

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

## FTP Adapter inbound lab

What this exercise is about .....	2
Lab requirements .....	2
What you should be able to do .....	2
Introduction .....	3
Exercise instructions .....	4
Part 1: Initialize the workspace and prepare for the lab.....	6
Part 2: Pass through scenario.....	7
2.1.  Configure pass through using external service wizard .....	9
2.2.  Add Java component .....	28
2.3.  Test pass through scenario.....	30
2.4.  Test pass through scenario with SplitBySize .....	33
Part 3: Content specific (non-pass through) scenario.....	37
3.1.  Configure content specific (non-pass through) scenario using external service wizard.....	39
3.2.  Add Java component .....	54
3.3.  Test non pass through scenario.....	56
3.4.  Test non pass through scenario with SplitByDelimiter.....	58
Part 4: Use default function selector and data binding .....	62
4.1.  Configure inbound using default function selector and data binding .....	63
4.2.  Add Java component .....	67
4.3.  Test defaults scenario .....	68
Part 5: Use 'Create a service from a typical pattern' .....	69
5.1.  Configure inbound using 'Create a service from a pattern (typical)' option .....	70
5.2.  Add Java component .....	76
5.3.  Test typical pattern scenario .....	78
What you did in this exercise .....	79
Task: Adding remote server to WebSphere Integration Developer test environment .....	80

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

### What this exercise is about

The objective of this lab is to provide you with an understanding of WebSphere Adapter for FTP and inbound event processing. In this lab you will deploy the WebSphere Adapter for FTP, using WebSphere Integration Developer, and integrate it with an SCA application that polls for inbound events and processes those inbound requests from the file system.

### Lab requirements

List of system and software required for the student to complete the lab.

- WebSphere Integration Developer V6.2 installed and updated with latest fixes
- WebSphere Process Server V6.2 Test Environment installed and updated latest fixes
- Extract Labfiles62.zip to your C:\ (your root) drive

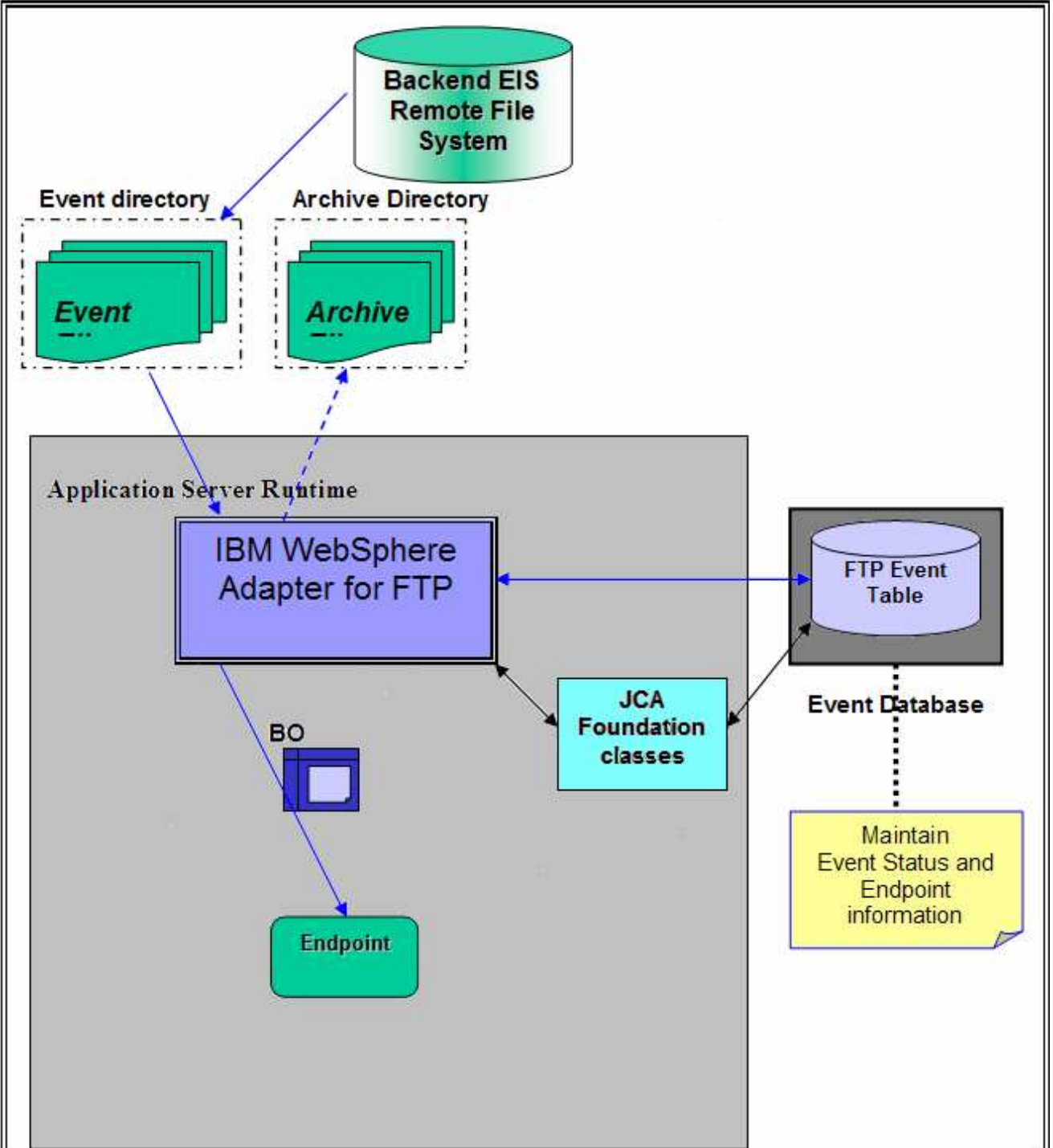
### What you should be able to do

At the end of this lab you should be able to:

- Import FTP adapter RAR file into WebSphere Integration Developer
  - Use External Service wizard to configure Activation Spec Properties, Resource Adapter Properties to generate Business Objects and other artifacts and then configure Function Selector, Data Binding and Data Handlers
  - Deploy the adapter application onto WebSphere Process Server
  - Test the deployed application using WebSphere Process Server test environment for both pass-through and non pass-through using different scenarios and patterns
  - Restore the server configuration
-

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

### Introduction



## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

The backend EIS is the source of events. When events are generated at the EIS, files are created by the EIS in the remote file system at a specific directory location. The same directory needs to be configured by you as the Event Directory for the adapter.

The adapter polls event files, based on configured Event File Mask and FTP Get Quantity, from the Event Directory and downloads to a Local Event Directory in the adapter machine for every FTP Poll Frequency time and archives the file in FTP Archive Directory of the FTP server (it specified by you).

The adapter splits the downloaded event files from Local Event Directory based on the configured SplittingFunctionClassname (splitting functionality) and SplitCriteria (splitting criteria used). User can implement his/her own class which contains the splitting logic which splits the event files into individual Business Objects. The adapter provides a Java™ interface which you have to implement in a class.

An entry is made in the Event Persistence table (given by EPEventTableName) for each BO with status as New Event. The adapter sends the record chunk, through a Function Selector and Data Binding, to the endpoint and the status in the Event Persistence table is updated based on successful (to PROCESSED state) or failed(to FAILED state) delivery to the endpoint. The event management framework takes care of delivering the event only once to the endpoint.

If archiving is enabled (if LocalArchiveDirectory has a valid value), the Business Objects are also archived in a configured Local Archive Directory, on the local file system, based on successful or failed delivery to the endpoint. Successful and failed events are written to different files. The entries from the Event Persistence table are deleted for each of the successfully processed BO's only. If Local ArchiveDirectory is not set, the event file is deleted after processing of all the BO's is completed.

If EventContentType in the Activation Spec Properties is not set or if it is not valid or the value does not match the entries present in annotation of the Wrapper data object then it is called PassThrough and data transformation will not happen in this case. During PassThrough the SplittingFunctionClassName and SplitCriteria are set to values (even if they are set to different values) such that splitting happens based on file size. In the PassThrough scenario you can have either Chunking or FilePassbyReference features.

---

## Exercise instructions

Some instructions in this lab are Windows operating-system specific. If you plan on running the lab on an operating-system other than Windows, you will need to run the appropriate commands, and use appropriate files (.sh or .bat) for your operating system. The directory locations are specified in the lab instructions using symbolic references, as follows:

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

Reference variable	Windows® location	AIX® or UNIX® location
<WID_HOME>	C:\Program Files\IBM\WID62	
<FTPADAPTER_HOME>	<WID_HOME>\ResourceAdapters\FTP_6.2.0.0\deploy	
<LAB_FILES>	C:\Labfiles62	/tmp/Labfiles62
<WORKSPACE>	<LAB_FILES>\FTPInbound\workspace	
<LOCAL_EVENT_DIR>	<LAB_FILES>\FTPInbound\LocalEventDir	
<LOCAL_ARCHIVE_DIR>	<LAB_FILES>\FTPInbound\LocalArchivetDir	
<FTPFILERS>	<LAB_FILES>\FTPFiles	
<TEMP>	C:\temp	/tmp

**Windows users' note:** When directory locations are passed as parameters to a Java program such as EJBdeploy or wsadmin, it is necessary to replace the backslashes with forward slashes to follow the Java convention. For example, replace C:\Labfiles62\ with C:/Labfiles62/.

### Instructions if using a remote server for testing

Note that the previous table is relative to where you are running WebSphere Integration Developer. The table below is related to where you are running the remote test environment:

Reference variable	Example: Remote Windows test server location	Example: Remote z/OS® test server location	Input your values for the remote location of the test server
<SERVER_NAME>	server1	sssr011	
<WAS_HOME>	C:\Program Files\IBM\WebSphere\AppServer	/etc/sscell/AppServer	
<HOSTNAME>	localhost	mvsxxx.rtp.raleigh.ibm.com	
<SOAP_PORT>	8880	8880	
<TELNET_PORT>	N/A	1023	
<PROFILE_NAME>	AppSrv01	default	
<USERID>	N/A	ssadmin	
<PASSWORD>	N/A	fr1day	

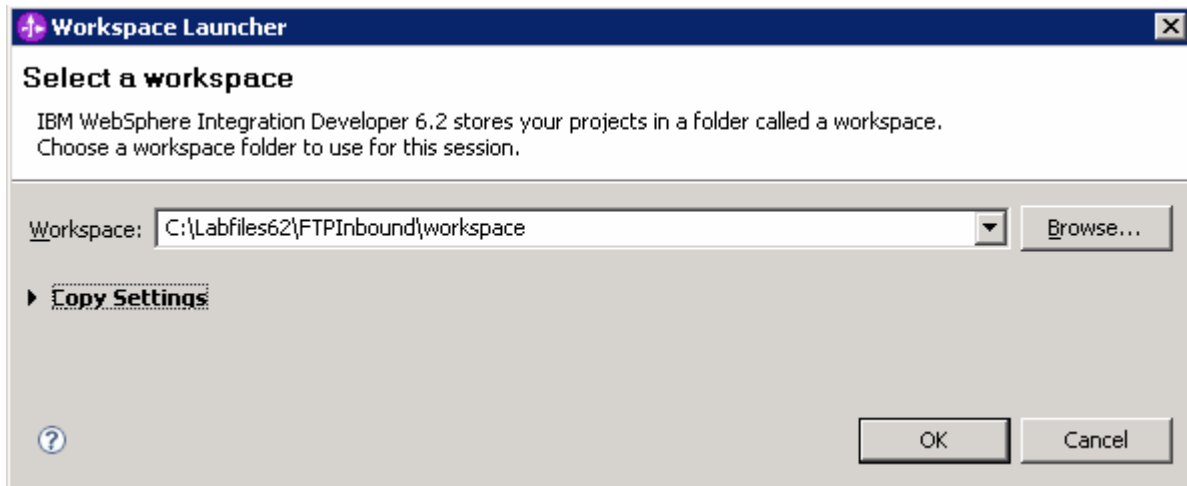
Instructions for using a remote testing environment, such as z/OS, AIX or Solaris, can be found at the end of this document, in the section "[Task: Adding remote server to WebSphere Integration Developer test environment](#)".


## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

## Part 1: Initialize the workspace and prepare for the lab

This part of the lab, you will start the WebSphere Integration Developer V6.2 with a new workspace and create required data source and database using the administrator console of WebSphere Process Server V6.1

- \_\_\_ 1. Extract the provided Labfiles62.zip to your C:\ (root) drive, if you have not already done so. This will create the necessary subdirectory structure to complete the lab, and provides you with sample text files
- \_\_\_ 2. Start the WebSphere Integration Developer V6.2 with a new workspace
  - \_\_\_ a. Select **Start > All Programs > IBM WebSphere Integration Developer > IBM WebSphere Integration Developer V6.2 > WebSphere Integration Developer V6.2**
  - \_\_\_ b. From the Workspace Launcher window, enter **<WORKSPACE>** for the Workspace field



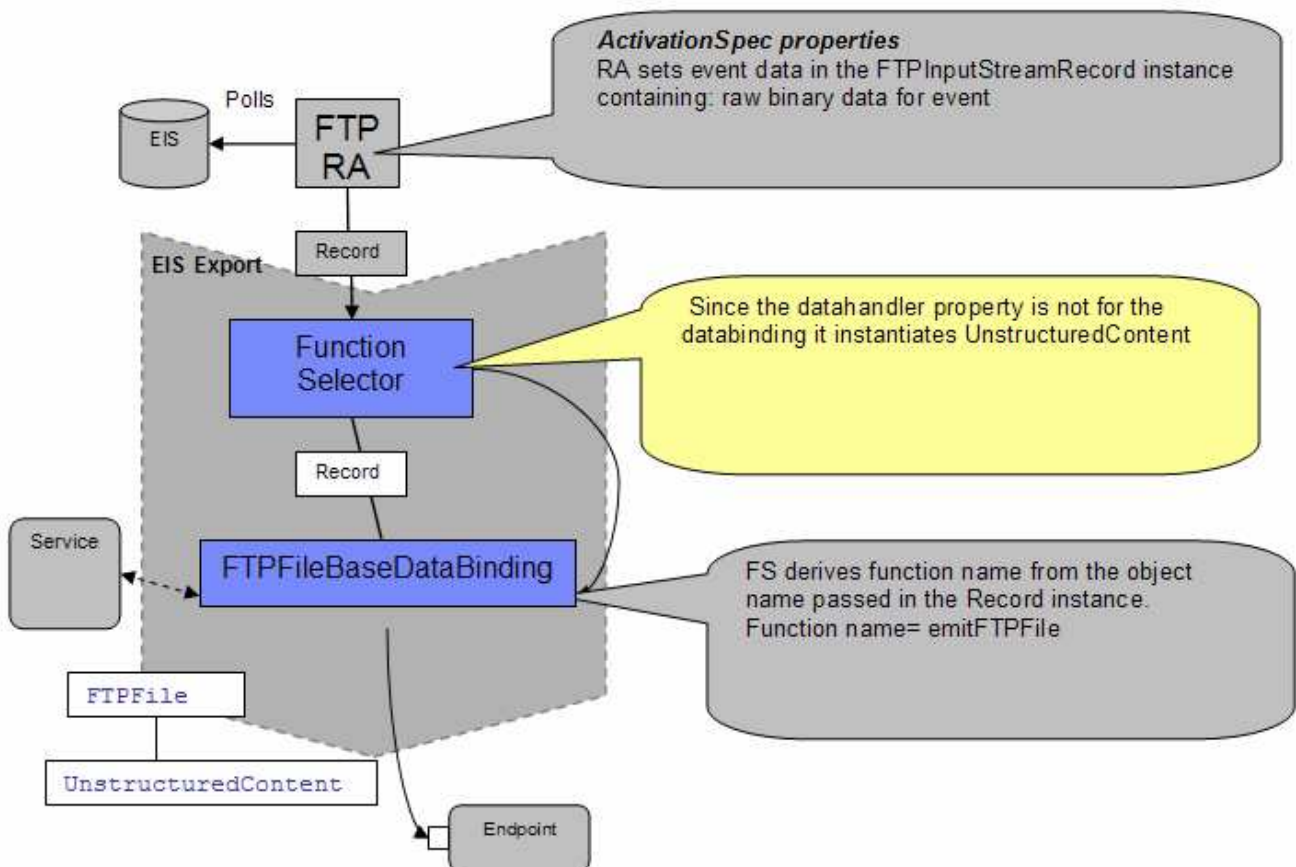
- \_\_\_ 3. Click the  button on the right corner to close the Welcome page and proceed with the workbench
- \_\_\_ 4. Follow the instructions of **'Configure data source'** task with these inputs and create the data source and data base required for this lab:
  - \_\_\_ a. Data source name: **FTP**
  - \_\_\_ b. JNDI name: **jdbc/FTP**
  - \_\_\_ c. Database name: **FTPDB**
- \_\_\_ 5. Create directory structure on your FTP Server
  - \_\_\_ a. Log onto FTP machine/FTP Server using your ftp user and its password
  - \_\_\_ b. Create an **EventDir** and an **ArchiveDir** under the user's home directory:
    - 1) mkdir **EventDir**
    - 2) mkdir **ArchiveDir**

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

## Part 2: Pass through scenario

Inbound support can be broadly classified into two flows, one that involves data transformation and another without it (pass-through). In this part of the lab, you will configure the pass through scenario using the new External Service option from the WebSphere Integration Developer and then test the configuration with different cases.

Pass through flow for the inbound scenario:



- Event data is picked from the event file based on SplitCriteria (it takes a value in bytes), converted into a Input Stream and set on FTPInputStreamRecord.
- The protocol specific information like event file name, directory name are also set in the FTPInputStreamRecord. If the PassThrough has a chunk file or if it is FilePassByReference or default one is also indicated.
  - In case of FilePassByReference the directory name corresponds to LocalArchiveDirectory and file name to event file name appended with timestamp.
  - In case of chunking the directory name indicates LocalEventDirectory, file name represents event file name and ChunkInfo represents the chunk details.
  - In case of normal PassThrough the directory name indicates LocalEventDirectory and file name represents the event file.

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- If the databinding does not contain information about the datahandler, then the inbound scenario is considered as PassThrough.
- The FTPInputStreamRecord is sent to the base class and from there to the Function Selector. If no matching DataBinding defined or the invoked DataBinding can not resolve to a BO, no content-specific data binding is invoked but UnstructuredContent SDO is instantiated. It is set with byte content present in the input stream object.
- Optionally the FTPFile wrapper is set with protocol specific information and UnstructuredContent is set in FTPFile
- FTPFile wrapper is directly sent to the endpoint or the wrapper data object is set in FTPFileBG and sent to the endpoint in case you want to use Business Graphs

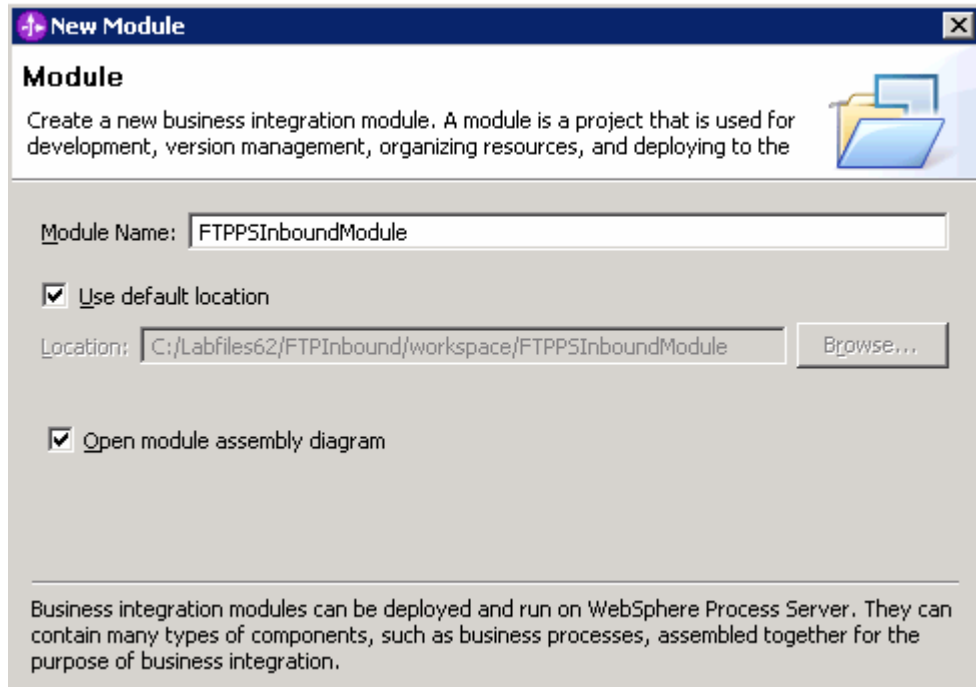


## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

## 2.1. Configure pass through using external service wizard

In this part of the lab you will use this new external service feature to create and configure the function selector, data binding and other required artifacts to test the inbound pass through scenario

- \_\_\_ 1. Create the module: FTPPSInboundModule
  - \_\_\_ a. From the Business Integration window, right-click and select **New > Module**
  - \_\_\_ b. From the New Module window, enter **FTPPSInboundModule** for the Module Name

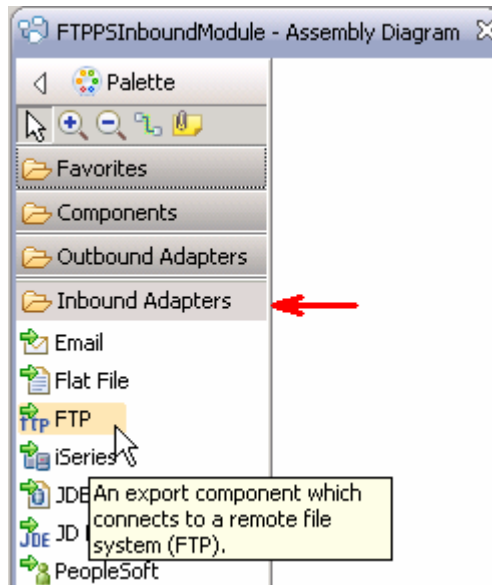


- \_\_\_ c. Ensure that the box next to **Open module assembly diagram** is checked and then click **Finish**
- You will now see a new module, FTPPSInboundModule, created from your Business Integration window

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

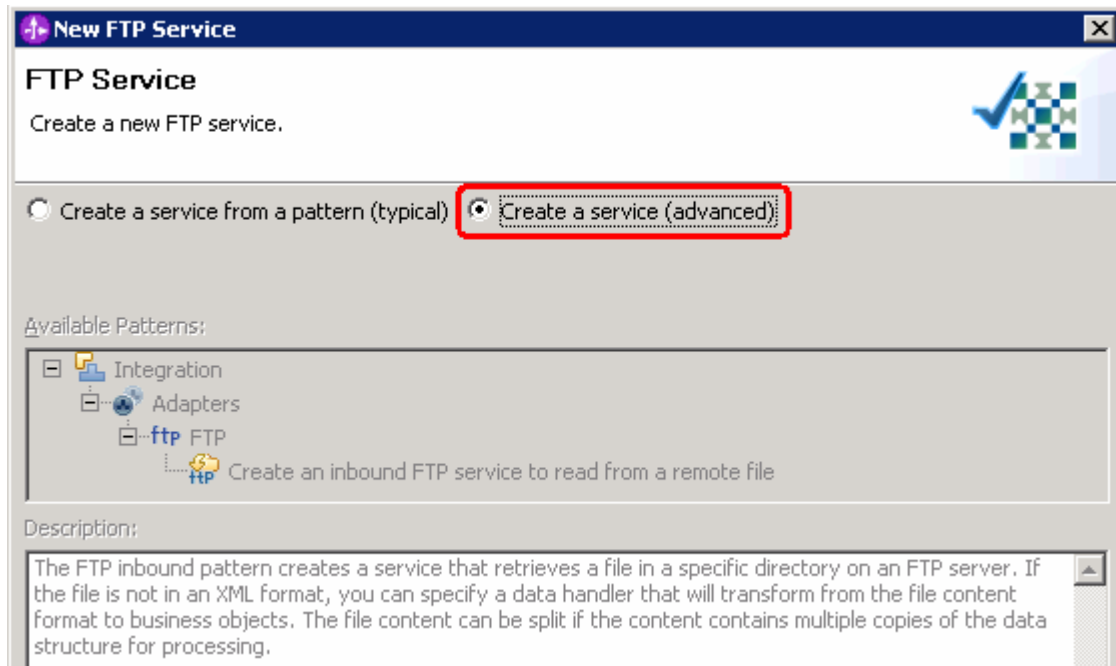
\_\_\_ 2. To start External Service from the Palette:

\_\_\_ a. From the **Palette** on the left side of Assembly Diagram, click **Inbound Adapters**:



\_\_\_ b. Under Inbound Adapters, click the **FTP** and then click the empty canvas of the assembly diagram. The New FTP File Service wizard is opened

\_\_\_ 3. From the FTP Service screen, select **Create a service (advanced)**



\_\_\_ a. Click **Next**

---

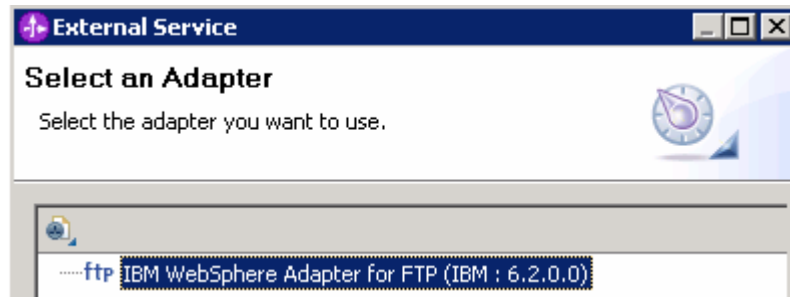
**Note:** You can also start the External Service from the **File menu** option:

---

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

From the main menu, select **File > New > External Service**. This opens an External Service wizard that helps you obtain a service which establishes connectivity with other systems. Select **Adapters > FTP** and click **Next**

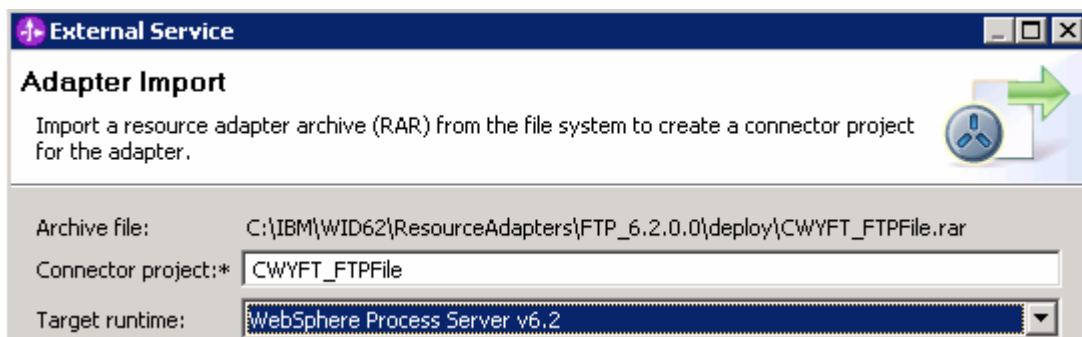
- \_\_\_ 4. On the Select an Adapter screen, select **IBM WebSphere Adapter for FTP (IBM : 6.2.0.0)** and click **Next**



- \_\_\_ 5. Adapter Import screen:

In this step, you will import a connector resource adapter archive from the file system into your WebSphere Integration Developer workspace. The adapter RAR file already exists under **<FTPADAPTER\_HOME>**.

- \_\_\_ a. The default Connector file is selected which is shipped along with WebSphere Integration Developer
- \_\_\_ b. Accept the default name for Connector project, **CWYFT\_FTPFile**. You can change it to any other name, but for this lab, you can leave the default name.
- \_\_\_ c. For Target server, ensure that **WebSphere Process Server v6.2** is selected



- \_\_\_ d. Click **Next**

**Note:** The resource adapter archive file is imported and a new connector project, **CWYFT\_FTPFile**, is listed under Business Integration view.

**Note:** If you are using the **File menu** option to start the External Service wizard, you are asked to select the **Processing Direction** at this point. Select the radio button next to **Inbound** and click **Next** to proceed to the next step.

- \_\_\_ 6. Service Configuration Properties:

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

\_\_\_ a. Deploy connector project: ensure that the default option **With module for use by single application** is selected

\_\_\_ b. Enter these for FTP system connection information:

- 1) Host name: **<FTP\_Machine\_Name>** (or IP Address of the machine that has FTP Server),  
for Ex: wsbeta181.austin.ibm.com
- 2) Remote directory: **full path of the EventDir created in on the machine where FTP server is existing** (for Ex: /home/wsbeta/EventDir)

---

**Note:** This is the directory from where adapter gets the event files. Alternatively, you can also replace the absolute directory path with WebSphere variables for the Event directory, Archive directory. Refer to '**FTP adapter – Processing COBOL copy book files lab**' for more details on this new feature introduced in V6.2.

---

- 3) Local directory: **<LOCAL\_EVENT\_DIR>**
- 4) Protocol: **FTP – file transfer protocol** (default)

---

**Note:** Refer to '**Install and configure SSH server**' for more details on **SFTP – secure shell (SSH) file transfer protocol**.

---

- 5) Port number: **21** (default)
- 6) **User name:** username using which you connect to your FTP server (for Ex: **root**)
- 7) **Password:** password for the above user to connect to your FTP server

Deploy connector project: With module for use by single application

Connection properties: Use properties below

Connection properties

FTP system connection information

Host name: \* wsbeta181.austin.ibm.com

Remote directory:\* /home/wsbeta/EventDir

Local directory: \* C:\Labfiles62\FTPInbound\LocalEventDir **Browse...**

Protocol: FTP - file transfer protocol

Port number: 21

The user name and password will not be encrypted and will be stored as plain text.

User name: root

Password: \*\*\*\*\*

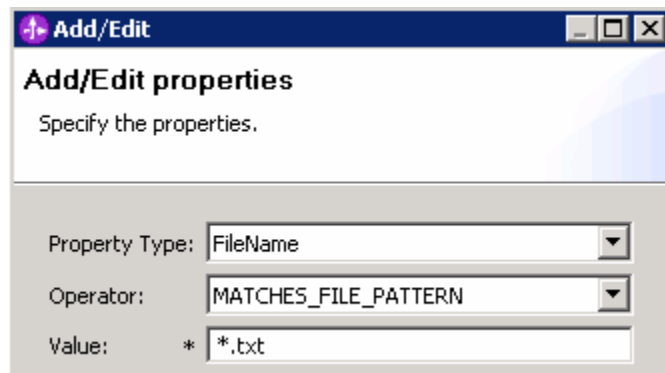
IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

\_\_\_ 7. Rule Editor: Configure rules for event filtering

**Note:** You will configure three rules to match three different file name patterns. Adapter polls only those files, which match the rule that is configured here.

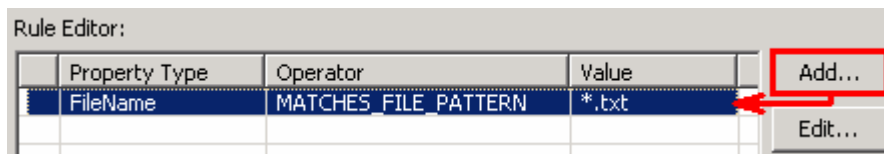
\_\_\_ a. Define a rule to filter only .txt files:

- 1) Click **Add...** next to the Rule Editor table
- 2) From the Add/Edit window, provide the values as shown below:
  - a) Property Type: select **FileName** from the drop down list
  - b) Operator: select **MATCHES\_FILE\_PATTERN** from the drop down list
  - c) Value: **\*.txt**

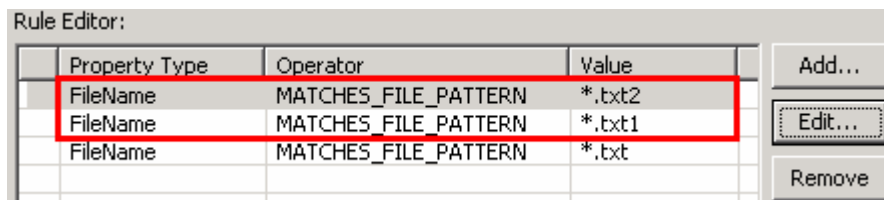


d) Click **Finish**

3) You should see a new rule entry in the table as shown below:

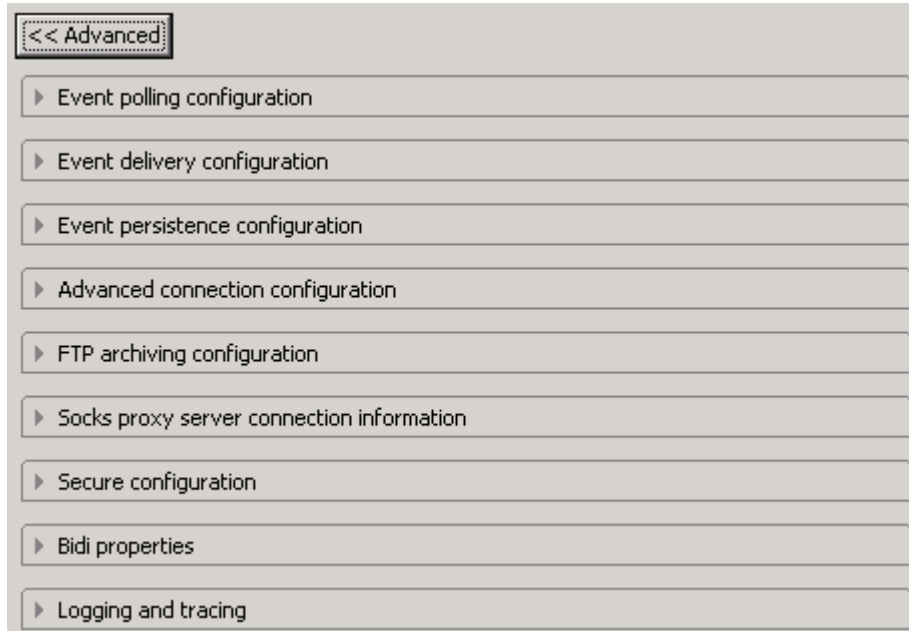


\_\_\_ b. Repeat the above instructions listed in step 8.a and create two more rules as shown below:



## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

\_\_ c. Click **Advanced >>** to see the hidden advanced properties that can be configured:



You can click each of the configuration and review the options available under it. For this lab, you will need only some of these properties.

\_\_ d. Event polling configuration: This has all the polling configuration details and for this lab, you can accept the defaults.

\_\_ e. Event delivery configuration:

- 1) **Ensure once-only event delivery:** You should check this box only if you are using data source and table name in the Event persistence configuration (below). If this property is set to true, while using in-memory capability (explained below), the adapter will log a warning message. By default this is selected and you can accept the default selection.

\_\_ f. Event persistence configuration:

- 1) Ensure that the **Auto create event table** is **checked**
- 2) Event recovery table name: **FTPPSTABLE**
- 3) Event recovery data source (JNDI) name: **jdbc/FTP**

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

**Note:** This represents the JNDI name of the Data Source used by Event Persistence to get the JDBC database connection. The Data Source must be created in the WebSphere Process Server. You should enter the data source JNDI name that you created in Step 3 of Part 1.

Event persistence configuration

Auto create event table

Event recovery table name: FTPPSTABLE

Event recovery data source (JNDI) name: jdbc/FTP

User name used to connect to event data source:

Password used to connect to event data source:

Database schema name:

**Note:** The Event recovery data source (JNDI) name is **not mandatory** from V6.1. Now, the adapter can use **in-memory representation** of event table to store all the necessary information. Adapter uses this feature when event database information is not configured during inbound event polling. This feature will not support the capability of handling “Ensure once-only event delivery”.

\_\_\_ g. Advanced connection properties:

- 1) **Pass only file name and directory, not the content (FilePassByReference):** Accept the defaults. This property determines if the adapter needs to load the contents of the file or just provide information on the directoryName and fileName. If the value is true, the adapter just provides the directory name and file name. This is **not selected by default**.
- 2) **Include business object delimiter in the file content (IncludeEndBODelimiter):** Accept the defaults. When this property is set to true (selected) the delimiter (given in Split criteria) is appended to the BO content before further processing is done. This property is valid only if you are splitting the event files based on a delimiter, that is., only if the Split function class name (below) is com.ibm.j2ca.utils.filesplit.SplitByDelimiter. This is **not selected by default**.
- 3) **Split function class name:** Accept the defaults. This value takes a fully qualified class name of the class to be used in order to split the event file. It takes two values as of now:
  - **com.ibm.j2ca.utils.filesplit.SplitBySize:** a class which splits the event file based on event file size. This is the **default** value for the split function class name.
  - **com.ibm.j2ca.utils.filesplit.SplitByDelimiter:** a class which splits the event file based on delimiter(used to separate BO's in event file)

The delimiter or file size is specified in split criteria attribute.
- 4) **Specify the criteria to split file content:** Accept the defaults. This attribute takes different values based on value set in Split function class name.
  - If split function class name is set to **com.ibm.j2ca.utils.filesplit.SplitBySize**, then split criteria must contain a valid number which represents the size in bytes. If event file size is greater than this value, it is split into chunks of this value and so many chunks are posted. If event file size is less than this value the entire event file is posted in one shot. When split criteria=0 (**default**), chunking is disabled.

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

- If split function class name is set to **com.ibm.j2ca.utils.filesplit.SplitByDelimiter**, then split criteria must contain the delimiter which separates the BO's in the event file.

---

**Note:** You will use the SplitByDelimiter class later in the content specific scenario.

---

\_\_ h. FTP archiving configuration:

- 1) Local Archive directory: click **Browse...** and select **<LOCAL\_ARCHIVE\_DIR>**
- 2) Remote archive directory: **full path of the ArchiveDir created in on the machine where FTP server is existing** (for Ex: /home/wsbeta/ArchiveDir)

FTP archiving configuration

Specify local archive directory to enable archiving on the local system, specify remote archive directory to enable archiving on the remote system.

Local archive directory: C:\Labfiles62\FTPInbound\LocalArchiveDir **Browse...**

File extension for local archive: original

Success file extension for local archive: success

Failure file extension for local archive: fail

Remote archive directory: /home/wsbeta/ArchiveDir

File extension for remote archive:

- \_\_\_ 8. **Secure configuration:** Refer to the new lab 'Install and configure SSH server' for more details on this new feature

Secure configuration

Enable remote server authentication for SFTP protocol

Host key file:  Browse...

Private key file:  Browse...

Passphrase:

- \_\_\_ 9. **Logging and tracing:** Refer to the new lab 'Log and confidential trace lab' for more details on this new feature

Logging and tracing

Adapter ID:\* 001

Disguise user data as "XXX" in log and trace files.

- \_\_\_ 10. For this lab, you are not going to use the J2C authentication. So, **uncheck** the box next to **Specify a Java Authentication and Authorization Services (JAAS) alias security credentials**.

Service properties

Specify a Java Authentication and Authorization Services (JAAS) alias security credential.

J2C authentication data entry:



## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

**Function Selector Configuration:** Function Selectors are required in order to map between events generated by resource adapters and the appropriate SCA export function name.

There are two function selectors provided by the adapter foundation classes that are supported by the FTP adapter - FilenameFunctionSelector and EmbeddedNameFunctionSelector.

FilenameFunctionSelector is used for generic FTP business object, where the object name cannot be determined from the event file. The FilenameFunctionSelector is a rule-based function selector that can match a regular expression on a file name to an object name. This is represented in properties as a 2-column table, with N rows. So, you can have a mapping as follows: For any event file with a .txt extension, the corresponding object name is FTPBG. The endpoint method name generated by the function selector in this case is emitFTPBG. You need to set this same name in the Native method property after you add the operation.

You can also have same rules multiple times. If more than one rule matches, the function selector will return the object name based on the first matching rule.

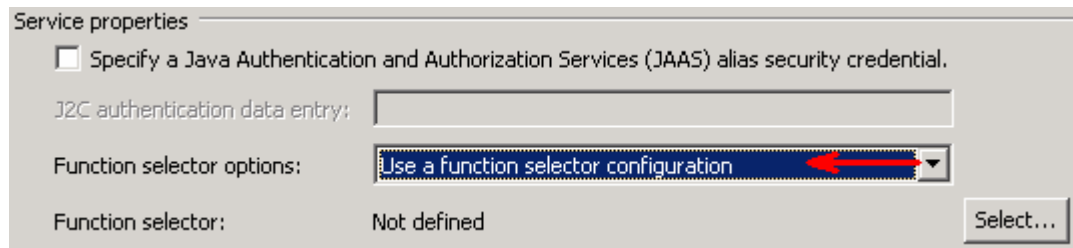
The function selector can be configured with multiple “rules”, each of which contains an object name, and a regular expression to match against the file name. In the next part of this lab, you will create three such rules.

- \_\_\_ 11. Under Service properties, for Function selector options, select **Use a function selector configuration** from the drop down menu

---

**Note:** If you select **Use default function selector ‘FilenameFunctionSelector’** option for the Function selector, the adapter automatically creates a function selector with FilenameFunctionSelector as the class name. But, for this lab, you need to define three rules at the function selector and so you need to use the **‘Use a function selector configuration’** option.

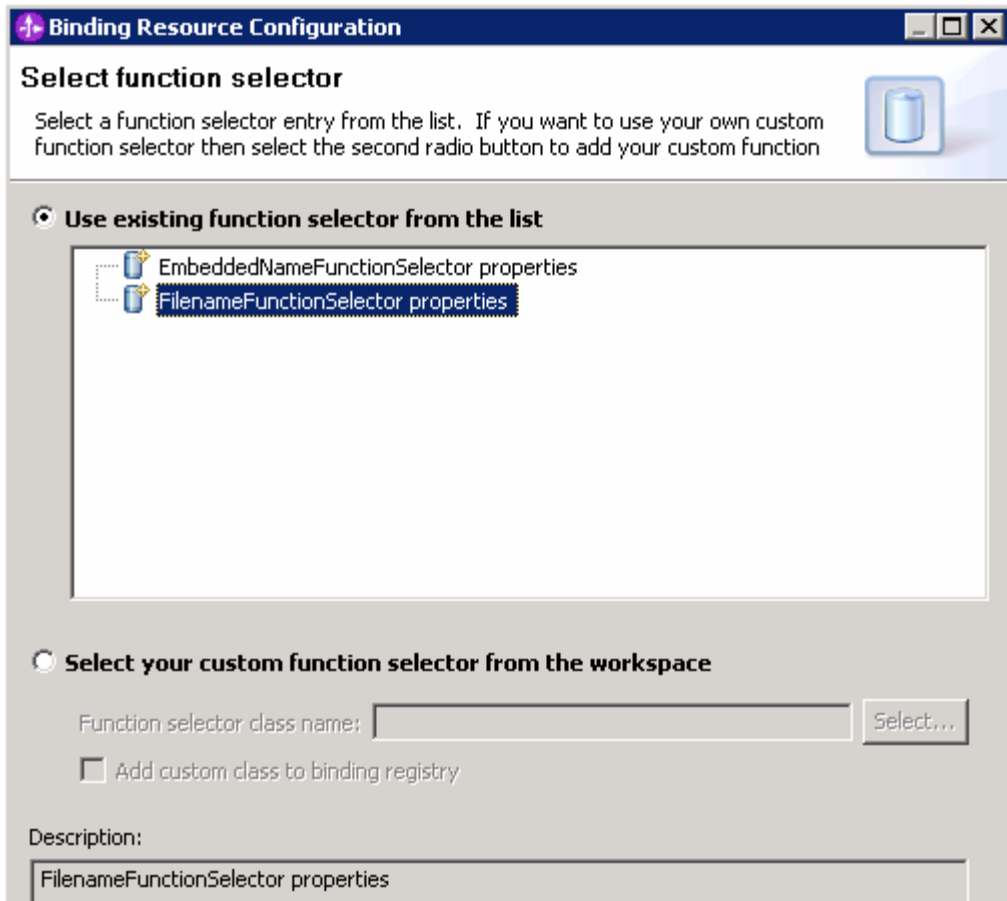
---



- \_\_\_ a. Click **Select** next to Function selector. The Binding Resource Configuration window is opened

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_ b. Under 'Use existing function selector from the list', select **FilenameFunctionSelector properties**



- \_\_ c. Click **Next**

- \_\_\_ 12. **Define Function selector rules:** In this step, you will define three different rules and object names associated with those rules.

When the adapter gets an event file, the function selector looks at the file type (ex: .txt, .tmp) and if the type matches with any of the rules defined in this table, it will associate the corresponding Object name of that rule to that event file.

Object Name	Rule	Operation Name	EIS function name = emit + Object Name
FTPBG	.*txt	emitFTPFileBG	emitFTPBG
FTPTType1	.*txt1	emitFTPFile1	emitFTPTType1
FTPTType2	.*txt2	emitFTPFile2	emitFTPTType2

**Note:** Make a note of the Object name you specify here as you are going to use this in the later part of the lab while specifying the EIS function name.

- \_\_ a. Click **Add...** next to the Function selector rules table

IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

\_\_ b. From the Add/Edit pop-up window, enter these:

1) Object name: **FTPBG**

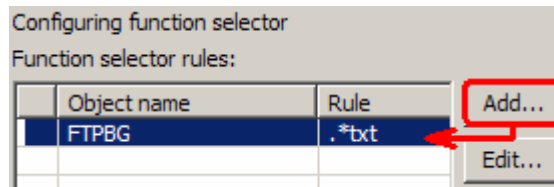
2) Rule: **\*.txt**

A screenshot of a dialog box with two input fields. The first field is labeled 'Object name:' and contains the text 'FTPBG'. The second field is labeled 'Rule:' and contains the text '\*.txt'.

**Note:** You have now created a rule for the files of type **.txt** and an Object name, **FTPBG**, corresponding to those file type

3) Click **Finish** from Add/Edit window

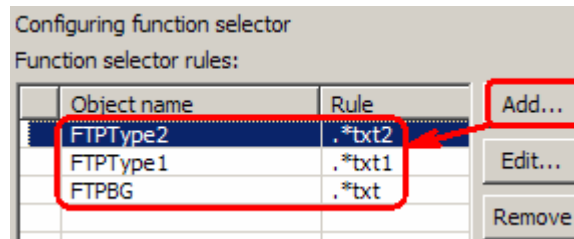
\_\_ c. The above created rule is populated under Function selector rules:



\_\_ d. Repeat Steps 9.a and 9.b to add these function selector rules:

Object name	Rule
FTPType1	*.txt1
FTPType2	*.txt2

\_\_ e. You should see these three rules:

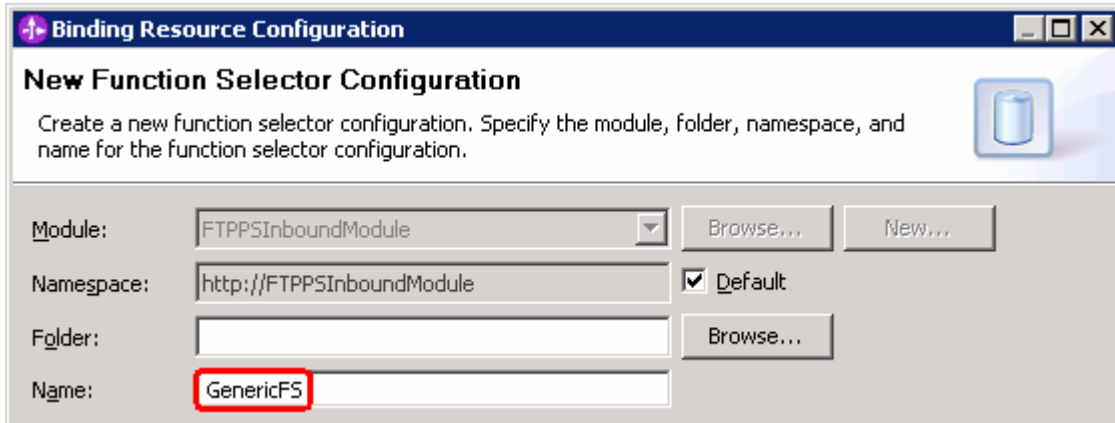


\_\_ f. Click **Next**

\_\_ g. Ensure that the selected module is **FTPPSInboundModule**

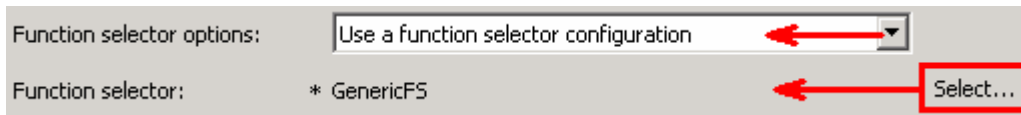
IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

\_\_ h. For **Name**, enter any string. For Ex: **GenericFS**



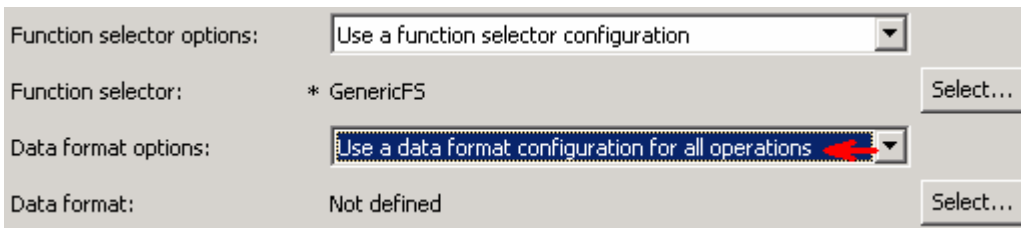
\_\_ i. Click **Finish**

\_\_\_ 13. You will now be back to External Service window and the function selector created in the above steps is populated:



\_\_\_ 14. You can define data binding in two places - service level (current screen of External Service wizard) or later at the method level (Operations screen of the External Service wizard). In this lab, you will define data binding at the service level (from this screen)

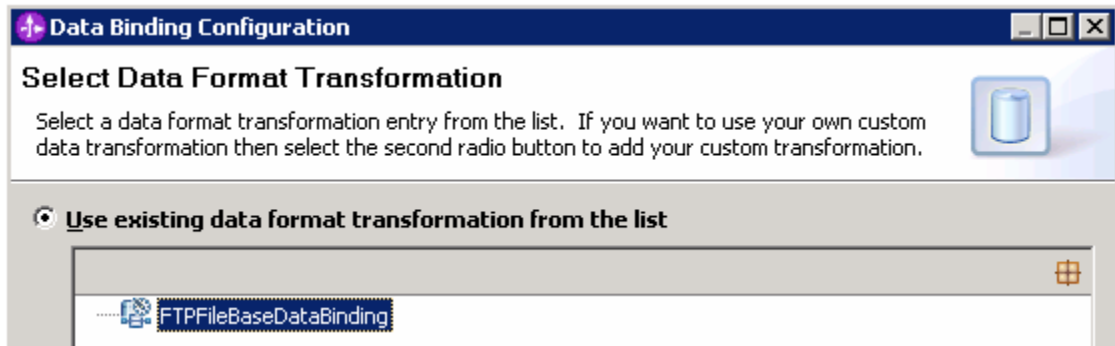
\_\_ a. From the dropdown menu next to Data format options, select '**Use a data binding configuration for all operations**'



\_\_ b. Click **Select...** next to **Data format**. A Binding Resource Configuration window is opened

### IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_ c. Select the radio button for **'Use existing data format transformation from the list'** and then select **FTPFileBaseDataBinding**



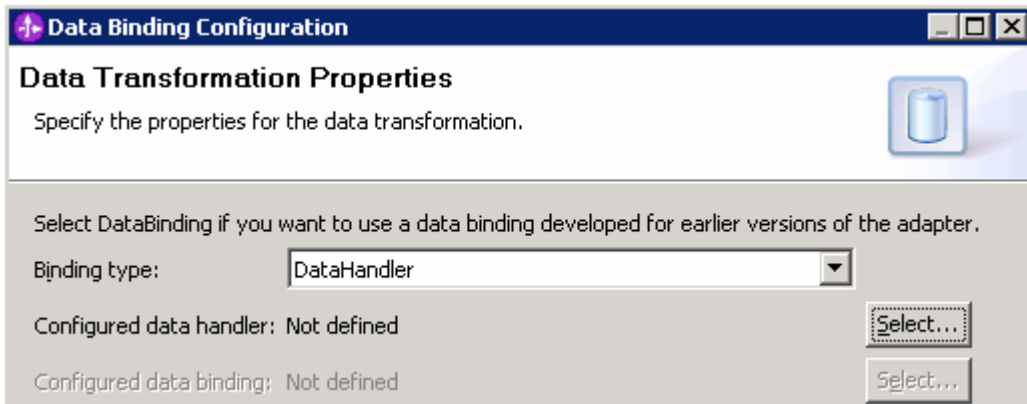
- \_\_ d. Click **Next**

---

**Note: Data Handler Configuration:** Since you are doing the pass through scenario, you do not need to configure any data handler.

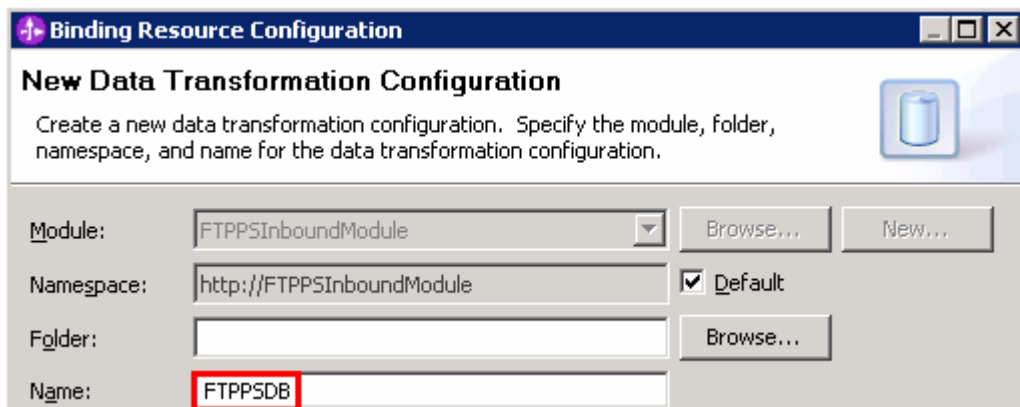
---

- \_\_ e. Click **Next** from the Data Transformation Properties screen



- \_\_ f. Note that the selected module is **FTPPSInboundModule**

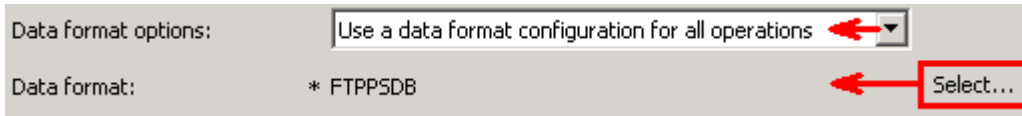
- 1) For the **Name**, enter **FTPPSDB**



- 2) Click **Finish**

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

\_\_\_ g. Now the **FTPPSDB** should be displayed for Data format



\_\_\_ 15. Check the box next to **Change logging properties for wizard** to view the output location of the log file and the logging level. You can change the logging level using the drop down menu.

\_\_\_ a. Click **Next**

Following screen is the Operations screen where you can define all your operations.

From V6.1, you can select from three different Data types for any operation:

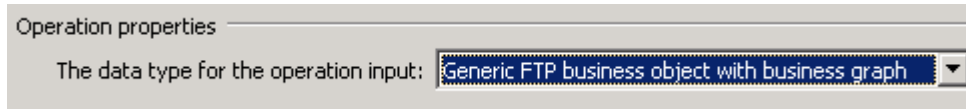
- User defined type
- Generic FTP business object
- Generic FTP business object with business graph

The pass through scenario of this lab demonstrates the creation of three operations (for the three rules you have defined in the previous steps) using the Generic FTPFile business object and Generic FTPFile business object with business graph data types.

**Add emitFTPFileBG Operation:**

\_\_\_ 16. From the Operations screen, click **Add...**

\_\_\_ a. Add Operation window is opened. Select **Generic FTP business object with business graph** for the Data type and click **Next**



You are now back to Operation window and because you have chosen the data type with business graph, the Input type is populated as **FTPFileBG**

\_\_\_ b. For Operation name, enter any name, for Ex: **emitFTPFileBG**

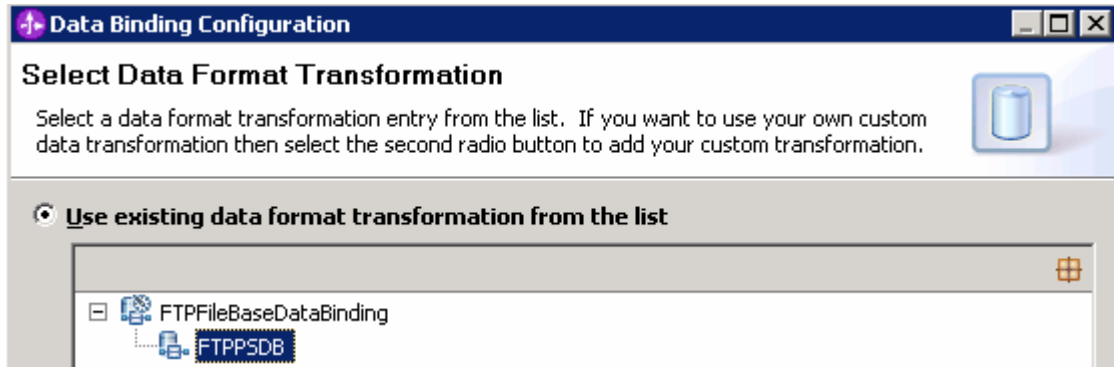
Define Data format:

\_\_\_ c. For **Data format options**, select **Use a data binding configuration** from the drop down list

\_\_\_ d. Click **Select...** next to **Data format**. A Binding Resource Configuration window is opened

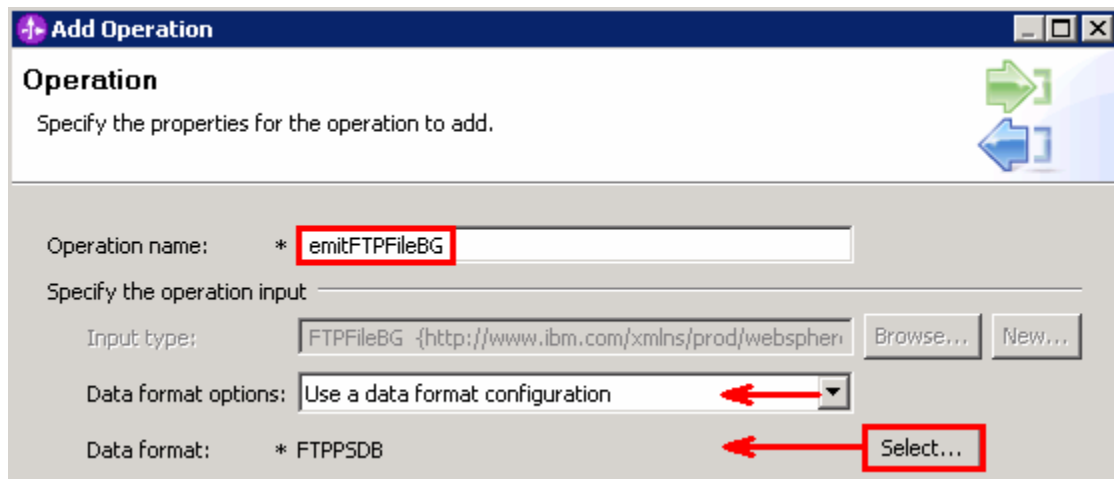
IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ e. Ensure that the radio button for 'Use existing data format transformation from the list' and then select **FTPFileBaseDataBinding > FTTPSDB**

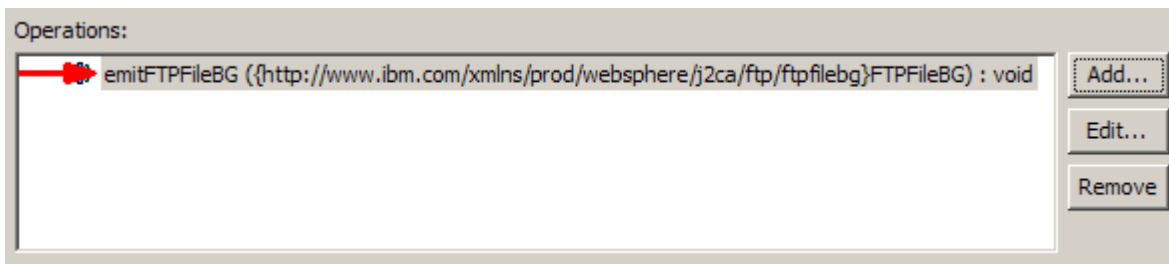


- 1) Click **Finish**

- \_\_\_ f. Now the **FTTPSDB** is displayed for **Data format** in the Add Operation window:



- \_\_\_ g. Click **Finish**. The above defined operation, **emitFTPFileBG**, is populated under Operations list



**Add emitFTPFile1 Operation:**

- \_\_\_ 17. From the Operations screen, click **Add...**

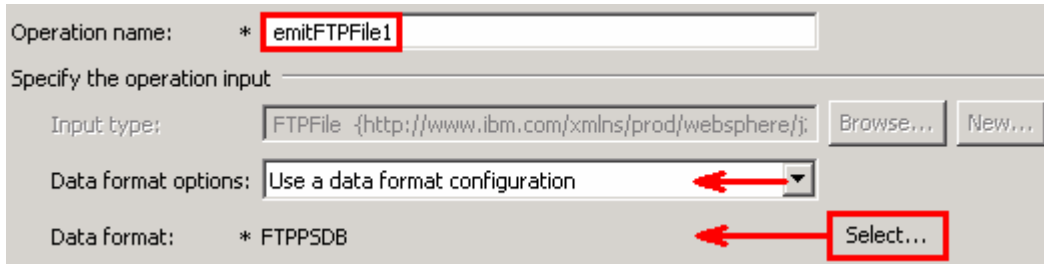
- \_\_\_ a. Select **Generic FTP business object** for the Data type and click **Next**



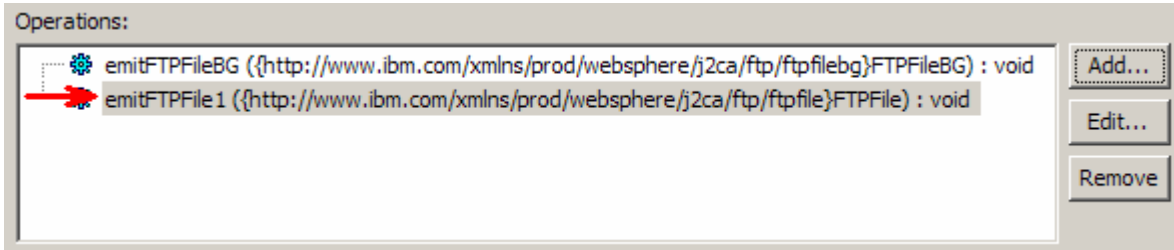
### IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

You are now back to Operation window and because you have chosen the data type with no business graph, the Input type is populated as **FTPFile**

- \_\_\_ b. For Operation name, enter **emitFTPFile1**
- \_\_\_ c. Follow the instructions used to define data format for emitFlatFileBG in the previous step, and select the same **FTPPSDB** as Data format
- \_\_\_ d. Now the **FTPPSDB** is displayed for Data format in the Add Operation window:



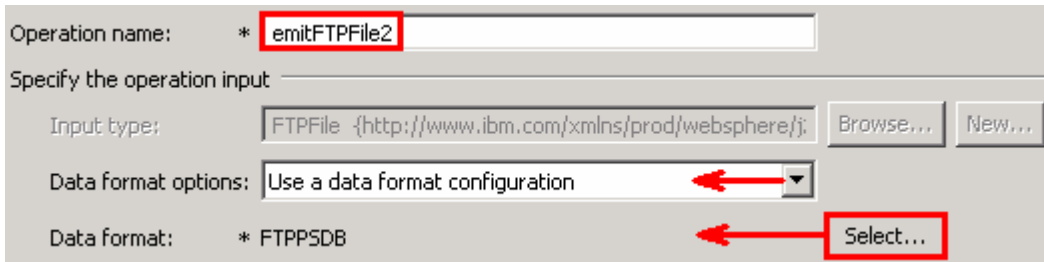
- \_\_\_ e. Click **Finish**. The above defined operations, **emitFTPFile1**, is populated under Operations list



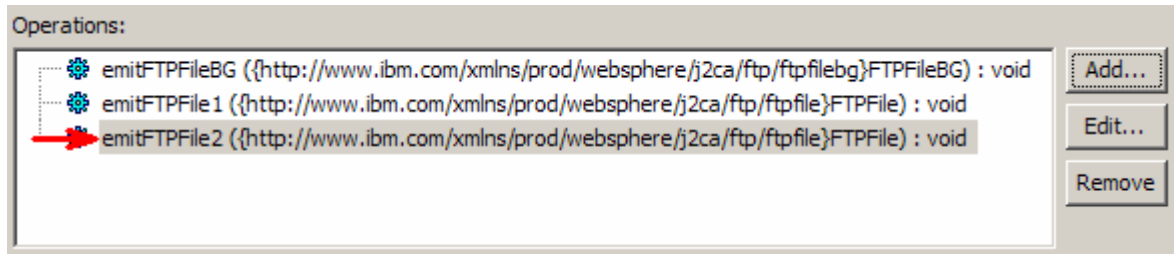
#### Add emitFTPFile2 Operation

- \_\_\_ 18. Repeat steps of **Add emitFTPFile1** to add one more operation, emitFTPFile2, with these changes:

- \_\_\_ a. Operation Name: **emitFTPFile2**



- \_\_\_ 19. Click **Finish**. The above defined operations, **emitFTPFile2**, is populated under Operations list



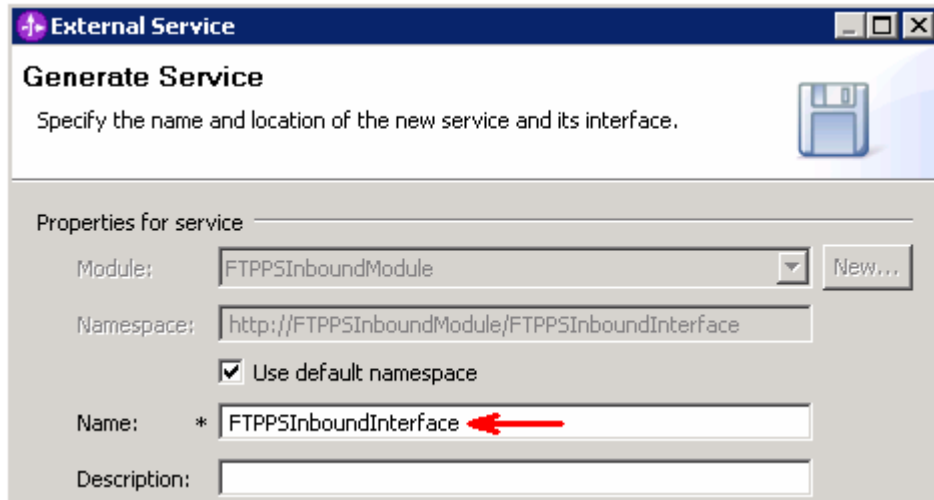


IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

\_\_\_ a. Click **Next** from Operations screen

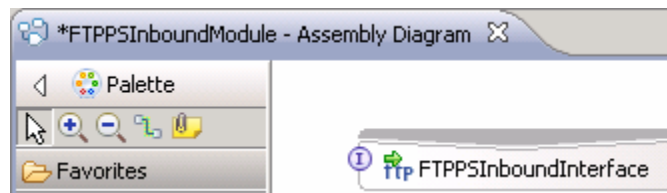
\_\_\_ 20. From Generate Artifacts screen:

\_\_\_ a. For **Name**, enter **FTPPSInboundInterface**



\_\_\_ b. Click **Finish**

\_\_\_ 21. The Assembly diagram for FTPPSInboundModule is opened with an Export component, FTPPSInboundInterface:



\_\_\_ a. Save (**Ctrl + S**) changes to your assembly diagram

**Note:** You need to manually change the Native method names.

The table below summarizes the rules, operations you have defined so far in this lab. Note the last column, which shows the Native method name for each of the rule and operation defined:

Object Name	Rule	Operation Name	Native method name = emit + Object Name
FTPBG	.*txt	emitFTPFileBG	<b>emitFTPBG</b>
FTPTYPE1	.*txt1	emitFTPFile1	<b>emitFTPTYPE1</b>
FTPTYPE2	.*txt2	emitFTPFile2	<b>emitFTPTYPE2</b>

\_\_\_ 22. Change the Native method binding name

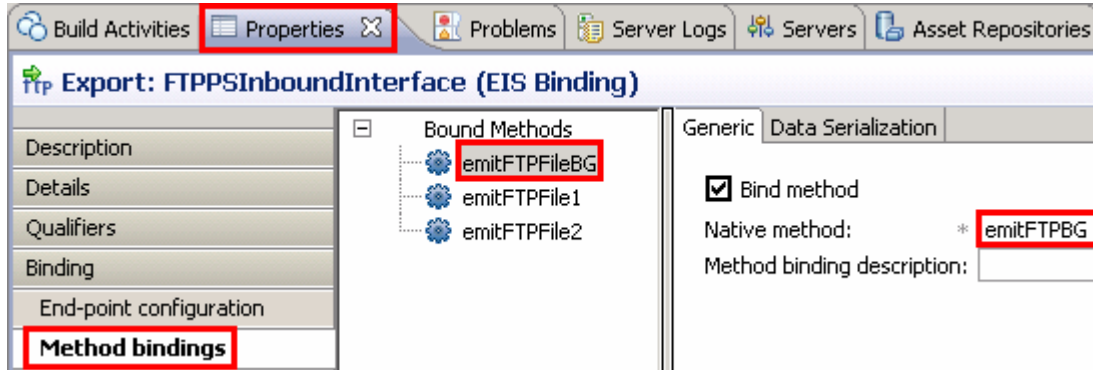
\_\_\_ a. Ensure that the FTPPSInboundInterface is selected from the Assembly diagram

\_\_\_ b. From the bottom panel, select **Properties > Binding > Method bindings**

\_\_\_ c. From the Bound Methods list, click **emitFTPFileBG**

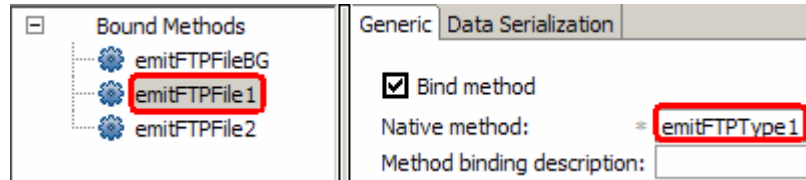
IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

\_\_\_ d. From the right side, ensure that the Generic tab is selected, and then change the **Native method** to **emitFTPBG**

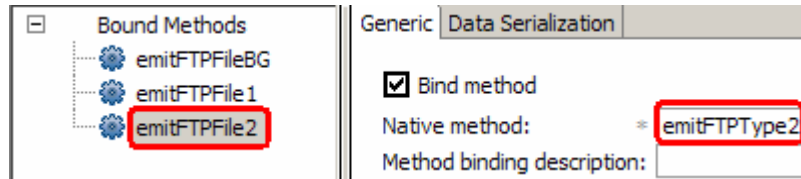


\_\_\_ e. From the Bound Methods list, click **emitFTPFile1**

\_\_\_ f. Similarly, for **emitFTPFile1**, change the **Native method** to **emitFTPTYPE1**



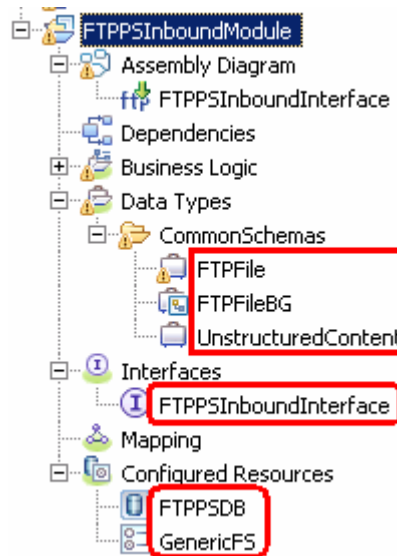
\_\_\_ g. Similarly, for **emitFTPFile2**, change the **Native method** to **emitFTPTYPE2**



\_\_\_ h. Save (**Ctrl + S**) your changes

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ 23. Review the FTPPSInboundModule: The generated **Data Types**, **Interface**, and Function selector (**GenericFS**) and Data binding (**FTPSSDB**) under Configured Resources can be found inside FTPPSInboundModule



You can open each of these generated artifacts and business objects and review the properties inside.

Review the created methods inside the interface:

- \_\_\_ a. From the Business Integration view, expand FTPPSInboundModule > Interfaces and then double-click **FTPSSInboundInterface** to open it
- \_\_\_ b. You should see these three operations:

Name	Type
emitFTPFileBG	
Input(s)	emitFTPFileBGInput
emitFTPFile1	
Input(s)	emitFTPFile1Input
emitFTPFile2	
Input(s)	emitFTPFile2Input
	FTPFile

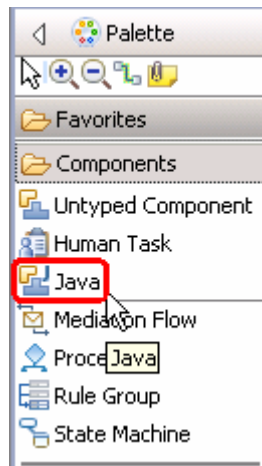
- \_\_\_ c. Close the interface, FTPPSInboundInterface


## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

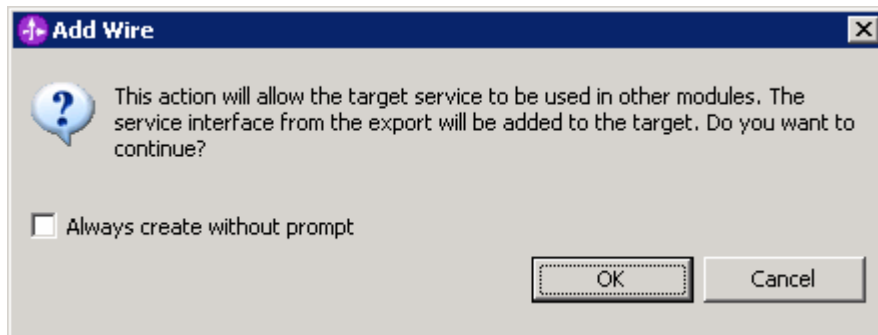
## 2.2. Add Java component

In this part of the lab, you will add a Java component and then wire the component to the existing Export interface. The Java component is your endpoint.

- \_\_\_ 1. Open the assembly diagram for FTPPSInboundModule (if it is already not open)
  - \_\_\_ a. From the business integration view, expand **FTPPSInboundModule** and double click **Assembly diagram**
- \_\_\_ 2. Drop a Java component to onto the assembly diagram
  - \_\_\_ a. From the **Palette**, click **Components** to expand it



- \_\_\_ b. Click **Java** and then click the empty space of FTPPSInboundModule assembly diagram. This will place a new component, **Component1** on the assembly diagram.
- \_\_\_ 3. Wire the FTPPSInboundInterface to the Component1
  - \_\_\_ a. Select the **wire** () icon from the Palette
  - \_\_\_ b. Click **FTPPSInboundInterface** and then click **Component1** to wire them together
  - \_\_\_ c. Select **OK** for the Add Wire pop-up window:



- \_\_\_ d. From the top of the Palette, click the **Selection Tool** icon () to get back to the normal cursor mode

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_ e. Right-click the empty space of the Assembly diagram and select **Arrange Contents Automatically** from the pop-up menu

Your assembly diagram for FTPPSInboundModule will look like this:



- \_\_ f. Right-click **Component1** and select **Generate Implementation** from the pop-up menu
- \_\_ g. On the **Generate Implementation** panel, select **default package**, and click **OK**
- \_\_ h. **Component1Impl.java** is opened in Assembly editor. Scroll down to the method **emitFTPFileBG** that needs to be implemented and add this code under that method:

```
System.out.println("*****Inside emitFTPFileBG: type .txt
files*****");
```

- \_\_ i. Similarly add the print statement to **emitFTPFile1** method:

```
System.out.println("*****Inside emitFTPFile1:type .txt1
files*****");
```

- \_\_ j. And finally, add the print statement to **emitFTPFile2** method:

```
System.out.println("*****Inside emitFTPFile2:type .txt2
files*****");
```

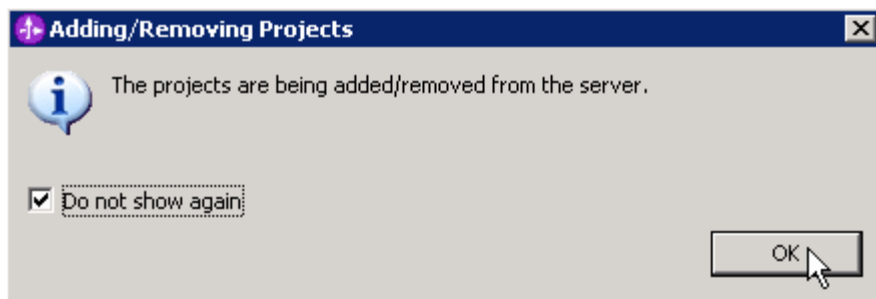
- \_\_ k. Save (**Ctrl + S**) and close Component1Impl.java
- \_\_ l. Save (**Ctrl + S**) and close Assembly diagram: FTPPSInboundModule

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

## 2.3. Test pass through scenario

In this part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application Inbound processing for the Pass through scenario.

- \_\_\_ 1. Add the project to the WebSphere Test Environment server
  - \_\_\_ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects...** from the pop-up menu
  - \_\_\_ b. From the Add and Remove Projects window, select **FTPPSInboundModuleApp** under Available projects panel and click **Add >**
  - \_\_\_ c. You will now see the **FTPPSInboundModuleApp** added to the **Configured projects**
  - \_\_\_ d. Click **Finish** and wait until the project is being published onto the server. The server is started in Debug mode if it is not already started before
  - \_\_\_ e. Click **OK** from Adding/Removing Projects pop-up. Optionally, you can select 'Do not show again' so that you are not required to do this next time when you start the test client

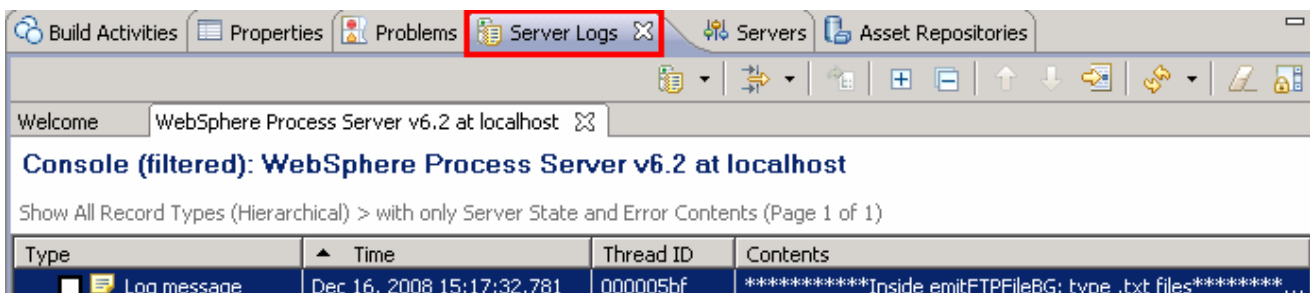


- \_\_\_ 2. Test the application by providing three different input files:

**Note:** For your convenience, three test files, **sample.txt**, **sample.txt1**, **sample.txt2** are placed in **<FTPFILES>**.

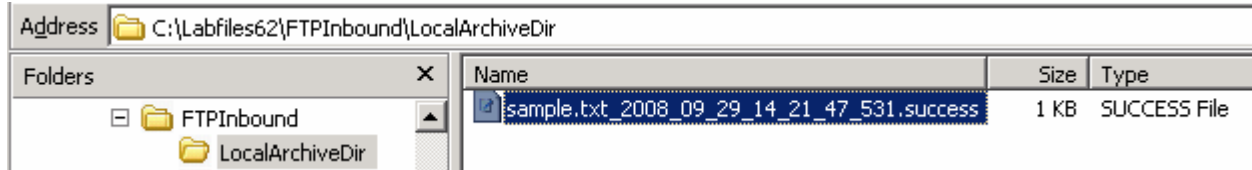
### Test scenario 1: .txt files

- \_\_\_ a. On the machine where the FTP Server is running, put **sample.txt** file in the **EventDir**. The adapter will poll the copied file from the event directory and will transfer it to the archive directory
- \_\_\_ b. Because you have placed a **.txt** file, it will pass through the **emitFTPFileBG** method and you should see this message in your **Server Logs** view (or SystemOut.log):



IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ c. Check the <LOCAL\_EVENT\_DIR> on your local machine. The file is quickly moved from this directory to the local archive directory
- \_\_\_ d. Check the **ArchiveDir** of your FTP server which should contain the same file name appended with year, month, date, system time, and processed as you have given
- \_\_\_ e. Check the <LOCAL\_ARCHIVE\_DIR> subdirectory which should contain an archive of the event file, with the same file name appended with year, month, date, system time, and success

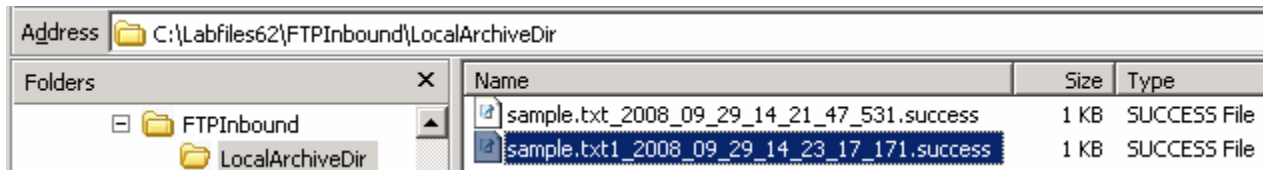


**Test scenario 2: .txt1 files:**

- \_\_\_ f. On the machine where the FTP Server is running, put **sample.txt1** file in the **EventDir**. The adapter will poll the copied file from the event directory and will transfer it to the archive directory
- \_\_\_ g. Because you have placed a **.txt1** file, it will pass through the **emitFTPFile1** method and you should see this message in your **Server Logs** view (or SystemOut.log):

Type	Time	Thread ID	Contents
Log message	Dec 16, 2008 15:17:32.781	000005bf	*****Inside emitFTPFileBG: type .txt files***** ...
Log message	Dec 16, 2008 15:18:32.781	000005bf	*****Inside emitFTPFileBG: type .txt1 files***** ...

- \_\_\_ h. Check the <LOCAL\_EVENT\_DIR> on your local machine. The file is quickly moved from this directory to the local archive directory
- \_\_\_ i. Check the **ArchiveDir** of your FTP server which should contain the same file name appended with year, month, date, system time, and processed as you have given
- \_\_\_ j. Check the <LOCAL\_ARCHIVE\_DIR> subdirectory which should contain an archive of the event file, with the same file name appended with year, month, date, system time, and success



**Test scenario: .txt2 files:**

- \_\_\_ k. On the machine where the FTP Server is running, put **sample.txt2** file in the **EventDir**. The adapter will poll the copied file from the event directory and will transfer it to the archive directory
- \_\_\_ l. Because you have placed a **.txt2** file, it will pass through the **emitFTPType2** method and you should see this message in your **Server Logs** view (or SystemOut.log):

Type	Time	Thread ID	Contents
Log message	Dec 16, 2008 15:17:32.781	000005bf	*****Inside emitFTPFileBG: type .txt files***** ...
Log message	Dec 16, 2008 15:18:32.781	000005bf	*****Inside emitFTPFileBG: type .txt1 files***** ...
Log message	Dec 16, 2008 15:19:33.531	000005bf	*****Inside emitFTPFileBG: type .txt2 files***** ...

- \_\_\_ m. Check the <LOCAL\_EVENT\_DIR> on your local machine. The file is quickly moved from this directory to the local archive directory

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ n. Check the **ArchiveDir** of your FTP server which should contain the same file name appended with year, month, date, system time, and processed as you have given:

```
ftp> pwd
257 "/home/wsbeta/ArchiveDir"
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
sample.txt1_2007_10_13_14_09_36_218
sample.txt2_2007_10_13_14_09_46_250
sample.txt_2007_10_13_14_09_26_218
226 directory send OK.
ftp: 110 bytes received in 0.00Seconds 110000.00Kbytes/sec.
ftp>
```

- \_\_\_ o. Check the <LOCAL\_ARCHIVE\_DIR> subdirectory which should contain an archive of the event file, with the same file name appended with year, month, date, system time, and success



- \_\_\_ 3. Restore the Sever Configuration
- \_\_\_ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects...** from the pop-up menu
- \_\_\_ b. Select **FTPPSInboundModuleApp** under Configured projects and click < **Remove**
- \_\_\_ c. Click **Finish** after you see the application moved to Available projects. Wait until the application is being unpublished



## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

### 2.4. Test pass through scenario with SplitBySize

In this last part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application Inbound processing for the pass through scenario with split by size.

- \_\_\_ 1. Configure/change the adapter properties from the Properties view:
  - \_\_\_ a. Change to Business Integration perspective if you are in a different perspective
    - 1) Select **Window > Open Perspective > Other...**
    - 2) From the Select Perspective window, select **Business Integration (default)** and click **OK**
  - \_\_\_ b. Expand **FTPPSInboundModule** and double-click **FTPPSInboundModule** to open it in Assembly Editor
  - \_\_\_ c. Click **FTPPSInboundInterface** from the Assembly Editor and select **Properties** tab from the bottom
  - \_\_\_ d. Select **Binding** under Properties and select **End-point configuration** under Binding itself and then select the **Connection** tab
  - \_\_\_ e. Now under **ActivationSpec Properties**, you can review the properties that were given during the external service wizard
  - \_\_\_ f. Scroll down to **Advanced Properties**
    - 1) File transfer type: select **ascii** from the drop down list
    - 2) Select the box for '**Split file content based on the size (bytes) or delimiter**'
    - 3) Split function class name: **com.ibm.j2ca.utils.filesplit.SplitBySize** (default)
    - 4) Specify criteria to split file content: **1000**

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

**Note:** The value, 1000, you entered for the field, Specify criteria to split the content, is the size of the input file in Bytes.

## Advanced connection configuration

Retrieve files with this pattern:	<input type="text" value="*.*"/>	
Sort event files:	<input type="text" value="no sort"/>	<input type="button" value="v"/>
Encoding used by FTP server:	<input type="text"/>	<input type="button" value="Select..."/>
File content encoding:	<input type="text"/>	<input type="button" value="Select..."/>
FTP server connection mode:	<input type="text" value="active"/>	<input type="button" value="v"/>
File transfer type:	<input type="text" value="ascii"/>	<input type="button" value="v"/>
Number of files to get at a time:	<input type="text" value="10"/>	
Number of poll periods between downloads:	<input type="text" value="5"/>	
<i>Populate the fully qualified class name of the custom parser that is used to parse the "ls -l" output. This is used only when the "ls -l" output deviates from standard output.</i>		
Custom parser class name:	<input type="text"/>	<input type="button" value="Browse..."/>
<input type="checkbox"/> Pass only file name and directory, not the content <input type="checkbox"/> Include business object delimiter in the file content <input checked="" type="checkbox"/> Split file content based on the size (bytes) or delimiter.		
Split function class name:	<input type="text" value="com.ibm.j2ca.utils.filesplit.SplitBySize"/>	<input type="button" value="Browse..."/>
Specify criteria to split file content:	<input type="text" value="1000"/>	

\_\_ g. Click Assembly diagram (or any where else so that the save button is enabled) and then save **(Ctrl +S)** your changes

\_\_\_ 2. Modify Java code:

\_\_ a. From the assembly diagram of **FTPPSInboundModule**, double-click **Component1**

\_\_ b. **Component1Impl.java** is opened in Assembly editor. Scroll down to the method **emitFTPFileBG** that needs to be implemented and add this code under that method:

```
System.out.println("*****Inside emitFTPFileBG: type .txt files*****");
DataObject FTPFile = emitFTPFileBGInput.getDataObject("FTPFile");
DataObject Unstructured = FTPFile.getDataObject("Content");
String astext = Unstructured.getString("AsText");
System.out.println("File content-----> "+astext);
```

**Note:** You can also copy the Java code from **<FTPFILES>\SplitBySizeJavaCode.txt**.

\_\_ c. Save **(Ctrl + S)** and close Component1Impl.java

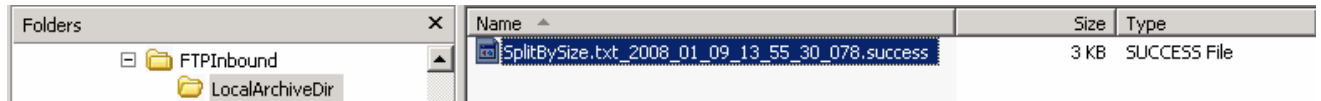
\_\_ d. Save **(Ctrl + S)** and close Assembly diagram: FTPPSInboundModule

\_\_\_ 3. Repeat Step 1 of part 2.3 to add the saved project **FTPPSInboundModuleApp** to the server



## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ f. Check the **<LOCAL\_ARCHIVE\_DIR>** subdirectory which should contain an archive of the event file, with the same file name appended with year, month, date, system time, and success:



The screenshot shows a file explorer window with a 'Folders' pane on the left and a main pane on the right. The 'Folders' pane shows a tree view with 'FTPInbound' and 'LocalArchiveDir'. The main pane shows a table of files with columns 'Name', 'Size', and 'Type'. A single file is listed: 'SplitBySize.txt\_2008\_01\_09\_13\_55\_30\_078.success' with a size of '3 KB' and type 'SUCCESS File'.

Name	Size	Type
SplitBySize.txt_2008_01_09_13_55_30_078.success	3 KB	SUCCESS File

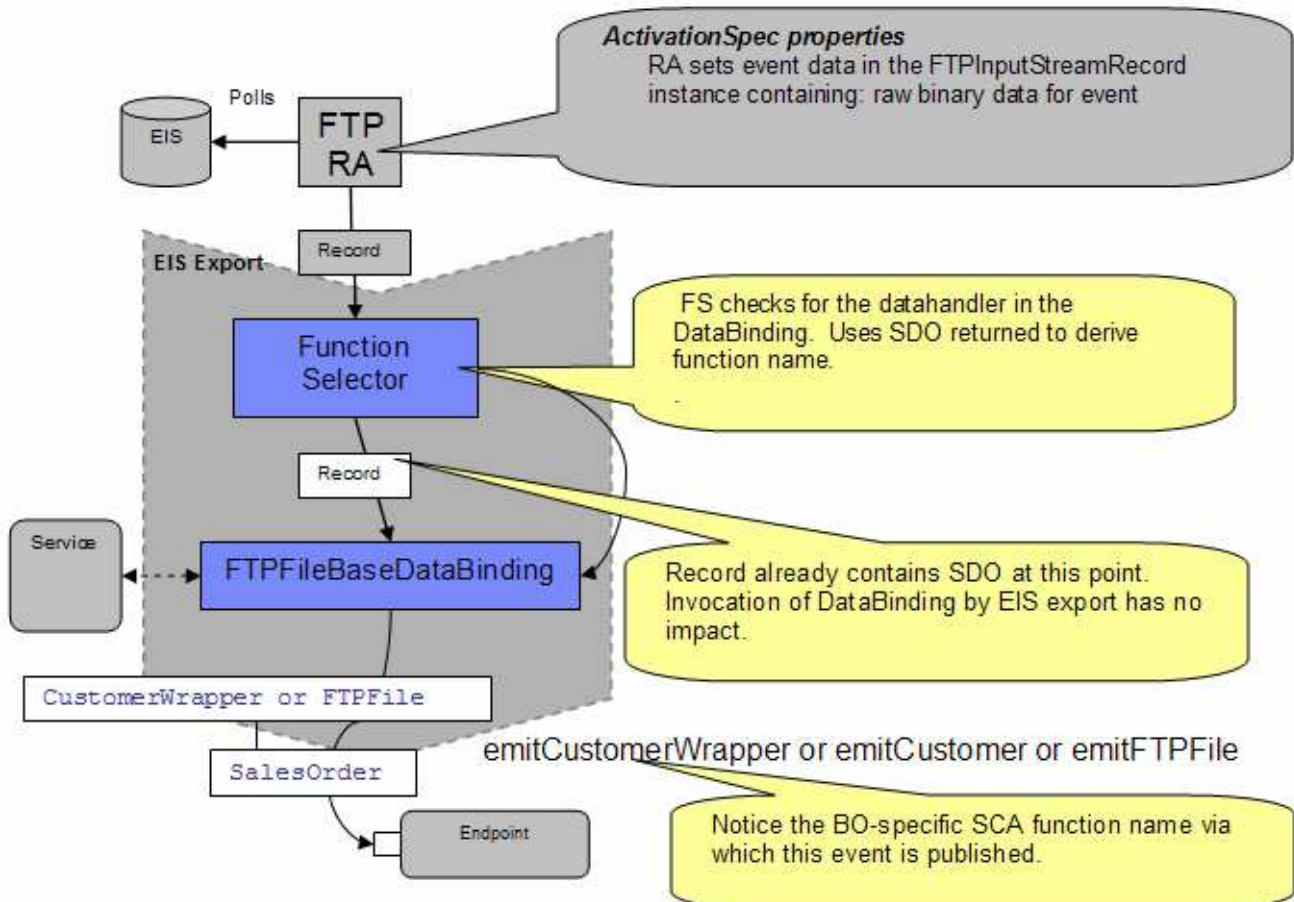
- \_\_\_ g. Restore the Sever Configuration: follow the instructions under **Step 3 of Part 2.3**

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

**Part 3: Content specific (non-pass through) scenario**

Of the two flows, you have just tested the pass through scenario that does not involve data transformation. In this part of the lab, you will configure the non-pass through scenario using the new External Service option from the WebSphere Integration Developer and then test the configuration with different cases.

Content specific flow for the inbound scenario:



- Event data is picked from the event file based on SplitCriteria (it contains the delimiter which actually separates the BO's in an event file), then converted into an input stream and then set on FTPInputStreamRecord.
- The data handler should be set with a valid value.
- Based on the wrapper generation selection, the protocol specific information such as event file name, directory name (LocalEventDirectory) which needs to be sent to the endpoint is also set in the FTPInputStreamRecord.
- The FTPInputStreamRecord is sent to the the Function Selector. The Function Selector will invoke that data binding and get back a content specific data object (Customer SDO). That is used to generate the function name and emit that. If content-specific data binding is not valid, the Data Transformation Framework will flag appropriate error.
- Optionally while sending protocol specific information like directoryName and fileName, you need the wrapper data object to hold this.

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- This content specific data object is set on the wrapper data object if wrapper is used. Protocol specific information is also set in the wrapper. The content specific data or wrapper is sent directly or wrapper is set in a BG and sent to the end point.

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

### 3.1. Configure content specific (non-pass through) scenario using external service wizard

In this part of the lab you will use this new external service feature to create and configure the function selector, **data handler**, and data binding and other required artifacts to test the inbound pass through scenario

- \_\_\_ 1. Create FTPCustomInboundModule
  - \_\_\_ a. From the Business Integration window, right-click and select **New > Module**
  - \_\_\_ b. From the New Module window, enter **FTPCustomInboundModule** for the Module Name
  - \_\_\_ c. Ensure that the box next to **Open module assembly diagram** is checked and then click **Finish**

You will now see a new module, FTPCustomInboundModule, created from your Business Integration window

- \_\_\_ 2. Import required business objects
  - \_\_\_ a. Expand FTPCustomInboundModule (if not already expanded), right-click **Data Types** and select **Import...** from the pop-up menu
  - \_\_\_ b. From the Import window, expand **General** and select **File System** and then click **Next**
  - \_\_\_ c. Enter From directory
    - 1) Click **Browse...** next to **From directory**
    - 2) From the Import from directory window, select **<FTPFILES>** and click **OK**

Now, you will see FTPFiles folder added on the left side, and all the xsds and other files under that folder on the right side.

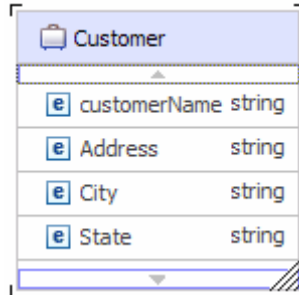
- \_\_\_ d. Select the box next to **Customer.xsd** and **Order.xsd**
- \_\_\_ e. Ensure that the **FTPCustomInboundModule** is selected for Into folder
- \_\_\_ f. Click **Finish** from the Import window

The Business Integration window is updated with the imported business objects.

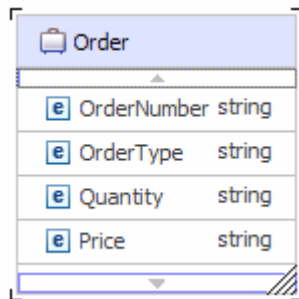
- \_\_\_ 3. Review imported business objects:
  - \_\_\_ a. Expand **FTPCustomInboundModule > Data Types** and you will now see two new data types **Customer** and **Order** under it

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

\_\_\_ b. Double-click **Customer** review the fields inside the object:



\_\_\_ 4. Now, double-click Order and review the fields inside the object:



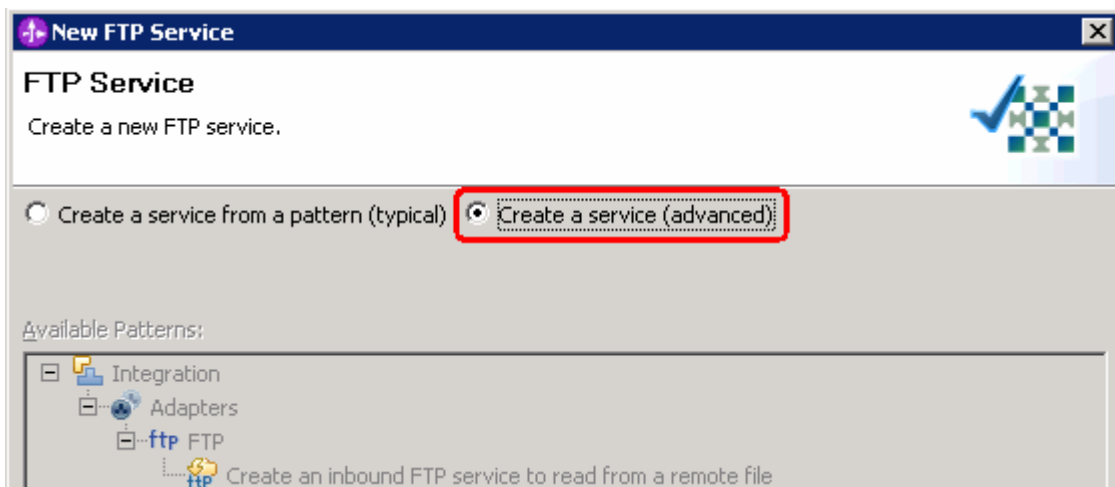
\_\_\_ 5. After reviewing, close the Customer business object from the Assembly editor

\_\_\_ 6. To start External Service from the Palette:

\_\_\_ a. From the **Palette** on the left side of Assembly Diagram, click **Inbound Adapters**:

\_\_\_ 7. Under Inbound Adapters, click the **FTP** and then click the empty canvas of the assembly diagram. The New FTP Service wizard is opened

\_\_\_ 8. From the FTP Service screen, select **Create a service (advanced)**



\_\_\_ a. Click **Next**

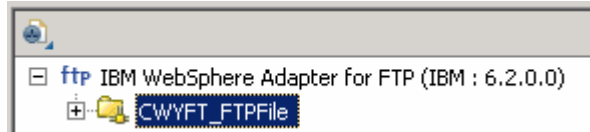


## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

**Note:** You can also start the External Service from the **File menu** option:

From the main menu, select **File > New > External Service**. This opens an External Service wizard that helps you obtain a service which establishes connectivity with other systems. Select **Adapters > FTP** and click **Next**

- \_\_\_ 9. On the Select an Adapter screen, expand **IBM WebSphere Adapter for FTP (IBM : 6.2.0.0)** and select **CWYFT\_FTPFile**



- \_\_\_ a. Click **Next**

**Note:** If you are using the **File menu** option to start the External Service wizard, you are asked to select the **Processing Direction** at this point. Select the radio button next to **Inbound** and click **Next** to proceed to the next step.

- \_\_\_ 10. Service Configuration Properties:

- \_\_\_ a. Deploy connector project: ensure that the default option **With module for use by single application** is selected

- \_\_\_ b. Enter these for FTP system connection information:

- 1) Host name: **<FTP\_Machine\_Name>** (or IP Address of the machine that has FTP Server),  
for Ex: wsbeta181.austin.ibm.com
- 2) Remote directory: **full path of the EventDir created in on the machine where FTP server is existing** (for Ex: /home/wsbeta/EventDir)

**Note:** This is the directory from where adapter gets the event files.

- 3) Local directory: **<LOCAL\_EVENT\_DIR>**
- 4) Protocol: **FTP - file transfer protocol** (default)
- 5) Port number: **21** (default)
- 6) **User name:** username using which you connect to your FTP server (for Ex: **root**)

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

7) **Password:** password for the above user to connect to your FTP server

\_\_ c. Click **Advanced >>** to see the hidden advanced properties that can be configured:

You can click each of the configuration and review the options available under it. For this lab, you will need only some of these properties.

\_\_ d. Event persistence configuration:

- 1) Ensure that the **Auto create event table** is **checked**
- 2) Event recovery table name: **FTPCustomTable**
- 3) Event recovery data source (JNDI) name: **jdbc/FTP**

---

**Note:** This represents the JNDI name of the Data Source used by Event Persistence to get the JDBC database connection. The Data Source must be created in the WebSphere Process Server. You should enter the data source JNDI name that you created in Step 3 of Part 1.

---

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

**Note:** The Event recovery data source (JNDI) name is **not mandatory** from V6.1. Now, the adapter can use **in-memory representation** of event table to store all the necessary information. Adapter uses this feature when event database information is not configured during inbound event polling. This feature will not support the capability of handling “Ensure once-only event delivery”.

\_\_ e. FTP archiving configuration:

- 1) Local Archive directory: click **Browse...** and select **<LOCAL\_ARCHIVE\_DIR>**
- 2) Remote archive directory: **full path of the ArchiveDir created in on the machine where FTP server is existing** (for Ex: /home/wsbeta/ArchiveDir)

FTP archiving configuration

Specify local archive directory to enable archiving on the local system, specify remote archive directory to enable archiving on the remote system.

Local archive directory: C:\Labfiles62\FTPInbound\LocalArchiveDir **Browse...**

File extension for local archive: original

Success file extension for local archive: success

Failure file extension for local archive: fail

Remote archive directory: /home/wsbeta/ArchiveDir

File extension for remote archive:

- \_\_\_ 11. For this lab, you are not going to use the J2C authentication. So, **uncheck** the box next to **Specify a Java Authentication and Authorization Services (JAAS) alias security credentials**.

Service properties

Specify a Java Authentication and Authorization Services (JAAS) alias security credential.

J2C authentication data entry:

**Function Selector Configuration:** Of the two function selectors that are supported by the FTP adapter, FilenameFunctionSelector was used for pass through scenario. The **EmbeddedNameFunctionSelector** is used in case of content-specific business objects, where the object name is embedded in the event file. The EmbeddedNameFunctionSelector will return the function name based on the content data. For example, if the content-specific business object is CustomerWrapperBG, the function returned by the function selector is emitCustomer. This function selector should be configured with a data handler. The data binding should be the adapter-specific WrapperDataBinding. The data binding should be configured to use the same data handler configured with the function selector.

- \_\_\_ 12. Under Service properties, for Function selector options, select **Use a function selector configuration** from the drop down menu

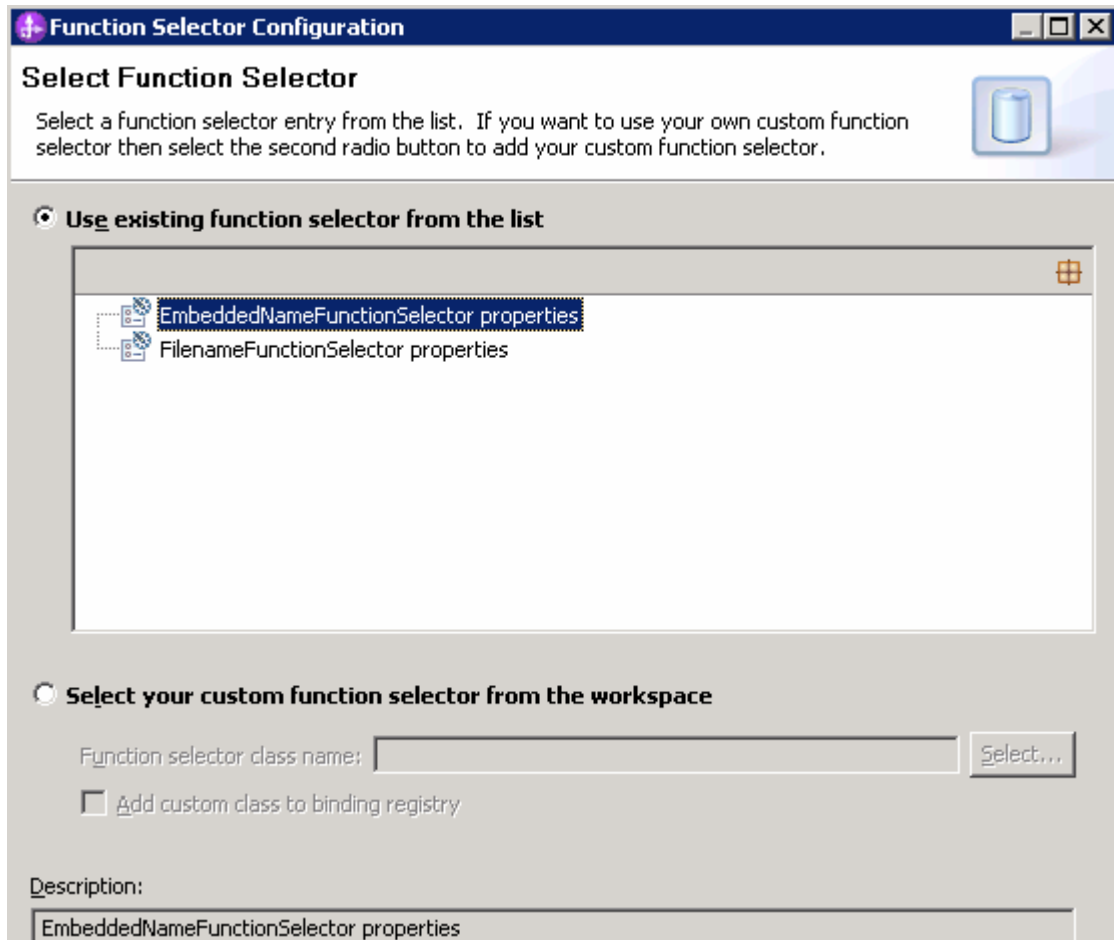
Function selector options: Use a function selector configuration

Function selector: \* Not defined **Select...**

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

**Note:** If you select **Use default function selector 'FilenameFunctionSelector'** option for the Function selector, the adapter automatically creates a function selector with `FilenameFunctionSelector` as the class name. For non-pass through scenario, you need to define a function selector that uses **EmbeddedNameFunctionSelector**.

- \_\_\_ a. Click **Select** next to Function selector. The Binding Resource Configuration window is opened
- \_\_\_ b. Under '**Use existing function selector from the list**', select **FilenameFunctionSelector properties**



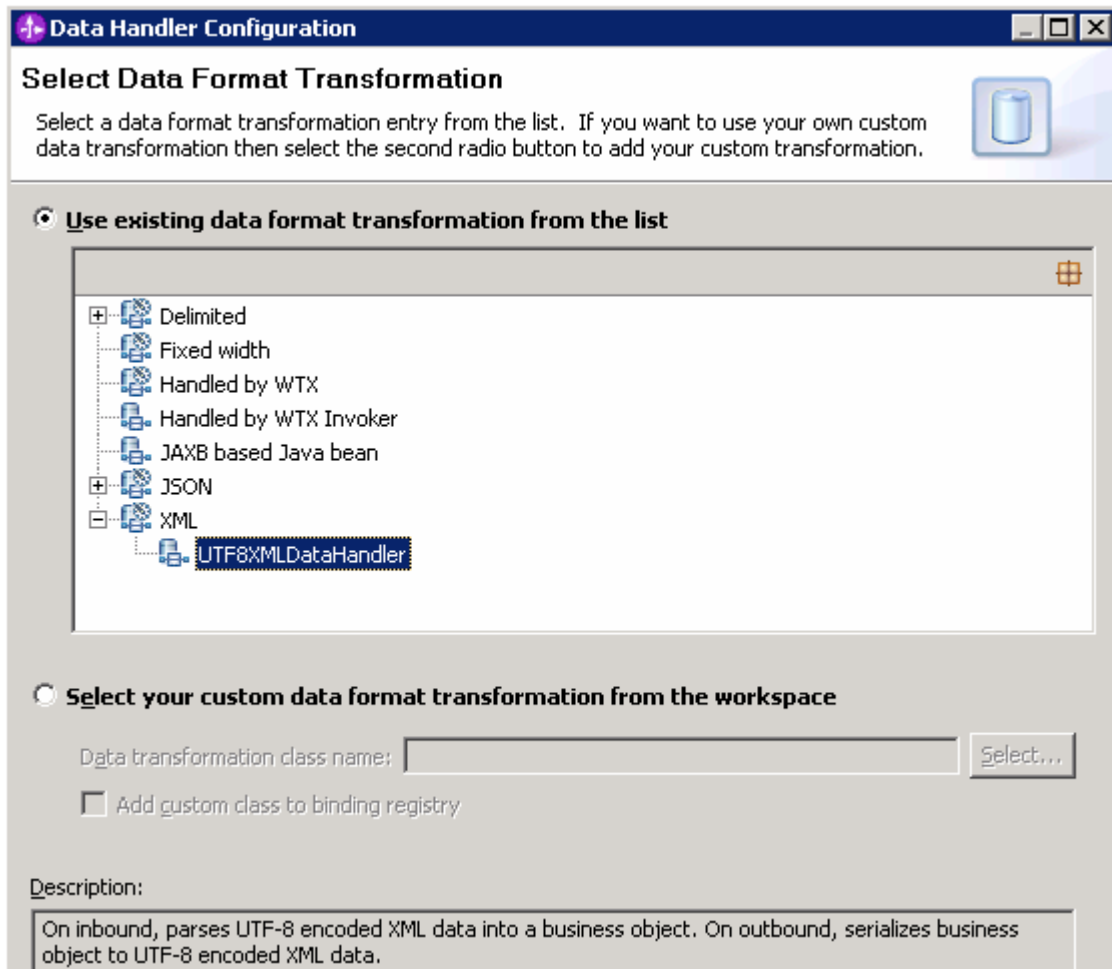
- \_\_\_ c. Click **Next**

**Data handler configuration:** The screen shown below is Function Selector Properties screen where you will define the data handler and encoding used in inbound processing

- \_\_\_ d. Click **Select...** next to Data handler configuration name. A Binding Resource Configuration window is opened for you to define the data handler
- \_\_\_ e. Under 'Use existing data format transformation from the list', select **XML > UTF8XMLDataHandler**

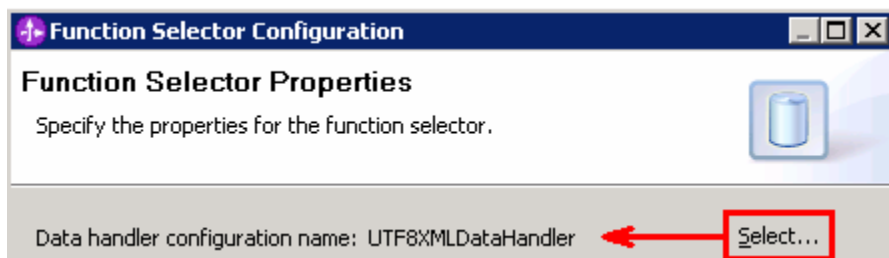
## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

**Note:** UTF8XMLDataHandler listed under XML is the predefined data handler with UTF-8 as the encoding. You can also select XML and then select the encoding of your choice in the next screen to define a data handler of your choice.



\_\_\_ f. Click **Finish**

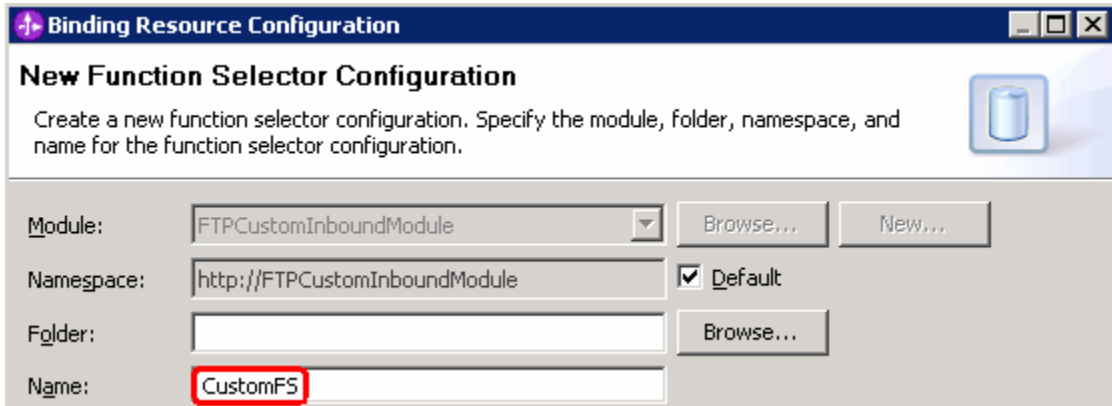
\_\_\_ g. From the next screen, the Data handler configuration name, **UTF8XMLDataHandler** is displayed which was selected in the previous steps



\_\_\_ h. Click **Next**

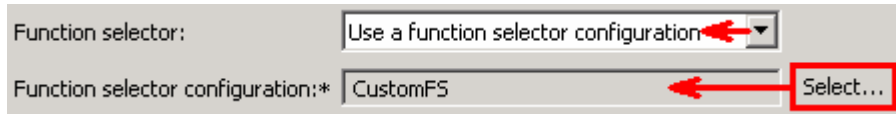
IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

\_\_ i. From the New Function Selector Configuration screen, enter **CustomFS** for **Name**



\_\_ j. Click **Finish**

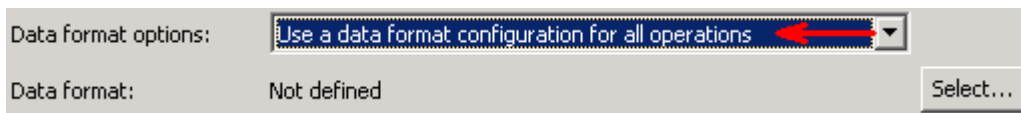
You are now done with configuring the Function Selector, CustomFS and, in that process, configured a data handler, CustomDH inside the function selector configuration wizard. You are back to the External Service wizard and the above configured function selector, CustomFS is displayed for Function Selector Configuration field:



**Data binding configuration:**

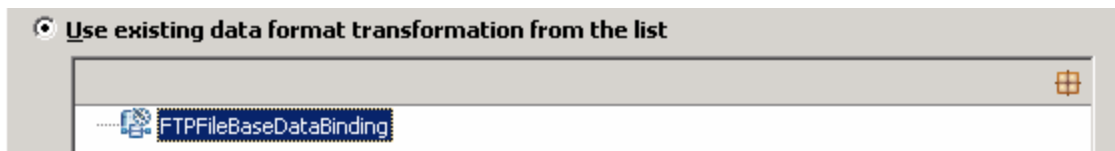
\_\_\_ 13. You can define data binding in two places - service level (current screen of External Service wizard) or later at the method level (Operations screen of the External Service wizard). In this lab, you will define data binding at the service level (from this screen)

\_\_ a. From the dropdown menu next to Data format options, select '**Use a data binding configuration for all operations**'



\_\_ b. Click **Select...** next to **Data format**. A Binding Resource Configuration window is opened

\_\_ c. Select the radio button for '**Use existing data format transformation from the list**' and then select **FTPFileBaseDataBinding**



\_\_ d. Click **Next**

\_\_ e. Select the **UTF8XMLDataHandler** for data handler:

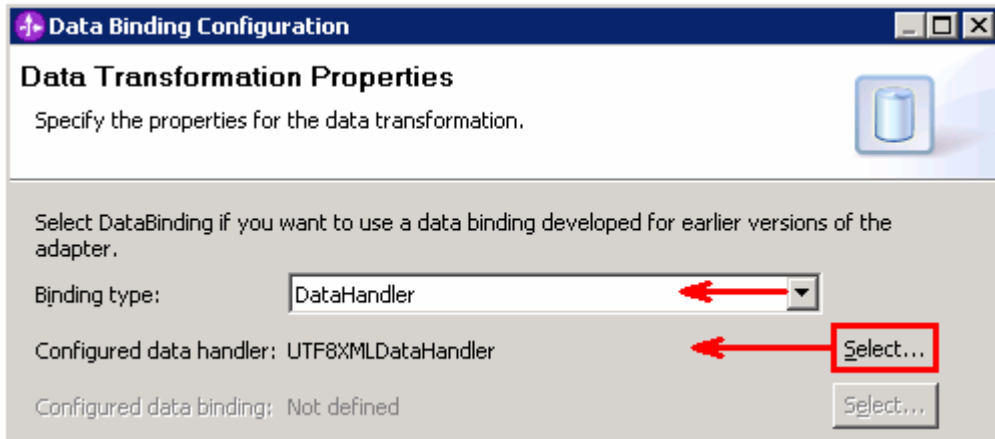
1) Click **Select** next to 'Configured data handler'

### IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

2) From the Binding Resource Configuration window, select **XML > UTF8XMLDataHandler** listed under 'Use existing data format transformation from the list'

\_\_\_ f. Click **Finish**

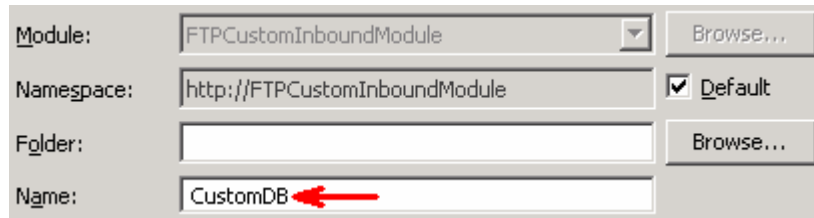
\_\_\_ g. Back to Data Transformation Properties and the selected data handler '**UTF8XMLDataHandler**' is displayed here:



\_\_\_ h. Click **Next**

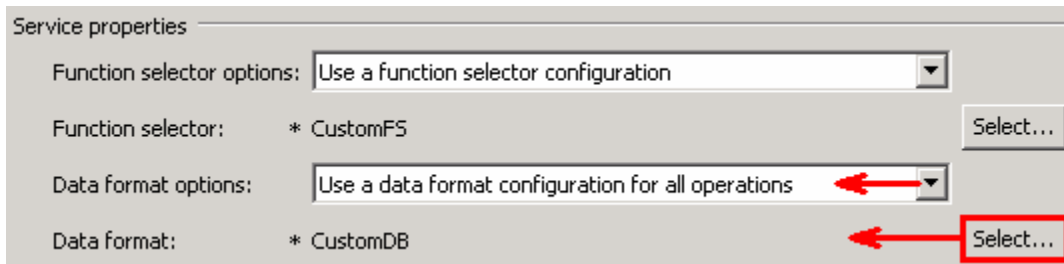
\_\_\_ i. Note that the selected module is **FTPCustomInboundModule**

1) For the **Name**, enter **CustomDB**



2) Click **Finish**

\_\_\_ j. Now the **CustomDB** should be displayed for Data binding configuration



\_\_\_ 14. Check the box next to **Change logging properties for wizard** to view the output location of the log file and the logging level and click **Next**

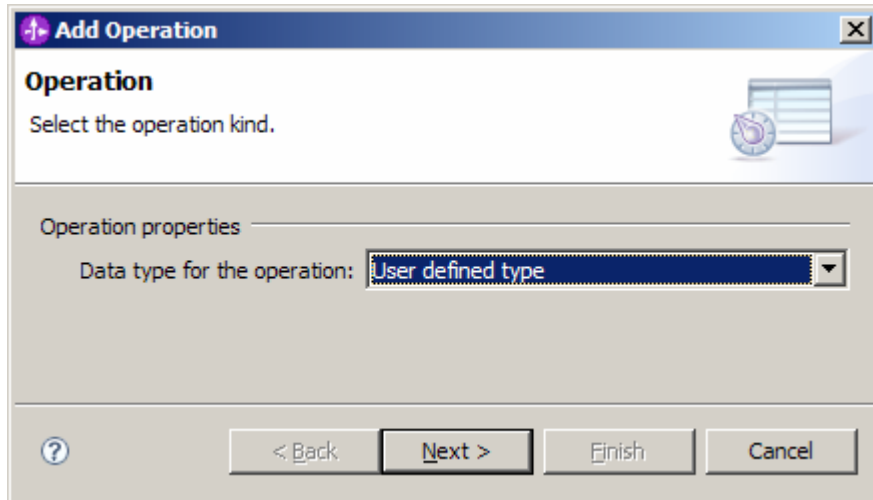
## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

**Define emitCustomerFile operation:**

\_\_\_ 15. From the Operations screen, click **Add...**

Add Operation window is opened

\_\_\_ a. Select **User defined type** for the Data type and click **Next**

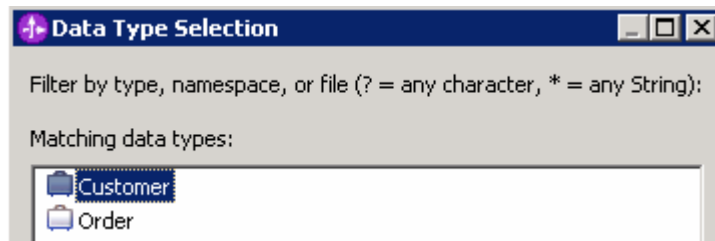


You are now back to Operation window and because you chose the User defined data type, the Input type is blank

\_\_\_ b. For Operation name, enter **emitCustomerFile**

\_\_\_ c. Define Input type:

- 1) Click **New...** next to **Input type** to open a New Business Object window
- 2) From this window, ensure that the Module selected is **FTPCustomInboundModule** and click **Next**
- 3) Click **Browse...** next to **Data type**
- 4) From the Data Type Selection window, select **Customer** under Matching data types:

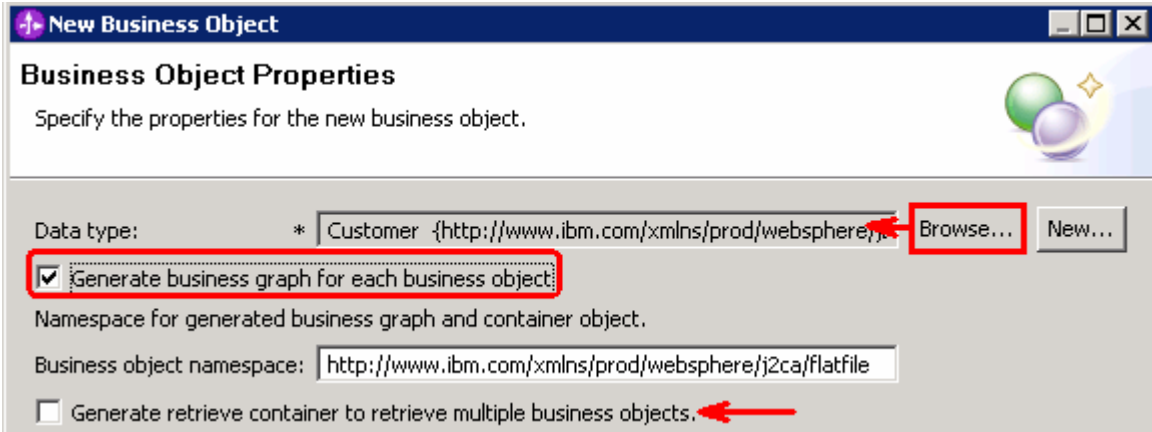


- 5) Click **OK**
- 6) From the Type Wizard, **check** the box next to **Generate business graph for each business object**
- 7) **Do not** check the box for '**Generate retrieve container to retrieve multiple business objects**'



IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

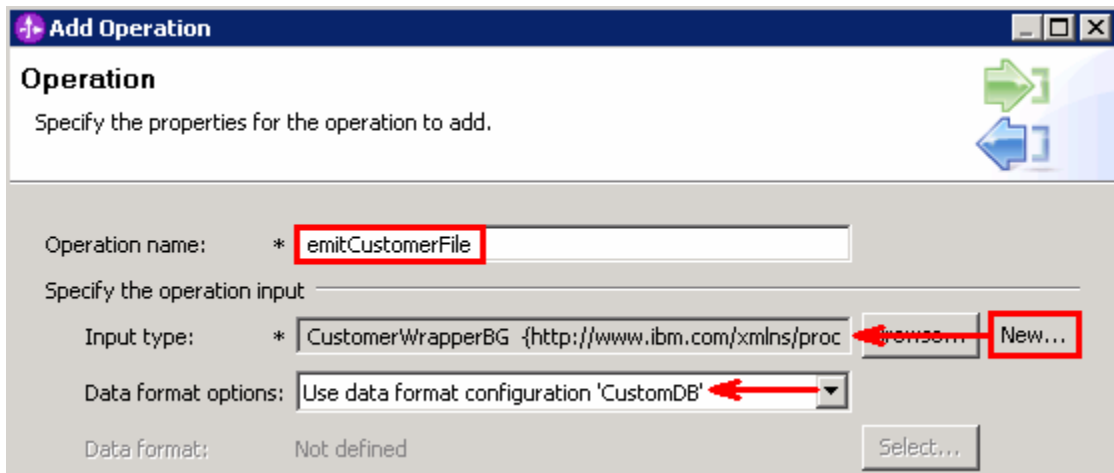
**Note:** The 'Generate retrieve container to retrieve multiple business objects' is used only during outbound retrieve operation.



8) Click **Finish**

\_\_\_ d. The Input Type in Add Operation window is populated with CustomerWrapperBG, because you have selected to have business graph (BG) generated

\_\_\_ e. For **Data format options**, accept the default selection **Use data format configuration 'CustomDB'** from the dropdown list



\_\_\_ f. Click **Finish**. Above defined operation, emitCustomerFile is populated in the Operations list.



## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

**Define emitOrderFile operation:**

\_\_\_ 16. From the Operations screen, click **Add...**

Add Operation window is opened

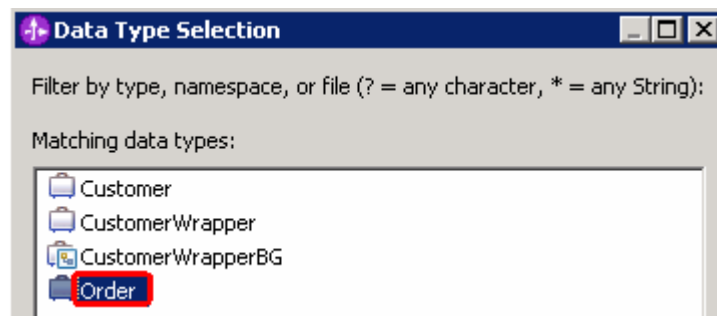
\_\_\_ a. Select **User defined type** for the Data type and click **Next**

You are back to Operation window and because you chose the User defined data type, the Input type is blank.

\_\_\_ b. For Operation name, enter **emitOrderFile**

\_\_\_ c. Define Input type:

- 1) Click **New...** next to **Data type** to open a Type Wizard window
- 2) From the next screen, ensure that **FTPCustomInboundModule** is selected and click **Next**
- 3) Click **Browse...** next to **Data type**
- 4) From the Data Type Selection window, select **Order** under Matching data types:



- 5) Click **OK**
- 6) **Do not** check the box next to **Generate business graph for each business object**
- 7) **Do not** check the box for '**Generate retrieve container to retrieve multiple business objects**'

---

**Note:** From V6.1, the generation of business graph is optional and you can leave this option unchecked. As a result, Adapter will not generate the BG for Order business object and you can confirm this by reviewing the generated data types from the business integration view of your WebSphere Integration Developer. The 'Generate retrieve container to retrieve multiple business objects' is used only during outbound retrieve operation.

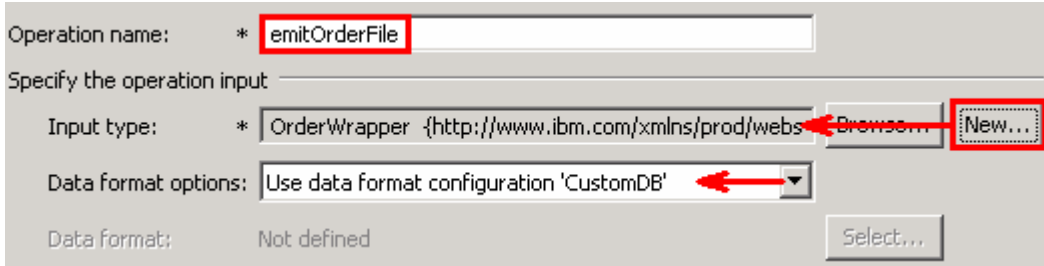
---



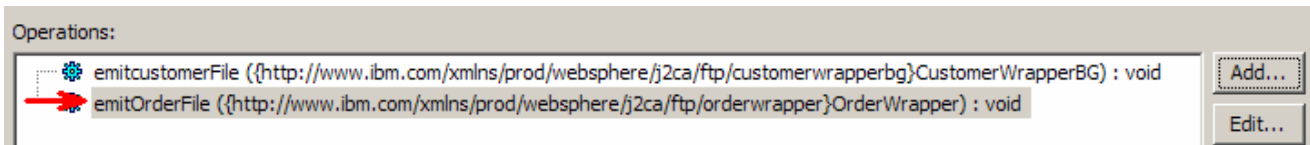
- 8) Click **Finish**

IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ d. The Input Type in Add Operation window is populated with **OrderWrapper**, because you have **not** selected to have business graph (BG) generated
- \_\_\_ e. For **Data format options**, accept the default selection **Use data format configuration 'CustomDB'** from the dropdown list

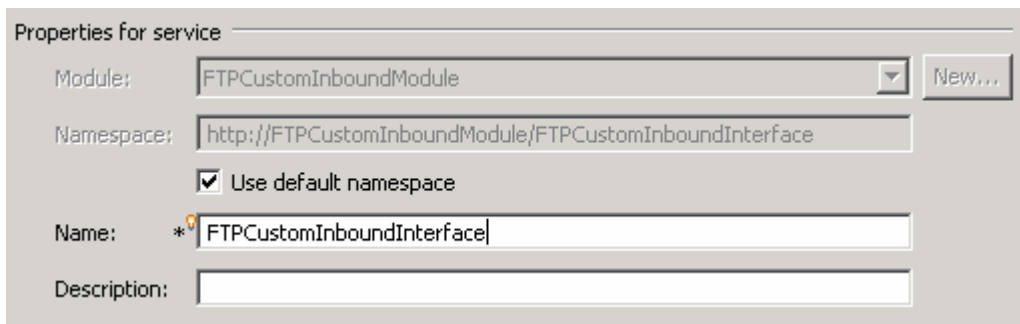


- \_\_\_ f. Click **Finish**
- \_\_\_ g. The above defined operation; emitOrderFile is populated in the Operations list

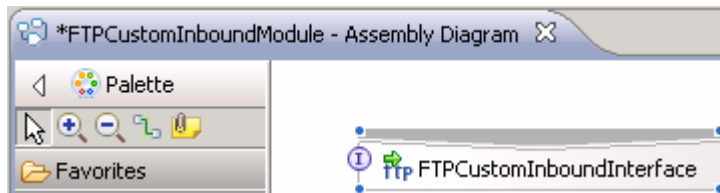


- \_\_\_ h. Click **Next**
- \_\_\_ 17. From the Generate Artifacts screen, enter these:

- \_\_\_ a. For Name, enter **FTPCustomInboundInterface**



- \_\_\_ b. Click **Finish**
- \_\_\_ 18. You will now see a new export component, **FTPCustomInboundInterface** in the assembly diagram of FTPCustomInboundModule



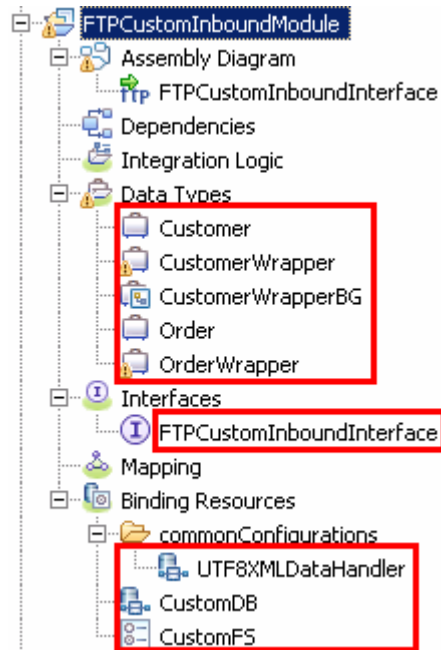
- \_\_\_ a. Save (**Ctrl+S**) your changes to the assembly diagram

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ 19. Review the generated Native method bindings for the defined operations:
- \_\_\_ a. Ensure that the FTPCustomInboundInterface is selected from the Assembly diagram
  - \_\_\_ b. From the bottom panel, select **Properties > Binding > Method bindings**

**Note:** The Native method name should be 'emit' added as a prefix to the business object name, that is., **Native method = emit + Business Object name**. In this case, for the processed business object **Customer**, the Native method is **emitCustomer** and for **Order**, it is **emitOrder**.

- \_\_\_ c. From the Bound Methods list, click **emitCustomerFile** and you should see **emitCustomer** as the **Native method**
  - \_\_\_ d. Now, click **emitOrderFile** from Bound Methods and you should see **emitOrder** as the **Native method**
- \_\_\_ 20. Review the FTPCustomInboundModule: The generated **Data Types**, **Interface**, and the Function selector (**CustomFS**) Data binding (**CustomDB**), and Data handler (**UTF8XMLDataHandler**) under Configured Resources can be found inside FTPCustomInboundModule

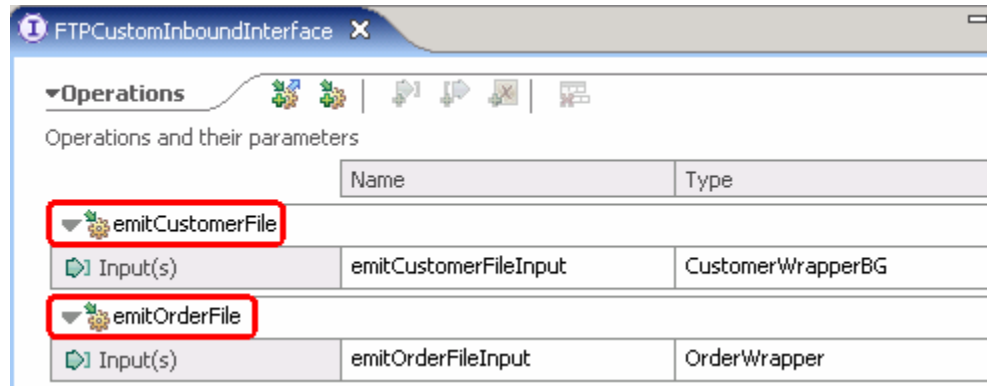


You can open each of these generated artifacts and business objects and review the properties inside.

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

Review the created methods inside the interface:

- \_\_\_ a. From the Business Integration view, expand FTPCustomInboundModule > Interfaces and then double-click **FTPCustomInboundInterface** to open it




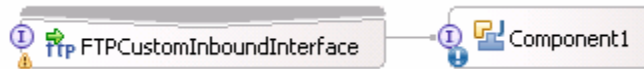
- \_\_\_ b. Close the interface, FTPCustomInboundInterface


## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

## 3.2. Add Java component

In this part of the lab, you will add a Java component to the module and then wire the export to the added component. Then you will add the custom Java code to the added module.

- \_\_\_ 1. Open the assembly diagram for FTPCustomInboundModule, if it is already not open
  - \_\_\_ a. From the business integration view, expand **FTPCustomInboundModule** and double click **Assembly diagram**
- \_\_\_ 2. Drop a Java component to onto the assembly diagram
  - \_\_\_ a. From the Palette, click **Components** to expand it
  - \_\_\_ b. Click **Java** and then click the empty space of FTPCustomInboundModule assembly diagram. This will place a new component, Component1 on the assembly diagram.
- \_\_\_ 3. Wire the FTPCustomInboundInterface to the Component1
  - \_\_\_ a. Select the **wire** () icon from the Palette
  - \_\_\_ b. Click **FTPCustomInboundInterface** and then click **Component1** to wire them together
  - \_\_\_ c. Select **OK** for the Add Wire pop-up window. Your assembly diagram for FTPCustomInboundModule will look like this:



- \_\_\_ d. From the top of the Palette, click the **Selection Tool** icon () to get back to the normal cursor mode
- \_\_\_ 4. Generate Java Implementation
    - \_\_\_ a. Right-click **Component1** and select **Generate Implementation** from the pop-up menu
    - \_\_\_ b. On the **Generate Implementation** panel, select default package, and click **OK**
    - \_\_\_ c. **Component1Impl.java** is opened in Assembly editor. Scroll down to the method **emitCustomerFile** that needs to be implemented and add this code under that method:

```
System.out.println("*****ENDPOINT emitCustomer*****");
DataObject wrapper =
emitCustomerFileInput.getDataObject("CustomerWrapper");
String filename = wrapper.getString("Filename");
System.out.println("FILENAME : "+filename);
DataObject customer = wrapper.getDataObject("Content");
String name = customer.getString("customerName");
System.out.println("NAME-----> "+name);
String address = customer.getString("Address");
System.out.println("ADDRESS--> "+address);
String city = customer.getString("City");
System.out.println("CITY-----> "+city);
String state = customer.getString("State");
System.out.println("STATE-----> "+state);
```

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

---

**Note:** The code is also available at <FTPFILES>\CustomerJavaCode.txt for your convenience

---

\_\_\_ d. Scroll down to the method **emitOrderFile** and add this code:

```
System.out.println("*****ENDPOINT emitOrder*****");
System.out.println("FILENAME :
"+emitOrderFileInput.getString("Filename"));
DataObject order = emitOrderFileInput.getDataObject("Content");
String OrderNumber = order.getString("OrderNumber");
System.out.println("ORDER NUMBER-----> "+OrderNumber);
String OrderType = order.getString("OrderType");
System.out.println("ORDER TYPE--> "+OrderType);
String Quantity = order.getString("Quantity");
System.out.println("QUANTITY-----> "+Quantity);
String Price = order.getString("Price");
System.out.println("PRICE-----> "+Price);
```

---

**Note:** The code is also available at <FTPFILES>\OrderJavaCode.txt for your convenience

---

\_\_\_ e. Save (**Ctrl + S**) and close Component1Impl.java

\_\_\_ 5. Save (**Ctrl + S**) and close Assembly diagram: FTPCustomInboundModule

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

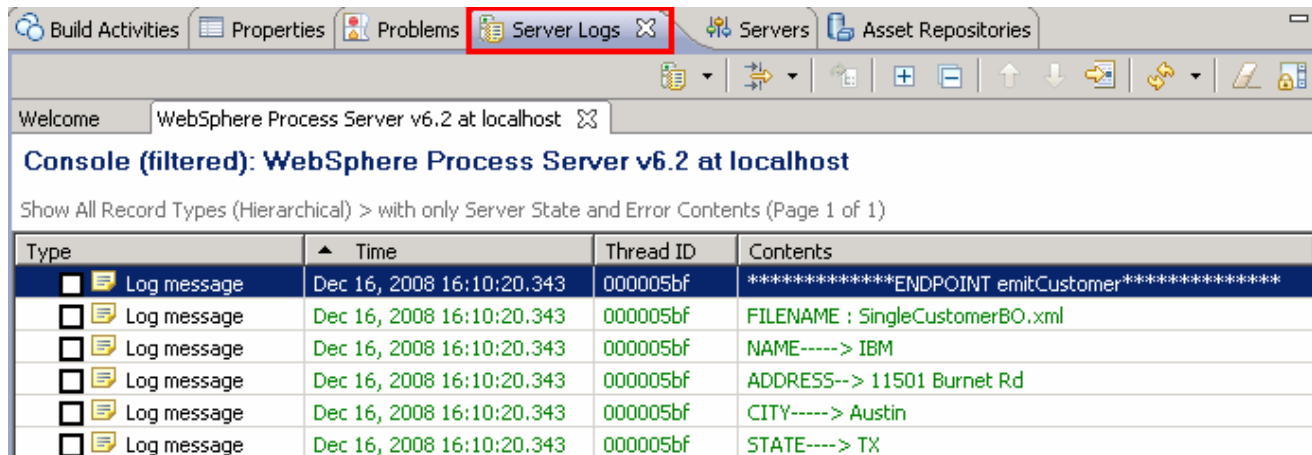
### 3.3. Test non pass through scenario

In this part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application Inbound processing for the non pass through Scenario with input file having a single business object.

- \_\_\_ 1. Add the project to the WebSphere Test Environment server
  - \_\_\_ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects...** from the pop-up menu
  - \_\_\_ b. From the Add and Remove Projects window, select **FTPCustomInboundModuleApp** under Available projects panel and click **Add >**
  - \_\_\_ c. You will now see the **FTPCustomInboundModuleApp** added to the **Configured projects**
  - \_\_\_ d. Click **Finish** and wait until the project is being published onto the server. The server is started in Debug mode if it is not already started before
- \_\_\_ 2. Put the input files in the event directory

**Note:** For your convenience, the test files **SingleCustomerBO.xml**, **SingleOrderBO.xml** are placed in **<FTPFILES>**.

- \_\_\_ a. On the machine where the FTP Server is running, put **SingleCustomerBO.xml** file in the **EventDir**. The adapter will poll the copied file from the event directory and will transfer it to the archive directory
- \_\_\_ b. Because you have placed a file that contains a BO of type Customer, it will pass through the **emitCustomerFile** method and you should see this message in your **Server Logs** view (or SystemOut.log):



The screenshot shows the 'Server Logs' window in the WebSphere Process Server v6.2 IDE. The console displays a filtered view of logs for the 'WebSphere Process Server v6.2 at localhost'. The logs show a sequence of messages from the 'emitCustomerFile' method, including the filename 'SingleCustomerBO.xml' and customer details: NAME: IBM, ADDRESS: 11501 Burnet Rd, CITY: Austin, and STATE: TX.

Type	Time	Thread ID	Contents
Log message	Dec 16, 2008 16:10:20.343	000005bf	*****ENDPOINT emitCustomer*****
Log message	Dec 16, 2008 16:10:20.343	000005bf	FILENAME : SingleCustomerBO.xml
Log message	Dec 16, 2008 16:10:20.343	000005bf	NAME----> IBM
Log message	Dec 16, 2008 16:10:20.343	000005bf	ADDRESS--> 11501 Burnet Rd
Log message	Dec 16, 2008 16:10:20.343	000005bf	CITY----> Austin
Log message	Dec 16, 2008 16:10:20.343	000005bf	STATE----> TX

- \_\_\_ c. On the machine where the FTP Server is running, put **SingleOrderBO.xml** file in the **EventDir**. The adapter will poll the copied file from the event directory and will transfer it to the archive directory

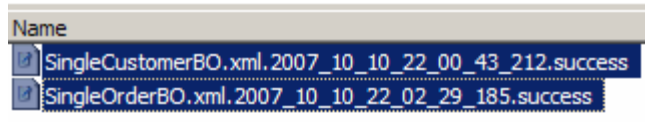


IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ d. Because you have placed a file that contains a BO of type Order, it will pass through the **emitOrderFile** method and you should see this message in your Server Logs view (or SystemOut.log):

Type	Time	Thread ID	Contents
Log message	Dec 16, 2008 16:10:20.343	000005bf	*****ENDPOINT emitCustomer*****
Log message	Dec 16, 2008 16:10:20.343	000005bf	FILENAME : SingleCustomerBO.xml
Log message	Dec 16, 2008 16:10:20.343	000005bf	NAME----> IBM
Log message	Dec 16, 2008 16:10:20.343	000005bf	ADDRESS--> 11501 Burnet Rd
Log message	Dec 16, 2008 16:10:20.343	000005bf	CITY----> Austin
Log message	Dec 16, 2008 16:10:20.343	000005bf	STATE----> TX
Log message	Dec 16, 2008 16:10:50.078	000005bf	*****ENDPOINT emitOrder*****
Log message	Dec 16, 2008 16:10:50.078	000005bf	FILENAME : SingleOrderBO.xml
Log message	Dec 16, 2008 16:10:50.078	000005bf	ORDER NUMBER----> ABC12345
Log message	Dec 16, 2008 16:10:50.078	000005bf	ORDER TYPE--> BULK
Log message	Dec 16, 2008 16:10:50.078	000005bf	QUANTITY----> 500
Log message	Dec 16, 2008 16:10:50.078	000005bf	PRICE----> 26.59

- \_\_\_ 3. You can also verify the results by reviewing the archive directory
  - \_\_\_ a. Check the **ArchiveDir** of your FTP server which should contain the same file name appended with year, month, date, system time, and processed
  - \_\_\_ b. Check the **<LOCAL\_ARCHIVE\_DIR>** subdirectory which should contain two archives of the event files, with the same file name appended with year, month, date, system time, and success



- \_\_\_ 4. Restore the Sever Configuration
  - \_\_\_ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects...** from the pop-up menu
  - \_\_\_ b. Select **FTPCustomInboundModuleApp** under Configured projects and click **< Remove**
  - \_\_\_ c. Click **Finish** after you see the application moved to Available projects. Wait until the application is being unpublished

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

### 3.4. Test non pass through scenario with SplitByDelimiter

In this last part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application Inbound processing for the non pass through Scenario with input file having multiple business objects separated by a delimiter.

- \_\_\_ 1. Configure/change the adapter properties from the Properties view:
  - \_\_\_ a. Change to Business Integration perspective if you are in a different perspective
    - 1) Select **Window > Open Perspective > Other...**
    - 2) From the Select Perspective window, select **Business Integration (default)** and click **OK**
  - \_\_\_ b. Expand **FTPCustomInboundModule** and double-click **FTPCustomInboundModule** to open it in Assembly Editor
  - \_\_\_ c. Click **FTPCustomInboundInterface** from the Assembly Editor and select **Properties** tab from the bottom
  - \_\_\_ d. Select **Binding** under Properties and select **End-point configuration** under Binding itself and then select the **Connection**
  - \_\_\_ e. Now click **Advanced>>** button at the bottom
  - \_\_\_ f. You can see all the properties that were given during the external service wizard
  - \_\_\_ g. Scroll down to **Advanced Properties:**
    - 1) Select the box for '**Split file content based on the size (bytes) or delimiter**'
    - 2) Split function class name: **com.ibm.j2ca.utils.filesplit.SplitByDelimiter** (you can also Browse... and select this class name)

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

## 3) Specify criteria to split file content: #####

Advanced connection configuration	
Retrieve files with this pattern:	<input type="text" value="*,*"/>
Sort event files:	<input type="text" value="no sort"/>
Encoding used by FTP server:	<input type="text"/> <input type="button" value="Select..."/>
File content encoding:	<input type="text"/> <input type="button" value="Select..."/>
FTP server connection mode:	<input type="text" value="active"/>
File transfer type:	<input type="text" value="binary"/>
Number of files to get at a time:	<input type="text" value="10"/>
Number of poll periods between downloads:	<input type="text" value="5"/>
Populate the fully qualified class name of the custom parser that is used to parse the "ls -l" output. This is used only when the "ls -l" output deviates from standard output.	
Custom parser class name:	<input type="text"/> <input type="button" value="Browse..."/>
<input type="checkbox"/> Pass only file name and directory, not the content	
<input type="checkbox"/> Include business object delimiter in the file content	
<input checked="" type="checkbox"/> Split file content based on the size (bytes) or delimiter.	
Split function class name:	<input type="text" value="com.ibm.j2ca.utils.filesplit.SplitByDelimiter"/> <input type="button" value="Browse..."/>
Specify criteria to split file content:	<input type="text" value="#####"/>

\_\_\_ h. Click Assembly diagram (or any where else so that the save button is enabled) and then save (Ctrl +S) your changes

\_\_\_ 2. Repeat Step 1 of part 3.3 to add the saved project **FTPCustomInboundModuleApp** to the server

\_\_\_ 3. Test the input file CustomerSplitByDelimiter.xml

**Note:** For your convenience, a test file **CustomerSplitByDelimiter.xml** is placed in <FTPFILES>. The file contains two Customer Business Objects separated by the delimiter #####.

\_\_\_ a. On the machine where the FTP Server is running, put **CustomerSplitByDelimiter.xml** file in the **EventDir**. The adapter will poll the copied file from the event directory and will transfer it to the archive directory

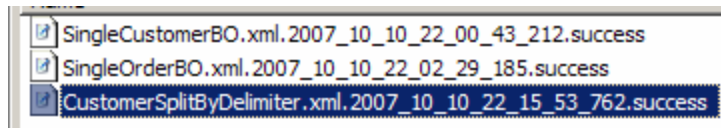
\_\_\_ b. Because you have placed a file that contains BOs of type Customer, it will pass through the **emitCustomerFile** method and you should see this message in your **Server Logs** view (or SystemOut.log):

IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

**Note:** You will see the successful event delivery message twice as there were two Business Objects present in the event file separated by the delimiter #####.

Type	Time	Thread ID	Contents
Log message	Dec 16, 2008 16:17:21.578	00000090	*****ENDPOINT emitCustomer*****
Log message	Dec 16, 2008 16:17:21.578	00000090	FILENAME : CustomerSplitByDelimiter.xml
Log message	Dec 16, 2008 16:17:21.578	00000090	NAME----> Michele
Log message	Dec 16, 2008 16:17:21.578	00000090	ADDRESS--> Da Vinci Ave.
Log message	Dec 16, 2008 16:17:21.578	00000090	CITY----> NewYork
Log message	Dec 16, 2008 16:17:21.578	00000090	STATE----> NY
Log message	Dec 16, 2008 16:17:21.593	000006fe	*****ENDPOINT emitCustomer*****
Log message	Dec 16, 2008 16:17:21.593	000006fe	FILENAME : CustomerSplitByDelimiter.xml
Log message	Dec 16, 2008 16:17:21.593	000006fe	NAME----> Michele
Log message	Dec 16, 2008 16:17:21.593	000006fe	ADDRESS--> Da Vinci Ave.
Log message	Dec 16, 2008 16:17:21.593	000006fe	CITY----> NewYork
Log message	Dec 16, 2008 16:17:21.593	000006fe	STATE----> NY

- 1) Check your <ARCHIVE\_DIR> which should contain an archive of the event file, with the same file name appended with year, month, date, system time, and success



4. Test the input file: **OrderSplitByDelimiter.xml**

**Note:** For your convenience, the test file **OrderSplitByDelimiter.xml** is placed in <LAB\_FILES>FTPfiles. The file contains two Order Business Objects separated by the delimiter #####.

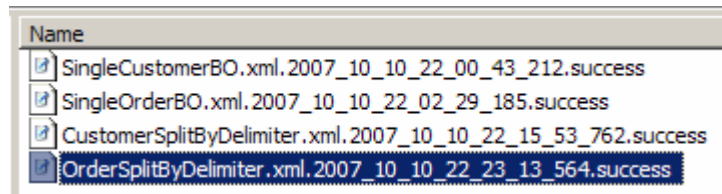
- a. On the machine where the FTP Server is running, put **OrderSplitByDelimiter.xml** file in the **EventDir**. The adapter will poll the copied file from the event directory and will transfer it to the archive directory
- b. Because you have placed a file that contains BOs of type Order, it will pass through the **emitOrderFile** method and you should see this message in your **Server Logs** view (or SystemOut.log):

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

**Note:** You will see the successful event delivery message twice as there were two Business Objects present in the event file separated by the delimiter #####.

Type	Time	Thread ID	Contents
Log message	Dec 16, 2008 16:19:11.140	000006fe	*****ENDPOINT emitOrder*****
Log message	Dec 16, 2008 16:19:11.140	000006fe	FILENAME : OrderSplitByDelimiter.xml
Log message	Dec 16, 2008 16:19:11.140	000006fe	ORDER NUMBER-----> ABC12345
Log message	Dec 16, 2008 16:19:11.140	000006fe	ORDER TYPE--> BULK
Log message	Dec 16, 2008 16:19:11.140	000006fe	QUANTITY-----> 500
Log message	Dec 16, 2008 16:19:11.140	000006fe	PRICE-----> 26.59
Log message	Dec 16, 2008 16:19:11.171	00000090	*****ENDPOINT emitOrder*****
Log message	Dec 16, 2008 16:19:11.171	00000090	FILENAME : OrderSplitByDelimiter.xml
Log message	Dec 16, 2008 16:19:11.171	00000090	ORDER NUMBER-----> ABC12345
Log message	Dec 16, 2008 16:19:11.171	00000090	ORDER TYPE--> BULK
Log message	Dec 16, 2008 16:19:11.171	00000090	QUANTITY-----> 500
Log message	Dec 16, 2008 16:19:11.171	00000090	PRICE-----> 26.59

- 1) Check your <ARCHIVE\_DIR> which should contain an archive of the event file, with the same file name appended with year, month, date, system time, and success



5. Repeat Step 4 of part 3.3 to restore the sever configuration

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

---

### **Part 4: Use default function selector and data binding**

This part of the lab will show you how to use the default use the default function selector and data binding options from the External Service wizard and generate other required artifacts.

When you use the default function selector, you cannot define the rules as you did in Part 2 and hence there will only be one method that handles all types of files.

When you use the default data binding, you cannot have multiple data types as in Part 3 and each data type is handled by different method. Instead, there will only be one method and one data type.

After running the External Service wizard, you will add the required Java component with implementation and then will continue to test the adapter.

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

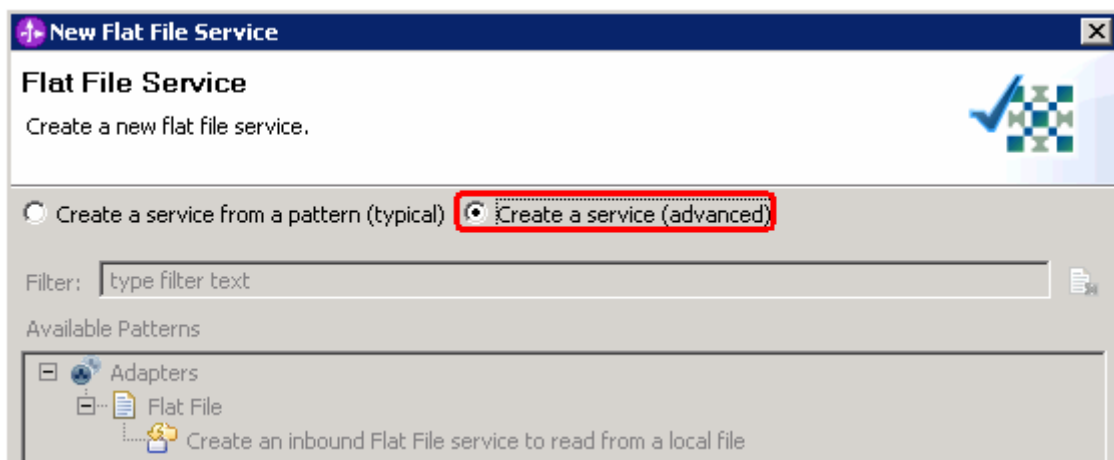
## 4.1. Configure inbound using default function selector and data binding

In this part of the lab you will use the default function selector and data binding options from the External Service wizard and generate other required artifacts to test the inbound scenario.

- \_\_\_ 1. Create FTPDefaultsInboundModule
  - \_\_\_ a. From the Business Integration window, right-click and select **New > Module**
  - \_\_\_ b. From the New Module window, enter **FTPDefaultsInboundModule** for the Module Name
  - \_\_\_ c. Ensure that the box next to **Open module assembly diagram** is checked and then click **Finish**

You will now see a new module, FTPDefaultsInboundModule, created from your Business Integration window

- \_\_\_ 2. To start External Service from the Palette:
  - \_\_\_ a. From the **Palette** on the left side of Assembly Diagram, click **Inbound Adapters**:
  - \_\_\_ b. Under Inbound Adapters, click the **FTP** and then click the empty canvas of the assembly diagram. The New FTP Service wizard is opened
- \_\_\_ 3. From the FTP Service screen, select **Create a service (advanced)**



- \_\_\_ a. Click **Next**

**Note:** You can also start the External Service from the **File menu** option:

From the main menu, select **File > New > External Service**. This opens an External Service wizard that helps you obtain a service which establishes connectivity with other systems. Select **Adapters > Flat File** and click **Next**.

- \_\_\_ 4. On the Select an Adapter screen, expand **IBM WebSphere Adapter for FTP (IBM : 6.2.0.0)** and select **CWYFT\_FTPFile**
  - \_\_\_ a. Click **Next**

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

**Note:** If you are using the **File menu** option to start the External Service wizard, you are asked to select the **Processing Direction** at this point. Select the radio button next to **Inbound** and click **Next** to proceed to the next step.

5. Service Configuration Properties:

a. Deploy connector project: ensure that the default option **With module for use by single application** is selected

b. Enter these for FTP system connection information:

- 1) Host name: **<FTP\_Machine\_Name>** (or IP Address of the machine that has FTP Server), for Ex: wsbeta181.austin.ibm.com
- 2) Remote directory: **full path of the EventDir created in on the machine where FTP server is existing** (for Ex: /home/wsbeta/EventDir)

**Note:** This is the directory from where adapter gets the event files.

3) Local directory: **<LOCAL\_EVENT\_DIR>**

4) Protocol: **FTP – file transfer protocol** (default)

5) Port number: **21** (default)

6) **User name:** username using which you connect to your FTP server (for Ex: **root**)

7) **Password:** password for the above user to connect to your FTP server

Deploy connector project: With module for use by single application

Connection properties: Use properties below

Connection properties

FTP system connection information

Host name: \* wsbeta181.austin.ibm.com

Remote directory:\* /home/wsbeta/EventDir

Local directory: \* C:\Labfiles62\FTPInbound\LocalEventDir

Protocol: FTP - file transfer protocol

Port number: 21

The user name and password will not be encrypted and will be stored as plain text.

User name: root

Password: \*\*\*\*\*

c. Click **Advanced >>** to see the hidden advanced properties that can be configured:

You can click each of the configuration and review the options available under it. For this part of the lab, you will need only some of these properties.



IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ d. Event persistence configuration: In this part of the lab, you will not use any JNDI instead use adapter's in-memory representation of event table to store all the necessary information

**Note:** The Event recovery data source (JNDI) name is **not mandatory** from V6.1. Now, the adapter can use **in-memory representation** of event table to store all the necessary information. Adapter uses this feature when event database information is not configured during inbound event polling. This feature will not support the capability of handling "Ensure once-only event delivery".

- \_\_\_ e. FTP archiving configuration:

- 1) Local Archive directory: click **Browse...** and select **<LOCAL\_ARCHIVE\_DIR>**
- 2) Remote archive directory: **full path of the ArchiveDir created in on the machine where FTP server is existing** (for Ex: /home/wsbeta/ArchiveDir)

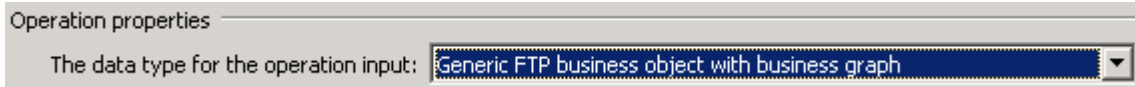
- \_\_\_ 6. For this lab, you are not going to use the J2C authentication. So, **uncheck** the box next to **Specify a Java Authentication and Authorization Services (JAAS) alias security credentials**.
- \_\_\_ 7. Under Service properties, for **Function selector options**, select **Use default function selector 'FilenameFunctionSelector'** from the drop down list
- \_\_\_ 8. For **Data format options**, select **Use default data binding 'FTPFileBaseDataBinding' for all operations** from the drop down list

- \_\_\_ 9. Check the box next to **Change logging properties for wizard** to view the output location of the log file and the logging level and click **Next**

IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

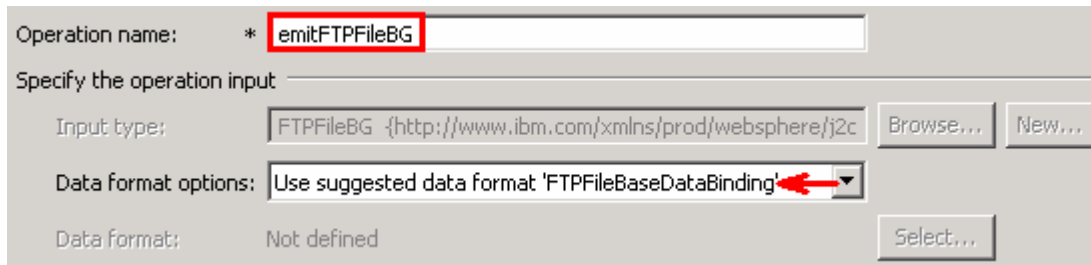
**Define emitFTPFileBG operation:**

- \_\_\_ 10. From the Operations screen, click **Add...**
  - \_\_\_ a. Add Operation window is opened. Select **Generic FTP business object with business graph** for the Data type and click **Next**



You are back to Operation window and because you have chosen the data type with business graph, the Input type is populated as **FTPFileBG**.

- \_\_\_ 11. For **Operation name**, enter any name, for Ex: **emitFTPFileBG**
- \_\_\_ 12. Accept the default selection, **Use suggested data format 'FTPFileBaseDataBinding'**, for **Data format options**

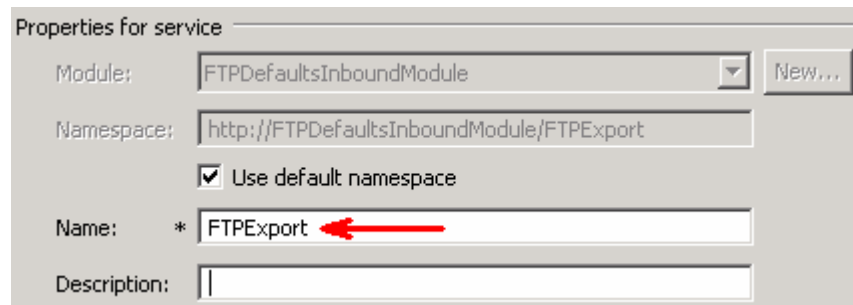


- \_\_\_ a. Click **Finish**. The above defined operation, **emitFTPFileBG**, is populated under Operations list



- \_\_\_ b. Click **Next** from Operations screen

- \_\_\_ 13. From Generate Service screen, accept the default value, **FTPEXport**, for **Name**





- \_\_\_ a. Click **Finish**

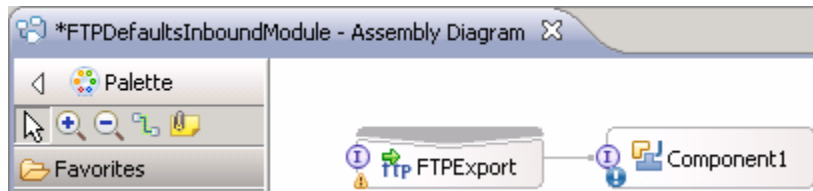
- \_\_\_ 14. The Assembly diagram for FTPDefaultsInboundModule is opened with an Export component, **FTPEXport**
- \_\_\_ 15. Save (**Ctrl + S**) changes to your assembly diagram

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

## 4.2. Add Java component

In this part of the lab, you will add a Java component and then wire the component to the existing Export interface. The Java component is your endpoint.

- \_\_\_ 1. Open the assembly diagram for FTPDefaultsInboundModule (if it is already not open)
  - \_\_\_ a. From the business integration view, expand **FTPDefaultsInboundModule** and double click **Assembly diagram**
- \_\_\_ 2. Drop a Java component to onto the assembly diagram
  - \_\_\_ a. From the **Palette**, click **Components** to expand it
  - \_\_\_ b. Click **Java** and then click the empty space of FTPDefaultsInboundModule assembly diagram. This will place a new component, **Component1** on the assembly diagram.
- \_\_\_ 3. Wire the FTPEXport to the Component1
  - \_\_\_ a. Select the **wire** () icon from the Palette
  - \_\_\_ b. Click **FTPEXport** and then click **Component1** to wire them together
  - \_\_\_ c. Select **OK** for the Add Wire pop-up window:
  - \_\_\_ d. From the top of the Palette, click the **Selection Tool** icon () to get back to the normal cursor mode
  - \_\_\_ e. Right-click the empty space of the Assembly diagram and select **Arrange Contents Automatically** from the pop-up menu. Your assembly diagram for FTPDefaultsInboundModule will look like this:



- \_\_\_ f. Right-click **Component1** and select **Generate Implementation** from the pop-up menu
- \_\_\_ g. On the **Generate Implementation** panel, select **default package**, and click **OK**
- \_\_\_ h. **Component1Impl.java** is opened in Assembly editor. Scroll down to the method **emitFTPFileBG** that needs to be implemented and add this code under that method:
 

```
System.out.println("*****Reached Endpoint*****");
```
- \_\_\_ i. Save (**Ctrl + S**) and close Component1Impl.java
- \_\_\_ j. Save (**Ctrl + S**) and close Assembly diagram: FTPDefaultsInboundModule

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE



### 4.3. Test defaults scenario

In this part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application Inbound processing for the pass through scenario.

- \_\_\_ 1. Add the project to the WebSphere Test Environment server
  - \_\_\_ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects...** from the pop-up menu
  - \_\_\_ b. From the Add and Remove Projects window, select **FTPDefaultsInboundModuleApp** under Available projects panel and click **Add >**
  - \_\_\_ c. You will now see the **FTPDefaultsInboundModuleApp** added to the **Configured projects**
  - \_\_\_ d. Click **Finish** and wait until the project is being published onto the server. The server is started in Debug mode if it is not already started before
- \_\_\_ 2. Test the application by providing three different input files:

**Note:** For your convenience, three test files, **sample.txt**, **sample.txt1**, **sample.txt2** are placed in **<FTPFILES>**.

- \_\_\_ a. On the machine where the FTP Server is running, put **sample.txt** (or any of the three sample files) file in the **EventDir**. The adapter will poll the copied file from the event directory and will transfer it to the archive directory
- \_\_\_ b. No matter what type of file you put in Event directory, it will pass through the only existing method, **emitFTPFileBG**, and you should see this message in your **Server Logs** view (or SystemOut.log):

 Log message	Dec 16, 2008 17:13:57.984	00000726	WSVR0221I: Application started: FTPDefaultsInboundModu...
 Log message	Dec 16, 2008 17:14:28.906	00000090	*****Reached Endpoint*****

Verify the results:

- \_\_\_ c. Check the **<LOCAL\_EVENT\_DIR>** on your local machine. The file is quickly moved from this directory to the local archive directory
  - \_\_\_ d. Check the **ArchiveDir** of your FTP server which should contain the same file name appended with year, month, date, system time, and processed as you have given
  - \_\_\_ e. Check the **<LOCAL\_ARCHIVE\_DIR>** subdirectory which should contain an archive of the event file, with the same file name appended with year, month, date, system time, and success
- \_\_\_ 3. Restore the Sever Configuration
    - \_\_\_ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects...** from the pop-up menu
    - \_\_\_ b. Select **FTPDefaultsInboundModuleApp** under Configured projects and click **< Remove**
    - \_\_\_ c. Click **Finish** after you see the application moved to Available projects. Wait until the application is being unpublished

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

---

### Part 5: Use ‘Create a service from a typical pattern’

In this part of the lab you will use the **typical pattern** option from the External Service wizard to create and configure the Function Selector, Data Binding and other required artifacts to test the inbound scenario.

Based on your selection, the Binding resources (data binding and function selector) are created which you will review later in this part.

After running the External Service wizard, you will continue to add the Java component with implementation and then test the adapter.

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

## 5.1. Configure inbound using ‘Create a service from a pattern (typical)’ option

In this part of the lab you will use the **typical pattern** from the External Service feature to create and configure the Function Selector

- \_\_\_ 1. Create the module: FTPTypicalInboundModule
  - \_\_\_ a. From the Business Integration window, right-click and select **New > Module**
  - \_\_\_ b. From the New Module window, enter **FTPTypicalInboundModule** for the Module Name
  - \_\_\_ c. Ensure that the box next to **Open module assembly diagram** is checked and then click **Finish**

You will now see a new module, **FTPTypicalInboundModule**, created from your Business Integration window and the Assembly diagram for the same module is opened in the Assembly Editor.

- \_\_\_ 2. Import required business objects
  - \_\_\_ a. Expand FTPTypicalInboundModule (if not already expanded), right-click **Data Types** and select **Import...** from the pop-up menu
  - \_\_\_ b. From the Import window, expand **General** and select **File System** and then click **Next**
  - \_\_\_ c. Enter From directory
    - 1) Click **Browse...** next to **From directory**
    - 2) From the Import from directory window, select **<FTPFILES >** and click **OK**

Now, you will see FTPFiles folder added on the left side, and all the xsds and files under that folder on the right side.

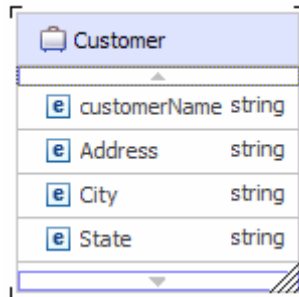
- \_\_\_ d. Select the box next to **Customer.xsd**
- \_\_\_ e. Ensure that the **FTPTypicalInboundModule** is selected for Into folder
- \_\_\_ f. Click **Finish** from the Import window

The Business Integration window is updated with the imported business objects.

- \_\_\_ 3. Review imported business object:
  - \_\_\_ a. Expand **FTPTypicalInboundModule > Data Types** and you will now see a new data type **Customer** and **Order** under it.

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

\_\_ b. Double-click **Customer** review the fields inside the object:



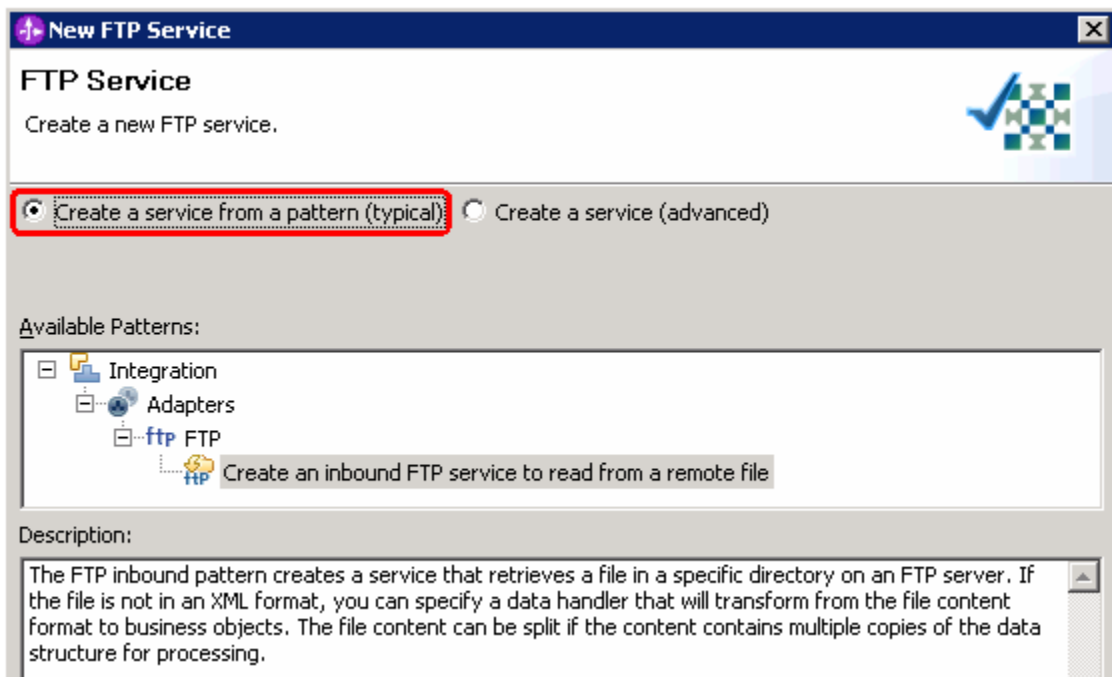
\_\_ c. After reviewing, close the Customer business object from the Assembly editor

\_\_\_ 4. To start External Service from the Palette:

\_\_ a. From the **Palette** on the left side of Assembly Diagram, click **Inbound Adapters**:

\_\_\_ 5. Under Inbound Adapters, click the **FTP** and then click the empty canvas of the assembly diagram. The New FTP Service wizard is opened

\_\_\_ 6. From the FTP Service screen, accept the default selection of **Create a service from a pattern (typical)**



\_\_ a. Click **Next**

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ 7. From the next **FTP service name** screen, for **Name**, accept the default value '**FTPEXport**' and click **Next**

- \_\_\_ 8. From the **Business object and location** screen, enter these:
- \_\_\_ a. Click **Browse...** next to **Business object** and a Data Type Selection window is opened
  - \_\_\_ b. Select **Customer** under Matching data types and click **OK**

- \_\_\_ c. For **FTP server host name**, enter **<FTP\_Machine\_Name>** (or IP Address of the machine that has FTP Server), for Ex: wsbeta181.austin.ibm.com
- \_\_\_ d. Click **Test connection**, next to host name and you should get this pop-up window with success message:

- \_\_\_ e. Click **OK** from the above Test Results window
- \_\_\_ f. For **Remote directory**, enter **full path of the EventDir created on the machine where FTP server is existing** (for Ex: /home/wsbeta/EventDir)
- \_\_\_ g. For **Local staging directory**, click **Browse...** and select **<LOCAL\_EVENT\_DIR>**



## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

Your Business object and location screen should look like this:

**New Inbound FTP Service**

**Business object and location**  
Specify the business object and location for the input file.

**What business object do you want to read from the input file?**

Business object:

**Where is the input file?**

FTP server host name:

Remote directory:

**Where do you want to temporarily put the input file?**

Local staging directory:

\_\_\_ h. Click **Next**

\_\_\_ 9. From 'FTP server security credential' screen, enter these:

\_\_\_ a. Select the radio button next to **Using user name and password**

- 1) User name: **username using which you connect to your FTP server** (for Ex: wsbeta)
- 2) Password: **password for the above user to connect to your FTP server**

**New Outbound FTP Service**

**FTP server security credential**  
Specify the FTP server security credential.

**How do you want to specify the FTP server security credential?**

**Using an existing JAAS alias (recommended)**  
Java Authentication and Authorization Services (JAAS) alias is the recommended way for specifying security credentials.  
J2C authentication data entry:

**Using user name and password**  
The user name and password will not be encrypted and will be stored as plain text.

User name:

Password:

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

\_\_ b. Click **Next**

\_\_\_ 10. From the **Input file format and file content split option** screen, enter these:

\_\_ a. For input file format, accept the default **XML** selection

\_\_ b. For file content split option, accept the default selection, **None**

**New Inbound Flat File Service**

**Input file format and file content split option**  
Specify the input file format and the file content split option.

**What is the input file format?**

**XML**

**Other**  
Specify a data handler to transform the native data format to a business object.  
Data handler:

**Which file content split option do you want to use?**

**None**

**Split file content by fixed size**  
Size (in bytes):

**Split file content by delimiter**  
Delimiter:

**Split file content using a custom splitter**  
Custom splitter:    
Split criteria:

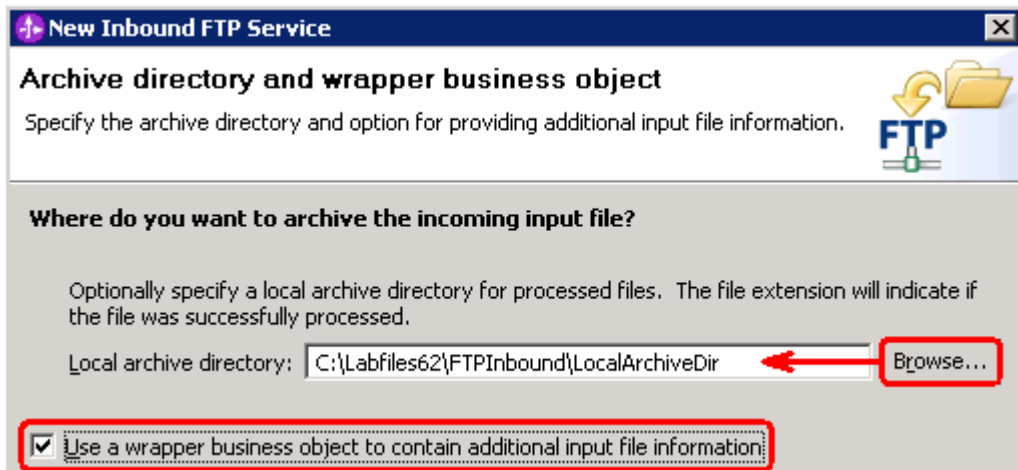
\_\_ c. Click **Next**

\_\_\_ 11. From the 'Archive directory and wrapper business object' screen, enter these:

\_\_ a. Click **Browse...** next to **Local archive directory** and select **<LOCAL\_ARCHIVE\_DIR>**

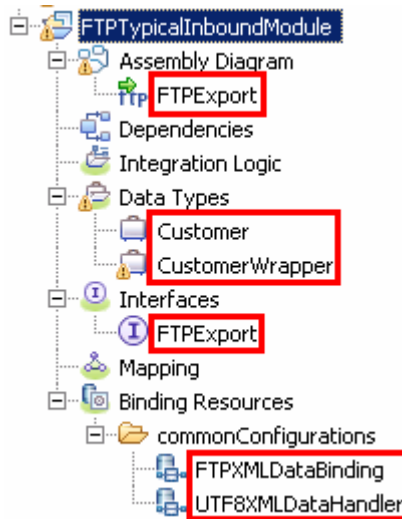
IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ b. Check the box next to **Use a wrapper business object to contain additional input file information**. This will generate a Customer Wrapper under the Data Types of your Module



- \_\_\_ c. Click **Finish**



- \_\_\_ 12. Save (**Ctrl + S**) changes to your assembly diagram
- \_\_\_ 13. Review the FTPTypicalInboundModule and the generated artifacts: The generated **Data Types**, **Interface**, Data handler (**XMLDataHandler**) and Data binding (**FTPFileXMLDataBinding**) under Configured Resources can be found under FTPTypicalInboundModule. You can open each of these generated artifacts, business objects and review the properties inside.



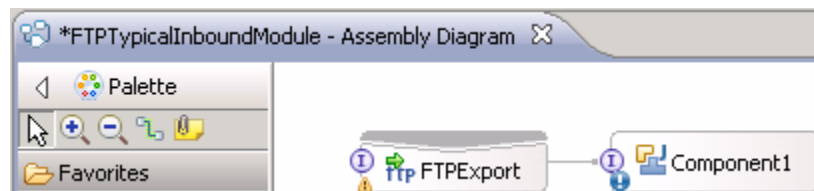
## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

## 5.2. Add Java component

In this part of the lab, you will add a Java component and then wire the component to the existing Export interface.

- \_\_\_ 1. Open the assembly diagram for FTPTypicalInboundModule (if it is already not open)
  - \_\_\_ a. From the business integration view, expand **FTPTypicalInboundModule** and double click **Assembly diagram**
- \_\_\_ 2. Drop a Java component to onto the assembly diagram
  - \_\_\_ a. From the **Palette**, click **Components** to expand it
  - \_\_\_ b. Click **Java** and then click the empty space of FTPTypicalInboundModule assembly diagram. This will place a new component, **Component1** on the assembly diagram.
- \_\_\_ 3. Wire the FTPEXport to the Component1
  - \_\_\_ a. Select the **wire** (  ) icon from the Palette
  - \_\_\_ b. Click **FTPEXport** and then click **Component1** to wire them together
  - \_\_\_ c. Select **OK** for the Add Wire pop-up window:
  - \_\_\_ d. From the top of the Palette, click the **Selection Tool** icon (  ) to get back to the normal cursor mode
  - \_\_\_ e. Right-click the empty space of the Assembly diagram and select **Arrange Contents Automatically** from the pop-up menu

Your assembly diagram for FTPTypicalInboundModule will look like this:



- \_\_\_ f. Right-click **Component1** and select **Generate Implementation** from the pop-up menu
- \_\_\_ g. On the **Generate Implementation** panel, select **default package**, and click **OK**

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_ h. **Component1Impl.java** is opened in Assembly editor. Scroll down to the method **emit** that needs to be implemented and add this code under that method:

```
System.out.println("*****ENDPOINT
emitCustomer*****");
System.out.println("FILENAME : "+emitInput.getString("Filename"));
DataObject customer = emitInput.getDataObject("Content");
String name = customer.getString("CustomerName");
System.out.println("NAME-----> "+name);
String address = customer.getString("Address");
System.out.println("ADDRESS--> "+address);
String city = customer.getString("City");
System.out.println("CITY-----> "+city);
String state = customer.getString("State");
System.out.println("STATE-----> "+state);
```

---

**Note:** The code is also available at <FTPFILES>\TypicalCustomerJavaCode.txt for your convenience

---

- \_\_ i. Save (**Ctrl + S**) and close Component1Impl.java
- \_\_ j. Save (**Ctrl + S**) and close Assembly diagram: FTPTypicalInboundModule

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

### 5.3. Test typical pattern scenario

In this part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application Inbound processing for the typical pattern with input file having single business object.

- \_\_\_ 1. Add the project to the WebSphere Test Environment server
  - \_\_\_ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects...** from the pop-up menu
  - \_\_\_ b. From the Add and Remove Projects window, select **FTPTypicalInboundModuleApp** under Available projects panel and click **Add >**
  - \_\_\_ c. You will now see the **FTPTypicalInboundModuleApp** added to the **Configured projects**
  - \_\_\_ d. Click **Finish** and wait until the project is being published onto the server. The server is started in Debug mode if it is not already started before
- \_\_\_ 2. Put the input files in the event directory

**Note:** For your convenience, the test files **SingleCustomerBO.xml**, is placed in **<FTPFILES>**.

- \_\_\_ a. On the machine where the FTP Server is running, put **SingleCustomerBO.xml** file in the **EventDir**. The adapter will poll the copied file from the event directory and will transfer it to the archive directory
- \_\_\_ b. The Customer BO is passed through the **emit** method and you should see this message in your **Server Logs** view (or SystemOut.log):

 Log message	Dec 16, 2008 17:22:43.750	00000623	*****ENDPOINT	emitCustomer*****...
 Log message	Dec 16, 2008 17:22:43.750	00000623	FILENAME :	SingleCustomerBO.xml
 Log message	Dec 16, 2008 17:22:43.750	00000623	NAME----	> IBM
 Log message	Dec 16, 2008 17:22:43.750	00000623	ADDRESS--	> 11501 Burnet Rd
 Log message	Dec 16, 2008 17:22:43.750	00000623	CITY----	> Austin
 Log message	Dec 16, 2008 17:22:43.750	00000623	STATE----	> TX

- \_\_\_ 3. You can also verify the results by reviewing the archive directory
  - \_\_\_ a. Check the **ArchiveDir** of your FTP server which should contain the same file name appended with year, month, date, system time, and processed
  - \_\_\_ b. Check the **<LOCAL\_ARCHIVE\_DIR>** subdirectory which should contain an archive of the event files, with the same file name appended with year, month, date, system time, and success
- \_\_\_ 4. Restore the Sever Configuration
  - \_\_\_ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects...** from the pop-up menu
  - \_\_\_ b. Select **FTPTypicalInboundModuleApp** under Configured projects and click **< Remove**
  - \_\_\_ c. Click **Finish** after you see the application moved to Available projects. Wait until the application is being unpublished

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

### **What you did in this exercise**

In this lab, you started with importing the FTP File Adapter RAR file into your WebSphere Integration Developer new workspace. Then, you made use of the External Service wizard available in WebSphere Integration Developer to specify Activation Spec Properties, define data binding, data handler, and function selector which, after deploying onto the server, will generate business objects and other artifacts.

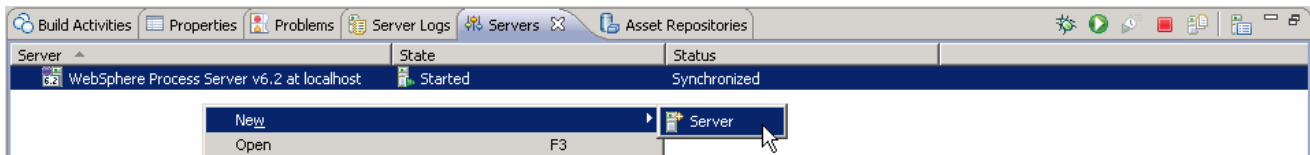
At the end of each part, you deployed and then tested the adapter application for the scenarios - pass-through (with and without SplitBySize) test scenario, two content specific or non pass through (with and without SplitByDelimiter) test scenarios, using all defaults (default data binding, function selector) scenario, and then finally using the typical pattern.

## IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

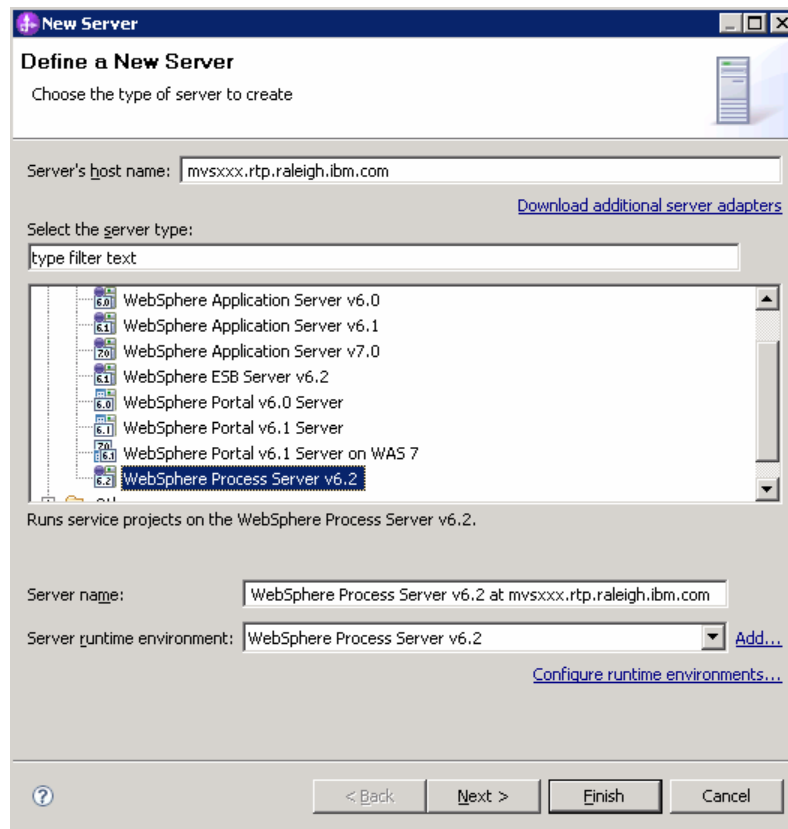
## Task: Adding remote server to WebSphere Integration Developer test environment

This task describes how to add a remote server to the WebSphere Integration Developer Test environment. This example uses a z/OS machine.

- \_\_\_ 1. Define a new remote server to WebSphere Integration Developer.
  - \_\_\_ a. Right click the background of the Servers view to access the pop-up menu.
  - \_\_\_ b. Select **New → Server**.



- \_\_\_ c. In the New Server dialog, specify the remote server's host name, **<HOSTNAME>**.
- \_\_\_ d. Ensure that the appropriate server type, **'WebSphere Process Server v6.2'** or **'WebSphere ESB Server v6.2'**, is highlighted in the server type list

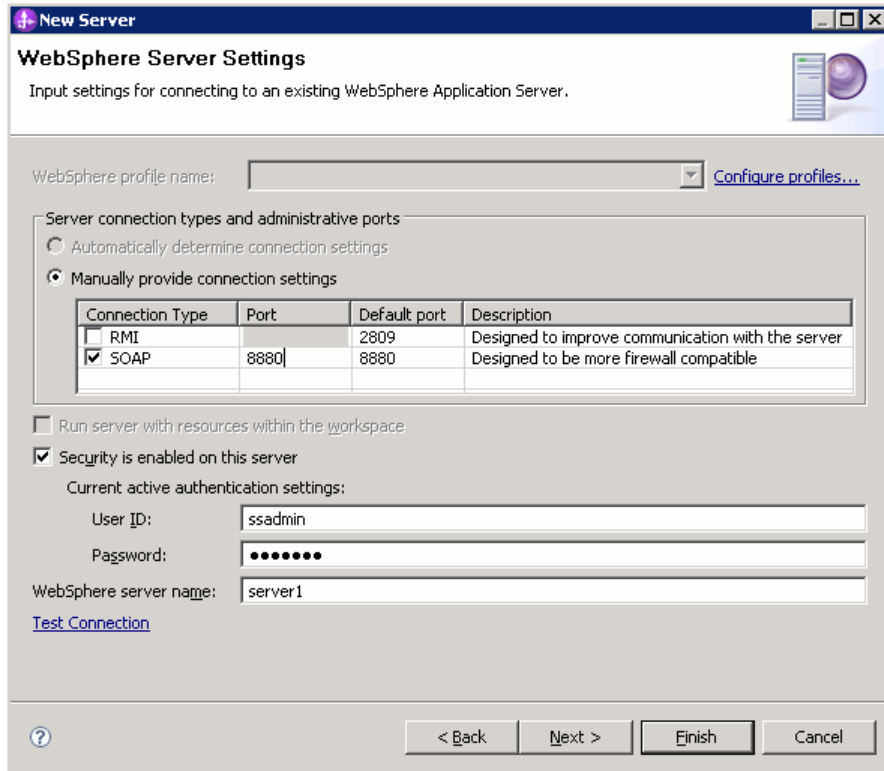


- \_\_\_ e. Click **Next**.



IBM WEBSHERE ADAPTER 6.2 – LAB EXERCISE

- \_\_\_ f. On the WebSphere Server Settings page, leave the radio button for **Manually provide connection settings** selected, and select the box for SOAP
- \_\_\_ g. Enter the correct setting (<SOAP\_PORT>) for **Port** column
- \_\_\_ h. If security is enabled on your server, select the box for ‘**Security is enabled on this server**’ and enter <USERID> for the user ID and <PASSWORD> for the password.



- \_\_\_ i. Click **Finish**.
- \_\_\_ j. The new server should be seen in the Server view.



- \_\_\_ 2. Start the remote server if it is not already started. WebSphere Integration Developer does not support starting remote servers from the Server view.
- \_\_\_ a. From a command prompt, telnet to the remote system if needed:

**'telnet <HOSTNAME> <TELNET\_PORT>'**

User ID : <USERID>

Password : <PASSWORD>

## IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

\_\_ b. Navigate to the bin directory for the profile being used:

**cd <WAS\_HOME>/profiles/<PROFILE\_NAME>/bin**

\_\_ c. Run the command file to start the server: **./startServer.sh <SERVER\_NAME>**

\_\_ d. Wait for status message indicating server has started:

```
ADMU3200I: Server launched. Waiting for initialization status
```

```
ADMU3000I: Server sssr01 open for e-business; process id is 0000012000000002
```