoras,
compatibilidad de las versiones, 1002
Class ID,
in MapForce integration, 1243
Clave,

Clave,
campos de archivos de texto, 410
clave de ordenación, 619
Clave de evaluación,
para su software de Altova, 1015
Clave de ordenación,
componente de ordenación, 619
Close, 1159
project, 1193
code generation, 1169, 1171
C#, 1160
C++, 1161
enumerations, 1207, 1217, 1218, 1219, 1221, 1223
Java, 1161
options, 1165
options for, 1185, 1186, 1187, 1188, 1189
sample, 1099
XSLT, 1163
code-from-char, 852
Code-generation,
options for, 1135, 1190
codificación, 981
archivo, 414, 425
configuración del componente, 981
del documento XQuery, 1309
marca de orden de bytes, 981
opciones del panel Consulta de la BD, 365
Código,
de salida (línea de comandos), 121
ejemplo de generación, 1034
funciones inline y tamaño del código, 767
generación, 1023
integrar código de MapForce, 1037
quitar nombre del esquema del código, 981
SPL, 1076
tipos integrados, 1091
validar código EDI en ejecución, 507
Código de salida, 121
Códigos clave,
para su software de Altova, 1015
Columna,
encabezados (primera fila como), 548
columnname-to-index, 887
Coma,
archivos CSV, 404
COM-API,
documentation, 1096
Combinación de teclas,
acceso rápido, 1002
Combinar,
varios archivos de entrada, 724
Comentario de bloque, 354
Comentario de línea, 354
Comillas,
delimitadores de campo, 414
parámetros de la línea de comandos, 653
Comments,
Adding to target files, 212
Cómo puedo..., 612
Comodín,
WHERE SQL, 316
Comodines,
uso de comillas en la línea de comandos, 653
compare, 895
Compatibilidad,
bases de datos, 230
información sobre bases de datos, 291
Compatibilidad con BD, 67
compatibilidad con esquemas XML,
de XPath 2.0 y de los motores XQuery, 1312
Compatibilidad con Unicode,
de los productos de Altova, 1357
Compatibilidad del código,
v2005R3, 1075
Compilación,
configuraciones de compilación C++, 1031
Compilador,
configuración, 1075
Compilar,
código C#, 1028
código C++, 1031
Compilar en,
archivo de ejecución de MapForce, 117
archivo de ejecución de MapForce Server, 118
compl.,
grupo de nodos del complemento, 41
Compleja,
entrada compleja definida por el usuario, 775
función (inline), 767
función definida por el usuario, 774, 780
salida compleja definida por el usuario, 780
Complemento,
aplicar la naturaleza MapForce, 960
crear asignación / proyecto nuevo, 953
de MapForce para Eclipse, 940
generación de código, 954

Complemento,
generar código a mano, 955
generar código automáticamente, 956
importar carpeta de ejemplos, 951
MapForce (editor, vista), 950
Complemento de MapForce,
aplicar la naturaleza MapForce, 960
crear asignación / proyecto nuevo, 953
Editor, Vista, Perspectiva, 950
generación de código, 954
generar código a mano, 955
generar código automáticamente, 956
importar carpeta de ejemplos, 951
Complementos,
MapForce para VS .NET, 932
Component, 1146
Componente,
agregar/quitar tablas, 238
ajustar hasta su tamaño óptimo, 28
asignar esquema en SQL Server, 337
asignar recurso global, 693
asignar un archivo SPS, 146
asignar un esquema en DB2, 324
base de datos, 67
cambiar configuración, 981
cambiar de BD, 981
contenido del componente XBRL, 584
definir la interfaz del usuario, 810
dominio y dimensiones, 593
EDI, configuración, 507
elementos eliminados, 108
entrada - valor predeterminado, 655
excepción, 676
formato pretty-print en el archivo de salida, 981
habilitar procesamiento de entrada, 981
nombre (configuración del componente), 418, 981
opción Mostrar todos los conceptos, 584
opciones de codificación, 981
ordenación de datos, 619
separador de subcomponentes, 507
usar componentes FlexText en MapForce, 446
varias entradas/ salidas, 722
varias entradas/salidas, 724
varios archivos de entrada / salida, 62
Componente de entrada,
valor predeterminado, 655
Componentes,
secuencia de procesamiento, 171
Components, 1153
multi file input/output, 63
comportamiento durante la implementación,
de funciones XSLT 2.0, 1308
Comprar software de Altova, 1015
Comprobación,
nodos, 662
Comprobación de tipos, 418
Compute when,
variable, 752
Con nombre,
plantilla (espacios de nombres), 796
concat, 852
Concatenar,
filtros, 617
Condición,
dividir una vez, 439
IF-Else ampliable, 974
Condición múltiple,
conmutador, 441
Conectarse,
a través de la API ODBC, 270
Consulta de la BD (conectarse a la BD), 342
Conector,
copiar con Ctrl, 150
crear asignación con, 150
emergente, 150
nombre, 989
propiedades, 103
Conectores,
copia total, 165
Conexión, 270
asistente, 270
configuración, 989
eliminar, 103
JDBC, 277
mover conectores primario/secundario, 103
nativa, 270
propiedades, 103
Conexión automática,
elementos secundarios, 103, 150
Conexión rápida,
asistente para la conexión, 270, 342
Conexiones,
basadas en el tipo, 165
Conexiones JDBC, 277
Configuración,
actualizar para mensajes múltiples, 533

Configuración,
agregar a recurso global, 690
archivo de texto de longitud fija, 425
C++ y C#, 1075
cambiar de componente, 981
cambiar de, recurso global, 695
carácter de relleno, 425
conector, 989
copiar ya existente, 690
finalización automática (panel Consulta de la BD), 367
firma digital, 225
fuentes de texto (Consulta de la BD), 368
general, 365
llamada WSDL, 923
nombre del componente, 418, 981
vista de resultados (Consulta de la BD), 368
Configuración de clave,
acciones de tabla, 281
Configuración del componente,
nombre del componente, 418, 981
Configuración predeterminada,
dimensión (identificador y período), 593
Configurar,
archivo MFF, 807
configuración del Editor SQL, 365
Confirmación,
997 de X12, 498
999 de X12, 500
X12 997/999. Segmento TA1, 502
Confirmación de implementación,
999 de X12, 500
Confirmación funcional,
997 de X12, 498
Conmutador, 464
Connection, 1155
Conservar conexiones,
tras la eliminación, 103
Conservar datos,
cuando se usa una asignación de valores, 633
Consolidar datos,
unir archivos XML, 651
Constante,
como valor predeterminado, 655
Constructor,
XSLT2, 800
constructores,
xs:ENTITY, 889
Consulta,
de base de datos, 342
Consulta de la BD,
características del editor SQL, 353
convertir texto en comentario, 354
ejecutar SQL, 352
filtrar tablas, 359
finalización automática, 354
generar SQL, 351, 357, 362
guardar y abrir scripts SQL, 353
marcadores, 356
opciones de finalización automática, 367
opciones de fuentes de texto, 368
opciones de la vista de resultados, 368
opciones del menú contextual, 361
opciones: codificación, 365
regiones, 356
ventana SQL, 350
Consultar,
XML en DB2, 329
contains, 852
Contenedor,
mostrar / ocultar compartimento, 433
omitir resultados de FlexText, 469
Contenedoras,
clases, 1002
Contenido mixto,
asignación, 157
asignación basada en el origen, 157
ejemplo de asignación, 162
método de asignación, 155
Contexto,
invalidar, 666
prioridad, 189
prioritario, 648
Contexto primario,
variable, 752
Contexto prioritario,
definir, 648
Contraer,
expandir compartimento, 433
regiones, 356
Contraseña,
firma digital, 225
Contrato de licencia para el usuario final, 1359, 1363
Controlador,
asistente para la conexión, 342
JDBC, 981
Conversión,

Conversión,
funciones booleanas, 646
Convertir,
en esquema de destino, 981
Convertir rutas de acceso,
en rutas absolutas en el código generado, 740
convert-to-utc, 869
Copia total, 165
conectores, 165
método de asignación, 155
resolver / eliminar conectores, 165
y filtros, 165
Copiar,
conector disponible, 150
Core,
funciones de la biblioteca, 825
Cortar,
mover conectores primario/secundario, 103
cos, 881
coundt-substring, 883
Count, 825, 1142, 1153, 1167, 1183
count() en XPath 2.0,
ver fn:count(), 1312
count, sum, avg,
función de agregado, 641
CR / LF,
reemplazar en bases de datos, 639
Crear,
crear asignación / proyecto nuevo, 953
función, 37
funciones definidas por el usuario, 757
proyecto de servicio web, 908
regiones, 356
Crear conexiones,
asistente para la conexión, 270
create-guid, 868
Criterio de ordenación,
cambiar, 619
CSV,
archivo de entrada, 414
archivo de salida, 414
asignación, 404
campo personalizado, 414
crear jerarquías (claves), 410
guardar como, 470
opciones de archivo, 414
secuencias de datos, 112
tipos de datos de campo, 414
Cuadrícula,
alineación, 86
guía de alineación, 86
current, 900
current-date, 890
current-dateTime, 890
current-time, 890
char-from-code, 852

D

Datapoint, 1156
date,
xsd:date, 507
date-from-datetime, 869
Datetime,
asignación utilizando funciones datetime, 507
datetime-add, 869
datetime-diff, 869
datetime-from-date-and-time, 869
datetime-from-parts, 869
datetime-to-xlsx, 887
date-to-xlsx, 887
Datos,
compatibilidad con secuencia de datos, 1044
filtrado, 41
nombre de origen de datos (DSN), 342
Datos de tabla,
ordenar, 619
Datos secundarios,
omitirlos en tablas secundarias, 282
day-from-datetime, 869
day-from-duration, 869
db, 864
DB2,
asignar datos MS Access a IBM DB2, 333
asignar datos XML, 324
como componente de destino, 331
consultar datos XML, 329
punto y coma terminador, 277
vista previa del contenido XML, 324
deep-equal() en XPath 2.0,
ver fn:deep-equal(), 1312
default-collation, 890
Definida por el usuario,
entrada compleja, 775

Definida por el usuario,
función (inline o estándar), 767
salida compleja, 780
salida si bool = false, 786
Definidas por el usuario,
cambiar de tipo, 757
cambiar el tipo de la función, 757
eliminar, 757
funciones, 757
funciones (complejas), 774, 780
funciones (restricciones), 757
funciones anidadas, 786
funciones de búsqueda, 769
funciones estándar, 769
funciones y archivos MFF de MapForce, 757
importar/exportar, 757
Degradado,
fondo, 1075
degrees, 881
Delimitador,
asumir presencia, 425
campo, 414
Delimitadores,
campos vacíos como ausentes, 414
definidos personalmente, 507
no imprimibles, 507
Derivados,
usar / asignar tipos derivados, 203
Deshacer,
cambios en los datos de la BD, 349
Destino,
componente IBM DB2, 331
documentos EDI, 507
elemento raíz, 645
FlexText, 450
varios esquemas, 46
ver varios esquemas, 50
Destino único,
varios orígenes, 651
Destinos,
varios destinos y generación de código, 1034
Detección automática,
tipo de datos del parámetro, 316
Dimensión,
configuración predeterminada (identificador y período), 593
mostrar en el componente XBRL, 593
mover, 593
transformación de asignación de valores, 599
Dinámicos,
archivos de entrada en tiempo de ejecución, 724
compatibilidad con varios archivos, 722
nombres de archivo como parámetros de entrada, 727
Disposición,
explorador, 357
distinct-values, 848
Distribución,
de productos de software de Altova, 1359, 1360, 1362
divide, 843
divide-integer, 881
Dividir,
una vez, 439
Dividir una vez,
basado en línea, 463
flotante, 462
longitud fija, 461
predeterminado, 459
División repetida,
delimitado (la línea empieza con), 458
delimitado basado en línea, 456
delimitado flotante, 454
longitud fija, 453
predeterminada, 452
Dll,
configuración del compilador, 1075
Document, 900, 1158
closing, 1159
creating new, 1133, 1168
filename, 1165
on closing, 1159
on opening, 1130
opening, 1134, 1168
path and name of, 1160
path to, 1166
retrieving active document, 1167
save, 1166
save as, 1166
Documentación,
definir hojas de estilos SPS, 142
Documentación general,
hoja de estilos SPS, 138
Documentar,
asignaciones, 132
Document-level,
examples of integration of XMLSpy, 1233
integration of MapForce, 1229, 1230, 1231, 1232
integration of MapForceControl, 1228

© 2014 Altova GmbH

Documents, 1131, 1167
retrieving, 1168
total number in collection, 1167
document-uri, 888
DOM type,
enumerations for C++, 1217
for C++, 1186
Dominio,
mostrar en el componente XBRL, 593
DoTransform,
archivo por lotes, 112
archivo por lotes de AltovaXML, 45
DoTransform bat,
transformar XML, 50
DoTransform.bat,
ejecutar con RaptorXML Server, 24
DSN,
asistente para la conexión, 342
nombre de origen de datos, 342
DTD,
origen y destino, 202
Duplicado,
agregar delante/detrás, 981
Duplicar,
conector, 150
elementos de entrada, 57
duration-add, 869
duration-from-parts, 869
duration-subtract, 869

E

Eclipse,
aplicar la naturaleza MapForce, 960
complemento de MapForce, 940
crear asignación / proyecto nuevo, 953
generación de código, 954
generar código automáticamente, 956
generar código de asignación a mano, 955
importar ejemplos de MapForce, 951
iniciar complemento de MapForce, 947
instalar complemento de MapForce, 941, 950
EDI,
actualizar para mensajes múltiples, 533
como componente de destino, 507
configuración del componente, 507
IATA PADIS, 537
prioridad de los separadores, 507
reglas de finalización automática EDIFACT, 510
reglas de finalización automática X12, 512
reglas de validación EDIFACT, 510
reglas de validación X12, 512
SAP IDocs, 535
segmento X12 TA1, 502
separadores, 482
separar entradas combinadas, 530
validación, 114
validación de código en ejecución, 507
X12 HIPAA, 540
Edición, 973
edifact, 480, 866
actualizar archivo de configuración para mensajes múltiples, 533
asignar a XML, 483
como destino de la asignación, 507
ejemplo, 520
entradas combinadas, 523
generación de código (clase de C++), 520
personalización global, 517
personalización local, 518
personalizar, 515
personalizar (configuración), 515, 520
reglas de finalización automática, 510
reglas de validación de destino, 510
terminología, 482
Edition, 1131
Editor SQL,
características, 353
configuración general, 365
convertir texto en comentario, 354
crear regiones, 356
ejecutar SQL, 352
finalización automática, 354
inserar regiones, 356
insertar comentarios, 354
insertar marcadores, 356
marcadores, 356
margen de marcadores, 356
quitar comentarios, 354
quitar marcadores, 356
quitar regiones, 356
usar regiones, 356
Ejecución,
compilar en, 117

Ejecución,
tiempo de espera (BD), 285
Ejecutar,
archivo SQL, 352
instrucciones SQL por separado, 352
SQL, 352
Ejecutar script SQL,
desde el panel Resultados, 319
Ejemplo,
asignación recursiva definida por el usuario, 678
Ejemplos,
carpeta del tutorial, 26
El contenido contiene, 464
El contenido empieza con, 464
element-available, 900
Elemento,
convertir en destino, 981
duplicar, 57
esquema, asignación, 34
filas (iterar), 407
predeterminado XBRL, 584
recursivo en XML Schema, 678
Elemento de destino,
asignación de varios orígenes, 52
Elemento de error,
fault de servicio web, 918
llamada a servicio web, 923
Elemento raíz,
crear nuevo, 103
Elemento secundarios,
eliminar, 103
Elementos,
definir elementos asignables Excel 2007, 548
desaparecidos, 108
obligatorios, 150
separar entradas combinadas, 530
Elementos desaparecidos, 108
Elementos marcados,
elementos desaparecidos, 108
Elementos secundarios,
conexión automática, 103, 150
Eliminación,
acción de base de datos, 264
Eliminar,
conexiones, 103
conexiones de copia total, 165
datos en tablas secundarias, 282
eliminaciones, elementos desaparecidos, 108
función definida por el usuario, 757
tablas del componente, 238
Eliminar registros,
antes de la acción de tabla, 281
Eliminar secundario,
y actualizar, 261
empty, 883
En cadena,
asignación, generación de código, 181
en contexto,
parámetro, 825
Encabezados,
fila Excel como encabezados de columnas, 548
Encoding,
default for output files, 1187, 1188
ends-with, 895
entidades de caracteres,
en el resultado HTML de la transformación XSLT, 1304
Entrada,
archivo (CSV), 414
archivo (texto), 425
como parámetros de la línea de comandos, 653
comparar nodos booleanos, 646
instancia XML, 981
parámetro de la línea de comandos, 655
parámetros opcionales, 786
separar entradas combinadas, 530
valor predeterminado, 786
varios archivos, 724
Entradas combinadas, 523
dividir en varios elementos (X12), 530
Enumerations, 1206, 1207, 1208, 1209, 1210, 1211, 1213, 1214, 1215, 1217, 1218, 1219, 1220, 1221
for MapForce View, 1223
in MapForceControl, 1300
Envuelta,
firma digital, 225
equal, 841
equal-or-greater, 841
equal-or-less, 841
Error,
definir excepciones, 676
validación, 114
Error handling,
general description, 1110
ErrorLevel,
ejecución de la línea de comandos, 121
ErrorMarkers, 1169, 1171

escape-uri, 895
Espacio de nombres,
plantilla con nombre, 796
espacio de nombres predeterminado de las funciones,
en las hojas de estilos XSLT 2.0, 1306
espacio de nombres predeterminado para las funciones,
para expresiones XPath 2.0 y XQuery 1.0, 1312
espacios de nombres,
del documento XQuery, 1309
en las hojas de estilos XSLT 2.0, 1306
espacios en blanco en el documento XML,
gestión en el motor XSLT 2.0 de Altova, 1306
Especificar valor,
componente de entrada - vista previa, 655
Esquema,
agregar ubicación al archivo de salida, 981
archivo de catálogo, 127
asignación de XML, 201
asignar a una BD, 319
asignar en componente SQL Server, 337
asignar en un componente DB2, 324
BD como origen, 67
elementos recursivos, 678
generador de código, 1018
generar automáticamente a partir de un archivo XML, 28
insertar taxonomía XBRL, 580
objeto raíz, 342
quitar nombre del código generado, 981
registrado en IBM DB2, 324
registrado en SQL Server, 337
varios destinos, 46
varios esquemas de origen, 981
ver varios destinos, 50
Esquema XML,
asignar a una BD, 319
asignar en componente SQL Server, 337
asignar en un componente DB2, 324
generar automáticamente, 28
registrado en IBM DB2, 324
registrado en SQL Server, 337
Estándar,
asignación con secundarios, 163
asignación de contenido mixto, 163
biblioteca XSLT, 189
frente a asignación basada en el origen, 163
función definida por el usuario, 769
método de asignación, 155, 156
Events,
of Document, 1159
Events for Project, 1191
Examinar,
Consulta de la BD, 357
Excel,
ejemplo de asignación entre Excel y XBRL, 601
genera varios archivos XML a partir de, 734
hojas con formato previo, 576
nodos duplicados, 548
primera fila como encabezados de columna, 548
Excel 2007,
asignar a XML, 568
asignar datos de BD a Excel, 571
elementos asignables (definir), 548
libro (definir elementos), 548
OOXML Excel 2007, 547
Excepción,
elemento de error fault de servicio web, 918
iniciar, 676
memoria insuficiente, 1073
Existe,
comprobación de nodos, 662
exists, 846
usar función not-exist para asignar nodos que faltan, 664
exp, 881
Expandir,
contraer compartimento, 433
regiones, 356
Explícita,
relación (relaciones locales), 303
Explícito,
tipo de datos, 800
Explorador,
aplicar filtros, 359
filtrar, 359
generar SQL, 351
generar SQL para tablas, 361
Exportar,
funciones definidas por el usuario, 757
Expresión,
regular, 860
Expresiones regulares, 860
Extensiones de Altova,
funciones de gráficos (ver funciones de gráficos), 1317

F

fadians, 881
false, 889
Fault,
llamada a servicio web, 923
servicio web, 918
Fecha,
filtrar registros de fecha de la BD, 661
Fija,
archivos de longitud fija, 403
división repetida, 453
Fijo,
archivo de entrada, 425
archivo de salida, 425
campo personalizado, 425
crear nuevo (vacío), 425
Fila,
como encabezado de columna, 548
Filas,
asignación desde archivos de texto, 410
asignar datos a filas (archivos de texto), 407
de Excel, 734
iterar en elementos, 407
File,
multi file input / output, 63
File:,
(default), 63
item in component, 63
Filtrado,
asignar elementos primarios, 617
componente: consejos, 617
Filtrar,
complemento, 41
conector de copia total, 165
datos, 41
en el explorador de BD, 359
objetos de la BD, 359
registros por fecha, 661
unir archivos XML, 651
Filtro,
contexto prioritario, 617
no concatenar, 617
Finalización automática,
editor SQL, 354
find-substring, 883
Firma,
configuración, 225
Firma digital,
activar, 221
configuración, 225
Firmar,
firma digital, 221
FlexText,
como componente de destino, 450
Introducción, 432
origen de datos, 449
plantillas de FlexText en MapForce, 446
Tutorial, 436
FLF,
guardar como, 476
secuencias de datos, 112
floor, 843
Flotante,
dividir una vez, 462
división repetida, 454
FlowForce,
implementar en, 967
Flujo de trabajo,
iniciar, recurso global, 707
usando recursos globales, 701
Flujo de trabajo en la aplicación,
usando recursos globales, 701
fn:base-uri de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:collection de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:count() en XPath 2.0,
y los espacios en blanco, 1312
fn:current-date de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:current-dateTime de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:current-time de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:data de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:deep-equal() en XPath 2.0,
y los espacios en blanco, 1312
fn:id de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:idref de XPath 2.0,
compatibilidad con los motores de Altova, 1313

© 2014 Altova GmbH

fn:index-of de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:in-scope-prefixes de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:last() en XPath 2.0,
y los espacios en blanco, 1312
fn:lower-case de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:normalize-unicode de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:position() en XPath 2.0,
y los espacios en blanco, 1312
fn:resolve-uri de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:static-base-uri de XPath 2.0,
compatibilidad con los motores de Altova, 1313
fn:upper-case de XPath 2.0,
compatibilidad con los motores de Altova, 1313
Fondo,
degradado, 1075
format-dateTime, 828
format-guid-string, 883
format-number, 828, 900
Formato plano,
asignación, 418
From, 891
Fuentes,
varias fuentes y generación de código, 1034
FullName, 1160, 1194
Función, 839, 864, 866, 886, 994
agregado, 641
agregar, 189
agregar función XQuery personal, 801
agregar función XSLT 2.0 personal, 800
agregar función XSLT personal, 796
anidada definida por el usuario, 786
biblioteca, 189
biblioteca core, 825
biblioteca lang, 868
biblioteca xlsx, 887
biblioteca xpath2, 888
biblioteca xslt, 898
buscar en la biblioteca, 189
cambiar el tipo de una función definida por el usuario, 757
compleja (inline), 767
consulta, 189
conversión booleana, 646
datetime, 507
de búsqueda definida por el usuario, 769
de suma SUM, 802
definida por el usuario, 680, 757
definida por el usuario (cambiar de tipo), 757
definir, 756
entrada como parámetro, 653
estándar definida por el usuario, 769
exportar función definida por el usuario, 757
extensible, 189
generador, 269
generador visual, 757
IF-Else ampliable, 974
implementación, 812
importar función definida por el usuario, 757
inline, 767
personalizada, 189
restricciones en funciones definidas por el usuario, 757
set-null (para bases de datos), 311
xmlexists (consultas en DB2), 329
función count(),
de XPath 1.0, 1304
función definida por el usuario,
recursiva, 680
función last(),
de XPath 1.0, 1304
función position(),
de XPath 1.0, 1304
funciones,
asignación de, 37
Funciones XSLT 2.0, 1308
referencia, 823
XPath 2.0 y XQuery 1.0, 1312
Funciones de extensión .NET,
constructores, 1346
conversiones de tipos de datos (de .NET a XPath/XQuery), 1349
conversiones de tipos de datos (de XPath/XQuery a .NET), 1348
información general, 1344
métodos de instancia, campos de instancia, 1348
métodos estáticos, campos estáticos, 1347
para XSLT y XQuery, 1344
Funciones de extensión .NET para XSLT y XQuery,
ver Funciones de extensión .NET, 1344
Funciones de extensión de scripts MSXSL, 1350
Funciones de extensión Java,
archivos de clases definidos por el usuario, 1336
archivos JAR definidos por el usuario, 1339

© 2014 Altova GmbH

Funciones de extensión Java,
constructores, 1340
conversiones de tipos de datos (de Java a Xpath/XQuery), 1343
conversiones de tipos de datos (de XPath/XQuery a Java), 1342
información general, 1335
métodos de instancia, campos de instancia, 1342
métodos estáticos, campos estáticos, 1341
para XSLT y XQuery, 1335
Funciones de extensión Java para XSLT y XQuery,
ver Funciones de extensión Java, 1335
Funciones de extensión para XSLT y XQuery, 1317
Funciones de gráficos,
código de muestra, 1323
ejemplo, 1332
estructura de los datos del gráfico para, 1327
funciones externas,
en el documento XQuery, 1309
Funciones para bases de datos,
set-null, 311
Funciones utilizadas por, 138
Funciones XPath 2.0,
información general sobre, 1312
información sobre la implementación, 1312
ver fn: y el nombre de la función, 1312
Funciones XPath compatibles,
ver fn: y la función pertinente, 1313
Funciones XQuery 1.0,
información general sobre, 1312
información sobre la implementación, 1312
ver fn: y el nombre de la función, 1312
Funciones XSLT 2.0,
comportamiento durante la implementación, 1308
véase fn: y la función pertinente, 1308
function-available, 900

G

Generación de código,
clase de C++ (EDIFACT / X12), 520
convertir rutas de acceso en rutas absolutas, 740
de asignaciones en cadena, 181
nombre de salida predeterminado del archivo, 45
parámetros de entrada, 653
versión compatible con clases contenedoras, 1002
y ruta de acceso absoluta, 46
Generado,
nombre de salida del archivo, 45
Generador,
de funciones definidas por el usuario, 757
función, 269
Generador de código, 1018
Generador visual de funciones, 757
Generales,
opciones (panel Consulta de la BD), 365
Generar, 50
código, 1023
código a partir de un esquema, 1018
código C#, 1028
código C++, 1031
código Java, 1026
código y funciones inline, 767
ejemplo de código, 1034
esquema XML automáticamente, 28
firma digital, 221
SQL, 351
varios archivos Java de destino, 50
varios XSLT de destino, 50
generate-id, 900
gestión de los espacios en blanco,
y las funciones XPath 2.0, 1312
get-fileext, 837
get-folder, 837
GetRootDatapoint, 1212
Globalresource.xml,
definición de recursos, 688
Gradients,
in background, 1189
Gráficos de llamadas,
hoja de estilos SPS, 138
greater, 841
group-adjacent, 848
group-by, 848
group-ending-with, 848
group-starting-with, 848
Grupo de nodos,
complemento, 41
Grupos,
bucles y jerarquías, 192
Guardar,
datos de la ventana Resultados, 50
resultado XML, 112
scripts SQL, 353

© 2014 Altova GmbH

Guardar como CSV, 470
Guardar como FLF, 476
Guardar valor, 478
Guía,
de alineación, alineación automática de componentes, 86
guid, 269

H

Habilitar procesamiento de entrada,
optimización, 981
Health Level 7,
ejemplo, 607
Herramientas, 1002
HighlightMyConnections, 1174
HighlightMyConnectionsRecursively, 1174
HIPAA,
X12, 540
Hipercubo,
dominios/dimensiones XBRL, 593
HL7, 480
sub subcampo, 507
HL7 2.6 - 3.x,
ejemplo, 607
Hoja de cálculo,
a archivos XML de salida, 734
Hoja de estilos,
pestañas de StyleVision, 97
Hoja de estilos XSLT 2.0,
declaraciones de espacios de nombres en, 1306
Hojas de cálculo,
Excel y datos de MapForce, 576
Hojas de estilos,
definir hojas de estilos para la documentación, 142
Hojas Excel con formato previo,
enviar datos, 576
hour-from-datetime, 869
hour-from-duration, 869
HRESULT,
and error handling, 1110
HTML,
documentación de la asignación, 132
integration of MapForce, 1245
pestaña, de la ventana Asignación, 76
Stylevision Power Stylesheets, 144
HTML example,
of MapForceControl integration, 1243, 1244, 1245

I

IATA PADIS, 537
IBM DB2,
asignar datos desde la BD de origen, 333
asignar datos XML, 324
como componente de destino, 331
consultar datos XML, 329
incrustar un esquema XML en un componente, 324
información, 297
Icono,
obligatorio (resaltado), 150
Icono de entrada,
asignación, 150
Icono de salida,
asignación, 150
Icono del conector,
emergente, 150
Iconos,
de la ventana Mensajes (panel Consulta de la BD), 362
de la ventana Resultados (panel Consulta de la BD), 362
IDocs,
documento de intercambio SAP, 535
IF-Else,
ampliable, 974
Implementación,
función, 812
Implementar,
asignación de MapForce, 118
en FlowForce Server, 967
servicio web, 911
Implícito,
tipo de datos, 800
Importar,
funciones definidas por el usuario, 757
Imprimir,
delimitadores no imprimibles, 507
IN,
SQL WHERE, 316
Include,
XSLT, 796
XSLT 2.0, 800
index-to-columnname, 887
Información general, 1353

Información legal, 1359
Información sobre derechos de autor, 1359
Información técnica, 1353
Iniciar,
complemento para Eclipse, 947
Inline, 757
funciones y tamaño del código, 767
Inline / Estándar,
funciones definidas por el usuario, 767
Inserción,
acción de base de datos, 246
Insertar, 974
base de datos, 72
comentario de bloque, 354
comentario de línea, 354
comentarios, 354
componentes WHERE SQL, 313
llamada a servicio web, 923, 929
marcadores, 356
operador WHERE SQL, 316
regiones, 356
Insertar restantes,
acción de tabla, 254
después de la acción de tabla omisión, 267
Insertar todo,
acción de tabla, 282
ninguna acción de tabla después de Insertar todo, 282
InsertXMLFile, 1175
InsertXMLSchema, 1175
InsertXMLSchemaWithSample, 1176
Instalación,
carpeta Examples, 26
Instalar,
complemento para Eclipse, 941
Instancia,
elementos XBRL obligatorios, 601
instancia XML de entrada, 981
instancia XML de salida, 981
y el nombre de componente, 659
Instancia de texto,
componente FlexText, 449
Instancia XML,
entrada, 981
ruta de acceso absoluta, 46
salida, 981
Instrucción,
ejecutar instrucción SQL, 352
tiempo de espera de ejecución, 285
Integrado,
motor de ejecución, icono BUILTIN, 112
Integrar,
en C#, 1040
en C++, 1042
en Java, 1038
Integrar código de MapForce, 1037
Integrating,
MapForce in applications, 1226
Intercalación,
componente de ordenación, 619
configuración regional, 619
punto de código Unicode, 619
Intercalación de la configuración regional, 619
intercalaciones,
en el documento XQuery, 1309
en XPath 2.0, 1312
Intercambio,
documentos de intercambio (SAP IDoc), 535
Introducción,
a FlexText, 432
generador de código, 1019
Introducción a MapForce, 3, 12
Invalidar,
contexto, 666
iSeries,
deshabilitar opción de tiempo de espera, 324
is-not-null, 864
is-null, 864
ISO,
8601 (formato fecha y hora), 507
is-xsi-nil, 846
Item, 1142, 1153, 1168, 1183
Iteración,
contexto prioritario, 648
Iterar,
en archivos de texto, 407
IU,
definir la interfaz del usuario del componente, 810

J

Java, 1250
asignar datos de BD, 72
code generation, 1161
código, 1018

Java, 1250
configuración, 1075
crear servicio web Java, 911
generar código, 1026
generar varios destinos, 50
integrar código generado, 1038
options, 1164, 1185, 1189
varios destinos, 46
JavaScript,
error handling, 1110
JDBC,
convertir ADO y ODBC en, 1002
Jerarquía,
de archivos de texto, 410
nodo de FlexText, 467
tabla, 244
Jerarquías,
bucles y grupos, 192
JScript,
code-generation sample, 1099

L

La línea empieza con,
división repetida, 458
lang, 893, 898
funciones de la biblioteca, 868
last, 890, 898
last() en XPath 2.0,
ver fn:last(), 1312
leapyear, 869
left, 883
left-trim, 883
Lenguajes,
y compatibilidad con varios archivos y archivos dinámicos, 722
less, 841
Lib,
configuración del compilador, 1075
Library type,
enumerations for C++, 1218
for C++, 1186, 1187
Libro,
Excel 2007 (definir elementos), 548
Licencia, 1363
información sobre, 1359
Licencia del producto de software, 1363
Licencias,
para su software de Altova, 1015
LIKE,
SQL WHERE, 316
Línea de comandos, 121
código de salida, 121
configuración predeterminada y vista previa, 655
nombre de componente, 659
nombres de archivos de entrada dinámicos, 724
parámetro de entrada, 653
parámetros, 121
parámetros de entrada, 727
parámetros y valores de entrada, 653
Línea nueva,
en la instrucción WHERE SQL, 316
Local,
esquemas locales (archivos de catálogo), 127
relación de BD, 303
local-name, 893, 898
local-name-from-QName, 868
log, 881
log10, 881
logical-and, 841
logical-not, 841
logical-or, 841
logical-xor, 880
Logo,
display on startup, 1189
option for printing, 1189
Longitud fija,
asignación, 418
configuración de archivo de texto, 425
dividir una vez, 461
Lotes,
automatizar procesamiento, 22
lowercase, 883
lower-case, 895
Llamada, 923
configurar llamada WSDL, 923
elemento de error fault de servicio web, 923
Llamada a servicio web,
insertar, 923, 929
Llamar a servicios web, 922

M

main-mfd-filepath, 837
manespace-uri, 893
Mantener datos sin cambios,
pasar a través de una asignación de valores, 633
Manual del usuario,
ver Ayuda en pantalla, 1014
MapForce,
API, 1096
complemento para Eclipse, 940
complemento para VS .NET, 932
integration, 1226
introducción, 3, 12
motor, 112
parent, 1176
terminología, 14
MapForce API,
accessing, 1280
MapForce command table, 1266
MapForce integration,
and clients, 1226, 1228
example of, 1243, 1244, 1245
MapForce Server,
automatizar asignaciones, 22
MapForce view,
enumerations for, 1223
MapForceCommand,
in MapForceControl, 1282
MapForceCommands,
in MapForceControl, 1284
MapForceControl, 1285
documentation of, 1226
example of integration at application level, 1243, 1244, 1245
examples of integration at document level, 1233
integration at application level, 1227
integration at document level, 1228, 1229, 1230, 1231, 1232
integration using C#, 1234
integration using HTML, 1245
integration using Visual Basic, 1265
object reference, 1281
MapForceControlDocument, 1291
MapForceControlPlaceHolder, 1297
MapForceView, 1164, 1173
application, 1174
Mapping,
inserting XML file, 1175
inserting XML Schema file, 1175
MappingMap,
toTargetSchemaName, 45
Marca de orden de bytes,
configuración del componente, 981
Marcadores,
insertar, 356
margen de marcadores, 356
navegar, 356
quitar, 356
matches, 895
match-pattern, 883
max, 825, 881
Medición de licencias,
en los productos de Altova, 1361
Memoria,
insuficiente - excepciones, 1073
Memoria insuficiente, 1073
Mensajes,
actualizar archivo de configuración para mensajes múltiples, 533
iconos del panel Consulta de la BD, 362
ventana (panel Consulta de la BD), 362
Mensajes múltiples,
actualizar configuración EDI para, 533
Menú,
Archivo, 967
Ayuda, 1013
conexión, 989
Edición, 973
Función, 994
herramientas, 1002
Insertar, 974
Resultados, 997
Vista, 999
Menú contextual,
para tablas (panel Consulta de la BD), 361
Método,
reservar nombre, 1073
Métodos de asignación,
basada en el destino, 156
estándar, 156
estándar / mixto / copia total, 155
MFC support,
fo C++, 1187
mfd-filepath, 837

© 2014 Altova GmbH

MFF,
archivo de biblioteca, 805
archivo mff.xsd, 805
y funciones definidas por el usuario, 757
Microsoft® Office 2007,
asignaciones de datos, 547
millisecond-from-duration, 869
min, 825, 881
min, max,
función de agregado, 641
minOccurs/maxOccurs,
optimizar el procesamiento de entrada, 981
minute-from-datetime, 869
minute-from-duration, 869
MisDocumentos,
archivos de ejemplo, 26
missisecond-from-datetime, 869
Mixto,
asignación estándar, 163
Modo,
compatible con v2005R3, 1075
Modo de compatibilidad,
v2005R3, 1075
módulos de bibliotecas,
en el documento XQuery, 1309
modulus, 843
month-from-datetime, 869
month-from-duration, 869
Motor,
MapForce, 112
Motor de MapForce,
motor de ejecución integrado, 112
Motor XQuery 1.0,
información sobre, 1309
Motor XSLT 1.0,
limitaciones y comportamiento durante la implementación, 1304
Motor XSLT 1.0 de Altova,
limitaciones y comportamiento durante la implementación, 1304
Motor XSLT 2.0,
compatibilidad con versiones anteriores, 1306
información general sobre, 1306
información sobre, 1306
Motor XSLT 2.0 de Altova,
información general sobre, 1306
información sobre, 1306
Motores,
de los productos de Altova, 1356
Mover,
conectores primario/secundario, 103
dimensión en el componente XBRL, 593
MS Access,
información, 291
versiones 2000 y 2003, 230
MS Access 2000,
compatibilidad, 67
MS SQL Server,
compatibilidad, 67
información, 292
MS Visual Studio .NET,
complemento de MapForce, 932
MSXML 6.0,
biblioteca, 1002
msxsl:script, 1350
Multi file,
processing (tutorial), 63
Múltiple,
condiciones (conmutador), 464
división (condición conmutador), 441
Múltiples,
acciones de tabla, 254
destinos y generación de código, 1034
fuentes y generación de código, 1034
multiply, 843
MySQL,
información, 295

N

Name, 893, 898, 1133, 1165, 1196
Namespaces,
and wildcards (xs:any), 216
namespace-uri, 898
namespace-uri-from-QName, 868
Nativa,
conexión, 270
Navegar,
marcadores, 356
negative, 880
NewDocument, 1133, 1168
NewProject, 1134
nil,
xsi:nil, 208

Nillable,
no compatible, 1073
No imprimibles,
delimitadores, 507
node, 893
node-name, 888
Nodo, 467
asignar nodos que faltan, 664
comparar booleanos, 646
comprobación, 662
posición, 666
sumar varios nodos, 802
Nodos,
duplicados en Excel, 548
obligatorios, 150
nodos que contienen espacios en blanco en el documento XML,
y cómo se gestionan en el motor XSLT 1.0, 1304
Nodos que faltan,
asignar nodos que faltan, 664
Nombre,
conector, 989
Nombre de componente,
ejecución de la línea de comandos, 659
Nombre de la aplicación, 1075
Nombres de esquema,
quitar nombre del esquema del nombre de la tabla, 981
normalización de caracteres,
en el documento XQuery, 1309
normalize-space, 852
normalize-unicode, 895
not-equal, 841
not-exist,
asignar nodos que faltan, 664
not-exists, 846
now, 869
Nueva línea,
reemplazar caracteres especiales, 639
Nulo,
valores que pueden ser nulos, 208
Null,
función substitute-null, 311
funciones, 311
number, 828, 893
numeric, 880

O

Object,
reference, 1128
Object model,
overview, 1098
Object tree navigation,
Application, 1131, 1135
AppOutputLine, 1137, 1141
AppOutputLines, 1142
AppOutputLineSymbol, 1144, 1145
Component, 1146, 1150
Components, 1153
Document, 1159, 1165
Documents, 1167, 1168
ErrorMarker, 1169, 1170
ErrorMarkers, 1171
MapForceView, 1174, 1176
Mapping, 1178, 1181
Mappings, 1183
Options, 1185, 1189
Project, 1193, 1197
ProjectItem, 1200, 1205
Objeto raíz,
seleccionar, 342
Objetos globales,
de SPL, 1081
Obligatorios,
asignar elementos obligatorios para instancia XBRL, 601
nodos/elementos, 150
Obsolete, 1165
Ocultar,
mostrar compartimento, 433
ODBC,
conexión nativa, 270
Office Open XML,
Excel 2007, 547
Omisión,
acción de base de datos, 267
Omitir, 469
datos secundarios de entrada, 282
OnDocumentClosed, 1159
OnDocumentOpened, 1130
OnProjectClosed, 1192
OnProjectOpened, 1130

© 2014 Altova GmbH

OOXML,
asignar a XML, 568
asignar datos de BD a OOXML, 571
Excel 2007, 547
Opción,
opciones de archivos CSV, 414
Opción de configuración,
todos los conceptos XBRL, 584
Opcional,
parámetros de entrada, 786
Opciones,
finalización automática (panel Consulta de la BD), 367
fuentes de texto (Consulta de la BD), 368
generales, 365
vista de resultados (Consulta de la BD), 368
OpenDocument, 1134, 1168
OpenProject, 1134
Optimización,
habilitar procesamiento de entrada, 981
Options, 1135, 1185
for code generation, 1165, 1187, 1188
for Java, 1164
Oracle,
compatibilidad, 67
información, 294
Orden,
en el que se procesan los componentes, 171
Ordenación,
componente, 619
Ordenar,
datos en la ventana Resultados, 362
icono de columna en la ventana Resultados, 362
tablas (Consulta de la BD), 357
Ordenar datos,
componente de ordenación, 619
ORDER BY,
componente WHERE SQL, 313
Origen,
campos vacíos como ausentes, 414
Origen de datos,
FlexText, 449
Output directory,
for code-generation files, 1185
for XSLT generated output, 1190
Output encoding,
default used, 1187, 1188
Overview,
of API de MapForce, 1097

P

PADIS,
IATA, 537
Panel Consulta de la BD, 342
Paquete base, 1075
Para la primera condición desencadenada, 464
Para todas las condiciones desencadenadas, 464
Parámetro,
de entrada dinámico, 724
en contexto, 825
función de entrada como, 653, 655
generación de código, 653
línea de comandos, 121, 653
opcional, 786
salida, 786
usar comodines, 653
valor predeterminado, 655
WHERE SQL, 316
Parámetro de entrada,
dinámico, 724
línea de comandos, 653
y generación de código, 653
y nombres de archivo dinámicos, 727
Parámetro-nombre,
nombre de componente, 659
Parámetros,
ampliar en funciones, 189
Parámetros de la línea de comandos,
comodines entre comillas, 653
Parámetro-valor,
archivo de instancia, 659
Parcial,
importación de base de datos, 309
Parent, 1135, 1141, 1142, 1145, 1150, 1153, 1168, 1170, 1171, 1181, 1183, 1197
parse-dateTime, 828
parse-number, 828
Pasar datos,
sin cambiar a través de una asignación de valores, 633
Path, 1166, 1198
PDF,
documentación de la asignación, 132
Stylevision Power Stylesheets, 144
Período de evaluación,

Período de evaluación,
de los productos de software de Altova, 1359, 1360, 1362
Personales,
funciones XQuery, 801
funciones XSLT, 796
funciones XSLT 2.0, 800
Personalización,
configurar X12, 524
global de EDIFACT, 517
global de X12, 526
local de EDIFACT, 518
X12, 527
Personalización global,
EDIFACT, 517
X12, 526
Personalización local,
EDIFACT, 518
X12, 527
Personalizada,
biblioteca, 189
función, 189
Personalizado,
carácter de relleno, 425
Personalizar,
EDIFACT, 515
transacciones ANSI X12, 523
Pestañas,
pestañas de resultados de StyleVision, 97
pi, 881
Plantilla,
con nombre (sumar), 802
llamar, 796
usar plantillas de FlexText en MapForce, 446
Plantilla con nombre,
sumar nodos, 802
Plataformas,
para productos de Altova, 1354
Posición,
de la secuencia filtrada, 666
nodo / contexto, 666
position, 846, 890, 898
position() en XPath 2.0,
ver fn:position(), 1312
positive, 880
pow, 881
Predeterminado,
parámetro - componente de entrada, 655
valor de entrada, 786
Predeterminado/a,
configuración, recurso global, 690
Predeterminados,
XBRL, 584
Preguntas frecuentes sobre MapForce, 1016
Presentación, 433
Pretty-print,
en el archivo de salida, 981
Primario,
asignación y filtros, 617
Primera fila,
como encabezados de columna, 548
Prioridad,
función, 189
separadores en archivos EDI, 507
y filtros, 617
Procesador XQuery,
de los productos de Altova, 1356
Procesadores,
descargar, 1016
Procesadores XSLT,
de los productos de Altova, 1356
Procesamiento,
automatizar asignaciones, 22
Processing Instructions,
Adding to target files, 212
Programming language,
enumerations for, 1219
Project, 1191
creating new, 1134
file name, 1196
file name and path, 1194
on opening, 1130
opening, 1134
path with filename, 1198
saving, 1198
Project type,
enumerations for C#, 1221
for C#, 1187
for Java, 1189
Propiedades,
tabla de asignación de valores, 637
Protocolos,
WSDL compatibles, 905
Proyecto,
crear servicio web, 908
Punto de código,
intercalación, 619

Punto y coma,
terminar JDBC y DB2, 277

Q

QName, 868
funciones para, 895
string (dimensiones), 599
QName-as-string, 868
Quit, 1135
Quitar,
comentario de bloque, 354
comentario de línea, 354
comentarios, 354
conexión, 103
conexiones de copia total, 165
marcadores, 356
nombre del esquema, 981
regiones, 356
tablas del componente, 238

R

Raíz,
elemento del destino, 645
objeto raíz (esquema de BD), 342
random, 881
RaptorXML Server,
ejecutar una transformación, 24
Recuperación,
modo, 349
Recursiva,
asignación definida por el usuario, 678
función definida por el usuario, 680
Recursivas,
llamadas recursivas en funciones, 767
Recurso,
bases de datos como, 711
carpeta, 697
propiedades del recurso global, 717
Recurso global,
activar, 695
asignar a componente, 693
base de datos como, 711
carpetas como, 697
configuración predeterminada, 690
copiar configuración, 690
flujo de trabajo, 701
iniciar el flujo de trabajo, 707
propiedades, 717
Recursos globales,
archivo de definición, 688
barra de herramientas, 689
definir archivo de recursos, 690
Referencia, 966
funciones de MapForce, 823
Regenerar resultados, 246
Regex, 860
Regiones,
contraer, 356
crear, 356
expandir, 356
insertar, 356
quitar, 356
Registrar el software de Altova, 1015
Registro,
generar XML a partir de, 732
Relación,
conservar/descartar, 300
crear, 303
Relaciones,
Agregar relaciones de tabla, 303
Relaciones de BD,
conservar/descartar, 300
Relaciones de tablas, 244
Relativas,
rutas de acceso (ventajas), 740
Relleno,
caracteres (archivo de longitud fija), 425
remove-fileext, 837
remove-folder, 837
remove-timezone, 869
replace, 883, 895
replace-fileext, 837
Requisitos,
servicios web, 904
Requisitos de memoria, 1354
Reservar,
nombre de método, 1073
resolve-filepath, 837
resolve-uri, 888
Restablecer,

© 2014 Altova GmbH

Restablecer,
conectores secundarios, 103
Resultado de la transformación,
recursos globales, 701
Resultados, 112, 981, 997
agregar ubicación del esquema al archivo de salida, 981
guardar datos, 50
iconos del panel Consulta de la BD, 362
instancia XML, 981
motor de ejecución integrado BUILTIN, 112
pseudo-SQL, 246
validación, 114
validar, 114
validar XML, 112
ventana, 112
ventana (panel Consulta de la BD), 362
ventana, nivel de zoom, 112
Retener datos,
pasar a través de una asignación de valores, 633
reversefind-substring, 883
right, 883
right-trim, 883
round, 843
round-half-to-even, 894
round-precision, 843
RTF,
documentación de la asignación, 132
pestaña, de la ventana Asignación, 76
Stylevision Power Stylesheets, 144
Ruta de acceso,
absoluta durante la generación de código, 46

S

Salida,
archivo (CSV), 414
archivo (texto), 425
definida por el usuario si bool = false, 786
parámetro, 786
varios archivos, 724
Salto de línea,
en la instrucción WHERE SQL, 316
reemplazar caracteres especiales, 639
SAP,
IDocs, 535
Save, 1166, 1198
SaveAs, 1166
Saved, 1166, 1198
Script,
ANT, 1026
Scripts en XSLT/XQuery,
ver Funciones de extensión, 1317
schemanativetype, 1078
Sección,
CDATA, 214
second-from-datetime, 869
second-from-duration, 869
Secuencia,
asignaciones entre secuencias de datos, 1044
de procesamiento de los componentes, 171
posición, 666
posición del elemento, 666
Secuencia de procesamiento,
de los componentes de una asignación, 171
Secuencias de datos,
XML, CSV y FLF, 112
Secundarios,
asignación estándar con secundarios, 163
Segmento,
TA1, 502
Seleccionar,
datos de la tabla (panel Consulta de la BD), 362
Separada,
firma digital, 225
Separador,
de subcomponentes, 507
definidos por el usuario, 507
prioridad en archivos EDI, 507
sub subcampo, 507
Separadores,
EDI, 482
Serialización de QName,
cuando es el resultado de una función XPath 2.0, 1313
Servicio web,
crear C#, 916
crear Java, 911
crear proyecto, 908
elemento de error fault, 918
implementar Java, 911
llamar, 922
Servicios web,
requisitos, 904
set-empty, 848
set-null, 864

© 2014 Altova GmbH

set-null, 864
funciones para valores null, 311
set-xsi-nil, 846
ShowItemTypes, 1176
ShowLibraryFunctionHeader, 1176
shutdown,
of application, 1135, 1136
Signo de interrogación,
elementos desaparecidos, 108
sin, 881
Sitio web de Altova, 1016
SO,
para productos de Altova, 1354
SOAP,
elemento de error fault de servicio web, 918
Software de acompañamiento,
descargar, 1016
Soporte para MapForce, 1016
Soporte técnico para MapForce, 1016
SPL, 1076
bloques de código, 1077
condiciones, 1084
foreach, 1085
objetos globales, 1081
subrutinas, 1087
usar archivos, 1082
variables, 1080
SPS,
documentación definida por el usuario, 142
hojas de estilos predefinidas para documentar asignaciones, 138
Stylevision Power Stylesheet, 144
SQL, 281
abrir script, 353
comandos en la ventana Resultados, 246
consulta directa de la BD, 342
ejecutar instrucciones, 352
eliminar datos antes de la acción de tabla, 281
en acciones de tabla, 281
generar instrucciones SQL, 351
guardar scripts SQL, 353
pseudo-SQL en la ventana Resultados, 246
regenerar resultados, 246
tiempo de espera de ejecución, 285
ventana de la Consulta de la BD, 350
SQL Server,
asignar datos XML, 337
incrustar esquema XML, 337
vista previa de contenido XML, 337
SQL WHERE,
ORDER BY, 313
SQL/XML,
consultar datos XML, 329
sqrt, 881
starts-with, 852, 895
startup,
of application, 1136
Status, 1136, 1208
sting-compare-ignore-case, 883
string, 828, 888
qname (dimensiones), 599
string-compare, 883
string-join, 825
string-length, 852
Stylevision,
asignación en cadena, componente final, 176, 181
asignar un archivo SPS a un componente, 146
definir hojas de estilos SPS para documentar asignaciones, 142
hojas de estilos de taxonomías XBRL, 579
hojas de estilos en la vista previa, 144
HTML, PDF, RTF, Word2007+, 144
parámetros de la línea de comandos, 121
pestaña, de la ventana Asignación, 76
pestañas de resultados de la asignación, 97
sub subcampo,
separador, 507
Subir,
elemento en el componente, 981
substitute-missing, 846
substitute-missing-with-xsi-nil, 846
substitute-null, 864
función para bases de datos, 311
substring, 852
substring-after, 852, 895
substring-before, 852, 895
Subtract, 843, 891
sum, 825
Suma,
nodos en XSLT 1.0, 802
Sustituir,
nodo que falta, 662
Sybase,
información, 296
system-property, 900

T

TA1,
segmento, 502
Tabla,
acciones de tabla de base de datos, 282
agregar/quitar del componente, 238
búsqueda, asignación de valores, 628
eliminar datos antes de la acción de tabla, 281
icono de acciones de tabla, 254
jerarquía, 244
menú contextual, 361
múltiples acciones de tabla, 254
omitir datos en tablas secundarias, 282
quitar nombre del esquema, 981
quitar nombre del esquema de la tabla, 981
relaciones (conservar/descartar), 300
vista previa, 324, 337
visualizar primario/secundario, 244
Tabla de búsqueda,
asignar nodos que faltan, 664
dimensiones de asignación de valores, 599
propiedades, 637
tabla de asignación de valores, 628
Tablas raíz, 244
tan, 881
Taxonomía,
insertar, 580
Tecla de acceso rápida,
teclado, 1002
Teclado,
tecla de acceso rápida, 1002
Teclas de acceso rápido,
nivel de zoom de la ventana de resultados, 112
Terminación,
punto y coma en IBM DB2 y JDBC, 277
Terminología, 14
EDIFACT, 482
Texto,
archivos (definir campos de clave), 410
asignación, 418
asignar archivos de texto, 403
asignar texto a filas, 407
crear archivo de texto nuevo, 425
iterador (filas), 407
Texto delimitado entre, 414, 425
Tiempo de espera,
base de datos, 285
iSeries (deshabilitar), 324
time,
xsd:time, 507
time-from-datetime, 869
time-to-xlsx, 887
timezone, 869
Tipo,
convertir en tipo de destino, 981
Tipo complejo,
ordenar, 619
Tipo de datos,
detección automática WHERE SQL, 316
explícito / implícito, 800
Tipo simple,
ordenar, 619
Tipos,
integrados, 1091
tipos derivados (xsi:type), 203
tipos de datos,
campo, 414
en XPath 2.0 y XQuery 1.0, 1312
to-date, 866
to-datetime, 866
Todos los conceptos,
configuración del componente XBRL, 584
Todos los conceptos (sin procesar),
configuración del componente XBRL, 584
to-duration, 866
Tokenize, 852, 856
Tokenize-by-length, 852, 856
tokenize-regexp, 852
to-time, 866
Transacción,
definir / configurar, 282
Transacciones,
personalizar X12, 523
Transformación por lotes,
DoTransform, 112
Transformación XML canónica,
firma digital, 225
Transformaciones,
RaptorXML Server, 24
Transformar,
datos de entrada, asignación de valores, 628
DoTransform.bat, 50

© 2014 Altova GmbH

translate, 852
Tratar,
campos vacíos como ausentes, 470, 476
true, 889
Tutorial, 26
carpeta Examples, 26
FlexText, 436
TXT,
tipos de datos de campo, 414

U

UN/EDIFACT, 480
asignar a XML, 483
como destino de la asignación, 507
ejemplo, 520
personalizar, 515
personalizar (configuración), 515, 520
reglas de finalización automática, 510
reglas de validación de destino, 510
terminología, 482
unary-minus, 881
Unicode,
intercalación de punto de código, 619
Unidades predeterminadas,
mostrar XBRL, 601
Unir,
archivos XML, 651
unparsed-entity-uri, 900
uppercase, 883
upper-case, 895
URI,
en documentos DTD, 202
URI de espacio de nombres,
DTD, 202
Usar,
recursos globales, 695
Uso de Internet,
en los productos de Altova, 1358
Usuario,
separadores definidos por, 507

V

Vacío,
archivo de texto (crear nuevo), 425
Vacíos,
campos vacíos (tratar como ausentes), 470, 476
Validación,
de código en ejecución, 507
validación con esquema del documento XML,
para XQuery, 1309
Validador,
de los productos de Altova, 1355
Validar,
datos de la ventana Resultados, 50
proyecto de asignación, 114
resultados, 114
XML, 112
Valor,
componente de entrada - vista previa, 655
guardar, 478
predeterminado, 786
Valor asignado,
configuración de clave (acción de tabla), 281
Variable,
casos de uso, 746
insertar, 746
intermedia, 746
parámetro WHERE SQL, 316
Variables,
de SPL, 1080
Variables intermedias, 746
Varias,
entras / salidas, 722
Varias tablas,
en un XML, 643
Varios,
destinos y Java, 46
elementos de entrada, 57
esquemas de destino, 46
esquemas de origen, 981
ver varios esquemas de destino, 50
Varios archivos,
compatibilidad con, lenguajes, 722
entrada / salida, 62
entrada/salida, 724

© 2014 Altova GmbH

Varios archivos de origen,
en un solo destino, 651
Varios archivos XML,
a partir de un archivo Excel, 734
a partir de un solo XML de origen, 729
a partir de una BD, 732
Varios elementos consecutivos,
Edifact - X12, 523
Varios orígenes,
asignar a un solo destino, 52
Ventana Asignación,
alineación automática, 86
pestañas de StyleVision, 76
Version, 1133
Versión,
compatibilidad de las clases contenedoras, 1002
View,
of MapForce, 1173
Visible, 1136
Vista, 999
Vista previa,
motor de MapForce, 112
Stylevision Power Stylesheets, 144
tablas y contenido, 324
tables and content, 337
valor del componente de entrada, 655
Visual Basic,
error handling, 1110
integration of MapForce, 1265
Visual Studio,
versiones compatibles (generación de código), 1002
Visual Studio .NET,
abrir archivos de MapForce en, 933
complemento de MapForce, 932
diferencias entre MapForce y MapForce para Visual Studio, 936
Volver a ejecutar un script SQL,
regenerar resultados, 246
VS .NET,
compilar servicio web C#, 916
crear/implementar servicio web C#, 916
VS NET 2002/2003, 1028

W

weekday, 869
weeknumber, 869
Where,
componente WHERE SQL, 313
consultar datos XML en DB2, 329
operador WHERE SQL, 316
Where de SQL,
consultar datos XML, 329
WHERE SQL,
componente (insertar), 313
detección automática campo tipo de datos, 316
operadores, 316
parámetro, 316
salto de línea, 316
Wildcards,
xs:any - xs:anyAtrribute, 216
WindowHandle, 1136
Windows,
compatibilidad con productos de Altova, 1354
Word,
documentación de la asignación, 132
Word2007+,
pestaña, de la ventana Asignación, 76
Stylevision Power Stylesheets, 144
WSDL,
compatibilidad con 2.0, 905
llamada a servicio web de entrada/salida, 923, 929
protocolos compatibles, 905

X

X12,
actualizar archivo de configuración para mensajes múltiples, 533
como destino de la asignación, 507
configurar personalización, 524
confirmación, 502
confirmación de implementación 999, 500
confirmación funcional 997, 498
entradas combinadas, 523
generación de código (clase de C++), 520
HIPAA, 540
local, 527
personalización global, 526
reglas de finalización automática, 512
reglas de validación de destino, 512
segmento TA1, 502

© 2014 Altova GmbH

X12,
separar entradas combinadas, 530
xbrl, 579, 886
asignación de datos entre BD y taxonomía, 605
asignación de valores, 599
asignar archivos de entrada/salida, 579
ejemplo de asignación entre Excel y XBRL, 601
elementos obligatorios para archivo de instancia, 601
elementos predeterminados, 584
hipercubos (dominios/dimensiones), 593
hojas de estilos de taxonomías de Stylevision, 579
insertar documento/taxonomía, 580
presentación/composición del componente, 584
xbrl-measure-currency, 886
xbrl-measure-pure, 886
xbrl-measures-shares, 886
Xerces,
bibliotecas, 1002
compatibilidad, 1075
xlsx,
funciones de la biblioteca, 887
OOXML, 547
XML, 112
asignar a UN/EDIFACT, 483
asignar datos de BD a OOXML, 571
asignar datos XML a componentes IBM DB2 de destino, 331
asignar datos XML desde DB2, 324
asignar OOXML a XML, 568
asignar varias tablas a, 643
asignar y consultar datos XML, 329
configuración de la firma, 225
consultar datos XML en DB2, 329
generar esquema XML a partir de, 28
guardar resultado, 112
secuencias de datos, 112
validar resultados, 112
vista previa de contenido XML, 324, 337
XML a base de datos,
asignación, 238
XML en XML, 201
xmlexists,
función de consulta, 329
xpath, 898
en consultas DB2 XML, 329
sumar varios nodos, 802
xpath2,
biblioteca, 189
funciones de la biblioteca, 888
XQuery,
agregar funciones personales, 801
funciones, 189
Funciones de extensión, 1317
y bases de datos, 230
xs:,
constuctores, 889
xs: any (xs:anyAttribute), 216
xs:date, 890
xs:QName,
ver QName, 1313
xs:time, 890
xsd,
date / time, 507
Xsi:nil,
valores que pueden ser nulos, 208
xsi:type,
asignación a tipos derivados, 203
xsl:preserve-space, 1304
xsl:strip-space, 1304
XSLT,
agregar funciones personales, 796
biblioteca estándar, 189
code generation, 1163
espacio de nombres de la plantilla, 796
Funciones de extensión, 1317
funciones de la biblioteca, 898
generar varios destinos, 50
options, 1190
pestaña, de la ventana Asignación, 76
XSLT 1.0/2.0, 45
archivo por lotes DoTransfrom, 45
generar (tutorial), 45
XSLT 2.0,
agregar funciones personales, 800
xslx-to-date, 887
xslx-to-datetime, 887
xslx-to-time, 887

Y

year-from-datetime, 869
year-from-duration, 869

Z

Z-A,
componente de ordenación, 619
zona horaria implícita,
en las funciones XPath 2.0, 1312
Zoom,
nivel en la ventana de resultados, 112

© 2014 Altova GmbH