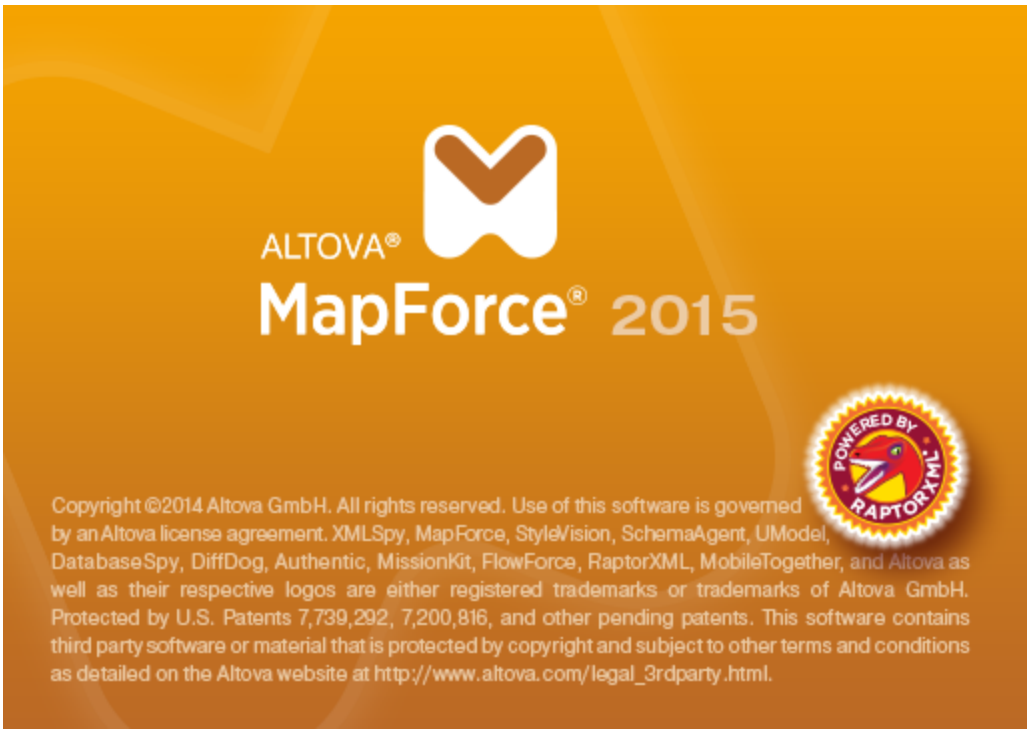
Manual del usuario y referencia del programador



Manual del usuario y referencia de Altova MapForce 2015

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 2015 3

2 Características nuevas 6

3 Introducción a MapForce 10

3.1 Terminología ... 12

4 RaptorXML Server 18

5 Tutorial de MapForce 20

5.1 Configuración del entorno de asignación ... 22

5.1.1 Agregar componentes al panel Asignación ... 23

5.2 Crear una asignación ... 27

5.2.1 Asignaciones entre elementos de un esquema ... 28

5.2.2 Uso de las funciones en las asignaciones de datos ... 31

5.2.3 Filtrado de datos ... 35

5.3 Generación de código XSLT 1.0 y 2.0 ... 39

5.4 Trabajar con varios esquemas y documentos de destino ... 40

5.4.1 Crear un segundo componente de destino ... 41

5.4.2 Ver y generar el resultado de varios esquemas de destino ... 44

5.5 Asignar varios componentes de origen a uno de destino ... 46

5.5.1 Crear la asignación ... 48

5.5.2 Duplicar elementos de entrada ... 51

5.6 Trabajar con varios archivos de entrada/salida ... 56

5.6.1 Procesar varios archivos por cada componente de entrada/salida ... 57

6 Interfaz del usuario 62

6.1 Ventana Bibliotecas ... 64

6.2 Panel Asignación ... 66

6.3 Panel XSLT / XSLT2 ... 68

6.4 Panel Resultados ... 69
6.5 Paneles relacionados con StyleVision ... 71
6.6 Ventana Vista general ... 72
6.7 Ventana Mensajes ... 73

7 Funcionamiento general 76

7.1 Mover y conservar conectores ... 77
7.2 Elementos desaparecidos ... 82
7.3 Seleccionar el lenguaje de transformación ... 86
7.4 Vista previa del resultado de la transformación ... 87
7.5 Vista previa del código XSLT ... 88
7.6 Validar la asignación y su resultado ... 89
7.7 Parámetros de la línea de comandos ... 92
7.8 Archivos de catálogo en MapForce ... 94
7.9 Archivos SPS en la vista previa de MapForce ... 99
7.9.1 Asignar un archivo SPS a un componente ... 101

8 Asignaciones entre componentes 106

8.1 Métodos de asignación de datos ... 111
8.1.1 Basada en el destino (asignación estándar) ... 112
8.1.2 Basada en el origen (asignación de contenido mixto) ... 113
Asignación de contenido mixto ... 113
Ejemplo de contenido mixto ... 118
Usar asignaciones estándar en elementos de contenido mixto ... 119
8.1.3 Conexiones de copia total ... 121
8.2 Configuración de la conexión ... 124
8.3 Conexiones y resultados de la asignación ... 126
8.4 Secuencia de procesamiento de los componentes ... 127
8.5 Asignaciones en cadena / componentes de paso a través ... 130
8.5.1 Asignaciones en cadena: paso a través activo ... 132
8.5.2 Asignaciones en cadena: paso a través inactivo ... 137
8.5.3 Ejemplo de asignación en cadena ... 142
8.6 Uso de las funciones ... 145
8.7 Bucles, grupos y jerarquías ... 148
8.8 Reglas y estrategias de asignación de datos ... 149

9 Datos de origen y destino 156

9.1 XML y esquemas XML ... 157
9.1.1 Usar documentos DTD como componentes de esquema ... 158
9.1.2 Asignar tipos derivados de XML Schema ... 159
9.1.3 Uso de nombres QName ... 161
9.1.4 Valores nulos / que pueden ser nulos ... 164
9.1.5 Comentarios e instrucciones de procesamiento ... 167
9.1.6 Secciones CDATA ... 169
9.1.7 Comodines xs:any / xs:anyAttribute ... 171
9.2 Asignación de datos entre HL7 v3 y esquemas XML ... 176

10 Consejos prácticos 178

10.1 Componentes de filtrado: consejos ... 180
10.2 Componentes de ordenación: ordenar secuencias de entrada ... 182
10.3 Asignación de valores: transformar datos de entrada ... 190
10.3.1 Pasar datos sin cambiarlos a través de una asignación de valores ... 195
10.3.2 Propiedades de las asignaciones de valores ... 199
10.4 Funciones de agregado: min, max, sum, count, avg ... 201
10.5 Elemento raíz de los documentos de destino ... 203
10.6 Comparación booleana de los nodos de entrada ... 204
10.7 Nodo de contexto prioritario ... 206
10.8 Unir varios archivos en un solo archivo de destino ... 208
10.9 Línea de comandos: definir parámetros de entrada ... 210
10.10 Parámetros de entrada: configuración predeterminada y vista previa ... 211
10.11 Nombres de los componentes ... 215
10.12 Comprobación, posición y agrupamiento de nodos ... 216
10.12.1 Asignar nodos que faltan con not-exists ... 218
10.12.2 Posición de los elementos de contexto en una secuencia ... 220
10.12.3 Agrupamiento de nodos / contenido de nodos ... 223
10.13 Asignación recursiva definida por el usuario ... 230
10.13.1 Definir una función recursiva definida por el usuario ... 232

11 Recursos globales 240

11.1 Recursos globales: archivos ... 241
11.1.1 Definir y agregar recursos globales ... 242
11.1.2 Asignar un recurso global ... 245

11.1.3 Usar y activar un recurso global ... 247
11.2 Recursos globales: carpetas ... 249
11.3 Recursos globales: flujo de trabajo en la aplicación ... 253
11.3.1 Iniciar el flujo de trabajo en la aplicación ... 259
11.4 Recursos globales: propiedades ... 263

12 Archivos de entrada/salida dinámicos por componente 268

12.1 Nombres de archivo dinámicos: entrada/salida ... 270
12.2 Nombres de archivo dinámicos como parámetros de entrada ... 273
12.3 Crear varios archivos XML a partir de un solo XML de origen ... 275
12.4 Rutas de acceso de archivo relativas y absolutas ... 278

13 Variables intermedias 284

13.1 Variables: casos de uso ... 290

14 Bibliotecas y funciones 294

14.1 Definir funciones definidas por el usuario ... 295
14.1.1 Parámetros de las funciones ... 301
14.1.2 Funciones definidas por el usuario inline y normales ... 305
14.1.3 Crear una función de búsqueda sencilla ... 307
14.1.4 Función definida por el usuario compleja: con un nodo XML como entrada ... 312
Definir componentes de entrada complejos ... 313
14.1.5 Función definida por el usuario compleja: con un nodo XML como salida ... 318
Definir componentes de salida complejos ... 318
14.1.6 Ejemplo de función definida por el usuario ... 324
14.2 Agregar funciones XSLT y XQuery personales ... 331
14.2.1 Agregar funciones XSLT 1.0 personales ... 332
14.2.2 Agregar funciones XSLT 2.0 personales ... 336
14.2.3 Funciones de agregado: sumar nodos en XSLT 1.0/2.0 ... 337
14.3 Referencia de funciones ... 340
14.3.1 Biblioteca core ... 341
aggregate (funciones de agregado) ... 341
conversion (funciones de conversión) ... 344
file path (funciones de ruta de acceso) ... 351
generator (funciones generador) ... 353
logical (funciones lógicas) ... 355

math (funciones matemáticas) ... 357
node (funciones de nodo) ... 360
sequence (funciones de secuencia) ... 363
string (funciones de cadena) ... 366
Ejemplos de la función tokenize ... 370
Expresiones regulares ... 374
14.3.2 Biblioteca xpath2 ... 378
accessors (descriptores de acceso) ... 378
anyURI ... 378
boolean (funciones booleanas) ... 379
constructores ... 379
context (funciones de contexto) ... 380
duration, date and time (funciones de duración, fecha y hora) ... 381
node (funciones de nodo) ... 383
numeric (funciones numéricas) ... 384
qname-related (funciones para QName) ... 385
string (funciones de cadena) ... 385
14.3.3 Biblioteca xslt ... 388
xpath ... 388
xslt ... 390

15 Referencia del usuario 394

15.1 Archivo ... 395
15.2 Edición ... 400
15.3 Insertar ... 401
15.4 Componente ... 403
15.5 Conexión ... 408
15.6 Función ... 413
15.7 Resultados ... 416
15.8 Vista ... 418
15.9 Herramientas ... 420
15.10 Ventanas ... 428
15.11 Ayuda ... 429
15.11.1 Contenido, índice, buscar ... 430
15.11.2 Activación, formulario de pedido, registro, actualizaciones ... 431
15.11.3 Otros comandos ... 433

16 Anexos 436

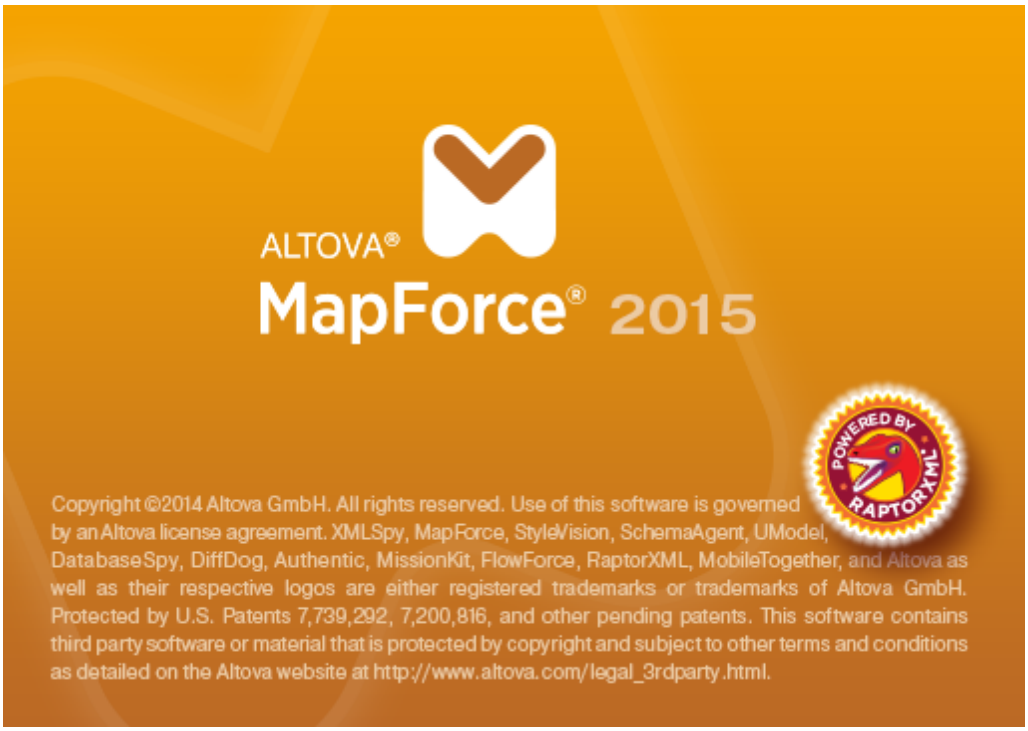
16.1 Información sobre motores de Altova ... 437
16.1.1 Información sobre motores XSLT y XQuery ... 438
XSLT 1.0 ... 438
XSLT 2.0 ... 438
XQuery 1.0 ... 440
16.1.2 Funciones XSTL y XPath/XQuery ... 444
Funciones de extensión de Altova ... 445
Funciones XSLT ... 446
Funciones XPath/XQuery: fecha y hora ... 449
Funciones XPath/XQuery: funciones de cadena ... 462
Funciones XPath/XQuery: funciones varias ... 467
Funciones de extensión varias ... 476
Funciones de extensión Java ... 476
Archivos de clases definidos por el usuario ... 478
Archivos JAR definidos por el usuario ... 481
Constructores ... 482
Métodos estáticos y campos estáticos ... 482
Métodos de instancia y campos de instancia ... 483
Tipos de datos: conversión de XPath/XQuery en Java ... 484
Tipos de datos: conversión de Java en XPath/XQuery ... 485
Funciones de extensión .NET ... 485
Constructores ... 488
Metodos estáticos y campos estáticos ... 489
Métodos de instancia y campos de instancia ... 489
Tipos de datos: conversión de XPath/XQuery en .NET ... 490
Tipos de datos: conversión de .NET en XPath/XQuery ... 491
Scripts MSXSL para XSLT ... 491
16.2 Datos técnicos ... 495
16.2.1 Requisitos de SO y memoria ... 496
16.2.2 Validador XML de Altova ... 497
16.2.3 Motores XSLT y XQuery de Altova ... 498
16.2.4 Compatibilidad con Unicode ... 499
16.2.5 Uso de Internet ... 500
16.3 Información sobre licencias ... 501
16.3.1 Distribución electrónica de software ... 502
16.3.2 Activación del software y medición de licencias ... 503
16.3.3 Derechos de propiedad intelectual ... 504
16.3.4 Contrato de licencia para el usuario final de Altova ... 505

Altova MapForce 2015

MapForce 2015

1 MapForce 2015

MapForce® 2015 Basic 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: 04/09/2014

Altova MapForce 2015

Características nuevas

2 Características nuevas

Novedades destacadas de **MapForce 2015**:

- Nuevo argumento `language` para las funciones `format-date` y `format-dateTime`
- Nueva función de secuencia: `replicate-item`

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 conservar los conectores después de eliminar componentes
- 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 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

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

- Altova publica el rapidísimo motor de transformación RaptorXML Server

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

- Configuración de final de línea definida por el usuario para los archivos de resultados.
- Actualizaciones y mejoras internas

Novedades destacadas de **MapForce 2013**:

- 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

Novedades destacadas de **MapForce 2012**:

- Alineación automática de componentes en la ventana de asignación
- Avisos para conectarse al nodo primario de destino
- Reglas específicas que determinan la secuencia en la que se procesan los componentes en una asignación

 © 2014 Altova GmbH

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

- Variables intermedias

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

- 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**
-

Novedades destacadas de **MapForce 2011**:

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

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

- 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

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

- Conexión automática de conectores secundarios idénticos al mover un conector primario
- Posibilidad de acortar cadenas de entrada antes de procesarlas

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
- Mejoras en el tratamiento de contenido mixto en elementos XML

Novedades destacadas de **MapForce 2009 SP1**:

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

Novedades destacadas de **MapForce 2009**:

- Posibilidad de usar EDI HL7 versión 3.x XML como componentes de origen y destino
- Agrupación de nodos o del contenido del nodo
- Posibilidad de filtrar los datos según la posición del nodo en una secuencia
- Compatibilidad con QName
- Búsqueda de nodos y elementos en componentes

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

- Función para generar esquemas XML para archivos XML automáticamente
- 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

 © 2014 Altova GmbH

Altova MapForce 2015

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 se pueden asignar a otros documentos XML. La transformación se realiza por medio de hojas de estilos XSLT 1.0 o 2.0 generadas automáticamente.

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 <%MAP-API%> (control ActiveX)
- Definición de bibliotecas XSLT 1.0 y 2.0 personalizadas
- Compatibilidad con funciones XPath 2.0 en XSLT 2.0 y XQuery

 © 2014 Altova GmbH

- 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 4.2 / 4.3 / 4.4
- 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 4.2 / 4.3 / 4.4 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.

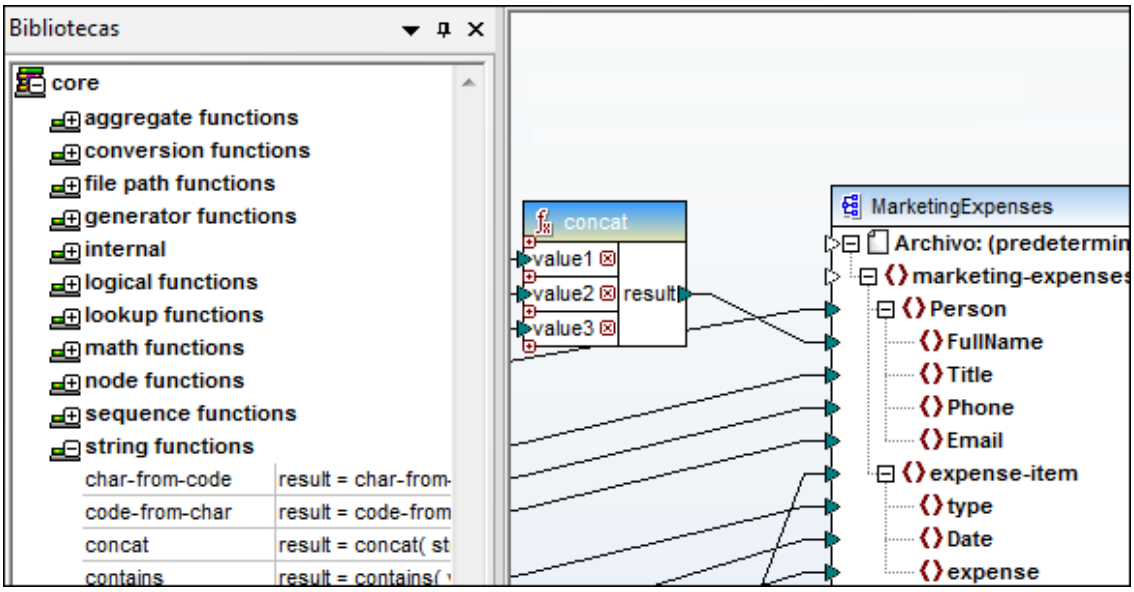
Biblioteca
Una *biblioteca* es una colección de funciones visible en la ventana Bibliotecas. Hay varios tipos de funciones, como funciones básicas, funciones propias de determinados lenguajes de programación, funciones definidas por el usuario y funciones personalizadas. Para más información, consulte 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
-
- Tipos de funciones: XSLT/XSLT2, 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.

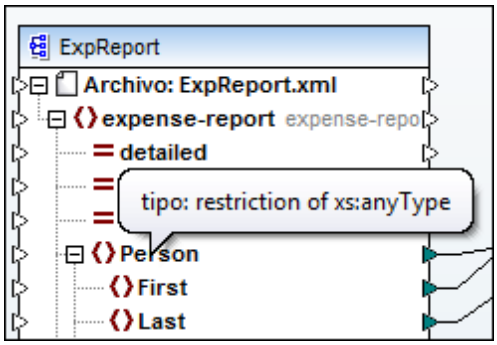


Con Java seleccionado

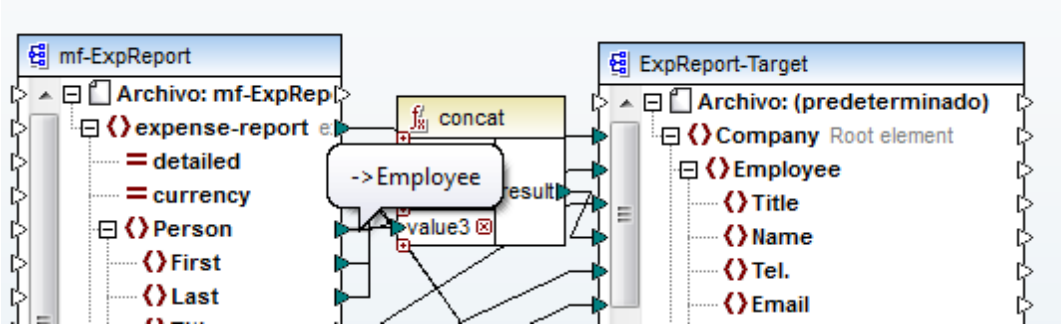
Elemento
Un *elemento* representa los datos que se pueden asignar de un componente a otro. Un elemento puede ser de varios tipos: **elemento**, **atributo**.

 © 2014 Altova GmbH

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.



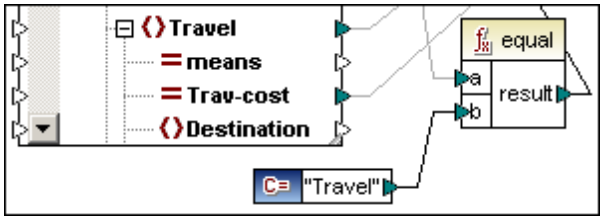
Conector
Un *conector* es la **línea** que une dos iconos. El conector representa la **asignación** existente entre los dos conjuntos de datos que los iconos representan. Para más información consulte la sección Asignaciones entre componentes.

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).

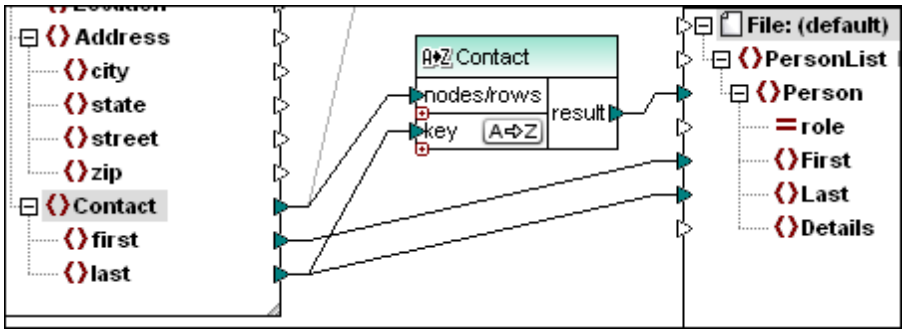


Variable

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.

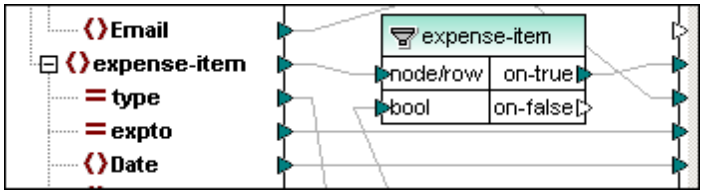
Componente de ordenación

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*).



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**.

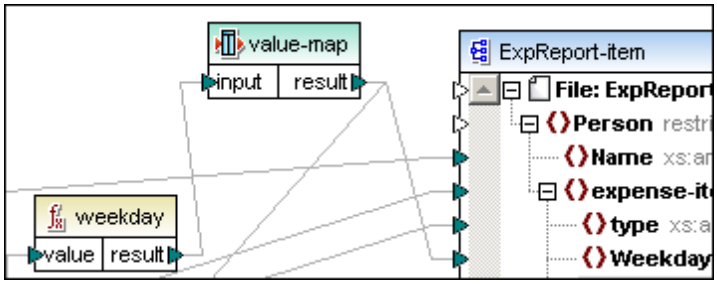


El parámetro de salida **on-false** produce el grupo de nodos definido por la asignación. Para más información consulte el apartado Trabajar con varios esquemas y documentos de destino.

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.

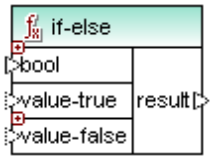
 © 2014 Altova GmbH



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 **`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 2015

RaptorXML Server

4 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, 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 Altova MapForce Server y FlowForce Server.

Nota: puede obtener una vista previa del código XSLT con el motor de ejecución integrado.

 © 2014 Altova GmbH

Altova MapForce 2015

Tutorial de MapForce

5 Tutorial de MapForce

Este tutorial le enseñará a realizar varias tareas en MapForce 2015, 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
- 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\Datos de programa\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)
- `Tut-ExpReport-multi.mfd` La asignación de datos del informe de gastos (con varios componentes de destino)

 © 2014 Altova GmbH

- `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.

5.1 Configuración del entorno de asignación

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

Objetivo

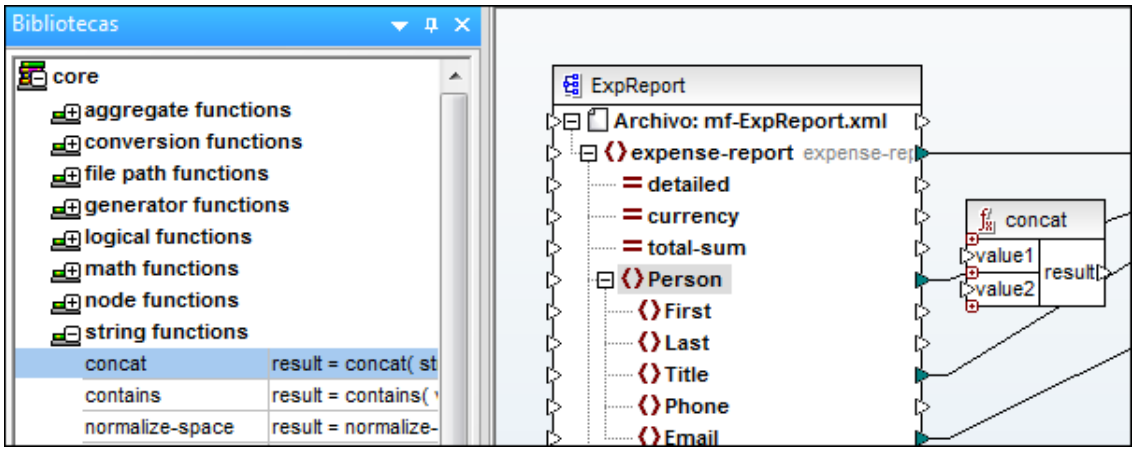
En esta sección del tutorial aprenderá a configurar el entorno de asignación en MapForce. En concreto, aprenderá a:

- 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.



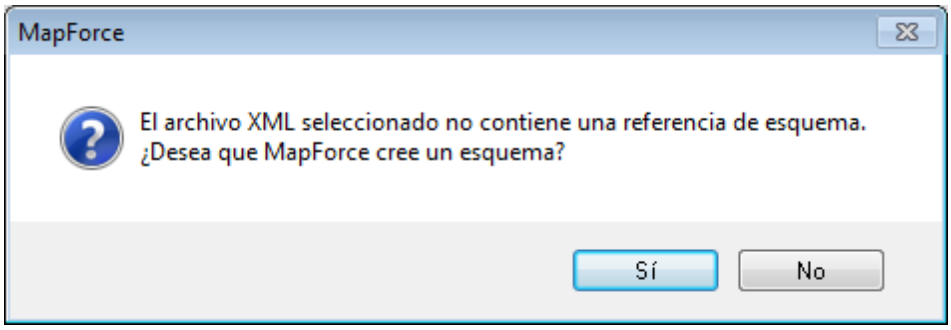
Con XSLT seleccionado

 © 2014 Altova GmbH

5.1.1 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**.

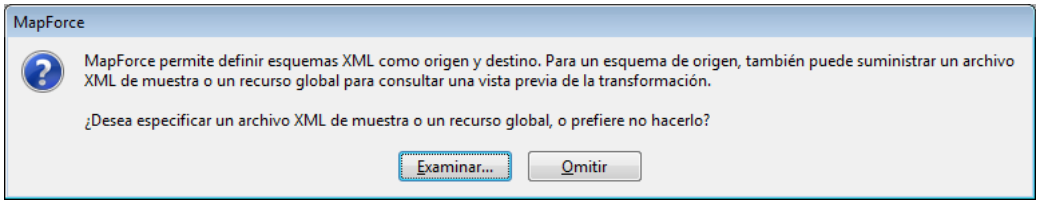
MapForce puede generar automáticamente un esquema XML a partir de un archivo XML ya existente si no está disponible el esquema XML. Si MapForce no encuentra ningún esquema XML asociado al archivo XML insertado con el comando **Insertar archivo o esquema XML**, aparece automáticamente este cuadro de diálogo.



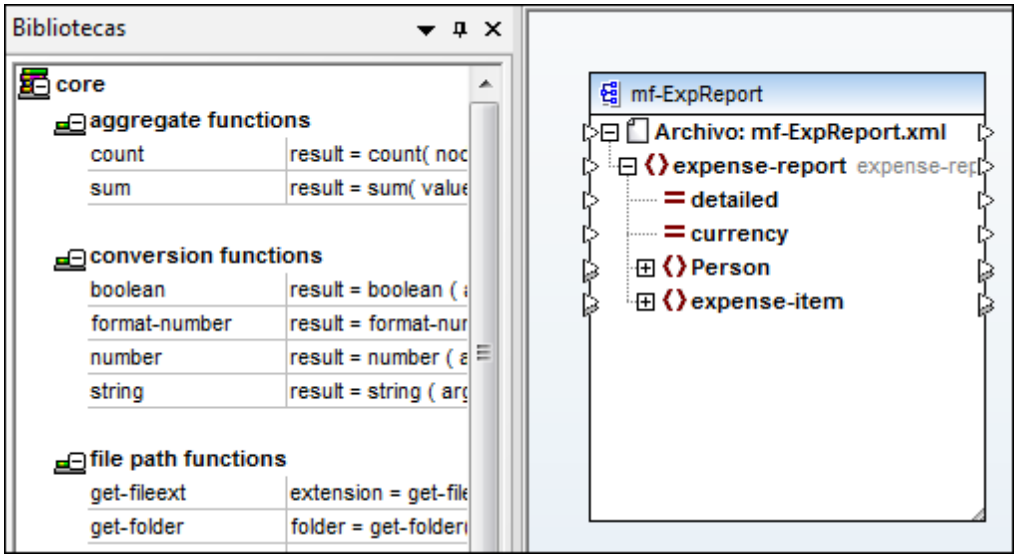
Cuando se genera un esquema a partir de un archivo XML, los tipos de datos de los elementos/atributos deben deducirse del documento XML de instancia y puede que no sean exactamente los tipos deseados. Por eso, recomendamos que revise el esquema generado y compruebe si representa con precisión los datos 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 `...\MapForce2015\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.

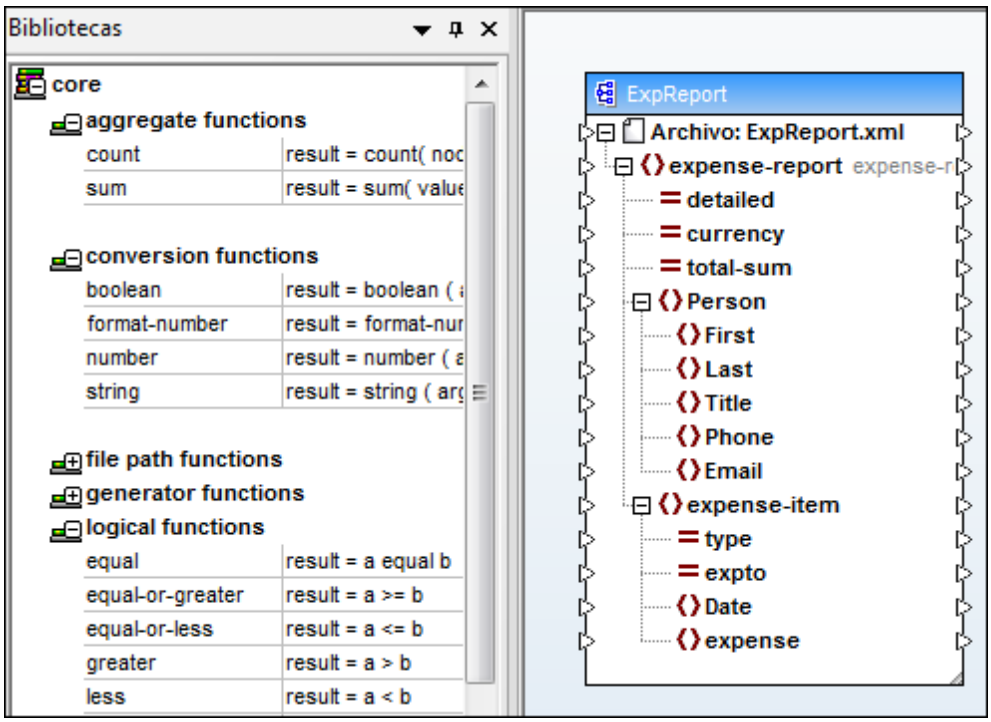


3. Pulse el botón **Examinar...** y seleccione el archivo **`mf-ExpReport.xml`**.
   El componente de esquema de origen aparece ahora en el panel **Asignación**.



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.



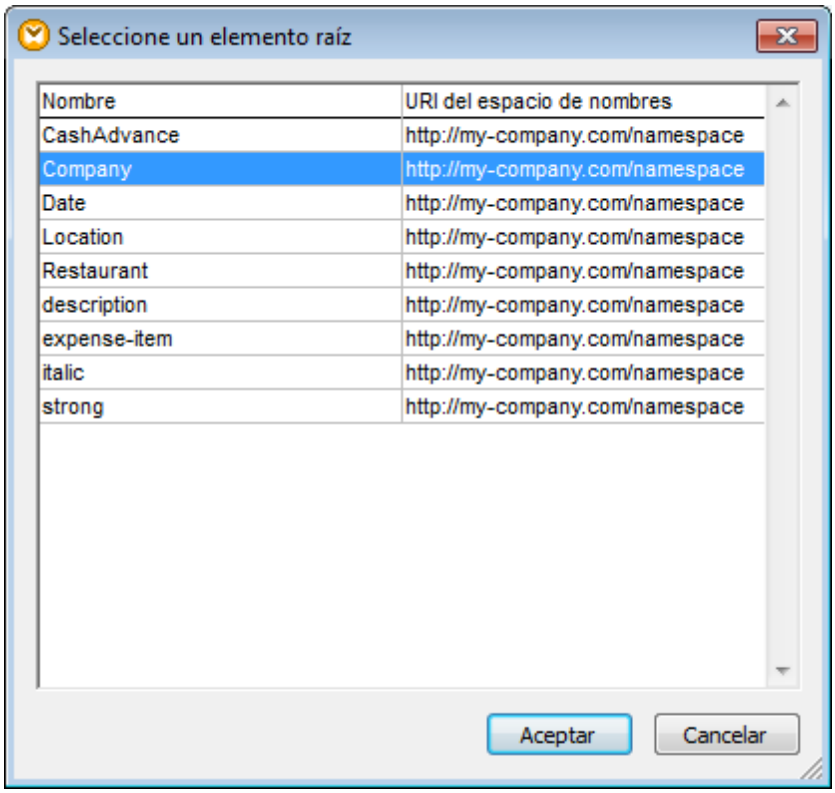
Con XSLT seleccionado

Para crear el componente de esquema de destino:

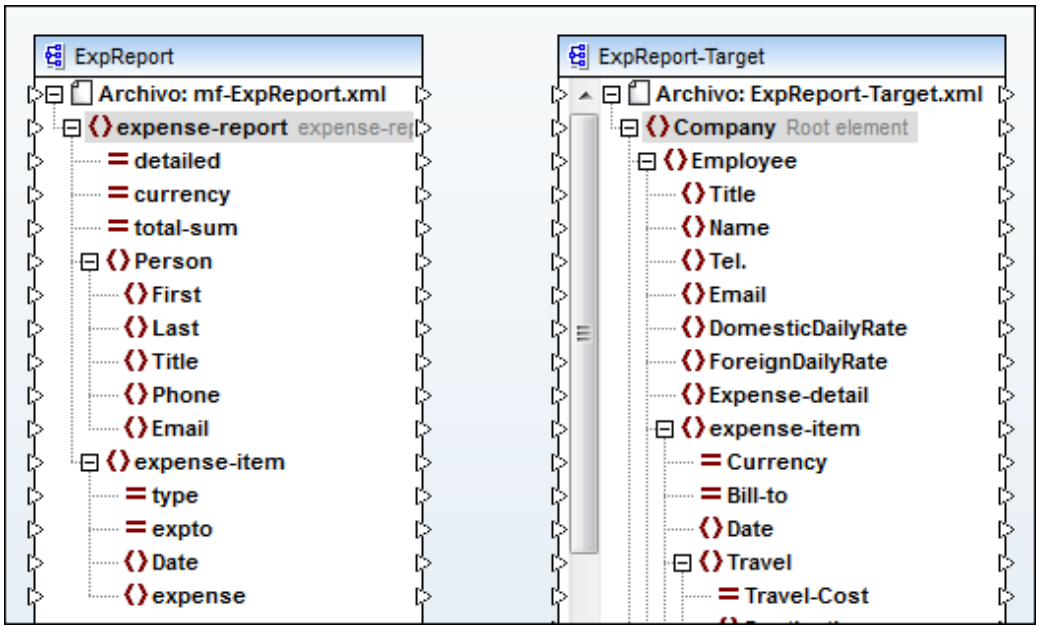
1. Haga clic en el icono **Insertar archivo o esquema XML** o seleccione la opción de menú **Insertar | Archivo o esquema XML...**.

 © 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

5.2 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**.

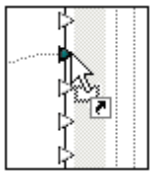
Sitio web de Altova: Integración de datos y asignación de datos XML

5.2.1 Asignaciones entre elementos de un esquema

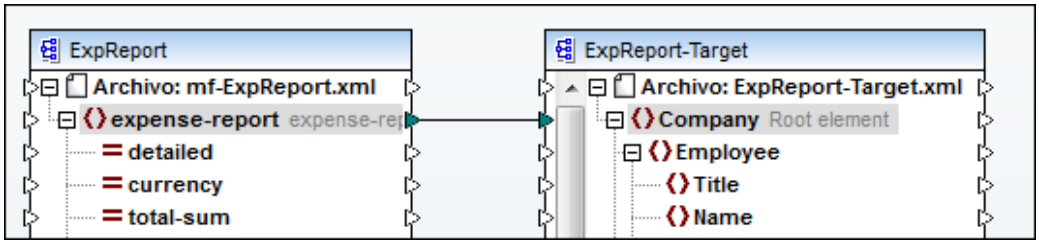
En este apartado aprenderá a definir asignaciones entre los elementos del esquema de origen y los del esquema de destino.

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

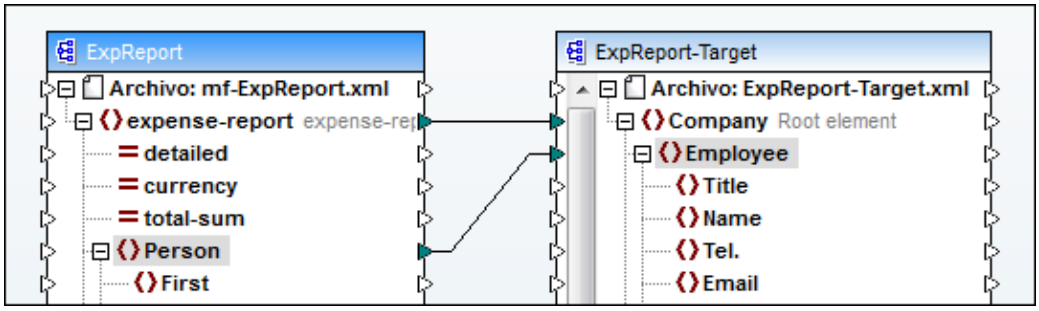
1. Haga clic en el elemento **`expense-report`** del esquema de origen **`mf-ExpReport`** y arrastre el mouse.
   Automáticamente aparece una línea de conexión que surge del icono de salida del elemento y está unida al puntero del mouse, que ahora tiene otra forma.
2. Mueva el puntero del mouse hasta 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.

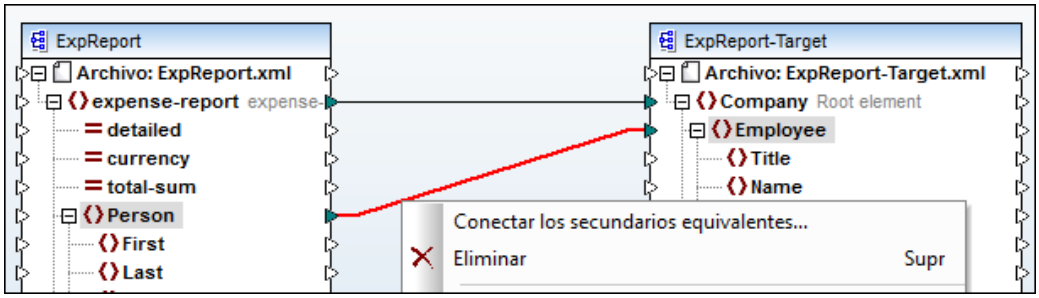


3. Use el método que acabamos de describir para crear una asignación entre los elementos **`Person`** y **`Employee`**.

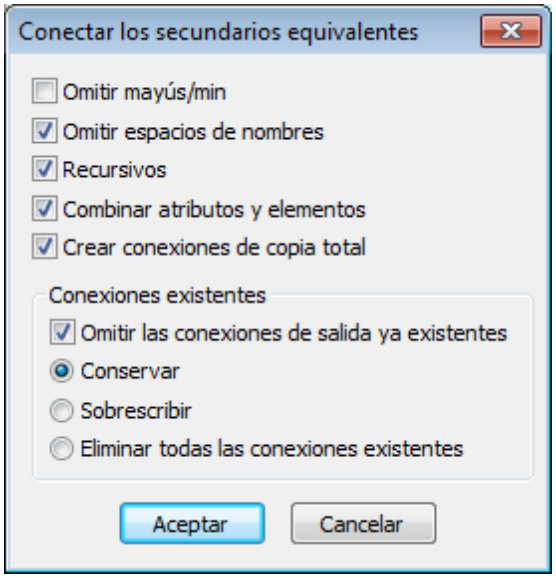


   Si está activo el icono **Conectar automáticamente los secundarios equivalentes** , entonces los elementos **`Title`** y **`Email`** se conectan automáticamente. Si no es así:
4. Haga clic con el botón secundario en el conector de **`Person`** y seleccione la opción **Conectar los secundarios equivalentes**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

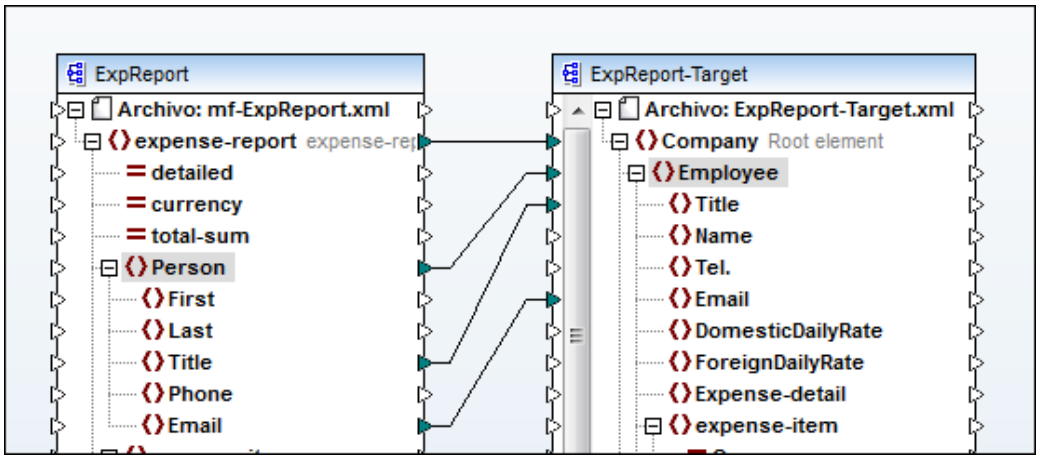
 © 2014 Altova GmbH



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

5.2.2 Uso de las funciones en las asignaciones de datos

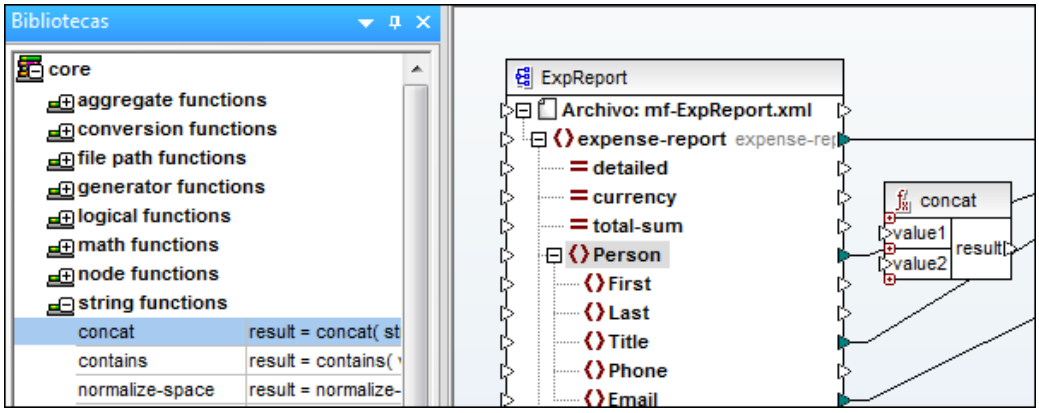
El objetivo de este apartado es combinar dos conjuntos de datos del esquema de origen y colocar el resultado en un solo elemento del documento de destino. Para ello será necesario:

- 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.

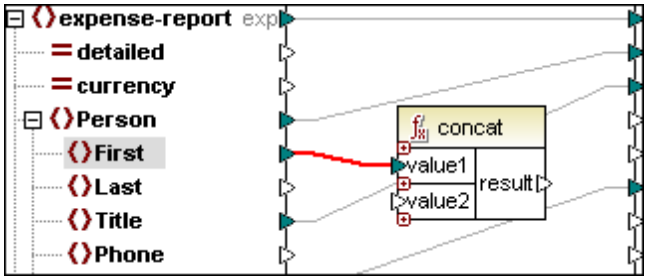
Para combinar elementos usando funciones:

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**.

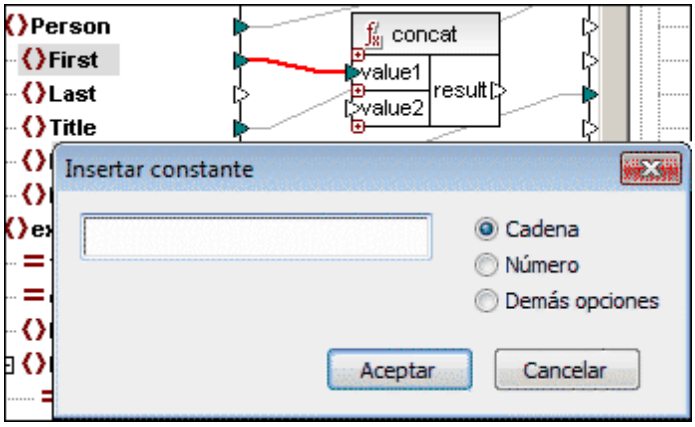


Con XSLT seleccionado

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`**.

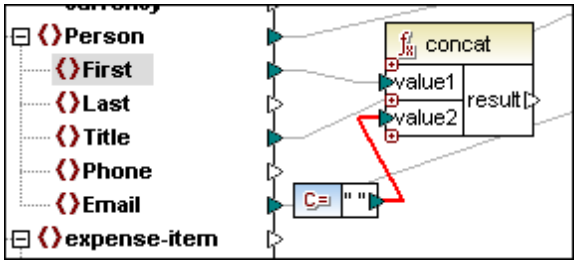


3. Haga clic con el botón secundario en el fondo del panel, cerca de la entrada **`value2`** y seleccione la opción **Insertar constante...** del menú contextual para insertar un componente de constante.

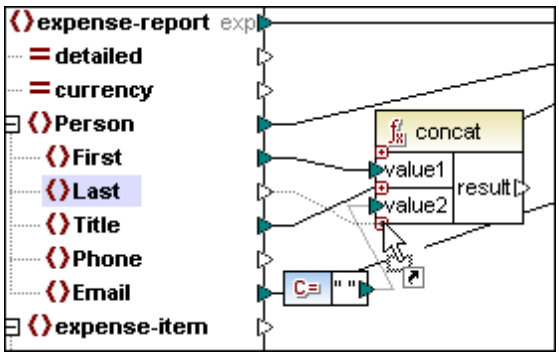


4. Introduzca un carácter de espacio en el cuadro de texto y haga clic en **Aceptar**. Ahora aparece el componente de constante en el área de trabajo. Su contenido aparece junto al icono de salida.

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

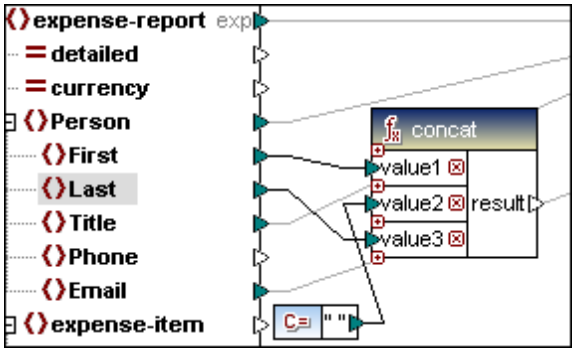


6. En el componente **`mf-ExpReport`** haga clic en el elemento **`Last`** y arrastre una línea de conexión hasta el icono "**+**" de la función **`concat`**, justo debajo de la entrada **`value2`**. El puntero del mouse cambia de forma, indicando cuándo puede "soltar" el conector.

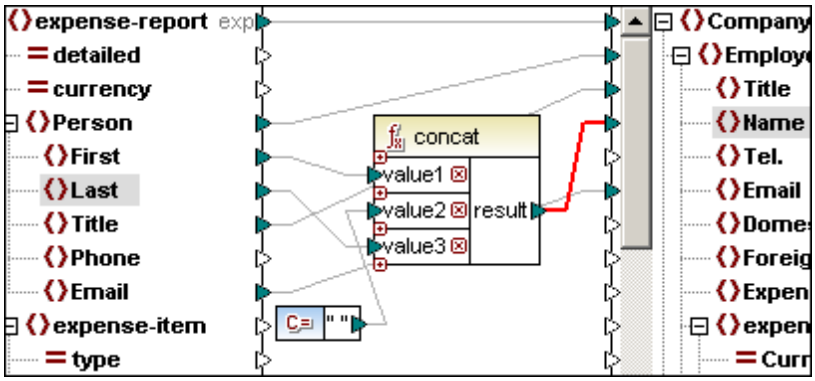


Automáticamente se amplía la función **`concat`** con un elemento más (es decir, un valor más), al que se conecta el elemento **`Last`**.

 © 2014 Altova GmbH



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



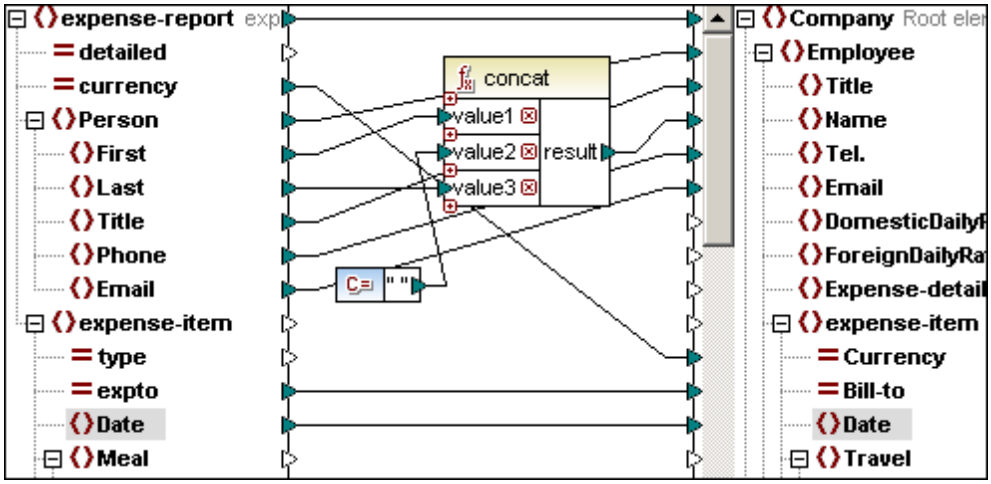
8. Cambie al panel **Resultados** para ver el resultado de la asignación actual.

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.

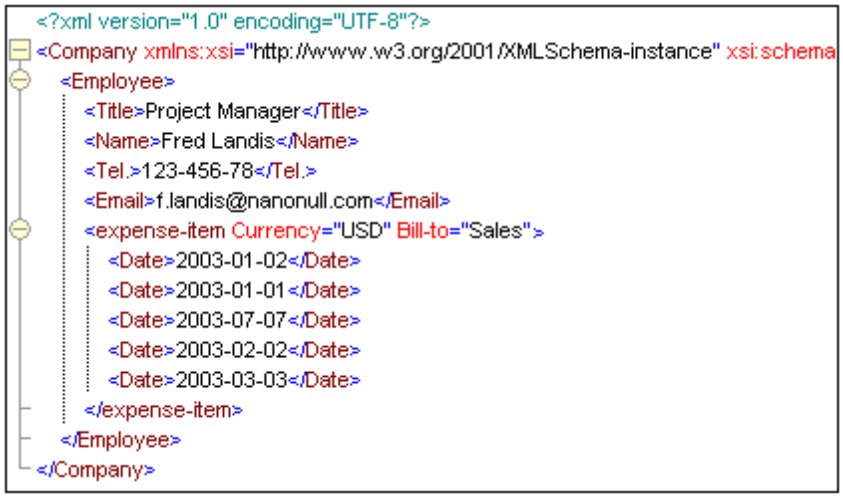
Asignación de los demás datos personales

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

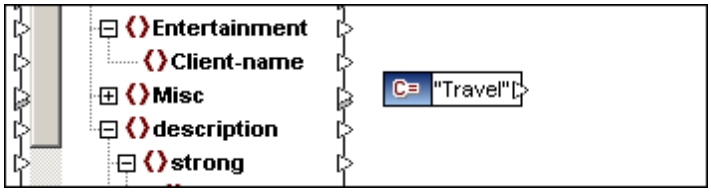
5.2.3 Filtrado de datos

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:

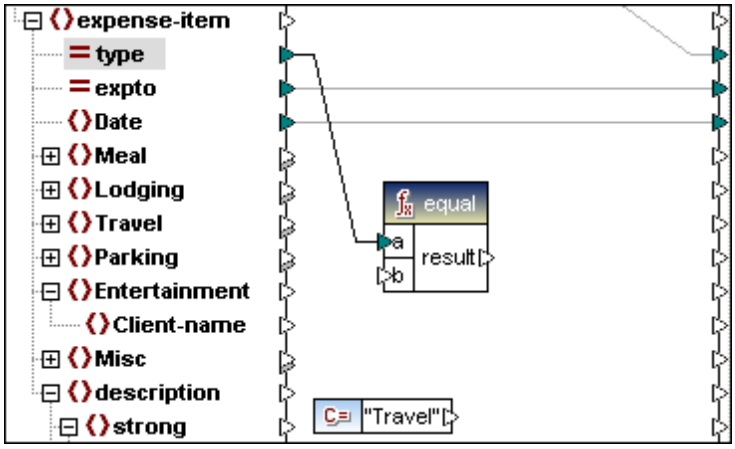
- Usar la función **`equal`** para probar el valor de un elemento de origen
- Usar una **constante** para suministrar la cadena de comparación que debe probarse
- Usar el **componente de filtrado** que pasa los datos de **`Travel`**, si el valor de entrada **`bool`** da **`true`** como resultado
- Colocar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

Para filtrar los datos:

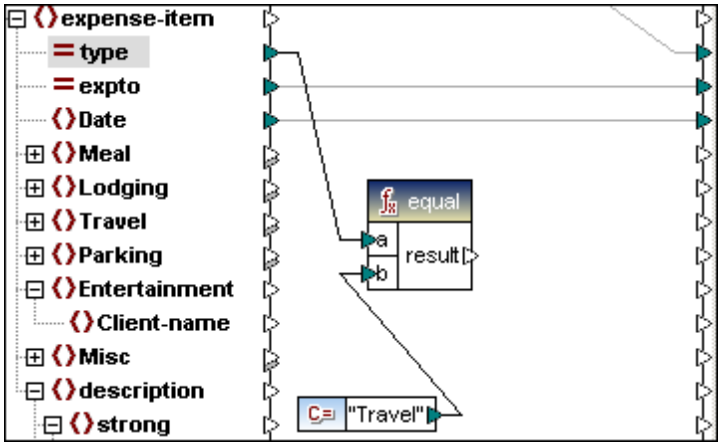
1. Haga clic en el icono **Insertar constante** e introduzca la cadena **`Travel`** en el campo de entrada.



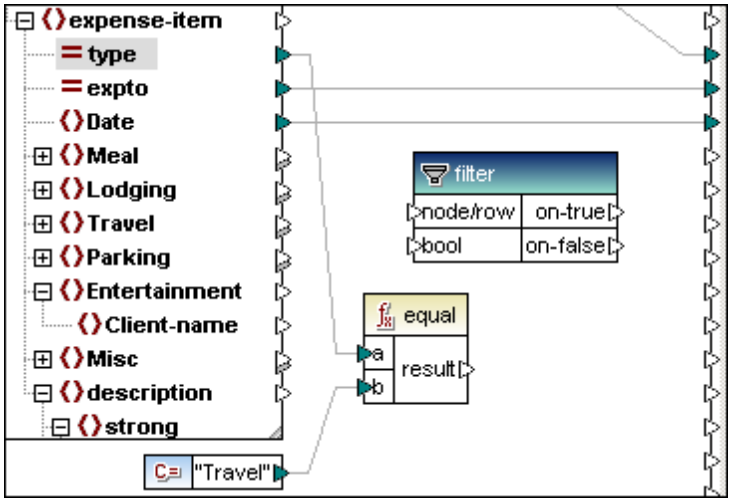
2. En la ventana Bibliotecas, expanda el grupo **logical functions** de la biblioteca **core** y arrastre la función lógica **`equal`** hasta el panel **Asignación**.
3. Conecte el elemento **`type`** (de **`expense-item`**) del esquema de origen con el parámetro **`a`** de la función **`equal`**.



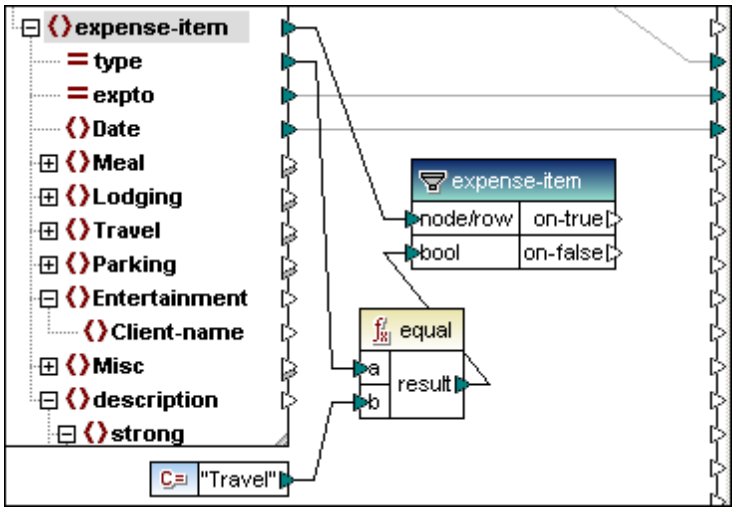
4. Conecte el icono de salida de la **constante** "Travel" con el parámetro **`b`** de la función **`equal`**.



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



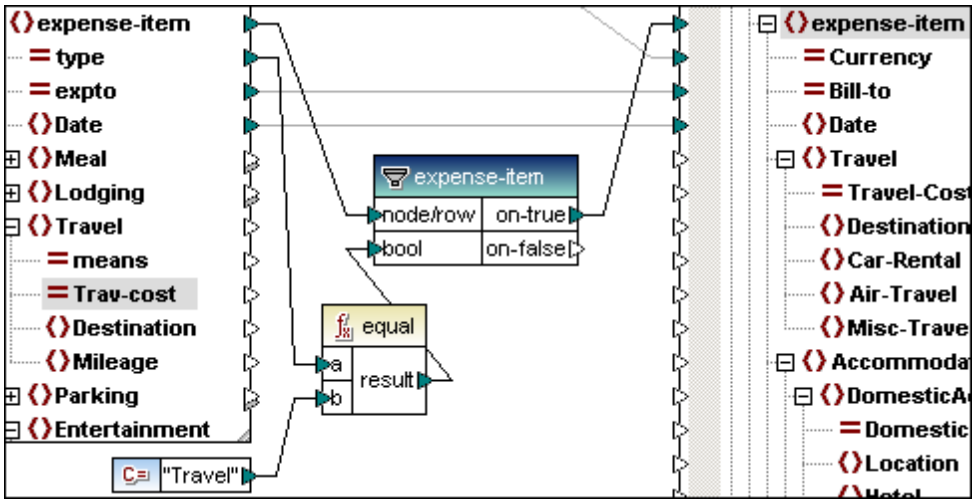
6. Conecte el icono de salida **`result`** de la función **`equal`** con el parámetro **`bool`** del componente **`filter`**.
7. Conecte el elemento **`expense-item`** del esquema de origen con el parámetro **`node/row`** del componente **`filter`**.



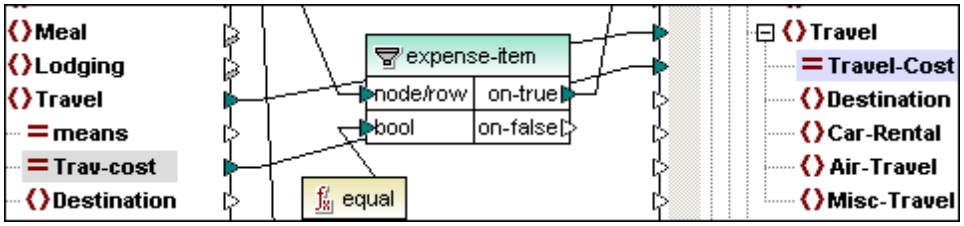
Observe cómo el nombre del componente de filtrado ahora pasa a ser **`expense-item`**.

 © 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

5.3 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>`).

También puede obtener una vista previa del código XSLT generado en MapForce (consulte el apartado *Obtener una vista previa del código XSLT*).

Transformar el archivo XML
La carpeta donde se guarda el archivo XSLT también contiene un archivo por lotes llamado **`DoTransform.bat`** que usa RaptorXML Server 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.

5.4 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
- Ver los resultados de cada componente de destino
- Generar código XSLT 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.

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.

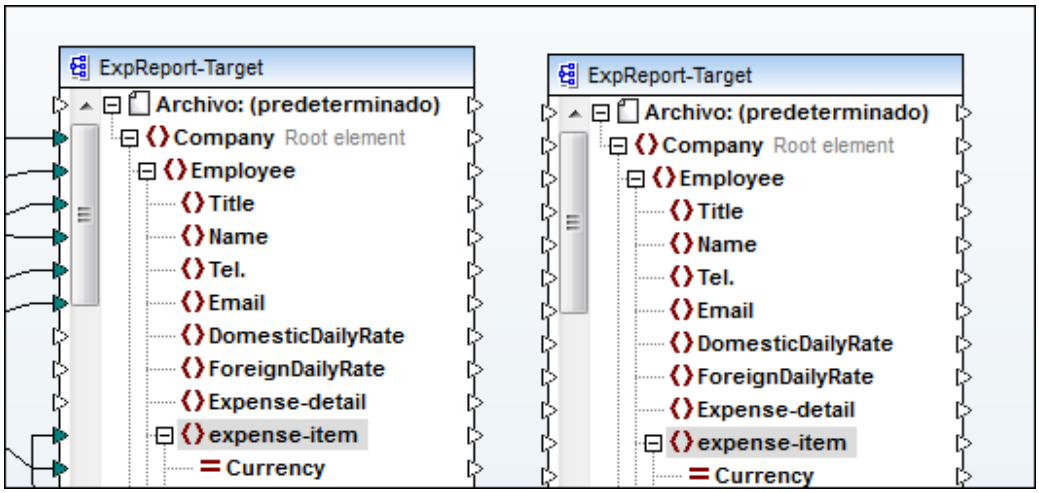
 © 2014 Altova GmbH

5.4.1 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).

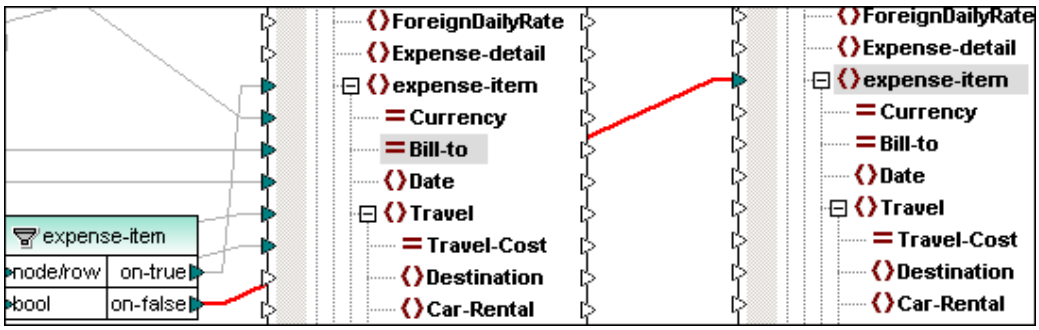
Para crear el segundo componente de esquema de destino:

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`**).

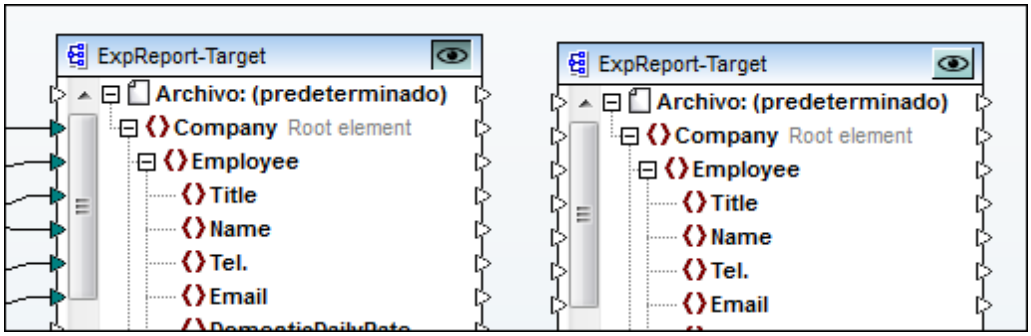


Para filtrar los datos que no son gastos de viaje y eliminarlos:

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**, o **Resultados**.

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**:

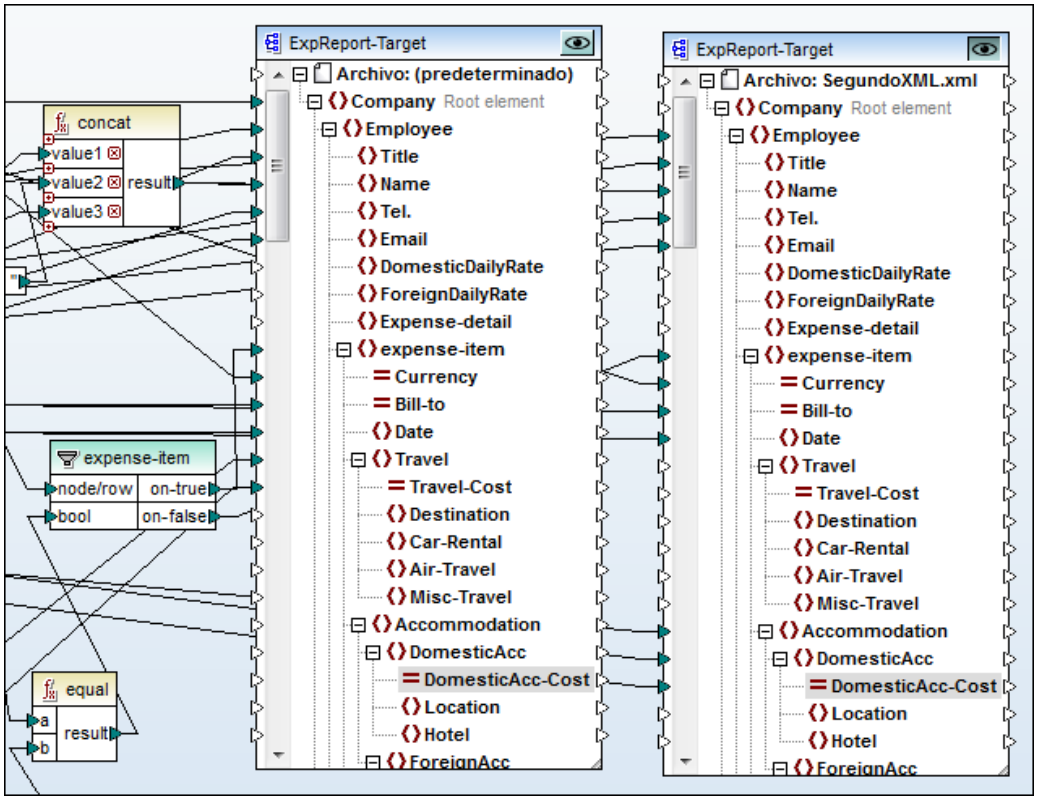
- Entre **`result`** y **`Name`**

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`**

 © 2014 Altova GmbH

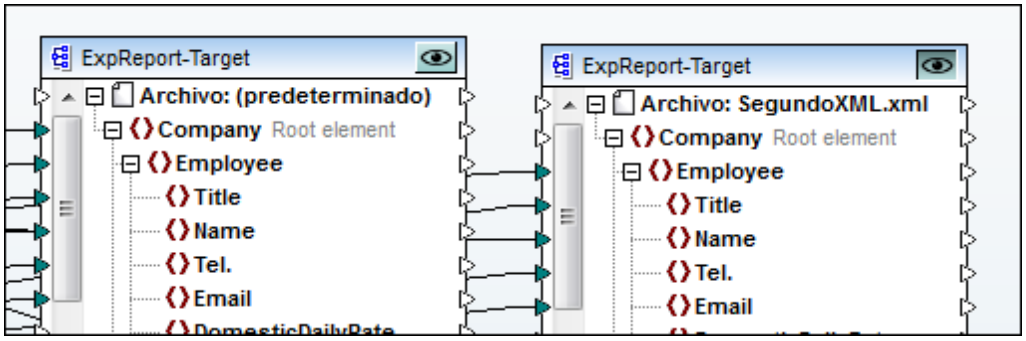

ExpReport-Target
Archivo: (predeterminado)
Company Root element
Employee
Title
Name
Tel.
Email
DomesticDailyRate
ForeignDailyRate
Expense-detail
expense-item
Currency
Bill-to
Date
Travel
Travel-Cost
Destination
Car-Rental
Air-Travel
Misc-Travel
Accommodation
DomesticAcc
DomesticAcc-Cost
Location
Hotel
ForeignAcc
concat
value1
value2
value3
result
expense-item
node/row
bool
on-true
on-false
equal
a
b
result
ExpReport-Target
Archivo: SegundoXML.xml
Company Root element
Employee
Title
Name
Tel.
Email
DomesticDailyRate
ForeignDailyRate
Expense-detail
expense-item
Currency
Bill-to
Date
Travel
Travel-Cost
Destination
Car-Rental
Air-Travel
Misc-Travel
Accommodation
DomesticAcc
DomesticAcc-Cost
Location
Hotel
ForeignAcc

5.4.2 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.

5.5 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

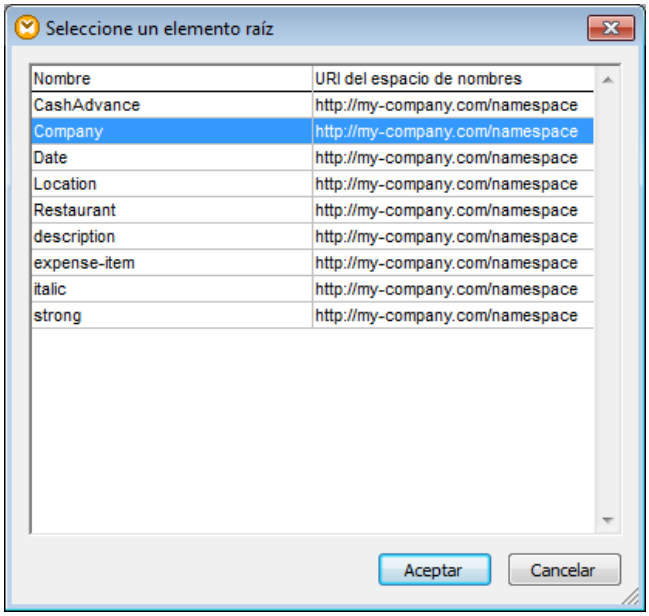
\MapForceExamples\Tutorial.

5.5.1 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`.

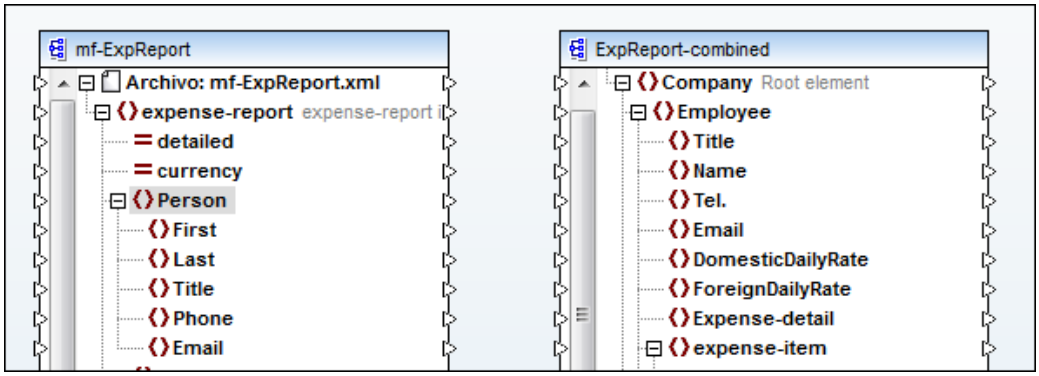
Para crear el entorno de asignación de datos:

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

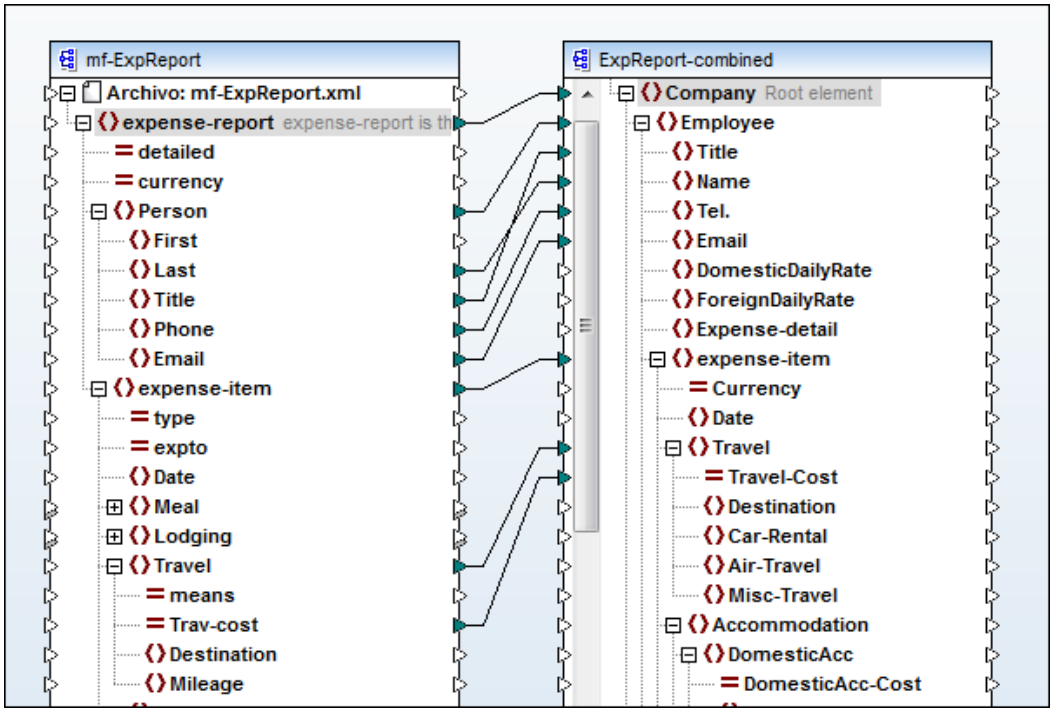


Asignación de los componentes

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`**.

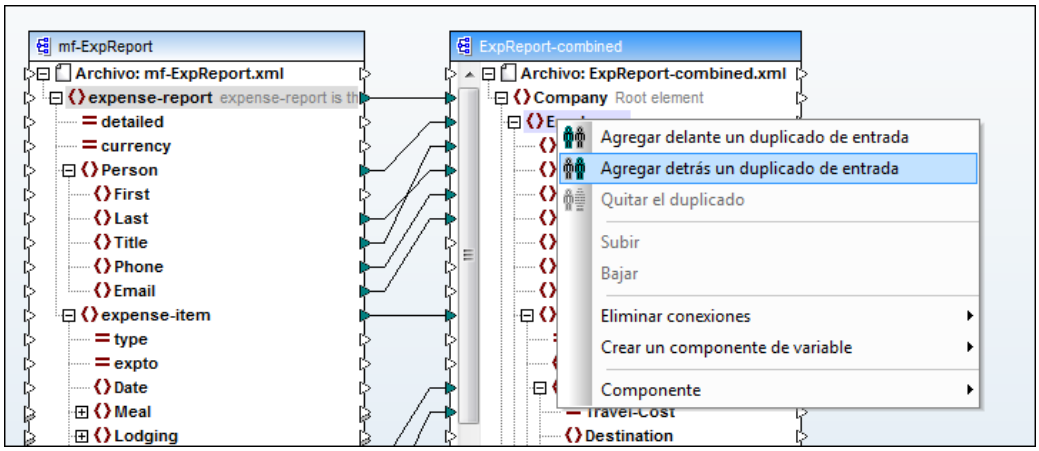
© 2014 Altova GmbH

5.5.2 Duplicar elementos de entrada

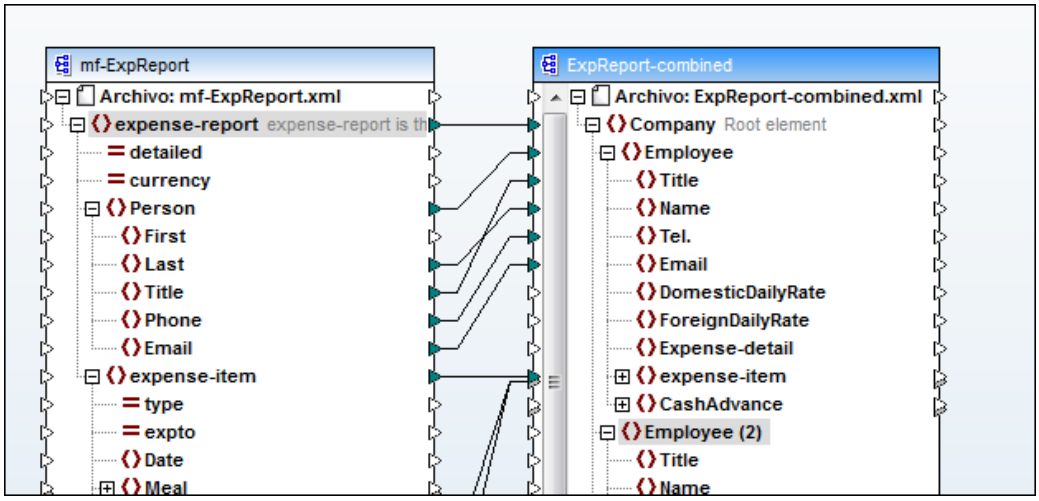
Para poder realizar asignaciones entre componentes de origen diferentes y un solo elemento de destino es necesario duplicar los **elementos de entrada** del componente de destino. Esto se consigue añadiendo un segundo archivo XML de origen y creando asignaciones desde sus elementos hasta la "misma" entrada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

Duplicar elementos de entrada:

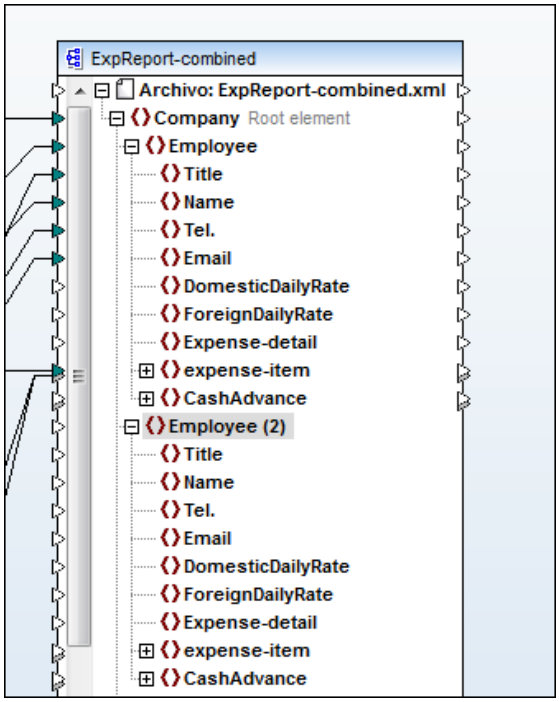
1. Haga clic con el botón secundario en el elemento **`Employee`** del archivo XML de destino.
2. En el menú contextual seleccione la opción **Agregar detrás un duplicado de entrada**.



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.



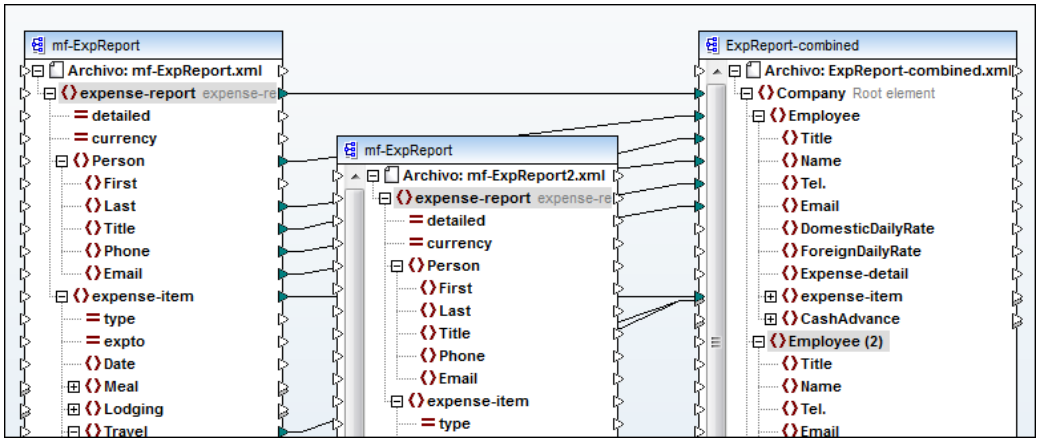
Ahora puede usar estos elementos duplicados como destino del **segundo** archivo XML de origen.

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.

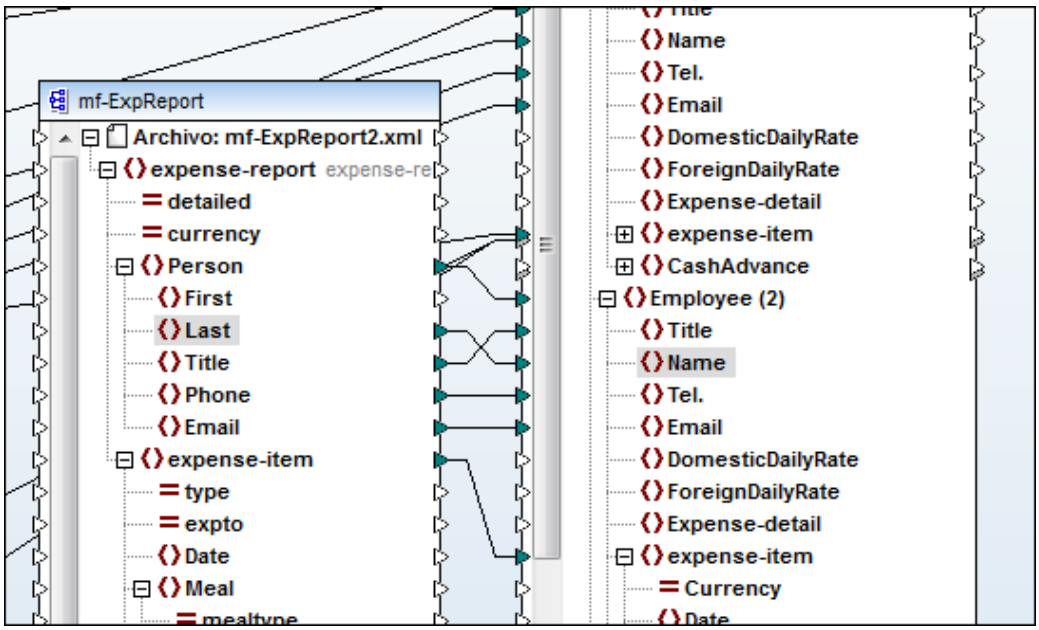
Para insertar otro componente de origen:

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:



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.

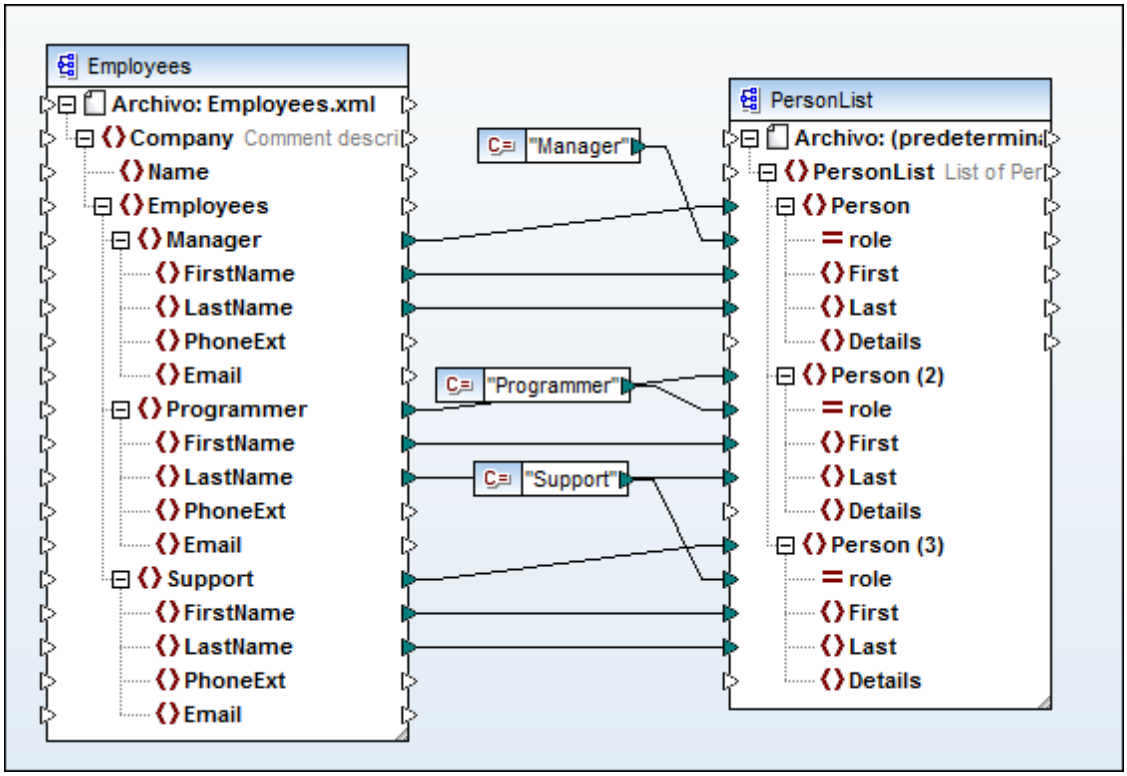
Ejemplo

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>
```

5.6 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.

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

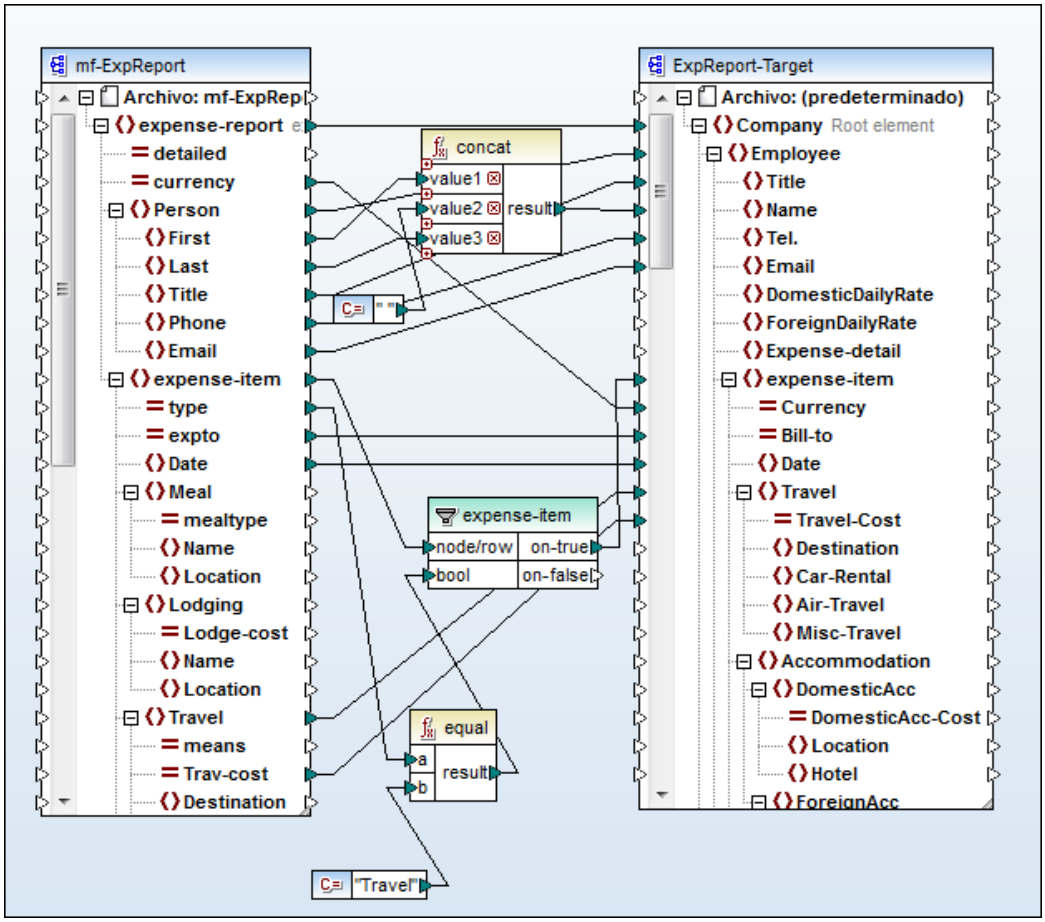
- `mf-ExpReport.xml` Archivo XML de entrada utilizado en la sección anterior
- `mf-ExpReport2.xml` El segundo archivo XML de entrada
- `Tut-ExpReport-multi.mfd` El archivo de asignación para este ejemplo

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

 © 2014 Altova GmbH

5.6.1 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

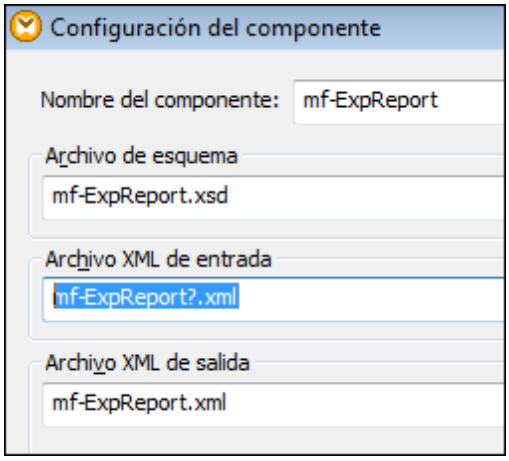
Para procesar varios archivos MapForce utiliza el carácter comodín "?" en el nombre de archivo del archivo XML de entrada. El carácter "?" puede remplazarse con un carácter o no remplazarse.

Para procesar varios archivos:

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

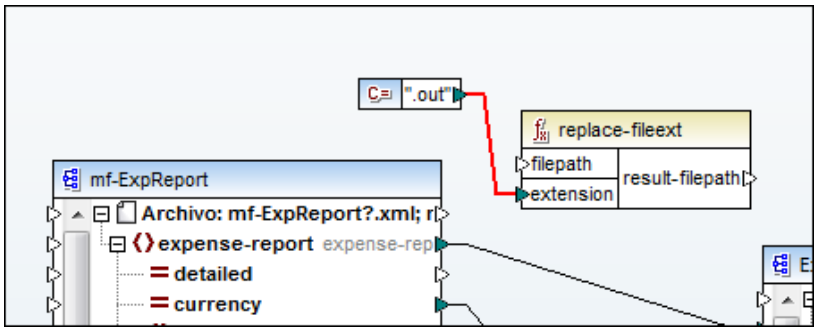
en el icono **XSLT2** de la barra de herramientas.

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*.

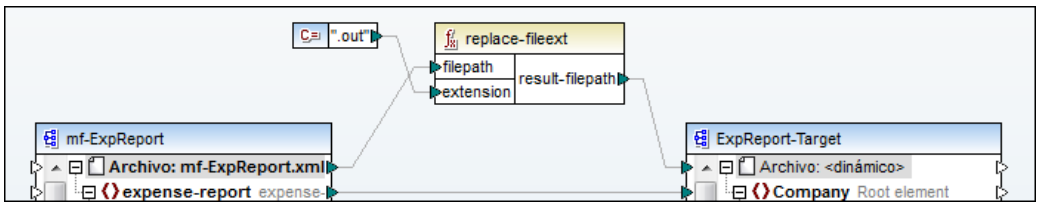


Los nombres de archivo admiten los caracteres comodín **?** y *. Observe que se introdujo una ruta de acceso relativa porque el archivo `Tut-ExpReport.mfd` está disponible en la carpeta `...\Tutorial` (si quiere puede escribir una ruta de acceso absoluta).

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

 © 2014 Altova GmbH

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

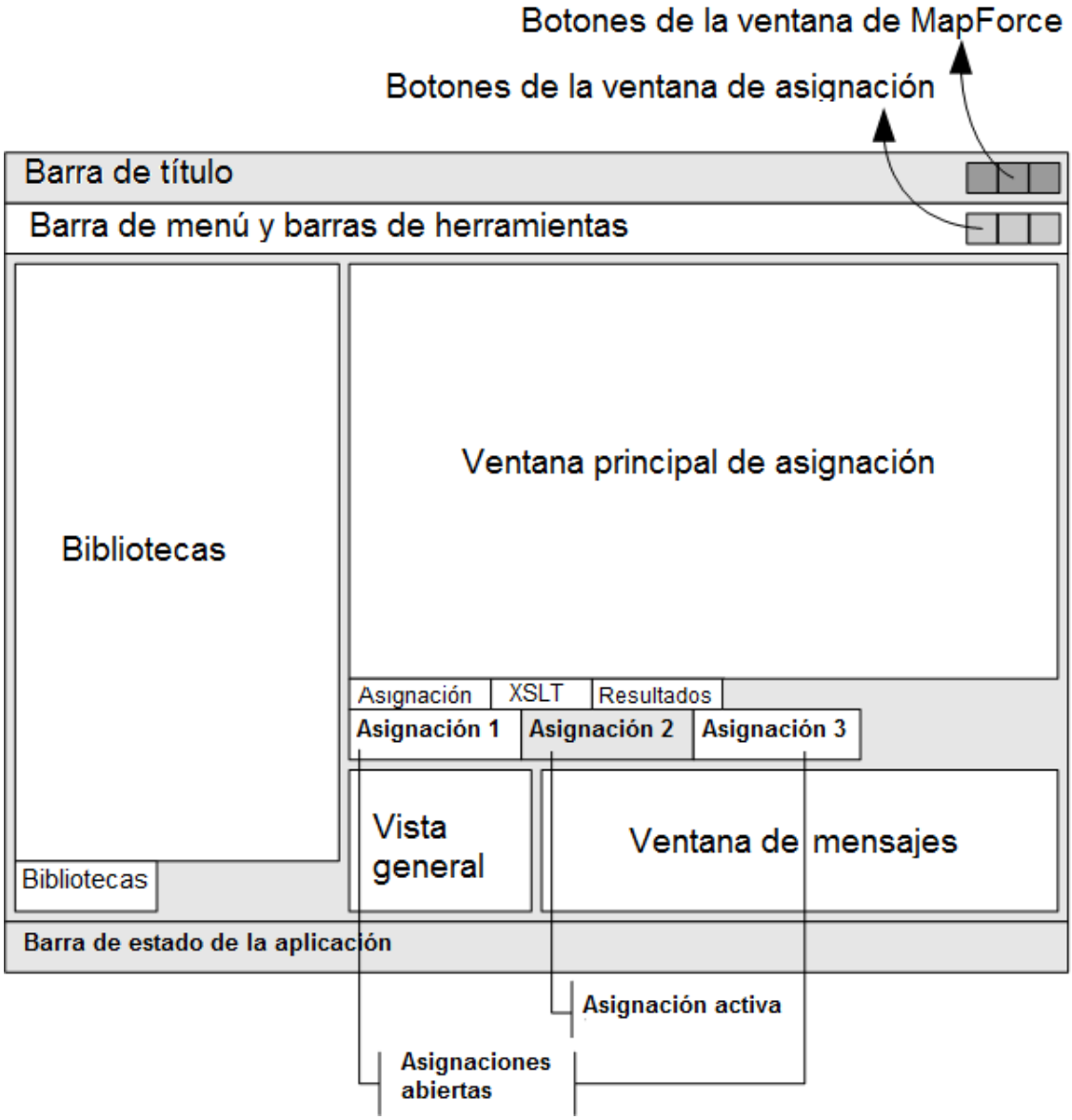
Altova MapForce 2015

Interfaz del usuario

6 Interfaz del usuario

MapForce se divide en tres partes:

- la sección izquierda, que incluye la ventana Bibliotecas
- la ventana de asignación principal (compuesta por los paneles **Asignación**, **XSLT**, **XSLT2**, 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 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

 © 2014 Altova GmbH

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.

Ventana Bibliotecas

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

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** y **XSLT2** ofrece una vista previa del código de transformación en el lenguaje seleccionado.
- 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).

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.

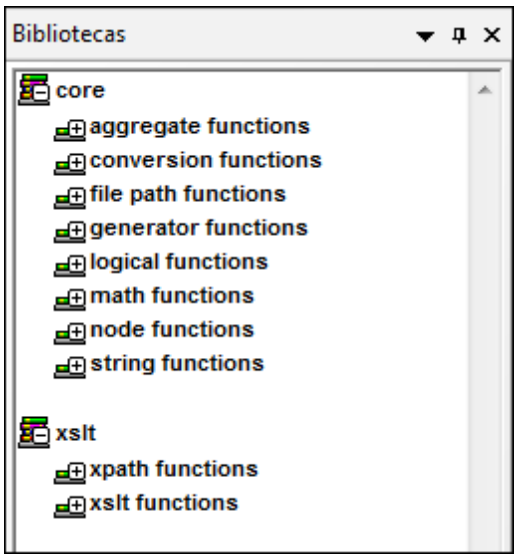
La ventana Mensajes muestra mensajes de advertencia y errores producidos durante el proceso de asignación. Si hace clic en un mensaje, el elemento afectado se resalta en la ventana de asignación.

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. 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).

6.1 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

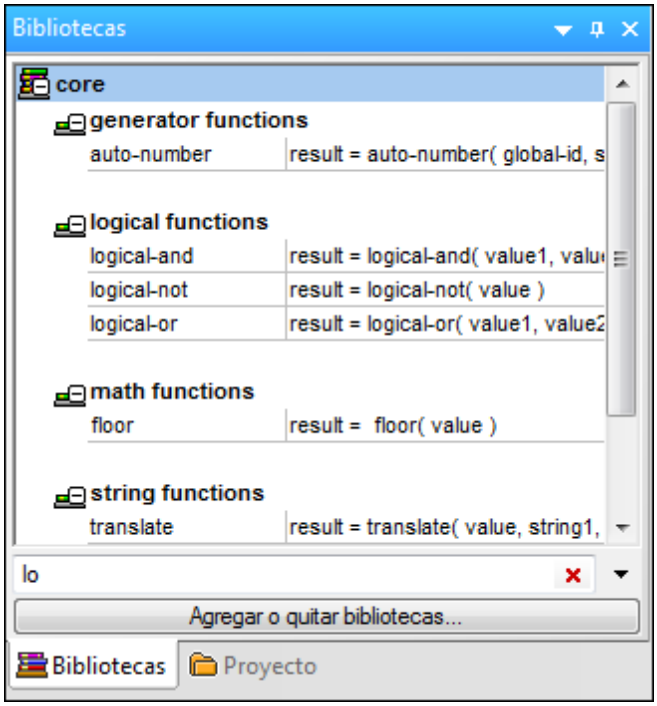
Las bibliotecas estándar **core** y **xslt** se cargan automáticamente y el usuario no necesita añadirlas. La biblioteca **core** agrupa las funciones que se pueden utilizar para producir todo tipo de datos de salida: XSLT. Las demás bibliotecas (**xslt**, **xslt2**, **xpath2**, **lang**, etc.) contienen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

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

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

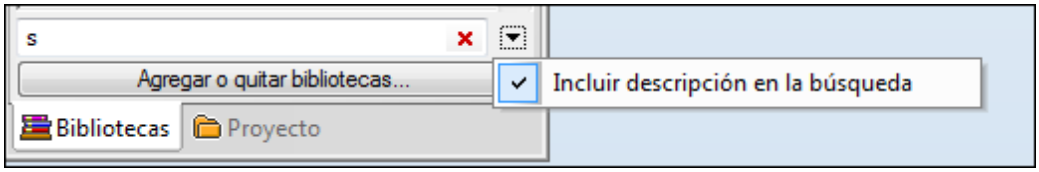


Con XSLT seleccionado

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.



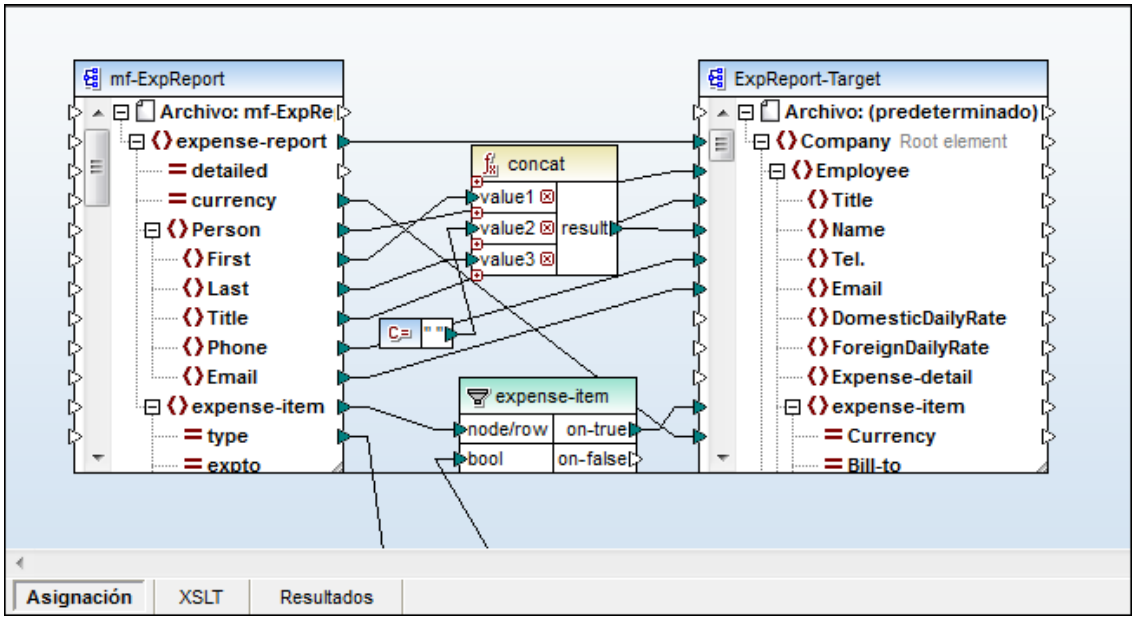
Agregar bibliotecas de funciones nuevas

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

Nota: puede definir funciones/bibliotecas personalizadas para XSLT y XSLT 2.

6.2 Panel Asignación

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



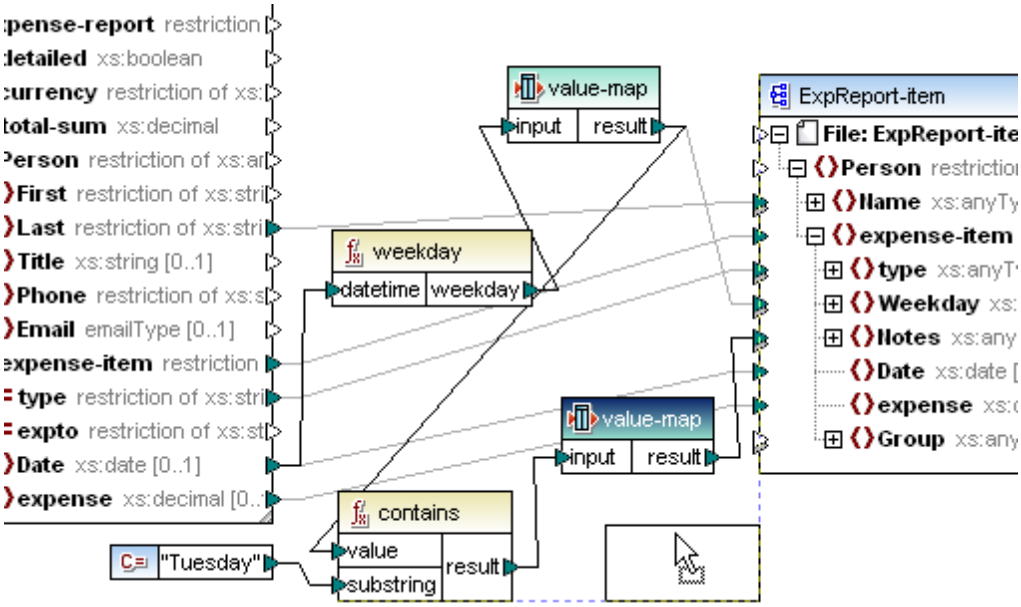
El panel **Asignación** muestra los elementos gráficos utilizados para crear la asignación (transformación) entre los componentes. Los conectores conectan los iconos de entrada y salida de cada elemento del esquema. Los elementos del esquema pueden ser atributos o elementos.

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

6.3 Panel XSLT / XSLT2

El panel **XSLT / XSLT2** 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 | 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

6.4 Panel Resultados

El panel **Resultados** muestra una vista previa del resultado de la ejecución de la asignación. El resultado que aparece en este panel depende de qué lenguaje de transformación se seleccione para la ejecución.

```
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:/Documents/Altova/MapForce2013/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>
```

Asignación | XSLT | **Resultados**

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.

```
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** .

Teclas de acceso rápido

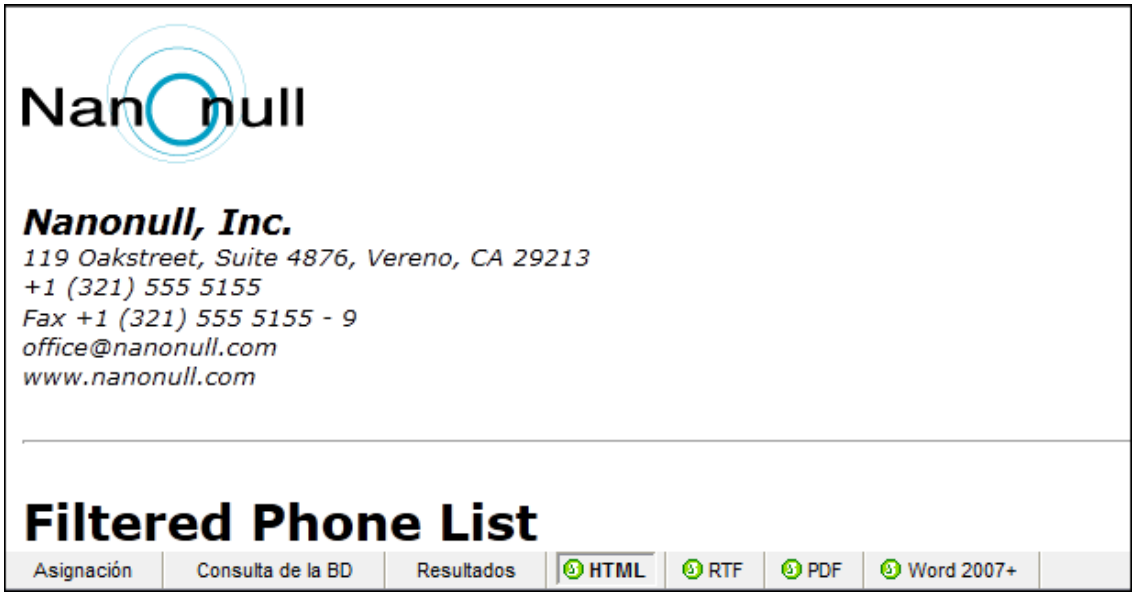
En el panel **Resultados** puede usar estas teclas de acceso rápido del teclado y del teclado numérico

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

© 2014 Altova GmbH

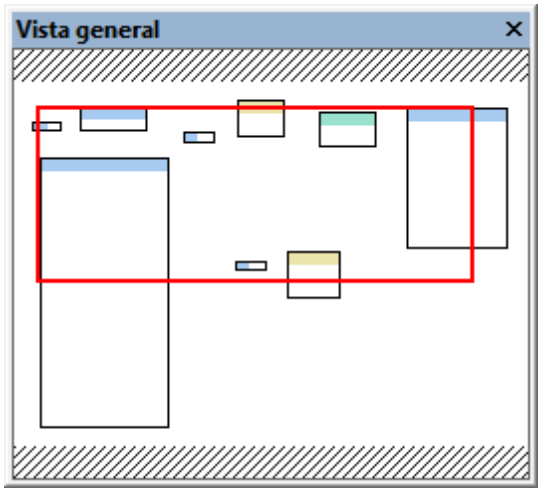
6.5 Paneles relacionados con StyleVision

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.



6.6 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**.

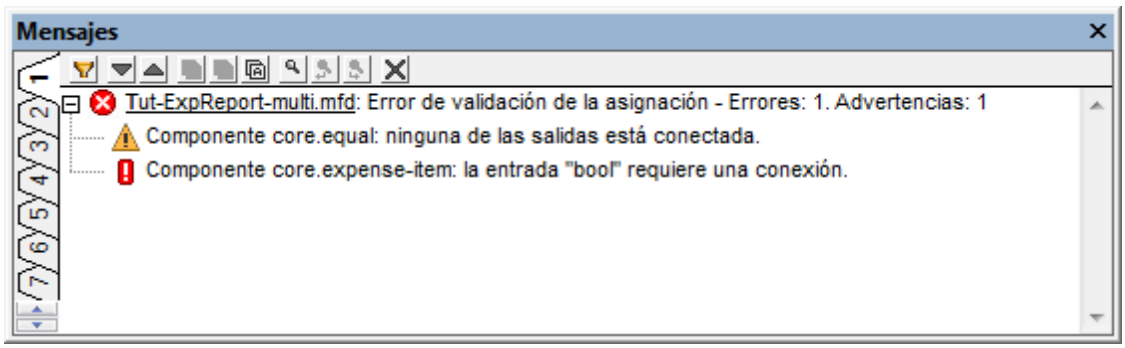


Cuando se hace clic en la ventana Vista general, el recuadro rojo muestra la parte central de la asignación.

© 2014 Altova GmbH

6.7 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 2015

Funcionamiento general

7 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
- Seleccionar el lenguaje de transformación
- Obtener una vista previa del resultado de la transformación
- Validar la asignación y el resultado de la asignación
- Parámetros de la línea de comandos
- Usar los archivos de catálogo

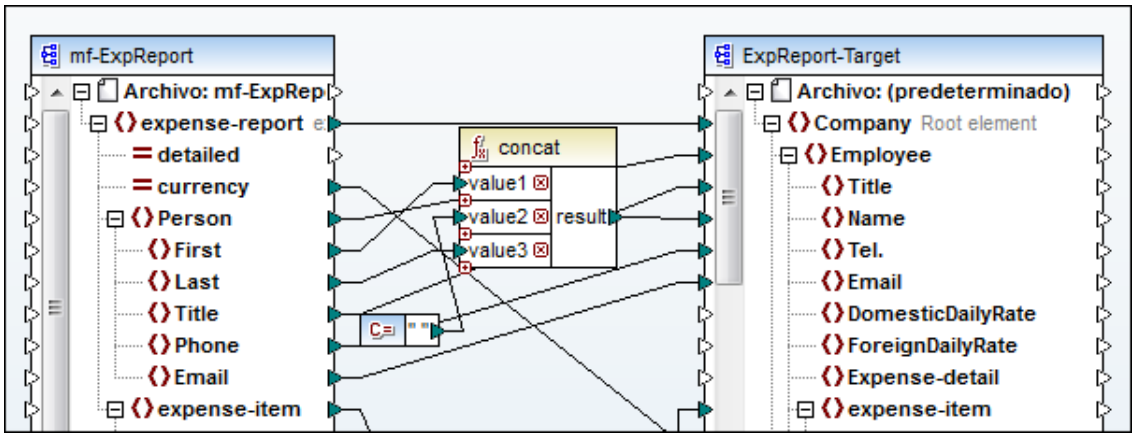
© 2014 Altova GmbH

7.1 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.

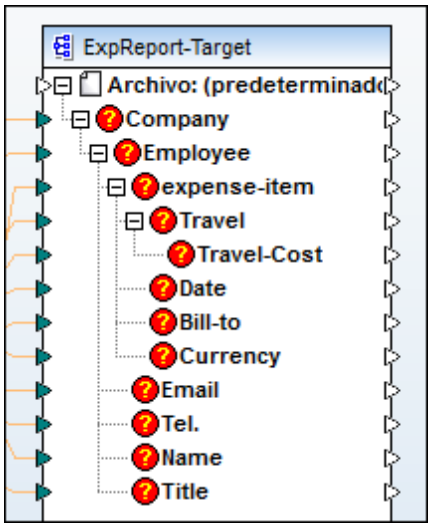
Esta característica es muy práctica si, por ejemplo, tiene una asignación y cambia el elemento raíz del esquema de destino. Esto le obligaría a crear otra vez las conexiones de los elementos secundarios a mano.

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



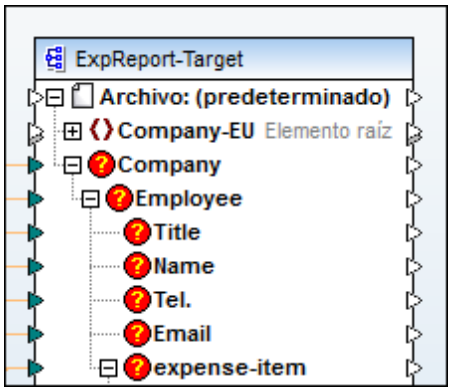
Si por cualquier motivo, el elemento raíz del esquema de destino cambia (a **`Company-EU`**, por ejemplo) recibimos un mensaje avisando de que el archivo ha cambiado.

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.

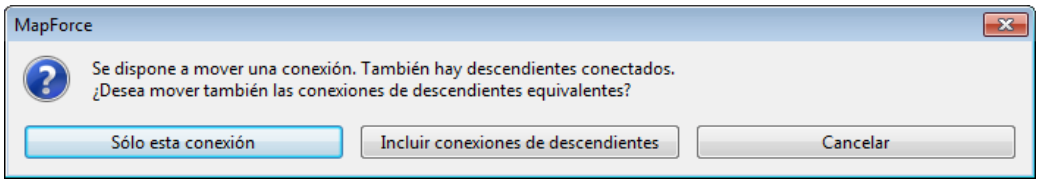
3. Seleccione el nuevo elemento raíz (**`Company-EU`**) y haga clic en **Aceptar**.



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

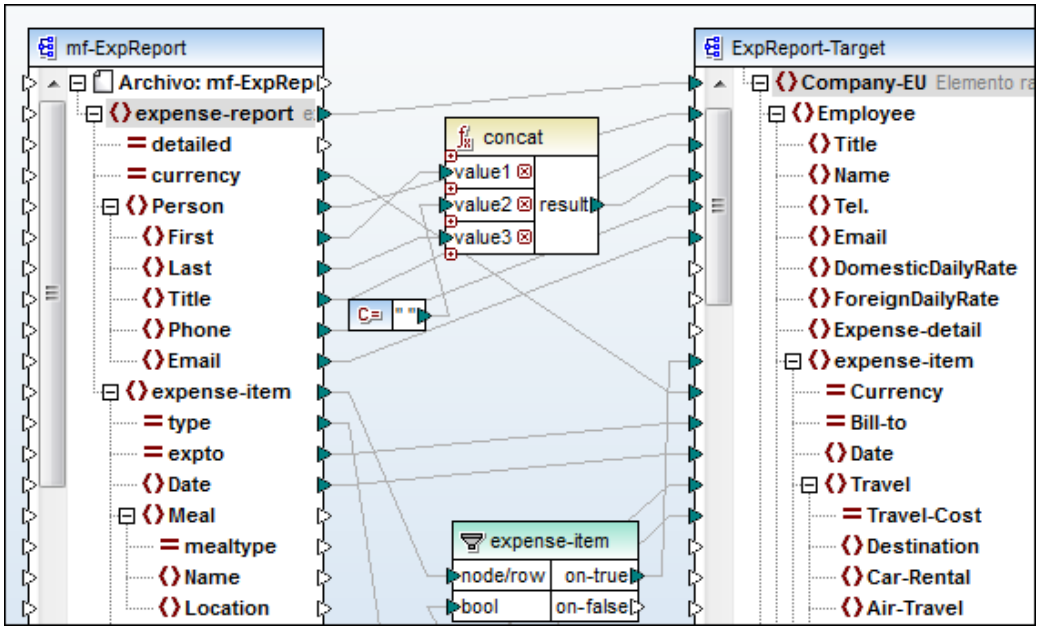
4. Haga clic en el conector del elemento **`Company`** y arrástrelo hasta el nuevo elemento raíz **`Company-EU`**.

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.

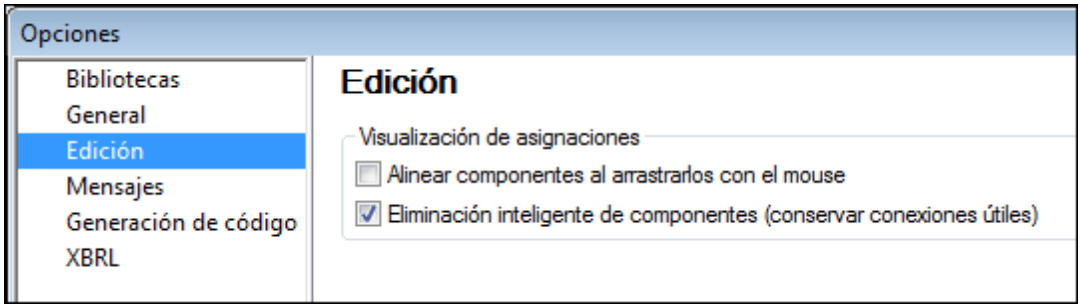
También puede cambiar de elemento raíz haciendo clic en el componente y seleccionando la opción de menú **Componente | Cambiar de elemento raíz**.

Conservar conectores tras la eliminación de componentes

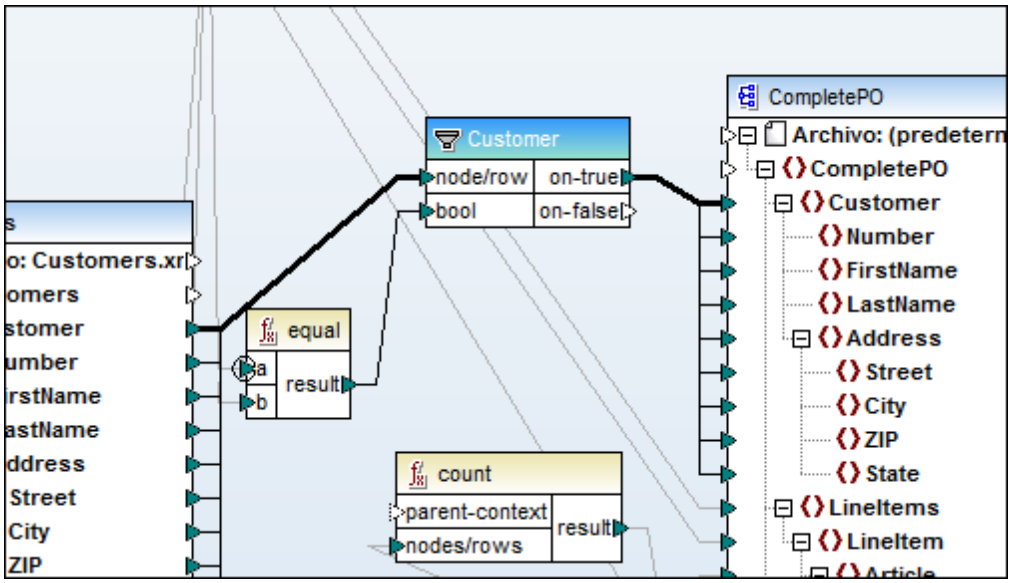
Ahora puede decidir qué ocurre cuando se elimina un componente que tiene varias conexiones (secundarias) con otro componente, como por ejemplo un filtro o componente de ordenación. Esta opción es muy práctica porque permite conservar todos los conectores secundarios, en lugar de tener que restaurarlos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

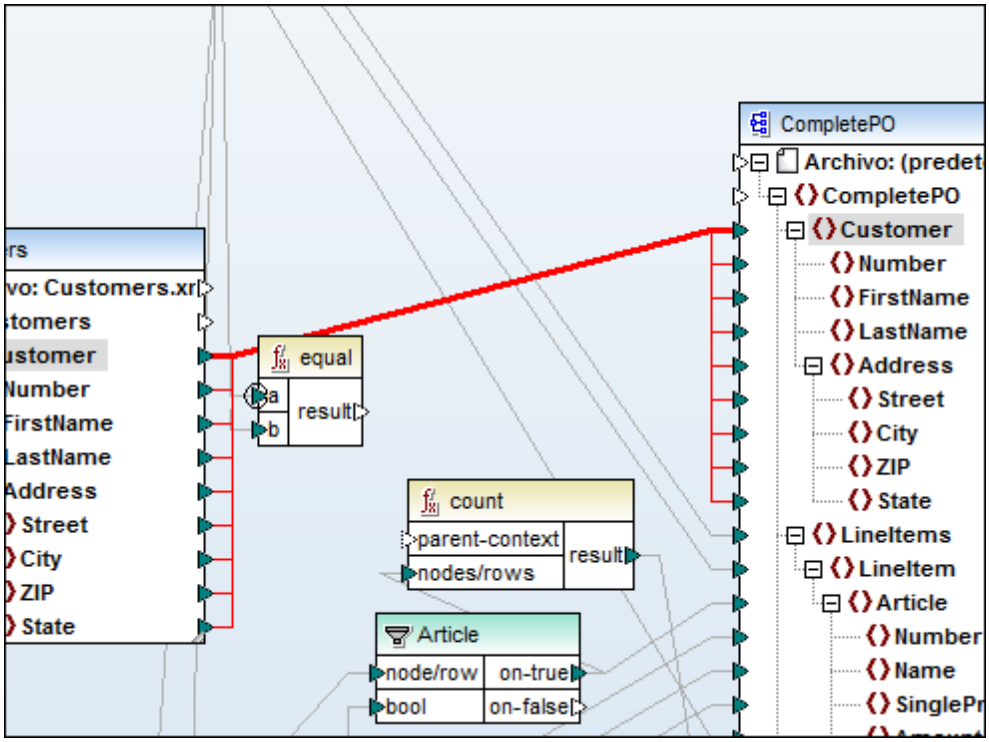


Por ejemplo, tome la asignación `CompletePO.mfd` de la carpeta `...\MapForceExamples` y active la casilla *Eliminación inteligente de componentes (conservar conexiones útiles).* Observe que el filtro **`Customer`** es una conexión de copia total que tiene muchos elementos secundarios conectados.



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.

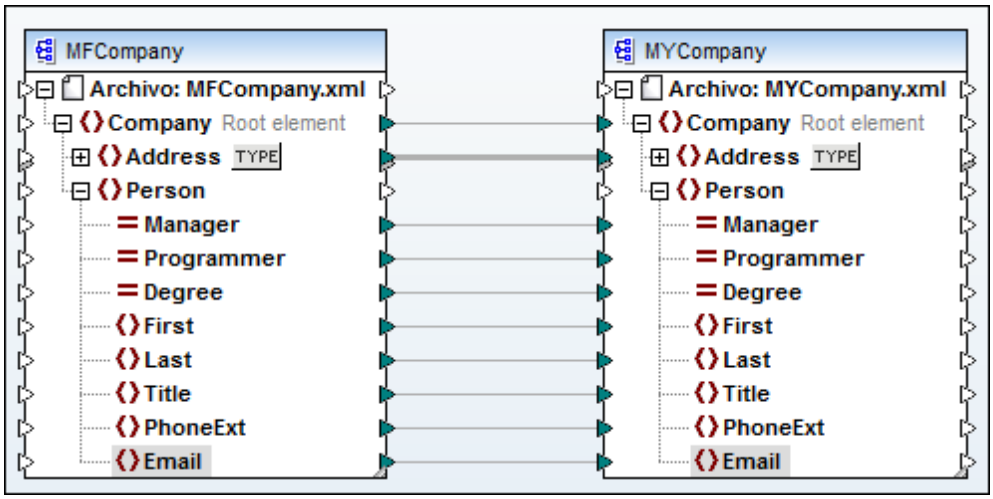
7.2 Elementos desaparecidos

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`.

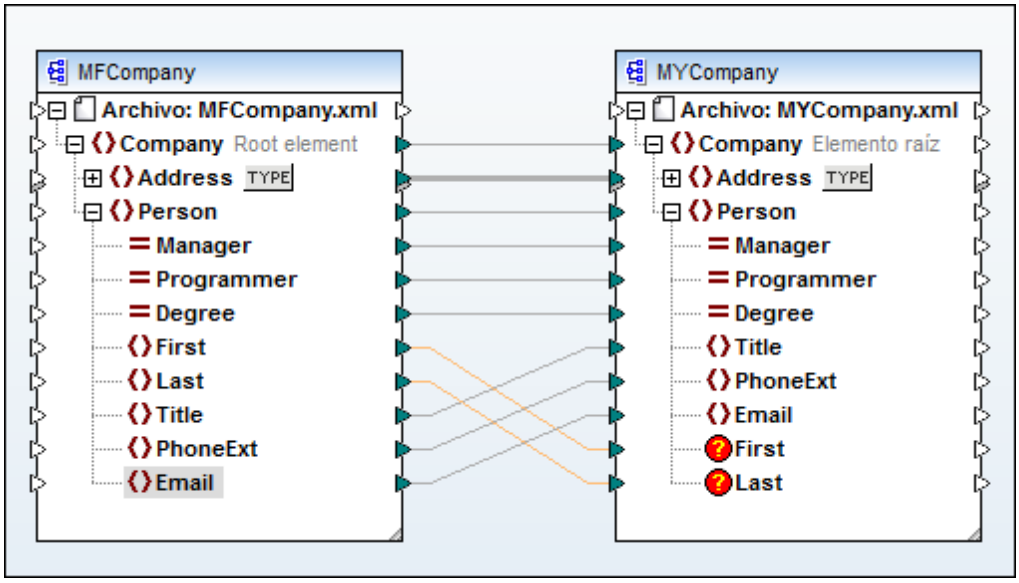
Creamos un componente de origen con el esquema `MFCompany.xsd` y el archivo de instancia `MFCompany.xml`. Después creamos un componente de destino con el esquema `MyCompany.xsd` y el archivo de instancia `MyCompany.xml`.

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.

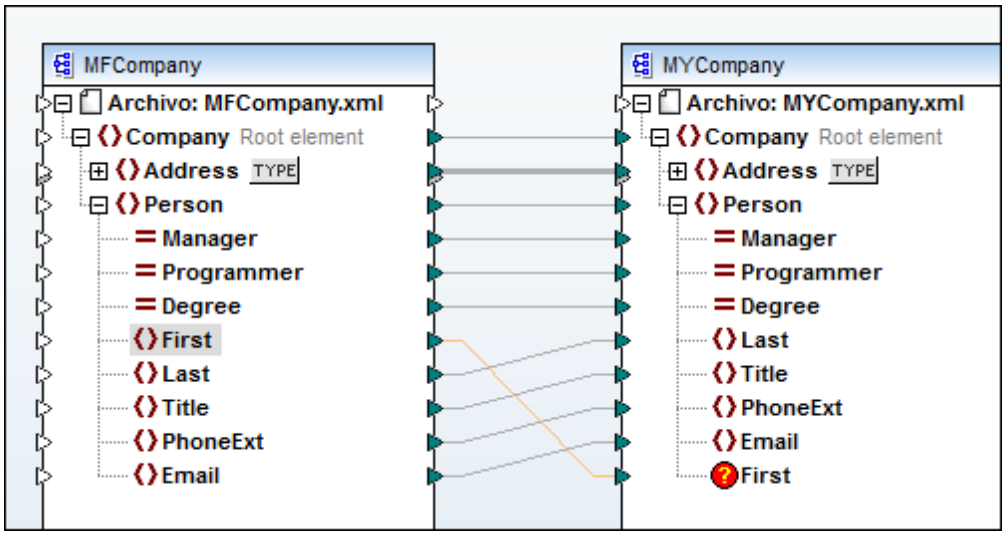
 © 2014 Altova GmbH



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.

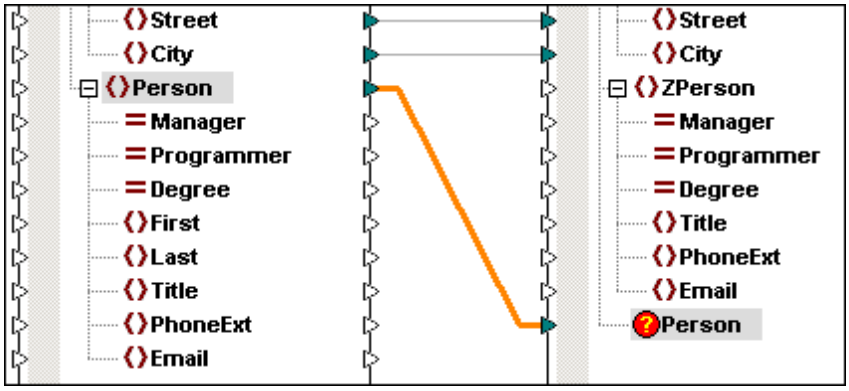
Recuerde que puede ver una vista previa de la asignación (o generar código), pero si lo hace en este momento, en la ventana Mensajes aparecen advertencias. Durante la generación de código y en la vista previa de resultados se ignoran los elementos que faltan.

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**).

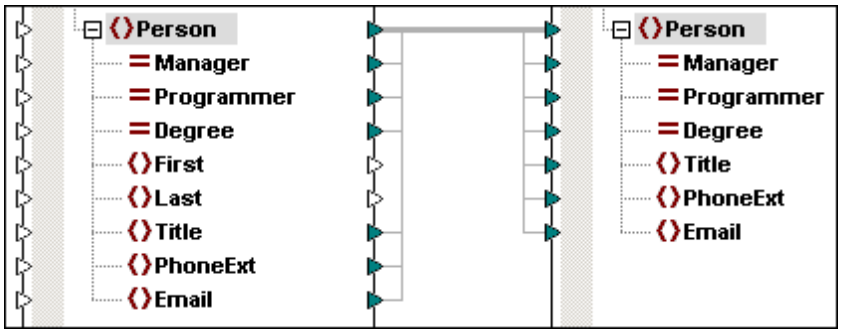


Qué ocurre cuando un elemento cambia de nombre

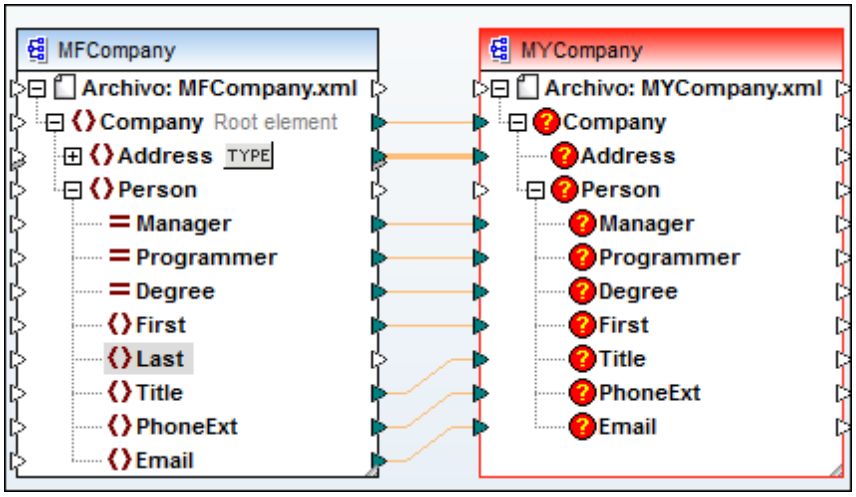
Si se cambia el nombre de un elemento primario (p. ej. imaginemos que ahora **`Person`** se llama **`ZPerson`**), entonces el conector original del elemento primario se conserva y los elementos secundarios y sus conectores se eliminan.



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.

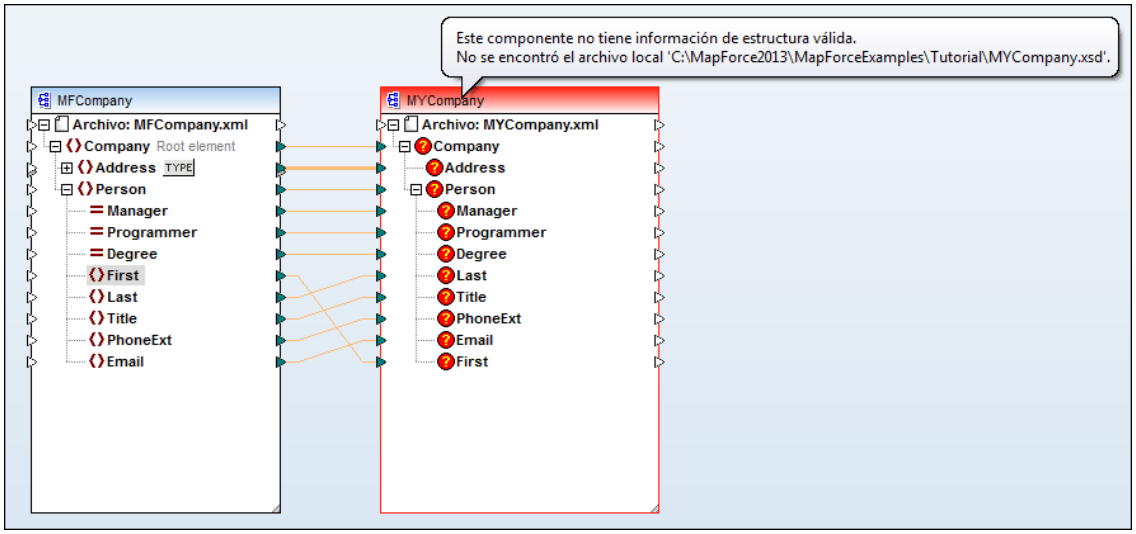


Qué ocurre cuando desaparece el origen de datos o cambia de nombre
Si el origen de datos de un componente (es decir, el esquema se elimina o cambia de nombre, entonces aparecen resaltados todos los elementos que contenía. El componente aparece ahora con un marco rojo, lo cual indica que no existe una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

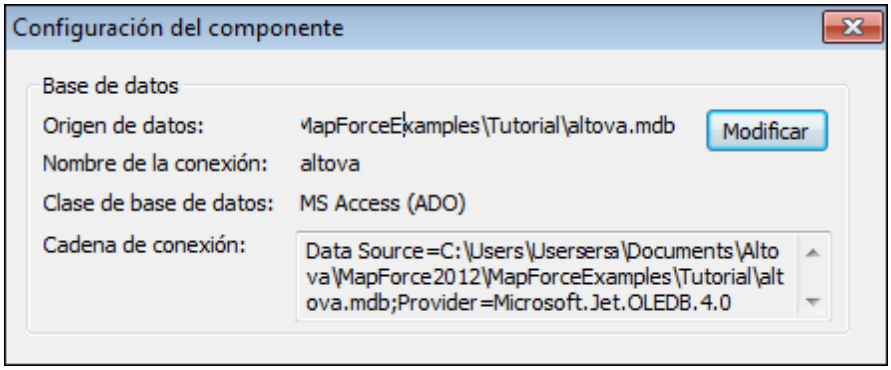


Si pasa el puntero del mouse sobre el componente que aparece en rojo, se abre un mensaje con información.

 © 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*.



Todas las conexiones válidas/correctas se conservan cuando seleccionamos un esquema que tiene la misma estructura.

7.3 Seleccionar el lenguaje de transformación

Estos son los lenguajes de transformación de datos que puede elegir:

- XSLT 1.0
- XSLT 2.0

Hay dos maneras de seleccionar el lenguaje de transformación:

- Haciendo clic en el lenguaje que quiere usar en el menú **Resultados**.
- Haciendo clic en el lenguaje que quiere usar en la barra de herramientas Selección del lenguaje.

Temas relacionados:

- Vista previa del resultado de la transformación

© 2014 Altova GmbH

7.4 Vista previa del resultado de la transformación

En MapForce puede obtener una vista previa del resultado de la asignación de datos sin necesidad de ejecutar y compilar el código generado con un procesador o compilador externo. Por lo general es buena idea obtener una vista previa del resultado de la transformación dentro de MapForce directamente antes de intentar procesar el código generado fuera de MapForce.

Para ofrecer una vista previa del resultado de la asignación MapForce ejecuta la asignación y muestra el resultado en el panel **Resultados**. Los datos del panel Resultados se pueden validar y guardar y también puede buscar texto con el comando **Buscar** (**Ctrl + F**).

Para obtener una vista previa del resultado de la transformación:

- Haga clic en el panel **Resultados** de la ventana principal de MapForce. MapForce ejecuta la asignación utilizando el lenguaje de transformación que está seleccionado en la barra de herramientas Selección del lenguaje y muestra el resultado en el panel **Resultados**.

Hay dos maneras de guardar el resultado de la transformación:

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

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

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

La vista previa también se puede configurar en la pestaña *General* del cuadro de diálogo "Opciones".

Temas relacionados:

- Seleccionar el lenguaje de transformación

7.5 Vista previa del código XSLT

Para obtener una vista previa del código XSLT generado con MapForce seleccione el lenguaje de transformación XSLT 1.0 o XSLT 2.0 (consulte el apartado *Seleccionar el lenguaje de transformación*).

Hay dos maneras de obtener una vista previa del código XSLT 1.0 (o XSLT 2.0) generado:

- Haciendo clic en el panel **XSLT** de la ventana principal de MapForce (para obtener una vista previa del código XSLT 1.0).
- Haciendo clic en el panel **XSLT2** de la ventana principal de MapForce (para obtener una vista previa del código XSLT 2.0).

Nota: el panel **XSLT** o **XSLT2** se habilita si está seleccionado el lenguaje de transformación XSLT o XSLT2 respectivamente.

© 2014 Altova GmbH

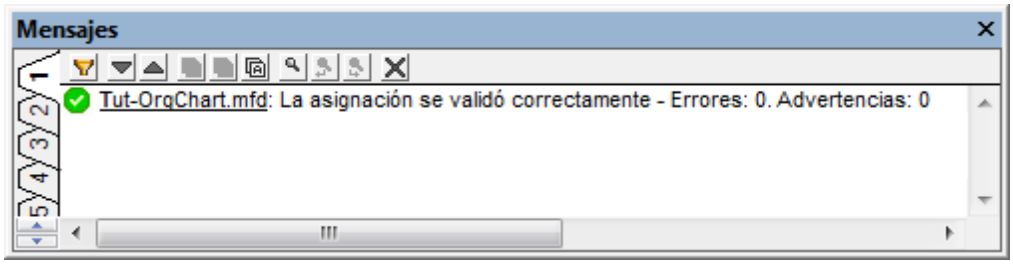
7.6 Validar la asignación y su resultado

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 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

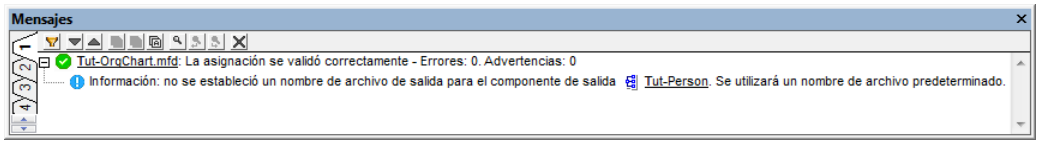
Para validar una asignación:

- 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.

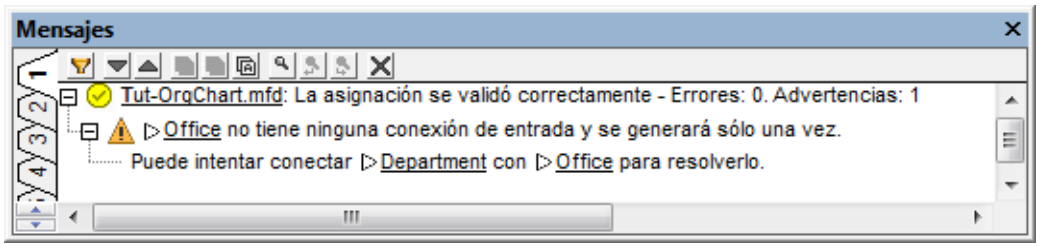
Mensajes de validación

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*

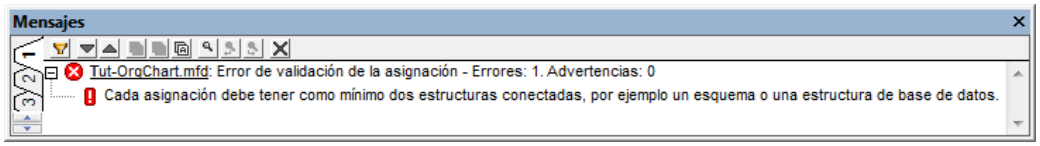


Las **advertencias** alertan de riesgos sin impedir que continúe el proceso de asignación y de vista previa de los resultados de la transformación. Por tanto, una asignación válida tiene cero errores pero puede tener varias advertencias.



- 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**.



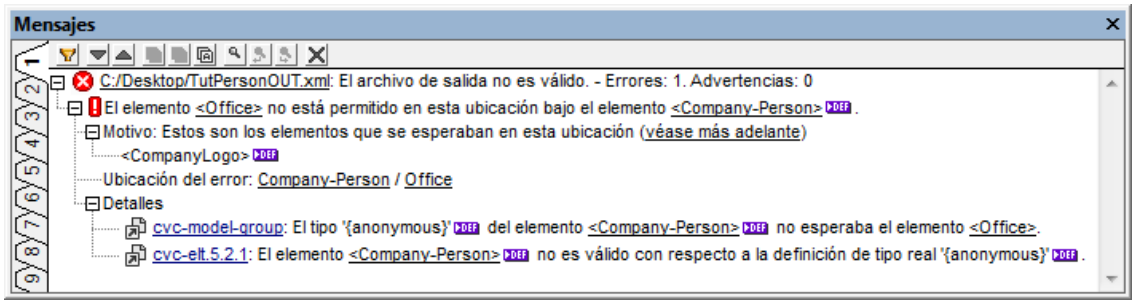
Validar los resultados de una asignación

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.

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*).

 © 2014 Altova GmbH

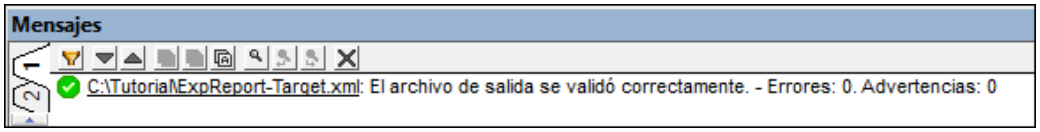


El mensaje de validación contiene algunos hipervínculos que ofrecen más información:

- 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.

7.7 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

/XSLT	genera código XSLT 1.0
/XSLT2	genera código XSLT 2.0
/GLOBALRESOURCEFILE *nombrearchivorecglob*	usa los recursos globales definidos en el archivo de recursos globales especificado
/GLOBALRESOURCECONFIG *configrecglob*	usa la configuración de recursos globales especificada

dirsalida
El directorio donde se debe poner el código de asignación generado. *Dirsalida* es opcional. Si no se aporta la ruta de salida, se usa el directorio actual/de trabajo. Si no se especifica, los nombres de archivo son relativos al directorio actual/de trabajo. Cuando decimos *relativos,* queremos decir que solamente se incluye el nombre de archivo y no la ruta de acceso completa con la letra de la unidad de disco duro.

opciones
Especifica varias opciones:

/**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

 © 2014 Altova GmbH

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*.

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 todos los archivos XSLT y usar todos los recursos globales del archivo de recursos globales de la configuración especificada
Mapforce.exe *nombrearchivo* **/XSLT** *dirsalida* **/GLOBALRESOURCEFILE** *nombrearchivorecursosglobales* **/GLOBALRESOURCECONFIG** *nombreconfiguración*

7.8 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`:

 © 2014 Altova GmbH

- `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.

`%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

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.

© 2014 Altova GmbH

7.9 Archivos SPS en la vista previa de MapForce

MapForce puede ofrecer una vista previa / representar componentes de esquema XML 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.

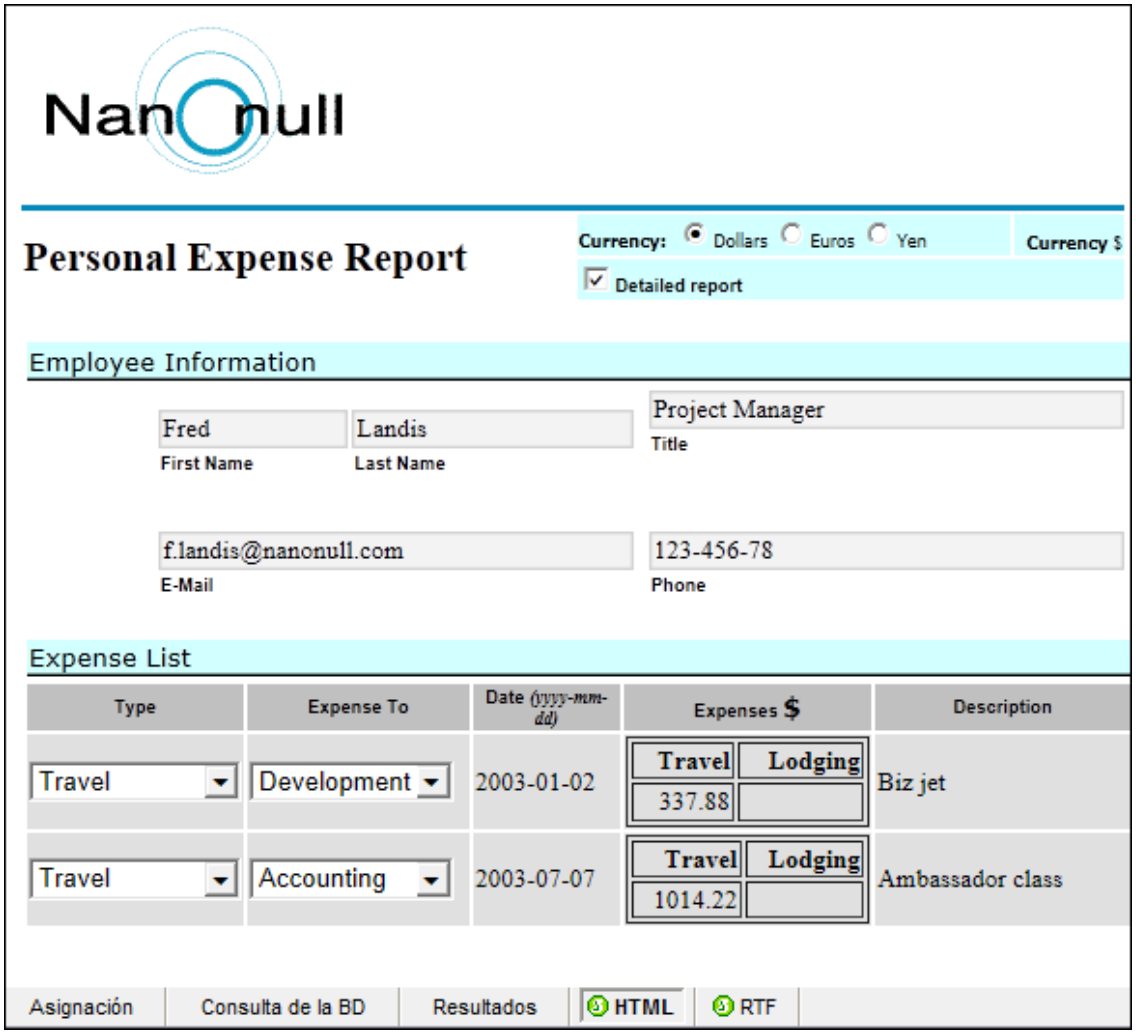
Por tanto, si StyleVision está instalado en el equipo y el componente de destino tiene asociado un archivo SPS, en MapForce hay varios paneles adicionales:

- 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.

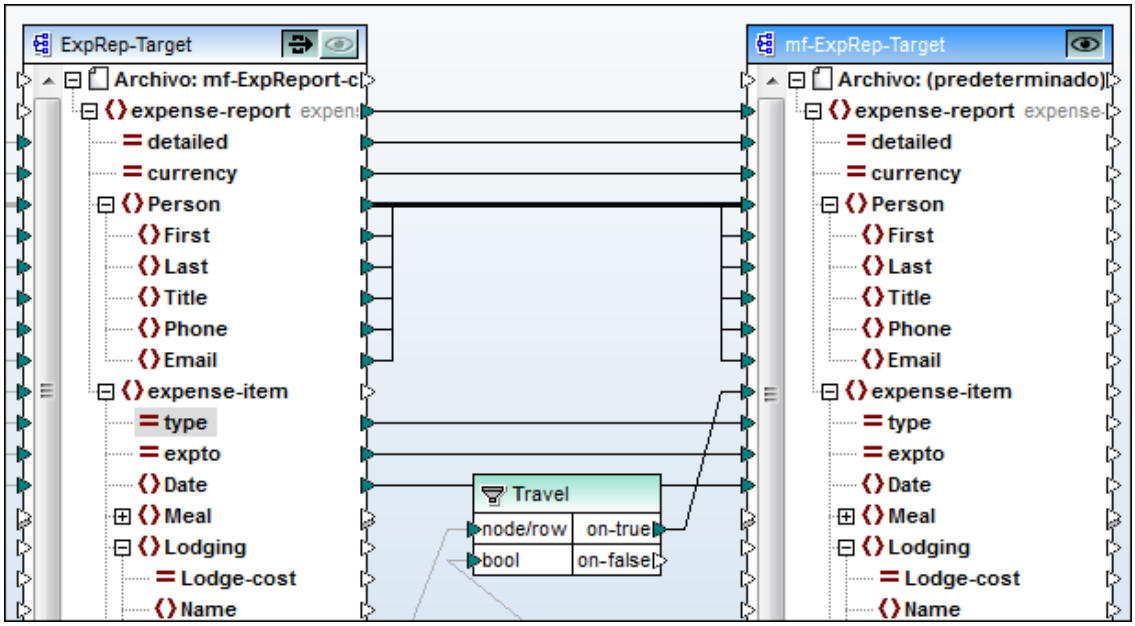
Si se trata de StyleVision Professional Edition, MapForce incluye los paneles **HTML** y **RTF** solamente.

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 .


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

7.9.1 Asignar un archivo SPS a un componente

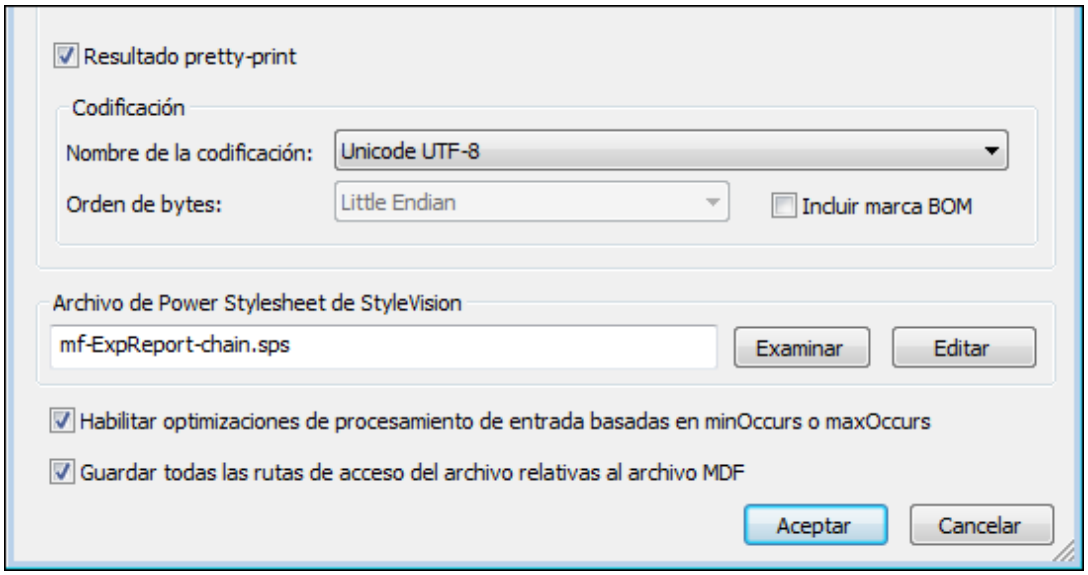
En este apartado usamos como ejemplo el archivo `Tut-ExpReport-chain.mfd`, disponible en la carpeta `...\MapForceExamples\Tutorial`. Para más información sobre asignaciones en cadena, consulte el apartado Asignaciones en cadena: componentes de paso a través.



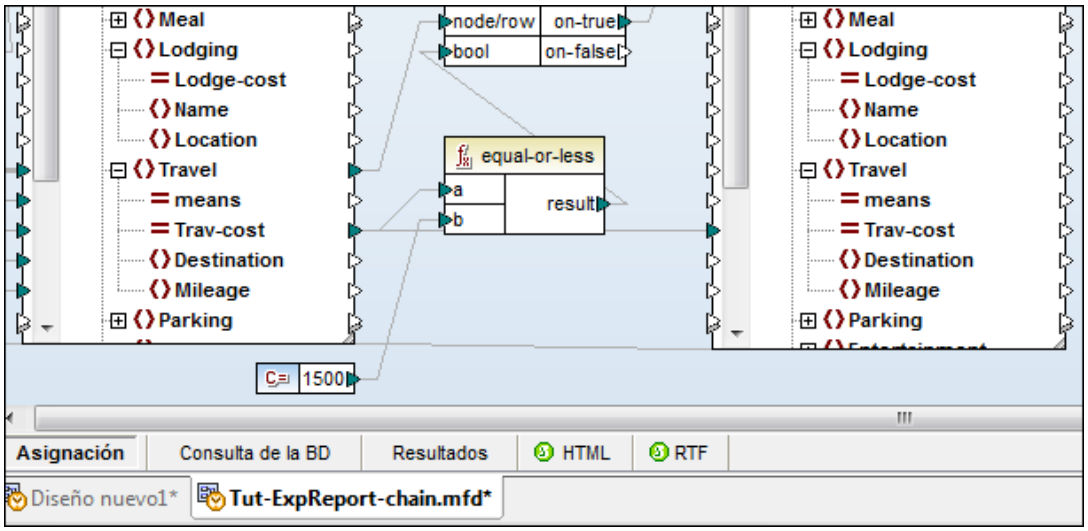
Para asignar una plantilla SPS a un componente de MapForce:

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. **mf-ExpRep-Target**). Aparece el cuadro de diálogo "Configuración del componente".

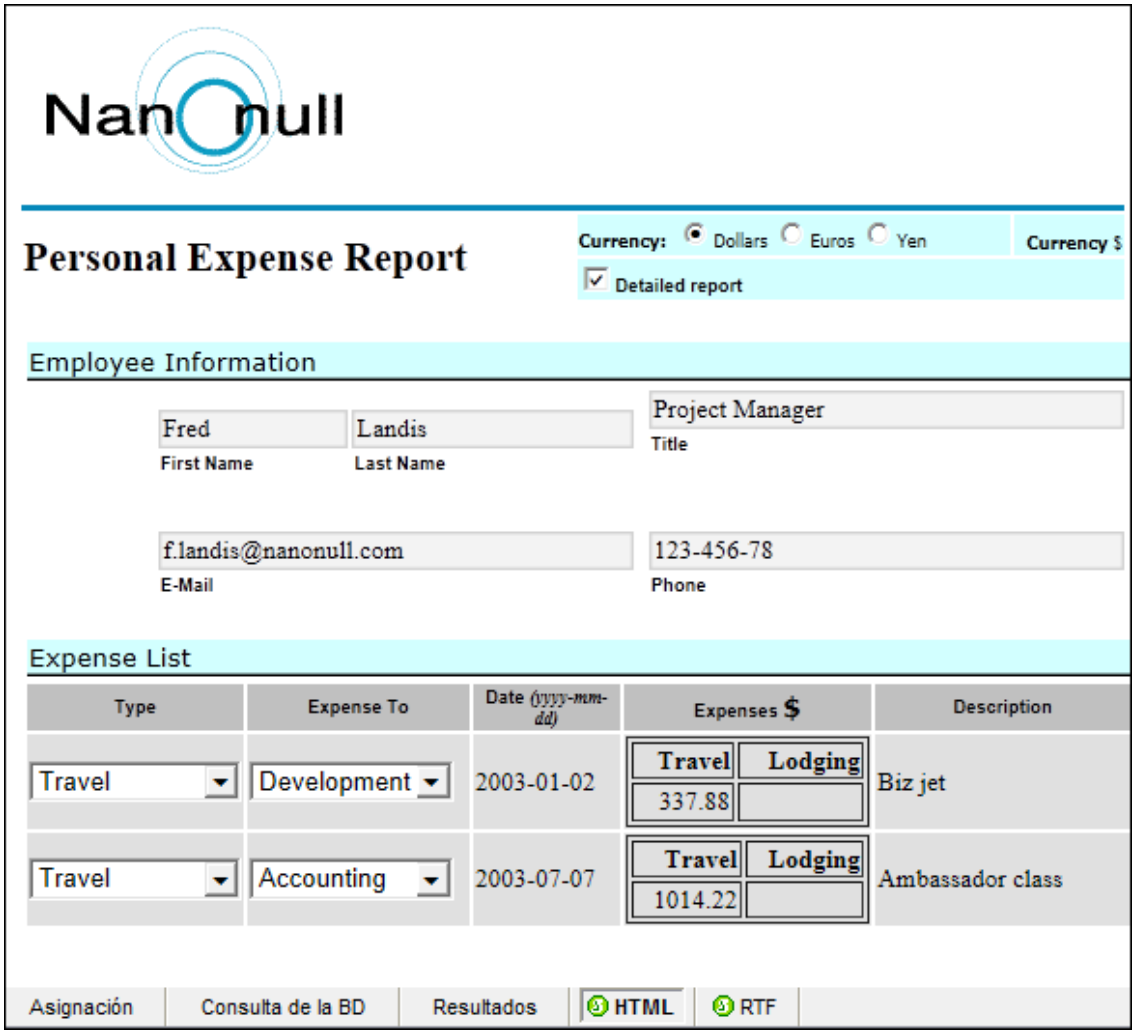


2. Haga clic en el botón **Examinar** para seleccionar el archivo SPS de StyleVision y después haga clic en **Aceptar**.



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.

 © 2014 Altova GmbH



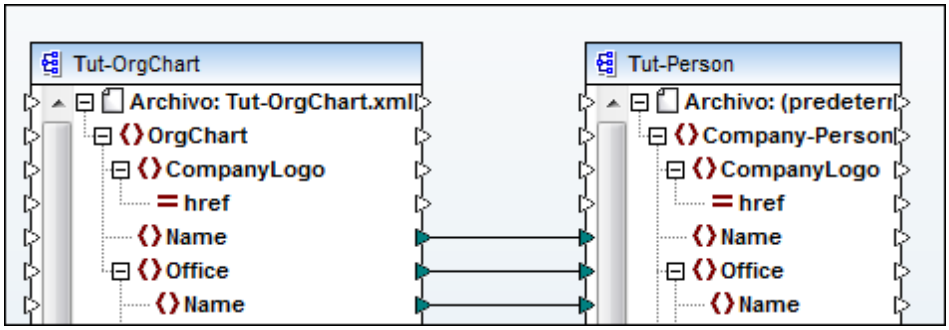
Nota: Si en el cuadro de diálogo "Configuración del componente" hace clic en el botón **Crear...**, MapForce solicita un nombre para el archivo SPS y abre StyleVision, donde puede crear el archivo SPS. Si el componente ya tiene asignado un archivo SPS, el cuadro de diálogo "Configuración del componente"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

Altova MapForce 2015

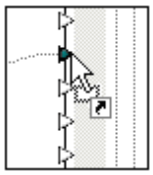
Asignaciones entre componentes

8 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 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.



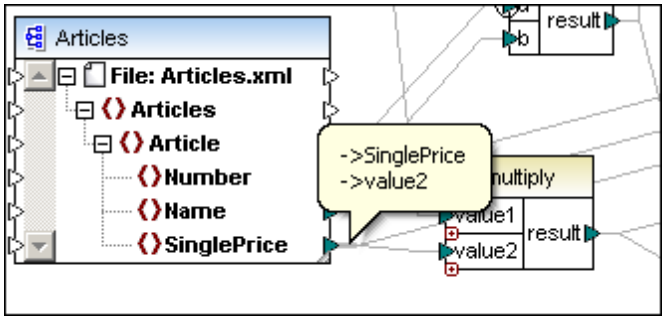
Para seleccionar el icono del elemento/atributo también puede hacer clic en su **nombre**. Igualmente, si suelta el conector en el nombre del elemento/atributo de destino, se selecciona automáticamente su icono.

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

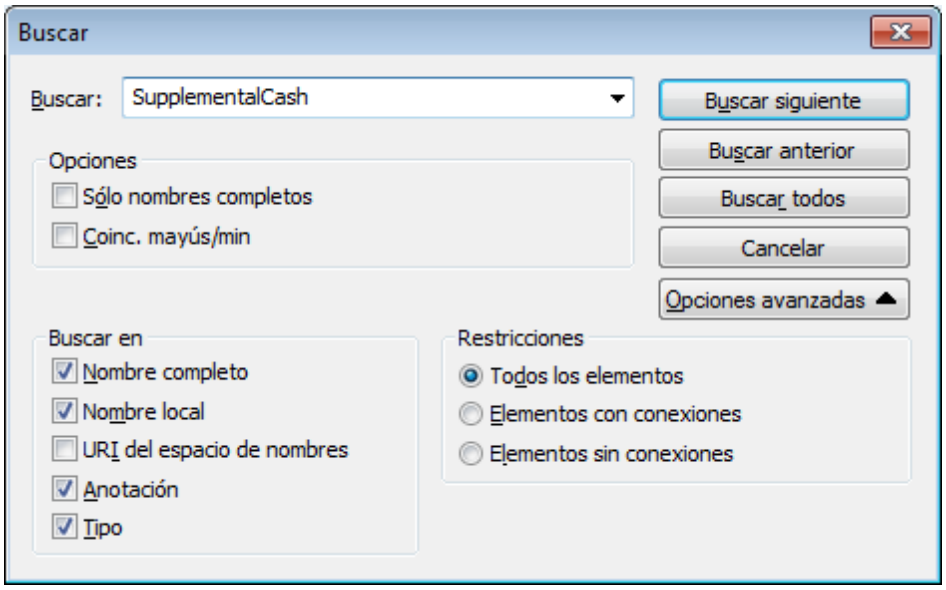


Para **mover** un conector de un elemento a otro elemento, haga clic el icono de salida del conector y arrástrelo hasta su nueva posición.

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:

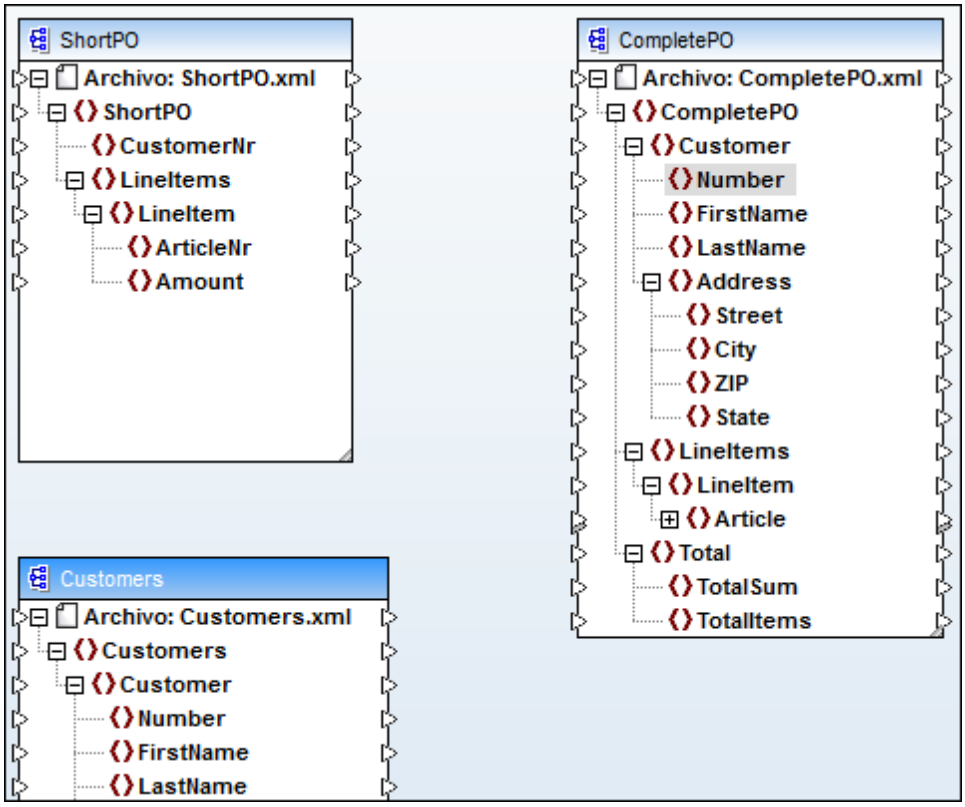
- Los elementos de **esquema** (atributos/elementos) tienen un icono de entrada y otro de salida.
-
- Los componentes de esquema y de otro tipo presentes dentro de funciones definidas por el usuario solamente tienen iconos de salida.
- Los elementos duplicados solamente tienen iconos de entrada. Así 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:

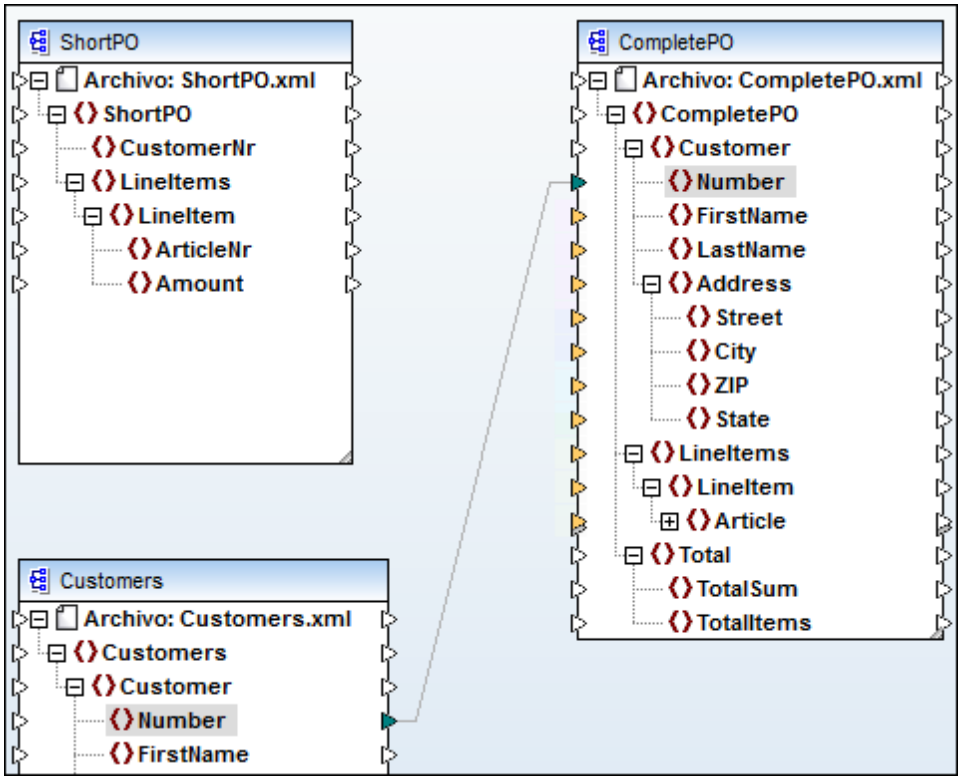
- Los elementos cuyo parámetro `minOccurs` es igual o mayor que `1` (en los componentes EDI y XML).
- Los campos que no admiten valores `null` (en los componentes de BD).
- Todos los nodos de las llamadas y respuestas WSDL.
- Los nodos XBRL definidos como obligatorios.
- Algunos parámetros concretos 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 `...\MapForce2015\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

8.1 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:

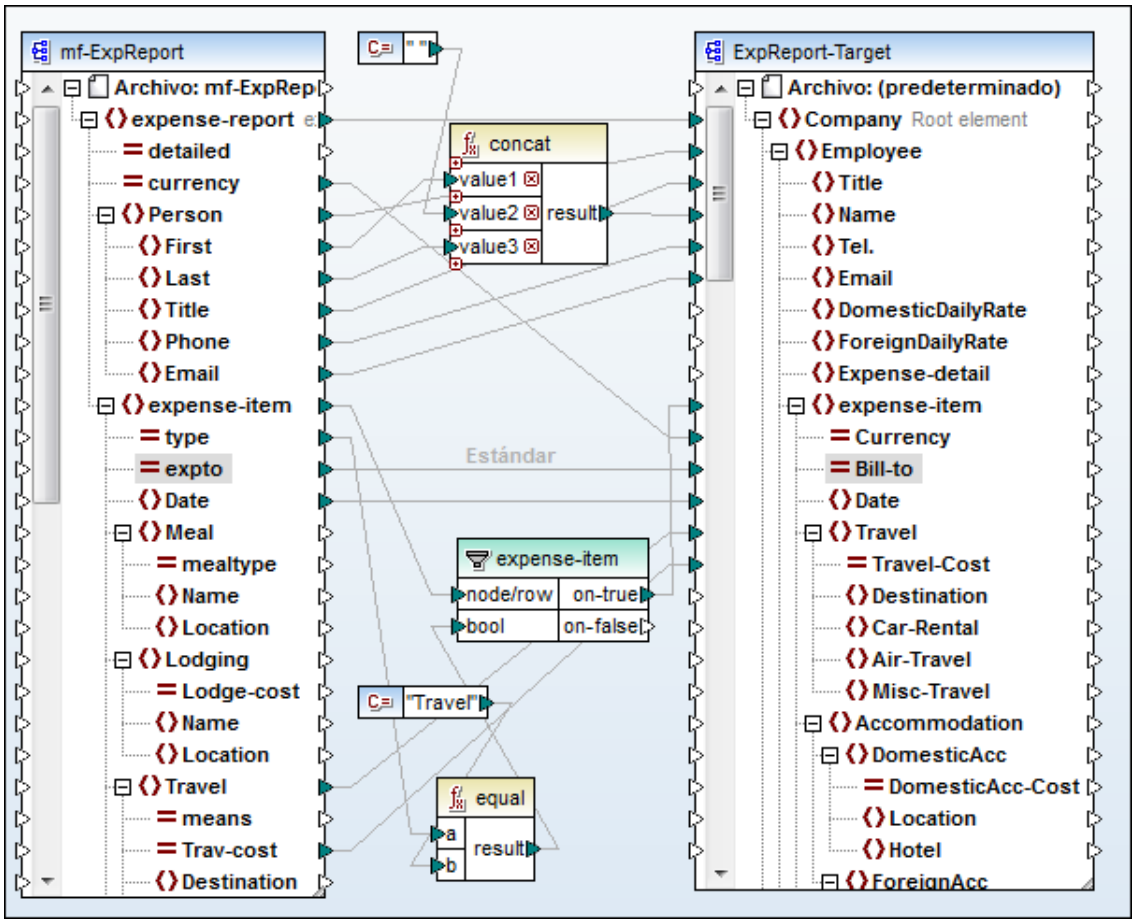
- ver resaltados en negro los conectores que están conectados **directamente** con el componente seleccionado o
- ver resaltados en negro los conectores relacionados con el componente seleccionado, desde el componente de origen hasta los de destino.

8.1.1 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.

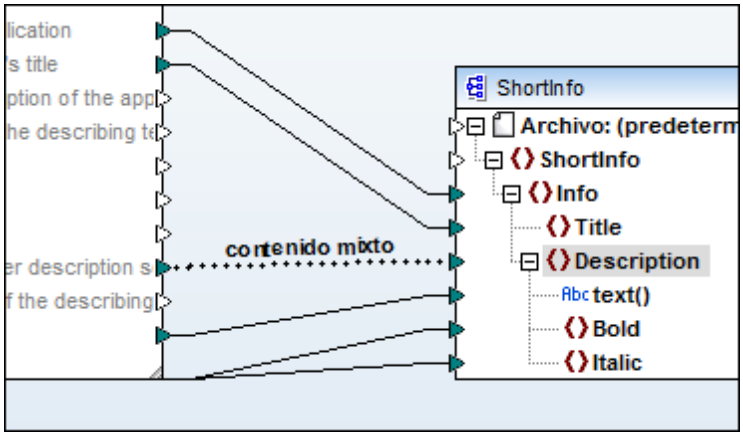


 © 2014 Altova GmbH

8.1.2 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`)
- como componentes de destino:
  - elementos **complexType** de XML Schema (incluidos elementos de contenido mixto)

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>
```

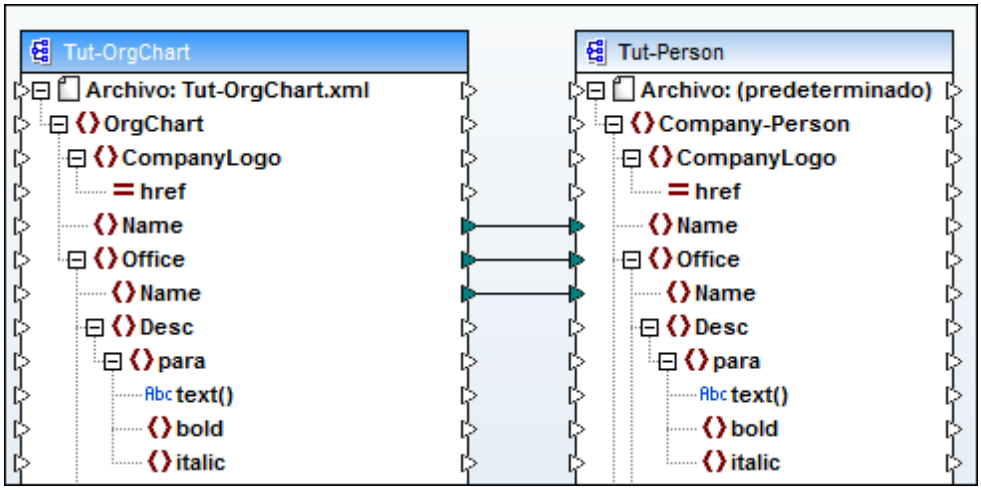
Observe el orden en el que aparecen los nodos **`text`** y los nodos **`bold`**/**`italic`** de `Nanonull, Inc.` en el archivo XML:

```
<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

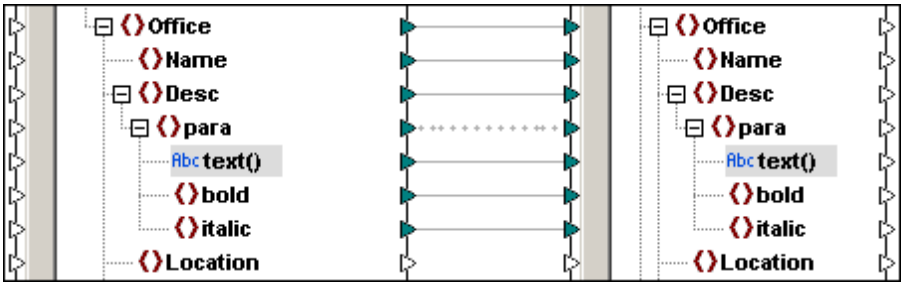
El resultado de esta asignación inicial aparece a continuación. Observe que el resultado incluye `Organization Chart` y el nombre de cada una de las oficinas.

 © 2014 Altova GmbH

```
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
```

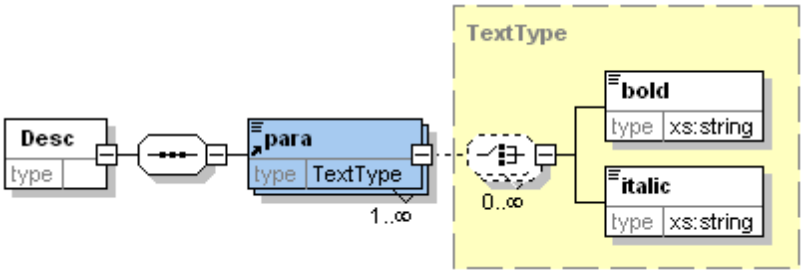
Asignación del elemento `para`

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.



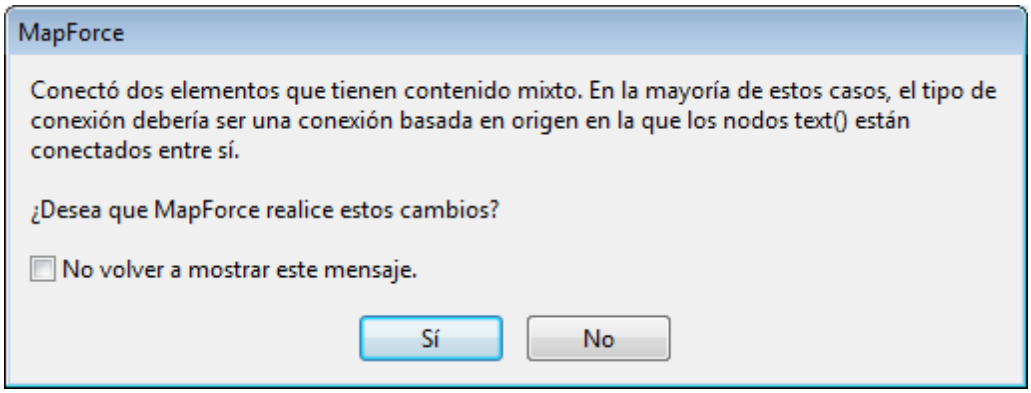
Fijémonos en las siguientes propiedades del elemento **`para`**:

- **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.

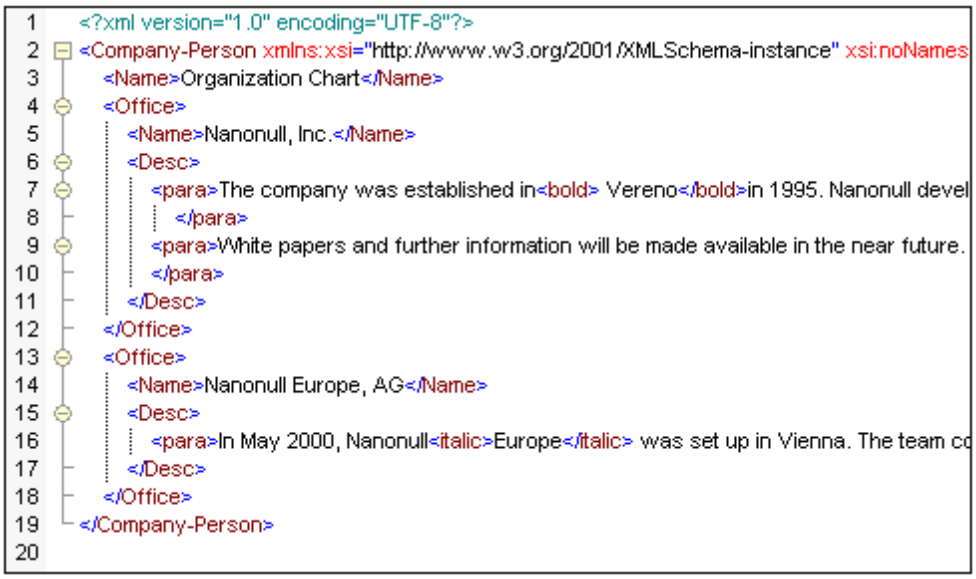


3. Haga clic en **Sí** para crear una conexión de contenido mixto.

   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.

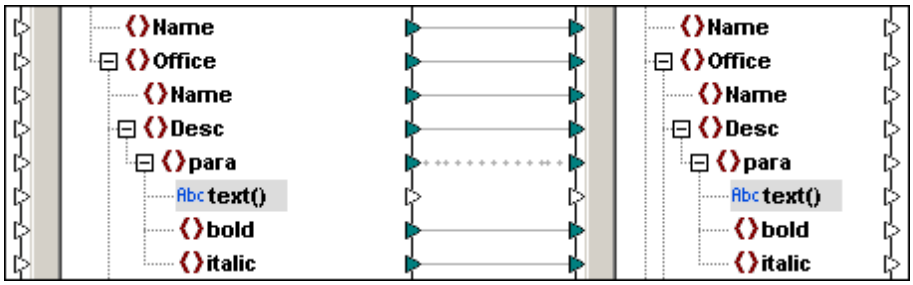
```
3    <Name>Organization Chart</Name>
4    <Office>
5      <Name>Nanonull, Inc.</Name>
6      <Desc>
7        <para>The company was established in<bold> Vereno</bold>in 1995. Nanonull develops
     nanoelectronic technologies for<italic>multi-core processors.</italic>February 1999 saw the
     unveiling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

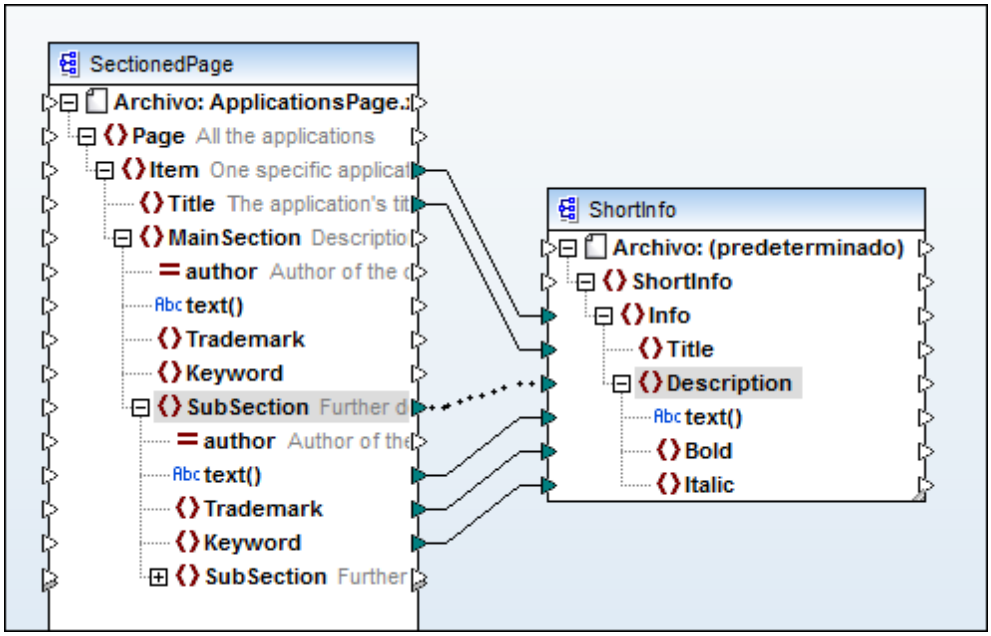
Aquí puede ver un pequeño fragmento del archivo XML de origen utilizado en este ejemplo.

```
<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.

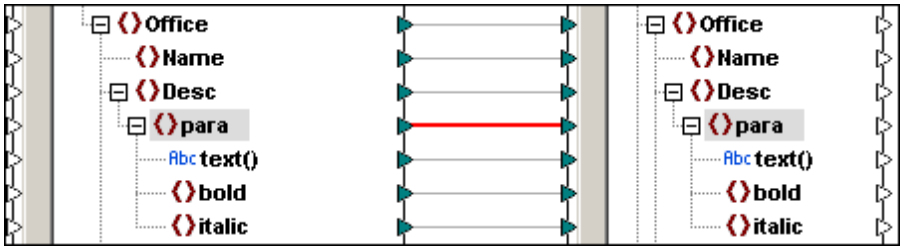
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <ShortInfo xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="
   C:/PROGRA~1/Altova/MapForce2005/MapForceExamples/ShortInfo.xsd">
3    <Info>
4      <Title>XMLSpy</Title>
5      <Description>Altova <Bold>XMLSpy</Bold>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

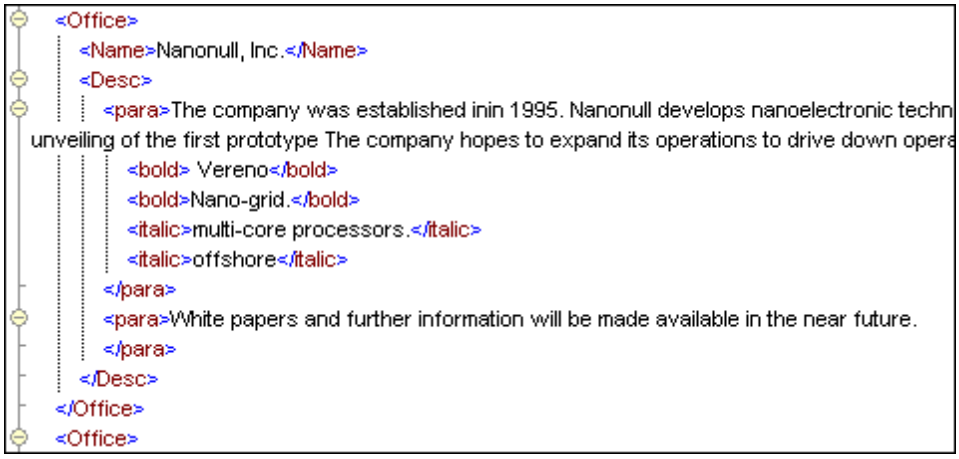
Esta sección describe los resultados que se obtienen al definir asignaciones **estándar** (usando conectores estándar) para nodos de **contenido mixto**. Para este ejemplo utilizamos el archivo `Tut-OrgChart.mfd`, que está en la carpeta `...\MapForceExamples\Tutorial`.

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**.

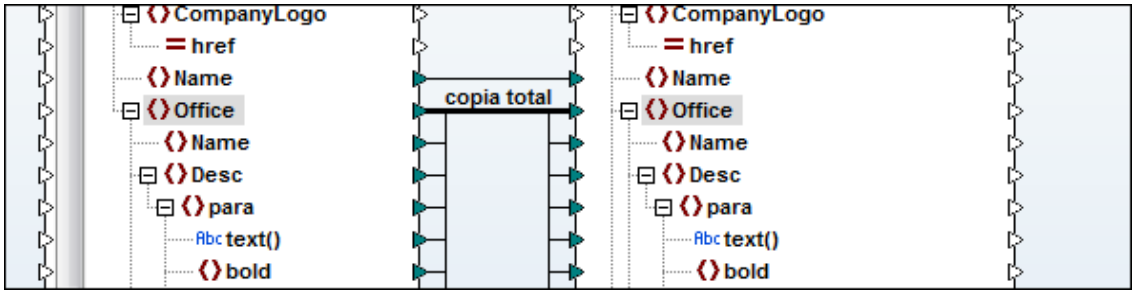
Es decir, por cada elemento **`para`**, se presenta primero el nodo **`text()`**, después todos los elementos **`bold`** y después todos los elementos **`italic`**.

 © 2014 Altova GmbH

8.1.3 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.

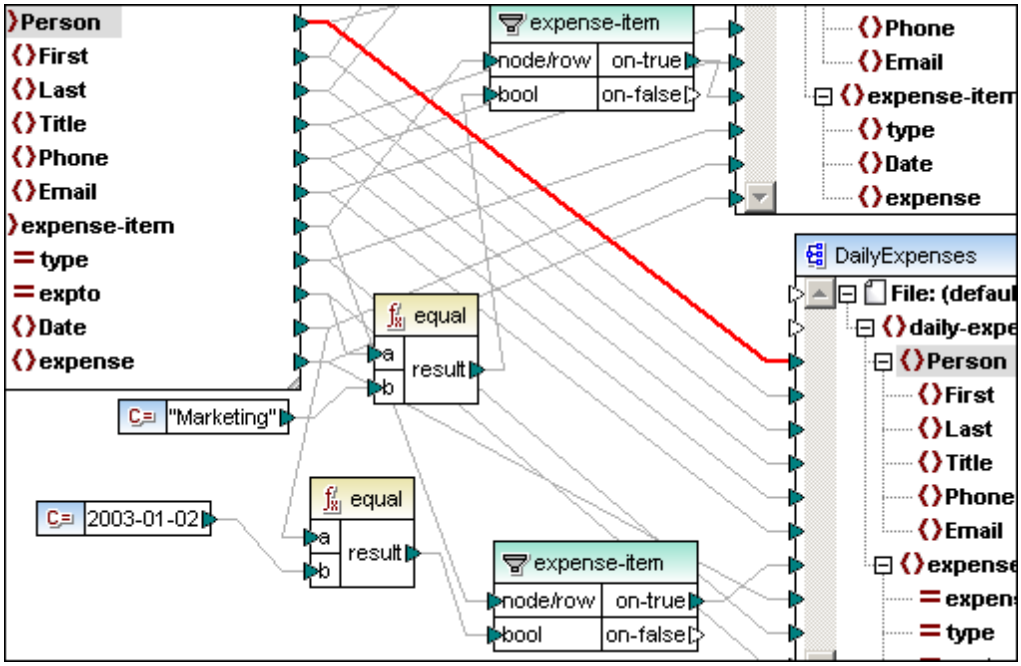
Tenga en cuenta que solamente se compara el nombre de los elementos secundarios, no su tipo.

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

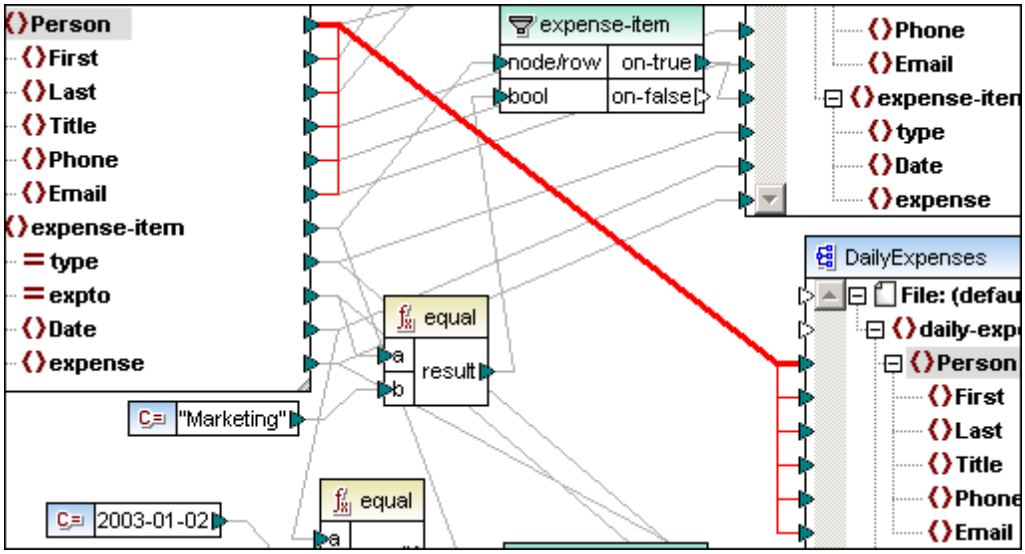
Para el ejemplo que aparece a continuación usamos el archivo `MarketingAndDailyExpenses.mfd`, guardado en la carpeta `...\MapForceExamples`.

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**.

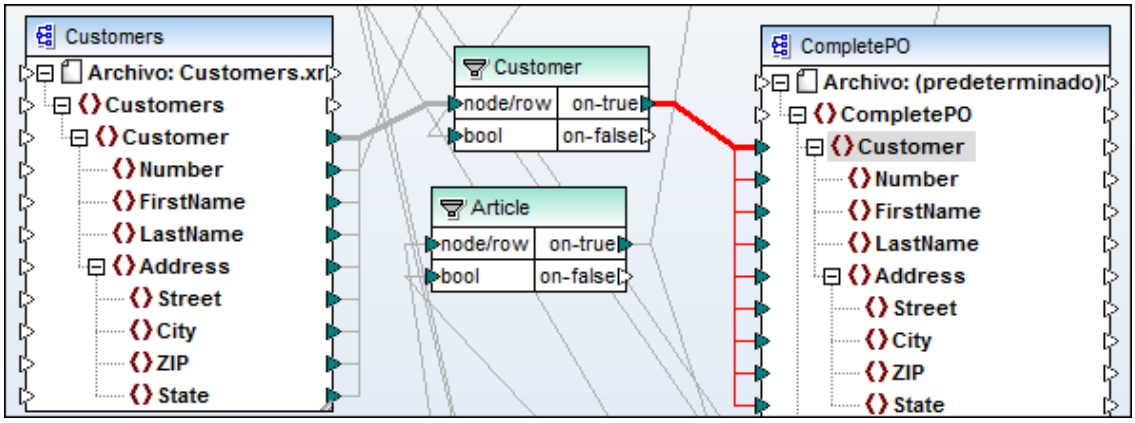
2. Aparece otro cuadro de diálogo. Haga clic en **Resolver la conexión de copia total**.
La conexión de copia total se sustituye por varios conectores.

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,
- 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.



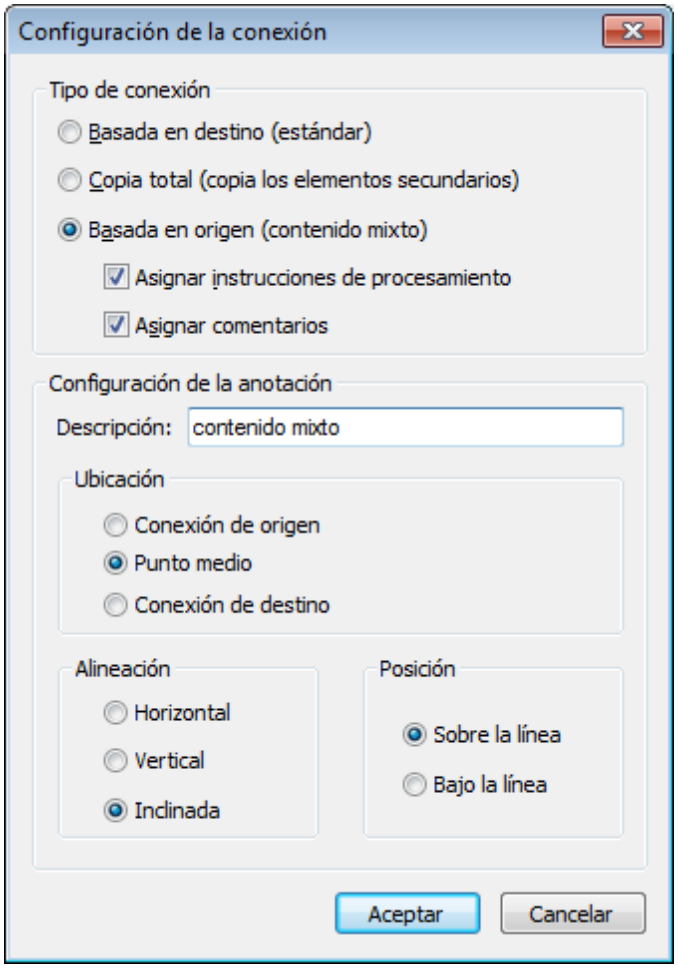
Para definir una conexión de copia total a través de un componente de filtrado:

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.

8.2 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.

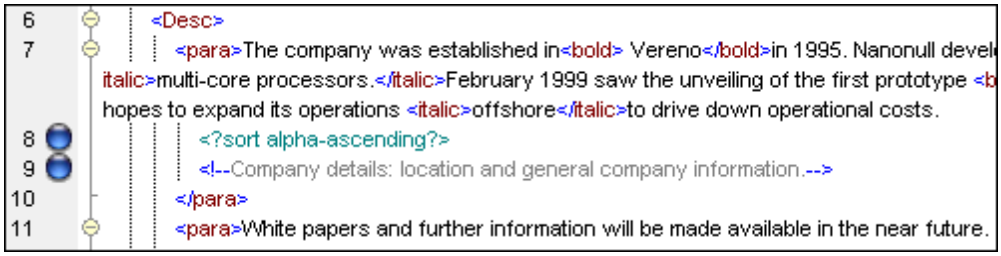


Tipo de conexión
Para los elementos complexType (incluso para los que no contienen nodos de texto), puede elegir uno de estos tipos de conexión:

- **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.

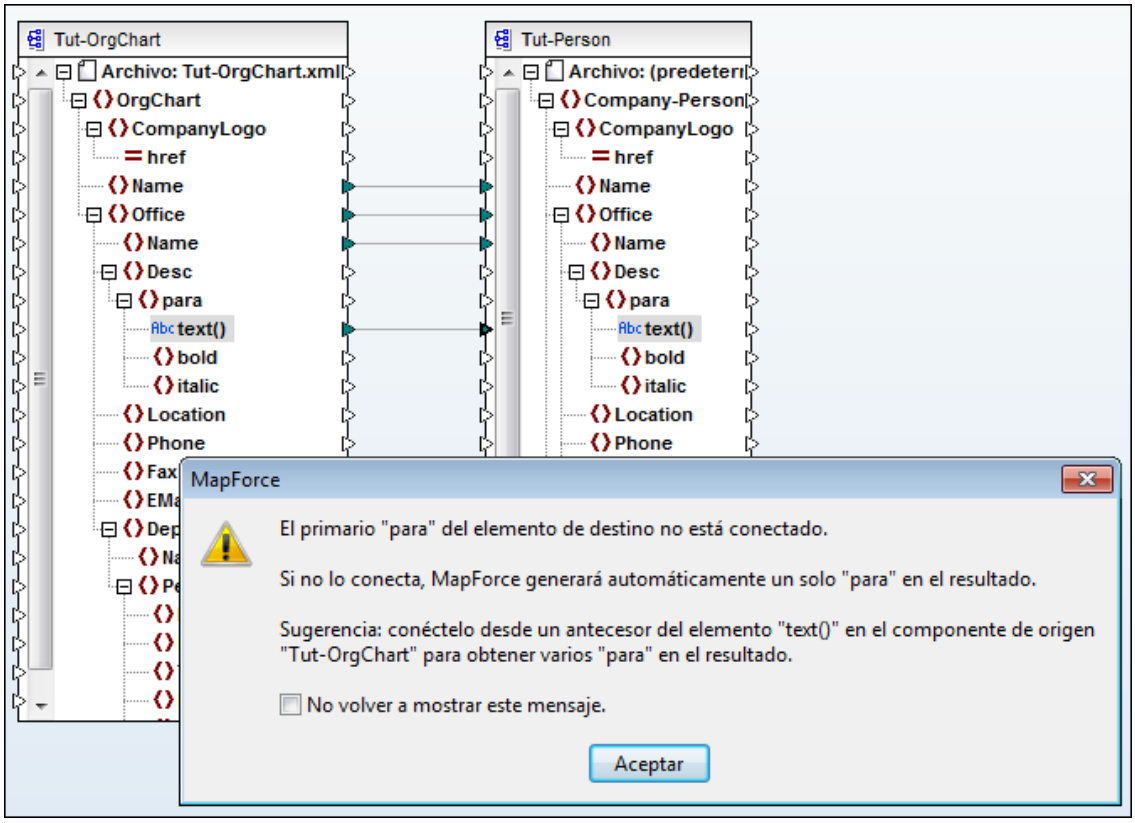
8.3 Conexiones y resultados de la asignación

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

8.4 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.

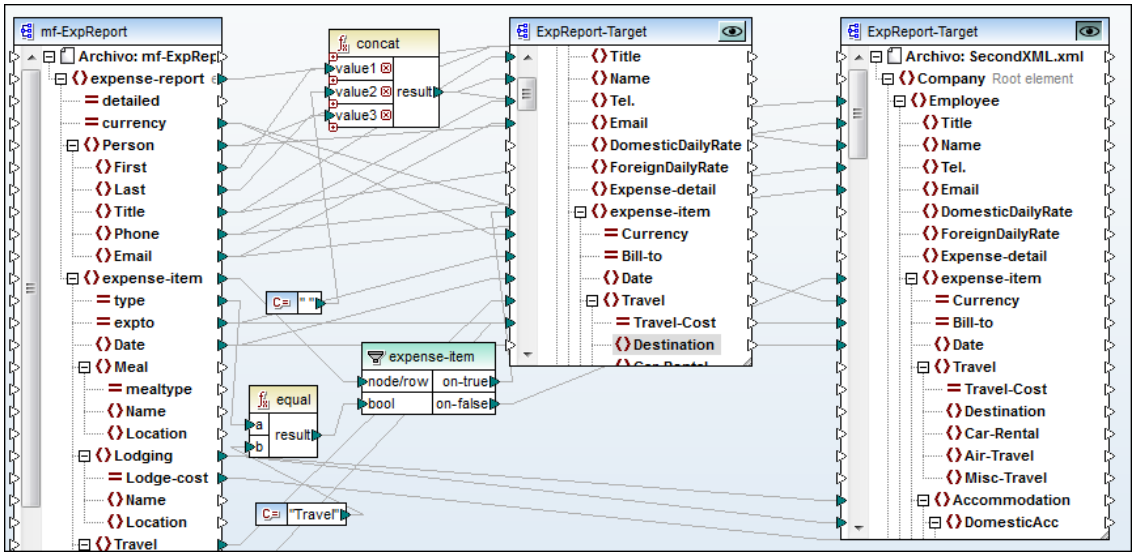
El orden en el que se procesan los componentes de destino se puede alterar cambiando la posición de los componentes de destino en el panel **Asignación**. La **posición** de un componente viene definida por la posición de su ángulo superior izquierdo.

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`.

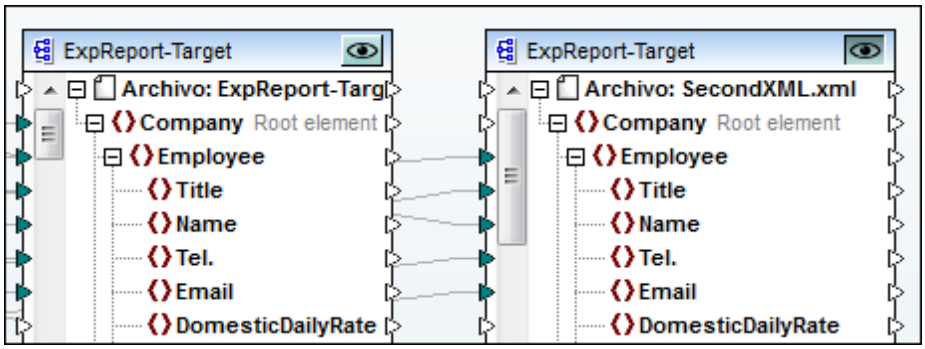
Para comprobarlo, abra el archivo `DoTransform.bat` (situado en la carpeta de salida indicada), que muestra el orden en el que se generan los archivos de salida. `ExpReport-Target.xml` es el primer archivo de salida generado por el archivo por lotes y `SecondXML.xml` es el segundo.

```
@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.

8.5 Asignaciones en cadena / componentes de paso a través

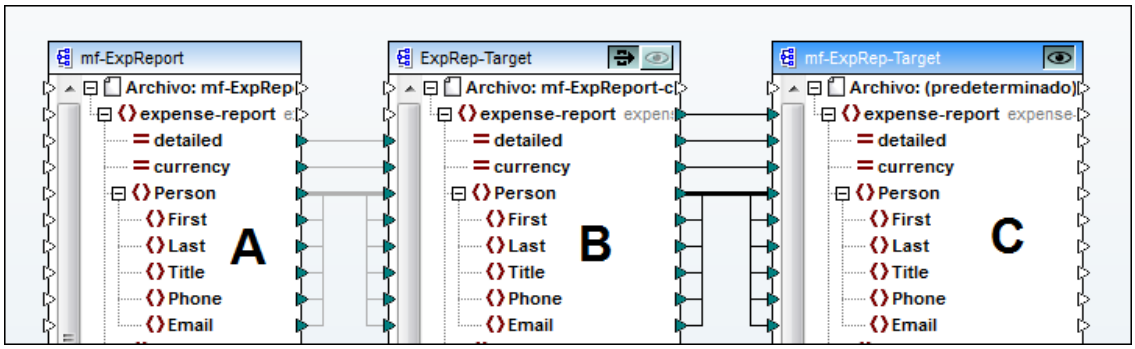
En MapForce puede trabajar con asignaciones compuestas por varios componentes encadenados. Las asignaciones en cadena son asignaciones en las que al menos uno de los componentes funciona tanto como componente de origen como de destino. Dichos componentes crean resultados que se utilizan como entrada 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.

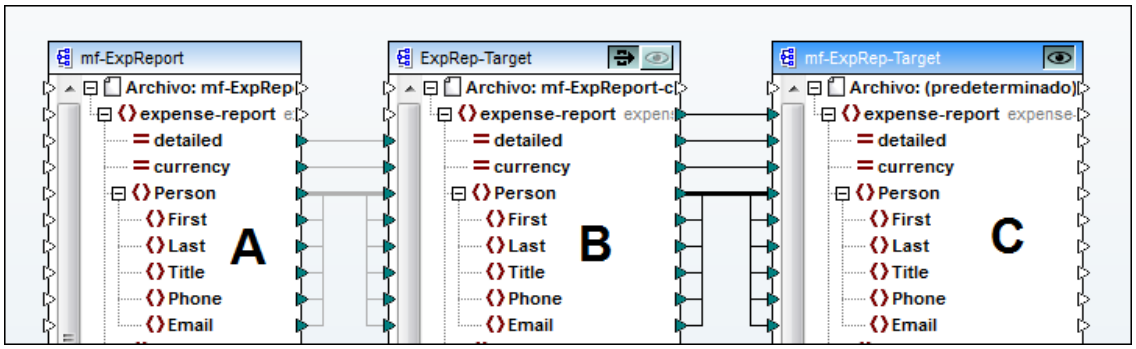
En nuestro ejemplo se generarán dos archivos de resultados. Esto se debe a que MapForce analiza automáticamente las dependencias entre todos los componentes y genera todos los resultados de los componentes intermedios y de destino en el orden correcto.

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.

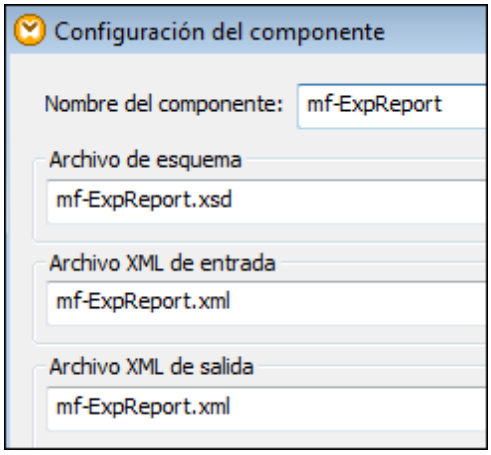
8.5.1 Asignaciones en cadena: paso a través activo

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).

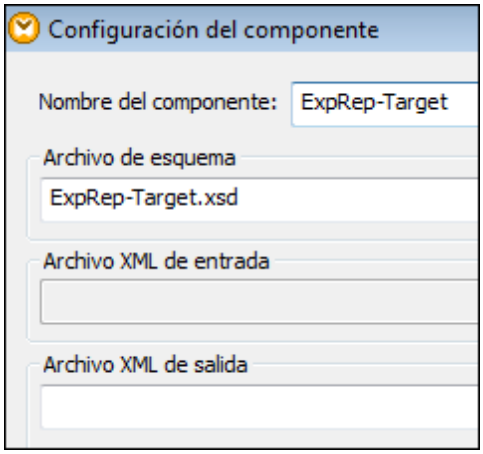


- El **componente intermedio B** con el paso a través activo:
  cuando el paso a través está activo, el campo *Archivo XML de entrada* del componente intermedio se desactiva automáticamente. No es necesario tener un nombre para el archivo porque los datos intermedios se almacenan en archivos temporales.

  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.

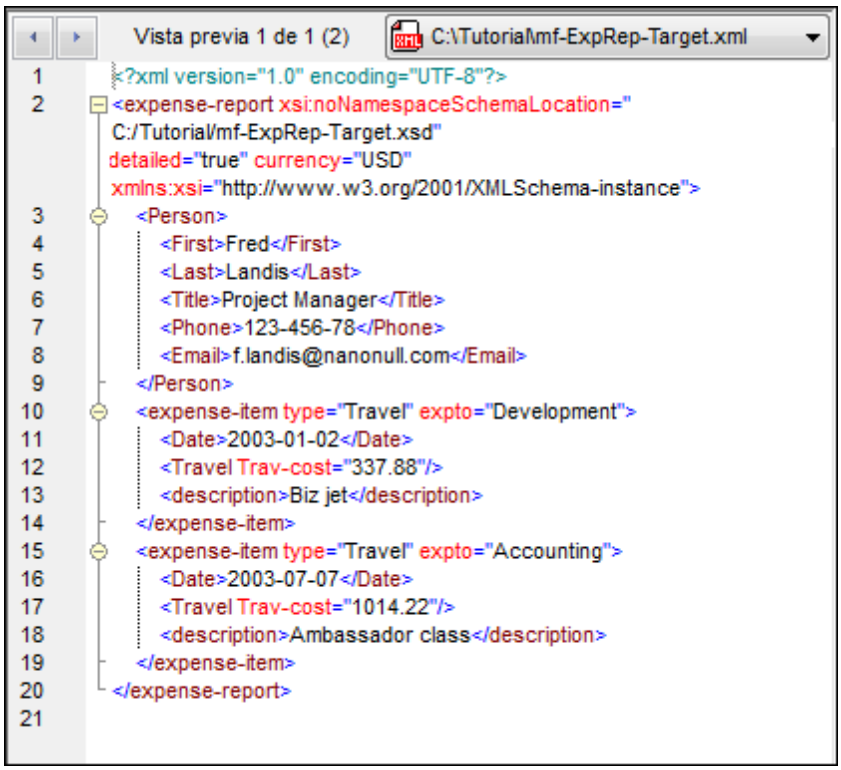
 © 2014 Altova GmbH



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

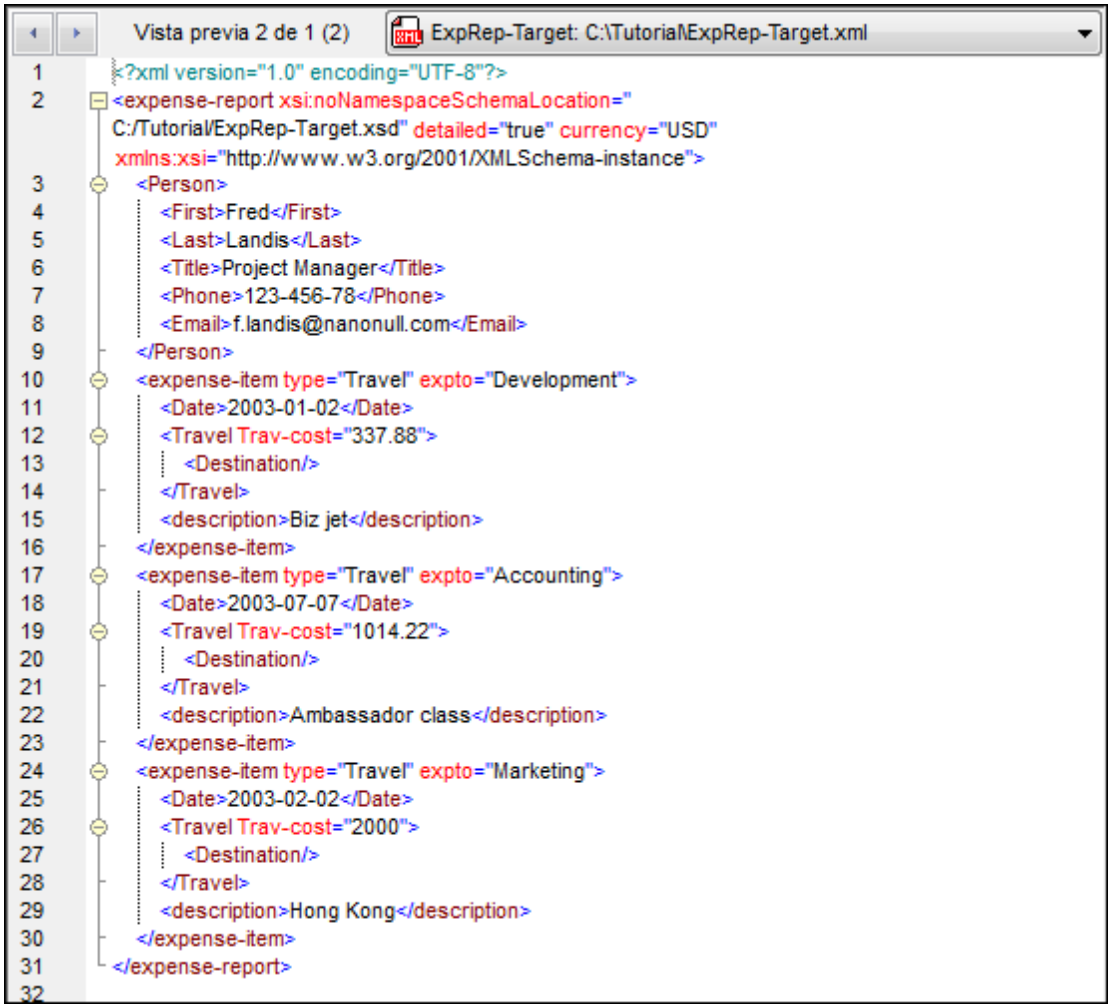
Pase al panel **Resultados** para ver una vista previa de los resultados en el motor de ejecución integrado.

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.

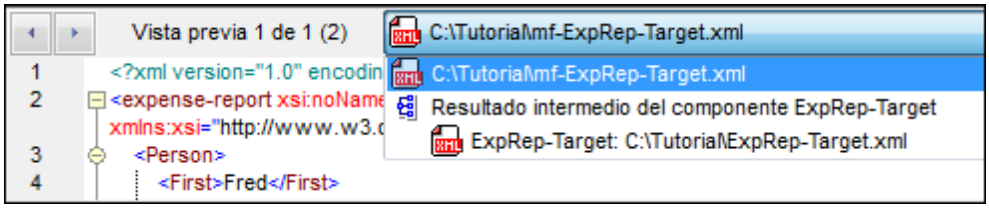


Vista previa 2:
Muestra el resultado de asignar los datos del componente A al componente intermedio B (es decir, todos los gastos de viaje). `ExpRep-Target.xml` es un nombre de archivo predeterminado

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

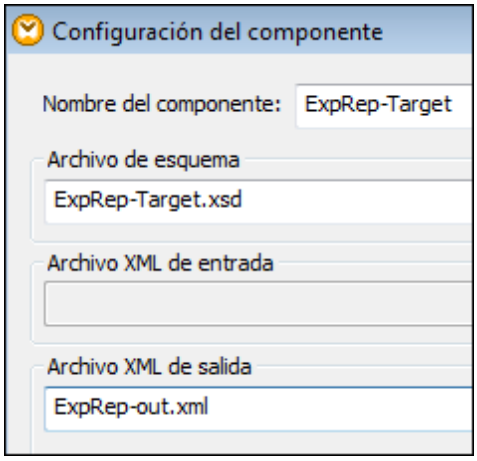


Nota importante:
Cada resultado tiene una ventana de vista previa propia. Haga clic en la flecha de la lista desplegable para ver los demás resultados.

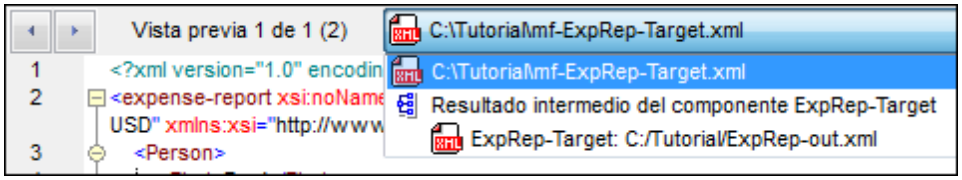


En la lista desplegable aparecen todos los archivos de resultados. El resultado final aparece al principio de la lista y el resultado intermedio al final. Haga clic en el resultado que desea ver o use las teclas de dirección del teclado para navegar por la lista y pulse **Entrar**.

 © 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`).



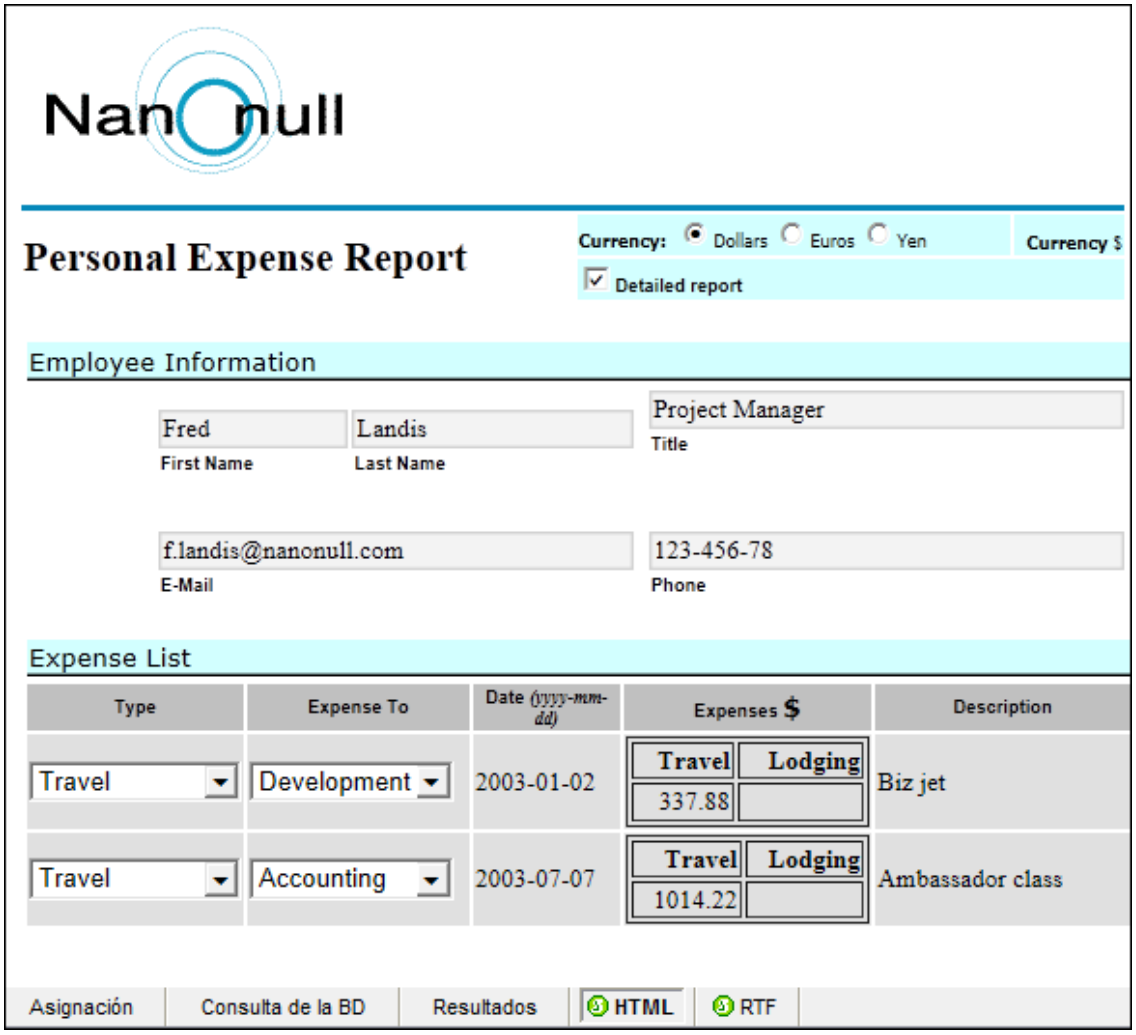
Cuando está activo el botón de paso a través, los archivos creados por el componente intermedio se guardan **automáticamente** como archivos temporales y se usan para seguir procesando los resultados de ese componente.

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** de StyleVision de MapForce.

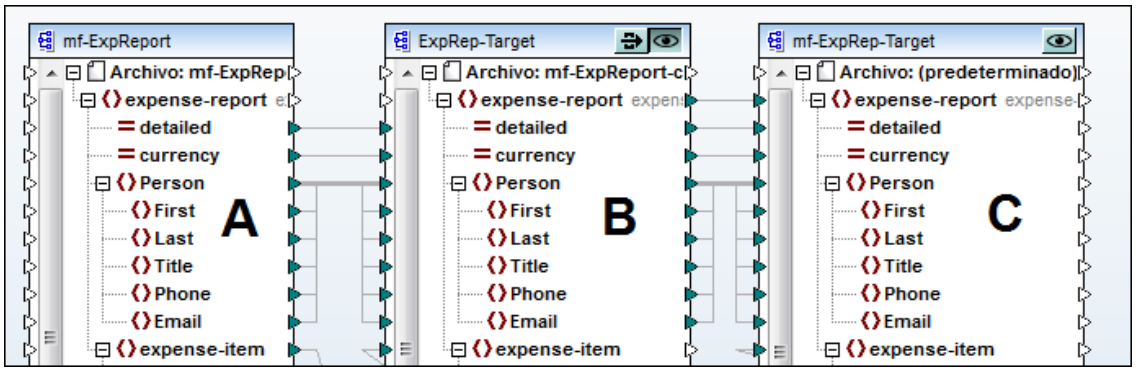


Recuerde que en los paneles de StyleVision de MapForce solamente se representan los resultados de los componentes de destino finales de la asignación en cadena. En estos paneles no se puede ver el resultado de componentes intermedios.

 © 2014 Altova GmbH

8.5.2 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.

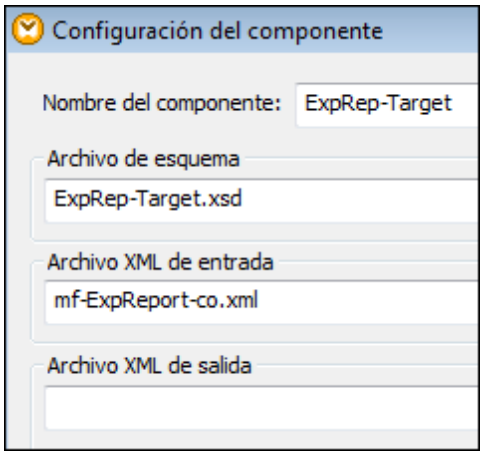
  Al cambiar al panel **Resultados** se pueden ver los resultados en el motor de ejecución integrado.

  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.



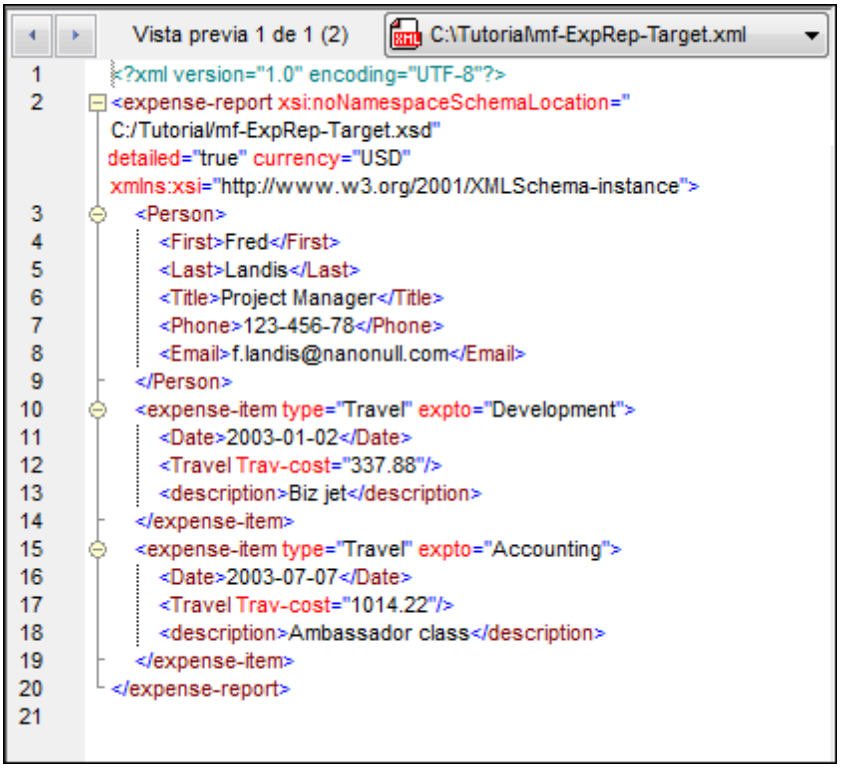
MapForce abre el archivo intermedio y asigna sus datos al componente C. Si existe el

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.

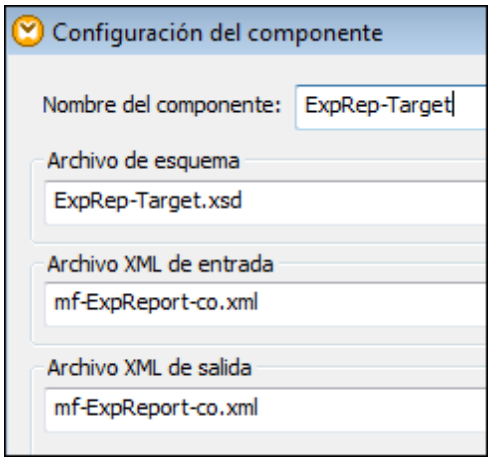
Guardar el resultado intermedio de una asignación

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**.

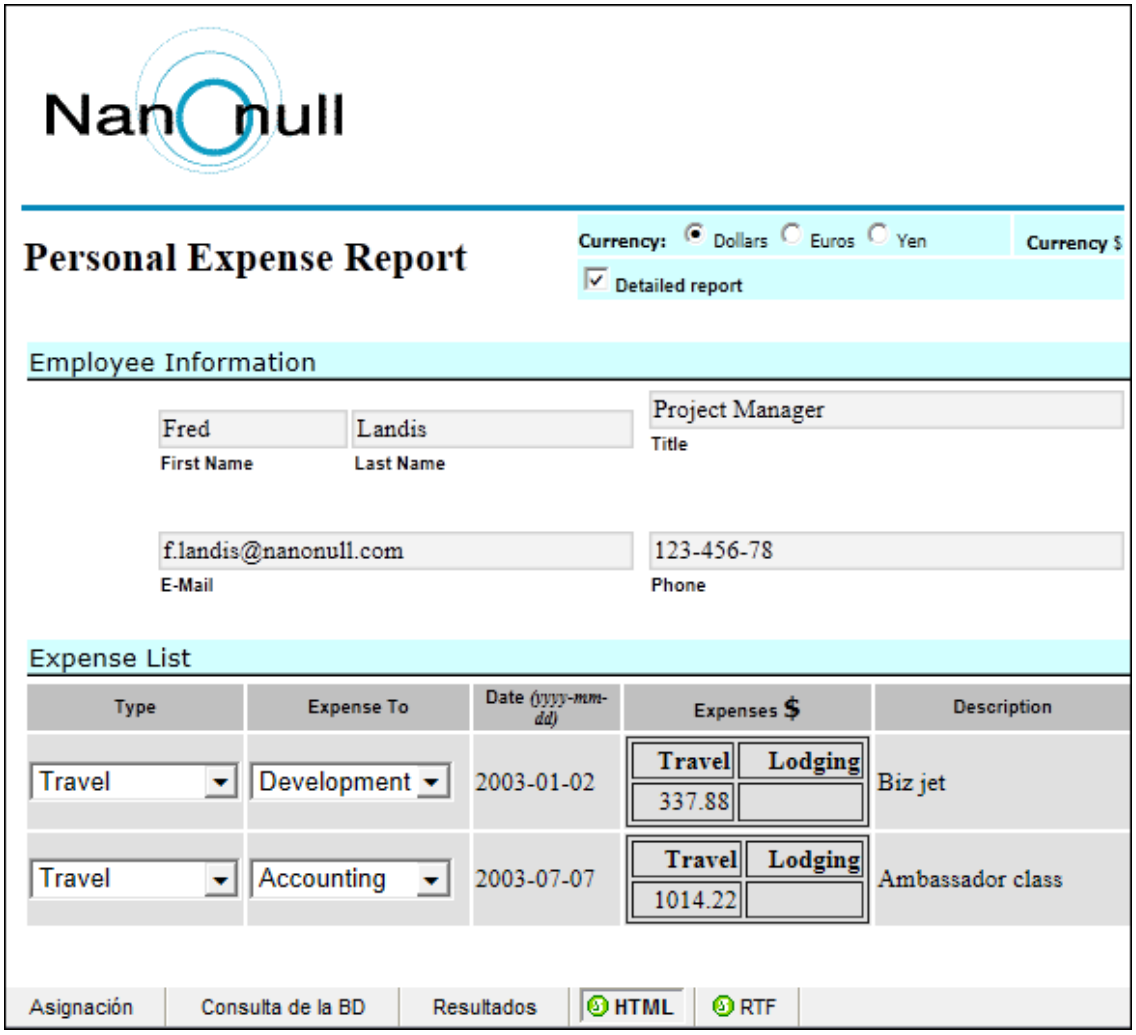


Nota: los archivos XML de entrada y salida **deben** tener el mismo nombre y ambos **deben estar presentes** si desea generar código y ejecutar la asignación desde la línea de comandos.

Visualizar los resultados con StyleVision

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

 © 2014 Altova GmbH

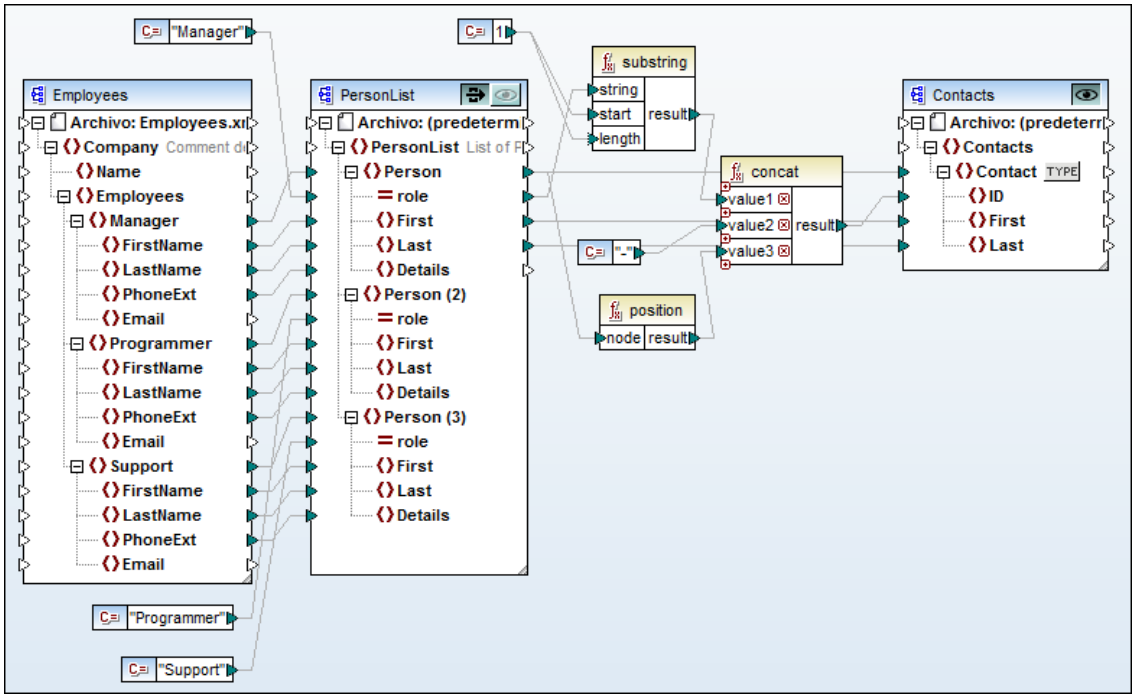

NanoNull
Personal Expense Report
Currency: Dollars Euros Yen
Currency $
Detailed report
Employee Information
Fred
First Name
Landis
Last Name
Project Manager
Title
f.landis@nanonull.com
E-Mail
123-456-78
Phone
Expense List
Type
Expense To
Date (yyyy-mm-dd)
Expenses $
Description
Travel
Development
2003-01-02
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

8.5.3 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)

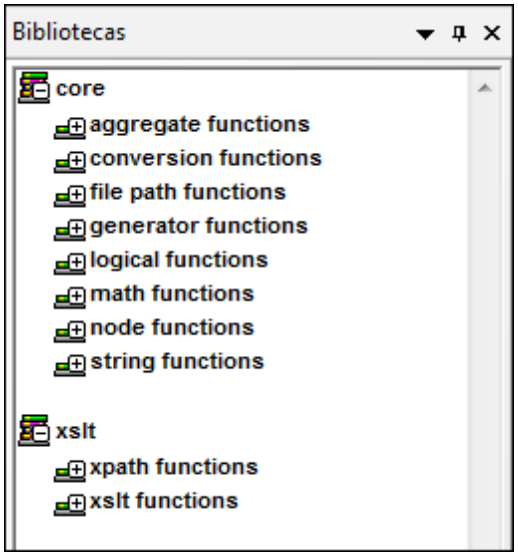
Vista previa 1 de 1 (2) C:\MapForceExamples\Contacts.xml

```
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

8.6 Uso de las funciones

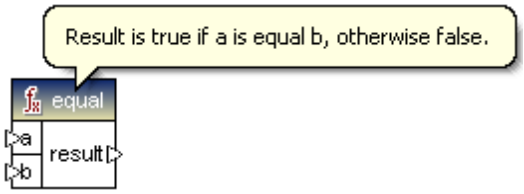
En la ventana Bibliotecas aparecen todas las funciones disponibles para el lenguaje seleccionado. Para ver/ocultar las funciones use los iconos + y - de cada biblioteca.



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.



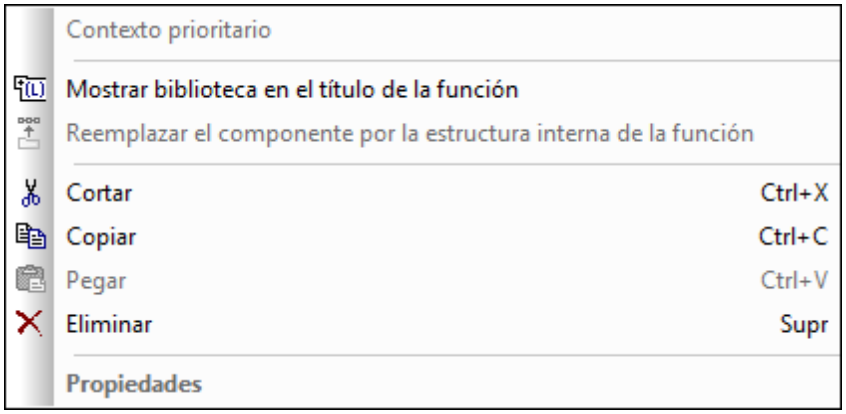
Para usar una función en la ventana de la asignación:

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.

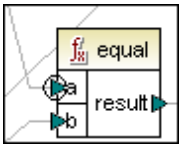
Recuerde que si pasa el puntero del mouse sobre la expresión "`result=....`" de la ventana Bibliotecas, aparece un mensaje emergente que describe la función con más detalle.

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, 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.

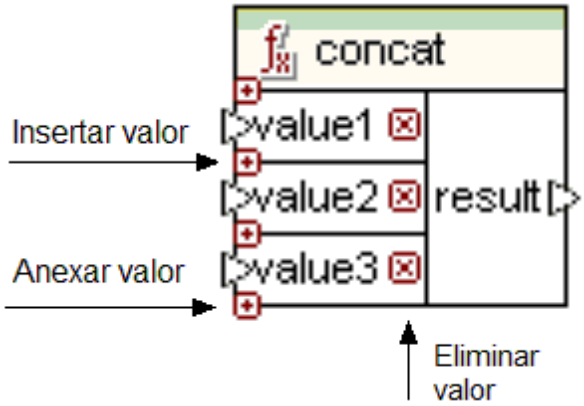
Funciones ampliables

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.

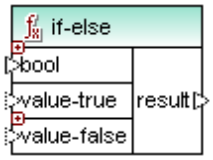
Al hacer clic en el icono **+** se inserta/anexa el mismo tipo de parámetro. Asimismo, al hacer clic en la marca **x** se elimina el parámetro.

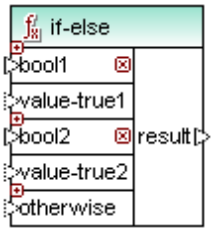
Nota: al soltar un conector en el icono **+** de la función, se inserta/anexa otro parámetro y la conexión se crea automáticamente.

 © 2014 Altova GmbH



Los parámetros de prueba IF de la función **IF-Else** también se pueden ampliar de esta forma.





8.7 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.

Asignación de valores (tabla de consulta)

Contexto prioritario

© 2014 Altova GmbH

8.8 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.

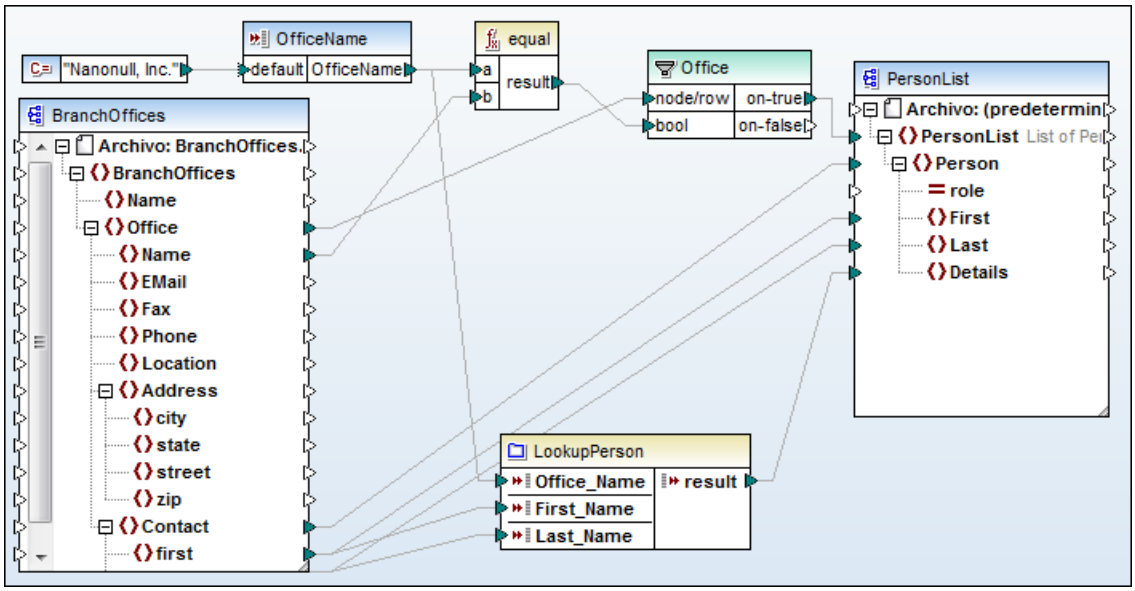
Si el elemento de origen tiene contenido simple (p. ej. una cadena de texto, un entero, etc.) y el de destino acepta contenido simple, entonces se copia el contenido y se convierte el tipo de datos, si es necesario.

Sin embargo, hay algunas excepciones.

Elementos *actuales* y elementos de *contexto*
MapForce muestra la estructura de un esquema 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:

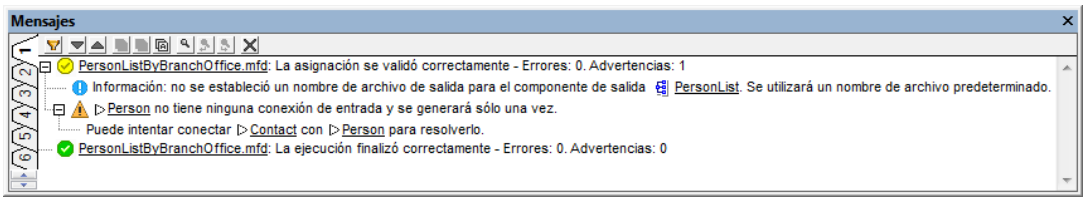
- En un principio, el contexto solamente contiene los componentes de origen, pero no contiene ningún nodo concreto. Al evaluar la asignación, MapForce procesa primero el nodo **raíz de destino** (**`PersonList`**) y después sigue con el resto de la jerarquía.
- El conector del nodo de **destino** llega 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 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 de entrada asociado o
- una **secuencia** de entre 0 y **varios nodos de instancia** del archivo 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)

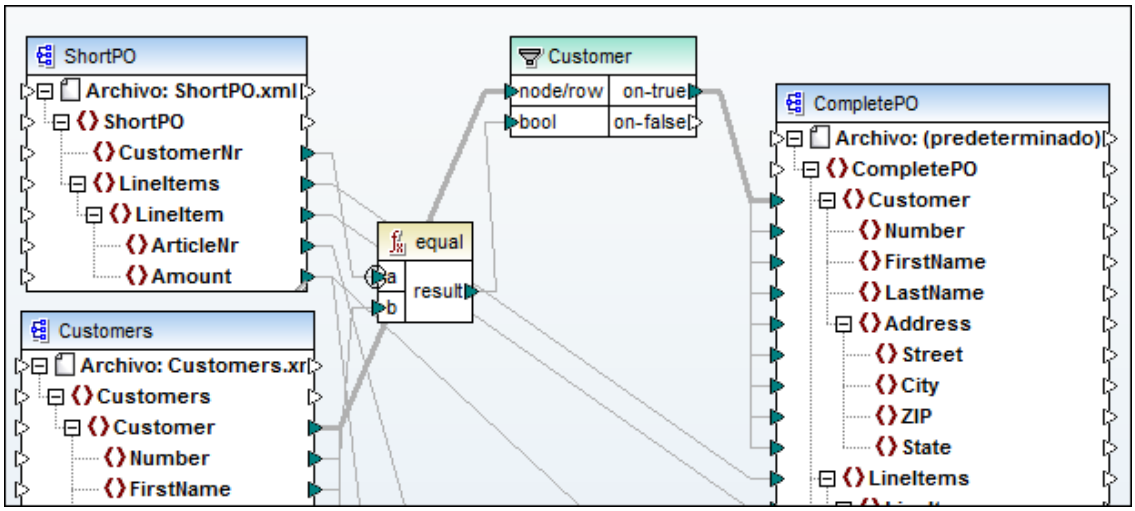
La secuencia de entrada de este tipo de funciones siempre se evalúa independientemente del nodo de destino actual del contexto de sus antecesores. Estos significa que si hay componentes

de filtrado 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`).



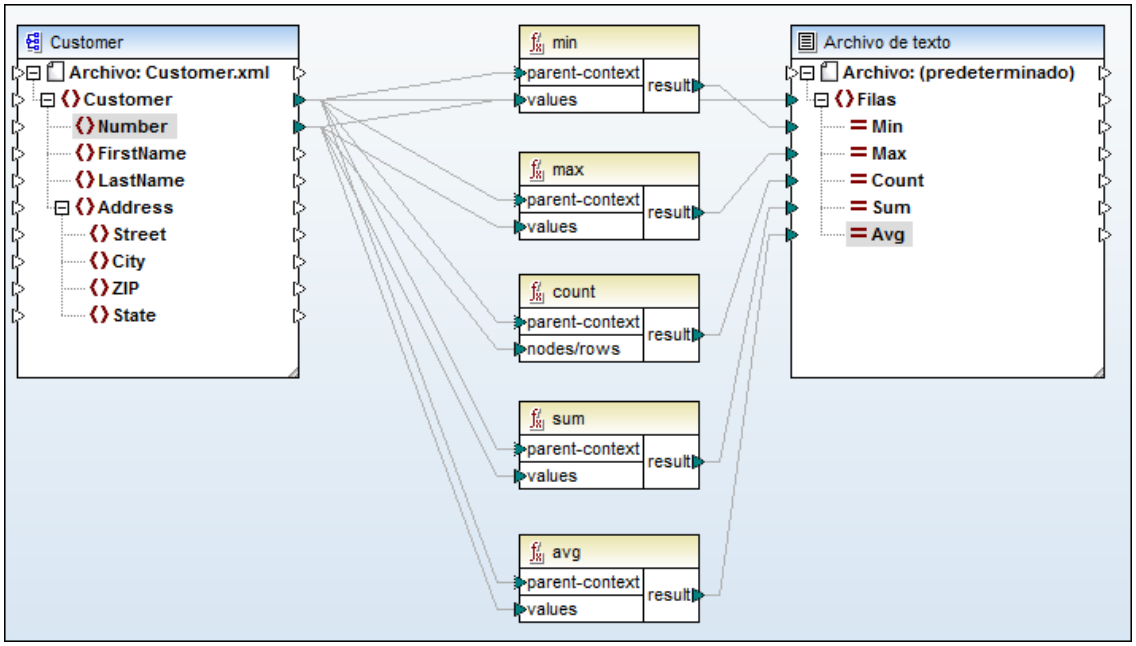
En este ejemplo el contexto prioritario obliga a evaluar **`ShortPO`**/**`CustomerNr`** antes de recorrer y filtrar los nodos de **`Customer`** del componente **Customers**. Para más información, consulte el ejemplo del apartado Nodo/Elemento con contexto prioritario.

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, 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 2015

Datos de origen y destino

9 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

© 2014 Altova GmbH

9.1 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

9.1.1 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:

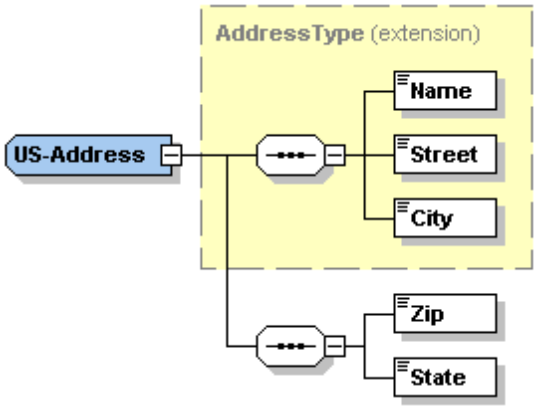
```
<!ATTLIST fo:root
        xmlns:fo CDATA #FIXED 'http://www.w3.org/1999/XSL/Format'
        ...
>
```

 © 2014 Altova GmbH

9.1.2 Asignar tipos derivados de XML Schema

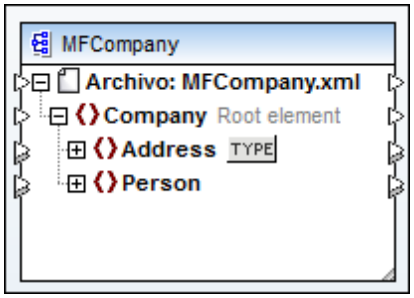
En MapForce puede realizar asignaciones entre tipos derivados de un tipo complejo. Los tipos derivados son tipos complejos de un esquema XML que usan el atributo `xsi:type` para identificar los tipos derivados.

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.

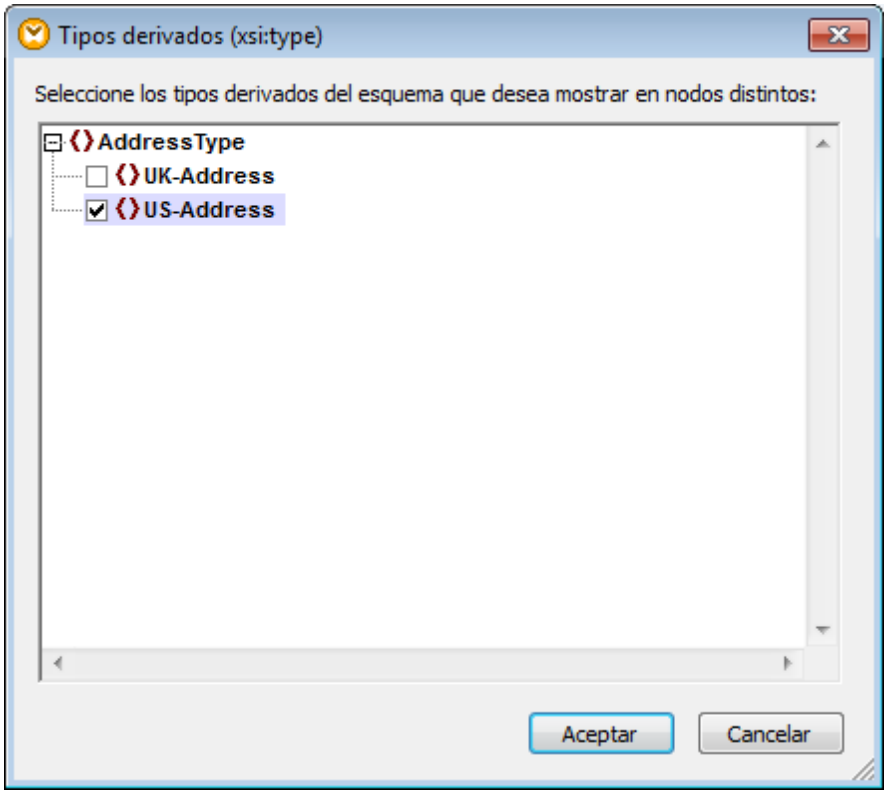


Asignaciones de datos a tipos derivados en MapForce:

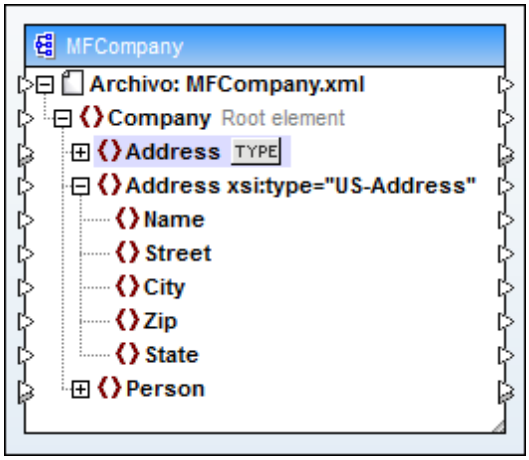
1. Inserte el esquema XML `MFCompany.xsd` (de la carpeta `...\Tutorial`) haga clic en **Omitir** y seleccione el elemento raíz `Company`.



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

9.1.3 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>
```

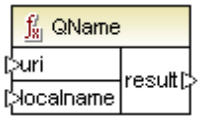
x es la referencia de espacio de nombres a **`"http://myCompany.com"`** y `<x:part>` es un QName válido porque:

x es el prefijo de espacio de nombres
part es la parteLocal, es decir, el nombre del elemento.

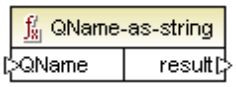
En las asignaciones entre componentes de origen y destino los prefijos de QName se resuelven.

MapForce ofrece varias funciones para trabajar con nombres QName (en la biblioteca de funciones **Lang**):

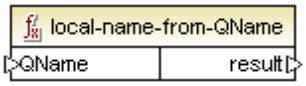
QName
Construye un QName a partir de un URI de espacio de nombres y una parte local. Utilice esta función para crear un QName en un componente de destino. Los parámetros **`uri`** y **`localname`** pueden venir dados por una constante.



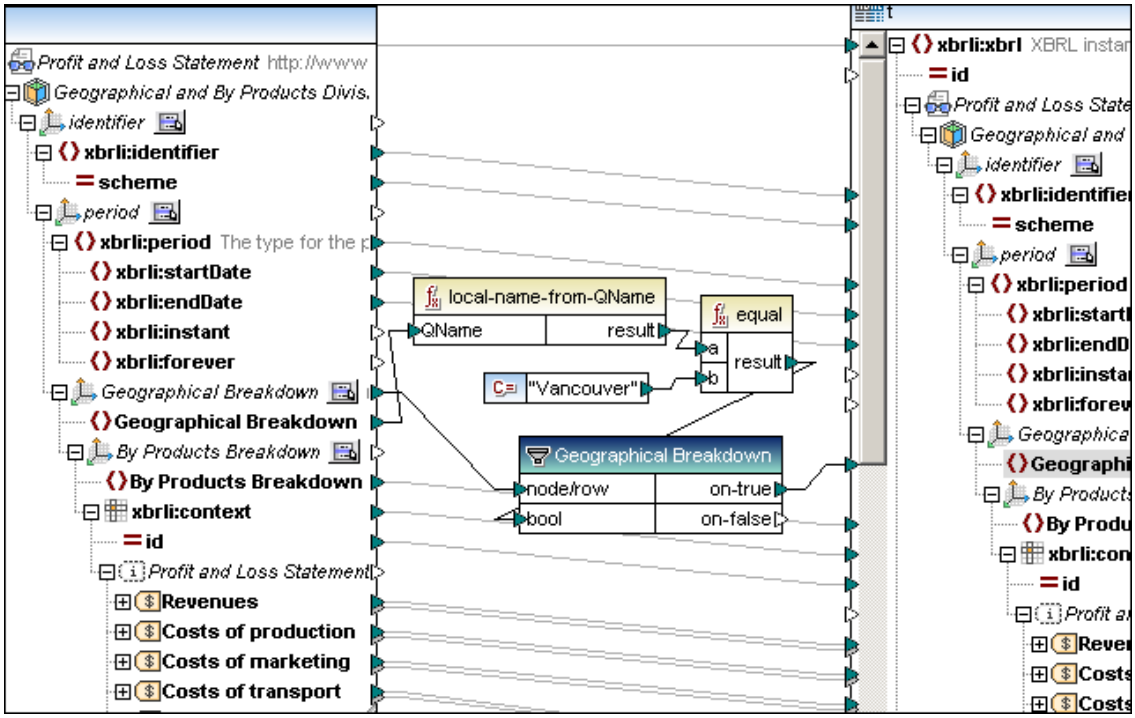
QName-as-string
Convierte un QName en una cadena de formato `{http://myCompany.com}local`.



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.

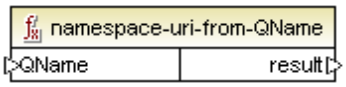
```
<xbrli:context id="Year2007_Vancouver_BeveragesTotal">
  <xbrli:entity>
    <xbrli:identifier scheme="http://www.stockexchange/ticker">ACME</xbrli:identifier>
    <xbrli:segment>
      <xbrldi:explicitMember dimension="d-g:GeographicalBreakdown">d-g:Vancouver</xbrl
      <xbrldi:explicitMember dimension="d-b:ByProductsBreakdown">d-b:BeveragesTotal</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.



9.1.4 Valores nulos / que pueden ser nulos

La especificación XML Schema permite que un elemento vacío (sin contenido) sea válido si en el esquema se definió el atributo `nillable="true"` para dicho elemento. En el documento XML de instancia, puede indicar que el valor de un elemento es nil añadiendo el atributo `xsi:nil="true"`. En este apartado describimos cómo se tratan estos elementos nulos en los componentes de origen y destino de MapForce.

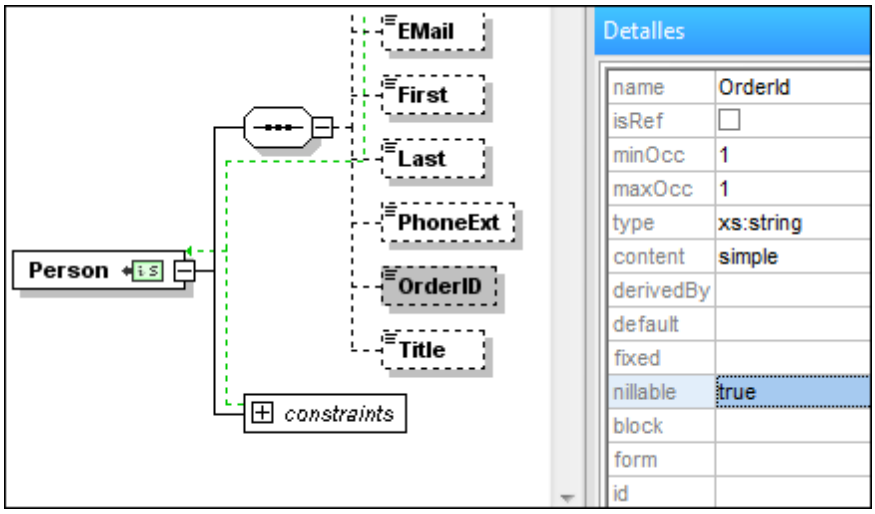
Diferencias entre 'xsi:nil' y 'nillable'
El atributo `xsi:nil="true"` se define en el **documento de instancia XML**.

```
14  <Person>
15    <PrimaryKey>2</PrimaryKey>
16    <ForeignKey>1</ForeignKey>
17    <EMail>biff@amail.com</EMail>
18    <First>biff</First>
19    <Last>bander</Last>
20    <PhoneExt>22</PhoneExt>
21    <OrderID xsi:nil="true"/>
22    <Title>IT services</Title>
23  </Person>
```

El atributo `xsi:nil="true"` indica que, aunque el elemento existe, no tiene ningún contenido. Recuerde que el atributo `xsi:nil="true"` afecta al valor de los elementos y no de los atributos. Los elementos que tienen el atributo `xsi:nil="true"` pueden tener más atributos, incluso si no tiene contenido.

El atributo `xsi:nil` no se presenta de forma explícita en la asignación gráfica de MapForce porque en la mayoría de los casos se gestiona automáticamente. Es decir, el nodo nulo (con el atributo `xsi:nil="true"`) existe, pero su contenido no.

Por el contrario, el atributo `nillable="true"` se define en el **esquema XML**. En MapForce puede estar presente tanto en el componente de origen como en el de destino.



Usar elementos que pueden ser nulos como origen de la asignación
MapForce comprueba el valor del atributo `xsi:nil` automáticamente cada vez que una asignación lee datos de elementos XML que tienen este atributo. Si el valor del atributo `xsi:nil` es `true`, se considera que su contenido no existe.

Cuando se crea una asignación **basada en destino** entre un elemento de origen que puede ser nulo (que tiene definido el atributo `xsi:nil`) y un elemento de destino que puede ser nulo y de **contenido simple** (un solo valor con atributos opcionales, pero sin elementos secundarios), MapForce añade el atributo `xsi:nil` al elemento de destino (p. ej. `<OrderID xsi:nil="true"/>`).

Cuando se crea una asignación de copia total entre un elemento de origen que puede ser nulo (que tiene definido el atributo `xsi:nil`) y un elemento de destino que puede ser nulo, MapForce añade el atributo `xsi:nil` al elemento de destino (p. ej., `<OrderID xsi:nil="true"/>`).

Para comprobar si un elemento de origen tiene el atributo `xsi:nil="true"`, utilice la función **is-xsi-nil**. Esta función devuelve `true` si el elemento es nulo y `false` si no lo es.

Para sustituir un elemento de origen nulo por otro valor, utilice la función **substitute-missing**.

Notas:
- Si se conecta la función **exists** con un elemento de origen nulo, la función devuelve `TRUE` porque el nodo de elemento existe aunque no tenga contenido.
- Si usa funciones que esperan valores simples (como las funciones **multiply** y **concat**) en elementos que tienen definido el atributo `xsi:nil`, no se obtiene ningún resultado porque no hay ningún contenido y no se puede extraer ningún valor. Por tanto, en este caso estas funciones se comportan como si el nodo de origen no existiera.

Usar elementos que pueden ser nulos como destino de la asignación
Cuando se crea una asignación **basada en destino** entre un elemento de origen que puede ser nulo y un elemento de destino que puede ser nulo de **contenido simple** (un solo valor con atributos opcionales, pero sin elementos secundarios), MapForce inserta el atributo `xsi:nil` en el elemento de destino (p. ej. <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. En casos como este puede definir una **conexión de copia total** para copiar el atributo `xsi:nil` en el elemento de destino.

Cuando se crea una asignación entre una **secuencia vacía** y un elemento de destino, el elemento no se crea en el destino, independientemente de su atributo `nil`.

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.

Si el nodo es de tipo simple, 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.

Nota: las funciones que generan `xsi:nil` no pueden pasarse a través de funciones ni componentes que solo operan en valores (como la función **if-else**, por ejemplo).

© 2014 Altova GmbH

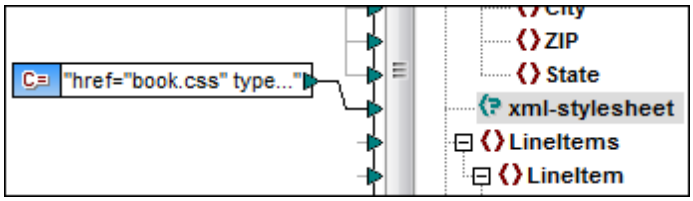
9.1.5 Comentarios e instrucciones de procesamiento

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.

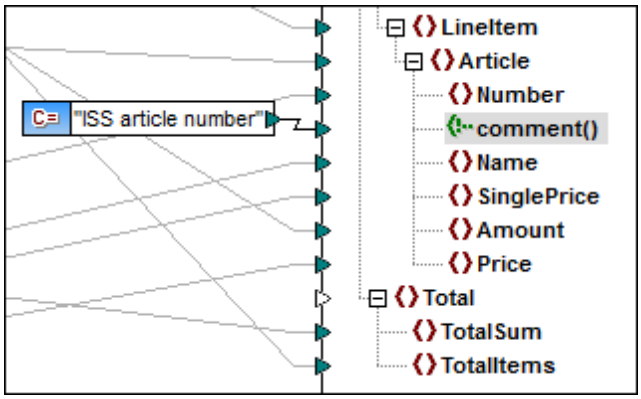


3. Ahora puede usar un componente de constante para suministrar el valor del atributo de la instrucción de procesamiento (p. ej. `href="book.css" Type="text/css"`).

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.

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.

© 2014 Altova GmbH

9.1.6 Secciones CDATA

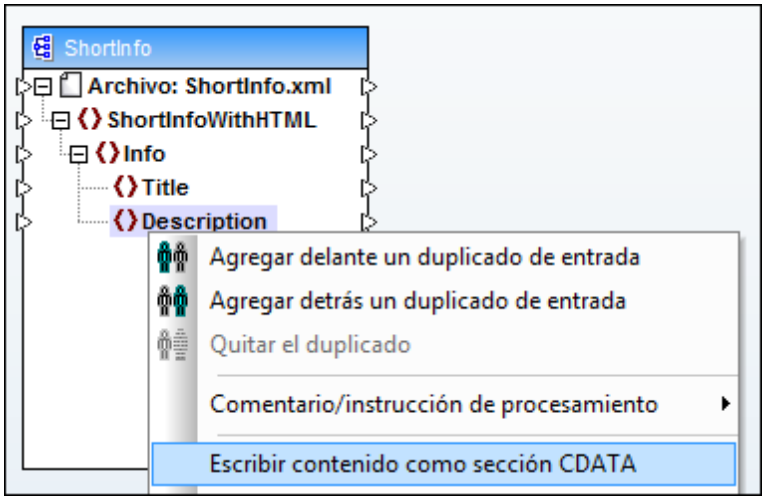
Las secciones CDATA sirven para excluir bloques de texto que contienen caracteres que de lo contrario serían interpretados como código. Las secciones CDATA empiezan con `<![CDATA[` y terminan con `]]>`.

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.

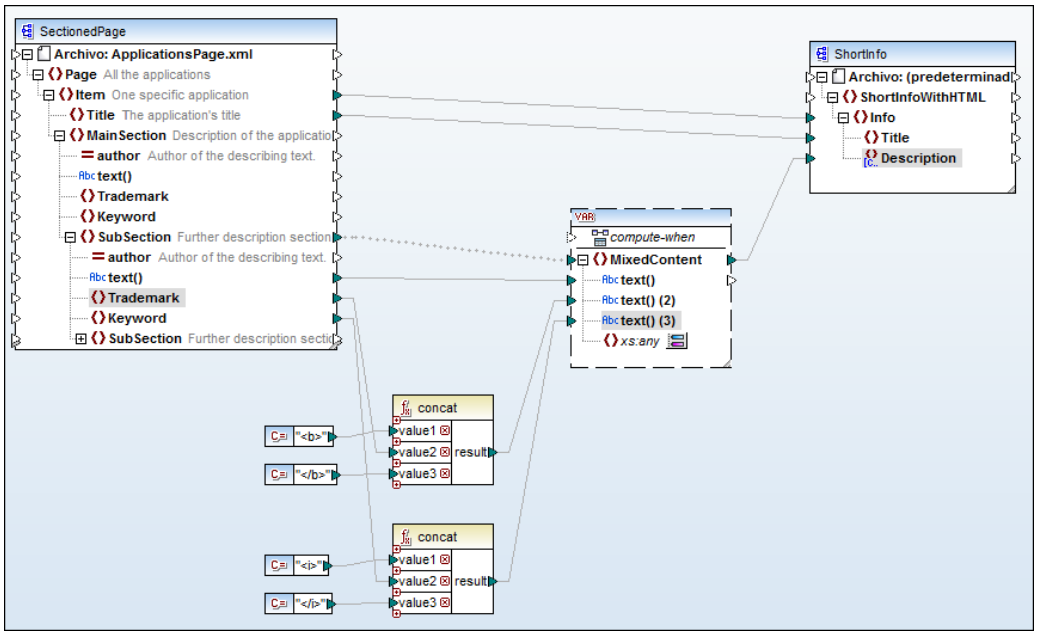
Nota: también puede definir secciones CDATA en nodos duplicados y en nodos xsi:type.

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`**.

- 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.

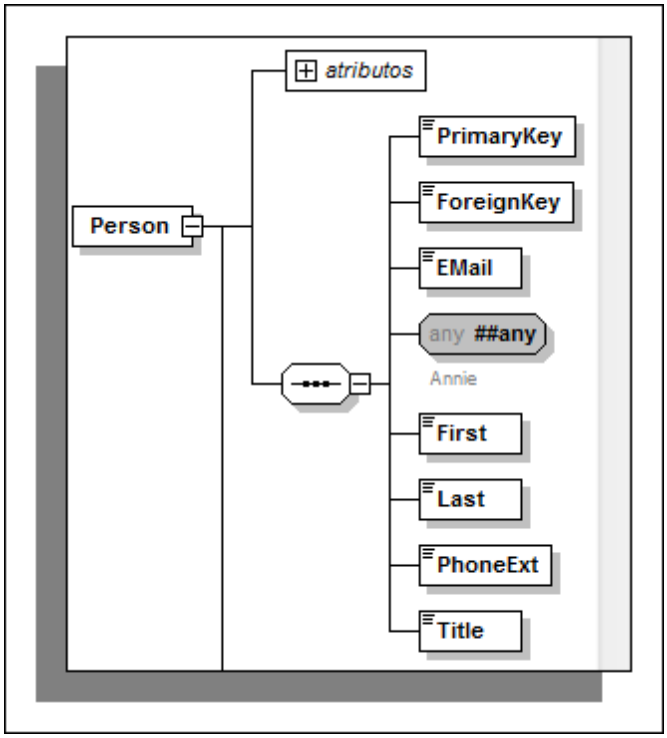


Al hacer clic en la pestaña **Resultados** aparece la sección CDATA que contiene el texto con marcado.

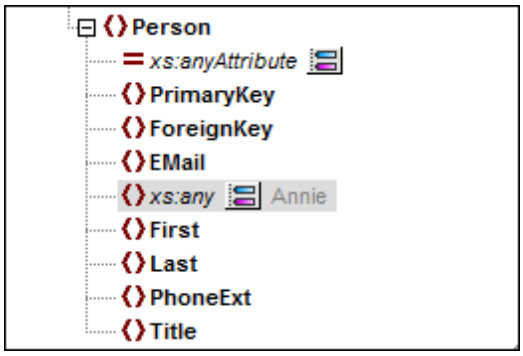
```
7    <Info>
8      <Title>MapForce</Title>
9      <Description><![CDATA[Altova <b>MapForce</b> 2014 Enterprise Edition is the premier <i>XML</i>
/ <i>database</i> / <i>flat file</i> / <i>EDI</i> data mapping tool that auto-generates mapping code in
<i>XSLT</i> 1.0/2.0, <i>XQuery</i> , <i>Java</i> , <i>C++</i> and <i>C#</i> . It is the definitive tool for
data integration and information leverage.]]></Description>
10   </Info>
```

9.1.7 Comodines xs:any / xs:anyAttribute

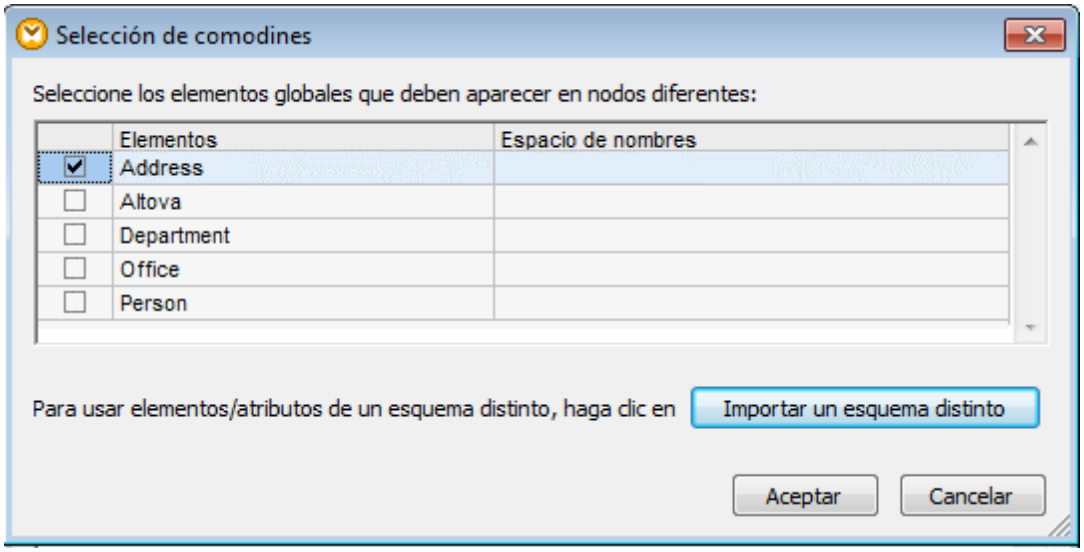
Los comodines `xs:any` (y `xs:anyAttribute`) permiten usar cualquier elemento o atributo del esquema XML correspondiente. Por ejemplo, en la imagen siguiente puede ver el elemento "any" en la vista **Esquema** de XMLSpy.



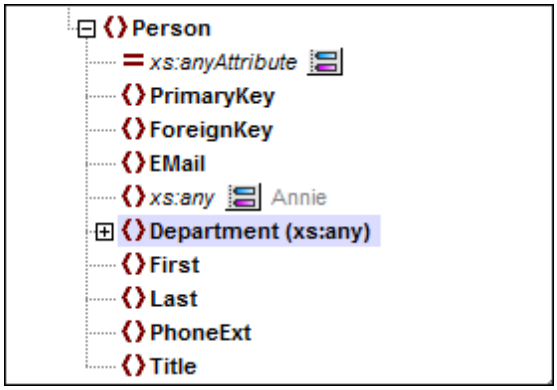
En MapForce, la estructura de este esquema puede verse haciendo clic en el botón de selección situado a la derecha del elemento `xs:any` (y de `xs:anyAttribute`).



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.



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.

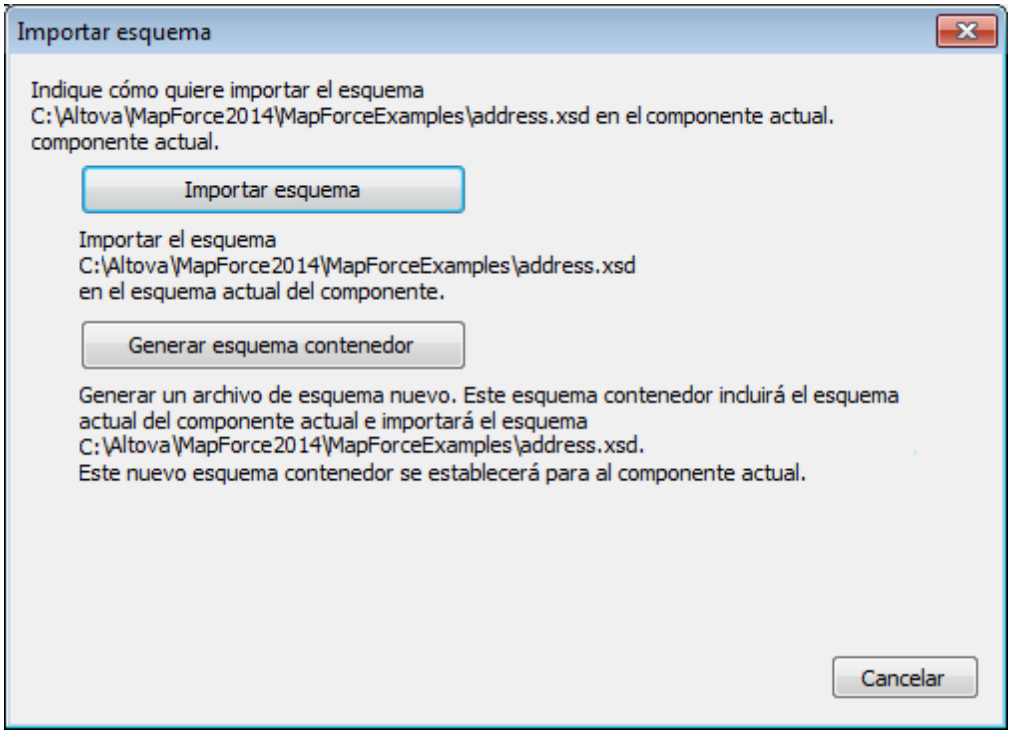
Para eliminar un elemento comodín:

- 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.

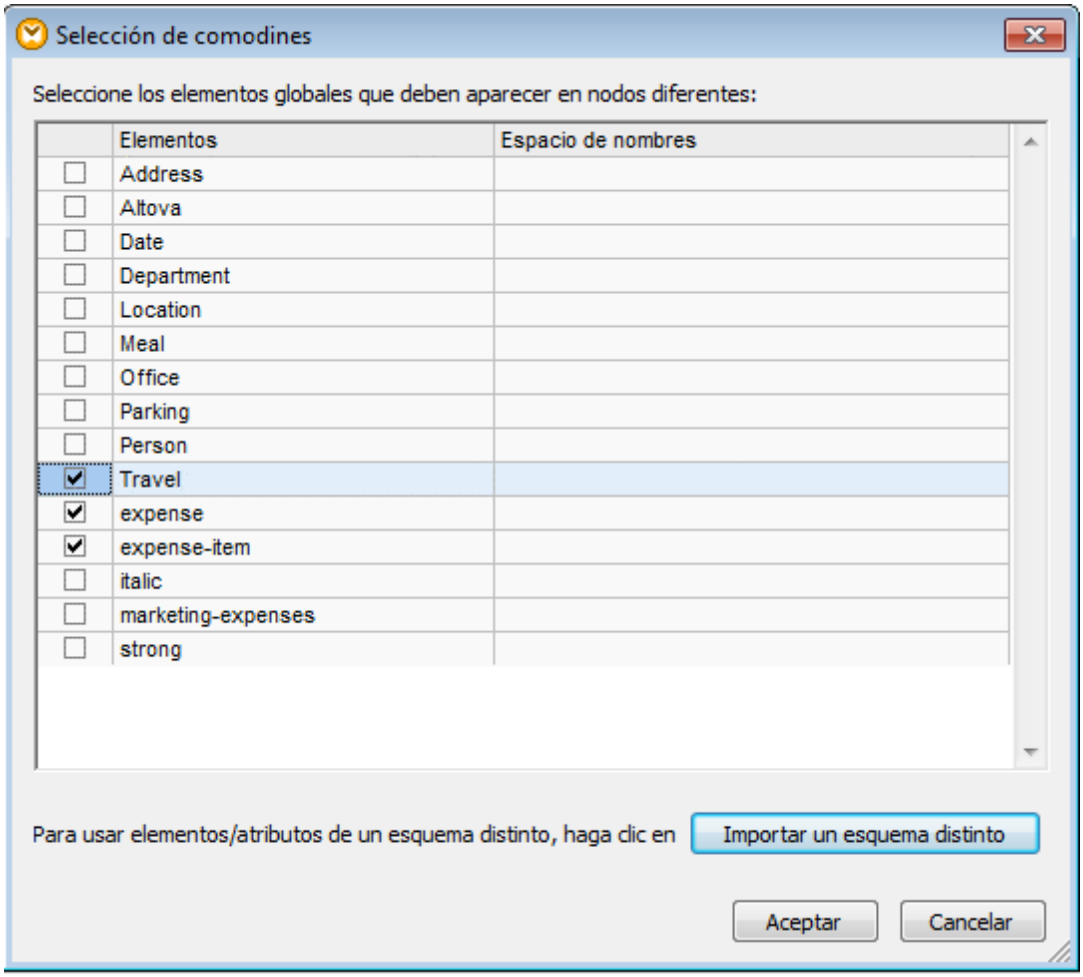
 © 2014 Altova GmbH



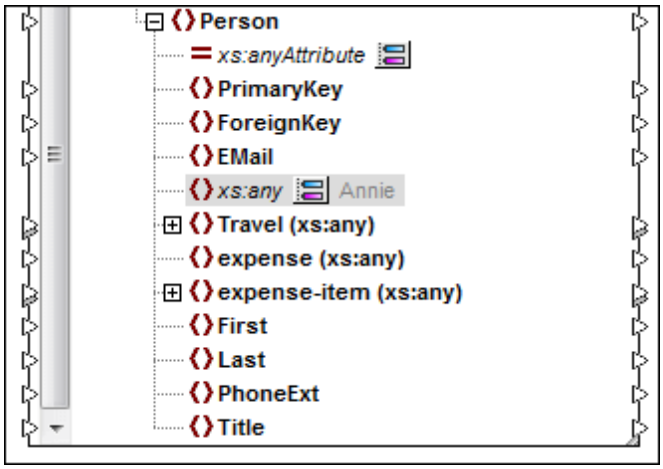
3. Haga clic en el botón correspondiente.

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`.



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.

9.2 Asignación de datos entre HL7 v3 y esquemas XML

MapForce 2015 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\Common2015\Schemas\ hl7v3`
Equipos Windows	`C:\Archivos de programas\Altova\Common2015\Schemas\ hl7v3`
Equipos Windows7	`C:\Archivos de programas\Altova\Common2015\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.

 © 2014 Altova GmbH

Altova MapForce 2015

Consejos prácticos

10 Consejos prácticos

En esta sección se describen tareas corrientes de la creación de asignaciones de datos.

General:

Cómo puedo...	Consulte este apartado
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
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	Asignación de valores: transformar datos de entrada
Ejecutar MapForce desde la **línea de comandos**	Parámetros de la línea de comandos
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
Probar nodos: nodos existentes / nodos no existentes	Comprobación de nodos: existe/no existe

 © 2014 Altova GmbH

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
Definir el nodo que debe actuar como nodo de contexto en un archivo de origen	Nodo de contexto prioritario (priority context)

10.1 Componentes de filtrado: consejos

Para más información sobre el filtrado de datos XML, consulte el apartado Filtrado de datos.

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.

 © 2014 Altova GmbH

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.

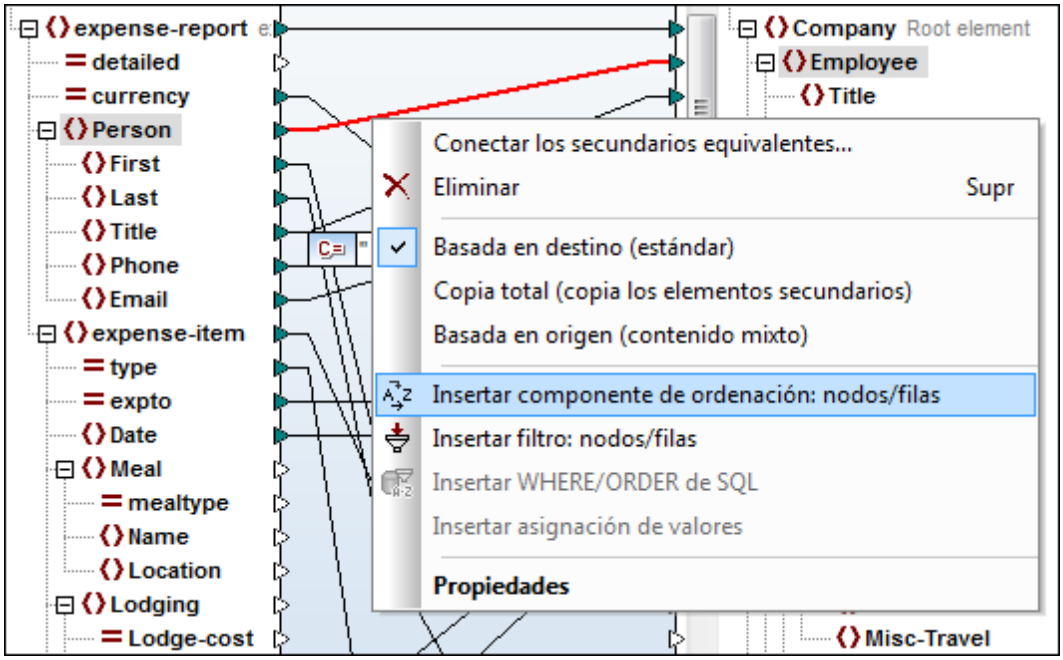
Si un componente de excepción se conecta a una de las salidas del filtro, la condición de excepción se comprueba cuando se ejecutan las asignaciones a las otras salidas del filtro.

10.2 Componentes de ordenación: ordenar secuencias de entrada

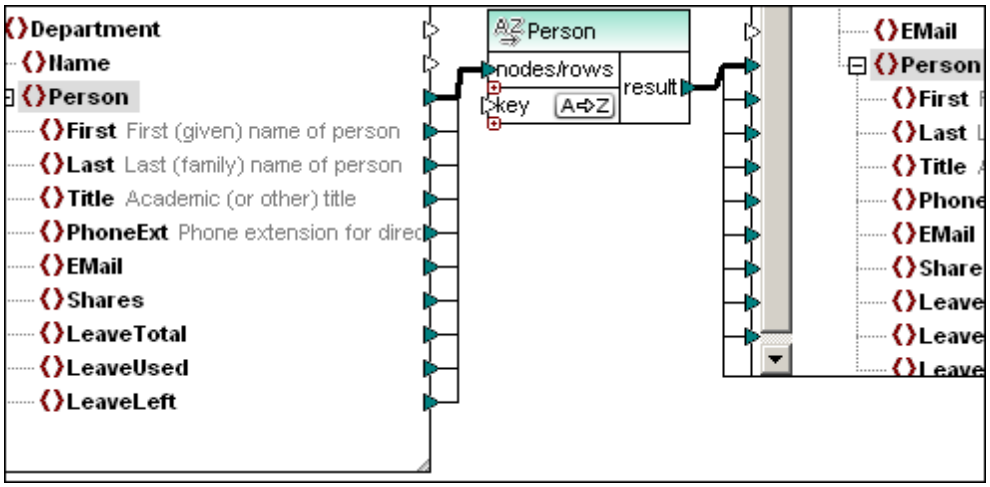
Para ordenar datos de entrada según un criterio de ordenación, use un componente de **ordenación**. Este componente se puede usar con XSLT2, XQuery y con el motor de ejecución integrado.

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**.

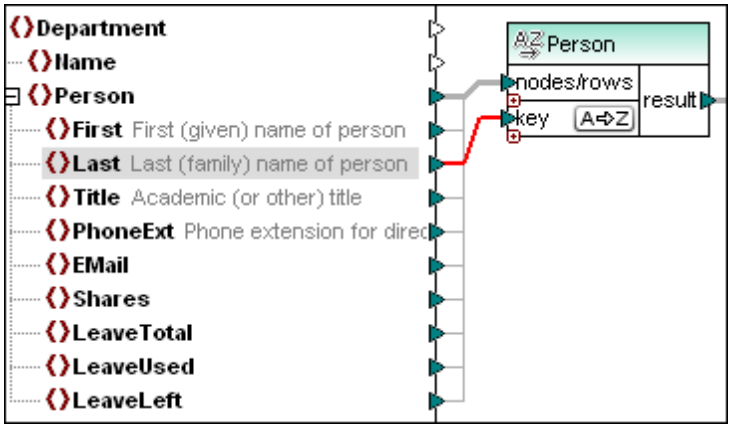


Esto inserta un componente de ordenación, que se conecta automáticamente a los componentes de origen y destino.

Para definir el criterio de ordenación:

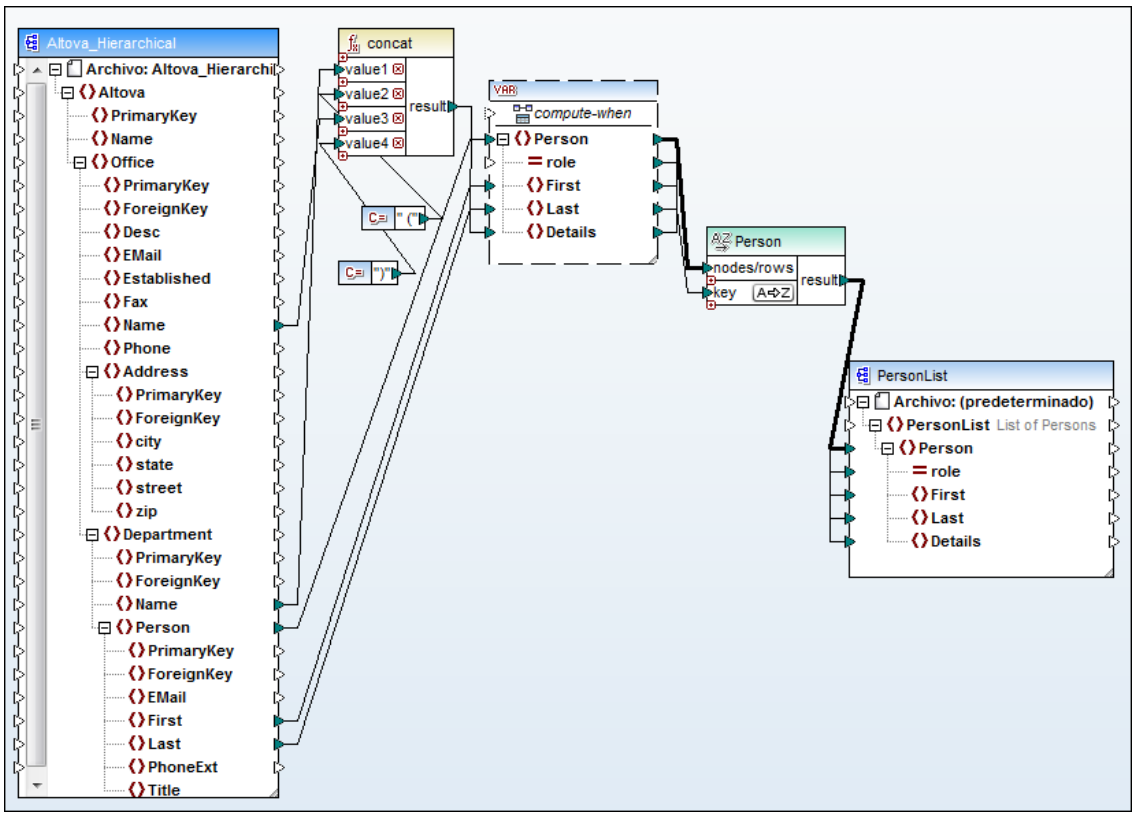
 © 2014 Altova GmbH

- 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**).



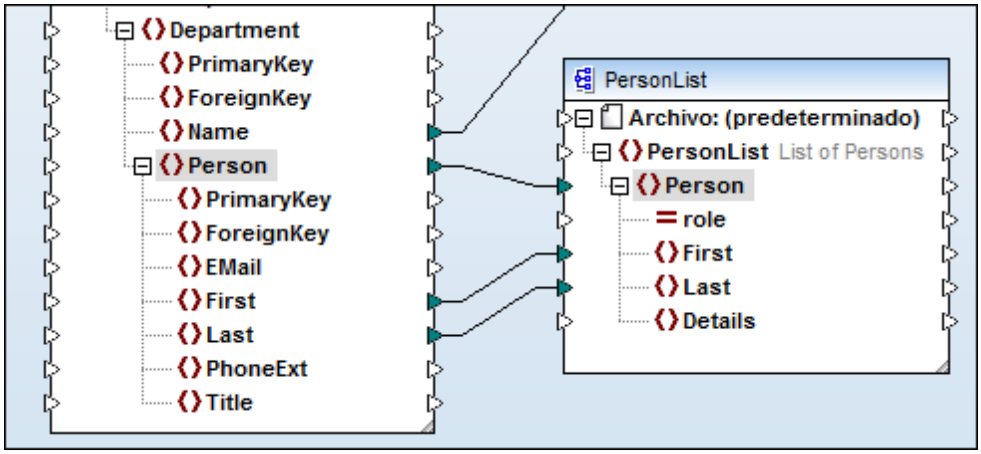
En los resultados de la asignación los elementos **Person** aparecerán ordenados según el elemento **Last**.

Tomemos como ejemplo el archivo `Altova_Hierarchical_Sort.mfd` de la carpeta `MapForceExamples`.



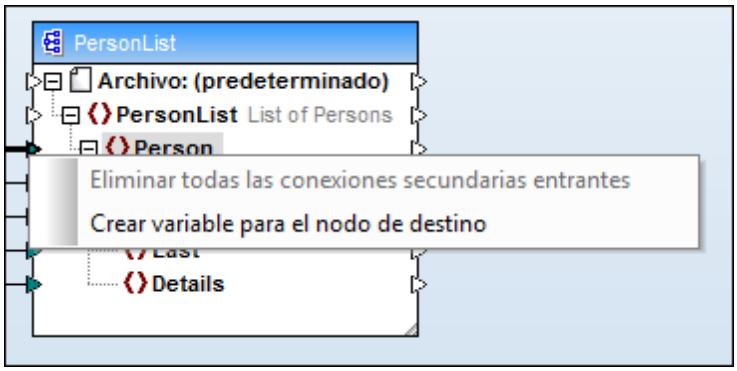
El objetivo es ordenar alfabéticamente las personas de cada oficina e incluir información detallada sobre los nombres de departamento y sobre la oficina. Para ello, este ejemplo usa una variable que permite acceder a los elementos primarios (primarios del nodo **Person**).

Este era el estado inicial de esta asignación:

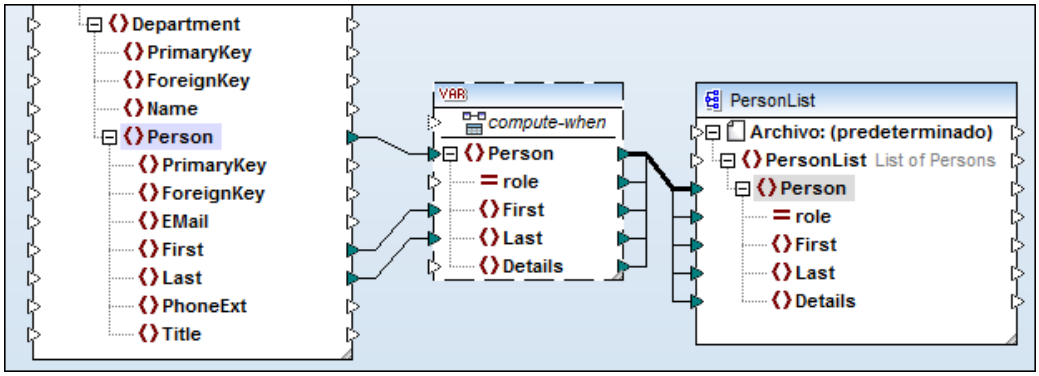


Puesto que deseamos ordenar los elementos **Person**, se conectan los elementos **Person** de ambos componentes (y también se conectan los secundarios que tienen el mismo nombre).

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**.



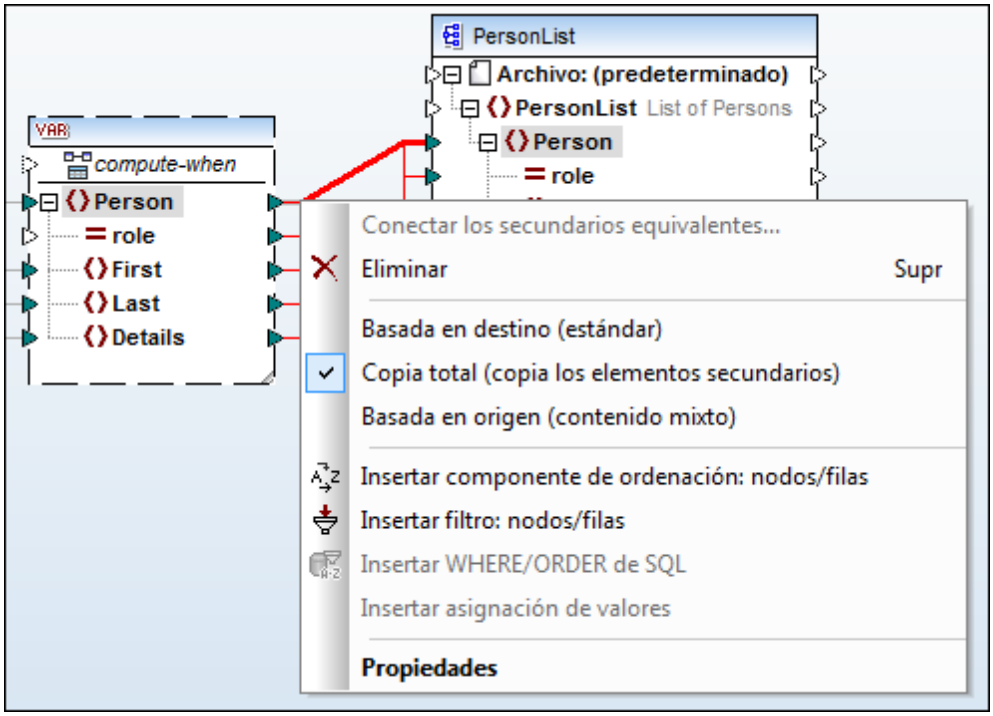
Esto inserta una variable intermedia compleja entre los componentes y se conecta a los elementos secundarios que tienen el mismo nombre. Para más información consulte el apartado Variables intermedias.



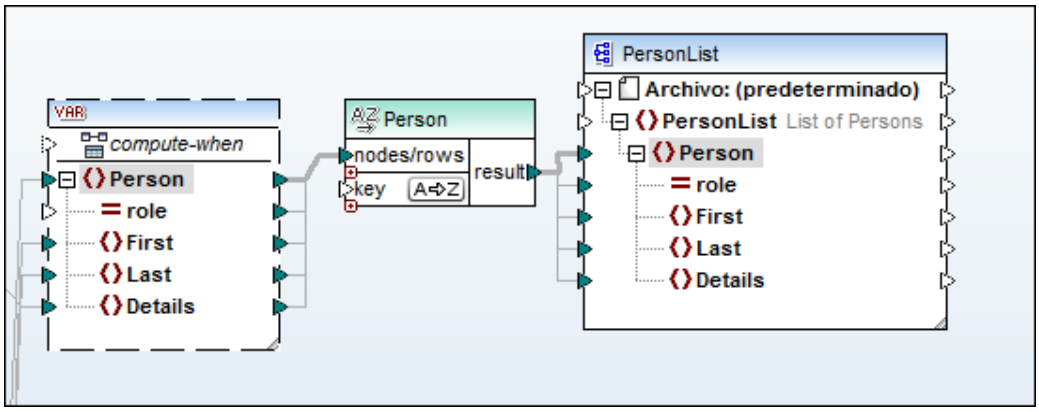
Puesto que queremos ordenar los elementos **Person** según su elemento **Last**, ahora insertamos el componente de ordenación.

 © 2014 Altova GmbH

3. Haga clic con el botón secundario en el conector de copia total y seleccione **Insertar componente de ordenación: nodos/filas**.

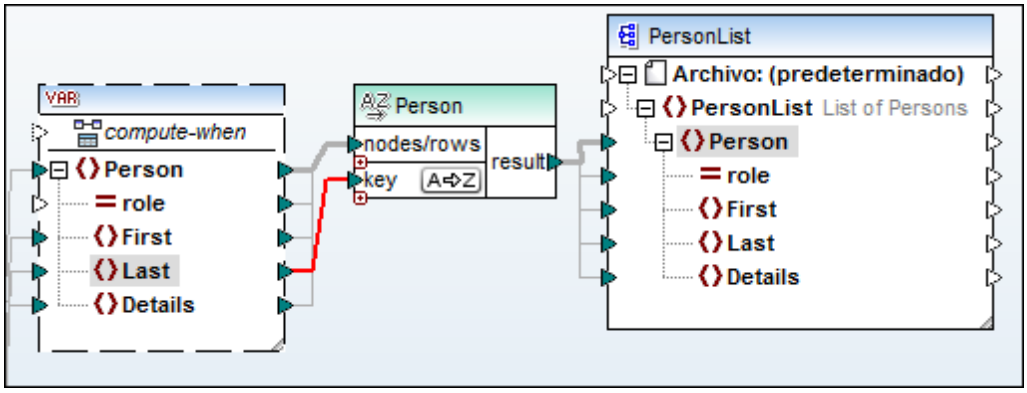


Esto inserta el componente de ordenación, que automáticamente se conecta a los elementos pertinentes.

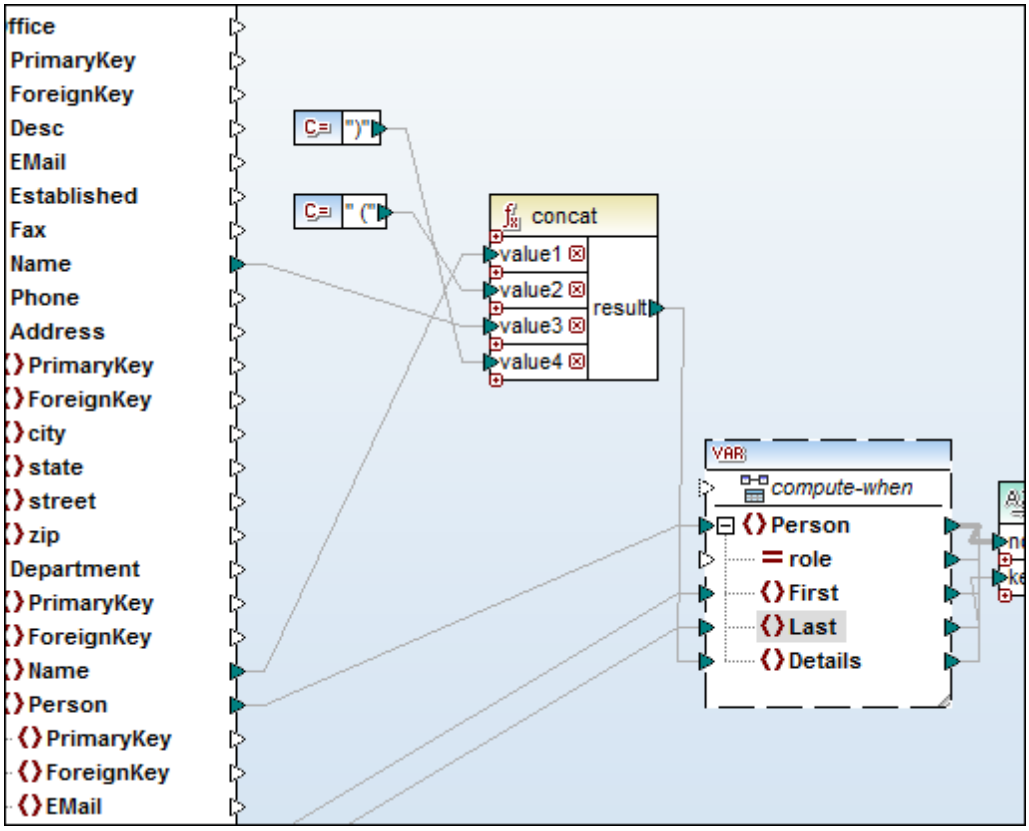


Lo única que falta es definir el criterio de ordenación.

4. Conecte el elemento **`Last`** de la variable intermedia al parámetro **`key`** del componente de ordenación.



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.

 © 2014 Altova GmbH

```
<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>
```

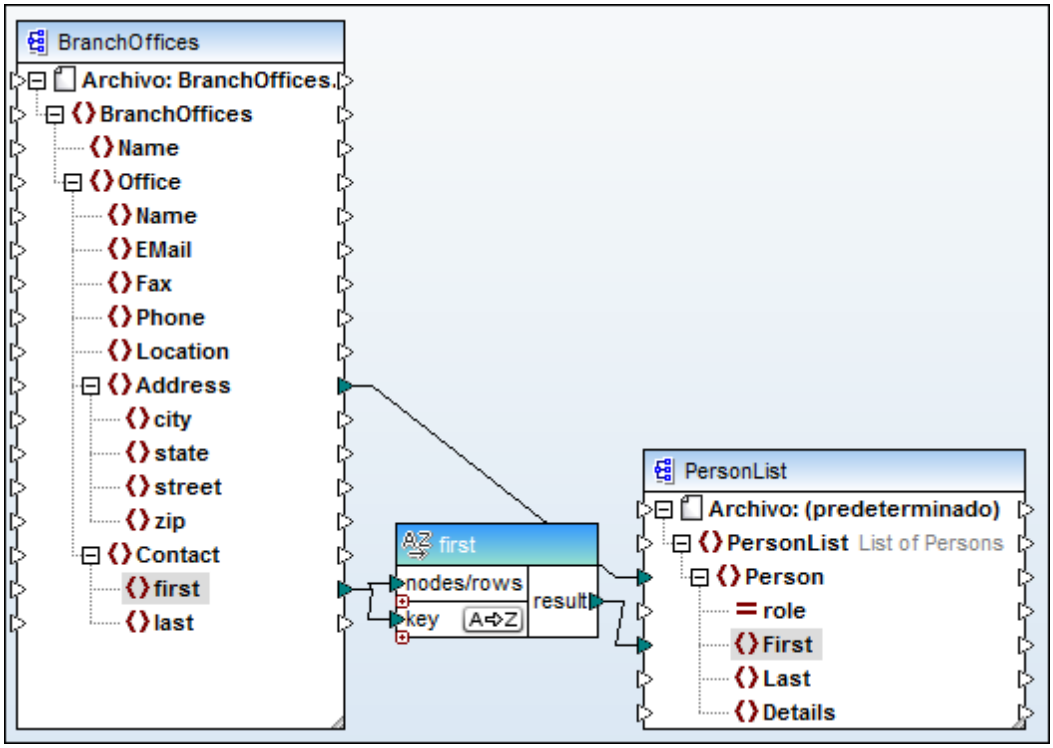
Ahora las personas siguen estando ordenadas por apellido, pero se incluye información adicional (del elemento **`Details`**). El nombre de la oficina y del departamento aparece ahora debajo de cada persona porque la variable intermedia permite acceder a los datos primarios de cada nodo secundario. Esto solamente es posible gracias a la variable intermedia.

Para invertir la secuencia de clasificación:

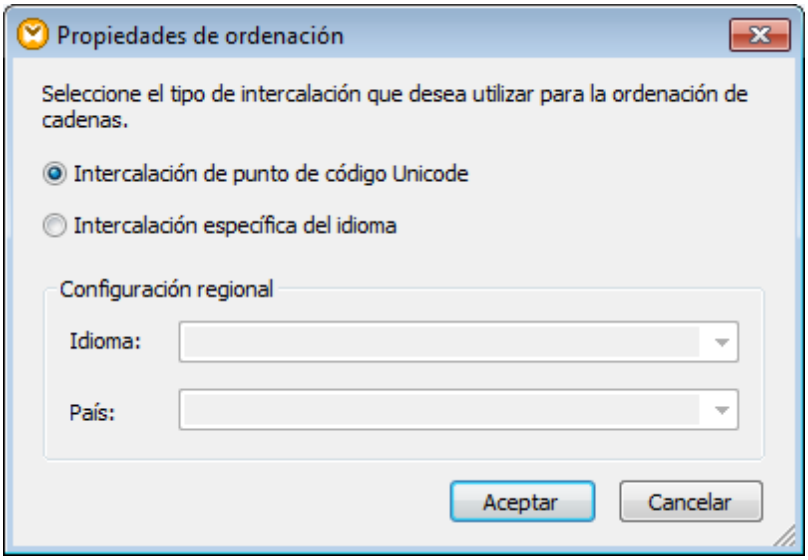
- Haga clic en el icono A⇨Z del componente de ordenación. Ahora el icono es Z➔A, indicando que se invirtió la secuencia.

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.



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.

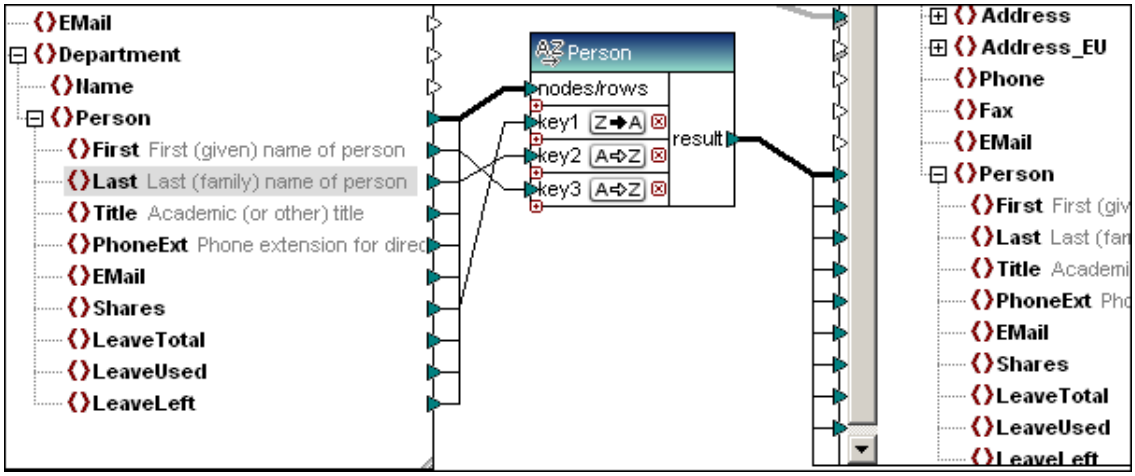
Intercalación específica del idioma: esta opción permite definir el idioma y la variación del

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.

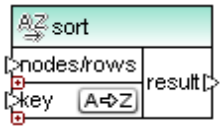
- Al hacer clic en el icono **+** se añade un criterio nuevo de ordenación (es decir, **key2**) en el componente.
- Al hacer clic en el icono **x** se elimina el criterio correspondiente.
- Al colocar un conector sobre un icono **+** se inserta/añade automáticamente un criterio nuevo conectado al elemento 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`**.

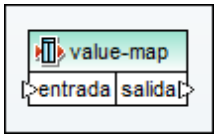
10.3 Asignación de valores: transformar datos de entrada

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.

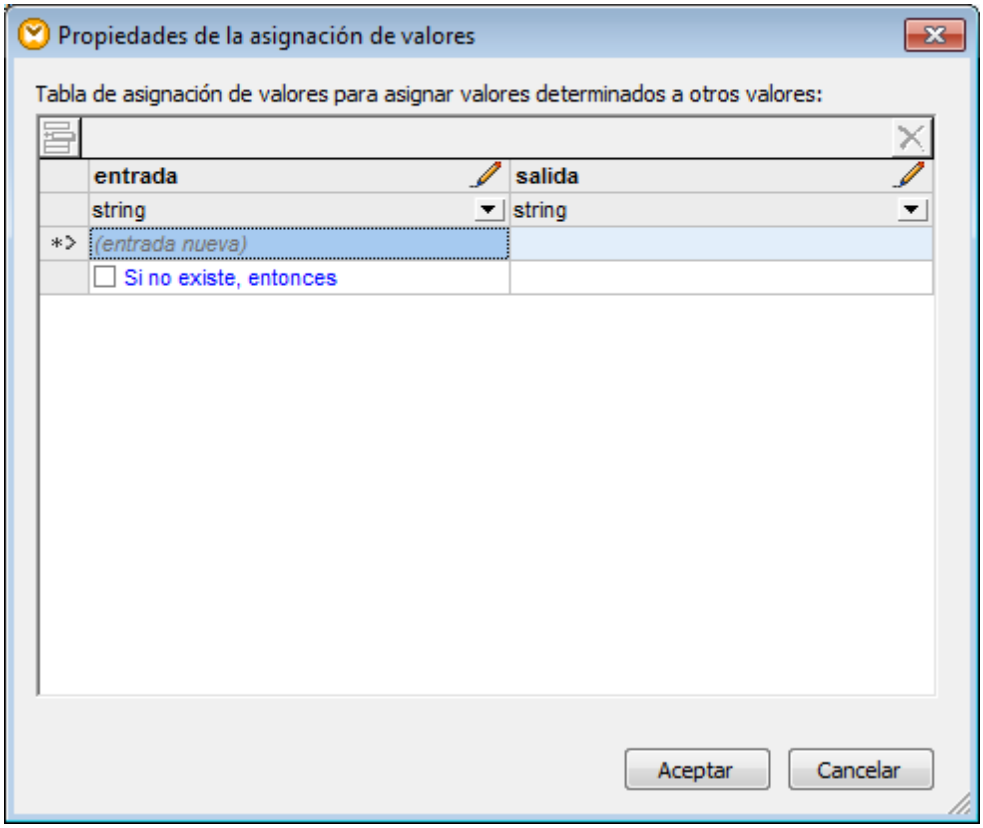
Nota: si quiere recuperar/filtrar solamente ciertos datos y no otros, use un componente de **filtrado**. Para más información consulte Filtrado de datos.

Para usar una asignación de valores:

1. Seleccione la opción de menú **Insertar | Asignación de valores** o haga clic en el icono .

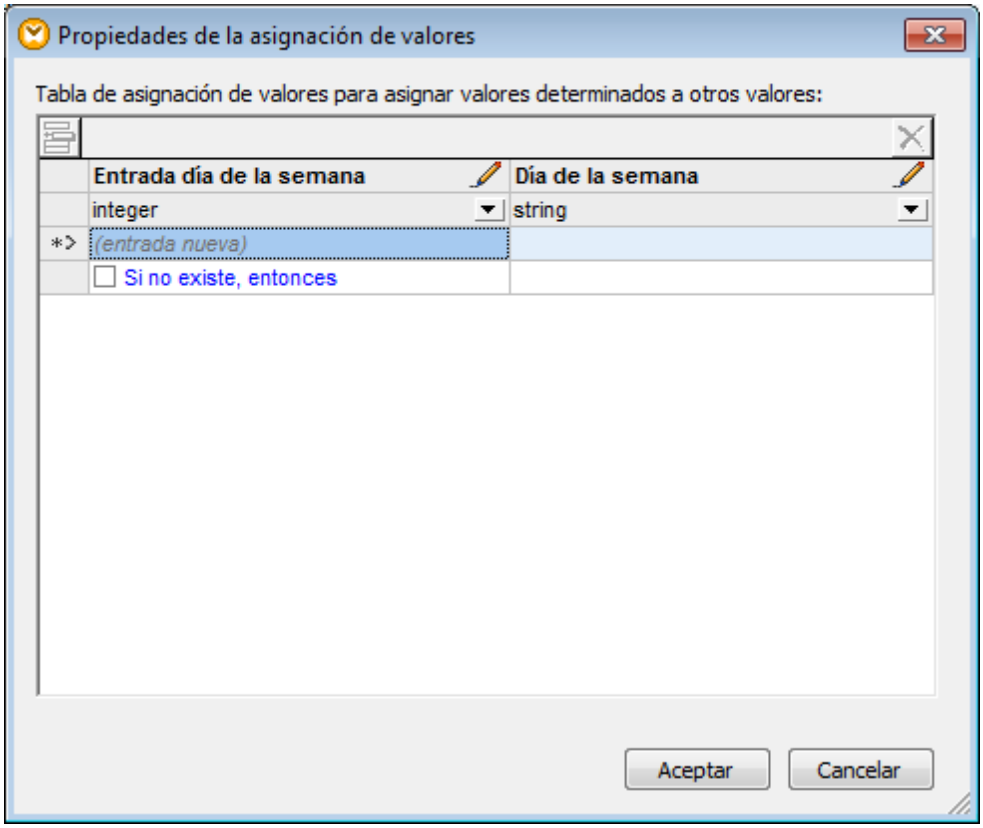


2. Haga doble clic en la asignación de valores (componente **`value-map`**) para abrir la tabla de la asignación de valores.

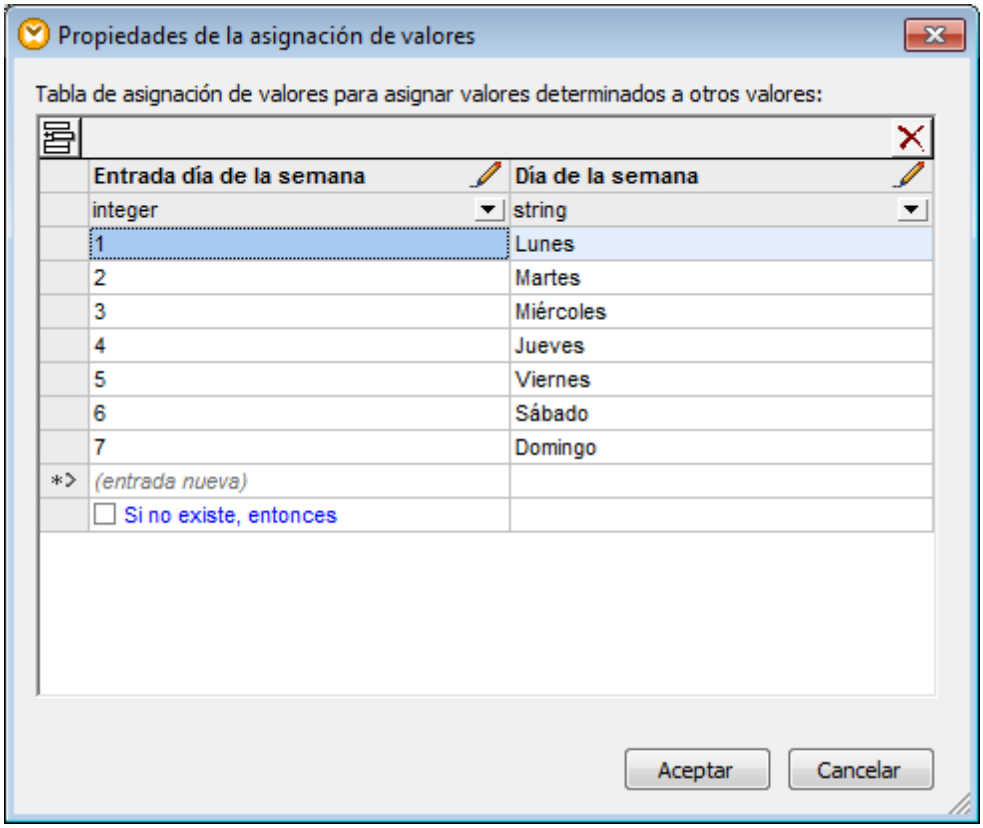


3. Haga clic en los encabezados de columna y escriba **`Entrada día de la semana`** en la primera columna y **`Día de la semana`** en la segunda.

 © 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`**).

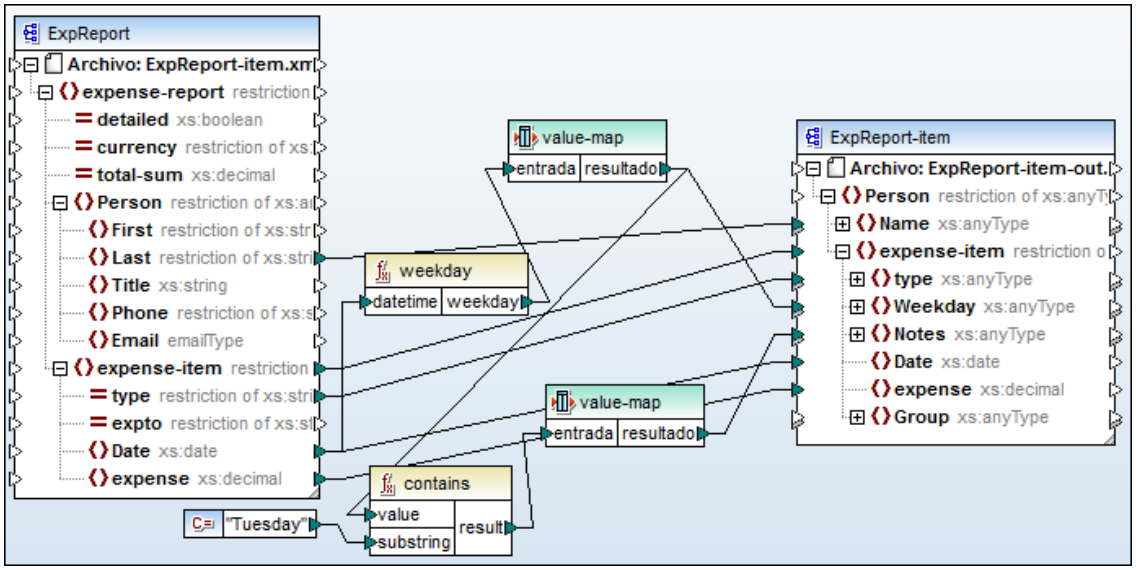


Nota: marque la casilla *Si no existe, entonces* y escriba un valor para definir un valor de salida alternativo en caso de que los valores suministrados no existan en la entrada. Para pasar a través de datos de origen sin cambiarlos, consulte el apartado Pasar datos sin cambiarlos a través de una asignación 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)

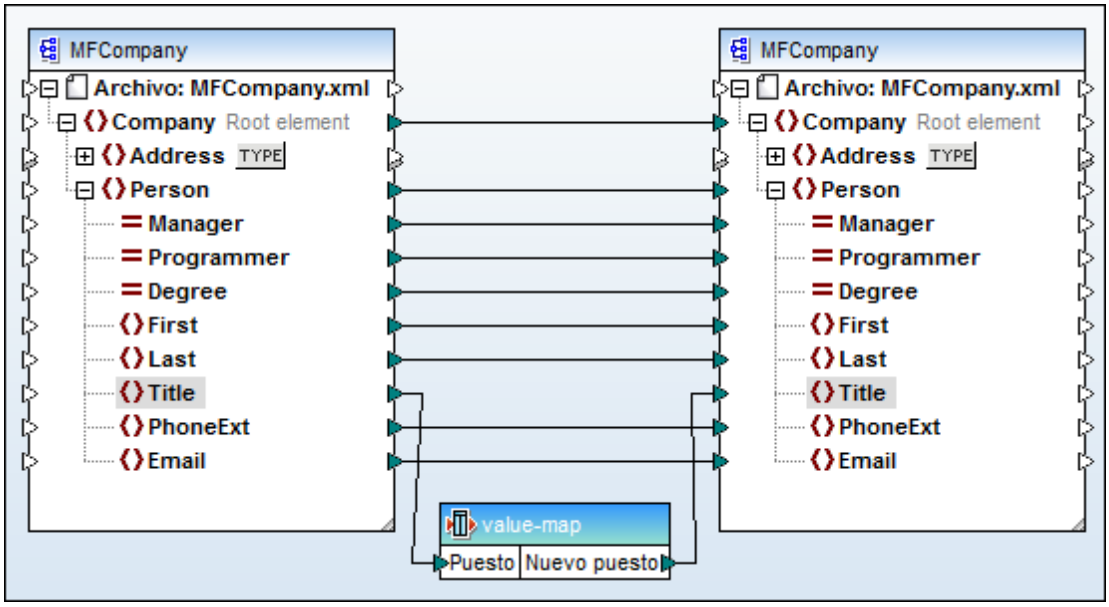
solamente puede ser el valor booleano **`true`** o **`false`**. Por tanto, en el campo **`entrada`** de la tabla de la asignación de valores debe introducir el valor que desea probar, p. ej. **`true`**.

© 2014 Altova GmbH

10.3.1 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.

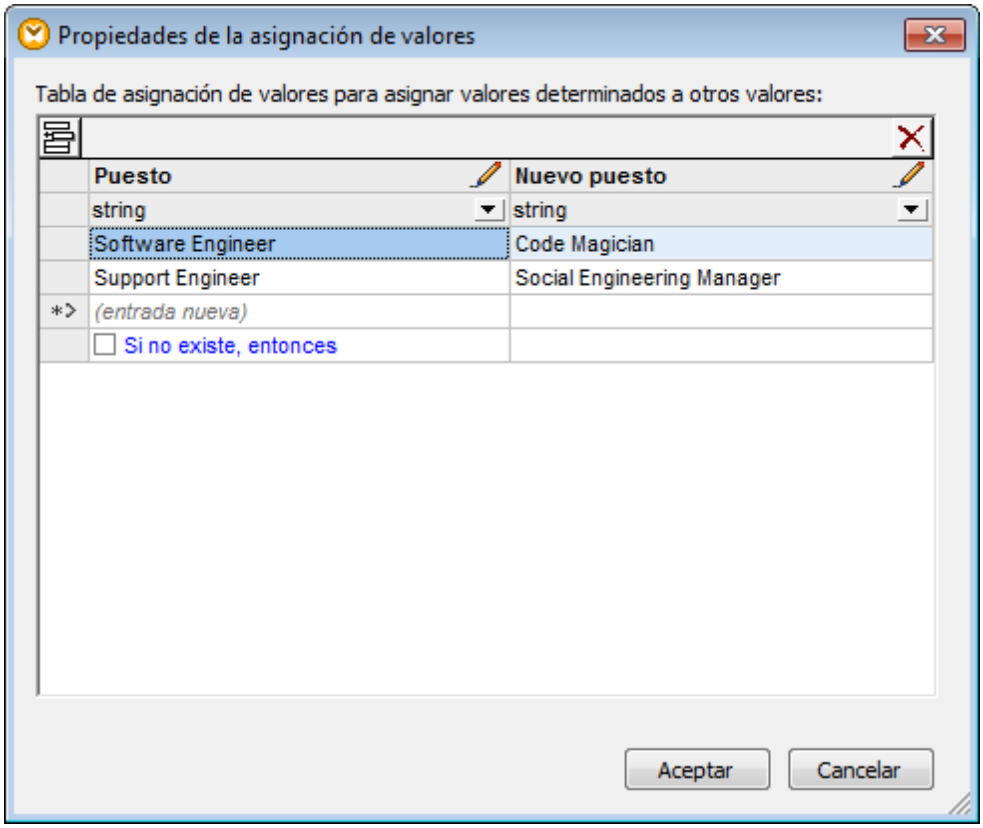
Por ejemplo, imaginemos que una compañía tiene una subsidiaria y necesita cambiar el nombre de dos puestos de trabajo de la subsidiaria y conservar el nombre de los demás puestos tal y como están.



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>
```

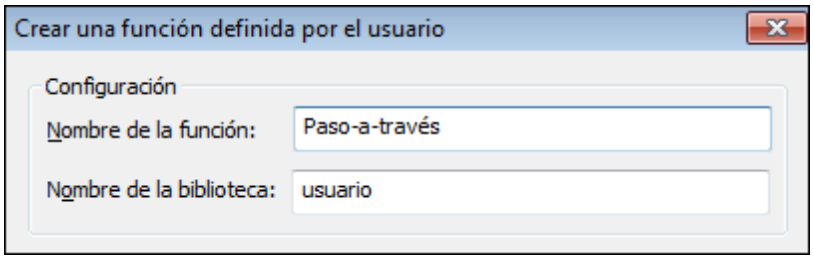
En el caso de empleados que no tengan ninguno de los dos puestos del componente de asignación de valores, el elemento **`Title`** se elimina en el archivo de salida.



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`**.

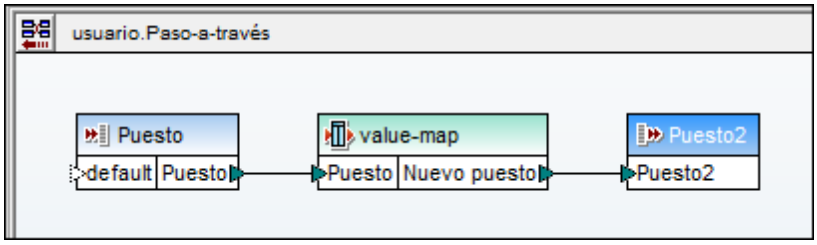
Solución:
Crear una función definida por el usuario que contenga el componente de asignación de valores y usar la función **`substitute-missing`** para suministrar los datos originales para los nodos vacíos.

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**.

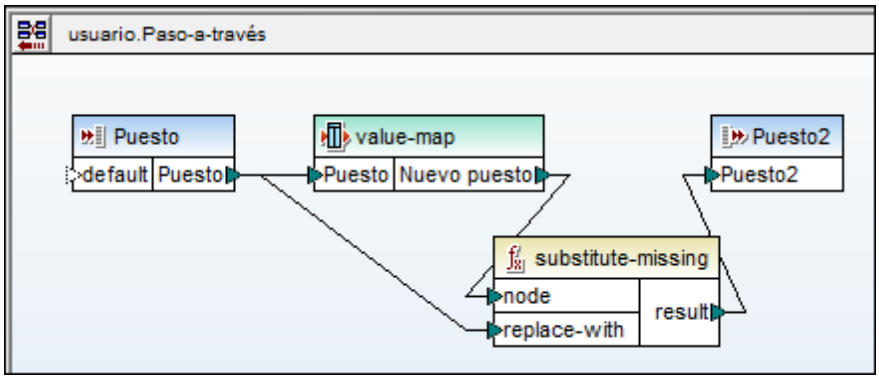


2. Escriba el nombre de la función, p. ej. Paso-a-través y haga clic en **Aceptar**.

 © 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

10.3.2 Propiedades de las asignaciones de valores

Iconos:

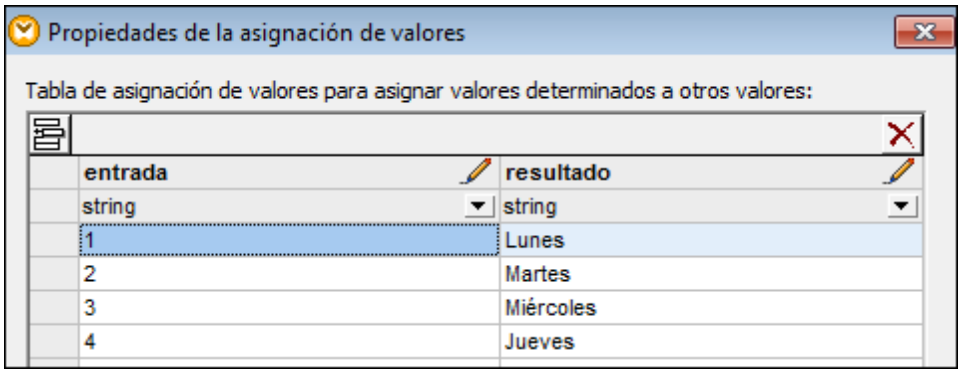
Inserta una fila nueva antes de la fila activa.

Elimina la fila activa.

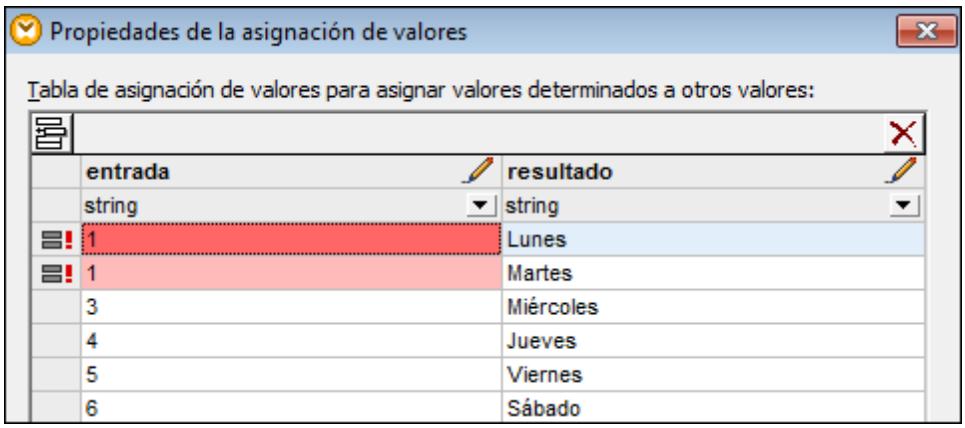
Permite **editar** el encabezado de la columna.

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).



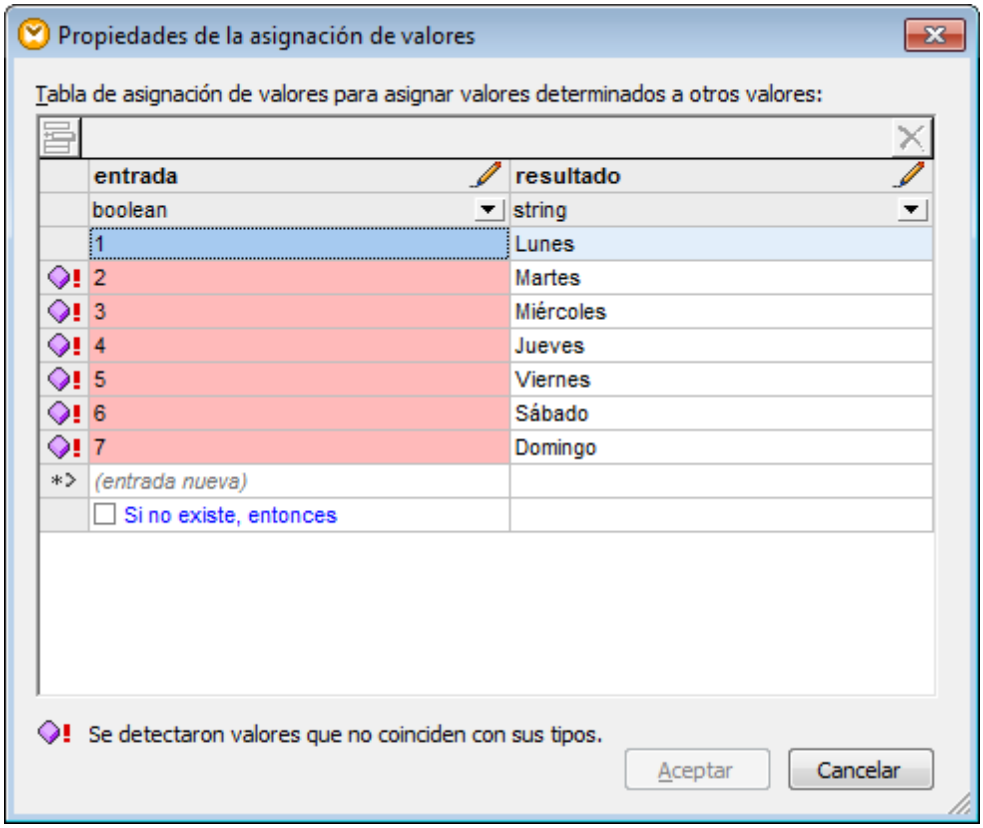
Usar valores de entrada únicos:
Los valores introducidos en la columna de entrada deben ser únicos. Si introduce dos veces el mismo valor, ambos se resaltan para que corrija el error.



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`.

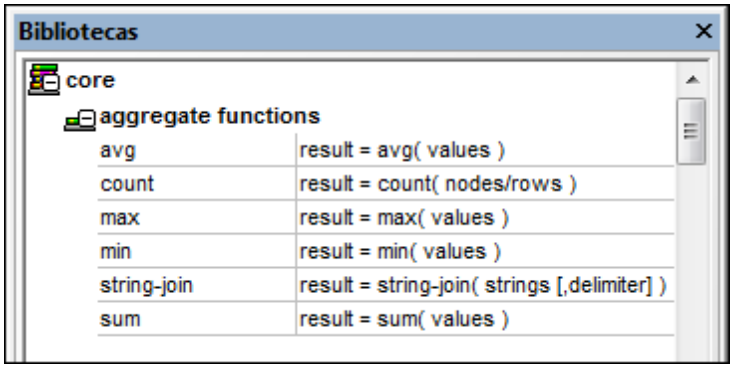


 © 2014 Altova GmbH

10.4 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`.

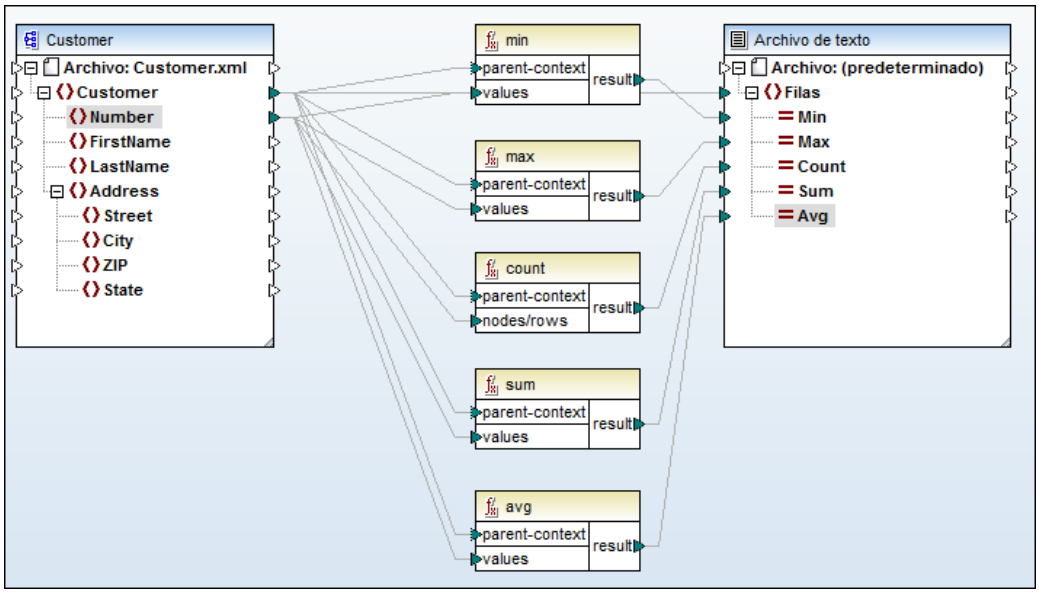
Consulte también el apartado Funciones de agregado: sumar nodos en XSLT 1.0/2.0 para aprender a crear funciones de agregado usando plantillas con nombre.



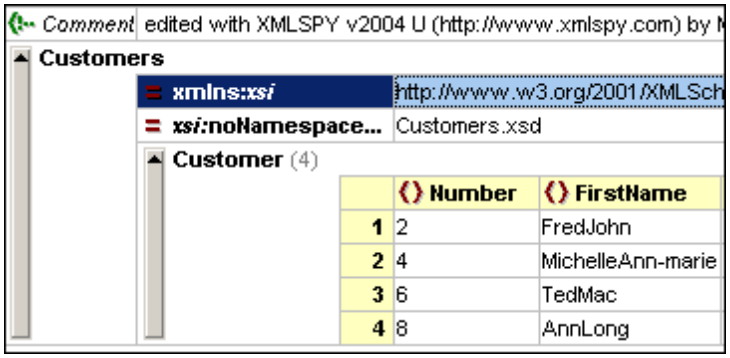
Con Java seleccionado

Las funciones de agregado tienen dos entradas:

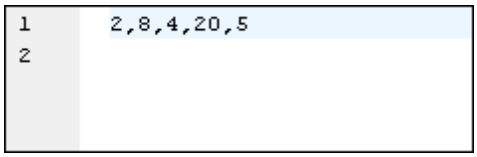
- la entrada **`values`** (**`nodes/rows`**) se conecta al elemento del componente de origen que suministra los datos, en este caso el elemento **`Number`**.
- la entrada **`parent-context`** se conecta al elemento en el que se desea iterar (es decir, el contexto. En este caso se quiere iterar en todos 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

10.5 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 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.

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.

10.6 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:

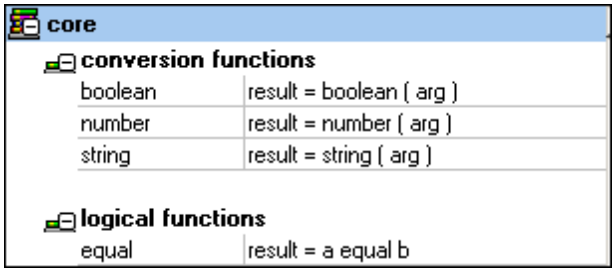
El tipo de datos utilizado para la comparación **siempre es el más general, es decir, el menos restrictivo**, de los dos tipos de entrada.

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

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**).

10.7 Nodo de contexto prioritario

Cuando aplicamos una función a distintos elementos del esquema, 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 datos de origen debe realizarse la búsqueda primero o qué nodo debe recorrerse primero y establezca el contexto prioritario en su conector.

El ejemplo de la imagen siguiente es una versión simplificada del archivo `CompletePO.mfd`, disponible en la carpeta `...\MapForceExamples`.

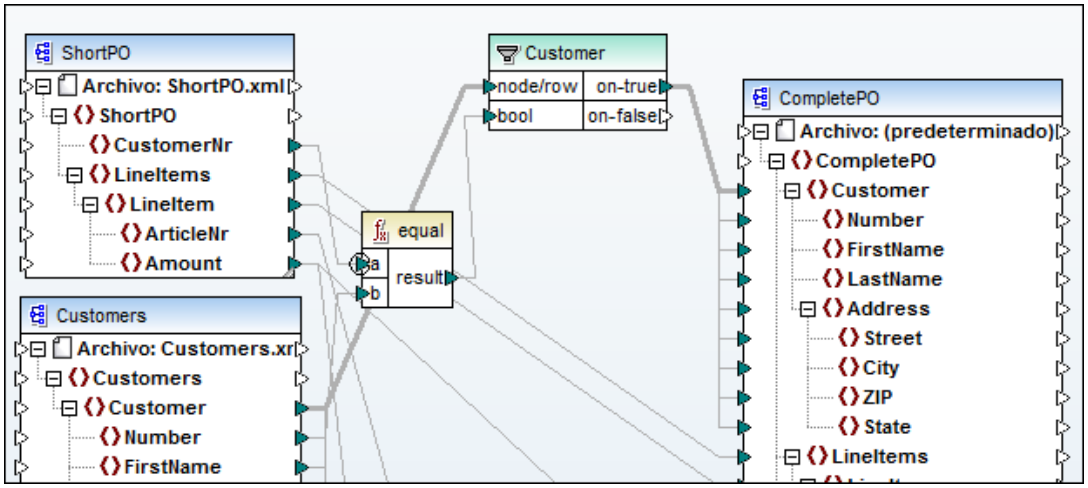
Observe que esta asignación tiene varios componentes de origen. **ShortPO, Customers** y **Articles** son esquemas que tienen asociados archivos XML de instancia. Los datos de todos estos componentes se asignan al archivo/esquema XML **CompletePO**. El nodo de contexto prioritario aparece señalado con un círculo.

- El nodo **`CustomerNr`** del esquema de origen **ShortPO** se compara con el elemento **`Number`** del archivo **Customers**.
- **`CustomerNr`** se designa como **contexto prioritario** y se coloca en el parámetro **`a`** de la función **`equal`**.
- En el archivo **Customers** se busca el **mismo** número **una vez**. El parámetro **`b`** contiene el elemento **`Number`** del archivo **Customers**.
- Si se encuentra el número, el resultado se pasa al parámetro **`bool`** de la función **`filter`**.
- El parámetro **node/row** pasa los datos de **Customers** a **`on-true`** cuando el parámetro **`bool`** es **`true`**, es decir, cuando se encuentra el mismo número.
- Se pasan los demás datos de **Customers** (elementos **`Number`**, **`FirstName`**, **`LastName`**) y todos se conectan a los elementos correspondientes del esquema de destino.

Al designar como **contexto prioritario** el parámetro **`b`** de la función **`equal`** (es decir, el elemento **`Number`**) conseguimos esto:

- MapForce carga el primer **`Number`** en el parámetro **`b`**
- Lo compara con el elemento **`CustomerNr`** del parámetro **`a`**
- Si no son iguales, MapForce carga el siguiente **`Number`** en el parámetro **`b`** y lo comprueba con **`a`**
- Se itera en cada elemento **`Number`** del archivo y se intenta buscar ese número en **`ShortPO`**.

 © 2014 Altova GmbH

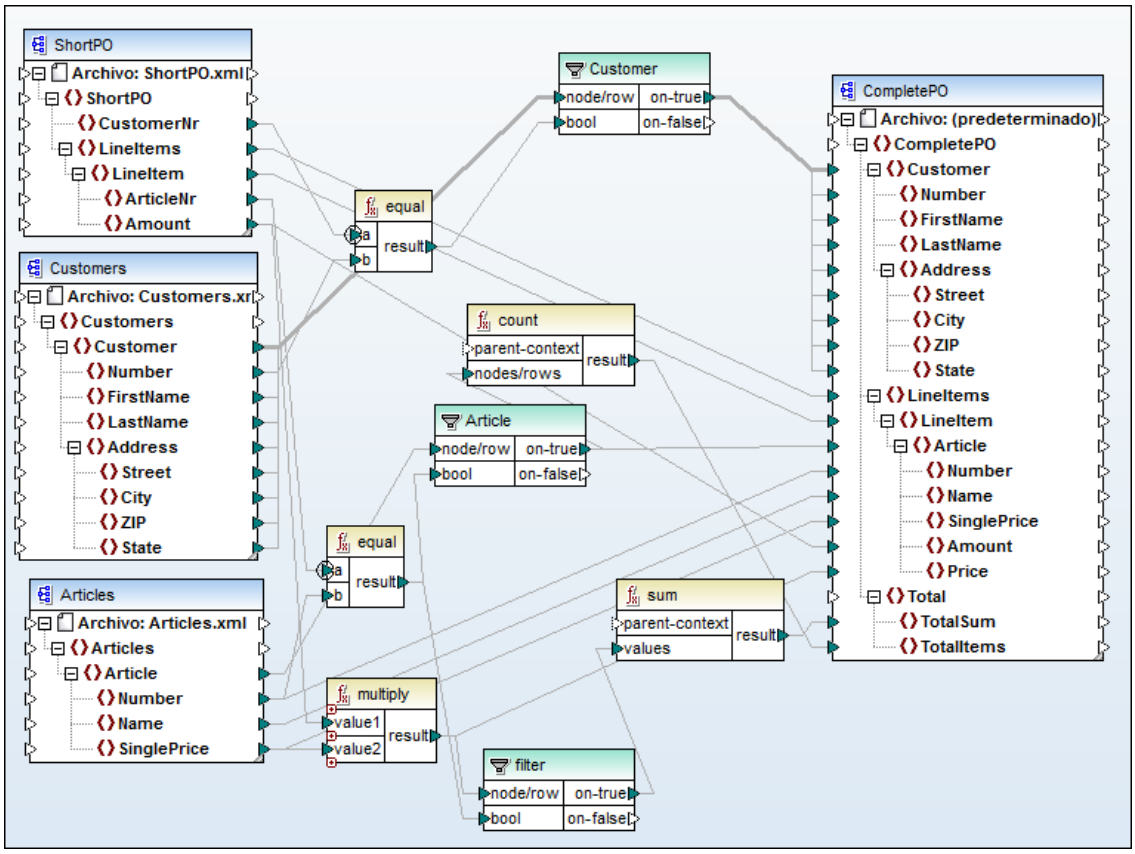

ShortPO
Archivo: ShortPO.xml
ShortPO
CustomerNr
LineItems
LineItem
ArticleNr
Amount
Customers
Archivo: Customers.xr
Customers
Customer
Number
FirstName
equal
a
b
result
Customer
node/row
bool
on-true
on-false
CompletePO
Archivo: (predeterminado)
CompletePO
Customer
Number
FirstName
LastName
Address
Street
City
ZIP
State
LineItems

10.8 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.

10.9 Línea de comandos: definir parámetros de entrada

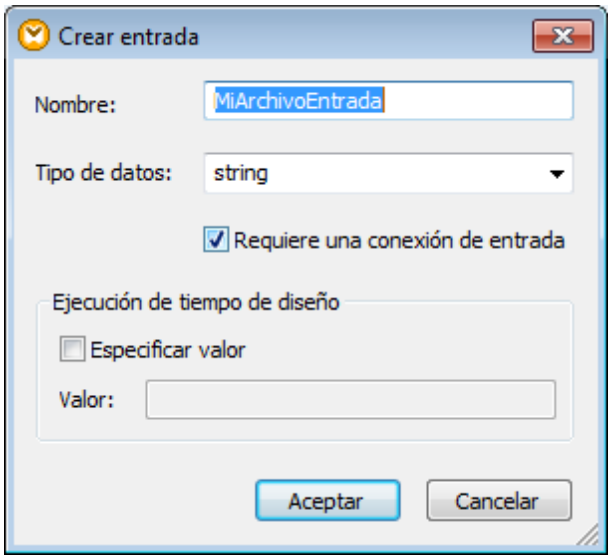
En MapForce puede crear componentes de entrada que funcionen como parámetros en la ejecución de la línea de comandos de una asignación. Estos componentes de entrada sirven para definir el contenido del parámetro cuando la asignación se ejecuta desde la línea de comandos y los valores que debe tomar cuando 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**.

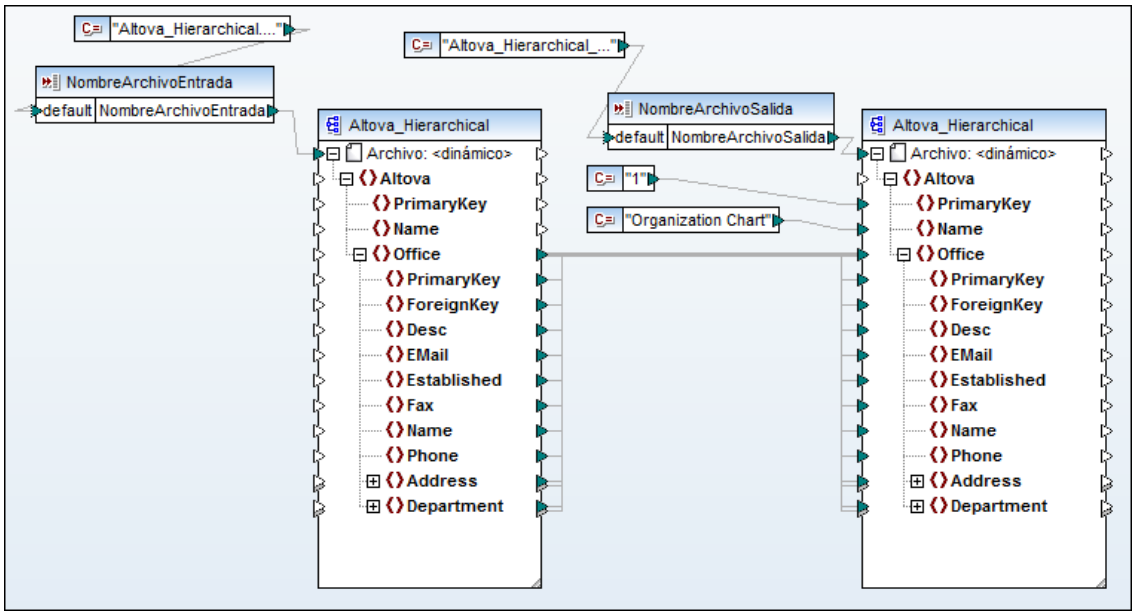
   Los demás campos de este cuadro de diálogo sirven para especificar los valores que se deben usar cuando se ejecuta la asignación en el panel **Resultados** (vista previa) y no se usan en la ejecución desde la línea de comandos. Para más información consulte el apartado Parámetros de entrada: configuración predeterminada y vista previa.

 © 2014 Altova GmbH

10.10 Parámetros de entrada: configuración predeterminada y vista previa

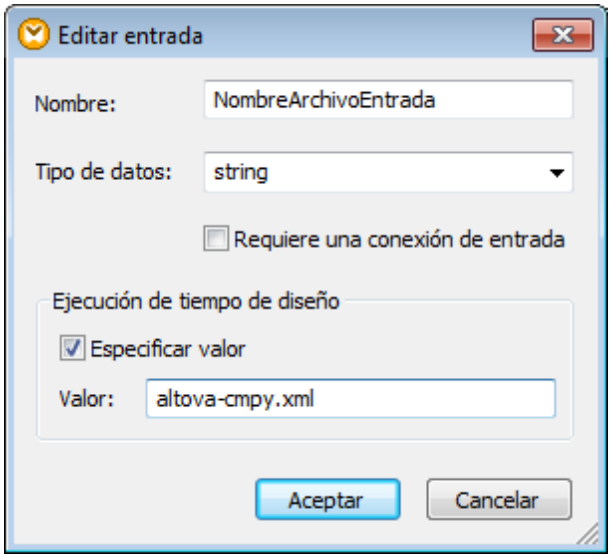
La asignación que aparece en la imagen siguiente (el archivo `FileNamesAsParameters.mfd` guardado en la carpeta `...\MapForceExamples`) usa dos componentes de entrada: uno suministra el nombre del archivo de origen y el otro suministra el del archivo de salida (a los elementos **`Archivo:<dinámico>`** 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**.

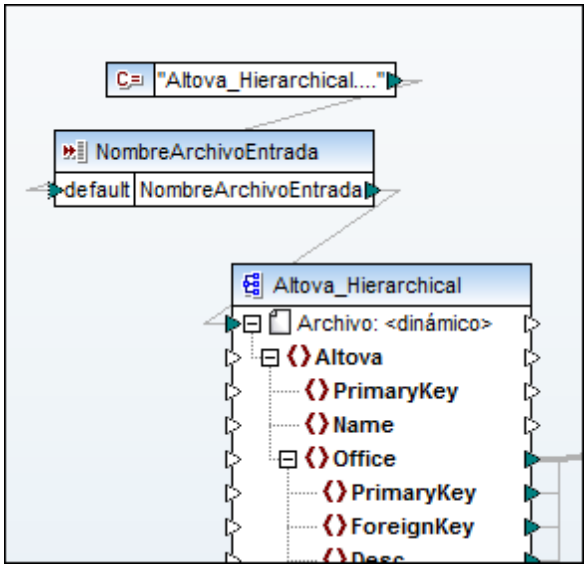
   El valor introducido en el campo *Valor:* del grupo *Ejecución de tiempo de diseño* solamente se aplica a la **vista previa** de resultados en el panel **Resultados**. Este valor no se usa para la generación de código ni para la ejecución de asignaciones desde la línea de comandos.

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`**).

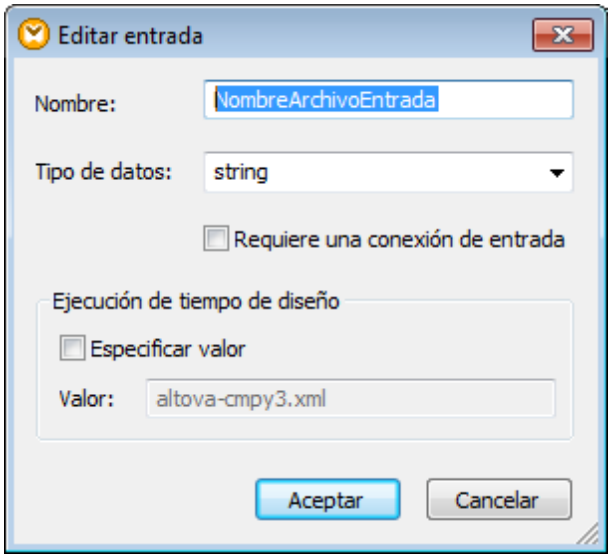
1. Ahora puede usar una **constante** para suministrar el valor predeterminado **`default`**, p. ej `Altova_Hierarchical.xml`.
2. Cree la constante y conéctela al elemento **`default`** del componente de entrada.

 © 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.

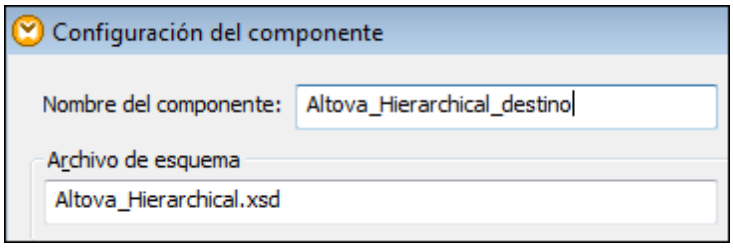
Usar valores/parámetros predeterminados `default` en la ejecución de la línea de comandos

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

10.11 Nombres de los componentes

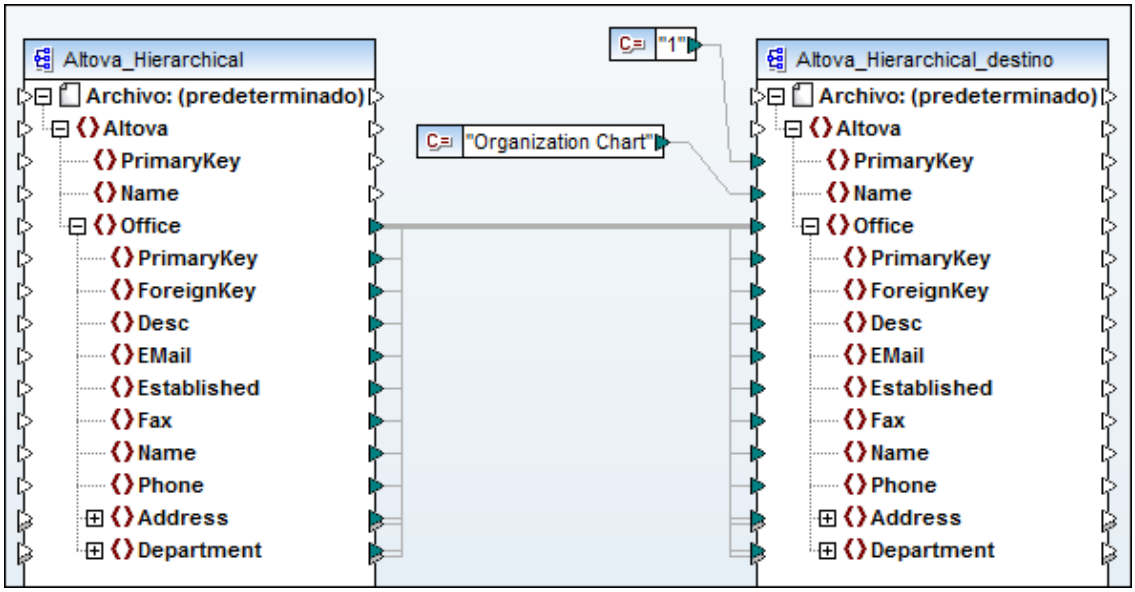
Todos los componentes de MapForce basados en archivos pueden tener un nombre de componente definido por el usuario. El nombre de componente aparece en el primer cuadro de texto del diálogo "Configuración del componente".



Debe tener en cuenta los siguientes aspectos de los nombres de componente:

- El nombre del componente se utiliza para obtener el nombre de los scripts XSLT generados.

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.

- El componente de origen se llama **Altova_Hierarchical**
- El componente de destino se llama **Altova_Hierarchical_destino**

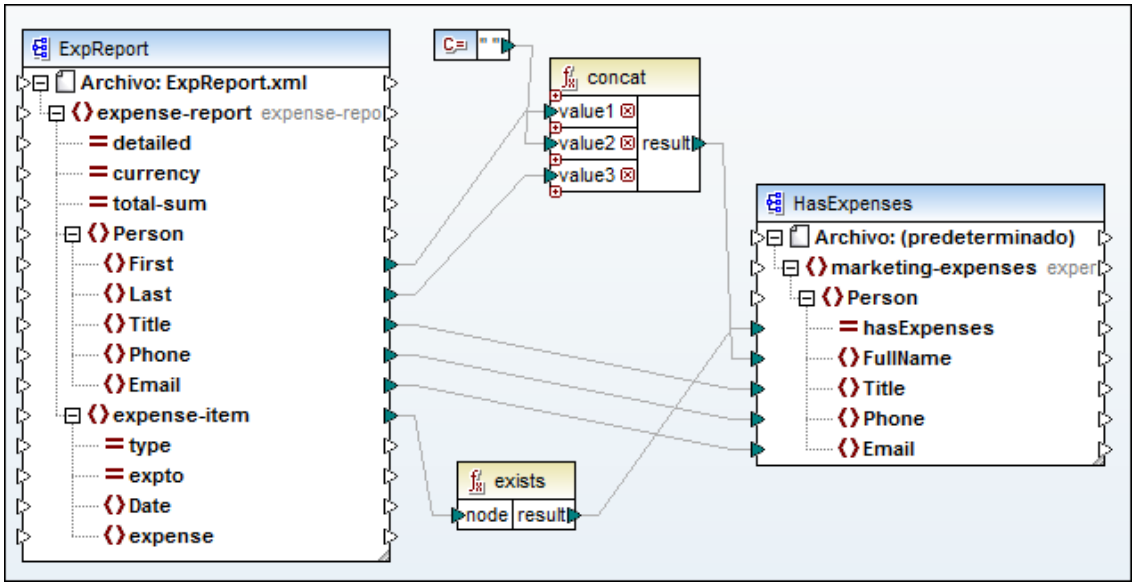
10.12 Comprobación, posición y agrupamiento de nodos

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.



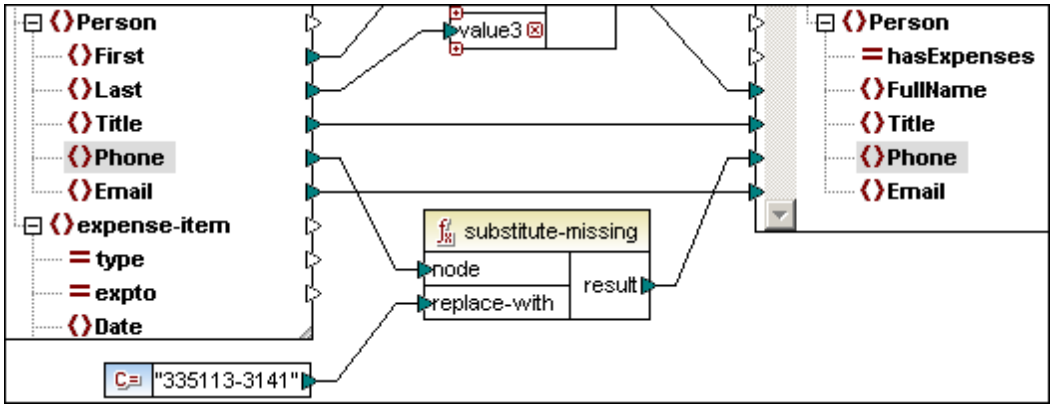
Función `not-exists`

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



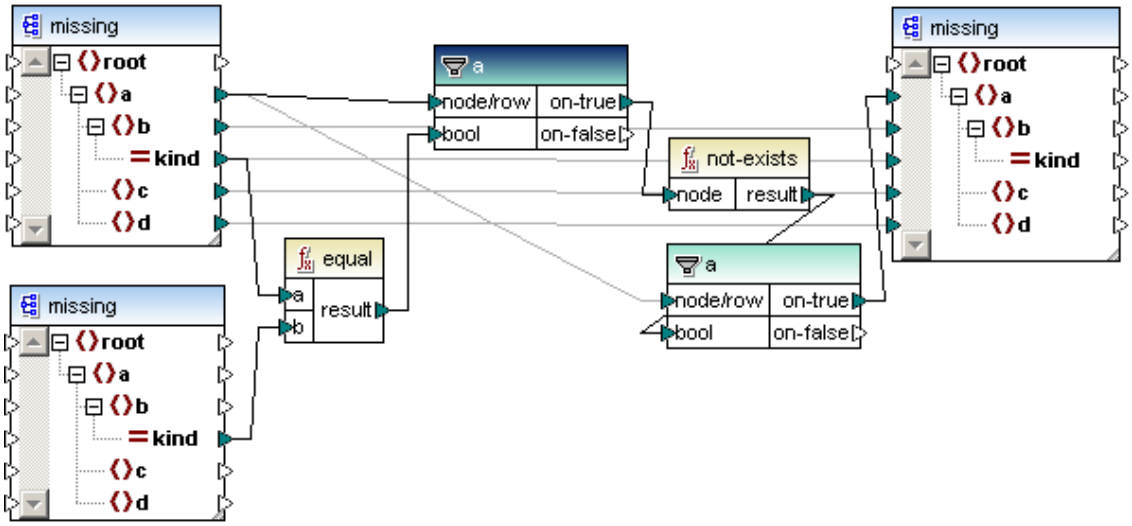
En el ejemplo de la imagen anterior, se comprueba si existe el nodo **Phone** en el archivo XML de instancia. Si el nodo no existe, se asigna el valor suministrado por la constante.

10.12.1 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.

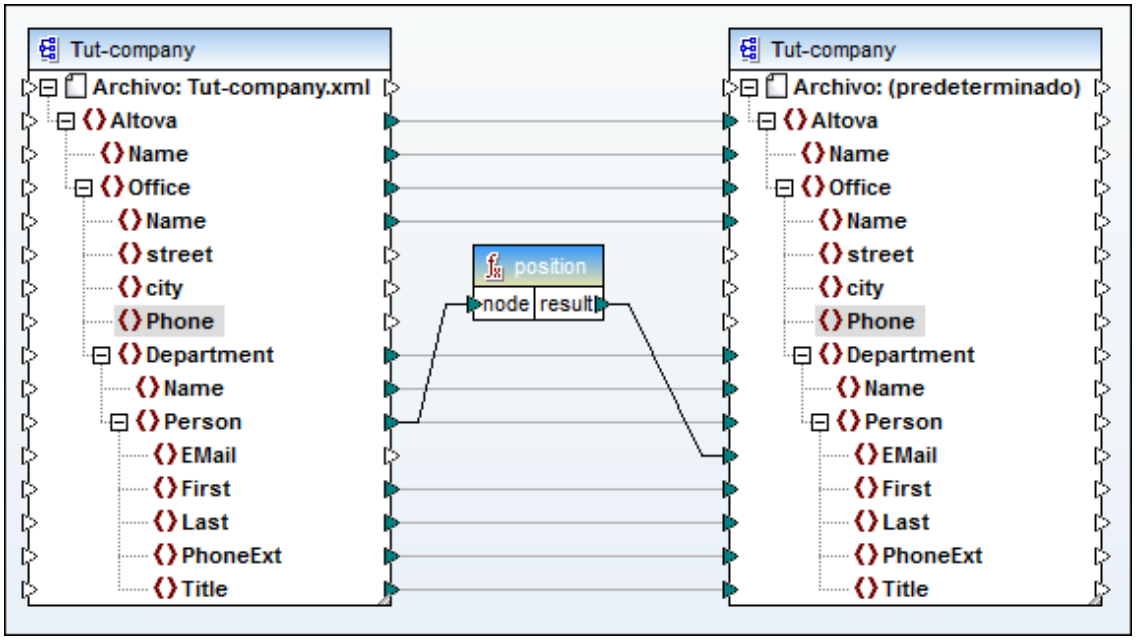
```
<root xsi:noNamespaceSchemaLocation="C:/DOCUME
    <a>
        <b kind="3">b3</b>
        <c>c3</c>
        <d>d3</d>
    </a>
</root>
```

10.12.2 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>
```

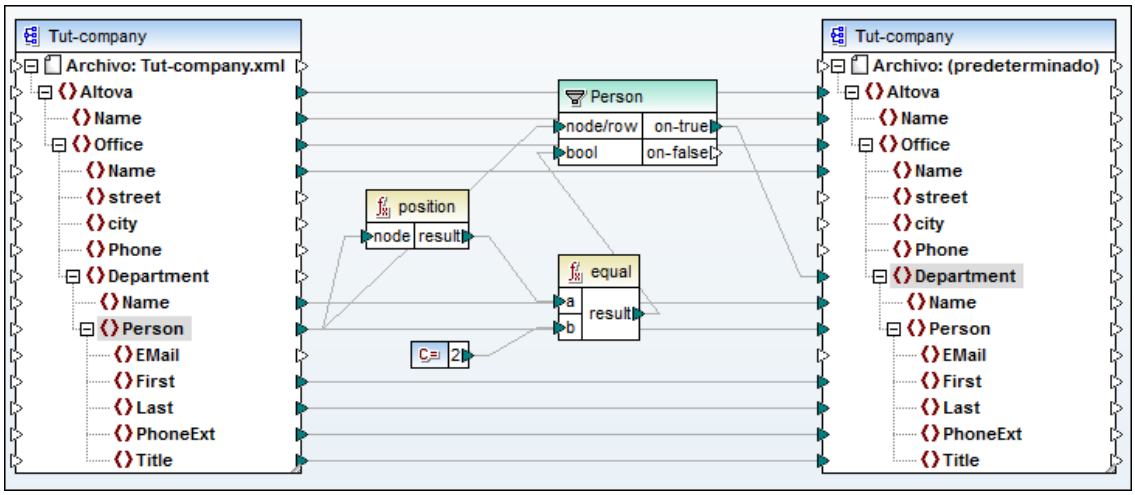
Filtrar determinados nodos por medio de la función position

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.

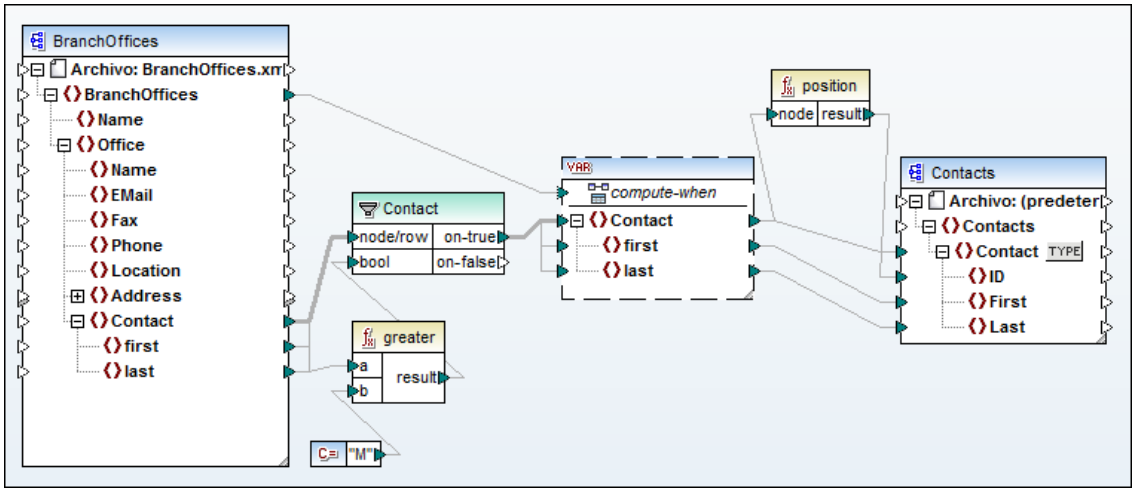
```
<Office>
  <Name>Microtech, Inc.</Name>
  <Department>
    <Name>Admin</Name>
    <Person>
      <EMail>b.bander@microtech.com</EMail>
      <First>Bert</First>
      <Last>Bander</Last>
      <Title>Accounts Receivable</Title>
    </Person>
  </Department>
  <Department>
    <Name>Sales and Marketing</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.



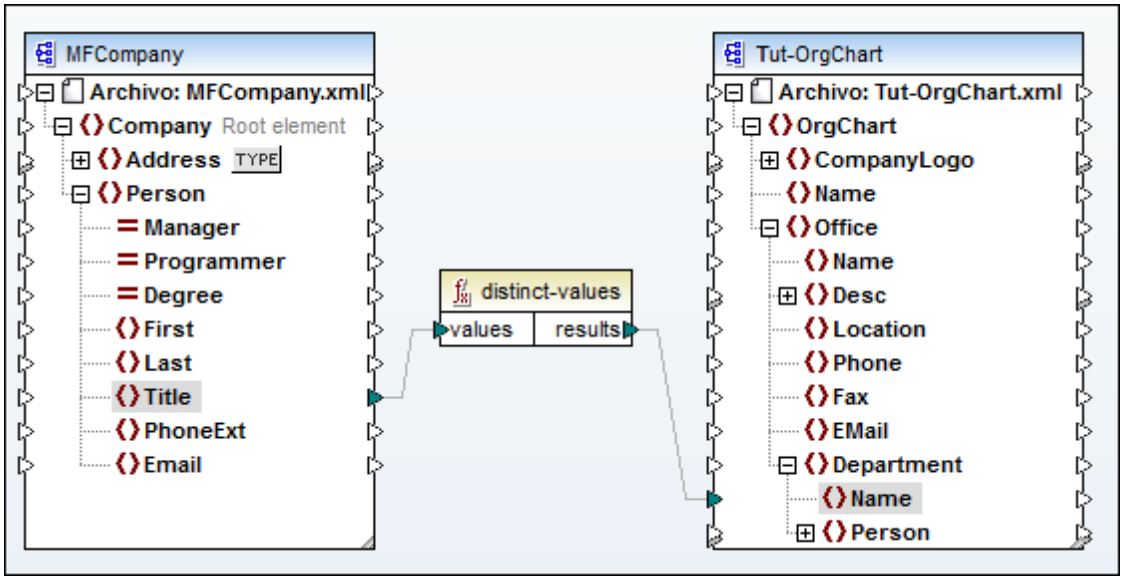
 © 2014 Altova GmbH

10.12.3 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.

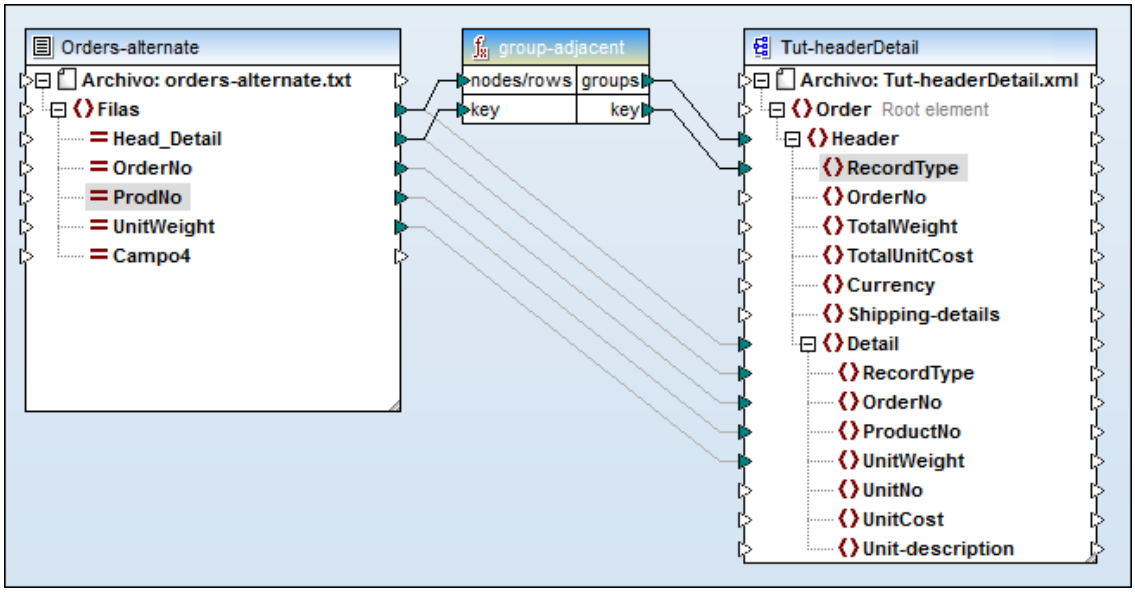


Observe que la secuencia que los elementos **`Title`** tenían en el componente de origen se conserva en el componente de destino.

```
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

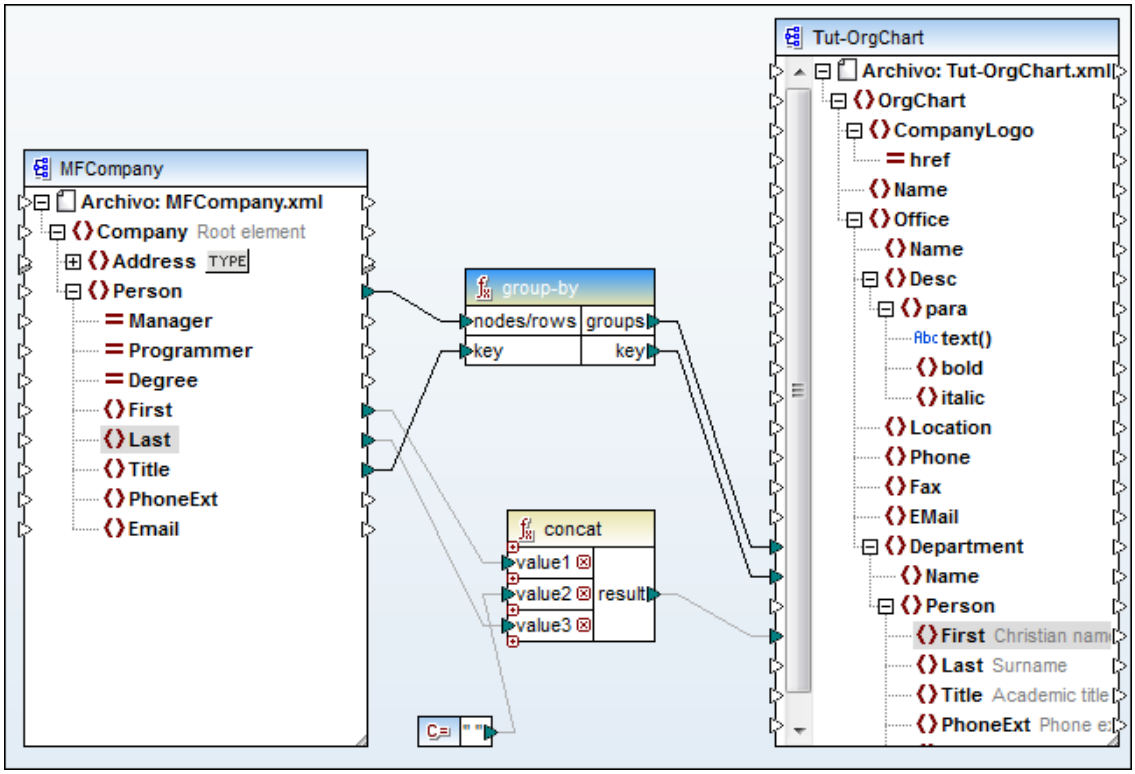
```
3   <Header>
4     <RecordType>H</RecordType>
5     <Detail>
6       <RecordType>H</RecordType>
7       <OrderNo>111</OrderNo>
8       <ProductNo>G-889-11</ProductNo>
9       <UnitWeight>9</UnitWeight>
10    </Detail>
11  </Header>
12  <Header>
13    <RecordType>D</RecordType>
14    <Detail>
15      <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

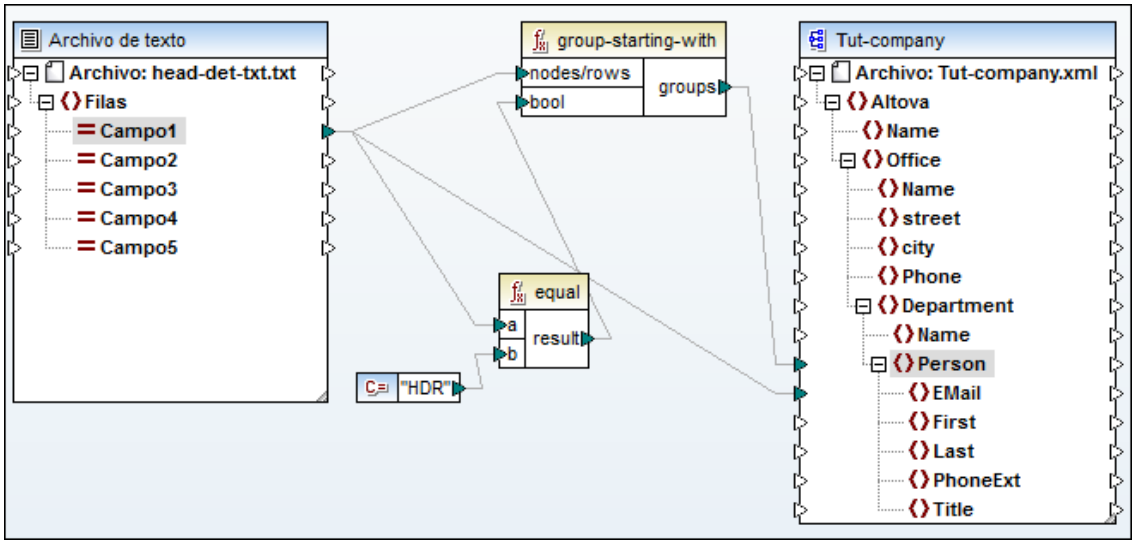
```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <OrgChart xsi:schemaLocation="http://www.xmlspy.com/schemas/orgchart C:/DOCU
3    <Office>
4      <Department>
5        <Name>Office Manager</Name>
6        <Person>
7          <First>Vernon Callaby</First>
8          <First>Steve Meier</First>
9        </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.

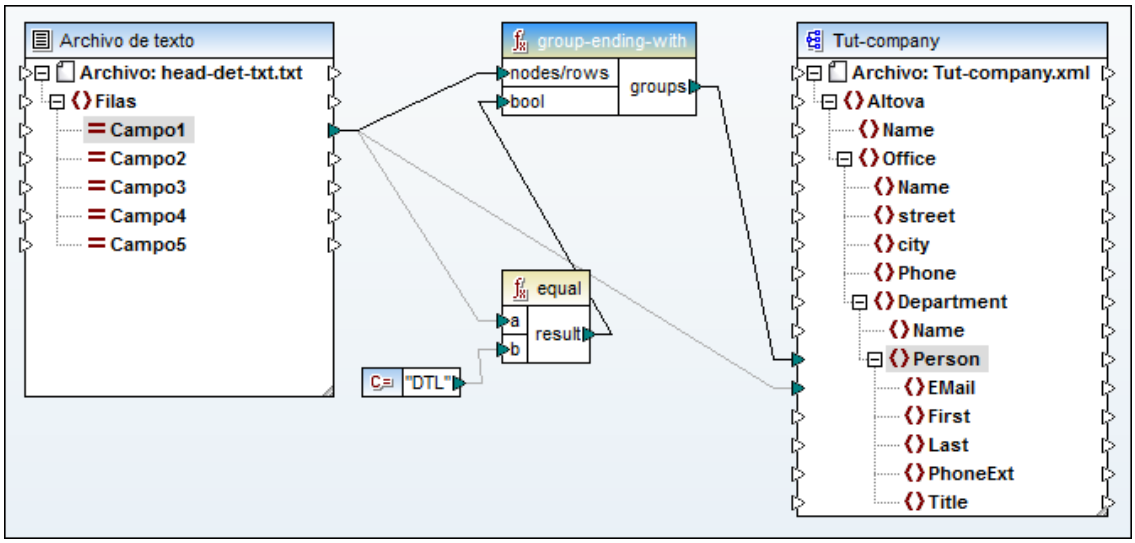
```
3   <Office>
4     <Department>
5       <Person>
6         <EMail>HDR</EMail>
7         <EMail>DTL</EMail>
8         <EMail>TXT</EMail>
9         <EMail>DTL</EMail>
10        <EMail>DTL</EMail>
11      </Person>
12      <Person>
13        <EMail>HDR</EMail>
14        <EMail>DTL</EMail>
15      </Person>
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.

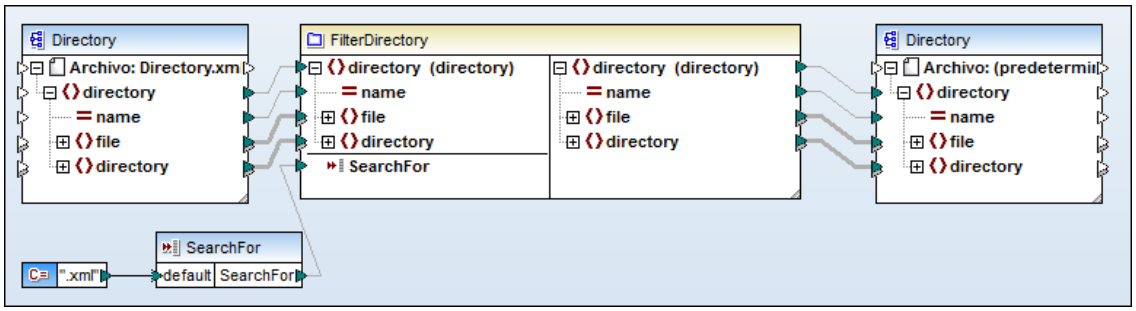
Función `set-empty`

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.

10.13 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.

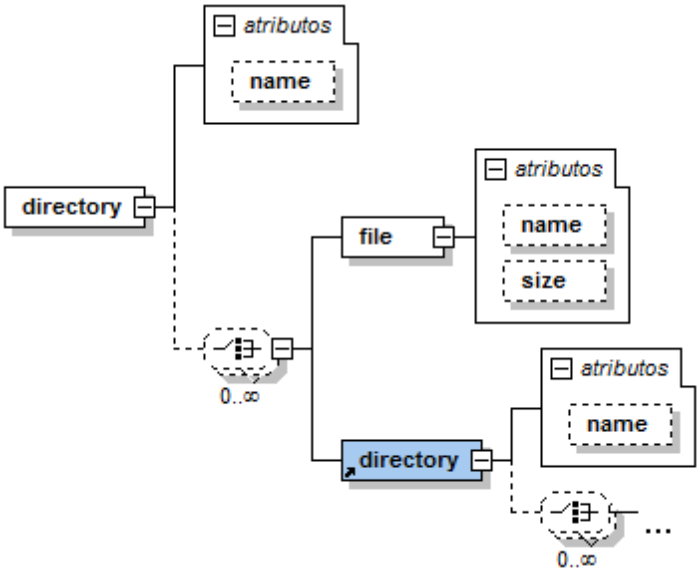


El **archivo de origen** que contiene los datos sobre archivos y directorios para esta asignación es el archivo `Directory.xml`. Este archivo XML suministra los datos sobre archivos y directorios en forma jerárquica (*imagen siguiente*).

```
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.

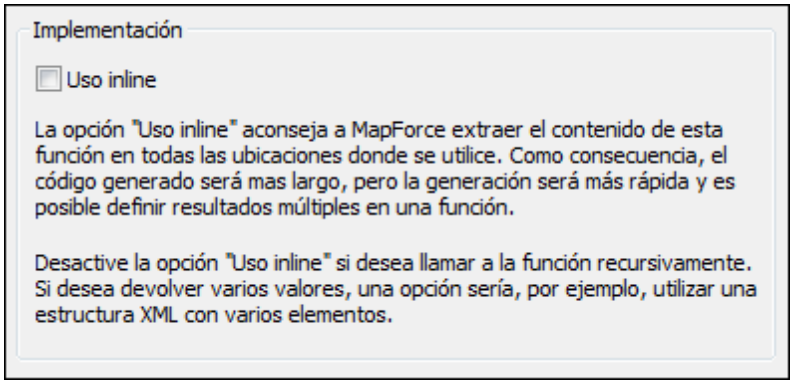
 © 2014 Altova GmbH


atributos
name
directory
file
atributos
name
size
0..∞
atributos
name
directory
0..∞
...

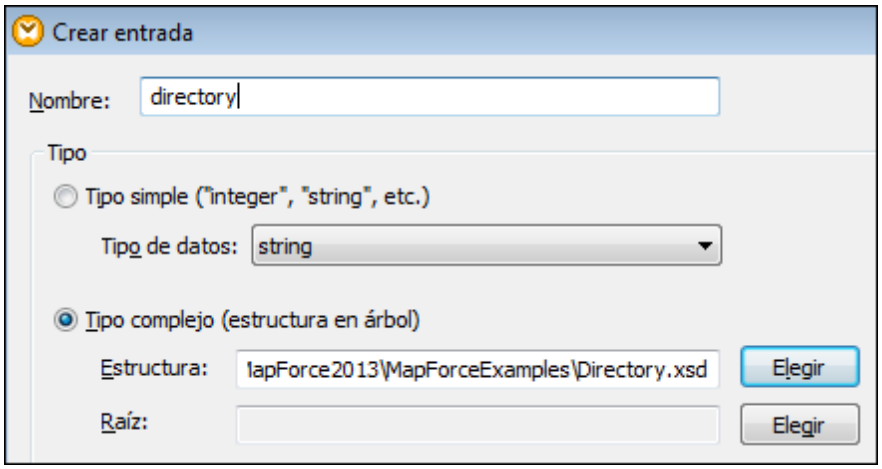
10.13.1 Definir una función recursiva definida por el usuario

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**.

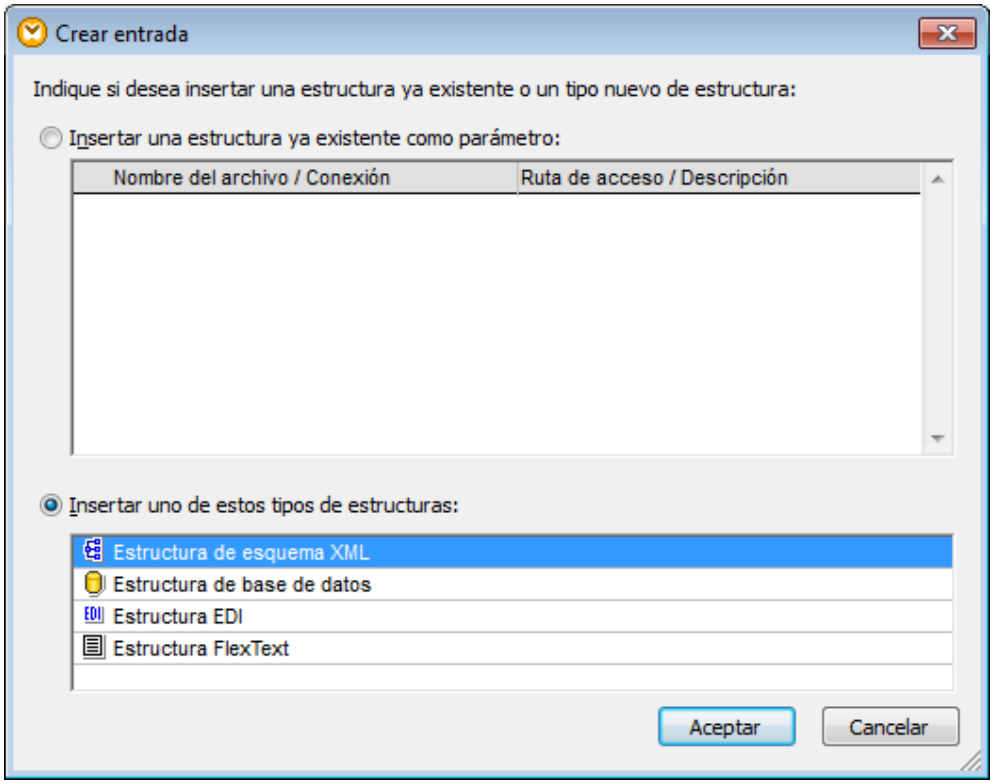


   Ahora está en el panel **FilterDirectory**, donde puede crear la función definida por el usuario.
3. Seleccione la opción de menú **Función | Insertar componente de entrada**.
4. En el cuadro de diálogo que aparece escriba el nombre del componente de entrada (p. ej. **`directory`**) y haga clic en el botón de opción *Tipo complejo (estructura en árbol)*.

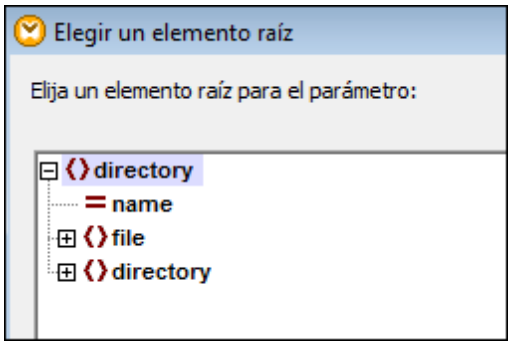


5. Haga clic en el botón **Elegir**. En el nuevo cuadro de diálogo haga clic en la entrada `Estructura de esquema XML` (en el panel inferior) y confirme la selección con **Aceptar**.

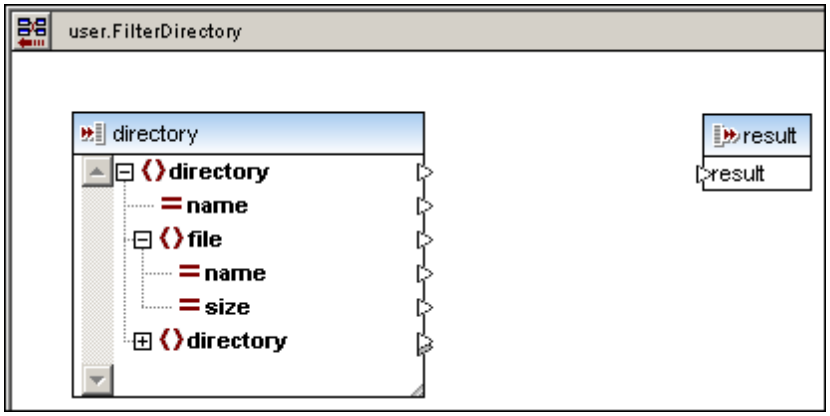
© 2014 Altova GmbH



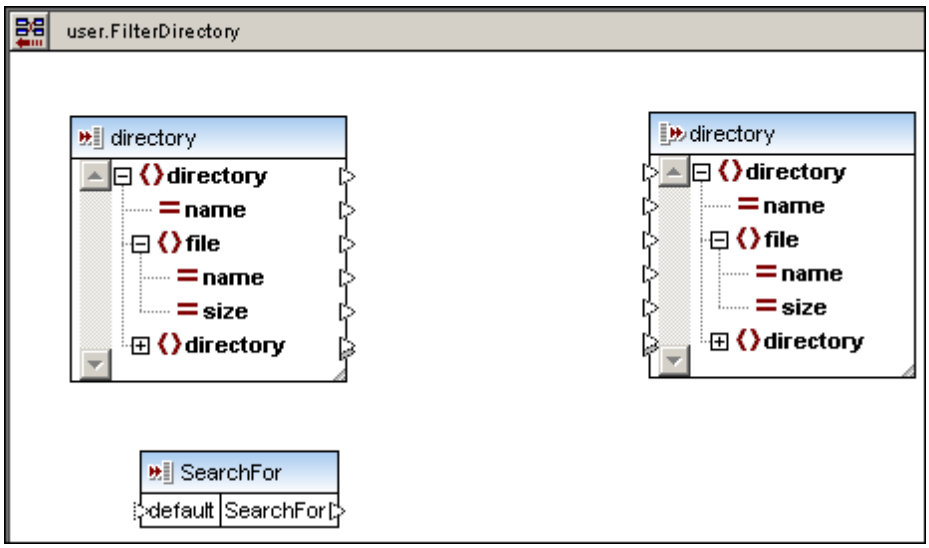
6. Seleccione el archivo `Directory.xsd` de la carpeta `...\MapForceExamples` y haga clic en **Abrir**.
7. Aparece el cuadro de diálogo "Elegir un elemento raíz". Seleccione la entrada `directory` (*imagen siguiente*) y haga clic en **Aceptar**.



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.

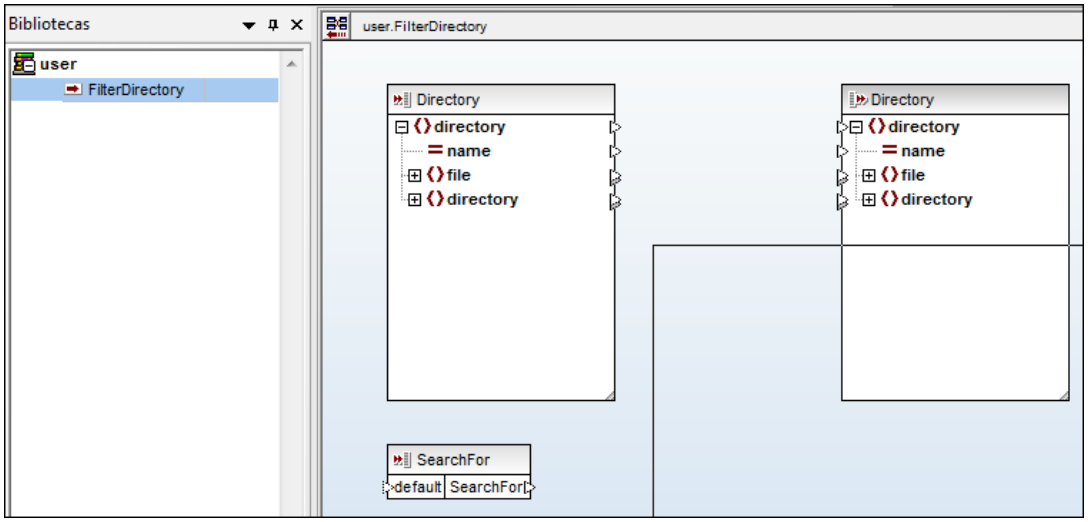


Insertar la función recursiva definida por el usuario

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.

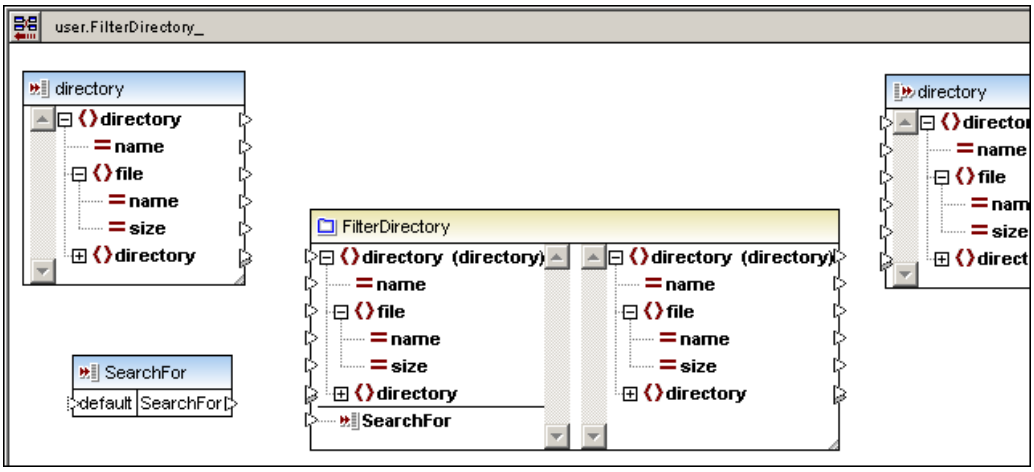
1. Busque la función **`FilterDirectory`** en la sección **`user`** de la ventana Bibliotecas.
2. Haga clic en **`FilterDirectory`** y arrastre la función hasta el panel de diseño **FilterDirectory** donde trabajó hasta ahora.

 © 2014 Altova GmbH

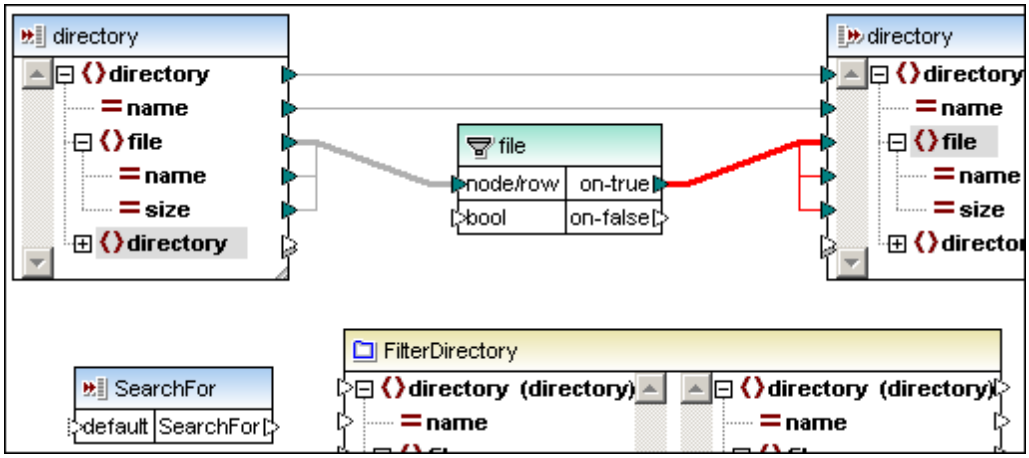


Con Java seleccionado

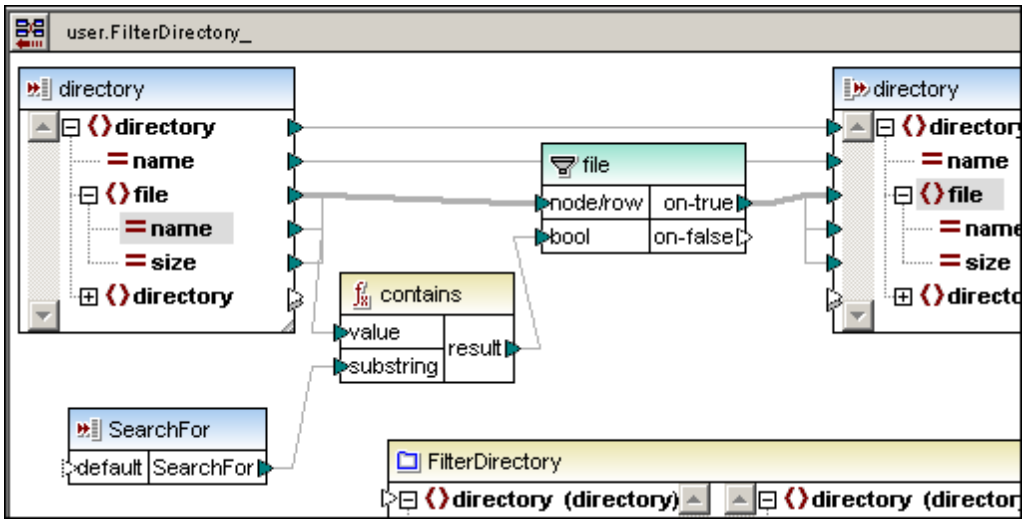
La función definida por el usuario aparece como componente recursivo en la ventana de la función definida por el usuario.



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**.



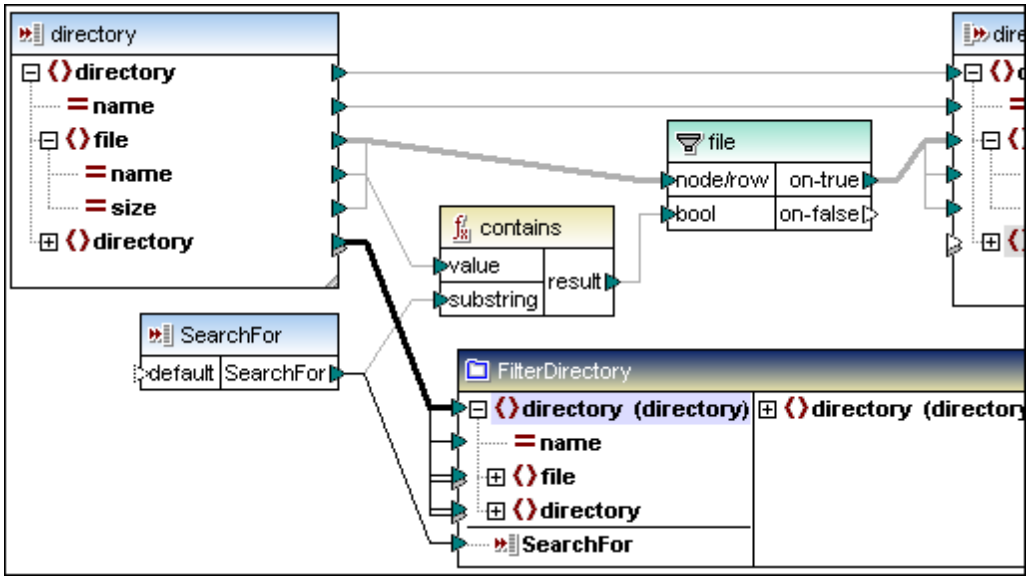
8. Para terminar conecte el elemento **`SearchFor`** del componente de entrada con el elemento **`SearchFor`** de la función definida por el usuario.

Definir la recursión

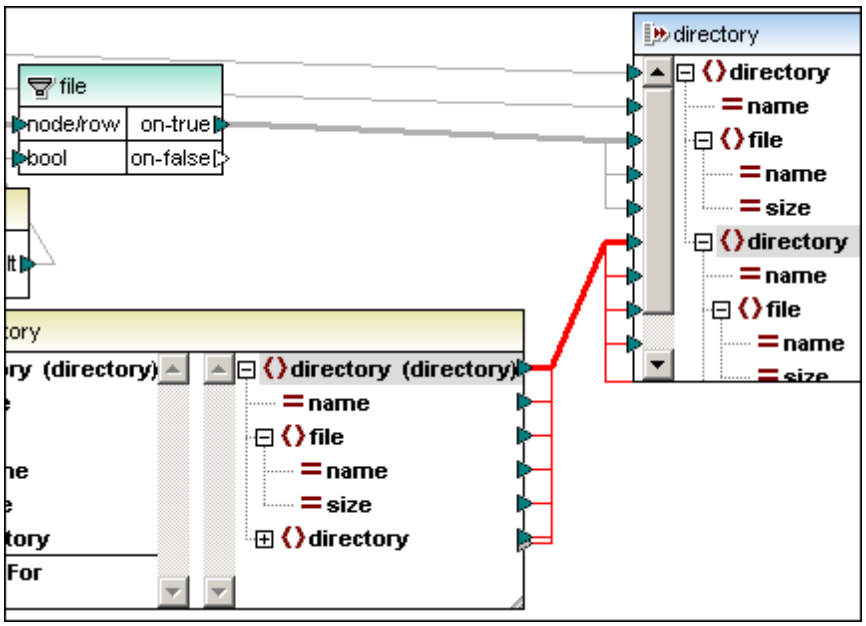
Ya hemos definido un nivel de recursividad del directorio. Ahora tenemos que definir cómo se procesa un subdirectorio.

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.

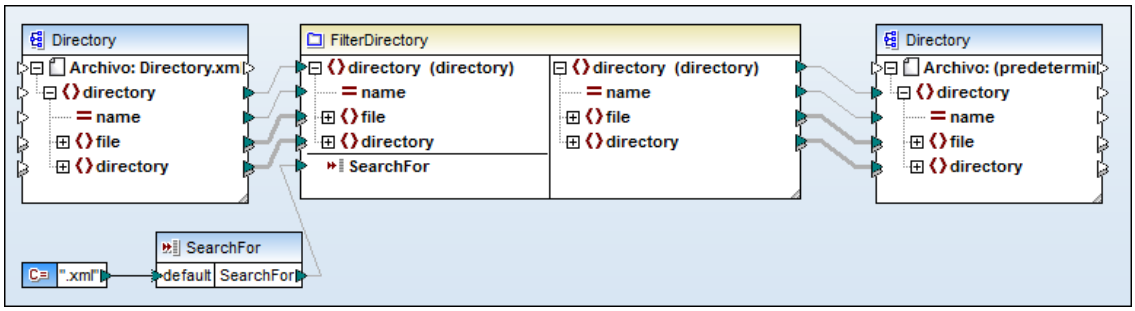
Haga clic en el icono situado en la parte superior del panel para volver al panel principal de asignación y seguir definiendo la asignación.

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 2015

Recursos globales

11 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 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

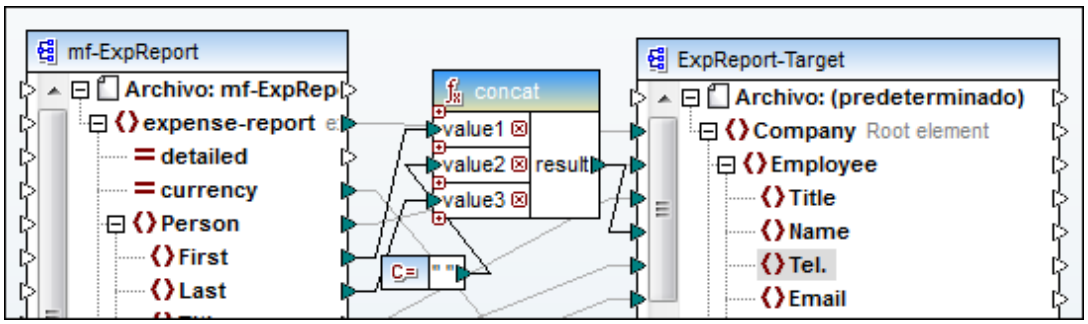
11.1 Recursos globales: archivos

En MapForce puede definir como recurso global cualquier archivo de entrada/salida de un componente, p. ej. archivos XML, esquemas XML, 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**.

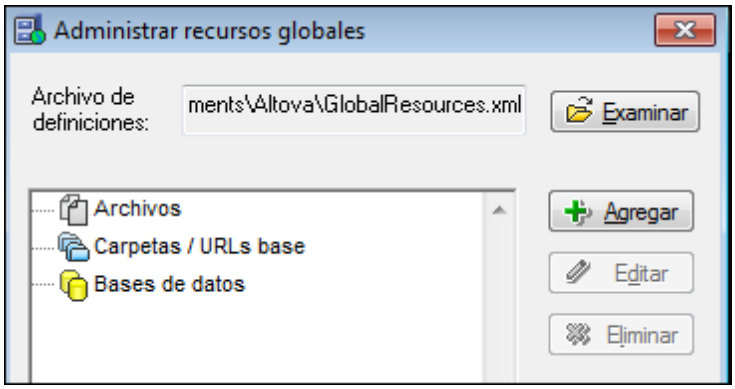
  Para este ejemplo utilizaremos el archivo `Tut-ExpReport.mfd` disponible en la carpeta `...\MapForceExamples\Tutorial`.



11.1.1 Definir y agregar recursos globales

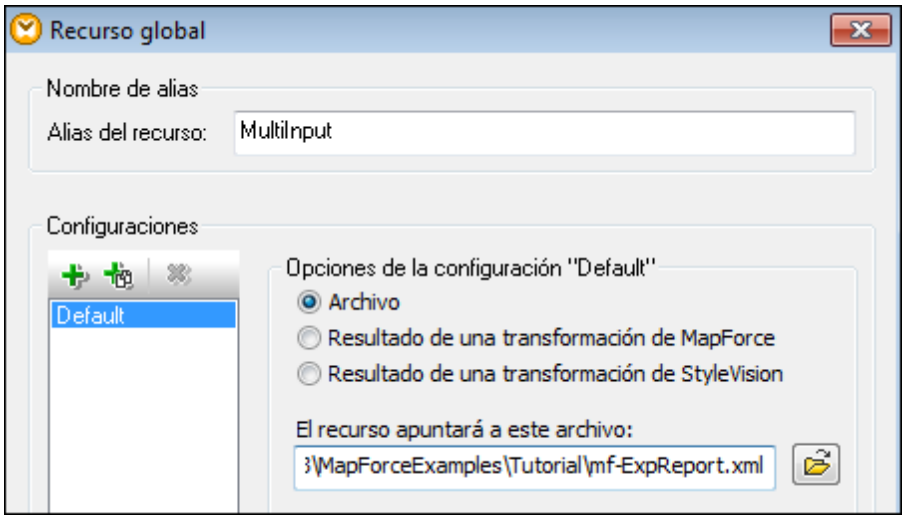
Definir/añadir recursos globales de archivo

1. Primero haga clic en el icono **Recursos globales** para abrir este cuadro de diálogo.



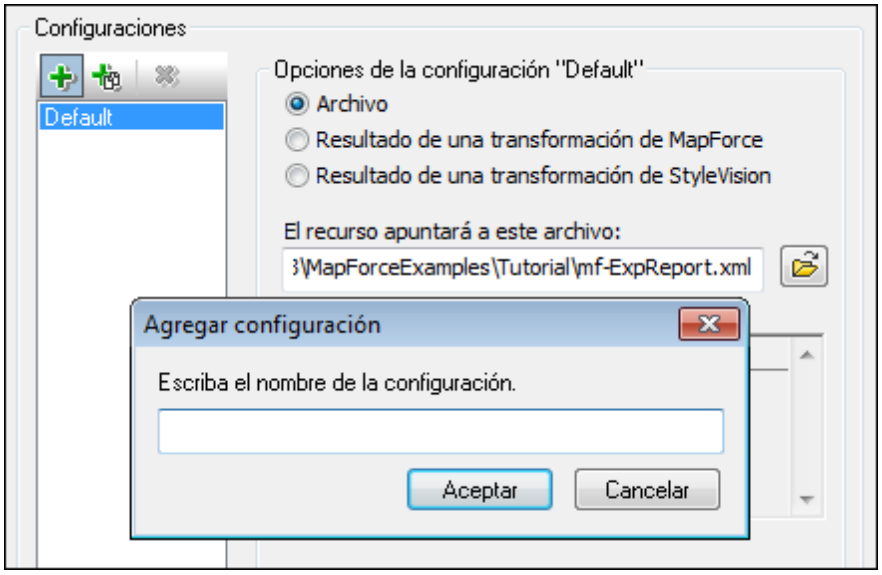
Así se representa un archivo de recursos globales vacío.

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`.

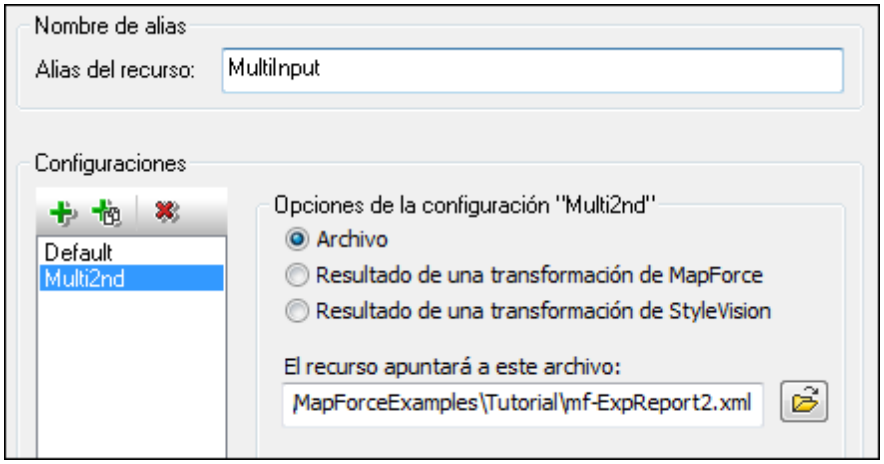


5. Haga clic en el icono **Agregar** del grupo de opciones *Configuraciones* para añadir una configuración nueva al *Alias* actual. Recuerde que con el icono **Copiar configuración** puede copiar la configuración seleccionada y guardarla con otro nombre.

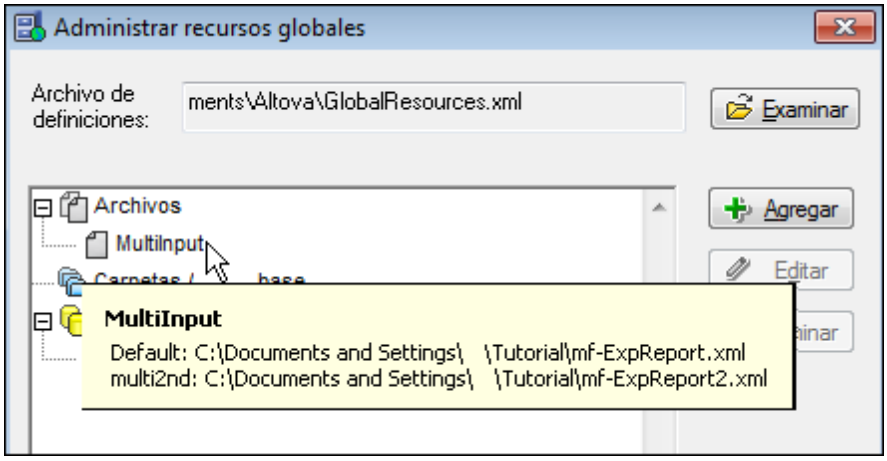
 © 2014 Altova GmbH



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.

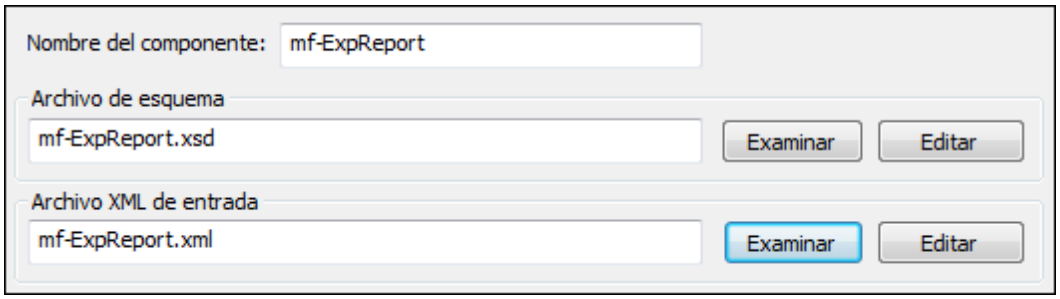
 © 2014 Altova GmbH

11.1.2 Asignar un recurso global

Asignar recursos globales a un componente

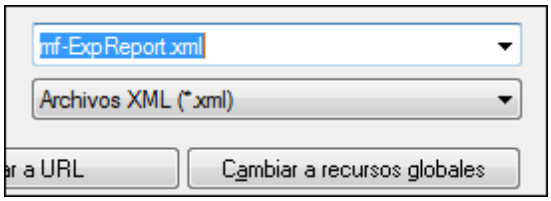
Ahora tenemos que asignar el recurso global al componente que debe utilizarlo, es decir, el archivo `mf-ExpReport.xml` que se utiliza como archivo de origen de la asignación.

1. Haga doble clic en el componente **mf-ExpReport** y después haga clic en el botón **Examinar** del campo *Archivo XML de entrada*.

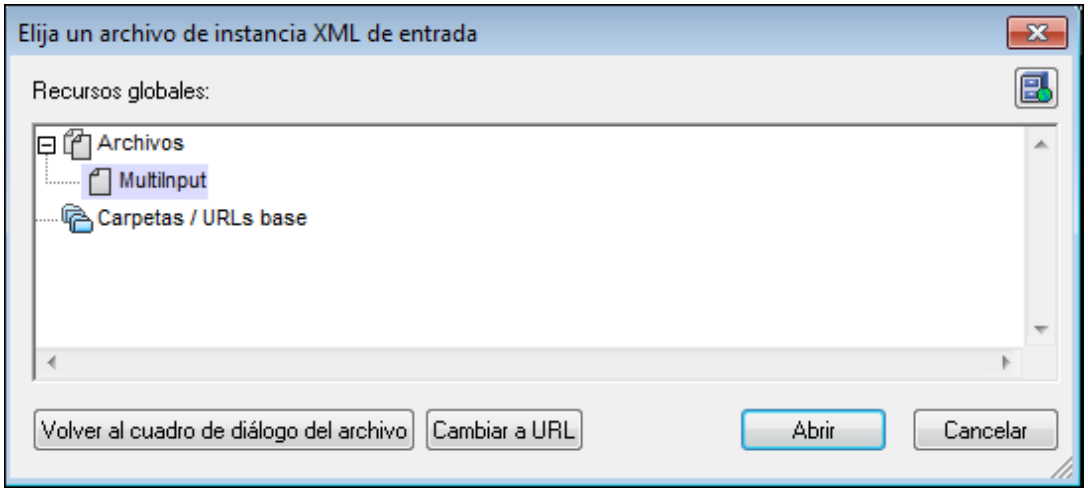


Se abre el cuadro de diálogo "Elija un archivo XML de instancia de entrada".

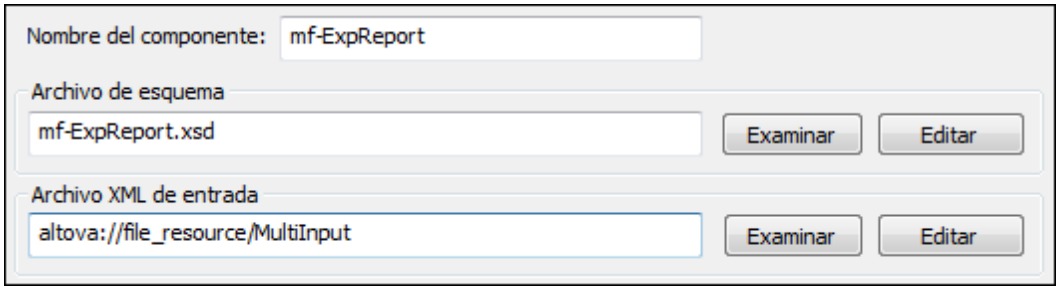
2. Haga clic en el botón **Cambiar a recursos globales**, situado en la parte inferior del cuadro de diálogo.



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.

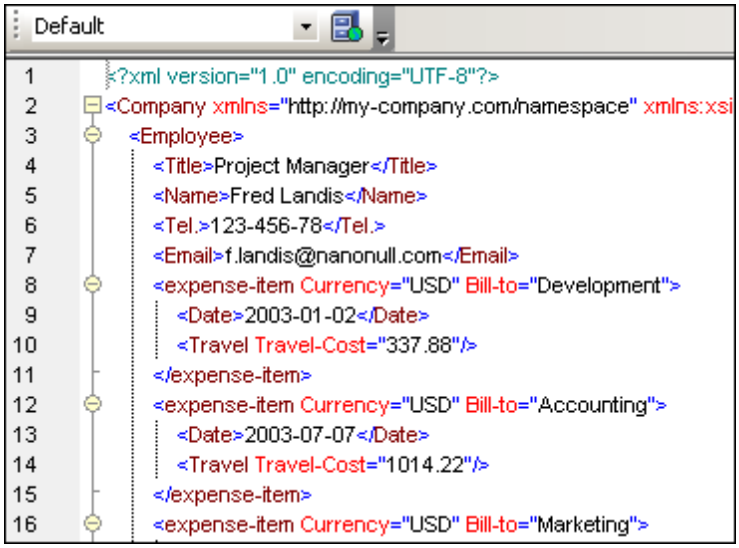
© 2014 Altova GmbH

11.1.3 Usar y activar un recurso global

Usar/activar un recurso global

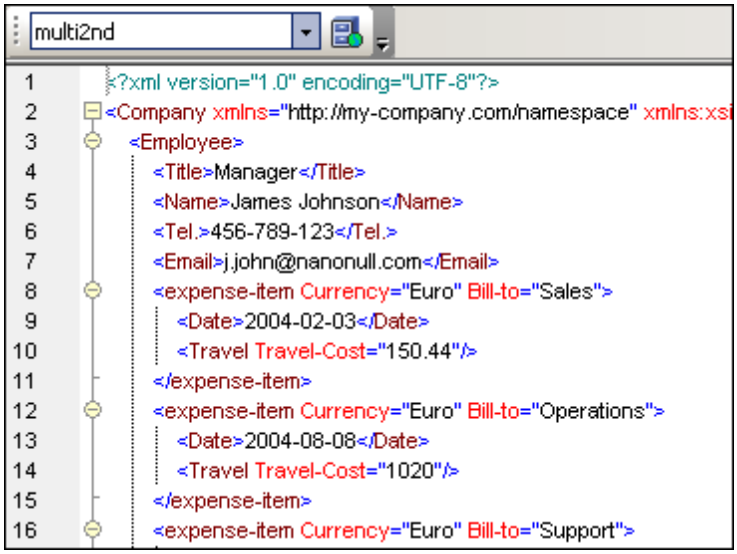
Llegados a este punto, la configuración activa es la configuración **`Default`**, que definimos en el primer paso para el alias `MultiInput`. Esto se puede ver en la barra de herramientas Recursos globales, que siempre indica la configuración activa.

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

11.2 Recursos globales: carpetas

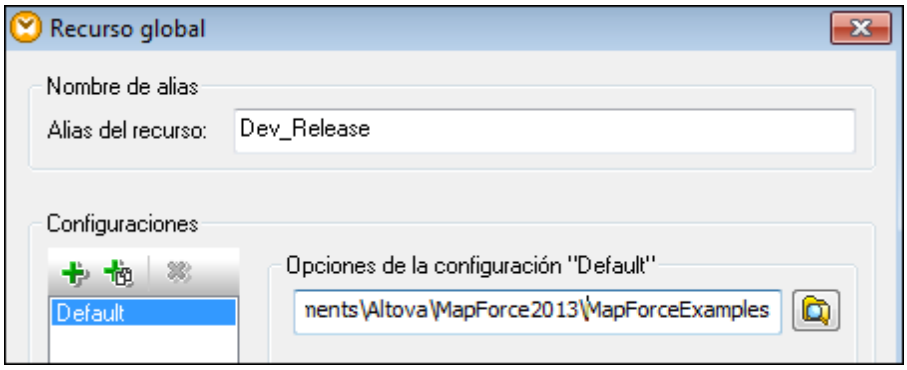
Las carpetas también se pueden definir como recurso global. Esto significa que los componentes de entrada pueden contener archivos que hacen referencia a carpetas diferentes, unas para la fase de desarrollo y otras para la fase de release, por ejemplo.

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.

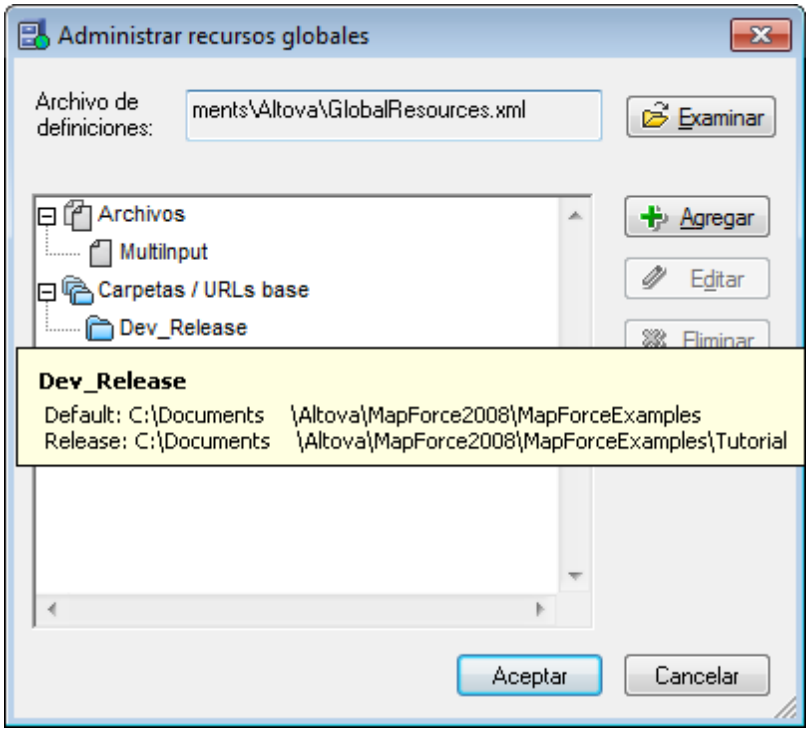
Para este ejemplo utlizamos el archivo de asignación `global-folder.mfd` guardado en la carpeta `...\MapForceExamples\Tutorial`.

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`.



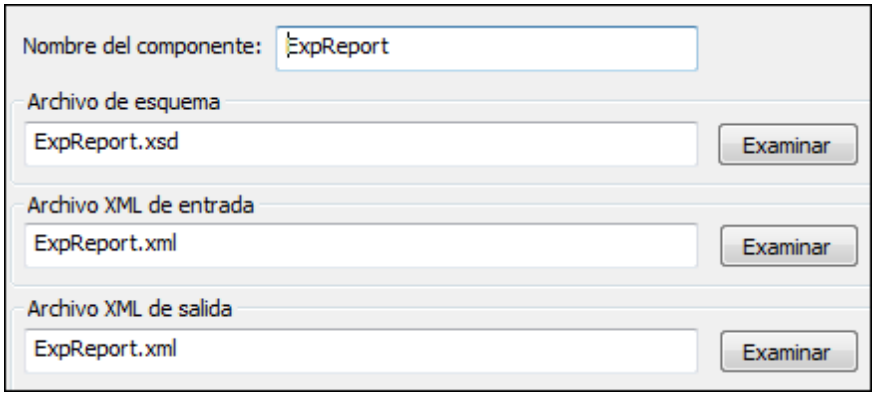
5. Haga clic en el icono **Agregar** del grupo de opciones *Configuraciones* para añadir una configuración nueva al *Alias* actual. Recuerde que con el icono **Copiar configuración** puede copiar la configuración seleccionada y guardarla con otro nombre.
6. Escriba el nombre de la nueva configuración, 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**.

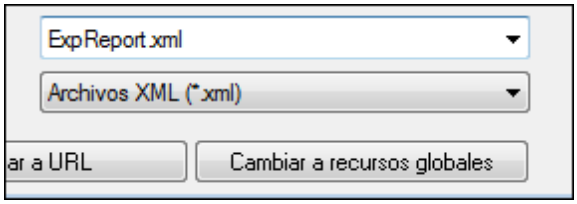
Asignar recursos globales de carpeta:

1. Haga doble clic en el componente **ExpReport** y después pulse el botón **Examinar** del campo *Archivo XML de entrada*.



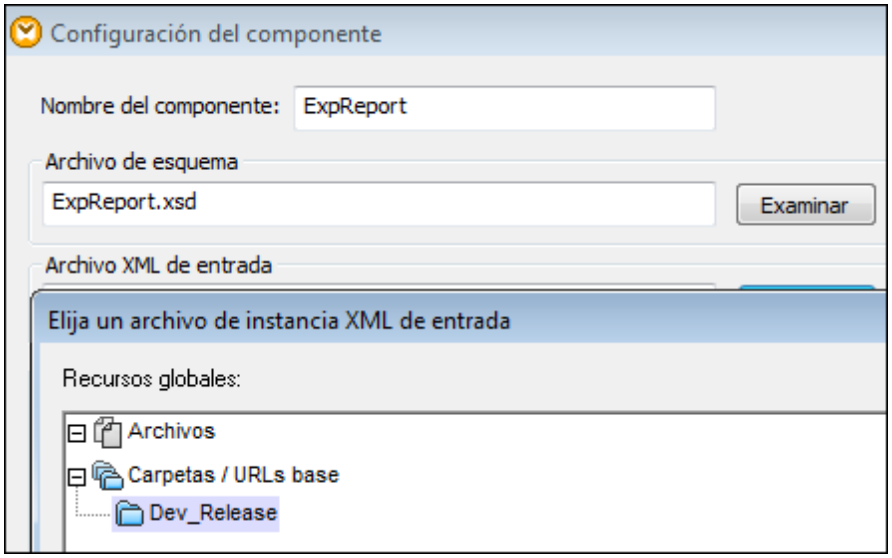
Aparece el cuadro de diálogo "Elija un archivo XML de instancia de entrada".

2. Haga clic en el botón **Cambiar a recursos globales**, situado en la parte inferior del cuadro de diálogo.



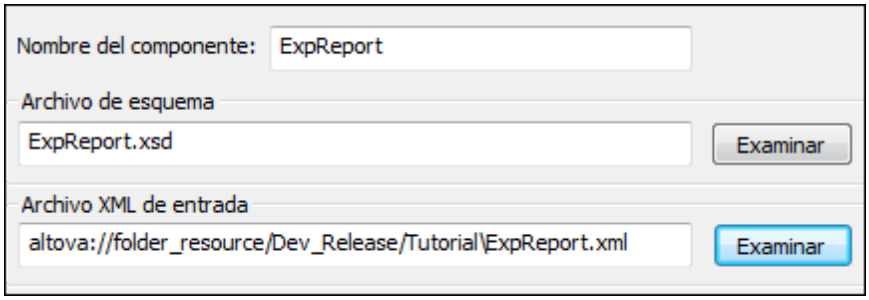
3. Seleccione el recurso que desea asignar como entrada para el componente (en este

caso `Dev_Release`) y haga clic en **Abrir**.



Aparece el cuadro de diálogo "Abrir".

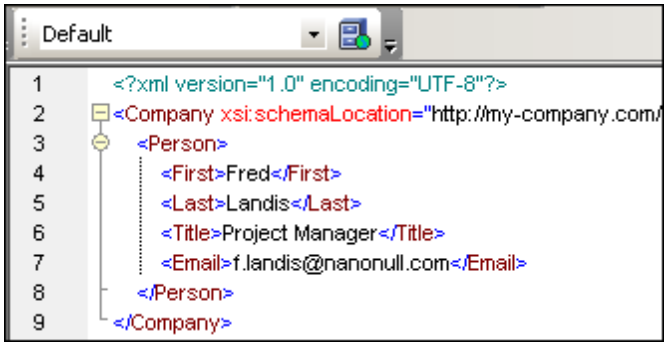
4. Seleccione el nombre del **archivo** que debe servir de **recurso** de archivo de desarrollo y para el release en ambas carpetas, p. ej. `ExpReport.xml`.
   Para terminar de asignar el recurso de carpeta haga clic en **Aceptar**.



No olvide que este archivo está disponible en ambas carpetas pero tiene diferente contenido.

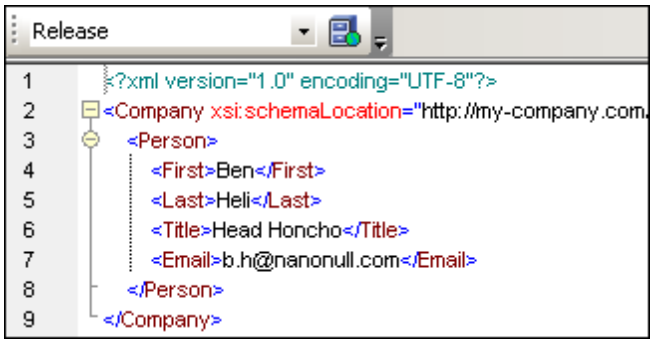
Cambiar de recurso de carpeta en tiempo de ejecución:

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`.



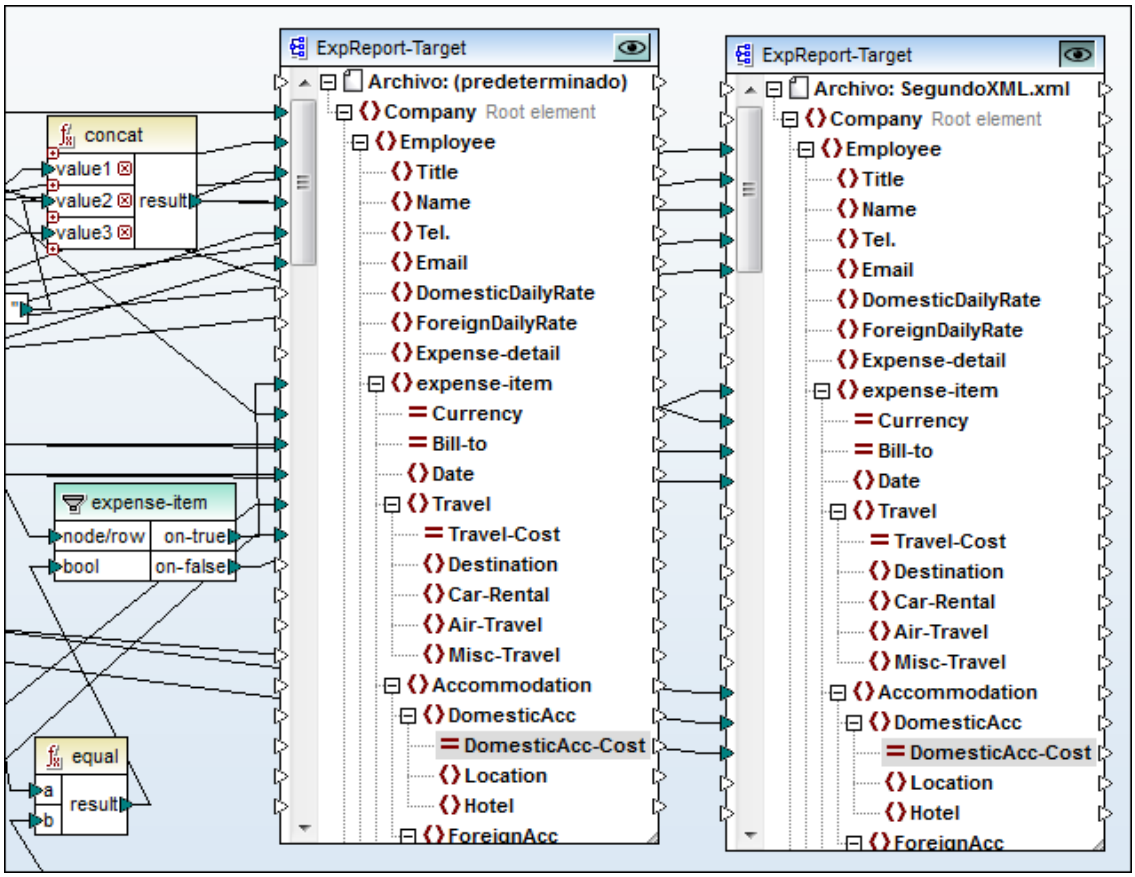
Ahora podemos ver el resultado obtenido usando la carpeta de release `.../MapforceExamples/Tutorial`.

 © 2014 Altova GmbH

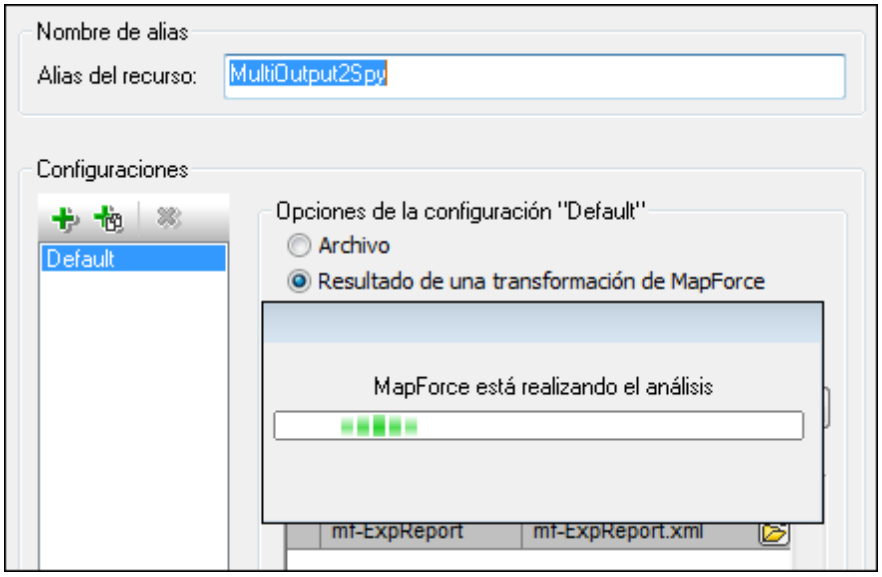
11.3 Recursos globales: flujo de trabajo en la aplicación

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`.

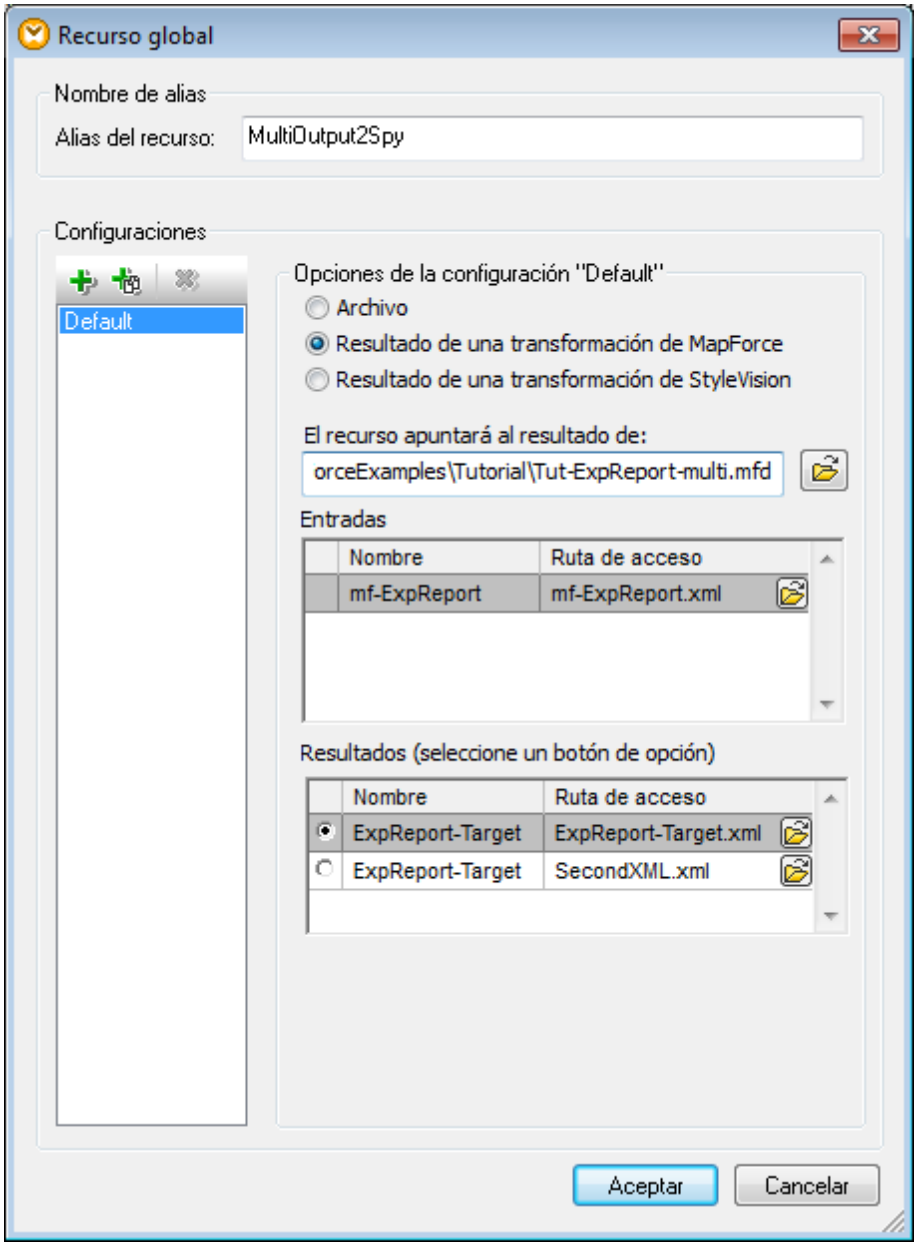


1. Haga clic en el icono **Recursos globales**.
2. Aparece el cuadro de diálogo de recursos globales. Haga clic en **Agregar** y seleccione la opción **Archivo**.
3. Escriba el nombre de alias del recurso, p. ej. `MultiOutput2Spy`
4. Seleccione el botón de opción *Resultado de una transformación de MapForce* y después haga clic en el icono **Examinar**.
5. Seleccione el archivo de asignación `Tut-ExpReport-multi.mfd`.

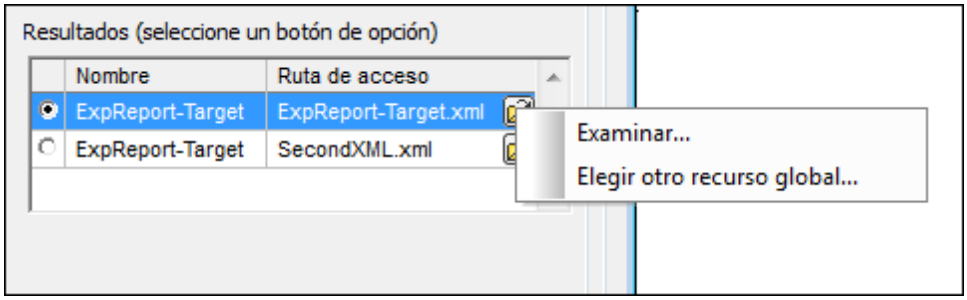


MapForce analiza la asignación y muestra los archivos de entrada y salida en recuadros diferentes.

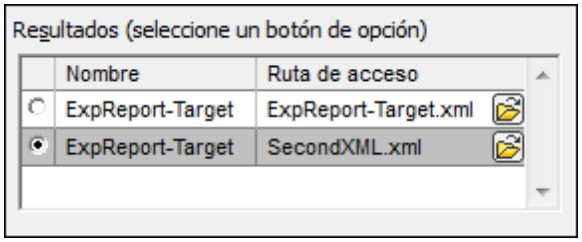
© 2014 Altova GmbH



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`.



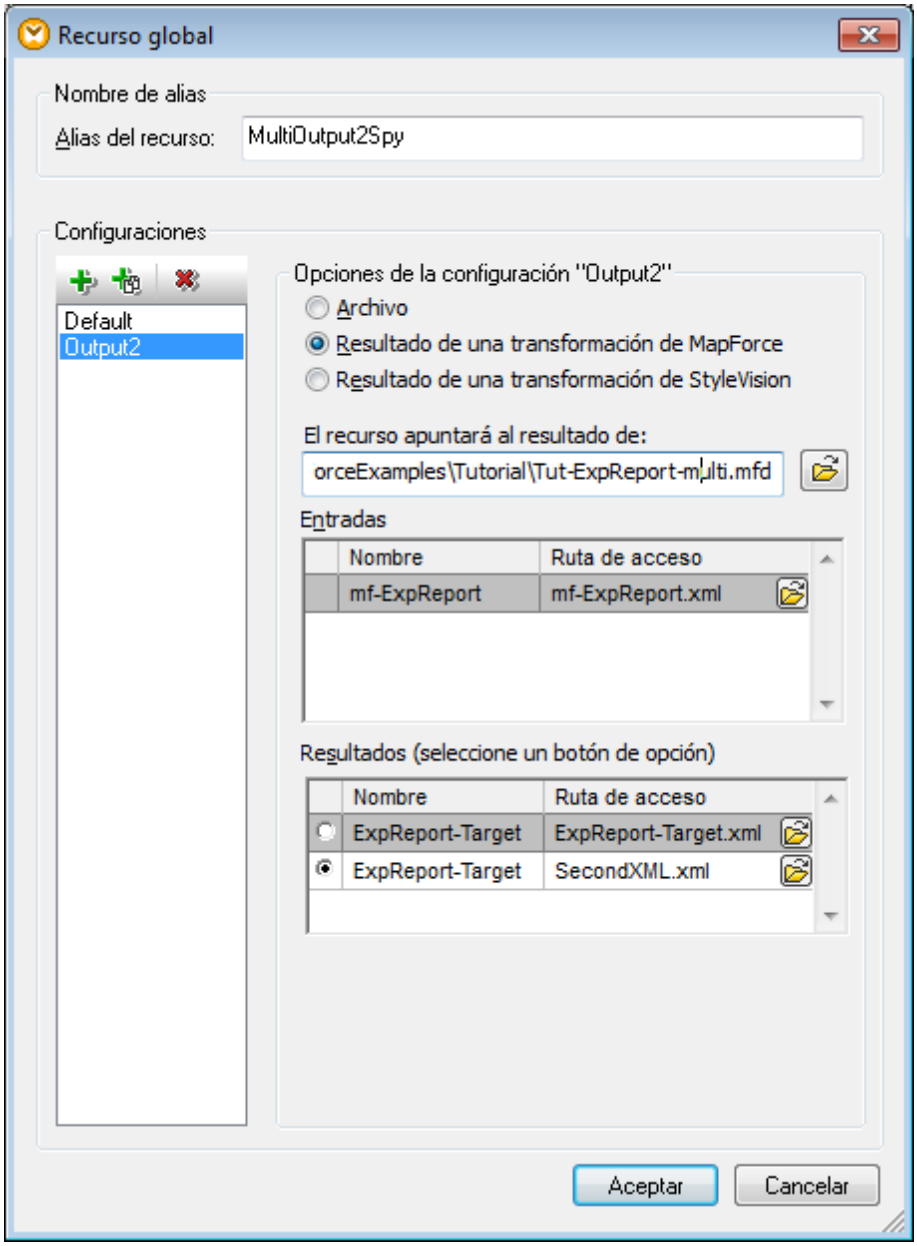
8. Escriba la ubicación nueva (p. ej. `C:\Temp`) y haga clic en **Guardar**. Esta ubicación puede ser diferente de la ubicación definida en la configuración del componente.
9. Haga clic en el icono del grupo *Configuraciones* (del cuadro de diálogo de recursos globales) para añadir una configuración 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`.



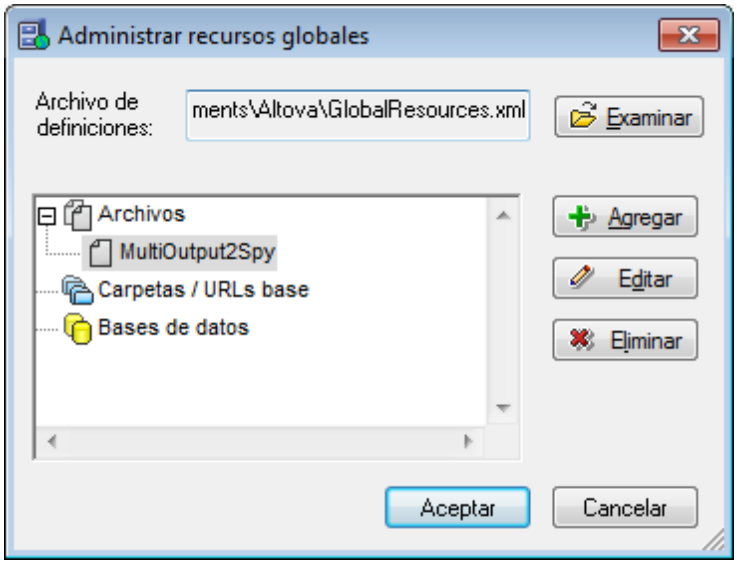
14. Haga clic en el icono y seleccione **Examinar** en el menú contextual para definir la ubicación nueva del archivo de salida, p. ej. `C:\Temp`.

    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.



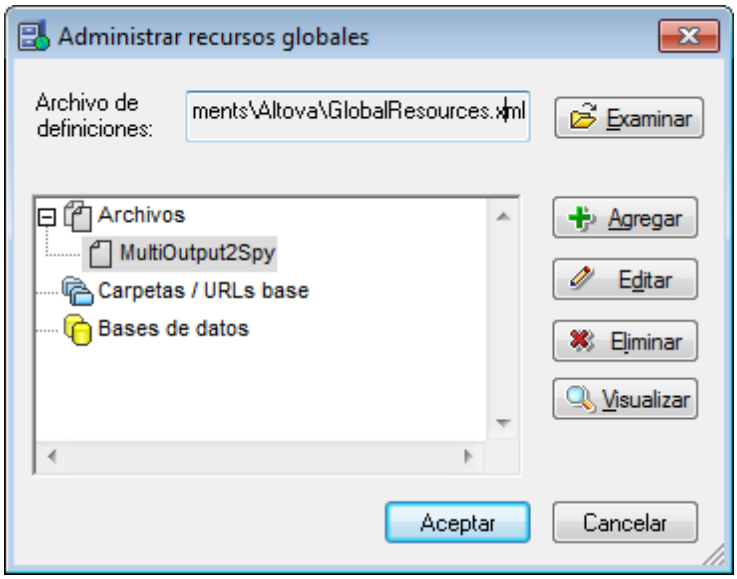
16. Haga clic en **Aceptar** para finalizar la fase de definición de recursos globales.

© 2014 Altova GmbH

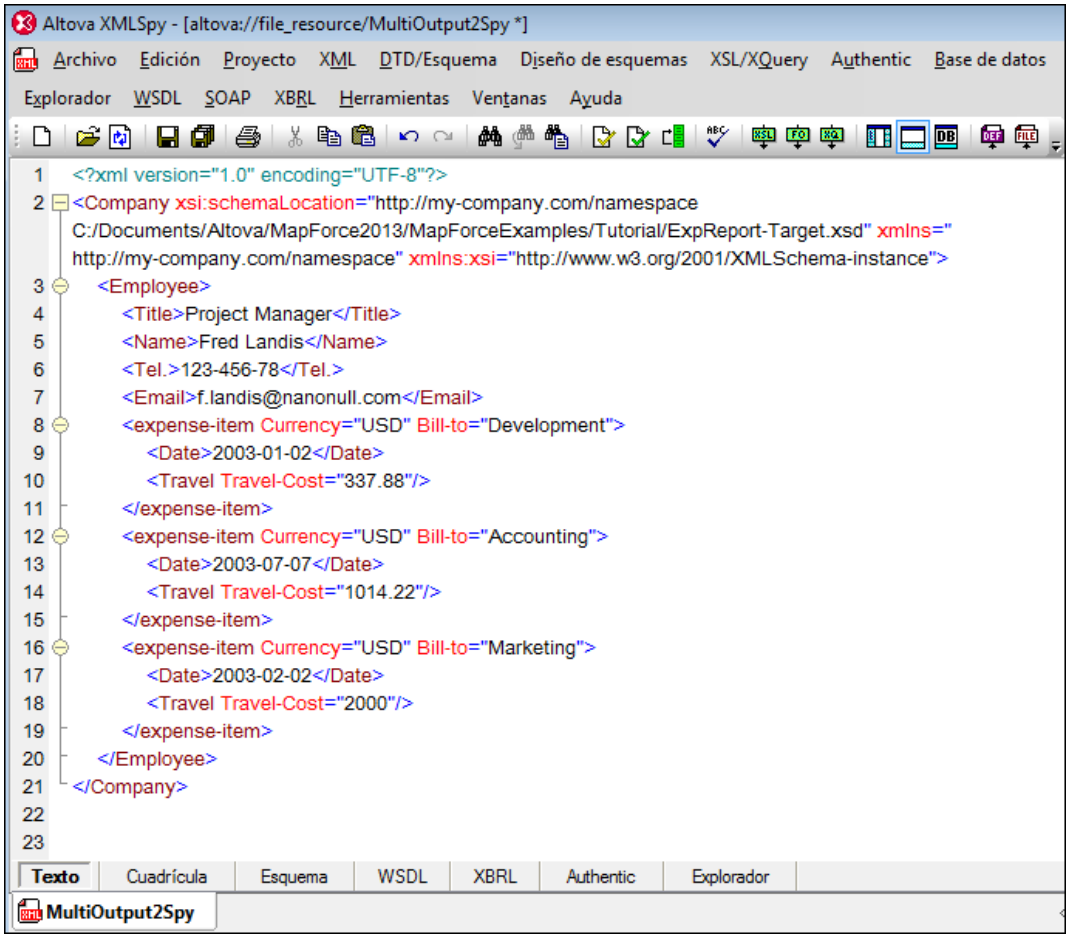
11.3.1 Iniciar el flujo de trabajo en la aplicación

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.



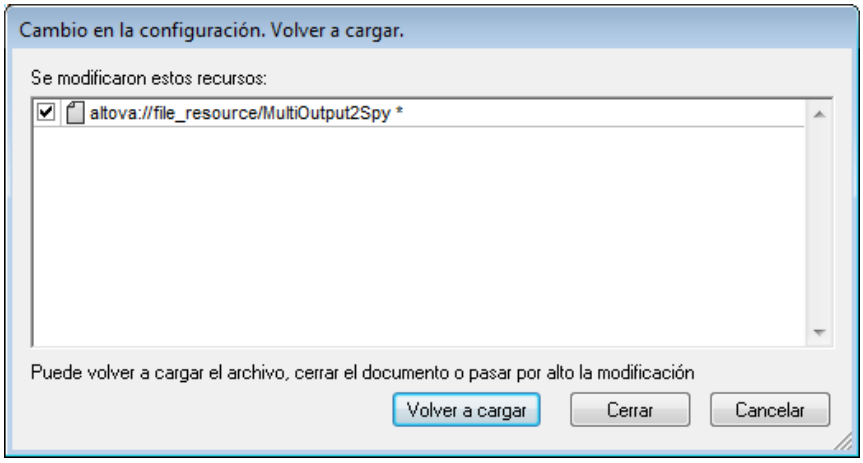
Nota:

- 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`.

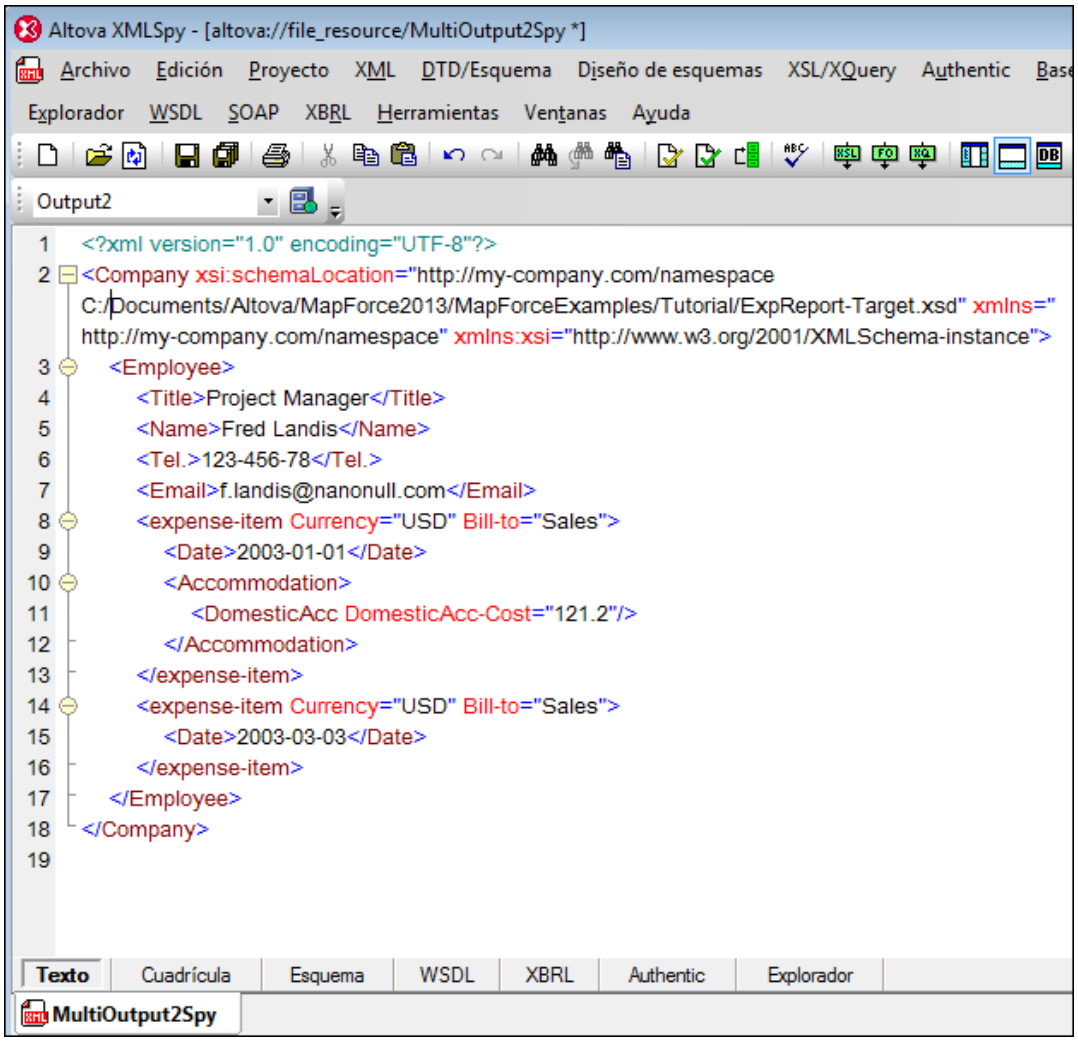
Para recuperar el archivo de salida que contiene todos los gastos menos los de viaje:

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

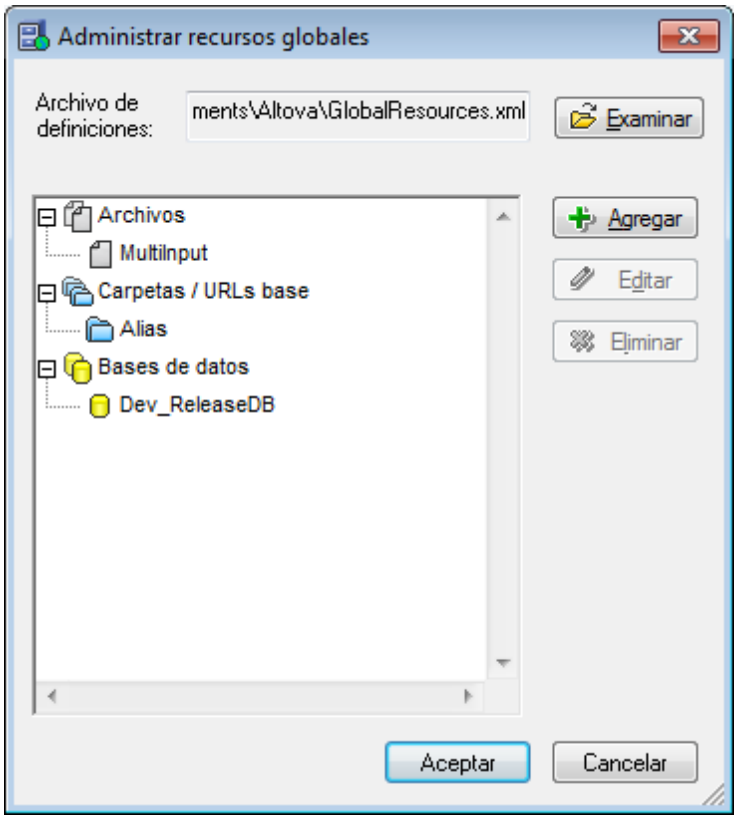
11.4 Recursos globales: propiedades

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.

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**.

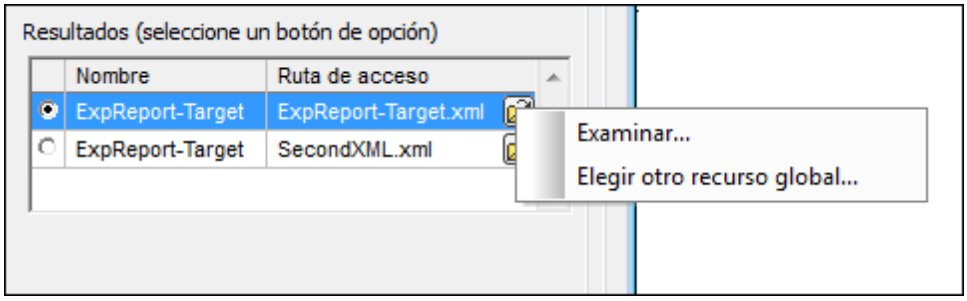
Para ver el resultado del flujo de trabajo de una aplicación:
Si una aplicación (p. ej. XMLSpy) llama a otra aplicación (p. ej. MapForce), entonces en el cuadro de diálogo "Administrar recursos globales" hay un botón más, el botón **Visualizar**.

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). 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*.

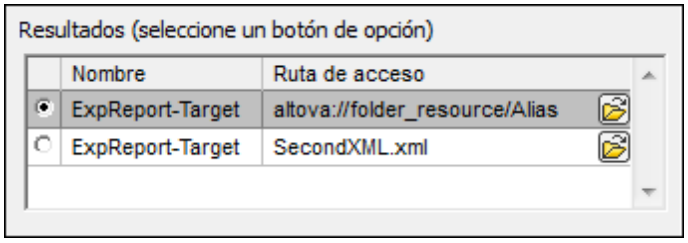
Seleccionar como recurso global los resultados de una transformación de MapForce
En una transformación de MapForce que tiene más de un archivo de resultados, puede definir uno de ellos como recurso global, seleccionando su botón de opción.



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.

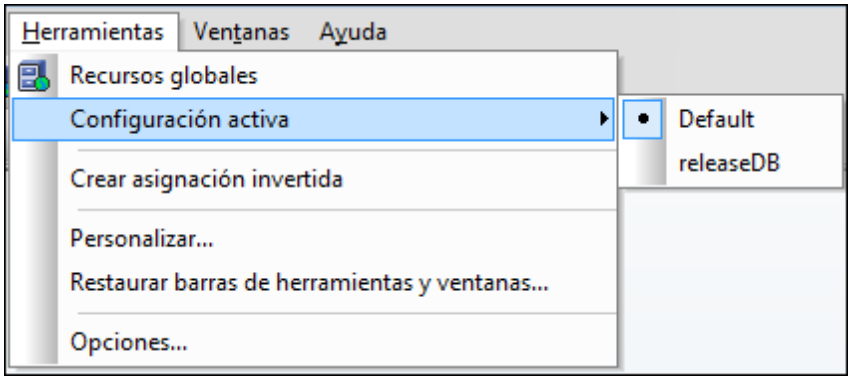
  La configuración seleccionada es la configuración activa en la aplicación. Es decir, allá donde se use un recurso global (o alias), se cargará el recurso asociado a la configuración activa de cada alias.

  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 2015

Archivos de entrada/salida dinámicos por componente

12 Archivos de entrada/salida dinámicos por componente

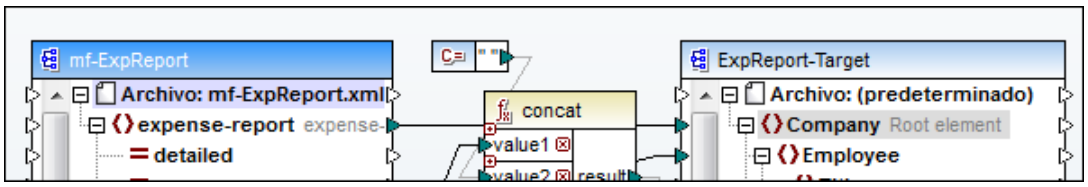
MapForce es capaz de procesar varios archivos de entrada/salida por componente y, por tanto, puede procesar todos los archivos de un directorio o de un grupo de archivos del directorio, usando caracteres de comodín en el componente de origen.

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

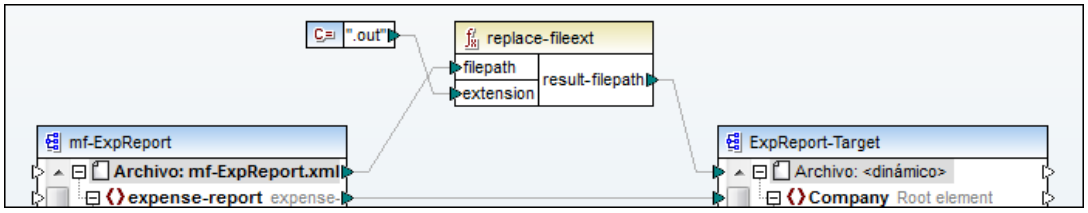
Observe el elemento **`Archivo:`** situado al principio de los componentes de esta asignación:



- 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.



Lenguajes de programación de MapForce compatibles con el uso de archivos dinámicos y comodines:

 © 2014 Altova GmbH

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)	*

* 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

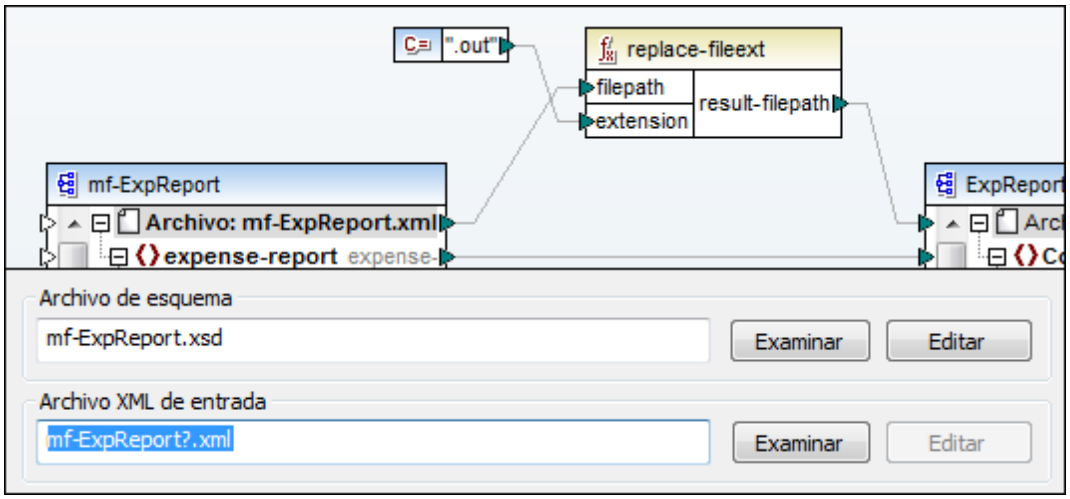
12.1 Nombres de archivo dinámicos: entrada/salida

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.

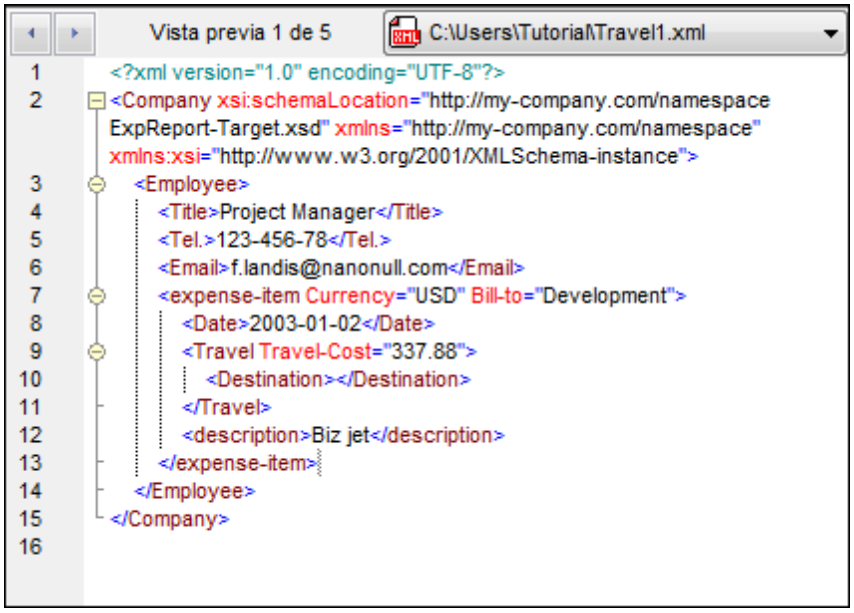
  Puede suministrar una secuencia de nombres de archivo mediante:

  - un archivo XML

Vista previa de asignaciones con entradas/salidas dinámicas

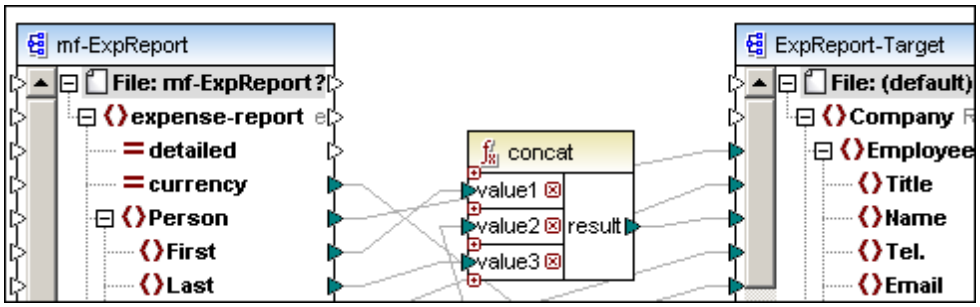
Al pasar al panel **Resultados** aparece el resultado de la asignación en una ventana de vista previa. Si la asignación produce varios archivos de salida, como en el ejemplo siguiente (*Vista previa 1 de 5*), cada archivo tiene su propia pantalla de vista previa en el panel **Resultados**. Haga clic 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.

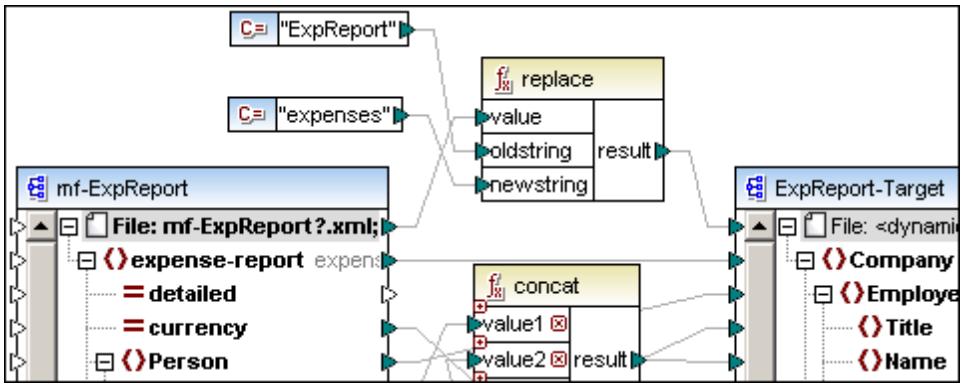


Varias entradas / varias salidas
Para asignar varios archivos a varios archivos de destino, es necesario generar nombres de archivo de salida únicos. En algunos casos, los nombres de archivo de salida se pueden derivar de cadenas de los datos de entrada. En otros casos, lo más práctico es derivar el 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).

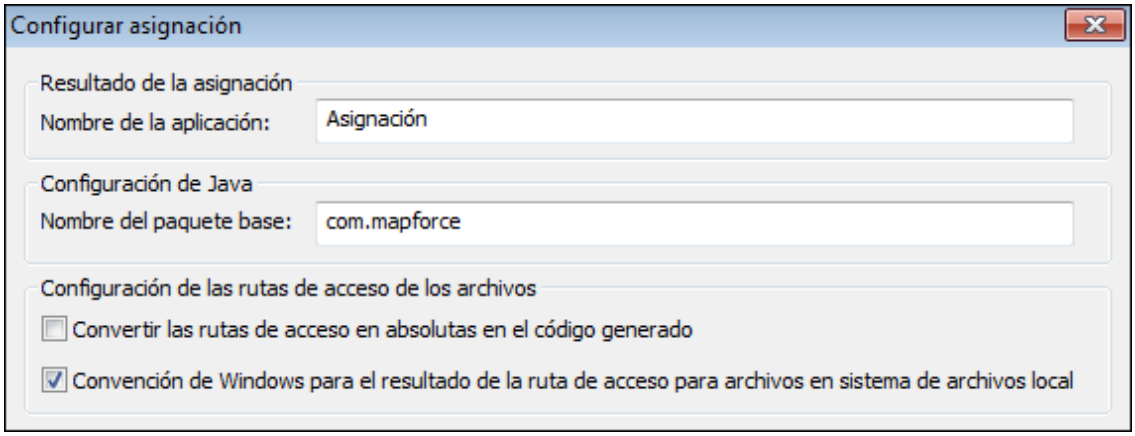
Nota: recomendamos no conectar los nodos **`Archivo:`** **directamente** sin usar funciones de procesamiento. Si lo hace, los archivos de entrada se sobrescribirán cuando se ejecute la

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.

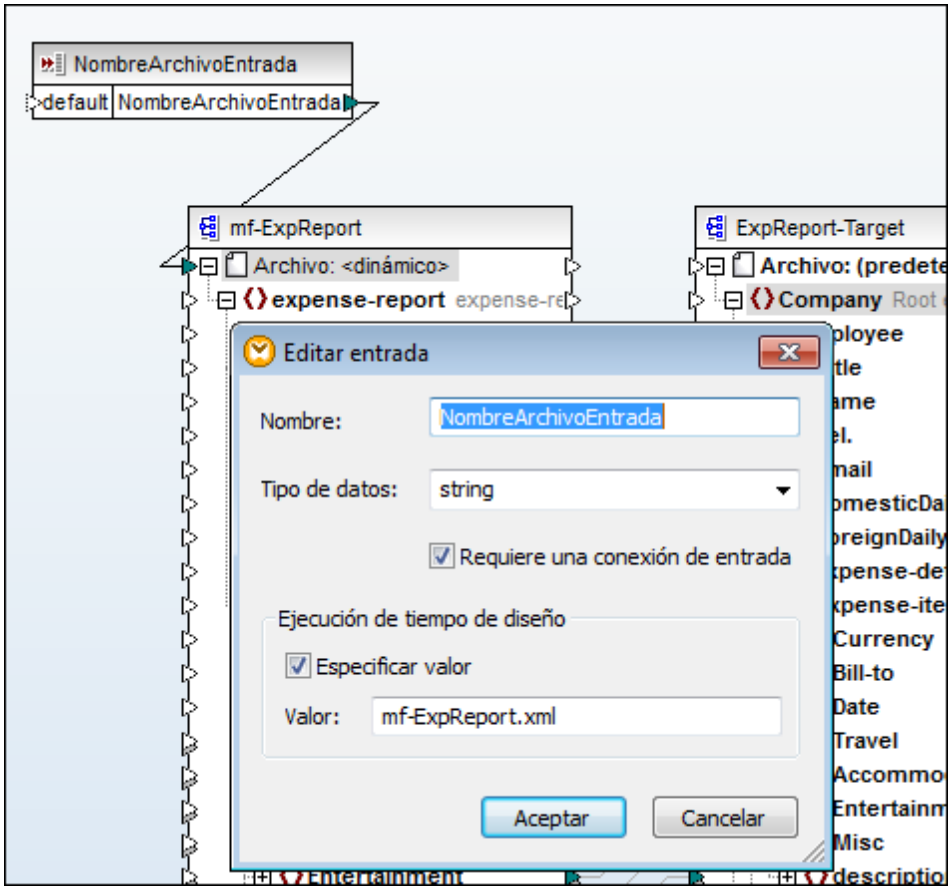
Si marca esta casilla, las rutas de acceso de tipo `archivo://URI` se convierten automáticamente en una ruta de archivo de Windows completa (p. ej. `C:\...`) para simplificar el resto del procesamiento.

 © 2014 Altova GmbH

12.2 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

12.3 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).

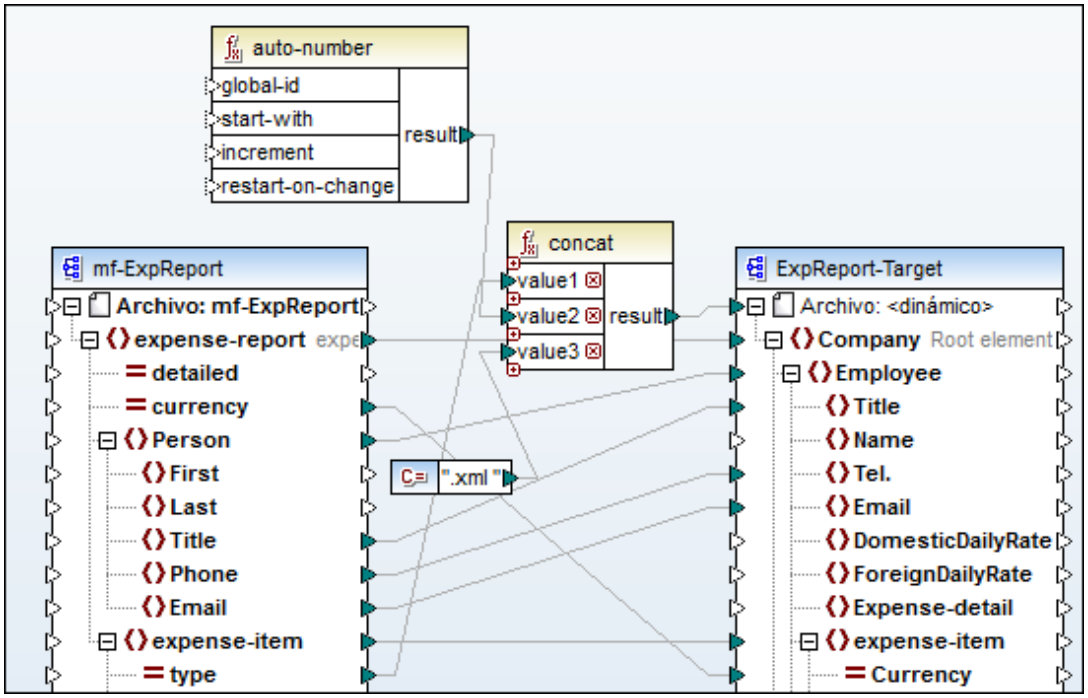
Person	
First	Fred
Last	Landis
Title	Project Manager
Phone	123-456-78
Email	f.landis@nanonull.com

expense-item (5)

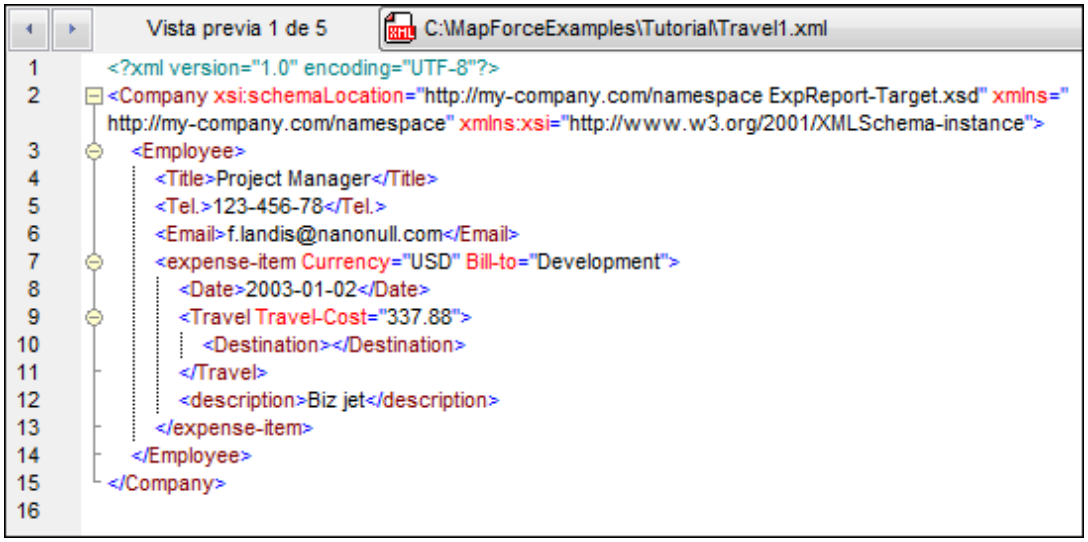
	type	expto	Date	Travel	Lodging
1	Travel	Development	2003-01-02	Travel Trav-cost=337.88	
2	Lodging	Sales	2003-01-01		Lodging
3	Travel	Accounting	2003-07-07	Travel Trav-cost=1014.22	
4	Travel	Marketing	2003-02-02	Travel Trav-cost=2000	
5	Meal	Sales	2003-03-03		

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**.



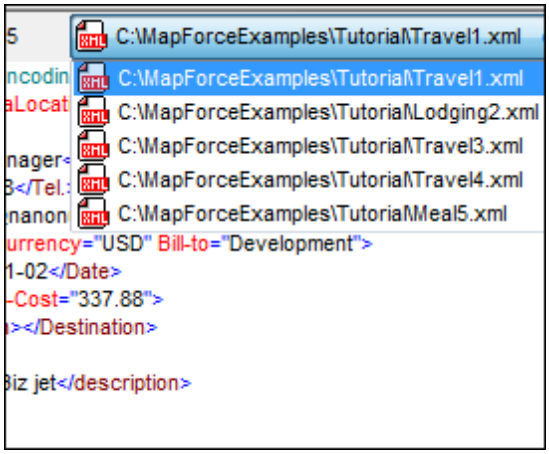
4. Cree las conexiones que aparecen en la imagen (entre **`type`** y **`value1`**, entre **`auto-number`** y **`value2`** y entre la constante y **`value3`**).
5. Conecte el parámetro de salida **`result`** de la función **`concat`** con el elemento **`Archivo:`** del componente de destino. Observe cómo 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.

12.4 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.

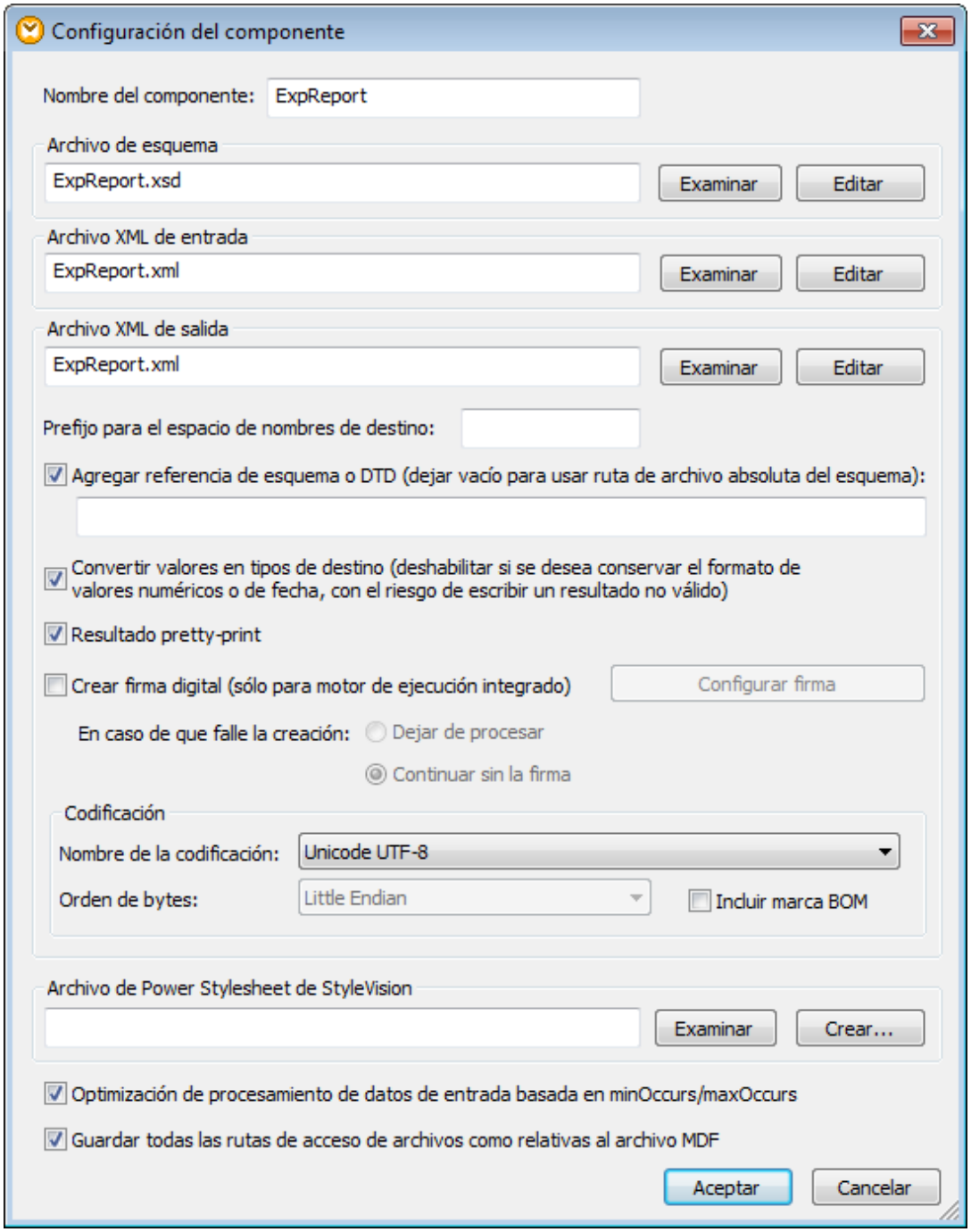
Guardar todas las rutas de acceso relativas en el archivo MFD
Esta nueva opción, disponible en los cuadros de diálogo de configuración de todos los componentes, sean del tipo que sean, guarda todas las rutas de acceso de archivo (del componente) como relativas a la ubicación del archivo MFD actual. En 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

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 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.

 © 2014 Altova GmbH

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.

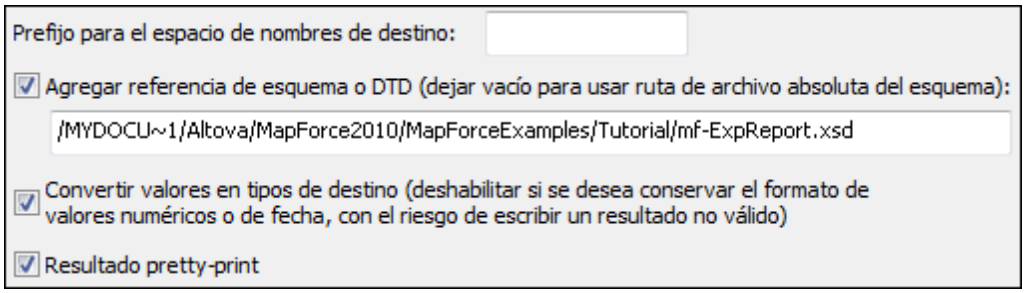
Recuerde que el nombre de archivo de instancia de origen también se utiliza para:

- 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.

Nota: en este cuadro de texto también se puede introducir una URL, p. ej. http://mylocation.com/mf-expreport.xsd.

Altova MapForce 2015

Variables intermedias

13 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

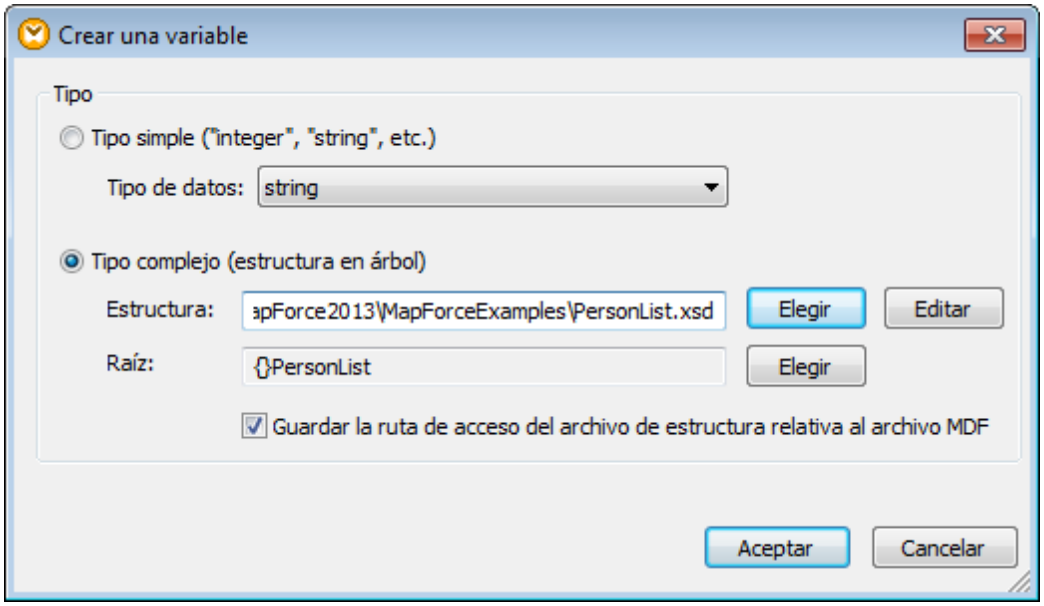
Asignaciones en cadena	Variables
Abarcan dos pasos totalmente independientes entre sí.	Se evalúan dependiendo del contexto/ámbito. Son controladas por una conexión **`compute-when`**
Cuando se ejecuta la asignación, los resultados intermedios se almacenan de forma externa en archivos XML	Cuando se ejecuta 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.

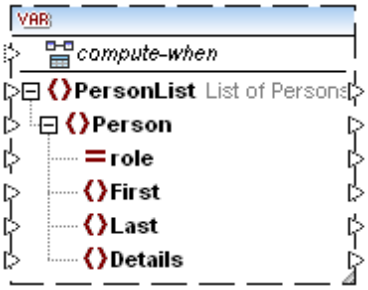
1. Seleccione **Insertar | Variable** o haga clic en el icono de la barra de herramientas. Aparece un cuadro de diálogo donde puede elegir si se inserta una variable de tipo simple o de tipo complejo.

 © 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.

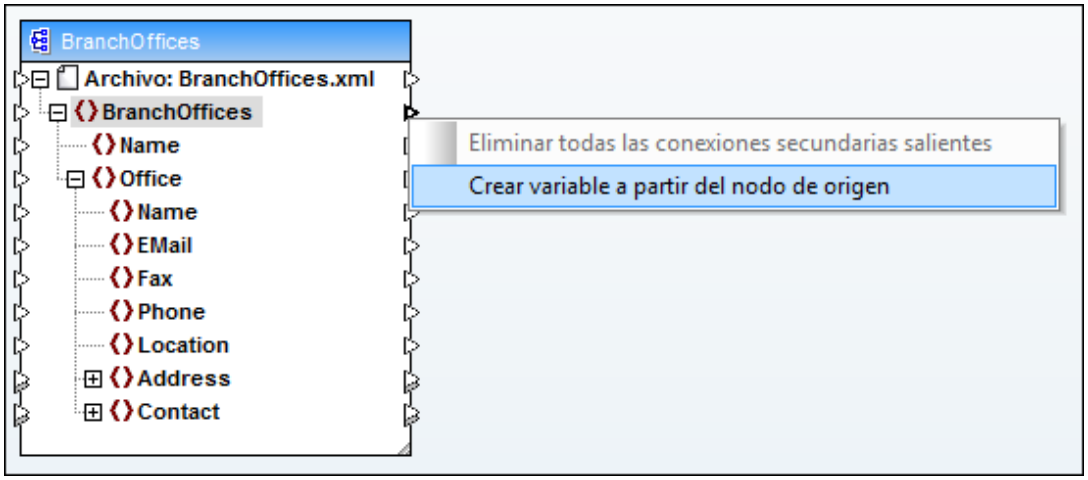
Ejemplo de variable compleja:



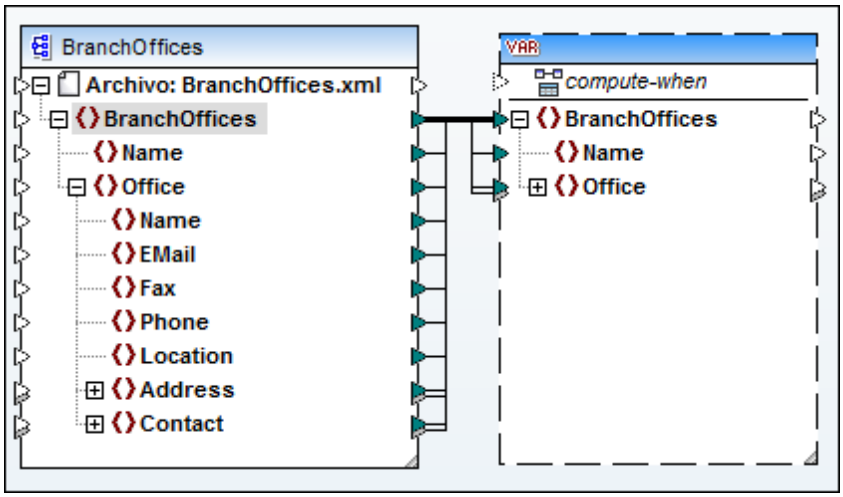
Ejemplo de variable simple (tienen un solo valor/nodo asignable, p. ej. una cadena de texto, un entero, etc. No obstante, el nodo **`value`** puede duplicarse.)

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**.

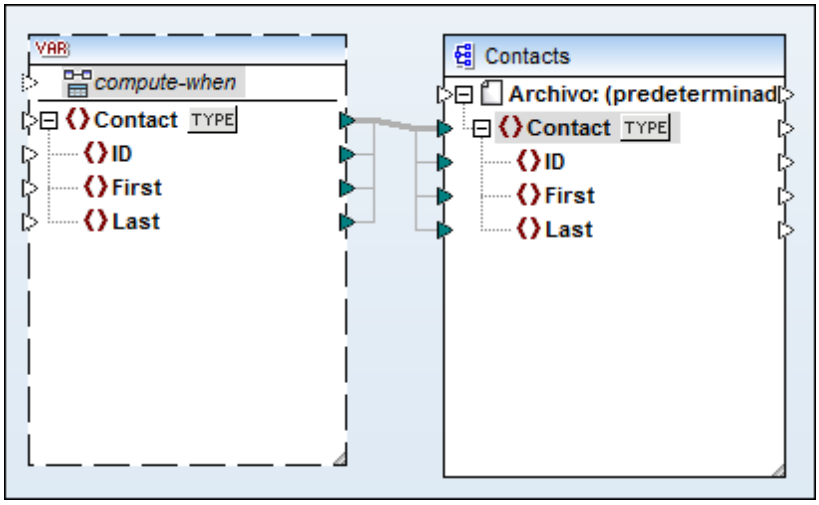


Esto crea una variable compleja usando el mismo esquema de origen y la variable se conecta automáticamente con todos los elementos del componente mediante una conexión de copia total.



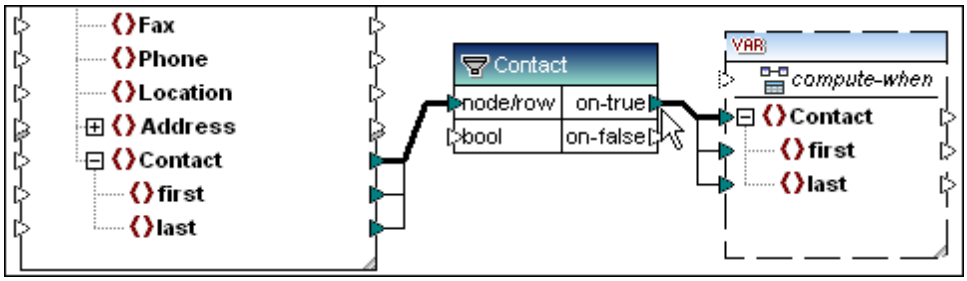
- Haga clic con el botón secundario en el icono de **entrada** de un componente de destino y en el menú contextual seleccione **Crear variable para el nodo de destino**.

 © 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.

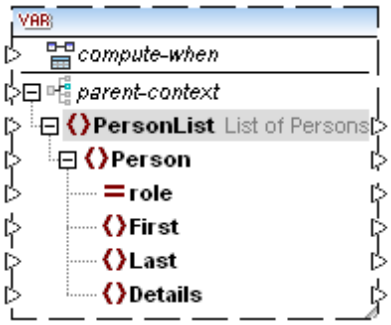
En una función definida por el usuario, la variable se evalúa cada vez que se llama a la función y antes de que se evalúe el resultado de la función propiamente dicha.

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:

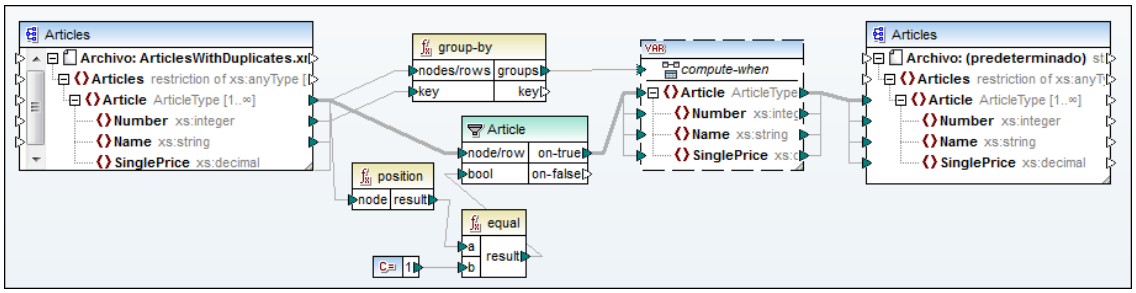
- Haga clic con el botón secundario en el nodo raíz de la variable y en el menú contextual seleccione **Agregar contexto primario**.
  Esto añade un nodo nuevo llamado **`parent-context`** a la jerarquía existente.



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.

13.1 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.

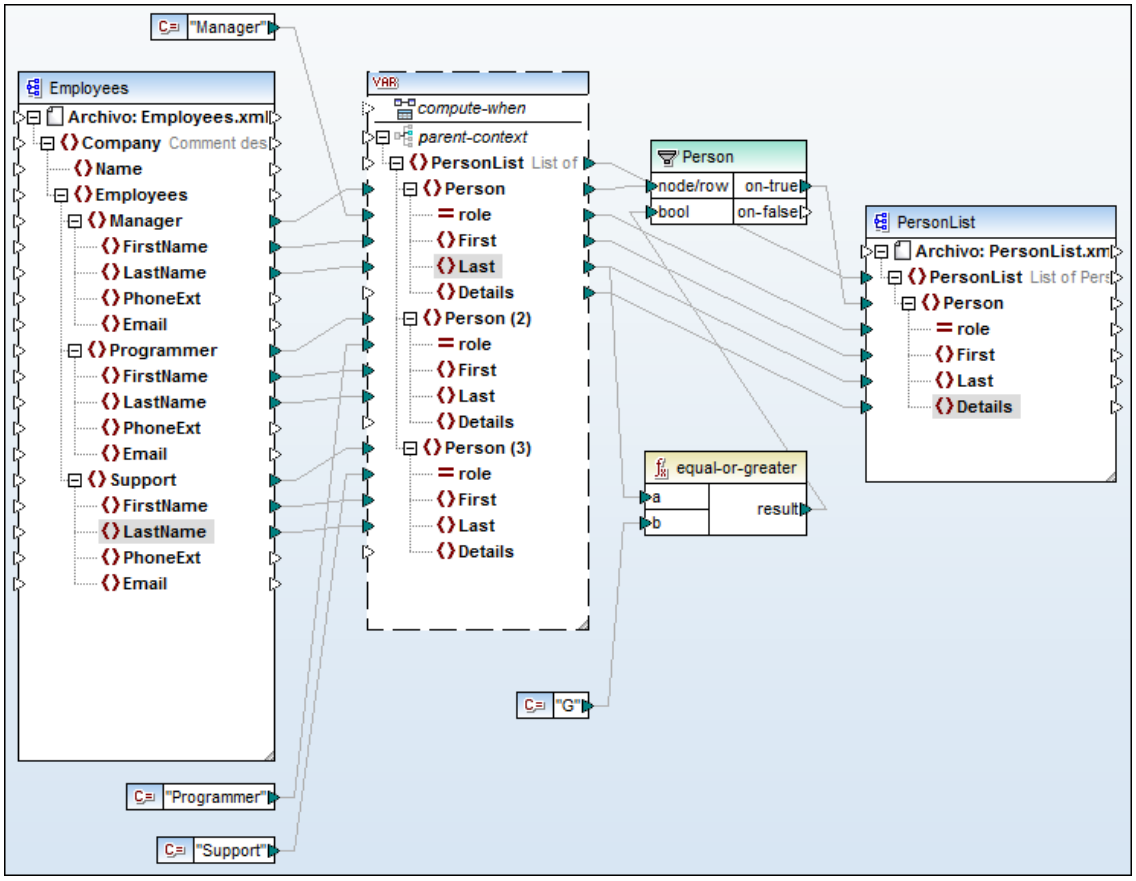
La salida del grupo (de la función **`group-by`**) está conectada al elemento **`compute-when`** de la variable. Esto significa que la variable se evalúa una vez por cada grupo, en el contexto del grupo. Esto establece un nivel de contexto adicional, como si hubiésemos conectado un elemento primario 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.

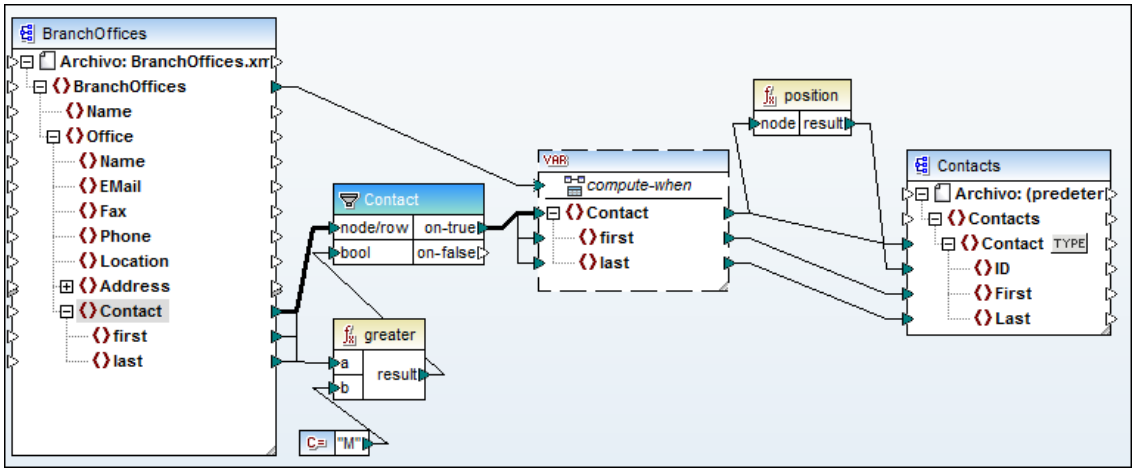


El nodo **`Person`** de la **variable** se duplicó dos veces y se añadió un filtro para seleccionar a las personas (nodos **`Person`**) cuyo apellido (**`Last`**) empieza con una letra superior a la G (G o H, I, etc.)

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.

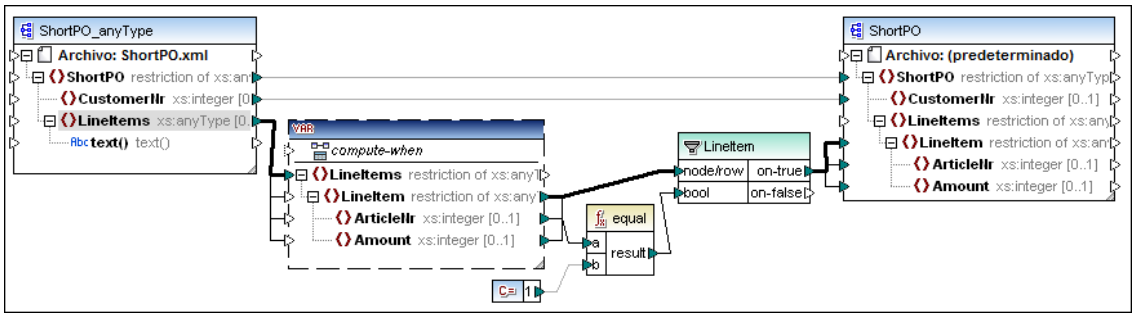
Después estos contactos se transmiten (desde la variable) hasta el componente de destino y la función **`position`** numera estos contactos de forma secuencial.



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 2015

Bibliotecas y funciones

14 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

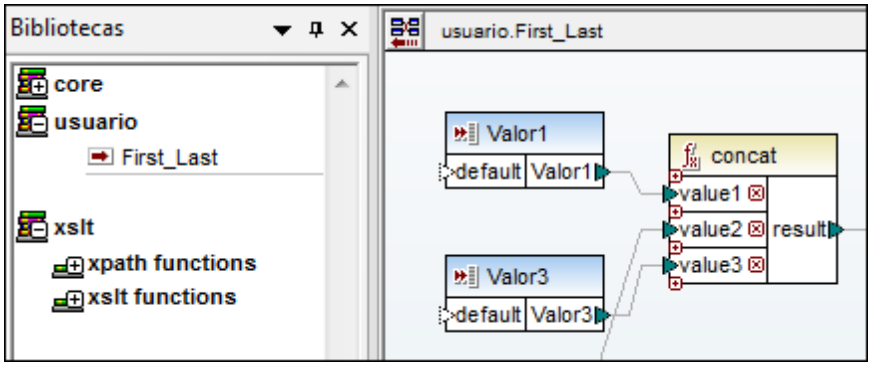
Referencia de funciones

© 2014 Altova GmbH

14.1 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, 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, o nodos XML.

Las funciones definidas por el usuario permiten:

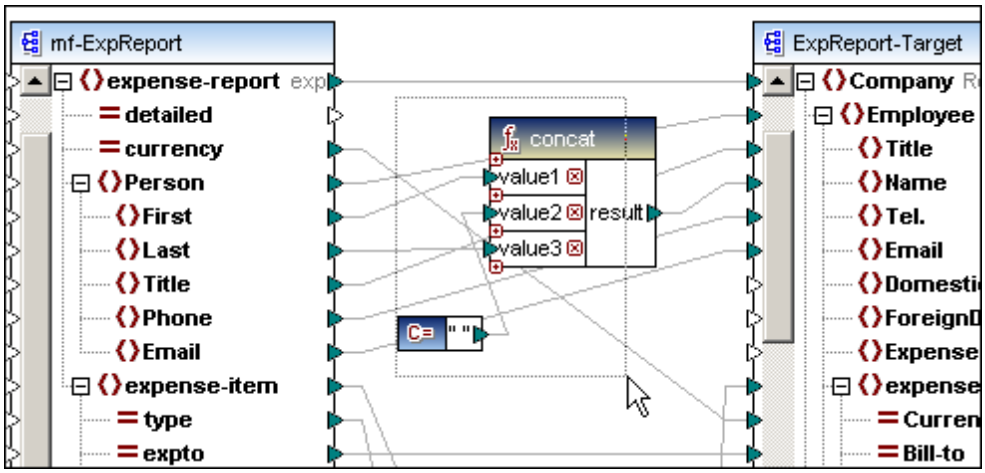
- reunir varias funciones de procesamiento en un solo componente (p. ej. para dar formato a un campo concreto o para buscar un valor)
- reutilizar estos componentes un número ilimitado de veces
- importar funciones definidas por el usuario en otras asignaciones (cargando el archivo de asignación como 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`.

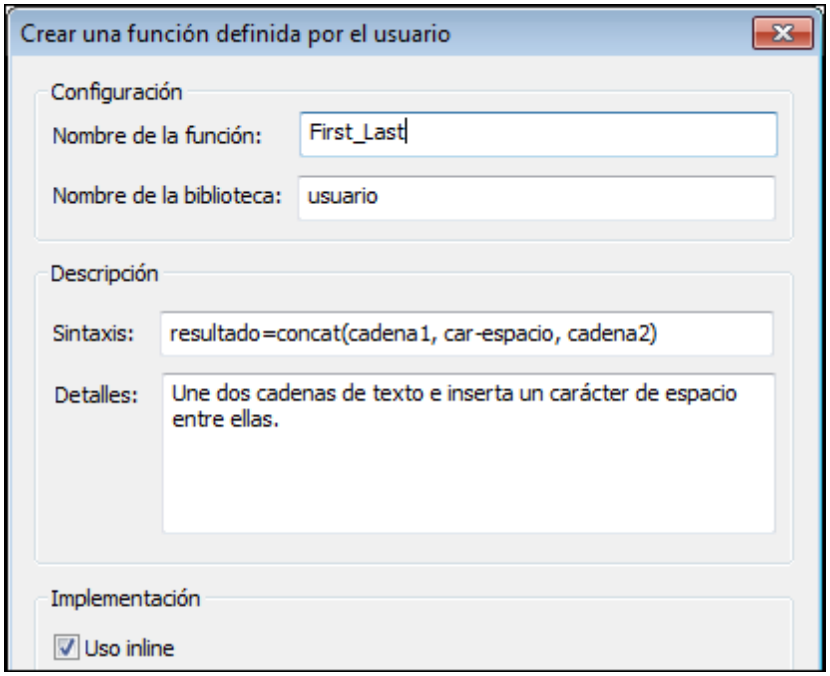
Para crear una función definida por el usuario a partir de componentes ya existentes:

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**).

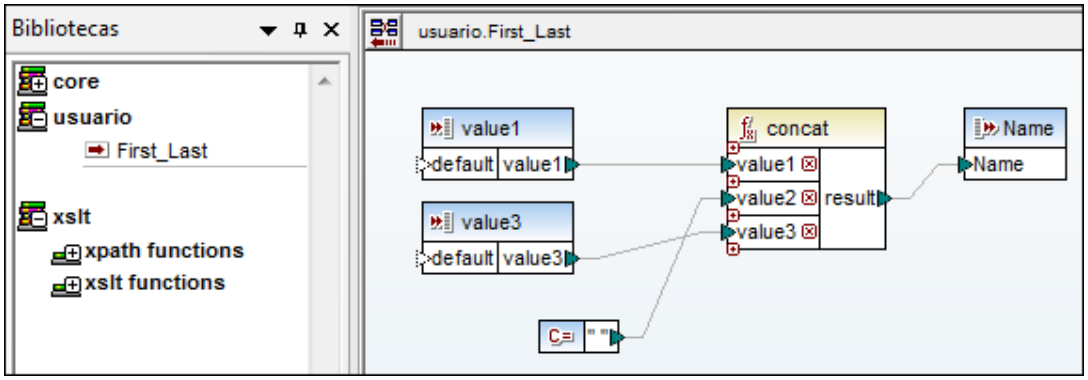


2. Seleccione la opción de menú **Función | Crear una función definida por el usuario a partir de la selección**.
3. Aparece un cuadro de diálogo donde puede escribir el nombre de la nueva función definida por el usuario (es decir, **`First_Last`**).
   Nota: los caracteres válidos para el nombre de 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

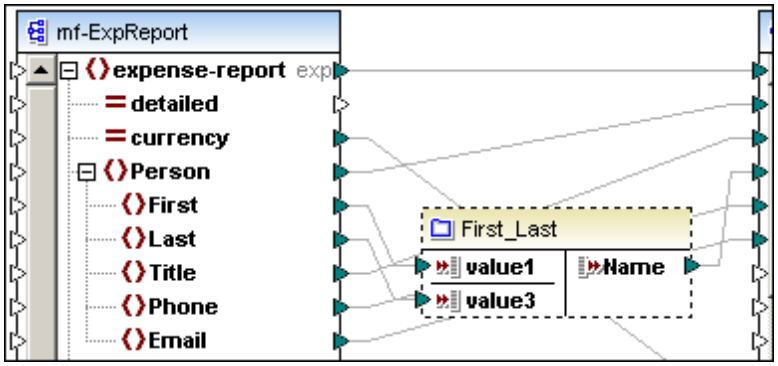


Los elementos que componen la función aparecen en un panel nuevo cuyo título es el nombre de la función. La biblioteca **usuario** aparece en el la ventana Bibliotecas y debajo el nombre de la función recién creada **`First_Last`**.



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.

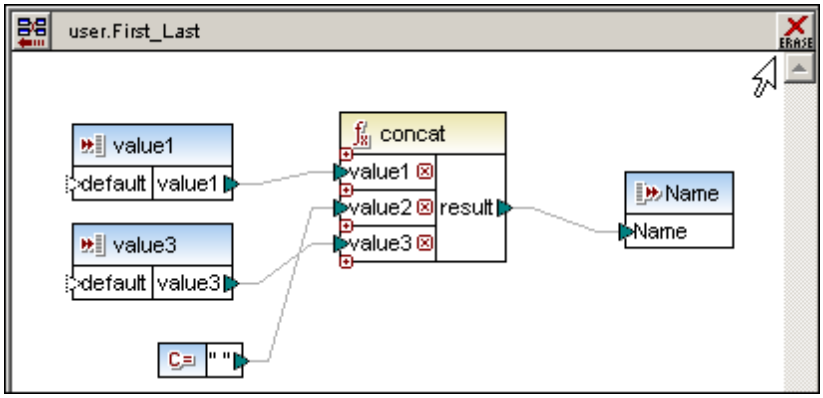
Pulse el botón **Atrás** para retroceder en el historial.

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:

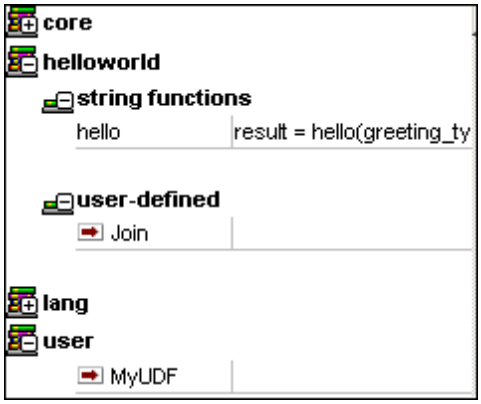
1. En la parte inferior de la ventana Bibliotecas haga clic en el botón **Agregar o quitar bibliotecas**. En el cuadro de diálogo que aparece haga clic en el botón **Agregar** y seleccione el archivo ***.mfd** cuyas funciones definidas por el usuario desea importar. Las funciones definidas por el usuario 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 .

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:

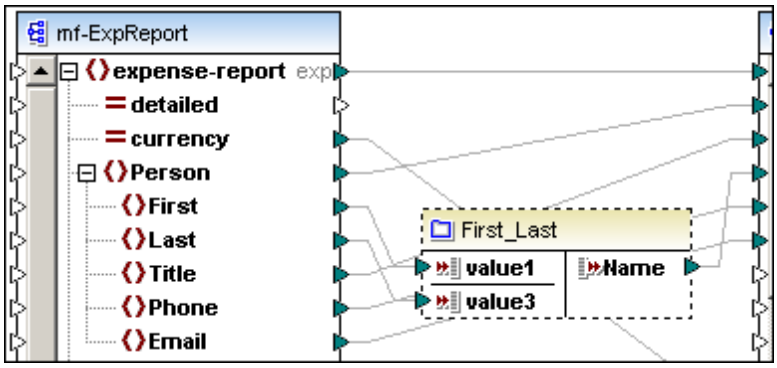
- la función **`hello`** de la biblioteca **`helloworld`** se importó de una biblioteca personal
- la función **`Join`** de la biblioteca **`helloworld`** es una función definida por el usuario presente en el archivo *.mfd actual
- la función **`MyUDF`** de la biblioteca **`user`** también es una función definida por el usuario presente en el archivo *.mfd actual



Con Java seleccionado

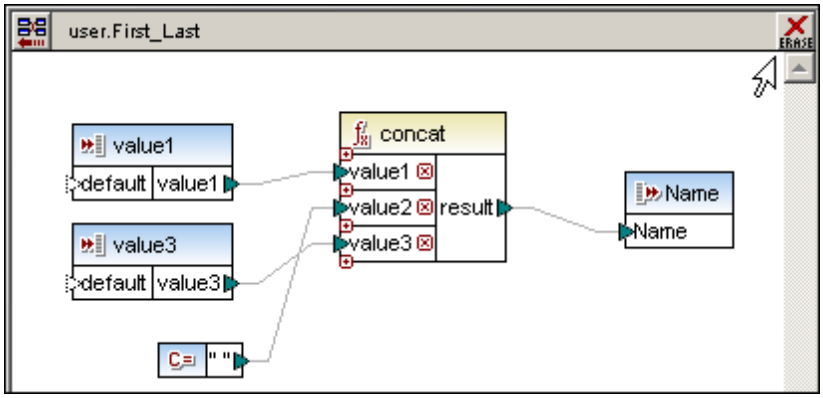
No olvide que, si realiza cambios en una función importada, cuando guarde el archivo ***.mfd** los cambios se aplicarán también a las asignaciones desde las que se importó la función.

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:

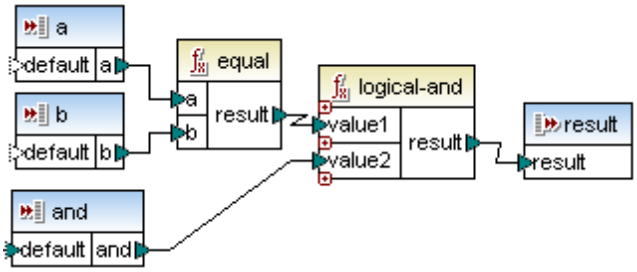
- la posición del componente y las acciones de ajuste de tamaño no se pueden deshacer.
- los componentes de entrada y salida recién añadidos se crean bajo la entrada y salida creada inmediatamente antes.
- puede mezclar parámetros complejos y simples. El orden de los parámetros se deduce de la posición del componente.

 © 2014 Altova GmbH

14.1.1 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**.

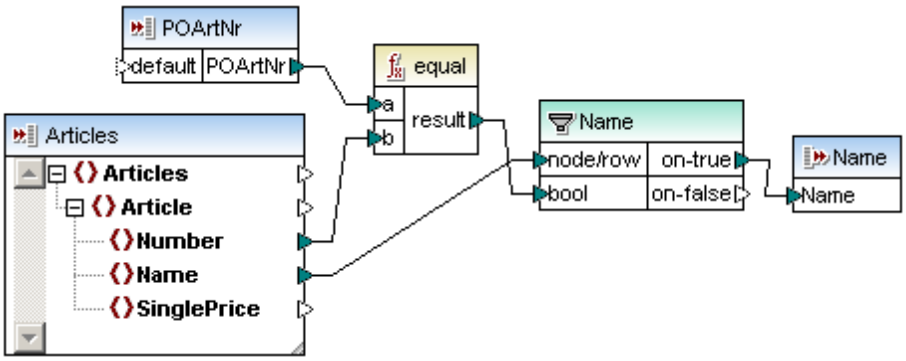
- Componentes/parámetros de entrada: **a**, **b**, **and**
- Componentes/parámetros de salida: **result**



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`

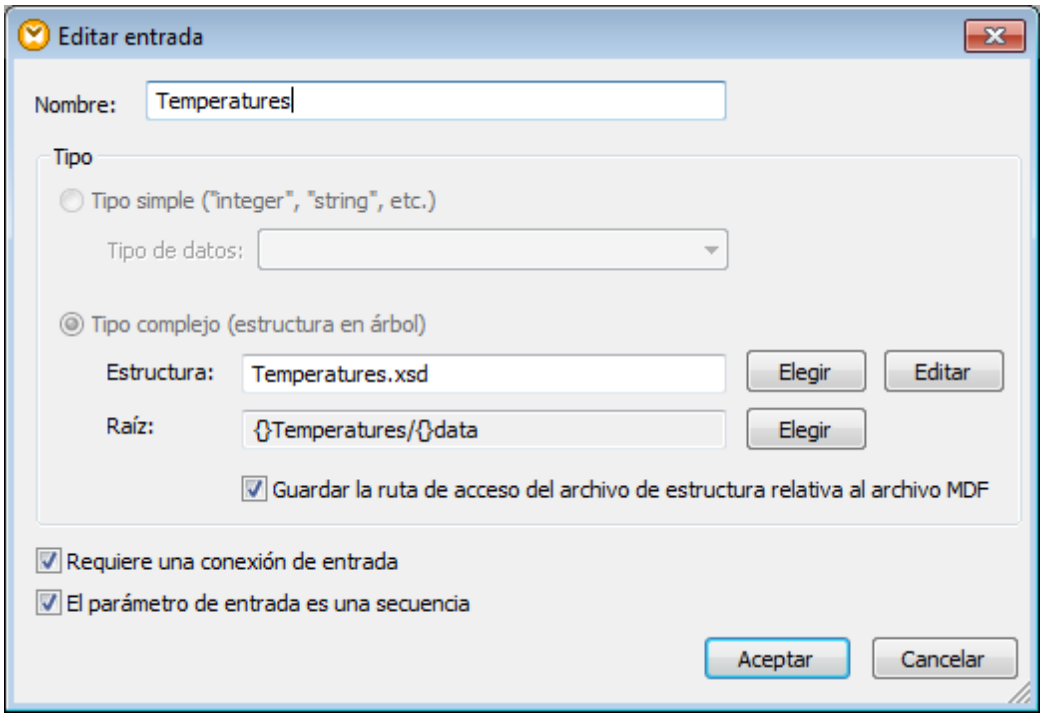
Nota: las funciones definidas por el usuario de las dos imágenes anteriores están disponibles en el archivo de ejemplo **`PersonListByBranchOffice.mfd`** , guardado en la carpeta `...\MapForceExamples`.

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.

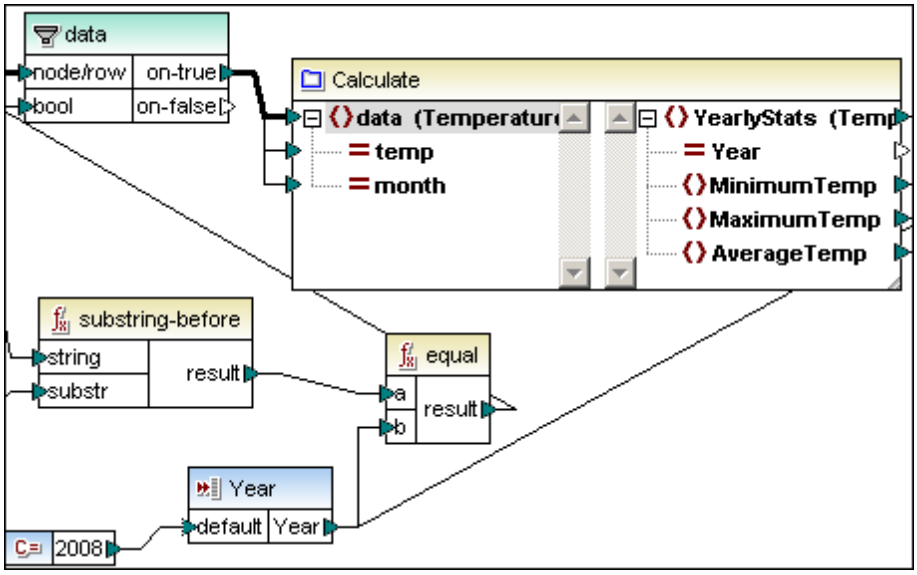
Si activa la casilla *El parámetro de entrada es una secuencia,* el componente trata la entrada como si fuera una secuencia de valores. Si desactiva la casilla, la entrada se trata como si fuera un valor único.



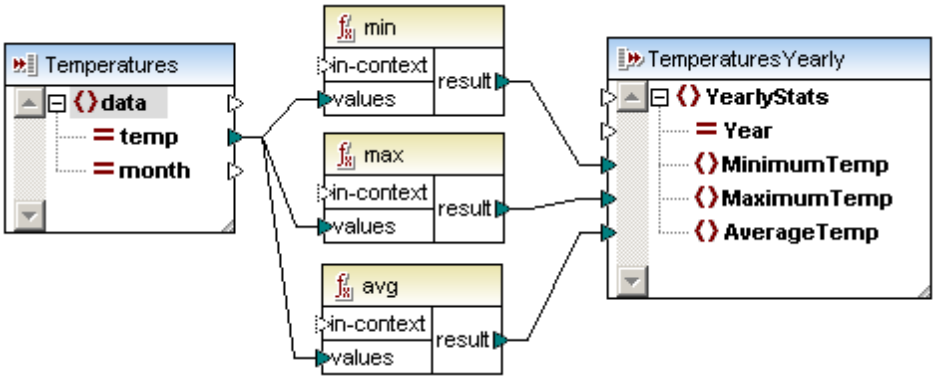
Este tipo de datos de entrada (de secuencia o no secuencia) determina **con qué frecuencia** se llama a la función.

- 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

14.1.2 Funciones definidas por el usuario inline y normales

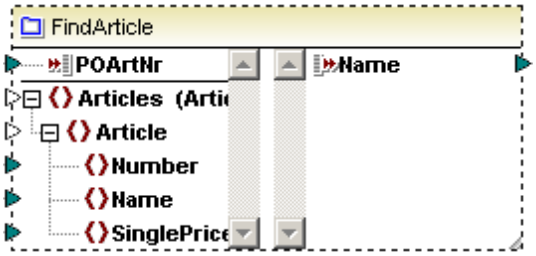
Las funciones inline son diferentes de las demás funciones por la manera en que se implementan cuando se genera código.

- 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.

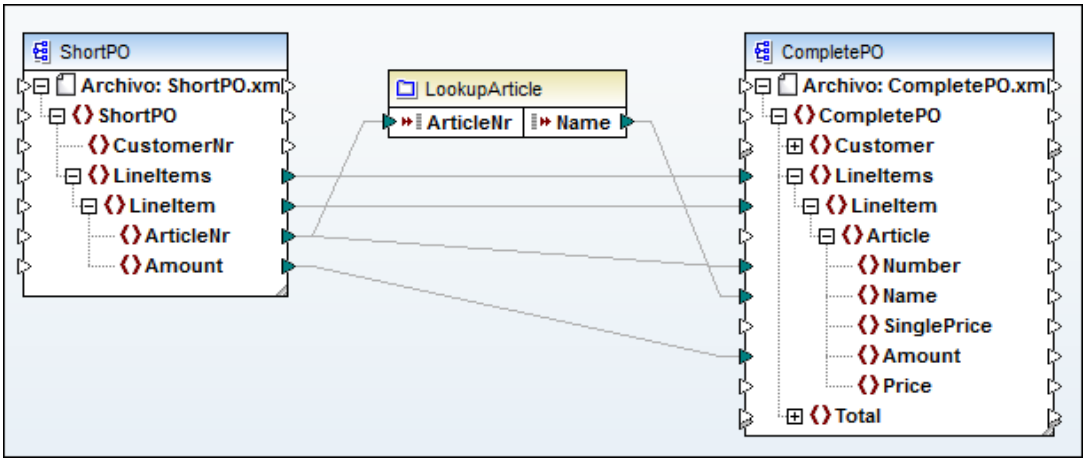
 © 2014 Altova GmbH

14.1.3 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**.

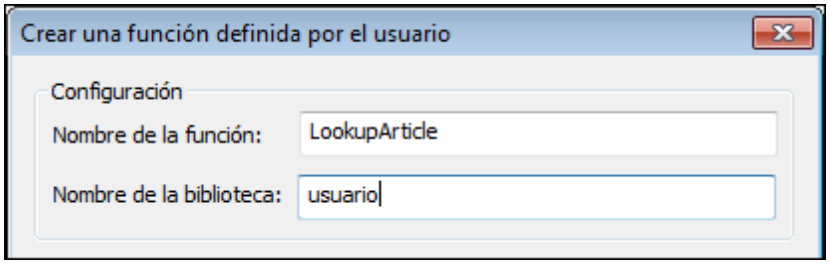


Instrucciones generales:

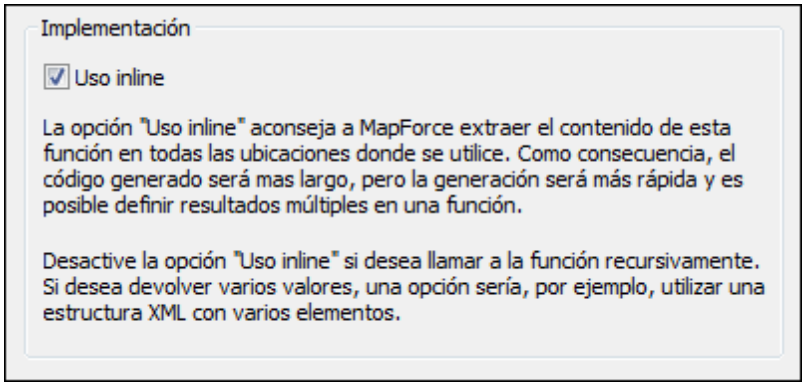
- Inserte el esquema `ShortPO.xsd` y asigne el archivo `ShortPO.xml` como archivo XML de entrada.
- Inserte el esquema `CompletePO.xsd` y seleccione el elemento raíz **`CompletePO`**.
- Inserte una función definida por el usuario nueva siguiendo los pasos descritos a continuación.

Para crear una función definida por el usuario desde cero:

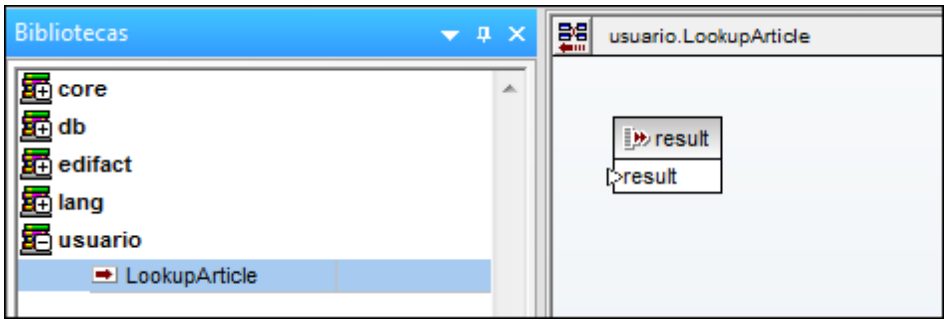
1. Seleccione la opción de menú **Función | Crear función definida por el usuario**.
2. En el cuadro de diálogo que aparece escriba el nombre de la función, p. ej. **`LookupArticle`**, y la biblioteca en la que desea almacenarla (p. ej. **`usuario`**).



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`**.



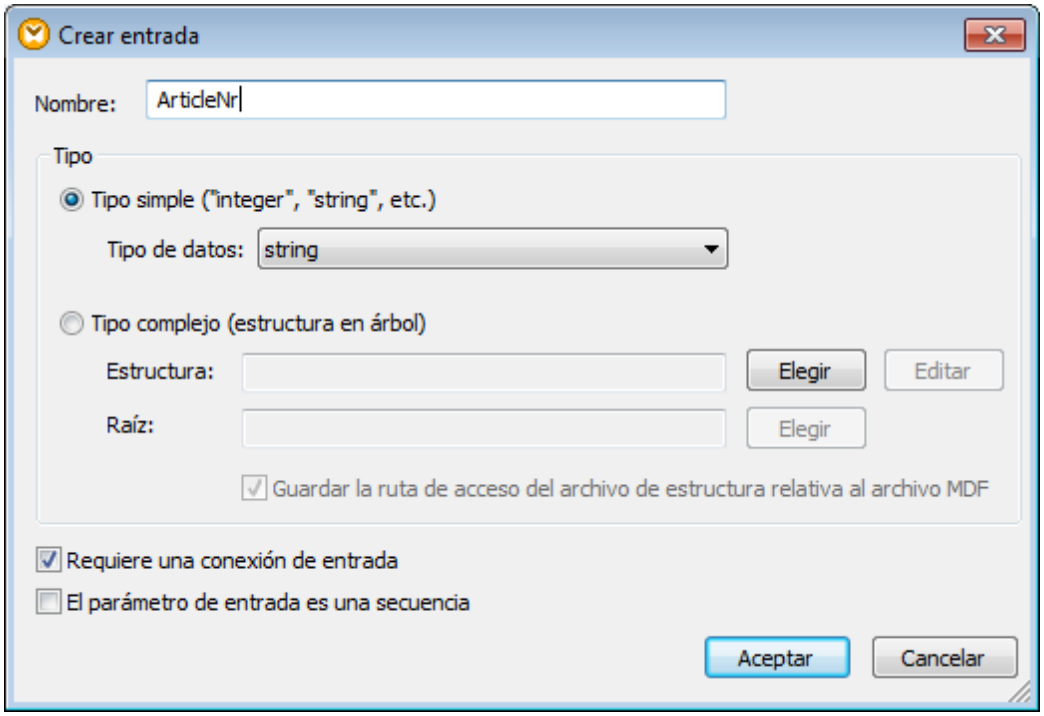
Con Java seleccionado

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`**.

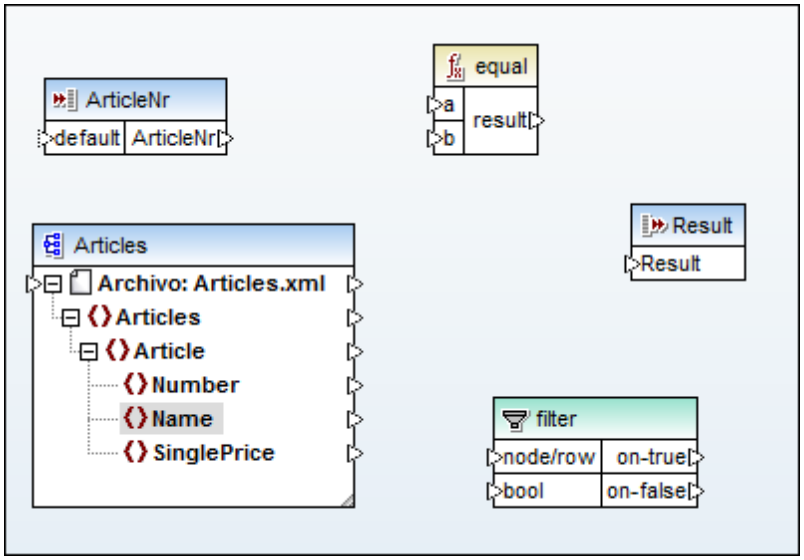
3. Haga clic en el icono **Insertar esquema o archivo XML** para insertar el esquema **`Articles`** y seleccionar el archivo XML **`Articles`** como origen de datos.
4. Haga clic en el icono **Insertar componente de entrada** para insertar un componente de entrada.
5. Escriba el nombre del parámetro 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.

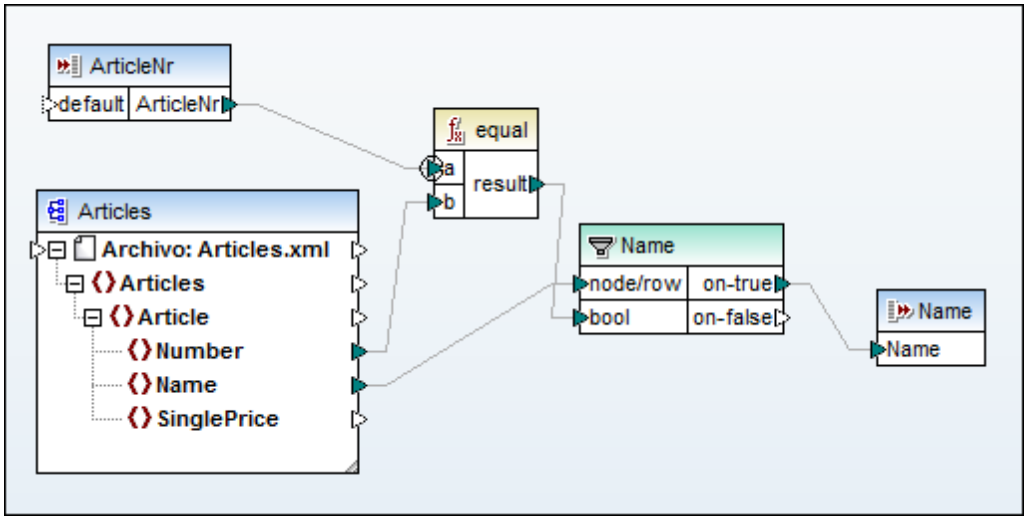
6. Inserte la función **`equal`** de la biblioteca **core | logical**.
7. Inserte un componente de filtrado haciendo clic en el icono **Insertar filtro** de la barra de herramientas.



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.



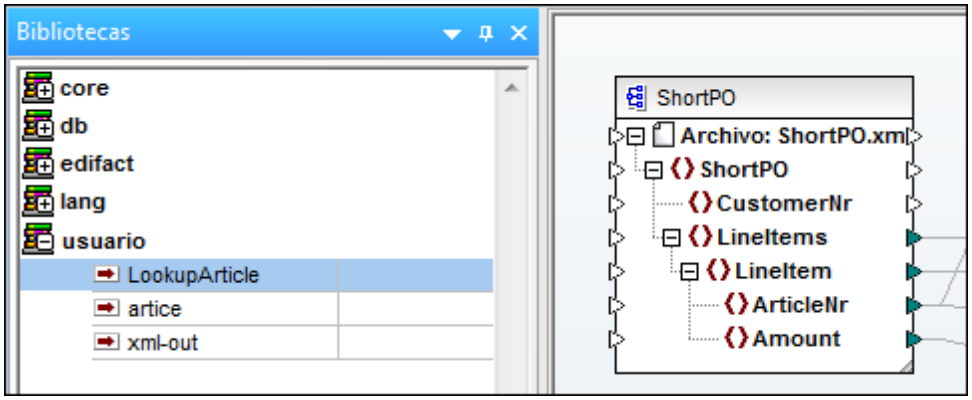
La función definida por el usuario ya está definida.

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**.



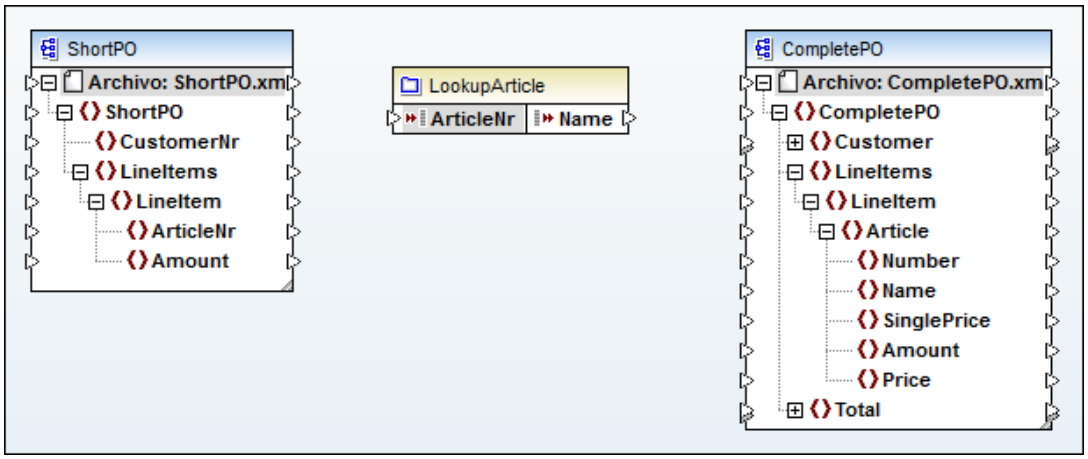
Con Java seleccionado

11. Arrastre la función **`LookupArticle`** hasta el panel **Asignación**.

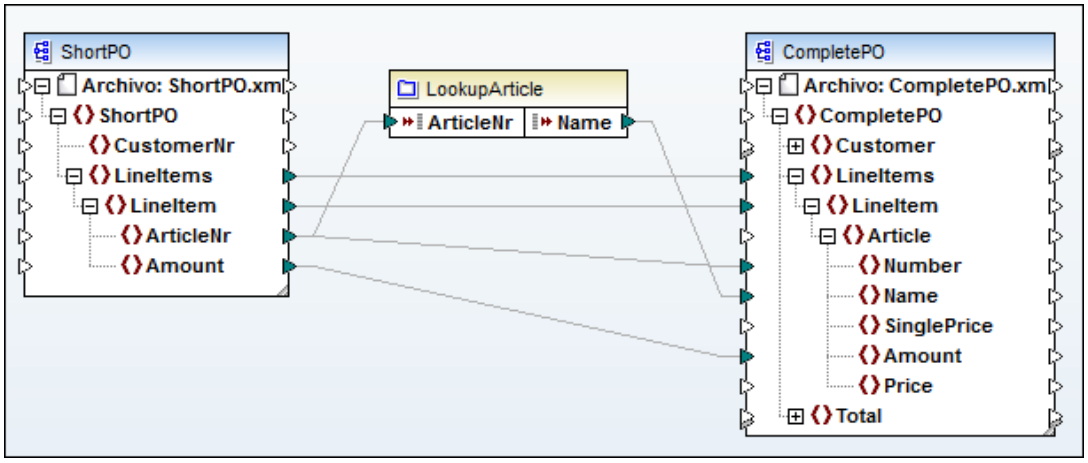
 © 2014 Altova GmbH

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.



14.1.4 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.

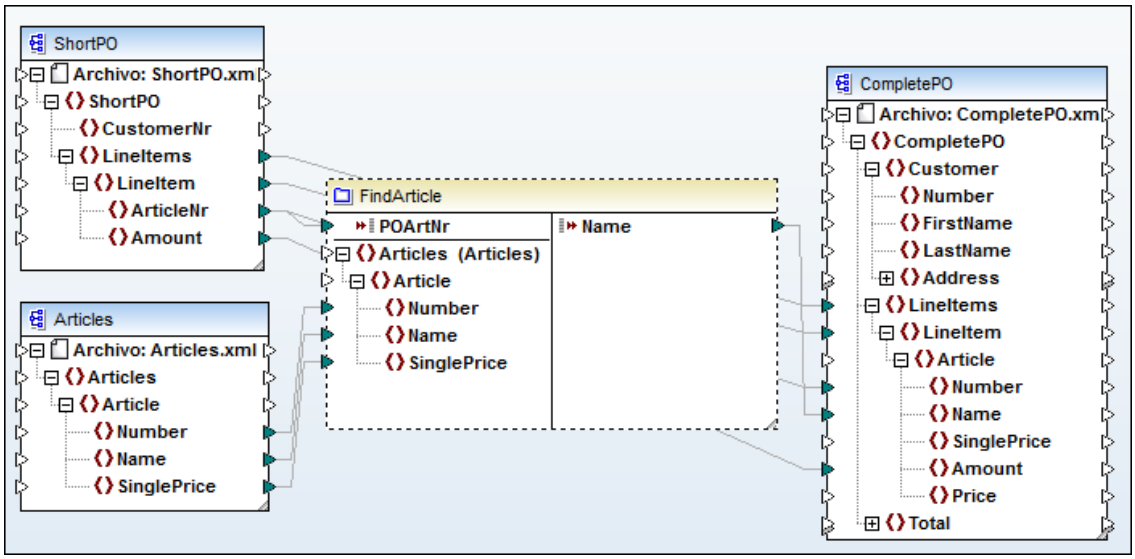
Tenga en cuenta que la función definida por el usuario **`FindArticle`** tiene dos partes.

La parte de la izquierda contiene los parámetros de entrada:

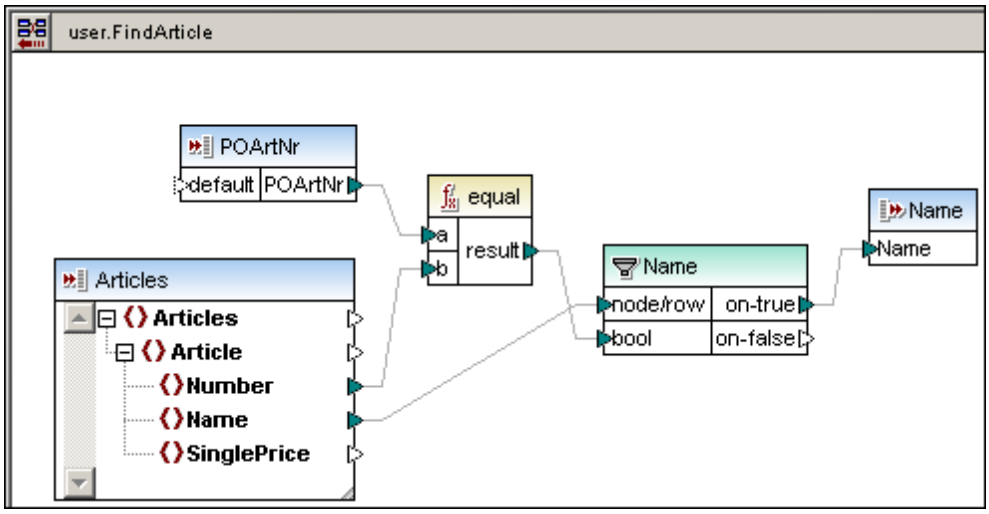
- un parámetro de entrada simple llamado **`POArtNr`**
- un parámetro de entrada complejo llamado **`Articles`**, cuyos nodos XML secundarios tienen asignaciones directas a otros componentes de la asignación

La parte de la derecha contiene:

- un parámetro de salida simple llamado **`Name`**.



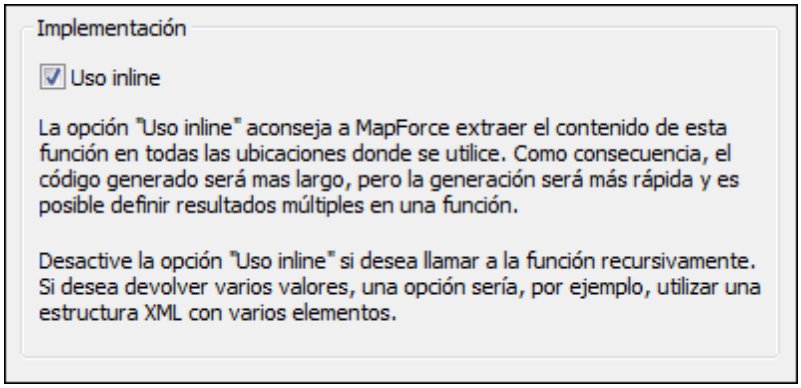
A continuación puede ver los componentes que constituyen la función definida por el usuario, con los dos componentes de entrada la izquierda y el de salida a la derecha.



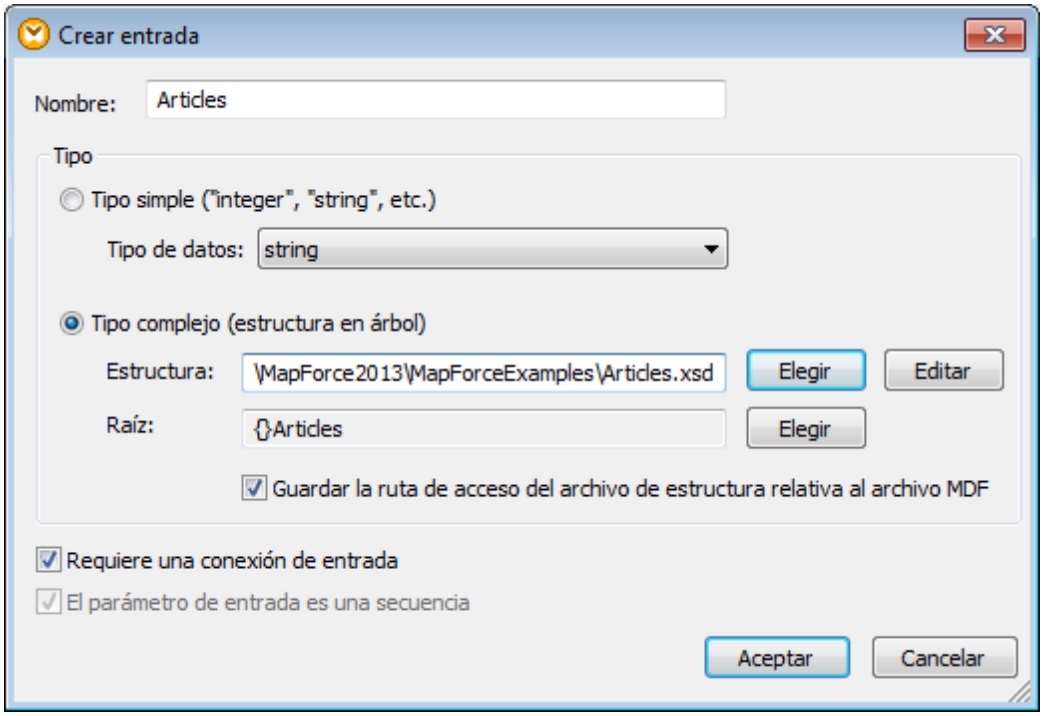
Definir componentes de entrada complejos

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.



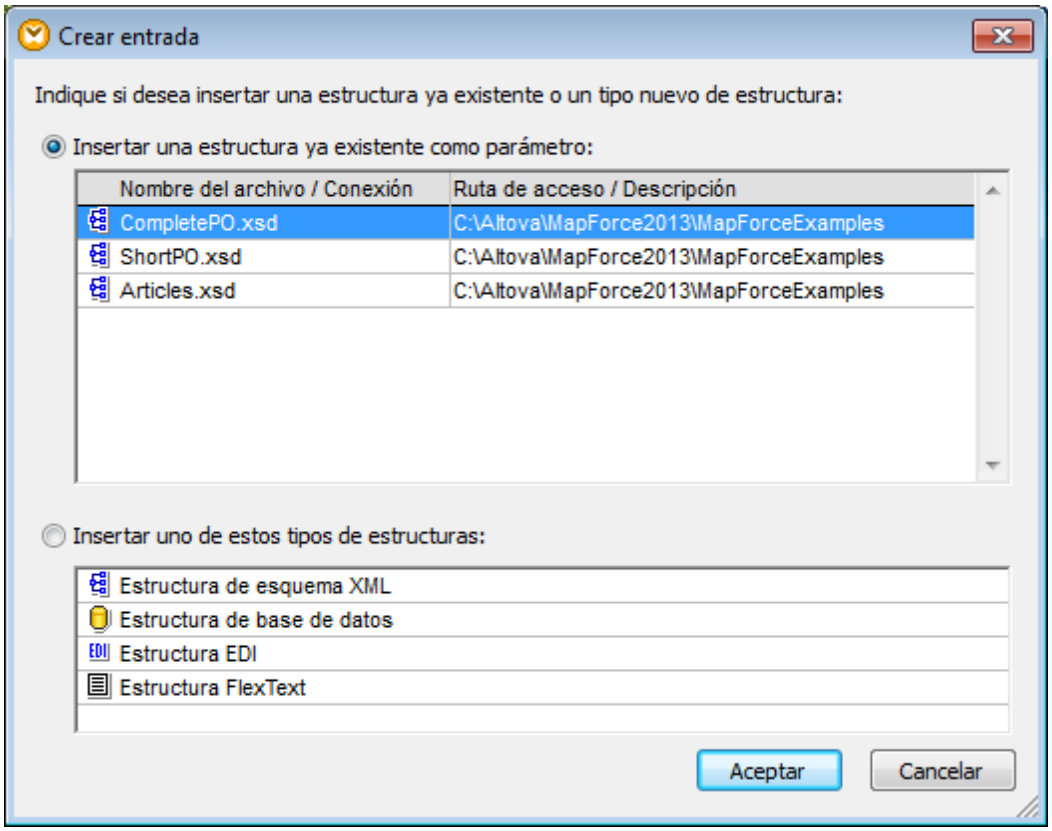
2. Haga clic en el icono **Insertar componente de entrada** de la barra de herramientas.
3. Escriba el nombre del componente de entrada en el campo *Nombre*.



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.

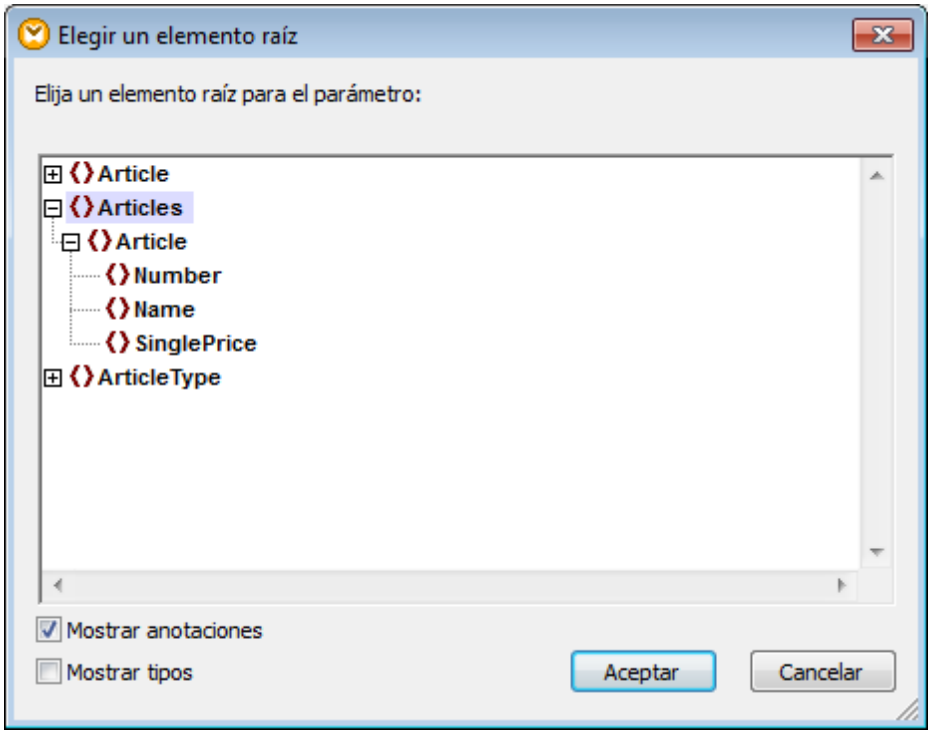
El primer cuadro de lista enumera los componentes ya **existentes** en la asignación (en este caso, tres esquemas). Es decir, la lista enumera todos los componentes que ya se insertaron en la asignación activa (p. ej. un esquema XML).

En el cuadro de lista inferior puede seleccionar una estructura de datos compleja **nueva** (es decir, un esquema XML).

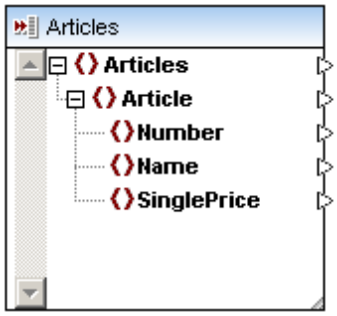


5. Seleccione el botón de opción *Insertar una estructura nueva*, después la opción **Estructura de esquema XML** y haga clic en **Aceptar**.
6. Seleccione **`Articles.xsd`** en el cuadro de diálogo Abrir.
7. Seleccione el elemento que desea usar como elemento raíz del componente (p. ej. **`Articles`**) y haga clic en **Aceptar**. Haga otra vez clic en **Aceptar** para cerrar el otro cuadro de diálogo.

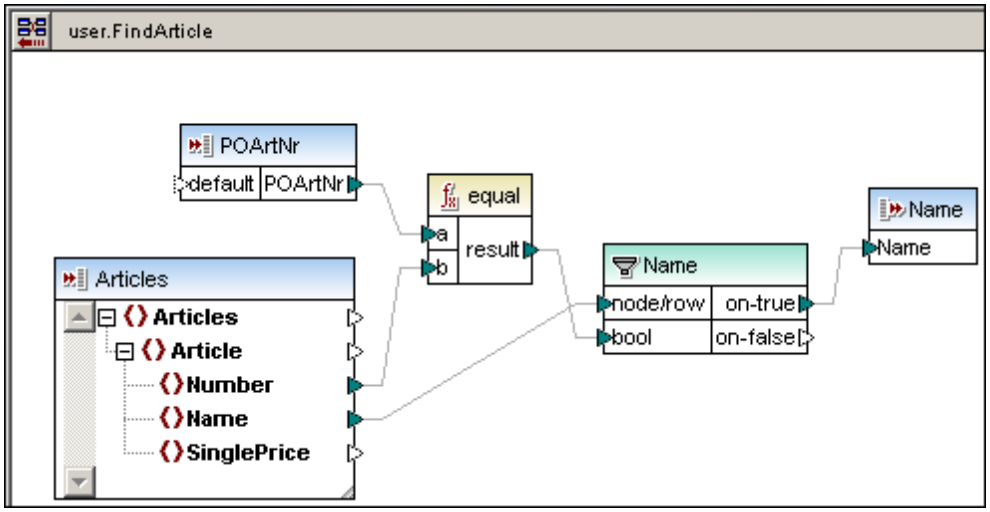
 © 2014 Altova GmbH



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.



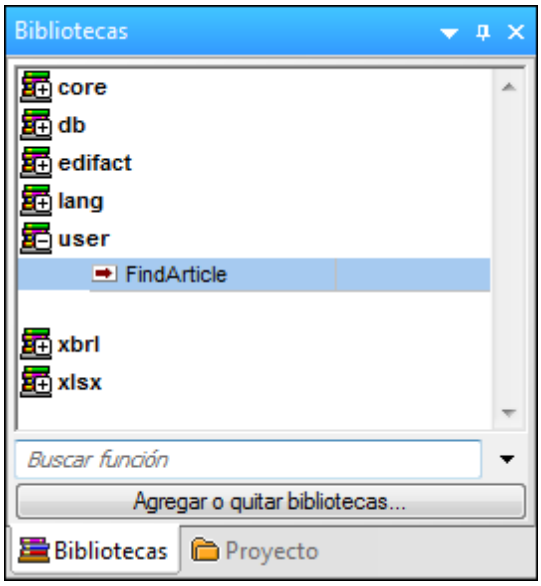
8. Inserte el resto de componentes que aparecen en la imagen siguiente, es decir, otro componente de entrada simple (**`POArtNr`**), un filtro, la función **`equal`** y un componente de salida (**`Name`**). Después conéctelos tal y como muestra la imagen.



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.

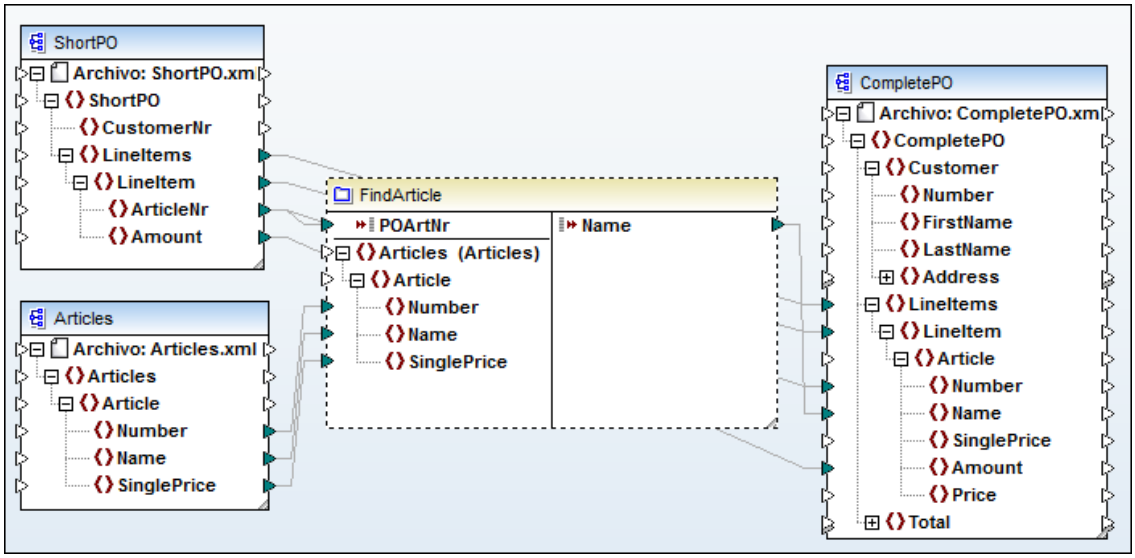
10. Haga clic en el icono **Volver** para volver a la asignación principal.
11. Inserte la función definida por el usuario nueva (arrastrándola desde la ventana Bibliotecas hasta el panel **Asignación**).



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>
```

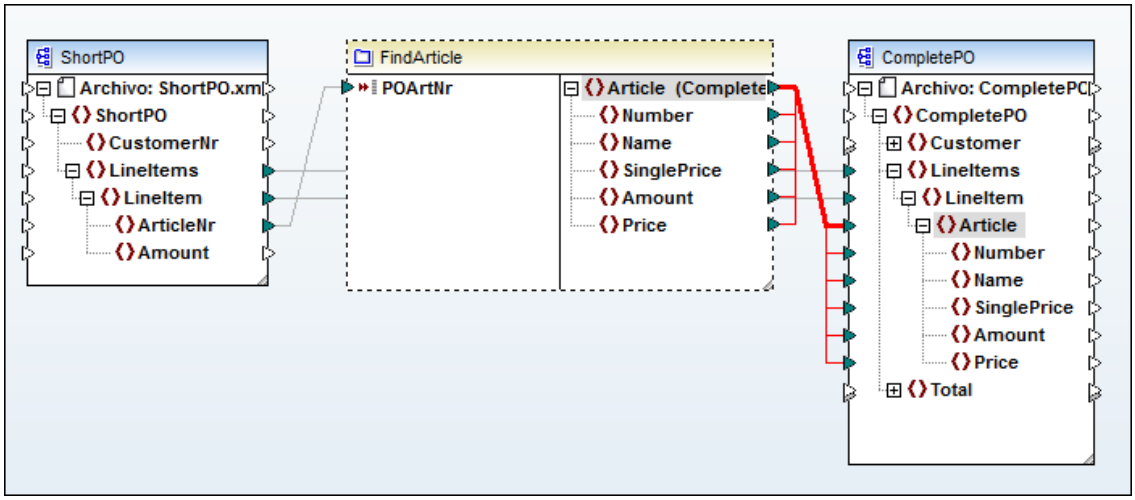
Nota: cuando cree conexiones de **copia total** entre un esquema y un parámetro de una función definida por el usuario, ambos componentes deben estar basados en el mismo esquema. Sin embargo, no es necesario que ambos tengan el mismo elemento raíz.

14.1.5 Función definida por el usuario compleja: con un nodo XML como salida

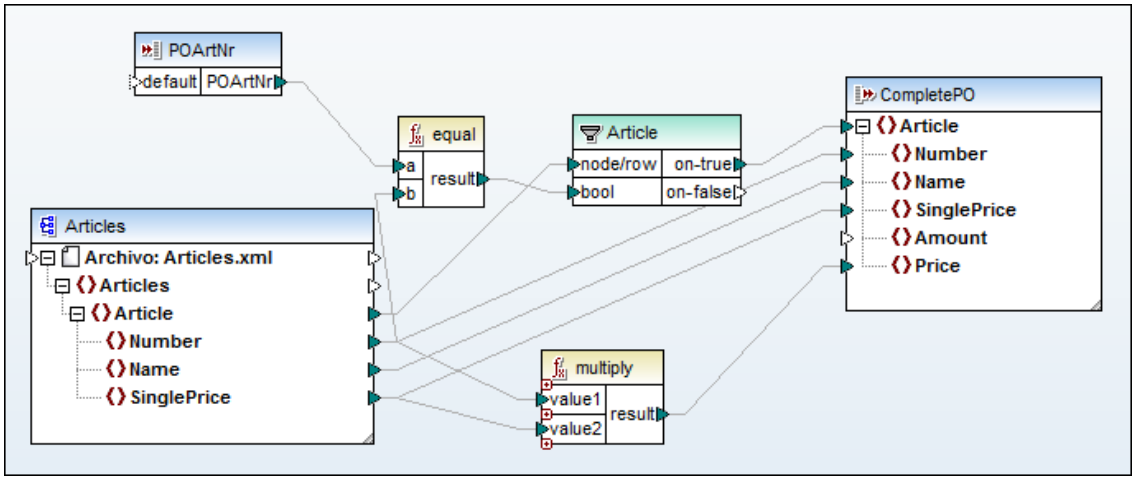
Este apartado utiliza el archivo de ejemplo `lookup-udf-out.mfd`, disponible en la carpeta `...\MapForceExamples`. A continuación explicamos cómo se define una función definida por el usuario de tipo inline con un componente de salida complejo.

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.

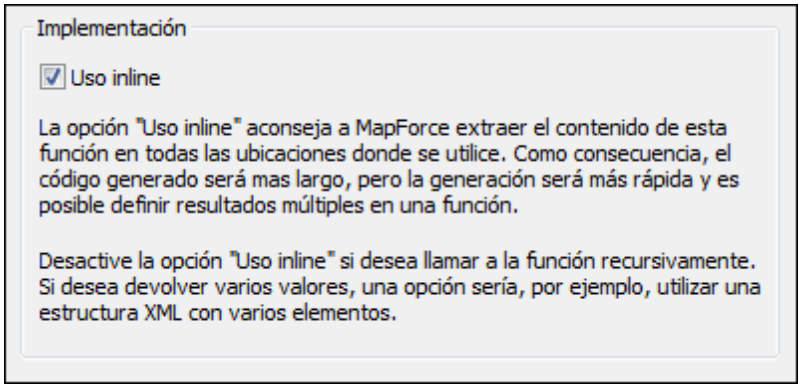


Definir componentes de salida complejos

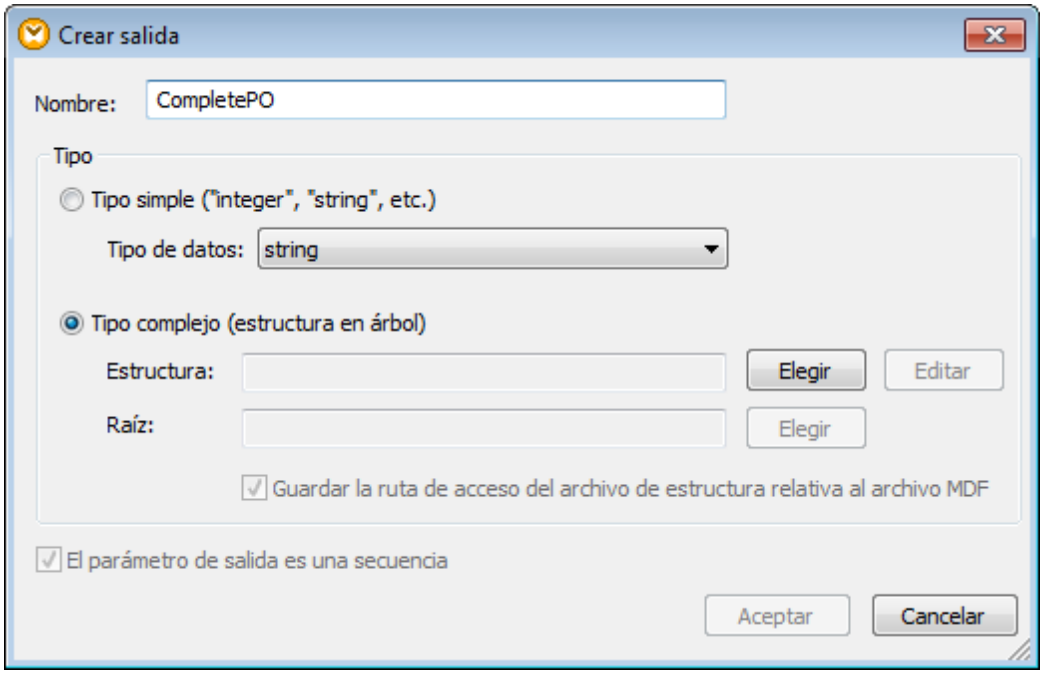
Para definir componentes de salida complejos:

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



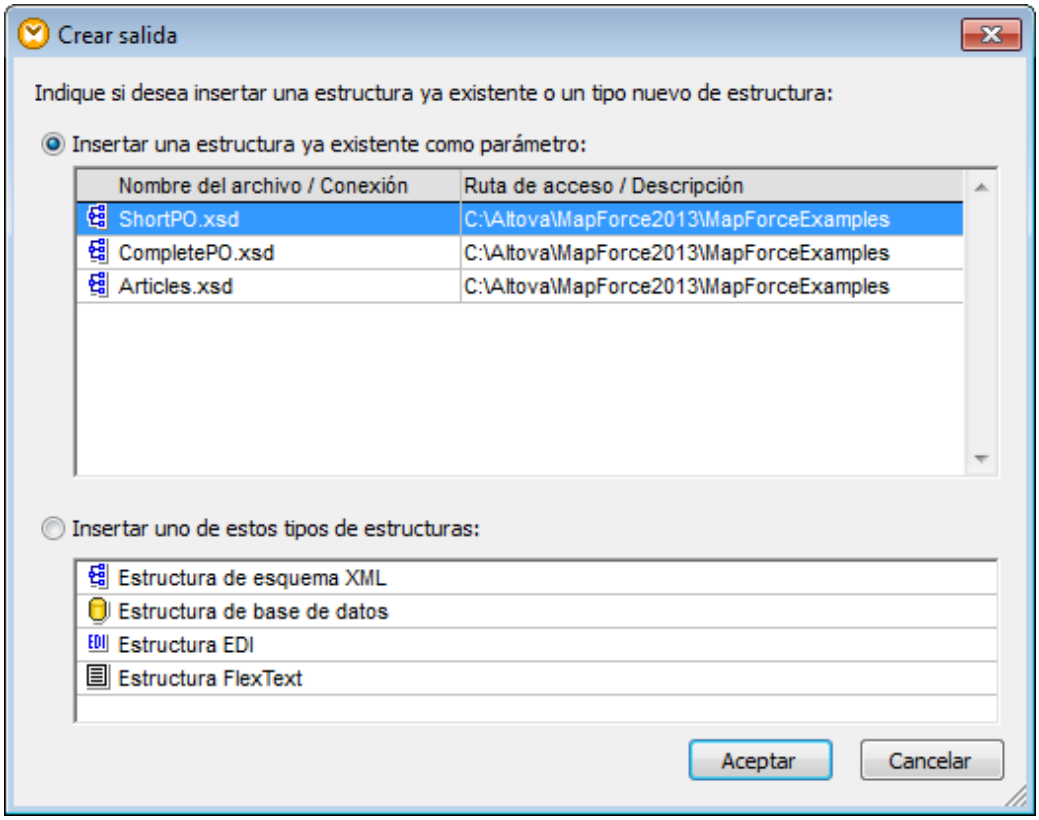
2. Haga clic en el icono **Insertar componente de salida** de la barra de herramientas. En el cuadro de diálogo escriba el nombre del componente de entrada en el campo *Nombre* (p. ej. CompletePO).



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.

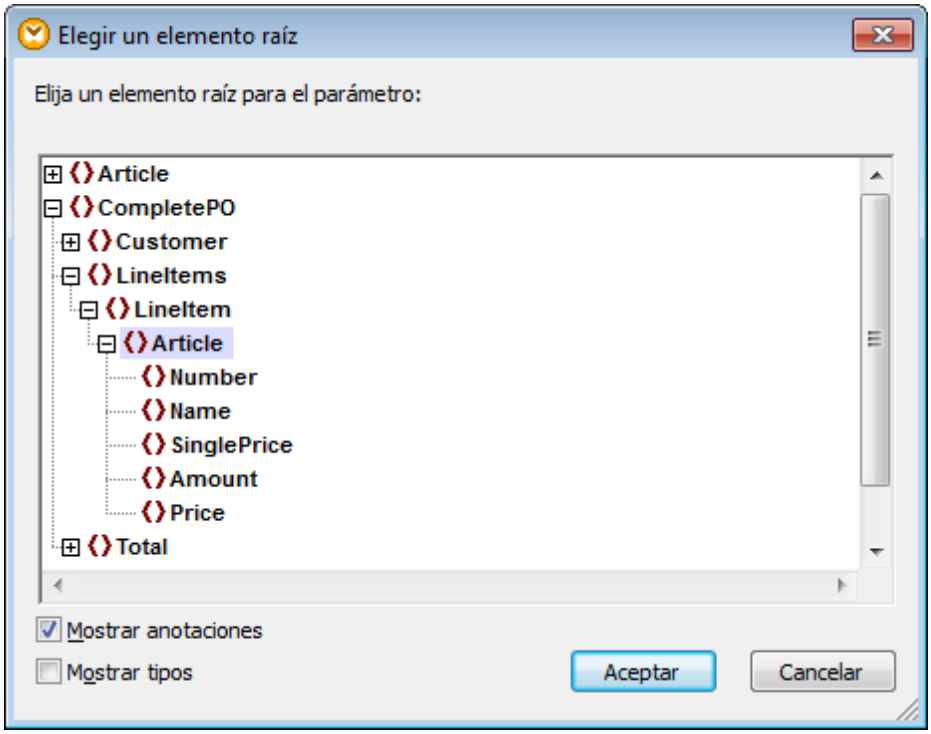
   El primer cuadro de lista enumera los componentes ya **existentes** en la asignación (en este caso, tres esquemas). Es decir, la lista enumera todos los componentes que ya se insertaron en la asignación activa (p. ej. un esquema XML).

   En el cuadro de lista inferior puede seleccionar una estructura de datos compleja **nueva** (es decir, un esquema XML).

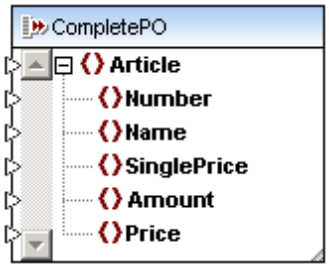


4. Seleccione el botón de opción *Insertar una estructura nueva*, seleccione la opción **Estructura de esquema XML** y haga clic en **Aceptar**.
5. Seleccione el esquema `CompletePO.xsd` en el cuadro de diálogo Abrir.
6. Haga clic en el elemento que desea usar como elemento raíz del componente (p. ej. **`Article`**) y haga clic en **Aceptar**. Haga otra vez clic en **Aceptar** para cerrar el otro cuadro de diálogo.

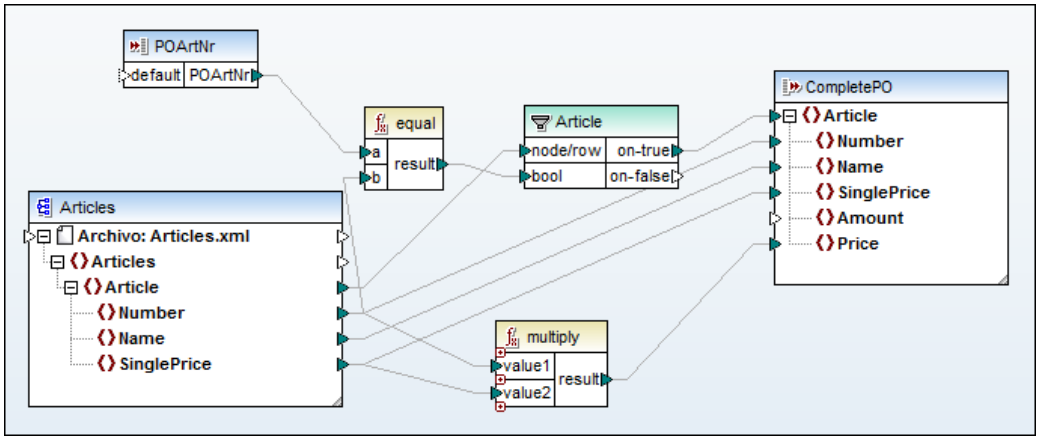
 © 2014 Altova GmbH



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.



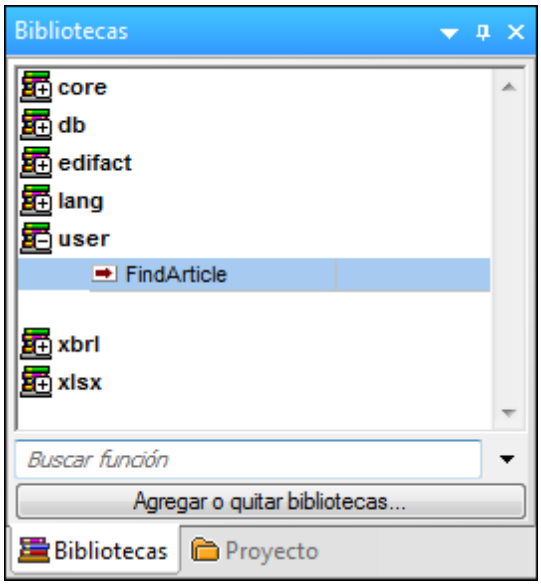
7. Inserte el esquema XML **Articles** en la función definida por el usuario y asigne el archivo `Articles.xml` como XML de instancia.
8. Inserte el resto de componentes que aparecen en la imagen siguiente, es decir: el componente de entrada simple **`POArtNr`**, un filtro, una función **`equal`** y 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**.

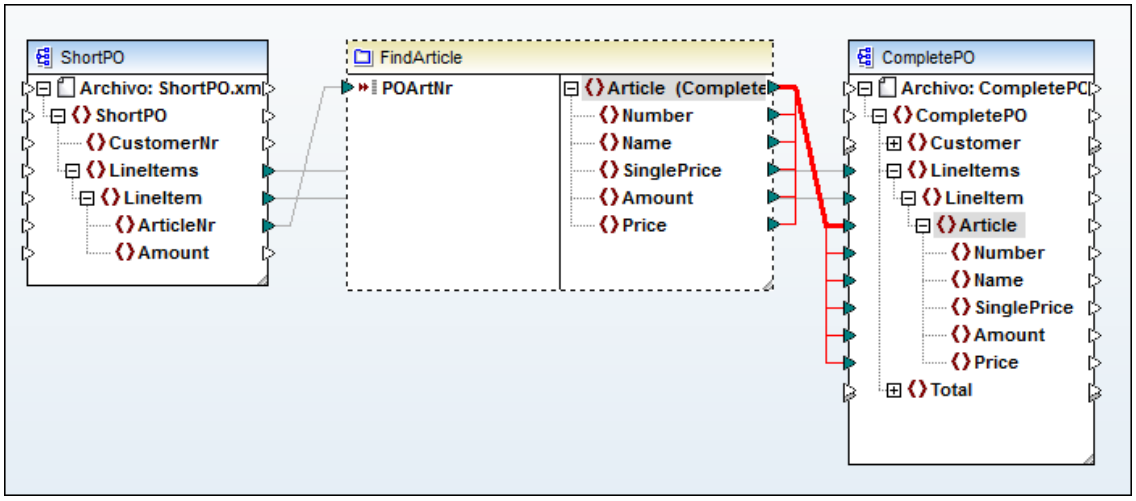
9. Haga clic en el icono **Volver** para volver a la asignación principal.
10. Inserte la función definida por el usuario nueva (arrastrándola desde la ventana Bibliotecas hasta el panel **Asignación**).



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:

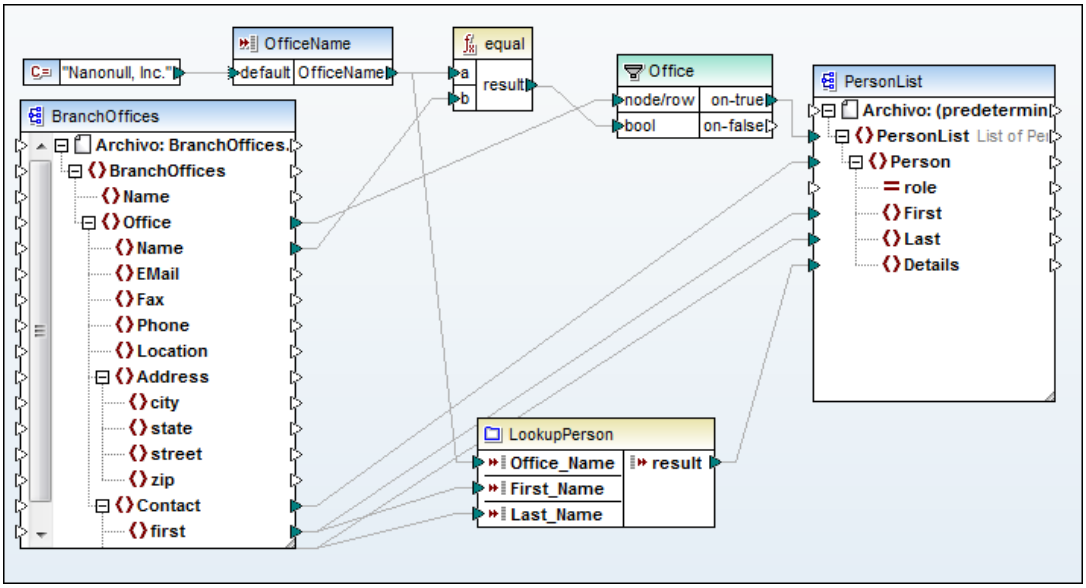
- un parámetro de salida complejo llamado **`Article (CompletePO)`** con nodos XML secundarios, que asigna los elementos filtrados (los que tienen el mismo número de artículo) al componente de destino **CompletePO**.

```
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>
```

14.1.6 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.

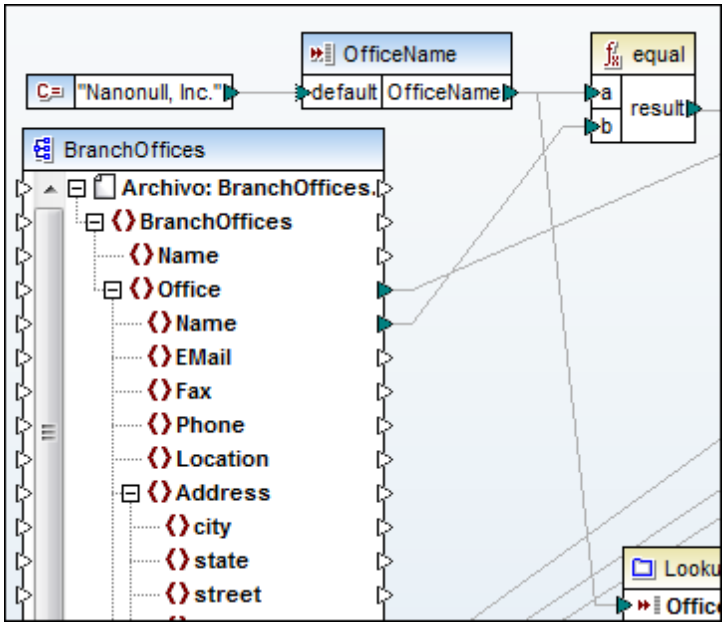


 © 2014 Altova GmbH

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**.

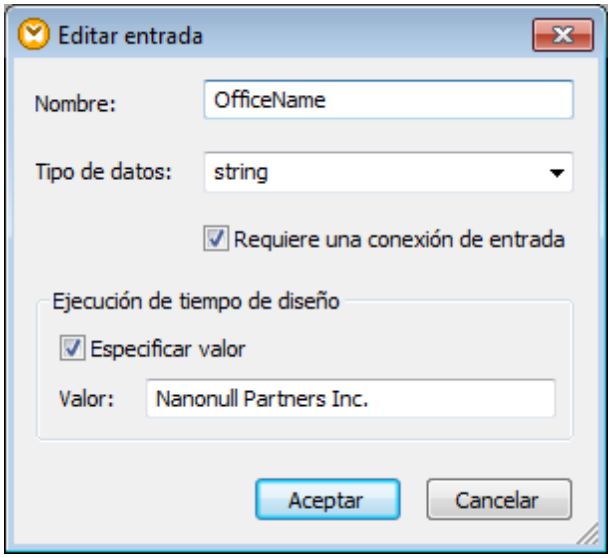


Para definir el valor de entrada:

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.

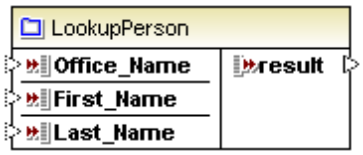
   Recuerde que los datos introducidos en este cuadro de diálogo se usan solamente en el modo de ***vista previa*** (es decir, al abrir el panel **Resultados**). Si no introduce un valor o desactiva la casilla *Especificar un valor*, se utilizan los datos asignados al icono de entrada **`default`**.

   Para más información consulte el apartado Línea de comandos: definir parámetros de entrada.



© 2014 Altova GmbH

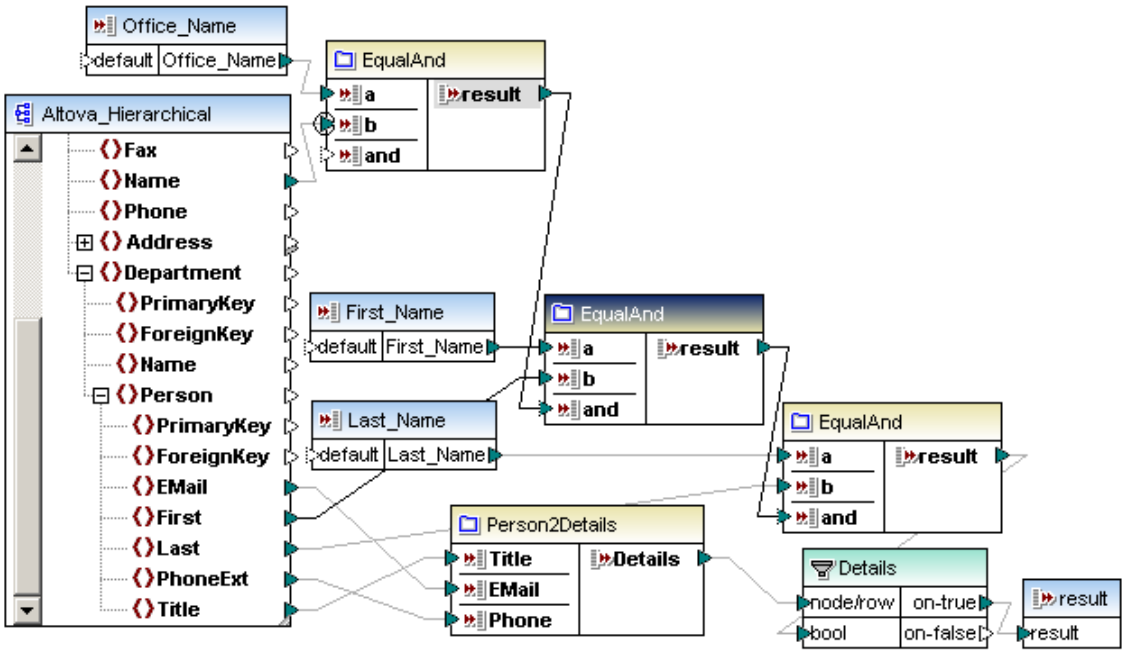
Componente LookupPerson



Al hacer doble clic en esta función definida por el usuario aparecen los componentes que la constituyen. Lo que hace este componente es:

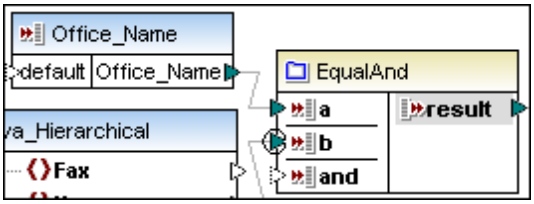
- **comparar** el contenido de **`Office`**, **`First`** y **`Last`** de `BranchOffices.xml` con los mismos campos del archivo `Altova_Hierarchical.xml`, usando los componentes de **entrada** y la función definida por el usuario **`EqualAnd`**.
- **reúne** los elementos **`Email`**, **`PhoneExt`** y **`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.



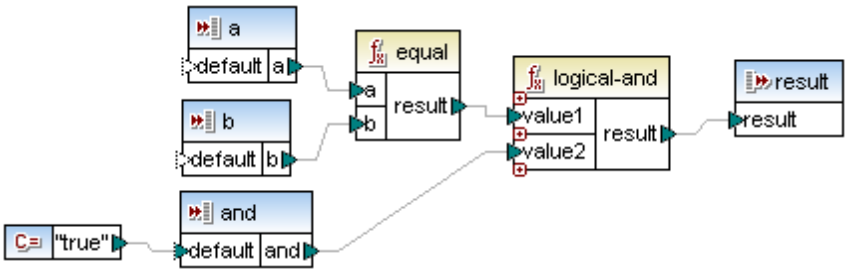
- Los tres componentes de entrada (**`Office_Name`**, **`First_Name`**, **`Last_Name`**) reciben sus datos del archivo XML `BranchOffices.xml`.
- El componente **`EqualAnd`** compara dos valores y ofrece un valor de comparación **opcional**, así como un valor predeterminado.
- **`Person2Details`** combina los tres campos de datos de **`Person`** y los pasa al componente de salida **`result`** a través de un filtro.

Componente EqualAnd



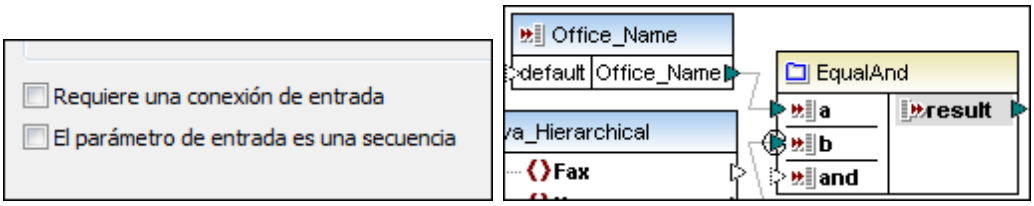
Al hacer doble clic en este componente definido por el usuario aparecen los componentes que lo constituyen. Lo que hace este componente es:

- 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`**.



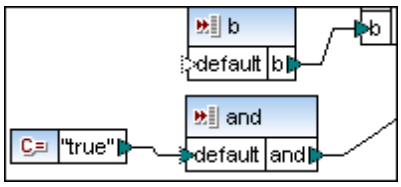
Parámetros opcionales

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`**).



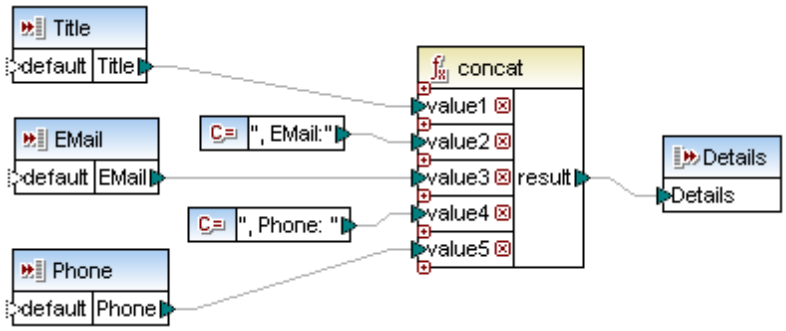
 © 2014 Altova GmbH

- 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

14.2 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 cuando el lenguaje de transformación elegido es XSLT.

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 la vista previa del código XSLT generado en el panel **XSLT**. Para más información consulte el anexo Información sobre motores XSLT y XQuery | XSLT 1.0.

Temas relacionados:

- Implementación del motor XSLT 1.0
- Implementación del motor XSLT 2.0

14.2.1 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.

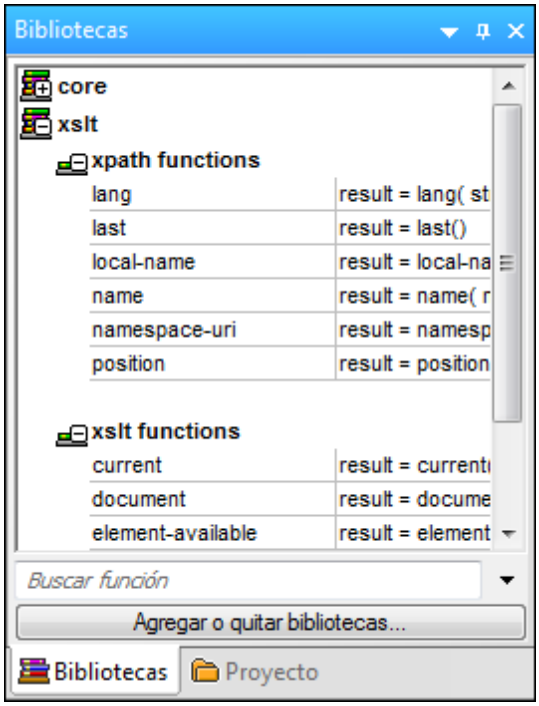
Para agregar una función XSLT personal:

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">
```

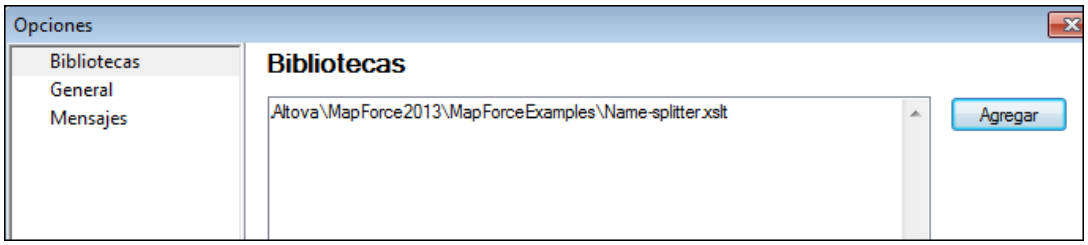
2. Haga clic en el botón **Agregar o quitar bibliotecas** y en el siguiente cuadro de diálogo haga clic en el botón **Agregar**.

 © 2014 Altova GmbH

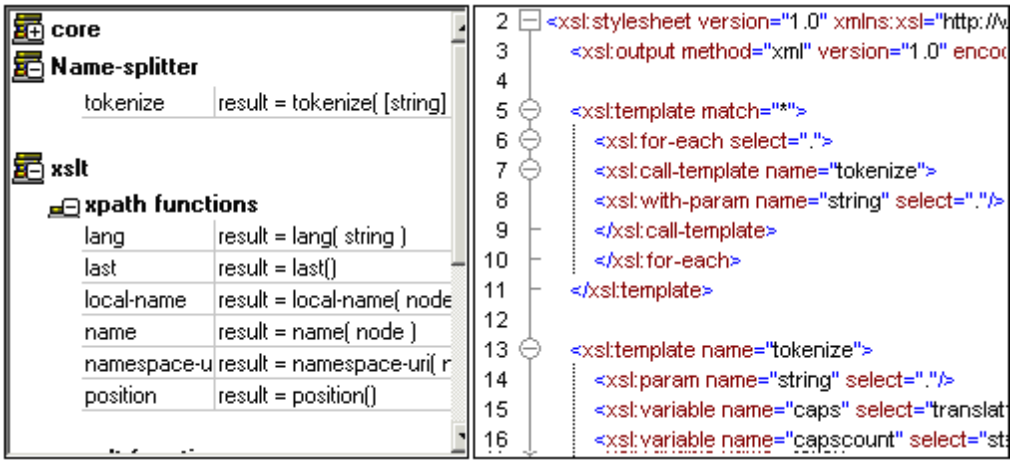


Con XSLT seleccionado

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.



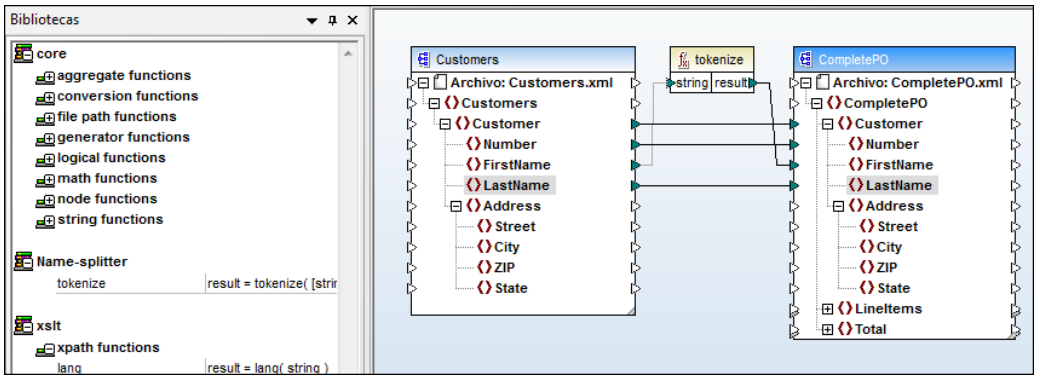
4. Haga clic en **Aceptar** para insertar la función nueva.



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.

14.2.2 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 ver cómo puede usar plantillas con nombre para sumar nodos consulte el apartado Funciones de agregado.

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.

Temas relacionados:

- Implementación del motor XSLT 2.0

 © 2014 Altova GmbH

14.2.3 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`:

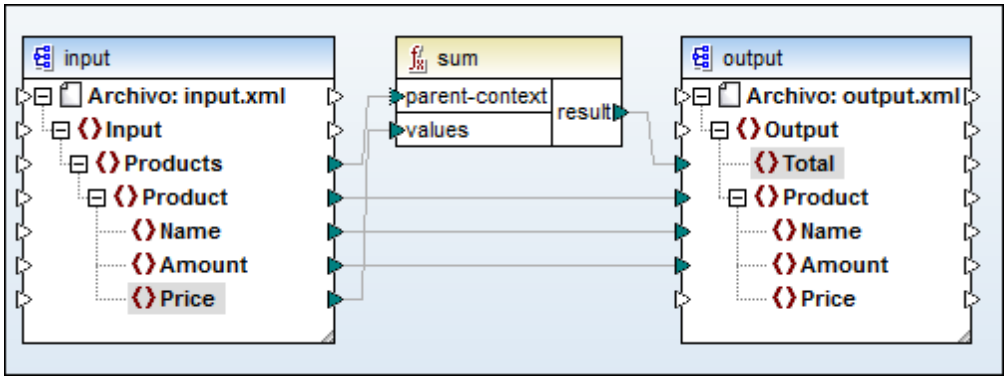
- `Summing-nodes.mfd`: el archivo de asignación
- `input.xml`: el archivo XML de entrada
- `input.xsd` y `output.xsd`: los esquemas de origen y destino respectivamente
- `Summing-nodes.xslt`: el archivo xslt que contiene una plantilla con nombre que suma los nodos

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).

```
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>
```

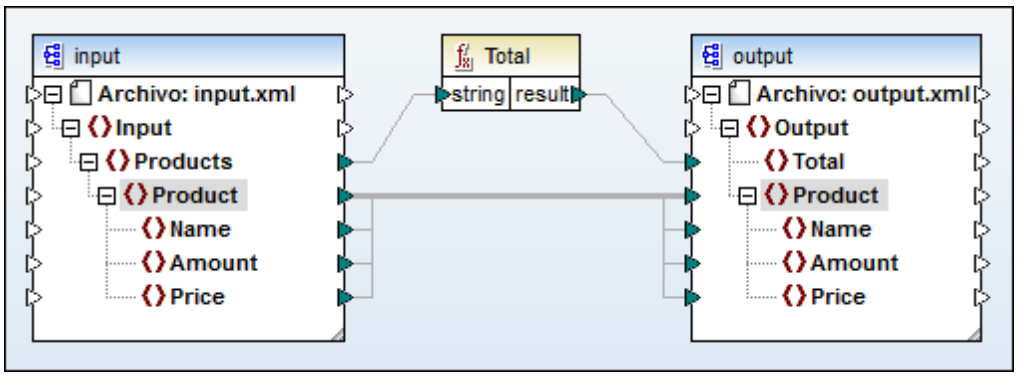
La siguiente imagen muestra la hoja de estilos XSLT que utiliza la plantilla con nombre **`Total`** y un parámetro único **`string`**. La plantilla recorre el archivo XML de entrada y suma todos los valores obtenidos por la expresión XPath `/Product/Price`.

```
<?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="
```

14.3 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.

MapForce ofrece estas bibliotecas:

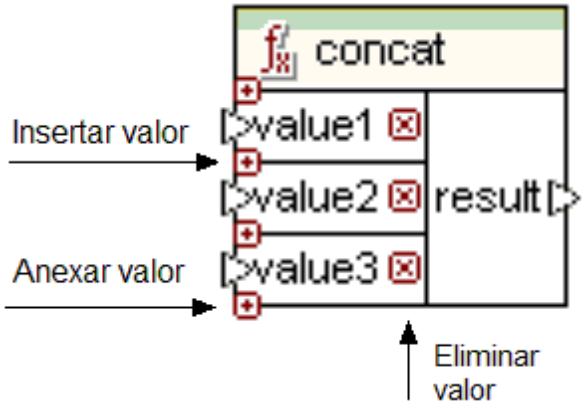
- **core**
- **xpath2**
- **xslt**

Funciones ampliables

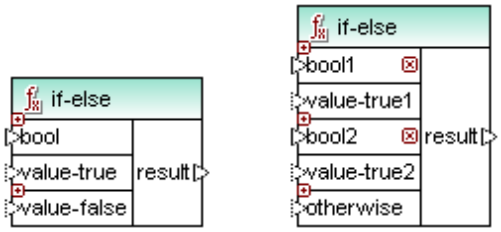
Hay varias funciones que se pueden ampliar: p. ej. las funciones **`concat`**, **`logical-and`**, **`logical-or`** e **`IF-ELSE`**. Esto significa que en este tipo de funciones puede insertar/anexar/eliminar parámetros dependiendo de sus requisitos.

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

14.3.1 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)

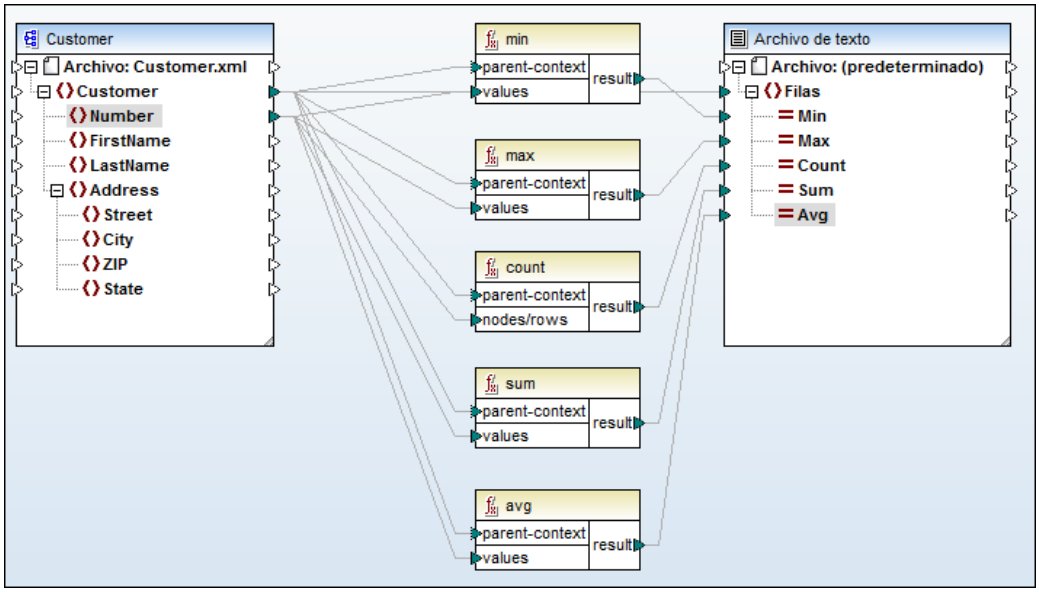
Las funciones de agregado de la biblioteca **core | aggregate** realizan operaciones en un conjunto o secuencia de valores de entrada. Los datos de entrada para las funciones **min**, **max**, **sum** y **avg** se convierten al tipo de datos `decimal` para poder procesarlos.

- 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.

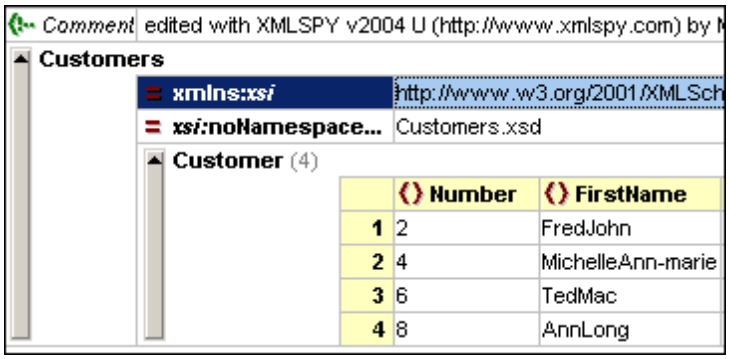
La asignación que aparece a continuación es la del archivo `Aggregates.mfd`, guardado en la carpeta `...\Tutorial` y explica cómo utilizar este tipo de funciones.

Las funciones de agregado tienen dos entradas:

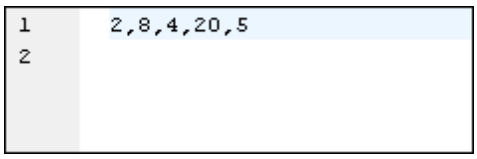
- la entrada **values** (**nodes/rows**) se conecta al elemento del componente de origen que suministra los datos, en este caso el elemento **Number**.
- la entrada **parent-context** se conecta al elemento en el que se desea iterar (es decir, el contexto. En este caso se quiere iterar en todos los elementos **Customer**). Este parámetro es opcional.



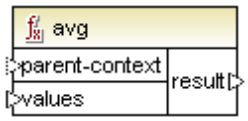
La instancia de entrada, en este caso, es un archivo XML que contiene estos datos:



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:



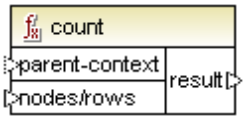
min=2, max=8, count=4, sum=20 y avg=5.



avg

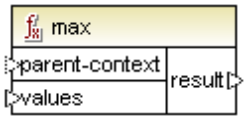
Devuelve el valor promedio de todos los valores de la secuencia de entrada. El promedio de un conjunto de valores vacío es un conjunto vacío. Esta función no está disponible en XSLT1.

 © 2014 Altova GmbH



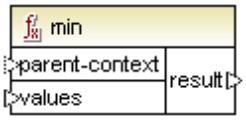
count

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.



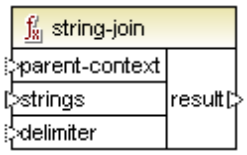
max

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.

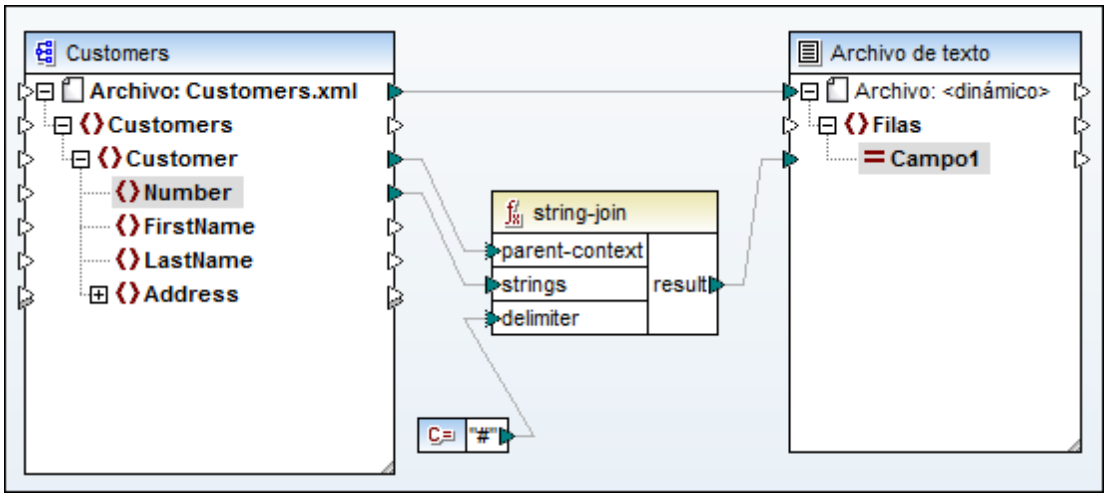


string-join

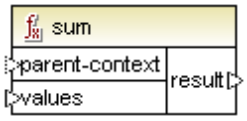
Enlaza todos los valores de la secuencia de entrada y forma una cadena delimitada por la cadena suministrada como delimitador al parámetro **`delimiter`**. Cuando se aplica a un conjunto vacío el resultado es una cadena vacía. Esta función no está disponible en XSLT1.

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`).

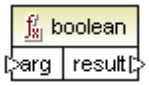


sum

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.

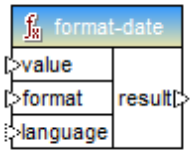
conversion (funciones de conversión)

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.



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

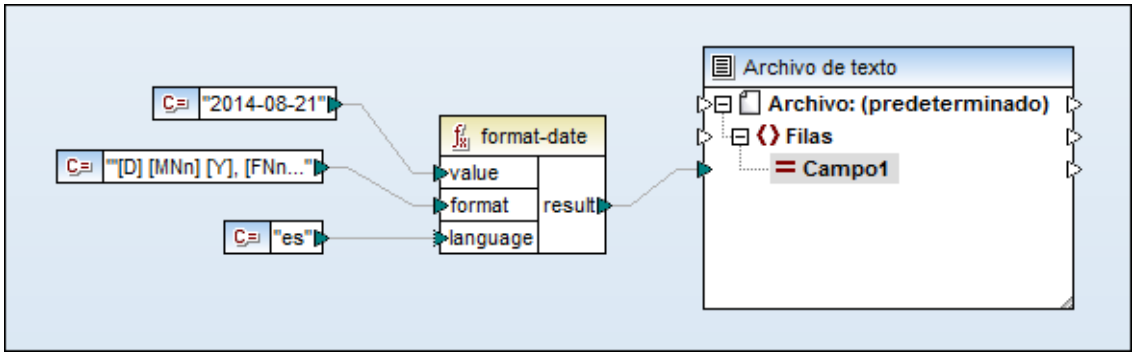
Convierte un valor de entrada `xs:date` en un valor de cadena y le aplica el formato indicado por las opciones especificadas.

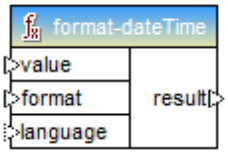
Argumento	Descripción
`value`	Fecha a la que se le debe aplicar formato.
`format`	Cadena de formato que identifica el formato que se debe aplicar a la fecha. Este argumento se usa igual que el argumento `format` de la función **`format-dateTime`**.
`language`	Argumento opcional que devuelve el nombre del mes y el día de la semana en el idioma seleccionado. Valores válidos: **en (predeterminado)** inglés **es** español **de** alemán

 © 2014 Altova GmbH

	ja	japonés

En la imagen siguiente, por ejemplo, el resultado es "21 August 2014, Thursday". Para traducir este valor al idioma español, defina el valor `es` para el argumento `language`.





format-dateTime

Convierte un valor `dateTime` en un valor de cadena.

Argumento	Descripción
`value`	el valor `dateTime` al que se debe aplicar formato.
`format`	Cadena de formato que identifica el formato que se debe aplicar al valor `dateTime`.
`language`	Argumento opcional que devuelve el nombre del mes y el día de la semana en el idioma seleccionado. Valores válidos: **en (predeterminado)** inglés **es** español **de** alemán **ja** japonés

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

 © 2014 Altova GmbH

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)**

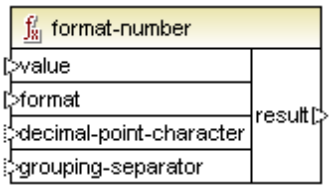
Nota: los modificadores N, n y Nn solamente son compatibles con estos componentes: M, d, D.

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
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:

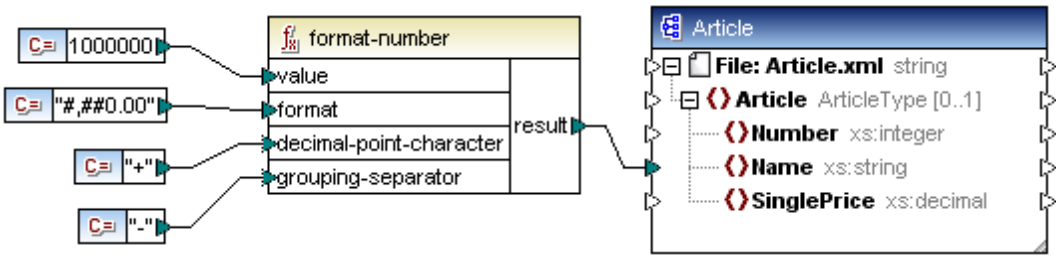
```
format := subformat (;subformat)?
  subformat := (prefix)? integer (.fraction)? (suffix)?
  prefix := cualquier carácter excepto un carácter especial
  suffix := cualquier carácter excepto un carácter especial
  integer := (#)* (0)* ( permitiendo aparecer a ',')
  fraction := (0)* (#)* ( permitiendo 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	**Predeterminado**	**Descripción**
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.

 © 2014 Altova GmbH



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>
```

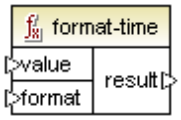
Redondeo

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.

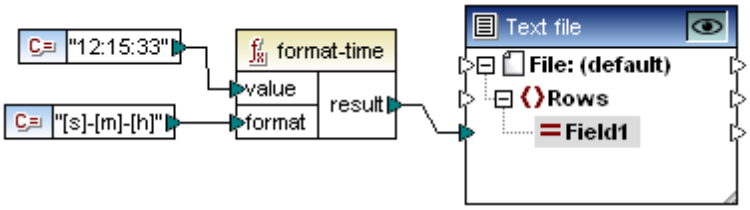
Número	formato de cadena	Resultado
1234.5	#,##0.00	1,234.50
123.456	#,##0.00	123.46
1000000	#,##0.00	1,000,000.00
-59	#,##0.00	-59.00
1234	###0.0###	1234.0
1234.5	###0.0###	1234.5
.00025	###0.0###	0.0003
.00035	###0.0###	0.0004
0.25	#00%	25%
0.736	#00%	74%
1	#00%	100%
-42	#00%	-4200%
-3.12	#.00;(#.00)	(3.12)
-3.12	#.00;#.00CR	3.12CR



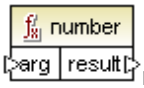
format-time

Convierte un valor de entrada `xs:time` en un valor de cadena

Por ejemplo

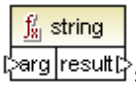


Resultado: 33-15-12



number

Convierte una cadena de entrada en un número. También convierte una entrada booleana en un número.



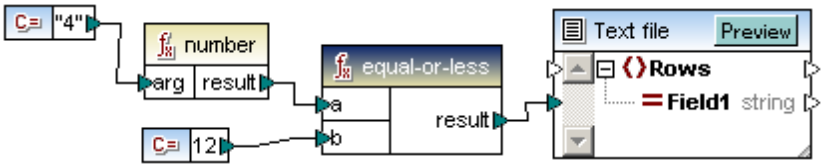
string

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.



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

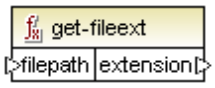
 © 2014 Altova GmbH

resultado de la comparación es "true".

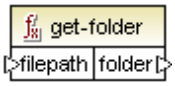
Recuerde que si la función **number** no se utiliza (si 4 no se conecta directamente al parámetro **a**), se realizaría una comparación de cadenas y el resultado sería "false".

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`"

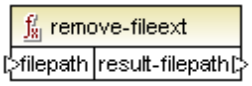


get-folder
Devuelve el nombre de la carpeta de la ruta de archivo, incluida la barra diagonal final o la barra diagonal inversa final.
P. ej. `c:/data/Sample.mfd` devuelve `c:/data/`

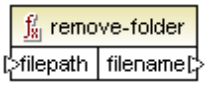
main-mfd-filepath
Devuelve la ruta completa del archivo `mfd` que contiene la asignación principal. Si el archivo `mfd` está sin guardar, la función devuelve una cadena vacía.

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.

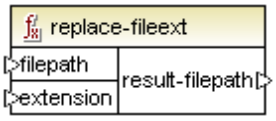


remove-fileext
Borra la extensión de la ruta de archivo, incluido el carácter "."
P. ej. `c:/data/Sample.mfd` devuelve `c:/data/Sample`.



remove-folder

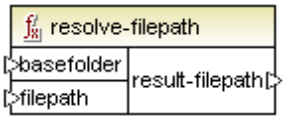
Borra el directorio de la ruta de acceso, incluida la barra diagonal final o la barra diagonal inversa final.
P. ej. `c:/data/Sample.mfd` devuelve `Sample.mfd`.



replace-fileext

Reemplaza la extensión de la ruta de acceso dada por el parámetro **`filepath`** con la extensión dada por el parámetro **`extension`**.

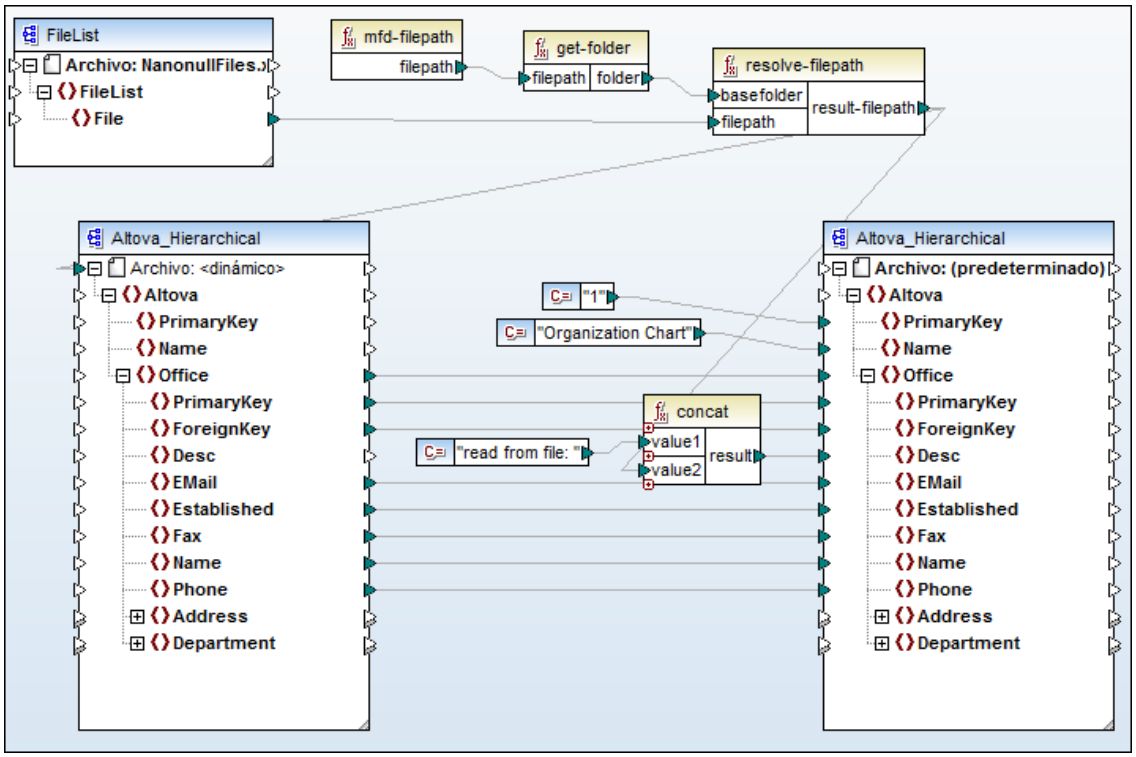
P. ej. si el parámetro **`filepath`** es la ruta de acceso `c:/data/Sample.mfd` y `.mfp` es la extensión del parámetro **`extension`**, la función devuelve `c:/data/Sample.mfp`.



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`.

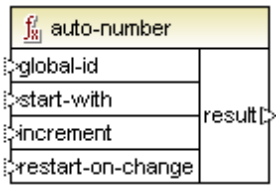


 © 2014 Altova GmbH

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**.



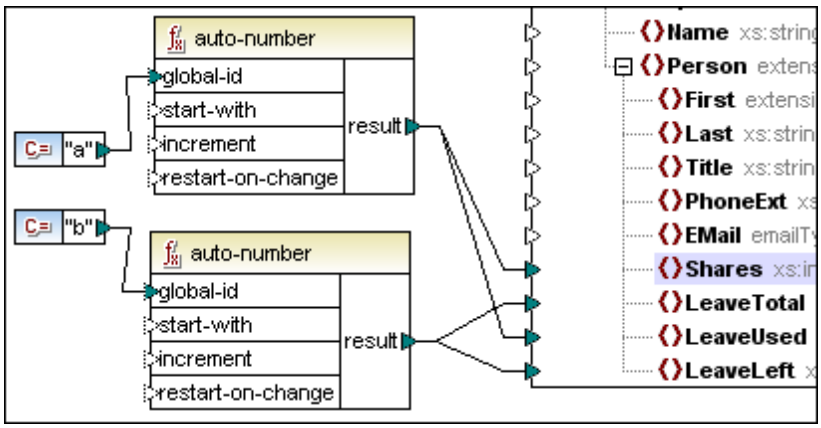
auto-number

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**).

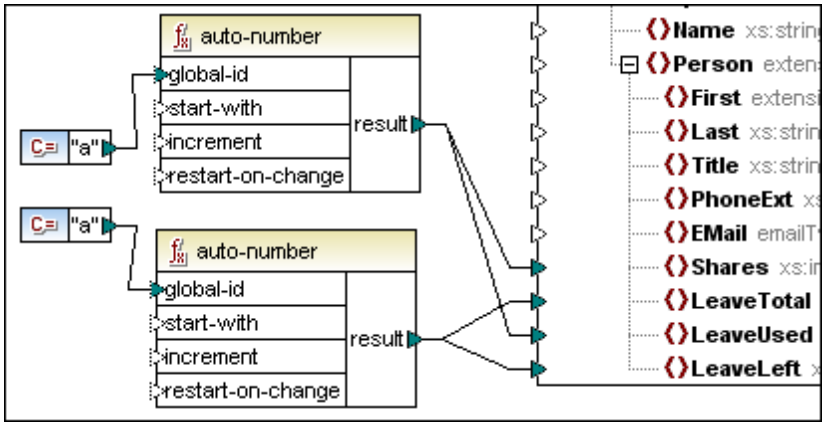
El resultado de la asignación es `1,1,2,2`. La primera función (con **global-id a**) aporta el primer 1 y la segunda (con **global-id b**) aporta el segundo 1.



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.



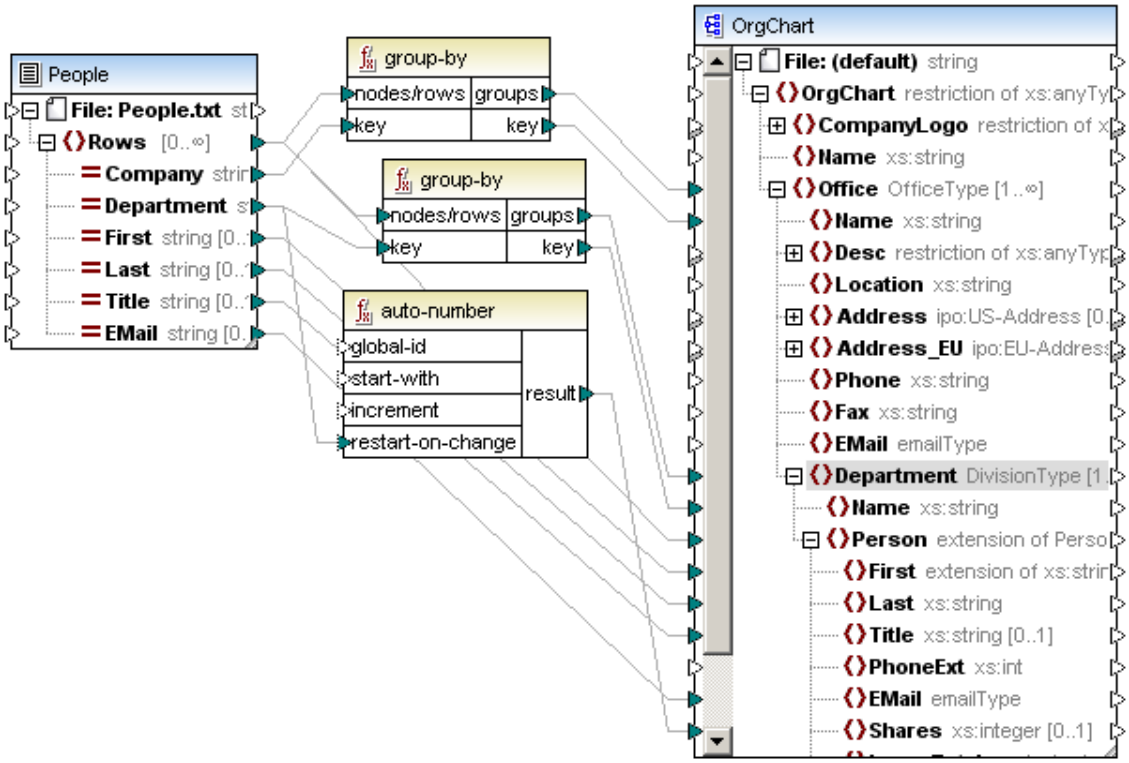
Parámetro `start-with`
El valor inicial utilizado para comenzar la secuencia de numeración automática. Valor predeterminado = 1.

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.

 © 2014 Altova GmbH



logical (funciones lógicas)

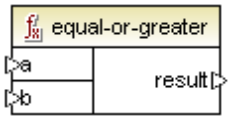
Las funciones lógicas de la biblioteca **core | logical** se usan por lo general para comparar datos de entrada y su resultado son los valores booleanos `true` o `false`. Estas funciones se suelen utilizar para probar datos antes de pasarlos, por medio de un filtro, a un subconjunto del componente de 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`**

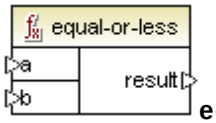
equal

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

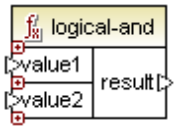
El resultado es **`true`** si **`a`** es igual o menor que **`b`**. De lo contrario, el resultado es **`false`**.

greater

El resultado es **`true`** si **`a`** es mayor que **`b`**. De lo contrario, el resultado es **`false`**.

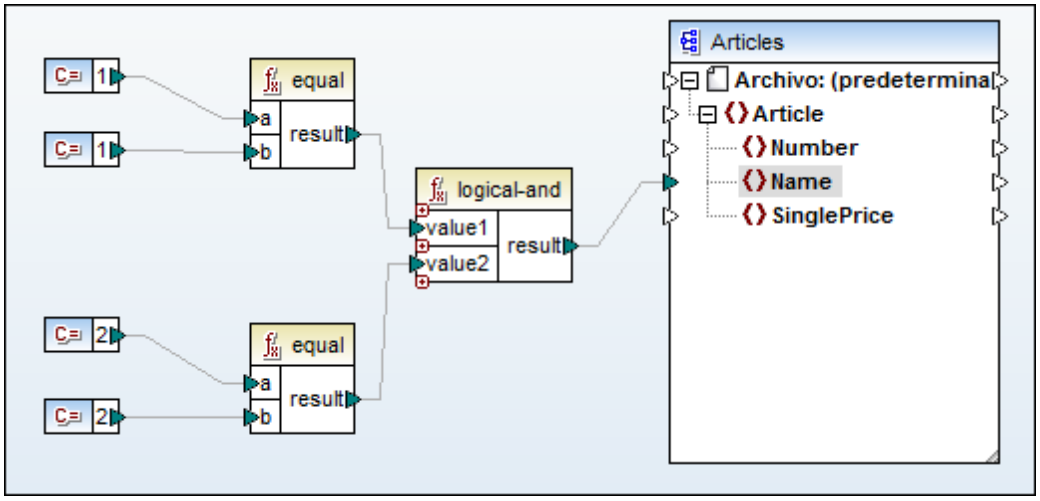
less

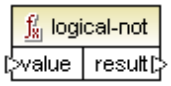
El resultado es **`true`** si **`a`** es menor que **`b`**. De lo contrario, el resultado es **`false`**.



logical-and

Si tanto **`value1`** como **`value2`** son **`true`**, el resultado es **`true`**. Si son diferentes, el resultado es **`false`**.





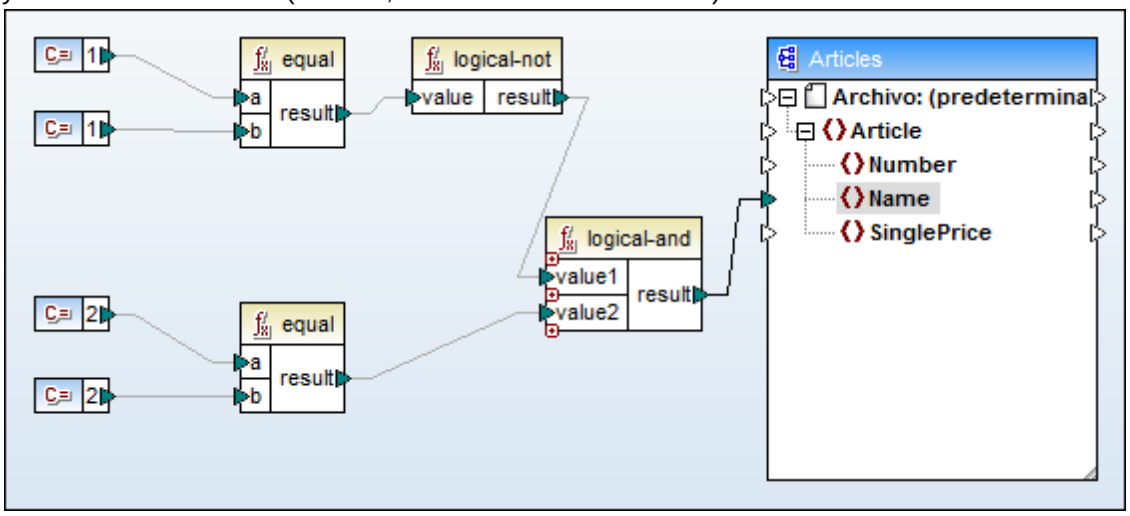
logical-not

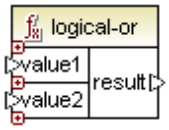
Invierte o le da la vuelta al estado lógico/resultado. Si el valor de entrada es **`true`**, el resultado de la función es **`false`**. Si el valor de entrada es **`false`**, el resultado de la función es **`true`**.

La función **`logical-not`** que aparece a continuación invierte el resultado de la función **`equal`**. La

 © 2014 Altova GmbH

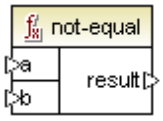
función **`logical-and`** ahora solamente devuelve el valor **`true`** si los valores booleanos de **`value1`** y **`value2`** son diferentes (es decir, **`true-false`** o **`false-true`**).





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

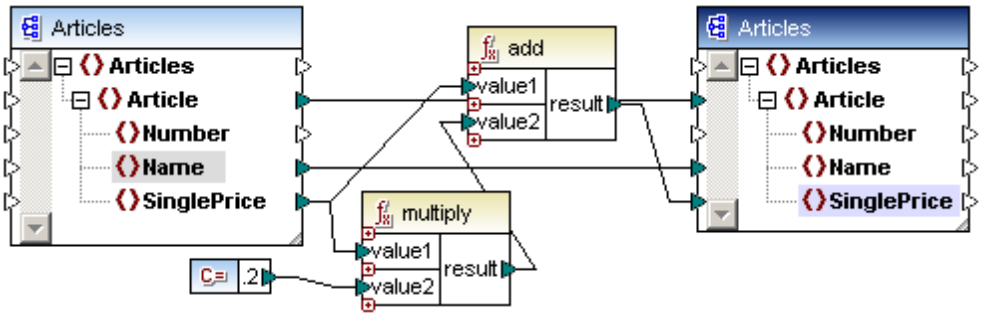
El resultado es **`true`** si **`a`** no es igual a **`b`**.

math (funciones matemáticas)

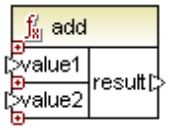
Las funciones matemáticas de la biblioteca **core | math** sirven para realizar operaciones matemáticas básicas con los datos. Tenga en cuenta que no se pueden usar para cálculos con datos `duration` ni `datetime`.

parámetros de entrada = **`value1 | value2`**
parámetro de salida = **`result`**

MapForce convierte automáticamente los valores de entrada en decimales para poder procesarlos.

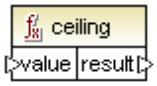


En la asignación de esta imagen, por ejemplo, se añade un 20% de impuestos sobre las ventas en los artículos asignados al componente de destino.



add

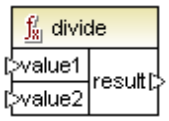
El resultado es el valor decimal resultante de sumar **`value1`** a **`value2`**.



ceiling

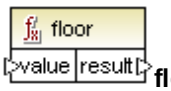
El resultado es el entero más pequeño que es mayor o igual que **`value`**, es decir, el valor entero mayor más próximo al valor decimal de entrada **`value`**.

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.

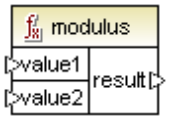


floor

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

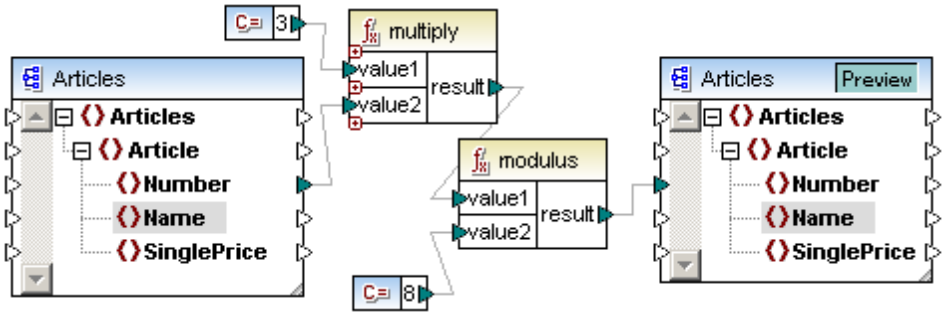


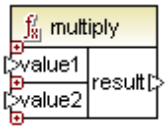
modulus

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.

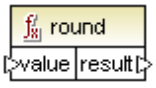
Cuando se aplica/utiliza la función **`modulus`** con **`value2`** = 8, los enteros restantes son 3, 6, 1 y 4.





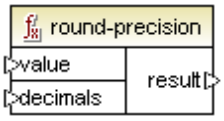
multiply

El resultado es el valor decimal resultante de multiplicar **`value1`** por **`value2`**.



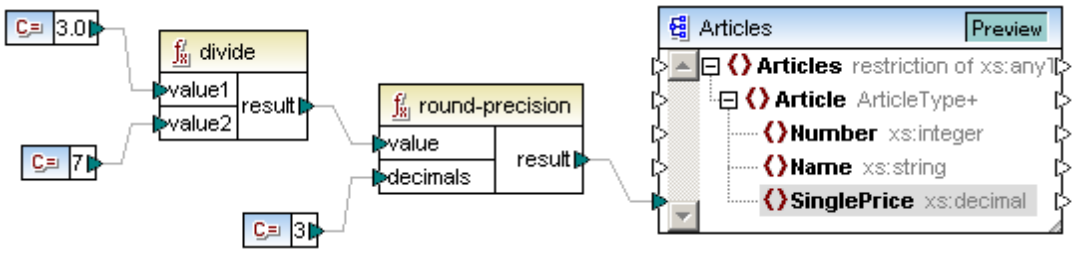
round

Devuelve el valor redondeado al entero más cercano. Cuando el valor esté justo entre dos enteros, se utilizará el algoritmo *Desempate la mitad alejándose del cero.* Por ejemplo, el valor `10.5` se redondearía hasta `11` y el valor `-10.5` se redondearía hasta `-10`.

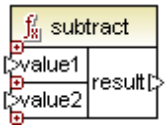


round-precision

El resultado es el valor decimal del número redondeado a las cifras decimales definidas por el parámetro **`decimals`**.



Por ejemplo, en la asignación anterior, el resultado es `0.429`.
Si quiere que este resultado aparezca correctamente en el archivo XML, debe estar asignado a un elemento de tipo `xs:decimal`.

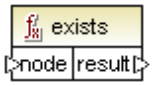


subtract

El resultado es el valor decimal resultante de restar **`value2`** a **`value1`**.

node (funciones de nodo)

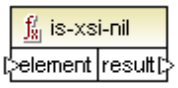
Las funciones de prueba de nodos de la biblioteca **core | node** sirven para comprobar si existen o no ciertos nodos en varios tipos de archivos de entrada (como XML, esquemas, texto, bases de datos, EDI, etc.) e incluso en los resultados de ciertas funciones. La función **`exists`** suele comprobar si hay una secuencia no vacía, es decir, si existe un nodo.



exists

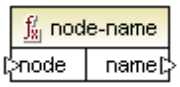
Devuelve el resultado **`true`** si el nodo existe. De lo contrario devuelve **`false`**.

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`.



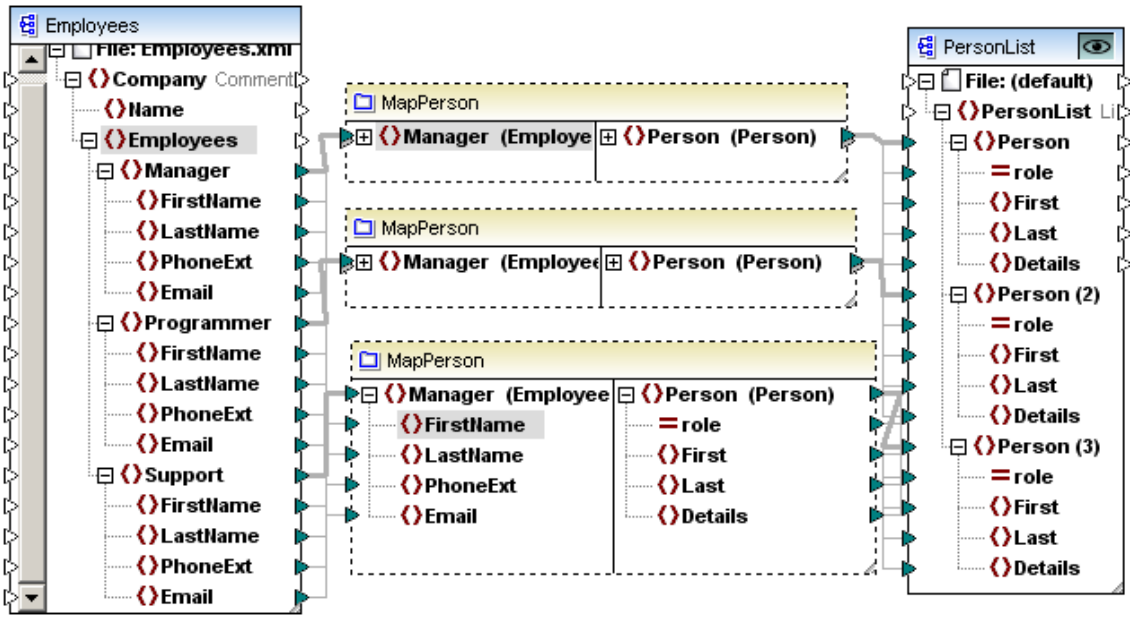
node-name

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.

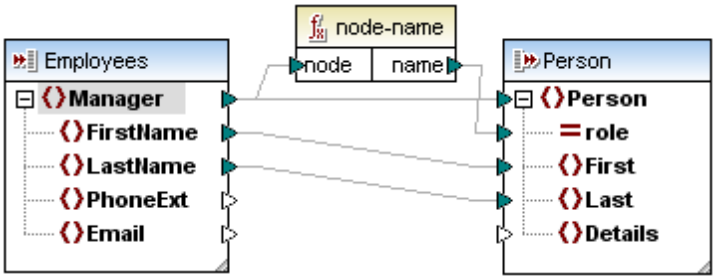
- No puede obtener un nombre de campos/tablas de BD.
- Esta función no es compatible con XBRL ni Excel.

 © 2014 Altova GmbH

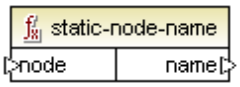
- No se puede obtener un nombre del nodo de entrada **`Archivo`**.
- Los nodos de servicio web se comportan como nodos XML, excepto que:
  - node-name no es compatible con `part`.
  - node-name no es compatible con el nodo raíz (de entrada o salida).



La función definida por el usuario **`MapPerson`** usa la función **`node-name`** para recuperar el nombre del nodo de entrada y colocarlo en el atributo `role`. El nodo raíz del esquema `Employees.xsd`, en la función definida por el usuario, es **`Manager`**.



`Manager` recibe sus datos de **fuera** de la función definida por el usuario y su contenido puede ser `Manager`, `Programmer` o `Support`. Estos son los datos que se pasan al atributo `role` del componente de destino **PersonList**.

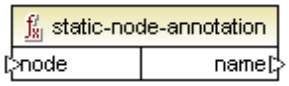


static-node-name

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

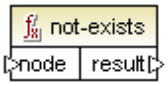
un texto adquirido del nodo conectado y, por tanto, está disponible en todos los lenguajes.



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.

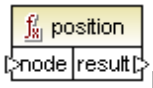
La conexión debe ser directa. No puede pasar por un filtro ni por una función definida por el usuario no inline. Se trata de una pseudo función, que en tiempo de generación se reemplaza con un texto adquirido del nodo conectado y, por tanto, está disponible en todos los lenguajes.



not-exists

Devuelve **`false`** si el nodo existe. De lo contrario, devuelve **`true`**.

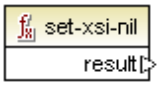
Consulte este apartado para ver un ejemplo de uso.



position

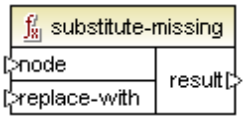
Devuelve la posición que tiene un nodo dentro de su secuencia.

Consulte este apartado para ver un ejemplo de uso.



set-xsi-nil

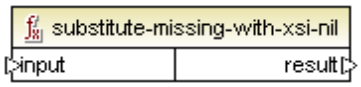
Define el nodo de destino como `xsi:nil`.



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

Para los nodos de contenido simple esta función reemplaza los valores que faltan o los valores nulos del componente de origen con el atributo `xsi:nil` en el nodo de destino.

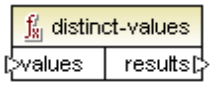
 © 2014 Altova GmbH

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.

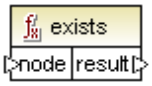
- El parámetro de entrada **key** tiene un tipo de datos arbitrario que se puede convertir en `string` en las funciones de secuencia **group-adjacent** y **group-by**.
- El parámetro de entrada **bool** tiene el tipo de datos **Boolean** en las funciones de secuencia **group-starting-with** y **group-ending-with**.
- 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.

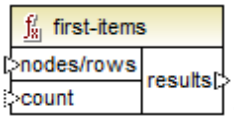
Consulte este apartado para ver un ejemplo de uso.



exists

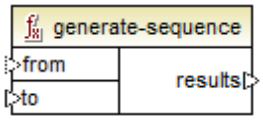
Devuelve **true** si el nodo existe. De lo contrario, devuelve **false**.

Consulte este apartado para ver un ejemplo de uso.



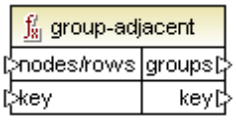
first-items

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**.

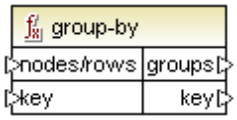


group-adjacent

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.

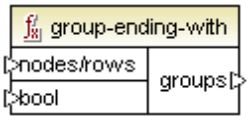
Consulte este apartado para ver un ejemplo de uso.



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.

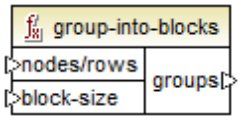
Consulte este apartado para ver un ejemplo de uso.



group-ending-with

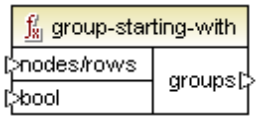
Agrupa la secuencia del parámetro de entrada **nodes/rows** en grupos, terminando con un grupo nuevo si el parámetro **bool** es `true`.

Consulte este apartado para ver un ejemplo de uso.



group-into-blocks

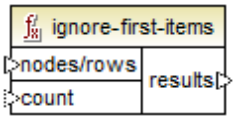
Agrupa la secuencia de entrada **nodes/rows** en bloques del mismo tamaño. El tamaño de los bloques viene dado por el parámetro **block-size**.



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**.

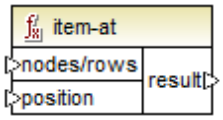
Consulte este apartado para ver un ejemplo de uso.



ignore-first-items

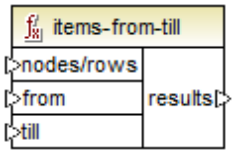
Pasa por alto los **X** primeros elementos de la secuencia de entrada, siendo **X** el número dado por el parámetro **count**. Por ejemplo, si se asigna el valor tres al parámetro **count** y un nodo primario al parámetro **nodes/row**, la función pasará por alto los tres primeros elementos del resultado.

 © 2014 Altova GmbH



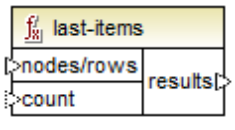
item-at

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`**.



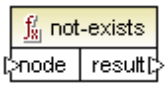
items-from-till

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

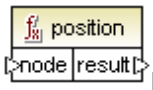
Devuelve los **`X`** últimos nodos de la secuencia **`nodes/rows`**, siendo X el número dado por el parámetro **`count`**.



not-exists

Si el nodo existe, la función devuelve **`false`**. De lo contrario, devuelve **`true`**.

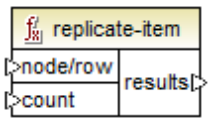
Consulte este apartado para ver un ejemplo de uso.



position

Devuelve la posición de un nodo dentro de la secuencia de la que forma parte.

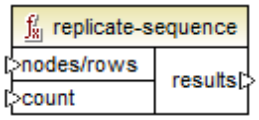
Consulte este apartado para ver un ejemplo de uso.



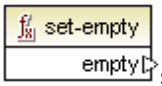
replicate-item

Repite cada elemento de la secuencia de entrada tantas veces como se indique en el parámetro **`count`**. Tenga en cuenta que el parámetro **`count`** se evalúa por cada elemento.

replicate-sequence

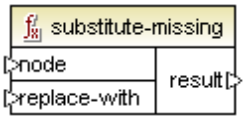


Copia **X** de la secuencia de entrada **nodes/rows** a la secuencia de salida, siendo **X** el número dado por el parámetro **count**.



set-empty

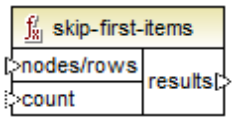
Devuelve una secuencia vacía.



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.

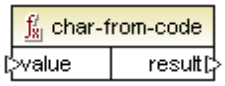


skip-first-items

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.

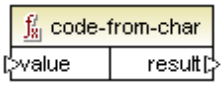
string (funciones de cadena)

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

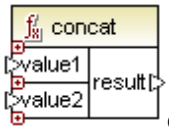
El resultado es el carácter que represente el valor Unicode decimal del parámetro **value**.



code-from-char

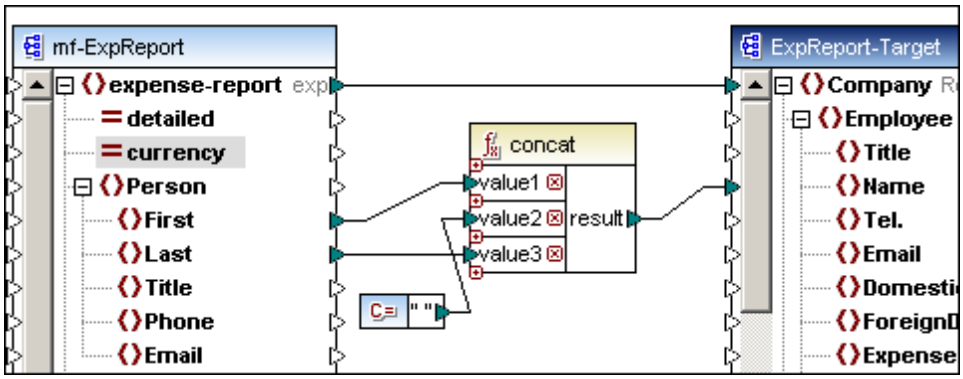
El resultado es el valor Unicode decimal del primer carácter del parámetro **value**.

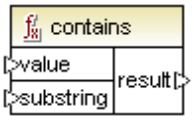
 © 2014 Altova GmbH



concat

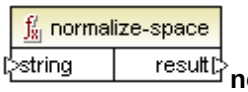
Concatena (anexa) dos o más valores dando lugar a una sola cadena. Todos los valores de entrada se convierten automáticamente en valores de tipo `string`.





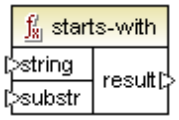
contains

El resultado es **`true`** si los datos asignados al parámetro de entrada **`value`** contienen la cadena asignada al parámetro de entrada **`substring`**.



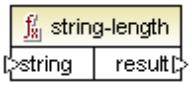
normalize-space

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

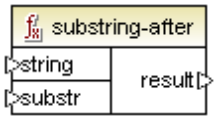
El resultado es el número de caracteres asignados al parámetro de entrada **`string`**.

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.

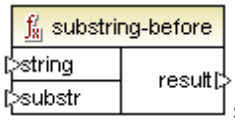
P. ej. la función **`substring("56789",2,3)`** da como resultado `678` (la cadena es `56789`, la subcadena debe empezar en la posición `2` de la cadena y debe tener una longitud de `3` caracteres)



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.

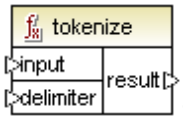
P. ej. la función **`substring-after("2009/01/04","/")`** da como resultado la subcadena `01/04`.



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`.

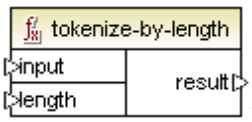


tokenize

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`**.



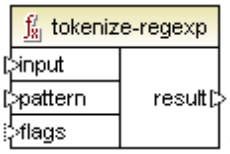
tokenize-by-length

 © 2014 Altova GmbH

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.

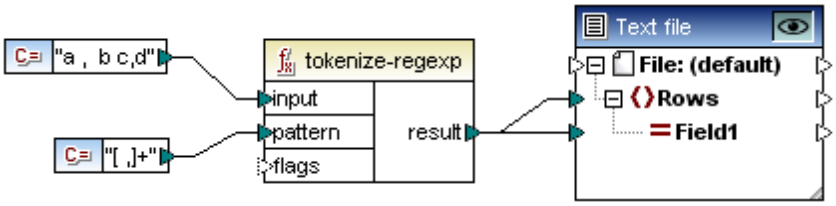
P. ej. la cadena **input** es ABCDEF y el parámetro **length** es 2 . El resultado es AB CD EF.

Para más información consulte el apartado Ejemplos de la función **tokenize**.



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.

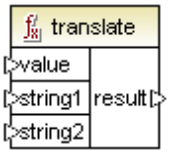
El cuantificador + especifica "una o más" apariciones de la clase de caracteres/cadena.

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).

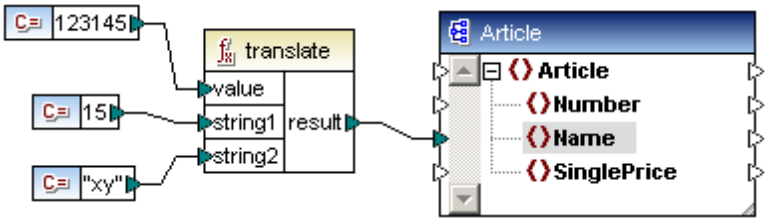
Consulte el apartado Expresiones regulares para más información.



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**"

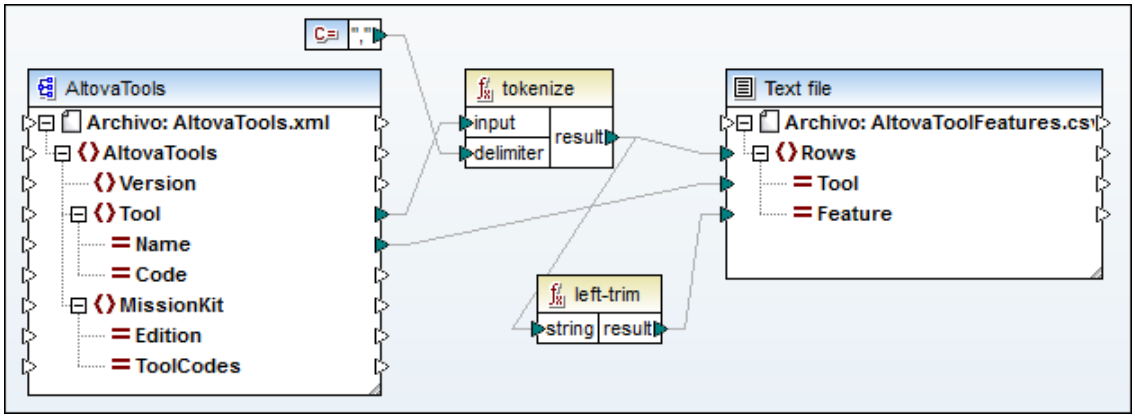
La cadena resultante es **ccbccabac**

Ejemplos de la función tokenize

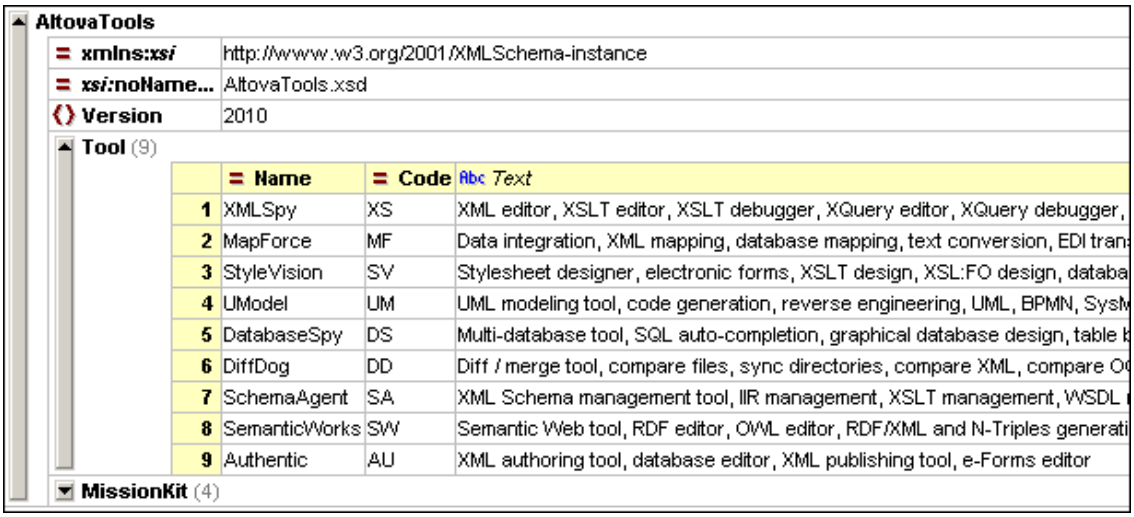
Ejemplo de uso de la función **tokenize**

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`

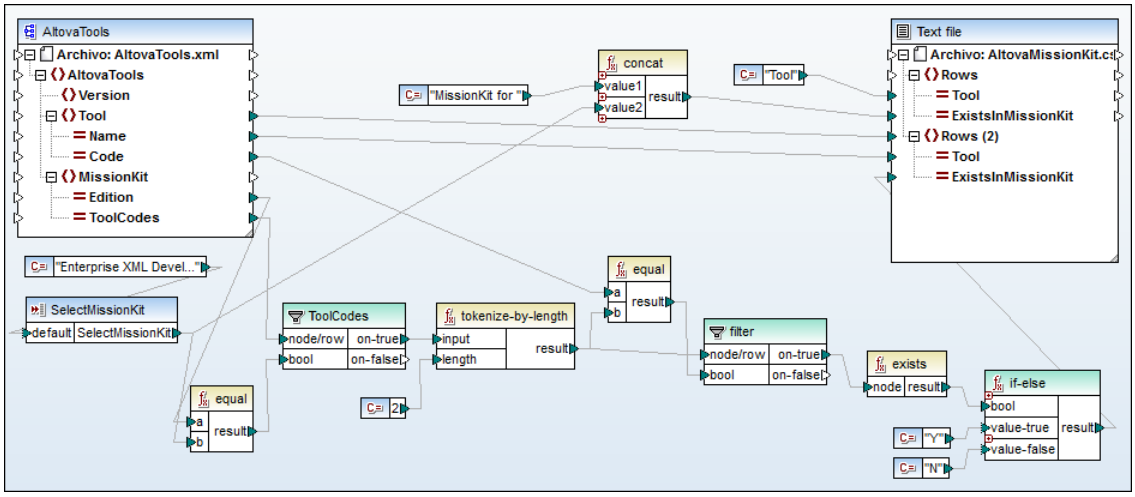
- Esto se repite con todos los trozos del elemento **`Tool`** actual y después con todos los demás elementos **`Tool`**.
- Abra el panel **Resultados** para ver el resultado de la asignación.

```
 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

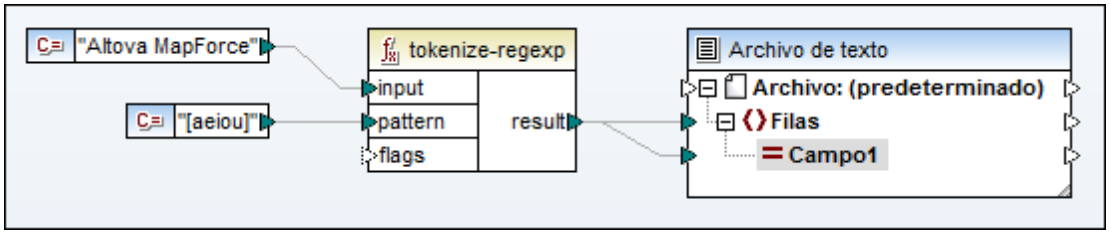
MapForce admite el uso de expresiones regulares en el parámetro **pattern** de la función **tokenize-regexp** para buscar cadenas concretas en el parámetro de entrada.

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

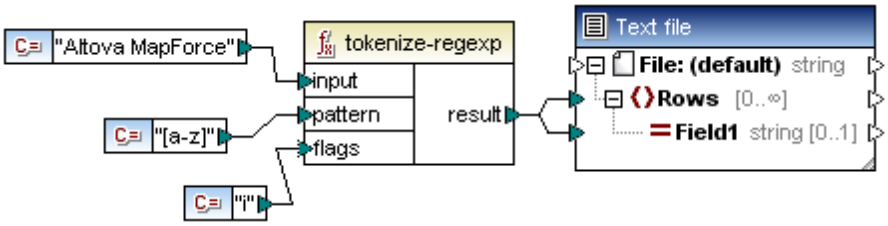
m
Si está presente, el proceso de búsqueda opera en el modo multilínea.

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.

14.3.2 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`.

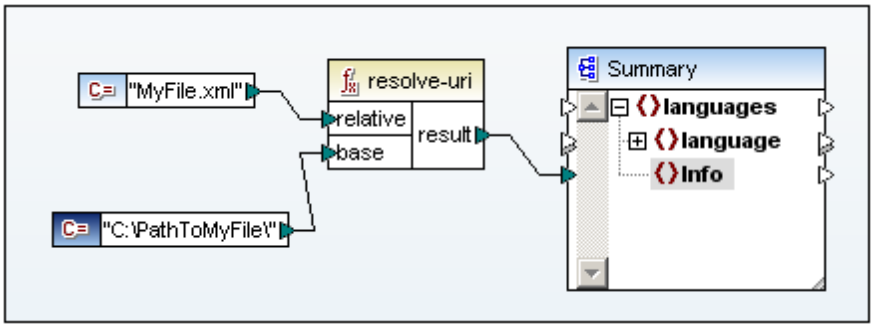
Cuando el argumento de entrada es un valor de un tipo atómico (por ejemplo `xs:decimal`), este valor atómico se convierte en un valor de tipo `xs:string`. Si el argumento de entrada es un nodo, se extrae el valor de cadena del nodo. (El valor de cadena de un nodo es una concatenación de los valores de 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`.

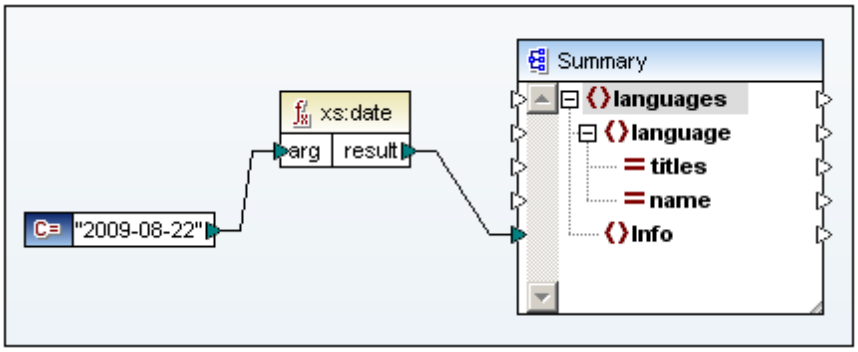
Función `false`

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.

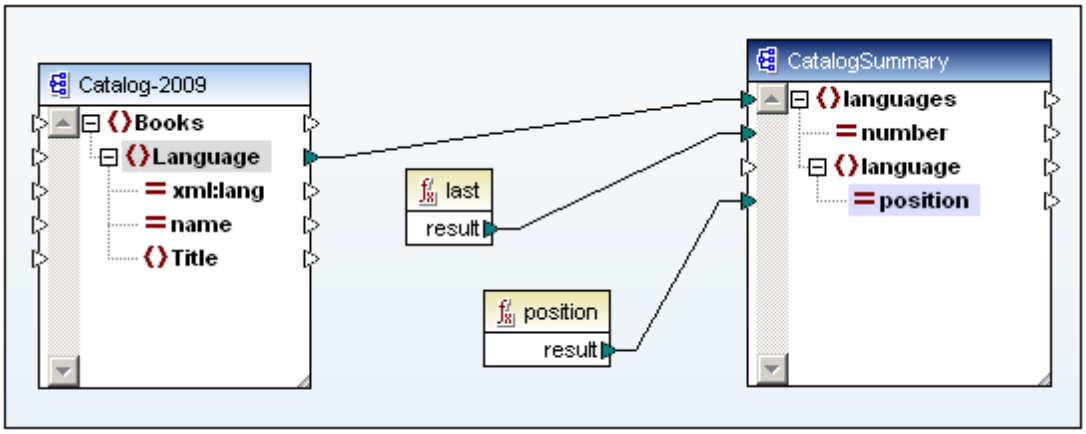
last, position

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

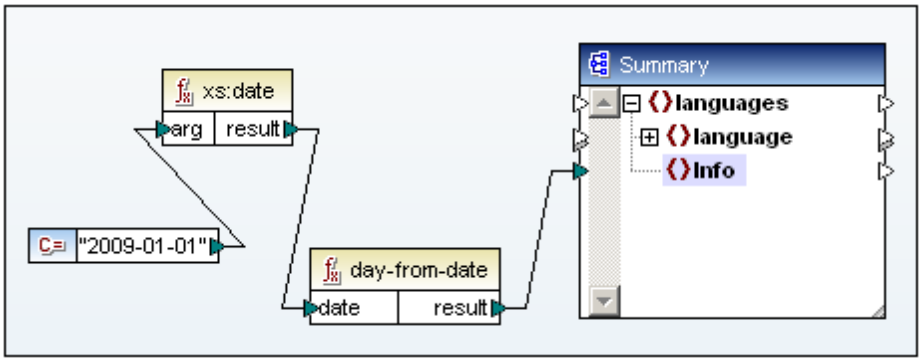
original se sustituye con el uso horario del segundo argumento.

- 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`

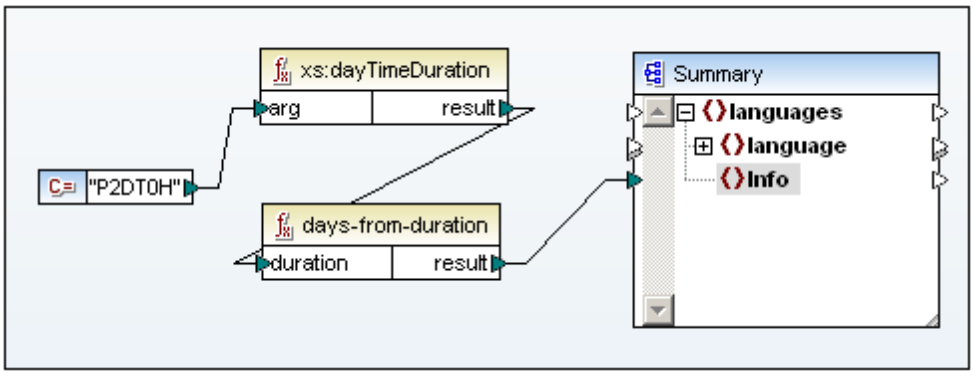
Cada función **`From`** extrae un componente de los (i) datos de fecha u hora o de los (ii) datos de duración. Los resultados son de tipo de datos `xs:decimal`.

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`.



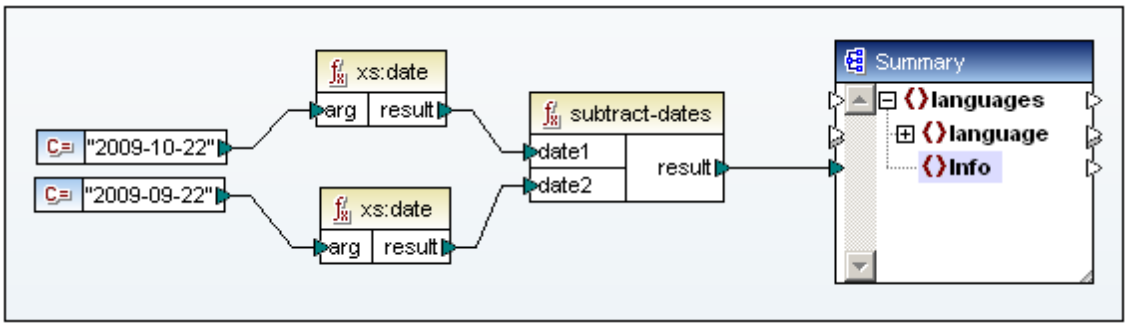
Funciones `Substract`

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.



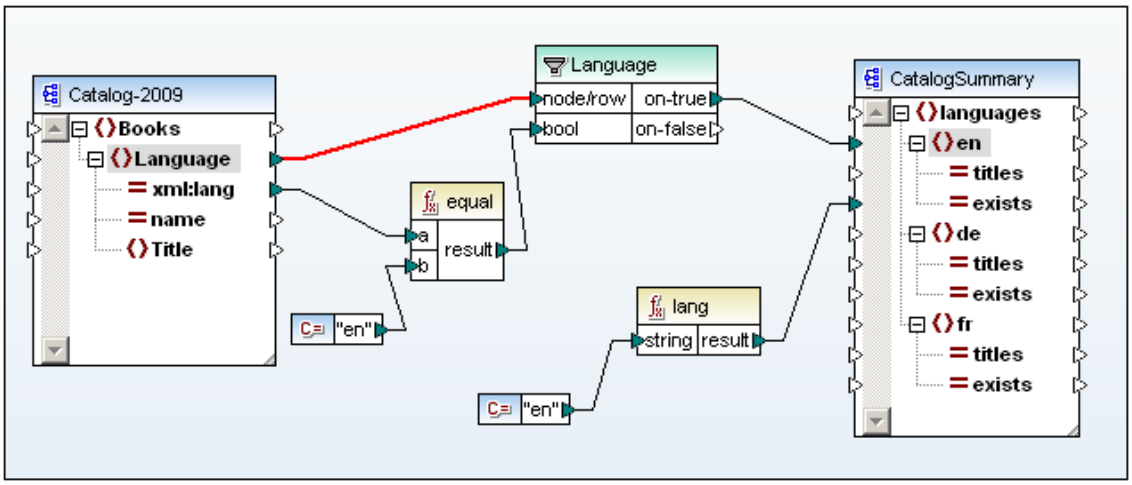
Nota: al pasar el puntero del mouse por el argumento de entrada de la función aparece el tipo de datos esperado del argumento.

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

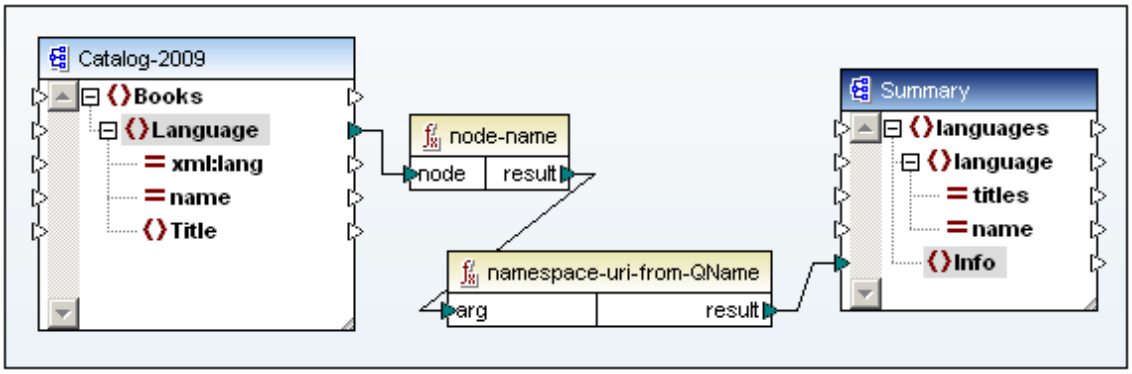
qname-related (funciones para QName)

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.

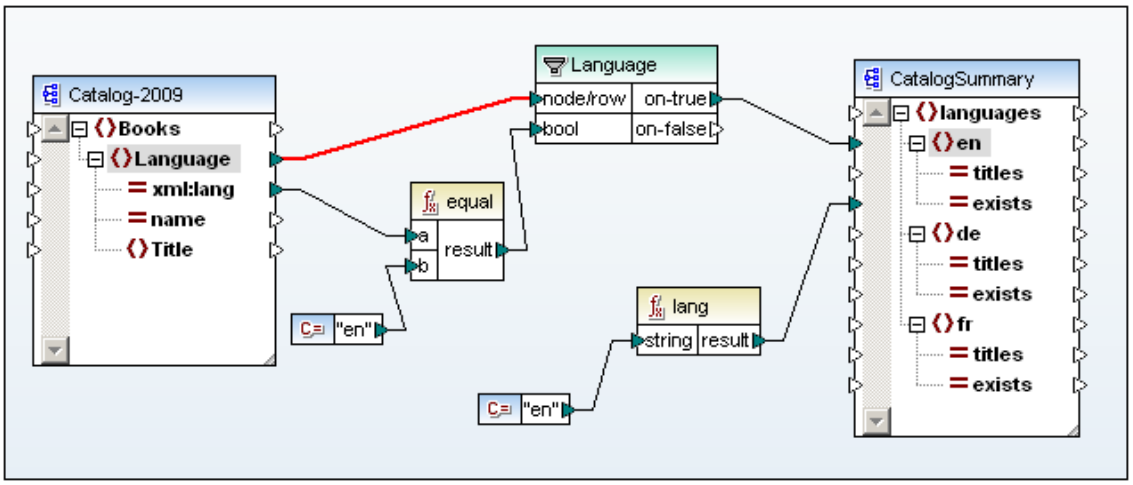
14.3.3 Biblioteca xslt

- 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:

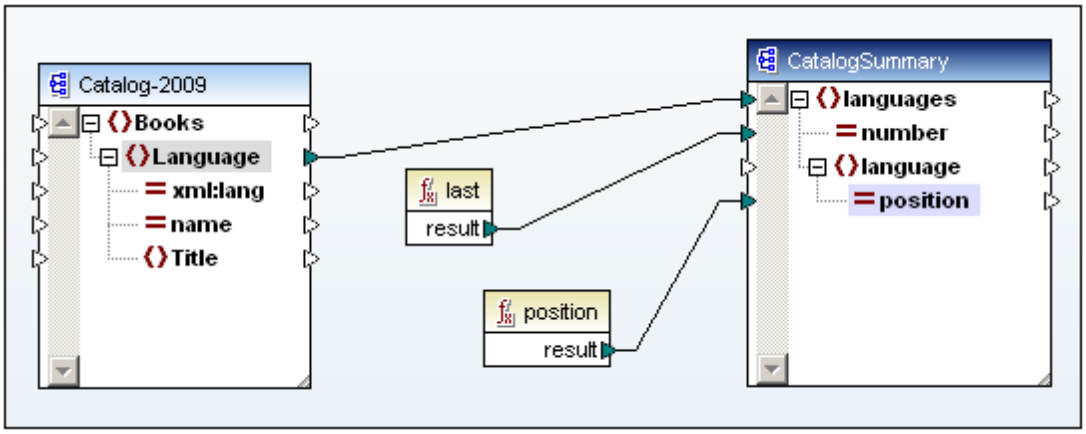
1. El elemento `Language` del esquema de origen tiene un atributo `xml:lang`.
2. Los nodos `Language` se filtran para que solamente se procesen aquellos cuyo atributo `xml:lang` tenga el valor `en` (la prueba de filtro se especifica en la función **`equal`**).
3. El nodo `Language` es el nodo de contexto 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**.



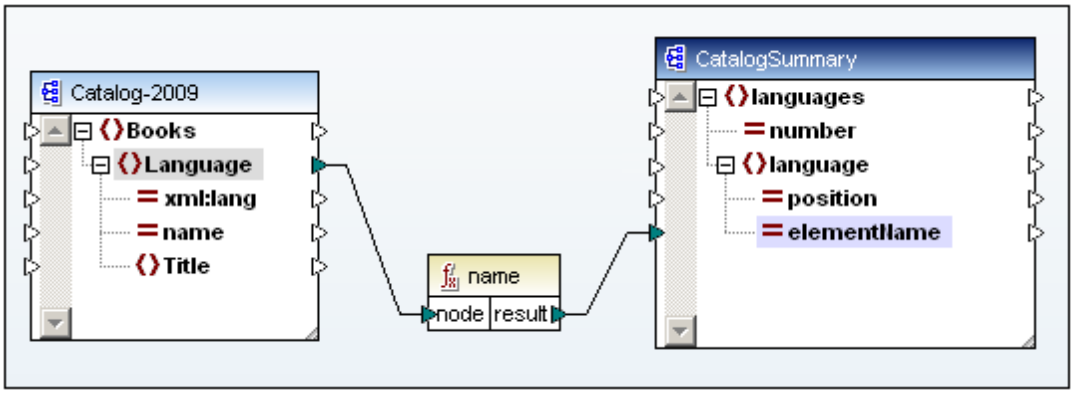
En el ejemplo, la función **last** devuelve la posición del último nodo del conjunto de nodos de contexto (el conjunto de nodos de los elementos `Language`) como valor del atributo `number`. Este valor también es el tamaño del conjunto de nodos, puesto que indica el número de nodos que contiene el conjunto.

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

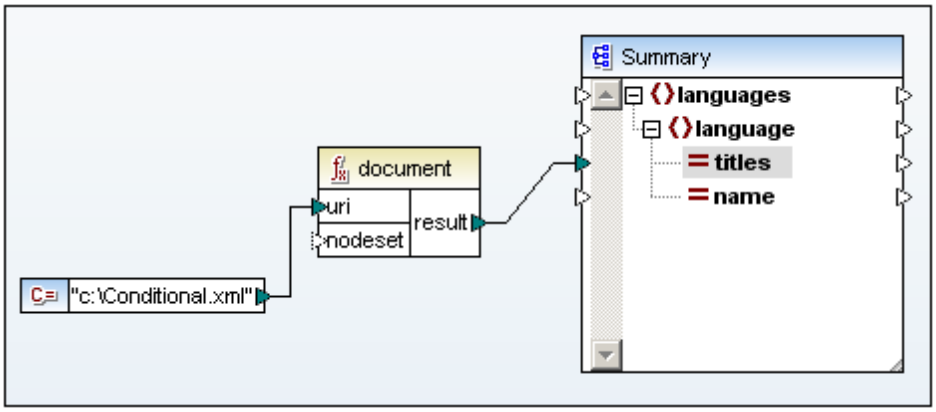
Las funciones de la biblioteca **xslt | xslt** son funciones XSLT 1.0. Pase el puntero del mouse por el argumento de entrada de la función para ver el tipo de datos esperado del argumento.

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.

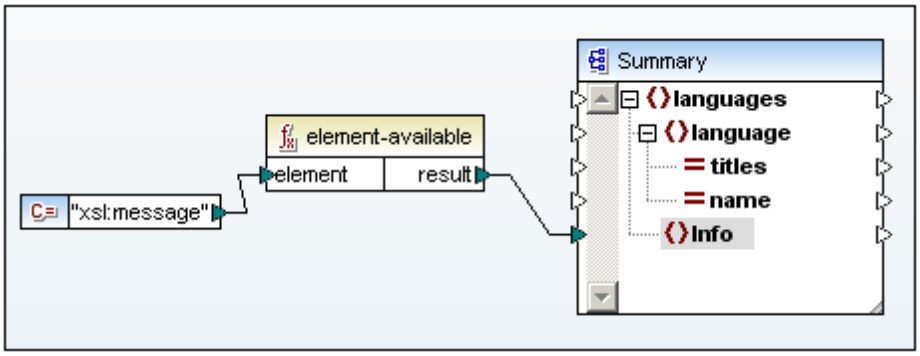


 © 2014 Altova GmbH

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

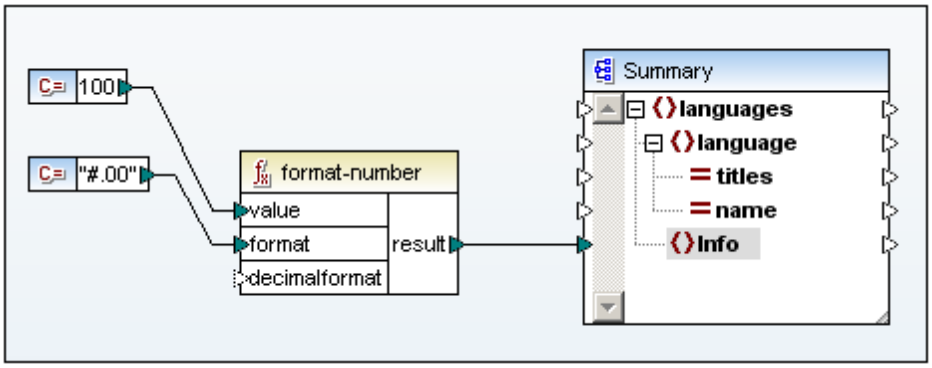
Esta función es similar a la función **element-available** y comprueba si el nombre de la función dado como argumento es compatible con el procesador XSLT.

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.



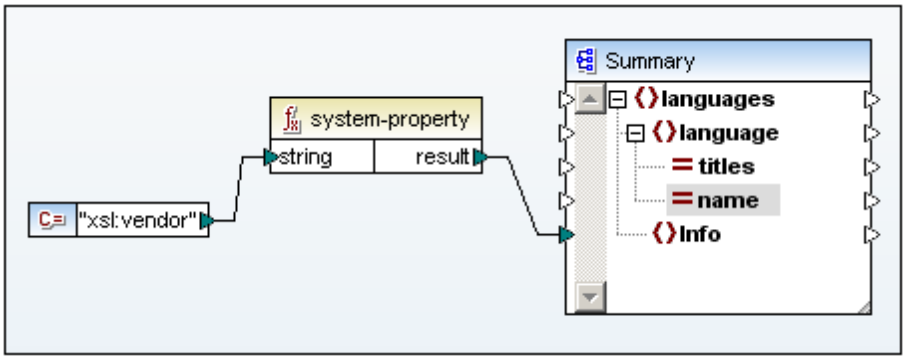
La función devuelve el número con formato de cadena.

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 2015

Referencia del usuario

15 Referencia del usuario

Este apartado enumera y describe todos los menús y opciones de menú de MapForce.

© 2014 Altova GmbH

15.1 Archivo

Nuevo
Crea un documento de asignación nuevo.

Abrir
Abre archivos de asignación (*.mfd) .

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.

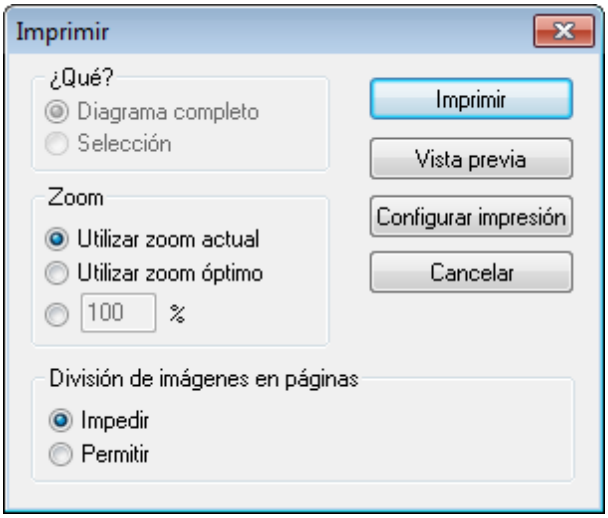
Guardar todos
Guarda todos los archivos que están abiertos.

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.

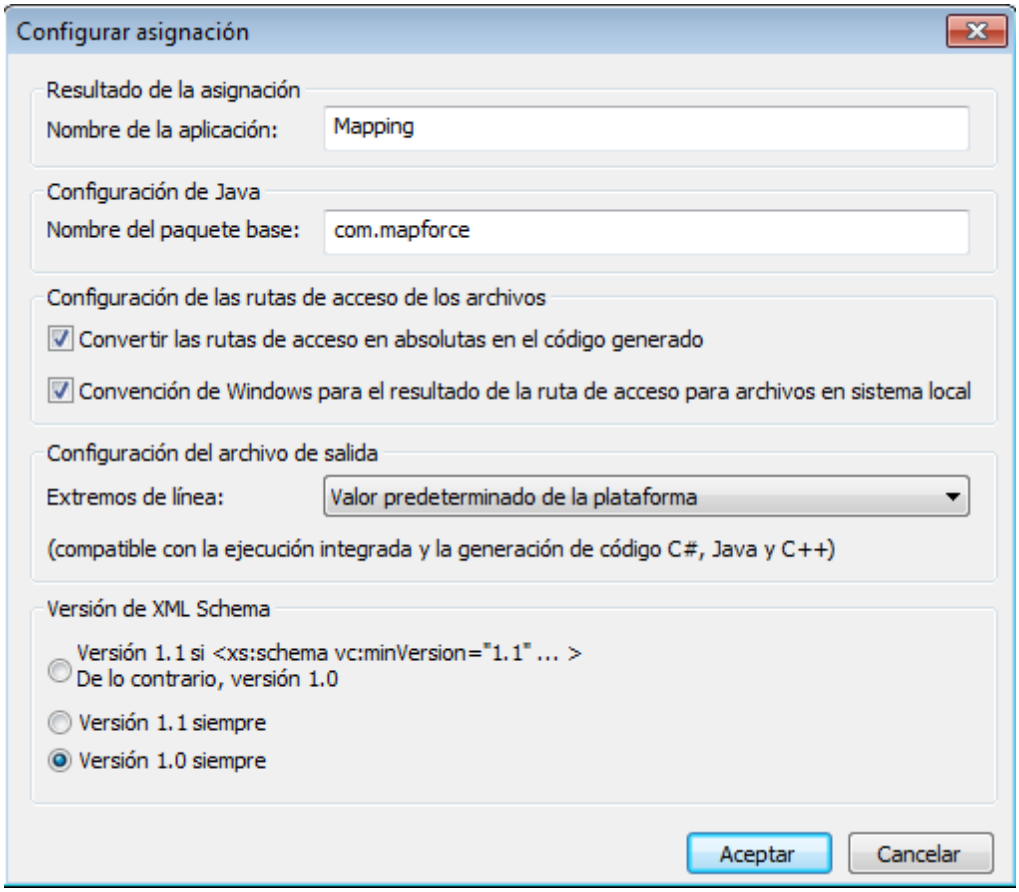
Vista previa de impresión
Abre el cuadro de diálogo "Imprimir" (ver comando **Imprimir**).

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 para los archivos de transformación generados.

Configuración de las rutas de acceso de los archivos

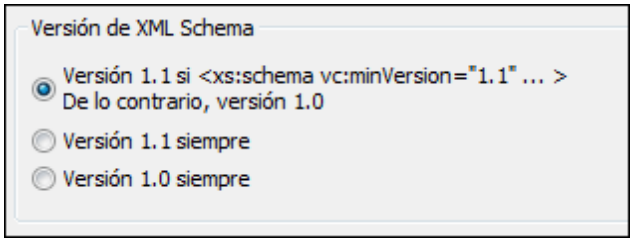
Convertir las rutas de acceso en absolutas en el código generado: garantiza la compatibilidad del código generado con archivos de asignación (`*.mfd`) creados en versiones de MapForce anteriores a la versión 2010. Consulte el apartado Rutas de archivo relativas y absolutas para más información.

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.

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.

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.



Notas:

- 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.
- 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.

Si cambia esta opción de configuración en una asignación, se volverán a cargar todos los esquemas que tengan la versión de esquema XML seleccionada y también podría cambiar su validez.

Generar código en el lenguaje seleccionado
Genera código en el lenguaje seleccionado en la barra de herramientas: XSLT / XSLT 2.

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**).

Archivos recientes
Una lista con archivos abiertos recientemente.

Salir

 © 2014 Altova GmbH

Cierra la aplicación. Antes aparece un aviso preguntando si desea guardar los archivos que estén sin guardar.

15.2 Edición

La mayoría de los comandos del menú **Edición** se habilitan 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** 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**, y **Resultados**.

 © 2014 Altova GmbH

15.3 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**, y **Resultados**. Si selecciona un archivo XML sin referencia de esquema, aparece un aviso preguntando si desea generar automáticamente el esquema XML correspondiente.

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

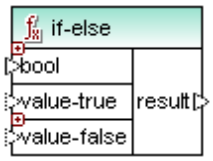
Inserta un componente con dos parámetros de entrada (**`node/row`** y **`bool`**) y dos de salida (**`on-true`** y **`on-false`**). Si el valor booleano es `true`, el valor del parámetro **`node/row`** se envía al parámetro **`on-true`**. Si el booleano es `false`, el valor del parámetro **`node/row`** se envía al parámetro **`on-false`**. Para ver un ejemplo de uso consulte el *Tutorial*.

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`**.

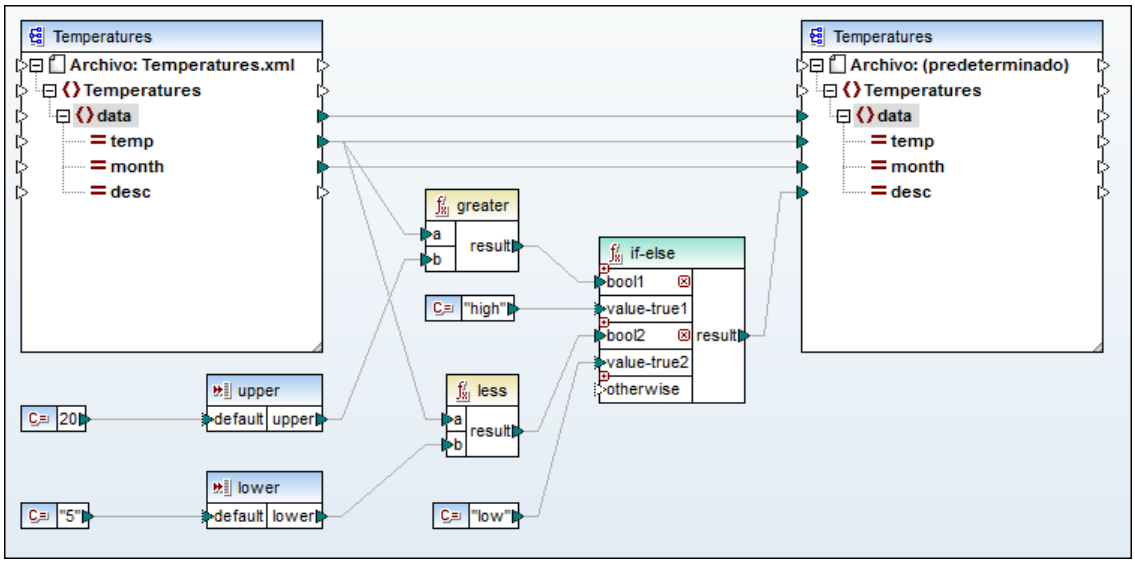


- El primer parámetro de entrada es **`bool`**, que contiene valores booleanos.
- Si el valor de **`bool`** es `true`, los datos del parámetro de entrada **`value-true`** se pasan al parámetro de salida **`result`**.
- Si el valor de **`bool`** es `false`, los datos del parámetro de entrada **`value-false`** 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**.
- 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"/>
```

 © 2014 Altova GmbH

15.4 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.

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*.

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.

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`)

El nombre del componente puede contener:

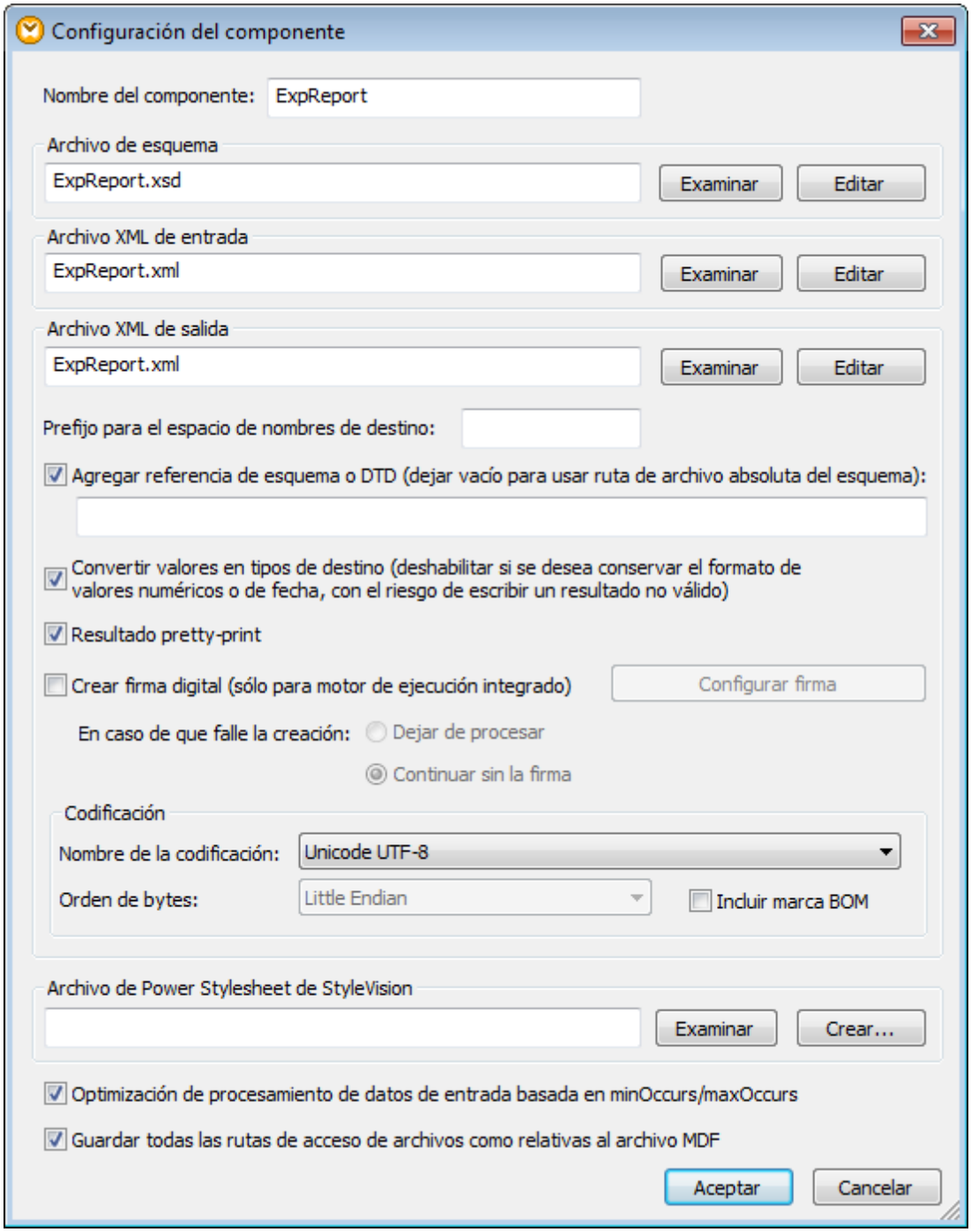
- espacios (p. ej. `Archivo de texto` o `Archivo Excel`). No está permitido usar espacios al principio ni al final de la cadena.
- puntos (p. ej. `Orders.EDI`)

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

 © 2014 Altova GmbH



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

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**.

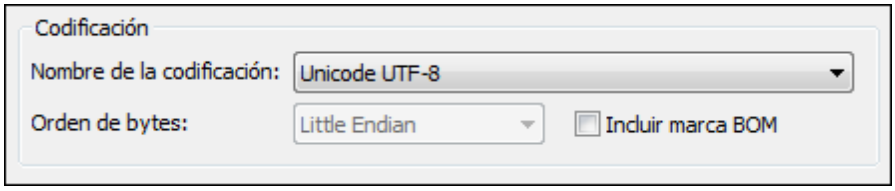
Desactive esta opción para conservar el formato exacto de los valores. Por ejemplo, para cumplir con la faceta `pattern` de un esquema que exige un número determinado de dígitos decimales en un valor numérico.

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.)

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 .

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*

 © 2014 Altova GmbH

- *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".

15.5 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.

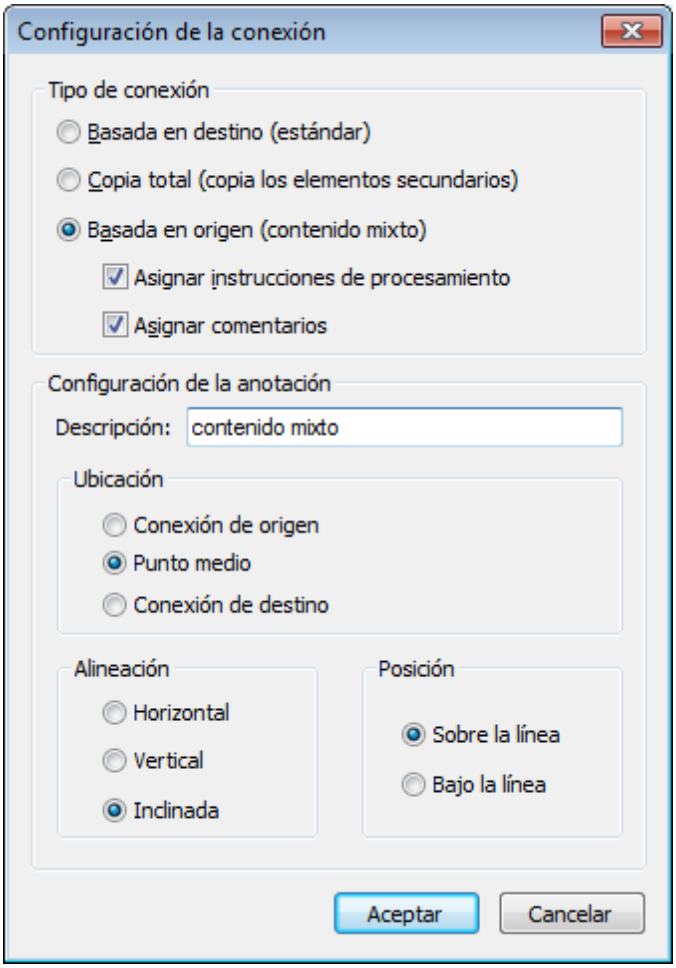
Copia total (copia los elementos secundarios)
Crea conectores para todos los elementos secundarios equivalentes, donde cada uno de los conectores secundarios aparece como una rama del conector primario. Para más información consulte el apartado Conexiones de copia total.

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.

 © 2014 Altova GmbH

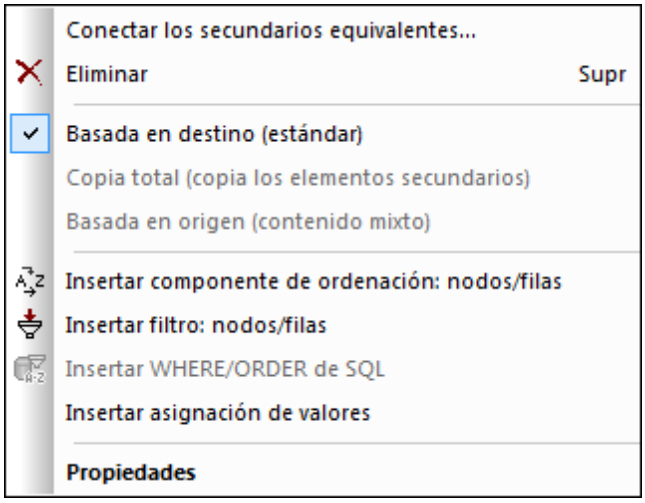


Configuración de la anotación:

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:



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.

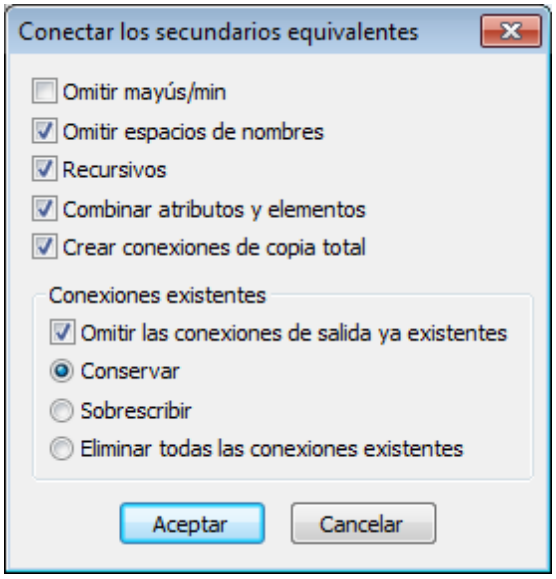
Basada en origen (contenido mixto)
Cambia el tipo de conector a una asignación basada en origen / de contenido mixto. Para más información consulte el apartado Asignación basada en origen y de contenido mixto.

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.**

 © 2014 Altova GmbH



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 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.

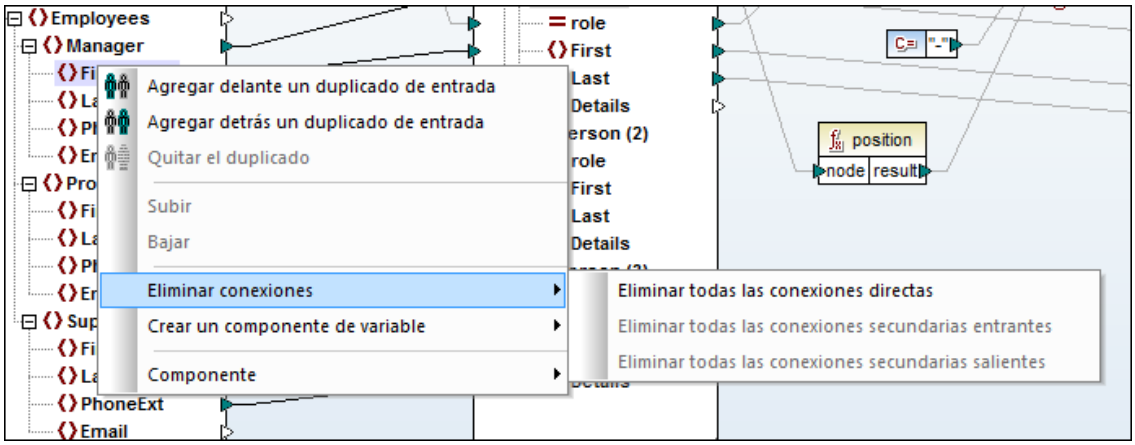
Conservar
Conserva las conexiones ya existentes.

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.

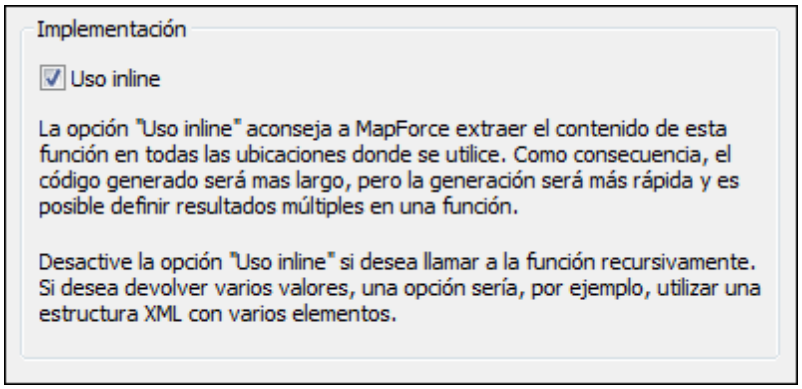
 © 2014 Altova GmbH

15.6 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**.

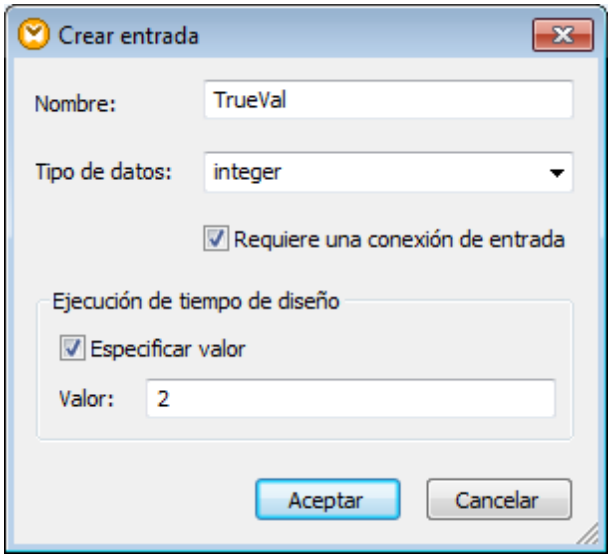


Quitar función
Elimina la función definida por el usuario activa. Haga doble clic en la barra de título de la función definida por el usuario y después seleccione **Función | Quitar función**. Es decir, este comando solamente está disponible mientras se edita la función definida por el usuario.

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.

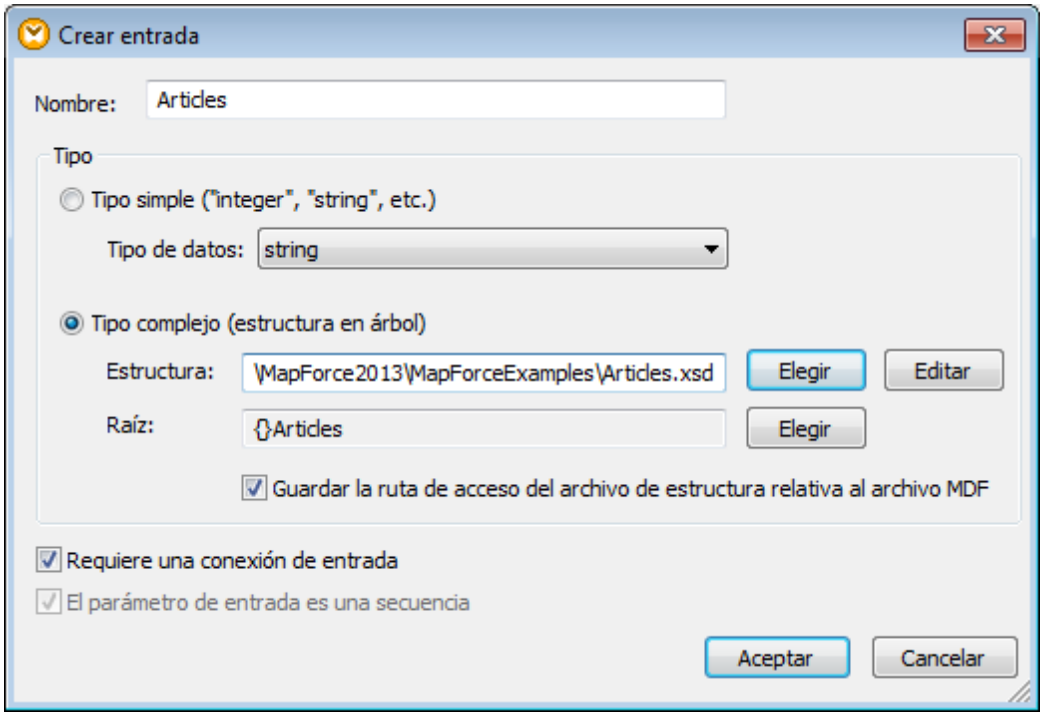
Insertar componente de entrada
Inserta un componente de entrada en la asignación o en la función definida por el usuario.

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.



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)



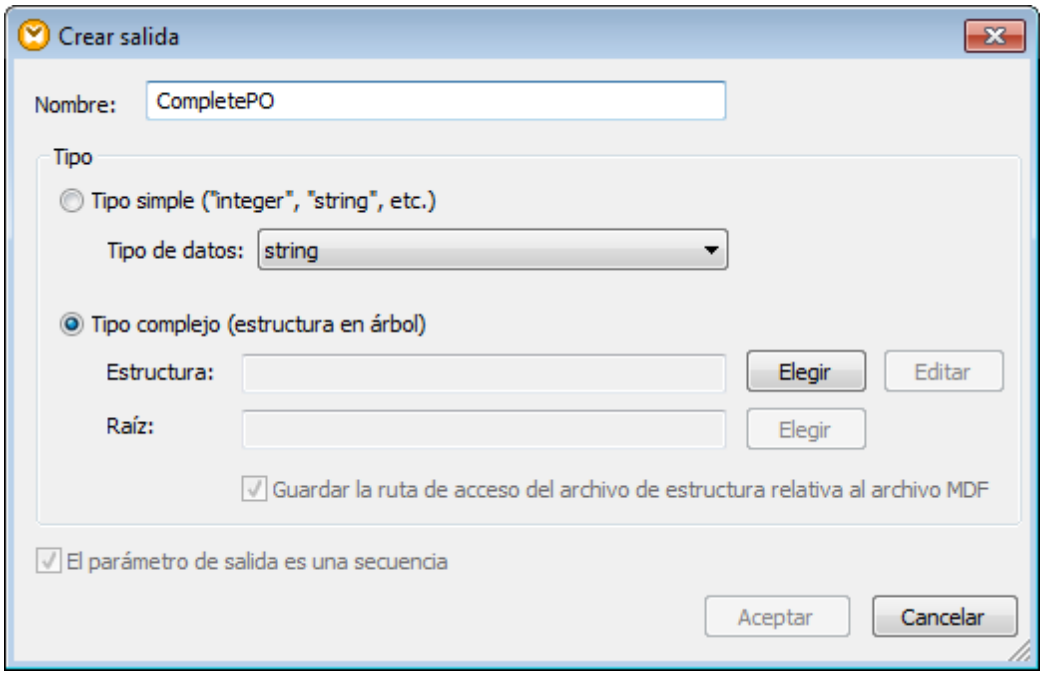
Insertar componente de salida

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

 © 2014 Altova GmbH

- salidas de tipo complejo (p. ej. estructuras basadas en esquemas)



15.7 Resultados

El primer grupo de opciones (**XSLT 1.0, XSLT 2.0, etc.**) sirve para definir el lenguaje de destino del código.

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**.

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**.

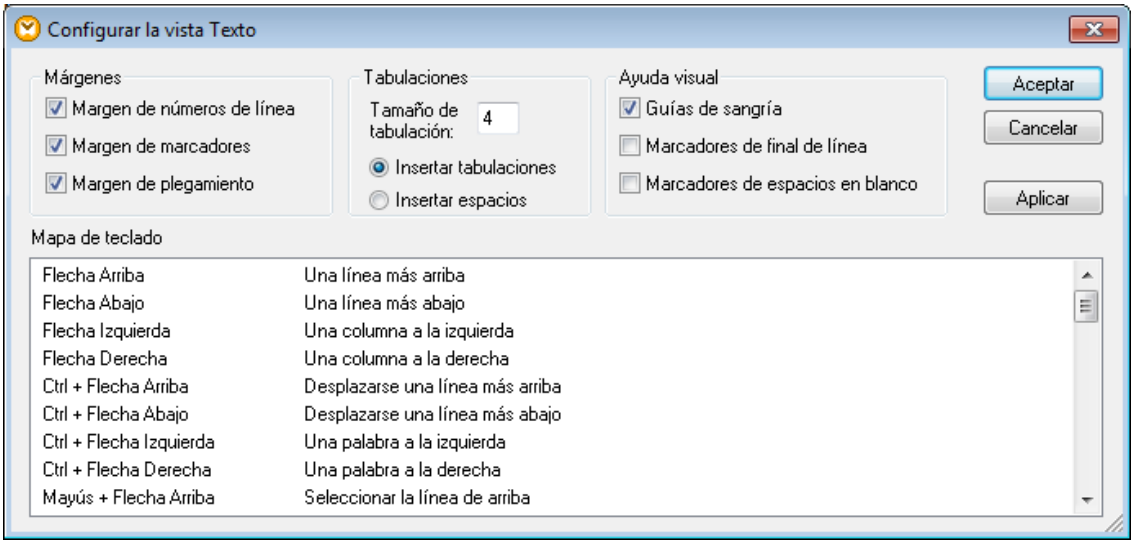
Marcador anterior
Navega hasta el marcador anterior en el panel **Resultados**.

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.

 © 2014 Altova GmbH



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**.

15.8 Vista

Mostrar anotaciones
Muestra las anotaciones del esquema XML 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.

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

 © 2014 Altova GmbH

Avanza por las asignaciones que están abiertas en el panel **Asignación**.

Barra de estado
Muestra/oculta la barra de estado que está visible bajo la ventana Mensajes.

Ventana de bibliotecas
Muestra/oculta la ventana Biblioteca, que contiene todas las funciones.

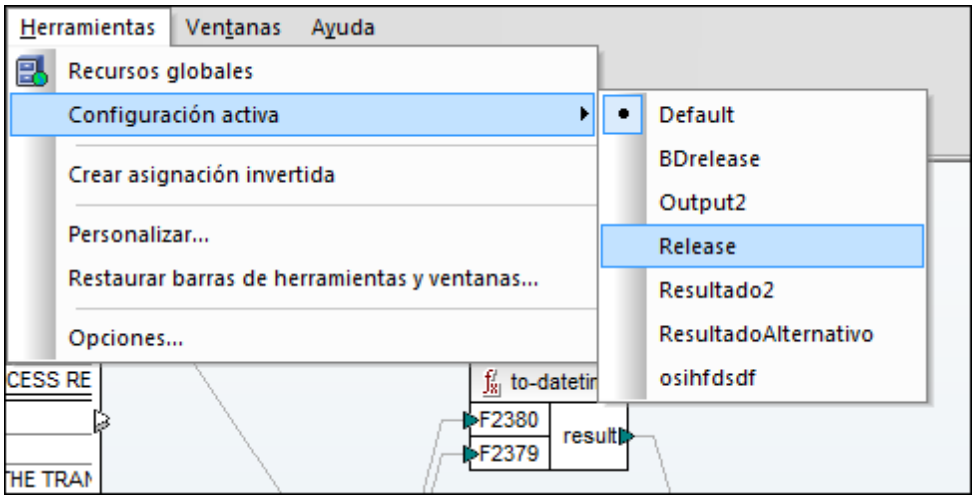
Mensajes
Muestra/oculta la ventana Mensajes, donde aparecen los resultados de la validación, etc. La ventana Mensajes se activa automáticamente cuando se genera código para mostrar el resultado de la validación.

Vista general
Muestra/oculta la ventana Vista general. Arrastre el rectángulo rojo por la ventana de vista general para navegar por el panel **Asignación**.

15.9 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.

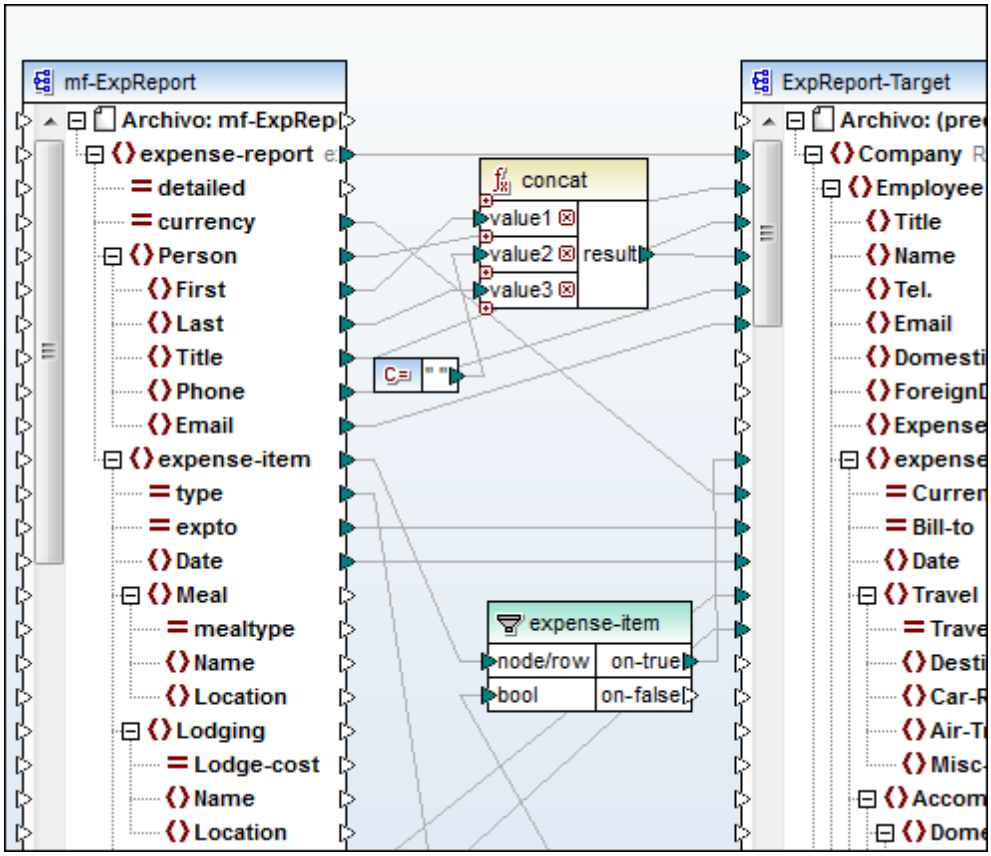
Configuración activa
Permite seleccionar un recurso global o cambiar el recurso global activo desde una lista que enumera todos los recursos y configuraciones del archivo XML de recursos globales. Seleccione la configuración deseada en el submenú.



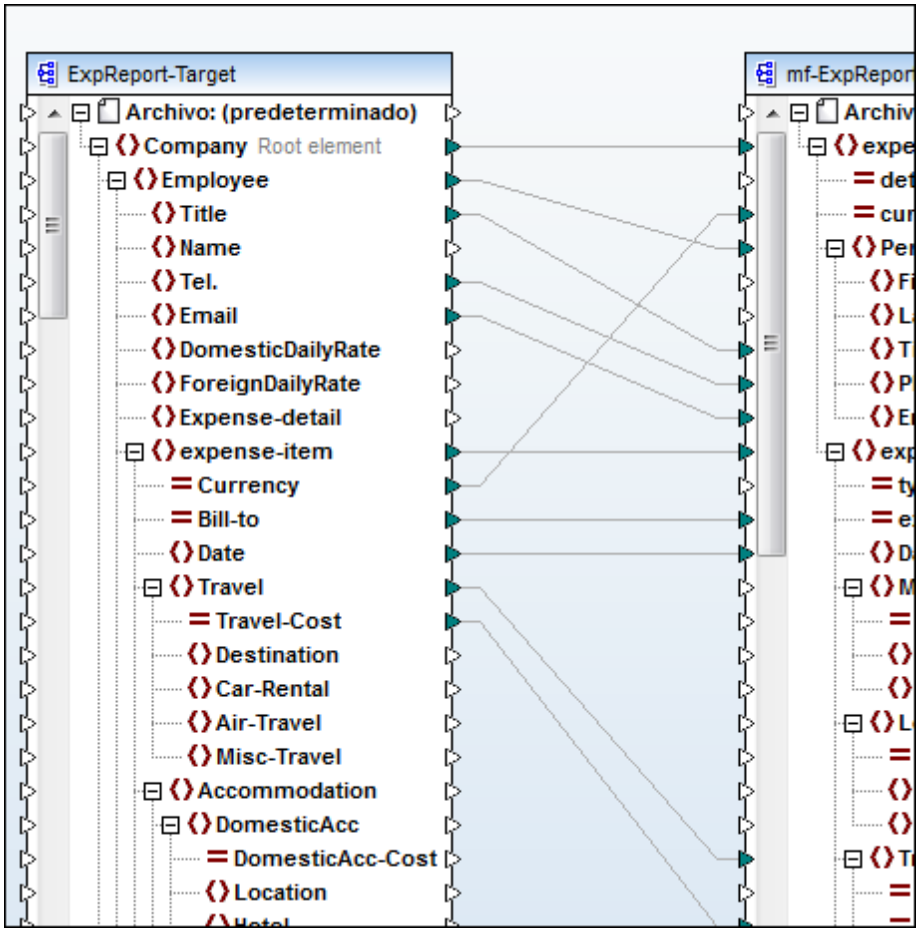
Crear asignación invertida
Crea una asignación invertida a partir de la asignación activa. Tenga en cuenta que el resultado no es una asignación completa porque en la asignación invertida solamente se conservan las conexiones directas entre los componentes de la asignación original. Es muy posible que 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`:

 © 2014 Altova GmbH



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...

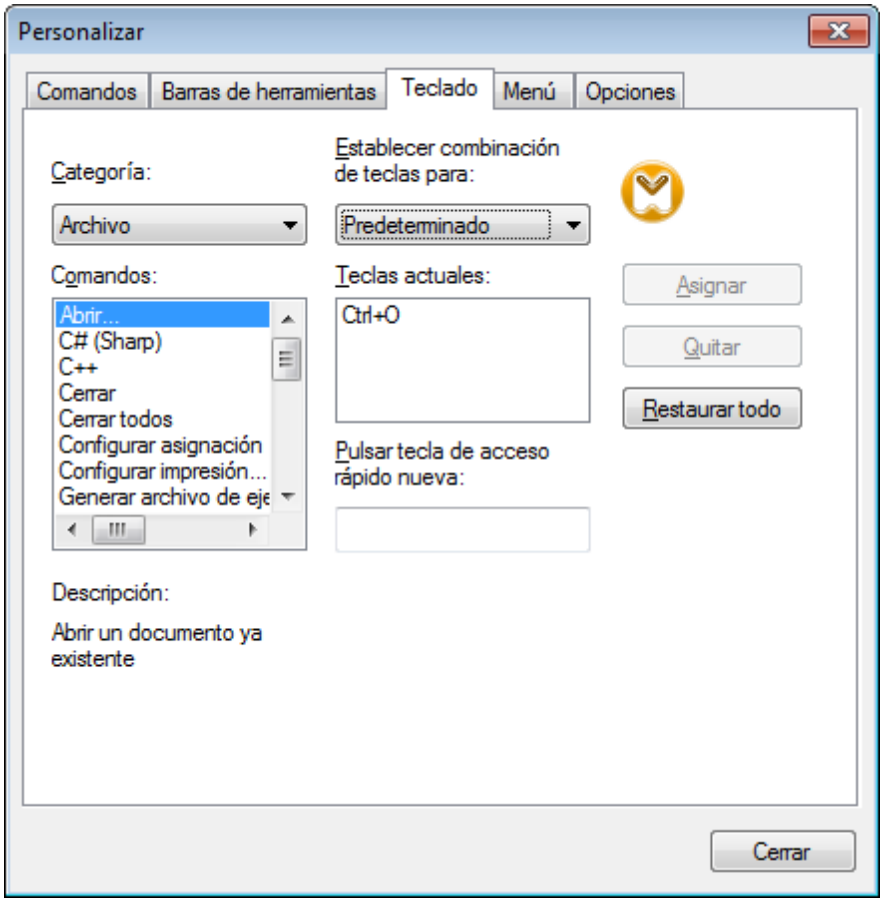
 © 2014 Altova GmbH

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

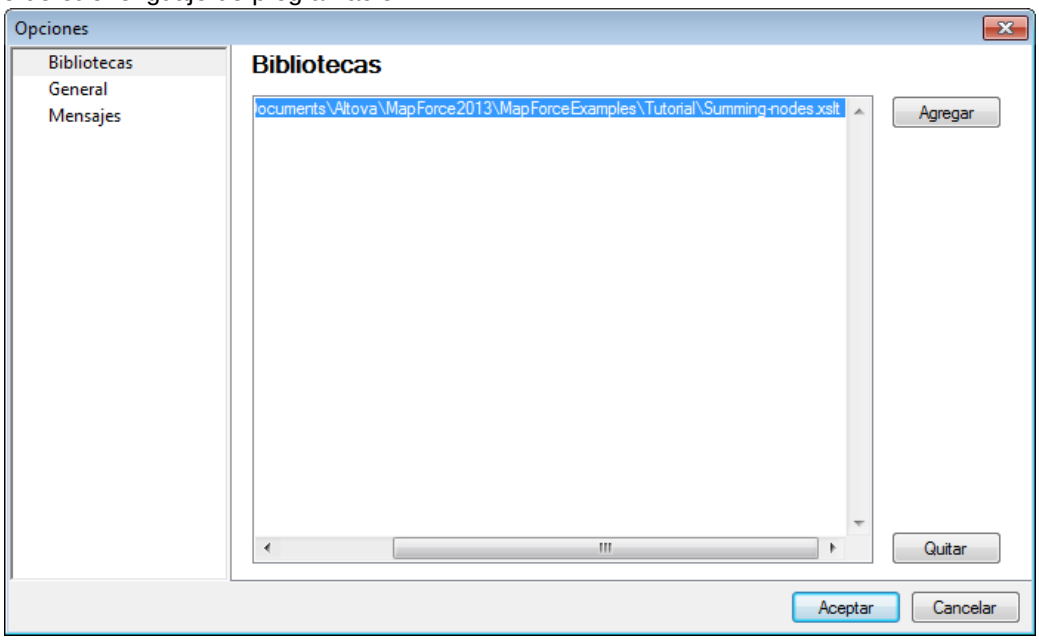
Opciones

Abre el cuadro de diálogo "Opciones" donde puede:

- Agregar o eliminar funciones XSLT 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.

Pestaña *Bibliotecas*:

- Desde aquí puede agregar o eliminar funciones/bibliotecas XSLT definidas por el usuario o de otro lenguaje de programación.



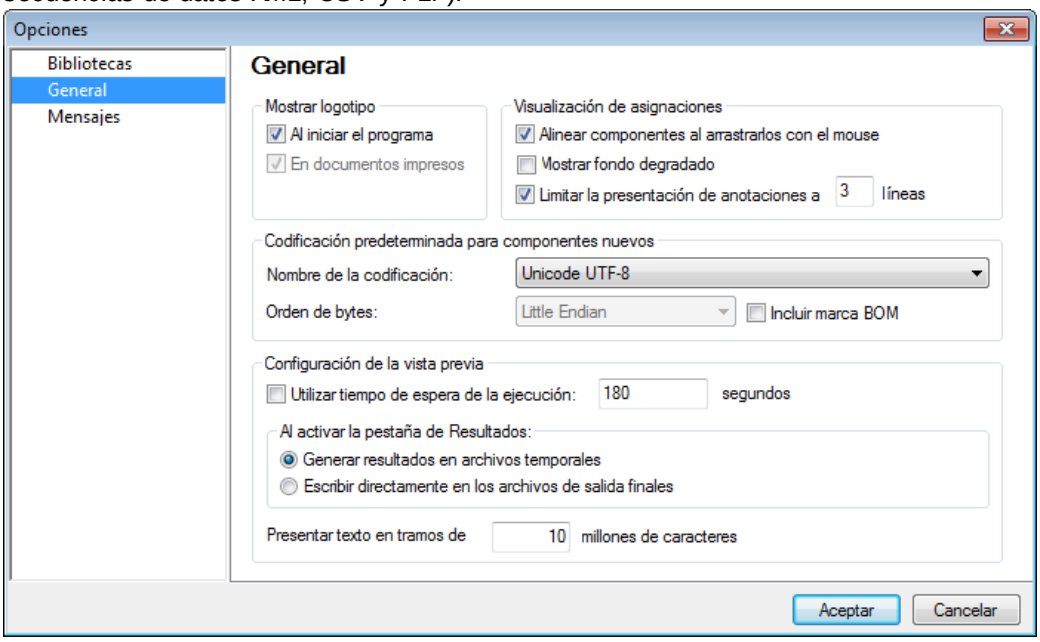
Pestaña *General*:

- 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**.

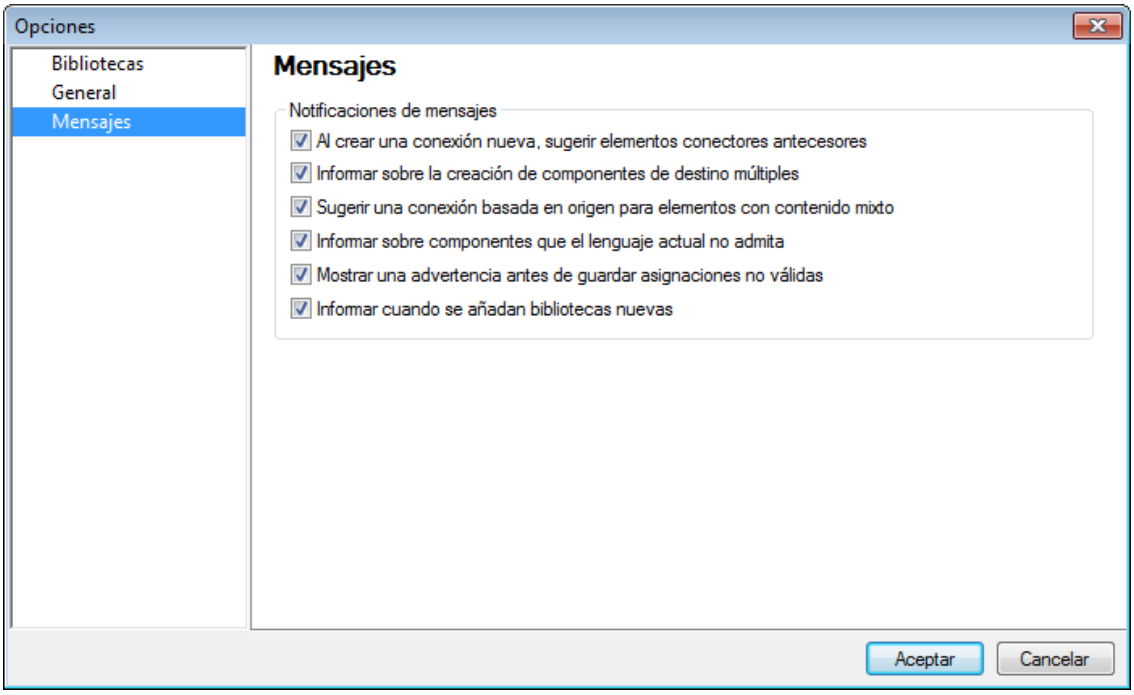
  Advertencia: habilite la opción *Escribir directamente en los archivos de salida finales* para sobrescribir los archivos de resultados sin solicitar confirmación previa.

- 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


Opciones
Bibliotecas
General
Mensajes
Mensajes
Notificaciones de mensajes
Al crear una conexión nueva, sugerir elementos conectores antecesores
Informar sobre la creación de componentes de destino múltiples
Sugerir una conexión basada en origen para elementos con contenido mixto
Informar sobre componentes que el lenguaje actual no admita
Mostrar una advertencia antes de guardar asignaciones no válidas
Informar cuando se añadan bibliotecas nuevas
Aceptar
Cancelar

15.10 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

15.11 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

15.11.1 Contenido, índice, buscar

Contenido

Descripción

Abre la ayuda en pantalla por la tabla de contenido. 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.

Índice

Descripción

Abre la ayuda en pantalla por el índice de palabras clave. 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.

Buscar

Descripción

Abre la ayuda en pantalla por la función de búsqueda. Para buscar un término en el documento de ayuda escriba el término de búsqueda en el campo de consulta y pulse la tecla **Entrar**. Después haga doble clic en uno de los temas de la lista para ver su contenido.

© 2014 Altova GmbH

15.11.2 Activación, formulario de pedido, registro, actualizaciones

Activación del software

Descripción

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

Descripción

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

Descripción

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

Descripción

Busca una versión más reciente del producto en el servidor de Altova y emite un mensaje a tal efecto.

© 2014 Altova GmbH

15.11.3 Otros comandos

Centro de soporte técnico

Descripción

Es un enlace al centro de soporte técnico del sitio web de Altova. El centro de soporte técnico incluye preguntas frecuentes, foros de debate y un formulario para ponerse en contacto con el equipo de soporte técnico de Altova.

Preguntas más frecuentes

Descripción

Es un enlace a la página de preguntas frecuentes del sitio web de Altova. Esta página se actualiza constantemente con las preguntas que recibimos de nuestros clientes.

Descargar herramientas gratis y componentes

Descripción

Es un enlace al centro de descargas de componentes del sitio web de Altova. Aquí puede descargar software adicional para usarlo con los productos de Altova, como procesadores XSLT y XSL-FO y paquetes de integración. Estos componentes suelen ser totalmente gratis.

MapForce en Internet

Descripción

Es un enlace al sitio web de Altova, donde encontrará más información sobre MapForce, otros productos de Altova y tecnologías relacionadas.

Acerca de MapForce

Descripción

Abre la pantalla de presentación de la aplicación, que incluye el número de versión del producto e información sobre copyright.

Altova MapForce 2015

Anexos

16 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

16.1 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.

16.1.1 Información sobre motores XSLT y XQuery

Los motores XSLT y XQuery de MapForce siguen las especificaciones del W3C y, por tanto, son más estrictos que otros motores anteriores de Altova, como los de las versiones antiguas de XMLSpy. Por consiguiente, MapForce señala algunos errores leves que antes no se notificaban en la versión anterior de estos motores.

Por ejemplo:

- Se notifica un error de tipo (`err:XPTY0018`) si el resultado de un operador de ruta de acceso contiene tanto nodos como no nodos.
- Se notifica un error de tipo (`err:XPTY0019`) si `E1` en una expresión XPath `E1/E2` no da como resultado una secuencia de nodos.

Si encuentra este tipo de errores, modifique el documento XSLT/XQuery o el documento de instancia según corresponda.

Esta sección describe características relacionadas con la implementación de los motores e incluye estos apartados:

- XSLT 1.0
- XSLT 2.0
- XQuery 1.0

XSLT 1.0

El motor XSLT 1.0 de MapForce cumple con la recomendación XSLT 1.0 del 16 de noviembre de 1999 y con la recomendación XPath 1.0 del 16 de noviembre de 1999, ambas del W3C.

Nota sobre la implementación
Cuando el atributo `method` de `xsl:output` tiene el valor HTML o si selecciona de forma predeterminada el formato de salida HTML, los caracteres especiales del archivo XML o XSLT se insertan en el documento HTML como referencias de caracteres HTML. Por ejemplo, el carácter ` ` (la referencia de carácter decimal para un espacio de no separación) se inserta como ` ` en el código HTML.

XSLT 2.0

Temas de este apartado:

- Especificaciones con las que cumple el motor
- Compatibilidad con versiones antiguas
- Espacios de nombres
- Compatibilidad con esquemas
- Comportamiento propio de esta implementación

Especificaciones
El motor XSLT 2.0 de MapForce cumple con la recomendación XSLT 2.0 del 23 de enero de 2007 y la recomendación XPath 2.0 del 14 de diciembre de 2010, ambas del W3C.

 © 2014 Altova GmbH

Compatibilidad con versiones antiguas

El motor XSLT 2.0 es compatible con versiones previas. Esto solamente es relevante cuando se utiliza el motor XSLT 2.0 para procesar una hoja de estilos XSLT 1.0. Tenga en cuenta que los resultados obtenidos con el motor XSLT 1.0 pueden ser diferentes a los obtenidos con el motor XSLT 2.0 en modo de compatibilidad con versiones antiguas.

Espacios de nombres

En su hoja de estilos XSLT 2.0 debe declarar estos espacios de nombres para poder usar los constructores de tipo y las funciones disponibles en XSLT 2.0. Los prefijos que aparecen a continuación son los que se suelen usar, pero puede usar otros prefijos si quiere.

Espacio de nombres	Prefijo	URI del espacio de nombres
Tipos XML Schema	`xs:`	`http://www.w3.org/2001/XMLSchema`
Funciones XPath 2.0	`fn:`	`http://www.w3.org/2005/xpath-functions`

Estos espacios de nombres se suelen declarar en el elemento `xsl:stylesheet` o en el elemento `xsl:transform`:

```
<xsl:stylesheet version="2.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:fn="http://www.w3.org/2005/xpath-functions"
    ...
</xsl:stylesheet>
```

Es necesario tener en cuenta que:

- El motor XSLT 2.0 utiliza el espacio de nombres Funciones XPath 2.0 y XQuery 1.0 como **espacio de nombres de funciones predeterminado**. Esto significa que puede usar funciones XPath 2.0 y XSLT 2.0 en su hoja de estilos sin prefijos. Si declara el espacio de nombres Funciones XPath 2.0 en su hoja de estilos con un prefijo, podrá usar el prefijo asignado en la declaración.
- Cuando se usan constructores de tipo y tipos del espacio de nombres XML Schema, el prefijo utilizado en la declaración de espacio de nombres se debe usar en la llamada al constructor de tipo (por ejemplo, `xs:date`).
- Algunas funciones XPath 2.0 se llaman igual que algunos tipos de datos de XML Schema. Por ejemplo, las funciones XPath `fn:string` y `fn:boolean` y los tipos de datos de XML Schema `xs:string` y `xs:boolean`. Por tanto, si usa la expresión `string('Hello')`, la expresión se evalúa como `fn:string('Hello')` y no como `xs:string('Hello')`.

Compatibilidad con esquemas
El motor XSLT 2.0 está preparado para esquemas de modo que puede usar tipos de esquema definidos por el usuario y la instrucción `xsl:validate`.

Comportamiento propio de esta implementación
Más abajo puede ver cómo se ocupa el motor XSLT 2.0 de algunos aspectos del comportamiento de las funciones XSLT 2.0 relacionadas con esta implementación.

xsl:result-document
También son compatibles estas codificaciones específicas de Altova: `x-base16tobinary` y `x-base64tobinary`.

function-available
Esta función mira si hay funciones del ámbito disponibles (XSLT 2.0, XPath 2.0 y funciones de extensión).

unparsed-text
El atributo `href` acepta (i) rutas de acceso relativas para archivos que estén en la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles estas codificaciones específicas de Altova: `x-binarytobase16` y `x-binarytobase64`.

unparsed-text-available
El atributo `href` acepta (i) rutas de acceso relativas para archivos que estén en la carpeta del URI base y (ii) rutas de acceso absolutas con o sin el protocolo `file://`. También son compatibles estas codificaciones específicas de Altova: `x-binarytobase16` y `x-binarytobase64`.

Nota: estos valores de codificación estaban implementados en el ya descatalogado AltovaXML pero ya no se utilizan (son obsoletos): `base16tobinary`, `base64tobinary`, `binarytobase16` y `binarytobase64`.

XQuery 1.0

Temas de este apartado:

- Especificaciones con las que cumple el motor
- Compatibilidad con esquemas
- Codificación
- Espacios de nombres
- Fuentes XML y validación
- Comprobación de tipos estática y dinámica
- Módulos biblioteca
- Módulos externos
- Intercalaciones
- Precisión de datos numéricos
- Compatibilidad con instrucciones XQuery

 © 2014 Altova GmbH

Especificaciones compatibles
El motor XQuery 1.0 de MapForce cumple con la recomendación XQuery 1.0 del 14 de diciembre de 2010 del W3C. El estándar XQuery concede libertad a la hora de implementar muchas características. A continuación explicamos cómo se implementaron estas características en el motor XQuery 1.0 de MapForce.

Compatibilidad con esquemas
El motor XQuery 1.0 está preparado para esquemas.

Codificación
El motor XQuery 1.0 es compatible con las codificaciones de caracteres UTF-8 y UTF-16.

Espacios de nombres
Se predefinen estos URI de espacios de nombres y sus enlaces asociados.

Espacio de nombres	Prefijo	URI del espacio de nombres
Tipos XML Schema	`xs:`	`http://www.w3.org/2001/XMLSchema`
Schema instance	`xsi:`	`http://www.w3.org/2001/XMLSchema-instance`
Funciones integradas	`fn:`	`http://www.w3.org/2005/xpath-functions`
Funciones locales	`local:`	`http://www.w3.org/2005/xquery-local-functions`

Es importante tener en cuenta que:

- El motor XQuery 1.0 entiende que los prefijos de la tabla anterior están enlazados con los correspondientes espacios de nombres.
- Como el espacio de nombres de funciones integradas (ver tabla) es el espacio de nombres de funciones predeterminado de XQuery, no es necesario usar el prefijo `fn:` cuando se invocan funciones integradas (por ejemplo, `string("Hello")` llamará a la función `fn:string`). No obstante, el prefijo `fn:` se puede utilizar para llamar a una función integrada sin necesidad de declarar el espacio de nombres en el prólogo de la consulta (por ejemplo: `fn:string("Hello")`).
- Puede cambiar el espacio de nombres de funciones predeterminado declarando la expresión `default function namespace` en el prólogo de la consulta.
- Cuando use tipos del espacio de nombres XML Schema, puede usar el prefijo `xs:` sin necesidad de declarar los espacios de nombres de forma explícita ni enlazar estos prefijos a los espacios de nombres en el prólogo de la consulta. (Ejemplo: `xs:date` y

xs:yearMonthDuration.) Si quiere usar otros prefijos para el espacio de nombres de XML Schema, estos se deben declarar en el prólogo de la consulta. (Ejemplo: declare namespace alt = "http://www.w3.org/2001/XMLSchema"; alt:date("2004-10-04").)

- Recuerde que los tipos de datos untypedAtomic, dayTimeDuration y yearMonthDuration se movieron del espacio de nombres XPath Datatypes al espacio de nombres XML Schema (es decir, ahora es xs:yearMonthDuration.)

Si se asignaron mal los espacios de nombres para funciones, constructores de tipo, pruebas de nodo, etc., se emite un error. Sin embargo, recuerde que algunas funciones se llaman igual que los tipos de datos de esquema (p. ej. fn:string y fn:boolean.) (Se definen xs:string y xs:boolean.) El prefijo del espacio de nombres determina si se usa la función o el constructor de tipo.

Documento XML de origen y validación

Los documentos XML que se utilizan para ejecutar un documento XQuery con el motor XQuery 1.0 deben tener un formato XML correcto. Sin embargo, no es necesario que sean válidos con respecto a un esquema XML. Si el archivo no es válido, el archivo no válido se carga sin información de esquema. Si el archivo XML está asociado a un esquema externo y es válido con respecto a dicho esquema, se genera información posterior a la validación de esquema, que se utilizará para evaluar la consulta.

Comprobación de tipos estática y dinámica

En la fase de análisis estático se revisan aspectos de la consulta como la sintaxis, si existen referencias externas (p. ej. para módulos), si las funciones y variables que se invocan están definidas, etc. Si se detecta un error en la fase de análisis estático, se notifica y la ejecución se interrumpe.

La comprobación dinámica de tipos se realiza en tiempo de ejecución, cuando la consulta se ejecuta. Si un tipo no es compatible con los requisitos de una operación, se emite un error. Por ejemplo, la expresión xs:string("1") + 1 devuelve un error porque la operación de suma no se puede llevar a cabo en un operando de tipo xs:string.

Módulos biblioteca

Los módulos biblioteca almacenan funciones y variables para poder volver a utilizarlas. El motor XQuery 1.0 es compatible con el uso de módulos almacenados en un **solo archivo XQuery externo**. Dicho archivo de módulo debe incluir una declaración module en su prólogo que apunte a un espacio de nombres de destino Por ejemplo:

```
module namespace libns="urn:module-library";
declare variable $libns:company := "Altova";
declare function libns:webaddress() { "http://www.altova.com" };
```

 © 2014 Altova GmbH

Todas las funciones y variables declaradas en el módulo pertenecen al espacio de nombres asociado al módulo. El módulo se importa en un archivo XQuery con la instrucción `import module` del prólogo de la consulta. La instrucción `import module` solamente importa funciones y variables declaradas directamente en el archivo de módulo biblioteca. Por ejemplo:

```
import module namespace modlib = "urn:module-library" at "modulefilename.xq";

if        ($modlib:company = "Altova")
then       modlib:webaddress()
else       error("No match found.")
```

Funciones externas

Las funciones externas son incompatibles con el motor XQuery 1.0, es decir, todas las expresiones que usen la palabra clave `external`. Por ejemplo:

```
declare function hoo($param as xs:integer) as xs:string external;
```

Intercalaciones

La intercalación predeterminada es la intercalación de puntos de código Unicode, que compara las cadenas de texto según sus puntos de código Unicode. Otras intercalaciones compatibles son las intercalaciones ICU que se enumeran aquí. Para usar una intercalación concreta, indique su URI tal y como aparece en la lista de intercalaciones compatibles. Las comparaciones de cadenas de texto, incluidas las comparaciones para las funciones `fn:max` y `fn:min`, se harán según la intercalación especificada. Si no se indica la opción de intercalación, se utiliza la intercalación de puntos de código Unicode predeterminada.

Precisión de tipos numéricos

- El tipo de datos `xs:integer` es de precisión arbitraria, es decir, puede representar un número de dígitos cualquiera.
- El tipo de datos `xs:decimal` tiene un límite de 20 dígitos después del punto decimal.
- Los tipos de datos `xs:float` y `xs:double` tienen una precisión limitada de 15 dígitos.

Compatibilidad con instrucciones XQuery

La instrucción `Pragma` no es compatible. Si se encuentra, se ignora y en su lugar se evalúa la expresión de reserva.

16.1.2 Funciones XSTL y XPath/XQuery

Esta sección enumera las funciones de extensión de Altova y otras funciones de extensión que se pueden utilizar con expresiones XPath y XQuery. Las funciones de extensión de Altova se pueden usar con los motores XSLT y XQuery de Altova y ofrecen algunas funciones más aparte de las que están disponibles en las bibliotecas de funciones definidas en los estándares del W3C.

Aspectos generales
Es necesario tener en cuenta estos puntos generales:

- A las funciones de las bibliotecas de funciones `core` definidas en las especificaciones W3C se les puede llamar sin un prefijo. Esto se debe a que los motores XSLT y XQuery leen funciones sin prefijo como si pertenecieran a un espacio de nombres de funciones predeterminado, que es el que se especifica en las especificaciones de las funciones XPath y XQuery`http://www.w3.org/2005/xpath-functions`. Si este espacio de nombres se declara explícitamente en un documento XSLT o XQuery, el prefijo utilizado en la declaración de espacio de nombres también se puede usar en el nombre de las funciones.
- Por lo general, si una función espera como argumento una secuencia de un elemento y se suministra una secuencia de más de un elemento, entonces se devuelve un error.
- Todas las comparaciones de cadena se realizan usando la intercalación de puntos de código Unicode.
- Los resultados que son QName se serializan de esta forma `[prefijo:]nombrelocal`.

Precisión de xs:decimal
El término *precisión* hace referencia al número de dígitos del número y la especificación exige un mínimo de 18 dígitos. Para las operaciones de división que dan un resultado de tipo `xs:decimal`, la precisión es de 19 dígitos después del punto decimal sin redondear.

Uso horario implícito
Cuando hay que comparar dos valores `date`, `time` o `dateTime`, es necesario conocer el uso horario de los valores que se deben comparar. Cuando el uso horario no se conoce de forma explícita, se usa el uso horario implícito, que se toma del reloj del sistema. Para probar cuál es su valor puede utilizar la función `fn:implicit-timezone()`.

Intercalaciones
La intercalación predeterminada es la intercalación de puntos de código Unicode, que compara las cadenas de texto según sus puntos de código Unicode. Otras intercalaciones compatibles son las intercalaciones ICU que aparecen más abajo. Para usar una intercalación determinada, aporte su URI tal y como aparece en la lista de intercalaciones compatibles (*ver más abajo*). Las comparaciones de cadenas de texto (incluidas las que usan las funciones `fn:max` y `fn:min`) se harán según la intercalación especificada. Si no se especifica la opción de intercalación, se usa la intercalación predeterminada de puntos de código Unicode.

Idioma	Identificadores URI

 © 2014 Altova GmbH

da: danés	da_DK
de: alemán	de_AT, de_BE, de_CH, de_DE, de_LI, de_LU
en: inglés	en_AS, en_AU, en_BB, en_BE, en_BM, en_BW, en_BZ, en_CA, en_GB, en_GU, en_HK, en_IE, en_IN, en_JM, en_MH, en_MP, en_MT, en_MU, en_NA, en_NZ, en_PH, en_PK, en_SG, en_TT, en_UM, en_US, en_VI, en_ZA, en_ZW
es: español	es_419, es_AR, es_BO, es_CL, es_CO, es_CR, es_DO, es_EC, es_ES, es_GQ, es_GT, es_HN, es_MX, es_NI, es_PA, es_PE, es_PR, es_PY, es_SV, es_US, es_UY, es_VE
fr: francés	fr_BE, fr_BF, fr_BI, fr_BJ, fr_BL, fr_CA, fr_CD, fr_CF, fr_CG, fr_CH, fr_CI, fr_CM, fr_DJ, fr_FR, fr_GA, fr_GN, fr_GP, fr_GQ, fr_KM, fr_LU, fr_MC, fr_MF, fr_MG, fr_ML, fr_MQ, fr_NE, fr_RE, fr_RW, fr_SN, fr_TD, fr_TG
it: italiano	it_CH, it_IT
ja: jaoonés	ja_JP
nb: noruego bokmål	nb_NO
nl: holandés	nl_AW, nl_BE, nl_NL
nn: noruego nynorsk	nn_NO
pt: portugués	pt_AO, pt_BR, pt_GW, pt_MZ, pt_PT, pt_ST
ru: ruso	ru_MD, ru_RU, ru_UA
sv: sueco	sv_FI, sv_SE

Eje del espacio de nombres
El eje del espacio de nombres se dejó de utilizar en XPath 2.0. Sin embargo, los ejes de espacio de nombres son compatibles con la aplicación. Para acceder a la información sobre el espacio de nombres con mecanismos de XPath 2.0, utilice las funciones `fn:in-scope-prefixes()`, `fn:namespace-uri()` y `fn:namespace-uri-for-prefix()`.

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. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Las funciones definidas en las especificaciones XPath/XQuery Functions del W3C se pueden usar en (i) expresiones XPath en contextos XSLT y en (ii) expresiones XQuery en documentos XQuery. En esta documentación las funciones que se pueden usar en el primer contexto (XPath en XSLT) llevan el símbolo **XP** y se les llama funciones XPath. Las funciones que se pueden usar en contextos XQuery llevan el símbolo **XQ** y funcionan como funciones XQuery. Las especificaciones XSLT del W3C también definen funciones que se pueden usar en expresiones XPath en documentos XSLT. Estas funciones llevan el símbolo **XSLT** y se les denomina funciones XSLT. Por cada función se indica en qué versión de XPath/XQuery y XSLT se puede usar (ver

símbolos más abajo). Las funciones de las bibliotecas de funciones XPath/XQuery y XSLT aparecen sin prefijo. Las funciones de extensión de otras bibliotecas, como las funciones de extensión de Altova, aparecen con un prefijo.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3

Funciones XSLT

Estas funciones solo se pueden usar en expresiones XPath en contextos XSLT (igual que las funciones XSLT 2.0 `current-group()` o `key()`). Estas funciones no funcionan en otro contexto que no sea XSLT (por ejemplo, en contextos XQuery). Recuerde que las funciones XSLT para XBRL solamente se pueden usar en las ediciones de los productos de Altova que son compatibles con XBRL.

Funciones XPath/XQuery

Las funciones XPath/XQuery (generales, de fecha y hora y de cadena) se pueden usar tanto en expresiones XPath en contextos XSLT como en expresiones XQuery.

Funciones XSLT

Las **funciones de extensión XSLT** pueden utilizarse en expresiones XPath en contextos XSLT y no funcionan en contextos que no sean XSLT (por ejemplo, en contextos XQuery).

Nota sobre espacios de nombres y lenguajes
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. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3

Funciones estándar

▾ **`distinct-nodes [altova:]`**

 © 2014 Altova GmbH

altova:distinct-nodes(*node())** como **node()*** XSLT1 XSLT2 XSLT3

Toma un conjunto de nodos como entrada y devuelve el mismo conjunto menos los nodos que tengan el mismo valor (es decir, devuelve los nodos que son únicos). La comparación se hace con la función XPath/XQuery `fn:deep-equal`.

Ejemplo

- **altova:distinct-nodes**(`country`) devuelve todos los nodos secundarios `country` excepto los que tengan el mismo valor.

evaluate [altova:]

altova:evaluate(ExpresiónXPath *as xs:string*[, **ValorDe$p1, ... ValorDe$pN**]**)**
XSLT1 XSLT2 XSLT3

Toma una expresión XPath, pasada como cadena, como argumento obligatorio. Devuelve el resultado de la expresión evaluada. Por ejemplo, **altova:evaluate('//Name[1]')** devuelve el contenido del primer elemento `Name` del documento. Observe que para pasar la expresión `//Name[1]` como cadena basta con ponerla entre comillas simples.

La función `altova:evaluate` puede tomar más argumentos, que son los valores de las variables del ámbito que se llaman `p1`, `p2`, `p3`... `pN`. Recuerde que (i) las variables deben definirse con nombres de tipo `pX`, siendo `X` un entero; (ii) los argumentos de la función `altova:evaluate` (*ver firma más abajo*), a partir del segundo argumento, ofrecen los valores de las variables, correspondiendo la secuencia de argumentos a la secuencia numérica de variables: `p1` corresponde a `pN` y el segundo argumento será el valor de la variable `p1`, el tercer argumento al de la variable `p2`, y así sucesivamente; (iii) los valores de las variables deben ser de tipo `item*`.

Ejemplo

```
<xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
<xsl:value-of select="altova:evaluate($xpath, 10, 20, 'hi')" />
```

da el resultado `"hi 20 10"`

En el ejemplo anterior puede observar que:

- El segundo argumento de la expresión `altova:evaluate` es el valor asignado a la variable `$p1`, el tercer argumento es el valor asignado a la variable `$p2` y así sucesivamente.
- Observe que el cuarto argumento de la función es un valor de cadena porque va entre comillas simples.
- El atributo `select` del elemento `xs:variable` suministra la expresión XPath. Como esta expresión debe ser de tipo `xs:string`, se pone entre comillas simples.

Más ejemplos

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, //Name[1])" />
  ```
  El resultado es el valor del primer elemento `Name`.

- ```
  <xsl:variable name="xpath" select="'$p1'" />
  <xsl:value-of select="altova:evaluate($xpath, '//Name[1]')" />
  ```
  El resultado es "//Name[1]"

La función de extensión `altova:evaluate()` es muy práctica cuando una expresión XPath

de la hoja de estilos XSLT contiene partes que se deben evaluar de forma dinámica. Por ejemplo, imagine que el usuario selecciona un criterio de ordenación y este criterio se almacena en el atributo `UserReq/@sortkey`. En la hoja de estilos podría tener esta expresión:

```
<xsl:sort select="altova:evaluate(../UserReq/@sortkey)"
order="ascending"/>
```

La función `altova:evaluate()` lee el atributo `sortkey` del elemento secundario `UserReq` del primario del nodo de contexto. Imagine que el valor del atributo `sortkey` es `Price`. En ese caso, la función `altova:evaluate()` devuelve `Price`, que se convierte en el valor del atributo `select`:

```
<xsl:sort select="Price" order="ascending"/>
```

Si esta instrucción `sort` aparece dentro del contexto de un elemento llamado `Order`, entonces los elementos `Order` se ordenan según el valor de los secundarios `Price`. Otra opción es que, si el valor de `@sortkey` fuera `Date`, por ejemplo, entonces los elementos `Order` se ordenarían según el valor de los secundarios `Date`. Es decir, el criterio de ordenación para `Order` se selecciona del atributo `sortkey` en tiempo de ejecución. Esto no sería posible con una expresión como:

```
<xsl:sort select="../UserReq/@sortkey" order="ascending"/>
```

En este caso, el criterio de ordenación sería el propio atributo `sortkey`, no `Price` ni `Date` (ni otro contenido actual de `sortkey`).

Nota: el contexto estático incluye espacios de nombres, tipos y funciones (pero no variables) del entorno de llamada. El URI base y el espacio de nombres predeterminado se heredan.

Más ejemplos

- Variables estáticas: `<xsl:value-of select="$i3, $i2, $i1" />`
  El resultado es los valores de las tres variables.
- Expresión XPath dinámica con variables dinámicas:
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="altova:evaluate($xpath, 10, 20, 30)" />
  ```
  El resultado es "30 20 10"
- Expresión XPath dinámica sin variables dinámicas:
  ```
  <xsl:variable name="xpath" select="'$p3, $p2, $p1'" />
  <xsl:value-of select="altova:evaluate($xpath)" />
  ```
  Error: no se definió la variable para $p3.

encode-for-rtf [altova:]

altova:encode-for-rtf(input *as xs:string*, **preserveallwhitespace** *as xs:boolean*, **preservenewlines** *as xs:boolean*) como **xs:string** XSLT2 XSLT3

Convierte la cadena de entrada en código para RTF. Los espacios en blanco y las líneas nuevas se conservan o no dependiendo del valor booleano especificado para los correspondientes parámetros.

[Subir]

Funciones XBRL
Las funciones XBRL de Altova solo funcionan en las ediciones de los productos de Altova que son compatibles con XBRL.

xbrl-footnotes [altova:]

altova:xbrl-footnotes(*node()***)** como **node()*** XSLT2 XSLT3
Toma un nodo como argumento de entrada y devuelve el conjunto de nodos de nota al pie XBRL al que hace referencia el nodo de entrada.

xbrl-labels [altova:]

altova:xbrl-labels(*xs:QName, xs:string***)** como **node()*** XSLT2 XSLT3
Toma dos argumentos de entrada: un nombre de nodo y la ubicación del archivo de taxonomía en el que está el nodo. La función devuelve los nodos de etiqueta XBRL asociados al nodo de entrada.

[Subir]

Funciones XPath/XQuery: fecha y hora

Las funciones de extensión de fecha y hora de Altova se pueden usar en expresiones XPath y XQuery y permiten procesar datos almacenados en tipos de datos XML Schema de fecha y hora. Estas funciones se pueden usar con los **motores XPath 3.0 y XQuery 3.0** de Altova y están disponibles en contextos XPath/XQuery.

Nota sobre espacios de nombres y lenguajes
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. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3

Funciones agrupadas según su funcionalidad

- Agregar una duración a xs:dateTime y devolver xs:dateTime
- Agregar una duración a xs:date y devolver xs:date
- Agregar una duración a xs:time y devolver xs:time
- Quitar la zona horaria de las funciones que generan la fecha/hora actual
- Devolver el día de la semana de una fecha como número entero
- Devolver el número de semana de una fecha como número entero
- Generar la fecha, la hora y el tipo de duración a partir de los componentes léxicos de cada

tipo
- Construir un tipo date, dateTime o a partir de la cadena de entrada
- Funciones para calcular la edad

Funciones por orden alfabético

altova:add-days-to-date
altova:add-days-to-dateTime
altova:add-hours-to-dateTime
altova:add-hours-to-time
altova:add-minutes-to-dateTime
altova:add-minutes-to-time
altova:add-months-to-date
altova:add-months-to-dateTime
altova:add-seconds-to-dateTime
altova:add-seconds-to-time
altova:add-years-to-date
altova:add-years-to-dateTime
altova:age
altova:age-details
altova:build-date
altova:build-duration
altova:build-time
altova:current-dateTime-no-TZ
altova:current-date-no-TZ
altova:current-time-no-TZ
altova:parse-date
altova:parse-dateTime
altova:parse-time
altova:weekday-from-date
altova:weekday-from-dateTime
altova:weeknumber-from-date
altova:weeknumber-from-dateTime

[Subir]

Agregar una duración a `xs:dateTime` XP3 XQ3

Estas funciones sirven para agregar una duración a **`xs:dateTime`** y devuelven **`xs:dateTime`**. El tipo `xs:dateTime` tiene el formato `CCYY-MM-DDThh:mm:ss.sss`. Se trata de la concatenación de los formatos `xs:date` y `xs:time` separados por la letra `T`. Si quiere, puede usar un sufijo de zona horaria`+01:00 (por ejemplo)`.

add-years-to-dateTime [altova:]

altova:add-years-to-dateTime(FechaHora *as xs:dateTime*, **Años** *as xs:integer***)** como **xs:dateTime** XP3 XQ3

Añade una duración en años un valor de fecha y hora. El segundo argumento es el número de años que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2024-01-15T14:00:00

 © 2014 Altova GmbH

- **altova:add-years-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -4) devuelve 2010-01-15T14:00:00

add-months-to-dateTime [altova:]

altova:add-months-to-dateTime(FechaHora *as xs:dateTime*, **Meses** *as xs:integer***)** como **xs:dateTime** XP3 XQ3

Añade una duración en meses a un valor de fecha y hora. El segundo argumento es el número de meses que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

Ejemplos

- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2014-11-15T14:00:00
- **altova:add-months-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -2) devuelve 2013-11-15T14:00:00

add-days-to-dateTime [altova:]

altova:add-days-to-dateTime(FechaHora *as xs:dateTime*, **Días** *as xs:integer***)** como **xs:dateTime** XP3 XQ3

Añade una duración en días a un valor de fecha y hora. El segundo argumento es el número de días que se deben añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

Ejemplos

- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), 10) devuelve 2014-01-25T14:00:00
- **altova:add-days-to-dateTime**(xs:dateTime("2014-01-15T14:00:00"), -8) devuelve 2014-01-07T14:00:00

add-hours-to-dateTime [altova:]

altova:add-hours-to-dateTime(FechaHora *as xs:dateTime*, **Horas** *as xs:integer***)** como **xs:dateTime** XP3 XQ3

Añade una duración en horas a un valor de fecha y hora. El segundo argumento es el número de horas que se deben añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo xs:dateTime.

Ejemplos

- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), 10) devuelve 2014-01-15T23:00:00
- **altova:add-hours-to-dateTime**(xs:dateTime("2014-01-15T13:00:00"), -8) devuelve 2014-01-15T05:00:00

add-minutes-to-dateTime [altova:]

altova:add-minutes-to-dateTime(FechaHora *as xs:dateTime*, **Minutos** *as xs:integer***)** como **xs:dateTime** XP3 XQ3

Añade una duración en minutos a un valor de fecha y hora. El segundo argumento es el número de minutos que se debe añadir al valor de fecha y hora dado como primer

argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), 45) devuelve 2014-01-15T14:55:00
- **altova:add-minutes-to-dateTime**(xs:dateTime("2014-01-15T14:10:00"), -5) devuelve 2014-01-15T14:05:00

add-seconds-to-dateTime [altova:]

altova:add-seconds-to-dateTime(FechaHora *as xs:dateTime*, **Segundos** *as xs:integer*) como **xs:dateTime** XP3 XQ3

Añade una duración en segundos a un valor de fecha y hora. El segundo argumento es el número de segundos que se debe añadir al valor de fecha y hora dado como primer argumento. El resultado es de tipo `xs:dateTime`.

Ejemplos

- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), 20) devuelve 2014-01-15T14:00:30
- **altova:add-seconds-to-dateTime**(xs:dateTime("2014-01-15T14:00:10"), -5) devuelve 2014-01-15T14:00:05

[Subir]

Agregar una duración a xs:date XP3 XQ3

Estas funciones agregan una duración a **xs:date** y devuelven **xs:date**. El tipo `xs:date` tiene el formato `CCYY-MM-DD`.

add-years-to-date [altova:]

altova:add-years-to-date(Fecha *as xs:date*, **Años** *as xs:integer*) como **xs:date** XP3 XQ3

Añade una duración en años a una fecha. El segundo parámetro es el número de años que se debe añadir a la fecha dada como primer argumento. El resultado es de tipo `xs:date`.

Ejemplos

- **altova:add-years-to-date**(xs:date("2014-01-15"), 10) devuelve 2024-01-15
- **altova:add-years-to-date**(xs:date("2014-01-15"), -4) devuelve 2010-01-15

add-months-to-date [altova:]

altova:add-months-to-date(Fecha *as xs:date*, **Meses** *as xs:integer*) como **xs:date** XP3 XQ3

Añade una duración en meses a una fecha. El segundo argumento es el número de meses que se debe añadir a la fecha dada como primer argumento. El resultado es de tipo `xs:date`.

Ejemplos

- **altova:add-months-to-date**(xs:date("2014-01-15"), 10) devuelve 2014-11-15
- **altova:add-months-to-date**(xs:date("2014-01-15"), -2) devuelve 2013-11-15

 © 2014 Altova GmbH

add-days-to-date [altova:]

altova:add-days-to-date(Fecha *as xs:date*, **Días** *as xs:integer*) como **xs:date**
XP3 XQ3
Añade una duración en días a una fecha. El segundo argumento es el número de días que se deben añadir a la fecha dad como primer argumento. El resultado es de tipo `xs:date`.

Ejemplos

- **altova:add-days-to-date**(xs:date("2014-01-15"), 10) devuelve 2014-01-25
- **altova:add-days-to-date**(xs:date("2014-01-15"), -8) devuelve 2014-01-07

[Subir]

Agregar una duración a xs:time XP3 XQ3

Estas funciones agregan una duración a **xs:time** y devuelven **xs:time**. El tipo `xs:time` tiene un formato léxico de este tipo `hh:mm:ss.sss`. Si quiere, puede añadir un sufijo de zona horaria. La letra Z indica (UTC). Las demás zonas horarias se representan con la diferencia que hay entre ellas y la zona UTC: `+hh:mm` o `-hh:mm`. Si falta el valor de zona horaria, se entiende que se desconoce (no se da por hecho que es UTC)

add-hours-to-time [altova:]

altova:add-hours-to-time(Hora *as xs:time*, **Horas** *as xs:integer*) como **xs:time**
XP3 XQ3
Añade una duración en horas a una hora. El segundo argumento es el número de horas que se debe añadir a la hora dada como primer argumento. El resultado es de tipo `xs:time`.

Ejemplos

- **altova:add-hours-to-time**(xs:time("11:00:00"), 10) devuelve 21:00:00
- **altova:add-hours-to-time**(xs:time("11:00:00"), -7) devuelve 04:00:00

add-minutes-to-time [altova:]

altova:add-minutes-to-time(Hora *as xs:time*, **Minutos** *as xs:integer*) como **xs:time** XP3 XQ3
Añade una duración en minutos a una hora. El segundo argumento es el número de minutos que se debe añadir a la hora dada como primer argumento. El resultado es de tipo `xs:time`.

Ejemplos

- **altova:add-minutes-to-time**(xs:time("14:10:00"), 45) devuelve 14:55:00
- **altova:add-minutes-to-time**(xs:time("14:10:00"), -5) devuelve 14:05:00

add-seconds-to-time [altova:]

altova:add-seconds-to-time(Hora *as xs:time*, **Segundos** *as xs:integer*) como **xs:time** XP3 XQ3
Añade una duración en segundos a una hora. El segundo argumento es el número de segundos que se debe añadir a la hora dada como primer argumento. El resultado es de tipo `xs:time`. El componente **Segundos** puede estar comprendido entre `0` y `59.999`.

Ejemplos

- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20) devuelve 14:00:20
- **altova:add-seconds-to-time**(xs:time("14:00:00"), 20.895) devuelve 14:00:20.895

[Subir]

Quitar la parte de zona horaria de los tipos de datos date/time XP3 XQ3

Estas funciones quitan la zona horaria de los valores **xs:dateTime**, **xs:date** o **xs:time** actuales. Tenga en cuenta que la diferencia entre xs:dateTime y xs:dateTimeStamp es que en esta última la parte de zona horaria es obligatoria (mientras que en la primera es opcional). Es decir, el formato de un valor xs:dateTimeStamp puede ser CCYY-MM-DDThh:mm:ss.sss±hh:mm o CCYY-MM-DDThh:mm:ss.sssZ. Si la fecha y la hora se leen del reloj del sistema como xs:dateTimeStamp, la función current-dateTime-no-TZ() se puede usar para quitar la zona horaria.

current-dateTime-no-TZ [altova:]

altova:current-dateTime-no-TZ() como **xs:dateTime** XP3 XQ3

Esta función no toma ningún argumento. Quita la parte de zona horaria de current-dateTime() (que es la fecha y hora actual según el reloj del sistema) y devuelve un valor de tipo xs:dateTime.

Ejemplos

Si la fecha y hora actual es **2014-01-15T14:00:00+01:00**:

- **altova:current-dateTime-no-TZ**() devuelve 2014-01-15T14:00:00

current-date-no-TZ [altova:]

altova:current-date-no-TZ() como **xs:date** XP3 XQ3

Esta función no toma ningún argumento. Quita la parte de zona horaria de la función current-date() (que es la fecha actual según el reloj del sistema) y devuelve un valor de tipo xs:date.

Ejemplos

Si la fecha actual es **2014-01-15+01:00**:

- **altova:current-date-no-TZ**() devuelve 2014-01-15

current-time-no-TZ [altova:]

altova:current-time-no-TZ() as **xs:time** XP3 XQ3

Esta función no toma ningún argumento. Quita la parte de zona horaria de current-time() (que es la hora actual según el reloj del sistema) y devuelve un valor de tipo xs:time.

Ejemplos

Si la hora actual es **14:00:00+01:00**:

- **altova:current-time-no-TZ**() devuelve 14:00:00

[Subir]

 © 2014 Altova GmbH

Obtener el día de la semana de `xs:dateTime` o `xs:date` XP3 XQ3

Estas funciones obtienen el día de la semana (como entero) de `xs:dateTime` o `xs:date`. Los días de la semana se numeran del `1` al `7` (usando el formato EE UU, es decir `Domingo =1`). En el formato europeo la semana empieza el lunes (es decir, `Lunes=1`). Para establecer el formato EE UU (`Domingo=1`) use el entero `0` allí donde se acepte un entero para indicar el formato.

weekday-from-dateTime [altova:]

altova:weekday-from-dateTime(FechaHora *as xs:dateTime***)** como **xs:integer** XP3 XQ3

Toma una fecha con hora como único argumento y devuelve el día de la semana de la fecha dada como número entero. Los días de la semana se numeran del 1 al 7 empezando por `Domingo=1`. Si necesita usar el formato europeo (donde `Lunes=1`), utilice la otra firma de esta función (*ver más abajo*).

Ejemplos

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00")) devuelve `2`, lo cual significa "Lunes".

altova:weekday-from-dateTime(DateTime *as xs:dateTime*, **Formato** *as xs:integer***)** como **xs:integer** XP3 XQ3

Toma una fecha con hora como primer argumento y devuelve el día de la semana de la fecha dada como número entero. Los días de la semana se numeran del 1 al 7 empezando por `Lunes=1`. Si el segundo argumento (**Formato**) es `0`, entonces los días de la semana se numeran del 1 al 7 empezando por `Domingo=1`. Si el segundo argumento es un entero distinto de `0`, entonces `Lunes=1`. Si falta el segundo argumento, la función se lee como en la firma anterior (*ver más arriba*).

Ejemplos

- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 1) devuelve `1`, lo cual significa "Lunes".
- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 4) devuelve `1`, lo cual significa "Lunes".
- **altova:weekday-from-dateTime**(xs:dateTime("2014-02-03T09:00:00"), 0) devuelve `2`, lo cual significa "Lunes".

weekday-from-date [altova:]

altova:weekday-from-date(Fecha *as xs:date***)** como **xs:integer** XP3 XQ3

Toma una fecha como único argumento y devuelve el día de la semana de la fecha dada como número entero. Los días de la semana se numeran del 1 al 7 empezando por `Domingo=1`. Si necesita usar el formato europeo (donde `Lunes=1`), utilice la otra firma de esta función (*ver más abajo*).

Ejemplos

- **altova:weekday-from-date**(xs:date("2014-02-03+01:00")) devuelve `2`, lo cual significa "Lunes".

altova:weekday-from-date(Fecha *as xs:date*, **Formato** *as xs:integer***)** como **xs:integer** XP3 XQ3

Toma una fecha como primer argumento y devuelve el día de la semana de la fecha dada

como número entero. Los días de la semana se numeran del 1 al 7 empezando por `Lunes=1`. Si el segundo argumento (**Formato**) es `0`, entonces los días de la semana se numeran del 1 al 7 empezando por `Domingo=1`. Si el segundo argumento es un entero distinto de `0`, entonces `Lunes=1`. Si falta el segundo argumento, la función se lee como en la firma anterior (*ver más arriba*).

Ejemplos

- **altova:weekday-from-date**(xs:date("2014-02-03"), 1) devuelve 1, lo cual significa "Lunes"
- **altova:weekday-from-date**(xs:date("2014-02-03"), 4) devuelve 1, lo cual significa "Lunes"
- **altova:weekday-from-date**(xs:date("2014-02-03"), 0) devuelve 2, lo cual significa "Lunes"

[Subir]

Devolver el número de semana de xs:dateTime o xs:date XP2 XQ1 XP3 XQ3

Estas funciones devuelven el número de semana (como número entero) de `xs:dateTime` o `xs:date`. El número de la semana está disponible en el formato de calendario estadounidense, europeo e islámico. La razón de que los números de semana difieran en cada uno de estos calendarios es que en cada uno de ellos se considera un día diferente para el inicio de la semana (p. ej. en el formato estadounidense el primer día de la semana es el domingo).

weeknumber-from-date [altova:]

altova:weeknumber-from-date(Fecha *como xs:date*, **Calendario** *como xs:integer***)** como **xs:integer** XP2 XQ1 XP3 XQ3

Devuelve como número entero el número de semana del argumento **Fecha** dado. El segundo argumento (**Calendario**) indica el sistema de calendario que se debe seguir.
Estos son los valores permitidos para el argumento **Calendario**:

- **0 = Calendario estadounidense** (la semana comienza el domingo)
- **1 = Calendario estándar ISO o europeo** (la semana comienza el lunes)
- **2 = Calendario islámico** (la semana comienza el sábado)

El valor predeterminado es **0**.

Ejemplos

- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 0) devuelve 13
- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 1) devuelve 12
- **altova:weeknumber-from-date**(xs:date("2014-03-23"), 2) devuelve 13
- **altova:weeknumber-from-date**(xs:date("2014-03-23")) devuelve 13

El día de la fecha de los ejemplos anteriores (2014-03-23) es un domingo. Por tanto, en este caso, el calendario estadounidense y el islámico van una semana por delante del calendario europeo.

weeknumber-from-dateTime [altova:]

altova:weeknumber-from-dateTime(FechaHora *como xs:dateTime*, **Calendario** *como*

xs:integer) como **xs:integer** XP2 XQ1 XP3 XQ3

Devuelve como entero el día de la semana del argumento **FechaHora** dado. El segundo argumento (**Calendario**) indica el sistema de calendario que se debe seguir.
Estos son los valores permitidos para el argumento **Calendario**:

- **0 = Calendario estadounidense** (la semana comienza el domingo)
- **1 = Calendario estándar ISO o europeo** (la semana comienza el lunes)
- **2 = Calendario islámico** (la semana comienza el sábado)

El valor predeterminado es **0**.

Ejemplos

- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 0) devuelve 13
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 1) devuelve 12
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00"), 2) devuelve 13
- **altova:weeknumber-from-dateTime**(xs:dateTime("2014-03-23T00:00:00")) devuelve 13

El día de dateTime de los ejemplos anteriores (2014-03-23T00:00:00) es un domingo. Por tanto, en este caso, el calendario estadounidense y el islámico van una semana por delante del calendario europeo.

[Subir]

Generar tipos de datos de fecha, hora y duración a partir de sus componentes léxicos XP3 XQ3

Estas funciones toman los componentes léxicos de los tipos de datos xs:date, xs:time y xs:duration como argumentos de entrada y los combinan para generar el tipo de datos correspondiente.

build-date [altova:]

altova:build-date(Año *as xs:integer*, **Mes** *as xs:integer*, **Fecha** *as xs:integer*) como **xs:date** XP3 XQ3

Los argumentos son el año, el mes y la fecha respectivamente. Se combinan para generar un valor de tipo xs:date. Los valores de los enteros deben estar en el intervalo de esa fecha en particular. Por ejemplo, el segundo argumento (para el mes) no puede ser mayor que 12.

Ejemplos

- **altova:build-date**(2014, 2, 03) devuelve 2014-02-03

build-time [altova:]

altova:build-time(Horas *as xs:integer*, **Minutos** *as xs:integer*, **Segundos** *as xs:integer*) como **xs:time** XP3 XQ3

El primer, segundo y tercer argumentos son la hora (0 - 23), los minutos (0 - 59) y los segundos (0 - 59) respectivamente. Se combinan para generar un valor de tipo xs:time. Los valores de los enteros deben estar dentro del intervalo correcto de esa parte de tiempo

concreta. Por ejemplo, el segundo argumento (`Minutos`) no puede ser mayor que `59`. Para añadir la parte de uso horario al valor, use la firma que aparece más abajo.

Ejemplos

- **altova:build-time**(23, 4, 57) devuelve 23:04:57

altova:build-time(Horas *as xs:integer*, **Minutos** *as xs:integer*, **Segundos** *as xs:integer*, **UsoHorario** *as xs:string*) como **xs:time** XP3 XQ3

The first, second, and third arguments are, respectively, the hour (`0` to `23`), minutes (`0` to `59`), and seconds (`0` to `59`) values. The fourth argument is a string that provides the timezone part of the value. The four arguments are combined to build a value of `xs:time` type. The values of the integers must be within the correct range of that particular time part. For example, the second (`Minutes`) argument should not be greater than `59`.

El primer, segundo y tercer argumentos son la hora (`0` - `23`), los minutos (`0` - `59`) y los segundos (`0` - `59`) respectivamente. El cuarto argumento es una cadena que aporta la parte de uso horario del valor. Los cuatro argumentos se combinan para generar un valor de tipo `xs:time`. Los valores de los enteros deben estar dentro del intervalo correcto de esa parte de tiempo concreta. Por ejemplo, el segundo argumento (`Minutos`) no puede ser mayor que `59`.

Ejemplos

- **altova:build-time**(23, 4, 57, '+1') devuelve 23:04:57+01:00

build-duration [altova:]

altova:build-duration(Años *as xs:integer*, **Meses** *as xs:integer*) como **xs:yearMonthDuration** XP3 XQ3

Toma dos argumentos para generar un valor de tipo `xs:yearMonthDuration`. El primer argumento da la parte `Years` del valor de duración, mientras que el segundo da la parte `Months`. Si el segundo (`Months`) es mayor o igual que `12`, el entero se divide por 12. El cociente se añade al primer argumento para aportar la parte `Years` del valor de duración, mientras que el resto (de la división) da la parte `Months`. Para generar una duración de tipo `xs:dayTimeDuration`, consulte la firma siguiente.

Ejemplos

- **altova:build-duration**(2, 10) devuelve P2Y10M
- **altova:build-duration**(14, 27) devuelve P16Y3M
- **altova:build-duration**(2, 24) devuelve P4Y

altova:build-duration(Días *as xs:integer*, **Horas** *as xs:integer*, **Minutos** *as xs:integer*, **Segundos** *as xs:integer*) como **xs:dayTimeDuration** XP3 XQ3

Toma cuatro argumentos y los combina para generar un valor de tipo `xs:dayTimeDuration`. El primer argumento da la parte `Days` del valor de duración, el segundo, el tercero y el cuarto dan las partes `Hours`, `Minutes` y `Seconds` respectivamente. Los tres argumentos de tiempo se convierten a un valor equivalente en cuanto a la unidad mayor siguiente y el resultado se utiliza para calcular el valor total de la duración. Por ejemplo, 72 segundos se convierte en `1M+12S` (1 minuto y 12 segundos) y este valor se usa para calcular el valor total de la duración. Para generar una duración de tipo `xs:yearMonthDuration`, consulte la firma anterior.

Ejemplos

- **altova:build-duration**(2, 10, 3, 56) devuelve P2DT10H3M56S
- **altova:build-duration**(1, 0, 100, 0) devuelve P1DT1H40M

 © 2014 Altova GmbH

- **altova:build-duration**(1, 0, 0, 3600) devuelve P1DT1H

[Subir]

Construir tipos de datos date, dateTime y time a partir de una cadena de entrada XP2 XQ1 XP3 XQ3

Estas funciones toman cadenas como argumentos y construyen tipos de datos xs:date, xs:dateTime o xs:time. La cadena de entrada se analiza para los componentes del tipo de datos en función del argumento patrón dado.

parse-date [altova:]

altova:parse-date(Fecha *como xs:string*, **PatrónFecha** *como xs:string*) como **xs:date** XP2 XQ1 XP3 XQ3

Devuelve la cadena de entrada **Fecha** como valor **xs:date**. El segundo argumento (**PatrónFecha**) indica el patrón (secuencia de componentes) de la cadena de entrada. El argumento **PatrónFecha** se describe con los especificadores que aparecen a continuación y con cualquier separador de componentes (consulte los ejemplos más abajo).

D	Fecha
M	Mes
Y	Año

El patrón **PatrónFecha** debe coincidir con el patrón de **Fecha**. Como el resultado es de tipo **xs:date**, el resultado siempre tendrá el formato léxico **YYYY-MM-DD**.

Ejemplos

- **altova:parse-date**(xs:string("06-03-2014"), "[D]-[M]-[Y]") devuelve 2014-03-06
- **altova:parse-date**(xs:string("06-03-2014"), "[M]-[D]-[Y]") devuelve 2014-06-03
- **altova:parse-date**("06/03/2014", "[M]/[D]/[Y]") devuelve 2014-06-03
- **altova:parse-date**("06 03 2014", "[M] [D] [Y]") devuelve 2014-06-03
- **altova:parse-date**("6 3 2014", "[M] [D] [Y]") devuelve 2014-06-03

parse-dateTime [altova:]

altova:parse-dateTime(FechaHora *como xs:string*, **PatrónFechaHora** *como xs:string*) como **xs:dateTime** XP2 XQ1 XP3 XQ3

Devuelve la cadena de entrada **FechaHora** como valor **xs:dateTime**. El segundo argumento (**PatrónFechaHora**) indica el patrón (secuencia de componentes) de la cadena de entrada. El argumento **PatrónFechaHora** se describe con los especificadores que aparecen a continuación y con cualquier separador de componentes (consulte los ejemplos más abajo).

D	Día
M	Mes
Y	Año

H	Hora
m	minutos
s	segundos

El patrón **PatrónFechaHora** debe coincidir con el patrón de **FechaHora**. Como el resultado es de tipo **xs:dateTime**, el resultado siempre tendrá el formato léxico **YYYY-MM-DDTHH:mm:ss**.

Ejemplos

- **altova:parse-dateTime**(xs:string("06-03-2014 13:56:24"), "[D]-[M]-[Y] [H]:[m]:[s]") devuelve 2014-03-06T13:56:24
- **altova:parse-dateTime**("time=13:56:24; date=06-03-2014", "time=[H]:[m]:[s]; date=[D]-[M]-[Y]") devuelve 2014-03-06T13:56:24

parse-time [altova:]

altova:parse-time(Hora *como xs:string*, **PatrónHora** *como xs:string*) como **xs:time** XP2 XQ1 XP3 XQ3

Devuelve la cadena de entrada **Hora** como valor **xs:time**. El segundo argumento (**PatrónHora**) indica el patrón (secuencia de componentes) de la cadena de entrada. El argumento **PatrónHora** se describe con los especificadores que aparecen a continuación y con cualquier separador de componentes (consulte los ejemplos más abajo).

H	Hora
m	minutos
s	segundos

El patrón **PatrónHora** debe coincidir con el patrón de **Hora**. Como el resultado es de tipo **xs:time**, el resultado siempre tendrá el formato léxico **HH:mm:ss**.

Ejemplos

- **altova:parse-time**(xs:string("13:56:24"), "[H]:[m]:[s]") devuelve 13:56:24
- **altova:parse-time**("13-56-24", "[H]-[m]") devuelve 13:56:00
- **altova:parse-time**("time=13h56m24s", "time=[H]h[m]m[s]s") devuelve 13:56:24
- **altova:parse-time**("time=24s56m13h", "time=[s]s[m]m[H]h") devuelve 13:56:24

[Subir]

Funciones para calcular la edad XP3 XQ3

Estas funciones devuelven la edad que se calcula obteniendo la diferencia (i) entre la fecha del argumento de entrada y la fecha actual o (ii) entre las fechas de los dos argumentos de entrada. La función **altova:age** devuelve la edad en años, mientras que la función **altova:age-details** devuelve la edad en forma de una secuencia de tres enteros (años, meses y días).

age [altova:]

 © 2014 Altova GmbH

altova:age(FechaInicio *as xs:date*) como **xs:integer** XP3 XQ3

Devuelve un entero que es la edad *en años* de algún objeto, contando a partir de la fecha de inicio dada como argumento y hasta la fecha actual (tomada del reloj del sistema). Si el argumento de entrada es un año o más después que la fecha actual, el valor devuelto será negativo.

Ejemplos

Si la fecha actual es **2014-01-15**:

- `altova:age(xs:date("2013-01-15"))` devuelve `1`
- `altova:age(xs:date("2013-01-16"))` devuelve `0`
- `altova:age(xs:date("2015-01-15"))` devuelve `-1`
- `altova:age(xs:date("2015-01-14"))` devuelve `0`

altova:age(FechaInicio *as xs:date*, **FechaFinal** *as xs:date*) como **xs:integer** XP3 XQ3

Devuelve un entero que es la edad *en años* de algún objeto, contando a partir de la fecha de inicio dada como primer argumento y hasta la fecha dada como segundo argumento. El valor devuelto será negativo si el primer argumento es un año o más después que el segundo argumento.

Ejemplos

- `altova:age(xs:date("2000-01-15"), xs:date("2010-01-15"))` devuelve `10`
- `altova:age(xs:date("2000-01-15"), current-date())` devuelve `14` si la fecha actual es `2014-01-15`
- `altova:age(xs:date("2014-01-15"), xs:date("2010-01-15"))` devuelve `-4`

age-details [altova:]

altova:age-details(FechaEntrada *as xs:date*) como (**xs:integer**)* XP3 XQ3

Devuelve tres enteros que son los años, meses y días respectivamente que hay entre la fecha dada como argumento y la fecha actual (tomada del reloj del sistema). La suma del valor devuelto nos da el tiempo total transcurrido entre ambas fechas (entre la fecha dada y la fecha actual). La fecha de entrada puede tener un valor anterior o posterior a la fecha actual, pero esto no se indica en el valor devuelto por medio de un signo negativo o positivo. El valor devuelto siempre es positivo.

Ejemplos

Si la fecha actual es **2014-01-15**:

- `altova:age-details(xs:date("2014-01-16"))` devuelve `(0 0 1)`
- `altova:age-details(xs:date("2014-01-14"))` devuelve `(0 0 1)`
- `altova:age-details(xs:date("2013-01-16"))` devuelve `(1 0 1)`
- `altova:age-details(current-date())` devuelve `(0 0 0`

altova:age-details(Fecha1 *as xs:date*, **Fecha2** *as xs:date*) como (**xs:integer**)* XP3 XQ3

Devuelve tres enteros que son los años, meses y días que hay entre las dos fechas dadas por los argumentos. La suma del valor devuelto nos da el tiempo total transcurrido entre las dos fechas de entrada. Da igual cuál de las dos fechas se da como primer argumento, la más antigua o la más reciente. El valor devuelto no indica si la fecha de entrada es anterior o posterior a la fecha actual. Es decir, el valor devuelto siempre es positivo.

Ejemplos

- `altova:age-details(xs:date("2014-01-16"), xs:date("2014-01-15"))` devuelve

(0 0 1)
- **altova:age-details**(xs:date("2014-01-15"), xs:date("2014-01-16")) devuelve (0 0 1)

[Subir]

Funciones XPath/XQuery: funciones de cadena

Estas funciones de extensión XPath/XQuery generales son compatibles con la versión actual de su producto de Altova y se pueden usar en (i) expresiones XPath en contextos XSLT o en (ii) expresiones XQuery en documentos XQuery.

Nota sobre espacios de nombres y lenguajes
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. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3

---

camel-case [altova:]

altova:camel-case(CadenaEntrada *como xs:string*) como **xs:string** XP3 XQ3
Devuelve la cadena de entrada **CadenaEntrada** escrita en CamelCase. La cadena se analiza usando la expresión regular **'\s'** (que es la forma abreviada del carácter espacio en blanco). El primer carácter que no sea un espacio en blanco situado después de un espacio en blanco o de una secuencia de espacios en blanco consecutivos se pondrá en mayúsculas. El primer carácter de la cadena de salida se pondrá en mayúsculas.

Ejemplos
- **altova:camel-case**("max") devuelve Max
- **altova:camel-case**("max max") devuelve Max Max
- **altova:camel-case**("file01.xml") devuelve File01.xml
- **altova:camel-case**("file01.xml file02.xml") devuelve File01.xml File02.xml
- **altova:camel-case**("file01.xml file02.xml") devuelve File01.xml File02.xml
- **altova:camel-case**("file01.xml -file02.xml") devuelve File01.xml -file02.xml

altova:camel-case(CadenaEntrada *como xs:string*, **CaracteresDivisión** *como*

xs:string, **EsExpReg** *como xs:boolean*) como **xs:string** XP3 XQ3
Devuelve la cadena de entrada **CadenaEntrada** escrita en CamelCase usando los **CaracteresDivisón** para determinar qué caracteres desencadenan el siguiente uso de mayúsculas. El argumento **CaracteresDivisón** se usa como expresión regular **cuando EsExpReg = true()** o como caracteres planos cuando r **EsExpReg = false()**. El primer carácter de la cadensa de salida se escribe con mayúsculas.

Ejemplos
- **altova:camel-case**("setname getname", "set|get", true()) devuelve setName getName
- **altova:camel-case**("altova\documents\testcases", "\", false()) devuelve Altova\Documents\Testcases

char [altova:]

altova:char(Posición *as xs:integer*) como **xs:string** XP3 XQ3
Devuelve una cadena que contiene el carácter que está en la posición indicada por el argumento **Posición** en la cadena que se obtiene al convertir el valor del elemento de contexto en xs:string. La cadena resultante estará vacía si en la posición indicada no existe ningún carácter.

Ejemplos
Si el elemento de contexto es **1234ABCD**:
- **altova:char**(2) devuelve 2
- **altova:char**(5) devuelve A
- **altova:char**(9) devuelve la cadena vacía
- **altova:char**(-2) devuelve la cadena vacía

altova:char(CadenaEntrada *as xs:string*, **Posición** *as xs:integer*) como **xs:string** XP3 XQ3
Devuelve una cadena que contiene el carácter que está en la posición indicada por el argumento **Posición** en la cadena dada por el argumento **CadenaEntrada**. La cadena resultante estará vacía si en la posición indicada no existe ningún carácter.

Ejemplos
- **altova:char**("2014-01-15", 5) devuelve -
- **altova:char**("USA", 1) devuelve U
- **altova:char**("USA", 1) devuelve la cadena vacía
- **altova:char**("USA", -2) devuelve la cadena vacía

first-chars [altova:]

altova:first-chars(X *as xs:integer*) como **xs:string** XP3 XQ3
Devuelve una cadena que contiene los **X** primeros caracteres de la cadena que se obtiene al convertir el valor del elemento de contexto en xs:string.

Ejemplos
Si el elemento de contexto es **1234ABCD**:
- **altova:first-chars**(2) devuelve 12
- **altova:first-chars**(5) devuelve 1234A
- **altova:first-chars**(9) devuelve 1234ABCD

altova:first-chars(CadenaEntrada *as xs:string*, **X** *as xs:integer*) como **xs:string** XP3 XQ3
Devuelve una cadena que contiene los **X** primeros caracteres de la cadena dada como argumento **CadenaEntrada**.

Ejemplos
- **altova:first-chars**("2014-01-15", 5) devuelve 2014-
- **altova:first-chars**("USA", 1) devuelve U

last-chars [altova:]

altova:last-chars(X *as xs:integer*) como **xs:string** XP3 XQ3
Devuelve una cadena que contiene los X últimos caracteres de la cadena que se obtiene al convertir el valor del elemento de contexto en xs:string.

Ejemplos
Si el elemento de contexto es **1234ABCD**:
- **altova:last-chars**(2) devuelve CD
- **altova:last-chars**(5) devuelve 4ABCD
- **altova:last-chars**(9) devuelve 1234ABCD

altova:last-chars(CadenaEntrada *as xs:string*, **X** *as xs:integer*) como **xs:string** XP3 XQ3
Devuelve una cadena que contiene los **X** últimos caracteres de la cadena dada como argumento **CadenaEntrada**.

Ejemplos
- **altova:last-chars**("2014-01-15", 5) devuelve 01-15-
- **altova:last-chars**("USA", 10) devuelve USA

pad-string-left [altova:]

altova:pad-string-left(Cadena *as xs:string*, **Repeticiones** *as xs:integer*, **CarácterRelleno** *as xs:string*) como **xs:string** XP3 XQ3
El argumento **CarácterRelleno** es un solo carácter que se incluye a la izquierda de la cadena dada como argumento **Cadena**. El argumento **Repeticiones** da el número de veces que se debe repetir el carácter de relleno a la izquierda de **Cadena**.

Ejemplos
- **altova:pad-string-left**('AP', 1, 'Z') devuelve 'ZAP'
- **altova:pad-string-left**('AP', 3, 'Z') devuelve 'ZZZAP'
- **altova:pad-string-left**('AP', 0, 'Z') devuelve 'AP'
- **altova:pad-string-left**('AP', 3, 'YZ') devuelve el error "Carácter de relleno demasiado largo".

pad-string-right [altova:]

altova:pad-string-right(Cadena *as xs:string*, **Repeticiones** *as xs:integer*, **CarácterRelleno** *as xs:string*) cpmo **xs:string** XP3 XQ3
El argumento **CarácterRelleno** es un solo carácter que se incluye a la derecha de la cadena dada como argumento **Cadena**. El argumento **Repeticiones** da el número de veces

que se debe repetir el carácter de relleno a la derecha de **Cadena**.

Ejemplos

- `altova:pad-string-right('AP', 1, 'Z')` devuelve `'APZ'`
- `altova:pad-string-right('AP', 3, 'Z')` devuelve `'APZZZ'`
- `altova:pad-string-right('AP', 0, 'Z')` devuelve `'AP'`
- `altova:pad-string-right('AP', 3, 'YZ')` devuelve el error "Carácter de relleno demasiado largo".

repeat-string [altova:]

altova:repeat-string(CadenaEntrada *as xs:string*, **Repeticiones** *as xs:integer*) como **xs:string** XP2 XQ1 XP3 XQ3

Genera una cadena que está compuesta por el primer argumento `CadenaEntrada` repetida tantas veces como indique el argumento `Repeticiones`.

Ejemplo

- `altova:repeat-string("Altova #", 3)`

  devuelve `Altova #Altova #Altova #"`

substring-after-last [altova:]

altova:substring-after-last(CadenaPrincipal *as xs:string*, **CadenaPrueba** *as xs:string*) como **xs:string** XP3 XQ3

Si **CadenaPrueba** se encuentra en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de **CadenaPrueba** en **CadenaPrincipal**. Si **CadenaPrueba** no está en **CadenaPrincipal**, entonces devuelve la cadena vacía. Si **CadenaPrueba** es una cadena vacía, entonces devuelve la **CadenaPrincipal** entera. Si **CadenaPrueba** aparece varias veces en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de la última **CadenaPrueba**.

Ejemplos

- `altova:substring-after-last('ABCDEFGH', 'B')` devuelve `'CDEFGH'`
- `altova:substring-after-last('ABCDEFGH', 'BC')` devuelve `'DEFGH'`
- `altova:substring-after-last('ABCDEFGH', 'BD')` devuelve `''`
- `altova:substring-after-last('ABCDEFGH', 'Z')` devuelve `''`
- `altova:substring-after-last('ABCDEFGH', '')` devuelve `'ABCDEFGH'`
- `altova:substring-after-last('ABCD-ABCD', 'B')` devuelve `'CD'`
- `altova:substring-after-last('ABCD-ABCD-ABCD', 'BCD')` devuelve `''`

substring-before-last [altova:]

altova:substring-before-last(CadenaPrincipal *as xs:string*, **CadenaPrueba** *as xs:string*) como **xs:string** XP3 XQ3

Si **CadenaPrueba** se encuentra en **CadenaPrincipal**, la función devuelve la subcadena que aparece después de **CadenaPrueba** en **CadenaPrincipal**. Si **CadenaPrueba** no está en **CadenaPrincipal**, entonces devuelve la cadena vacía. Si **CadenaPrueba** es una cadena vacía, entonces devuelve la **CadenaPrincipal** entera. Si **CadenaPrueba** aparece varias veces en **CadenaPrincipal**, la función devuelve la subcadena que aparece antes de la última **CadenaPrueba**.

Ejemplos

- **altova:substring-before-last**('ABCDEFGH', 'B') devuelve 'A'
- **altova:substring-before-last**('ABCDEFGH', 'BC') devuelve 'A'
- **altova:substring-before-last**('ABCDEFGH', 'BD') devuelve ''
- **altova:substring-before-last**('ABCDEFGH', 'Z') devuelve ''
- **altova:substring-before-last**('ABCDEFGH', '') devuelve ''
- **altova:substring-before-last**('ABCD-ABCD', 'B') devuelve 'ABCD-A'
- **altova:substring-before-last**('ABCD-ABCD-ABCD', 'ABCD') devuelve 'ABCD-ABCD-'

substring-pos [altova:]

altova:substring-pos(Cadena *as xs:string*, **CadenaBúsqueda** *as xs:string*) como **xs:integer** XP3 XQ3

Devuelve la posición de carácter de la primera instancia de **CadenaBúsqueda** en **Cadena**. La posición de carácter se devuelve como número entero. El primer carácter de **CadenaBúsqueda** tiene la posición 1. Si **CadenaBúsqueda** no aparece dentro de **Cadena**, la función devuelve el entero 0. Para buscar la segunda instancia de **CadenaBúsqueda**, etc. use la otra firma de esta función.

Ejemplos

- **altova:substring-pos**('Altova', 'to') devuelve 3
- **altova:substring-pos**('Altova', 'tov') devuelve 3
- **altova:substring-pos**('Altova', 'tv') devuelve 0
- **altova:substring-pos**('AltovaAltova', 'to') devuelve 3

altova:substring-pos(Cadena *as xs:string*, **CadenaBúsqueda** *as xs:string*, **Entero** *as xs:integer*) como **xs:integer** XP3 XQ3

Devuelve la posición de carácter de **CadenaBúsqueda** en **Cadena**. La búsqueda de **CadenaBúsqueda** empieza en la posición de carácter dada por el argumento **Entero** (es decir, no se busca en la subcadena anterior a esta posición). El entero devuelto, sin embargo, es la posición que la cadena encontrada tiene en **Cadena**. Esta firma es muy práctica si quiere buscar la segunda posición, etc. de una cadena que aparece varias veces dentro de **Cadena**. Si **CadenaBúsqueda** no aparece en **Cadena**, la función devuelve el entero 0.

Ejemplos

- **altova:substring-pos**('Altova', 'to', 1) devuelve 3
- **altova:substring-pos**('Altova', 'to', 3) devuelve 3
- **altova:substring-pos**('Altova', 'to', 4) devuelve 0
- **altova:substring-pos**('Altova-Altova', 'to', 0) devuelve 3
- **altova:substring-pos**('Altova-Altova', 'to', 4) devuelve 10

trim-string [altova:]

altova:trim-string(CadenaEntrada *as xs:string*) como **xs:string** XP3 XQ3

Esta función toma un argumento xs:string, quita los espacios en blanco iniciales y finales y devuelve un xs:string "recortado".

Ejemplos

- **altova:trim-string**(" Hello World ")) devuelve "Hello World"
- **altova:trim-string**("Hello World ")) devuelve "Hello World"
- **altova:trim-string**(" Hello World")) devuelve "Hello World"

 © 2014 Altova GmbH

- **altova:trim-string**("Hello World")) devuelve "Hello World"
- **altova:trim-string**("Hello World")) devuelve "Hello World"

trim-string-left [altova:]

altova:trim-string-left(CadenaEntrada *as xs:string***)** como **xs:string** XP3 XQ3

Esta función toma un argumento `xs:string`, quita los espacios en blanco iniciales y devuelve un `xs:string` recortado por la izquierda.

Ejemplos

- **altova:trim-string-left**(" Hello World ")) devuelve "Hello World "
- **altova:trim-string-left**("Hello World ")) devuelve "Hello World "
- **altova:trim-string-left**(" Hello World")) devuelve "Hello World"
- **altova:trim-string-left**("Hello World")) devuelve "Hello World"
- **altova:trim-string-left**("Hello World")) devuelve "Hello World"

trim-string-right [altova:]

altova:trim-string-right(CadenaEntrada *as xs:string***)** como **xs:string** XP3 XQ3

Esta función toma un argumento `xs:string`, quita los espacios en blanco finales y devuelve una cadena `xs:string` recortada por la derecha.

Ejemplos

- **altova:trim-string-right**(" Hello World ")) devuelve " Hello World"
- **altova:trim-string-right**("Hello World ")) devuelve "Hello World"
- **altova:trim-string-right**(" Hello World")) devuelve " Hello World"
- **altova:trim-string-right**("Hello World")) devuelve "Hello World"
- **altova:trim-string-right**("Hello World")) devuelve "Hello World"

Funciones XPath/XQuery: funciones varias

Estas funciones de extensión XPath/XQuery generales son compatibles con la versión actual de su producto de Altova y se pueden usar en (i) expresiones XPath en contextos XSLT o (ii) en expresiones XQuery en documentos XQuery.

Nota sobre espacios de nombres y lenguajes
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. Tenga en cuenta que en futuras versiones del producto algunas funciones pueden dejar de ser compatibles o su comportamiento puede cambiar. Por tanto, consulte siempre la documentación del producto para conocer el funcionamiento de estas funciones en cada versión del producto.

Funciones XPath (en expresiones XPath en XSLT):	XP1 XP2 XP3
Funciones XSLT (en expresiones XPath en XSLT):	XSLT1 XSLT2 XSLT3
Funciones XQuery (en expresiones XQuery en XQuery):	XQ1 XQ3

Funciones de numeración automática

generate-auto-number [altova:]

altova:generate-auto-number(ID *as xs:string*, **StartsWith** *as xs:double*, **Increment** *as xs:double*, **ResetOnChange** *as xs:string*) como xs:integer XP1 XP2 XQ1 XP3 XQ3

Genera un número cada vez que se llama a la función. El primer número, que se genera cuando se llama a la función por primera vez, viene dado por el argumento `StartsWith`. Cada llamada posterior genera un número nuevo, que se incrementa en función del valor especificado en el argumento `Increment`. De hecho, la función `altova:generate-auto-number` crea un contador llamado como indique el argumento `ID` y este contador se incrementa cada vez que se llama a la función. Si el valor del argumento `ResetOnChange` cambia con respecto al valor que tenía en la llamada anterior, entonces el valor del número que se debe generar se restablece con el valor de `StartsWith`. También puede restablecer la numeración automática con la función `altova:reset-auto-number`.

Ejemplo

- **altova:generate-auto-number**("ChapterNumber", 1, 1, "SomeString")
  Devuelve un número cada vez que se llama a la función, empezando por `1` y con un incremento de `1` con cada llamada a función. Si el cuarto argumento continúa siendo `"SomeString"` en las llamadas posteriores, el incremento continuará. Cuando cambie el valor del cuarto argumento, se restaura el valor 1 del contador (llamado `ChapterNumber`). El valor de `ChapterNumber` también se puede restaurar llamando a la función `altova:reset-auto-number`: `altova:reset-auto-number("ChapterNumber")`.

reset-auto-number [altova:]

altova:reset-auto-number(ID *as xs:string*) XP1 XP2 XQ1 XP3 XQ3

Esta función restaura el número del contador de numeración automática especificado en el argumento `ID`. El número se reemplaza con el número indicado en el argumento `StartsWith` de la función `altova:generate-auto-number` que creó el contador especificado en el argumento `ID`.

Ejemplos

- **altova:reset-auto-number**("ChapterNumber") restablece el número del contador de numeración automática llamado `ChapterNumber` que se creó con la función `altova:generate-auto-number`. El número se reemplaza con el valor del argumento `StartsWith` de la función `altova:generate-auto-number` que creó `ChapterNumber`.

[Subir]

Funciones numéricas

hex-string-to-integer [altova:]

altova:hex-string-to-integer(CadenaHex *as xs:string*) como **xs:integer** XP3 XQ3

Toma un argumento de cadena que es el equivalente Base-16 de un entero del sistema decimal (Base-10) y devuelve un entero decimal.

Ejemplos

 © 2014 Altova GmbH

- **altova:hex-string-to-integer**('1') devuelve 1
- **altova:hex-string-to-integer**('9') devuelve 9
- **altova:hex-string-to-integer**('A') devuelve 10
- **altova:hex-string-to-integer**('B') devuelve 11
- **altova:hex-string-to-integer**('F') devuelve 15
- **altova:hex-string-to-integer**('G') devuelve un error
- **altova:hex-string-to-integer**('10') devuelve 16
- **altova:hex-string-to-integer**('01') devuelve 1
- **altova:hex-string-to-integer**('20') devuelve 32
- **altova:hex-string-to-integer**('21') devuelve 33
- **altova:hex-string-to-integer**('5A') devuelve 90
- **altova:hex-string-to-integer**('USA') devuelve un error

integer-to-hex-string [altova:]

altova:integer-to-hex-string(Entero *as xs:integer***)** como **xs:string** XP3 XQ3

Toma el argumento **Entero** y devuelve su equivalente Base-16 en forma de cadena.

Ejemplos

- **altova:integer-to-hex-string**(1) devuelve '1'
- **altova:integer-to-hex-string**(9) devuelve '9'
- **altova:integer-to-hex-string**(10) devuelve 'A'
- **altova:integer-to-hex-string**(11) devuelve 'B'
- **altova:integer-to-hex-string**(15) devuelve 'F'
- **altova:integer-to-hex-string**(16) devuelve '10'
- **altova:integer-to-hex-string**(32) devuelve '20'
- **altova:integer-to-hex-string**(33) devuelve '21'
- **altova:integer-to-hex-string**(90) devuelve '5A'

[Subir]

Funciones de secuencia

attributes [altova:]

altova:attributes(NombreAtributo *as xs:string***)** como **attribute()*** XP3 XQ3

Devuelve todos los atributos cuyo nombre local coincida con el nombre dado como argumento de entrada (NombreAtributo). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje attribute::.

Ejemplos

- **altova:attributes**("MiAtributo") devuelve MiAtributo()*

altova:attributes(NombreAtributo *as xs:string*, **OpcionesBúsqueda** *as xs:string***)** como **attribute()*** XP3 XQ3

Devuelve todos los atributos cuyo nombre local coincida con el nombre dado como argumento de entrada (NombreAtributo). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje attribute::. El segundo argumento es una cadena con marcas de búsqueda. Estas son las marcas disponibles:

r = habilita la búsqueda de expresiones regulares. En este caso, NombreAtributo debe ser

una cadena de búsqueda de expresión regular;
i = la búsqueda no tiene en cuenta el uso de mayúsculas y minúsculas;
p = incluye el prefijo de espacio de nombres en la búsqueda. En este caso, `NombreAtributo` debe contener el prefijo de espacio de nombres (p. ej.: `altova:MiAtributo`).

Las marcas pueden escribirse en cualquier orden y no hace falta utilizar todas. Si usa marcas no válidas, se genera un error. También puede usar una cadena vacía para el segundo argumento. Esto tiene el mismo efecto que usar solo el primer argumento. Sin embargo, no está permitido usar una secuencia vacía.

Ejemplos

- **altova:attributes**("MiAtributo", "rip") devuelve MiAtributo()*
- **altova:attributes**("MiAtributo", "pri") devuelve MiAtributo()*
- **altova:attributes**("MiAtributo", "") devuelve MiAtributo()*
- **altova:attributes**("MiAtributo", "Rip") devuelve un error de marca desconocida.
- **altova:attributes**("MiAtributo",) devuelve un error diciendo que falta el segundo argumento.

elements [altova:]

altova:elements(NombreElemento *as xs:string***)** como **elemento()*** XP3 XQ3

Devuelve todos los elementos cuyo nombre local coincida con el nombre dado como argumento de entrada (`NombreElemento`). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje `child::`.

Ejemplos

- **altova:elements**("MiElemento") devuelve MiElemento()*

altova:elements(NombreElemento *as xs:string***, OpcionesBúsqueda** *as xs:string***)** como **elemento()*** XP3 XQ3

Devuelve todos los elementos cuyo nombre local coincida con el nombre dado como argumento de entrada (`NombreElemento`). La búsqueda tiene en cuenta el uso de mayúsculas y minúsculas y se lleva a cabo en el eje `child::`. El segundo argumento es una cadena con marcas de búsqueda. Estas son las marcas disponibles:

r = habilita la búsqueda de expresiones regulares. En este caso, `NombreElemento` debe ser una cadena de búsqueda de expresión regular;
i = la búsqueda no tiene en cuenta el uso de mayúsculas y minúsculas;
p = incluye el prefijo de espacio de nombres en la búsqueda. En este caso, `NombreElemento` debe contener el prefijo de espacio de nombres (p. ej.: `altova:MiElemento`).

Las marcas pueden escribirse en cualquier orden y no hace falta utilizar todas. Si usa marcas no válidas, se genera un error. También puede usar una cadena vacía para el segundo argumento. Esto tiene el mismo efecto que usar solo el primer argumento. Sin embargo, no está permitido usar una secuencia vacía.

Ejemplos

- **altova:elements**("MiElemento", "rip") devuelve MiElemento()*
- **altova:elements**("MiElemento", "pri") devuelve MiElemento()*
- **altova:elements**("MiElemento", "") devuelve MiElemento()*
- **altova:elements**("MiElemento", "Rip") devuelve un error de marca desconocida.
- **altova:elements**("MiElemento",) devuelve un error diciendo que falta el segundo

 © 2014 Altova GmbH

argumento.

find-first [altova:]

altova:find-first((Secuencia *como item()**)**, (Condición(Elemento-Secuencia** *como xs:boolean*)**)** como **item()?** **XP3** **XQ3**

Esta función toma dos argumentos. El primero es una secuencia de uno o varios elementos de cualquier tipo de datos. El segundo argumento, **Condición**, es una referencia a una función XPath que toma un argumento (es decir, su aridad es 1) y devuelve un valor binario. Cada elemento de **Secuencia** se envía a su vez a la función a la que se hace referencia en **Condición**. Nota: recuerde que esta función solo toma un argumento. El primer elemento de **Secuencia** que consiga que la función de **Condición** dé **true()** como resultado se devuelve como resultado de **altova:find-first** y la iteración se detiene.

Ejemplos

- **altova:find-first**(5 to 10, function($a) {$a mod 2 = 0}) devuelve xs:integer 6

El argumento **Condición** remite a la función inline XPath 3.0 **function()**, que declara una función inline llamada **$a** y después la define. Cada elemento del argumento Secuencia de **altova:find-first** se envía a su vez como valor de entrada a **$a**. El valor de entrada se prueba en la condición en la definición de función ($a mod 2 = 0). El primer valor de entrada que cumpla la condición se devuelve como resultado de **altova:find-first** (en este caso **6**).

- **altova:find-first**((1 to 10), (function($a) {$a+3=7})) devuelve xs:integer 4

Más ejemplos

Si existe el archivo **C:\Temp\Customers.xml**:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve xs:string C:\Temp\Customers.xml

Si no existe el archivo **C:\Temp\Customers.xml** pero existe **http://www.altova.com/index.html**:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve xs:string http://www.altova.com/index.html

Si no existe el archivo **C:\Temp\Customers.xml** y tampoco existe **http://www.altova.com/index.html**:

- **altova:find-first**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) no devuelve ningún resultado

Notas sobre los ejemplos anteriores

- La función XPath 3.0 doc-available toma un solo argumento de cadena, que se usa como URI, y devuelve true si en el URI dado se encuentra un nodo de documento. El documento que está en el URI dado debe ser un documento XML.
- La función doc-available se puede usar para **Condición**, el segundo argumento de altova:find-first, porque solamente toma un argumento (aridad=1), porque toma

un `item()` como entrada (una cadena que se usa como URI) y devuelve un valor binario.

- Recuerde que solamente se hace referencia a la función `doc-available` pero no se le llama. El sufijo `#1` que se anexa a la función indica una función cuya aridad es 1. Es decir, `doc-available#1` simplemente significa "*Utilizar la función doc-availabe() que tiene aridad=1, pasándole como solo argumento a su vez cada uno de los elementos de la primera secuencia.*" Como resultado, se pasarán las dos cadenas a **`doc-available()`**, que utiliza la cadena como URI y prueba si existe un nodo de documento en el URI. Si existe, entonces `doc-available()` da como resultado `true()` y esa cadena se devuelve como resultado de la función **`altova:find-first`**. Nota sobre la función `doc-available()`: las rutas de acceso relativas se resuelven en relación al URI base actual, que es por defecto el URI del documento XML desde el que se carga la función.

find-first-combination [altova:]

altova:find-first-combination((Sec-01 *como item()**), **(Sec-02** *como item()**), **(Condición(Elem-Sec-01, Elem-Sec-02** *como xs:boolean*)) como **item()*** XP3 XQ3

Esta función toma tres argumentos:

- Los dos primeros (**Sec-01** y **Sec-02**) son secuencias de uno o más elementos de cualquier tipo de datos.
- El tercero (**Condición**) es una referencia a una función XPath que toma dos argumentos (su aridad es 2) y devuelve un valor binario.

Los elementos de **Sec-01** y **Sec-02** se pasan en pares ordenados (cada par está formado por un elemento de cada secuencia) como argumentos de la función de **Condición**. Los pares se ordenan de la siguiente manera:

```
Si   Sec-01 = X1, X2, X3 ... Xn
Y  Sec-02 = Y1, Y2, Y3 ... Yn
Entonces (X1 Y1), (X1 Y2), (X1 Y3) ... (X1 Yn), (X2 Y1), (X2 Y2) ... (Xn
Yn)
```

El primer par ordenado que consiga que la función de **Condición** dé como resultado **`true()`** se devuelve como resultado de **`altova:find-first-combination`**. Recuerde que (i) si la función de **Condición** recorre los pares de argumentos dados y no consigue dar **`true()`** como resultado ni una vez, entonces **`altova:find-first-combination`** devuelve *Sin resultados;* (ii) el resultado de **`altova:find-first-combination`** siempre será un par de elementos (de cualquier tipo de datos) o ningún elemento.

Ejemplos

- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) devuelve la secuencia de xs:integers (11, 21)
- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) devuelve la secuencia de xs:integers (11, 22)
- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 34}) devuelve la secuencia de xs:integers (11, 23)

find-first-pair [altova:]

altova:find-first-pair((Sec-01 *como item()**), **(Sec-02** *como item()**), **(Condición(Elem-Sec-01, Elem-Sec-02** *como xs:boolean*)) como **item()*** XP3 XQ3

Esta función toma tres argumentos:

- Los dos primeros (**Sec-01** y **Sec-02**) son secuencias de uno o más elementos de cualquier tipo de datos.
- El tercero (**Condición**) es una referencia a una función XPath que toma dos argumentos (su aridad es 2) y devuelve un valor binario.

Los elementos de **Sec-01** y **Sec-02** se pasan en pares ordenados como argumentos de la función de **Condición**. Los pares se ordenan de la siguiente manera:

```
Si   Sec-01 = X1, X2, X3 ... Xn
Y  Sec-02 = Y1, Y2, Y3 ... Yn
Entonces (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

El primer par ordenado que consiga que la función de **Condición** dé como resultado **true()** se devuelve como resultado de **altova:find-first-pair**. Recuerde que (i) si la función de **Condición** recorre los pares de argumentos dados y no consigue dar **true()** como resultado ni una vez, entonces **altova:find-first-pair** devuelve *Sin resultados;* (ii) el resultado de **altova:find-first-pair** siempre será un par de elementos (de cualquier tipo de datos) o ningún elemento.

Ejemplos

- `altova:find-first-pair(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32})` devuelve la secuencia de `xs:integers (11, 21)`
- `altova:find-first-pair(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33})` devuelve *Sin resultados*

Observe que en los dos ejemplos anteriores el orden de los pares es: `(11, 21) (12, 22) (13, 23)...(20, 30)`. Por ese motivo el segundo ejemplo no obtiene resultados (porque ningún par ordenado consigue sumar 33).

find-first-pair-pos [altova:]

altova:find-first-pair-pos((Sec-01 *como item()**), **(Sec-02** *como item()**), **(Condición(Elem-Sec-01, Elem-Sec-02** *como xs:boolean*)) como **xs:integer** XP3 XQ3

Esta función toma tres argumentos:

- Los dos primeros (**Sec-01** y **Sec-02**) son secuencias de uno o más elementos de cualquier tipo de datos.
- El tercero (**Condición**) es una referencia a una función XPath que toma dos argumentos (su aridad es 2) y devuelve un valor binario.

Los elementos de **Sec-01** y **Sec-02** se pasan en pares ordenados como argumentos de la función de **Condición**. Los pares se ordenan de la siguiente manera:

```
Si   Sec-01 = X1, X2, X3 ... Xn
Y  Sec-02 = Y1, Y2, Y3 ... Yn
Entonces (X1 Y1), (X2 Y2), (X3 Y3) ... (Xn Yn)
```

La posición de índice del primer par ordenado que consiga que la función de **Condición** dé como resultado **true()** se devuelve como resultado de **altova:find-first-pair-pos**. Recuerde que si la función de **Condición** recorre los pares de argumentos dados y no da como resultado **true()** ni una sola vez, entonces **altova:find-first-pair-pos** devuelve

Sin resultados.

Ejemplos

- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 32}) devuelve 1
- **altova:find-first-pair**(11 to 20, 21 to 30, function($a, $b) {$a+$b = 33}) devuelve *Sin resultados*

Observe que en los dos ejemplos anteriores el orden de los pares es: (11, 21) (12, 22) (13, 23)...(20, 30). En el primer ejemplo el primer par consigue que la función de **Condición** dé como resultado **true()** y, por tanto, se devuelve la posición de índice que tienen en la secuencia (**1**). El segundo ejemplo, sin embargo, devuelve *Sin resultados* porque ningún par consigue sumar 33.

find-first-pos [altova:]

altova:find-first-pos((Secuencia *como item()**)**, (Condición(Elem-Sec** *como xs:boolean*)**)** como **xs:integer** **XP3** **XQ3**

Esta función toma dos argumentos. El primer argumento es una secuencia de uno o varios elementos de cualquier tipo. El segundo argumento (**Condición**) es una referencia a una función XPath que toma un argumento (su aridad es 1) y devuelve un valor binario. Cada elemento de **Secuencia** se envía a su vez a la función a la que se hace referencia en **Condición**. (Recuerde que esta función toma un solo argumento.) El primer elemento de **Secuencia** que consiga que la función de **Condición** dé como resultado **true()** devuelve la posición de índice que tiene en **Secuencia** como resultado de **altova:find-first-pos** y la iteración se detiene.

Ejemplos

- **altova:find-first-pos**(5 to 10, function($a) {$a mod 2 = 0}) devuelve xs:integer 2

El argumento **Condición** hace referencia a la función inline XPath 3.0 **function()**, que declara una función inline llamada **$a** y después la define. Cada elemento del argumento **Sequence** de **altova:find-first-pos** se pasa a su vez como valor de entrada de **$a**. El valor de entrada se prueba en la condición de la definición de función ($a mod 2 = 0). La posición de índice que tiene en la secuencia el primer valor de entrada que cumple la condición se devuelve como resultado de **altova:find-first-pos** (en este caso es la posición de índice **2**, porque **6** es el primer valor (de la secuencia) que cumple la condición y su posición de índice en la secuencia es **2**).

- **altova:find-first-pos**((2 to 10), (function($a) {$a+3=7})) devuelve xs:integer 3

Más ejemplos

Si existe el archivo **C:\Temp\Customers.xml**:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve 1

Si no existe el archivo **C:\Temp\Customers.xml** pero existe **http://www.altova.com/index.html**:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) devuelve 2

Si no existe el archivo **C:\Temp\Customers.xml** y tampoco existe **http://www.altova.com/index.html**:

- **altova:find-first-pos**(("C:\Temp\Customers.xml", "http://www.altova.com/index.html"), (doc-available#1)) no devuelve ningún resultado

Notas sobre los ejemplos anteriores

- La función XPath 3.0 doc-available toma un solo argumento de cadena, que se usa como URI, y devuelve true si en el URI dado se encuentra un nodo de documento. El documento que está en el URI dado debe ser un documento XML.
- La función doc-available se puede usar para **Condición**, el segundo argumento de altova:find-first-pos, porque solamente toma un argumento (aridad=1), porque toma un item() como entrada (una cadena que se usa como URI) y devuelve un valor binario.
- Recuerde que solamente se hace referencia a la función doc-available pero no se le llama. El sufijo #1 que se anexa a la función indica una función cuya aridad es 1. Es decir, doc-available#1 simplemente significa "*Utilizar la función doc-availabe() que tiene aridad=1, pasándole como solo argumento a su vez cada uno de los elementos de la primera secuencia.*" Como resultado, se pasarán las dos cadenas a **doc-available()**, que utiliza la cadena como URI y prueba si existe un nodo de documento en el URI. Si existe, entonces doc-available() da como resultado true() y esa cadena se devuelve como resultado de la función **altova:find-first-pos**. Nota sobre la función doc-available(): las rutas de acceso relativas se resuelven en relación al URI base actual, que es por defecto el URI del documento XML desde el que se carga la función.

substitute-empty [altova:]

altova:substitute-empty(PrimeraSecuencia *as item()**, **SegundaSecuencia** *as item()***)** como **item()*** XP3 XQ3

Si **PrimeraSecuencia** está vacío, la función devuelve **SegundaSecuencia**. Si **PrimeraSecuencia** no está vacío, la función devuelve **PrimeraSecuencia**.

Ejemplos

- **altova:substitute-empty**((1,2,3), (4,5,6)) devuelve (1,2,3)
- **altova:substitute-empty**((), (4,5,6)) devuelve (4,5,6)

[Subir]

Funciones para trabajar con identificadores URI

get-temp-folder [altova:]

altova:get-temp-folder() como xs:string XP2 XQ1 XP3 XQ3

Esta función no toma ningún argumento. Devuelve la ruta de acceso de la carpeta temporal del usuario actual.

Ejemplo

- **altova:get-temp-folder()**
  en un equipo Windows devuelve (más o menos) `C:\Usuarios\<Usuario>\AppData\Local\Temp\` como valor de tipo `xs:string`.

[Subir]

Funciones de extensión varias

Los lenguajes de programación como Java y C# ofrecen varias funciones predefinidas que no están disponibles como funciones XQuery/XPath ni XSLT. Un ejemplo son las funciones matemáticas de Java `sin()` y `cos()`. Si los diseñadores de hojas de estilos XSLT y consultas XQuery tuvieran acceso a estas funciones, el área de aplicación de sus hojas de estilos y consultas aumentaría y su trabajo sería un poco más sencillo.

Los motores XSLT y XQuery de los productos de Altova admiten el uso de funciones de extensión en Java y .NET, así como scripts MSXSL para XSLT.

Esta sección describe cómo usar funciones de extensión y scripts MSXSL en hojas de estilos XSLT y documentos XQuery. Las funciones de extensión pueden organizarse en varios grupos:

- Funciones de extensión Java
- Funciones de extensión .NET
- Scripts MSXSL para XSLT

En los apartados de esta sección nos ocupamos de tres aspectos fundamentales: (i) cómo se llaman las funciones en sus respectivas bibliotecas, (ii) qué reglas deben seguirse para convertir los argumentos de una llamada a función en el formato de entrada necesario de la función y (iii) qué reglas deben seguirse para la conversión del tipo devuelto.

Requisitos
Para que estas funciones de extensión funcionen es necesario tener Java Runtime Environment (para las funciones Java) y .NET Framework 2.0 o superior (para las funciones .NET) instalado en el equipo que ejecuta la transformación XSLT o XQuery.

Funciones de extensión Java

Puede usar una función de extensión Java dentro de una expresión XPath o XQuery para invocar un constructor Java o llamar a un método Java (estático o de instancia).

Un campo de una clase Java se trata como un método sin argumentos. Un campo puede ser estático o de instancia. Más adelante describimos cómo se accede a los campos estáticos y de instancia.

Este apartado tiene varias partes:

- Constructores Java
- Métodos estáticos y campos estáticos
- Métodos de instancia y campos de instancia

 © 2014 Altova GmbH

- Tipos de datos: conversión de XPath/XQuery en Java
- Tipos de datos: conversión de Java en XPath/XQuery

Formato de la función de extensión

La función de extensión de la expresión XPath/XQuery debe tener este formato `prefijo:nombreFunción()`.

- La parte `prefijo:` identifica la función de extensión como función Java. Lo hace asociando la función de extensión con una declaración de espacio de nombres del ámbito, cuyo URI debe empezar por `java:` (*ver ejemplos más abajo*). La declaración de espacio de nombres debe identificar una clase Java, por ejemplo: `xmlns:myns="java:java.lang.Math"`. Sin embargo, también puede ser simplemente: `xmlns:myns="java"` (sin los dos puntos), dejando la identificación de la clase Java a la parte `nombreFunción()` de la función de extensión.
- La parte `nombreFunción()`identifica el método Java al que se llama y presenta los argumentos para el método (*ver ejemplos más abajo*). Sin embargo, si el URI de espacio de nombres identificado por la parte `prefijo:` no identifica una clase Java (*ver punto anterior*), entonces la clase Java debe identificarse en la parte `nombreFunción()`, antes de la clase y separada de la clase por un punto (*ver el segundo ejemplo XSLT que aparece más abajo*).

Nota: la clase a la que se llama debe estar en la ruta de acceso de clase del equipo.

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*.)
- El archivo de clases no está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el archivo de clases. (*ver ejemplo*.)
- El archivo de clases está en un paquete. El archivo XSLT/XQuery está en una carpeta cualquiera. (*ver ejemplo*.)
- El archivo de clases no está en un paquete. El archivo XSLT/XQuery está una carpeta cualquiera. (*ver ejemplo*.)

Imaginemos que tenemos un archivo de clases que no está en un paquete y que está en la misma carpeta que el documento XSLT/XQuery. En este caso, puesto que en la carpeta se encuentran todas las clases, no es necesario especificar la ubicación del archivo. La sintaxis que se utiliza para identificar una clase es esta:

`java:nombreClase`

donde

`java:` indica que se está llamando a una función definida por el usuario (por defecto se cargan las clases Java del directorio actual)
`nombreClase` es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método.

El archivo de clases está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el paquete Java
El código que aparece a continuación llama al método `getVehicleType()` de la clase `Car` del paquete `com.altova.extfunc`. El paquete `com.altova.extfunc` está en la carpeta `JavaProject`. El archivo XSLT también está en la carpeta `JavaProject`.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
```

```
        xmlns:car="java:com.altova.extfunc.Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

El archivo de clases no está en un paquete. El archivo XSLT/XQuery está en la misma carpeta que el archivo de clases

El código que aparece a continuación llama al método `getVehicleType()` de la clase `Car` del paquete `com.altova.extfunc`. El archivo de clases `Car` está en esta carpeta: `JavaProject/com/altova/extfunc`. El archivo XSLT también está en la carpeta `JavaProject/com/altova/extfunc`.

```
<xsl:stylesheet version="2.0"
        xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:fn="http://www.w3.org/2005/xpath-functions"
        xmlns:car="java:Car" >
<xsl:output exclude-result-prefixes="fn car xsl fo xs"/>

<xsl:template match="/">
   <a>
   <xsl:value-of select="car:getVehicleType()"/>
   </a>
</xsl:template>

</xsl:stylesheet>
```

El archivo de clases está en un paquete. El archivo XSLT/XQuery está en una carpeta cualquiera

El código que aparece a continuación llama al método `getCarColor()` de la clase `Car` del paquete `com.altova.extfunc`. El paquete `com.altova.extfunc` está en la carpeta `JavaProject`. El archivo XSLT está en otra carpeta cualquiera. En este caso debe especificarse la ubicación del paquete dentro del URI como una cadena de consulta. La sintaxis es esta:

```
java:nombreClase[?ruta=uri-del-paquete]
```

donde

`java:` indica que se está llamando a una función Java definida por el usuario
`uri-del-paquete` es el URI del paquete Java

nombreClase es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método. El ejemplo de código que aparece a continuación explica cómo se accede a un archivo de clases que está ubicado en un directorio que no es el directorio actual.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:com.altova.extfunc.Car?path=file:///C:/
JavaProject/" >

<xsl:output exclude-result-prefixes="fn car xsl xs"/>

<xsl:template match="/">
   <xsl:variable name="myCar" select="car:new('red')" />
   <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
</xsl:template>

</xsl:stylesheet>
```

El archivo de clases no está en un paquete. El archivo XSLT/XQuery está una carpeta cualquiera

El código que aparece a continuación llama al método getCarColor()de la clase Car del paquete com.altova.extfunc. El paquete com.altova.extfunc está en la carpeta JavaProject. El archivo XSLT está en otra carpeta cualquiera. En este caso debe especificarse la ubicación del paquete dentro del URI como una cadena de consulta. La sintaxis es esta:

```
java:nombreClase[?ruta=uri-del-archivoClases]
```

donde

java: indica que se está llamando a una función Java definida por el usuario
uri-del-archivoClases es el URI de la carpeta donde se ubica el archivo de clases
nombreClase es el nombre de la clase del método elegido

La clase se identifica en un URI de espacio de nombres y el espacio de nombres se usa como prefijo para la llamada al método. El ejemplo de código que aparece a continuación explica cómo se accede a un archivo de clases que está ubicado en un directorio que no es el directorio actual.

```
<xsl:stylesheet version="2.0"
       xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
       xmlns:xs="http://www.w3.org/2001/XMLSchema"
       xmlns:fn="http://www.w3.org/2005/xpath-functions"
       xmlns:car="java:Car?path=file:///C:/JavaProject/com/altova/
```

```
extfunc/" >

        <xsl:output exclude-result-prefixes="fn car xsl xs"/>

        <xsl:template match="/">
           <xsl:variable name="myCar" select="car:new('red')" />
           <a><xsl:value-of select="car:getCarColor($myCar)"/></a>
        </xsl:template>

        </xsl:stylesheet>
```

Nota: cuando se presenta una ruta de acceso por medio de una función de extensión, la ruta de acceso se añade al ClassLoader.

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.

Constructores

Una función de extensión se puede usar para llamar a un constructor Java. A todos los constructores se les llama con la pseudofunción `new()`.

Si el resultado de una llamada a un constructor Java se puede convertir de manera implícita a tipos de datos XPath/XQuery, entonces la llamada a la función de extensión Java devuelve una secuencia que es un tipo de datos XPath/XQuery. Si el resultado de una llamada a un constructor Java no se puede convertir a un tipo de datos XPath/XQuery adecuado, entonces el constructor crea un objeto Java contenido con un tipo que es el nombre de la clase que devuelve ese objeto Java. Por ejemplo, si se llama a un constructor para la clase `java.util.Date` (`java.util.Date.new()`), entonces se devuelve un objeto que tiene el tipo `java.util.Date`. Puede que el formato léxico del objeto devuelto no coincida con el formato léxico de un tipo de datos XPath y, por tanto, su valor debe convertirse al formato léxico del tipo de datos XPath pertinente y después al tipo de datos XPath.

Puede hacer dos cosas con el objeto Java creado por un constructor:

- Puede asignar el objeto a una variable:
  ```
  <xsl:variable name="currentdate" select="date:new()"
  xmlns:date="java:java.util.Date" />
  ```
- Puede pasar el objeto a una función de extensión (*ver métodos de instancia y campos de instancia*):
  ```
  <xsl:value-of select="date:toString(date:new())"
  xmlns:date="java:java.util.Date" />
  ```

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)" />
```

 © 2014 Altova GmbH

```
<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>
```

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: conversión de XPath/XQuery en Java

Cuando se llama a una función Java desde dentro de una expresión XPath/XQuery, el tipo de datos de los argumentos de la función es importante a la hora de determinar a cuál de las clases Java que tienen el mismo nombre se llama.

En Java se siguen estas reglas:

- Si hay más de un método Java con el mismo nombre, pero cada método tiene un número diferente de argumentos, entonces se selecciona el método Java que mejor se ajusta al número de argumentos de la llamada a función.
- Los tipos de datos de cadena, numéricos y booleanos de XPath/XQuery (*ver lista más abajo*) se convierten de forma implícita en el tipo de datos Java correspondiente. Si el tipo XPath/XQuery suministrado se puede convertir a más de un tipo Java (p. ej. `xs:integer`), entonces se selecciona el tipo Java que se declaró para el método seleccionado. Por ejemplo, si el método Java al que se llama es `fx(decimal)` y el tipo de datos XPath/XQuery suministrado es `xs:integer`, entonces `xs:integer` se convierte en el tipo de datos Java `decimal`.

La tabla que aparece a continuación enumera las conversiones implícitas de los tipos de cadena, numéricos y booleanos XPath/XQuery en tipos de datos Java.

`xs:string`	`java.lang.String`
`xs:boolean`	`boolean` (primitivo), `java.lang.Boolean`
`xs:integer`	int, long, short, byte, float, double y sus clases contenedoras, como `java.lang.Integer`
`xs:float`	`float` (primitivo), `java.lang.Float`, `double` (primitivo)
`xs:double`	`double` (primitivo), `java.lang.Double`
`xs:decimal`	`float` (primitivo), `java.lang.Float`, `double`(primitivo), `java.lang.Double`

Los subtipos de los tipos de datos XML Schema de la tabla anterior (que se usan en XPath y XQuery) también se convierten en los tipos Java correspondientes al tipo antecesor del subtipo.

En algunos casos quizás no sea posible seleccionar el método Java correcto usando la información dada. Por ejemplo, imagine que:

 © 2014 Altova GmbH

- 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: conversión de Java en XPath/XQuery

Cuando un método Java devuelve un valor y el tipo de datos del valor es un tipo de cadena, numérico o booleano, entonces se convierte en el tipo de datos XPath/XQuery correspondiente. Por ejemplo, los tipos de datos Java `java.lang.Boolean` y `boolean` se convierten en `xsd:boolean`.

Las matrices unidimensionales devueltas por las funciones se extienden en una secuencia. Las matrices multidimensionales no se convierten y, por tanto, deberían ser contenidas.

Cuando se devuelve un objeto Java contenido o un tipo de datos que no es de cadena, numérico ni booleano, puede garantizar la conversión del tipo XPath/XQuery necesario usando primero un método Java (p. ej. `toString`) para convertir el objeto Java en una cadena. En XPath/XQuery la cadena se puede modificar para ajustarse a la representación léxica del tipo necesario y convertirse después en dicho tipo (usando la expresión `cast as`, por ejemplo).

Funciones de extensión .NET

Si trabaja en la plataforma .NET desde un equipo Windows, puede usar funciones de extensión escritas en cualquier lenguaje .NET (p. ej. C#). Una función de extensión .NET se puede usar dentro de una expresión XPath/XQuery para invocar un constructor, una propiedad o un método (estático o de instancia) de una clase .NET.

A una propiedad de una clase .NET se le llama usando la sintaxis `get_NombrePropiedad()`.

Este apartado tiene varias partes:

- Constructores
- Métodos estáticos y campos estáticos
- Métodos de instancia y campos de instancia
- Tipos de datos: conversión de XPath/XQuery en .NET
- Tipos de datos: conversión de .NET en XPath/XQuery

Formato de la función de extensión

La función de extensión de la expresión XPath/XQuery debe tener este formato `prefijo:nombreFunción()`.

- La parte `prefijo:` está asociada a un URI que identifica la clase .NET.
- La parte `nombreFunción()` identifica el constructor, la propiedad o el método (estático o de instancia) dentro de la clase .NET y, si es necesario, suministra los argumentos.
- El URI debe empezar por `clitype:` (que identifica la función como función de extensión .NET).
- El formato `prefijo:nombreFunción()` de la función de extensión se puede usar con clases del sistema y con clases de un ensamblado cargado. No obstante, si se tiene que cargar una clase, será necesario suministrar parámetros que contengan la información necesaria.

Parámetros

Para cargar un ensamblado se usan estos parámetros:

`asm`	El nombre del ensamblado que se debe cargar.
`ver`	El número de versión (máximo cuatro enteros separados por puntos).
`sn`	El símbolo de clave del nombre seguro del ensamblado (16 dígitos hexadecimales).
`from`	Un URI que da la ubicación del ensamblado (DLL) que se debe cargar. Si el URI es relativo, es relativo al archivo XSLT o XQuery. Si está presente este parámetro, se ignoran los demás parámetros.
`partialname`	El nombre parcial del ensamblado. Se suministra a `Assembly.LoadWith.PartialName()`, que intentará cargar el ensamblado. Si está presente el parámetro `partialname`, se ignoran los demás parámetros.
`loc`	La configuración regional, por ejemplo, `en-US`. La configuración predeterminada es `neutral`.

Si el ensamblado se debe cargar desde un archivo DLL, use el parámetro `from` y omita el parámetro `sn`. Si el ensamblado se debe cargar desde el caché general de ensamblados (GAC), use el parámetro `sn` y omita el parámetro `from`.

Debe insertar un signo de interrogación final antes del primer parámetro y los parámetros deben separarse con un punto y coma (;). El nombre de parámetro da su valor con un signo igual (=), como en el ejemplo que aparece más abajo.

Ejemplos de declaraciones de espacios de nombres

Esto es un ejemplo de una declaración de espacio de nombres en XSLT que identifica la clase del sistema `System.Environment`:

```
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.

Constructores

Una función de extensión se puede usar para llamar a un constructor .NET. A todos los constructores se les llama con la pseudofunción `new()`. Si hay más de un constructor para una clase, entonces se selecciona el constructor que más se ajusta al número de argumentos suministrados. Si no se encuentra ningún constructor que coincida con los argumentos suministrados, entonces se genera el error `"No constructor found"`.

Constructores que devuelven tipos de datos XPath/XQuery
Si el resultado de una llamada a un constructor .NET se puede convertir de forma implícita en tipos de datos XPath/XQuery, entonces la función de extensión .NET devuelve una secuencia que es un tipo de datos XPath/XQuery.

Constructores que devuelven objetos .NET
Si el resultado de una llamada a un constructor .NET no se puede convertir a un tipo de datos XPath/XQuery adecuado, entonces el constructr crea un objeto .NET contenido con un tipo que es el nombre de la clase que devuelve dicho objeto. Por ejemplo, si se llama al constructor para la clase `System.DateTime` (con `System.DateTime.new()`), entonces se devuelve un objeto que tiene un tipo `System.DateTime`.

Puede que el formato léxico del objeto devuelto no coincida con el formato léxico de un tipo de datos XPath. En estos casos, el valor devuelto (i) debe convertirse al formato léxico del tipo de datos XPath pertinente y (ii) debe convertirse en el tipo de datos XPath necesario.

Se pueden hacer tres cosas con un objeto .NET creado con un constructor:

- Se puede usar dentro de una variable:
  ```
  <xsl:variable name="currentdate" select="date:new(2008, 4, 29)"
  xmlns:date="clitype:System.DateTime" />
  ```
- Se puede pasar a una función de extensión (ver *Métodos de instancia y campos de instancia*):
  ```
  <xsl:value-of select="date:ToString(date:new(2008, 4, 29))"
  xmlns:date="clitype:System.DateTime" />
  ```
- Se puede convertir en un tipo de cadena, numérico o booleano:
  ```
  <xsl:value-of select="xs:integer(data:get_Month(date:new(2008, 4, 29)))"
  xmlns:date="clitype:System.DateTime" />
  ```

 © 2014 Altova GmbH

Metodos estáticos y campos estáticos

La llamada a un método estático la hace directamente su nombre y se hace presentando los argumentos para el método. El nombre usado en la llamada debe ser el mismo que un método estático público de la clase especificada. Si el nombre del método y el número de argumentos que se dio en la llamada a función coincide con algún método de la clase, entonces los tipos de los argumentos presentados se evalúan para encontrar el resultado ideal. Si no se encuentra ninguna coincidencia, se emite un error.

Nota: un campo de una clase .NET se trata como si fuera un método sin argumentos. Para llamar a una propiedad se usa la sintaxis `get_nombrePropiedad()`.

Ejemplos

Este ejemplo de código XSLT muestra una llamada a un método con un argumento (`System.Math.Sin(arg)`):

```
<xsl:value-of select="math:Sin(30)" xmlns:math="clitype:System.Math"/>
```

Este ejemplo de código XSLT muestra una llamada a un campo (que se trata como si fuera un método sin argumentos) (`System.Double.MaxValue()`):

```
<xsl:value-of select="double:MaxValue()"
xmlns:double="clitype:System.Double"/>
```

Este ejemplo de código XSLT muestra una llamada a una propiedad (la sintaxis es `get_nombrePropiedad()`) (`System.String()`):

```
<xsl:value-of select="string:get_Length('my string')"
xmlns:string="clitype:System.String"/>
```

Este ejemplo de código XQuery muestra una llamada a un método con un argumento (`System.Math.Sin(arg)`):

```
<sin xmlns:math="clitype:System.Math">
   { math:Sin(30) }
</sin>
```

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: conversión de XPath/XQuery en .NET

Cuando se usa una función de extensión .NET dentro de una expresión XPath/XQuery, los tipos de datos de los argumentos de la función son importantes para determinar a cuál de los métodos .NET que tienen el mismo nombre se está llamando.

En .NET se siguen estas normas:

- Si en una clase hay varios métodos que tienen el mismo nombre, solamente se pueden seleccionar los métodos que tienen el mismo número de argumentos que la llamada a función.
- Los tipos de datos de cadena, numéricos y booleanos XPath/XQuery (*ver lista más abajo*) se convierten de forma implícita en el tipo de datos .NET correspondiente. Si el tipo XPath/XQuery suministrado se puede convertir en más de un tipo .NET (p. ej. `xs:integer`), entonces se selecciona el tipo .NET que se declaró para el método seleccionado. Por ejemplo, si el método .NET al que se está llamando es `fx(double)` y el tipo de datos XPath/XQuery suministrado es `xs:integer`, entonces se convierte `xs:integer` en el tipo de datos .NET `double`.

 © 2014 Altova GmbH

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: conversión de .NET en XPath/XQuery

Cuando un método .NET devuelve un valor y el tipo de datos del valor es un tipo de cadena, numérico o booleano, entonces se convierte en el tipo de datos XPath/XQuery correspondiente. Por ejemplo, el tipo de datos .NET `decimal` se convierte en `xsd:decimal`.

Cuando se devuelve un objeto .NET o un tipo de datos que no es de cadena, numérico ni booleano, puede garantizar la conversión del tipo XPath/XQuery necesario usando primero un método .NET (p. ej. `System.DateTime.ToString()`) para convertir el objeto .NET en una cadena. En XPath/XQuery la cadena se puede modificar para ajustarse a la representación léxica del tipo necesario y convertirse después en dicho tipo (usando la expresión `cast as`, por ejemplo).

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">
```

 © 2014 Altova GmbH

```
  <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

16.2 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

16.2.1 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

16.2.2 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.

16.2.3 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

16.2.4 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.

16.2.5 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

16.3 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.

16.3.1 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

16.3.2 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.

16.3.3 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

16.3.4 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

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, 57

/

/-,
parámetros en tiempo de ejecución (línea de comandos), 215

<

<dynamic>,
file input / output, 57

A

abs, 384
Absolutas,
rutas de acceso (ventajas y desventajas), 278
Acerca de MapForce, 433
Activar el software, 431
add, 357
Adjust-to-Timezone, 381
Advertencia,
validación, 89
Agregado,
función de agregado (usar plantillas con nombre), 337
funciones, 201
Agregar, 403
archivo de recursos globales, 242
duplicado delante/detrás, 403
funciones definidas por el usuario, 295
ubicación del esquema, 403
Ajustar tamaño,
componente "ajuste óptimo", 22
Ajuste óptimo,
doble clic en el icono de ajuste del tamaño, 22
Alineación automática,
líneas de guía, 23
Alinear,
componentes en la ventana de asignación, 23
Alinear automáticamente,
componentes de la asignación, 66
Altova XML,
DoTransform.bat, 39
Ampliar,
parámetros de las funciones, 145
Analizador,
integrado en los productos de Altova, 497
Analizador XML,
información sobre, 497
Analizador XML de Altova,
información sobre, 497
Anidadas,
funciones definidas por el usuario, 324
Anotación,
conector, 408
Any,
xs:any, 171
anyURI,
funciones, 378
Archivo, 395
agregar configuración de recurso, 242
catálogo, 94
definir recurso global, 242
Archivo de definición,
globalresource.xml, 240
Archivo de destino,
varios archivos de destino a partir de un solo archivo de origen, 275
Archivo de origen,
dividir entre varios archivos de destino, 275
Archivos,
nombres de archivo dinámicos como archivos de entrada, 273
varios archivos a partir de una BD, 275
Archivos XML,
a partir de un solo XML de origen, 275
Asignación,
basada en el destino, 119
basada en el origen, contenido mixto, 113
basada en el tipo, 121
conector, 106
elementos del esquema, 28

Asignación,
elementos secundarios, 77
estándar, 119
nombre del esquema de destino, 39
número de conectors, 106
propiedades, 77
secuencia de procesamiento, 127
tutorial, 20
validación, 89
validar estructura, 89
ventana, alineación automática, 23
Asignación automática,
elementos secundarios, 77
Asignación basada en el destino, 112
Asignación de valores,
pasar datos sin cambiarlos, 195
tabla de búsqueda, 190
tabla de búsqueda, propiedades, 199
Asignación en cadena,
mostrar componente final usando StyleVision, 132, 137
Asignar,
archivo SPS a componentes, 101
recurso global a componente, 245
ATTLIST,
URI de espacio de nombres del documento DTD, 158
auto-number, 353
avg, 341
Ayuda, 429
ver Ayuda en pantalla, 430
Ayuda en pantalla,
búsqueda, 430
contenido, 430
índice, 430
A-Z,
componente de ordenación, 182

B

Bajar,
elemento en el componente, 403
Barra de herramientas,
recurso global, 241
Basada en el destino,
asignación, 119
Basada en el origen,
asignación de contenido mixto, 113
frente a asignación estándar, 119
Basadas en el tipo,
conexiones, 121
Base,
tipo base (tipos derivados), 159
Base de datos,
cambiar de BD, 403
quitar nombre del esquema del código, 403
y varios componentes de origen, 403
base-uri, 378
Biblioteca,
agregar funciones XSLT, 332
agregar funciones XSLT 2.0, 336
buscar función en, 145
definir, 294
función, 145
importar funciones definidas por el usuario, 295
personal, 145
referencia de funciones, 340
XPath 2, 145
Bibliotecas,
y funciones definidas por el usuario, 295
BOM,
marca de orden de bytes, 403
Bool,
salida si bool = false, 324
boolean, 344
Booleano/a,
comparar nodos de entrada, 204
Borrar,
funciones definidas por el usuario, 295
Bucles,
grupos y jerarquías, 148
BUILTIN,
nombre de componente, 215
Built-in engine,
definition, 86
using, 86
Buscar,
función en biblioteca, 145
XSLT: panel Resultados, 400

C

Calcular una vez,
variable, 290

Call,
template, 332
Cambiar,
configuración, recurso global, 247
Cambiar de,
configuración, recurso global, 247
Carpetas,
como recurso global, 249
Catálogo,
archivo, 94
CDATA, 169
ceiling, 357
Centro de descarga de componentes,
del sitio web de Altova, 433
char-from-code, 366
Clases contenedoras,
compatibilidad de las versiones, 420
Clave,
clave de ordenación, 182
Clave de evaluación,
para su software de Altova, 431
Clave de ordenación,
componente de ordenación, 182
code-from-char, 366
Codificación, 403
configuración del componente, 403
marca de orden de bytes, 403
Código,
de salida (línea de comandos), 92
funciones inline y tamaño del código, 305
quitar nombre del esquema del código, 403
Código de salida, 92
Códigos clave,
para su software de Altova, 431
Combinación de teclas,
acceso rápido, 420
Combinar,
varios archivos de entrada, 270
Comillas,
parámetros de la línea de comandos, 210
Comments,
Adding to target files, 167
Cómo puedo..., 178
Comodines,
uso de comillas en la línea de comandos, 210
compare, 385
Compatibilidad con Unicode,
de los productos de Altova, 499
compl.,
grupo de nodos del complemento, 35
Compleja,
entrada compleja definida por el usuario, 313
función (inline), 305
función definida por el usuario, 312, 318
salida compleja definida por el usuario, 318
Componente,
ajustar hasta su tamaño óptimo, 22
asignar recurso global, 245
asignar un archivo SPS, 101
cambiar configuración, 403
cambiar de BD, 403
elementos eliminados, 82
entrada - valor predeterminado, 211
formato pretty-print en el archivo de salida, 403
habilitar procesamiento de entrada, 403
nombre (configuración del componente), 403
opciones de codificación, 403
ordenación de datos, 182
varias entradas/ salidas, 268
varias entradas/salidas, 270
varios archivos de entrada / salida, 56
Componente de entrada,
valor predeterminado, 211
Componentes,
secuencia de procesamiento, 127
Components,
multi file input/output, 57
Comprar software de Altova, 431
Comprobación,
nodos, 216
Compute when,
variable, 290
Con nombre,
plantilla (espacios de nombres), 332
concat, 366
Concatenar,
filtros, 180
Condición,
IF-Else ampliable, 401
Conector,
copiar con Ctrl, 106
crear asignación con, 106
emergente, 106
nombre, 408
propiedades, 77
Conectores,

Conectores,
copia total, 121
Conexión,
configuración, 408
eliminar, 77
mover conectores primario/secundario, 77
propiedades, 77
Conexión automática,
elementos secundarios, 77, 106
Conexiones,
basadas en el tipo, 121
Configuración,
agregar a recurso global, 242
cambiar de componente, 403
cambiar de, recurso global, 247
conector, 408
copiar ya existente, 242
nombre del componente, 403
Configuración del componente,
nombre del componente, 403
Conservar conexiones,
tras la eliminación, 77
Conservar datos,
cuando se usa una asignación de valores, 195
Consolidar datos,
unir archivos XML, 208
Constante,
como valor predeterminado, 211
Constructor,
XSLT2, 336
constructores,
xs:ENTITY, 379
contains, 366
Contenedoras,
clases, 420
Contenido mixto,
asignación, 113
asignación basada en el origen, 113
ejemplo de asignación, 118
método de asignación, 111
Context,
priority context, 206
Contexto,
invalidar, 220
prioridad, 145
Contexto primario,
variable, 290
Contrato de licencia para el usuario final, 501, 505
Controlador,
JDBC, 403
Conversión,
funciones booleanas, 204
Convertir,
en esquema de destino, 403
Convertir rutas de acceso,
en rutas absolutas en el código generado, 278
Copia total, 121
conectores, 121
método de asignación, 111
resolver / eliminar conectores, 121
y filtros, 121
Copiar,
conector disponible, 106
Core,
funciones de la biblioteca, 341
Cortar,
mover conectores primario/secundario, 77
count, 341
count, sum, avg,
función de agregado, 201
Crear,
función, 31
funciones definidas por el usuario, 295
Criterio de ordenación,
cambiar, 182
Cuadrícula,
alineación, 66
guía de alineación, 66
current, 390
current-date, 380
current-dateTime, 380
current-time, 380

D

Datos,
filtrado, 35
Datos de tabla,
ordenar, 182
default-collation, 380
Definida por el usuario,
entrada compleja, 313
función (inline o estándar), 305
salida compleja, 318

© 2014 Altova GmbH

Definida por el usuario,
salida si bool = false, 324
Definidas por el usuario,
cambiar de tipo, 295
cambiar el tipo de la función, 295
eliminar, 295
funciones, 295
funciones (complejas), 312, 318
funciones (restricciones), 295
funciones anidadas, 324
funciones de búsqueda, 307
funciones estándar, 307
funciones y archivos MFF de MapForce, 295
importar/exportar, 295
Derivados,
usar / asignar tipos derivados, 159
Destino,
elemento raíz, 203
varios esquemas, 40
ver varios esquemas, 44
Destino único,
varios orígenes, 208
Dinámicos,
archivos de entrada en tiempo de ejecución, 270
compatibilidad con varios archivos, 268
nombres de archivo como parámetros de entrada, 273
distinct-values, 363
Distribución,
de productos de software de Altova, 501, 502, 504
divide, 357
document, 390
document-uri, 378
DoTransform,
archivo por lotes de AltovaXML, 39
DoTransform bat,
transformar XML, 44
DoTransform.bat,
ejecutar con RaptorXML Server, 18
DTD,
origen y destino, 158
Duplicado,
agregar delante/detrás, 403
Duplicar,
conector, 106
elementos de entrada, 51

E

EDI,
validación, 89
Edición, 400
Ejemplo,
asignación recursiva definida por el usuario, 230
Ejemplos,
carpeta del tutorial, 20
element-available, 390
Elemento,
convertir en destino, 403
duplicar, 51
esquema, asignación, 28
recursivo en XML Schema, 230
Elemento de destino,
asignación de varios orígenes, 46
Elemento raíz,
crear nuevo, 77
Elemento secundarios,
eliminar, 77
Elementos,
desaparecidos, 82
obligatorios, 106
Elementos desaparecidos, 82
Elementos marcados,
elementos desaparecidos, 82
Elementos secundarios,
conexión automática, 77, 106
Eliminar,
conexiones, 77
conexiones de copia total, 121
eliminaciones, elementos desaparecidos, 82
función definida por el usuario, 295
En cadena,
asignación, generación de código, 137
en contexto,
parámetro, 341
ends-with, 385
Entrada,
como parámetros de la línea de comandos, 210
comparar nodos booleanos, 204
instancia XML, 403
parámetro de la línea de comandos, 211
parámetros opcionales, 324

Entrada,
valor predeterminado, 324
varios archivos, 270
equal, 355
equal-or-greater, 355
equal-or-less, 355
Error,
validación, 89
ErrorLevel,
ejecución de la línea de comandos, 92
escape-uri, 385
Espacio de nombres,
plantilla con nombre, 332
Especificar valor,
componente de entrada - vista previa, 211
Esquema,
agregar ubicación al archivo de salida, 403
archivo de catálogo, 94
asignación de XML, 157
elementos recursivos, 230
generar automáticamente a partir de un archivo XML, 22
quitar nombre del código generado, 403
varios destinos, 40
varios esquemas de origen, 403
ver varios destinos, 44
Esquema XML,
generar automáticamente, 22
Estándar,
asignación con secundarios, 119
asignación de contenido mixto, 119
biblioteca XSLT, 145
frente a asignación basada en el origen, 119
función definida por el usuario, 307
método de asignación, 111, 112
Existe,
comprobación de nodos, 216
exists, 360
usar función not-exist para asignar nodos que faltan, 218
Explícito,
tipo de datos, 336
Exportar,
funciones definidas por el usuario, 295
Expresión,
regular, 374
Expresiones regulares, 374
Extensiones de Altova,
funciones para gráficos, 445

F

false, 379
File,
multi file input / output, 57
File:,
(default), 57
item in component, 57
Filtrado,
asignar elementos primarios, 180
componente: consejos, 180
Filtrar,
complemento, 35
conector de copia total, 121
datos, 35
unir archivos XML, 208
Filtro,
contexto prioritario, 180
no concatenar, 180
floor, 357
FlowForce,
implementar en, 395
Flujo de trabajo,
iniciar, recurso global, 259
usando recursos globales, 253
Flujo de trabajo en la aplicación,
usando recursos globales, 253
format-dateTime, 344
format-number, 344, 390
From, 381
Función, 353, 413
agregado, 201
agregar, 145
agregar función XSLT 2.0 personal, 336
agregar función XSLT personal, 332
anidada definida por el usuario, 324
biblioteca, 145
biblioteca core, 341
biblioteca xpath2, 378
biblioteca xslt, 388
buscar en la biblioteca, 145
cambiar el tipo de una función definida por el usuario, 295
compleja (inline), 305
consulta, 145
conversión booleana, 204

Función, 353, 413
de búsqueda definida por el usuario, 307
de suma SUM, 337
definida por el usuario, 232, 295
definida por el usuario (cambiar de tipo), 295
definir, 294
entrada como parámetro, 210
estándar definida por el usuario, 307
exportar función definida por el usuario, 295
extensible, 145
generador visual, 295
IF-Else ampliable, 401
importar función definida por el usuario, 295
inline, 305
personalizada, 145
restricciones en funciones definidas por el usuario, 295
función definida por el usuario,
recursiva, 232
Funciones,
asignación de, 31
referencia, 340
Funciones de extensión .NET,
campos de instancia, 489
campos estáticos, 489
constructores, 488
conversiones de tipos de datos, 490, 491
métodos de instancia, 489
métodos estáticos, 489
para XSLT y XQuery, 485
resumen, 485
tipos de datos .NET en XPath/XQuery, 491
tipos de datos XPath/XQuery en .NET, 490
Funciones de extensión .NET para XSLT y XQuery,
ver Funciones de extensión .NET, 485
Funciones de extensión en scripts MSXSL,
msxsl:script, 491
Funciones de extensión Java,
archivos de clases definidos por el usuario, 478
archivos JAR definidos por el usuario, 481
campos de instancia, 483
campos estáticos, 482
constructores, 482
conversiones de tipos de datos, 484, 485
métodos de instancia, 483
métodos estáticos, 482
para XSLT y XQuery, 476
resumen, 476
tipos de datos Java en XPath/XQuery, 485
tipos de datos XPath/XQuery en Java, 484
Funciones de extensión Java para XSLT y XQuery,
ver Funciones de extensión Java, 476
Funciones de extensión para XSLT y XQuery, 476
function-available, 390

G

Generación de código,
convertir rutas de acceso en rutas absolutas, 278
de asignaciones en cadena, 137
nombre de salida predeterminado del archivo, 39
parámetros de entrada, 210
versión compatible con clases contenedoras, 420
y ruta de acceso absoluta, 40
Generado,
nombre de salida del archivo, 39
Generador,
de funciones definidas por el usuario, 295
Generador visual de funciones, 295
Generar, 44
código y funciones inline, 305
esquema XML automáticamente, 22
varios archivos Java de destino, 44
varios XSLT de destino, 44
generate-id, 390
get-fileext, 351
get-folder, 351
Globalresource.xml,
definición de recursos, 240
greater, 355
group-adjacent, 363
group-by, 363
group-ending-with, 363
group-starting-with, 363
Grupo de nodos,
complemento, 35
Grupos,
bucles y jerarquías, 148
Guardar,
datos de la ventana Resultados, 44
Guía,
de alineación, alineación automática de componentes, 66

H

Habilitar procesamiento de entrada,
optimización, 403
Health Level 7,
ejemplo, 176
Herramientas, 420
HL7 2.6 - 3.x,
ejemplo, 176
Hoja de estilos,
pestañas de StyleVision, 71
HTML,
pestaña, de la ventana Asignación, 62
Stylevision Power Stylesheets, 99

I

Icono,
obligatorio (resaltado), 106
Icono de entrada,
asignación, 106
Icono de salida,
asignación, 106
Icono del conector,
emergente, 106
IF-Else,
ampliable, 401
Implementar,
en FlowForce Server, 395
Implícito,
tipo de datos, 336
Importar,
funciones definidas por el usuario, 295
Include,
XSLT, 332
XSLT 2.0, 336
Información general, 495
Información legal, 501
Información sobre derechos de autor, 501
Información técnica, 495
Inline, 295
funciones y tamaño del código, 305
Inline / Estándar,
funciones definidas por el usuario, 305
Insertar, 401
Instalación,
carpeta Examples, 20
Instancia,
instancia XML de entrada, 403
instancia XML de salida, 403
y el nombre de componente, 215
Instancia XML,
entrada, 403
ruta de acceso absoluta, 40
salida, 403
Intercalación,
componente de ordenación, 182
configuración regional, 182
punto de código Unicode, 182
Intercalación de la configuración regional, 182
Introducción a MapForce, 3, 10
Invalidar,
contexto, 220
is-xsi-nil, 360
Iteration,
priority context, 206

J

Java,
generar varios destinos, 44
varios destinos, 40
JDBC,
convertir ADO y ODBC en, 420
Jerarquías,
bucles y grupos, 148

L

lang, 383, 388
last, 380, 388
Lenguajes,
y compatibilidad con varios archivos y archivos dinámicos, 268
less, 355
Licencia, 505
información sobre, 501
Licencia del producto de software, 505

Licencias,
para su software de Altova, 431
Línea de comandos, 92
código de salida, 92
configuración predeterminada y vista previa, 211
nombre de componente, 215
nombres de archivos de entrada dinámicos, 270
parámetro de entrada, 210
parámetros, 92
parámetros de entrada, 273
parámetros y valores de entrada, 210
Local,
esquemas locales (archivos de catálogo), 94
local-name, 383, 388
logical-and, 355
logical-not, 355
logical-or, 355
lower-case, 385

M

main-mfd-filepath, 351
manespace-uri, 383
Mantener datos sin cambios,
pasar a través de una asignación de valores, 195
Manual del usuario,
ver Ayuda en pantalla, 430
MapForce,
introducción, 3, 10
terminología, 12
MappingMap,
toTargetSchemaName, 39
Marca de orden de bytes,
configuración del componente, 403
matches, 385
max, 341
Medición de licencias,
en los productos de Altova, 503
Menú,
Archivo, 395
Ayuda, 429
conexión, 408
Edición, 400
Función, 413
herramientas, 420
Insertar, 401
Resultados, 416
Vista, 418
Métodos de asignación,
basada en el destino, 112
estándar, 112
estándar / mixto / copia total, 111
mfd-filepath, 351
MFF,
y funciones definidas por el usuario, 295
min, 341
min, max,
función de agregado, 201
minOccurs/maxOccurs,
optimizar el procesamiento de entrada, 403
MisDocumentos,
archivos de ejemplo, 20
Mixto,
asignación estándar, 119
modulus, 357
Motores,
de los productos de Altova, 498
Mover,
conectores primario/secundario, 77
MSXML 6.0,
biblioteca, 420
Multi file,
processing (tutorial), 57
multiply, 357

N

name, 383, 388
Namespaces,
and wildcards (xs:any), 171
namespace-uri, 388
nil,
xsi:nil, 164
node, 383
node-name, 378
Nodo,
asignar nodos que faltan, 218
comparar booleanos, 204
comprobación, 216
posición, 220
sumar varios nodos, 337
Nodos,

Nodos,
obligatorios, 106
Nodos que faltan,
asignar nodos que faltan, 218
Nombre,
conector, 408
Nombre de componente,
ejecución de la línea de comandos, 215
Nombres de esquema,
quitar nombre del esquema del nombre de la tabla, 403
normalize-space, 366
normalize-unicode, 385
not-equal, 355
not-exist,
asignar nodos que faltan, 218
not-exists, 360
Nulo,
valores que pueden ser nulos, 164
number, 344, 383

O

Obligatorios,
nodos/elementos, 106
Opcional,
parámetros de entrada, 324
Optimización,
habilitar procesamiento de entrada, 403
Orden,
en el que se procesan los componentes, 127
Ordenación,
componente, 182
Ordenar datos,
componente de ordenación, 182
Output,
previewing, 87

P

Parámetro,
de entrada dinámico, 270
en contexto, 341
función de entrada como, 210, 211
generación de código, 210
línea de comandos, 92, 210
opcional, 324
salida, 324
usar comodines, 210
valor predeterminado, 211
Parámetro de entrada,
dinámico, 270
línea de comandos, 210
y generación de código, 210
y nombres de archivo dinámicos, 273
Parámetro-nombre,
nombre de componente, 215
Parámetros,
ampliar en funciones, 145
Parámetros de la línea de comandos,
comodines entre comillas, 210
Parámetro-valor,
archivo de instancia, 215
parse-dateTime, 344
parse-number, 344
Pasar datos,
sin cambiar a través de una asignación de valores, 195
PDF,
Stylevision Power Stylesheets, 99
Período de evaluación,
de los productos de software de Altova, 501, 502, 504
Personales,
funciones XSLT, 332
funciones XSLT 2.0, 336
Personalizada,
biblioteca, 145
función, 145
Pestañas,
pestañas de resultados de StyleVision, 71
Plantilla,
con nombre (sumar), 337
llamar, 332
Plantilla con nombre,
sumar nodos, 337
Plataformas,
para productos de Altova, 496
Posición,
de la secuencia filtrada, 220
nodo / contexto, 220
position, 360, 380, 388
Predeterminado,
parámetro - componente de entrada, 211
valor de entrada, 324

© 2014 Altova GmbH

Predeterminado/a,
configuración, recurso global, 242
Preguntas frecuentes sobre MapForce, 433
Pretty-print,
en el archivo de salida, 403
Primario,
asignación y filtros, 180
Prioridad,
función, 145
y filtros, 180
Priority context,
defining, 206
Procesador XQuery,
de los productos de Altova, 498
Procesadores,
descargar, 433
Procesadores XSLT,
de los productos de Altova, 498
Processing Instructions,
Adding to target files, 167
Propiedades,
tabla de asignación de valores, 199
Punto de código,
intercalación, 182

Q

QName,
funciones para, 385
Quitar,
conexión, 77
conexiones de copia total, 121
nombre del esquema, 403

R

Raíz,
elemento del destino, 203
RaptorXML Server,
ejecutar una transformación, 18
Recursiva,
asignación definida por el usuario, 230
función definida por el usuario, 232
Recursivas,
llamadas recursivas en funciones, 305
Recurso,
carpeta, 249
propiedades del recurso global, 263
Recurso global,
activar, 247
asignar a componente, 245
carpetas como, 249
configuración predeterminada, 242
copiar configuración, 242
flujo de trabajo, 253
iniciar el flujo de trabajo, 259
propiedades, 263
Recursos globales,
archivo de definición, 240
barra de herramientas, 241
definir archivo de recursos, 242
Referencia, 394
funciones de MapForce, 340
Regex, 374
Registrar el software de Altova, 431
Relativas,
rutas de acceso (ventajas), 278
remove-fileext, 351
remove-folder, 351
replace, 385
replace-fileext, 351
Requisitos de memoria, 496
resolve-filepath, 351
resolve-uri, 378
Restablecer,
conectores secundarios, 77
Resultado de la transformación,
recursos globales, 253
Resultados, 403, 416
agregar ubicación del esquema al archivo de salida, 403
guardar datos, 44
instancia XML, 403
validación, 89
validar, 89
Retener datos,
pasar a través de una asignación de valores, 195
round, 357
round-half-to-even, 384
round-precision, 357
RTF,
pestaña, de la ventana Asignación, 62
Stylevision Power Stylesheets, 99

Ruta de acceso,
absoluta durante la generación de código, 40

S

Salida,
definida por el usuario si bool = false, 324
parámetro, 324
varios archivos, 270
Scripts en XSLT/XQuery,
ver Funciones de extensión, 476
Sección,
CDATA, 169
Secuencia,
de procesamiento de los componentes, 127
posición, 220
posición del elemento, 220
Secuencia de procesamiento,
de los componentes de una asignación, 127
Secundarios,
asignación estándar con secundarios, 119
set-empty, 363
set-xsi-nil, 360
Signo de interrogación,
elementos desaparecidos, 82
Sitio web de Altova, 433
SO,
para productos de Altova, 496
Software de acompañamiento,
descargar, 433
Soporte para MapForce, 433
Soporte técnico para MapForce, 433
SPS,
Stylevision Power Stylesheet, 99
starts-with, 366, 385
string, 344, 378
string-join, 341
string-length, 366
StyleVision,
asignación en cadena, componente final, 132, 137
asignar un archivo SPS a un componente, 101
hojas de estilos en la vista previa, 99
HTML, PDF, RTF, Word2007+, 99
pestaña, de la ventana Asignación, 62
pestañas de resultados de la asignación, 71
Subir,
elemento en el componente, 403
substitute-missing, 360
substitute-missing-with-xsi-nil, 360
substring, 366
substring-after, 366, 385
substring-before, 366, 385
Subtract, 357, 381
sum, 341
Suma,
nodos en XSLT 1.0, 337
Sustituir,
nodo que falta, 216
system-property, 390

T

Tabla,
búsqueda, asignación de valores, 190
quitar nombre del esquema, 403
quitar nombre del esquema de la tabla, 403
Tabla de búsqueda,
asignar nodos que faltan, 218
propiedades, 199
tabla de asignación de valores, 190
Tecla de acceso rápida,
teclado, 420
Teclado,
tecla de acceso rápida, 420
Terminología, 12
Tipo,
convertir en tipo de destino, 403
Tipo complejo,
ordenar, 182
Tipo de datos,
explícito / implícito, 336
Tipo simple,
ordenar, 182
Tipos,
tipos derivados (xsi:type), 159
Tokenize, 366, 370
Tokenize-by-length, 366, 370
tokenize-regexp, 366
Transformaciones,
RaptorXML Server, 18
Transformar,
datos de entrada, asignación de valores, 190

© 2014 Altova GmbH

Transformar,
DoTransform.bat, 44
Transformation language,
selecting, 86
translate, 366
true, 379
Tutorial, 20
carpeta Examples, 20

U

Unicode,
intercalación de punto de código, 182
Unir,
archivos XML, 208
unparsed-entity-uri, 390
upper-case, 385
URI,
en documentos DTD, 158
URI de espacio de nombres,
DTD, 158
Usar,
recursos globales, 247
Uso de Internet,
en los productos de Altova, 500

V

Validador,
de los productos de Altova, 497
Validar,
datos de la ventana Resultados, 44
proyecto de asignación, 89
resultados, 89
Valor,
componente de entrada - vista previa, 211
predeterminado, 324
Variable,
casos de uso, 284
insertar, 284
intermedia, 284
Variables intermedias, 284
Varias,
entras / salidas, 268
Varios,
destinos y Java, 40
elementos de entrada, 51
esquemas de destino, 40
esquemas de origen, 403
ver varios esquemas de destino, 44
Varios archivos,
compatibilidad con, lenguajes, 268
entrada / salida, 56
entrada/salida, 270
Varios archivos de origen,
en un solo destino, 208
Varios archivos XML,
a partir de un solo XML de origen, 275
Varios orígenes,
asignar a un solo destino, 46
Ventana Asignación,
alineación automática, 66
pestañas de StyleVision, 62
Versión,
compatibilidad de las clases contenedoras, 420
Vista, 418
Vista previa,
Stylevision Power Stylesheets, 99
valor del componente de entrada, 211
Visual Studio,
versiones compatibles (generación de código), 420

W

Wildcards,
xs:any - xs:anyAtrribute, 171
Windows,
compatibilidad con productos de Altova, 496
Word2007+,
pestaña, de la ventana Asignación, 62
Stylevision Power Stylesheets, 99

X

Xerces,
bibliotecas, 420
XML,
generar esquema XML a partir de, 22

© 2014 Altova GmbH

XML en XML, 157
xpath, 388
sumar varios nodos, 337
xpath2,
biblioteca, 145
funciones de la biblioteca, 378
XQuery,
funciones, 145
funciones de extensión, 476
xs:,
constuctores, 379
xs: any (xs:anyAttribute), 171
xs:date, 380
xs:time, 380
Xsi:nil,
valores que pueden ser nulos, 164
xsi:type,
asignación a tipos derivados, 159
XSLT,
agregar funciones personales, 332
biblioteca estándar, 145
espacio de nombres de la plantilla, 332
funciones de extensión, 476
funciones de la biblioteca, 388
generar varios destinos, 44
pestaña, de la ventana Asignación, 62
previewing generated code, 88
XSLT 1.0/2.0, 39
archivo por lotes DoTransfrom, 39
generar (tutorial), 39
XSLT 2.0,
agregar funciones personales, 336

Z

Z-A,
componente de ordenación, 182