IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

FTP Adapter outbound lab

What this exercise is about

The objective of this lab is to provide you with an understanding of the IBM WebSphere Adapter for FTP and outbound processing. In this lab you will deploy the WebSphere Adapter for FTP, using WebSphere Integration Developer, and integrate it with an SCA application that processes outbound requests to 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 with latest fixes

- FTP server installed and configured

- Extract Labfiles62.zip to your C:\ (your root) drive

What you should be able to do

- Import FTP adapter RAR file into WebSphere Integration Developer

- Use the external service wizard to configure Activation Spec Properties, Resource Adapter Properties to generate Business Objects and other artifacts and then define your Data Binding and Data Handler, and Operations

- Deploy the adapter application onto the WebSphere Process Server test environment

- 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

Introduction
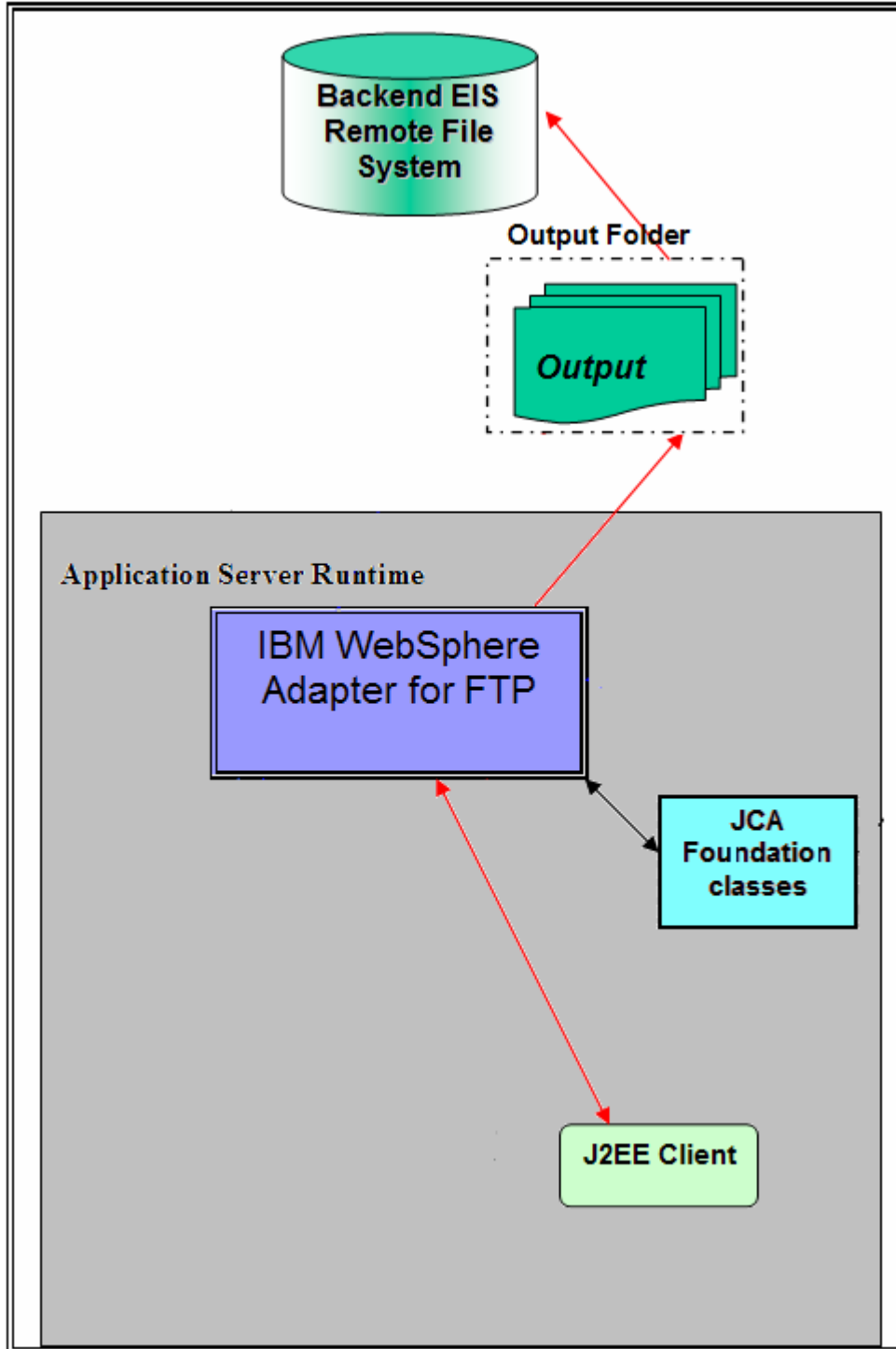
The Java™ EE Client makes a SCA call by giving the outbound operation name and the input data object and then the custom data binding is called and it invokes other content-specific data handler based on the ContentType set in the child data object.

The protocol specific properties like directory Name, file name which are required during outbound operation are populated in the wrapper data object present in BG. These values are set on to the FTPFileUnstructuredRecord in the custom data binding and sent to the adapter.

Exercise instructions

Some instructions in this lab might be specific for Windows platforms. If you run the lab on a platform other than Windows, you will need to run the appropriate commands, and use appropriate files (for example .sh in place of .bat) for your operating system. The directory locations are specified in the lab instructions using symbolic references as follows:

Reference variable	Windows location	Linux location
<WID_HOME>	C:\Program Files\IBM\WID62	
<WPS_HOME>	C:\<WID_HOME>\runtimes\bi_v62	
<FTPADAPTER_HOME>	<WID_HOME>\ResourceAdapters\FTP_6.2.0.0\deploy	
<LAB_FILES>	C:\Labfiles62	/tmp/Labfiles62
<WORKSPACE>	<LAB_FILES>\FTPOutbound\workspace	
<OUT_DIR>	<LAB_FILES>\ FTPOutbound\output	
<RETRIVE_ARCHIVE>	<LAB_FILES>\ FTPOutbound\retrievearchive	
<FTPFILES>	<LAB_FILES>\FTPFiles	
<LOCAL_DIR>	<LAB_FILES>\FTPOutbound\LocalDir	
<TEMP>	C:\temp	/tmp

Windows users: When directory locations are passed as parameters to a Java program such as EJBdeploy or wsadmin, you must 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**".

WBPMv62_AdapterFTPOutboundLab.doc

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 extract the lab files to your local system.

____ 1. Extract the provided Labfiles62.zip to your C:\ (root) drive, if you have not already done so. This creates 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
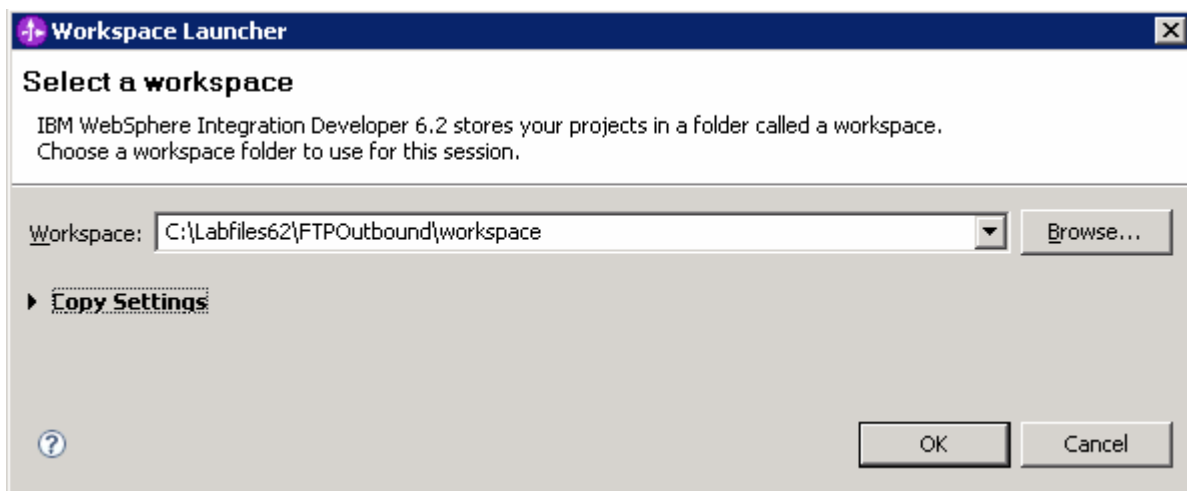
____ 3. Click the 📂 button on the right corner to close the Welcome page and proceed with the workbench

____ 4. Create directory structure on your FTP Server

　　__ a. Log onto your FTP Server using your ftp user and its password

　　__ b. Create an Output directory named **OutDir** under the user's home directory

　　　　1) **mkdir OutDir**

　　　　2) **mkdir RetrieveArchive**

Part 2: Review properties

This part of the lab will give you brief description of some the properties that are used in this lab at various stages.

- **New in V6.2 for Sequence File**: For each request, the adapter increments the number in the sequence file and the input type takes the sequence number that is currently stored in the sequence file. Sequence numbers are not maintained separately for different input data types.

  For compatibility with sequence files generated with previous versions of the adapter, where sequence numbers were maintained separately for different input data types, the adapter checks for all entries in the file that have the older format (<dirPath>/xyz.txt = 2, where xyz.txt is the file name and 2 is the sequence number to be used when the adapter receives another Create request on the same file). The adapter searches for all such sequence numbers for each input type and uses the highest sequence number as the sequence number for the next input type. The adapter then overwrites the entire file with the new (incremented) sequence number.

  Important: Unless they are part of a cluster, two adapter instances should not access the same sequence file, because this can result in delayed processing of batch requests.

  If the sequence file is deleted manually, the sequences are lost and will start from 1 again. You can also reset the sequence by changing the sequence value in the sequence file.

- **Default target file name**: This value if specified at Managed Connection factory level is used as default to create the new file. You can use this along with Sequence file.

- **chunkFileName:** this is populated during Inbound or on Retrieve operation during outbound and the presence of this indicates that it is a chunked file. This is used for Inbound and Retrieve outbound operations where chunking is enabled.

- **fileContentEncoding:** This encoding is used while writing to the file. If this property is not specified, the RA tries to write without using any specific encoding. You can specify any Java supported encoding set like UTF-8 for this attribute. If the file content is non-English, the corresponding encoding needs to be chosen so the adapter uses the encoding while writing to the file system.

- **includeEndBODelimiter:** This is used during the outbound Create/Append/Overwrite operations. The File content is appended with the value of IncludeEndBODelimiter. For example, if the operation chosen is Append and the specified values for this property is ####, when the BO content is written to the file, the include BO Delimiter content is also appended as part of the BO content at the end of the file.

- **stagingDirectory:** This directory is used only for the create and overwrite operations. The file is written to the staging directory completely and then just renamed programmatically to the Output directory specified in the directoryPath

- **generateUniqueFile:** During outbound Create operation the adapter creates a unique file when this property is true. When this property is set to true the adapter ignores any value set for file name property.

  The name of the unique file generated by FTP adapter will have this format:

  A random number prefixed by '**ftpa**' and with an extension '**.tmp**'. For example, ftpa23423.tmp

---

Note: If the **Sequence file** has a value and an output file name is specified, with the 'generateUniqueFile' is set, the 'generateUniqueFile' property takes precedence.

- **createFileIfNotExists:** During Append and Overwrite operations, if the file does not exist, then the adapter creates the file when this property is set to true. If this property is false and file does not exist then the adapter flags an error.

  And while creating file for this condition, if 'generateUniqueFile' is also set to true, then the adapter generates a unique file. At this time the adapter ignores the value present in file name property. If the file to be appended does not exist and this property is set to false, a RecordNotFoundException is thrown to the calling component.

- **splitFunctionClassName:** This value takes a fully qualified class name of the class to be used in order to split the retrieved file during outbound retrieve operation. It takes two values as of now:

  com. ibm.j2ca.utils.filesplit.SplitBySize - a class which splits the file based on file size

  com.ibm.j2ca.utils.filesplit.SplitByDelimiter - a class which splits the file based on delimiter (used to separate BO's in event file)

  The delimiter or file size is given in SplitCriteria.

  If RetrieveContentType is null, then this is automatically set to class name which does splitting based on size.

- **splitCriteria:** This attribute takes different values based on value set in splitFunctionClassName.

  If splitFunctionClassName is set to com. ibm.j2ca.utils.filesplit.SplitByDelimiter, then splitCriteria must contain the delimiter which separates the BO's in the retrieved file.

  If splitFunctionClassName is set to com. ibm.j2ca.utils.filesplit.SplitBySize, then splitCriteria must contain a valid number which represents the size in bytes. If retrieved file size is greater than this value, it is split into chunks of this value and so many chunks are posted. If file size is less than this value the entire event file is posted in one shot. When SplitCriteria=0, chunking is disabled.

- **deleteOnRetrieve:** If this property is set to true, during Retrieve operation, after the file content is retrieved, the file is deleted from the directory on the file system.

- **archiveDirectoryforDeleteOnRetrieve:** If the deleteOnRetrieve property is set to true, the adapter will optionally archive (if this directory is valid) the file to this folder before it is deleted.

Part 3: Pass through scenario

Outbound support can be broadly classified into two flows, one that involves data transformation and another without it (pass-through). The incoming BO can be a content specific BO or a generic FTP BO. This part of the lab deals with the pass through:
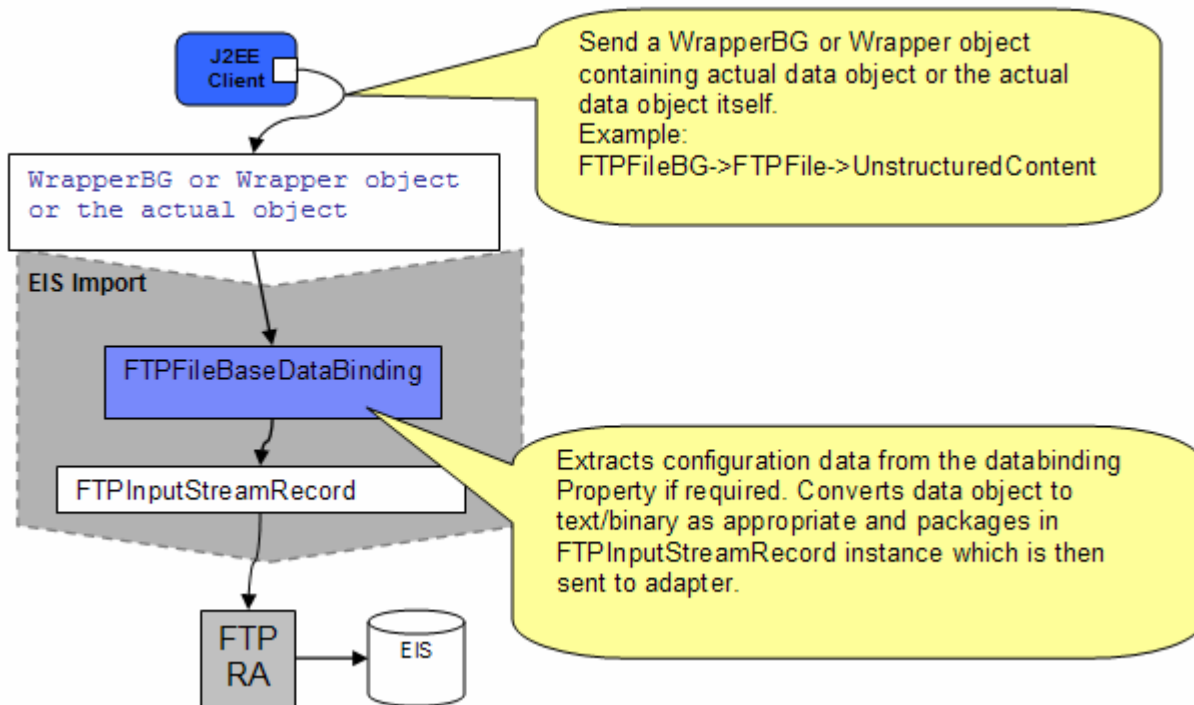

- In the Java EE client, if you intend to use business graph (FTPFileBG) along with the wrapper data object (FTPFile) present in the WrapperBG, business graph is populated for protocol specific information. The "Content" attribute of FTPFile is populated with an UnstructuredContent BO. The wrapper data object and the outbound operation name are sent as input by making an SCA call (or).

- If you intend to use the wrapper data object (FTPFile), it is populated with protocol specific information and the "Content" attribute of FTPFile is populated with an UnstructuredContent BO. The wrapper data object and the outbound operation name are sent as input by making an SCA call (or).

- Based on the Data Binding (FTPFileBaseDataBinding) configured while running the external service wizard, that particular data binding is called and it gets the actual data object either from wrapper data object if the wrapper data object is used or from the wrapper data object present in the business graph if the business graph is used or the data object itself if neither wrapper or business graph is used.

- The FTPFileBaseDataBinding recognizes the content as Unstructured and does a passThrough. It just instantiates the FTPInputStreamRecord, sets the actual content (as input stream) and optionally sets the protocol specific information when user needs wrapper.

- This Record object is sent to the adapter and the outbound operation is executed. The output of the outbound operation is an FTPStructuredRecord which is sent back to the configured Data Binding (FTPFileBaseDataBinding) for List, Exists and Retrieve operations. For Create, Append, OverWrite and ExecuteFTPScript, if output required is true, then the FTPStructuredRecord is sent back to the configured Data Binding (FTPFileBaseDataBinding) and a data object containing the output is sent back to the Java EE client.

3.1.　　Configure pass through using the external service wizard

In this part, you will use this new external service feature to create and configure the data binding, Operations, which generates the business objects and other artifacts.
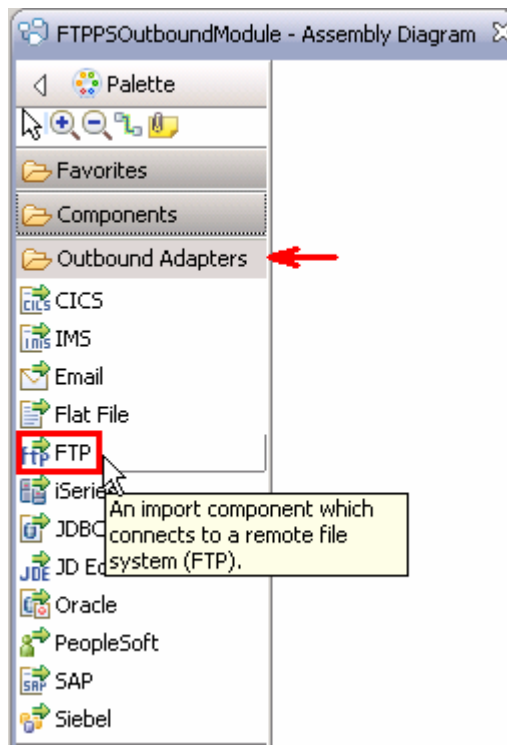
____ 1.　Create FTPPSOutboundModule

　__ a. From the Business Integration window, right-click and select **New > Module**

　__ b. From the New Module window, enter **FTPPSOutboundModule** for the Module Name

　__ c. Ensure that the box next to **Open module assembly diagram** is checked and then click **Finish**
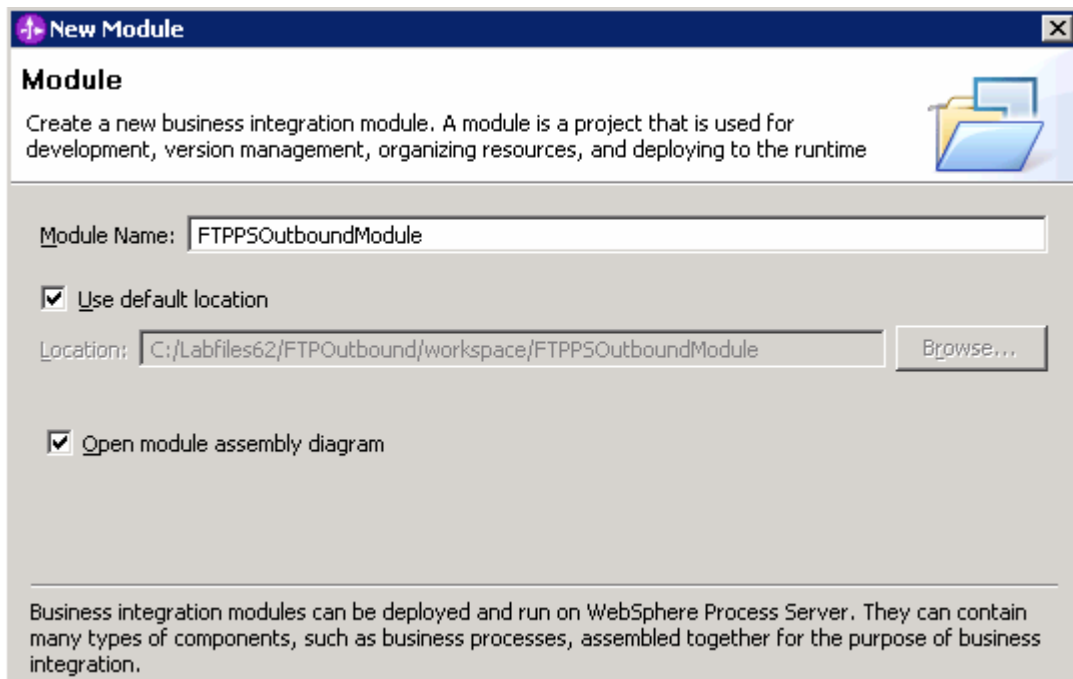
You will now see a new module, FTPPSOutboundModule, created in your Business Integration window

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

____ 2.   To start the external service from the Palette:

__ a. From the **Palette** on the left side of Assembly Diagram, click **Outbound Adapters**:

__ b. Under Outbound 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)**
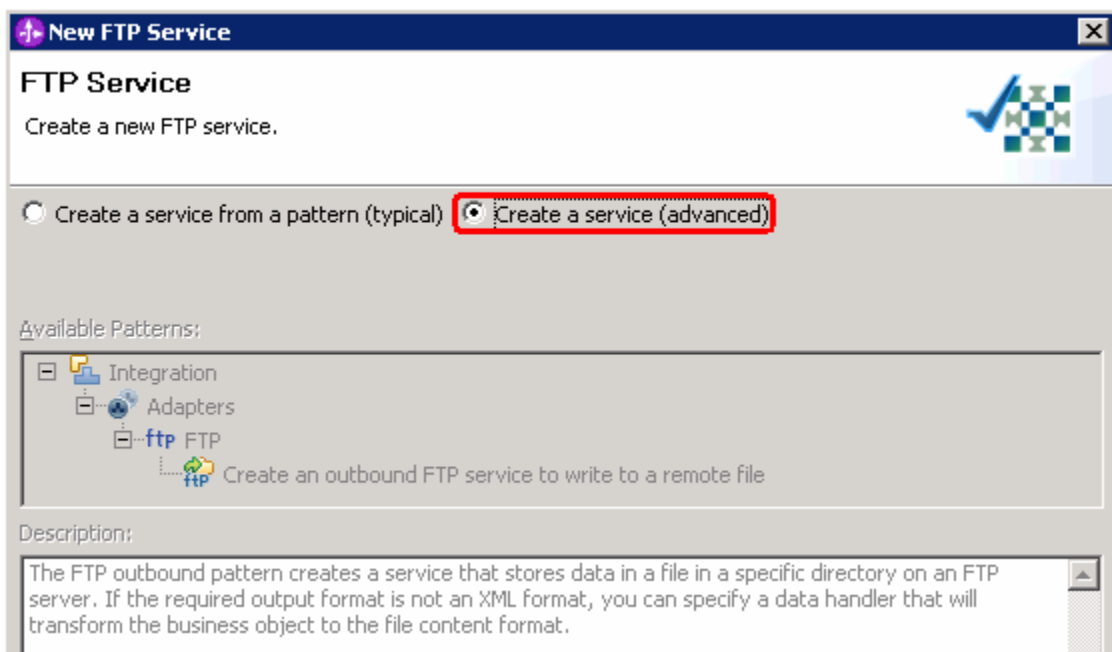
__ a. Click **Next**

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 the **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**
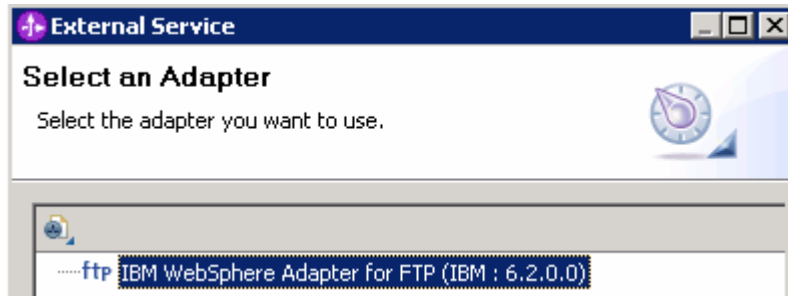
____ 5.   Adapter Import screen:

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
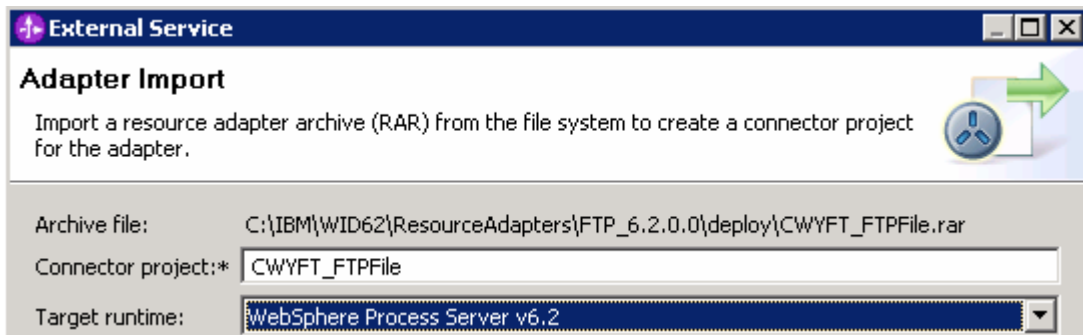
__ d. Click **Next**

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 **Outbound** and click **Next** to proceed to the next step.

____ 6.   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) Directory: **full path of the OutDir created in on the machine where FTP server is existing** (for Ex: /home/wsbeta/OutDir)

Note: This is the folder where the adapter will create the file. 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) 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.**

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

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

6) 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:
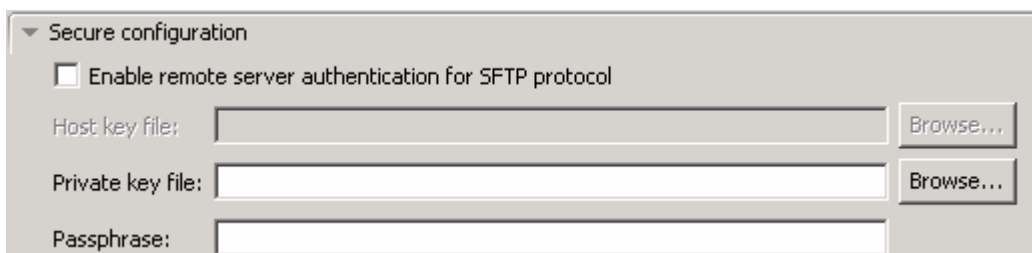

__ d. Click **Advanced connection configuration**

1) Click **Browse…** next to **Sequence file** and enter any location. For ex:
   <LABFILES>\FTPOutbound\SequenceFile.txt

Note: Refer to Review Properties part of this lab for the detailed explanation of this new feature. This file should already be created at the specified location.
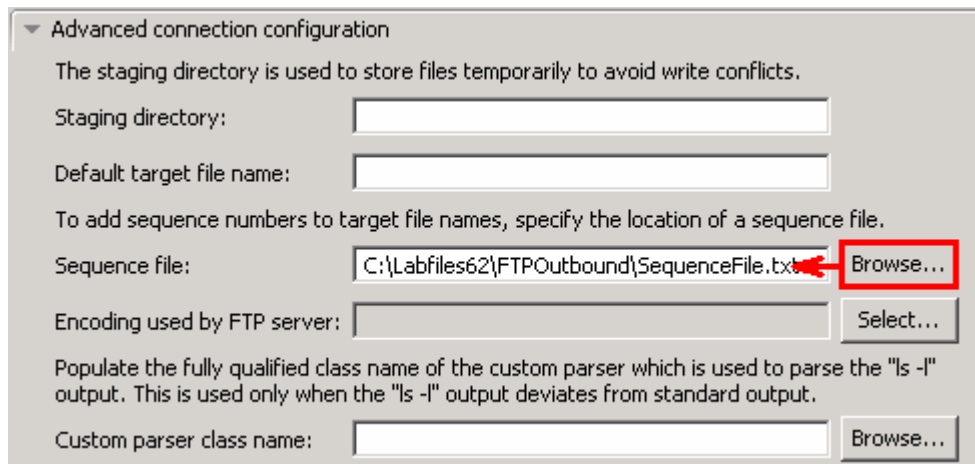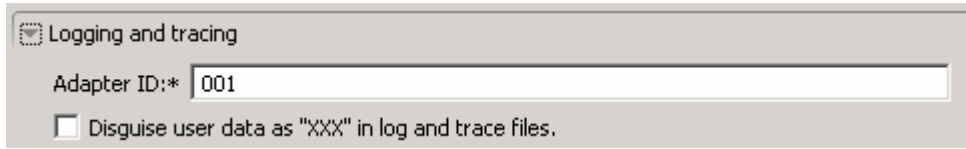

____ 7.   **Secure configuration**: Refer to the new lab '**Install and configure SSH server**' for more details on this new feature

____ 8. **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
>
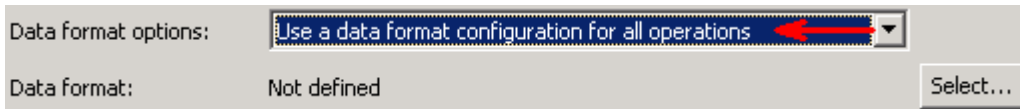> ☐ Disguise user data as "XXX" in log and trace files.

____ 9. 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**.
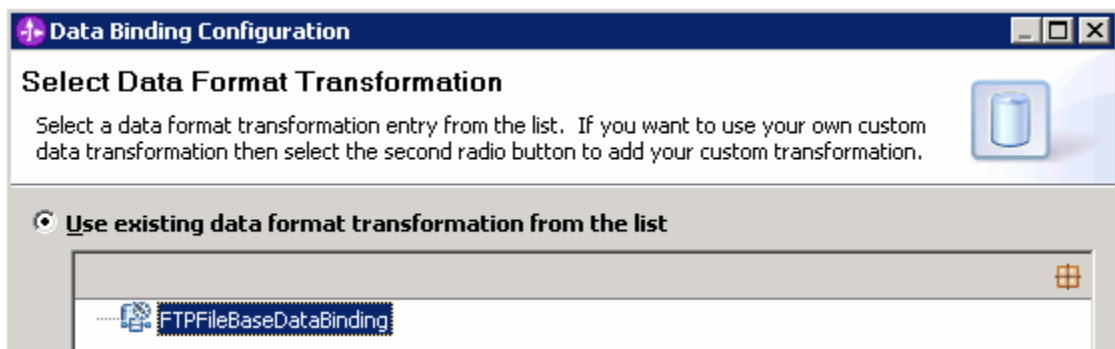
> Service properties
>
> ☐ Specify a Java Authentication and Authorization Services (JAAS) alias security credential.
>
> J2C authentication data entry:

____ 10. You can define data binding in two places - service level (current screen of the 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**'

> Data format options: Use a data format configuration for all operations   ◄ ▼
>
> Data format: Not defined   Select...

     __ 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**
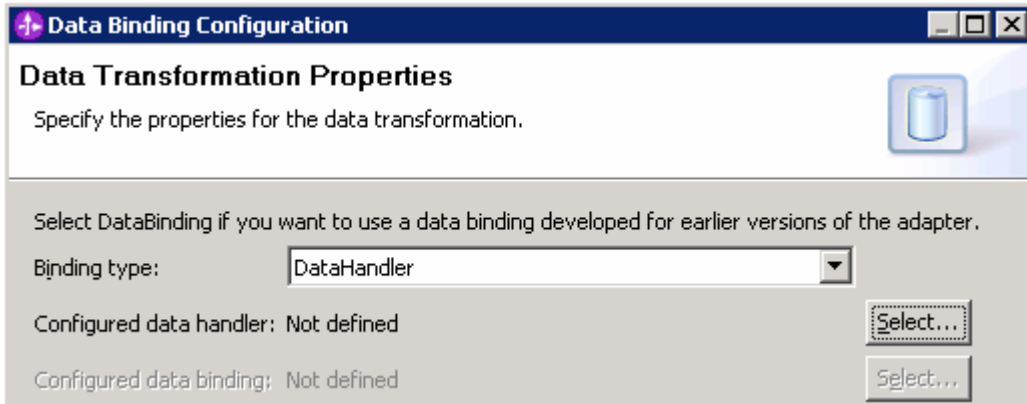
> **Data Binding Configuration**   _ □ ✕
>
> **Select Data Format Transformation**
>
> Select a data format transformation entry from the list. If you want to use your own custom data transformation then select the second radio button to add your custom transformation.
>
> ⊙ **Use existing data format transformation from the list**
>
>   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.
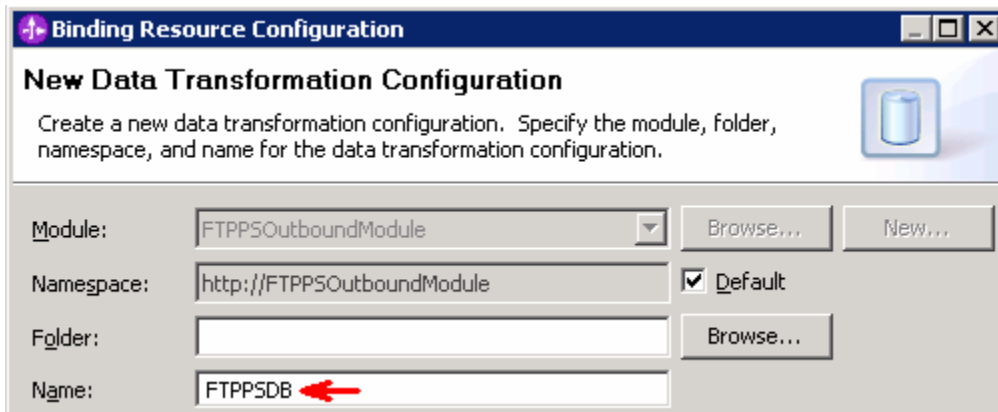

__ e. Click **Next** from the Data Transformation Properties screen
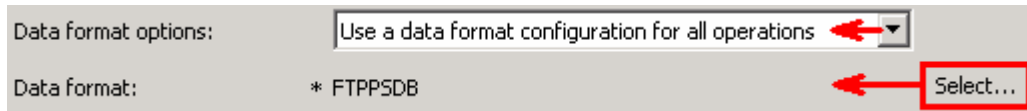

__ f. Note that the selected module is **FTPPSOutboundModule**

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

2) Click **Finish**

__ g. Now the **FTPPSDB** should be displayed for Data binding configuration

____ 11. 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.
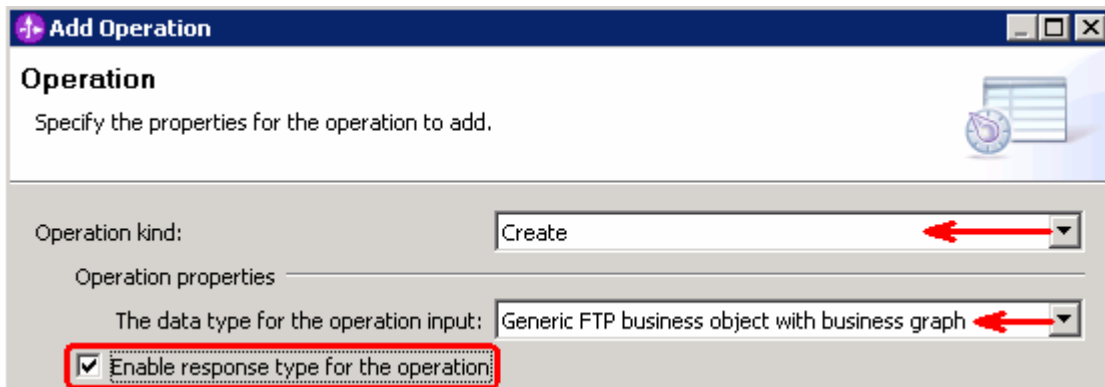
__ a. Click **Next**

Define Operations: In this screen, you will add the required operations that is supported by the adapter functions on the remote file system

---

Note: The precedence of the parameters is as follows: WrapperBO, Interaction Spec, and Managed Connection Factory. The adapter will first search for the parameters passed in the WrapperBO; if it is not available there, it will then subsequently search in the Interaction Spec, and then the Managed Connection Factory instance. **In this lab, for all the operations, you will enter the values at the WrapperBO level in the later part using the WebSphere Integration Developer test client.**

WBPMv6.2_AdapterFTPOutboundLab.doc

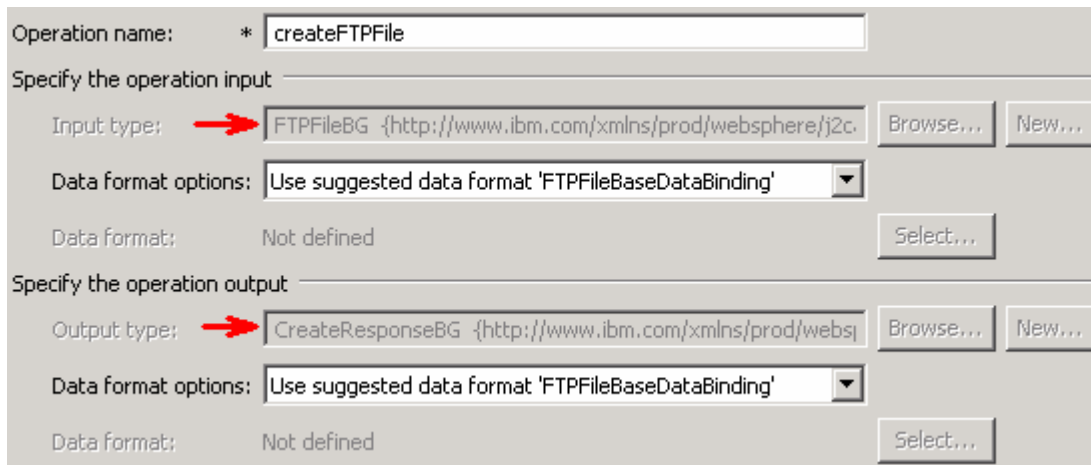**Define Operation: createFTPBG**

_____ 12. Click **Add…** to open Add Operation window

　　__ a. For **Operation kind**, select **Create** from the drop down list

　　__ b. For **Data type for operation**, select **Generic FTP business object with business graph** from the drop down list

　　__ c. Select the check box next to **'Enable response type for the operation'**

　　__ d. Click **Next**

The Data type for input and output are populated based on the selection of the Data type for the operation in the previous step. Since you have chosen Generic FTP business object with business graph, the **Input type** is **FTPFileBG** and because you have selected Output required box, the **Output type** is **CreateResponseBG**

　　__ e. For Operation name, enter **createFTPBG**

Define Data format for **input**:

　　__ f. For **Data format options**, select **Use a data binding configuration** from the dropdown list

　　__ g. Click **Select…** next to **Data format**. A Binding Resource Configuration window is opened

　　__ h. Ensure that the radio button for 'Use existing data format transformation from the list' and then select **FTPFileBaseDataBinding > FTPPSDB**
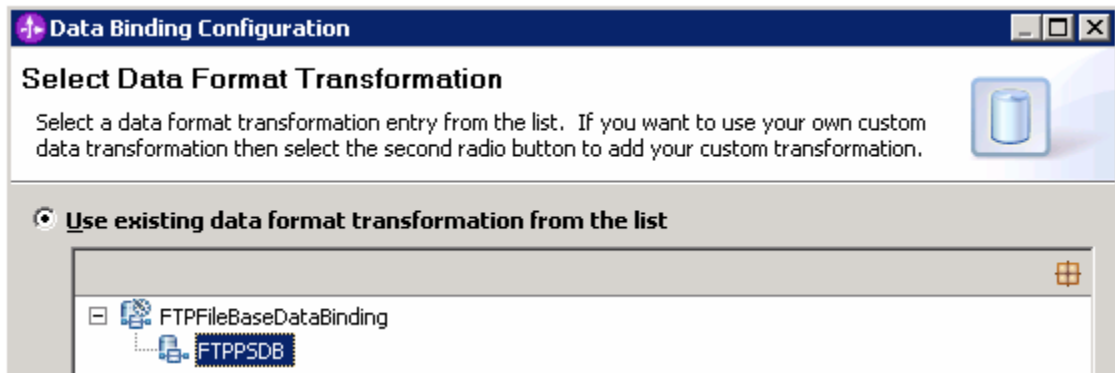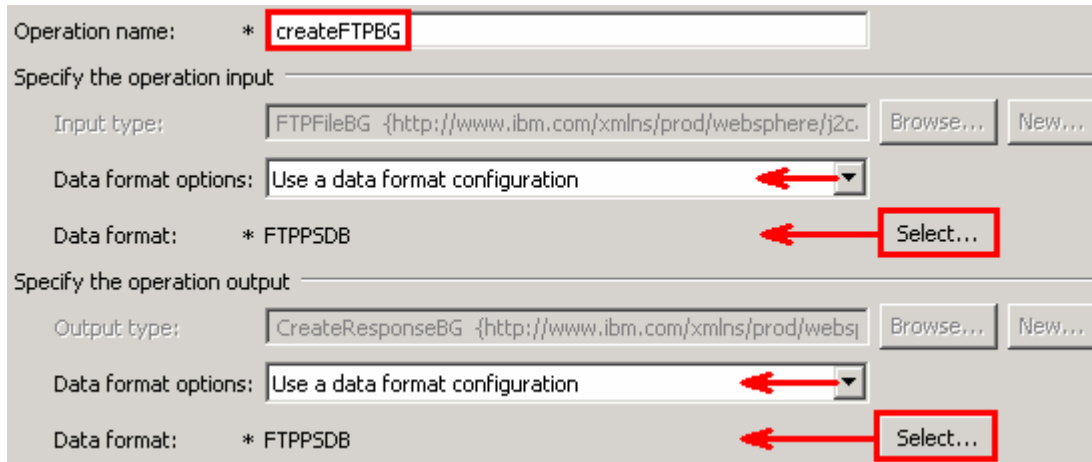
__ i. Click **Finish**

Define Data format for **output**:

__ j. Repeat the steps that you did to define the data format for input and select **FTPPSDB**

__ k. The Operation screen now should look like this:

__ l. Click **Finish** from the Add Operation window. The operation, createFTPBG, will now be displayed under Operations list

____ 13. You can click **Advanced >>** under 'InteractionSpec properties for createFTPBG' to review the properties available at Interaction spec level
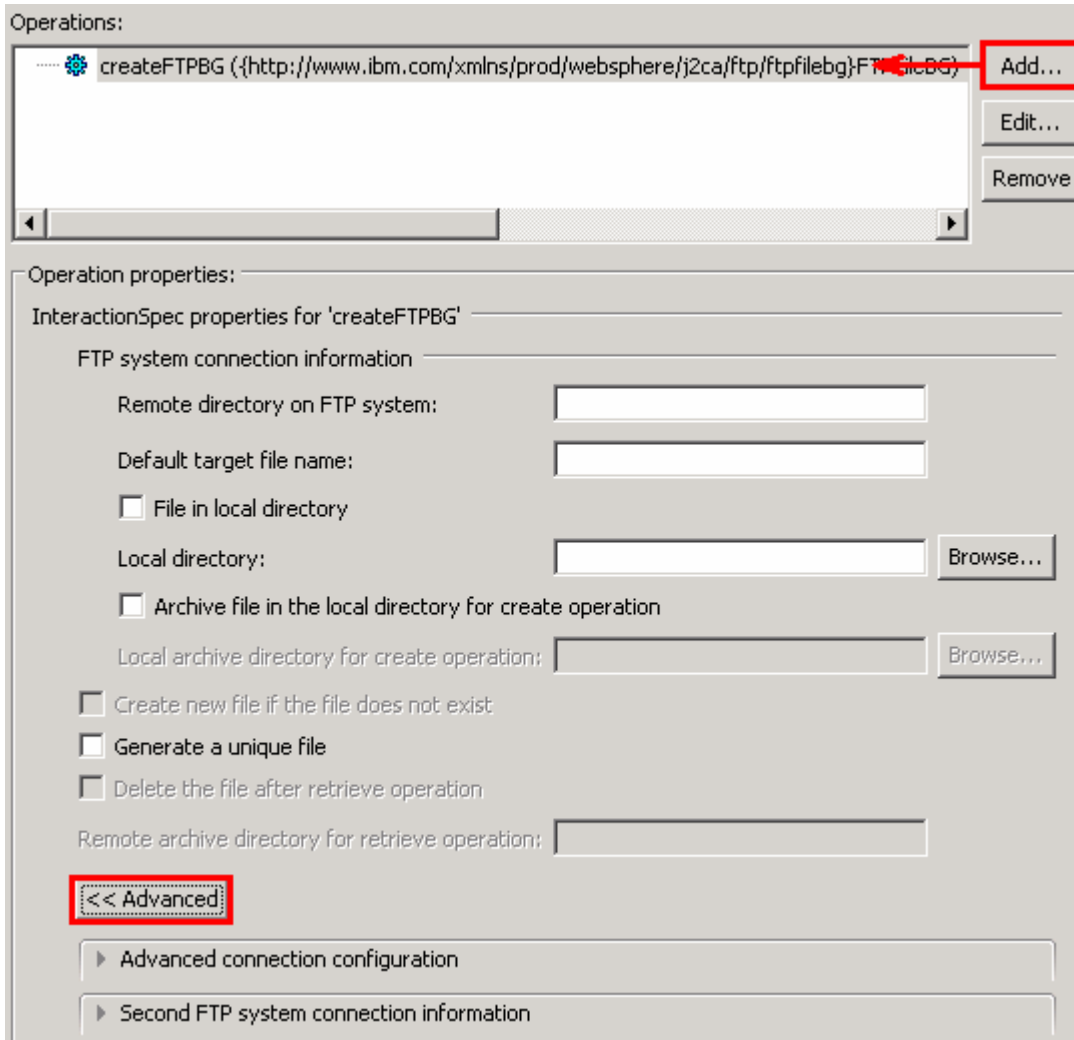
Define Operation: appendFTPBG

____ 14. Click **Add…** to open Add Operation window

__ a. For **Operation kind**, select **Append** from the drop down list

__ b. For **Data type for operation**, select **Generic FTP business object with business graph** from the drop down list

__ c. Select the check box next to **'Enable response type for the operation'**
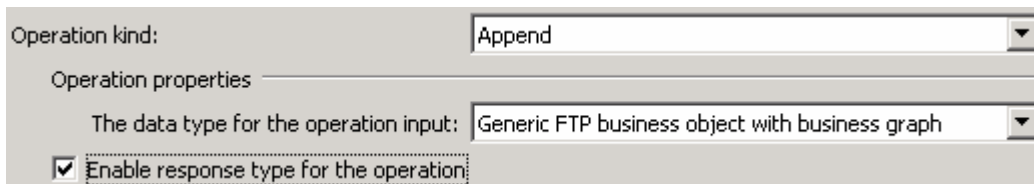
__ d. Click **Next**

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

The Data type for input and output are populated based on the selection of the Data type for the operation in the previous step. Since you have chosen Generic FTP business object with business graph, the **Input type** is **FTPFileBG** and because you have selected Output required box, the **Output type** is **AppendResponseBG**

Operation name: *	appendFTPFile
Specify the operation input	
Input type: →	FTPFileBG {http://www.ibm.com/xmlns/prod/websphere Browse... New...
Data format options:	Use suggested data format 'FTPFileBaseDataBinding' ▼
Data format:	Not defined Select...
Specify the operation output	
Output type: →	AppendResponseBG {http://www.ibm.com/xmlns/prod/ Browse... New...
Data format options:	Use suggested data format 'FTPFileBaseDataBinding' ▼
Data format:	Not defined Select...

___ e. For Operation name, enter **appendFTPBG**

Define Data format for **input**:

___ f. For **Data format options**, select **Use a data binding configuration** from the dropdown list

___ g. Click **Select…** next to **Data format**. A Binding Resource Configuration window is opened.

___ h. Ensure that 'Use existing data transformation from the list', select **FTPFileBaseDataBinding > FTPPSDB** and click **Finish**

Define Data format for **output**:

___ i. Repeat the steps you did to define data binding for input and define **FTPPSDB** for output

___ j. The Operation screen now should look like this:

Operation name: *	appendFTPBG
Specify the operation input	
Input type:	FTPFileBG {http://www.ibm.com/xmlns/prod/websphere Browse... New...
Data format options:	Use a data format configuration ← ▼
Data format: *	FTPPSDB ← Select...
Specify the operation output	
Output type:	AppendResponseBG {http://www.ibm.com/xmlns/prod/ Browse... New...
Data format options:	Use a data format configuration ← ▼
Data format: *	FTPPSDB ← Select...

___ k. Click **Finish** from the Add Operation window

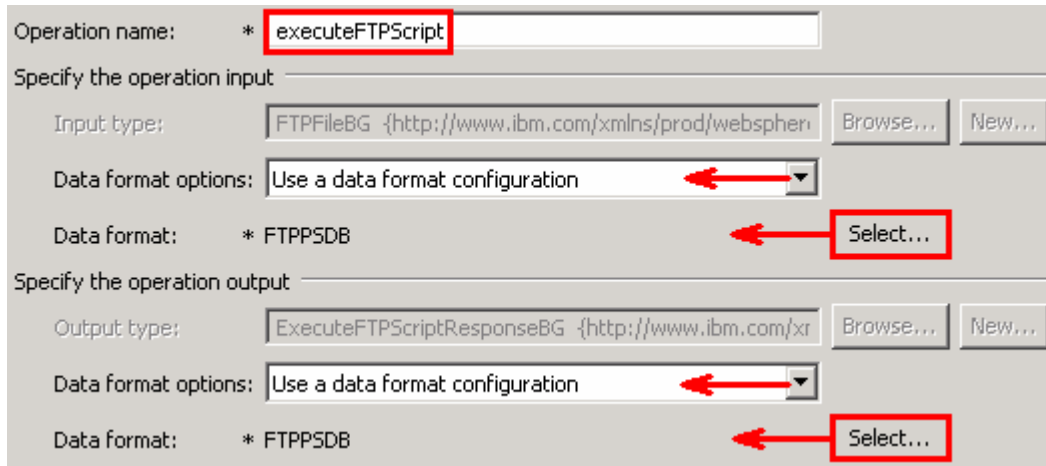The operation, appendFTPBG, will now be displayed under Operations list.

____ 15. You can click **Advanced >>** under 'InteractionSpec properties for appendFTPBG' to review the properties available at Interaction spec level
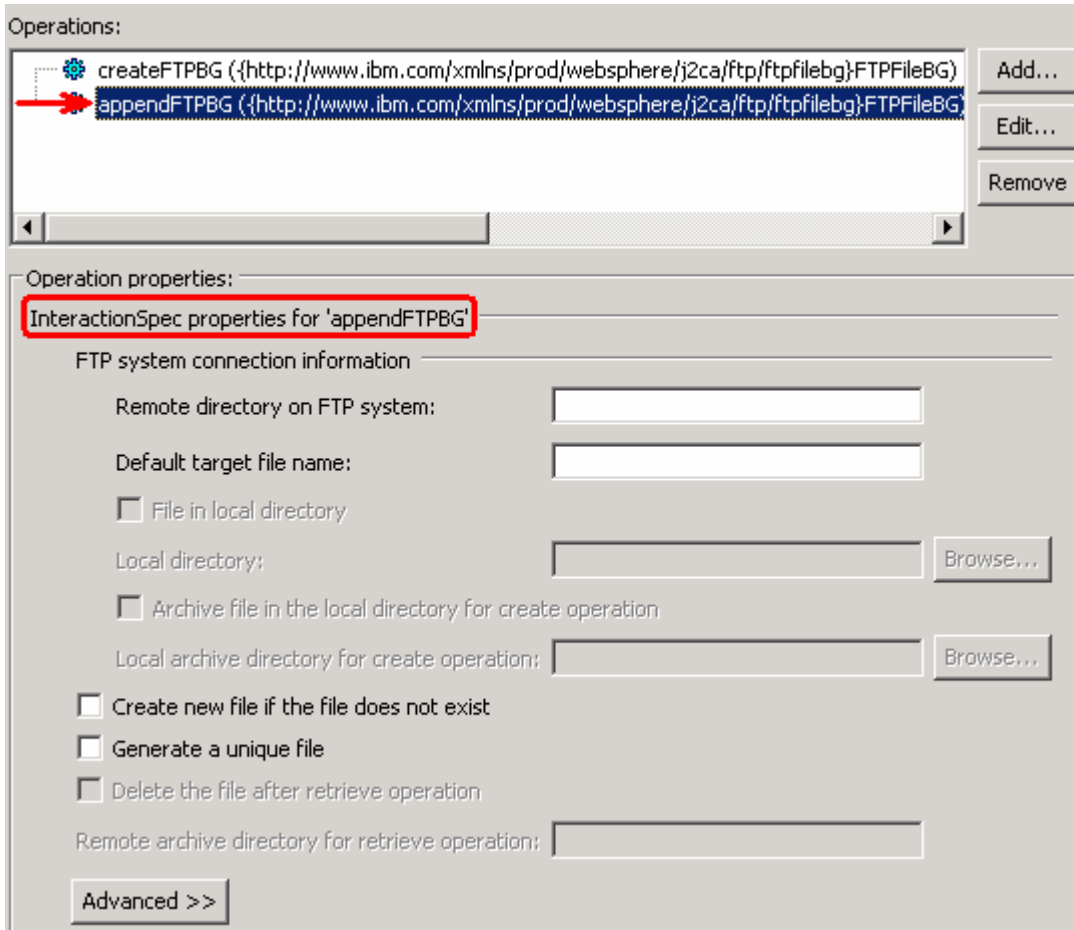
Define Operation: listFTP

____ 16. Click **Add…** to open Add Operation window

___ a. For **Operation kind**, select **List** from the drop down list

___ b. For **Data type for operation**, select **Generic FTP business object** from the drop down list

___ c. Note that the check box next to '**Enable response type for the operation'** is selected by default
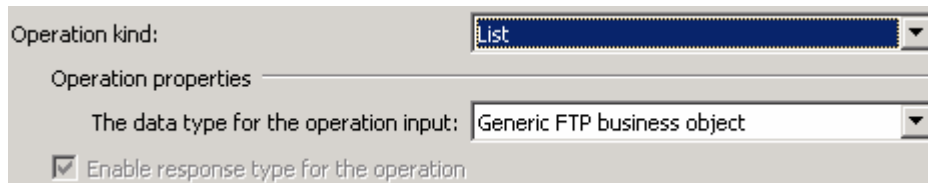
___ d. Click **Next**

The Data type for input and output are populated based on the selection of the Data type for the operation in the previous step. Since you have chosen Generic FTP business object, the **Input type** is **FTPFile** and because Output required box is selected by default, the **Output type** is **ListResponse**

__ e. For Operation name, enter **listFTP**

Define Data format for **input** and **output**:

__ f. Repeat the steps you did for Create or Append operation to define the data format and select **FTPPSDB** for both **input** and **output**

__ g. The Operation screen now should look like this:

__ h. Click **Finish** from the Add Operation window

The operation, listFTP, will now be displayed under Operations list.

__ i. You can click **Advanced >>** under 'InteractionSpec properties for listFTP' to review the properties available at Interaction spec level

Define Operation: retrieveFTP

_____ 17. For **Operation kind**, select **Retrieve** from the drop down list

__ a. For **Data type for operation**, select **Generic FTPFile business object** from the drop down list

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

__ b. Note that the check box for **'Enable response type for the operation'** is selected by default

__ c. Click **Next**

The Data type for input and output are populated based on the selection of the Data type for the operation in the previous step. Since you have chosen Generic FTPFile business object, the **Input type** is **FTPFile** and because the Output required is also selected, the **Output type** is **RetrieveResponseWrapper**. Also note that the Output type can be modified. But, for the pass through scenario, the output type is going to be the default, RetrieveResponseWrapper.

__ d. For Operation name, enter **retrieveFTP**

Define Data format for **input**:

__ e. Repeat the steps you did for Create or Append operation to define the data format and select **FTPPSDB** for both **input**

Define Data format for **output**:

__ f. Accept the default selection, **Use data format configuration 'FTPPSDB'**, from the drop down list

__ g. The Operation screen now should look like this:

__ h. Click **Finish** from the Add Operation window

The operation, retrieveFTP, will now be displayed under Operations list.

____ 18. You can click **Advanced >>** under 'InteractionSpec properties for retrieveFTP' to review the properties available at Interaction spec level

Define Operation: executeFTPScript

____ 19. Repeat steps under '**Define Operation: appendFTPBG**' (Step 14) of this part to with these inputs:

__ a. Operation kind (Step 14.a): **ExecuteFTPScript**

__ b. Operation name (Step 14.e): **executeFTPScript**

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

__ c. You should see this:

__ d. Click **Finish** from the Add Operation window

You have now defined four operations:

The operation, executeFTPScript, is now displayed under Operations list. So far, you have defined five operations:
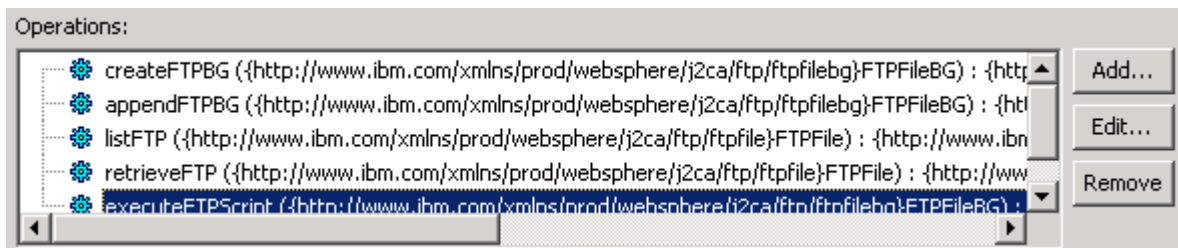
__ e. You can click **Advanced >>** under 'InteractionSpec properties for executeFTPScript' to review the properties available at Interaction spec level

__ f. Click **Next** from the Operations window

____ 20. From the Generate Artifacts screen, enter these:

__ a. For **Name**, enter **FTPPSOutboundInterface**

WBPMv6.2_AdapterFTPOutboundLab.doc
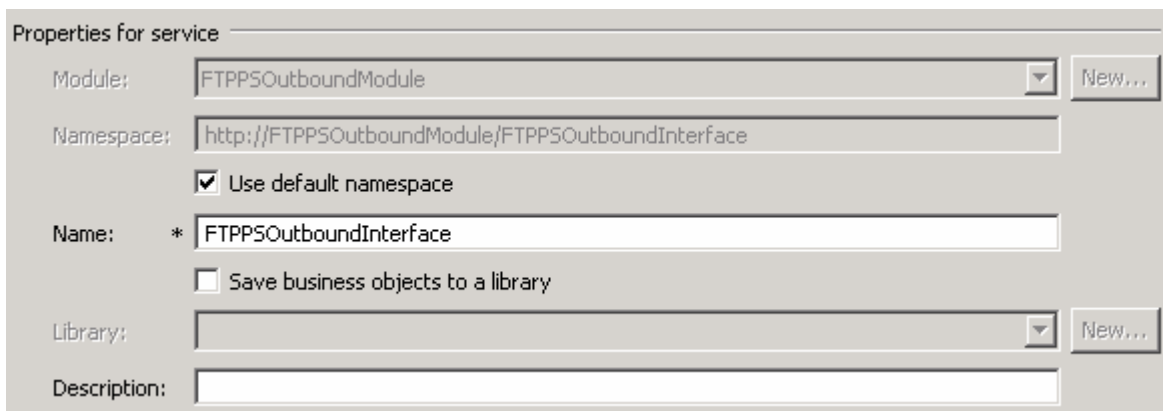
__ b. Click **Finish**

____ 21. You will now see a new import component, **FTPPSOutboundInterface** in the assembly diagram of FTPPSOutboundModule
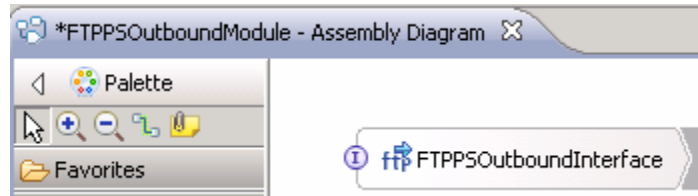
__ a. Save (**Ctrl+S)** your changes to the assembly diagram

____ 22. Review the FTPPSOutboundModule: The generated **Data Types**, **Interface**, and the Data binding (**FTPPSDB**) under Configured Resources can be found inside FTPPSOutboundModule
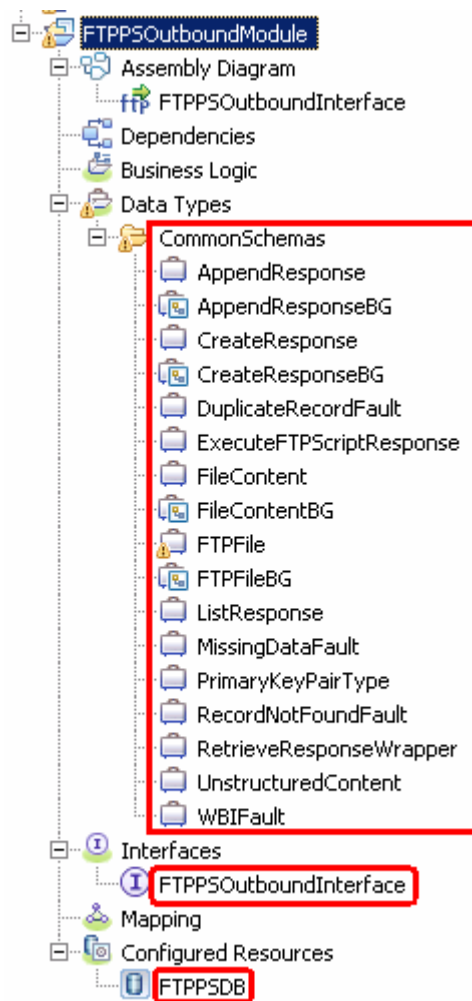
You can open each of these generated artifacts and business objects and review the properties inside.

Review the created methods inside the interface:

__ a. From the Business Integration view, expand FTPPSOutboundModule > Interfaces and then double-click **FTPPSOutboundInterface** to open it

__ b. You should see these five operations:

__ c. Close the interface, FTPPSOutboundInterface

WBPMv6.2_AdapterFTPOutboundLab.doc

3.2. Test pass through scenario

____ 1.   Start WebSphere Process Server (if not started already)

  __ a. From the **Servers** view of WebSphere Integration Developer, right click **WebSphere Process Server v6.2** and select Start from the pop-up menu

  __ b. Wait until the server status shows as **Started**

____ 2.   Add the project to the WebSphere Process Server Test Environment

  __ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects…** from the pop-up menu

  __ b. In the Add and Remove Projects window, select the **FTPPSOutboundModuleApp** project from the Available projects panel

  __ c. Click **Add >** to add it to the Configured projects panel

  __ d. The project is now moved to Configured projects. Click **Finish**

Wait for the project to be published to the server and you can confirm this by seeing this message in the console messages:

____ 3.   Open the test client for the module

  __ a. From the Business Integration perspective, right-click the **FTPPSOutboundModule** and select **Test > Test Module**

  __ b. The **FTPPSOutboundModule_Test** window is opened in the Assembly editor

You have five operations that were defined in the previous part in this module:

- createFTPBG

- appendFTPBG

- listFTP

- retrieveFTP

- executeFTPScript

Test Create operation:

____ 4.   Under **Detailed Properties**, for the **Operation** field, select **createFTPBG** from the drop down menu

Fill out the fields for Initial request parameters:

  __ a. For **DirectoryPath**, enter full path of the **OutDir** that is already created in your FTP server

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

__ b. For **Filename**, enter any name, for Ex: PassthroughTest.txt
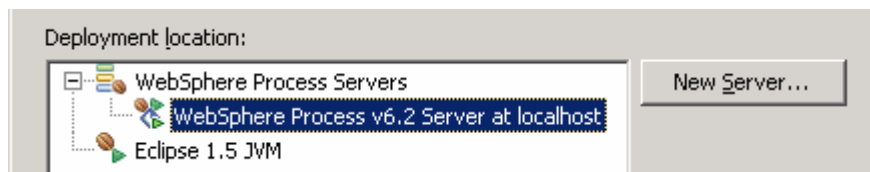

__ c. Under **Content**, for **AsText**, enter some test message, for Ex: **Testing Create operation for pass through**


__ d. Click **Continue** button under Events

__ e. From Deployment Location window, select **WebSphere Process Servers > WebSphere Process Server v6.2 at localhost** and click **Finish**


__ f. Provide Administrator User ID and Password

WBPMv6.2_AdapterFTPOutboundLab.doc

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

1) Optionally, select the box 'Use the authentication settings in the preference and never ask again' so that you do not have to enter the credentials next time when you start the test client
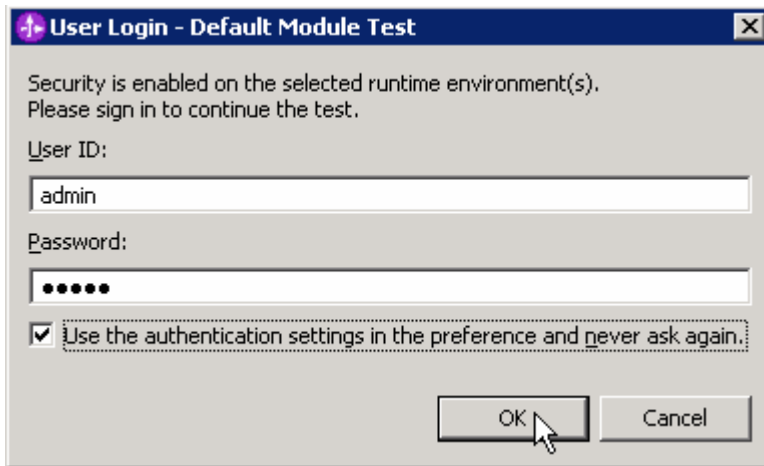
__ g. Click **OK** from Adding/Removing Projects pop-up. Optionally, you can select 'Do not show again' so that you are not required to do this next time when you start the test client
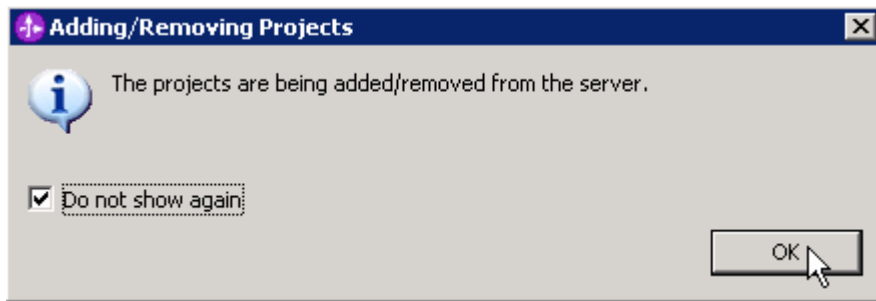
__ h. You should see a window similar to this, that contains the data you just entered in the previous steps:

Note: Since you have specified a Sequence File name while running the external service, the file name is created with '1' appended to it.

____ 5.   Verify the created file and its contents

__ a. You will see a new file, **PassThroughTest.1.txt** created under **OutDir** on your FTP server. Open that file to see the content

```
Testing Create operation for pass through
~
~
~
~
"PassThroughTest.1.txt" [noeol] 1L, 41C
```

__ b. Now browse to <LAB_FILES>\FTPOutbound and open the **SequenceFile.txt** file. You should see a numeric entry in the file. Each time the file is created, the adapter increases the number by one.

Test Append operation.

____ 6. Click **Invoke** ( ) under Events to start a new event

____ 7. Under **Detailed Properties**, for the **Operation** field, select **appendFTPBG** from the drop down menu

Fill out the fields for Initial request parameters:

__ a. For **DirectoryPath**, enter full path of the **OutDir** that is already created in your FTP server

__ b. For **Filename**, enter **PassThroughTest.1.txt** (the file created during the above Create operation)

▸ **General Properties**

▾ **Detailed Properties**

Configuration:	Default Module Test
Module:	FTPPSOutboundModule
Component:	FTPPSOutboundInterface
Interface:	FTPPSOutboundInterface
Operation:	appendFTPBG

Initial request parameters

Name	Type	Value
appendFTPBGInput	FTPFileBG	✔
verb	verb<string>	✔ CREATE
FTPFile	FTPFile	✔
DirectoryPath	string	✔
Filename	string	✔ PassThroughTest.1.txt

__ c. Under **Content**, for **AsText**, enter some test message, for Ex: **Test message for Append operation**

⊟ Content	UnstructuredContent	✔	
ContentType	string	✔	
ObjectName	string	✔	
AsText	string	✔	Test message for append operation
AsBinary	hexBinary	✔	00

__ d. Click **Continue** button under Events

__ e. From Deployment Location window (if opens), select **WebSphere Process Servers > WebSphere Process Server v6.2** and click **Finish**

__ f. You should see a window similar to this, which contains the data you just entered in the previous steps:

Events
⊟ Invoke (FTPPSOutboundInterface:createFTPBG)
⊟ Invoke started
Invoke (FTPPSOutboundInterface:createFTPBG)
Return (FTPPSOutboundInterface:createFTPBG)
Invoke returned
⊟ Invoke (FTPPSOutboundInterface:appendFTPBG)
⊟ Invoke started
Invoke (FTPPSOutboundInterface:appendFTPBG)
Return (FTPPSOutboundInterface:appendFTPBG)
Invoke returned

▸ **General Properties**

▾ **Detailed Properties**

Module: FTPPSOutboundModule

Component: FTPPSOutboundInterface

Interface: FTPPSOutboundInterface

Operation: appendFTPBG

Return parameters:

Name	Type	
⊟ appendFTPBGOutput	AppendResponseBG	✔
verb	string	✔
⊟ AppendResponse	AppendResponse	✔
Filename	string	✔ PassThroughTest.1.txt

____ 8.   Verify the created file and its contents

__ a. Browse to the **OutDir** on your FTP server

__ b. You will see a new file, **PassThroughTest.1.txt** created under that directory. Open the file and observe the appended content

```
Testing Create operation for pass through Test message for Append operation
~
~
~
~
"PassThroughTest.1.txt" [noeol] 1L, 74C
```

Test List operation:

____ 9.   Click **Invoke** ( ) under Events to start a new event

____ 10. Under **Detailed Properties**, for the **Operation** field, select **listFTP** from the drop down menu

____ 11. Fill out the fields for Initial request parameters:

 __ a. For **DirectoryPath**, enter full path of the **OutDir** that is already created in your FTP server

 __ b. You can leave all the other fields empty:

Detailed Properties

Configuration:	Default Module Test
Module:	FTPPSOutboundModule
Component:	FTPPSOutboundInterface
Interface:	FTPPSOutboundInterface
Operation:	listFTP

 __ c. Click **Continue** button under Events

 __ d. From Deployment Location window (if opens), select **WebSphere Process Servers > WebSphere Process Server v6.2** and click **Finish**

 __ e. Test client will return the list of files under the <OUT_DIR>. Adapter will list all the files present in the **OutDir** on the FTP server

Test Retrieve operation:

____ 12. Click **Invoke** ( ) under Events to start a new event

____ 13. Under **Detailed Properties**, for the **Operation** field, select **retrieveFTP** from the drop down menu

 Fill out the fields for Initial request parameters:

 __ a. For **Filename**, enter **PassThroughTest.1.txt** (The file name should already exist for retrieve operation. PassThroughTest.1.txt was created in the previous test)

 __ b. For **DeleteOnRetrieve**, change it to **true** from the drop down list

__ c. For **ArchiveDirectoryForDeleteOnRetrieve**, enter the full path of **RetrieveArchive** (this should already be created on your FTP server)

Name	Type	Value
⊟ 🖳 retrieveFTPInput	FTPFile	✔
⌐ 🔲 DirectoryPath	string	✔
⌐ 🔲 Filename	string	✔ PassThroughTest.1.txt
⌐ 🔲 ChunkInfo	string	✔
⌐ 🔲 FtpServerHostName	string	✔
⌐ 🔲 FtpServerEventDirectory	string	✔
⌐ 🔲 DataConnectionMode	string	✔
⌐ 🔲 FileTransferType	string	✔
⌐ 🔲 SecondServerDirectory	string	✔
⌐ 🔲 SecondServerUsername	string	✔
⌐ 🔲 SecondServerPassword	string	✔
⌐ 🔲 FileContentEncoding	string	✔
⌐ 🔲 IncludeEndBODelimiter	string	✔
⌐ 🔲 FileInLocalDirectory	boolean	✔ false
⌐ 🔲 LocalDirectoryPath	string	✔
⌐ 🔲 LocalArchivingEnabledForCreate	boolean	✔ false
⌐ 🔲 LocalArchiveDirForCreate	string	✔
⌐ 🔲 StagingDirectory	string	✔
⌐ 🔲 GenerateUniqueFile	boolean	✔ false
⌐ 🔲 CreateFileIfNotExists	boolean	✔ false
⌐ [■] ScriptFileParameters	string[]	6ᵔ
⌐ 🔲 SplittingFunctionClassName	string	✔
⌐ 🔲 SplitCriteria	string	✔
⌐ 🔲 DeleteOnRetrieve	boolean	✔ true
⌐ 🔲 ArchiveDirectoryForRetrieve	string	✔ home/wsbeta/RetrieveArchive

__ d. Click **Continue** button under Events

__ e. From Deployment Location window (if opens), select **WebSphere Process Servers > WebSphere Process Server v6.2** and click **Finish**

__ f. You should see a window similar to this, which contains the data you just entered in the previous steps:

WBPMv62_AdapterFTPOutboundLab.doc

____ 14. Verify the results

   __ a. The file **PassThroughTest.1.txt** is deleted from **OutDir** on your FTP server

   __ b. There is a new file created in RetrieveArchive directory on your FTP server. You can open the
           file in this directory to confirm the same contents

Test ExecuteFTPScript operation: This operation executes the commands present in a FTP Script file
(file in the adapter machine). The operation executes only those commands that are supported by the FTP
Server and ignores the rest. If the operation fails the adapter flags an FTPFileExecuteFTPScriptException.

Also the script file should not contain connection related commands like open as you use an already
established connection to run the commands. The location of the script file is given in DirectoryPath and
Filename. If the commands in the script file need to be run in a particular directory on the FTP Server, then
the script file should contain the first command to change to that directory. The list of commands ran and
their reply strings is returned back to the calling application after executing the script file. The adapter also
supports parameter substitution in the ftp script file (replacing parameters %1, %2 with actual values). The
values are sent as part of the request.

For your convenience, a script file, **ScriptFile.txt** is already created at <LABFILES>\FTPOutbound. Open
the file and make a note of the parameters that should be passed to this file from the test client:

%1 through %6 are replaced with the parameters specified in the test client.

%1 = any command to be performed on FTP server (Ex: pwd) - replaced by the SplitFileParameters[0]

%2 = the directory that you want to create in FTP server (FTPScriptDir) - replaced by the
SplitFileParameters[1]

%3 = the same directory specified for %2 (FTPScriptDir) - replaced by the SplitFileParameters[2]

%4 = location of any file that should be copied to New1.txt - replaced by the SplitFileParameters[3]

%5 = location of any file that should be copied to New2.txt - replaced by the SplitFileParameters[4]

%6 = any operation you want to do on New2.txt (for Ex: delete) - replaced by the
SplitFileParameters[5]

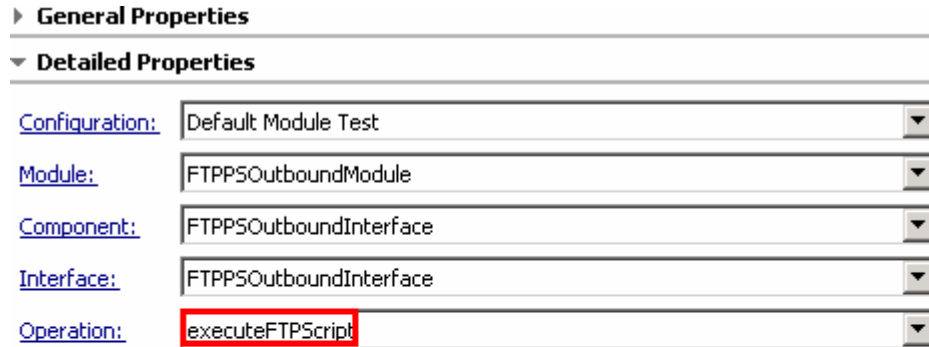____ 15. Click **Invoke** ( ) under Events to start a new event

____ 16. Under **Detailed Properties**, for the **Operation** field, select **executeFTPScript** from the drop down menu

Fill out the fields for Initial request parameters:

__ a. For **DirectoryPath**, enter **<LABFILES>\FTPOutbound**
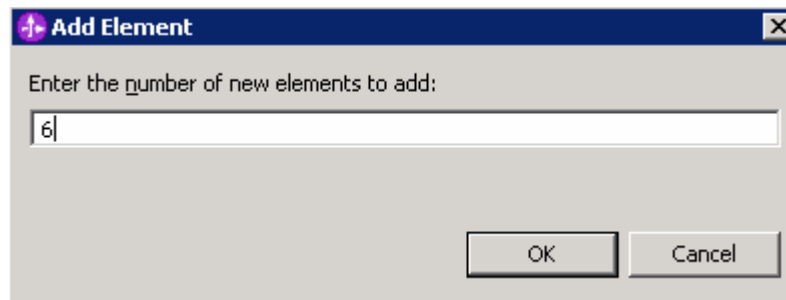
__ b. For **Filename**, enter **ScriptFile.txt**

__ c. Add the required script file parameters

1) Right-click **ScriptFileParameters** and then select **Add Elements** from the pop-up menu

2) Enter **6** from the Add Element window and click **OK**

__ d. You will see six ScriptFileParameters added. Enter these for each of the parameters

1) ScriptFileParamaters[0]: **pwd**

2) ScriptFileParamaters[1]: **FTPScriptDir**

3) ScriptFileParamaters[2]: **FTPScriptDir**

4) ScriptFileParamaters[3]: **C:\Labfiles62\FTPOutbound\ScriptContent.txt**

5) ScriptFileParamaters[4]: **C:\Labfiles62\FTPOutbound\ScriptContent.txt**

6) ScriptFileParamaters[5]: **delete**

ScriptFileParameters	string[]	6°
ScriptFileParameters[0]	string	✔ pwd
ScriptFileParameters[1]	string	✔ FTPScriptDir
ScriptFileParameters[2]	string	✔ FTPScriptDir
ScriptFileParameters[3]	string	✔ C:\Labfiles62\FTPOutbound\ScriptContent.txt
ScriptFileParameters[4]	string	✔ C:\Labfiles62\FTPOutbound\ScriptContent.txt
ScriptFileParameters[5]	string	✔ delete

__ e. Click **Continue** button under Events

__ f. From Deployment Location window (if opens), select **WebSphere Process Servers > WebSphere Process Server v6.2** and click **Finish**

____ 17.   Verify Results

__ a. You should see a window similar to this, which contains the results of all the ftp commands the were performed in the previous step:

Name	Type	Value
executeFTPScriptOutput	ExecuteFTPScriptResponse	✔
CommandTrace	string[]	6°
CommandTrace[0]	string	✔ pwd:257 "/root"...
CommandTrace[1]	string	✔ cd:250 Directory successfully changed....
CommandTrace[2]	string	✔ mkdir:257 "/download/guestusr/FTPScriptDir" created...
CommandTrace[3]	string	✔ cd:250 Directory successfully changed....
CommandTrace[4]	string	✔ pwd:257 "/download/guestusr/FTPScriptDir"...
CommandTrace[5]	string	✔ send:226 File receive OK....
CommandTrace[6]	string	✔ send:226 File receive OK....
CommandTrace[7]	string	✔ rename:250 Rename successful....
CommandTrace[8]	string	✔ delete:250 Delete operation successful....
CommandTrace[9]	string	✔ cd:250 Directory successfully changed....

__ b. Check for the **FTPScriptDir** in your FTP server which is created during this test

__ c. There is a new file, **New1_Renamed.txt** under FTPScriptDir. Open it and verify the content added to this file:

```
This file contains content to test createFTPScript operation
~
~
~
~
~
"New1_Renamed.txt" 1 line --100%--
```

3.3.　　Restore Server Configuration

____ 1.　Close the **FTPPSOutboundModule_Test** window and click **No** for the Save Resources window

____ 2.　Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects…** from the pop-up menu

____ 3.　Select **FTPPSOutboundModuleApp** under Configured projects and click **< Remove**

____ 4.　Click **Finish** after you see the application moved to Available projects. Wait until the application is unpublished

Part 4: Content specific (non-pass through) scenario

Of the two outbound flows, you have just tested the pass-through which does not involve data transformation. In this part of the lab you will perform the non-pass through that uses the data transformation:
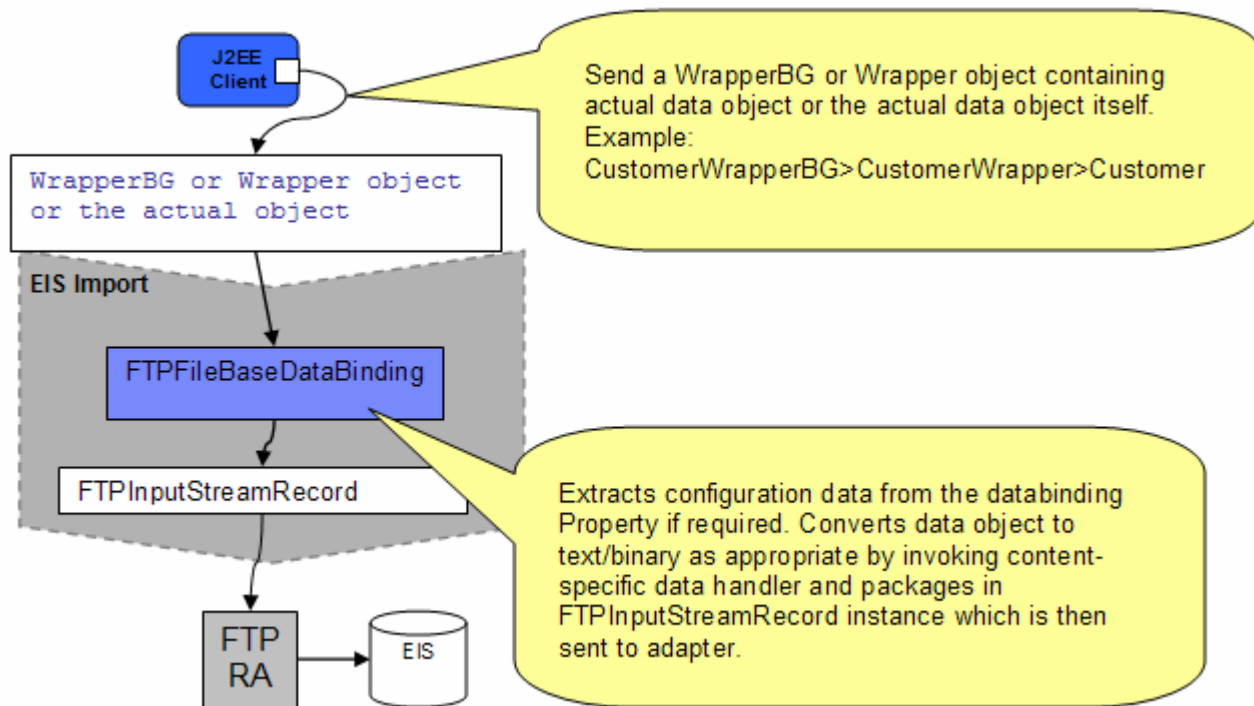
- In the Java EE client, if you intend to use business graph (CustomerWrapperBG) along with the wrapper data object (CustomerWrapper) present in the WrapperBG, it is populated for protocol specific information. The actual data object (for example: Customer) is set in the wrapper data object. The wrapper data object and the outbound operation name are sent as input by making an SCA call (or).

- If you intend to use the wrapper data object (CustomerWrapper), it is populated for protocol specific information and the actual data object (for example: Customer) is set in the wrapper data object. The wrapper data object and the outbound operation name are sent as input by making an SCA call (or).

- If want to send the actual data without wrapper or business graph then the protocol specific information should already be set in the interaction spec and the actual data object (for example: Customer) and the outbound operation name are sent as input by making an SCA call.

- Based on the Data Binding (FTPFileBaseDataBinding) configured while running the external service wizard, that particular data binding is called and it gets the actual data object either from wrapper data object if the wrapper data object is used or from the wrapper data object present in the business graph if the business graph is used or the data object itself if neither wrapper or business graph is used.

- The FTPFileBaseDataBinding checks the data handler information and it is invoked. If the data handler information is not valid, then appropriate error is thrown by the Data Transformation Framework.

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

- The output of the custom data binding (FTPFileBaseDataBinding) is an FTPInputStreamRecord. This Record object is sent to the adapter and the outbound operation is executed. The output of the outbound operation is an FTPStructuredRecord which is sent back to the configured Data Binding (FTPFileBaseDataBinding) for List, Exists and Retrieve operations. For Create, Append, OverWrite and ExecuteFTPScript, if output required is true, then the FTPStructuredRecord is sent back to the configured Data Binding (FTPFileBaseDataBinding) and a data object containing the output is sent back to the Java EE client.

4.1. Configure non-pass through using the external service wizard

In this part you use the new WebSphere Integration Developer feature, External Service, to create/configure the Data Binding, **Data Handler**, and Operations, which generates the business objects and other artifacts.

_____ 1.   Create FTPCustomOutboundModule

    __ a. From the Business Integration window, right-click and select **New > Module**

    __ b. From the New Module window, enter **FTPCustomOutboundModule** 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, FTPCustomOutboundModule, created in your Business Integration window

_____ 2.   Import required business objects

    __ a. Expand FTPCustomOutboundModule (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 ear files under that folder on the right side.

    __ d. Select the box next to **Customer.xsd**

    __ e. Ensure that the **FTPCustomOutboundModule** 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 the imported business objects:

    __ a. Expand **FTPCustomOutboundModule > Data Types** and you will now see a new data type **Customer** under it.

__ b. Double-click **Customer** to view its fields:
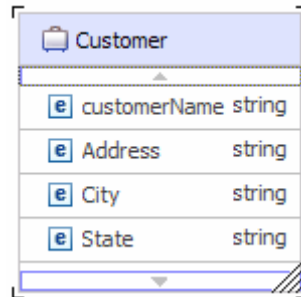
____ 4. After reviewing, close the Customer business object from the Assembly editor

____ 5. To start the external service from the Palette:

__ a. From the **Palette** on the left side of Assembly Diagram, click **Outbound Adapters**:

__ b. Under Outbound 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, select **Create a service (advanced)**
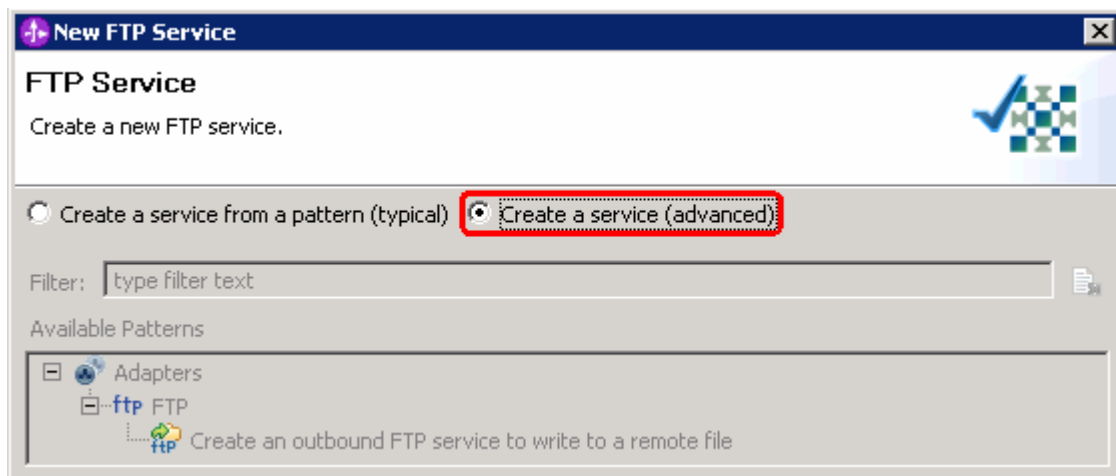
__ a. Click **Next**

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 > FTP** and click **Next**

____ 7.   On the Select an Adapter screen, select **IBM WebSphere Adapter for FTP (IBM : 6.2.0.0) > CWYFT_FTPFile** and click **Next**
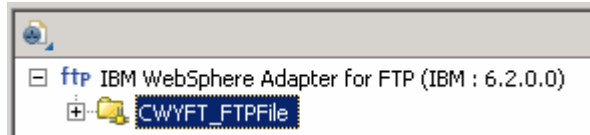
____ 8.   Service Configuration Properties:

__ 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) Directory: **full path of the OutDir created in on the machine where FTP server is existing** (for Ex: /home/wsbeta/OutDir)
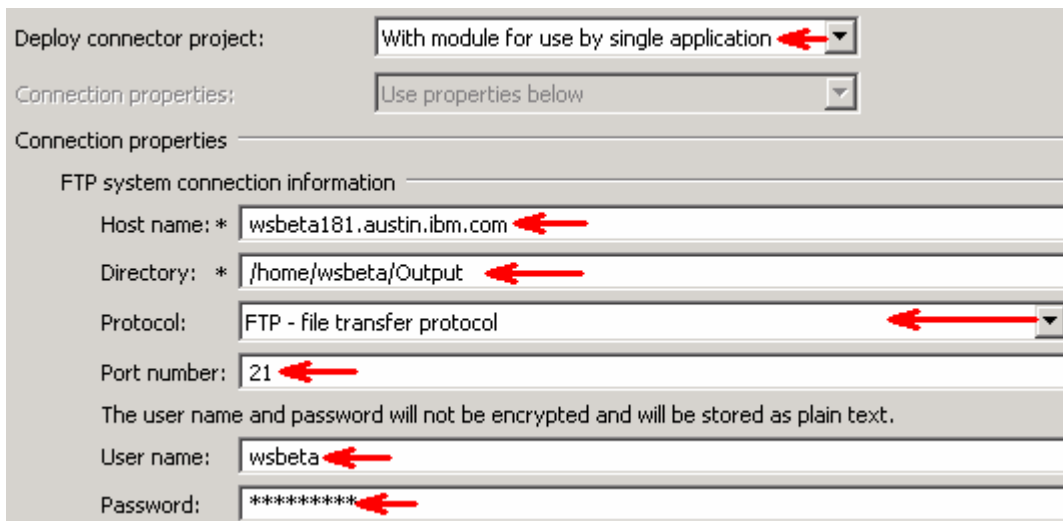
Note: This is the folder where the adapter will create the file.

---

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

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

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

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

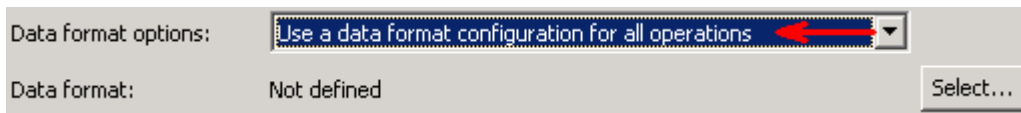__ c. Click **Advanced >>** to see the hidden advanced properties that can be configured:

____ 9.   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:

Data binding and Data handler configuration:

____ 10.   You can define data binding in two places - service level (current screen of the 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**'

Data format options: Use a data format configuration for all operations

Data format: Not defined   Select...

____ 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**

Data Binding Configuration

Select Data Format Transformation

Select a data format transformation entry from the list. If you want to use your own custom data transformation then select the second radio button to add your custom transformation.

⦿ **Use existing data format transformation from the list**

FTPFileBaseDataBinding
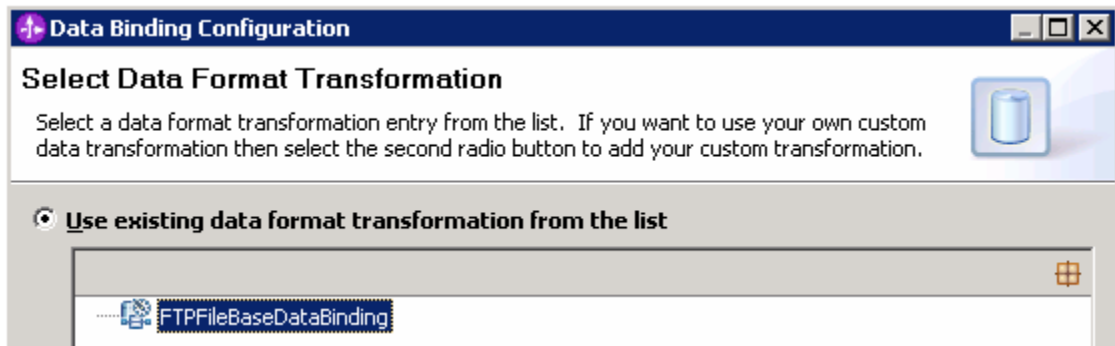
____ d. Click **Next**

____ e. From Data Transformation Properties screen, click **Select…** next to **Configured data handler**. A Binding Resource Configuration window is opened for you to define the data handler

____ f. Select the radio button for '**Use existing data format transformation from the list'** and then select **XML > UTF8XMLDataHandler**

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.
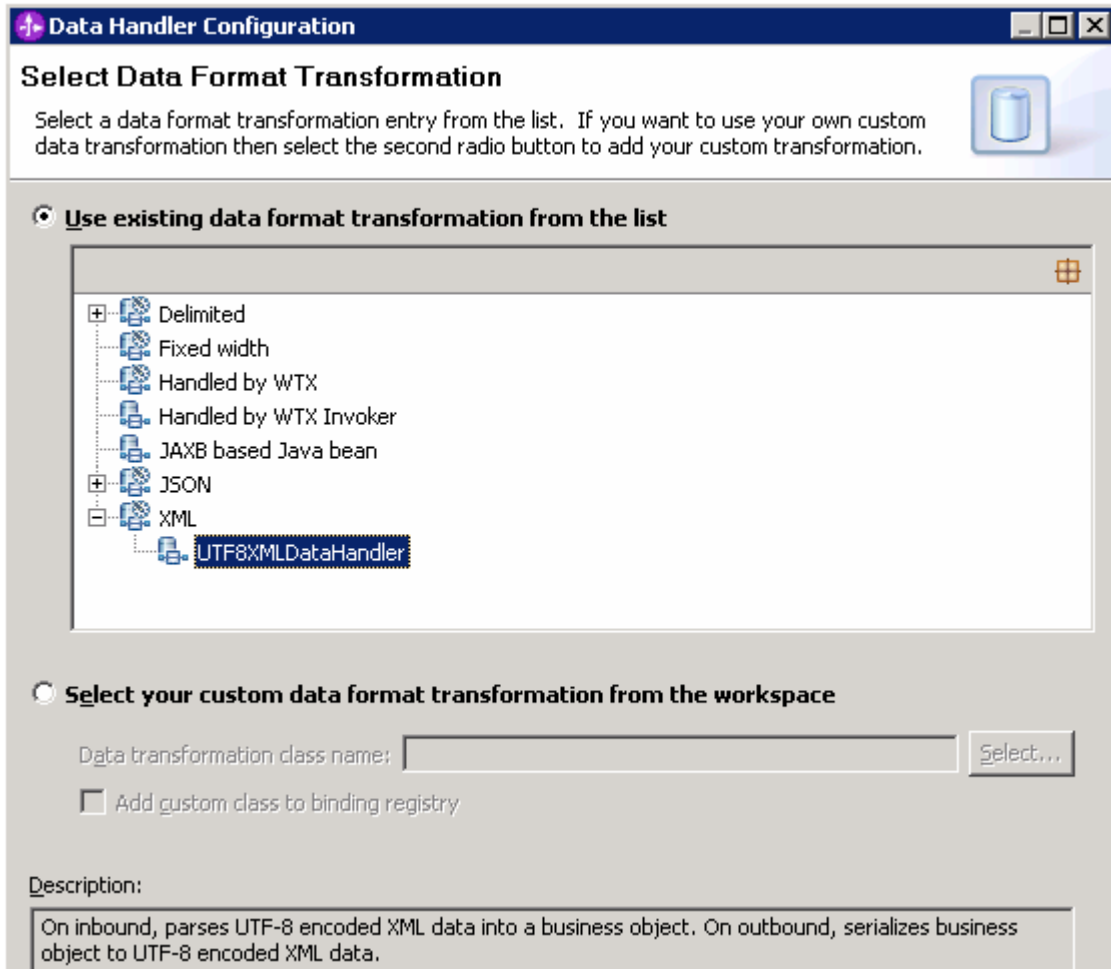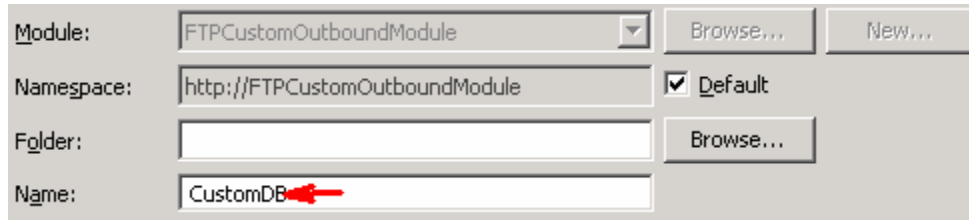

___ g. Click **Finish**

___ h. Back to 'Data Transformation Properties' screen, and the Configured data handler, **UT8XMLDatahandler** is displayed defined in the previous steps
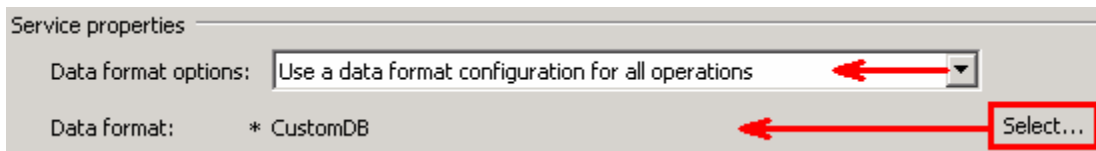
WBPMv6.2_AdapterFTPOutboundLab.doc
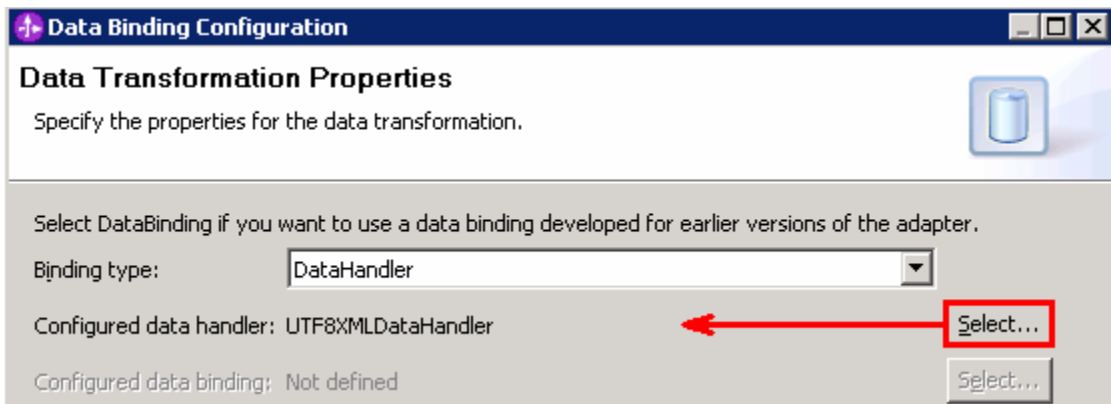
__ i. Click **Next**

__ j. Ensure that the module selected is FTPCustomOutboundModule and enter **CustomDB** for the Name of the data binding

Module:	FTPCustomOutboundModule ▾ Browse... New...
Namespace:	http://FTPCustomOutboundModule ☑ Default
Folder:	Browse...
Name:	CustomDB ◀

__ k. Click **Finish**

__ l. Now the **CustomDB** should be displayed for **Data format**

Service properties

Data format options: Use a data format configuration for all operations ◀ ▾

Data format: * CustomDB ◀ Select...

_____ 11. 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**

Define Operations: In this screen, you will add the required operations that is supported by the adapter functions on the remote file system

Note: The precedence of the parameters is as follows: WrapperBO, Interaction Spec, and Managed Connection Factory. The adapter will first search for the parameters passed in the WrapperBO; if it is not available there, it will then subsequently search in the Interaction Spec, and then the Managed Connection Factory instance. **In this lab, for all the operations, you will enter the values at the WrapperBO level in the later part using the WebSphere Integration Developer test client.**

Define Operation: createCustomFile:

_____ 12. Click **Add…** to open Add Operation window

__ a. For **Operation kind**, select **Create** from the drop down list

__ b. For **Data type for operation input**, select **User defined type** from the drop down list

__ c. Select the check box next to **Enable response type for the operation**

🔷 **Add Operation**   _ ☐ ✕

Operation

Specify the properties for the operation to add.

Operation kind: Create ◀ ▾

Operation properties

Data type for the operation: User defined type ◀ ▾
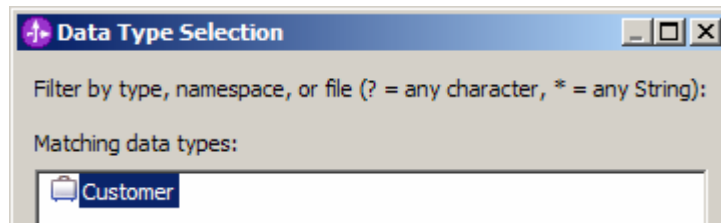
☑ Enable response type for the operation
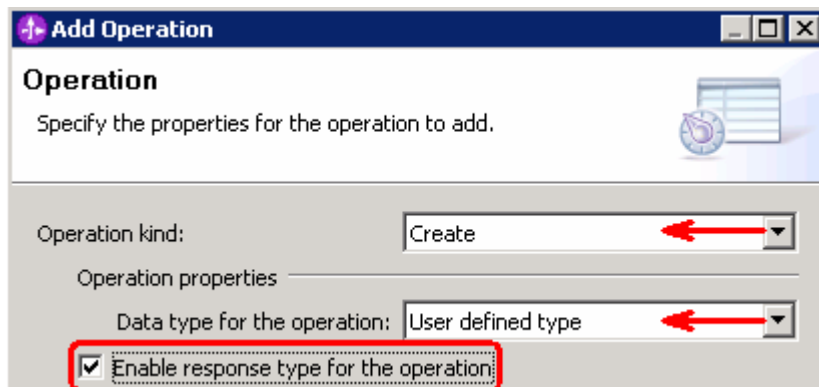
__ d. Click **Next**

The Data type for input and output are populated based on the selection of the Data type for the operation in the previous step. Since you have chosen User defined type, the **Input type** is **blank** and because you have selected Output required box, the **Output type** is **CreateResponse**

Operation name:	* create	
Specify the operation input		
Input type: ➡ *		Browse... New...
Data format options:	Use data format configuration 'CustomDB' ▾	
Data format:	Not defined	Select...
Specify the operation output		
Output type: ➡	CreateResponse {http://www.ibm.com/xmlns/prod/wel	Browse... New...
Data format options:	Use suggested data format 'FTPFileBaseDataBinding' ▾	
Data format:	Not defined	Select...

__ e. For Operation name, enter **createCustomFile**

__ f. Define Input type:

       1) Under **Specify the operation input**, click **New…** next to **Input type** to open a New Business Object window

       2) Ensure that the Module selected is **FTPCustomOutboundModule** and click **Next**

       3) From this window, Click **Browse…** next to Data type

       4) From the Data Type Selection window, select **Customer** under Matching data types:

Data Type Selection _ □ X
Filter by type, namespace, or file (? = any character, * = any String):
Matching data types:
Customer

       5) Click **OK**

__ g. From the Business Object window, **check** the box next to **Generate business graph for each business object**

__ h. **Do not** check the box for **'Generate retrieve container to retrieve multiple business objects'**

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

    __ i. Click **Finish**

In the Add Operation window, under Specify the operation input, you will see the Input type **CustomerWrapperBG** (because you have selected to have business graph (BG) generated):

Define Data format for **input**:

    __ j. Accept the default Data format options selection, **Use data format configuration 'CustomDB'**

Next define the Data format for **output**:

    __ k. For **Data binding**, select **Use a data binding configuration** from the dropdown list

    __ l. Click **Select…** next to **Data format**. A Binding Resource Configuration window is opened

    __ m. Ensure that the radio button for 'Use existing data format transformation from the list' and then select **FTPFileBaseDataBinding > CustomDB**

    __ n. Click **Finish**

    __ o. The Operation screen now should look like this:


    __ p. Click **Finish** from the Add Operation window

The operation, createCustomFile, will now be displayed under Operations list.

__ q. You can click **Advanced >>** under 'InteractionSpec properties for createCustomFile' to review the properties available at Interaction spec level
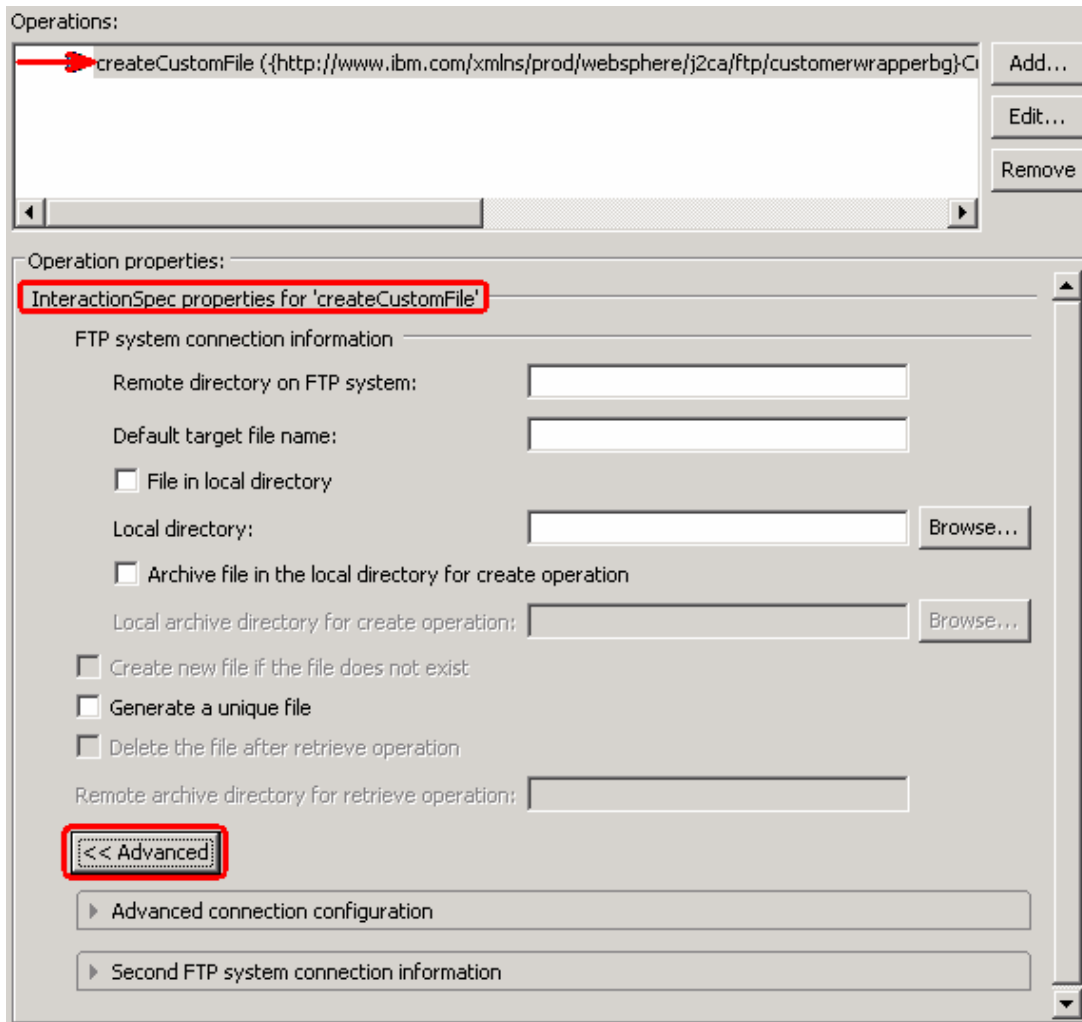
Add Operation: appendCustomFile:

_____ 13. Click **Add…** to open Add Operation window

__ a. For **Operation kind**, select **Append** from the drop down list

__ b. For **Data type for operation input**, select **User defined type** from the drop down list

__ c. Select the check box next to **Enable response type for the operation**
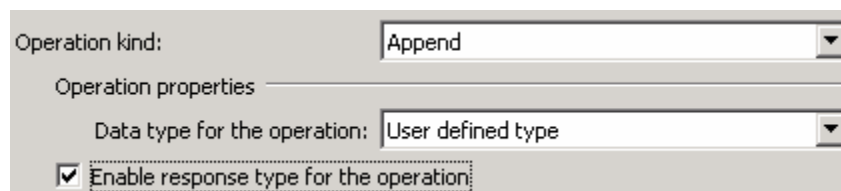
__ d. Click **Next**

You are back to Operation window and because you chose the User defined data type, the Input type and Output type is blank and because you have selected Output required box, the Output type is AppendResponse:

__ e. For Operation name, enter **appendCustomFile**

__ f. Define Input type:

      1) Under **Specify the operation input**, click **Browse…** next to **Input type**

      2) From the Data Type Selection window, select **CustomerWrapperBG** under Matching data types and click **OK**

Define Data format for **input**:

__ g. Accept the default Data format options selection, **Use data format configuration 'CustomDB'**

Define Data format for **output**:

__ h. Follow the instructions of defining data binding for createCustomFile and define **CustomDB**

__ i. You should now see this:

__ j. Click **Finish** from Add Operation window

The operation, appendCustomFile, will now be displayed under Operations list.

> __ k. You can click **Advanced >>** under 'InteractionSpec properties for appendCustomFile' to review the properties available at Interaction spec level
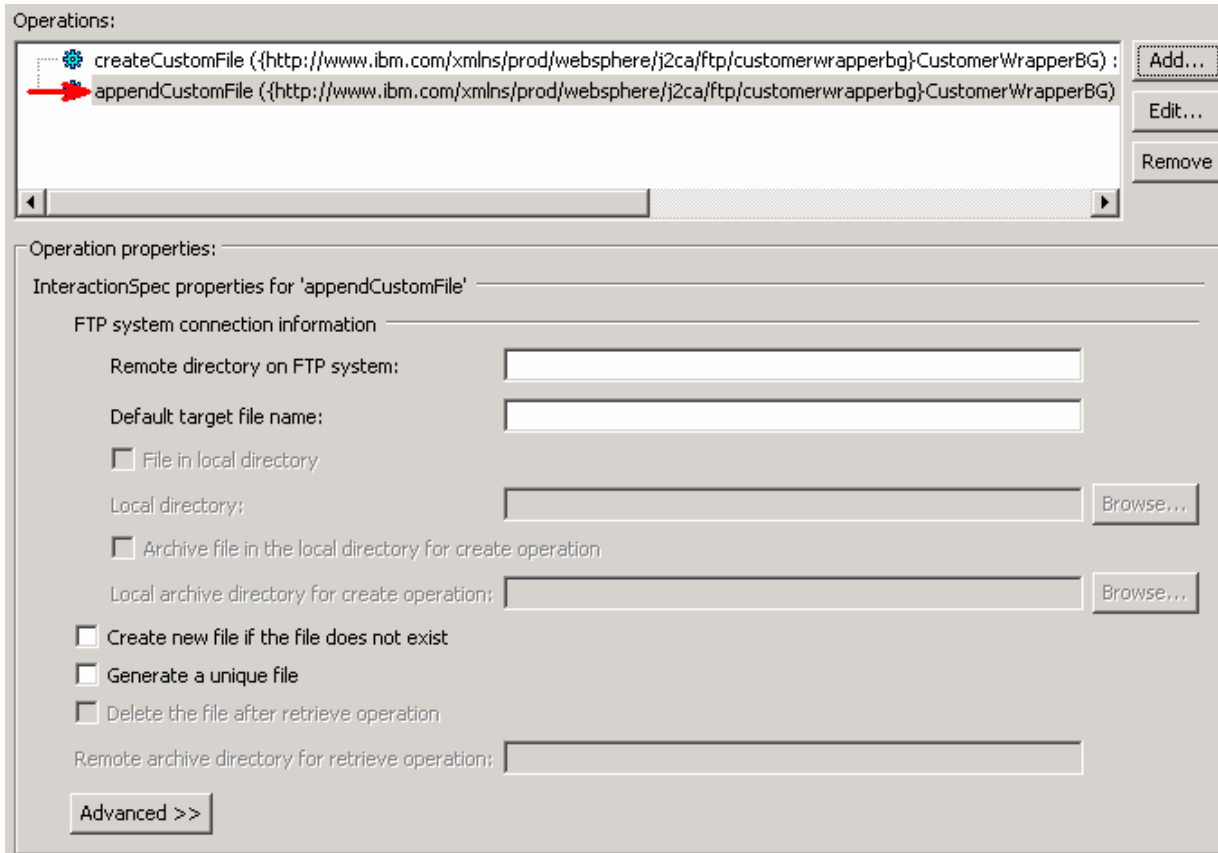
Add Operation: retrieveCustomFile

____ 14. Click **Add…** to open Add Operation window

> __ a. For **Operation kind**, select **Retrieve** from the drop down list
>
> __ b. For **Data type for operation**, select **Generic FTP business object** from the drop down list
>
> __ c. Note that the box next to **Enable response type for the operation** is checked by default
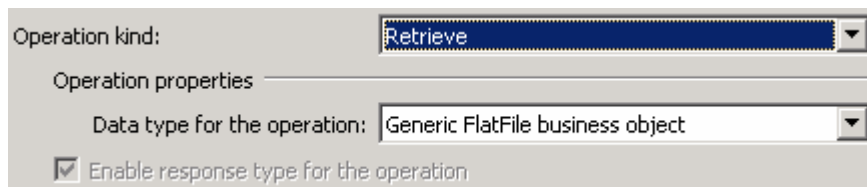
> __ d. Click **Next**

In the Add Operation window, under Specify the operation input, you will see the Input type **FTPFile** (because you have selected **not to** have business graph (BG)) and you will also see the Output type **RetrieveResponseWrapper** under Specify the operation output (because the Output was selected by default).

___ e. For Operation name, enter **retrieveCustomFile**

___ f. Define Data Binding type for **input**:

> 1) Follow the instructions of defining data binding for createCustomFile and select **CustomDB** for input data format

___ g. Define Data Binding type for **output**:

> 1) Accept the default Data format options selection, **Use data format configuration 'CustomDB'**

___ h. Define Output type:

> 1) Under **Specify the operation output**, click **New…** next to **Output type** to open a New Business Object window
>
> 2) Ensure that the Module selected is **FTPCustomOutboundModule** and click **Next**
>
> 3) From this window, Click **Browse…** next to **Data type**
>
> 4) From the Data Type Selection window, select **Customer** under Matching data types and click **OK**

___ i. From the Business Object window, **check** the box next to **Generate business graph for each business object**

___ j. **Check** the box for **'Generate retrieve container to retrieve multiple business objects'**

Note: Check this option, when you have multiple business objects in the file that the adapter is going to retrieve.

Data type: * Customer {http://www.ibm.com/xmlns/prod/websphere   Browse…   New…
☑ Generate business graph for each business object
Namespace for generated business graph and container object.
Business object namespace: http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp
☑ Generate retrieve container to retrieve multiple business objects.

___ k. Click **Finish**. You should see that the output type 'RetrieveResponseWrapper' is replaced with **CustomerRetrieveWrapperBG**

You should now see this:

___ I. Click **Finish** from the Add Operation window

The operation, retrieveCustomFile, will now be displayed under Operations list.

____ 15.   You can click **Advanced >>** under 'InteractionSpec properties for retrieveCustomFile' to review the properties available at Interaction spec level

__ a. Click **Next** from the Operations window

____ 16. From the Generate Artifacts screen, enter these:
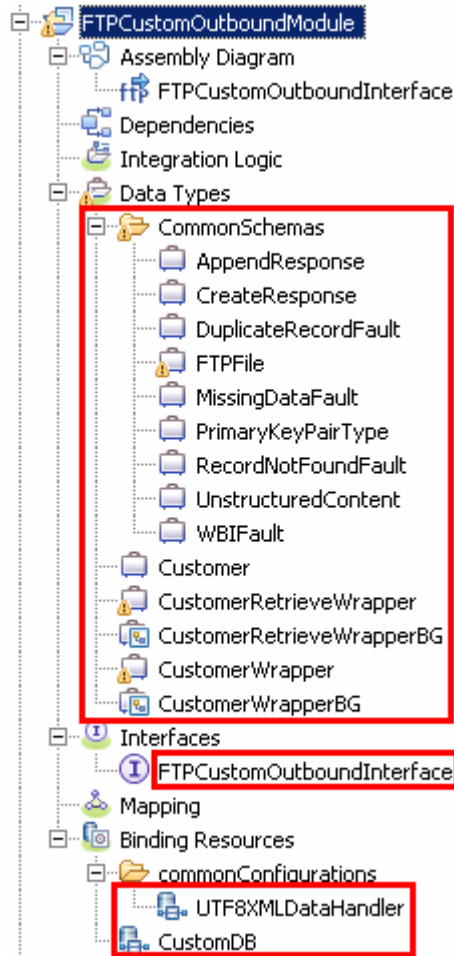
__ a. For **Name**, enter **FTPCustomOutboundInterface**
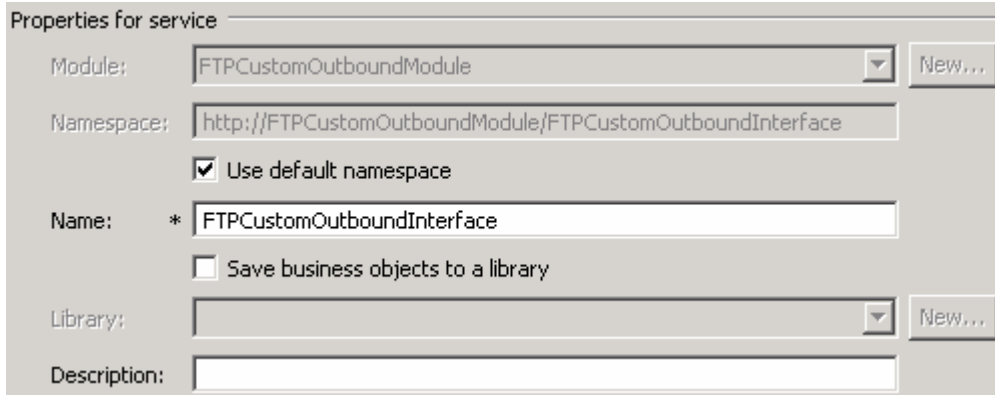
__ b. Click **Finish**

____ 17. You will now see a new import component, **FTPCustomOutboundInterface** in the assembly diagram of FTPCustomOutboundModule
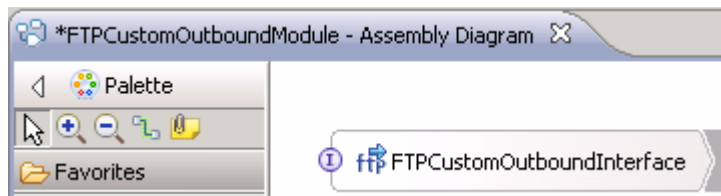
__ a. Save (**Ctrl+S)** your changes to the assembly diagram

____ 18. Review the FTPCustomOutboundModule: The generated **Data Types**, **Interface**, and the Data binding (**CustomDB**), and Data handler (**CustomDH**) under Configured Resources can be found inside FTPCustomOutboundModule

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 FTPCustomOutboundModule > Interfaces and then double-click **FTPCustomOutboundInterface** to open it

__ b. You should see these three operations:

I FTPCustomOutboundInterface ⊠		
▾**Operations** 🌼 🌼 | 🔖 🔖 🔖 | 🔳		
Operations and their parameters		
	Name	Type
▾🌼 createCustomFile		
▷️ Input(s)	createCustomFileInput	CustomerWrapperBG
▷️ Output(s)	createCustomFileOutput	CreateResponse
☒ Fault	MISSING_DATA	MissingDataFault
☒ Fault	WBIFault	WBIFault
☒ Fault	DUPLICATE_RECORD	DuplicateRecordFault
☒ Fault	RECORD_NOT_FOUND	RecordNotFoundFault
▾🌼 appendCustomFile		
▷️ Input(s)	appendCustomFileInput	CustomerWrapperBG
▷️ Output(s)	appendCustomFileOutput	AppendResponse
☒ Fault	MISSING_DATA	MissingDataFault
☒ Fault	WBIFault	WBIFault
☒ Fault	RECORD_NOT_FOUND	RecordNotFoundFault
▾🌼 retrieveCustomFile		
▷️ Input(s)	retrieveCustomFileInput	FTPFile
▷️ Output(s)	retrieveCustomFileOutput	CustomerRetrieveWrapperBG
☒ Fault	MISSING_DATA	MissingDataFault
☒ Fault	WBIFault	WBIFault
☒ Fault	RECORD_NOT_FOUND	RecordNotFoundFault

__ c. Close the interface, FTPCustomOutboundInterface

4.2.　　　Test content specific scenario

____ 1.　Start WebSphere Process Server (if not started already)

　__ a. From the **Servers** view of WebSphere Integration Developer, right click **WebSphere Process Server v6.2** and select Start from the pop-up menu

　__ b. Wait until the server status shows as **Started**

____ 2.　Add the project to the WebSphere Process Server Test Environment

　__ a. Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects…** from the pop-up menu

　__ b. In the Add and Remove Projects window, select the **FTPCustomOutboundModuleApp** project from the Available projects panel

　__ c. Click **Add >** to add it to the Configured projects panel

　__ d. The project now is moved to Configured projects. Click **Finish**

Wait for the project to be published to the server.

____ 3.　Open the test client for the module

　__ a. From the Business Integration perspective, right-click the **FTPCustomOutboundModule** and select **Test > Test Module**

　__ b. The **FTPCustomOutboundModule_Test** window is opened in the Assembly editor

You have three operations that were defined in the previous part in this module:

- createCustomFile

- appendCustomFile

- retrieveCustomFile

Test Create operation:

____ 4.　Under **Detailed Properties**, for the **Operation** field, select **createCustomFile** from the drop down menu

　Fill out the fields for Initial request parameter

　__ a. For **DirectoryPath**, **enter full path of the OutDir** that is already created in your FTP server. You do not need to provide the directory here if you had already specified during external service wizard

　__ b. For **Filename**, enter **UserDefined.xml**

　__ c. For **IncludeEndBODelimiter**, enter **#####**

　__ d. For **Content**, enter any random data. For Ex:

　　　1) Name: ABC

　　　2) Address: 11501 Burnet Rd

3) City: Austin

4) State: TX

__ e. Click **Continue** button under Events

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

__ f. From Deployment Location window, select **WebSphere Process Servers > WebSphere Process Server v6.2 at localhost** and click **Finish**

__ g. Provide Administrator User ID and Password

1) Optionally, select the box 'Use the authentication settings in the preference and never ask again' so that you do not have to enter the credentials next time when you start the test client
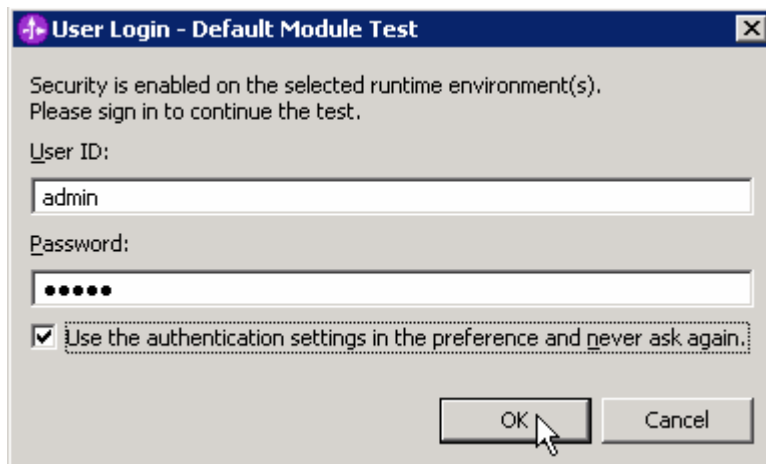
__ h. Click **OK** from Adding/Removing Projects pop-up, if asked. Optionally, you can select 'Do not show again' so that you are not required to do this next time when you start the test client

__ i. You should see a window similar to this, which contains the data you just entered in the previous steps:

____ 5.   Verify the created file and its contents

__ a. You will see a new file, **UserDefined.xml** created under OutDir on your FTP server. Open that file to see the content that was entered and ##### at the end of the file.

Test Append operation:

____ 6.   Click **Invoke** ( ) under Events to start a new event

____ 7.   Under **Detailed Properties**, for the **Operation** field, select **appendCustomFile** from the drop down menu

Fill out the fields for Initial request parameters:

__ a. For **DirectoryPath**, **enter full path of the OutDir** that is already created in your FTP server

__ b. For **Filename**, enter **UserDefined.xml** (The file name should already exist for append operation. PassThroughTest.txt was created during Create operation test)

__ c. For **IncludeEndBODelimiter**, enter **#####**

__ d. For **Content**, enter any random data. For Ex:

    1) Name: IBM

    2) Address: 11901 RTP

    3) City: Raleigh

    4) State: NC

Detailed Properties

Configuration:	Default Module Test
Module:	FTPCustomOutboundModule
Component:	FTPCustomOutboundInterface
Interface:	FTPCustomOutboundInterface
Operation:	appendCustomFile

Initial request parameters

Name	Type	
appendCustomFileInput	CustomerWrapperBG	✓
verb	verb<string>	
CustomerWrapper	CustomerWrapper	✓
DirectoryPath	string	✓ /home/wsbeta/OutDir
Filename	string	✓ UserDefined.xml
ChunkInfo	string	✓
FtpServerHostName	string	✓
FtpServerEventDirectory	string	✓
DataConnectionMode	string	✓
FileTransferType	string	✓
SecondServerDirectory	string	✓
SecondServerUsername	string	✓
SecondServerPassword	string	✓
FileContentEncoding	string	✓
IncludeEndBODelimiter	string	✓ #####
FileInLocalDirectory	boolean	✓ false
LocalDirectoryPath	string	✓
LocalArchivingEnabledForCreate	boolean	✓ false
LocalArchiveDirForCreate	string	✓
StagingDirectory	string	✓
GenerateUniqueFile	boolean	✓ false
CreateFileIfNotExists	boolean	✓ false
ScriptFileParameters	string[]	66
SplittingFunctionClassName	string	✓
SplitCriteria	string	✓
DeleteOnRetrieve	boolean	✓ false
ArchiveDirectoryForRetrieve	string	✓
Content	Customer	✓
CustomerName	string	✓ IBM
Address	string	✓ 11901 RTP
City	string	✓ Raleigh
State	string	✓ NC

__ e. Click **Continue** button under Events

__ f. From Deployment Location window (if opens), select **WebSphere Process Servers > WebSphere Process Server v6.2** and click **Finish**

__ g. You should see a window similar to this, which contains the data you just entered in the previous steps:

____ 8.   Verify the created file and its contents

__ a. Open the **UserDefined.xml** file under OutDir on your FTP server. You should see the content appended to the original content created during the create operation

Test Retrieve operation: In Retrieve operation, the adapter retrieves the file and parses it based on the configured SplittingFunctionClassName and SplitCriteria properties.

If splitting needs to be done based on file size, then splitFunctionClassName should be com.ibm.j2ca.utils.filesplit.SplitBySize which does the splitting functionality based on size and splitCriteria should be a number (a number representing the size in bytes). If the actual event file size is greater than this value, it is split into chunks else it is sent as one BO).

If splitting needs to be done based on a delimiter, then splitFunctionClassName should be com.ibm.j2ca.utils.filesplit.SplitByDelimiter which does this functionality and the splitCriteria (the delimiter which actually separates the BO's present in the event file) should be given.

This lab will guide you through the later splitting criteria, SplitByDelimiter.

____ 9.   Click **Invoke** (   ) under Events to start a new event

____ 10.   Under **Detailed Properties**, for the **Operation** field, select **retrieveCustomFile** from the drop down menu

Fill out the fields for Initial request parameters:

__ a. For **DirectoryPath**, **enter full path of the OutDir** that is already created in your FTP server.
This field is not required if you already specified the directory during external service wizard.

__ b. For **Filename**, enter **UserDefined.xml** (The file name should already exist for retrieve operation.
UserDefined.xml was created during Create operation test)

__ c. For LocalDirectoryPath, enter **<LOCAL_DIR>**

Note: LocalDirectoryPath is mandatory for Retrieve operation during the content specific scenario. If you
set the **FileInLocalDirectory** to **true** along with the LocalDirectoryPath, then UserDefined.xml is copied to
<LOCAL_DIR> and you can see the retrieved contents by opening the file copied to LocalDirectoryPath.
This lab will show you the case where you set only the LocalDirectoryPath and leave the
FileInLocalDirectory with the default false.

__ d. For **splitFunctionClassName**, enter **com.ibm.j2ca.utils.filesplit.SplitByDelimiter**

__ e. For **splitCriteria**, enter **#####**

Name	Type	Value
⊟ 🖳 retrieveCustomFileInput	FTPFile	✔
📋 DirectoryPath	string	✔
📋 Filename	string	✔ UserDefined.xml
📋 ChunkInfo	string	✔
📋 FtpServerHostName	string	✔
📋 FtpServerEventDirectory	string	✔
📋 DataConnectionMode	string	✔
📋 FileTransferType	string	✔
📋 SecondServerDirectory	string	✔
📋 SecondServerUsername	string	✔
📋 SecondServerPassword	string	✔
📋 FileContentEncoding	string	✔
📋 IncludeEndBODelimiter	string	✔
📋 FileInLocalDirectory	boolean	✔ false
📋 LocalDirectoryPath	string	✔ C:\Labfiles62\FTPOutbound\LocalDir
📋 LocalArchivingEnabledForCreate	boolean	✔ false
📋 LocalArchiveDirForCreate	string	✔
📋 StagingDirectory	string	✔
📋 GenerateUniqueFile	boolean	✔ false
📋 CreateFileIfNotExists	boolean	✔ false
[▣] ScriptFileParameters	string[]	66ᵒ
📋 SplittingFunctionClassName	string	✔ com.ibm.j2ca.utils.filesplit.SplitByDelimiter
📋 SplitCriteria	string	✔ #####

__ f. Click **Continue** button under Events

__ g. From Deployment Location window (if opens), select **WebSphere Process Servers >
WebSphere Process Server v6.2** and click **Finish**

____ 11.   Verify the results:

__ a. You should see a window similar to this, that contains the content of the two business objects existing in the UserDefined.xml file:

4.3.　　Restore server configuration

____ 1.　Close the **FTPCustomOutboundModule_Test** window and click **No** for the Save Resources window

____ 2.　Right-click **WebSphere Process Server v6.2** under the Servers view and select **Add and remove projects…** from the pop-up menu

____ 3.　Select **FTPCustomOutboundModuleApp** under Configured projects and click **< Remove**

____ 4.　Click **Finish** after you see the application moved to Available projects. Wait until the application is unpublished

Part 5: Use default 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 continue to test the adapter.

5.1. Configure outbound using 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 outbound scenario.

____ 1.   Create FTPDefaultsOutboundModule

  __ a. From the Business Integration window, right-click and select **New > Module**

  __ b. From the New Module window, enter **FTPDefaultsOutboundModule** 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, FTPDefaultsOutboundModule, created from your Business Integration window

____ 2.   To start the external service wizard from the Palette:

  __ a. From the **Palette** on the left side of Assembly Diagram, click **Outbound Adapters**:

  __ b. Under Outbound 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)**
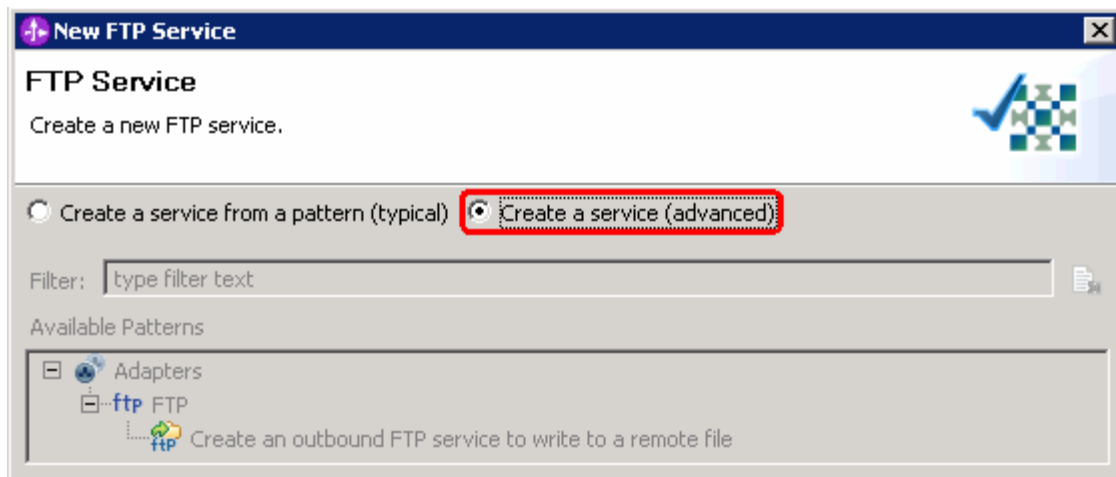
  __ a. Click **Next**

Note: You can also start the external service wizard 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**

_____ 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**

_____ 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) Directory: **full path of the OutDir created in on the machine where FTP server is existing** (for Ex: /home/wsbeta/OutDir)

Note: This is the folder where the adapter will create the file.

---

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

       4) Protocol: **FTP** (default)

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

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

   __ c. Optionally, click **Advanced >>** to see the hidden advanced properties that can be configured

_____ 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. For **Data format options**, select **Use default data binding 'FTPFileBaseDataBinding' for all operations** from the drop down list

____ 8. 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**

Define createFTPFileBG operation:

____ 9. From the Operations screen, click **Add…**

__ a. For **Operation kind**, select **Create** from the drop down list

__ b. For **Data type for the operation input**, select **Generic FTP business object with business graph** from the drop down list

__ c. Select the check box next to **Enable response type for the operation**
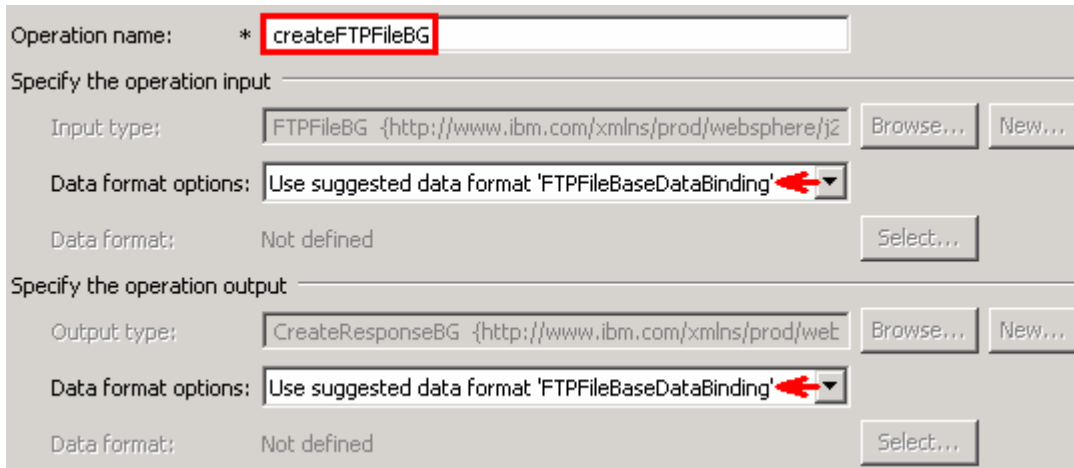

__ d. Click **Next**

The Data type for input and output are populated based on the selection of the Data type for the operation in the previous step. Since you have chosen Generic FTP business object with business graph, the **Input type** is **FTPFileBG** and because you have selected Output required box, the **Output type** is **CreateResponseBG**

____ 10. For **Operation name**, enter any name, for Ex: **createFTPFileBG**

____ 11.   Accept the default selection, **Use suggested data format 'FTPFileBaseDataBinding'**, as **Data format** for both input and output
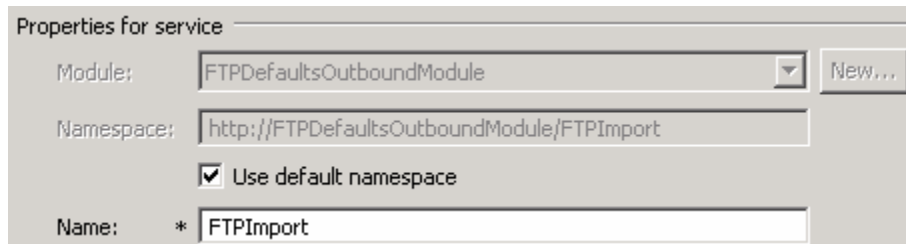
Operation name:	* createFTPFileBG
Specify the operation input	
Input type:	FTPFileBG {http://www.ibm.com/xmlns/prod/websphere/j2 Browse... New...
Data format options:	Use suggested data format 'FTPFileBaseDataBinding' ◄ ▼
Data format:	Not defined Select...
Specify the operation output	
Output type:	CreateResponseBG {http://www.ibm.com/xmlns/prod/web Browse... New...
Data format options:	Use suggested data format 'FTPFileBaseDataBinding' ◄ ▼
Data format:	Not defined Select...

__ a. Click **Finish**. The above defined operation, **createFTPFileBG**, is populated under Operations list

Operations:

createFTPFileBG ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfilebg}FTPFileBG) :   Add...

__ b. Click **Next** from Operations screen

____ 12.   From Generate Service screen, accept the default value, **FTPImport**, for **Name**

Properties for service	
Module:	FTPDefaultsOutboundModule ▼ New...
Namespace:	http://FTPDefaultsOutboundModule/FTPImport
	☑ Use default namespace
Name:	* FTPImport

__ a. Click **Finish**

____ 13.   The Assembly diagram for FTPDefaultsOutboundModule is opened with an Import component, **FTPImport**:
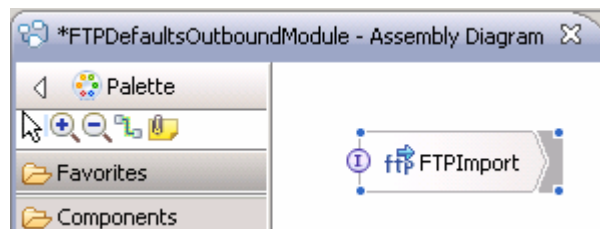
____ 14.   Save (**Ctrl + S**) changes to your assembly diagram

5.2.   Test all defaults scenario

In this part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application outbound 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 **FTPDefaultsOutboundModuleApp** under Available projects panel and click **Add >**

__ c. You will now see the **FTPDefaultsOutboundModuleApp** 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.   Open the test client for the module

__ a. From the Business Integration perspective, right-click the **FTPDefaultsOutboundModule** and select **Test > Test Module**

____ 3.   The **FTPDefaultsOutboundModule_Test** window is opened in the Assembly editor

____ 4.   Under **Detailed Properties**, for the **Operation** field, select **createFTPFileBG** from the drop down menu

Fill out the fields for Initial request parameters:

__ a. For **Filename**, enter any name, for Ex: **DefaultsTest.txt**

▸ **General Properties**

▾ **Detailed Properties**

Configuration:	Default Module Test
Module:	FTPDefaultsOutboundModule
Component:	FTPImport
Interface:	FTPImport
Operation:	createFTPFileBG

☐ Invoke export using binding

Initial request parameters

Name	Type	Value
☐ 🔧 createFTPFileBGInput	FTPFileBG	✔
⌐ verb	verb<string>	✔ CREATE
☐ 🔧 FTPFile	FTPFile	✔
⌐ DirectoryPath	string	✔
⌐ Filename	string	✔ DefaultsTest.txt
⌐ ChunkInfo	string	✔

__ b. For **AsText** under **Content**, enter some test message, for Ex: **Testing Create operation for defaults scenario**

__ c. Click **Continue** button under Events

__ d. From Deployment Location window, select **WebSphere Process Servers > WebSphere Process Server v6.2** and click **Finish**

__ e. You should see a window similar to this, which contains the data you just entered in the previous steps:

____ 5.   Verify the created file and its contents

__ a. You will see a new file, **DefaultsTest.txt** created under **OutDir** on your FTP server. Open that file to see the content

____ 6.   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 **FTPDefaultsOutboundModuleApp** 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

Part 6: 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 Data Binding and other required artifacts to test the outbound scenario.

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

After running the external service wizard, you will continue to test the adapter.

6.1. Configure outbound 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: FTPTypicalOutboundModule

    __ a. From the Business Integration window, right-click and select **New > Module**

    __ b. From the New Module window, enter **FTPTypicalOutboundModule** 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, **FTPTypicalOutboundModule**, 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 FTPTypicalOutboundModule (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 **FTPTypicalOutboundModule** 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 **FTPTypicalOutboundModule > Data Types** and you will now see a new data type **Customer** and **Order** under it.

__ b. Double-click **Customer** review the fields inside the object:
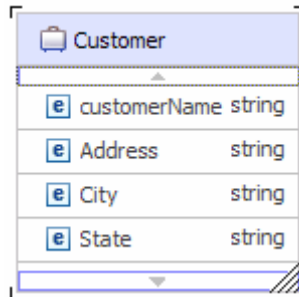
__ c. After reviewing, close the Customer business object from the Assembly editor

_____ 4.   To start the external service wizard from the Palette:

__ a. From the **Palette** on the left side of Assembly Diagram, click **Outbound Adapters**:

_____ 5.   Under Outbound 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)**
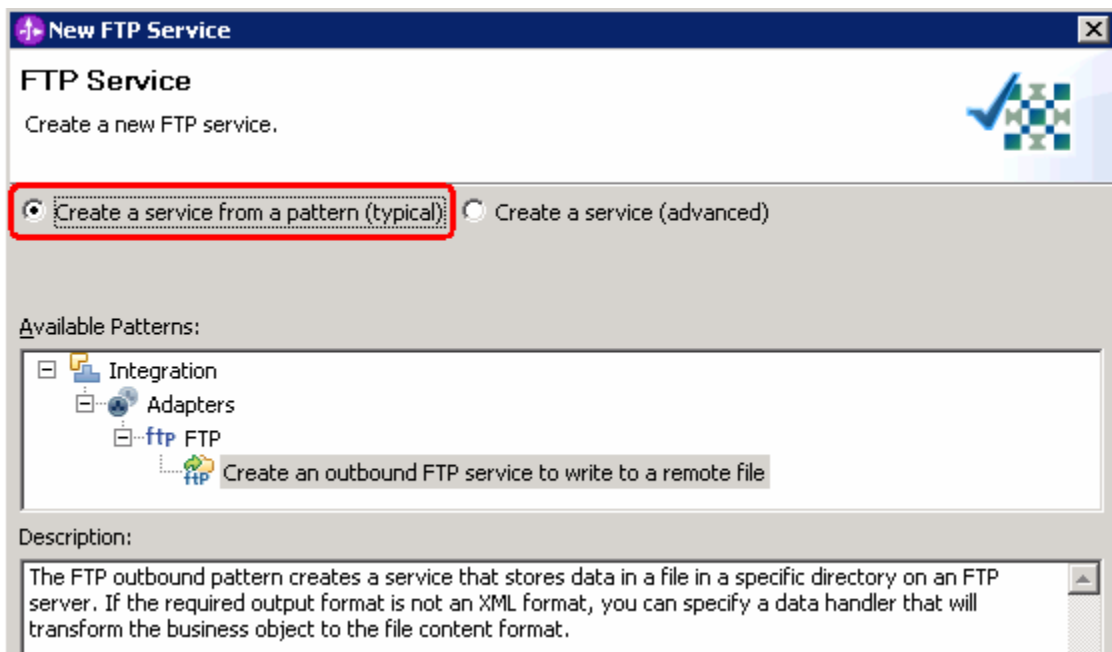
__ a. Click **Next**

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

____ 7.   From the next **FTP service name** screen, for **Name**, accept the default value '**FTPImport**' and click **Next**

Module:	FTPTypicalOutboundModule ▼	Browse... New...
Namespace:	http://FTPTypicalOutboundModule/FTPImport	☑ Default
Folder:		Browse...
Name:	FTPImport	

____ 8.   From the **Business object and FTP server credentials 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**

Data Type Selection

Filter by type, namespace, or file (? = any character, * = any String):

```
*
```
New...

Matching data types:

📋 Customer
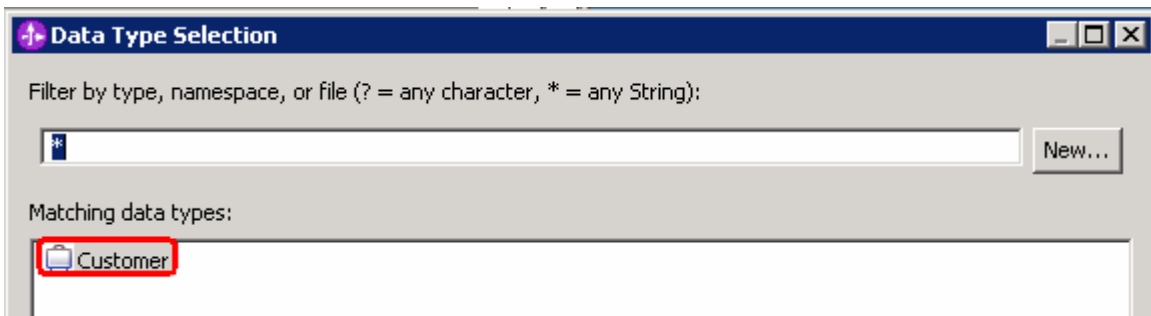
__ c. For **FTP server host name**, enter **<FTP_Machine_Name>** (or IP Address of the machine that has FTP Server), for Ex: wsbeta181.austin.ibm.com

__ d. Click **Test connection**, next to host name and you should get this pop-up window with success message:

Test Results

ⓘ   Connection successful

OK
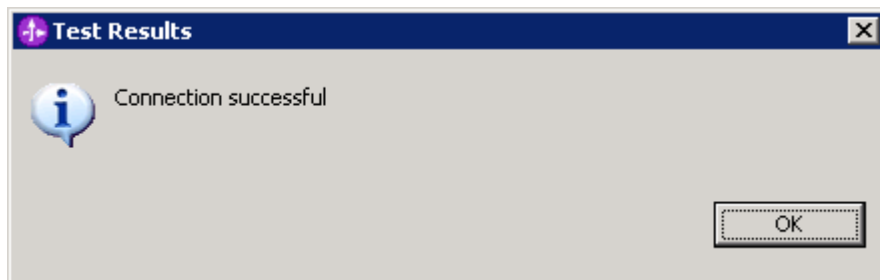
__ e. Click **OK** from the above Test Results window

__ f. For **Remote directory**, enter **full path of the OutDir created on the machine where FTP server is existing** (for Ex: /home/wsbeta/OutDir)

WBPMv62_AdapterFTPOutboundLab.doc

__ g. Your Business object and directory screen should look like this:

__ 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**

__ b. Click **Next**

____ 10.   From Output file name screen, enter these:

__ a. Select the radio button next to Generate a file name with an appended sequence number

1) Accept the default File name, **Customer.txt**

2) For Directory, click **Browse…** and navigate to select **<LOCAL_DIR>**

3) Accept the default value **Customer.seq** for Sequence file name

---

Note: If you select 'Generate a file name with and appended sequence number', the adapter will not generate the wrapper business object, instead uses the File name and Directory entered in this screen.

__ b. Click **Next**

____ 11.   From the Output file format screen, enter these:

   __ a. Select the radio button next to **XML**
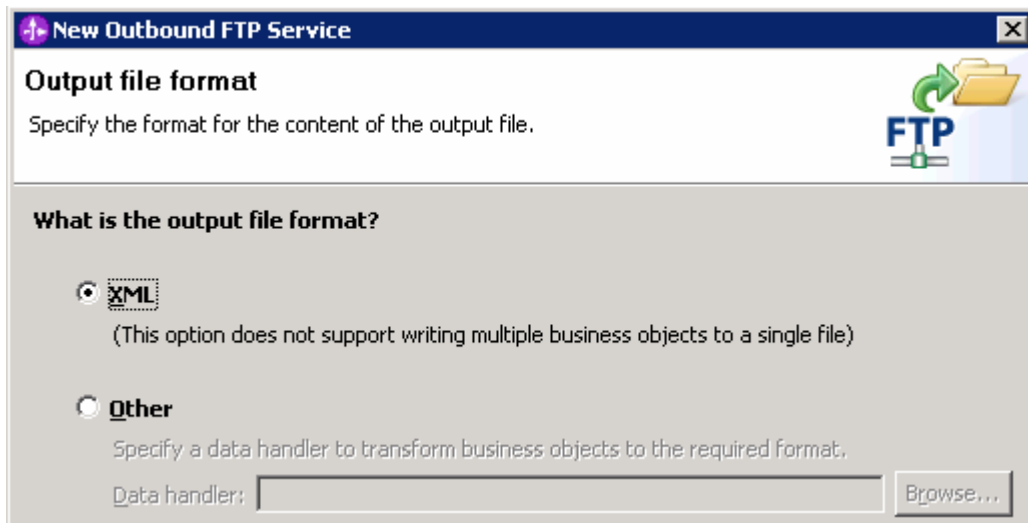
   __ b. Click **Finish**

____ 12.   Save (**Ctrl + S**) changes to your assembly diagram
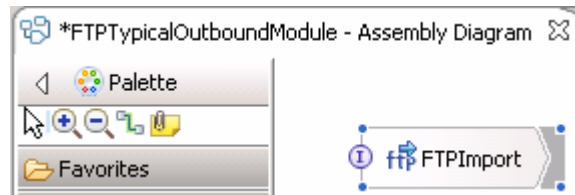
_____ 13. Review the FTPTypicalOutboundModule and the generated artifacts: The generated **Data Types**, **Interface**, Data handler (**UTF8XMLDataHandler**) and Data binding (**FTPXMLDataBinding**) under Configured Resources can be found under FTPTypicalOutboundModule. You can open each of these generated artifacts, business objects and review the properties inside.
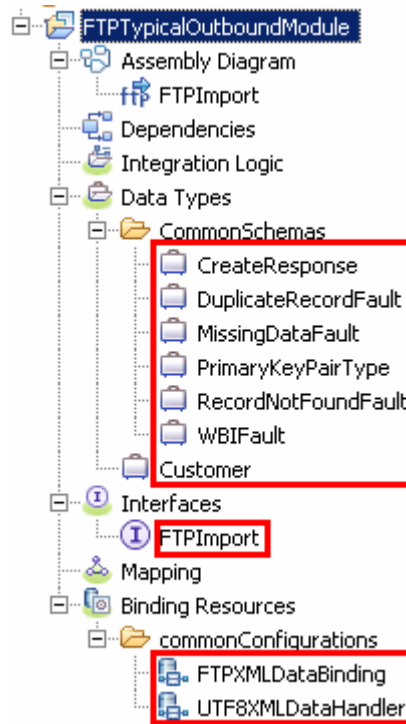
6.2.   Test typical pattern scenario

In this part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application Outbound 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 **FTPTypicalOutboundModuleApp** under Available projects panel and click **Add >**

 __ c. You will now see the **FTPTypicalOutboundModuleApp** 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

 Wait for the project to be published to the server.

____ 2.   Open the test client for the module

 __ a. From the Business Integration perspective, right-click the **FTPTypicalOutboundModule** and select **Test > Test Module**

____ 3.   The **FTPTypicalOutboundModule_Test** window is opened in the Assembly editor

____ 4.   Under **Detailed Properties**:

 __ a. Note the value for Operation, **create**

__ b. Fill out the fields for Initial request parameter:

      1) CustomerName: ABC

      2) Address: 11501 Burnet Rd

      3) City: Austin

      4) State: TX

▶ **General Properties**

▼ **Detailed Properties**

Configuration:	Default Module Test
Module:	FTPTypicalOutboundModule
Component:	FTPImport
Interface:	FTPImport
Operation:	create

☐ Invoke export using binding

Initial request parameters

Name	Type	Value
⊟ 🔲 createInput	Customer	✔
🔲 CustomerName	string	✔ ABC
🔲 Address	string	✔ 11501 Burnet Rd
🔲 City	string	✔ Austin
🔲 State	string	✔ TX

__ c. Click **Continue** button under Events

__ d. From Deployment Location window, select **WebSphere Process Servers > WebSphere Process Server v6.2** and click **Finish**

__ e. You should see a window similar to this, which contains the data you just entered in the previous steps:

Events

⊟ Invoke (FTPImport:create)
⊟ Invoke started
Invoke (FTPImport:create)
Return (FTPImport:create)
Invoke returned

▶ **General Properties**

▼ **Detailed Properties**

Module:	FTPTypicalOutboundModule
Component:	FTPImport
Interface:	FTPImport
Operation:	create

Return parameters:

Name	Type	Value
⊟ 🔲 createOutput	CreateResponse	✔
🔲 Filename	string	✔ Customer.1.txt

Note: Since you have specified a Sequence File name while running the external service wizard, the file name is created with '1' appended to it.

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

_____ 5.   Verify the created file and its contents

     __ a. Open Windows Explorer and browse to the subdirectory **<OUT_DIR>**

     __ b. You will see a new file, **Customer.1.txt** created under that directory. Open the file and you
            should see the content that was entered:

```
<?xml version="1.0" encoding="UTF-8"?>
<p:Customer xsi:type="p:Customer"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.ibm.com/xmlns/
prod/websphere/i2ca/ftp/customer">
<CustomerName>ABC</CustomerName>
<Address>11501 Burnet Rd</Address>
<City>Austin</City>
<State>TX</State>
</p:Customer>
```

_____ 6.   Now using windows explorer, navigate to **<LOCAL_DIR>** and you should see **Customer.seq** file
            created. Open this file using a note pad (or word pad) and you should see a number (2) entered in
            that file. The adapter increases the number by one, each time a file is created in this directory.

_____ 7.   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 **FTPTypicalOutboundModuleApp** 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

What you did in this exercise

In this lab, you imported the FTP Adapter RAR file into your WebSphere Integration Developer workspace and integrated it into an SCA application that creates a file to the file system. Next, you made use of the external service wizard available in WebSphere Integration Developer to specify Activation Spec Properties, define Data binding, Data handler, and Operations which, after deploying onto the server, will generate Business Objects and other artifacts for different scenarios.

At the end of each part, you deployed and then tested the adapter application for these scenarios - pass-through test scenario (create, append, list, retrieve, ExecuteFTPScript Operations), content specific or non pass through test scenario (create, append, and retrieve operations), using all defaults (default data binding) scenario (create operation), and then finally using the typical pattern (create operation).

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**.
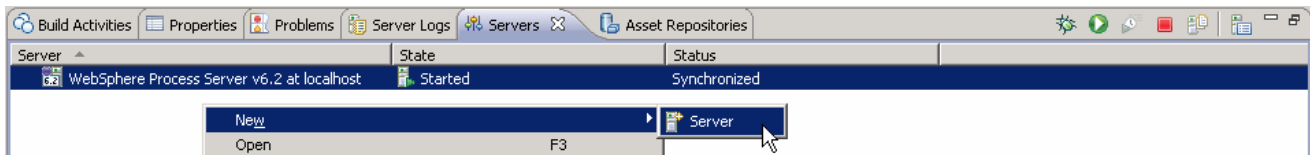
   __ c. In the New Server dialog, specify the remote server's host name, **<HOSTNAME>**.

   __ 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.**

WBPMv6.2_AdapterFTPOutboundLab.doc
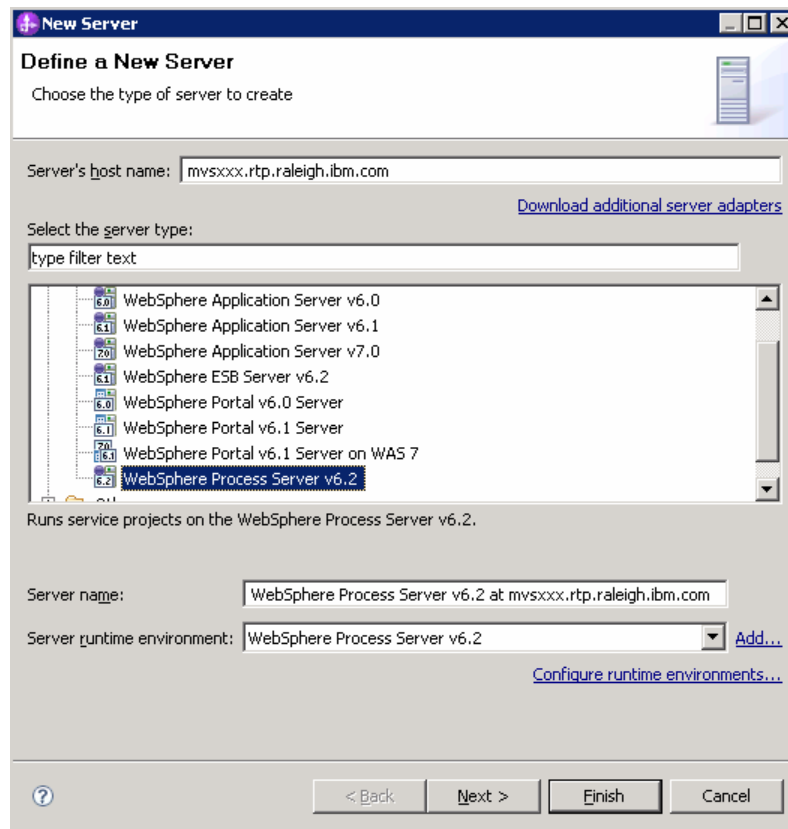
IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

__ 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.
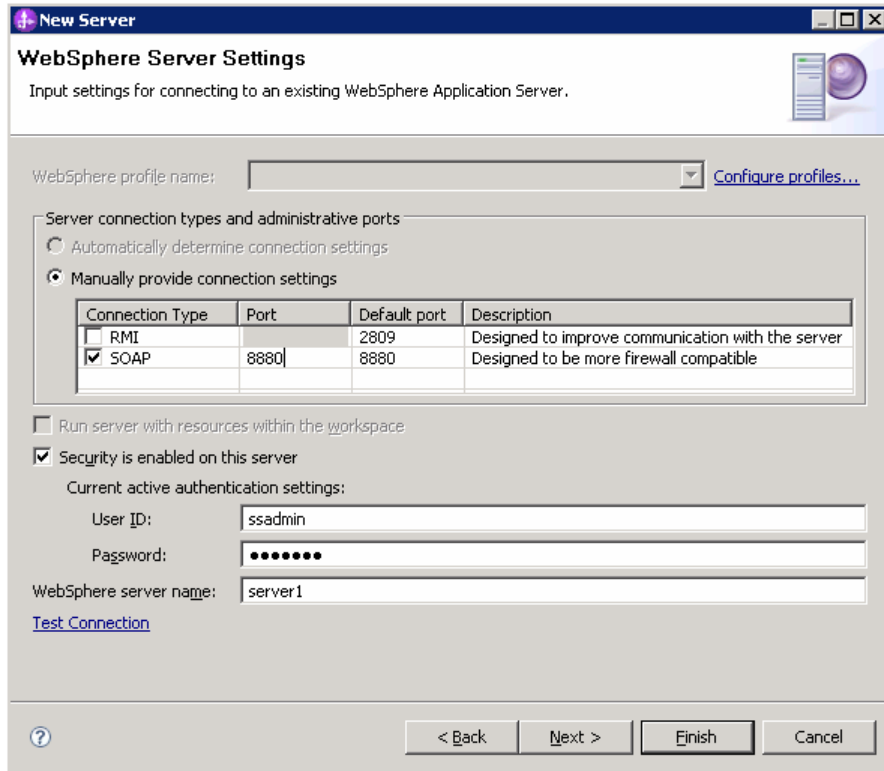
__ i. Click **Finish**.

__ 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
```