Log and confidential trace lab

What this exercise is about

The objective of this lab is to provide you with an understanding of some of new features in WebSphere Adapter for Flat Files V6.2 – usage of Adapter ID for logging and tracing details, hide confidential trace, and poll sub directories in event directory.

Lab requirements

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

- WebSphere Integration Developer V6.2 installed and updated with latest fixes

- WebSphere Process Server V6.2 Test Environment installed and updated latest fixes

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

What you should be able to do

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

- Import Flat File adapter RAR file into WebSphere Integration Developer

- Use External Service wizard to configure Activation Spec Properties, Resource Adapter Properties to generate Business Objects

- Deploy the adapter application onto WebSphere Process Server and enable tracing in WebSphere Process Server administrative console

- Test the deployed application using WebSphere Process Server test environment

- Load the trace file and review trace messages from the new Server Logs view

- Review the trace/log file contents for the 'confidential' messages replaced with 'XXX'

- Restore the server configuration

Introduction

The adapter provides the ability to protect sensitive or confidential data in log and trace files from being seen by those without authorization.

Log and trace files for the adapter can contain data, which might contain sensitive or confidential information. Sometimes it is necessary for these files to be seen by individuals without authorization to view to sensitive customer data, for example, a support specialist that must use the log and trace files to troubleshoot a problem.

To protect the data in situations like this, the adapter provides the HideConfidentialTrace property. The HideConfidentialTrace property specifies whether you want to prevent confidential user data from displaying in the adapter log and trace files. When this property is enabled, the adapter replaces the confidential data with XXX's.

The types of information that are considered potentially sensitive data and hidden are:

- The contents of a business object

- The contents of an event record

- User ID

- Business object data in an intermediate form, such as a comma-delimited version of a file

The types of information that are not considered user data and not hidden are:

- Business object schemas

- Transaction IDs

- Event IDs

- Call sequences

Exercise instructions

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

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

Reference variable	Windows location	AIX® or UNIX® location
<WID_HOME>	C:\Program Files\IBM\WID62	
<WPS_HOME>	C:\<WID_HOME>\runtimes/bi_v62	
<FFADAPTER_HOME>	<WID_HOME>\ResourceAdapters\FlatFile_6.2.0.0\deploy	
<LAB_FILES>	C:\Labfiles62	/tmp/Labfiles62
<WORKSPACE>	<LAB_FILES>\LogAndTrace\workspace	
<EVENT_DIR>	<LAB_FILES>\LogAndTrace\eventdir	
<ARCHIVE_DIR>	<LAB_FILES>\LogAndTrace\archivedir	
<FFFILES>	<LAB_FILES>\FFFiles	
<TEMP>	C:\temp	/tmp

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

Instructions if using a remote server for testing

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

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

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

Part 1: Initialize the workspace and prepare for the lab

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

____ 1.   Extract the provided Labfiles62.zip to your C:\ (root) drive, if you have not already done so. This will create the necessary subdirectory structure to complete the lab, and provides you with sample text files

____ 2.   Start the WebSphere Integration Developer V6.2 with a new workspace

__ a. Select **Start > All Programs > IBM WebSphere Integration Developer > IBM WebSphere Integration Developer V6.2 > WebSphere Integration Developer V6.2**

__ b. From the Workspace Launcher window, enter **<WORKSPACE>** for the Workspace field
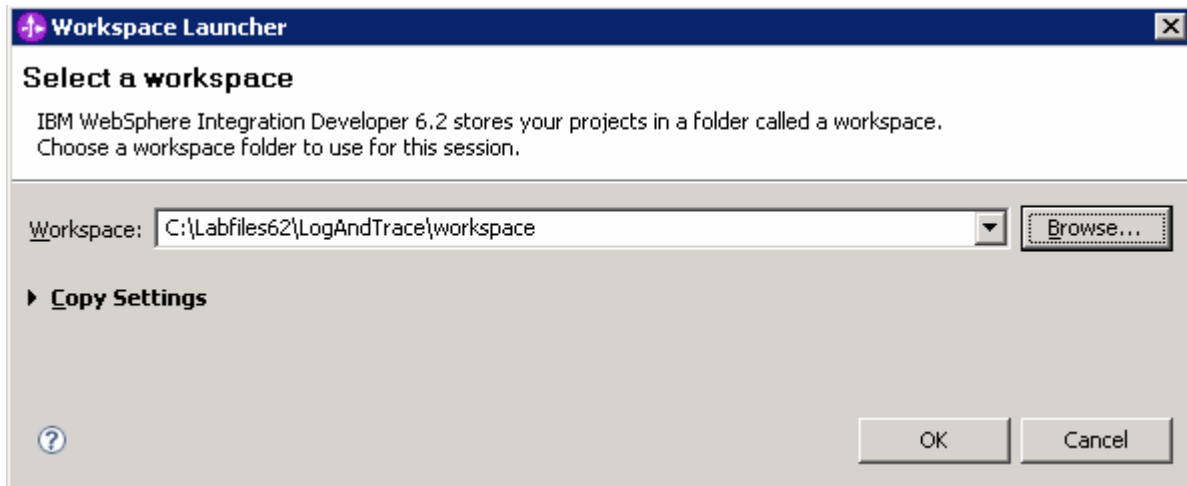
____ 3.   Click the [icon] button on the right to close the Welcome page and proceed with the workbench

____ 4.   Follow the instructions of "**Define WebSphere variables"** task to define the WebSphere variables listed in the table below, which are used later in this lab:

| Name	Value
LOGTRACE_EVENT	C:\Labfiles62\LogAndTrace\eventdir
LOGTRACE_ARCHIVE	C:\Labfiles62\LogAndTrace\archivedir

Note: In FlatFile adapter, you have to provide directory values (Event directory, Archive directory and so on, in activation spec and managed connection factory) while running external service wizard. Presently these values can be changed in the module (or) can be changed in deployed application from administration console. This is like hard coding the values of these properties. All local directories in ActivationSpec can be filled with **WebSphere environment variables**. This way hard coding of directory paths for these properties can be avoided. In WebSphere Process Server V6.1 and above, you can declare these values as Environment Variables using your WebSphere Process Server administration console 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.

WBPMv62_IEA_AdapterFFInboundLab.doc

Part 2: Configure non-pass through inbound scenario

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

____ 1.   Create the module: LogAndTraceModule

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

__ b. From the New Module window, enter **LogAndTraceModule** for the Module Name
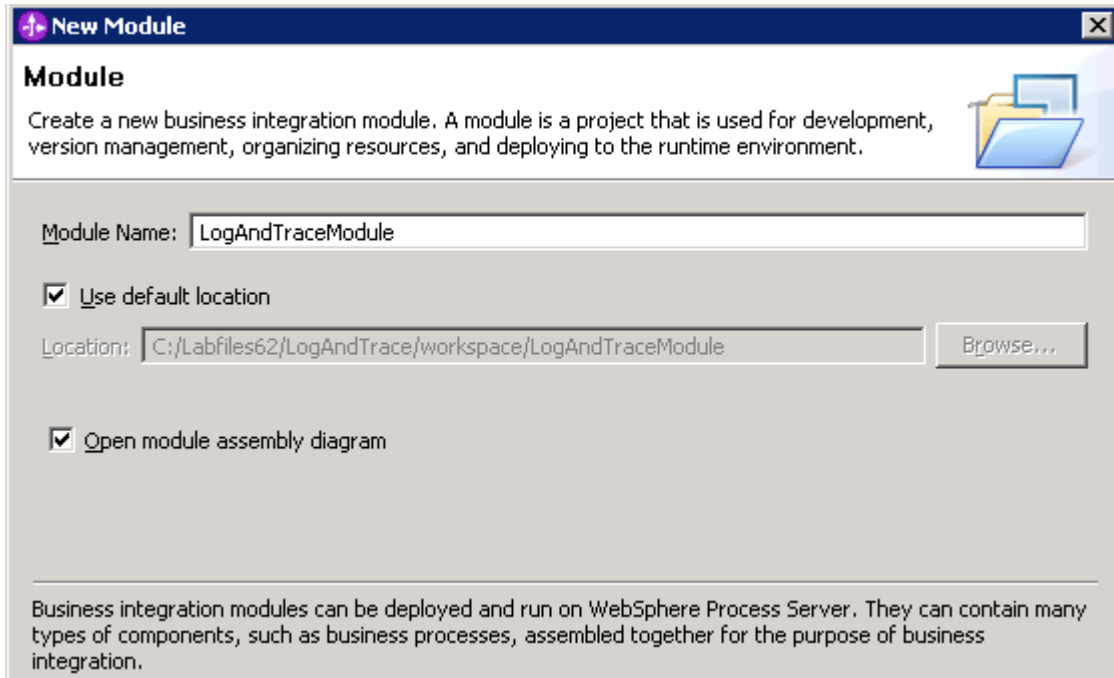
__ c. Ensure that the box next to **Open module assembly diagram** is checked and then click **Finish**

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

____ 2.   Import required business objects

__ a. Expand LogAndTraceModule (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 **<FFFILES >** and click **OK**

Now, you will see FFFiles 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 **LogAndTraceModule** is selected for Into folder

____ f. Click **Finish** from the Import window

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

_____ 3.   Review imported business objects:

____ a. Expand **LogAndTraceModule > Data Types** and you will now see two new data types **Customer** and **Order** under it.
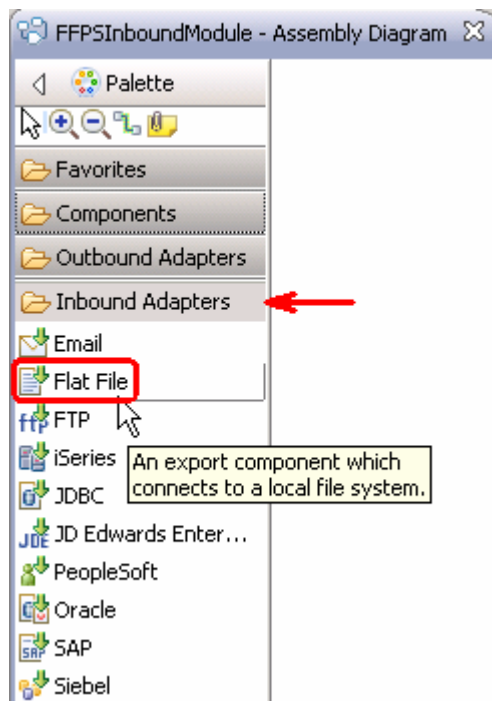
____ b. Double-click **Customer** review the fields inside the object:
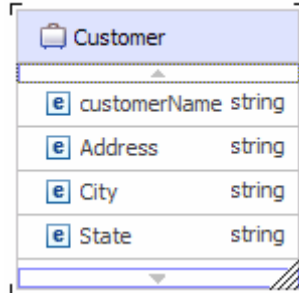
_____ 4.   After reviewing, close the Customer business object from the Assembly editor

_____ 5.   To start External Service from the Palette:
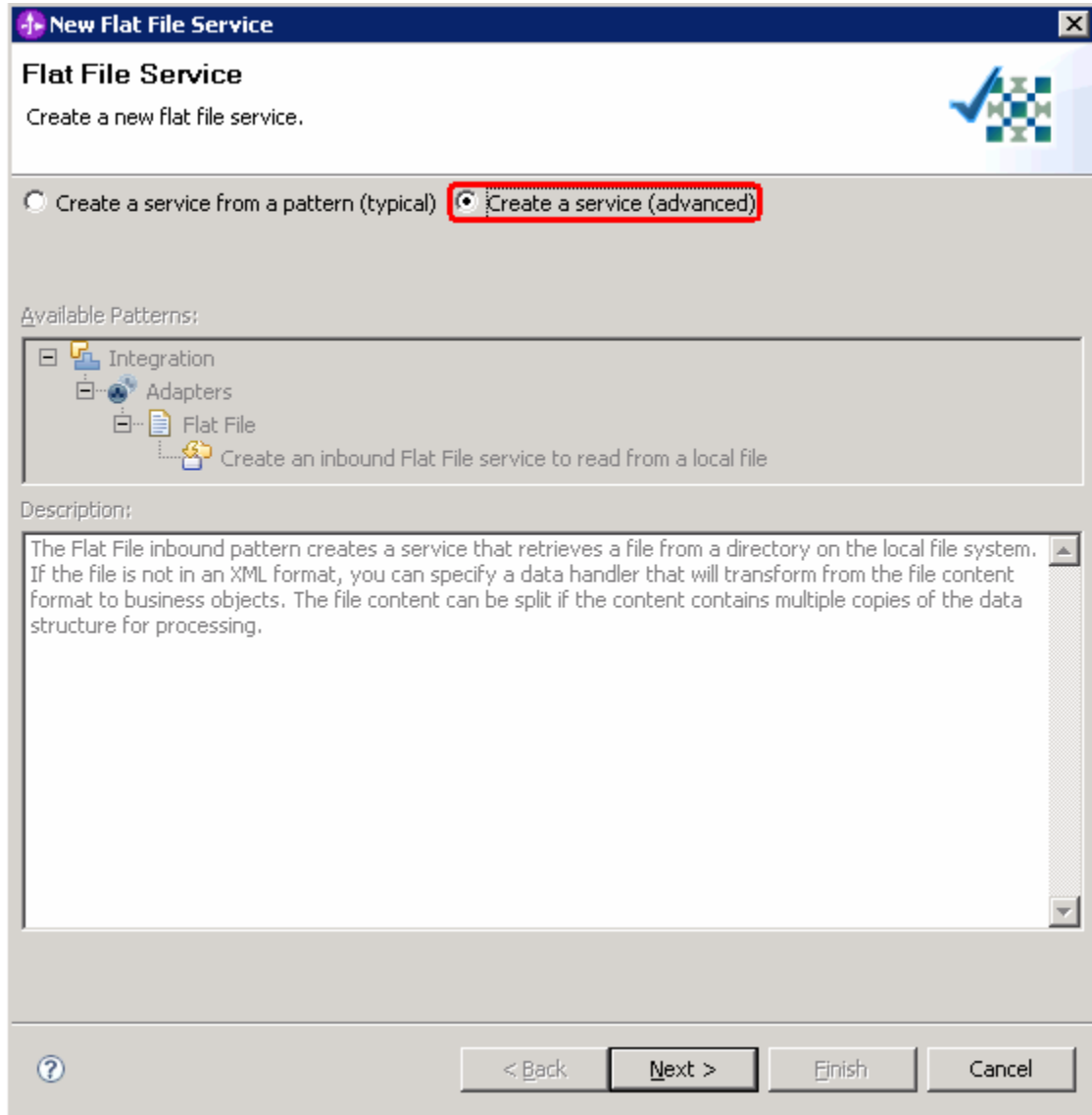
____ a. From the **Palette** on the left side of Assembly Diagram, click **Inbound Adapters**:

_____ 6.   Under Inbound Adapters, click the **Flat File** and then click the empty canvas of the assembly diagram. The New Flat File Service wizard is opened

____ 7.   From the Flat File Service screen, select **Create a service (advanced)**


    __ a. Click **Next**

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

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

Select **Adapters > Flat File** and click **Next**

____ 8.   On the Select an Adapter screen, select **IBM WebSphere Adapter for Flat Files (IBM : 6.2.0.0)** and click **Next**

*WBPMv62_IEA_AdapterFFInboundLab.doc*

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE
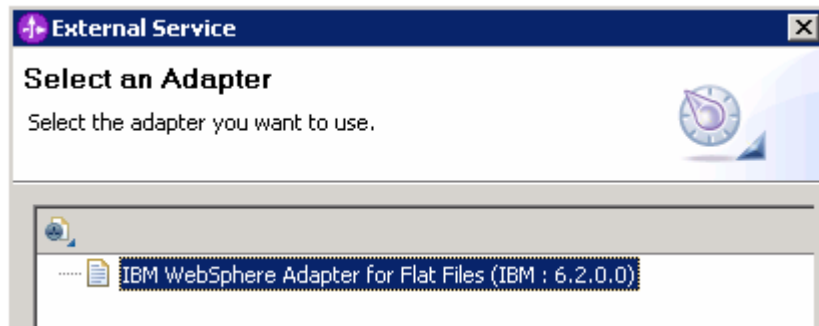
_____ 9.   Adapter Import screen:

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 **<FFADAPTER_HOME>**.

   __ a. The default Connector file is selected which is shipped along with WebSphere Integration Developer

   __ b. Accept the default name for Connector project, **CWYFF_FlatFile**. You can change it to any other name, but for this lab, you can leave the default name

   __ c. For Target server, ensure that **WebSphere Process Server v6.2** is selected
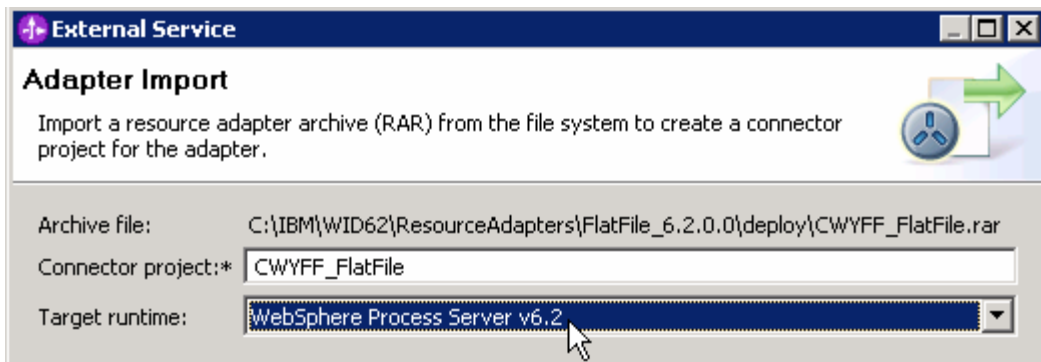
   __ d. Click **Next**

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

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

_____ 10.   Service Configuration Properties:

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

   __ b. Under Connection Configuration, for Event directory enter **${LOGTRACE_EVENT}**

2009 April, 13   *IBM WebSphere Adapter 6.2 – Lab exercise*   *Page 9 of 32*

WBPMv62_IEA_AdapterFFInboundLab.doc

Note: The WebSphere Environment Variable 'LOGTRACE_EVENT' is already created in previous parts of this lab.

Deploy connector project:	With module for use by single application ▼
Connection properties:	Use properties below ▼

Connection properties

File system connection information

Event directory:* ${LOGTRACE_EVENT}   Browse...

Note: Alternatively, you can also **Browse…** and select the event directory.

 __ 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 polling configuration: This has all the polling configuration details and for this lab, you can accept the defaults

 __ e. Event delivery configuration:

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

 __ f. Event persistence configuration:

Note: This represents the JNDI name of the Data Source used by Event Persistence to get the JDBC database connection. The Data Source must be created in the WebSphere Process Server.

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

 __ g. Advanced properties:

   1) Select the box for **Poll subdirectories in event directory**

Note: '**Poll subdirectories in event directory'** is the new Boolean property added in V6.1.2. This property is used for recursive polling. Flat File adapter polls the given 'Event directory' and all the sub-directories present in the 'Event directory'.

__ h. File archiving configuration:

> 1) Archive directory: **${LOGTRACE_ARCHIVE}**

Note: The WebSphere Environment Variable 'FF_ARCHIVE' is already created previously in Part 1, Step 5 of this lab.

__ i. **Logging and tracing**:

> 1) Enter ID: **001** (enter any numeric value)

Note: This property identifies the adapter instance in log and trace files and for PMI events. The adapter ID is used with an adapter-specific identifier, MyAdapterRA, to form the component name used by Log and Trace Analyzer, For example, if the adapter ID property is set to 001, the component ID is FFRA001 for Flat File adapter, FTPRA001 for FTP adapter, EMARA001 for Email Adapter. For inbound processing, this property is retrieved from the resource adapter properties. For outbound processing, it is retrieved from the managed connection factory properties:

Adapter	Component ID (Filter string)
Flat File	FFRA001
FTP	FTPRA001
Email	EMA001
JDBC	JDBC001

2) Select the box for '**Disguise user data as "XXX" in log and trace files**"

Note: If you set this property to **True (selected)**, the adapter will replace user data with a string of x's when writing to log and trace files. For inbound processing, this property is retrieved from the resource adapter properties. For outbound processing, it is retrieved from the managed connection factory properties.
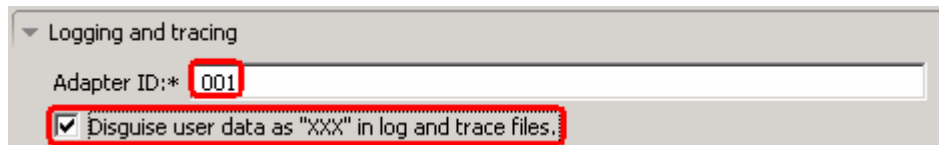

_____ 11. Under Service properties, for Function selector options, select **Use a function selector configuration** from the drop down menu

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

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

___ b. Under '**Use existing function selector from the list**', select **FilenameFunctionSelector properties**
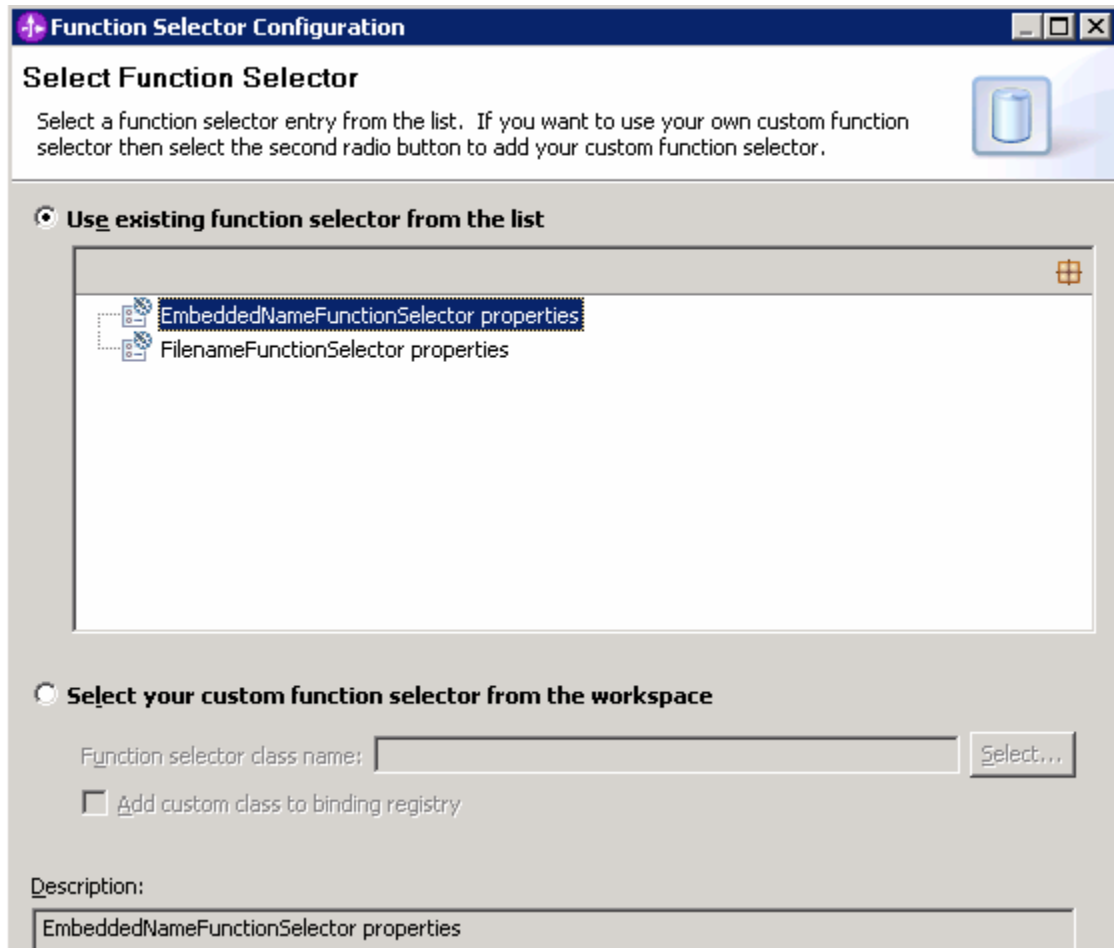
___ c. Click **Next**

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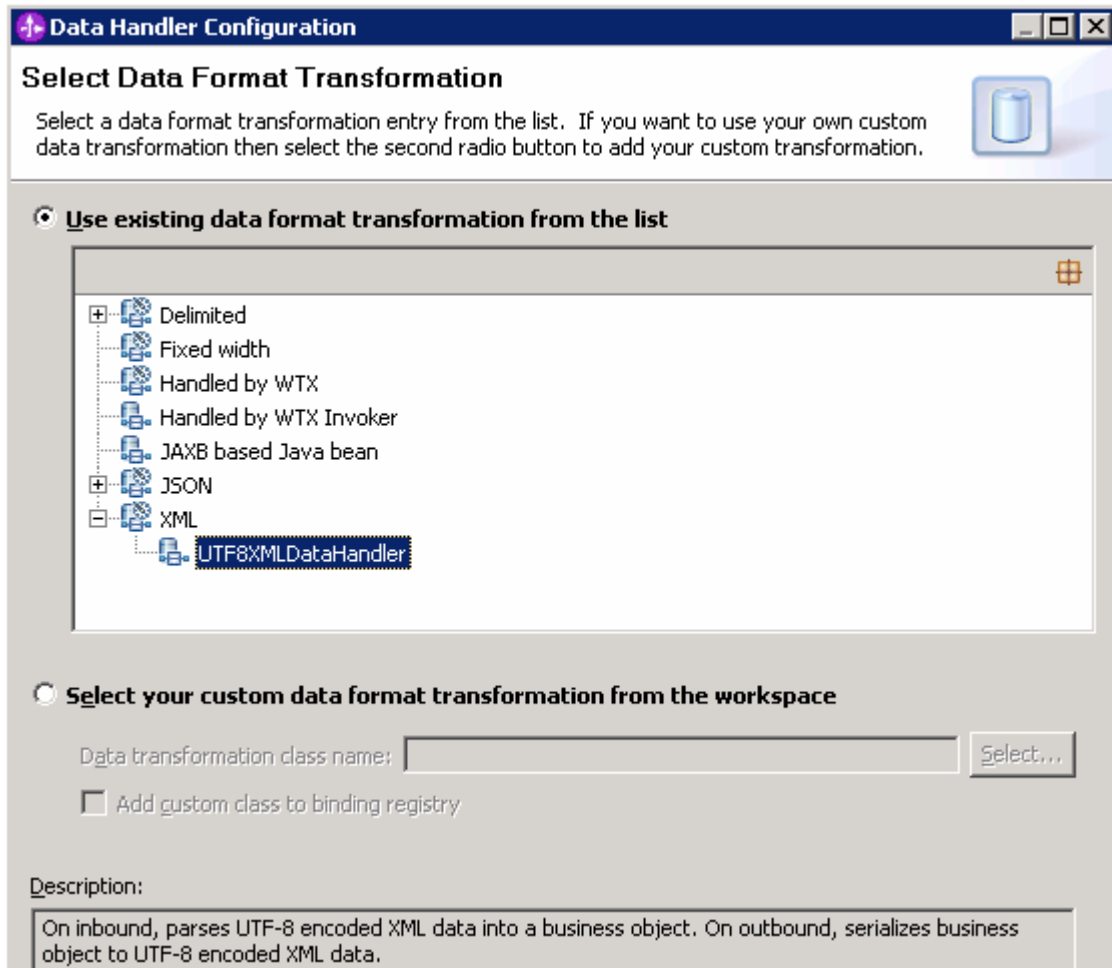
Data handler configuration:

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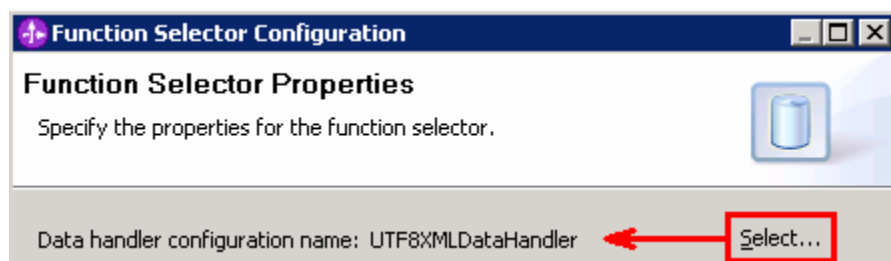
__ e. Under 'Use existing data format transformation from the list', select **XML > UTF8XMLDataHandler**

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

__ f. Click **Finish**

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

__ h. Click **Next**

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

Module or library:	LogAndTraceModule		Browse...	New...
Namespace:	http://LogAndTraceModule	☑ Default		
Folder:			Browse...	
Name:	CustomFS			

__ j. Click **Finish**

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

Service properties

Function selector options: Use a function selector configuration ◄

Function selector: * CustomFS ◄ Select...

Data binding configuration:

____ 12. You can define data binding in two places - service level (current screen of External Service wizard) or later at the method level (Operations screen of the External Service wizard). In this lab, you will define data binding at the service level (from this screen)

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

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 **FlatFileBaseDataBinding**

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**

⊞

FlatFileBaseDataBinding

__ d. Click **Next**

__ e. Select the **UTF8XMLDataHandler** for data handler:

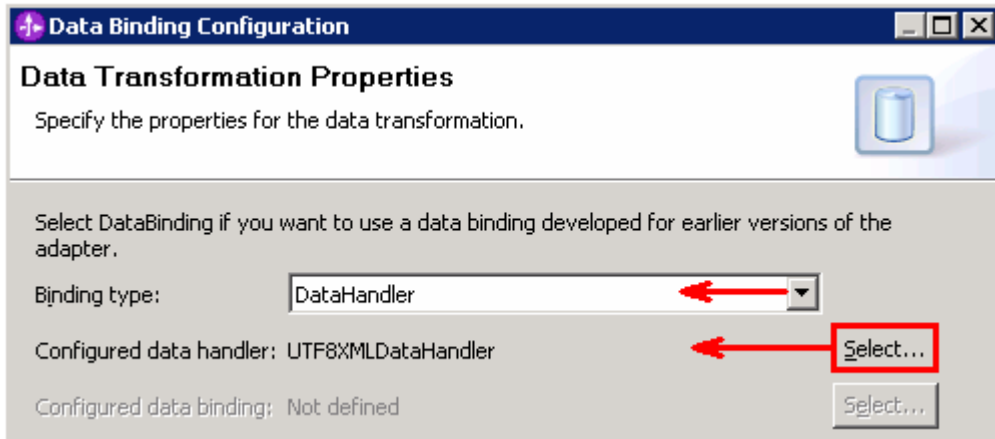　　　1) Click **Select** next to 'Configured data handler'

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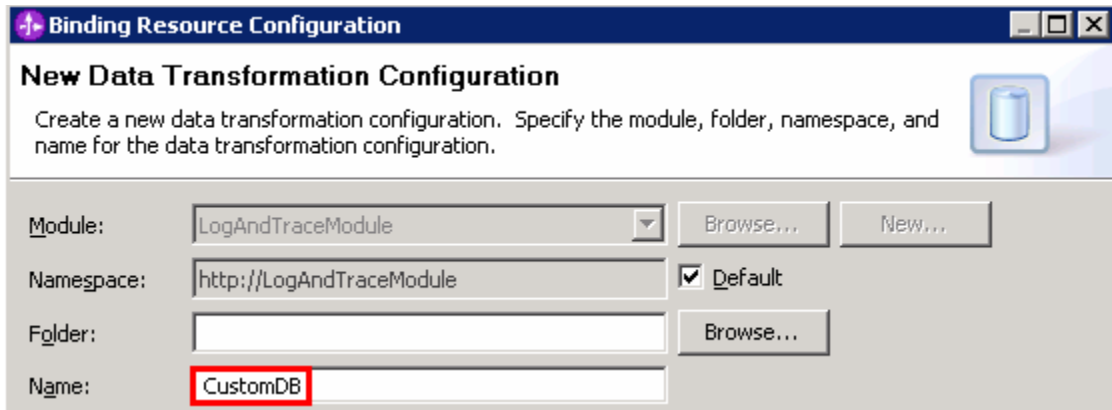
__ f. Click **Finish**

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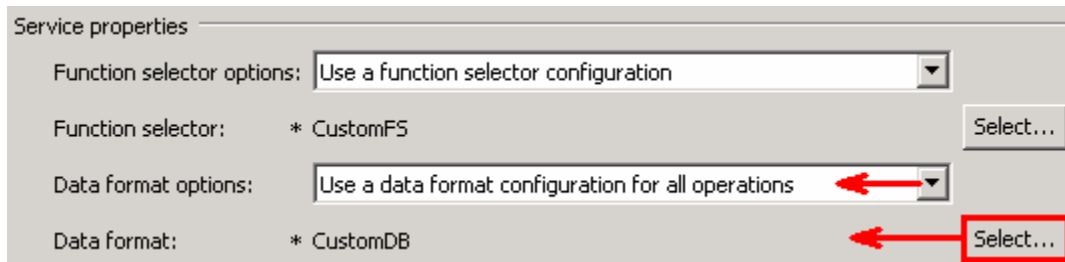is displayed here:

__ h. Click **Next**

__ i. Note that the selected module is **LogAndTraceModule**

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

2) Click **Finish**

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

2009 April, 13   *IBM WebSphere Adapter 6.2 – Lab exercise*   *Page 16 of 32*

WBPMv62_IEA_AdapterFFInboundLab.doc

____ 13. 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 emitCustomerFile operation:

____ 14. From the Operations screen, click **Add…**. Add Operation window is opened

___ a. Select **User defined type** for the Data type and click **Next**
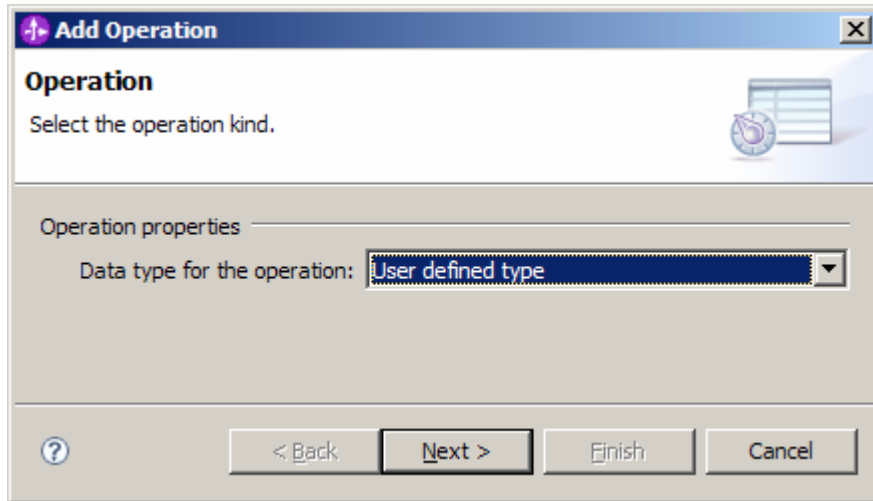
You are now back to Operation window and because you chose the User defined data type, the Input type is **blank** and because you have selected Output required box, the Output type is **CreateResponse**

___ b. For Operation name, enter **emitCustomerFile**

___ c. Define Input type:

1) Click **New…** next to **Input type** to open a New Business Object window

2) From this window, ensure that the Module selected is **LogAndTraceModule** and click **Next**

3) Click **Browse…** next to Data type

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

5) Click **OK**

6) From the wizard, **check** the box next to **Generate business graph for each business object**

*WBPMv62_IEA_AdapterFFInboundLab.doc*

      7) **Do not** check the box for **'Generate retrieve container to retrieve multiple business objects'**
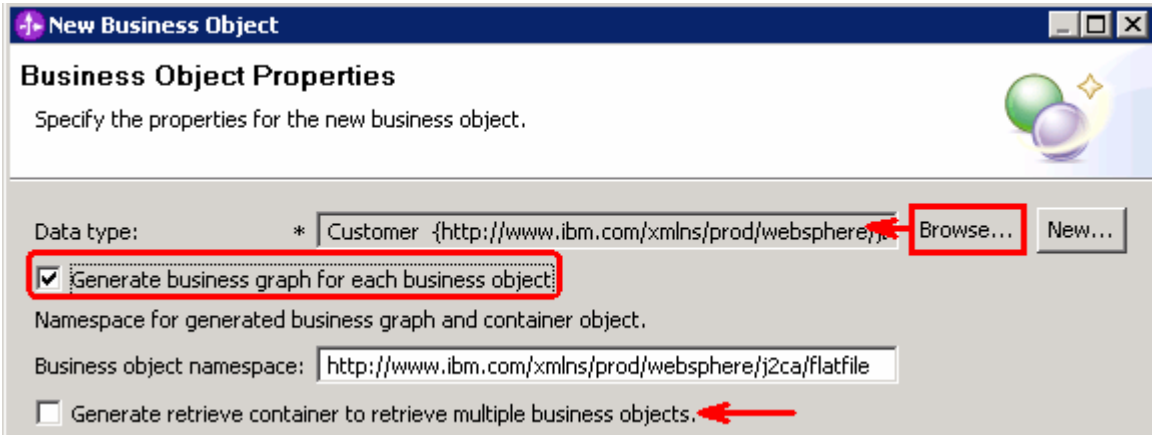
      8) Click **Finish**

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

__ e. For **Data format options**, accept the default selection **Use data format configuration 'CustomDB'** from the dropdown list
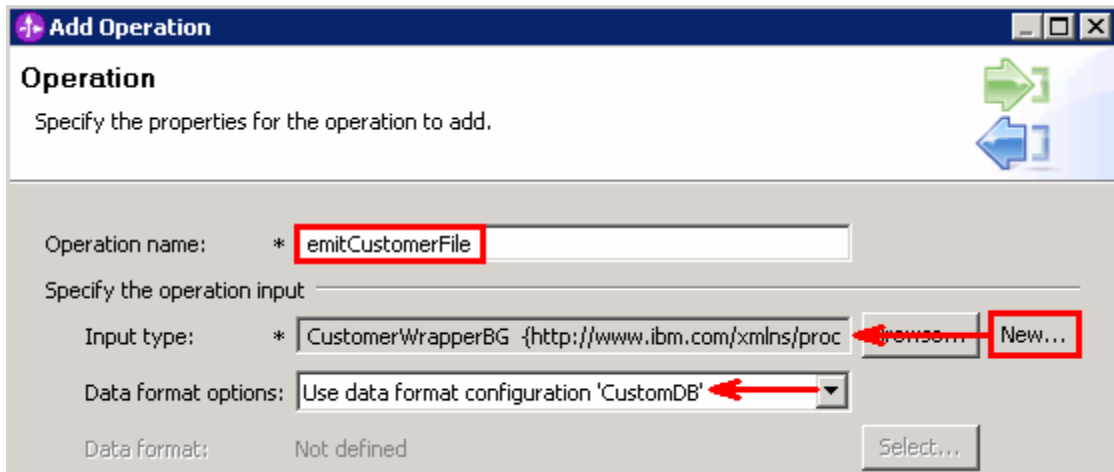
__ f. Click **Finish**

__ g. The above defined operation, emitCustomerFile, is populated in the Operations list
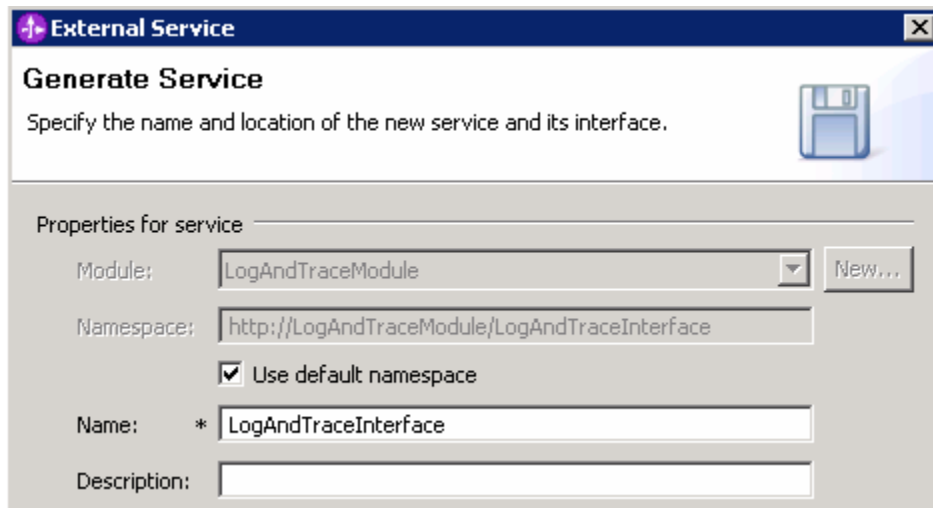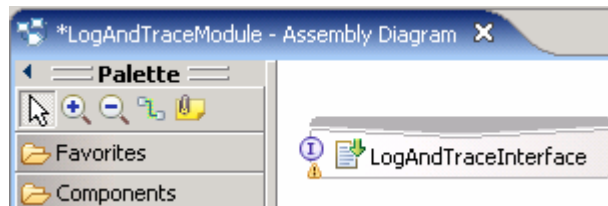

__ h. Click **Next**

____ 15. From the Generate Artifacts screen, enter these:

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

    __ b. Click **Finish**

____ 16. You will now see a new export component, **LogAndTraceInterface** in the assembly diagram of FFCustomInboundModule

    __ a. Save (**Ctrl+S)** your changes and close the assembly diagram

2.1. Add Java component

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

____ 1.   Open the assembly diagram for LogAndTraceModule, if it is already not open

    __ **a.** From the business integration view, expand **LogAndTraceModule** 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 LogAndTraceModule assembly diagram. This will place a new component, Component1 on the assembly diagram.

____ 3.   Wire the LogAndTraceInterface to the Component1

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

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

    __ c. Select **OK** for the Add Wire pop-up window:
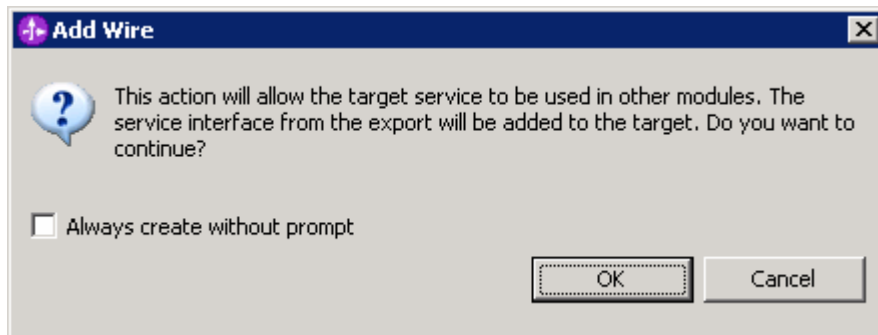
Your assembly diagram for LogAndTraceModule will look like this:
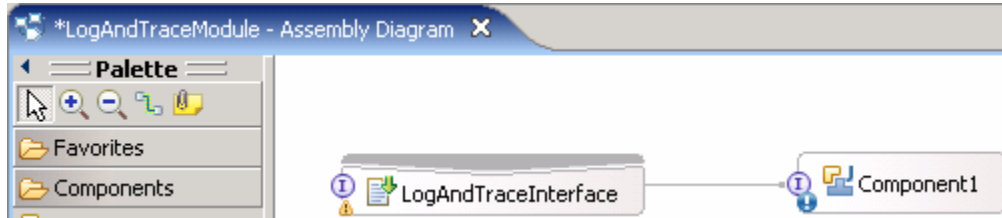
___ d. From the top of the Palette, click the **Selection Tool** icon (   ) to get back to the normal cursor mode

_____ 4.   Generate Java Implementation

___ a. Right-click **Component1** and select **Generate Implementation** from the pop-up menu

___ b. On the **Generate Implementation** panel, select default package, and click **OK**

___ c. **Component1Impl.java** is opened in Assembly editor. Scroll down to the method **emitCustomerFile(DataObject emitCustomerFileInput)** that needs to be implemented and add this code under that method:

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

System.out.println("STATE----> "+state);
```

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

___ d. Save (**Ctrl + S**) and close Component1Impl.java

_____ 5.   Save (**Ctrl + S**) and close Assembly diagram: LogAndTraceModule

Part 3: Deploy the application and enable tracing

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

____ 1.   Start the server (if not already started)

> __ a. From the Servers view, right-click the **WebSphere Process Server v6.2** and select **Start**

> __ b. Wait until the server status turns to **Started**

____ 2.   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 **LogAndTraceModuleApp** under Available projects panel and click **Add >**

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

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

____ 3.   Enable tracing

> __ a. Right-click WebSphere Process Server v6.2 and select **Administration > Run administrative console**

> __ b. From the 'Admin Console' window, enter Administrative User ID and Password and then click **Log in**
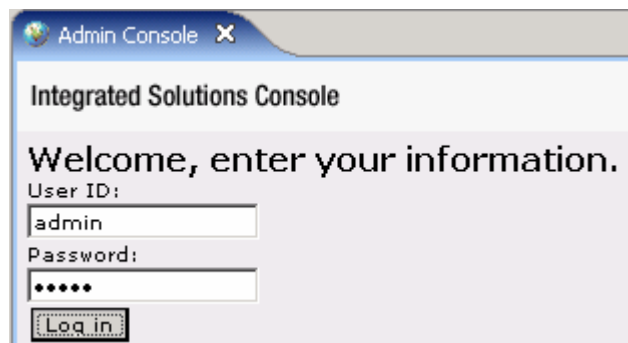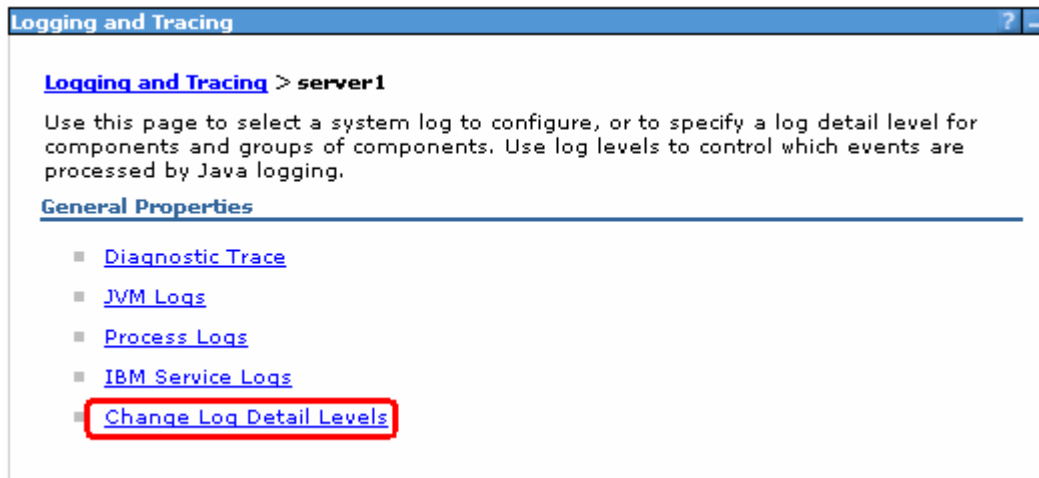

> __ c. Expand **Troubleshooting** and click **Logs and Trace**

__ d. From the Logging and Tracing screen, click **server1**
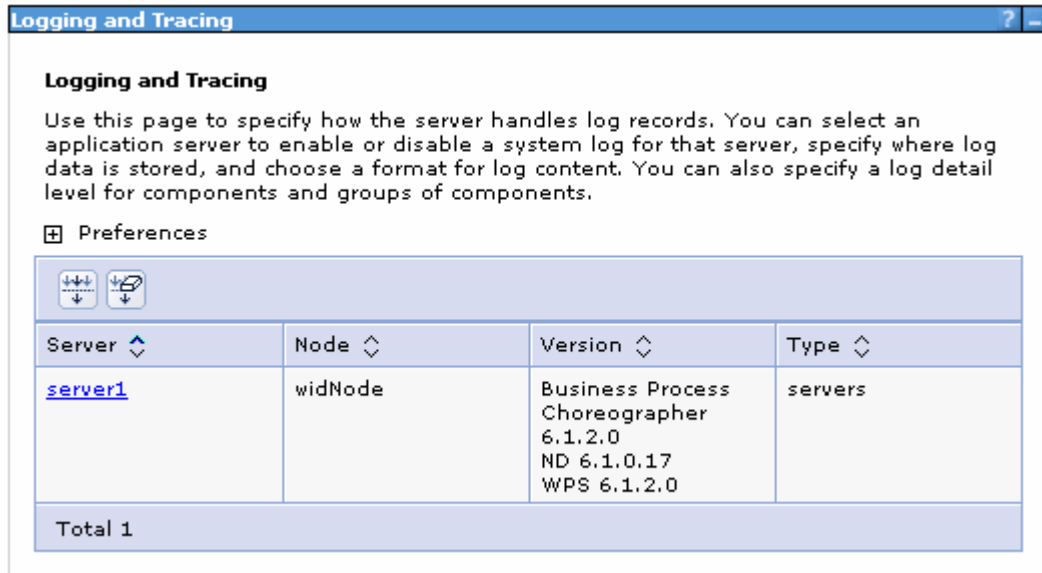
__ e. Click **Change Log Detail Levels** under General Properties

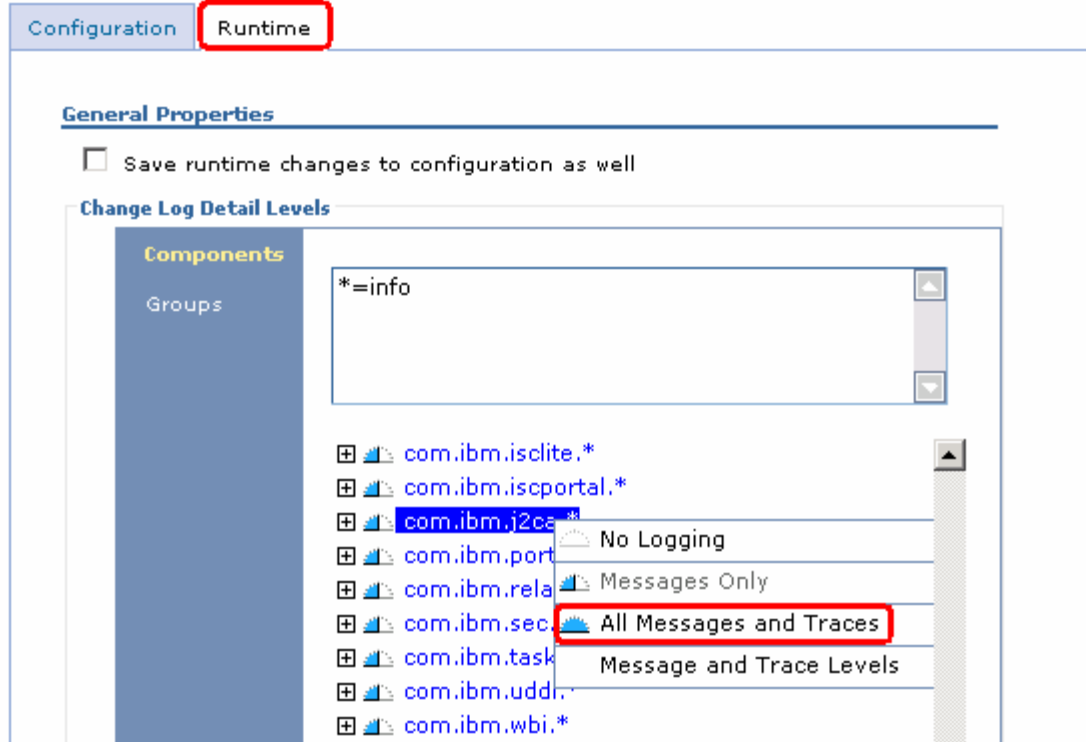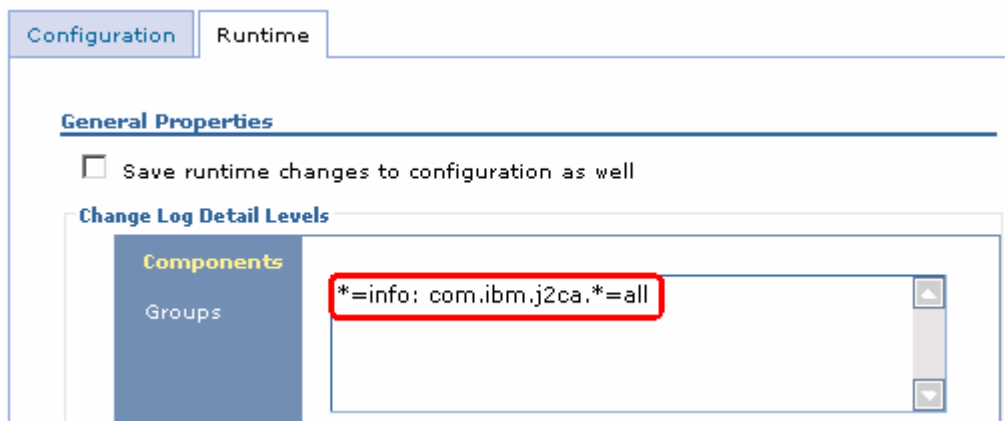__ f. Select the **Runtime** tab

WBPMv62_IEA_AdapterFFInboundLab.doc

__ g. Scroll down and click **com.ibm.j2ca.*** under Change Log Detail Levels and then select **All Messages and Traces**

__ h. You should see the below statement:


__ i. Click **OK** at the bottom

__ j. Click **Save** at the top to save all your changes

__ k. **Log Out** and close the administrative console window

Part 4: Load trace file and review confidential messages

_____ 1.   Load log/trace file contents from new **Server Logs** view

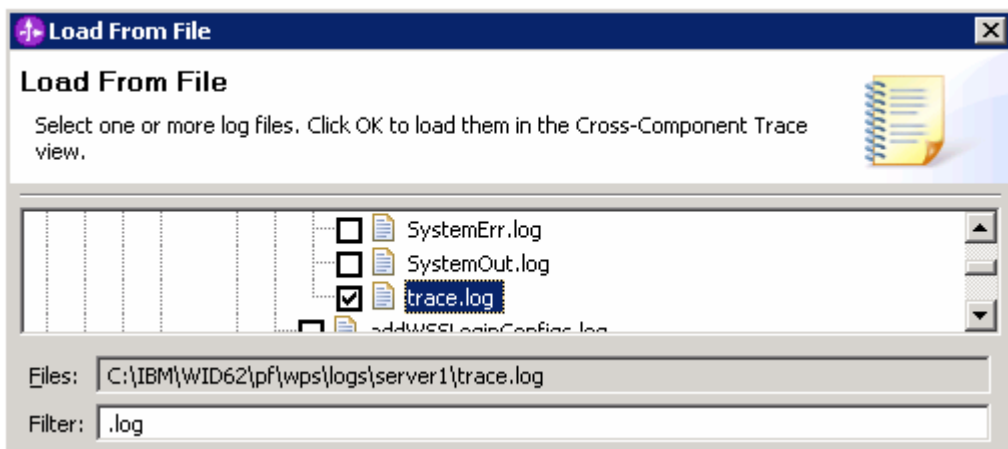__ a. Select Server Logs view located in the bottom pane

| 🔾 Build Activities | ▣ Properties | 🖹 Problems | 🗐 Server Logs ✕ | 🔧 Servers | 🗐 Asset Repositories |
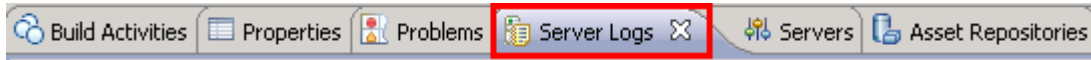
__ b. From the icons menu under Server Logs, click '**Load Server Console or Log**' icon (🗐):

__ c. Ensure that '**Load from file**' is selected and click **Browse…** next to Files

> 1) In the 'Load From File' window, browse to **<WPS_HOME>\logs\server1** and check the box for **trace.log**

Load From File

Select one or more log files. Click OK to load them in the Cross-Component Trace view.

- ☐ 🖹 SystemErr.log
- ☐ 🖹 SystemOut.log
- ☑ 🖹 trace.log
- ☐ 🖹 addWSSLoginConfigs.log

Files: C:\IBM\WID62\pf\wps\logs\server1\trace.log

Filter: .log

> 2) Click **OK**

__ d. You are back to 'Load Server Console or Log' window and the above selected trace file is populated here:

Load Server Console or Log

Load a server console or log files

Select a server console or log files to load into the Server Logs view.

◉ Load from file

Files: C:\QCert\WID62\pf\wps\logs\server1\trace.log   Browse…

○ Load from server console

Server: WebSphere Process Server v6.2 at localhost

○ Load from server log directory

Server: WebSphere Process Server v6.2 at localhost
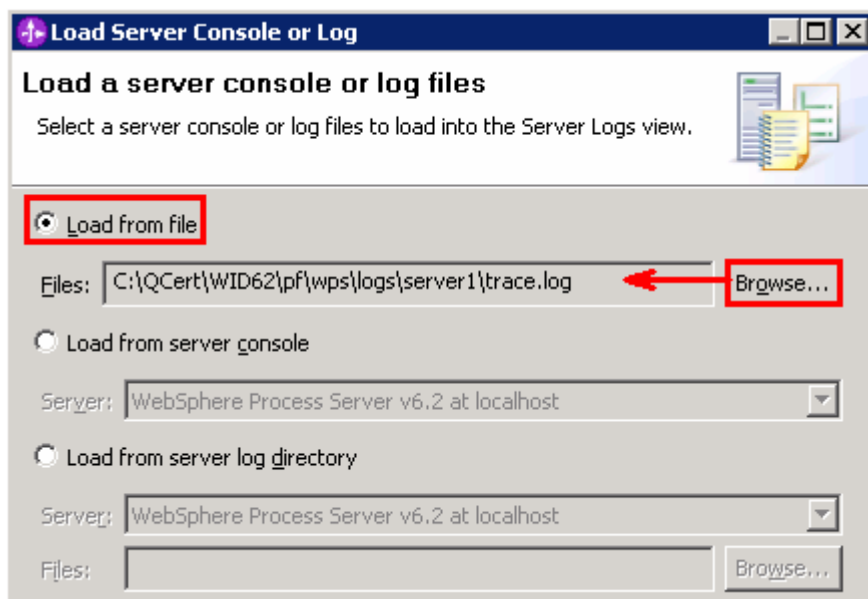
Files:

__ e. Click **Finish**

_____ 2.   Provide the input file:

Note: For your convenience, a sample test file, **SingleCustomerBO.xml**, which contains one customer business object, is placed in **<FFFILES>.**

__ a. Copy the **sample.txt** file from <FFFILES> to **<EVENT_DIR>/subdir**. The adapter will poll the copied file from the sub directory under event directory and will transfer it to the archive directory

__ b. Because you have placed a file that contains customer business object, it will pass through the **emitCustomerFile** method and you should see this message in the console (or SystemOut.log):

```
SystemOut     O ************ENDPOINT emitCustomer**************
SystemOut     O FILENAME : SingleCustomerBO.xml
SystemOut     O NAME-----> IBM
SystemOut     O ADDRESS--> 11501 Burnet Rd
SystemOut     O CITY-----> Austin
SystemOut     O STATE----> TX
```

__ c. To verify your test results, check the **<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
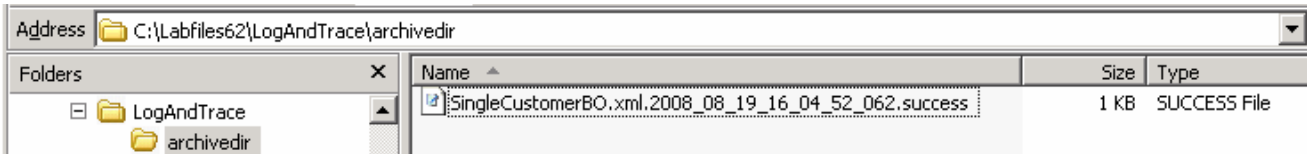
__ d. The log/trace file should be loaded and displayed in the **Server Logs** view as shown below. The log records show all the messages of trace/log file. Note that some messages have '**FFRA001**' in their description and this is because you specified the Adapter ID as 001 while configuring the adapter:
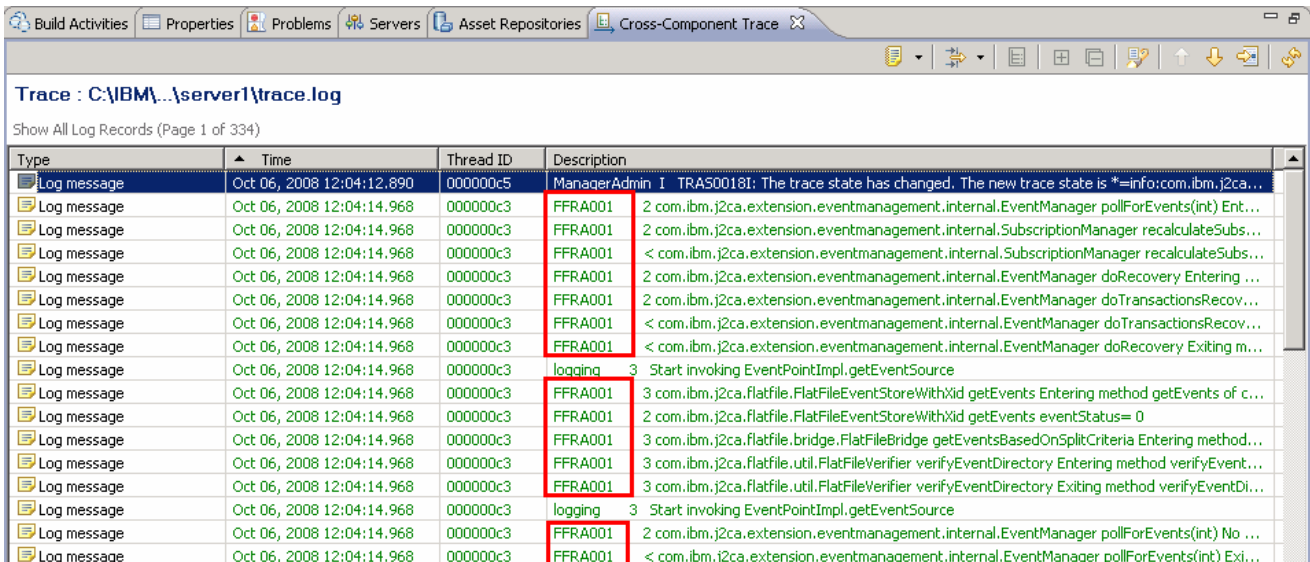
_____ 3.   Review trace file for confidential messages:

- **Flat File**: When confidential tracing is enabled, BO data for both inbound and outbound is replaced by XXXXX. For outbound (create, append, and overwrite) the BO is not masked since the BO content is in the request and at the databinding, it is not possible to differentiate if confidential tracing was enabled or not. Things that are not considered confidential for this purpose, includes anything that is internal to the adapter or runtime, and did not originate inside the EIS. This includes, but is not limited to: Transaction IDs, Event IDs, BO schemas, and Call sequences.

- **FTP**: When confidential tracing is enabled: BO data for both inbound and outbound is replaced by XXXXX. Username is replaced by XXXXX. Password will always be written to the trace file as ***** irrespective of whether confidential tracing is enabled or not. FTP URL is traced to the trace file. This is part of each trace message (for example: ftp://localhost/eventDir). This is required because when you are using multiple Activation Spec (AS) or MCFs with each AS polling one dir on the same FTP server, it is easier to identify to which AS the traces belong to. For outbound (create, append, and overwrite) the BO is not masked since the BO content is in the request and at the databinding it is not possible to differentiate if confidential tracing was enabled or not. Things that are not considered confidential for this purpose, includes anything that is internal to the adapter or runtime, and did not originate inside the EIS. This includes, but is not limited to: Transaction IDs Event IDs BO Schemas Call sequences

- **EMail**: When confidential tracing is enabled: BO data for both inbound and outbound is replaced by XXXXX. Username is replaced by XXXXX. Password will always be written to the trace file as ***** irrespective of whether confidential tracing is enabled or not. Things that are not considered confidential for this purpose, includes anything that is internal to the adapter or runtime, and did not originate inside the EIS. This includes, but is not limited to: Transaction IDs Event IDs BO Schemas Call sequences

___ a. Open your windows explorer and open <WPS_HOME>\logs\server1\trace.log using note pad

___ b. You should see 'XXXX' replaced for all the confidential data:

```
[10/3/08 11:58:04:015 CDT] 00000018 FFRA001        1 com.ibm.j2ca.flatfile.FlatFileResourceAdapter
traceActivationSpecProperties EP_UserName = XXXX
[10/3/08 11:58:04:015 CDT] 00000018 FFRA001        1 com.ibm.j2ca.flatfile.FlatFileResourceAdapter
traceActivationSpecProperties EP_Password = XXXX
[10/3/08 11:58:04:015 CDT] 00000018 FFRA001        1 com.ibm.j2ca.flatfile.FlatFileResourceAdapter
traceActivationSpecProperties EP_TableName = XXXX
[10/3/08 11:58:04:015 CDT] 00000018 FFRA001        1 com.ibm.j2ca.flatfile.FlatFileResourceAdapter
traceActivationSpecProperties EP_SchemaName = XXXX
[10/3/08 11:58:04:015 CDT] 00000018 FFRA001        1 com.ibm.j2ca.flatfile.FlatFileResourceAdapter
traceActivationSpecProperties EP_DataSource_JNDIName = XXXX
[10/3/08 11:58:04:015 CDT] 00000018 FFRA001        < com.ibm.j2ca.flatfile.FlatFileResourceAdapter
```

```
[10/3/08 11:52:05:812 CDT] 000000e0 FFRA001        3 com.ibm.j2ca.flatfile.util.FlatFileUtil
getTotalBOsFromEventId Total number of BOs is 1
[10/3/08 11:52:05:812 CDT] 000000e0 FFRA001        3
com.ibm.j2ca.flatfile.FlatFileEventStorewithXid getRecordForEvent BO Content : XXXXX
[10/3/08 11:52:05:812 CDT] 000000e0 FFRA001        3
com.ibm.j2ca.flatfile.FlatFileEventStorewithXid getRecordForEvent DirectoryPath set in the
record object is C:\Labfiles62\LogAndTrace\eventdir
```

_____ 4.   Restore the Sever Configuration

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

___ b. Select **LogAndTraceModuleApp** 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 started with importing the Flat File Adapter RAR file into your WebSphere Integration Developer new workspace. 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 Function selector which, after deploying onto the server, will generate Business Objects and other artifacts. Then you continued to complete building the application by adding a Java component and implementation.

Next, you deployed the adapter application onto WebSphere Process Server and enable tracing in WebSphere Process Server administrative console and tested the application for sample input file containing single customer business object.

You loaded the trace file into the Server Logs view, and reviewed the messages matching Adapter ID. Finally, you reviewed the trace file contents for the 'confidential' messages replaced with 'XXX'.

Task: Adding remote server to WebSphere Integration Developer test environment

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

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

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

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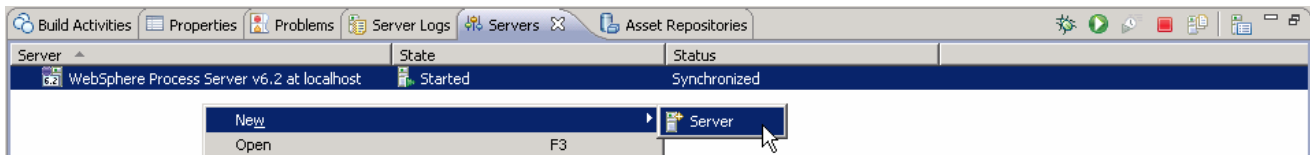
    __ c. In the New Server dialog, specify the remote server's host name, **<HOSTNAME>**.

    __ d. Ensure that the appropriate server type, '**WebSphere Process Server v6.2**' or '**WebSphere ESB Server v6.2**', is highlighted in the server type list
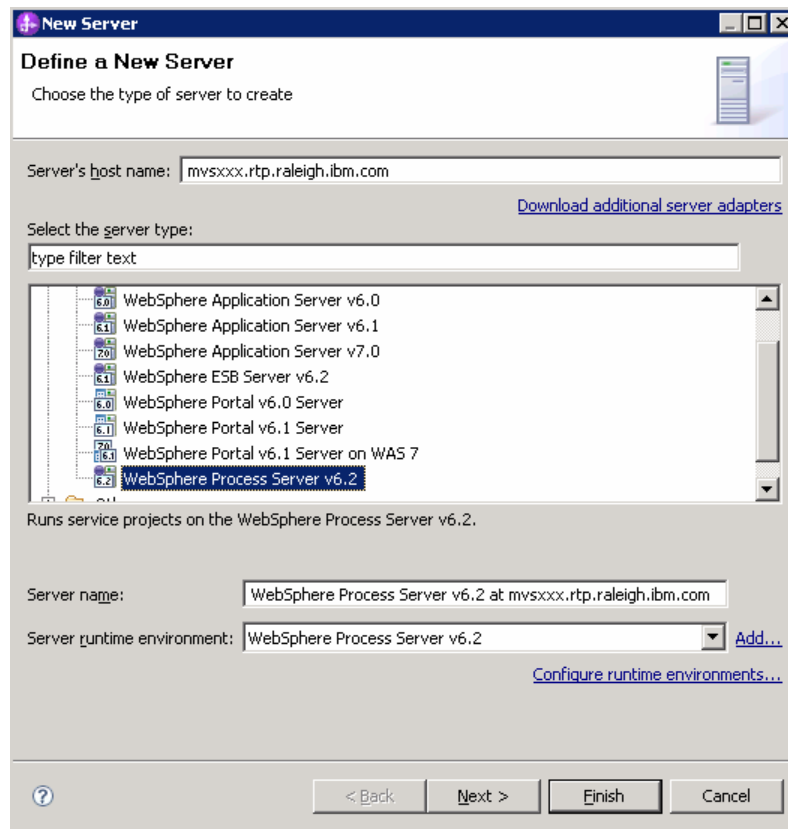

    __ e. Click **Next.**

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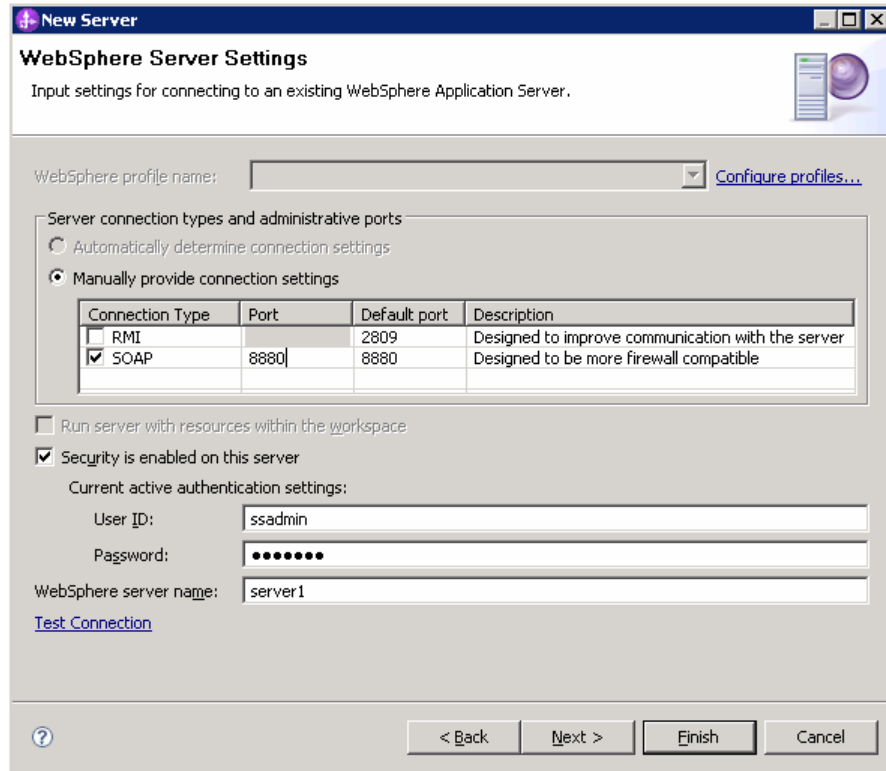
__ i. Click **Finish**.

__ j. The new server should be seen in the Server view.
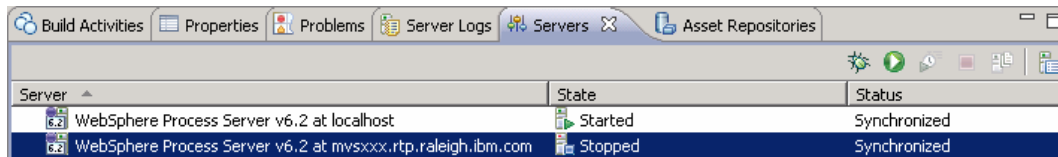
____ 2. Start the remote server if it is not already started. WebSphere Integration Developer does not support starting remote servers from the Server view.

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

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

User ID : **<USERID>**

Password : **<PASSWORD>**

WBPMv62_IEA_AdapterFFInboundLab.doc

__ 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
```

IBM WEBSPHERE ADAPTER 6.2 – LAB EXERCISE

This page is left intentionally blank.

WBPMv62_IEA_AdapterFFInboundLab.doc