# FTP Adapter outbound lab

What th	nis exercise is about1
Lab rec	quirements2
What y	ou should be able to do2
Introdu	ction3
Exercis	se instructions4
Part 1:	Initialize the workspace and prepare for the lab
Part 2:	Review properties7
Part 3:	Pass through scenario9
3.1.	Configure pass through using the external service wizard10
3.2.	Test pass through scenario
3.3.	Restore Server Configuration
Part 4:	Content specific (non-pass through) scenario
4.1.	Configure non-pass through using the external service wizard40
4.2.	Test content specific scenario
4.3.	Restore server configuration
Part 5:	Use default data binding65
5.1.	Configure outbound using data binding
5.2.	Test all defaults scenario
Part 6:	Use 'Create a service from a typical pattern'72
6.1.	Configure outbound using 'Create a service from a pattern (typical)' option73
6.2.	Test typical pattern scenario
What y	ou did in this exercise
Task: A	Adding remote server to WebSphere Integration Developer test environment

# 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

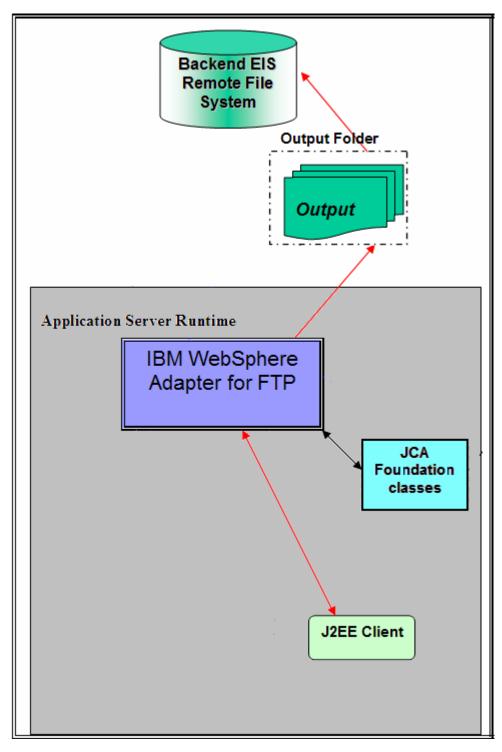
List of system and software required for the student to complete the lab

- 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 passthrough and non pass-through using different scenarios and patterns
- Restore the server configuration

# Introduction



The Java<sup>™</sup> 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></wid_home>	C:\Program Files\IBM\WID62	
<wps_home></wps_home>	C:\ <wid_home>\runtimes\bi_v62</wid_home>	
<ftpadapter_home></ftpadapter_home>	<wid_home>\ResourceAdapters\FTP_6.2.0.0\deploy</wid_home>	
<lab_files></lab_files>	C:\Labfiles62	/tmp/Labfiles62
<workspace></workspace>	<lab_files>\FTPOutbound\workspace</lab_files>	
<out_dir></out_dir>	<lab_files>\ FTPOutbound\output</lab_files>	
<retrive_archive></retrive_archive>	<lab_files>\ FTPOutbound\retrievearchive</lab_files>	
<ftpfiles></ftpfiles>	<lab_files>\FTPFiles</lab_files>	
<local_dir></local_dir>	<lab_files>\FTPOutbound\LocalDir</lab_files>	
<temp></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 <sup>®</sup> test server location	Input your values for the remote location of the test server
<server_name></server_name>	server1	sssr011	
<was_home></was_home>	C:\Program Files\IBM\WebSphere\AppServer	/etc/sscell/AppServer	
<hostname></hostname>	localhost	mvsxxx.rtp.raleigh.ibm.com	
<soap_port></soap_port>	8880	8880	
<telnet_port></telnet_port>	N/A	1023	
<profile_name></profile_name>	AppSrv01	default	
<userid></userid>	N/A	ssadmin	
<password></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".

# 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

🚯 Workspace Launcher		×
Select a workspace		
IBM WebSphere Integration Developer 6.2 stores your projects in a folder called a workspace. Choose a workspace folder to use for this session.		
Workspace: C:\Labfiles62\FTPOutbound\workspace	•	Browse
▶ Topy Settings		
Ок		Cancel

- 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.

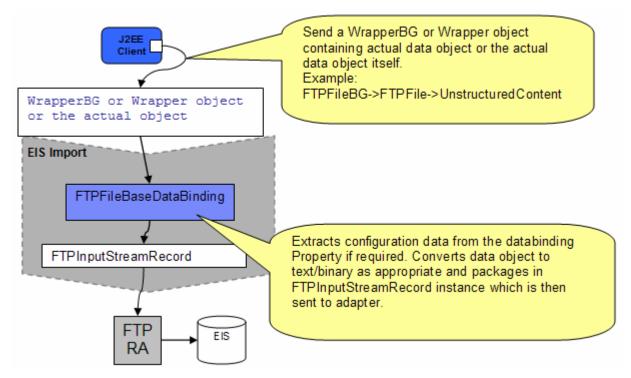
If splitFunctionClassName is set to com. ibm.j2ca.utils.filesplit.SplitByDelimiter, then splitCriteria must contain the delimiter which separates the BO's in the retrieved file.

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**

🚯 New Module	×
<b>Module</b> Create a new business integration module. A module is a project that is used for development, version management, organizing resources, and deploying to the runtime	
Module Name: FTPPSOutboundModule	
Use default location     Location: C:/Labfiles62/FTPOutbound/workspace/FTPPSOutboundModule	Browse
Open module assembly diagram	
Business integration modules can be deployed and run on WebSphere Process Server. The many types of components, such as business processes, assembled together for the purp integration.	

You will now see a new module, FTPPSOutboundModule, created in your Business Integration window

- \_\_\_\_\_2. To start the external service from the Palette:
  - \_\_\_\_a. From the **Palette** on the left side of Assembly Diagram, click **Outbound Adapters**:

	e - Assembly Diagram	x
Contraction   Contraction		
E Favorites		
Components	_	
a cics		
i IMS		
Flat File		
iSerie An import compo	pept which	
JDBC connects to a rer	note file	
📸 Oracle		
🚰 PeopleSoft ன SAP		
😴 Siebel		

- \_\_\_\_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)

🚯 New FTP Service	×
FTP Service	C
Create a new FTP service.	21
C Create a service from a pattern (typical) 💽 Create a service (advanced)	
<u>A</u> vailable Patterns:	
E La Integration	
E-ftp FTP	
Create an outbound FTP service to write to a remote file	
Description:	
The FTP outbound pattern creates a service that stores data in a file in a specific directory on an FTP server. If the required output format is not an XML format, you can specify a data handler that will transform the business object to the file content format.	*

#### \_\_\_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** 

🚯 External Service	_ 🗆 🗙
Select an Adapter Select the adapter you want to use.	0
IBM WebSphere Adapter for FTP (IBM : 6.2.0.0)	

\_\_\_ 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

🚯 External Service	
Adapter Import	
Import a resource ad for the adapter.	apter archive (RAR) from the file system to create a connector project
Archive file:	C:\IBM\WID62\ResourceAdapters\FTP_6.2.0.0\deploy\CWYFT_FTPFile.rar
Connector project:*	CWYFT_FTPFile
Target runtime:	WebSphere Process Server v6.2

\_\_\_ 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 serve	r' for more details o	n 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

Deploy connector proje	ct: With module for use by single application 🔫 🔽
Connection properties:	Use properties below
Connection properties	
FTP system connec	tion information
Host name: *	wsbeta181.austin.ibm.com <del></del>
Directory: *	/home/wsbeta/Output
Protocol:	FTP - file transfer protocol 🛛 🗧 🚽
Port number:	21 🗲 🗕
The user name	and password will not be encrypted and will be stored as plain text.
User name:	wsbeta <del> –</del>
Password:	*****

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

<< Advanced
Advanced connection configuration
Socks proxy server connection information
Second FTP system connection information
Secure configuration
Logging and tracing
Bidi properties

# \_\_\_\_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.

<ul> <li>Advanced connection configuration         The staging directory is used to store files temporarily to avoid write conflicts.         Staging directory:         Default target file name:         To add sequence numbers to target file names, specify the location of a sequence file.         Sequence file:         C:\Labfiles62\FTPOutbound\SequenceFile.tx+++         Browse         Populate the fully qualified class name of the custom parser which is used to parse the "ls -l" output. This is used only when the "ls -l" output deviates from standard output.         Browse         Browse</li></ul>				
Staging directory:         Default target file name:         To add sequence numbers to target file names, specify the location of a sequence file.         Sequence file:       C:\Labfiles62\FTPOutbound\SequenceFile.txtcf         Encoding used by FTP server:       Select         Populate the fully qualified class name of the custom parser which is used to parse the "Is -I" output. This is used only when the "Is -I" output deviates from standard output.	Ŧ	Advanced connection configura	tion	
Default target file name:         To add sequence numbers to target file names, specify the location of a sequence file.         Sequence file:       C:\Labfiles62\FTPOutbound\SequenceFile.txtcBrowse         Encoding used by FTP server:       Select         Populate the fully qualified class name of the custom parser which is used to parse the "Is -I" output. This is used only when the "Is -I" output deviates from standard output.		The staging directory is used to	store files temporarily to avoid write conflicts.	
To add sequence numbers to target file names, specify the location of a sequence file. Sequence file: C:\Labfiles62\FTPOutbound\SequenceFile.txtet Browse Encoding used by FTP server: Select Populate the fully qualified class name of the custom parser which is used to parse the "Is -I" output. This is used only when the "Is -I" output deviates from standard output.		Staging directory:		
Sequence file:       C:\Labfiles62\FTPOutbound\SequenceFile.tx       Browse         Encoding used by FTP server:       Select         Populate the fully qualified class name of the custom parser which is used to parse the "Is -I" output. This is used only when the "Is -I" output deviates from standard output.		Default target file name:		
Encoding used by FTP server: Select Populate the fully qualified class name of the custom parser which is used to parse the "Is -I" output. This is used only when the "Is -I" output deviates from standard output.		To add sequence numbers to target file names, specify the location of a sequence file.		
Populate the fully qualified class name of the custom parser which is used to parse the "Is -I" output. This is used only when the "Is -I" output deviates from standard output.		Sequence file:	C:\Labfiles62\FTPOutbound\SequenceFile.tx	Browse
output. This is used only when the "Is -I" output deviates from standard output.		Encoding used by FTP server:		Select
Custom parser class name: Browse				e the "Is -I"
		Custom parser class name:		Browse

\_\_\_7. Secure configuration: Refer to the new lab 'Install and configure SSH server' for more details on this new feature

👻 Secure configura	tion	
🔲 Enable remot	e server authentication for SFTP protocol	
Host key file:		Browse
Private key file:		Browse
Passphrase:		

8. Logging and tracing: Refer to the new lab 'Log and confidential trace lab' for more details on this new feature

C Logging and tracing		
Adapter ID:*	001	
Disguise user data as "XXX" in log and trace files.		

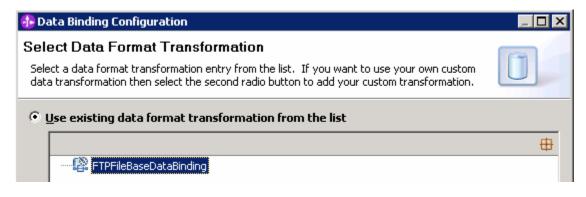
9. For this lab, you are not going to use the J2C authentication. So, **uncheck** the box next to **Specify a Java Authentication and Authorization Services (JAAS) alias security credentials**.

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



\_\_\_\_ 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

	🚯 Data Binding Configu	ration	_ 🗆 🗙
	Specify the properties for	the data transformation.	
Select DataBinding if you want to use a data binding developed for earlier versions of the			the adapter.
	Binding type:	DataHandler 🗾	
	Configured data handler:	Not defined	<u>S</u> elect
	Configured data binding;	Not defined	S <u>e</u> lect

- \_\_\_\_\_f. Note that the selected module is **FTPPSOutboundModule** 
  - 1) For the Name, enter FTPPSDB

🤂 Binding Resource Configuration 📃 🗖 🗙						
New Data 7	New Data Transformation Configuration					
Create a new namespace, ar						
<u>M</u> odule:	FTPPSOutboundModule Browse	New				
Namespace:	http://FTPPSOutboundModule					
F <u>o</u> lder:	Browse					
N <u>a</u> me:	FTPPSDB 🗲					

2) Click Finish

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

Data format options:	Use a data format configuration for all operations	<b>-</b>	
Data format:	* FTPPSDB		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. 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.

#### **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'

🚯 Add Operation	
Operation	
Specify the properties for the operation to add.	
Operation kind:	Create 🔫 🔫
Operation properties	
The data type for the operation input:	Generic FTP business object with business graph 🔫 🚽
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** 

Operation name: *	createFTPFile			
Specify the operation inp				
Input type: 🔶	FTPFileBG {http://www.ibm.com/xmlns/prod/websphere/j2c-	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/websj	Browse,,,	New	
Data format options:	Use suggested data format 'FTPFileBaseDataBinding'			
Data format:	Not defined	Select		

#### \_\_\_\_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

🚯 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			
	<b>#</b>		
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:

Operation name:	* createFTPBG				
Specify the operation input					
Input type:	FTPFileBG {http://www.ibm.com/xmlns/prod/websphere/j2c.	Browse,	New		
Data format options	: Use a data format configuration 🛛 🔫 🛨				
Data format:	* FTPPSDB 🛛 🗧	Select			
Specify the operation output					
Output type;	CreateResponseBG {http://www.ibm.com/xmlns/prod/websj	Browse,	New		
Data format options	: Use a data format configuration 🛛 🔫 🛨				
Data format:	* FTPPSDB	Select			

\_\_\_\_I. 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

Operations:		
🏶 createFTPBG ({http://www.ibm.com/xmlns/proc	d/websphere/j2ca/ftp/ftpfilebg}F <b>746</b> 6666	, Add
		Edit
-1		Remove
		<u></u>
Operation properties:		
InteractionSpec properties for 'createFTPBG'		
FTP system connection information		
Remote directory on FTP system:		
Default target file name:		
File in local directory		
Local directory:	E	Browse
Archive file in the local directory for create	e operation	
Local archive directory for create operation:	E	Browse
$\square$ Create new file if the file does not exist		
🔲 Generate a unique file		
$\square$ Delete the file after retrieve operation		
Remote archive directory for retrieve operation:		
<< Advanced		
Advanced connection configuration		
Second FTP system connection information		

#### **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'

Operation kind:	Append	•
Operation properties		
The data type for the operation input:	Generic FTP business object with business graph	•
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 **AppendResponseBG** 

Operation name: *	appendFTPFile		
Specify the operation inp	out		
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 ou	tput		
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 ir	nput		
Input type:	FTPFileBG {http://www.ibm.com/xmlns/prod/websphen	Browse	New
Data format option	s: Use a data format configuration 🛛 🔫 🛨		
Data format:	* FTPPSDB	Select	
Specify the operation o	utput		
Output type:	AppendResponseBG {http://www.ibm.com/xmlns/prod/	Browse,	New
Data format option	s: 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

Operations:	
createFTPBG ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfilebg}FTP	
appendFTPBG ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfilebg}FT	PFileBG) Edit
	Remove
	- Nonove
Operation properties: InteractionSpec properties for 'appendFTPBG'	
FTP system connection information	
Remote directory on FTP system:	
Default target file name:	_
File in local directory	
Local directory:	Browse
$\square$ Archive file in the local directory for create operation	
Local archive directory for create operation:	Browse
Create new file if the file does not exist	
🗖 Generate a unique file	
Delete the file after retrieve operation	
Remote archive directory for retrieve operation:	_
Advanced >>	

## 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

Operation kind:	List
Operation properties	
The data type for the operation input:	Generic FTP business object
Enable response type for the operation	1

\_\_\_ 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** 

Operation name: *	listFTPFile		
Specify the operation inp	ut		
Input type: 🔶	FTPFile {http://www.ibm.com/xmlns/prod/websphere/j;	Browse	New
Data format options:	Use suggested data format 'FTPFileBaseDataBinding'		
Data format:	Not defined	Select	
Specify the operation out	put		
Output type: 🔶	ListResponse -{http://www.ibm.com/xmlns/prod/webspl	Browse	New
Data format options:	Use suggested data format 'FTPFileBaseDataBinding'		
Data format:	Not defined	Select,	

\_\_\_\_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:

Operation name: *	listFTP		
Specify the operation inp	ut		
Input type:	FTPFile {http://www.ibm.com/xmlns/prod/websphere/j;	Browse,	New
Data format options:	Use a data format configuration		
Data format: *	FTPPSDB	Select	
Specify the operation out	:put		
Output type;	ListResponse {http://www.ibm.com/xmlns/prod/webspl	Browse,	New
Data format options:	Use a data format configuration		
Data format: *	FTPPSDB	Select	

\_\_\_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

\_\_\_\_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:

Operation name: *	retrieveFTP		
Specify the operation inp	ut		
Input type:	FTPFile {http://www.ibm.com/xmlns/prod/websphere/j;	Browse Ne	BW
Data format options:	Use a data format configuration		
Data format: *	FTPPSDB	Select	
Specify the operation ou	put		
Output type: *	RetrieveResponseWrapper {http://www.ibm.com/xmln	Browse Ne	ew
Data format options:	Use data format configuration 'FTPPSDB' 🗧 🗲 📩		
Data format:	Not defined	Select	

\_\_\_\_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

\_\_\_\_ c. You should see this:

Operation name:	* executeFTPScript		
Specify the operation in	nput		
Input type:	FTPFileBG {http://www.ibm.com/xmlns/prod/websphere	Browse	New
Data format option	s: Use a data format configuration 🛛 🔫 🛨		
Data format:	* FTPPSDB	Select	
Specify the operation of	utput		
Output type;	ExecuteFTPScriptResponseBG {http://www.ibm.com/xr	Browse	New
Data format option	s: Use a data format configuration 🛛 🛶 🛨		
Data format:	* FTPPSDB	Select	

\_\_\_\_ 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:

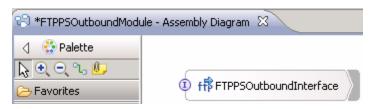
Operations:	
createFTPBG ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfilebg}FTPFileBG) : {http://	Add
appendFTPBG ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfilebg}FTPFileBG) : {htt	
🛛 🕂 🕲 listFTP ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfile}FTPFile) : {http://www.ibn	Edit
<pre>     retrieveFTP ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfile}FTPFile) : {http://ww     websphere/j2ca/ftp/ftpfile}FTPFile) : {http://ww     websphere/j2ca/ftp/ftpfilebo}ETPFileBG) : </pre>	Remove
🐘 🏶 executeETPScript ({http://www.ibm.com/xmlps/prod/webspbere/i2ca/ftp/ftpfilebo}ETPEileBG) : 🎴	

- \_\_\_\_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

Properties for serv	vice	
Module;	FTPPSOutboundModule	New
Namespace;	http://FTPPSOutboundModule/FTPPSOutboundInterface	
	✓ Use default namespace	
Name: *	FTPPSOutboundInterface	
	Save business objects to a library	
Library:		New
Description:		

## \_\_\_\_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

Ė

22. Review the FTPPSOutboundModule: The generated **Data Types**, **Interface**, and the Data binding (**FTPPSDB**) under Configured Resources can be found inside FTPPSOutboundModule

🛱 🖓 Assembly Diagram
FTPPSOutboundInterface
🚊 🔁 Data Types
🖻 🕞 CommonSchemas
🚽 🗐 AppendResponse
🔒 💼 AppendResponseBG
📄 🛄 CreateResponse
<ul> <li>AppendResponseBG</li> <li>CreateResponse</li> <li>CreateResponseBG</li> <li>DuplicateRecordFault</li> <li>ExecuteFTPScriptResponse</li> <li>FileContent</li> <li>FileContentBG</li> <li>FTPFile</li> <li>FTPFile</li> <li>FTPFileBG</li> <li>ListResponse</li> <li>MissingDataFault</li> <li>PrimaryKeyPairType</li> <li>RecordNotFoundFault</li> <li>RetrieveResponseWrapper</li> <li>UnstructuredContent</li> </ul>
🕂 💭 DuplicateRecordFault
🕂 🛱 ExecuteFTPScriptResponse
🕂 🛱 FileContent
🕂 🗓 FileContentBG
🕞 💭 FTPFile
🔒 🗓 FTPFileBG
🕛 🛄 ListResponse
🕛 💭 MissingDataFault
🕛 🛄 PrimaryKeyPairType
🕂 💭 RecordNotFoundFault
🕂 📋 RetrieveResponseWrapper
🕂 🛱 UnstructuredContent
I I WDIFAUL
🖃 🕘 Interfaces
- 🍐 Mapping
🖻 💿 Configured Resources
FTPPSDB

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:

Operations	🗱 🎶 🖒 🕼 🕷 👘				
perations and their pa	1				
	Name	Туре			
▼ ScreateFTPBG					
Input(s)	createFTPBGInput	FTPFileBG			
🕼 Output(s)	createFTPBGOutput	CreateResponseBG			
🔀 Fault	MISSING_DATA	MissingDataFault			
🔀 Fault	WBIFault	WBIFault			
🔀 Fault	DUPLICATE_RECORD	DuplicateRecordFault			
🔀 Fault	RECORD_NOT_FOUND	RecordNotFoundFault			
<b>▼</b> ‱appendFTPBG					
Input(s)	appendFTPBGInput	FTPFileBG			
📫 Output(s)	appendFTPBGOutput	AppendResponseBG			
🔀 Fault	MISSING_DATA	MissingDataFault			
🔀 Fault	WBIFault	WBIFault			
🔀 Fault	RECORD_NOT_FOUND	RecordNotFoundFault			
<b>▼</b> WistFTP					
Input(s)	listFTPInput	FTPFile			
📫 Output(s)	listFTPOutput	ListResponse			
🔀 Fault	MISSING_DATA	MissingDataFault			
🔀 Fault	WBIFault	WBIFault			
🔀 Fault	RECORD_NOT_FOUND	RecordNotFoundFault			
🕶 🌠 listFTP					
沟 Input(s)	listFTPInput	FTPFile			
📫 Output(s)	listFTPOutput	ListResponse			
🔀 Fault	MISSING_DATA	MissingDataFault			
🔀 Fault	WBIFault	WBIFault			
😹 Fault	RECORD_NOT_FOUND	RecordNotFoundFault			
➡ 20 TetrieveFTP					
🕼 Input(s)	retrieveFTPInput	FTPFile			
📫 Output(s)	retrieveFTPOutput	RetrieveResponseWrapper			
🔀 Fault	MISSING_DATA	MissingDataFault			
🔀 Fault	WBIFault	WBIFault			
🔀 Fault	RECORD_NOT_FOUND	RecordNotFoundFault			
🕶 👹 executeFTPScrip	bt.				
🕅 Input(s)	executeFTPScriptInput	FTPFile			
🕼 Output(s)	executeFTPScriptOutput	ExecuteFTPScriptResponse			
🔀 Fault	MISSING_DATA	MissingDataFault			
🔀 Fault	WBIFault	WBIFault			
🔀 Fault	RECORD_NOT_FOUND	RecordNotFoundFault			

\_\_\_ c. Close the interface, FTPPSOutboundInterface

# 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

\_\_\_\_b. For Filename, enter any name, for Ex: PassthroughTest.txt

General Pro	General Properties	
<ul> <li>Detailed Pro</li> </ul>	perties	
Configuration:	Default Module Test	•
Module:	FTPPSOutboundModule	•
Component:	FTPPSOutboundInterface	•
Interface:	FTPPSOutboundInterface	•
Operation:	createFTPBG	•

Initial request parameters

Ę	🖞   🕞   🗖		
	Name	Туре	Value
	🖃 🏪 createFTPBGInput	FTPFileBG	<ul> <li>✓</li> </ul>
	💭 verb	verb <string></string>	✓ CREATE
	🗄 📲 FTPFile	FTPFile	✓
	🖳 🛄 DirectoryPath	string	×
	💭 Filename	string	<ul> <li>PassThroughTest.txt</li> </ul>

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

🗄 🖳 Content	UnstructuredContent	✓
ContentType	string	✓
DojectName	string	✓
抑 AsText	string	<ul> <li>Testing Create operation for pass through</li> </ul>
🛄 AsBinary	hexBinary	✓ 0

\_\_\_\_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

Deployment location:	
🖃 🔤 WebSphere Process Servers	New Server
Eclipse 1.5 JVM	

\_\_\_\_f. Provide Administrator User ID and Password

 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

🚯 User Login - Default Module Test 🛛 🗙
Security is enabled on the selected runtime environment(s). Please sign in to continue the test. User ID:
admin
Password:
•••••
$\checkmark$ Use the authentication settings in the preference and never ask again.
OK Cancel

\_\_\_ 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

🚯 Adding/Removing Projects	×
The projects are being added/removed from the server.	
Do not show again	

\_\_\_\_ 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.

Events	General Properties		
↓ - □	<ul> <li>Detailed Properties</li> </ul>		
<ul> <li>K Invoke (FTPPSOutboundInterface:createFTPBG)</li> <li>K Invoke started</li> <li>Invoke (FTPPSOutboundInterface:createFTPBG)</li> <li>Return (FTPPSOutboundInterface:createFTPBG)</li> <li>Invoke returned</li> </ul>	Module: <u>FTPPSOutboundModul</u> Component: <u>FTPPSOutboundInterf</u> Interface: <u>FTPPSOutboundInterf</u> Operation: <u>createFTPBG</u>	ace	
	<u>R</u> eturn parameters:		
	📴 8: 🚾		
	Name	Туре	Value
	🖃 🖳 createFTPBGOutput	CreateResponseBG	¥
	verb	string	8
	🗄 🖳 CreateResponse	CreateResponse	✓
	Filename	string	✓ PassThroughTest.1.txt

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 (
- \_\_\_\_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 Prop	General Properties		
<ul> <li>Detailed Pro</li> </ul>	<ul> <li>Detailed Properties</li> </ul>		
Configuration:	Default Module Test	•	
Module:	FTPPSOutboundModule	•	
Component:	FTPPSOutboundInterface	•	
Interface:	FTPPSOutboundInterface	•	
Operation:	appendFTPBG	•	

#### Initial request parameters

Ę			
	Name	Туре	Value
	appendFTPBGInput	FTPFileBG	✓
	····Ì⊐ verb	verb <string></string>	✓ CREATE
	P-P- 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	<ul> <li>Test message for append operation</li> </ul>
🛄 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	General Properties			
🏞 ▾ 🔳 🖆 😥 🔳 🔛	<ul> <li>Detailed Properties</li> </ul>			
<ul> <li>Invoke (FTPPSOutboundInterface:createFTPBG)</li> <li>Invoke started</li> <li>Invoke (FTPPSOutboundInterface:createFTPBG)</li> <li>Return (FTPPSOutboundInterface:createFTPBG)</li> <li>Invoke returned</li> <li>Invoke (FTPPSOutboundInterface:appendFTPBG)</li> <li>Invoke started</li> <li>Invoke (FTPPSOutboundInterface:appendFTPBG)</li> </ul>	Module:     ETPPSOutboundMode       Component:     FTPPSOutboundInter       Interface:     FTPPSOutboundInter       Operation:     appendFTPBG       Return parameters:     Image: Im	<u>face</u>		
Return (FTPPSOutboundInterface:appendFTPBG)	Name	Туре		
Invoke returned	🖃 🖳 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 pas	throug	rest	message	for	Append	operation
~									
~									
~									
~									
"PassTh	roughTes	st.1.txt"	[noeol]	1L, 74C					
	0			-					

#### Test List operation:

- \_\_\_\_ 9. Click Invoke (
- 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:

<ul> <li>Detailed Properties</li> </ul>			
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

Events	General Properties			
🏞 ▾ 🔳 🍰 🗭 🔳 🔛	<ul> <li>Detailed Properties</li> </ul>			
<ul> <li>Invoke (FTPPSOutboundInterface:createFTPBG)</li> <li>Invoke started</li> <li>Invoke (FTPPSOutboundInterface:createFTPBG)</li> <li>Return (FTPPSOutboundInterface:createFTPBG)</li> <li>Invoke returned</li> <li>Invoke (FTPPSOutboundInterface:appendFTPBG)</li> <li>Invoke started</li> <li>Invoke (FTPPSOutboundInterface:appendFTPBG)</li> </ul>	Module:     FTPPSOutboundModule       Component:     FTPPSOutboundInterfa       Interface:     FTPPSOutboundInterfa       Operation:     listFTP       Return parameters:     Image: State	ace		
🤎 Return (FTPPSOutboundInterface:appendFTPBG)	Name Name	Туре		
🔲 Invoke returned	🗆 🖳 listFTPOutput	ListResponse	✓	
🖃 ừ Invoke (FTPPSOutboundInterface:listFTP)	🗄 🗄 🗄 ListOfFileNames	List <object></object>	×	
🖃 隆 Invoke started	ListOfFileNames	String	<ul> <li>PassThroughTest.1.txt</li> </ul>	
💚 Invoke (FTPPSOutboundInterface:listFTP)				
Return (FTPPSOutboundInterface:listFTP)     Invoke returned				

#### Test Retrieve operation:

\_\_\_\_ 12. Click Invoke (

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	Туре	Value
🖃 🖳 retrieveFTPInput	FTPFile	✓
🖳 🛄 DirectoryPath	string	×
江 Filename	string	PassThroughTest.1.txt
江 ChunkInfo	string	×
- In FtpServerHostName	string	×
- EtpServerEventDirectory	string	×
- IntaConnectionMode	string	×
- IleTransferType	string	✓
- Implication SecondServerDirectory	string	✓
- Implication SecondServerUsername	string	✓
- Implication SecondServerPassword	string	✓
- IleContentEncoding	string	✓
- İncludeEndBODelimiter	string	✓
FileInLocalDirectory	boolean	✓ false
- IocalDirectoryPath	string	✓
- IocalArchivingEnabledForCreate	boolean	✓ false
- IocalArchiveDirForCreate	string	✓
- Implimite StagingDirectory	string	✓
🏣 GenerateUniqueFile	boolean	✓ false
	boolean	✓ false
[] ScriptFileParameters	string[]	66
- ImplittingFunctionClassName	string	×
🎞 SplitCriteria	string	✓
- IndeteOnRetrieve	boolean	✓ true
	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:

Events	General Properties		
↓ - □ ↓ 0 □ □	<ul> <li>Detailed Properties</li> </ul>		
Invoke (FTPPSOutboundInterface:createFTPBG)     Invoke started     Invoke started     Invoke (FTPPSOutboundInterface:createFTPBG)     Invoke (FTPPSOutboundInterface:createFTPBG)     Invoke returned     Invoke (FTPPSOutboundInterface:appendFTPBG)     Invoke (FTPPSOutboundInterface:appendFTPBG)     Invoke (FTPPSOutboundInterface:appendFTPBG)     Invoke (FTPPSOutboundInterface:appendFTPBG)	Module: ETPPSOutboundModule Component: ETPPSOutboundInterface Interface: ETPPSOutboundInterface Operation: retrieveFTP Return parameters:		
<ul> <li>Invoke started</li> <li>Invoke (FTPPSOutboundInterface:appendFTPBG)</li> </ul>	E. 8. 🚾		
Return (FTPPSOutboundInterface:appendFTPBG)	Name Name	Туре	Value
Invoke returned	🖃 🖳 retrieveFTPOutput	RetrieveResponseWr	✓
Invoke returned % Invoke (FTPPSOutboundInterface:listFTP)	Content	RetrieveResponseWr Object[] <object></object>	✓ ✓
🖂 ĭ Invoke (FTPPSOutboundInterface:listFTP)	Content	Object[] <object></object>	
<ul> <li>□ M Invoke (FTPPSOutboundInterface:listFTP)</li> <li>□ P Invoke started</li> </ul>	Content	Object[] <object> FileContent</object>	× *
W Invoke (FTPPSOutboundInterface:listFTP)     Nvoke started     Invoke (FTPPSOutboundInterface:listFTP)	Content	Object[] <object> FileContent String</object>	× *
W Invoke (FTPPSOutboundInterface:listFTP)     Nvoke started     Invoke (FTPPSOutboundInterface:listFTP)     W Return (FTPPSOutboundInterface:listFTP)	Content	Object[] <object> FileContent String UnstructuredContent</object>	✓ ✓ PassThroughTest.1.bxt
<ul> <li>➡ Invoke (FTPPSOutboundInterface:listFTP)</li> <li>➡ Invoke started</li> <li>➡ Invoke (FTPPSOutboundInterface:listFTP)</li> <li>➡ Invoke (FTPPSOutboundInterface:listFTP)</li> <li>■ Invoke returned</li> </ul>	Content	Object[] <object> FileContent String UnstructuredContent String</object>	PassTwoughTest.1.bxt
<ul> <li>Invoke (FTPPSOutboundInterface:listFTP)</li> <li>▷ Invoke started</li> <li>♥ Invoke (FTPPSOutboundInterface:listFTP)</li> <li>♥ Return (FTPPSOutboundInterface:listFTP)</li> <li>■ Invoke returned</li> <li>♥ Invoke (FTPPSOutboundInterface:retrieveFTP)</li> </ul>	Content	Object[] <object> FileContent String UnstructuredContent String String</object>	PassThroughTest.1.txt  PassThroughTest.1.txt  http://www.ibm.com/mins/prod/websphere/j2ca/ttp/retrieveresponsewrapper
Winvoke (FTPPSOutboundInterface:listFTP)     Winvoke started     Winvoke (FTPPSOutboundInterface:listFTP)     Winvoke (FTPPSOutboundInterface:listFTP)     Winvoke returned     Winvoke (FTPPSOutboundInterface:retrieveFTP)     Winvoke started	Content	Object] <object> FileContent String UnstructuredContent String String String [B</object>	PassThroughTest.1.txt  PassThroughTest.1.txt   http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/retrieveresponsewrapper  http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/retrieveresponsewrapper  Testing Create operation for pass throughTest message for Append operation

\_\_\_\_ 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:

🕞 ScriptFile.txt - Notepad	_ 🗆 🗵
<u>File E</u> dit F <u>o</u> rmat <u>V</u> iew <u>H</u> elp	
%1 cd /download/guestusr mkdir %2 cd %3 pwd send %4 New1.txt send %5 New2.txt rename New1.txt New1_Renamed.txt %6 New2.txt cd /download/guestusr	
•	•

%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 (

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** 
  - General Properties

<ul> <li>Detailed Pro</li> </ul>	Detailed Properties			
Configuration:	Default Module Test	•		
Module:	FTPPSOutboundModule	•		
Component:	FTPPSOutboundInterface	•		
Interface:	FTPPSOutboundInterface	•		
Operation:	executeFTPScript	•		

Initial request parameters

8 8 6			
Name	Туре	Value	
executeFTPScript1	Input FTPFile	×	
📃 🖳 DirectoryPath	string	<ul> <li>C:\Labfiles62\FTPOutbound</li> </ul>	
💭 Filename	string	<ul> <li>ScriptFile.txt</li> </ul>	

- \_\_\_\_ 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

🚯 Add Element		×
Enter the number of new elements to add:		
6		
	ОК	Cancel

- \_\_\_\_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

	string[]	60°
Important ScriptFileParameters[0]	string	🖌 pwd
— İ ScriptFileParameters[1]	string	✓ FTPScriptDir
Important ScriptFileParameters[2]	string	✓ FTPScriptDir
師 ScriptFileParameters[3]	string	<ul> <li>C:\Labfiles62\FTPOutbound\ScriptContent.txt</li> </ul>
[] ScriptFileParameters[4]	string	<ul> <li>C:\Labfiles62\FTPOutbound\ScriptContent.txt</li> </ul>
💭 ScriptFileParameters[5]	string	🗸 delete

#### \_\_\_\_e. Click **Continue** button under Events

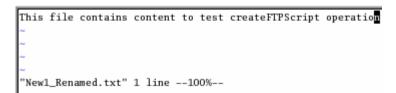
#### \_\_\_\_\_f. From Deployment Location window (if opens), select **WebSphere Process Servers > WebSphere Process Server v6.2** and click **Finish**

# \_\_\_\_ 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	Туре	Value
🖃 🏪 executeFTPScriptOutput	ExecuteFTPScriptResponse	✓
🗄 🗉 🔲 CommandTrace	string[]	60
🏳 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:

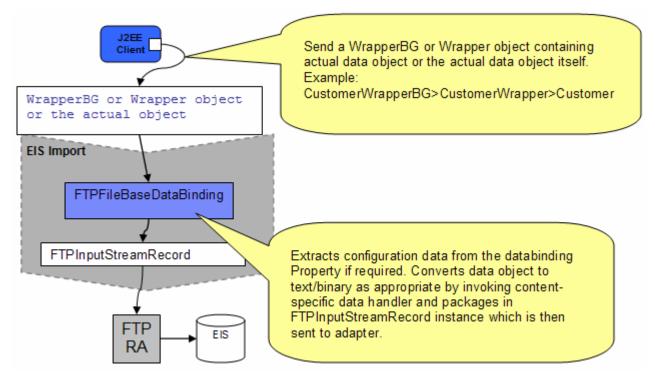


# 3.3. Restore Server Configuration

- 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.

 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:

🛱 Customer	
<u></u>	
customerName	string
e Address	string
e City	string
e State	string

- \_\_\_\_\_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)**

🚯 New FTP Service	×
FTP Service	/x =
Create a new FTP service.	- <b>1</b> 00
C. Couche a complex from a contract (horizon)	
Create a service from a pattern (typical) Create a service (advanced)	
Filter: type filter text	Ē.
Available Patterns	
Adapters	
Erete an outbound FTP service to write to a remote file	

\_\_\_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

٩	
	ttp IBM WebSphere Adapter for FTP (IBM : 6.2.0.0)
	🕀 🔍 CWYFT_FTPFile

- \_\_\_\_\_ 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
    - 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

Deploy connector projec	t: With module for use by single application 🔫 🔽
Connection properties:	Use properties below
Connection properties	
FTP system connec	tion information
Host name: *	wsbeta181.austin.ibm.com <del>&lt;</del>
Directory: *	/home/wsbeta/Output
Protocol:	FTP - file transfer protocol 🛛 🗧 🚽
Port number:	21 🗲 🗕
The user name	and password will not be encrypted and will be stored as plain text.
User name:	wsbeta <del> –</del>
Password:	*****

\_\_\_\_ 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**.



#### 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	m
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	
	<b>#</b>
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.

Data Handler Configuration	_ 0
elect Data Format Transformation	
elect a data format transformation entry from the list. If you want to use your own custom lata transformation then select the second radio button to add your custom transformation.	
🖲 Use existing data format transformation from the list	
	±
E Belimited	
Fixed width	
Handled by WTX	
I I I I I I I I I I I I I I I I I I I	
E E UTF8XMLDataHandler	
Select your custom data format transformation from the workspace	
Data transformation class name:	Select
Add gustom class to binding registry	
escription:	
On inbound, parses UTF-8 encoded XML data into a business object. On outbound, serializes b	usiness
object to UTF-8 encoded XML data.	

## \_\_\_ g. Click Finish

\_\_\_\_h. Back to 'Data Transformation Properties' screen, and the Configured data handler, UT8XMLDatahandler is displayed defined in the previous steps

🚯 Data Binding Con	figuration	
Data Transform	ation Properties	
Specify the propertie	s for the data transformation.	
Select DataBinding if	you want to use a data binding developed for earlier	versions of the adapter.
Binding type:	DataHandler	•
Configured data han	dler: UTF8XMLDataHandler 🦛	<u>S</u> elect
Configured data bind	ling; Not defined	Select

- \_\_\_ i. Click Next
- \_\_\_\_j. Ensure that the module selected is FTPCustomOutboundModule and enter **CustomDB** for the Name of the data binding

<u>M</u> odule:	FTPCustomOutboundModule	Browse,	New
Name <u>s</u> pace:	http://FTPCustomOutboundModule	🔽 Default	
F <u>o</u> lder:		Browse	
Name:	CustomDB		

#### \_\_\_ k. Click Finish

\_\_\_\_ I. Now the **CustomDB** should be displayed for **Data format** 

Service properties						
	Data format options	:	Use a data format configuration for all operatio	ns		
	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 operati	on to add.	
Operation kind:	Create	
Operation properties	,	
Data type for the operation:	User defined type	<b>←</b>
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 inp	ut					
Input type: 🔶 *		Browse	New			
Data format options:	Use data format configuration 'CustomDB'					
Data format:	Not defined	Select				
Specify the operation ou	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:



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.

Data type:	* Customer {http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customer} Browse New
Generate business gra	aph for each business object
Namespace for generated	business graph and container object.
Business object namespac	e: http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp
Generate retrieve con	tainer to retrieve multiple business objects.

\_\_\_\_ 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):

Operation name:	* (	createCustomFile		
Specify the operation input –				
Input type:	* [	$\label{eq:customerw} CustomerWrapperBG_{http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customerwrapperbg} \ref{eq:customerwrapperbg} \ref{eq:customerwrapperbg} \label{eq:customerwrapperbg}$	Bromsonn	New

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
- \_\_\_I. 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:

Operation name: *	createCustomFile	
Specify the operation inp	put	
Input type: *	CustomerWrapperBG {http://www.ibm.com/xmlns/pro	Browco New
Data format options:	Use data format configuration 'CustomDB' 🔫 🕂	
Data format:	Not defined	Select
Specify the operation ou	tput	
Output type;	CreateResponse {http://www.ibm.com/xmlns/prod/wel	Browse, New
Data format options:	: Use a data format configuration 🛛 🔫 🔫	
Data format: *	< CustomDB	Select

\_\_\_\_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

Operations:	
	wrapperbg}C Add
	Edit
	Remove
Operation properties:	
InteractionSpec properties for 'createCustomFile'	
FTP system connection information	
Remote directory on FTP system:	
Default target file name:	
File in local directory	
Local directory:	Browse
Archive file in the local directory for create operation	
Local archive directory for create operation:	Browse
Create new file if the file does not exist	
🦳 Generate a unique file	
$\square$ Delete the file after retrieve operation	
Remote archive directory for retrieve operation:	
[<< Advanced]	
Advanced connection configuration	
Second FTP system connection information	

#### 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

Operation kind:	Append	•	
Operation properties			
Data type for the operation:	User defined type	•	
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:

Operation name:	append		
Specify the operation inp	out		
Input type: 🔶 *		Browse	New
Data format options	Use data format configuration 'CustomDB'		
Data format:	Not defined	Select	
Specify the operation ou	tput		
Output type: 🔶	AppendResponse {http://www.ibm.com/xmlns/prod/we	Browse	New
Data format options	Use suggested data format 'FTPFileBaseDataBinding' 💌		
Data format:	Not defined	Select	

- \_\_\_\_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:

Operation name: *	appendCustomFile	
Specify the operation inp	ut	
Input type: *	CustomerWrapperBG {http://www.ibm.com/xmlns/proc	Browse New
Data format options:	Use data format configuration 'CustomDB' 🔫 —— 🔽	
Data format:	Not defined	Select
Specify the operation out	:put	
Output type:	AppendResponse {http://www.ibm.com/xmlns/prod/we	Browse New
Data format options:	Use a data format configuration	
Data format: *	CustomDB	Select

\_\_\_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

Operations:		
createCustomFile ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customerwrapperbg}	CustomerWrapperBG) :	Add
appendCustomFile ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customerwrapperbg)	{CustomerWrapperBG}	Edit
-		Remov
	Þ	
Operation properties:		
InteractionSpec properties for 'appendCustomFile'		
FTP system connection information		
Remote directory on FTP system:		
Default target file name:		
File in local directory		
Local directory:	Br	owse,,,
$\square$ Archive file in the local directory for create operation		
Local archive directory for create operation:	Br	owse,,,
Create new file if the file does not exist		
🔲 Generate a unique file		
Delete the file after retrieve operation		
Remote archive directory for retrieve operation:		
Advanced >>		
Advanced >>		

#### 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

Operation kind:	Retrieve
Operation properties	
Data type for the operation:	Generic FlatFile business object 💽
Enable response type for the	operation

\_\_\_ 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
  - 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/webspherenererererererererererererererererere
🔽 Generate business gr	aph for each business object
Namespace for generate	d business graph and container object.
Business object namespa	ce: http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp
🔽 Generate retrieve cor	ntainer 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:

Operation name: *	retrieveCustomFile		
Specify the operation inp	ut		
Input type:	FTPFile {http://www.ibm.com/xmlns/prod/websphere/j;	Browse	New
Data format options:	Use a data format configuration		
Data format: *	CustomDB <	Select	
Specify the operation out	put		
Output type: *	CustomerRetrieveWrapperBG {http://www.ibm.com/	Bronsonn	New
Data format options:	Use data format configuration 'CustomDB' 🛛 🗲 🛨		
Data format:	Not defined	Select	

\_\_\_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

Operations:	
<pre>     createCustomFile ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customerwrapperbg}CustomerWrapperBe     appendCustomFile ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customerwrapperbg}CustomerWrapperBe     retrieveCustomFile ({http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfile}FTPFile) : {http://www.ibm.com/ </pre>	3G)
	▶
Operation properties:	
InteractionSpec properties for 'retrieveCustomFile'	
FTP system connection information	
Remote directory on FTP system:	
Default target file name:	
File in local directory	
Local directory:	Browse
$\square$ Archive file in the local directory for create operation	
Local archive directory for create operation:	Browse,
Create new file if the file does not exist	
🗖 Generate a unique file	
Delete the file after retrieve operation	
Remote archive directory for retrieve operation:	
Advanced >>	

- \_\_\_\_a. Click **Next** from the Operations window
- \_\_\_\_\_16. From the Generate Artifacts screen, enter these:

## \_\_\_\_a. For Name, enter FTPCustomOutboundInterface

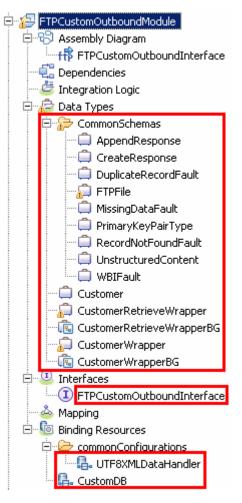
Properties for serv	vice	
Module;	FTPCustomOutboundModule	New
Namespace;	http://FTPCustomOutboundModule/FTPCustomOutboundInterface	
	Use default namespace	
Name: *	FTPCustomOutboundInterface	
	🔲 Save business objects to a library	
Library:	<b>_</b>	New
Description:		

- \_\_\_ b. Click Finish
- \_\_\_\_\_ 17. You will now see a new import component, **FTPCustomOutboundInterface** in the assembly diagram of FTPCustomOutboundModule

🕄 *FTPCustomOutboundModule - Assembly Diagram 🛛				
👌 😳 Palette				
🔓 🗨 🗨 🖫 🕼				
🕞 Favorites	ff     FTPCustomOutboundInterface			

\_\_\_\_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:

) FTPCustomOutboundI	interface 🛛		
•Operations	🕺 🏂 🕼 🖓 👘 🐺		
Operations and their pa	arameters		
	Name	Туре	
▼ <sup>™</sup> CreateCustomFil	e		
DI Input(s)	createCustomFileInput	CustomerWrapperBG	
(C) Output(s)	createCustomFileOutput	CreateResponse	
🔀 Fault	MISSING_DATA	MissingDataFault	
🔀 Fault	WBIFault	WBIFault	
🔀 Fault	DUPLICATE_RECORD	DuplicateRecordFault	
🔀 Fault	RECORD_NOT_FOUND	RecordNotFoundFault	
✓ ¥iji appendCustomFile			
DI Input(s)	appendCustomFileInput	CustomerWrapperBG	
📫 Output(s)	appendCustomFileOutput	AppendResponse	
🔀 Fault	MISSING_DATA	MissingDataFault	
🔀 Fault	WBIFault	WBIFault	
🔀 Fault	RECORD_NOT_FOUND	RecordNotFoundFault	
▼ <sup>™</sup> gretrieveCustomF	=ile		
DI Input(s)	retrieveCustomFileInput	FTPFile	
(C) 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

#### Detailed Properties

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

Initial reguest parameters

Name	Туре			
E ereateCustomFileInput	CustomerWrapperBG	✓		
	verb <string></string>	¥.		
E CustomerWrapper	CustomerWrapper			
DirectoryPath	string	✓ /home/wsbeta/Ou		
	string	<ul> <li>UserDefined.xml</li> </ul>		
	string			
FtpServerHostName	string	· ·		
	string	· ·		
	string	· ·		
	string	· ·		
	string	~		
	string	· ·		
	string	~		
	string	× ×		
	string	<pre>// #####</pre>		
	boolean	✓ false		
	string	<ul> <li>✓ raise</li> </ul>		
LocalArchivingEnabledForCreate	boolean	✓ false		
	string			
	string	· ·		
	boolean	✓ false		
	boolean	✓ false		
[I] ScriptFileParameters	string[]	60		
SplittingFunctionClassName	string	✓		
	string	· ·		
	boolean	✓ false		
	string			
	Customer	*		
	string	ABC		
	-	✓ ADC ✓ 11501 Burnet Rd		
	string string	<ul> <li>Austin</li> </ul>		
City	string	V TX		

\_\_\_\_e. Click **Continue** button under Events

\_\_\_\_\_f. From Deployment Location window, select WebSphere Process Servers > WebSphere Process Server v6.2 at localhost and click Finish

🚯 Deployment Location	
Select Deployment Location Specify a runtime location where this test will deploy.	E
Deployment location:	
🖃 🔤 WebSphere Process Servers	New Server
WebSphere Process v6.2 Server at localhost	
Eclipse 1.5 JVM	
Mode: Run	
$\square$ Use this as the default and do not ask again	

- \_\_\_\_g. Provide Administrator User ID and Password
  - 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

🚯 User Login - Default Module Test 🛛 🗙
Security is enabled on the selected runtime environment(s). Please sign in to continue the test.
User ID:
admin
Password:
•••••
$\checkmark$ Use the authentication settings in the preference and <u>n</u> ever ask again.
OK Cancel
v

\_\_\_\_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:

Events	General Properties		
↓ ■ ⇒ ○ ■ □	<ul> <li>Detailed Properties</li> </ul>		
Invoke (FTPCustomOutboundInterface:createCustomFile)     F Invoke started     Invoke (FTPCustomOutboundInterface:createCustomFile)     Return (FTPCustomOutboundInterface:createCustomFile)     Invoke returned	Module: <u>FTPCustomOutboundMo</u> Component: <u>FTPCustomOutboundInt</u> Interface: <u>FTPCustomOutboundInt</u> Operation: createCustomFile	erface	
	Return parameters:		
	Name	Туре	
	🖃 🖳 createCustomFileOutput	CreateResponseBG	✓
	verb	string	*
		CreateResponse	✓
	📃 🛄 Filename	string	<ul> <li>UserDefined.xml</li> </ul>

- 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.

<pre>%xml version="1.0" encoding="UTF-8"?&gt;</pre>	Martin and Anna Anna Anna Anna Anna Anna Anna	
cp:Customer xsi:type="p:Customer" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.ibm.com/xmlns/prod/websphere/	j2ca/ftp/custon	mer">
<customername>ABC</customername>		
<address>11501 Burnet Rd</address>		
<city>Austin</city>		
<state>78758</state>		
#####		
~		
		666
"UserDefined.xml" 8 lines12%	1,1	A11

#### 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 reguest parameters

ž 🔹 🏧				
Name	Туре			
🖃 🖳 appendCustomFileInput	CustomerWrapperBG	✓		
江 verb	verb <string></string>	8.		
🗄 🖳 Customer Wrapper	CustomerWrapper	✓		
DirectoryPath	string	✓ /home/wsbeta/OutDir		
💭 Filename	string	<ul> <li>UserDefined.xml</li> </ul>		
[ ChunkInfo	string	×		
FtpServerHostName	string	×		
	string	×		
	string	✓		
FileTransferType	string	✓		
- EcondServerDirectory	string	×		
🖳 💭 SecondServerUsername	string	×		
	string	×		
FileContentEncoding	string	×		
IncludeEndBODelimiter	string	✓ #####		
- IIII FileInLocalDirectory	boolean	🗸 false		
	string	×		
	boolean	🗸 false		
	string	×		
StagingDirectory	string	×		
🖳 🛄 GenerateUniqueFile	boolean	🗸 false		
CreateFileIfNotExists	boolean	🗸 false		
[I] ScriptFileParameters	string[]	66		
	string	×		
	string	×		
DeleteOnRetrieve	boolean	🗸 false		
- Interview ArchiveDirectoryForRetrieve	string	×		
🗄 📲 Content	Customer	✓		
	string	✓ IBM		
	string	🗸 11901 RTP		
(İla City	string	🗸 Raleigh		
💭 State	string	NC 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:

Events		General Properties				
♦ - 🔳 🔆 🔎 🔳 🔛						
	🖃 K Invoke (FTPCustomOutboundInterface:createCustomFile)	Mo	odule:	FTPCustomOutboundMod	tule	
Invoke started     Invoke (FTPCustomOutboundInterface:createCustomFile)     Invoke (FTPCustomOutboundInterface:createCustomFile)		Co	Component: FTPCustomOutboundInterface Interface Interface			
Invoke returned		Operation: appendCustomFile				
🖃 ừ Invoke (FTPCustomOutboundInterface:appendCustomFile)		Return parameters:				
Invoke started     Invoke (FTPCustomOutboundInterface:appendCustomFile)		_	i 84 🚾			
	🛃 Return (FTPCustomOutboundInterface:appendCustomFile)			Name	Туре	
	Invoke returned		🗆 💾 a	ppendCustomFileOutput	AppendResponse	×
			ĝ	🗆 Filename	string	✓ UserDefined.xml

- 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 (
  - 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	Туре	Value
🖃 🏪 retrieveCustomFileInput	FTPFile	✓
DirectoryPath	string	×
Filename	string	✓ UserDefined.xml
🎞 ChunkInfo	string	*
- EtpServerHostName	string	✓
- In FtpServerEventDirectory	string	✓
💭 DataConnectionMode	string	✓
🖳 💭 FileTransferType	string	✓
- EcondServerDirectory	string	✓
- EcondServerUsername	string	✓
- EcondServerPassword	string	✓
- IIII FileContentEncoding	string	✓
- IncludeEndBODelimiter	string	✓
FileInLocalDirectory	boolean	✓ false
- IocalDirectoryPath	string	C:\Labfiles62\FTPOutbound\LocalDir
- IocalArchivingEnabledForCreate	boolean	✓ false
- IocalArchiveDirForCreate	string	✓
- ImagingDirectory	string	✓
	boolean	✓ false
CreateFileIfNotExists	boolean	✓ false
[I] ScriptFileParameters	string[]	ഒറ
- Internation ClassName	string	com.ibm.j2ca.utils.filesplit.SplitByDelimiter
	string	✓ <b>####</b> #

#### \_\_\_\_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:

vents	General Properties		
≱ - ■ \$ 0 ■ .	<ul> <li>Detailed Properties</li> </ul>		
🖃 🐩 Invoke (FTPCustomOutboundInterface:createCustomFile)	Module: FTPCustomOutboundModule		
🖃 隆 Invoke started			
Invoke (FTPCustomOutboundInterface:createCustomFile)	Component: FTPCustomOutboundInterface		
Return (FTPCustomOutboundInterface:createCustomFile)	Interface: <u>FTPCustomOutboundInterface</u>		
Invoke returned	Operation: retrieveCustomFile		
🖃 🐩 Invoke (FTPCustomOutboundInterface:appendCustomFile)	Return parameters:		
🖃 隆 Invoke started			
Invoke (FTPCustomOutboundInterface:appendCustomFile)			
Return (FTPCustomOutboundInterface:appendCustomFile)	Name	Туре	
Invoke returned	🗆 🖳 retrieveCustomFileOutput	RetrieveResponseWrapper	✓
🖃 🐩 Invoke (FTPCustomOutboundInterface:retrieveCustomFile)	🖃 💷 Content	Object[unknown][]	66
🖃 隆 Invoke started	E	Customer	×
Invoke (FTPCustomOutboundInterface:retrieveCustomFile)	CustomerName	string	🗸 ABC
Return (FTPCustomOutboundInterface:retrieveCustomFile)	- In Address	string	🗸 11501 Burnet R
Invoke returned	City	string	🗸 Austin
	🛄 🛄 State	string	🖌 TX
	🖻 🖳 Content[1]	Customer	✓
	CustomerName	string	🗸 IBM
	- In Address	string	🗸 11901 RTP
	City	string	🗸 Raleigh
	💭 State	string	V NC
	SavedFileToLocalDirectory	boolean	8

## 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 New FTP Service	<u> </u>
FTP Service	4.2.
Create a new FTP service.	<b>100</b>
C Create a service from a pattern (typical) 💽 Create a service (advanced)	
Filter: type filter text	E.
Available Patterns	
Adapters  Adapters	

## \_\_\_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

Deploy connector project:	With module for use by single application 🔫 🔽			
Connection properties;	Use properties below			
Connection properties				
FTP system connection informa	ation			
Host name: * wsbeta181	Host name: * 🛛 wsbeta181.austin.ibm.com 🔫 💳			
Directory: * /home/wsb	eta/Output 🔫			
Protocol: FTP - file tr	ansfer protocol 🖌 🚽			
Port number: 21 🔫 🗕	•			
The user name and passw	The user name and password will not be encrypted and will be stored as plain text.			
User name: 🛛 wsbeta 🔫				
Password: *******	<del>~</del>			

\_\_\_\_ 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

Service properties  Services (JAAS) alias security credential.				
J2C authentication data entry;				
Data format options:	Use default data format 'FTPFileBaseDataBinding' for all operations			
Data format:	Not defined	Select		

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

Operation kind:	Create	•
Operation properties		
The data type for the operation input:	Generic FTP business object with business graph	•
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 inp	put		
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 ou	tput		
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

Oper	atio	ns:	
	÷	$create {\tt FTPFileBG} ({\tt http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/ftpfilebg} {\tt FTPFileBG}):$	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:

🕄 *ETPDefaultsOutbour	dModule - Assembly Diagram	R
	· · · · · · · · · · · · · · · · · · ·	
👌 🛟 Palette		
,}I€, ⊝, °L 🕼		
🕞 Favorites	I fr FTPImport	
🗁 Components		-

\_\_\_\_ 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

🔻 Detai	led	Properties
---------	-----	------------

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

Invoke export using binding

Initial request parameters

Ę				
	Name	Туре	Value	
	🖃 🖳 createFTPFileBGInput	FTPFileBG	✓ ▲	
		verb <string></string>	✓ CREATE	
		FTPFile	✓	
	🖳 🛄 DirectoryPath	string	✓	
	Filename	obring >	✓ DefaultsTest.txt	
	💭 ChunkInfo	string	✓	

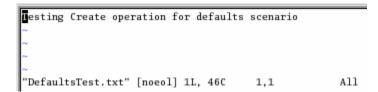
\_\_\_ b. For AsText under Content, enter some test message, for Ex: Testing Create operation for defaults scenario

ArchiveDirectoryForRetrieve	string	✓
	UnstructuredContent	✓
ContentType	string	✓
🚛 ObjectName	string	✓
- 💭 AsText	String 🍃	<ul> <li>Testing Create operation for defaults scenario</li> </ul>
🛄 AsBinary	hexBinary	✓ 0
		▼

- \_\_\_\_ 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	General Properties			
🎓 🕶 📑 🖗 🛛 🔳 🔛	▼ Detailed Properties			
<ul> <li>Invoke (FTPImport:createFTPFileBG)</li> <li>Invoke started</li> <li>Invoke (FTPImport:createFTPFileBG)</li> <li>Return (FTPImport:createFTPFileBG)</li> </ul>		ETPDefaultsOutboundModule		
	Component: <u>FTPImport</u> Interface: <u>FTPImport</u>			
Invoke returned	Operation: <u>createFTPFileBG</u> Return parameters:			
	Name	Туре	Value	
	🖃 🔚 createFTPFileBGOut	CreateResponseBG	✓	
	👘 💭 verb	verb <string></string>	*	
	🗄 🖳 🔁 CreateResponse	CreateResponse	×	
	🛄 🛄 Filename 🗕	string	🔶 🗸 DefaultsTest.txt	

- \_\_\_\_ 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:

ſ	📋 Customer	7
	e customerName	string
	e Address	string
	e City	string
	e State	string

- \_\_\_\_ 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)

🚯 New FTP Service 🛛 🔀
FTP Service
Create a new FTP service.
• Create a service from a pattern (typical) • Create a service (advanced)
<u>A</u> vailable Patterns:
E Lintegration
E S Adapters
Create an outbound FTP service to write to a remote file
Description:
The FTP outbound pattern creates a service that stores data in a file in a specific directory on an FTP server. If the required output format is not an XML format, you can specify a data handler that will transform the business object to the file content format.

\_\_\_a. Click Next

\_\_\_\_7. From the next FTP service name screen, for Name, accept the default value 'FTPImport' and click Next

Module:	FTPTypicalOutboundModule	Browse	New
Name <u>s</u> pace:	http://FTPTypicalOutboundModule/FTPImport	🔽 Default	
F <u>o</u> lder:		Browse	
N <u>a</u> me:	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)

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

🌆 New Outbound FTP Service			
Business object and FTP server credentials			
Specify the business object, FTP server host name and remote directory where the business object contents will be written.			
What business object do you want to write to the output file?			
Business object: Customer {http://www.ibm.com/xmlns/prod/websphere/j2ca/ftp/customer}			
Where do you want to create the output file?			
FTP server host name: wsbeta181.austin.ibm.com			
Remote directory: /homw/wsbeta/OutDir			
h. Click <b>Next</b>			
9. From 'FTP server security credential' screen, enter these:			
a. Select the radio button next to Using user name and password			
1) User name: username using which you connect to your FTP server (for Ex: wsbeta)			
2) Password: password for the above user to connect to your FTP server			
🚯 New Outbound FTP Service 🛛 🔀			
FTP server security credential			
Specify the FTP server security credential.			

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

## C Using an existing JAAS alias (recommended)

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

## Using user name and password

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

<u>U</u> ser name:	wsbeta	
Password:	*****	

\_\_\_ 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.

🚯 New Outbound FTP Service	×
Output file name	
Specify the method for naming the output file.	FTP
How should the output file be named?	
• <u>G</u> enerate a file name with an appended sequence number	
File name: Customer.txt 🔫	
Specify the sequence file that the adapter will create and use to store the sequer	nce number
Directory: C:\Labfiles62\FTPOutbound\LocalDir <	Browse
Sequence file name: Customer.seq	
C Use FTP server to generate a file name	
O Use a wrapper business object to specify the output file name	
Generated wrapper business object: CustomerWrapper	
$\square$ Add an append operation for writing multiple business objects to a single file	

\_\_\_b. Click Next

- \_\_\_\_\_ 11. From the Output file format screen, enter these:
  - \_\_\_\_a. Select the radio button next to XML

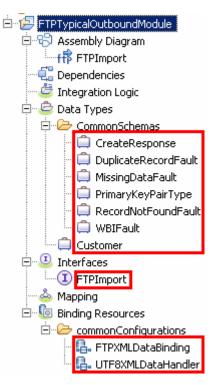
🚯 New Outbound FTP Service	×
Output file format Specify the format for the content of the output file.	FTP
What is the output file format?	
(This option does not support writing multiple business objects to a single file)	
O <u>O</u> ther	
Specify a data handler to transform business objects to the required format.	
Data handler:	Browse

## \_\_\_ b. Click Finish

\_\_\_\_\_ 12. Save (Ctrl + S) changes to your assembly diagram

😚 *FTPTypicalOutboundModule - Assembly Diagram		
👌 😳 Palette		
<b>}®,⊙,°⊾ <u>®</u>,</b>		•
🔁 Favorites	I frit FTPImport	2.

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

Ę	ž 🖞 🗖				
	Name	Туре	Value		
	🖃 🏪 createInput	Customer	¥		
	🖳 🛄 CustomerName	string	ABC		
	🖳 💭 Address	string	🗸 11501 Burnet Rd		
	🖳 🛄 City	string	🗸 Austin		
	🛄 State	string	V 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	General Properties		
🏞 ▾ 🗐 🔆 🍋 🕒 🔛	<ul> <li>Detailed Properties</li> </ul>		
<ul> <li>M Invoke (FTPImport:create)</li> <li>N Invoke started</li> <li>Invoke (FTPImport:create)</li> <li>Return (FTPImport:create)</li> <li>Invoke returned</li> </ul>	<u>Module: FTPTypicalOutbounc</u> Component: <u>FTPImport</u> Interface: <u>FTPImport</u> Operation: <u>create</u> Return parameters:	<u>IModule</u>	
	Name	Type CreateResponse	Value 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.

- \_\_\_\_ 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="1/2: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 - passthrough 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.

Server     State     Status       Image: State     State     Synchronized	🗞 Build Activities 🔲 Properties 🔝 Problems 🔞 Ser	ver Logs 👫 Servers 🔀 🚺 Asset	Repositories	🏇 🜔 🖉 🔳 🔑 🛅 🗖 🖗
	Server 🔺	State	Status	
New Course	📰 WebSphere Process Server v6.2 at localhost	🛼 Started	Synchronized	
Now https://www.				
The way is a server the server th	Ne <u>w</u>	· · · · · · · · · · · · · · · · · · ·	Server	
Open F3	Open	F3 T	VP	

- \_\_\_\_ 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

🚯 New Server		_	
Define a New Server Choose the type of server to	create		
Server's <u>h</u> ost name: mvsxxx Select the <u>s</u> erver type: type filter text	.rtp.raleigh.ibm.com	Download additional server ad	lapters
WebSphere Prod	lication Server v6.1 lication Server v7.0 Server v6.2 al v6.0 Server al v6.1 Server al v6.1 Server on WAS 7	2.	
Server na <u>m</u> e: Server <u>r</u> untime environment:	WebSphere Process Server v6	6.2 at mvsxxx.rtp.raleigh.ibm.com 6.2 <b>Configure runtime environm</b>	<u>Add</u> ents
0	< <u>B</u> ack	Next > Einish Can	cel

\_\_\_e. Click Next.

- \_\_\_\_\_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.

🚯 New Server				
WebSphere Server Settings Input settings for connecting to an existing WebSphere Application Server.				
WebSphere profile name;			Configure profiles	
Server connection types a	nd administrat	ive ports		
C Automatically determin	e connection s	ettings		
• Manually provide conn	ection settings			
Connection Type	Port	Default port	Description	
RMI SOAP	8880	2809 8880	Designed to improve communication with the server Designed to be more firewall compatible	
	0000	0000	Designed to be more mewait compatible	
Run server with resource	os within the w	orken nen		
Security is enabled on th		urkspace.		
Current active authenti		-,		
User ID:	ssadmin	,		
-				
Pa <u>s</u> sword:				
WebSphere server na <u>m</u> e:	server1			
Test Connection				
0		< <u>E</u>	lack <u>N</u> ext > <u>Finish</u> Cancel	

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

🗞 Build Activities 🔲 Properties 🔝 Problems 🔞 Server Logs 🦇 S	ervers 🛛 🔓 Asset Repositories	- 0
		🏇 🜔 🖉 🗉 💷 🛅
Server 🔺	State	Status
🔀 WebSphere Process Server v6.2 at localhost	🚡 Started	Synchronized
🛃 WebSphere Process Server v6.2 at mvsxxx.rtp.raleigh.ibm.com	🗎 Stopped	Synchronized

- 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>

\_\_\_\_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