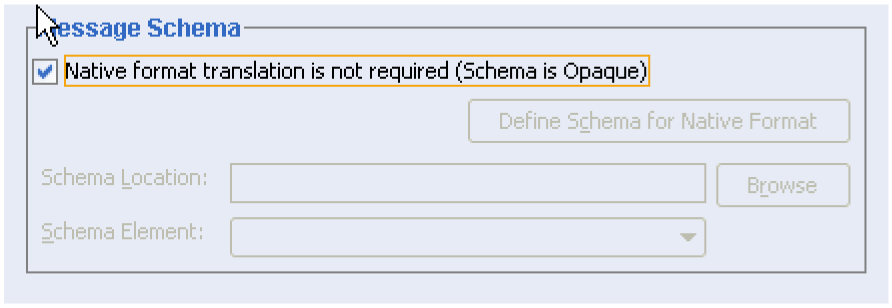
File and FTP Adapter Advanced Features

Getting File Header without File Payload

If a process requires only the file information (File Name and Directory), implement the following steps

1. Build a Partnerlink with read operation using the Adapter Wizard. On the “Message Schema” form of the Adapter wizard, select the checkbox for “Native format translation is not required (Schema is Opaque)”.



[✓] Native format translation is not required (Schema is Opaque)

This would create a WSDL similar to the one below.

```
     <binding name="Read_binding" type="tns:Read_ptt">
     <pc:inbound_binding  />
         <operation name="Read">
       <jca:operation
           LogicalDirectory="InputDir"
           ActivationSpec="oracle.tip.adapter.file.inbound.FileActivationSpec"
           DeleteFile="false"
           IncludeFiles=".*\.txt"
           ExcludeFiles="dummy"
           PollingFrequency="60"
           MinimumAge="0"
           OpaqueSchema="true">
       </jca:operation>
       <input>
         <jca:header message="hdr:InboundHeader_msg" part="inboundHeader"/>
       </input>
         </operation>
     </binding>
```

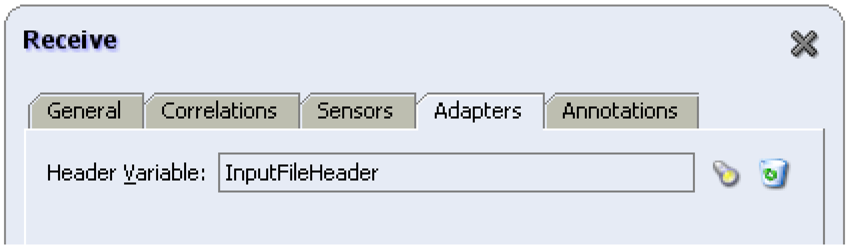
2. Manually add the UseHeaders property to the JCA operation.

```
     <binding name="Read_binding" type="tns:Read_ptt">
     <pc:inbound_binding  />
         <operation name="Read">
       <jca:operation
           LogicalDirectory="InputDir"
           ActivationSpec="oracle.tip.adapter.file.inbound.FileActivationSpec"
           DeleteFile="false"
```

```
          IncludeFiles=".*\.txt"
          ExcludeFiles="dummy"
          PollingFrequency="60"
          MinimumAge="0"
          OpaqueSchema="true"
          UseHeaders="true">
      </jca:operation>
      <input>
        <jca:header message="hdr:InboundHeader_msg" part="inboundHeader"/>
      </input>
        </operation>
    </binding>
```

3. Create a variable using the fileAdapterInboundHeader.wsdl.

4. Specify the variable in the Adapter tab of the partner link.



Header Variable: InputFileHeader

When the process is launched, the File / FTP Adapter will pass the File Name and Directory information in the header variable.

Transfer Large file from one location to another

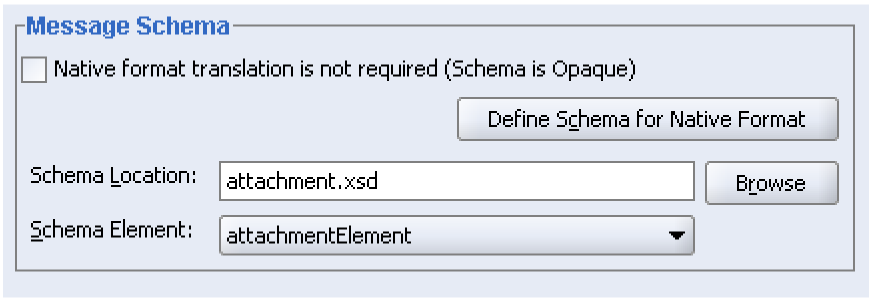
The attachment feature of 10.1.3.3 can be used transfer files larger than 7MB from source to destination without any translation or transformation. To use this feature implement the following steps

1. Create the following XSD and add/import the Schema into the project.

```
<schema xmlns=http://www.w3.org/2001/XMLSchema
targetNamespace=http://xmlns.oracle.com/pcbpel/adapter/file/attachment
/ elementFormDefault="qualified">
  <element name="attachmentElement">
    <complexType>
      <attribute name="href" type="string"/>
    </complexType>
  </element>
</schema>
```

☞ The attached XSD can be used with the File/FTP Adapter. attachment.xsd

2. Build File/FTP partnerlink with read operation using the Adapter Wizard. On the "Message Schema" form select the attachment.xsd as the schema and select the attachmentElment as the schema element.



☐ Native format translation is not required (Schema is Opaque)

Schema Location: attachment.xsd

Schema Element: attachmentElement

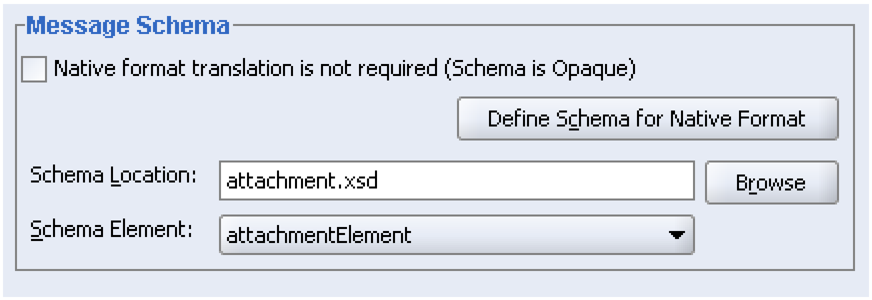
This would create a WSDL similar to the one below.

```
     <pc:inbound_binding  />
         <operation name="Read">
       <jca:operation
           LogicalDirectory="InputDir"
           ActivationSpec="oracle.tip.adapter.file.inbound.FileActivationSpec"
           DeleteFile="true"
           IncludeFiles=".*\..*"
           ExcludeFiles="dummy"
           PollingFrequency="60"
           MinimumAge="0"
           OpaqueSchema="false">
       </jca:operation>
```

3. Manually add the AsAttachment property to the JCA operation.

```
     <pc:inbound_binding  />
         <operation name="Read">
       <jca:operation
           LogicalDirectory="InputDir"
           ActivationSpec="oracle.tip.adapter.file.inbound.FileActivationSpec"
           DeleteFile="true"
           IncludeFiles=".*\..*"
           ExcludeFiles="dummy"
           PollingFrequency="60"
           MinimumAge="0"
           OpaqueSchema="false"
           AsAttachment="true" >
       </jca:operation>
```

4. Build File/FTP partnerlink with write operation using the Adapter Wizard. On the "Message Schema" form select the attachment.xsd as the schema and select the attachmentElment as the schema element.


Message Schema

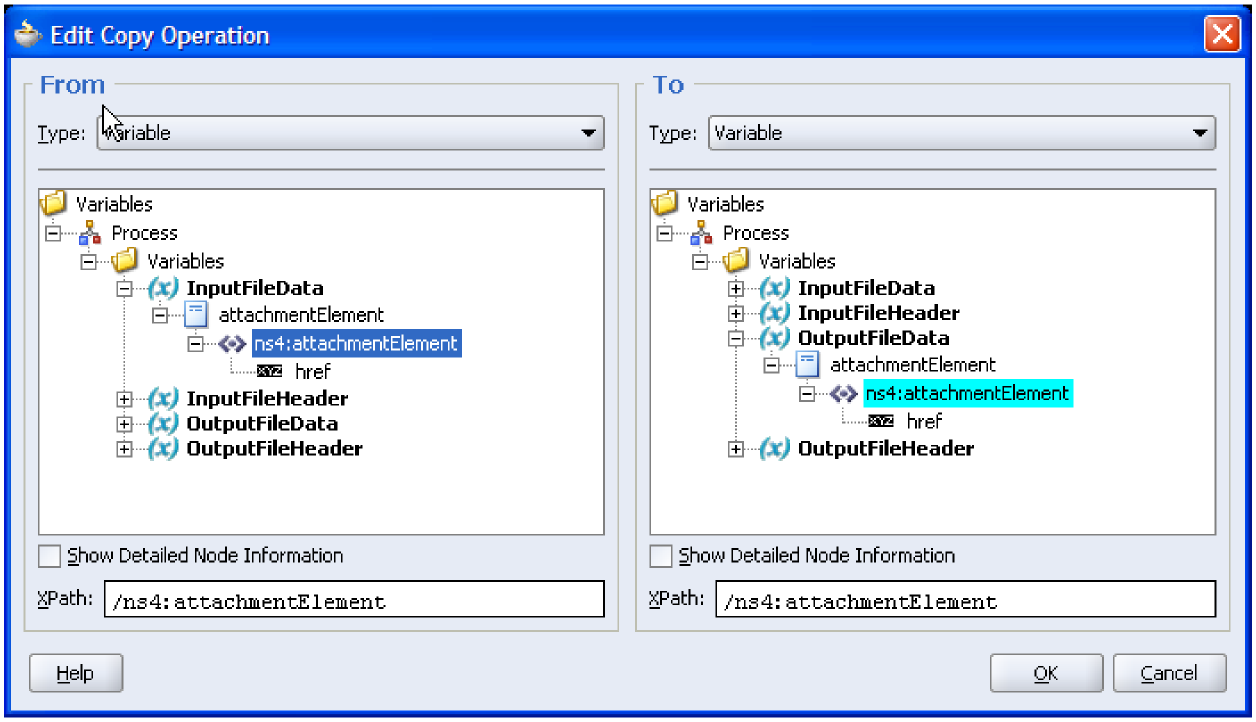
[ ] Native format translation is not required (Schema is Opaque)

Define Schema for Native Format

Schema Location: attachment.xsd

Schema Element: attachmentElement

5. Before the invoke activity for the write partnerlink, copy the href element from the input variable to the output variable.


Edit Copy Operation

From

Type: Variable

```
Variables
  Process
    Variables
      InputFileData
        attachmentElement
          ns4:attachmentElement
            href
      InputFileHeader
      OutputFileData
      OutputFileHeader
```

[ ] Show Detailed Node Information

XPath: /ns4:attachmentElement

To

Type: Variable

```
Variables
  Process
    Variables
      InputFileData
      InputFileHeader
      OutputFileData
        attachmentElement
          ns4:attachmentElement
            href
      OutputFileHeader
```

[ ] Show Detailed Node Information

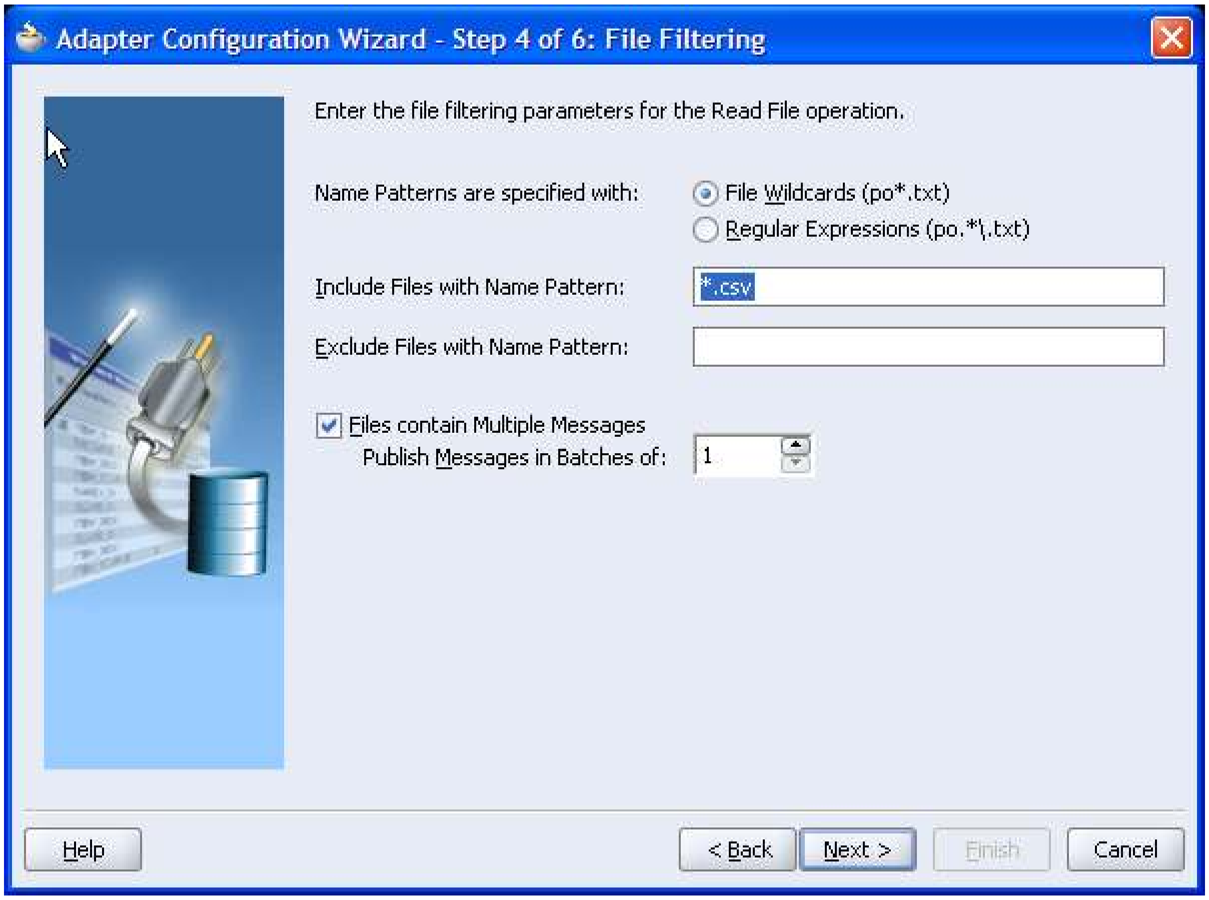
XPath: /ns4:attachmentElement

☞ The attachment capability is supported for Read operation only.

Processing Large File Using Debatching

For Files larger than 7MB, the debatching capability of File/FTP Adapter should be used, if the lines (or group of lines) can be processes independently. To debatch the files, implement the following

1. Build File/FTP partnerlink with read operation using the Adapter Wizard. On the “File Filtering” form select the “File contain Multiple Messages” checkbox and specify the number of lines (or groups of lines).


Adapter Configuration Wizard - Step 4 of 6: File Filtering

Enter the file filtering parameters for the Read File operation.

Name Patterns are specified with: [✓ File Wildcards (po*.txt)] Regular Expressions (po.*\.txt)

Include Files with Name Pattern: *.csv

Exclude Files with Name Pattern:

[✓] Files contain Multiple Messages

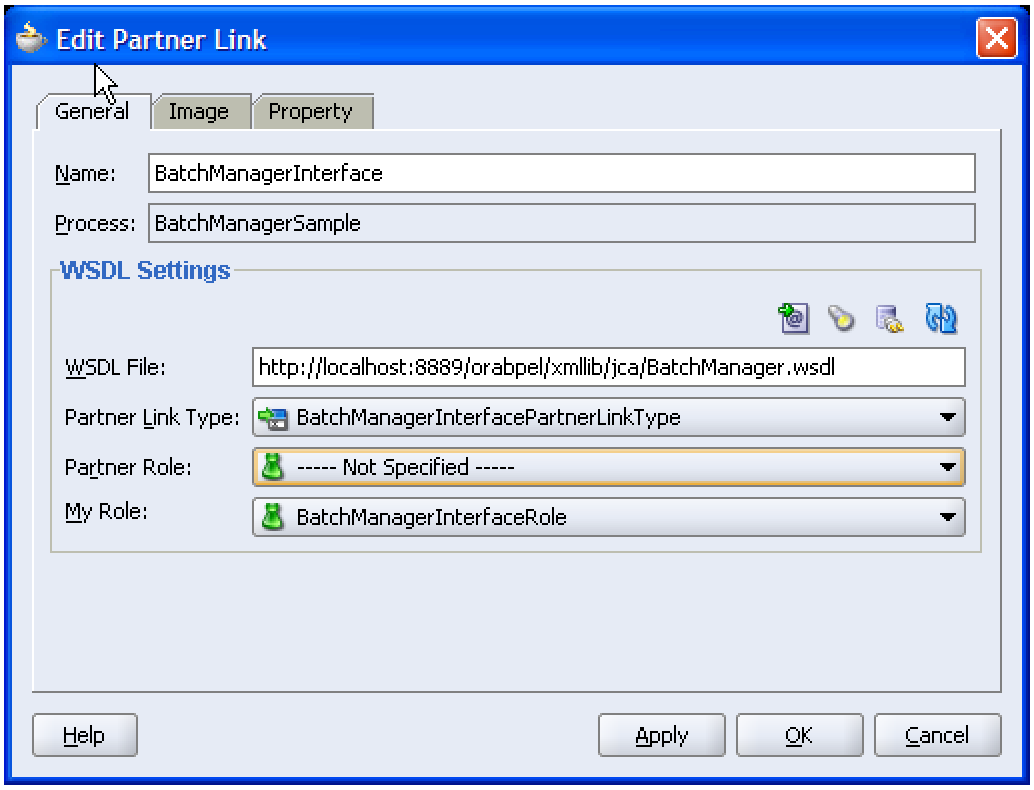
Publish Messages in Batches of: 1

☞ When a file is debatched, all BPEL processes are executed independently. The File/FTP Adapter can be configured to send notification to a controller process using the Batch Notification Manager. The next section provides the details of the Batch Notification Manager.

Debatching with a Batch Notification Manager

As mentioned above, the File/FTP Adapter can be configured to send notification to a controller process using the Batch Notification Manager. To implement the Batch Manager Process perform the following

1. Create a BPEL process using the "Empty BPEL Process" template and add a partnerlink using the WSDL document http://<bpel_host>:<bpel_port>/orabpel/xmllib/jca/BatchManager.wsdl


Edit Partner Link

Name: BatchManagerInterface

Process: BatchManagerSample

WSDL Settings

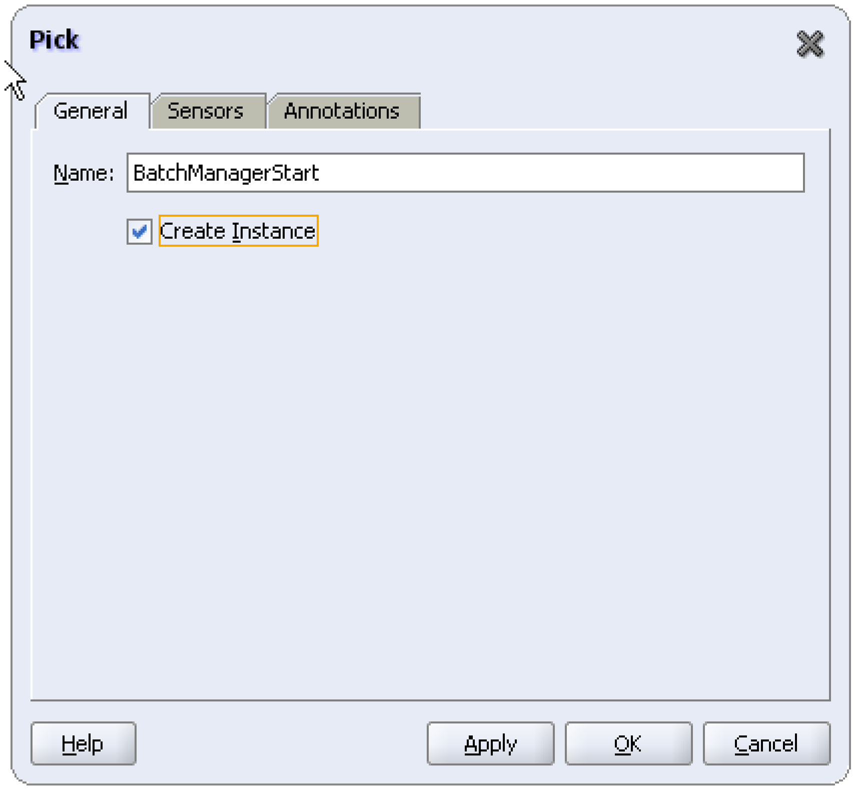
WSDL File: http://localhost:8889/orabpel/xmllib/jca/BatchManager.wsdl

Partner Link Type: BatchManagerInterfacePartnerLinkType

Partner Role: ----- Not Specified -----

My Role: BatchManagerInterfaceRole

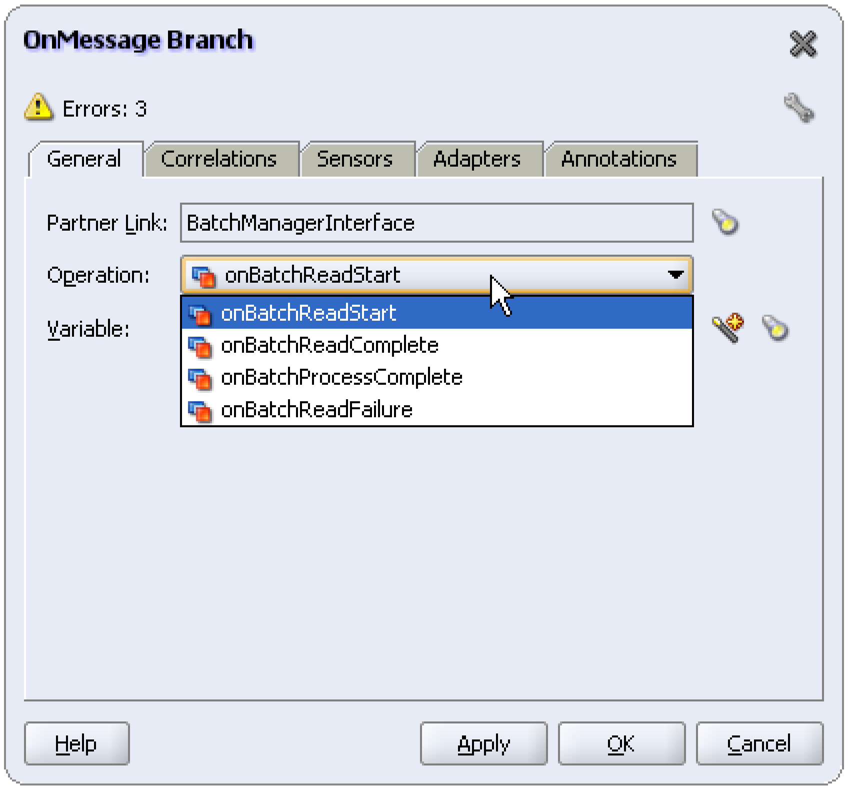
2. Add a pick and select the “Create Instance” check box.


Pick

Name: BatchManagerStart

[✓] Create Instance

3. Add an onMessage branch and select the Partnerlink created in step 1. Select the operation from the operation drop down list box.



Partner Link: BatchManagerInterface

Operation: onBatchReadStart

Variable:

onBatchReadStart
onBatchReadComplete
onBatchProcessComplete
onBatchReadFailure

Currently the Batch Manager Interface has the following operations

Operation	Message Type	Description
batchReadInitiateMessage	batchReadInitiateElement	The batchReadInitiateMessage message is raised with the File/FTP Adapter starts reading the file.
batchReadCompleteMessage	BatchReadCompleteElement	The batchReadInitiateMessage message is raised with the File/FTP Adapter has completed reading the file and started the BPEL processes..
batchProcessCompleteMessage	BatchProcessCompleteElement	Not Used.
batchReadFailureMessage	BatchReadFailureElement	The batchReadFailureMessage message is raised with the File/FTP Adapter issues wiith initiating a BPEL process for the batch.

All the element types mentioned above have the following elements

Element	Data Type	Description
batchId	String	The batchID is a unique identifier for the set of processes launched using debatching. The batchID is hash of the inbound file name and timestamp.
batchMetaData	String	
batchDescription	String	
batchPartialSize	Long	
process	String	The Process ID for the debatched Process.
domain	String	The domain of the debatched Process.

In the Batch Manager process, the ora:batchProcessActive(String batchId, String processId) and ora:batchProcessCompleted(String batchId, String processId) functions can be used to get information about the state of the debatched process.

☞ The ora:batchProcessActive(String batchId, String processId) and ora:batchProcessCompleted(String batchId, String processId) functions are not listed in expression builder in Jdeveloper.

4. Add the desired business logic in the sequence for the onMessage branch.

5. Add a new activation parameter batchNotificationHandler to the bpel.xml of the debatched process.

```
<property name="batchNotificationHandler">
          domain:encrypted-password|BatchNotificationProcess
</property>
```

The encrypted password is generated using the {SOA_HOME}/bpel/ bin/encrypt.[sh/bat] utility.

For setting the password during deployment, customized ant task should be used. The domain and password should be specified in the custom build.ml in the {PROCESS_HOME}/bpel/ using ant token <domain-name>.domain.locator.

```
 <property name="batchNotificationHandler">
          ${<domain-name>.domain.locator}|BatchNotificationProcess
</property>
```

A request would have to be submitted to add the domain.locator property for the domain where the Batch Notification Manager process is deployed.

The sample bpel.xml and build.xml shows an example of Batch Notification Process.

***Sample** bpel.xml:*

```
<?xml version = '1.0' encoding = 'UTF-8'?>
<BPELSuitcase>
   <BPELProcess id="DebatchSample" src="DebatchSample.bpel">
      <partnerLinkBindings>
         <partnerLinkBinding name="ReadContacts">
            <property name="wsdlLocation">ReadAddress.wsdl</property>
         </partnerLinkBinding>
         <partnerLinkBinding name="WriteContacts">
            <property
name="wsdlLocation">WriteContacts.wsdl</property>
            <property name="retryInterval">60</property>
            <property name="CONTACT_FILE_OUTPUT_DIR"
type="LogicalDirectory">D:\Temp\Integration-
Data\Debatching\Out</property>
         </partnerLinkBinding>
      </partnerLinkBindings>
      <activationAgents>
         <activationAgent
className="oracle.tip.adapter.fw.agent.jca.JCAActivationAgent"
partnerLink="ReadContacts">
            <property name="portType">Read_ptt</property>
            <property name="CONTACT_FILE_ARCHIVE_DIR"
type="LogicalArchiveDirectory">D:\Temp\Integration-
Data\Debatching\Archive</property>
            <property name="CONTACT_FILE_INPUT_DIR"
type="LogicalDirectory">D:\Temp\Integration-
Data\Debatching\In</property>
            <property
name="batchNotificationHandler">default:7747F4E13622D35AF789FE878DBB53
339A31B0C43B52175A|BatchManagerSample</property>
            <property name="rejectedMessageHandlers">

bpel://default:7747F4E13622D35AF789FE878DBB53339A31B0C43B52175A|Activa
tionError|initiate|message
                file://D:\Temp\Integration-
Data\Debatching\RejectedMessages
            </property>
            <property name="clusterGroupId">localhost</property>
         </activationAgent>
      </activationAgents>
   </BPELProcess>
</BPELSuitcase>
```

Sample Custom build.xml:

```
<?xml version="1.0" encoding="windows-1252" ?>
<project default="Replace" basedir=".">
```

```
    <target name="Replace">

        <echo>
--------------------------------------------------------------
| Performing substitutions bpel process ${process.name}, revision
${rev}
--------------------------------------------------------------
        </echo>
    <bpelc rev="${rev}" input="${process.dir}/bpel/bpel.xml"
out="${process.dir}/output" home="${bpel.home}">
       <customize>
         <activationAgent
className="oracle.tip.adapter.fw.agent.jca.JCAActivationAgent"
partnerLink="ReadContacts">
            <property name="portType">Read_ptt</property>
            <property
name="CONTACT_FILE_ARCHIVE_DIR">/tmp/debtachsample/archive</property>
            <property
name="CONTACT_FILE_INPUT_DIR">/tmp/debtachsample/in</property>
            <property name="batchNotificationHandler">
                      ${default.domain.locator}|BatchManagerSample
            </property>
            <property name="rejectedMessageHandlers">

bpel://${default.domain.locator}|ODRejectedMessageLogger|initiate|mess
age

file://${rejectedMessage.directory}/${domain}/${process.name}/rejected
Messages
            </property>
            <property
name="fatalErrorFailoverProcess">bpel://${default.domain.locator}|ODFa
talErrorLogger|initiate|message</property>
            <property
name="clusterGroupId">${cluster.group.id}</property>
         </activationAgent>
        </customize>
      </bpelc>
    </target>
</project>
```

6. To get the batch d in the debatched process, add the highlighted lines to the fileAdapterInboundHeader.wsdl created in the BPEL project directory (file location <ProjectDir>/bpel)

```
<definitions
     name="fileAdapter"
     targetNamespace="http://xmlns.oracle.com/pcbpel/adapter/file/"
     xmlns:tns="http://xmlns.oracle.com/pcbpel/adapter/file/"
     xmlns="http://schemas.xmlsoap.org/wsdl/" >
```

```
    <types>
        <schema attributeFormDefault="qualified"
elementFormDefault="qualified"

targetNamespace="http://xmlns.oracle.com/pcbpel/adapter/file/"
                xmlns="http://www.w3.org/2001/XMLSchema"

xmlns:FILEAPP="http://xmlns.oracle.com/pcbpel/adapter/file/">
          <element name="InboundFileHeaderType">
                <complexType>
                    <sequence>
                        <element name="fileName" type="string"/>
                        <element name="directory" type="string"/>
                        <element name="size" type="string"/>
                        <element name="batch" type="string"/>
                        <element name="batchIndex" type="string"/>
                    </sequence>
                </complexType>
            </element>
        </schema>
    </types>

    <!-- Header Message -->
    <message name="InboundHeader_msg">
      <part element="tns:InboundFileHeaderType" name="inboundHeader"/>
   </message>

</definitions>
```

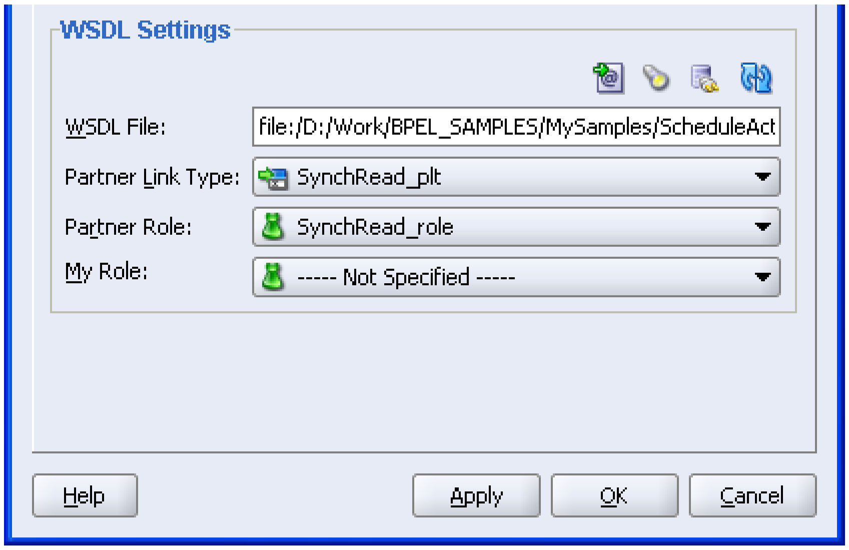
Processing All Lines of Large File in one process

Using the Chunk Read capability large file can be read in chunks and processed by a single BPEL Process.

☞ Patch 6316950 (FILEADAPTER HARPER ENHANCEMENTS: SYNC WRITE AND CHUNKED INTERACTION SPEC) is required to implement this functionality.

BPEL Designer does not support the Chunk Read capability. To use this feature the following changes have to done manually after a partner link is implemented

1. Create the Partnerlink using "Synchronous File Read" or "Synchronous File Get" operation of the File Adapter and FTP Adapter respectively. If an XSD does not exist for the file format, the native schema wizard can be used to generate the XSD. At this stage BPEL Designer would create a Partnerlink that look like the diagram shown below



WSDL File: file:/D:/Work/BPEL_SAMPLES/MySamples/ScheduleAct
Partner Link Type: SynchRead_plt
Partner Role: SynchRead_role
My Role: ----- Not Specified -----

2. The following modifications will have to made manually to the WSDL generated in the step above
    a. Replace all occurrences of SynchRead with SynchChunkedRead
    b. In the jca:operation, change the InteractionSpec to oracle.tip.adapter.file.outbound.ChunkedInteractionSpec
    c. In the jca:operation, add the desired batch size using the parameter ChunkSize

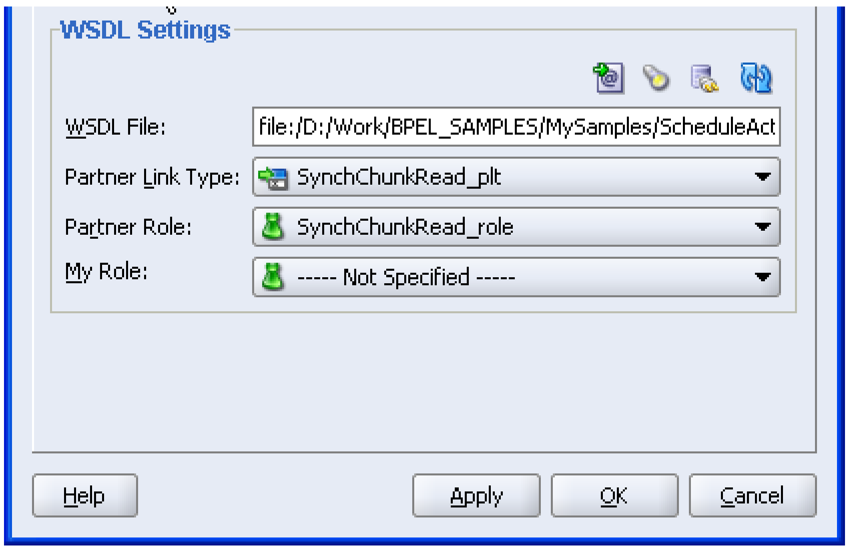
WDSL Sample

```
<definitions
     name="ChunkReadFile"

targetNamespace="http://xmlns.oracle.com/pcbpel/adapter/file/ChunkRead
File/"
     xmlns="http://schemas.xmlsoap.org/wsdl/"

xmlns:tns="http://xmlns.oracle.com/pcbpel/adapter/file/ChunkReadFile/"
     xmlns:plt="http://schemas.xmlsoap.org/ws/2003/05/partner-link/"
     xmlns:jca="http://xmlns.oracle.com/pcbpel/wsdl/jca/"
     xmlns:imp1="http://xmlns.oracle.com/ReadFileHeader"
     xmlns:hdr="http://xmlns.oracle.com/pcbpel/adapter/file/"
    >
    <import namespace="http://xmlns.oracle.com/pcbpel/adapter/file/"
location="fileAdapterOutboundHeader.wsdl"/>
    <types>
      <schema
targetNamespace="http://xmlns.oracle.com/pcbpel/adapter/file/ChunkRead
File/"
              xmlns="http://www.w3.org/2001/XMLSchema" >
        <import namespace="http://xmlns.oracle.com/ReadFileHeader"
schemaLocation="AP_EXPENSE_NA_TDM.xsd" />
        <element name="empty"><complexType/></element>
      </schema>
    </types>
    <message name="Empty_msg">
        <part name="Empty" element="tns:empty"/>
    </message>
    <message name="InvoiceFile_msg">
        <part name="InvoiceFile" element="imp1:InvoiceFile"/>
    </message>
    <portType name="SynchChunkedRead_ptt">
        <operation name="SynchChunkedRead">
            <input message="tns:Empty_msg"/>
            <output message="tns:InvoiceFile_msg"/>
        </operation>
    </portType>
    <binding name="SynchChunkedRead_binding"
type="tns:SynchChunkedRead_ptt">
    <jca:binding  />
        <operation name="SynchChunkedRead">
      <jca:operation
          LogicalDirectory="InputDir"

InteractionSpec="oracle.tip.adapter.file.outbound.ChunkedInteractionSp
ec"
          ChunkSize="5"
          LogicalArchiveDirectory="ArchiveDir"
          DeleteFile="true"
          FileName="InputFile"
          OpaqueSchema="false" >
      </jca:operation>
      <input>
        <jca:header message="hdr:OutboundHeader_msg"
part="outboundHeader"/>
      </input>
        </operation>
    </binding>
    <service name="ChunkReadFile">
        <port name="SynchChunkedRead_pt"
binding="tns:SynchChunkedRead_binding">
      <jca:address location="eis/FileAdapter" />
        </port>
    </service>
  <plt:partnerLinkType name="SynchChunkedRead_plt" >
    <plt:role name="SynchChunkedRead_role" >
      <plt:portType name="tns:SynchChunkedRead_ptt" />
    </plt:role>
  </plt:partnerLinkType>
</definitions>
```

3. Edit the Partnerlink created using the sync Read.
    d. Place the cursor in the WSDL File Field and Tab out. This would set the Partnerlink Type to SyncChunkRead_plt and reset Partner Role.

e. Select SyncChunkRead_Role as the Partner Role.

At this stage the WSDL setting would as follows:



WSDL File: file:/D:/Work/BPEL_SAMPLES/MySamples/ScheduleAct
Partner Link Type: SynchChunkRead_plt
Partner Role: SynchChunkRead_role
My Role: ----- Not Specified -----

4. Add the highlighted lines to the fileAdapterOutboundHeader.wsdl created in the BPEL project directory (file location <ProjectDir>/bpel)

```
<definitions
     name="fileAdapter"
     targetNamespace="http://xmlns.oracle.com/pcbpel/adapter/file/"
     xmlns:tns="http://xmlns.oracle.com/pcbpel/adapter/file/"
     xmlns="http://schemas.xmlsoap.org/wsdl/" >

    <types>
        <schema attributeFormDefault="qualified" elementFormDefault="qualified"
                targetNamespace="http://xmlns.oracle.com/pcbpel/adapter/file/"
                xmlns="http://www.w3.org/2001/XMLSchema"
                xmlns:FILEAPP="http://xmlns.oracle.com/pcbpel/adapter/file/">
            <element name="OutboundFileHeaderType">
                <complexType>
                    <sequence>
                        <element name="fileName" type="string"/>
                        <element name="directory" type="string"/>
                        <element name="lineNumber" type="string"/>
                        <element name="columnNumber" type="string"/>
                        <element name="recordNumber" type="string"/>
                        <element name="isMessageRejected" type="string"/>
                        <element name="rejectionReason" type="string"/>
                        <element name="NoDataFound" type="string"/>
                        <element name="isEOF" type="string"/>
                    </sequence>
                </complexType>
            </element>
        </schema>
    </types>
    <!-- Header Message -->
    <message name="OutboundHeader_msg">
      <part element="tns:OutboundFileHeaderType" name="outboundHeader"/>
   </message>

</definitions>
```

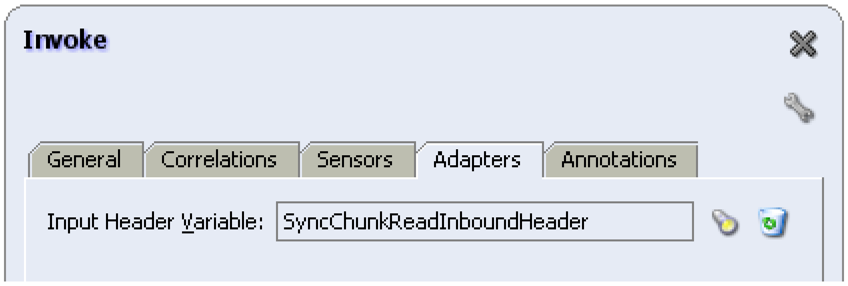
5. Create two header variables (inboundHeader and outboundHeader) using the fileAdapterOutboundHeader.wsdl.

6. Initialize the two header variables as follows:
   f. For the Inbound Header Variable (inboundHeader) set the Directory and filename to the file that has to be read. Also set the lineNumber, columnNumber and recordNumber fields to “”.
   g. For the Outbound Header Variable (outboundHeader) set the isEOF field to “false”.

7. Create a loop with the condition `outboundHeader.isEOF != ‘true’`

8. Inside the loop, add a invoke activity to the Partnerlink for the SyncChunkRead and create the input and output variables.

9. On the Adapter Tab, add the Inbound Header Variable as the header variable for the activity.



Input Header Variable: SyncChunkReadInboundHeader

10. Using the source view, manually add a bpelx:outputHeaderVariable using the Output Header Variable

```
    <invoke name="ReadInvoiceData" partnerLink="ChunkReadFile"
            portType="ns3:SynchChunkedRead_ptt"
            operation="SynchChunkedRead"
            inputVariable="ReadInvoiceDataInput"
            outputVariable="ReadInvoiceDataOutput"
            bpelx:outputHeaderVariable="SyncChunkReadOutboundHeader"
            bpelx:inputHeaderVariable="SyncChunkReadInboundHeader"/>
```

11. Assign the output header variable to the input header variable. This is required to read the next set of records.

12. The output variable in the invoke step would have the data read from the file.

13. The `isMessageRejected` and `rejectionReason` in the output header should be used for error handling. Also, if the last read operation has no records, the `NoDataFound` field in the output header is set to 'true'

☞ If a file/ftp based activation is used, the activation should not delete the file as it would be required for the SyncChuckRead operation.

☞ The SyncChuckRead operation does not delete the file generated.

Specifying Output Directory while writing a file.

By default the fileAdapterInboundHeader.wsdl only has the fileName element. To be able to specify output directory at runtime, implement the following steps.

1. Build a Partnerlink with write operation.

2. Add the highlighted lines to the fileAdapterOutboundHeader.wsdl created in the BPEL project directory (file location <ProjectDir>/bpel)

```
<definitions
     name="fileAdapter"
     targetNamespace="http://xmlns.oracle.com/pcbpel/adapter/file/"
     xmlns:tns="http://xmlns.oracle.com/pcbpel/adapter/file/"
     xmlns="http://schemas.xmlsoap.org/wsdl/" >

    <types>
        <schema attributeFormDefault="qualified" elementFormDefault="qualified"
                targetNamespace="http://xmlns.oracle.com/pcbpel/adapter/file/"
                xmlns="http://www.w3.org/2001/XMLSchema"
                xmlns:FILEAPP="http://xmlns.oracle.com/pcbpel/adapter/file/">
            <element name="OutboundFileHeaderType">
                <complexType>
                    <sequence>
                        <element name="fileName" type="string"/>
                        <element name="directory" type="string"/>
                    </sequence>
                </complexType>
            </element>
        </schema>
    </types>

    <!-- Header Message -->
    <message name="OutboundHeader_msg">
      <part element="tns:OutboundFileHeaderType" name="outboundHeader"/>
   </message>
```

```
</definitions>
```

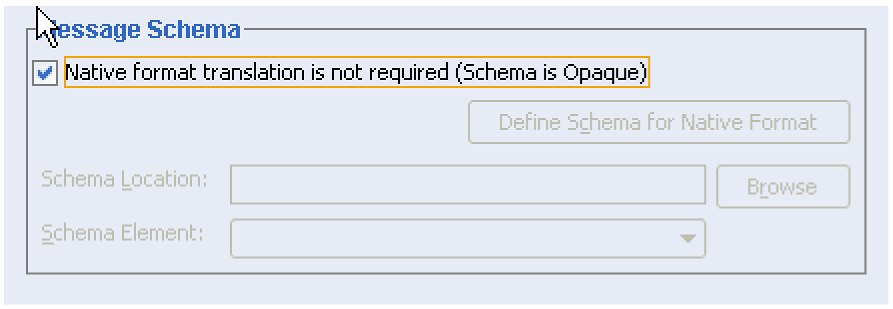
3. Create a header variable using the fileAdapterOutboundHeader.wsdl and assign the Output directory and file name.

4. Assign the header variable using the Adapter tab of the invoke activity for the Partnerlink.

Appending to a File

To append to a file, implement the following steps

1. Build a Partnerlink with write operation using the Adapter Wizard.



☑ Native format translation is not required (Schema is Opaque)

This would create a WSDL similar to the one below.

```
      <jca:operation
          LogicalDirectory="CONTACT_FILE_OUTPUT_DIR"
InteractionSpec="oracle.tip.adapter.file.outbound.FileInteractionSpec"
          FileNamingConvention="Contact%SEQ%.txt"
          NumberMessages="1"
          OpaqueSchema="false">
      </jca:operation>
```

2. Manually add the Append property to the JCA operation.

```
      <jca:operation
          LogicalDirectory="CONTACT_FILE_OUTPUT_DIR"

InteractionSpec="oracle.tip.adapter.file.outbound.FileInteractionSpec"
          FileNamingConvention="Contact%SEQ%.txt"
          NumberMessages="1"
          OpaqueSchema="false"
          Append="true" >
      </jca:operation>
```