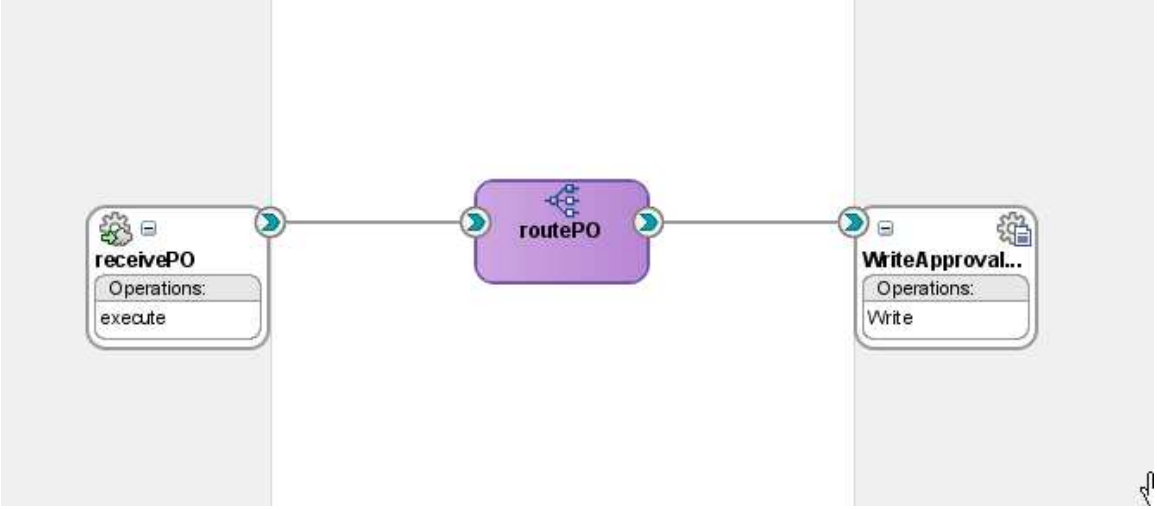
3 Creating the Purchase Order Routing Service

3.1 Introduction

Note: The solution for this chapter can be found in `c:\po\solutions\ch3`.

We will now create another application that will accept new purchase orders and forward them to the fulfillment systems. In our case we will simulate these fulfillment end systems with a simple file adapter that will capture to disk all processed purchase orders.

We will build this application in an incremental fashion, starting with a very simple ESB flow that receives orders coming over SOAP and write them to file, as follows:



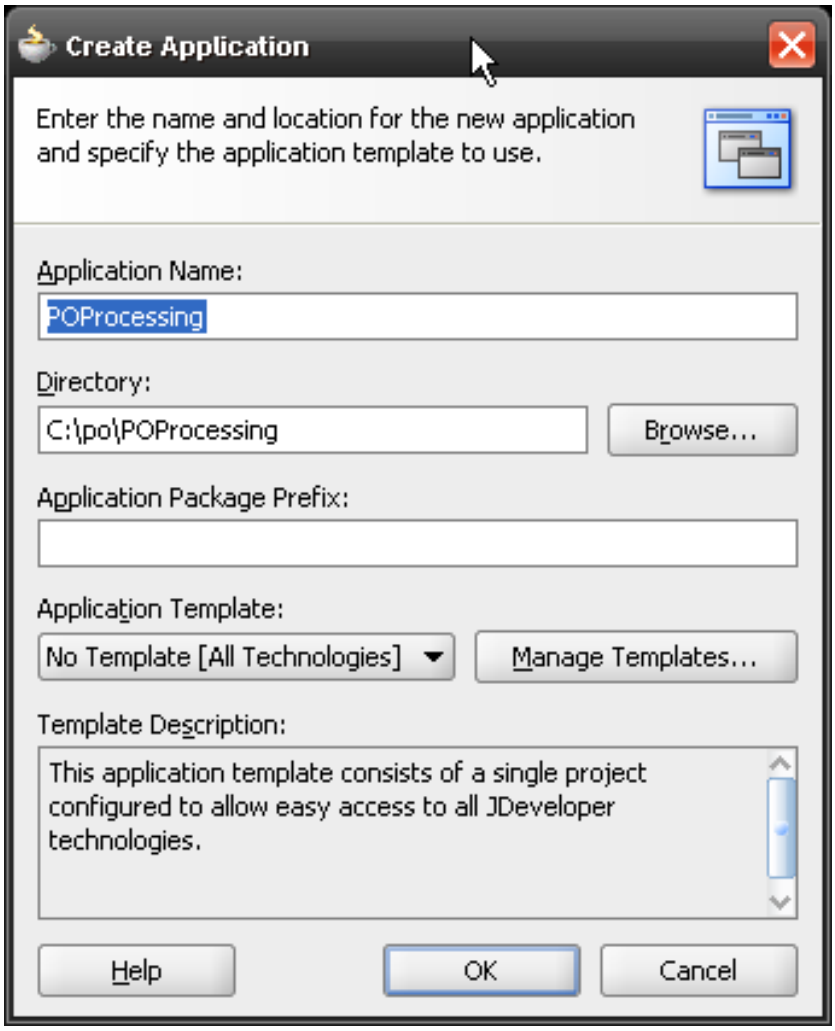
3.2 Designing the flow

You will create a new composite application that will be invoked as a service from your main application to be designed later.

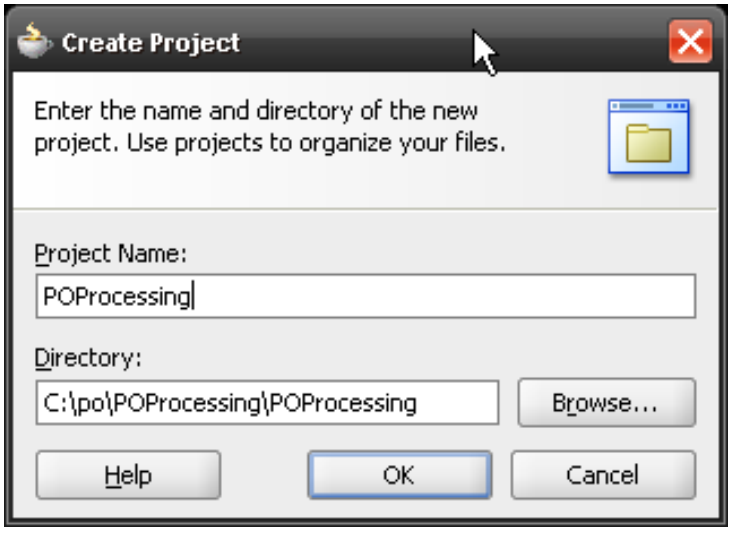
3.2.1 Creating a new application

1. Create a new application following the same sequence of steps as for the previous CreditCardStatus service. As an alternative, instead of using the File/New menu, you can choose "New Application" from the application menu.

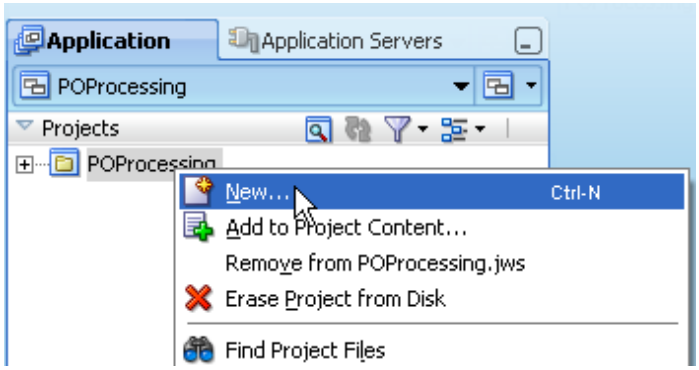
2. Name your application **POProcessing** and make its location in `c:\po`.

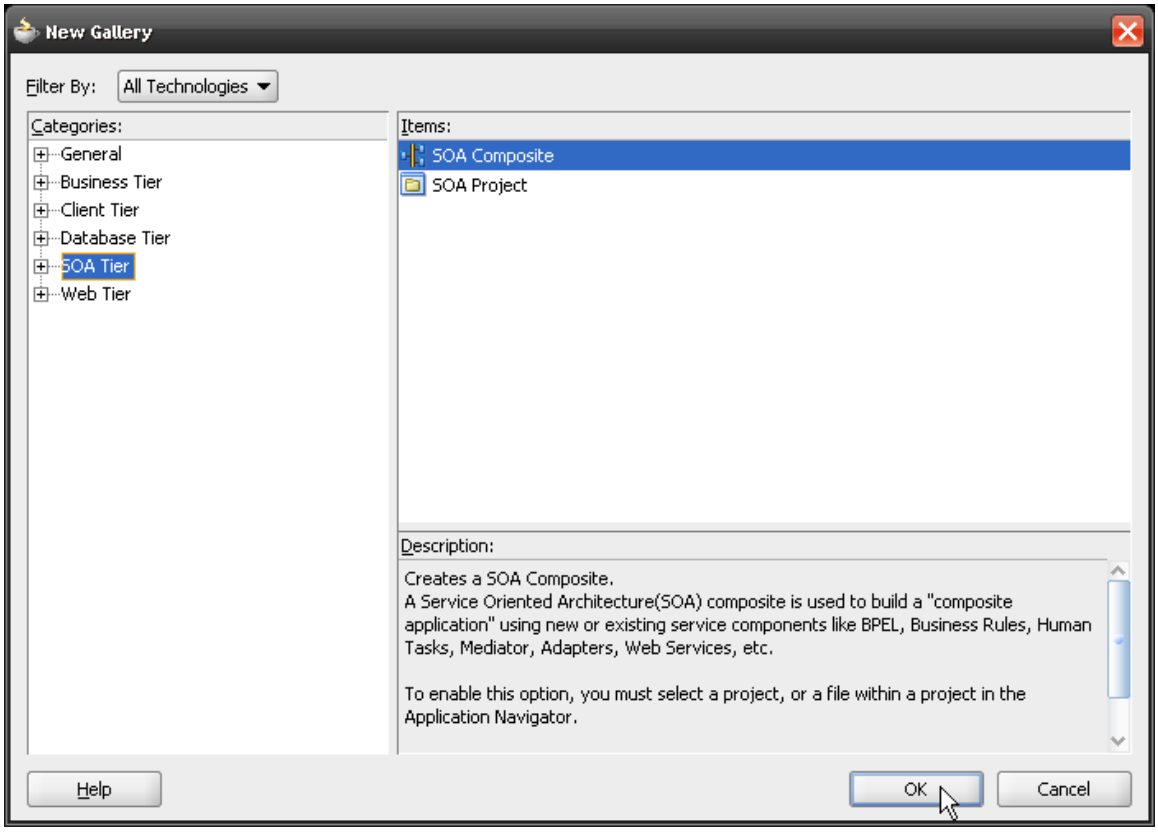


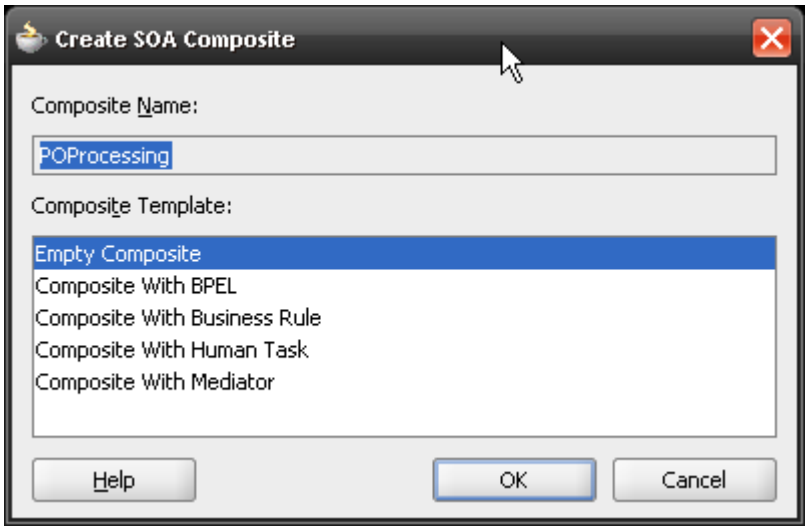
3. Use the same name for the project.



4. Add a SOA Composite to your project.

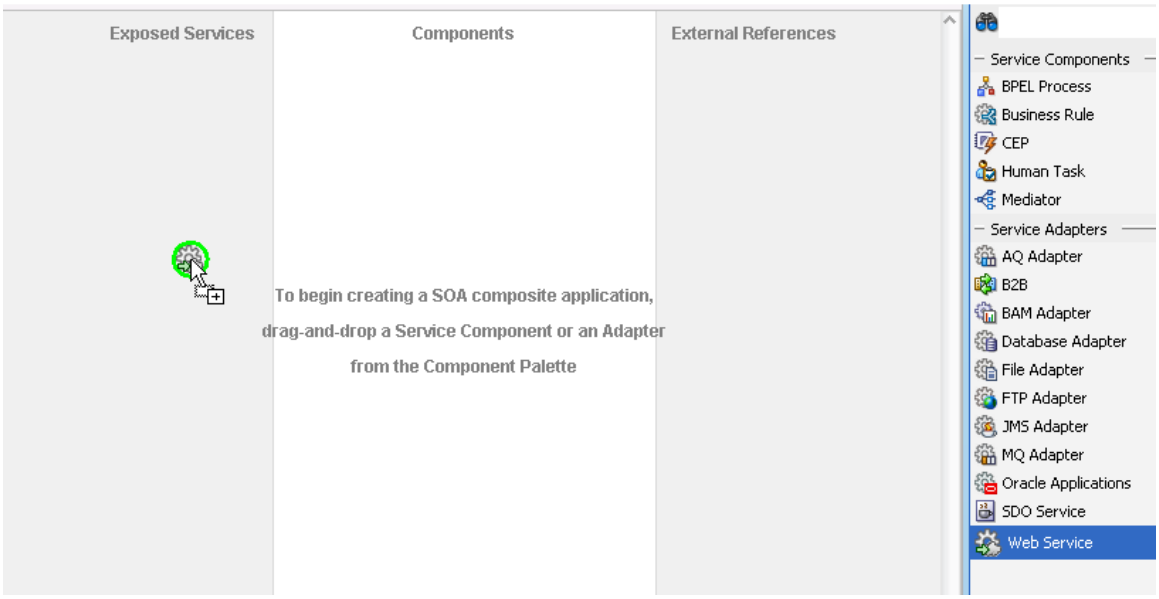

Application
Application Servers
POProcessing
Projects
POProcessing
New...
Ctrl-N
Add to Project Content...
Remove from POProcessing.jws
Erase Project from Disk
Find Project Files


New Gallery
Filter By:
All Technologies
Categories:
General
Business Tier
Client Tier
Database Tier
SOA Tier
Web Tier
Items:
SOA Composite
SOA Project
Description:
Creates a SOA Composite.
A Service Oriented Architecture(SOA) composite is used to build a "composite application" using new or existing service components like BPEL, Business Rules, Human Tasks, Mediator, Adapters, Web Services, etc.
To enable this option, you must select a project, or a file within a project in the Application Navigator.
Help
OK
Cancel

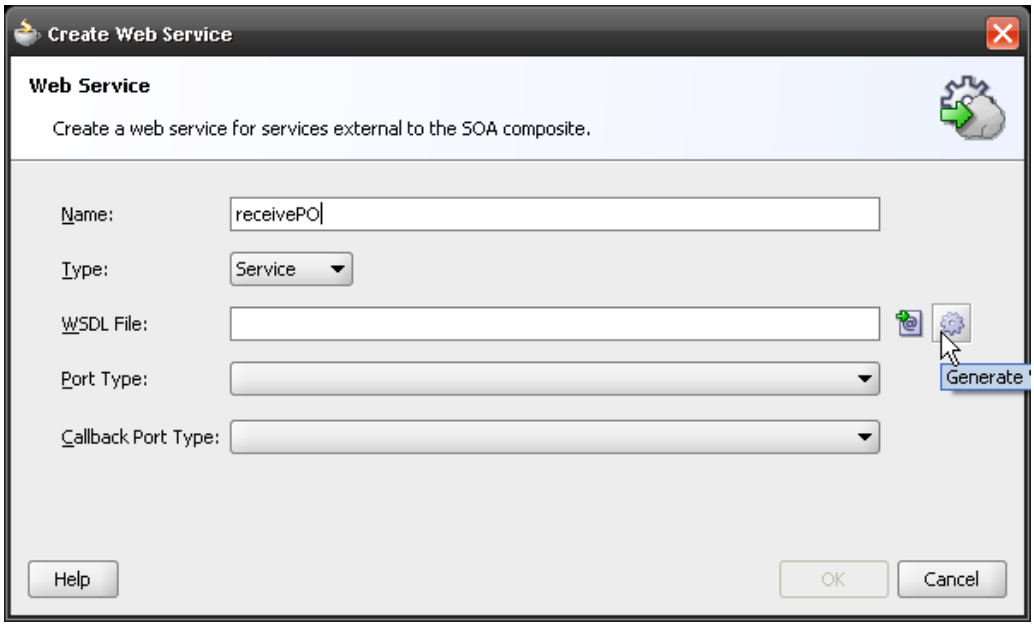


3.2.2 Adding the service interface

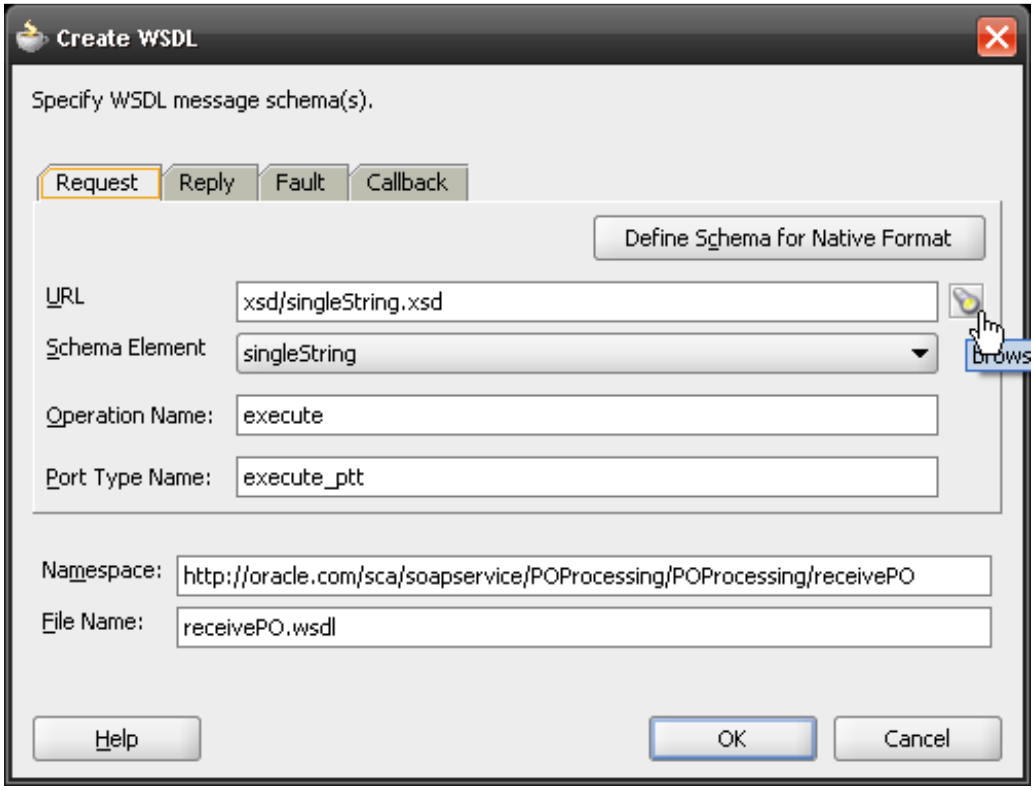
1. Start by creating the service we will use to expose this composite over SOAP. Drag and drop a **Web Service** activity on the **Exposed Services** swim lane

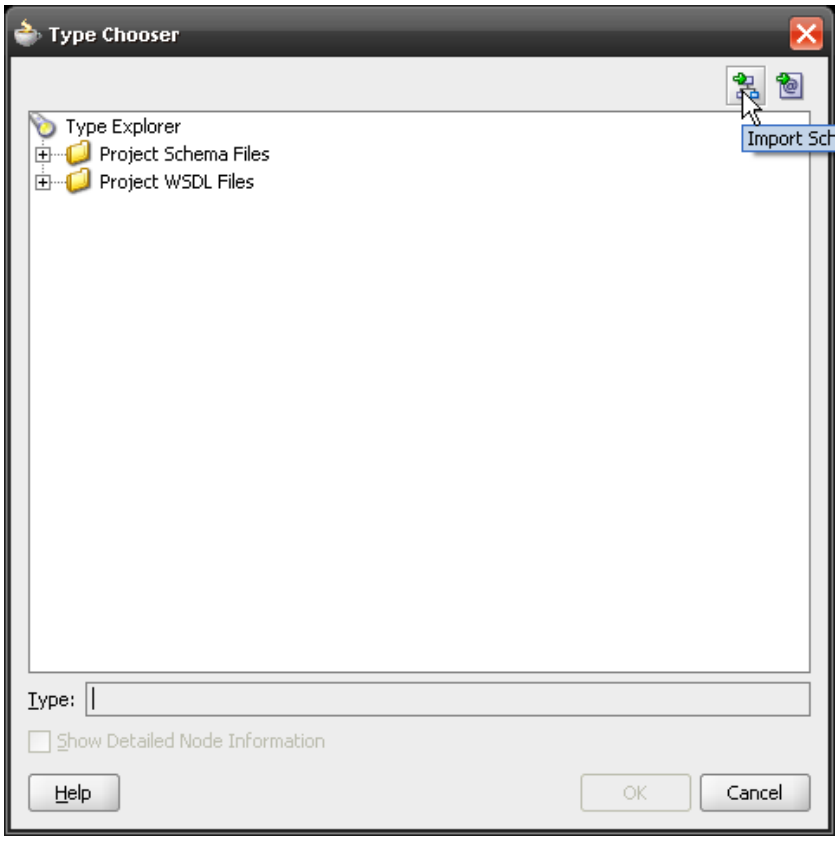


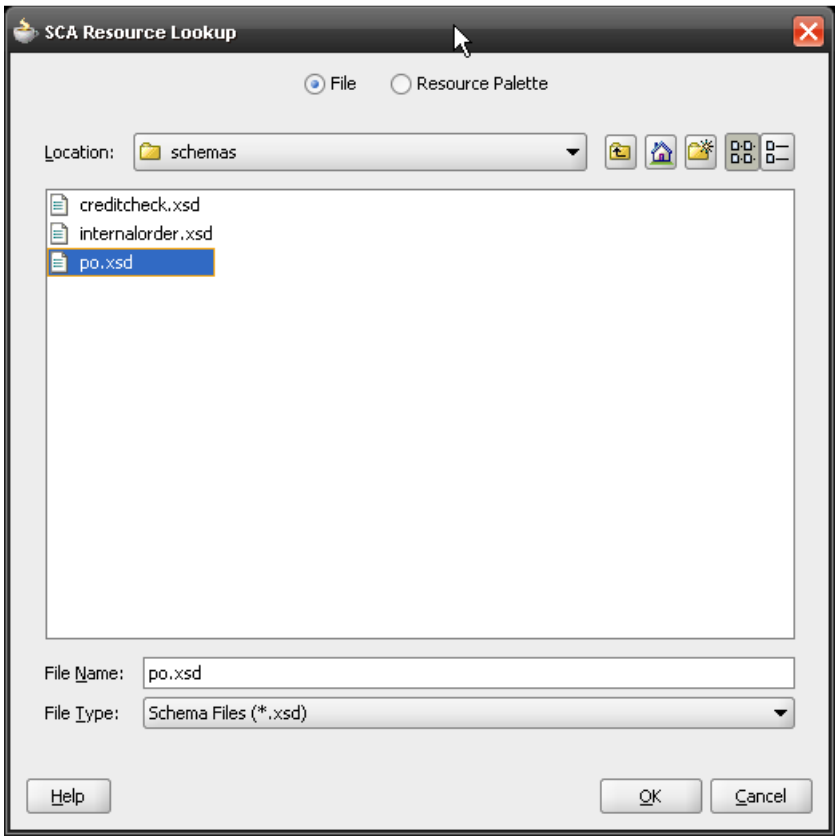
2. Name it **receivePO** and create a WSDL from a schema by clicking the cog icon.



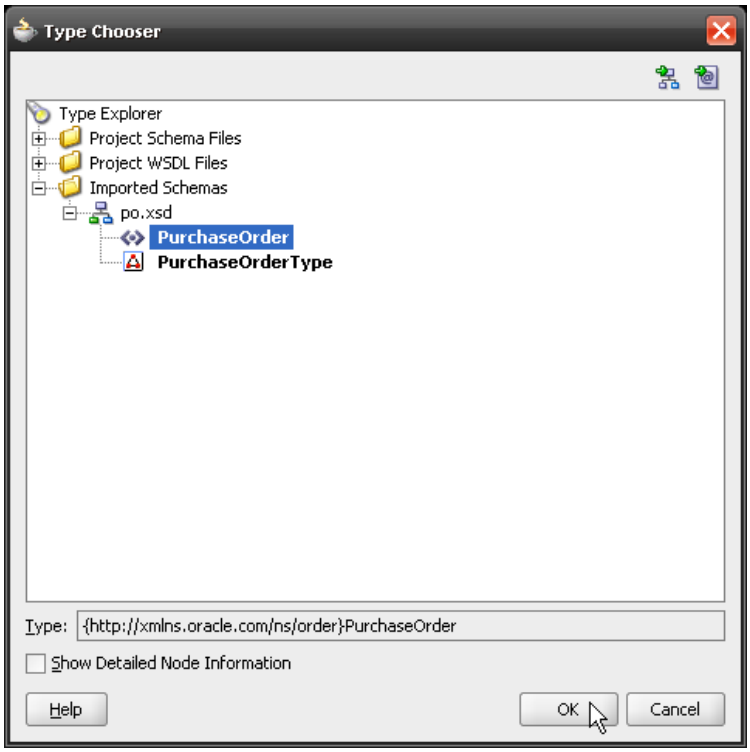
3. Browse for the `po.xsd` schema (which you'll find under `c:\po\schemas`).



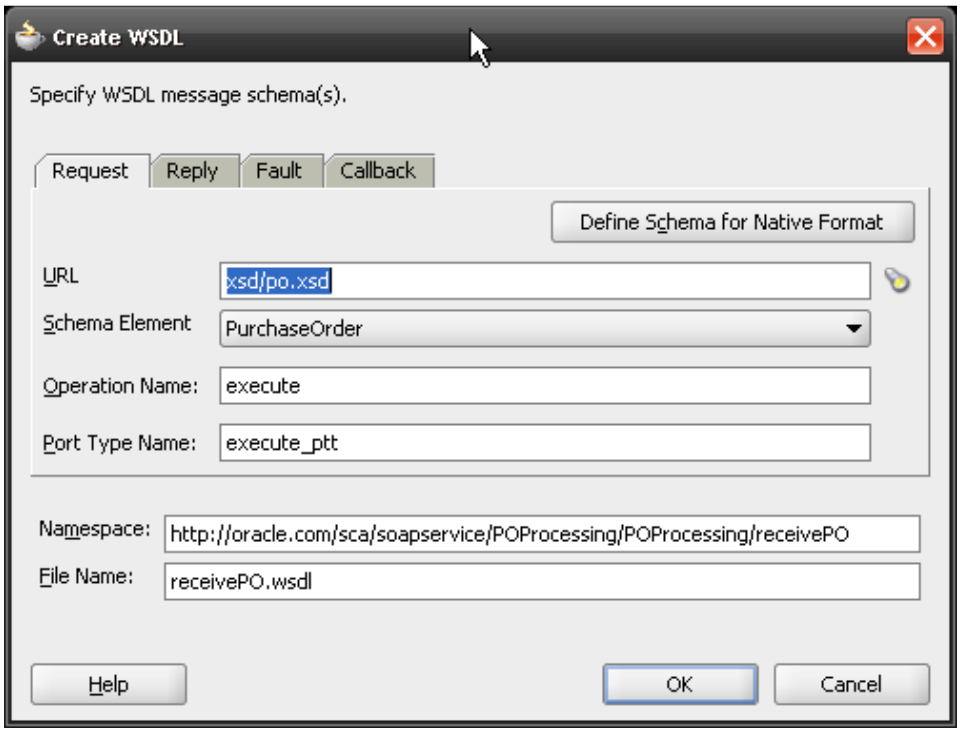




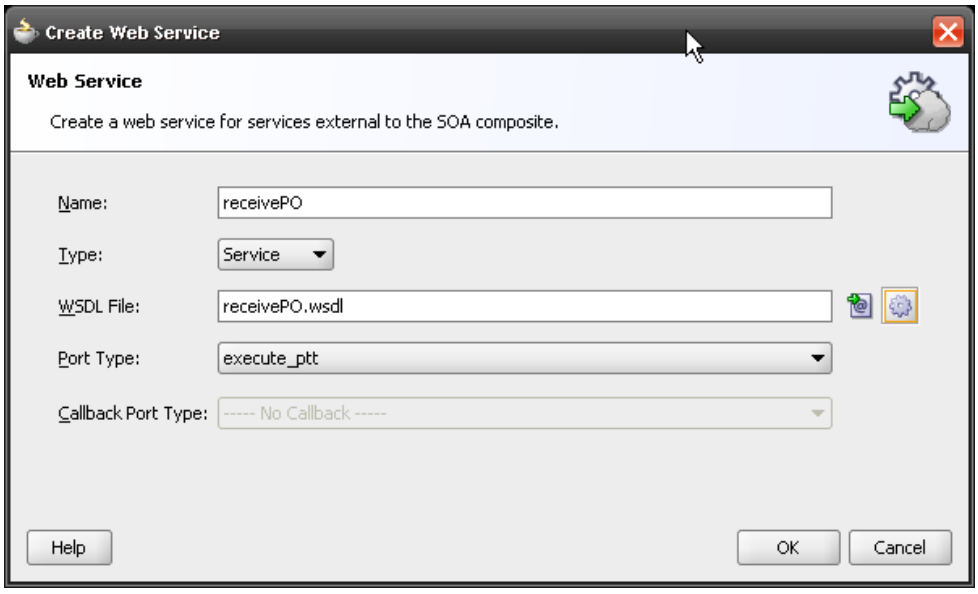
4. Select the **PurchaseOrder** element.



5. This is service is a one-way invocation type, also known as a fire-and-forget service. So there is no need to specify a reply or callback.



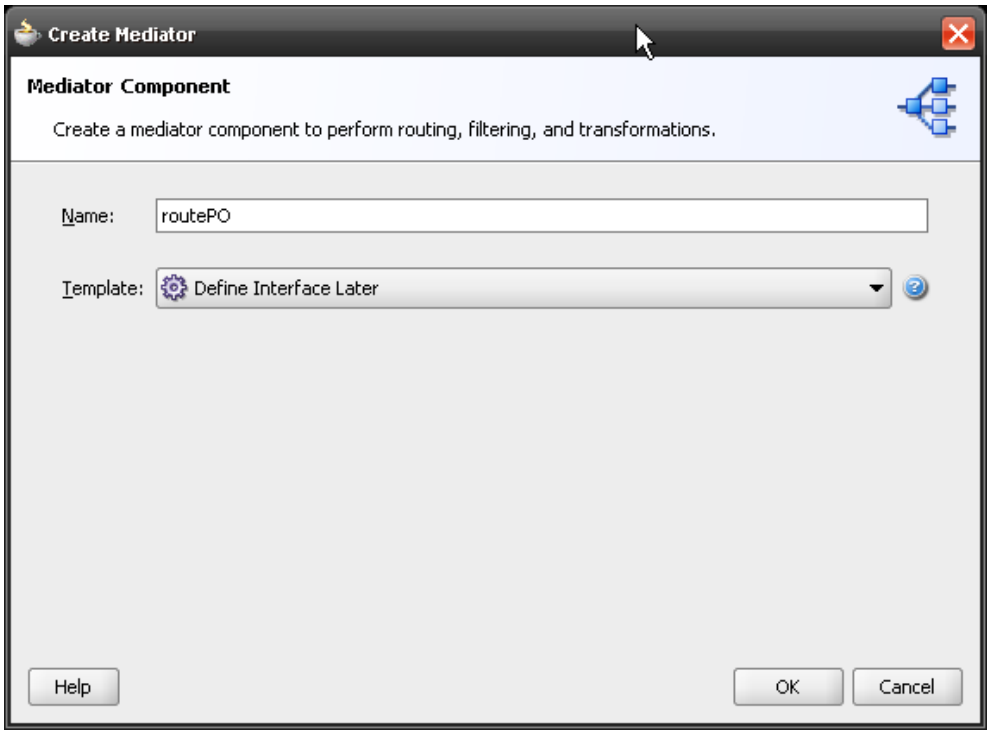
6. Click **OK**.



7. Click **OK** to close the web service binding dialog.

3.2.3 Adding the routing component

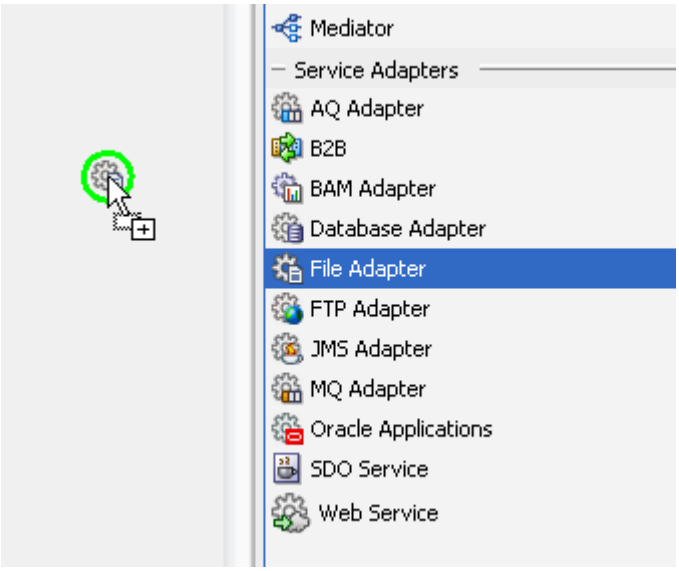
1. Now add a Mediator to the composite by dragging one from the palette to the canvas. Call it **routePO** and select the **Define Interface Later** template.



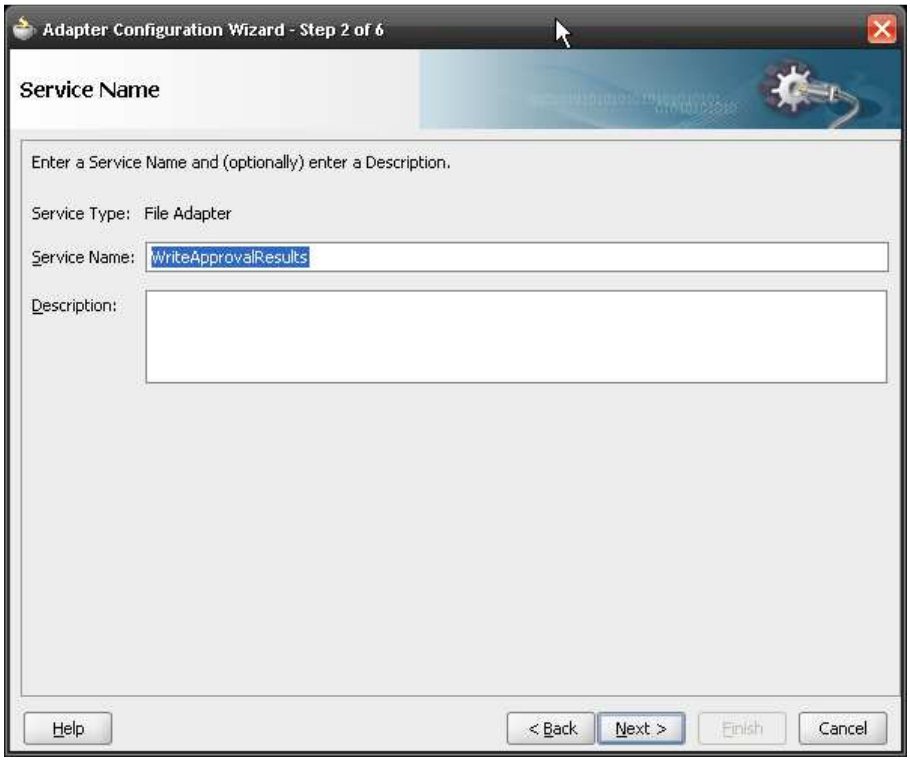
2. Click **OK**.

3.2.4 Adding the File Adapter

1. Drag-and-drop a **File Adapter** to the **References** swim lane. This file adapter will append each new order to a text file.



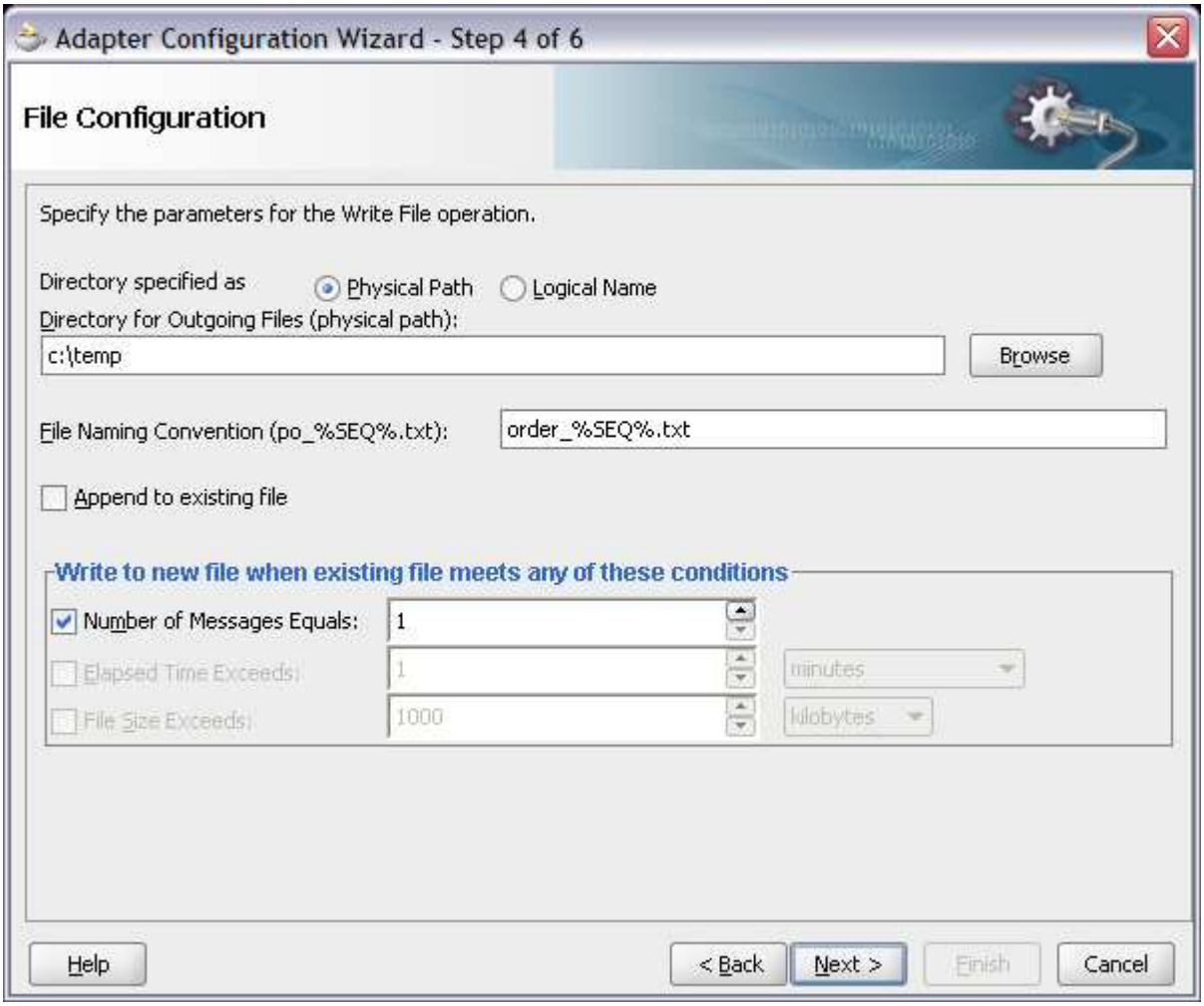
2. Name the service **WriteApprovalResults**.



3. Click **Next**.
4. Select the **Write File** operation.

5. Click **Next**.
6. Specify the following settings, leaving all others with their default values:
   - **Directory for Outgoing Files:** c:\temp

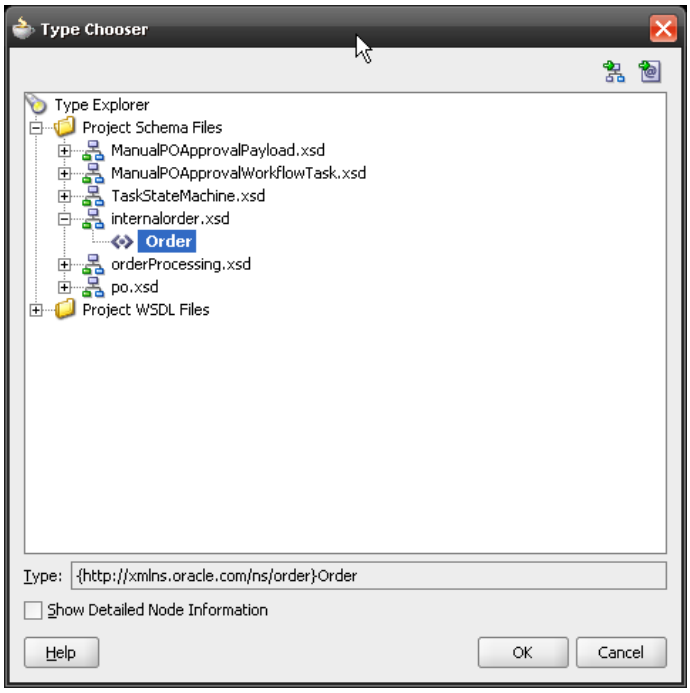
     Note: You may need to create this directory if it doesn't exist.
   - **File Naming Convention:** order_%SEQ%.txt

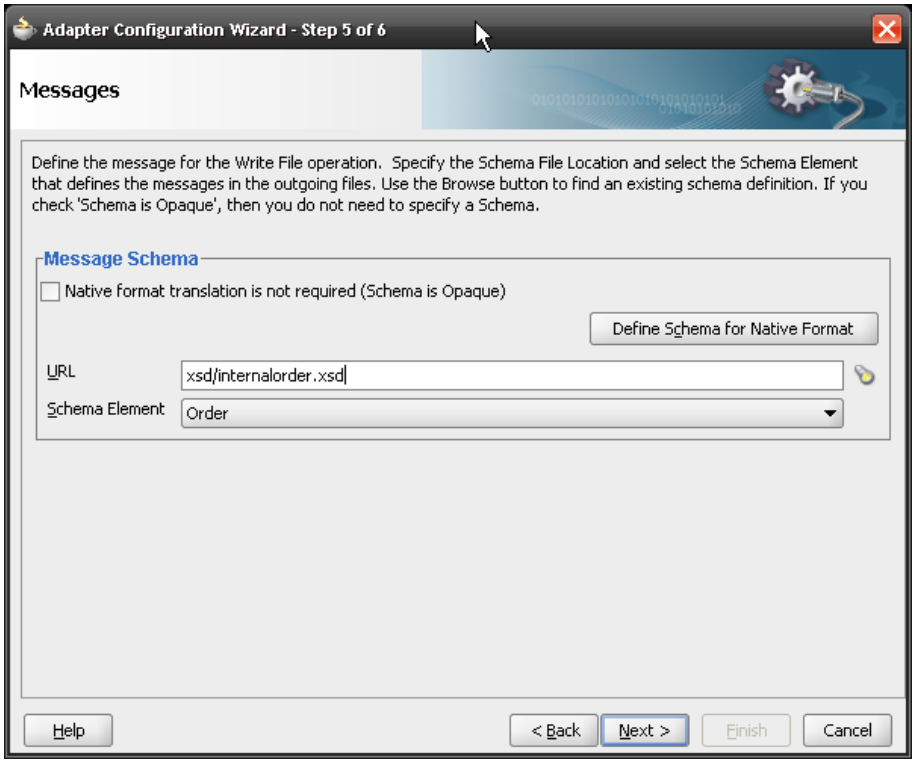


7. Click **Next**.

8. Browse for a schema that represents the content we will write.

9. Click the **Import Schema** File button and navigate to `c:\po\schemas\internalorder.xsd` to open the file, then select **Order**.

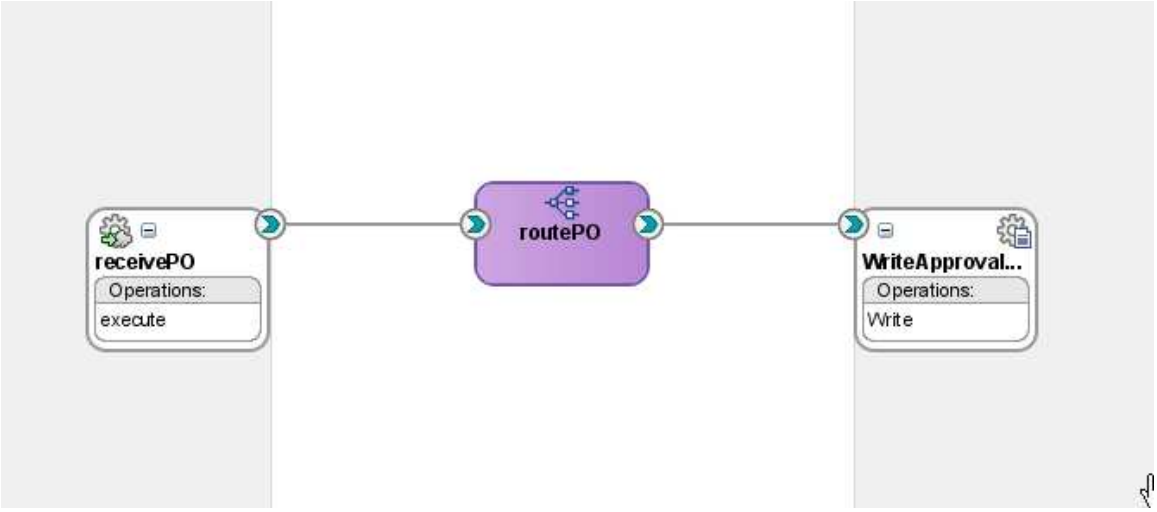




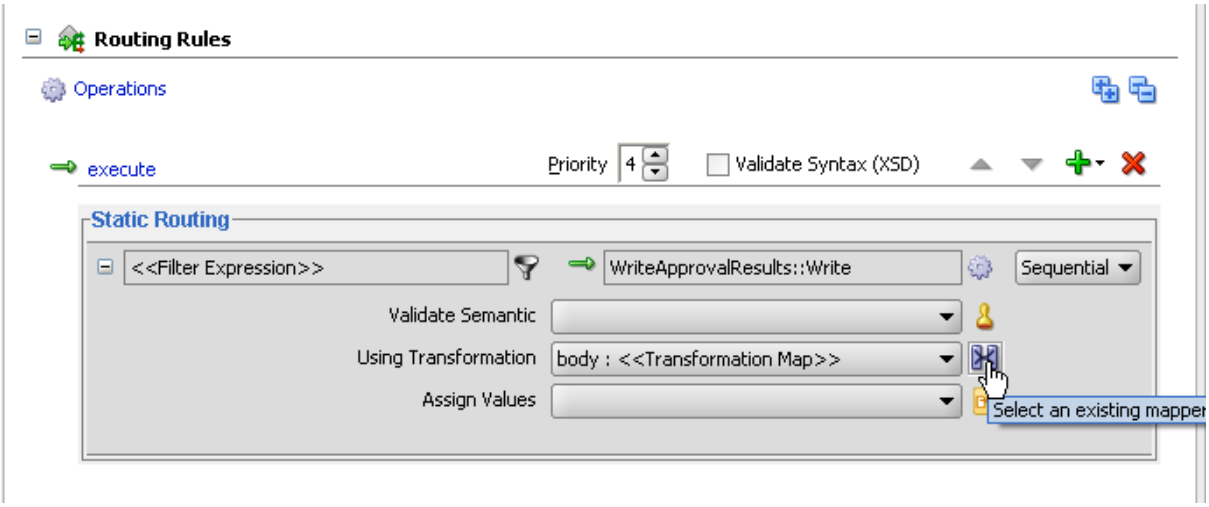
10. Click next, then complete the File Adapter wizard and return to the composite.

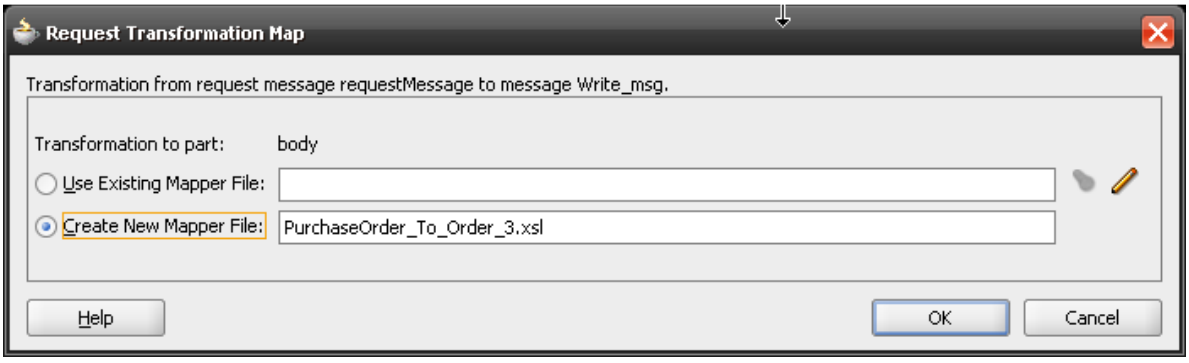
3.2.5 Wiring the components and adding a transformation

1. Wire the components as follows.

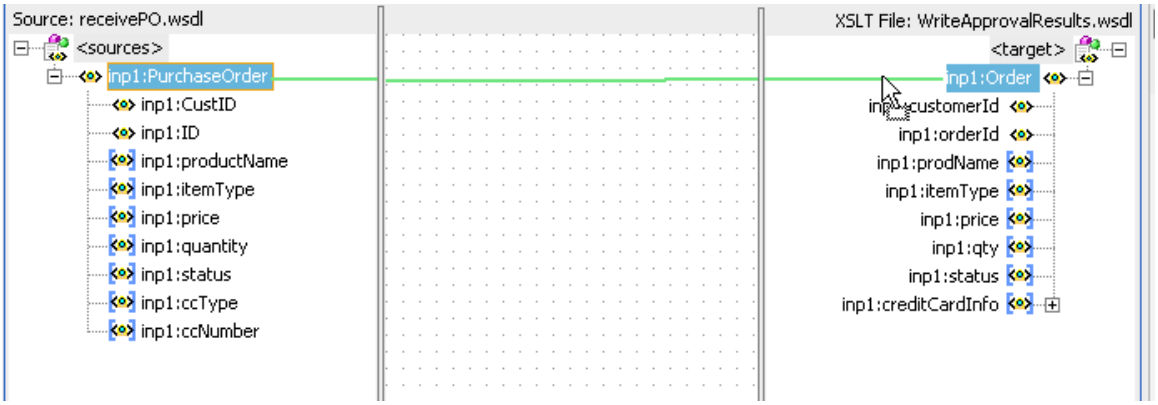


2. Double-click the Mediator component to create the mapping between the inbound PO and the internal order format that we will log to file.

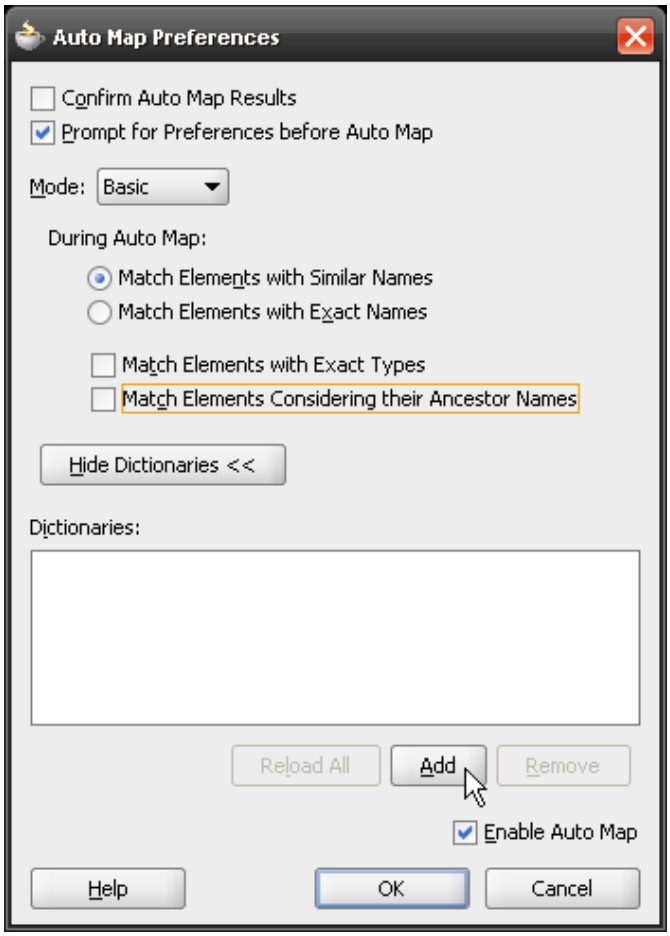




3. Drag-and-drop **Purchase Order** from the source side to **Order** on the target side.

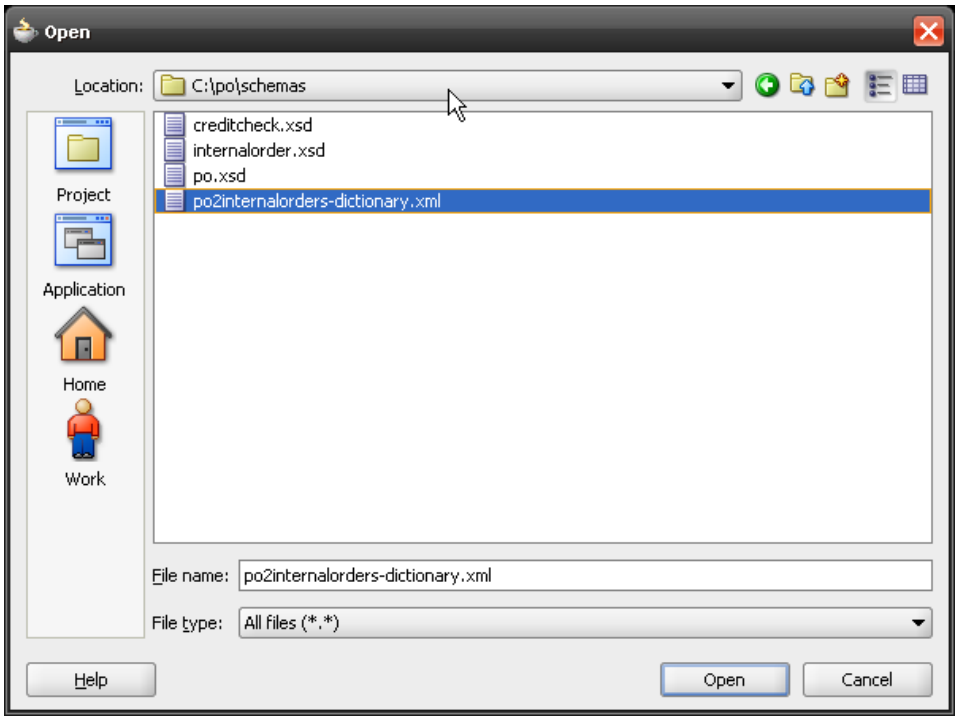


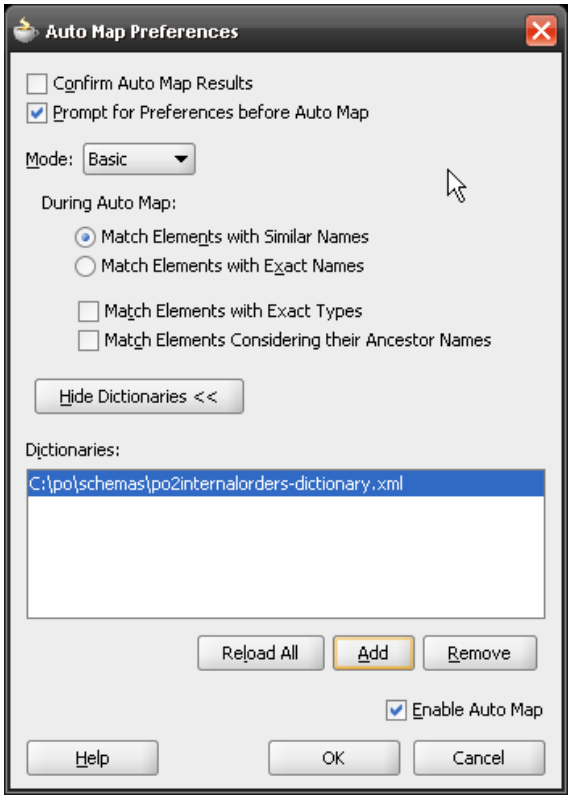
4. You will be prompted for auto-mapping preferences:



Uncheck **Match Elements Considering their Ancestor Names** and click on **Show Dictionaries**. Select **Add**. To help us in our mapping, we are going to leverage a dictionary created by our business analysts and that lists common synonyms in use across our data objects (ex: "qty" is sometimes used instead of "quantity", some departments use "ID" instead of "ordered", etc.).

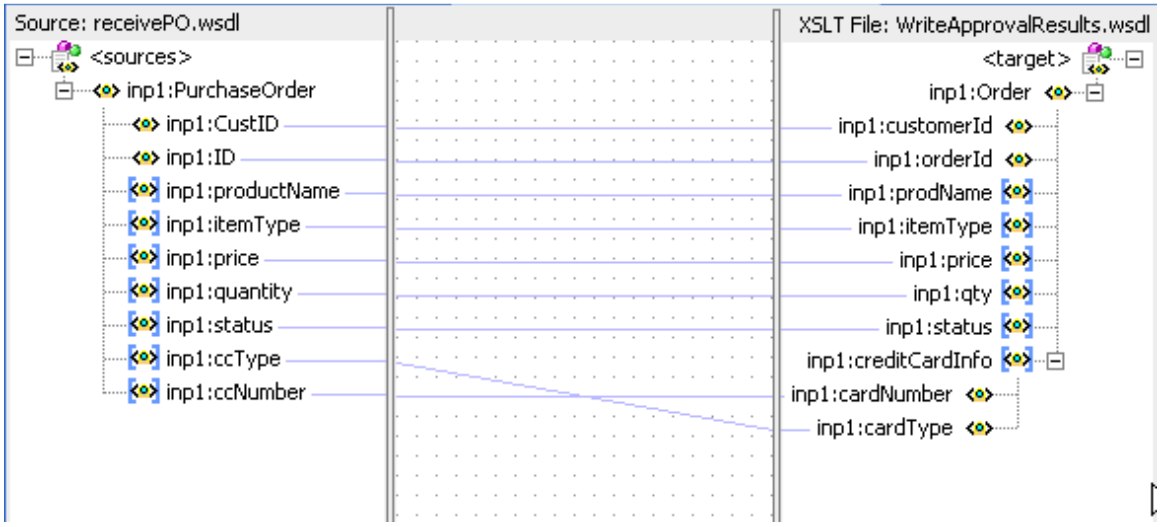
5. Browse for `c:\po\schemas\po2internalorders-dictionary.xml`.





6. Note that even without a dictionary, the auto-mapping feature will be able to identify and automatically take care of many of these fields, but a dictionary, customized to your company helps improve its accuracy.

7. The resultant mapping should look like this:



8. Save and close both the mapping and the Mediator editor to return to the composite.
9. At this point in time you have a fully-functional ESB flow that can be deployed and tested before adding more components to it.

   Select **Save-All** to save all of your edits.

3.3 Deploying the application

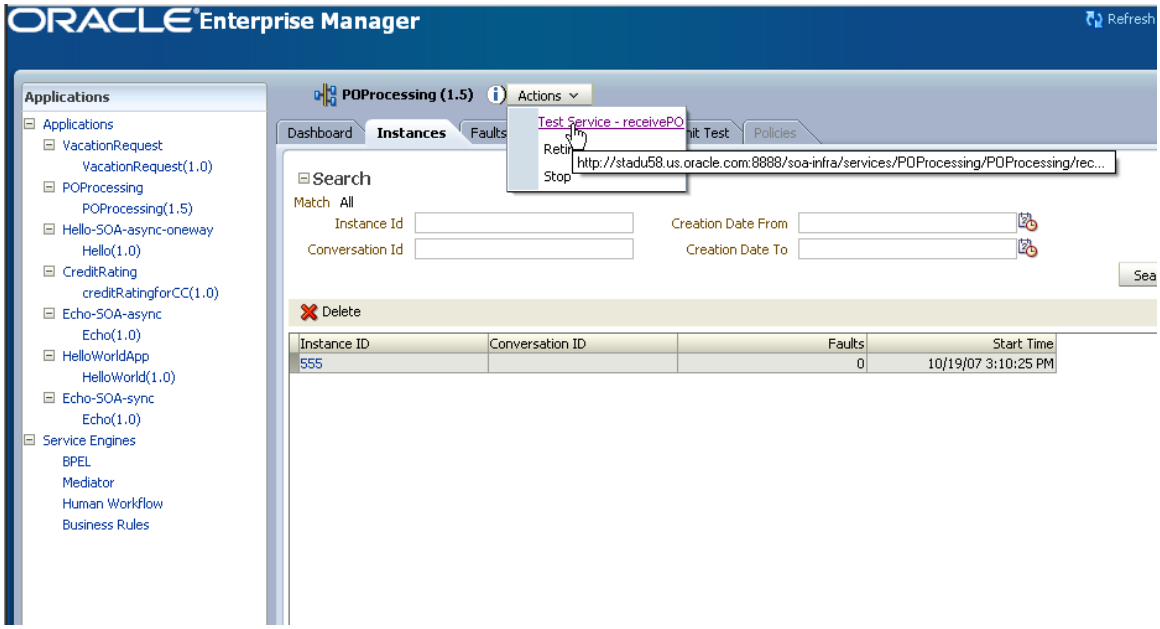
Deploy the application in the same way as before using the **Deploy** command on the Application Menu. This time you can select connection you created earlier instead of creating a new one.

Read **Appendix A Deploying and Running a Composite Application** to refresh your memory on how to deploy if you need to.

3.4 Testing the application

Read **Appendix A Deploying and Running a Composite Application** if you can't remember how to run and test your application.

1. Open the SOA Console.
2. Click on **POProcessing** and in the **Actions** list choose to test your service.

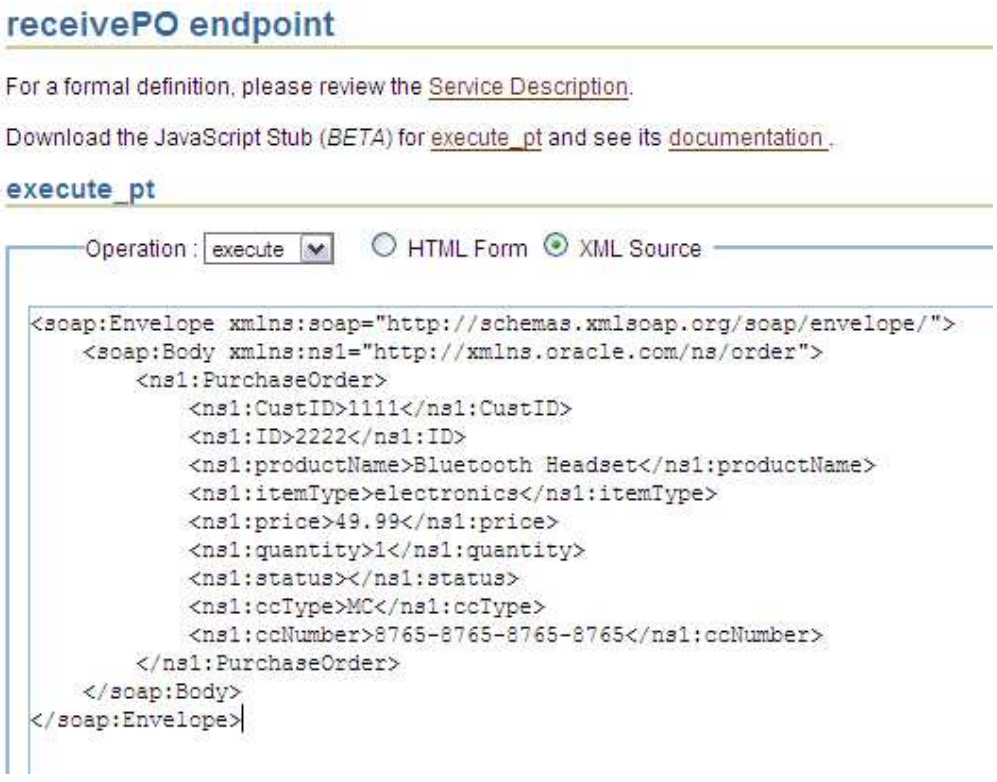


3. Enter a small order by doing one of the following:

   - Type the values into in to the HTML form.
   - Click **XML Source** so you can paste in the XML payload. This is the recommended way. Open the following file in a text editor:

     `c:\po\input\po-small-Headsetx1.xml`

     Copy the entire contents and paste them into the large text field in your browser:



4. Click **Invoke**.

5. The **Test Result** screen won't have any response because this is a one-way invocation with no reply or callback. However the call to the File Adapter service will have resulted in a new file in `c:\temp`, called `order_n.txt`, where *n* is a sequence number like 1, 2, 3, etc.

   You can open that file with a text editor and examine it (as shown in Figure 1). Notice how field names have been translated by the mapping and are different from the input.

Figure 1 Output from order_n.txt file

```
<?xml version="1.0" ?><inp1:Order xmlns:inp1="http://xmlns.oracle.com/ns/order">
   <inp1:customerId>1111</inp1:customerId>
   <inp1:orderId>2222</inp1:orderId>
   <inp1:prodName>Bluetooth Headset</inp1:prodName>
   <inp1:itemType>electronics</inp1:itemType>
   <inp1:price>49.99</inp1:price>
   <inp1:qty>1</inp1:qty>
   <inp1:status/>
   <inp1:creditCardInfo>
      <inp1:cardNumber>8765-8765-8765-8765</inp1:cardNumber>
      <inp1:cardType>MC</inp1:cardType>
   </inp1:creditCardInfo>
</inp1:Order>
```

6. You can also return to the SOA Console, click on the application or click the **Refresh** link. In the **Last 5 Instances** section click the most recent instance id to see the execution flow.