InterConnect in SOA perspectief

Gebruik van de FTP/file-adapter

Interfacen met files is al zo oud als de file zelf. Dat betekent dat dergelijke interfaces nog erg veel voorkomen. Dat het tegenwoordig hip is om met webservices te interfacen is in de praktijk dus niet altijd bruikbaar. Wanneer een van de legacy-applicaties die met files interfacet, wordt vervangen, blijft vaak de noodzaak bestaan de file-interface in stand te houden voor de andere oude legacy-applicatie. Zeker bij bedrijven die OracleAS Integration InterConnect reeds hebben ingezet voor interfacing, is de inzet van de FTP/file-adapter een goede optie voor het interfacen met files.

In dit artikel wordt dit uitgewerkt aan de hand van een eenvoudig voorbeeld. Uitgangspunt daarbij is een interface volgens het Publish-Subscribe principe, waarbij de Publishing Application al is ingericht en de FTP/file-adapter als Subscribing Application optreedt. Daarbij wordt een datafile gecreëerd die *geen* XML-structuur bevat, zoals bij veel legacy-applicaties het geval is.
Eerst wordt bekeken hoe OracleAS Integration InterConnect zich verhoudt tot SOA, en wordt even kort de architectuur van OracleAS Integration InterConnect geschetst.
Het SOA-geweld komt met bakken over ons heen en we duiken er met z'n allen bovenop. Dat is een prima ontwikkeling in het licht van de standaardisatie van onder meer integratie. In de visie van Oracle is OracleAS Integration InterConnect onderdeel van hun SOA-oplossing. Het is bedoeld voor de connectiviteit van zowel legacy als bijvoorbeeld ook Oracle BPEL Process Manager. Dat biedt dus uitstekende mogelijkheden om bestaande interfaces naadloos te combineren met het gebruik van nieuwe technologieën als webservices. Voor bedrijven die hun integratie al hebben gerealiseerd met OracleAS Integration InterConnect, bestaat daarmee de mogelijkheid hun bestaande infrastructuur beschikbaar te stellen als SOA.

Globale architectuur

In tegenstelling tot een ESB (Enterprise Service Bus) waarbij alle services aanknopen op een 'globaal netwerk', berust de architectuur van InterConnect op het 'Hub and Spoke'-principe. De hub is de centrale naaf die alle berichten verwerkt van de applicaties die als spaken zijn ingeplugd. Dit heeft onder meer als voordeel dat alle berichten volgens één standaardprotocol via de hub worden verzonden, waarbij FIFO wordt toegepast met behulp van Oracle AQ. Daarnaast wordt iedere applicatie met één adapter benaderd, waardoor slechts één verbinding noodzakelijk is per applicatie. De techniek van het verzenden van een bericht is losgekoppeld van het mappen van de data-attributen. Een nadeel van deze architectuur is dat de hub een single point of failure kan zijn. Oracle levert een groot aantal standaardadapters, zoals bijvoorbeeld database-, AQ-, SMTP-, HTTP- en FTP/file-adapter. Daarnaast levert Oracle nog Application Adapters onder een aparte licentie, onder meer voor SAP en Siebel.

Zoom in op FTP/file-adapter

Doel van de adapter

Wanneer een legacy-applicatie moet interfacen met behulp van files, kan de FTP/file-adapter worden gebruikt. Deze adapter kan zowel XML- als niet-XML-files lezen en schrijven. Het verwerken van niet-XML-files gebeurt aan de hand van een vooraf gedefinieerde XML-structuur. Voor de definitie van deze XML-structuur wordt de d3l.dtd. D3L (Data Definition Description Language) lijkt in eerste instantie complex, maar uit het volgende voorbeeld zal blijken dat dit erg mee valt. Er kan op redelijk eenvoudige wijze een opmaak van een datafile worden geconfigureerd.
Zonder dat ook maar één regel code wordt geschreven, kan met deze adapter een datafile worden gelezen of geschreven. Daarbij kan de datafile in iedere directory op ieder filesysteem worden verwerkt. De adapter kan files rechtstreeks vanuit een directory verwerken of via een FTP-server.

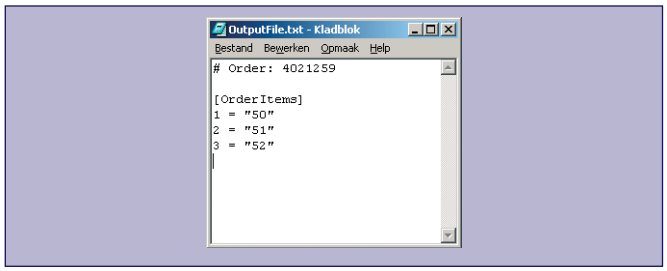
Onderdelen FTP/file-adapter

De FTP/file-adapter bestaat uit een aantal onderdelen. Design Time wordt de opmaak van de file geconfigureerd met een

D3L XML file. Daarnaast wordt de adapter als applicatie in iStudio ingevoerd, waardoor de data-attributen uit de Applicatie View kunnen worden gemapt met die van de Common View. Op de applicatieserver wordt de adapter geïnstalleerd. Daarbij wordt in de adapter.ini file geconfigureerd hoe de files run time worden verwerkt.

Configureer Design Time

D3L is een XML-structuur (DTD) waarmee de structuur van een niet-XML-datafile kan worden geconfigureerd in XML en vertaald naar een Application View in iStudio. Een datafile kan bijvoorbeeld bestaan uit gestructureerde records van bytes en of karakters.
De D3L XML-configuratie van een te verwerken datafile wordt Design Time geïmporteerd in iStudio om de Application View te maken. Voor Run Time verwerking van de datafile wordt deze XML-file bij de FTP/file-adapter geplaatst, zodat de adapter deze kan gebruiken bij de vertaling.
Hieronder wordt een voorbeeld uitgewerkt van de opbouw van een D3L-file voor een voorbeeld datafile. De voorbeeld datafile OutputFile.txt die middels de FTP/file-adapter wordt geladen, ziet er als volgt uit:



Afbeelding 1. Datafile OuputFile.txt

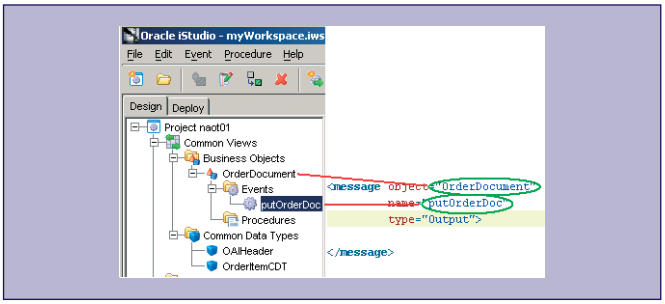
De file bestaat uit een aantal vaste tekstelementen als '# Order' en '[OrderItems]'. Verder zijn er dynamische data-elementen die vanuit een andere applicatie worden gevuld, waaronder een array van orderitems. Hierna is een basisopzet van de D3L-file 'OrderDocD3L.xml' weergegeven die correspondeert met de Common View in iStudio. Dit is een uitgeklede versie die hierna verder wordt aangevuld.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE message SYSTEM "d31.dtd">
<!-- file: OrderDocD3L.xml -->

<!-- Onderstaande message correspondeert met de configuratie in istudio
object is de naam van het Business Object,
name is de naam van het Event en
type is de Name van de Attributes bij het event van de Application View -->
<message object="OrderDocument" name="putOrderDoc" type="Output">

</message>
```

Afbeelding 2. Basis D3L-file OrderDocD3L.xml



Afbeelding 3. Common View-D3L mapping

Het attribuut 'object' van het element message verwijst naar de naam van het Business Object in iStudio en het attribuut 'name' verwijst naar de naam van het Event in de Common View. Hiermee wordt de link gelegd met het Event, zodat Run Time de juiste mapping kan plaatsvinden.

Als deze D3L-file later in iStudio wordt geïmporteerd, wordt de naam 'Output' van het attribuut 'type' als Application View opgeslagen.

Hieronder wordt het type 'Output' verder gevuld met de attributen die in de datafile staan.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE message SYSTEM "d31.dtd">

<message object="OrderDocument" name="putOrderDoc" type="Output">
  <struct id="Output">

    <!-- Maak een veld OrderPrompt, dat in istudio wordt
    gevuld met de literal "# Order:" -->
    <field name="OrderPrompt"> <termstring endchar=" " /> </field>

    <!-- Maak een veld OrderNo, dat in istudio wordt
    gevuld met attribute "OrderId" -->
    <field name="OrderId"> <termstring endchar="\r\n\r\n" /> </field>

    <!-- Maak een veld OrderItemTag dat in istudio wordt
    gevuld met "[OrderItems]" -->
    <field name="OrderItemTag"> <termstring endchar="\r\n" /> </field>

  </struct>
</message>
```

Afbeelding 4. D3L-file OrderDocD3L.xml met Output struct

De D3L-file is nu binnen het element 'message' verder aangevuld met het element 'struct', met als id 'Output'. 'Output' legt de verbinding naar het type 'Output' van het element 'message'. Binnen de struct zitten 3 'field'-elementen, die ook werkelijk de velden in de datafile vertegenwoordigen. De 'name'-attributen zijn de attribuutnamen, die in iStudio worden gebruikt voor de mapping. Bij het element 'termstring' kan worden aangegeven wat het karakter is waarmee het dataveld wordt afgesloten. In de datafile wordt het attribuut OrderPrompt (# Order:) gevolgd door het attribuut OrderId (4021259), gescheiden door een spatie. In de D3L-file wordt dit gerealiseerd door de spatie als terminator te zetten na het attribuut OrderPrompt (<termstring endchar=" " />).

In de datafile wordt het attribuut OrderId (4021259) gevolgd door het attribuut OrderItemTag ([OrderItems]) gescheiden door 2 harde returns. In de D3L-file wordt dit gerealiseerd door de 2 harde returns (\r carriage return + \n line feed) als terminator te zetten na het attribuut OrderId (<termstring endchar=" \r\n\r\n" />). Dat geldt ook voor het attribuut OrderItemTag.
Er bestaan nog veel meer elementen die in combinatie met het 'field' element gebruikt kunnen worden, zoals 'limstring' voor delimiters of 'pfxstring' voor strings met een vaste lengte.
Hieronder wordt een nieuwe structuur gedefinieerd voor de array, die later wordt gebruikt als type voor het attribuut OrderItemArray.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE message SYSTEM "d3l.dtd">

<message object="OrderDocument" name="putOrderDoc" type="Output">

  <!-- Maak een structuur type voor de orderitems dat hieronder wordt
  gebruikt in de struct met id Output bij field name OrderItemArray. -->
  <struct id="ItemRegelType">
    <!-- Maak een array structuur voor het vullen van de tabel met
    orderitems. -->
    <field name="OrderItems">
      <limarray contchar="\n" endchar="\r\n">
        <struct>
          <field name="Volgnr"> <termstring endchar=" " /> </field>
          <field name="OrderItem"> <termstring endchar="\r" />  </field>
        </struct>
      </limarray>
    </field>
  </struct>

  <struct id="Output">
    <field name="OrderPrompt"> <termstring endchar=" " />  </field>
    <field name="OrderId"> <termstring endchar="\r\n\r\n" /> </field>
    <field name="OrderItemTag"> <termstring endchar="\r\n" /> </field>

    <!-- Gebruik het hierboven gedefinieerde type ItemRegelType voor
    het vullen van de tabel met orderitems. -->
    <field name="OrderItemArray"> <typeref type="ItemRegelType" /> </field>

  </struct>
</message>
```

Afbeelding 5. D3L-file OrderDocD3L.xml met ItemRegelType struct

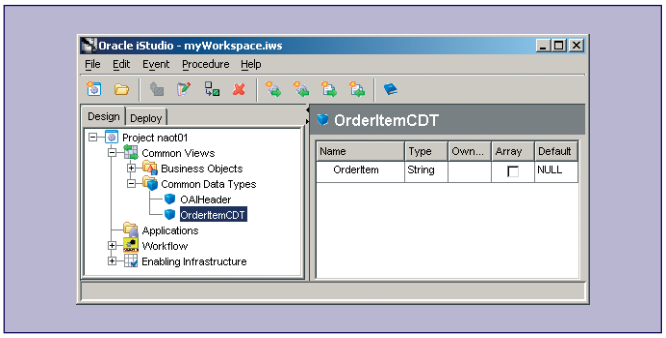
De D3L-file is nu binnen het element 'message' verder aangevuld met het element 'struct', met als id 'ItemRegelType'. Deze struct wordt bij het attribuut OrderItemArray onder in de file gebruikt als type.
Binnen de struct 'ItemRegelType' zit het 'field'-element OrderItems die een array van twee field-elementen bevat, namelijk 'Volgnr' en 'OrderItem'. Ook deze 'name'-attributen zijn de attribuutnamen die in iStudio worden gebruikt voor de mapping.
In de datafile wordt het attribuut OrderItemTag ([OrderItems]) gevolgd door het attribuut OrderItems. Het attribuut OrderItems is een array van de attributen 'Volgnr' en 'OrderItem' (bijvoorbeeld 1 = "50").
In de D3L-file wordt dit gerealiseerd door het element 'limarray'. Binnen dit array element zijn de twee velden 'Volgnr' en 'OrderItem' gedefinieerd, gescheiden door een spatie (<termstring endchar=" " />).
In de datafile wordt iedere OrderItem regel gevolgd door een harde return. In de D3L-file wordt dit gerealiseerd door de carriage return (\r) als terminator te zetten na het attribuut OrderItem (<termstring endchar=" \r" />) en de line feed (\n) te zetten als *scheidingsteken* van de array (<limarray contchar="\n"). De array wordt afgesloten met een harde return middels het attribuut endchar van het element limarray (<limarray contchar="\n" *endchar="\r\n"*>).
Hiermee is de gehele configuratie van de D3L-file voor dit voorbeeld compleet. Er bestaan echter nog veel meer mogelijkheden voor het laden en vertalen van andere type attributen. Kijk daarvoor in Appendix C van de Oracle AS Integration InterConnect User's Guide.

Configureer Design Time met iStudio

Nu de D3L-file gereed is, kan deze worden gebruikt voor het creëren van de Common View en de Application View in iStudio.

Common Data Type als array

Voor het creëren van de Common View wordt eerst een Business Object en Event aangelegd. De structuur van de attributen in het Event van de Common View moet in ieder geval de data elementen uit de datafile bevatten. In dit geval zijn dat het OrderId en een array van OrderItems. De overige elementen in de datafile kunnen tijdens de mapping van de Application View aan de Common View worden toegevoegd.
De structuur van de Common View wordt in dit voorbeeld handmatig in iStudio aangelegd. Daarvoor wordt eerst een Common Data Type OrderItemCDT aangelegd, waarin het data-attribuut OrderItem zit, zoals is aangegeven in afbeelding 6.

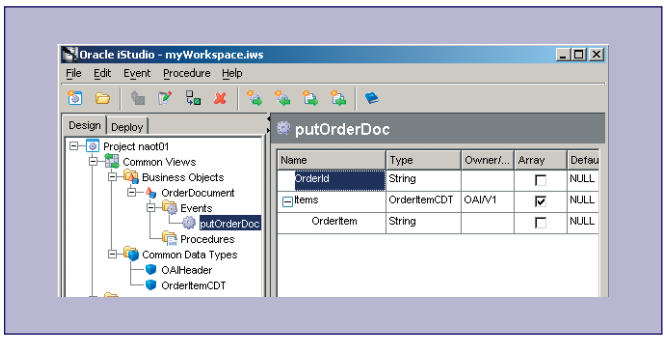


Afbeelding 6. Common Data Type OrderItemCDT.

Dit Data Type wordt gebruikt voor het creëren van de array-structuur 'Items' in het Event putOrderDoc. Daarvoor worden handmatig de attributen OrderId en Items toegevoegd. Het Type van 'Items' wordt gewijzigd in het Common Data Type OrderItemCDT, en het Array-vinkje wordt gezet. Na het opslaan ziet de structuur van het event er als volgt uit.

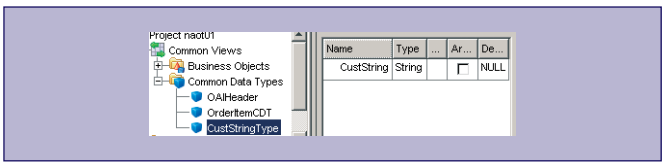
Application Data Type als array

Nu de Common View gereed is kan de Application View worden geconfigureerd en gemapt aan de Common View. Hiervoor



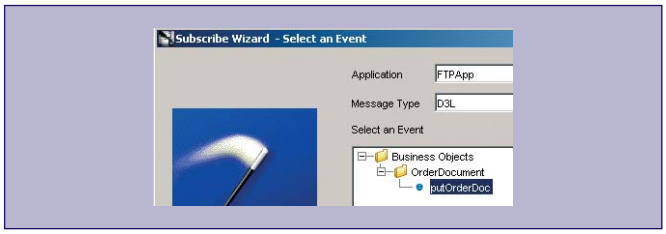
Afbeelding 7. Creëer Common View Event putOrderDoc.

wordt eerst nog even een Common Data Type CustStringType van het type String aangemaakt in de Common View.



Afbeelding 8. Creëer Common Data Type CustString.

Dit Type wordt straks gebruikt als type voor een aantal variabelen, dat wordt gebruikt voor het creëren van de vaste stukken tekst in de datafile (bijvoorbeeld [OrderItems]). Nu wordt een nieuwe Application FTPApp gecreëerd als Subscribed Event van Message Type D3L, gebaseerd op het eerder aangemaakte Event putOrderDoc.



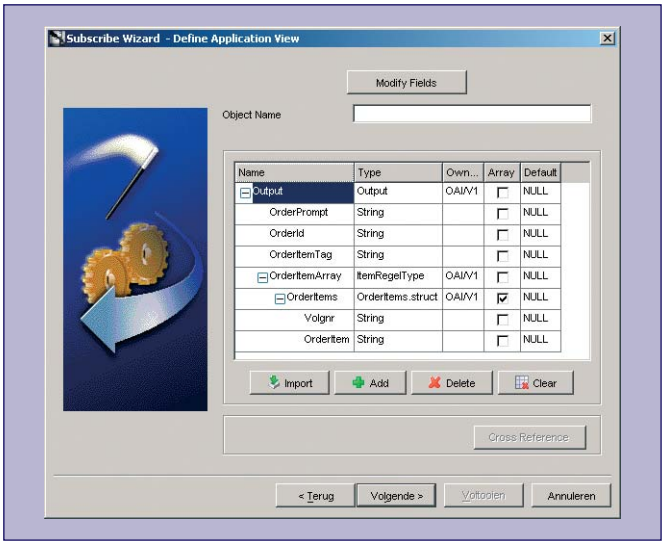
Afbeelding 9. Creëer Application FTPApp.

Nu kan de Application View worden gecreëerd middels een import van de eerder gemaakte D3L-file OrderDocD3L.xml. Dit geeft het volgende resultaat:

De naam 'Output' van deze Application View komt van het attribuut 'type' bij het element 'message' in de D3L-file. Deze Application View wordt nu gemapt aan de Common View.

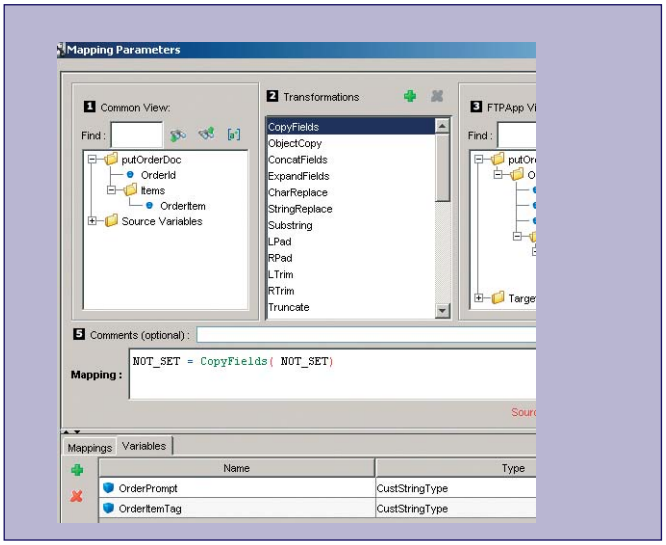
Prompts in file

Voor de vaste stukken tekst '# Order:' en '[OrderItems]' in de datafile, worden respectievelijk de twee variabelen OrderPrompt en OrderItemTag geconfigureerd. Voor het Type van deze variabele, wordt het eerder gecreëerde Common



Afbeelding 10. Geïmporteerde Application View

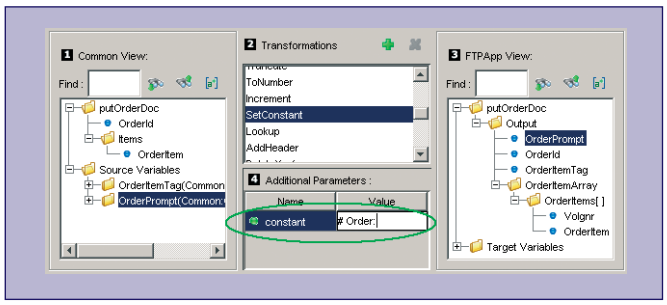
Data Type CustStringType gebruikt. De variabelen kunnen worden gecreëerd onder in het scherm 'Mapping Parameters' onder de tab 'Variables'.



Afbeelding 11. Creëer Variables

In de Common View onder 'Source Variables', kunnen deze variabelen nu worden gebruikt voor de mapping met de Application View.

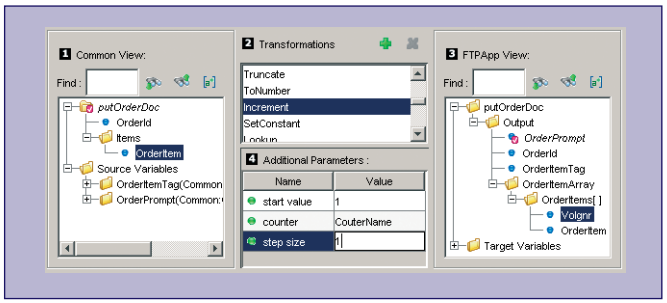
De variabele OrderPrompt wordt met de Transformatie 'SetConstant' gemapt aan OrderPrompt. Onder de Transformations bij '4 Additional Parameters' kan de literal '# Order:' worden ingevoerd. Zo kan de mapping van de OrderItemTag op dezelfde wijze worden gedaan. Het OrderId kan middels de Transformatie CopyFields worden gemapt aan OrderId.



Afbeelding 12. Map Variables.

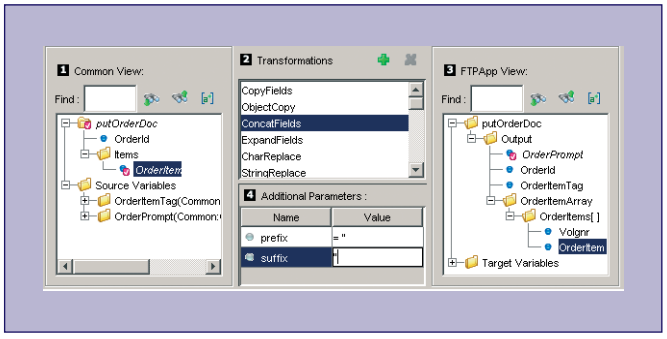
Sequence in datafile

In de datafile staat een sequence voor ieder OrderItem. Dit kan worden bereikt middels de Transformation 'Increment', tussen Common View attribuut OrderItem en Application View attribuut Volgnr. Bij '4 Additional Parameters' kan de sequence in detail worden geconfigureerd.



Afbeelding 13. Map sequence.

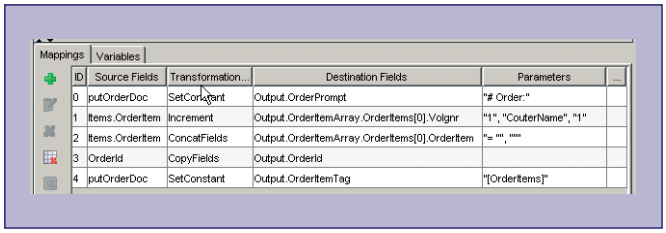
Het data-attribuut OrderItem in de datafile, wordt voorafgegaan door een =, een spatie en een aanhalingsteken, en wordt gevolgd door een aanhalingsteken. Dit kan in de mapping worden bereikt middels de Transformation ConcatFields.



Afbeelding 14. ConcatFields Transformatie.

Bij "4 Additional Parameters" kunnen de prefix (= ") en de suffix (") worden ingegeven. De spatie voor de =, is in de D3L-file opgenomen als afsluitend karakter van het item Volgnr (name="Volgnr"> <termstring endchar=" " />).

Hiermee is de mapping compleet en ziet het complete plaatje er als volgt uit.



Afbeelding 15. Mappings

Configureer Run Time

Nu de Design Time configuratie voltooid is, wordt de Run Time configuratie van de FTP/file-adapter verder uitgevoerd op het file systeem waar de adapter is geïnstalleerd. In de file adapter.ini wordt nog een aantal settings gedaan.

D3L-file

De adapter kan zowel XML als D3L-files verwerken. Dit wordt aangegeven bij de parameter ota.type. In dit geval wordt deze parameter als volgt gezet:

```
ota.type=D3L
```

De adapter moet nu nog weten hoe de datafile is opgemaakt. Dat is eerder geconfigureerd in de D3L-file OrderDocD3L.xml. Deze file wordt aan de adapter bekend gemaakt op de volgende manier:

```
ota.d3ls= OrderDocD3L.xml
```

Deze XML-file wordt in de directory van de FTP/file-adapter geplaatst waar ook de adapter.ini staat. Zo kan de adapter deze file vinden voor de Run Time verwerking van de datafile.

File systeem locatie

De adapter wordt in dit geval gebruikt als file adapter. Dat betekent dat de adapter de file op het filesysteem zet in de directory die wordt aangegeven bij de volgende parameter:

```
ota.send.endpoint=file:///\\SomeWidowsFileServer\
somePath\someOutputDirectory
```

Dit is de notatie die geldt voor een Windows-filesysteem. Voor een Unix-filesysteem kan natuurlijk het absolute pad worden opgenomen zonder de extra slashes ervoor. Nu kan de adapter worden gestart en getest.

Overige mogelijkheden

De FTP/file-adapter kan files natuurlijk ook lezen aan de hand van een D3L-file. In dat geval dient de in de adapter.ini de para-

meter ota.receive.endpoint worden geconfigureerd. De adapter kan tevens files via een FTP-server verzenden en ontvangen. Ook dit kan in de adapter.ini worden aangegeven. Wanneer gegevens moeten worden bewerkt tijdens het lezen of schrijven van de file, kan bij de parameter file.sender.customizer_ class een Java class worden gezet die deze bewerkingen uitvoert. Nu de D3L structuur duidelijk is, kan deze tevens worden ingezet ten bate van de SMTP- en HTTP-adapter.

Conclusie

Het gebruik van de FTP/file-adapter is een krachtig alternatief bij interfacing met behulp van files, zeker bij bedrijven die Oracle Integration InterConnect reeds hebben ingezet. Zonder één regel code te schrijven kan een file zowel worden gelezen als geschreven in iedere directory op ieder filesysteem. Dat Oracle AS Integration Interconnect naadloos in de SOA-oplossing van Oracle past, maakt de toepassing van de FTP/file-adapter nog krachtiger.

Documentatie

Oracle Application Server Integration InterConnect Adapter for FTP Installation and User's Guide 10g Release 2 (10.1.2)
http://download-uk.oracle.com/docs/cd/B14099_19/integrate.1012/b14073/toc.htm

Oracle Application Server Integration InterConnect User's Guide
10g Release 2 (10.1.2)
Appendix C Using the Data Definition Description Language
http://download-uk.oracle.com/docs/cd/B14099_19/integrate.1012/b14069/appx_d3l.htm#i632676

Aard Jan Kram is Senior Oracle Consultant bij Truston Services B.V.