FTP adapter – Processing COBOL copybook files lab

What this exercise is about 1
Lab requirements 1
What you should be able to do 2
Introduction 2
Exercise instructions 2
Part 1: Initialize the workspace and prepare for the lab 5
1.1. Review new properties 7
Part 2: Inbound scenario 9
2.1. Prepare for the inbound scenario 10
2.2. Configure inbound using the external service wizard 15
2.3. Add Java component 21
2.4. Test inbound scenario 23
Part 3: Outbound scenario 25
3.1. Prepare for the outbound scenario 26
3.2. Configure outbound using external service wizard 28
3.3. Test outbound scenario 36
Solution instructions 43
What you did in this exercise 45
Task: Adding remote server to WebSphere Integration Developer test environment 46

What this exercise is about

The objective of this lab is to provide you with an understanding of how to use the 'external data' wizard in WebSphere Integration Developer to generate business object definitions from a COBOL program source file. Then, the lab guides you through the configuration of inbound and outbound processing using these business object definitions to process COBOL copybook files.

Lab requirements

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

- WebSphere Integration Developer V7.0 installed and updated with latest fixes
- WebSphere Process Server V7.0 Test Environment installed and updated latest fixes

- Extract Labfiles70.zip to your C:\ (your root) drive

What you should be able to do

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

- Import FTP adapter RAR file into WebSphere Integration Developer
- Use External Data wizard to generate business object and wrapper definitions from a COBOL program source file
- Use External Service wizard to configure Activation Spec Properties, Resource Adapter Properties to generate Business Objects and other artifacts
- Deploy the adapter application onto WebSphere Process Server
- Test the deployed application using WebSphere Process Server test environment
- Restore the server configuration

Introduction

Use the external data wizard to generate a business object definition for a COBOL program source file. After you have generated the business object definition, you can optionally rerun the external data wizard to generate a wrapper business object definition from the generated business object.

You will use external service wizard to configure FTP adapter to convert COBOL copybook files to business objects during the inbound processing and convert business objects to COBOL copybook files during outbound processing.

Exercise instructions

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

Reference variable	Windows location	AIX® or UNIX® location
<WID_HOME>	C:\Program Files\IBM\WID70	
<WPS_HOME>	C:\<WID_HOME>\runtimes/bi_v70	
<FTPADAPTER_HOME>	<WID_HOME>\ResourceAdapters\FTP_7.0.0.0\deploy	
<LAB_FILES>	C:\Labfiles70	/tmp/Labfiles70
<WORKSPACE>	<LAB_FILES>\FTPCobol\workspace	
<EVENT_DIR>	<LAB_FILES>\FTPCobol\eventdir	
<ARCHIVE_DIR>	<LAB_FILES>\FTPCobol\archivedir	
<FTPFILES>	<LAB_FILES>\FTPFiles	
<TEMP>	C:\temp	/tmp

Windows users' note: When directory locations are passed as parameters to a Java™ program such as EJBdeploy or wsadmin, it is necessary to replace the backslashes with forward slashes to follow the Java convention. For example, replace C:\Labfiles70\ with C:/Labfiles70/

Instructions if using a remote server for testing

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

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

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

Part 1: Initialize the workspace and prepare for the lab

This part of the lab, you will extract the required lab files and start the WebSphere Integration Developer V6.2 with a new workspace and then define required WebSphere Environment Variables using the administrative console of WebSphere Process Server V7.0

____ 1. Extract the provided Labfiles70.zip to your C:\ (root) drive, if you have not already done so. This will create the necessary subdirectory structure to complete the lab, and provides you with sample text files

____ 2. Create directory structure on your FTP Server

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

__ b. Create these directories under the user's home directory:

1) mkdir **EventDir**

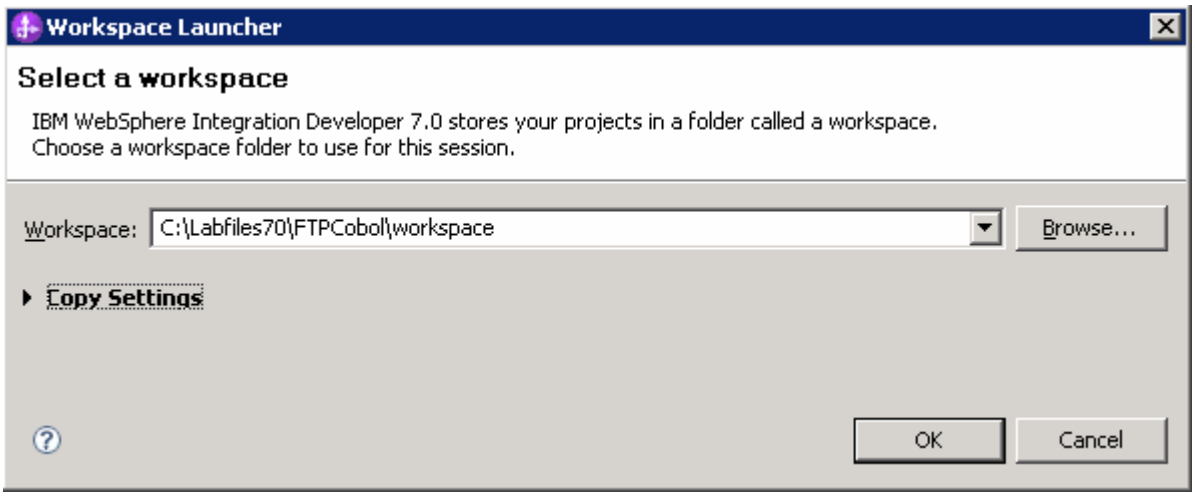
2) mkdir **ArchiveDir**

3) mkdir **OutDir**

____ 3. Start the WebSphere Integration Developer V7.0 with a new workspace

__ a. Select **Start > All Programs > IBM WebSphere Integration Developer > IBM WebSphere Integration Developer V7.0 > WebSphere Integration Developer V7.0**

__ b. From the Workspace Launcher window, enter **<WORKSPACE>** for the Workspace field



____ 4. Click the  button on the right corner to close the Welcome page and proceed with the workbench

____ 5. Follow the instructions of "**Define WebSphere variable"** lab to define the WebSphere variables listed in the table below, which are used later in this lab:

Note: In FlatFile adapter, you have to give directory values (Event directory, Archive directory, Output directory and so on, needed values in activation spec and managed connection factory) while running external service wizard. Presently these values can be changed in the module and re-deploy it (or) can be changed in deployed application from administration console. This is like hard coding the values of these properties. All local directory (including event and archive) properties in ActivationSpec can be filled with WebSphere environment variables. This way hard coding of directory paths for these properties can be avoided. From WebSphere Process Server V6.1, you can declare these values as environment variables and specify the environment variable name in the external service wizard. So when you deploy this application, the environment variable name is replaced with actual value and used by the adapter. This is very helpful if the values of these properties need to be changed.

Name	Value
FTP_EVENT	/root/EventDir
FTP_ARCHIVE	/root/ArchiveDir
FTP_LOCAL_EVENT	C:\Labfiles70\FTPCobol\LocalEventDir
FTP_LOCAL_ARCHIVE	C:\Labfiles70\FTPCobol\LocalArchiveDir
FTP_OUT	/root/OutDir

1.1. Review new properties

This part of the lab will give you brief description of some the new properties that are not used in this lab.

- **New in V7.0: FTPS Connection mode:** Specify the connection mode (Implicit or Explicit) to define the data protection level with which the data is transferred between the client and the server can be configured.
  - **Explicit** – In an explicit mode, the connection begins with an unencrypted FTP connection. When any sensitive information such as password needs to be sent, the client explicitly issues a request to switch to a secure FTP connection. After the successful SSL negotiation, a secure command channel is established between the client and the server. Explicit mode works with the default port 21 and is compliant with RFC 2228 commands. RFC 2228 specifies the mechanism for authenticating connections and confidential data transfer between the client and the server and this is referred as explicit mode. The AUTH command is used for specifying the security mechanism for the explicit mode. The client sends AUTH command (AUTH SSL/TLS) to the FTPS server and switches to a secure command connection.
  - **Implicit** – In an implicit mode, the communication between the client and server is setup immediately in secure mode. The text information exchanged between the client and server is in an encrypted form. The default port for implicit mode is 990.

- **New in V7.0: Data channel protection level**:
  - **Private** – Data is transferred in encrypted form: Select Private, if the data transfer between the Adapter and the FTPS server should be in an encrypted form
  - **Clear** – Data is transferred as clear text: Select Clear, if the data transfer between the Adapter and the FTPS server should be in clear text form

  Note: The default value is set to **Private**

Note: Server authentication can be enabled while establishing a SSL connection. While using SSL, FTPS server sends its certificate to the FTP client for verification. The FTP client verifies the certificate to ascertain that it is communicating with the intended FTP server. To enable this verification process, the FTP server's certificate should be present in the client's trust store. Following three properties are used for this server authentication.

- **New in V7.0: Keystore type**: Specify type of the keystore
  - JKS
  - PKCS12

Note: Use keytool utility, if you want to import server's certificate into client's trust store. For example, enter the command keytool -import -v -alias serverCert -file server.cert -keystore clientTrustStore where server.cert is the certificate of the server and clientTrustStore is the trust store of the client. Ensure that the value of Keystore type property is same as the type used while creating the keystore.

- **New in V7.0: Truststore file***: Specify the path of the truststore file that contains the certificates of the servers trusted by the adapter

- **New in V7.0: Truststore password:** Specify the password of the truststore. It is used to check the integrity of the truststore data. If the value is not specified, the integrity check is not performed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS

Note: Client authentication can be enabled while establishing a SSL connection. When using SSL/TLS, FTPS server requests for the client's certificate. The FTPS server verifies the certificate sent by the client to ascertain that it is communicating with the intended client. To enable this verification process, the FTPS server has to support client authentication and the client's certificate should be present at the server's trust store. At the client's end, client's keystore information has to be available for the exchange of the certificate to take place. Following three properties are used for this client authentication along with the Keystore type.

- **New in V7.0: Keystore file:** Specify the path of the keystore file. The keystore file will contain the private key entry of the FTPS client. It is also accompanied by a certificate chain for the corresponding public key

- **New in V7.0: Keystore password**: Specifiy the password of the keystore. It is used to check the integrity of the keystore data. If the value is not specified, integrity check is not executed. It is applicable only if the protocol value is set to FTP over SSL or FTP over TLS

- **New in V7.0: Key password:** Set the Key password to the password provided while creating the key in the keystore. This value is required to extract the certificate from the keystore while establishing a SSL connection.

Note: All these properties are applicable only if the protocol value is set to FTP over SSL or FTP over TLS

- **New In V7.0**: **Enable remote verification**: It checks whether the control and data connections are established with the same host (typically the machine from which you establish a connection to the FTP server). If not, the connection fails. By default, **Enable remote verification** check box is selected. This property is applicable to only FTP and FTPS protocols.

Part 2: Inbound scenario

In this part, you will import the FTP adapter RAR into the new workspace and use the external data wizard to generate a business object definition for a COBOL program source file. After you have generated the business object definition, you can optionally rerun the external data wizard to generate a wrapper business object definition from the generated business object.

You will make use of the new 'COBOL, C, or PL/I data binding' introduced in V6.0, while running the External service wizard to generate the required artifacts to convert COBOL copybook files to business objects during inbound processing.

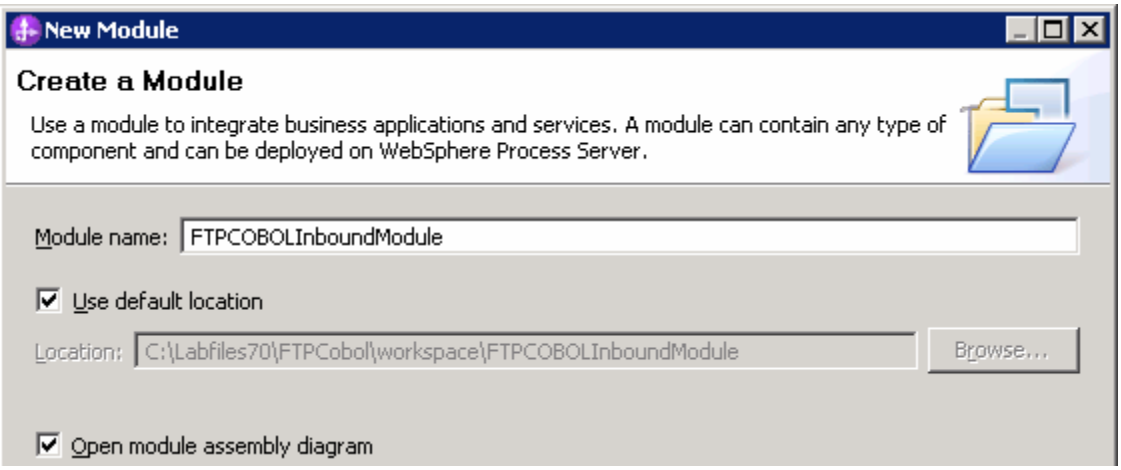
2.1. Prepare for the inbound scenario

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 FTPCOBOLInboundModule

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

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



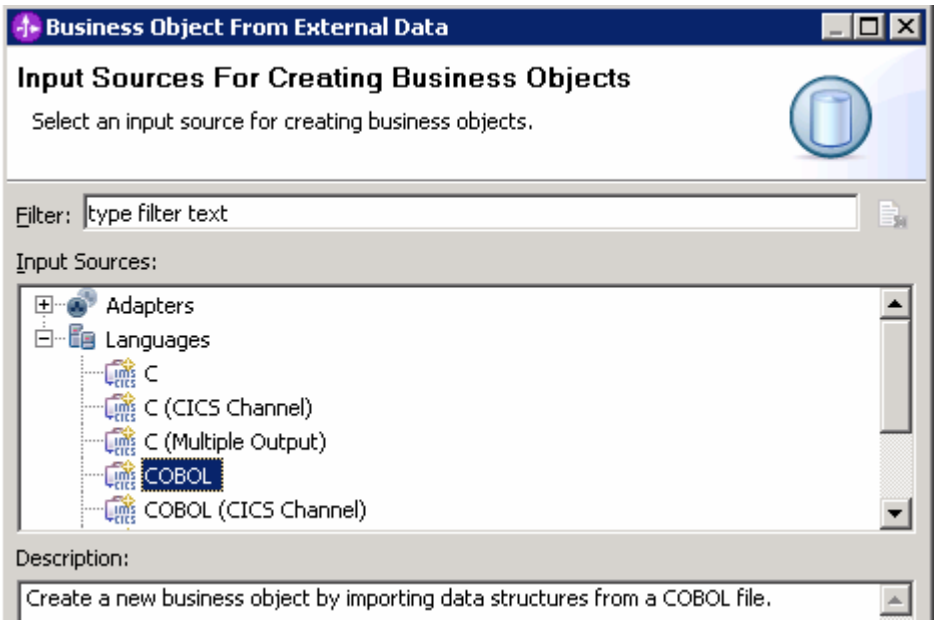
__ c. Ensure that the box next to **Open module assembly diagram** is checked and then click **Finish**

__ d. You will now see a new module, FTPCOBOLInboundModule, created from your Business Integration window

____ 2. Generate the business object definition for the COBOL program source file

__ a. In the Business Integration section of the window, right-click the module, FTPCOBOLInboundModule, and select **New > Business Object from External Data**

__ b. From the Business Object window, select **Languages > COBOL**

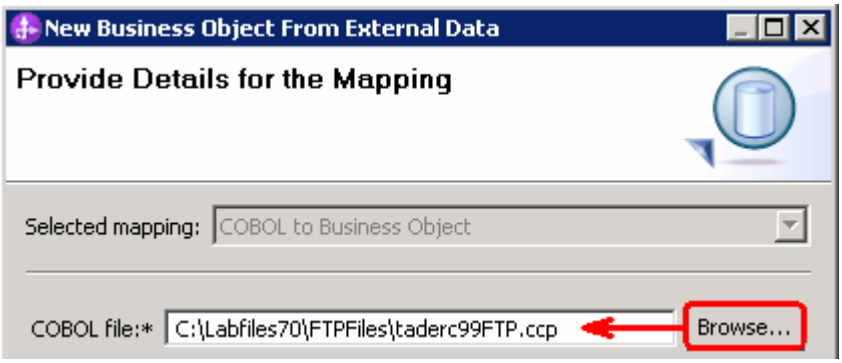


__ c. Click **Next**

__ d. In the Business Object Mapping Details window,

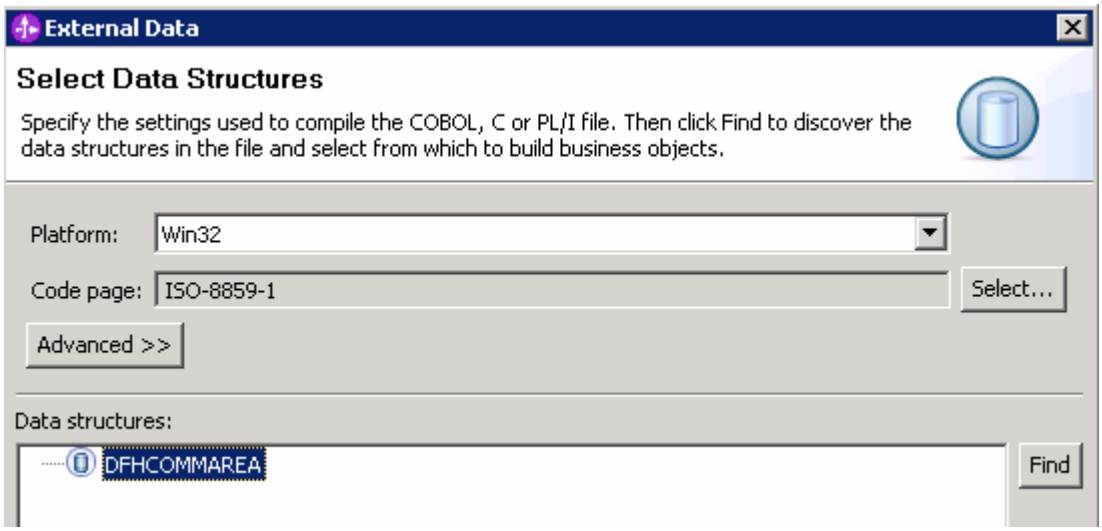
1) Notice that the **Select mapping** value is **COBOL to Business Object**

2) Click **Browse…** and select the **traderc99FTP.ccp** file located in **<FTPFILES>**



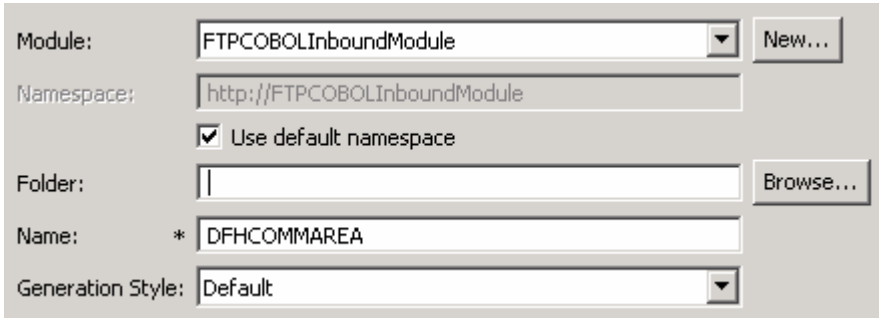
3) Click **Next**

__ e. In the Select Data Structures window, the new business object, called **DFHCOMMAREA**, is displayed

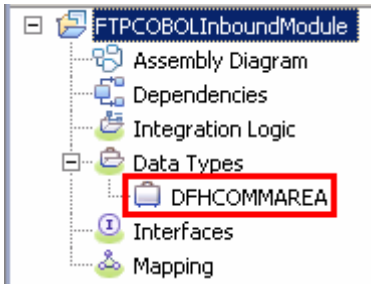


__ f. Select **DFHCOMMAREA** and click **Next**

__ g. In the Generate Business Objects screen, ensure that the selected Module is FTPCOBOLInboundModule and click **Finish**



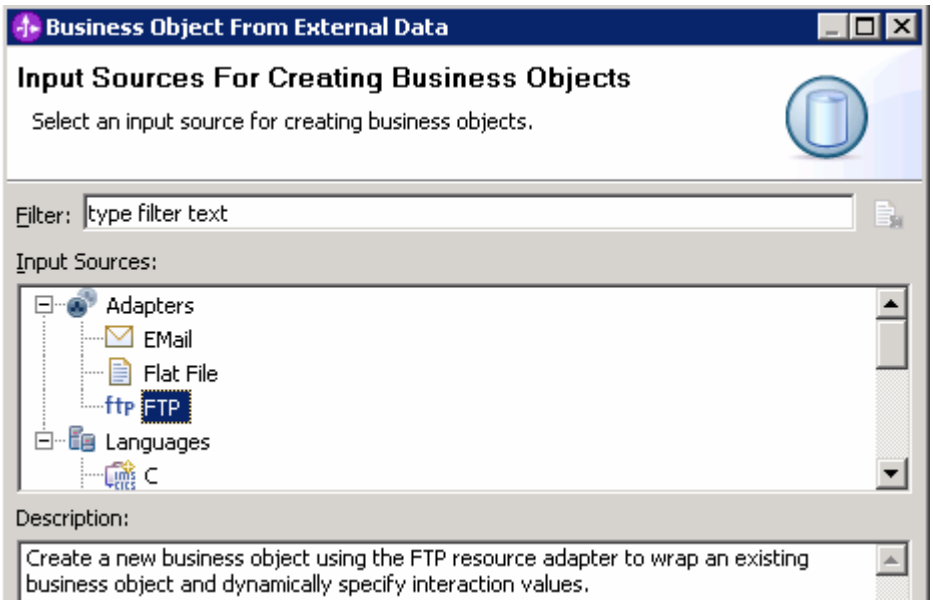
A business object, called DFHCOMMAREA, is created in the module



____ 3. Generate a wrapper business object definition. Wrapper business object definitions wrap existing business object definitions with additional function.

__ a. In the Business Integration section of the window, right-click the module and select **New > Business Object from External Data**

__ b. From the External Data window, select **Adapters > FTP** under Available Types:



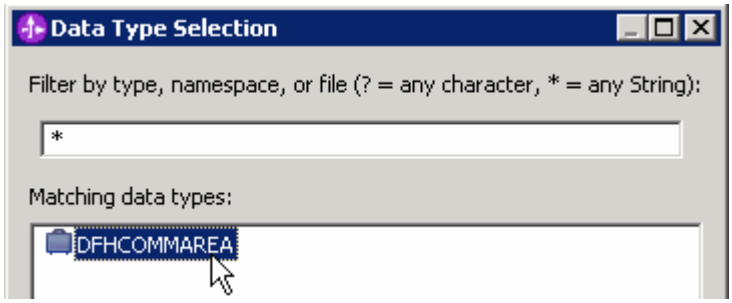
__ c. From the next screen, ensure that the correct module, **FTPCOBOLInboundModule**, is selected and click **Next**

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

__ d. Provide the below values in Business Object Properties screen:

1) Data type: click **Browse…**. A Data Type selection window is opened

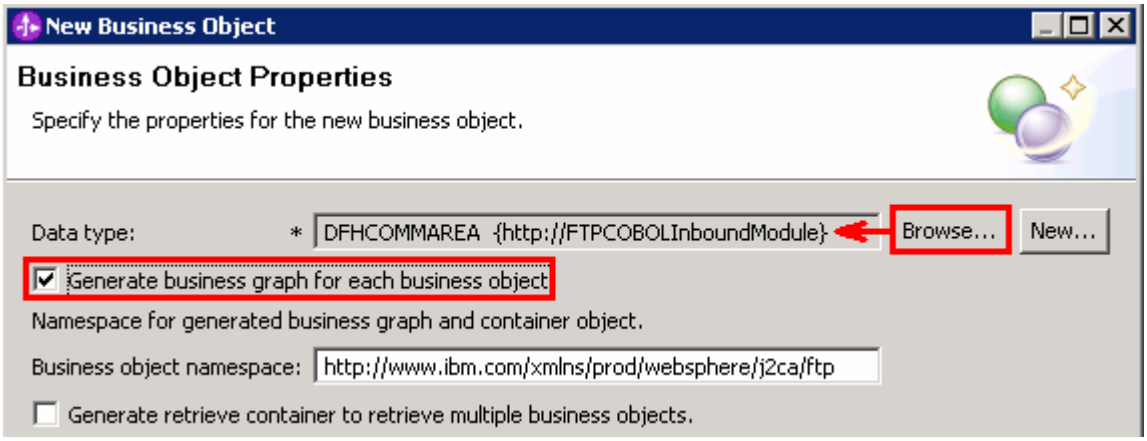
2) Select **DFHCOMMAREA** under Matching data types



3) Click **OK**

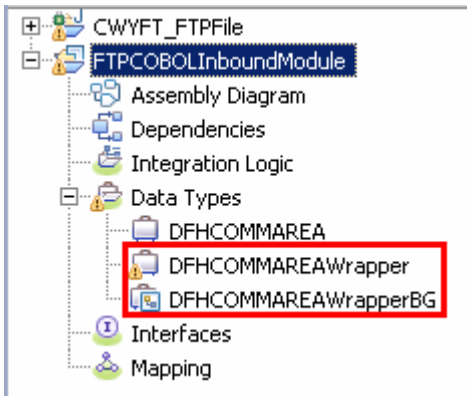
__ e. To generate a business graph, select the **Generate business graph for each business object** check box

__ f. **Do not** select the **Generate retrieve container to retrieve multiple business objects** check box. This new options is used and explained in outbound scenario



__ g. Click **Finish**

A wrapper business object and a business graph, called **DFHCOMMAREAWrapper** and **DFHCOMMAREAWrapperBG**, as shown in the figure below, are listed for the current module in the Business Integration window



2.2. Configure inbound using the external service wizard

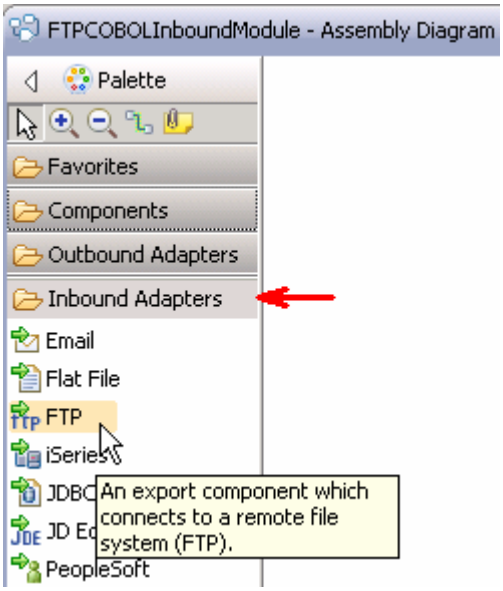
In this part of the lab you will use the new external service wizard and use the new 'COBOL, C, or PL/I data binding' to create the required artifacts to test the inbound scenario.

____ 1. Ensure that the FTPCOBOLInboundModule assembly diagram is opened in Assembly editor

__ a. In the Business Integration view, expand FTPCOBOLInboundModule and double-click Assembly Diagram

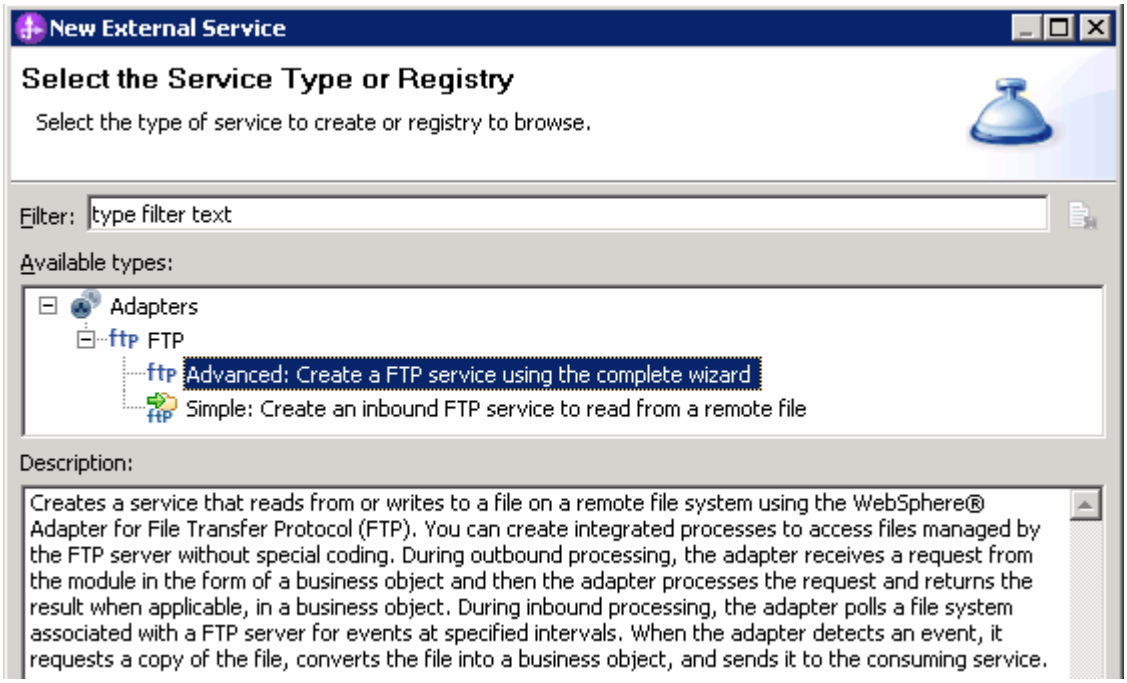
____ 2. To start External Service from the Palette:

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



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

____ 3. From the New External Service window, expand **Adapters > FTP** and select **Advanced: Create a FTP service using the complete wizard**



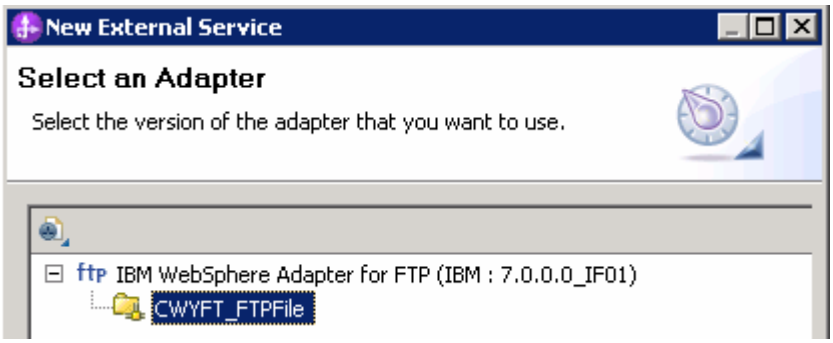
__ **a.** Click **Next**

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**

____ 4. On the Select an Adapter screen, select **IBM WebSphere Adapter for FTP (IBM : 7.0.0.0_IF01) > CWYFT_FTPFile** and 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: wsbeta149.austin.ibm.com

Note: Alternatively you can also define a WebSphere variable for the host name and replace it here.

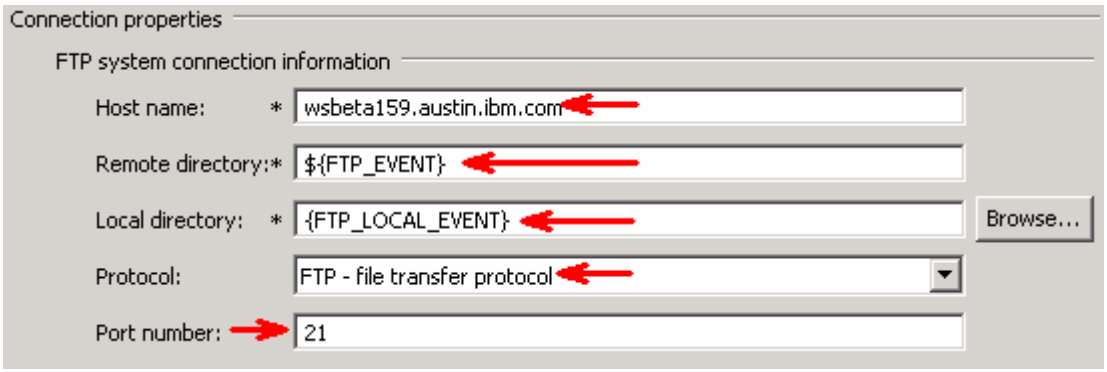
2) Remote directory: **${FTP_EVENT}**

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

3) Local directory: **${FTP_LOCAL_EVENT}**

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

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



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

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

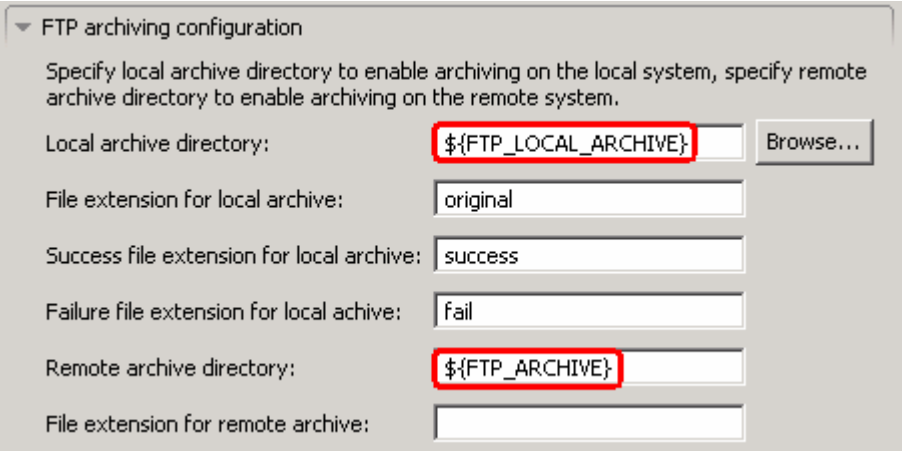
__ d. Event persistence configuration: In this part of the lab, you will not use any JNDI instead use adapter's in-memory representation of event table to store all the necessary information

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

__ e. FTP archiving configuration:

1) Local Archive directory: click **Browse…** and select **${FTP_LOCAL_ARCHIVE}**

2) Remote archive directory: **${FTP_ARCHIVE}**



____ 6. **Secure configuration**: Refer to '**Install and configure SSH server lab**' and '**FTP Adapter – SFTP support lab**' for more details on this new feature

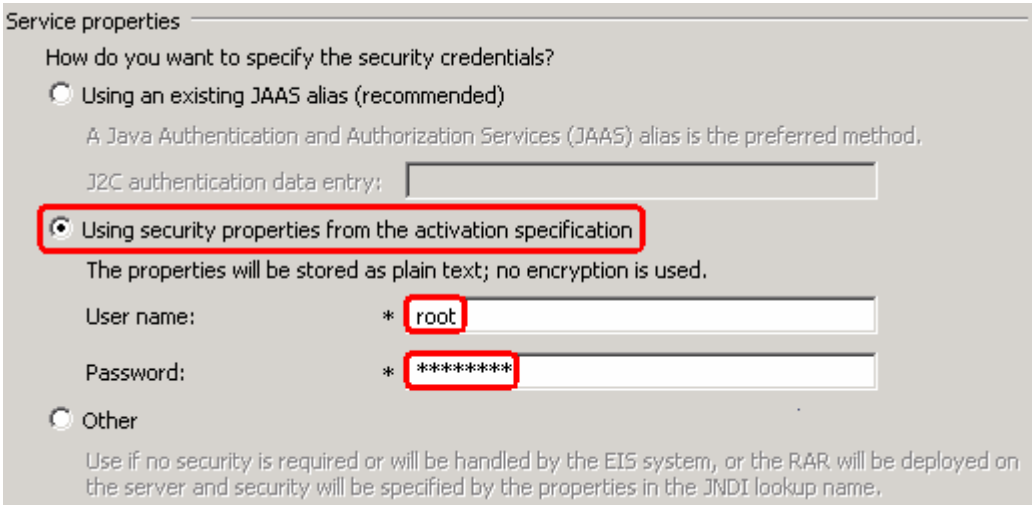
____ 7. **Logging and tracing**: Refer to the new lab '**Log and confidential trace lab**' for more details on this new feature

____ 8. For this lab, you are going to use the security properties from the activation specification.

__ a. Select the check box next to **Using security properties from the activation specification**

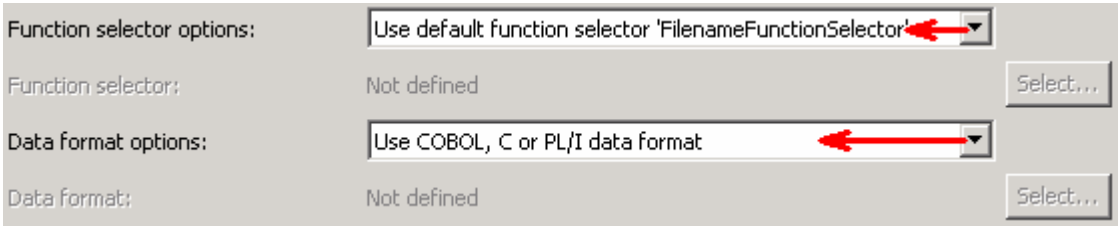
__ b. **User name**: username using which you connect to your FTP server (for Ex: **root**)

__ c. **Password**: password for the user to connect to your FTP server



____ 9. Under Service properties, for **Function selector options**, select **Use default function selector 'FilenameFunctionSelector'** from the drop down list

____ 10. For **Data format options**, select **Use COBOL, C or PL/I data format** from the drop down list

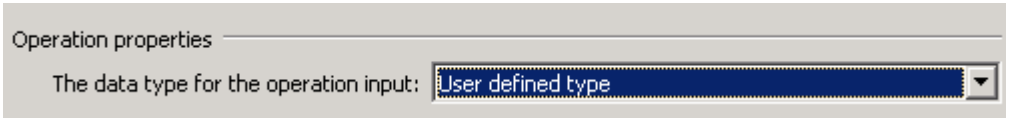


____ 11. Check the box next to **Change logging properties for wizard** to view the output location of the log file and the logging level and click **Next**

Define emitFTPCOBOL operation:

____ 12. From the Operations screen, click **Add…**

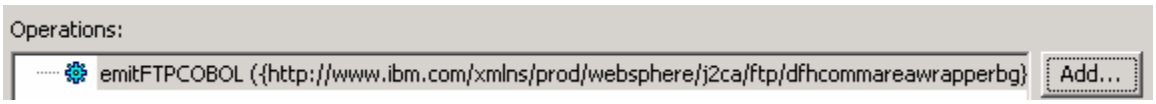
__ a. Add Operation window is opened. Select **User Defined Type** for the Data type and click **Next**



____ 13. For **Operation name**, enter any name, for Ex: **emitFTPCOBOL**

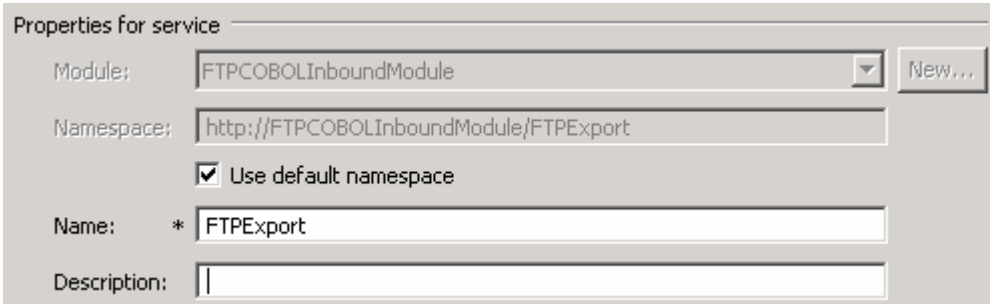
____ 14. For **Input type**, click **Browse** and select **DFHCOMMAREAWrapperBG** from the Data Type Selection window

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



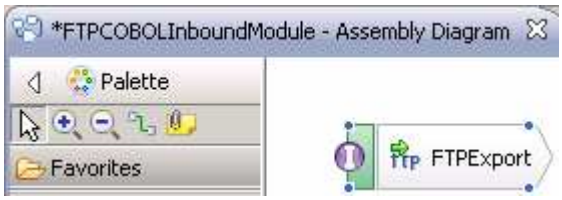
__ b. Click **Next** from Operations screen

____ 15. From Generate Service screen, accept the default value, **FTPExport**, for **Name**



__ a. Click **Finish**

____ 16. The Assembly diagram for FTPCOBOLInboundModule is opened with an Export component, **FTPExport**:



____ 17. Save (**Ctrl + S**) changes to your assembly diagram

2.3. Add Java component

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

____ 1. Open the assembly diagram for FTPCOBOLInboundModule (if it is already not open)

__ **a.** From the business integration view, expand **FTPCOBOLInboundModule** and double click **Assembly diagram**

____ 2. Drop a Java component to onto the assembly diagram

__ a. From the **Palette**, click **Components** to expand it

__ b. Click **Java** and then click the empty space of FTPCOBOLInboundModule assembly diagram. This will place a new component, **Component1** on the assembly diagram.

____ 3. Wire the FTPExport to the Component1

__ a. Select the **wire** ( ) icon from the Palette

__ b. Click **FTPExport** and then click **Component1** to wire them together

__ c. Select **OK** for the Add Wire pop-up window:

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

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

Your assembly diagram for FTPCOBOLInboundModule will look like this:

__ f. Right-click **Component1** and select **Generate Implementation** from the pop-up menu

__ g. On the **Generate Implementation** panel, select **default package**, and click **OK**

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

Note: The Java code is also available at **<FTPFILES>\ COBOLJavaCode.txt**

```
System.out.println("*************ENDPOINT
emitCOBOLFile**************");
DataObject wrapper =
emitFTPCOBOLInput.getDataObject("DFHCOMMAREAWrapper");
String filename = wrapper.getString("Filename");
System.out.println("File Name : "+filename);
DataObject customer = wrapper.getDataObject("Content");
String customernumber = customer.getString("CustomerNumber");
System.out.println("NAME-----> "+customernumber);
String firstname = customer.getString("FirstName");
```

```
System.out.println("First Name--> "+firstname);
String lastname = customer.getString("LastName");
System.out.println("Last Name--> "+lastname);
String street = customer.getString("Street");
System.out.println("Street--> "+street);
String city = customer.getString("City");
System.out.println("City-----> "+city);
String country = customer.getString("Country");
System.out.println("Country----> "+country);
String phone = customer.getString("Phone");
System.out.println("Phone-----> "+phone);
String postalcode = customer.getString("PostalCode");
System.out.println("Postal Code----> "+postalcode);
```

__ i. Save (**Ctrl + S**) and close Component1Impl.java

__ j. Save (**Ctrl + S**) and close Assembly diagram: FTPCOBOLInboundModule

2.4. Test inbound scenario

In this part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application inbound processing for the COBOL copybook scenario.

____ 1. Add the project to the WebSphere Test Environment server

__ a. Right-click **WebSphere Process Server v7.0** under the Servers view and select **Add and remove projects…** from the pop-up menu

__ b. From the Add and Remove Projects window, select **FTPCOBOLInboundModuleApp** under Available projects panel and click **Add >**

__ c. You will now see the **FTPCOBOLInboundModuleApp** added to the **Configured projects**

__ d. Click **Finish** and wait until the project is being published onto the server. The server is started in Debug mode if it is not already started before

____ 2. Test the application by providing input files:

Note: For your convenience, **COBOLTest.txt** is placed in **<FTPFILES>.**

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

__ b. The file will pass through the only existing method, **emitFTPCOBOL**, and you should see this message in your **Server Logs** view (or SystemOut.log):

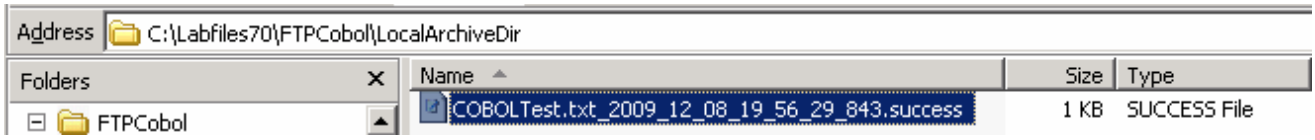
Type	Time	Thread ID	Contents
Log message	Dec 17, 2008 14:05:15.484	000006fe	**********ENDPOINT emitCOBOLFile**********
Log message	Dec 17, 2008 14:05:15.484	000006fe	File Name : COBOLTest.txt
Log message	Dec 17, 2008 14:05:15.484	000006fe	NAME-----> FTP11
Log message	Dec 17, 2008 14:05:15.484	000006fe	First Name--> Joe
Log message	Dec 17, 2008 14:05:15.484	000006fe	Last Name--> Sean
Log message	Dec 17, 2008 14:05:15.484	000006fe	Street--> Burnet Rd
Log message	Dec 17, 2008 14:05:15.484	000006fe	City-----> Austin
Log message	Dec 17, 2008 14:05:15.484	000006fe	Country----> USA
Log message	Dec 17, 2008 14:05:15.484	000006fe	Phone-----> 5128380000
Log message	Dec 17, 2008 14:05:15.484	000006fe	Postal Code----> 78758

Verify the results:

__ c. Check the **<LOCAL_EVENT_DIR>** on your local machine. The file is quickly moved from this directory to the local archive directory

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

__ e. Check the **<LOCAL_ARCHIVE_DIR>** subdirectory which should contain an archive of the event file, with the same file name appended with year, month, date, system time, and success



____ 3. Restore the Sever Configuration

__ a. Right-click **WebSphere Process Server v7.0** under the Servers view and select **Add and remove projects…** from the pop-up menu

__ b. Select **FTPCOBOLInboundModuleApp** 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 3: Outbound scenario

In this part, you will use the external data wizard to generate a business object definition for a COBOL program source file. After you have generated the business object definition, you can optionally rerun the external data wizard to generate a wrapper business object definition from the generated business object.

You will make use of the new ‘COBOL, C, or PL/I data binding’ introduced in V6.2, while running the External service wizard to generate the required artifacts to convert business objects to COBOL copybook files during outbound processing.

3.1. Prepare for the outbound scenario

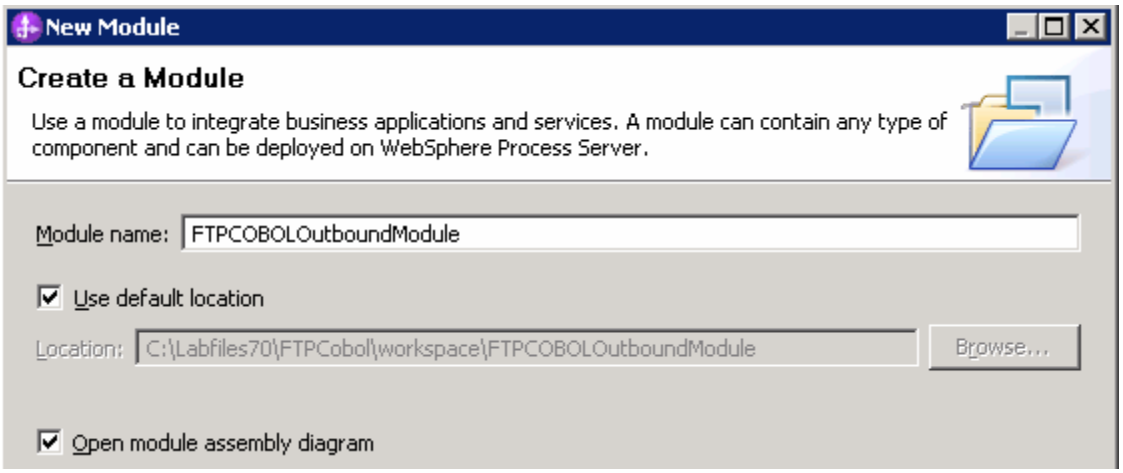
In this part of the lab you will create a module, use the external data wizard to generate a business object definition for a COBOL program source file. After you have generated the business object definition, you can optionally rerun the external data wizard to generate a wrapper business object definition from the generated business object.

____ 1. Create FTPCOBOLOutboundModule

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

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

__ c. Ensure that the box next to **Open module assembly diagram** is checked



__ d. Click **Finish**

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

____ 2. Follow the instructions in **Steps 2 and 3 of Part 2.1** to generate business object and a wrapper definition under **FTPCOBOLOutboundModule**

____ 3. Generate a wrapper business object definition for retrieve operation.

__ a. In the Business Integration section of the window, right-click the module and select **New > Business Object From External Data**

__ b. From the External Data window, select **Adapters > FTP** under Available Types:

__ c. From the next screen, ensure that the correct module, **FTPCOBOLOutboundModule**, is selected and click **Next**

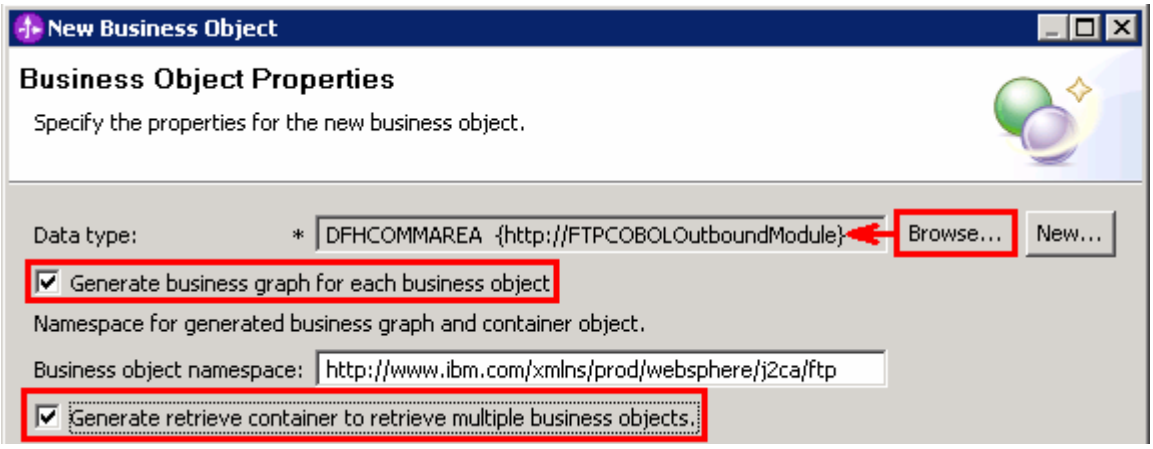
__ d. Provide the below values in Business Object Properties screen:

1) Data type: click **Browse…**. A Data Type selection window is opened

2) Select **DFHCOMMAREA** under Matching data types and click **OK**

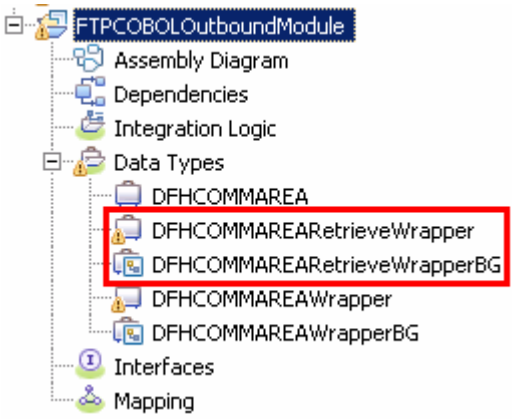
__ e. To generate a business graph, select the **Generate business graph for each business object** check box

__ f. Select the **Generate retrieve container to retrieve multiple business objects** check box



__ g. Click **Finish**

__ h. A wrapper business object and a business graph, called **DFHCOMMAREARetrieveWrapper** and **DFHCOMMAREARetrieveWrapperBG**, as shown in the figure below, are listed for the current module in the Business Integration window. This **DFHCOMMAREARetrieveWrapperBG** is used during External Service wizard in the next part for retrieve operation.



3.2. Configure outbound using external service wizard

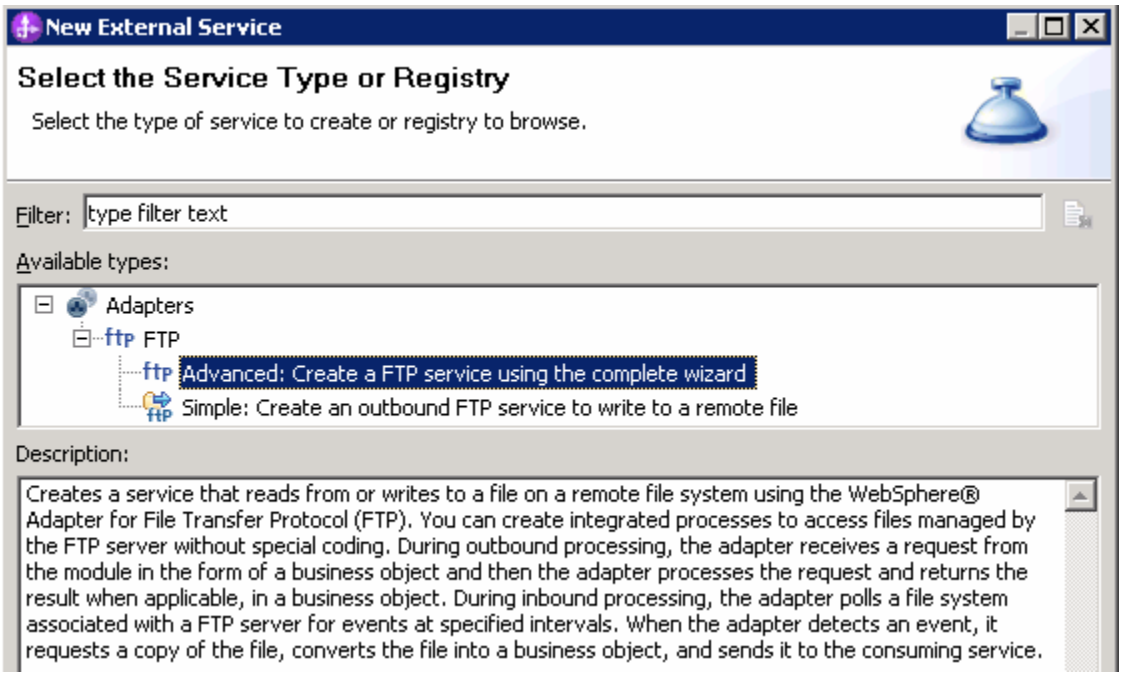
In this part of the lab you will use the default function selector and data binding options from the External Service wizard and generate other required artifacts to test the outbound scenario.

____ 1. 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

____ 2. From the New External Service window, expand **Adapters > FTP** and select **Advanced: Create a FTP service using the complete wizard**



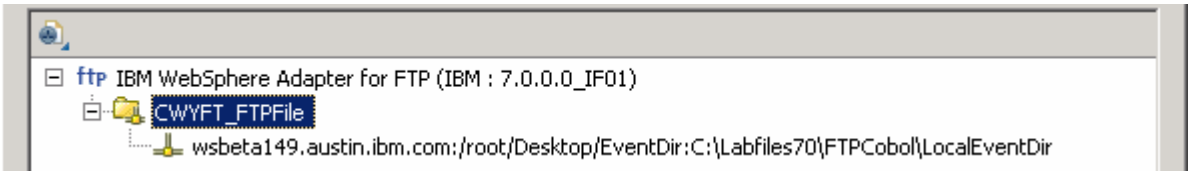
__ **a.** Click **Next**

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**

____ 3. On the Select an Adapter screen, expand **IBM WebSphere Adapter for FTP (IBM : 7.0.0.0_IF01)** and select **CWYFT_FTPFile**



__ a. Click **Next**

____ 4. 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: wsbeta159.austin.ibm.com

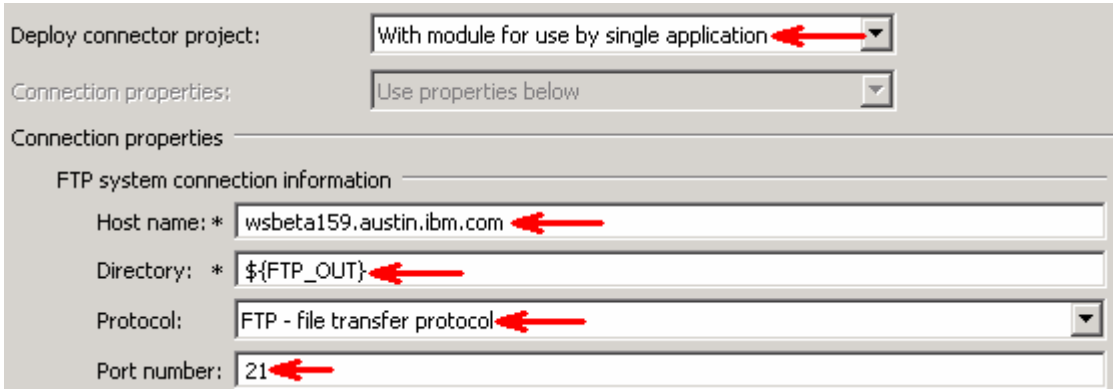
2) Directory: **${FTP_OUT}**

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

---

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

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



____ 5. Optionally, click **Advanced >>** to see the hidden advanced properties that can be configured:

__ a. **Secure configuration**: Refer to the '**Install and configure SSH server lab**' and '**FTP Adapter – SFTP support lab**' for more details on this new feature

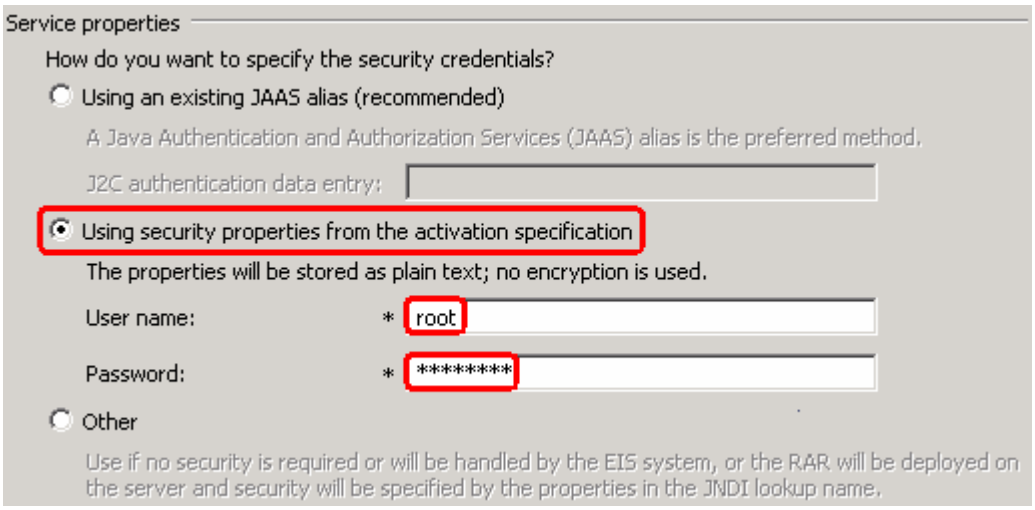
__ b. **Logging and tracing**: Refer to the new lab '**Log and confidential trace lab**' for more details on this new feature

____ 6. For this lab, you are going to use the security properties from the activation specification.

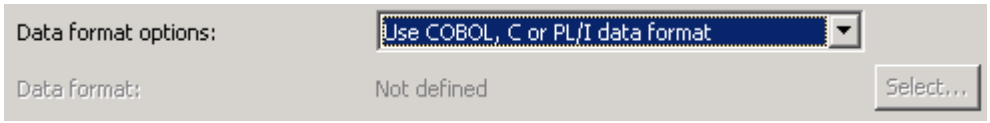
__ a. Select the check box next to **Using security properties from the activation specification**

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

__ c. **Password**: password for the user to connect to your FTP server



____ 7. For **Data format options**, select **Use COBOL, C or PL/I data format** from the drop down list



____ 8. Check the box next to **Change logging properties for wizard** to view the output location of the log file and the logging level and click **Next**

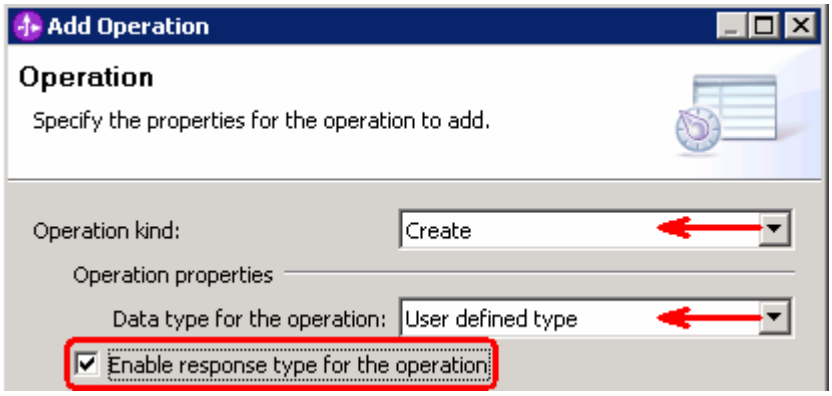
Define createFTPCOBOL operation:

____ 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 **User defined type** from the drop down list

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

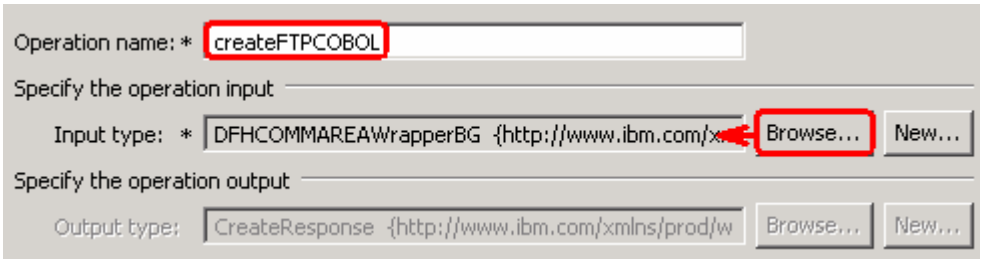


__ d. Click **Next**

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

__ e. For **Operation name**, enter any name, for Ex: **createFTPCOBOL**

__ f. For **Input type**, click **Browse** and select **DFHCOMMAREAWrapperBG** from the Data Type Selection window



__ g. Click **Finish**. The defined operation, **createFTPCOBOL**, is populated under Operations list

Define appendFTPCOBOL operation:

____ 10. From the Operations screen, click **Add…**

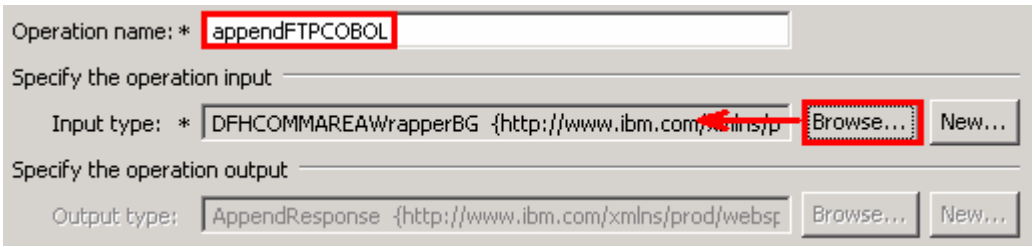
__ a. In the Add Operation window, select **Append** from the drop down menu for Operation kind

__ b. Select **User Defined Type** for the Data type

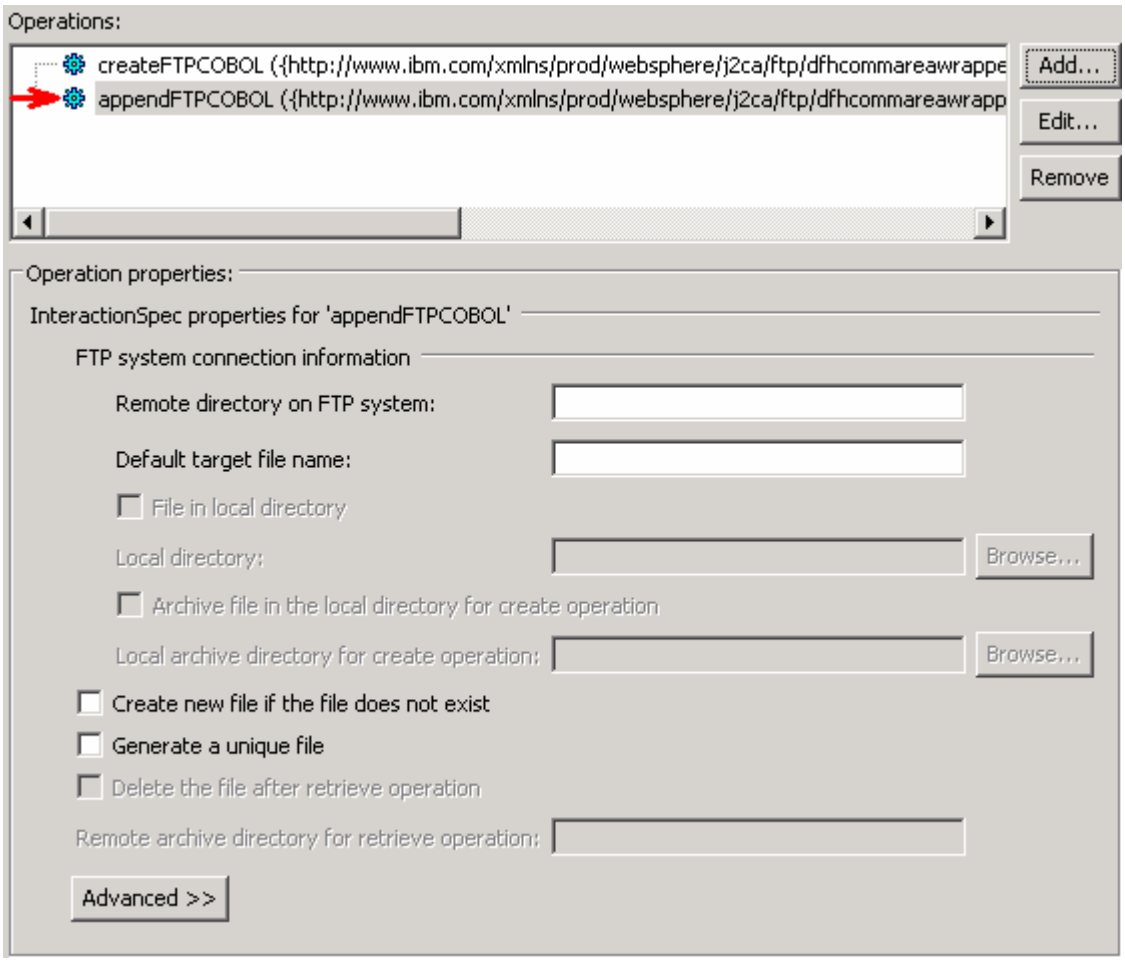
__ c. Select the check box next to **Enable response type for the operation** and click **Next**

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

__ e. For Input type, click **Browse** and select **DFHCOMMAREAWrapperBG** from the Data Type Selection window



__ f. Click **Finish**. The defined operation, **appendFTPCOBOL**, is populated under Operations list



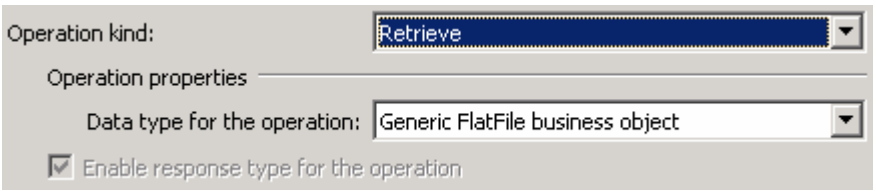
Define retrieveFTPCOBOL operation:

____ 11. 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 FlatFile business object** from the drop down list

__ c. Note that the box next to **Enable response type for the operation** is checked by default



__ d. Click **Next**

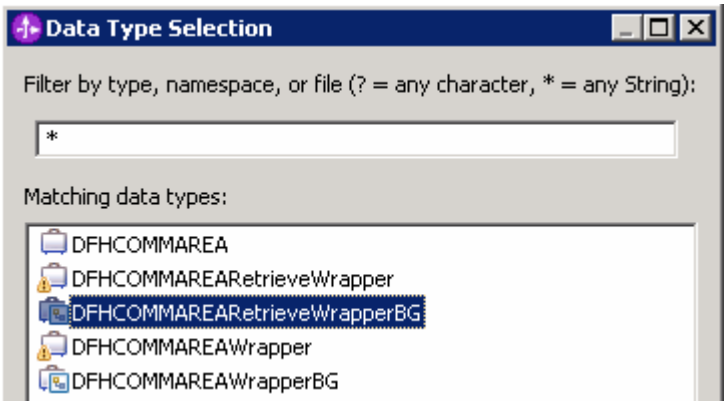
In the Add Operation window, under Specify the operation input, you will see the Input type **FTPFile** (because you have selected **not to** have business graph (BG)) and you will also see the Output type **RetrieveResponseWrapper** under Specify the operation output (because the Output was selected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

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

__ f. Define Output type:

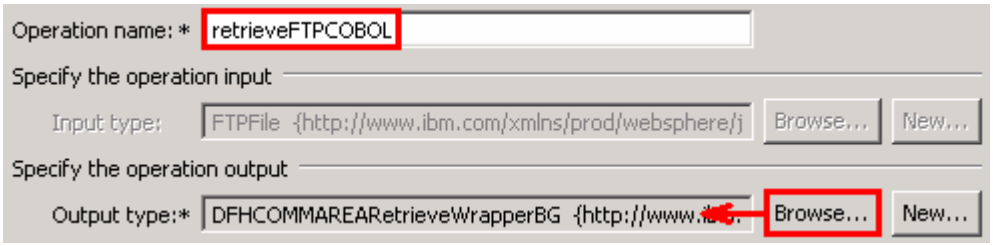
1) Under **Specify the operation output**, click **Browse…** next to **Output type** to open a Data Type Selection window

2) From the Data Type Selection window, select **DFHCOMMAREARetrieveWrapperBG** under Matching data types



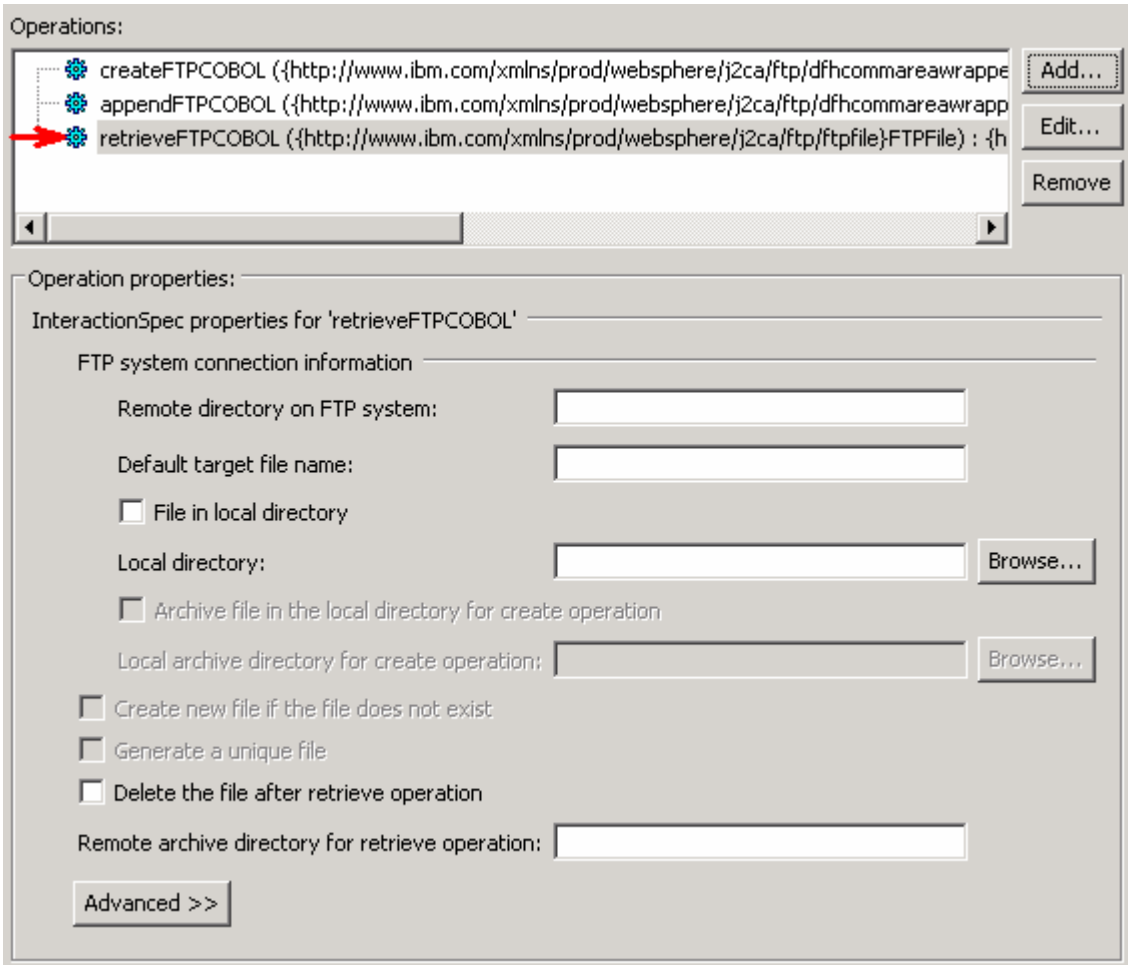
3) Click **OK**

__ g. You should now see this:



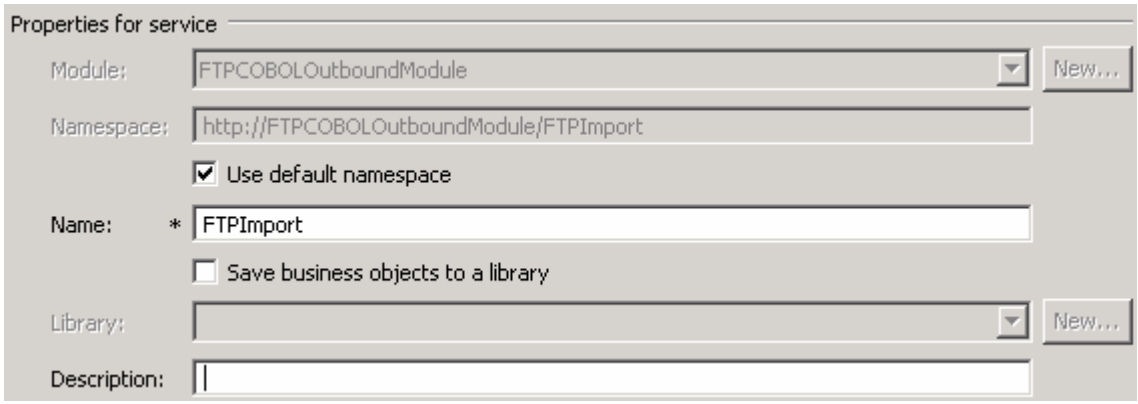
__ h. Click **Finish** from the Add Operation window. The operation, retrieveFTPCOBOL, will now be displayed under Operations list.

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



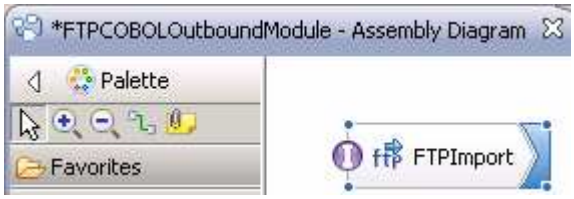
__ j. Click **Next** from Operations screen

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



__ a. Click **Finish**

____ 13. The Assembly diagram for FTPCOBOLOutboundModule is opened with an Import component, **FTPImport**:



____ 14. Save (**Ctrl + S**) changes to your assembly diagram

3.3. Test outbound 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 COBOL copybook create scenario.

____ 1. Add the project to the WebSphere Test Environment server

__ a. Right-click **WebSphere Process Server v7.0** under the Servers view and select **Add and remove projects…** from the pop-up menu

__ b. From the Add and Remove Projects window, select **FTPCOBOLOutboundModuleApp** under Available projects panel and click **Add >**

__ c. You will now see the **FTPCOBOLOutboundModuleApp** 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 **FTPCOBOLOutboundModule** and select **Test > Test Module**

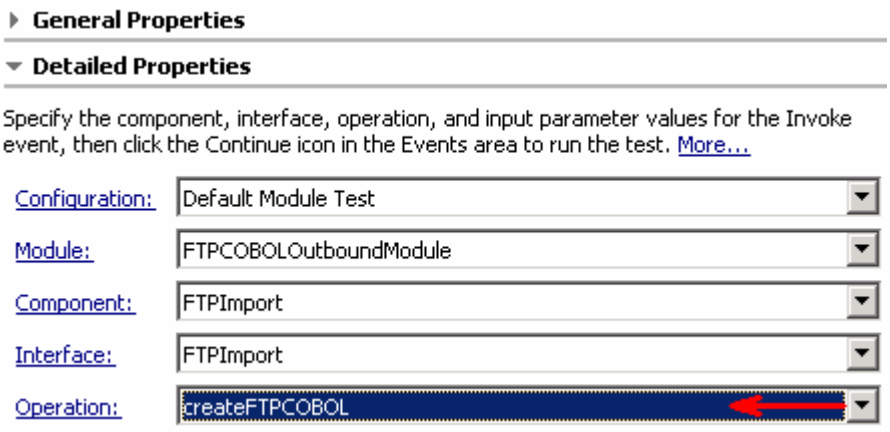
__ b. The **FTPCOBOLOutboundModule_Test** window is opened in the Assembly editor

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

- createFTPCOBOL
- appendFTPCOBOL
- retrieveFTPCOBOL

Test Create operation:

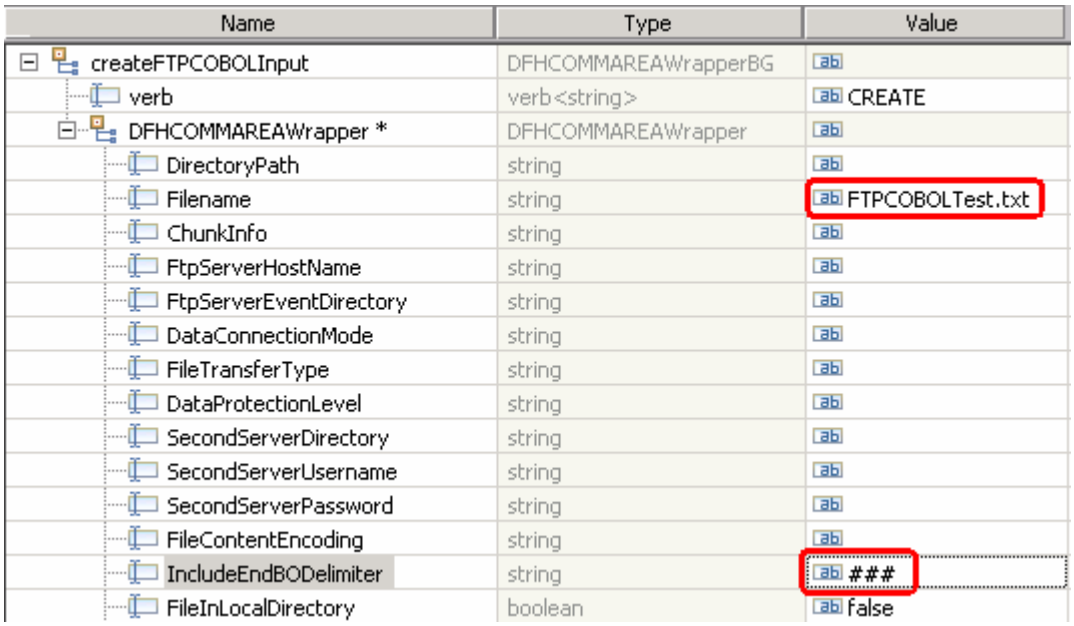
____ 3. Under **Detailed Properties**, for the **Operation** field, select **createFTPCOBOL** from the drop down menu



Fill out the fields for Initial request parameters:

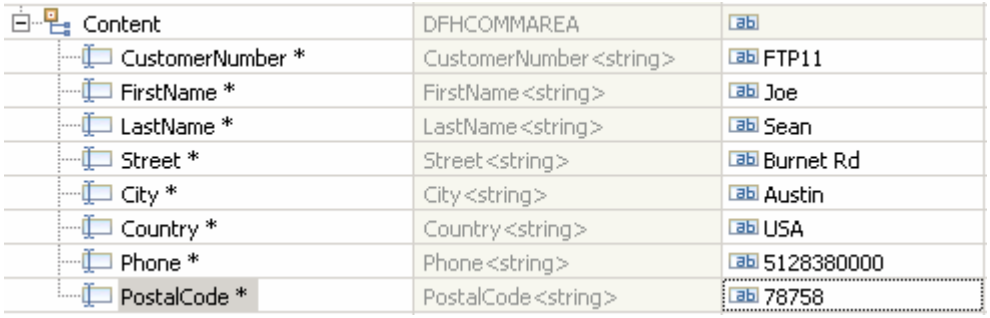
__ a. For **Filename**, enter any name, for Ex: **FTPCOBOLTest.txt**

__ b. For **includeEndBODelimiter**, enter ###



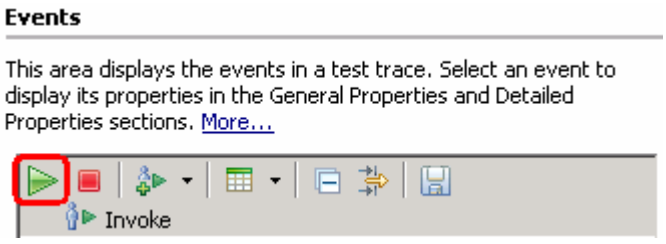
Name	Type	Value
createFTPCOBOLInput	DFHCOMMAREAWrapperBG	
verb	verb<string>	CREATE
DFHCOMMAREAWrapper *	DFHCOMMAREAWrapper	
DirectoryPath	string	
Filename	string	FTPCOBOLTest.txt
ChunkInfo	string	
FtpServerHostName	string	
FtpServerEventDirectory	string	
DataConnectionMode	string	
FileTransferType	string	
DataProtectionLevel	string	
SecondServerDirectory	string	
SecondServerUsername	string	
SecondServerPassword	string	
FileContentEncoding	string	
IncludeEndBODelimiter	string	###
FileInLocalDirectory	boolean	false

__ c. For the fields under **Content**, enter any values:

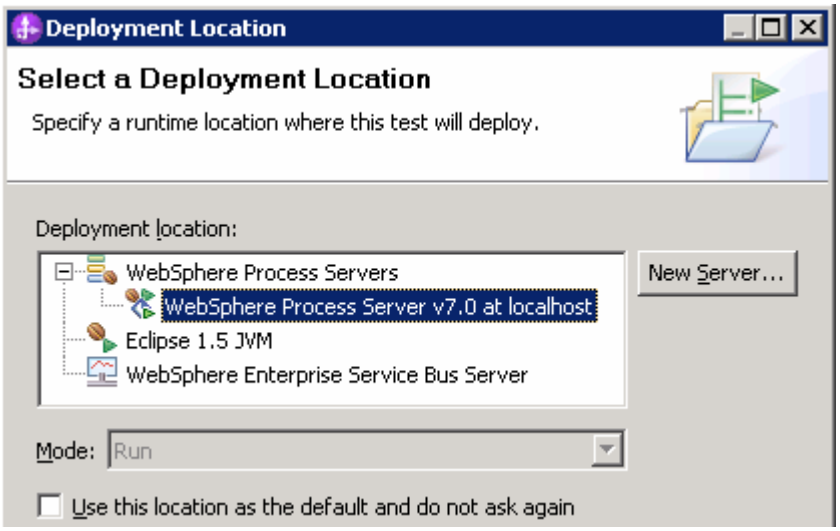


Content	DFHCOMMAREA	
CustomerNumber *	CustomerNumber<string>	FTP11
FirstName *	FirstName<string>	Joe
LastName *	LastName<string>	Sean
Street *	Street<string>	Burnet Rd
City *	City<string>	Austin
Country *	Country<string>	USA
Phone *	Phone<string>	5128380000
PostalCode *	PostalCode<string>	78758

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

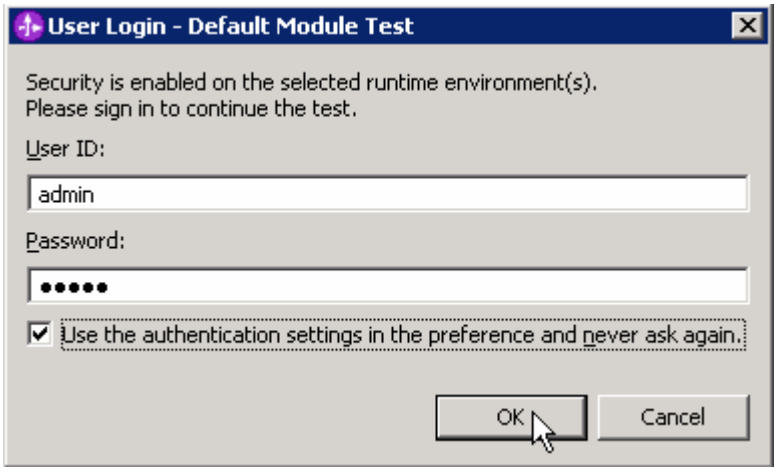


__ e. From Deployment Location window, select **WebSphere Process Servers > WebSphere Process Server v7.0 at localhost** and click **Finish**



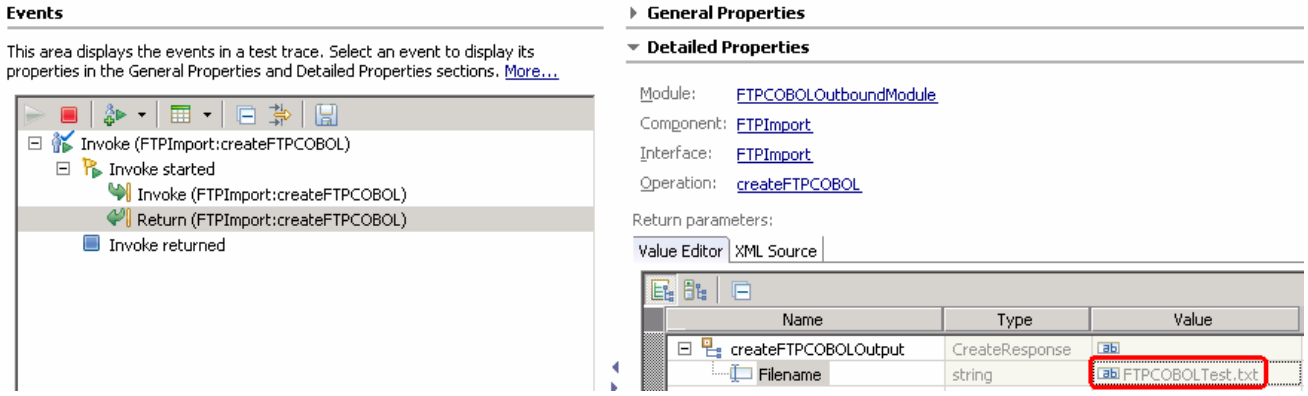
__ f. Provide Administrator User ID and Password

1) Optionally, select the box 'Use the authentication settings in the preference and never ask again' so that you do not have to enter the credentials next time when you start the test client



__ g. Verify the created file and its contents:

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



2) You will see a new file, **FTPCOBOLTest.txt** created under **OutDir** on your FTP server. Open that file to verify the content

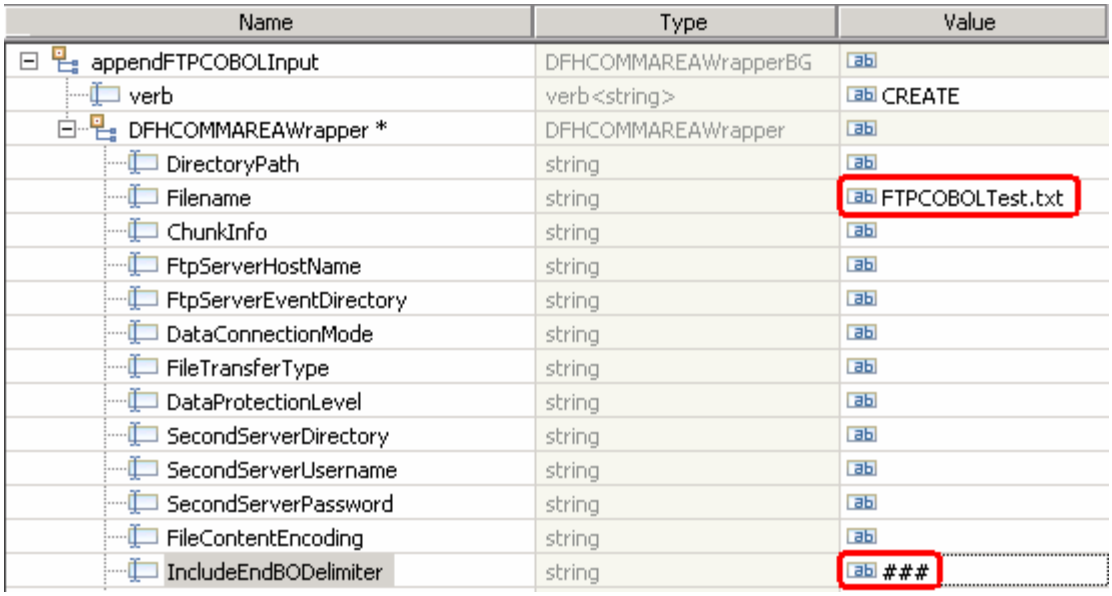
Test Append operation:

____ 4. Under **Detailed Properties**, note that the Operation is **appendFTPCOBOL**

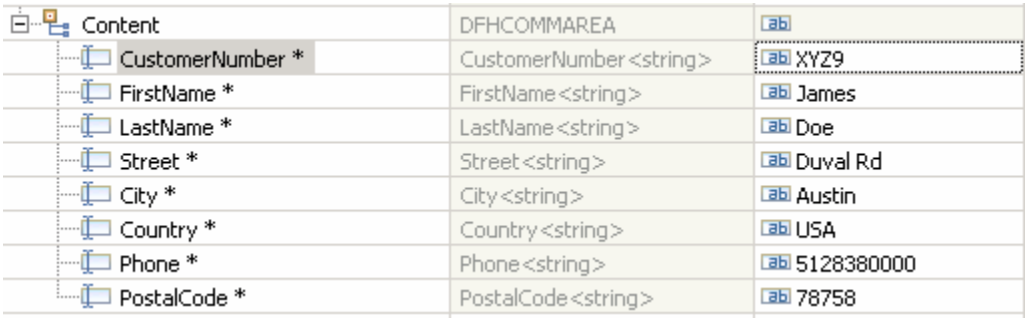
__ a. Fill out the fields for Initial request parameters:

__ b. For **fileName**, enter **FTPCOBOLTest.txt**

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



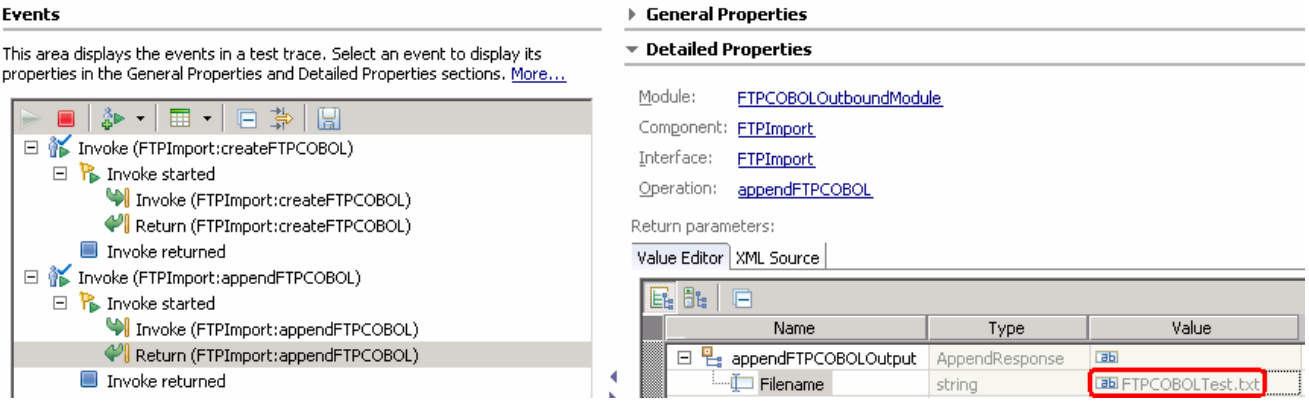
__ d. Enter the values of your choice for the fields under **Content**



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

__ f. Verify the results:

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



__ g. To verify your test results, open the FTPCOBOLTest.txt file from **OutputDir** of your FTP server and it should contain the text entered under content field in your test client. The highlighted text is the appended content and note the end BO delimiter, ###, at the end of each COBOL BO

Test Retrieve operation:

____ 5. Under **Detailed Properties**, note that the Operation is **retrieveCOBOL**

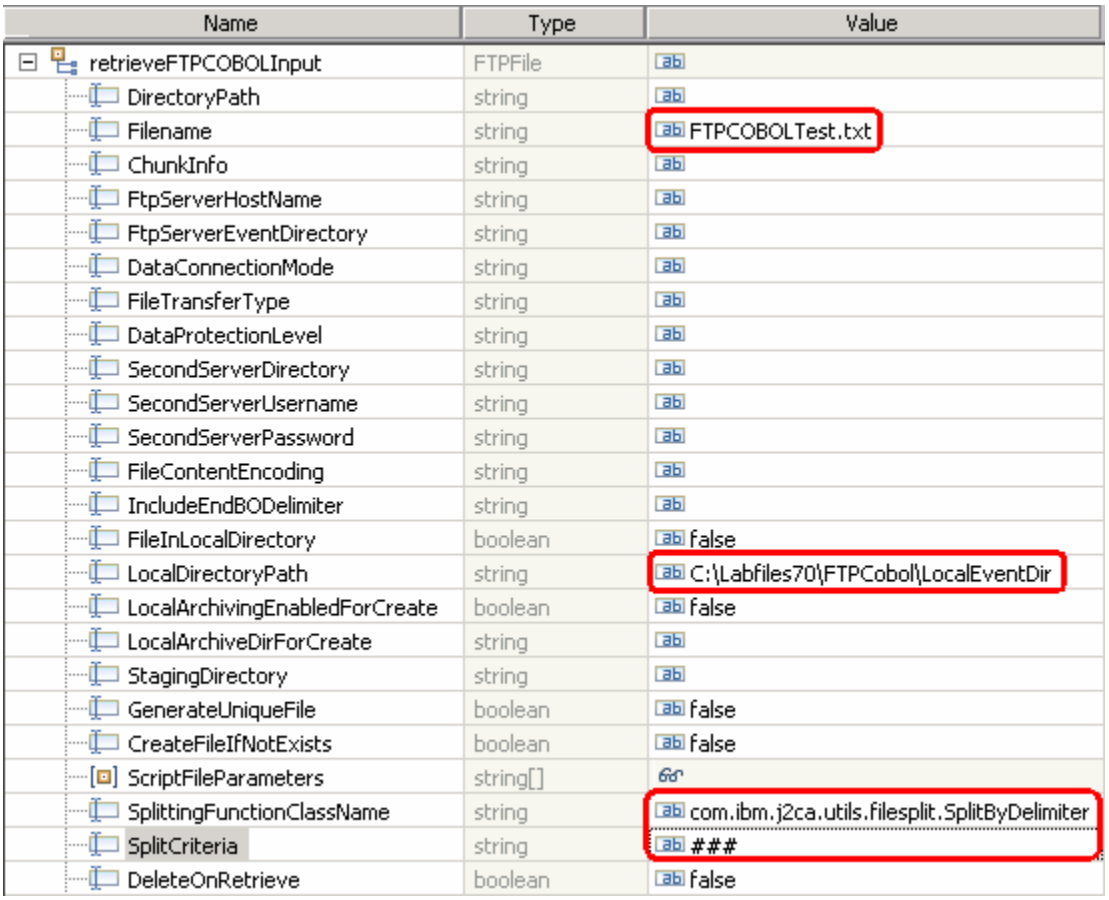
__ a. Fill out the fields for Initial request parameters:

__ b. For **filename**, enter **FTPCOBOLTest.txt**

__ c. **LocalDirectoryPath:** provide absolute path to any local directory

__ d. **splitFunctionClassName**: **com.ibm.j2ca.utils.filesplit.SplitByDelimiter**

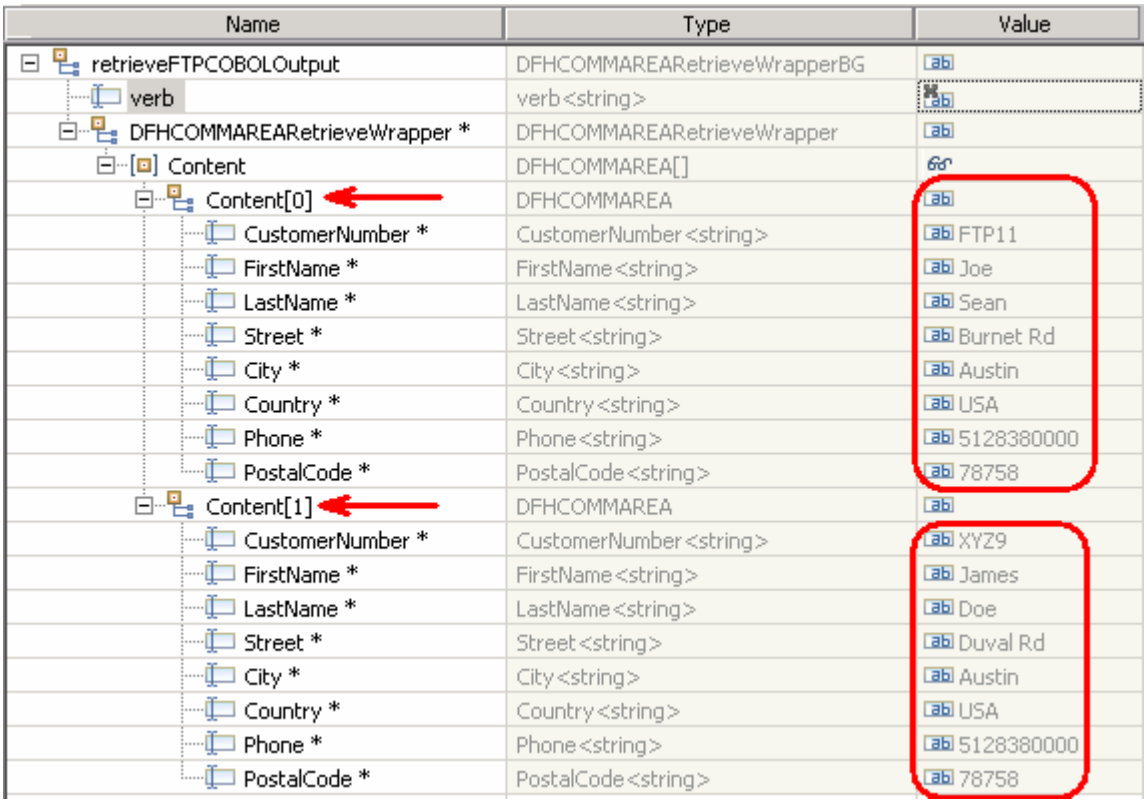
__ e. **splitCriteria**: ###



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

__ g. Verify the results:

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



____ 6. Restore the Sever Configuration

__ a. Right-click **WebSphere Process Server v7.0** under the Servers view and select **Add and remove projects…** from the pop-up menu

__ b. Select **FTPCOBOLOutboundModuleApp** 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

Solution instructions

____ 1. Start WebSphere Integration Developer V7.0 with a new workspace

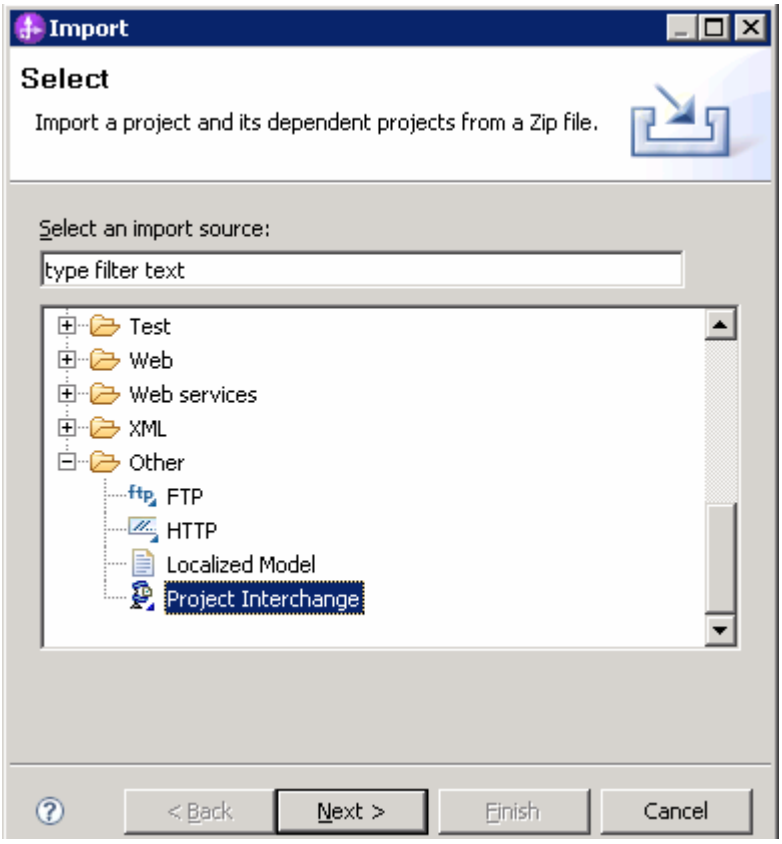
__ a. Follow the instructions outlined in **Part 1** of this exercise

____ 2. Import the Project Interchange

__ a. Import the project interchange file **FTPCOBOL_PI.zip** from **<LAB_FILES>\FTPCobol\solution** directory

__ b. Select **File → Import** from the menu

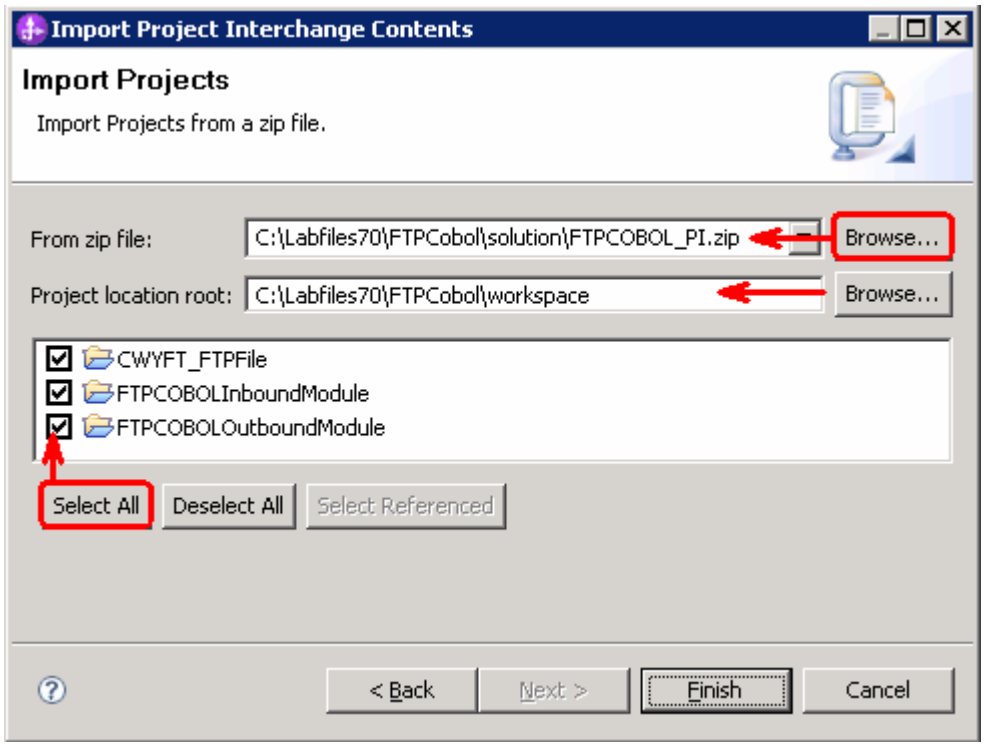
__ c. Select **Other → Project Interchange** in the **Import** dialog and click **Next**



__ d. For the **From zip file**, click on the **Browse** button and select the **FTPCOBOL_PI.zip** in the **<LAB_FILES>>\ FTPCobol\solution** directory

__ e. Enter **<LAB_FILES>\FTPCobol\workspace** for the **Project location root**

__ f. Click the **Select All** button. This will select all the modules: **CWYFT_FTPFile, FTPCOBOLInboundModule,** and **FTPCOBOLOutboundModule**



__ g. Click **Finish**

____ 3. Continue with **Part 2.4** of this lab to test the inbound scenario

____ 4. Continue with **Part 3.3** of this lab to test the outbound scenario

What you did in this exercise

In this lab, you started with importing the FTP Adapter RAR file into your WebSphere Integration Developer new workspace. Next, you made use of the External Data wizard to generate business object and wrapper definitions from a COBOL program source file. Then you continued to with External Service wizard available in WebSphere Integration Developer to generate other artifacts for inbound and then for outbound.

At the end of each part, you deployed and tested the adapter application for operations defined during the external service wizard.

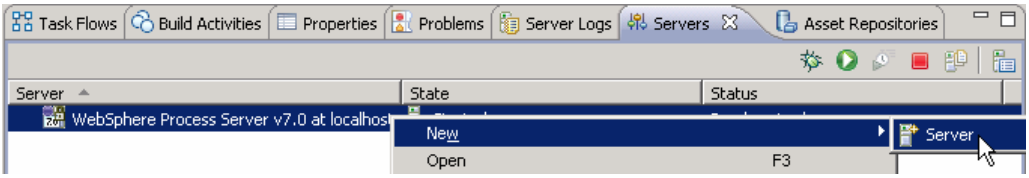
Task: Adding remote server to WebSphere Integration Developer test environment

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

____ 1. Define a new remote server to WebSphere Integration Developer.

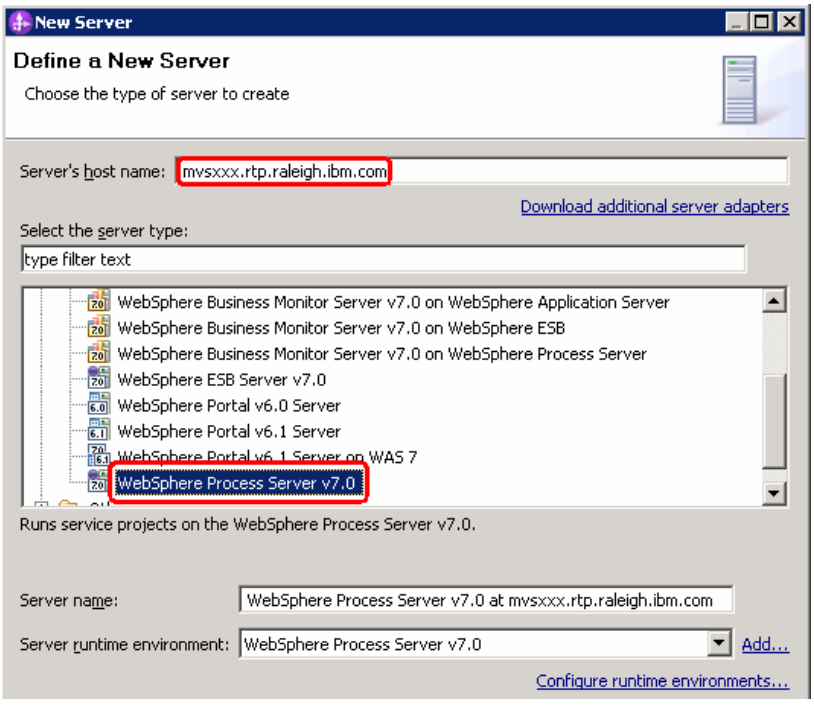
__ a. Right click the background of the Servers view to access the pop-up menu.

__ b. Select **New → Server**.



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

__ d. Ensure that the appropriate server type, '**WebSphere Process Server v7.0**' or '**WebSphere ESB Server v7.0**', is highlighted in the server type list

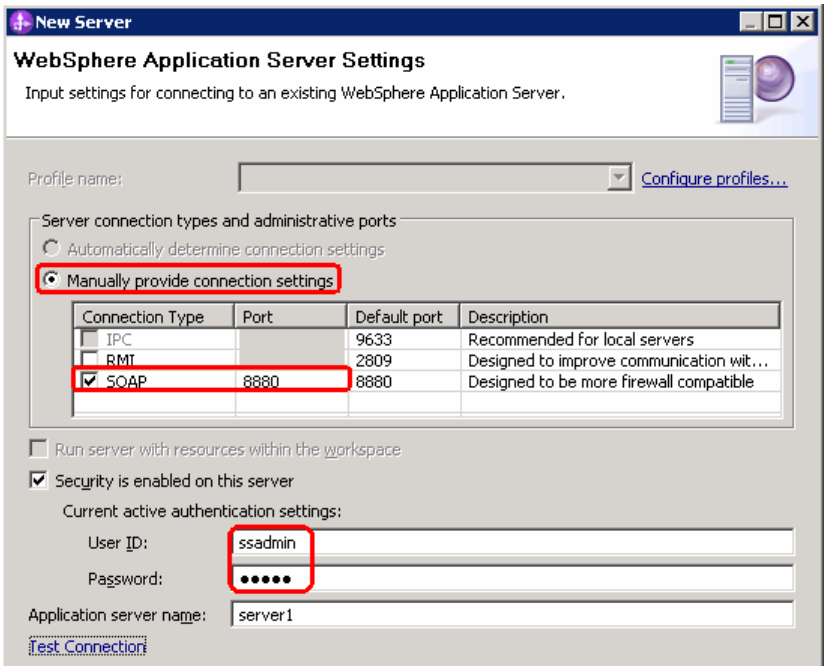


__ e. Click **Next**

__ f. On the WebSphere Server Settings page, leave the radio button for **Manually provide connection settings** selected, and select the box for SOAP

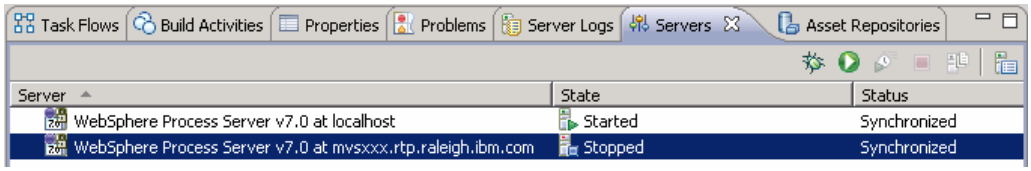
__ g. Enter the correct setting (**<SOAP_PORT>**) for **Port** column

__ 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.



__ i. Click **Finish**.

__ j. The new server should be seen in the Server view.



____ 2. Start the remote server if it is not already started. WebSphere Integration Developer does not support starting remote servers from the Server view.

__ a. From a command prompt, telnet to the remote system if needed:

'**telnet <HOSTNAME> <TELNET_PORT>**'

User ID : **<USERID>**

Password : **<PASSWORD>**

__ 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
```