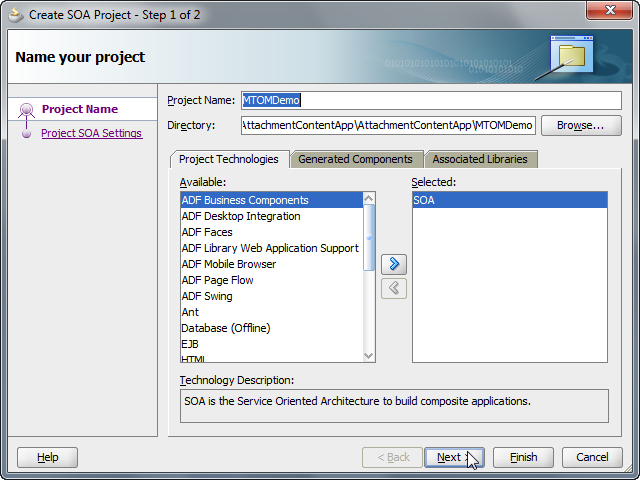
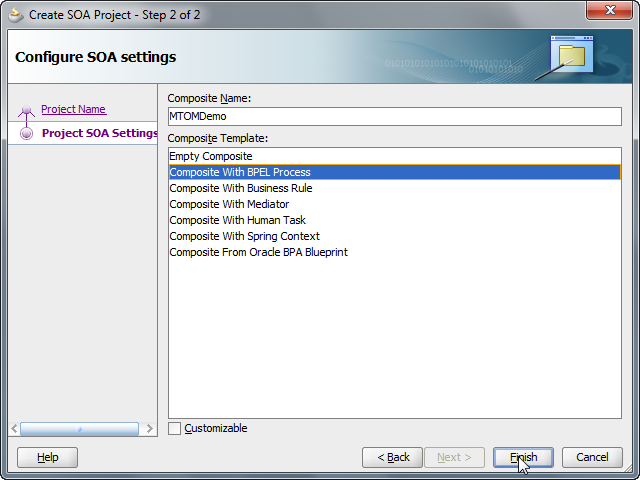
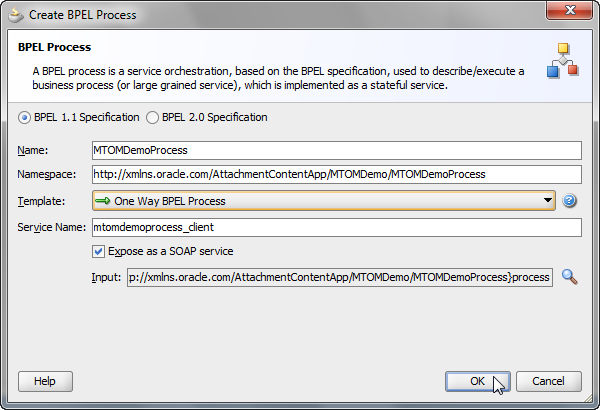
**Section 1: Create the Service that receives the MTOM Message**

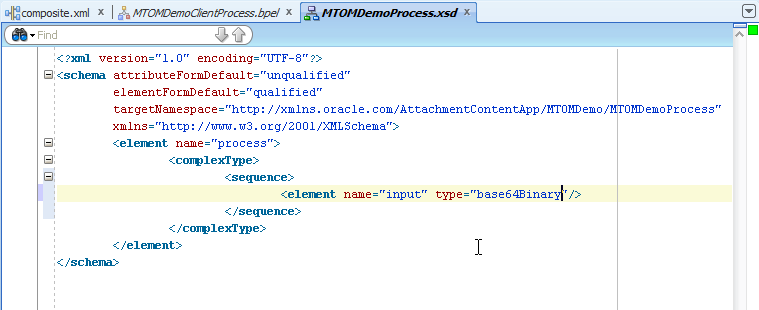
**Step 1:** Create a new JDeveloper SOA project and name it MTOMDemo. Create a new Composite with a One-Way BPEL Process as shown below.



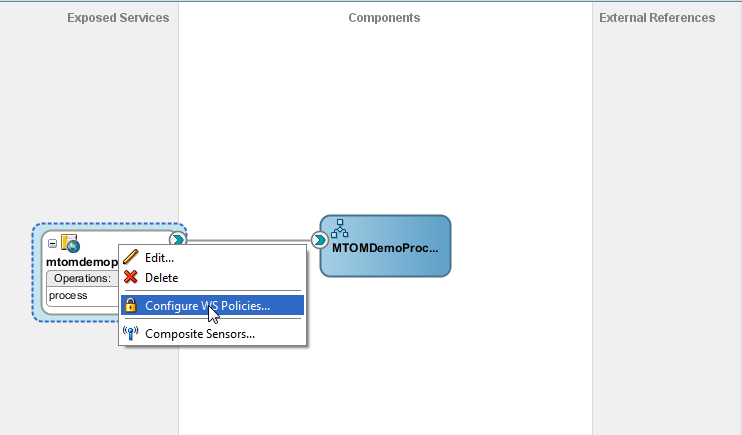


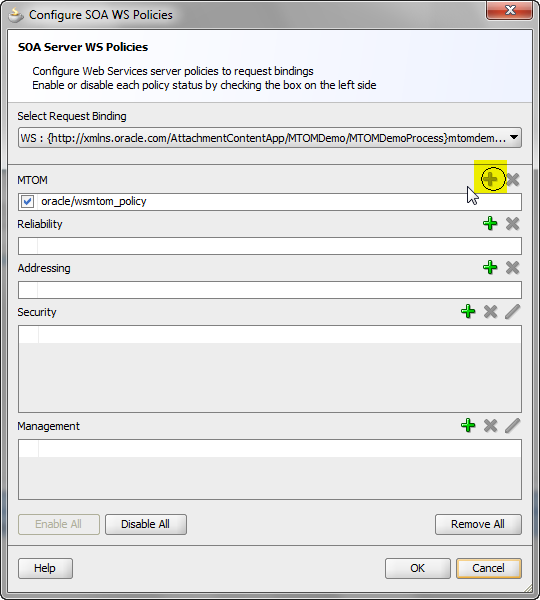


**Step 2:** Open up th generated XSD and update the datatype of the input element to base64Binary indicating that the field will actually hold the encoded binary content.

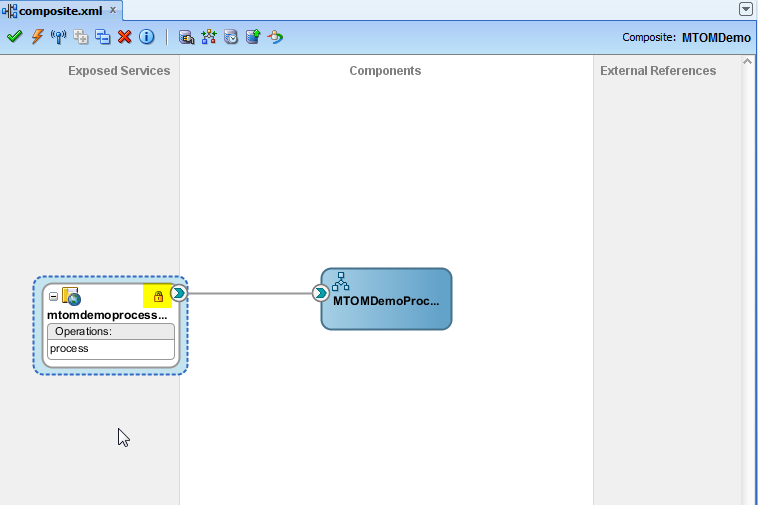


**Step 3:** Open the composite editor and right click on the Service interface and configure it to use oracle/wsmtom\_policy.

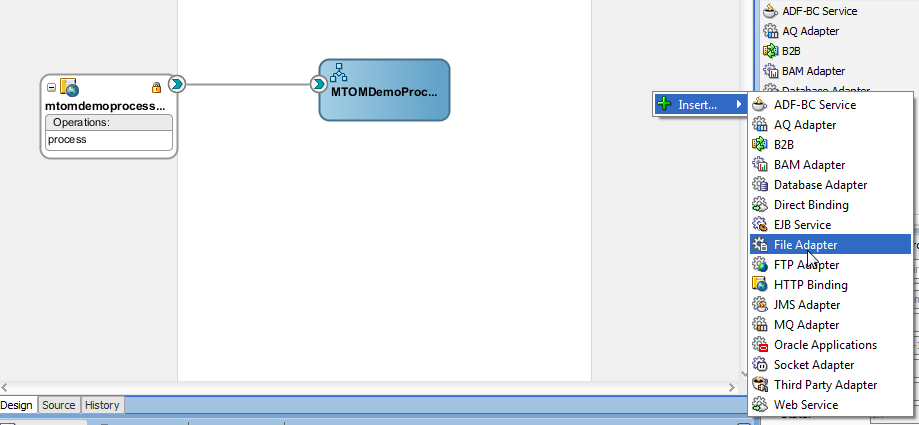


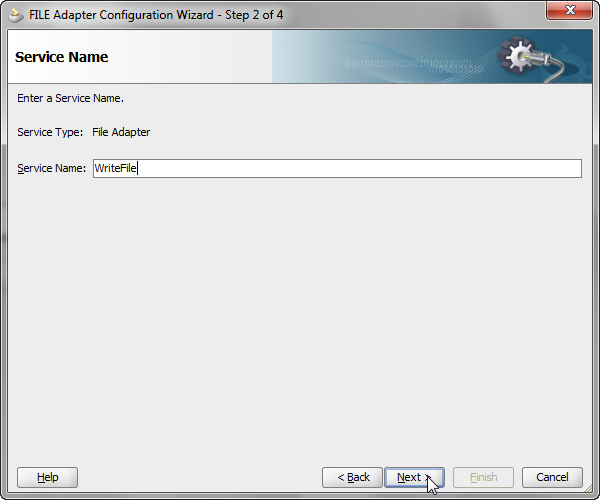


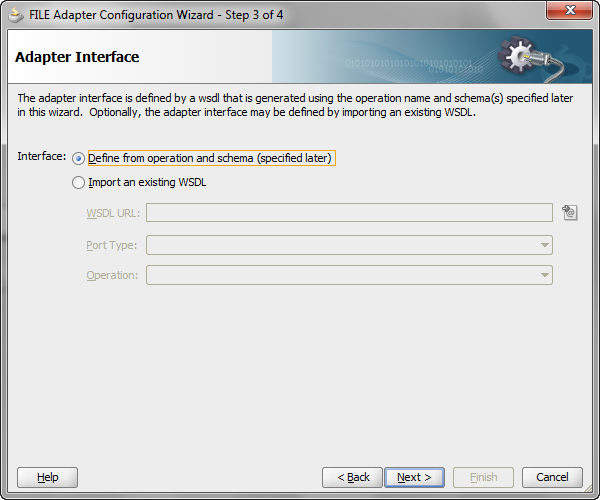
Step 4: Once you are done, you can see a *“lock”* icon on your service interface in the composite editor.

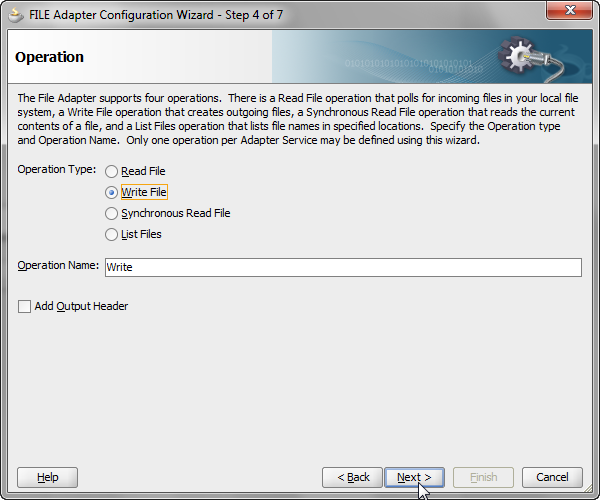


**Step 5:** Next, model an outbound File Write interaction by following the steps shown below.

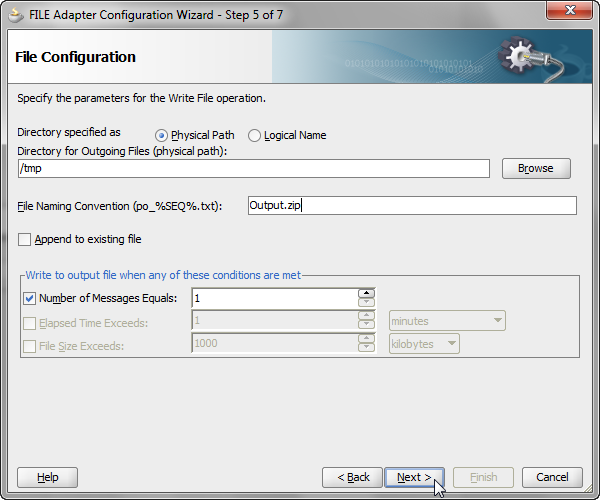


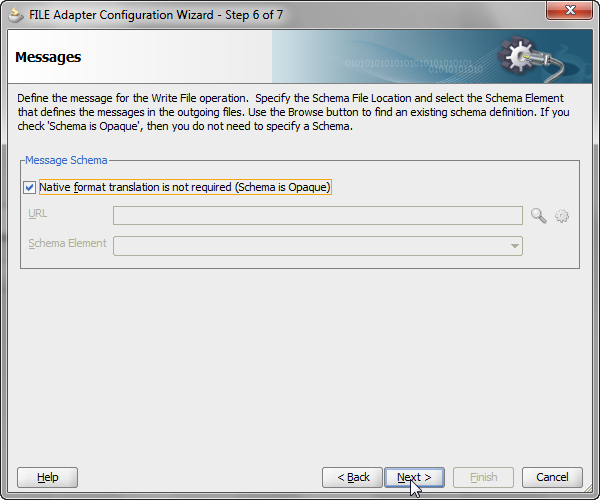




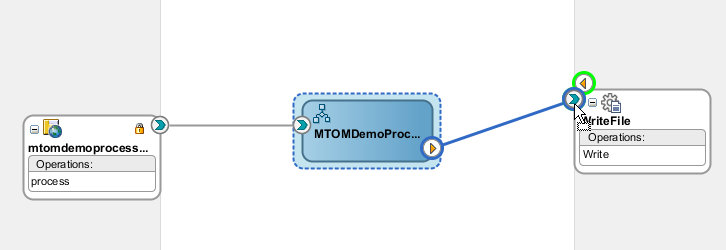


You can choose any directory of your choice for writing the file, and also any file name of your choice. Also choose an opaque datatype for the message definition in the dialog that follows.

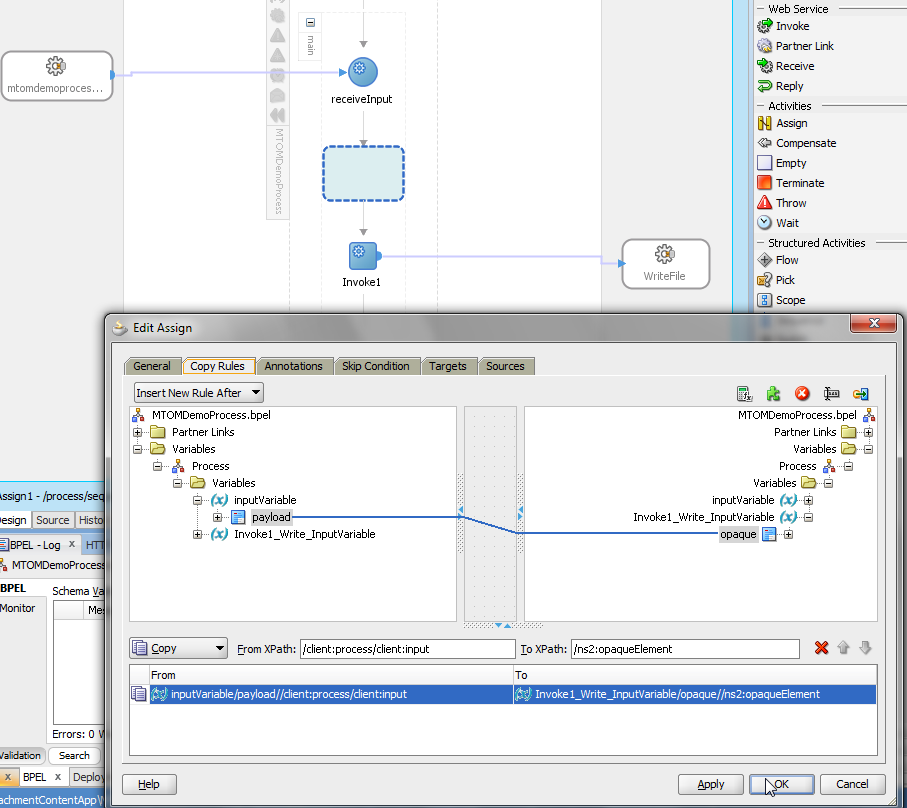


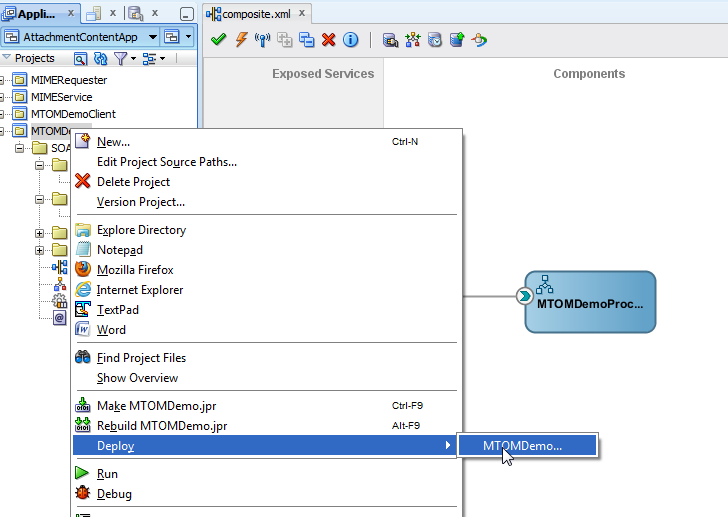


**Step 7:** Next wire the BPEL to the Adapter Service reference, and open up the BPEL process for editing.



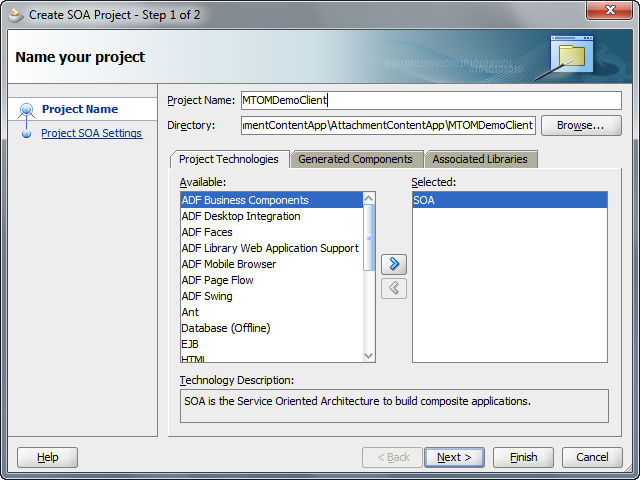
**Step 8:** Wire the <receive> and <invoke> activities appropriately with the default choices. Next, drag and drop an assign activity before the <invoke> and assign the *<input>* element in the source to *opaqueElement* in the target. Save all the files and deploy the composite .

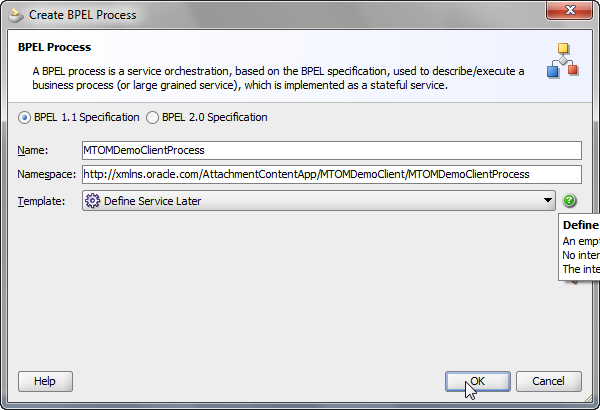




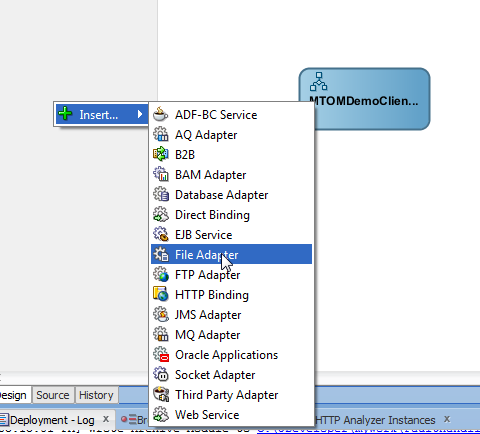
**Section 2: Create the Client Process that invokes the above service and passes the payload via MTOM.**

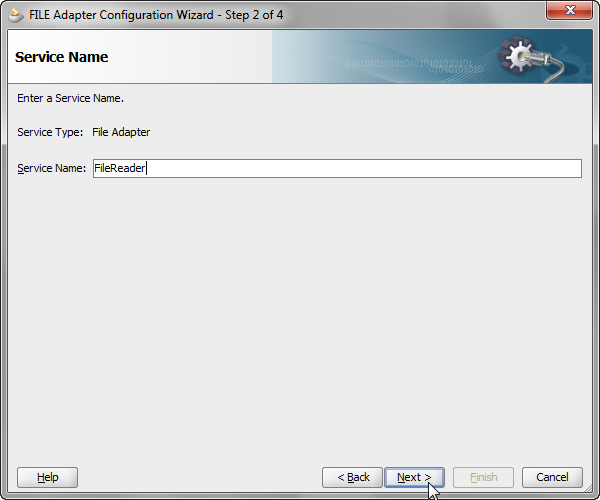
**Step 1:** Create a new JDeveloper SOA project and name it MTOMDemo. Create a new Composite with a BPEL Process as shown below. Don’t choose an interface WSDL yet – we will create it via an inbound File Adapter.

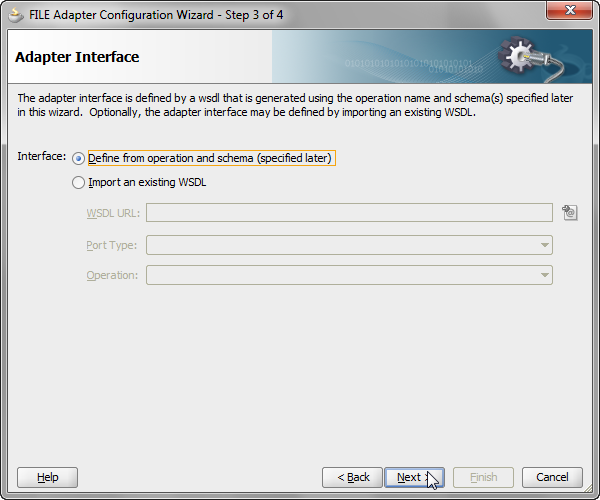


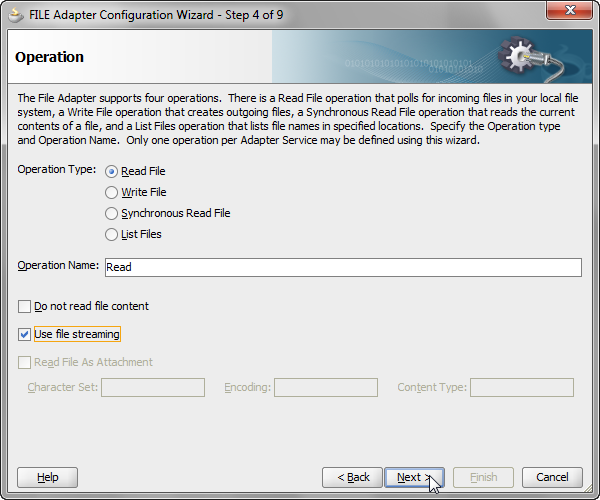


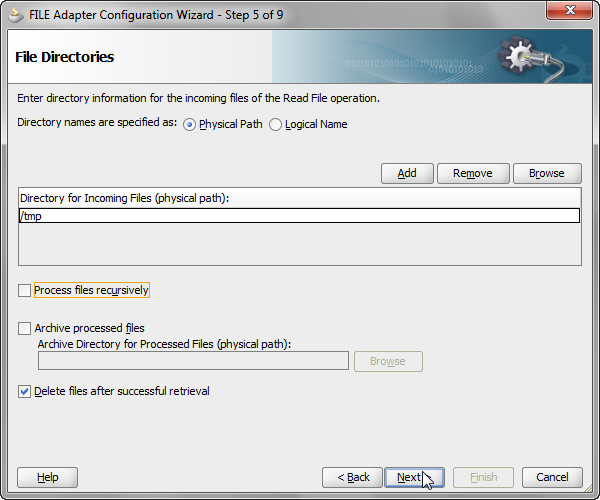
Step 2: Model an inbound File Read interaction as shown below. Choose opaque content , and a directory and file naming pattern of your choice.

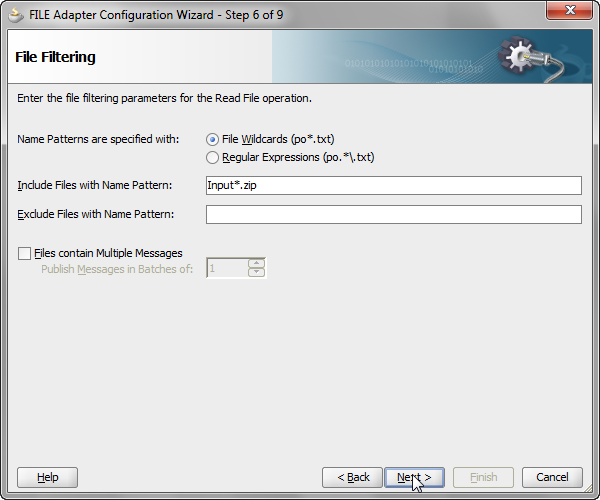


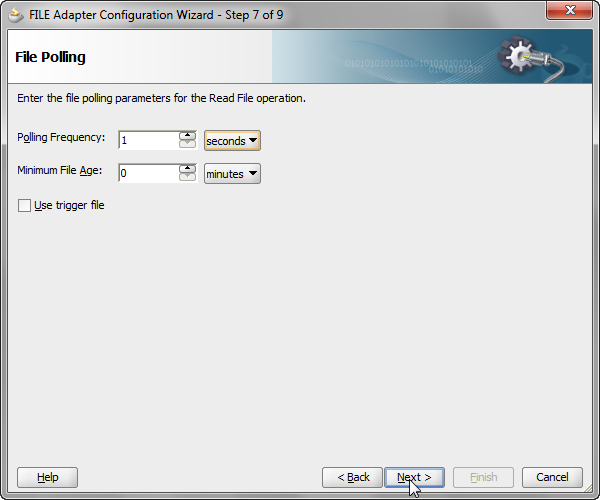


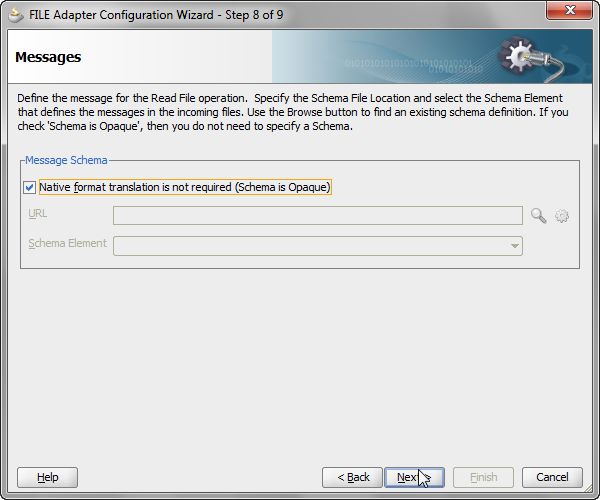




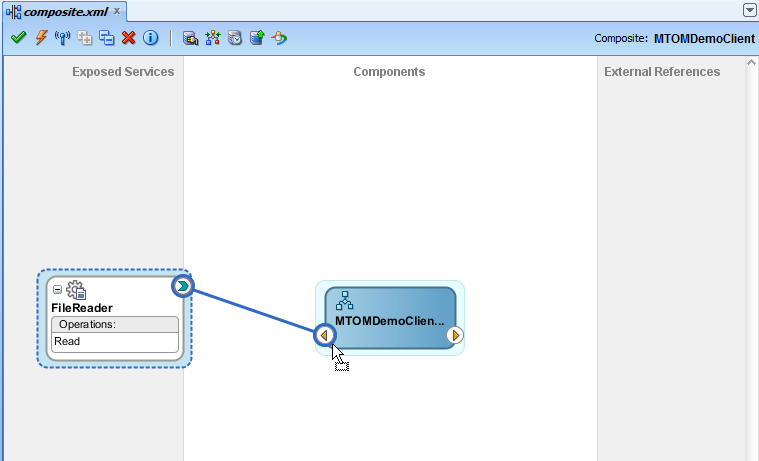




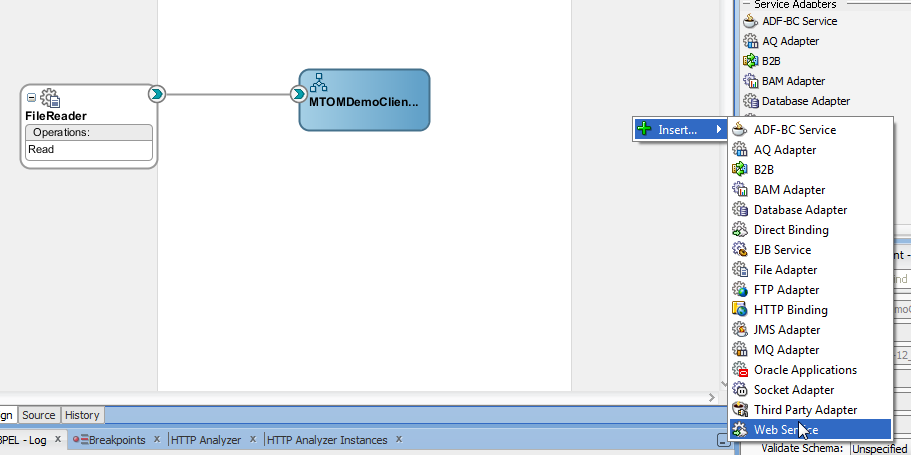


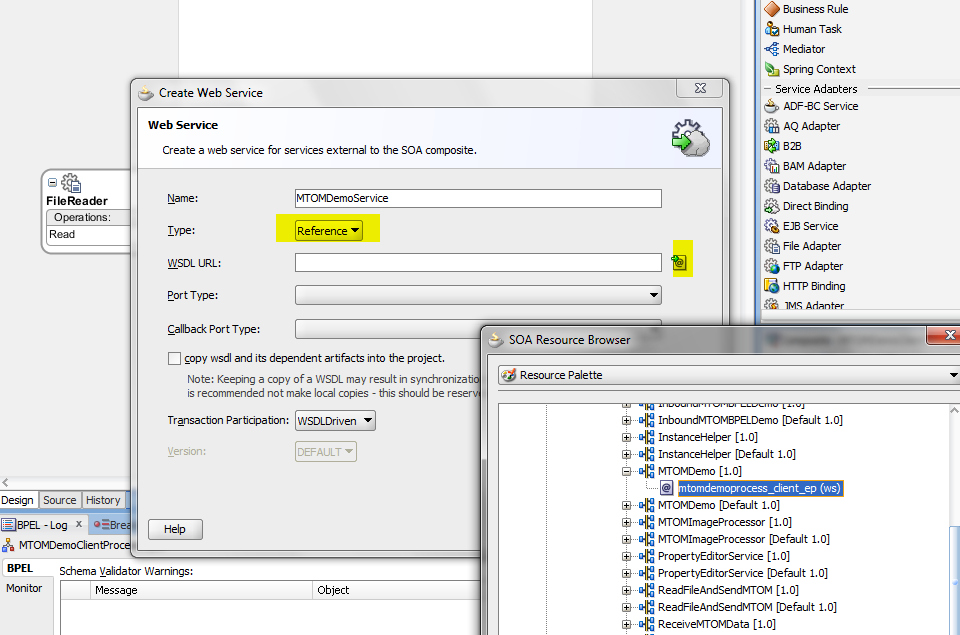


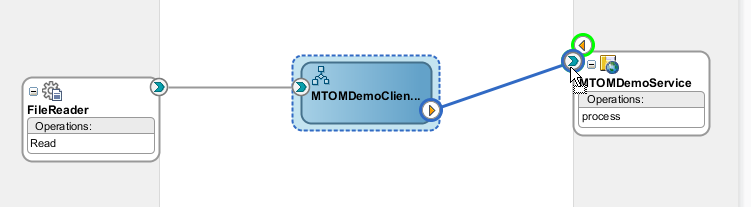
Step 3: Wire the inbound service interface to the BPEL.



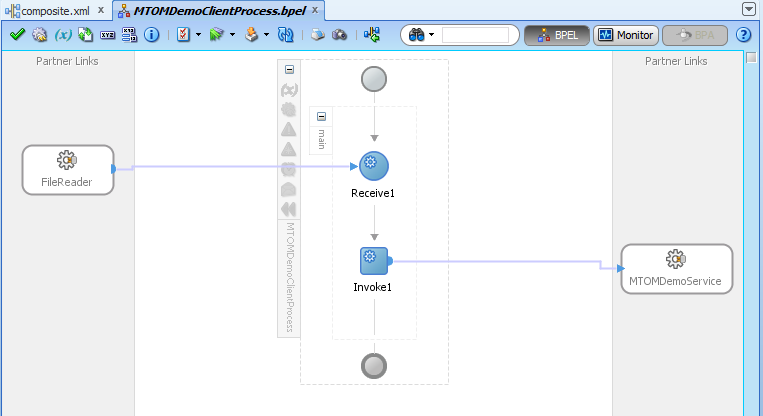
Step 4: Create a Service reference and point it to the endpoint of the MTOMDemo Service created earlier.

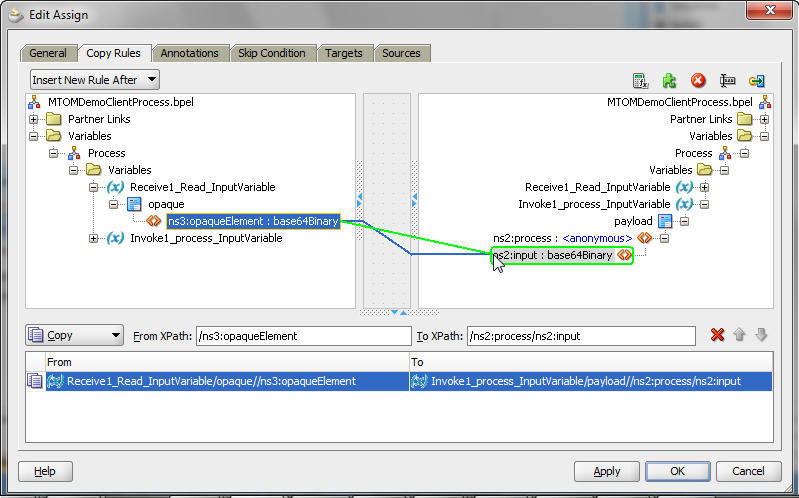




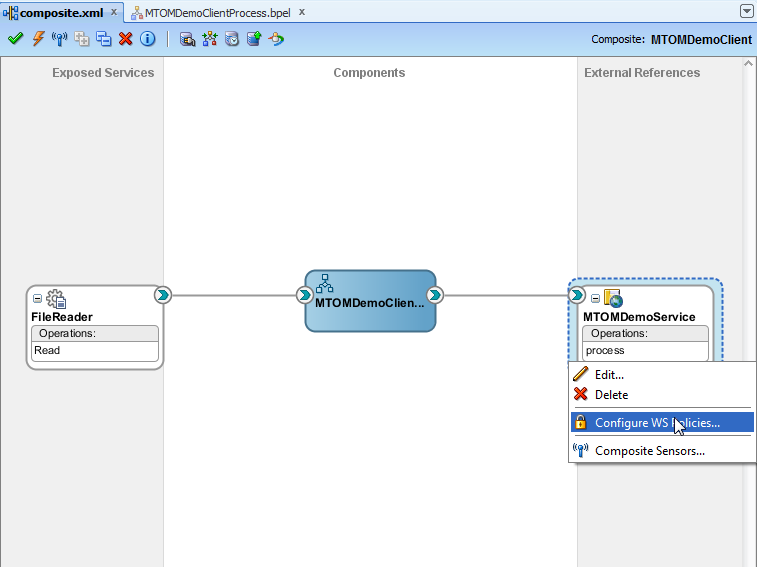


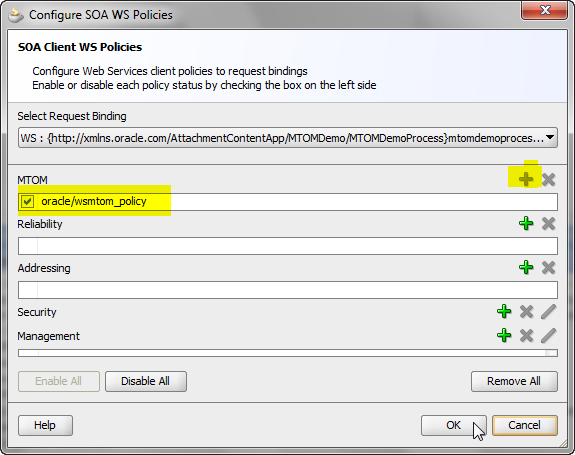
**Step 5:** Wire the BPEL Process <receive> and <invoke> and then drag and drop an assign activity in between.

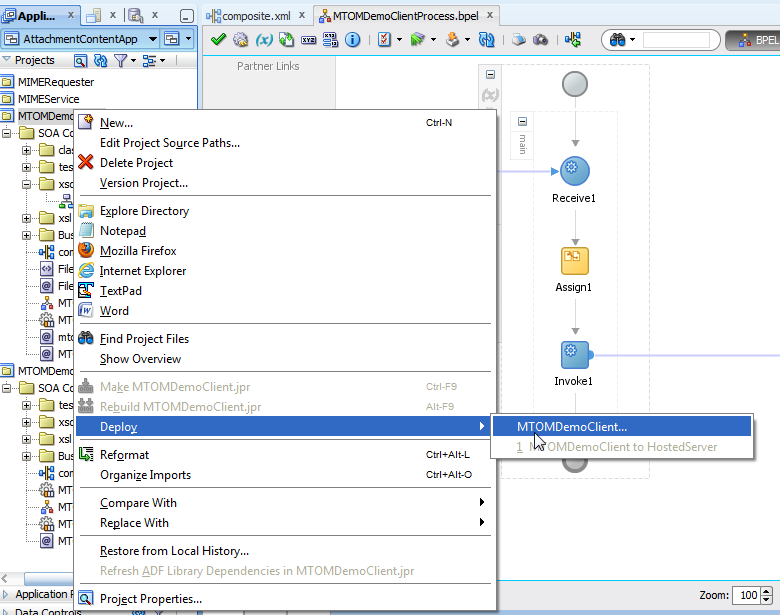




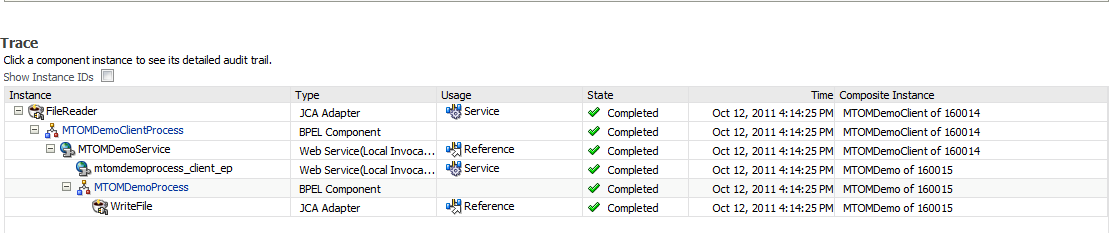
**Step 6**: Configure the Service Reference to use the MTOM policy to ensure that MTOM is active. Following this, deploy the composite project.







**Step 7:** Drop a file into the polling directory so as to trigger the processing. You should be able to see the composite execution on the console.



**Step 8:** If you tunnel and see the message you can notice two interesting behavior.

If the size of the binary attachment is less than 1024 bytes, the binary data is base64 encoded and inline within the SOAP envelope. If not, it appears within a separate MIME part referred via xop:include.

