IBM WEBSPHERE ADAPTER 7.0 – LAB EXERCISE

E-mail Adapter inbound lab

What this exercise is about

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

Lab requirements

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

- WebSphere Integration Developer V7.0 installed and updated with latest fixes

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

- Install and configure the hMailServer (your e-mail server for this lab) and Mozilla Thunderbird (your e-mail client for this lab) by following the instructions in the lab: **Install and configure e-mail server and client**

- 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 E-mail adapter RAR file into WebSphere Integration Developer

- Use External Service wizard to configure Activation Spec Properties, Resource Adapter Properties to generate Business Objects and other artifacts and configure Function Selector, Data Binding and Data Handlers

- Deploy the adapter application onto WebSphere Process Server

- Test the deployed application using WebSphere Process Server test environment for pass-through, non pass-through, and Fixed structure e-mail scenarios

- Restore the server configuration

Introduction
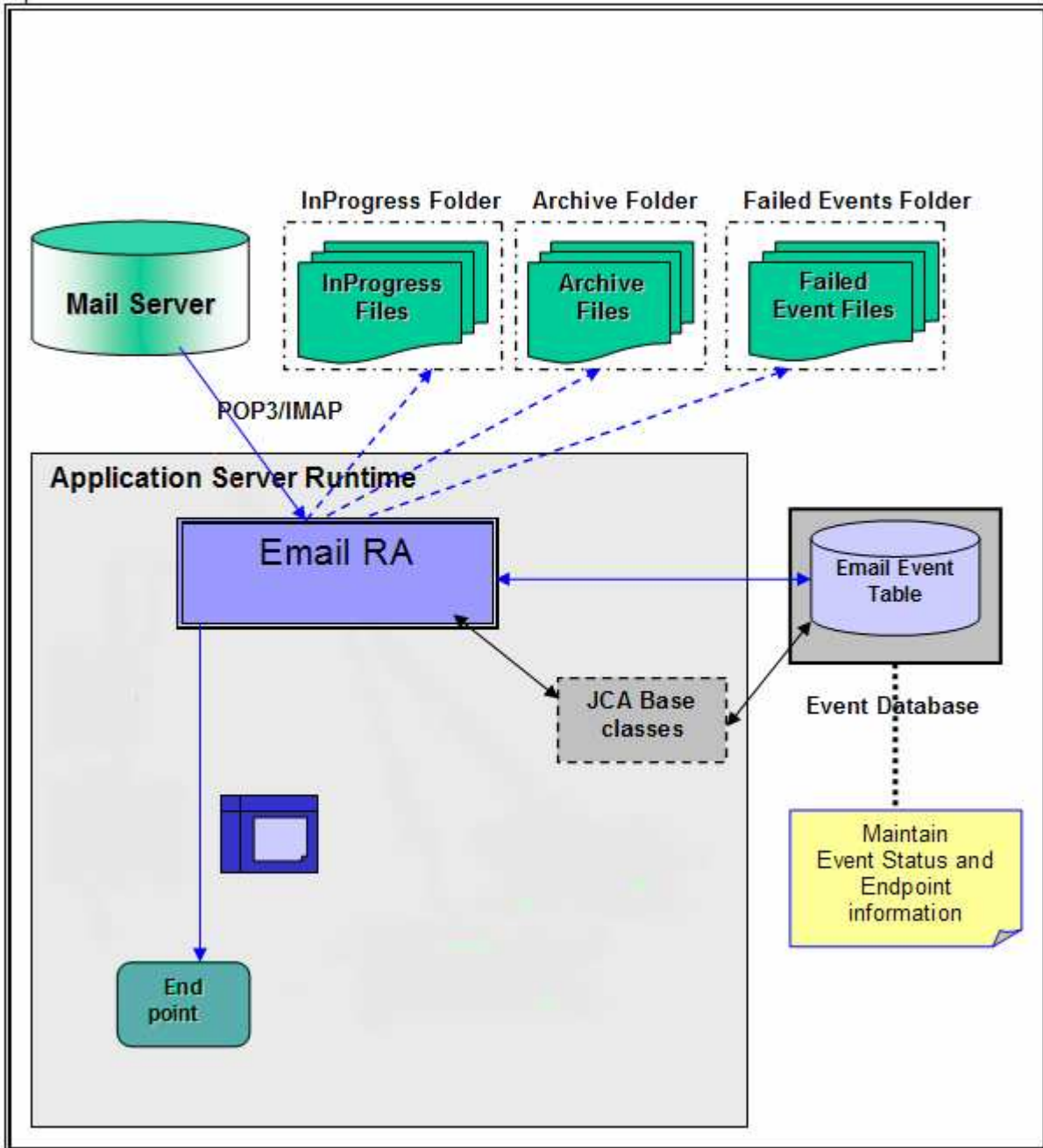
The E-mail RA interacts with the mail server using JavaMail API that handles the underlying SMTP and IMAP/POP3 protocols. The adapter supplies the necessary parameters required by JavaMail to communicate to the mail server.

The E-mail RA polls on the PollFolders for that user on the specified mail server at regular intervals, and picks up all the unread mails as events. You can specify multiple e-mail folders for a user's mail account within the PollFolders property. The multiple e-mail folders are then polled sequentially in the same poll cycle. Optionally, you can specify search criteria to pick up event mails and when search criteria is specified, all unread mails meeting the search criteria, are picked up for polling. When no search criteria are specified, all unread mails are picked up by default.

WBPMv70_AdapterEMailInbounLab.doc

IBM WEBSPHERE ADAPTER 7.0 – LAB EXERCISE

The E-mail RA writes all the polled mails to the InProgress file -folder, as files saved in the RFC822 format. The name of the RFC822 format file is the Message_ID of the polled E-mail. The E-mail RA transforms the mail to an E-mail BO and delivers the same to configured endpoint. The event management framework takes care of delivering the event only once to the endpoint.

The contents of each E-mail are parsed into an EmailRecord object. The EmailRecord is routed through the Function Selector to the E-mail-specific data-binding. The E-mail-specific data-binding is aware of the structure of the E-mail BO. It will parse the EmailRecord structure, and route the content of the mail and the attachments to the mime-specific data-bindings, according to their mime-types. It then receives back the parsed structures and combines them back into the EmailBO, and sends it out to the configured endpoint.

The E-mail Event Database is part of the Event Persistence Framework. The event persistence feature ensures that the event is delivered to the end point once and only once.

Once the mail is read and stored in the InProgress folder, it is deleted from the mail server (per the POP3 specifications). Once a mail is processed, the mail is explicitly marked as deleted on the Mail Server (per the IMAP specifications). The mail is archived or deleted from the InProgress folder. The archiving is based on the values for the properties ArchiveFolder and the FailedEventsFolder.

If the ArchiveFolder is specified, the successfully processed mails are moved into the ArchiveFolder from the InProgressFolder. If blank, then the successfully processed mails are just deleted from the InProgressFolder.

If the FailedEventsFolder is specified, the mails that failed processing are moved into the FailedEventsFolder from the InProgressFolder. If blank, then the failed mails are just deleted from the InProgressFolder.

Exercise instructions

Some instructions in this lab might be 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_70	
<EMAILADAPTER_HOME>	<WID_HOME>\ResourceAdapters\Email_7.0.0.0\deploy	
<LAB_FILES>	C:\Labfiles70	/tmp/Labfiles70
<WORKSPACE>	<LAB_FILES>\EmailInbound\workspace	
<EMAILFILES>	<LAB_FILES>\EmailFiles	
<STAGING>	<LAB_FILES>\EmailInbound\StagingDir	
<ARCHIVE>	<LAB_FILES>\EMailInbound\ArchiveDir	
<FAILED>	<LAB_FILES>\EMailInbound\FailedEventDir	
<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 following table 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**".

WBPMv70_AdapterEMailInbounLab.doc

Part 1: Initialize the workspace and prepare for the lab

This part of the lab, you will start the WebSphere Integration Developer V7.0 with a new workspace and create required data source and database 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. 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
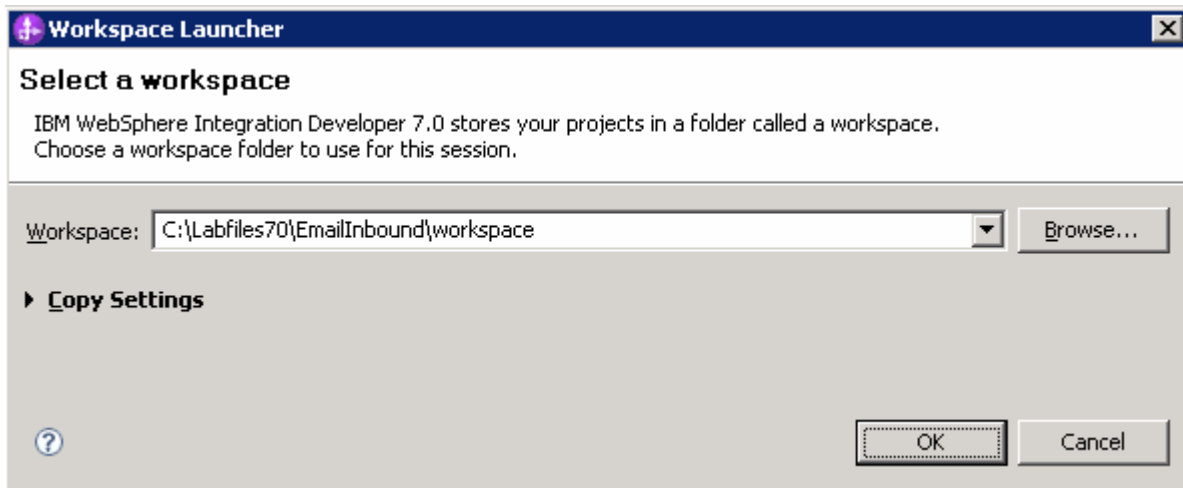
____ 3. Click the [icon] button on the right corner to close the Welcome page and proceed with the workbench

____ 4. Follow the instructions in the "**Configure data source**" task with these inputs and create the data source and data base required for this lab:

    __ a. Data source name: **Email**

    __ b. JNDI name: **jdbc/Email**

    __ c. Database name: **EmailDB**

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

Name	Value
EMAIL_STAGING	C:\Labfiles70\EmailInbound\StagingDir
EMAIL_ARCHIVE	C:\Labfiles70\EmailInbound\ArchiveDir
EMAIL_FAILED	C:\Labfiles70\EmailInbound\FailedEventDir

Note: In E-mail adapter, you have to provide directory values (Staging directory, Archive directory, Failed event 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. From WebSphere Process Server V6.1, 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.

Part 2: Pass through scenario

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

2.1. Configure pass through using the external service wizard

In this part of the lab you will use this new External Service feature to create and configure the Function Selector, Data Binding and other required artifacts to test the inbound pass through scenario

____ 1.   Create the module: EmailPSInboundModule

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

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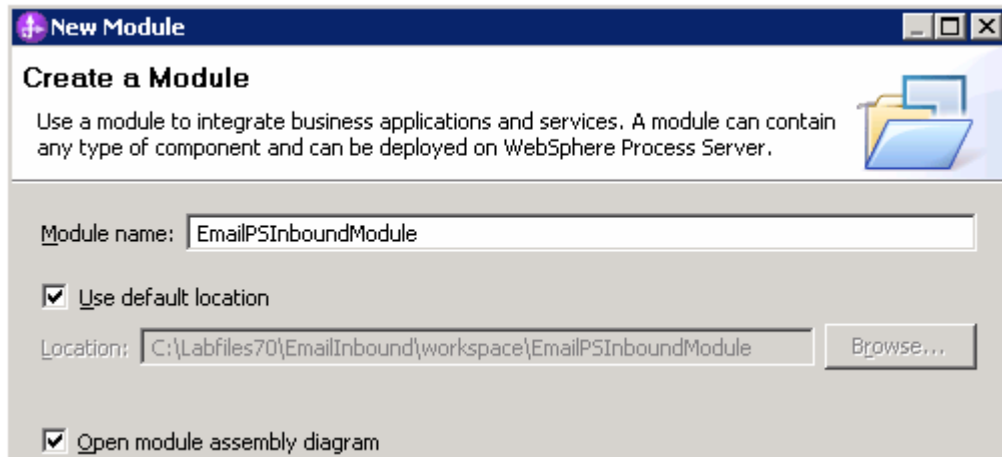

__ c. Ensure that the box next to **Open module assembly diagram** is checked and then click **Finish**

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

____ 2.   To start External Service from the Palette:

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

__ b. Under Inbound Adapters, click the **Email** and then click the empty canvas of the assembly diagram. The New Flat File Service wizard is opened
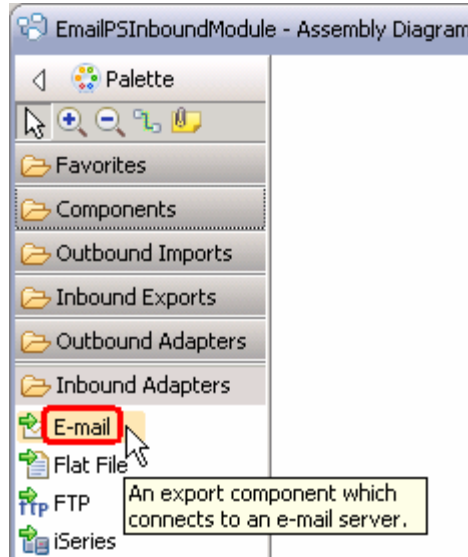
Note: You can also start the External Service from the **File menu** option:

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 > EMail** and click **Next**

____ 3. On the Select an Adapter screen, select **IBM WebSphere Adapter for Email (IBM : 7.0.0.0)** and click **Next**
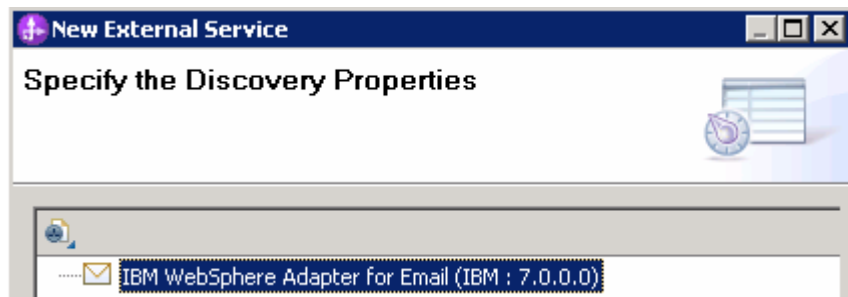
____ 4. Adapter Import screen:

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 **<EMAILADAPTER_HOME>**.

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

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

WBPMv70_AdapterEMailInbounLab.doc

__ c. For Target server, ensure that **WebSphere Process Server v7.0** is selected
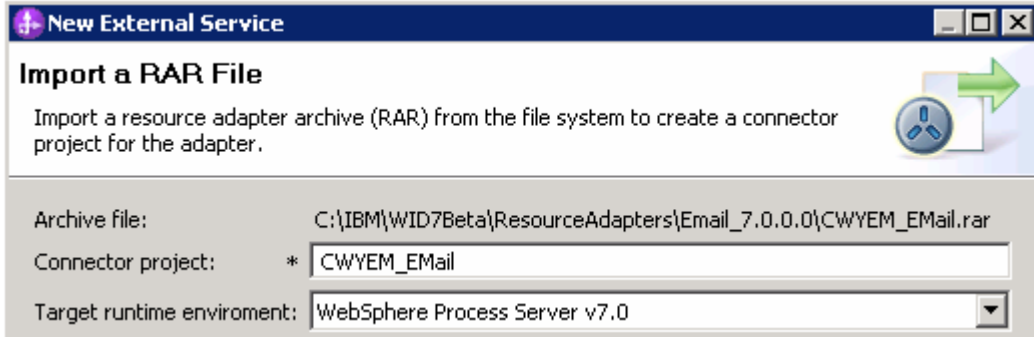
__ d. Click **Next**

Note: The resource adapter archive file is imported and a new connector project, **CWYEM_EMail**, 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.

____ 5.   Service Configuration Properties:

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

__ b. Enter E-mail system connection information:

1) For Staging directory, enter **${EMAIL_STAGING}**

Note: WebSphere Environment Variable '**EMAIL_STAGING**' should already be defined using the instructions provided in **Part 1, Step 5** of this lab.

2) Host name: **<EmailServer_HostName>** (or IP Address of the machine that has E-mail Server), for Ex: **localhost**

3) Port number: **110** (default, you should change it to the correct port number if your E-mail server is running on a different port)

4) Protocol: accept the default **POP3**

5) Not that the **Poll folder** is by default selected as **Inbox** and is not available

IBM WEBSPHERE ADAPTER 7.0 – LAB EXERCISE

__ 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 need only some of these properties.

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

__ e. Event delivery configuration:

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

__ f. Event persistence configuration:

    1) Ensure that the **Auto create event table** is **checked**

    2) Event recovery table name: **EmailPSTable**

    3) Event recovery data source (JNDI) name: **jdbc/Email**

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

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".

Advanced properties:

__ g. Archive folder: **${EMAIL_ARCHIVE}** (All the processed events are archived in this folder)

__ h. Failed event folder: **${EMAIL_FAILED}** (All the failed events are archived in this folder)

Note: WebSphere Environment Variables '**EMAIL_ARCHIVE**' and '**EMAIL_FAILED**' should already be defined using the instructions provided in **Part 1, Step 5** of this lab.

__ i. Check the box next to **Emit individual business object from a multi-part e-mail**

Note: If you select **Emit individual business object from a multi-part e-mail**, the split for individual BOs is done at the RA level and each of the BO is emitted as generic E-mail. Each part of a multipart E-mail (for ex: E-mail containing body, attachments, and file references) is considered as an individual BO and emitted using E-mail Wrapper, where **each part's content is set in the mail content attribute**.

New: In V6.2, you can also send binary images either in the e-mail (including image or embedding image) or as an attachment, which are treated as different business objects.

___ j. Logging and tracing:

Note: Refer to '**Log and confidential trace lab**' for more details on this **new** confidential trace feature.

____ 6. For this lab, you are going to use security properties from the activation specification. So, select the radio button next to **Using security properties from the activation specification**

___ a. User name: **username using which you connect to your E-mail server**

Note: Use Emailuser1@aimcp101.austin.ibm.com, since it is configured as SMTP authentication user, if you followed the instructions in 'Install and configure e-mail server and client lab' to install and define users for hMailServer

---

___ b. Password: **password for the user to connect to your E-mail server**

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

The E-mail Function Selector is used by the E-mail RA to get back the SCA export function name corresponding to the event type sent out by the adapter.

____ 7. Under Service properties, for **Function selector options**, select **Use default function selector 'EmailFunctionSelector'** from the drop down list

---

Note: Follow the steps in **Appendix: Define Function selector** to manually define a function selector.

____ 8. 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 **EmailWrapperDataBinding**

__ d. Click **Next**

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

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

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

1) For the **Name**, enter **EmailPSDB**
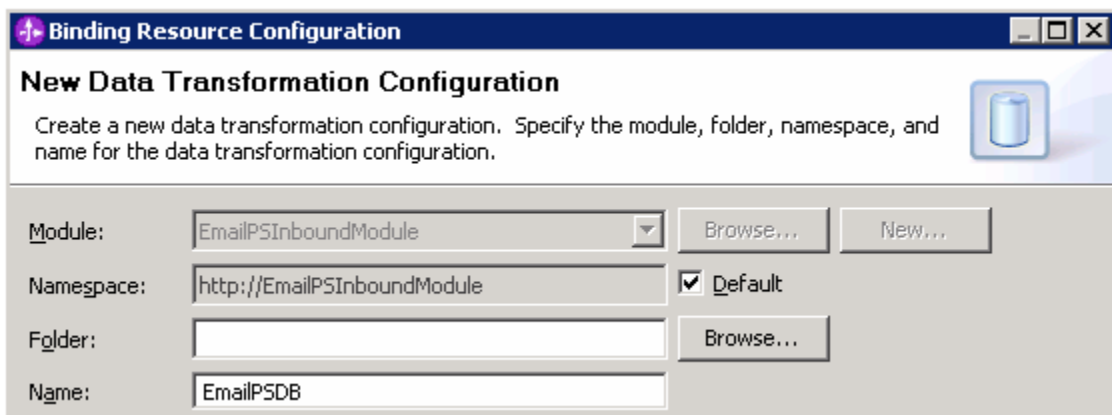
2) Click **Finish**

__ g. Now the **EmailPSDB** should be displayed for Data binding configuration
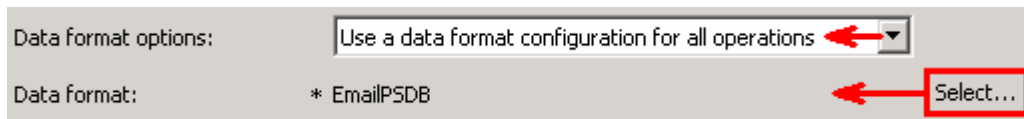
____ 9.   Check the box next to **Change logging properties for wizard** to view the output location of the log
file and the logging level. You can change the logging level using the drop down menu.

__ a. Click **Next** from Connection Properties screen

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

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

- Generic e-mail business object

- Generic e-mail business object with business graph

- User defined type

In this part of the lab, you will use the first option, **Generic e-mail business object**.

Create emitEmailPS Operation:

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

__ a. Add Operation window is opened. Select **Generic e-mail business object** for the Data type and click **Next**

> Operation properties
>
> The data type for the operation input: | Generic e-mail business object | ▼ |
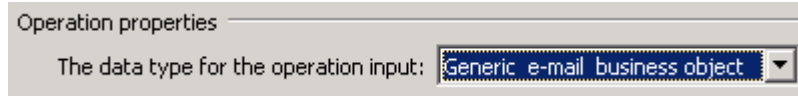
You are now back to Operation window and because you have chosen the 'Generic e-mail business object', the Input type is populated as **Email**

> Specify the operation input
>
> Input type: → | Email {http://www.ibm.com/xmlns/prod/websphere/j2ca/email/email} | Browse… | New… |
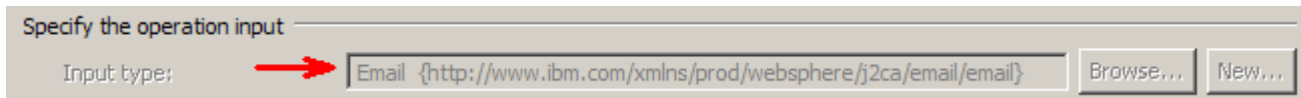
__ b. For Operation name, enter any name, for Ex: **emitEmailPS**

__ c. Note that the default Input type, **Email**

__ d. Select Data format:

> 1) Click **Select…** next to **Data format**. A Binding Resource Configuration window is opened
>
> 2) Ensure that the radio button for 'Use existing data format transformation from the list' and then select **EmailWrapperDataBinding > EmailPSDB**
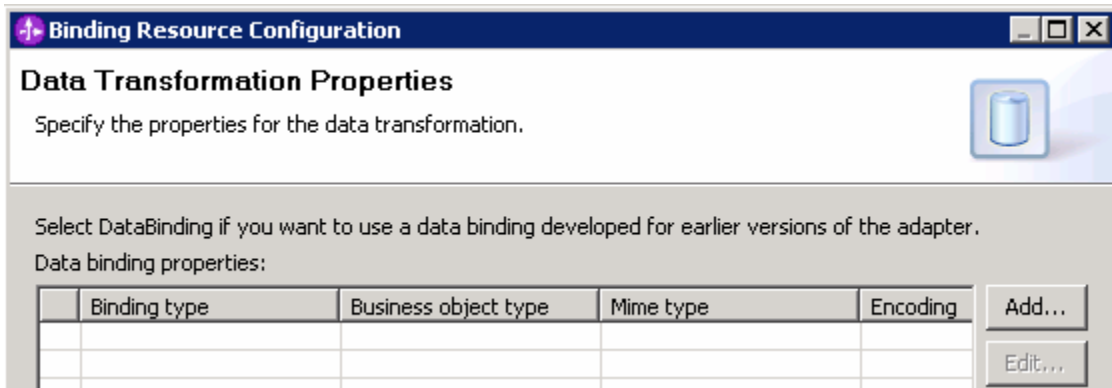
> **Data Binding Configuration**   _ □ ✕
>
> **Select Data Format Transformation**
>
> Select a data format transformation entry from the list. If you want to use your own custom data transformation then select the second radio button to add your custom transformation.
>
> ⊙ **Use existing data format transformation from the list**
>
> ⊟ 📇 EmailWrapperDataBinding
>   └─ 🔧 EmailPSDB
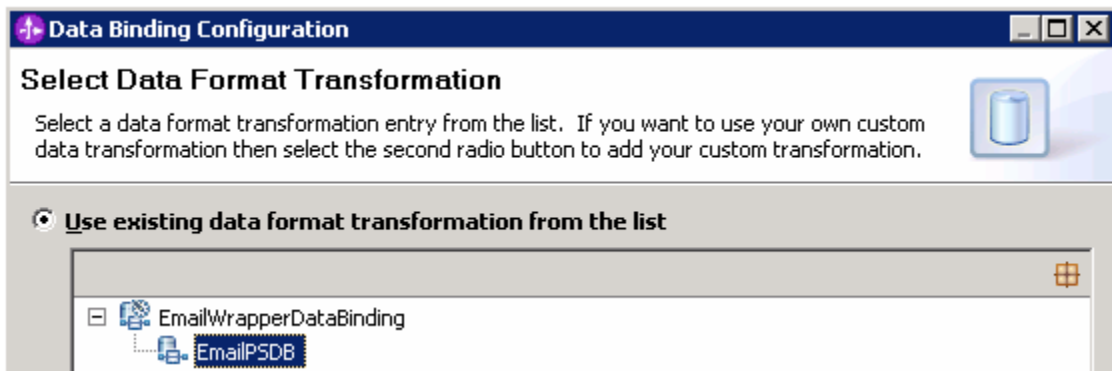
> 3) Click **Finish**

__ e. Now the **EmailPSDB** is displayed for Data format in the Add Operation window:

> Operation name: * | emitEmailPS |
>
> Specify the operation input
>
> Input type: | Email {http://www.ibm.com/xmlns/prod/websphere/j2c | Browse… | New… |
>
> Data format: * EmailPSDB   ← | Select… |
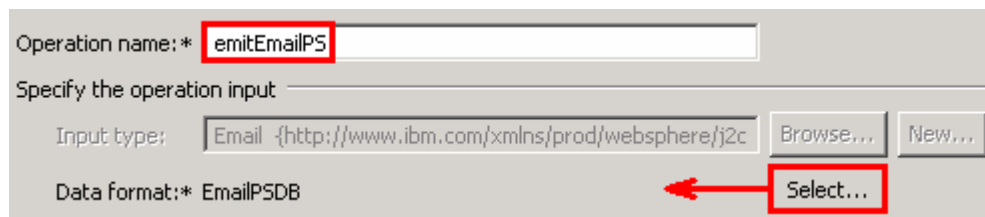
__ f. Click **Finish**

____ 11. The defined operation, **emitEmailPS**, is populated under Operations list

Operations:

emitEmailPS ({http://www.ibm.com/xmlns/prod/websphere/j2ca/email/email}Email) : void [Add...]

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

____ 12. From Generate Artifacts screen:

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

Properties for service

Module: EmailPSInboundModule   [▼] [New...]

Namespace: http://EmailPSInboundModule/EmailPSInboundExport

☑ Use default namespace

Name: * EmailPSInboundExport

Description:
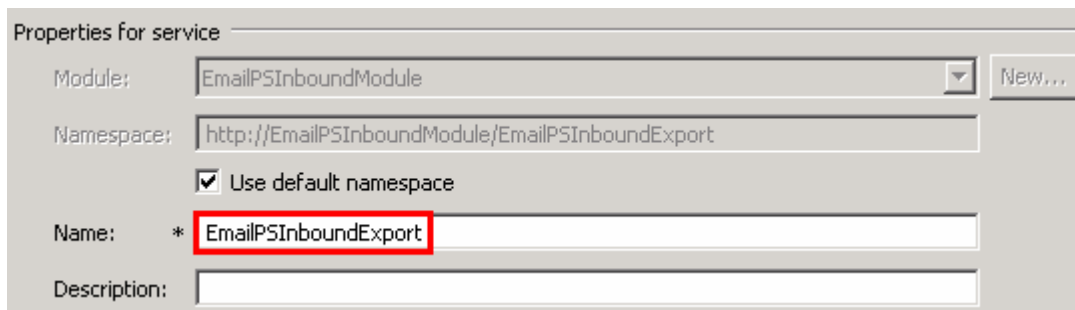
__ b. Click **Finish**

____ 13. The Assembly diagram for EmailPSInboundModule is opened with an export component, EmailPSInboundExport:
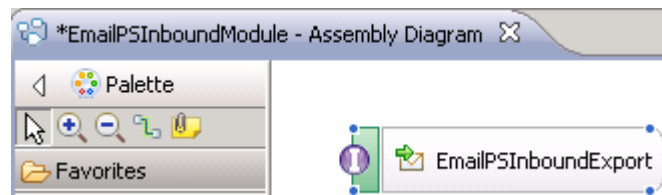
*EmailPSInboundModule - Assembly Diagram ☒

◁ Palette

Favorites

EmailPSInboundExport

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

2.2. Add Java component

In this part of the lab, you will add a Java component and then wire the component to the existing export interface. Then you will continue to add a simple print Java code to the added Java component.

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

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

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

    __ a. From the **Palette**, click **Components** to expand it
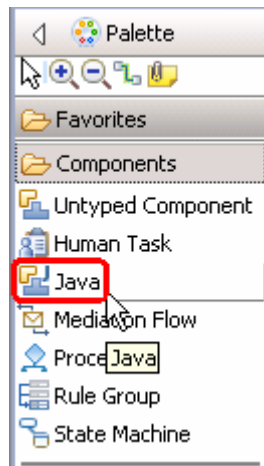
    __ b. Click **Java** and then click the empty space of EmailPSInboundModule assembly diagram. This will place a new component, **Component1** on the assembly diagram.

____ 3.   Wire the EmailPSInboundExport to the Component1

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

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

    __ c. Select **OK** for the Add Wire pop-up window:
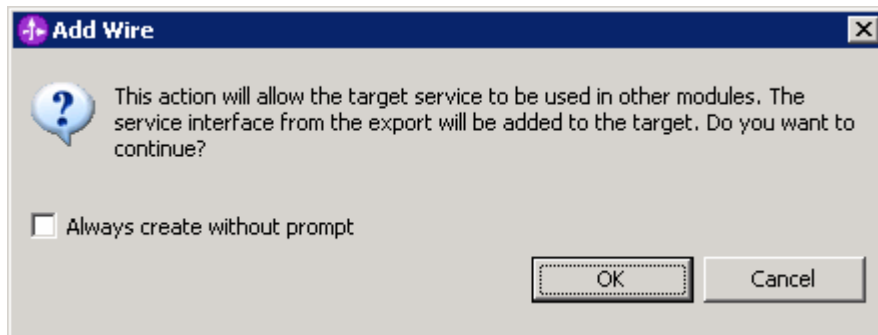
    __ d. From the top of the Palette, click the **Selection Tool** icon ( ) to get back to the normal cursor mode

IBM WEBSPHERE ADAPTER 7.0 – LAB EXERCISE

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

Your assembly diagram for EmailPSInboundModule 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 **emitEmailPS** that needs to be implemented and add this code under that method:

```
System.out.println("*********Reached Endpoint: EmailPS********");

DataObject unstructured =
emitEmailPSInput.getDataObject("mailContent");

System.out.println("**********Email Pass through content**********");

String astext = unstructured.getString("AsText");
System.out.println("As Text content-----> "+astext);
```

Note: The code is also available at **<EMAILFILES>\ PSInboundJavaCode.txt**

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

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

2.3.   Test pass through scenario

In this part of the lab, you will use the WebSphere Process Server Test Environment to test the SCA application Inbound processing for the pass through scenario. Before starting this testing part, you should already have completed the **Install and configure e-mail server and client lab** successfully to install and configure the hMailServer and Mozilla Thunderbird.

____ 1.   Start your e-mail server (if not started already)

    __ a. Select **Start > hMailServer >hMailServerAdministrator**

    __ b. From the hMailServer Administrator – Connect window, ensure that **localhost** is selected and click **Connect**
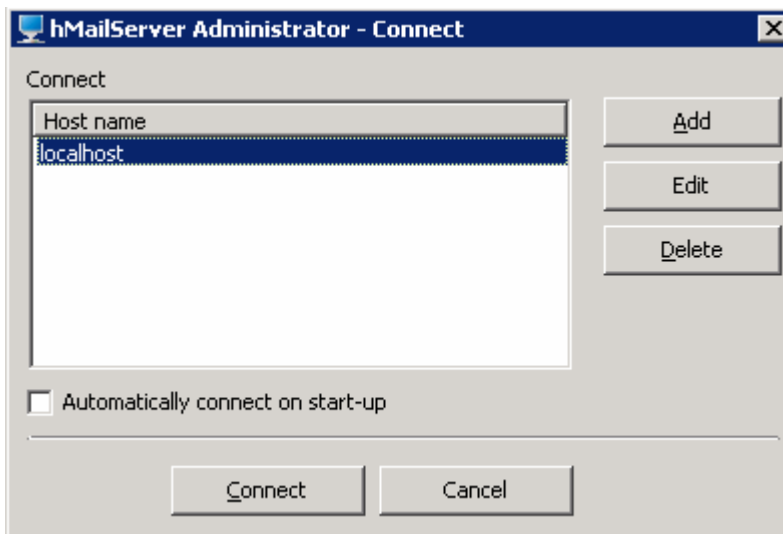
    __ c. Enter administration password provided while installing the hMailServer during the **Install and configure e-mail server and client lab**
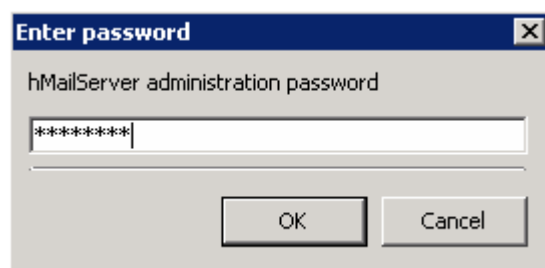
__ d. hMailServer Administrator window is opened and the Current status should show **Running**
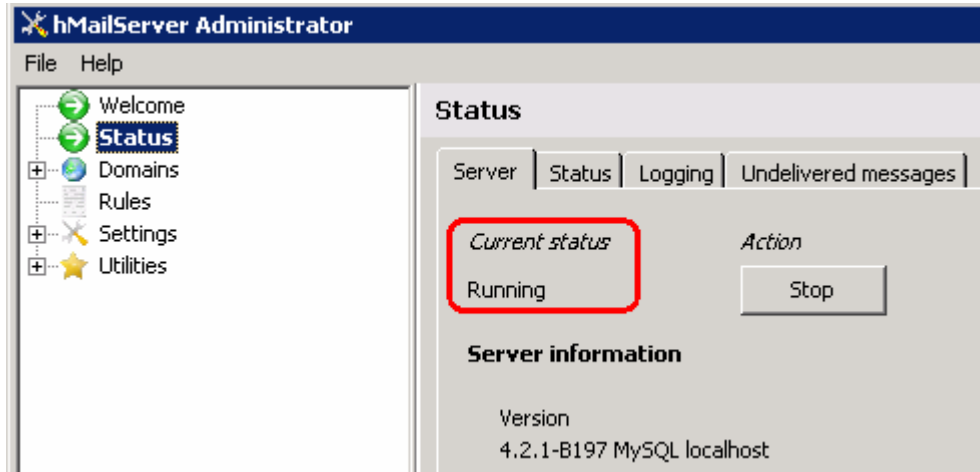
____ 2.   Add the project to the WebSphere Test Environment server

__ 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 **EmailPSInboundModuleApp** under Available projects panel and click **Add >**

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

____ 3.   Test the adapter application: send a pass through event e-mail with an attachment from the Mozilla Thunderbird client

---

Note: Alternatively, you can also send the same event e-mail with an attachment by running the batch file, **createEvents_sample_passthrough.bat**, located in **<EMAILFILES>\EmailEventCreator.** For more details on how to create e-mail events using the batch file, refer to the testing parts of Content specific and User defined scenarios explained later in this lab.

__ a. From the Mozilla Thunderbird, select **Emailuser2** and then click **Write** icon at the top of the window. A compose window is opened

__ b. To: Emailuser1@aimcp101.austin.ibm.com

__ c. Subject: **Email pass through test**

__ d. Body (Content): **Email pass through content**

__ e. Click **Attach** from the top of Compose window and select **<EMAILFILES>\InboundPSAttach.txt**. Note the content in this txt file so that you can verify in the coming steps.
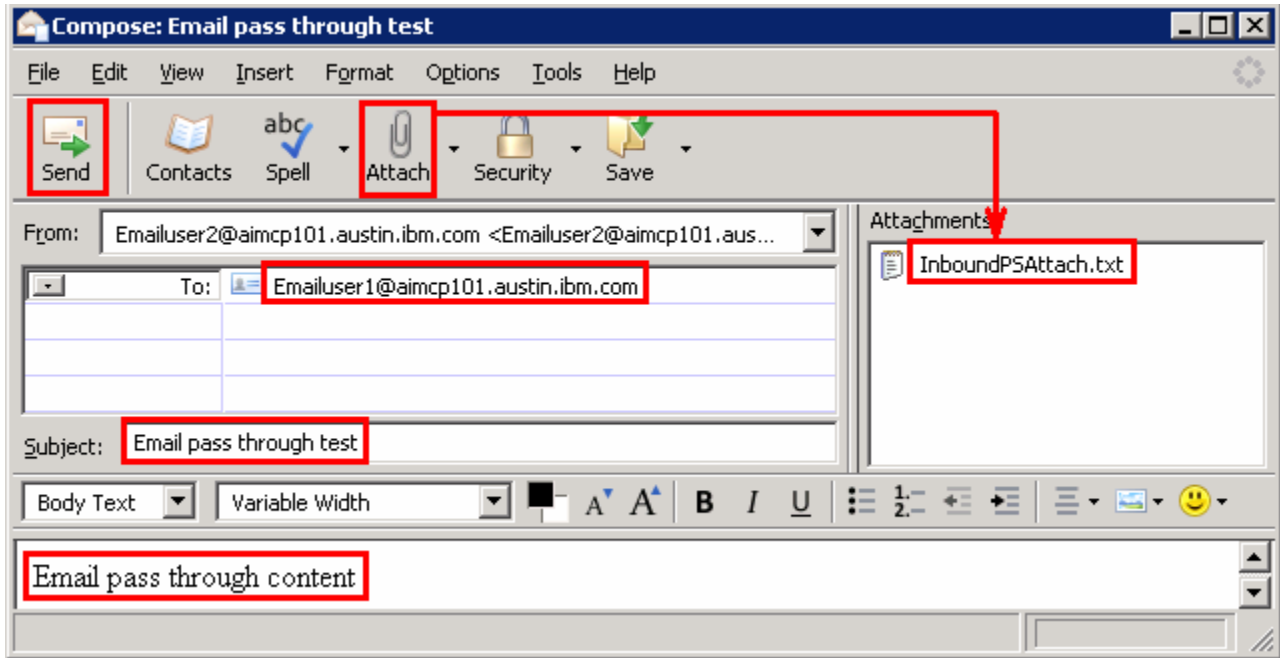
    __ f. Click **Send** from the top menu

    __ g. Select text only option if you get any pop-up window

    __ h. You can see the E-mail in the inbox of Emailuser1, but it is deleted quickly by the adapter after polling.

____ 4.   Verify the results:

    __ a. Because the adapter is running, the event E-mail will pass through the **emitEmailPS** method and you should see this message in your **Server Logs** view (or SystemOut.log):

Note: Since you have chosen the option '**Emit individual business object from a multi-part e-mail**', the content and the attachment are considered as individual BOs and hence it reached the end point twice once for the content and then for the attachment. And **each part's content is set in the mail content attribute**

Type		Time	Thread ID	Contents
☐ 🗐	Log message	Dec 8, 2009 11:36:05.453 …	00000084	WSVR0200I: Starting application: EmailPSInboundModuleApp
☐ 🗐	Log message	Dec 8, 2009 11:36:17.500 …	00000084	WSVR0221I: Application started: EmailPSInboundModuleApp
☐ 🗐	Log message	Dec 8, 2009 11:39:54.718 …	000001c8	************ENDPOINT emitEmailPS*************
☐ 🗐	Log message	Dec 8, 2009 11:39:54.718 …	000001c8	**********Email Pass through content**********
☐ 🗐	Log message	Dec 8, 2009 11:39:54.718 …	000001c8	As Text content-----> Email inbound pass through attachme..
☐ 🗐	Log message	Dec 8, 2009 11:39:54.718 …	000001c8	As Binary content-----> null
☐ 🗐	Log message	Dec 8, 2009 11:39:54.734 …	00000084	************ENDPOINT emitEmailPS*************
☐ 🗐	Log message	Dec 8, 2009 11:39:54.734 …	00000084	**********Email Pass through content**********
☐ 🗐	Log message	Dec 8, 2009 11:39:54.734 …	00000084	As Text content-----> Email pass through content
☐ 🗐	Log message	Dec 8, 2009 11:39:54.734 …	00000084	As Binary content-----> null

__ b. Check the **ArchiveDir** which should contain a folder INBOX (created by the adapter) and if you open that INBOX folder, you will see two new files with the name ending in Bodypart_0001 and Bodypart_0002

Folders		Name ▲	Size	Type
☐ 📁 EmailInbound		📄 48A32A84.5090208@aimcp101.austin.ibm.com _WebSphere_EmailRA_Bodypart_0001	1 KB	File
☐ 📁 ArchiveDir		📄 48A32A84.5090208@aimcp101.austin.ibm.com _WebSphere_EmailRA_Bodypart_0002	1 KB	File

__ c. Open the file name ending with **Bodypart_0001** under **INBOX** and observer that it is an archive of the content:

```
47826A60.1060104@aimcp101.austin.ibm.com _WebSphere_EmailRA_Bodypart_0001 - Notepad
File  Edit  Format  View  Help

Return-Path: <Emailuser2@aimcp101.austin.ibm.com>
Received: from [9.3.75.101] ([9.3.75.101])
        by localhost
        with hMailServer ; Mon, 7 Jan 2008 12:07:28 -0600
Message-ID: <47826A60.1060104@aimcp101.austin.ibm.com>_WebSphere_EmailRA_Bodypart_0001
Date: Mon, 07 Jan 2008 12:07:28 -0600
From: "Emailuser2@aimcp101.austin.ibm.com" <Emailuser2@aimcp101.austin.ibm.com>
User-Agent: Thunderbird 2.0.0.9 (windows/20071031)
MIME-Version: 1.0
To: Emailuser1@aimcp101.austin.ibm.com
Subject: Email pass through test
Content-Type: text/plain; charset=ISO-8859-1; format=flowed
Content-Transfer-Encoding: 7bit

Email pass through content
```

__ d. Open the file name ending with **Bodypart_0002** under **INBOX** and observe that it is an archive of the attachment:

```
47826A60.1060104@aimcp101.austin.ibm.com _WebSphere_EmailRA_Bodypart_0002 - Notepad
File  Edit  Format  View  Help

Return-Path: <Emailuser2@aimcp101.austin.ibm.com>
Received: from [9.3.75.101] ([9.3.75.101])
        by localhost
        with hMailServer ; Mon, 7 Jan 2008 12:07:28 -0600
Message-ID: <47826A60.1060104@aimcp101.austin.ibm.com>_WebSphere_EmailRA_Bodypart_0002
Date: Mon, 07 Jan 2008 12:07:28 -0600
From: "Emailuser2@aimcp101.austin.ibm.com" <Emailuser2@aimcp101.austin.ibm.com>
User-Agent: Thunderbird 2.0.0.9 (windows/20071031)
MIME-Version: 1.0
To: Emailuser1@aimcp101.austin.ibm.com
Subject: Email pass through test
Content-Type: text/plain; name=InboundPSAttach.txt; charset=us-ascii
Content-Transfer-Encoding: 7bit

Email inbound pass through attachment content
```

____ 5.   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 **EmailPSInboundModuleApp** 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: Content specific (non-pass through) scenario

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

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

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

_____ 1.   Create the module: EmailNonPSInboundModule

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

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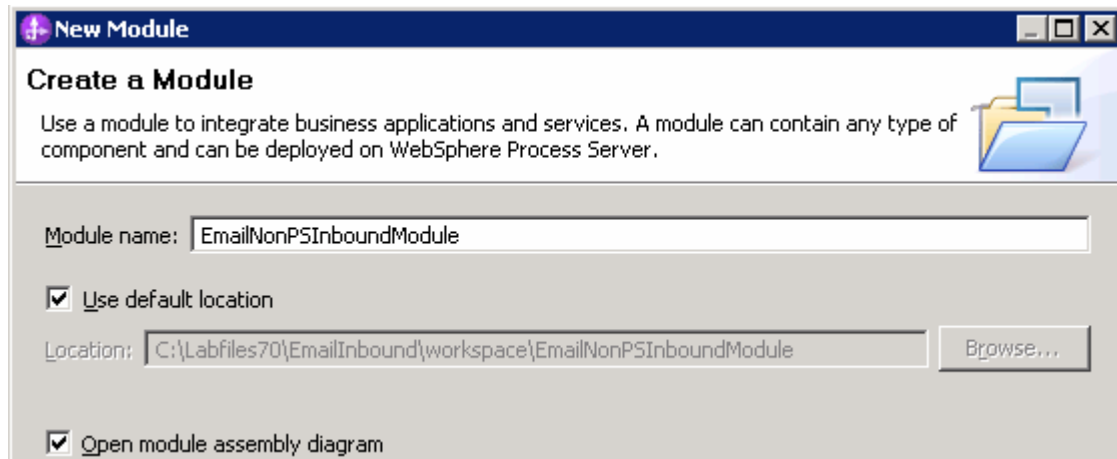
__ c. Ensure that the box next to **Open module assembly diagram** is checked and then click **Finish**

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

_____ 2.   Import required business objects

__ a. Expand EmailNonPSInboundModule (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 **<EMAILFILES>** and click **OK**

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

__ d. Select the boxes next to **Wbiaddress.xsd** and **Wbiphone.xsd**

__ e. Ensure that the **EmailNonPSInboundModule** is selected for Into folder

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

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

_____ 3.   Review the imported business objects:

IBM WEBSPHERE ADAPTER 7.0 – LAB EXERCISE

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

__ b. Double-click **Wbiaddress** to open it in assembly editor and then expand Wbiphone from assembly editor to view its fields
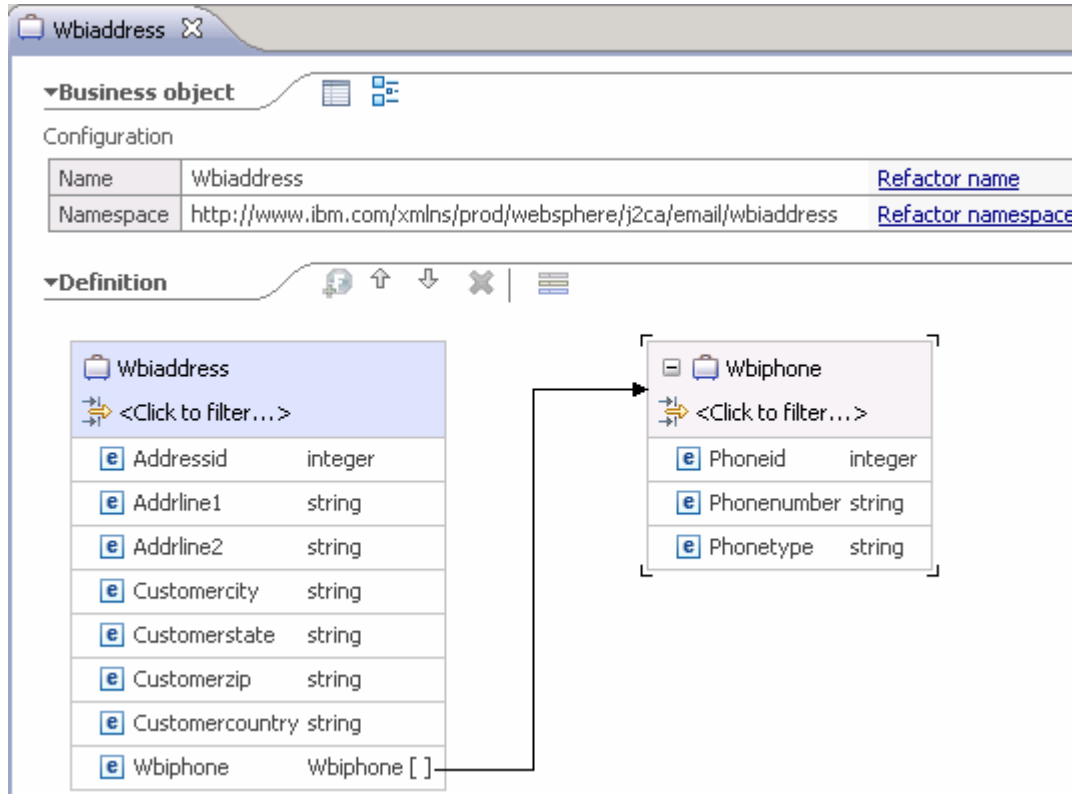
____ 4.   After reviewing, close the Customer business object from the Assembly editor

____ 5.   To start External Service from the Palette:

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

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

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 > EMail** and click **Next**

____ 6.   On the Select an Adapter screen, select **IBM WebSphere Adapter for Email (IBM : 7.0.0.0) > CWYEM_EMail** and click **Next**

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

____ 7.   Service Configuration Properties:
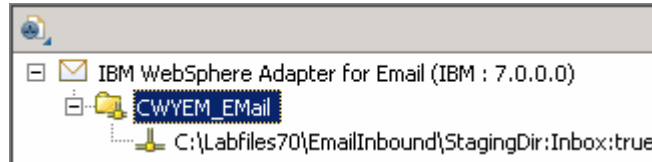
   __ a. Deploy connector project: ensure that the default option **With module for use by single application** is selected

   __ b. Enter E-mail system connection information:

      1) For Staging directory, enter **${EMAIL_STAGING}**

Note: WebSphere Environment Variable '**EMAIL_STAGING**' should already be defined using the instructions provided in **Part 1, Step 5** of this lab.

      2) Host name: **<EmailServer_HostName>** (or IP Address of the machine that has E-mail Server), for Ex: localhost

      3) Port number: **110** (default, you should change it to the correct port number if your E-mail server is running on a different port)

      4) Protocol: accept the default **POP3**

      5) Not that the **Poll folder** is by default selected as **Inbox** and is not available

   __ 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 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.

__ e. Event persistence configuration:

    1) Ensure that the **Auto create event table** is **checked**

    2) Event recovery table name: **EmailNonPSTable**

    3) Event recovery data source (JNDI) name: **jdbc/Email**

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

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".

Advanced properties:

__ f. Archive folder: All the processed events are archived in this folder

    1) Click **Browse…** next to it and select **<ARCHIVE>** and click **OK**

__ g. Failed event folder: All the failed events are archived in this folder

    1) Click **Browse…** next to it and select **<FAILED>** and click **OK**

Note: You can also use WebSphere Environment variable, as you did for the pass through scenario, instead of providing the full path of the directories here.

____ 8.   For this lab, you are going to use security properties from the activation specification. So, select the radio button next to **Using security properties from the activation specification**

   __ a. User name: **username using which you connect to your E-mail server**

Note: Use Emailuser1@aimcp101.austin.ibm.com, since it is configured as SMTP authentication user, if you followed the instructions in 'Install and configure e-mail server and client lab' to install and define users for hMailServer

   __ b. Password: **password for the user to connect to your E-mail server**

Function Selector Configuration:

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

Note: Follow the steps in **Appendix: Define Function selector** to manually define a function selector.

_____ 10. 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 **EmailWrapperDataBinding**

__ d. Click **Next**

__ e. Click **Add…** from Data binding properties table. Add/Edit window is opened

__ f. For **Binding type**, ensure that **DataHandler** is selected

__ g. Click **Browse…** next to **Business object type**

1) From Data Type Selection window, select **Wbiaddress** and click **OK**

WBPMv70_AdapterEMailInbounLab.doc

IBM WEBSPHERE ADAPTER 7.0 – LAB EXERCISE

You are now back to the Add/Edit screen and the defined business object, Wbiaddress, is populated in this screen:

___ h. For **Mime type**, select **text/xml** from the drop down list

___ i. Encoding: **UTF-8** (default)
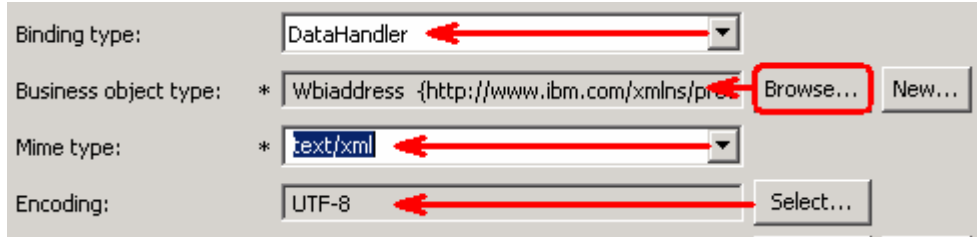
Data Handler Configuration:

___ j. Click **Select** next to **Data handler configuration**. A Binding Resource Configuration window is opened for you to define the data handler

___ k. Select **XML > UTF8XMLDataHandler** listed under **Use existing data format transformation from the list**

___ l. Click **Finish**

___ m. You are now done with defining the data handler and back to Add/Edit properties screen. The Data handler configuration name, **UTF8XMLDataHandler** is populated in this screen

__ n. Click **Finish** from the Add/Edit properties screen. You are now back to the Data binding properties screen with a new entry

__ o. Click **Next** from Data Transformation properties screen

__ p. Note that the selected module is **EmailNonPSInboundModule**

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

2) Click **Finish**

__ q. Now the **EmailNonPSDB** should be displayed for Data format

____ 11. Check the box next to **Change logging properties for wizard** to view the output location of the log file and the logging level. You can change the logging level using the drop down menu.

__ a. Click **Next**

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

IBM WEBSPHERE ADAPTER 7.0 – LAB EXERCISE

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

- Generic e-mail business object

- Generic e-mail business object with business graph

- User defined type

In this part of the lab, you will use the first option, **Generic e-mail business object with business graph**

Create emitEmailNonPS Operation:

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

__ a. For **Data type for the operation input**, select **Generic e-mail business object with business graph** from the drop down list

Operation properties

The data type for the operation input: | Generic e-mail business object with business graph ▼
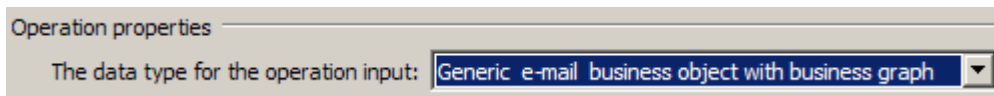
__ b. Click **Next**

The Input type is populated based on the selection of the Data type for the operation in the previous step. Since you have chosen Generic e-mail business object with business graph, the Input type is **EmailBG**

Specify the operation input

Input type: ⟶ EmailBG {http://www.ibm.com/xmlns/prod/websphere/j2ca/email/emailbg} | Browse... | New...
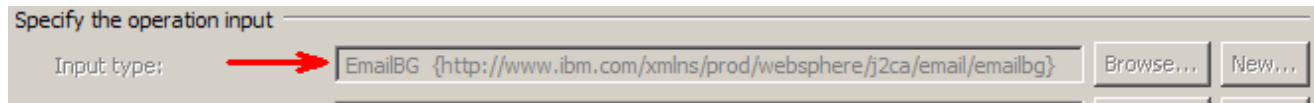
__ c. For Operation name, enter **emitEmailNonPS**

__ d. Select the Data format:

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

2) Ensure that the radio button for 'Use existing data format transformation from the list' and then select **EmailWrapperDataBinding > EmailNonPSDB**

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**

```
☐ 🗒 EmailWrapperDataBinding
     📇 EmailNonPSDB
```
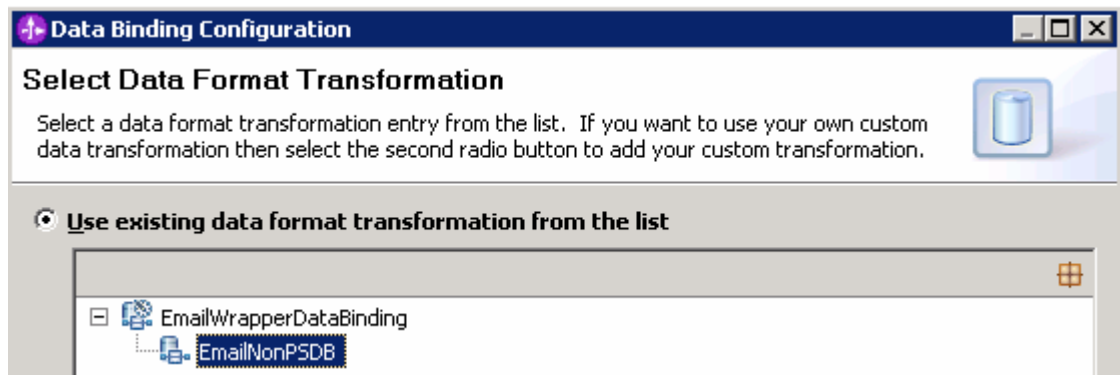
__ e. Click **Finish**

__ f. Now the **EmailNonPSDB** is displayed for Data binding configuration in the Add Operation window:

Operation name: * emitEmaiNonPS

Specify the operation input

Input type: EmailBG {http://www.ibm.com/xmlns/prod/websphere/: Browse... New...

Data format: * EmailNonPSDB ◄—— Select...
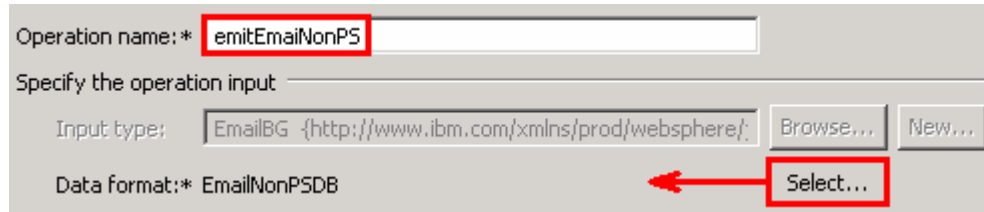
> __ g. Click **Finish**.

____ 13. The defined operation, **emitEmailNonPS**, is populated under Operations list:

Operations:

emitEmailNonPS ({http://www.ibm.com/xmlns/prod/websphere/j2ca/email/emailbg}EmailBG) : void   Add...

> __ a. Click **Next** from Operations screen

____ 14. From Generate Artifacts screen:

> __ a. For Name, enter **EmailNonPSInboundExport**

External Service

Generate Service

Specify the name and location of the new service and its interface.

Properties for service

Module: EmailNonPSInboundModule   New...

Namespace: http://EmailNonPSInboundModule/EmailNonPSInboundExport

☑ Use default namespace

Name: * EmailNonPSInboundExport

Description:
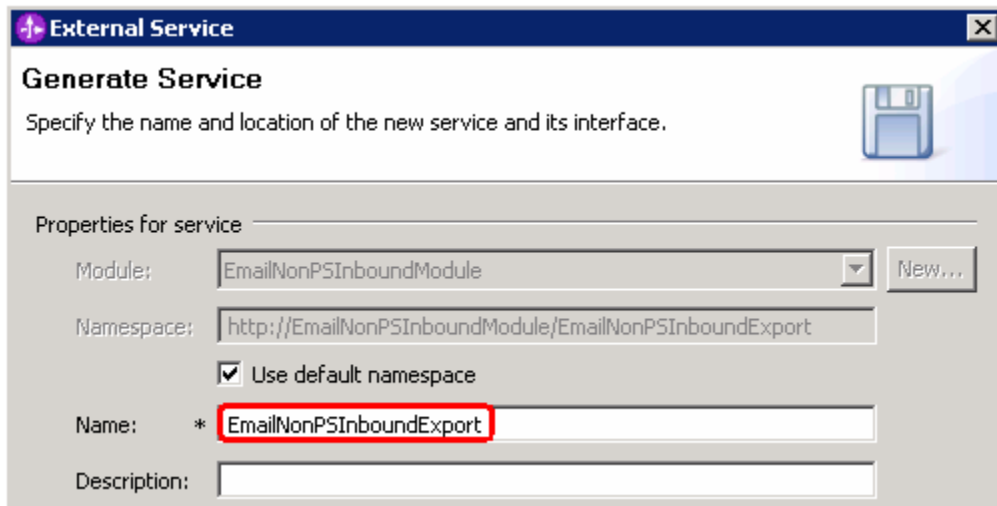
> __ b. Click **Finish**

____ 15. The Assembly diagram for EmailNonPSInboundModule is opened with an Export component, EmailNonPSInboundExport:

*EMailNonPSInboundModule - Assembly Diagram ✕

◁ Palette

Favorites

EmailNonPSInboundExport

> __ a. Save (**Ctrl + S**) changes to your assembly diagram

3.2. Add Java component

In this part of the lab, you will add a Java component and then wire the component to the existing export interface. Then you will continue to add a simple print Java code to the added Java component.

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

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

____ 3.   Wire the EmailNonPSInboundExport to the Component1

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

　　__ b. Click **EmailNonPSInboundExport** 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 EmailNonPSInboundModule will look like this:

____ 4.   Generate implementation for Component1

　　__ 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.

　　　　1) From the top of the Java implementation, under imports, type this to add one more import:

　　　　　　import java.util.*;

2) Scroll down to the method **emitEmailNonPS** that needs to be implemented and add this code under that method:

```
System.out.println("*****************ENDPOINT
emitNonPSEmail********************");
DataObject EMail = emitEmailNonPSInput.getDataObject("Email");
DataObject content = (DataObject) EMail.get("mailContent");
System.out.println("**************EMail Content***********");
String Addressid = content.getString("Addressid");
System.out.println("Address ID----> "+Addressid);
String AddrLine1 = content.getString("Addrline1");
System.out.println("Address Line1--> "+AddrLine1);
String AddrLine2 = content.getString("Addrline2");
System.out.println("Address Line2--> "+AddrLine2);
String CustomerCity = content.getString("Customercity");
System.out.println("Customer City----> "+CustomerCity);
String CustomerState = content.getString("Customerstate");
System.out.println("Customer State-----> "+CustomerState);
String CustomerZip = content.getString("Customerzip");
System.out.println("Customer Zip-----> "+CustomerZip);
String CustomerCountry = content.getString("Customercountry");
System.out.println("Customer Country-----> "+CustomerCountry);
//Get all the attachments as an ArrayList and iterate through the
list
List list = EMail.getList("mailAttachments");
String attachName = null;
DataObject tempSDO=null;
//Extract each attachment
for (int i = 0; i<list.size(); i++){
        Object attachmentObject = list.get(i);
        tempSDO = (DataObject)list.get(i);
//Attachment name and content
        attachName = tempSDO.getString("attachmentName");
        System.out.println("***ATTACHMENT NAME: "+attachName +"***");
        DataObject bo = (DataObject) tempSDO.get("appBOData");
        System.out.println("*************Attachment
        Content***********");
        String PhoneID = bo.getString("Phoneid");
        System.out.println("Phone ID-----> "+PhoneID);
        String PhoneNumber = bo.getString("Phonenumber");
        System.out.println("Phone Number--> "+PhoneNumber);
        String PhoneType = bo.getString("Phonetype");

        System.out.println("Phont Type--> "+PhoneType);

        }
```

Note: The Java code is also available at **<EMAILFILES>\ NonPSInboundJavaCode.txt**

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

__ e. Save (**Ctrl + S**) and close Assembly diagram: EmailNonPSInboundModule

3.3. Test non pass through scenario

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

____ 1. Start your e-mail server (if not started already)

  __ a. Select **Start > hMailServer >hMailServerAdministrator**

  __ b. From the hMailServer Administrator – Connect window, ensure that **localhost** is selected and click **Connect**

  __ c. hMailServer Administrator window is opened and the Current status should show **Running**

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

  __ c. You will now see the **EmailNonPSInboundModuleApp** 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
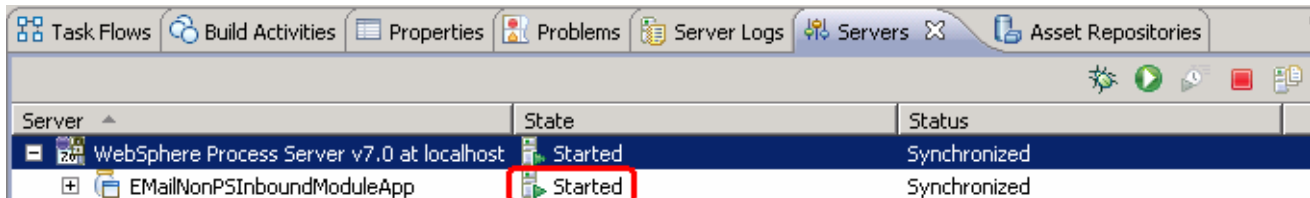
____ 3. Review and modify the batch file to generate required event E-mails

  __ a. From windows explorer, browse to **<EMAILFILES\CreateFixedEvents**

Note: For your convenience, the files under **<EMAILFILES>\CreateFixedEvents** are already modified for this lab.

  __ b. Review the batch file, createEvents.bat, under CreateFixedEvents folder:

  Structure of **createEvents.bat** file:

  <host> <port> <number of events to be created> <from> <to> <cc> <bcc> <subject> <content> <path for event file name> <Boolean for pass through/non pass through>

  Where, Path for event file name - is the path on the local folder where the event file attachment has been placed.

  Boolean for pass-through/non-pass-through – has true or false values. **True** indicates pass-through and **False** indicates non-pass-through.

__ c. Review **createEvents_sample_nonpassthrough.bat** file under CreateFixedEvents folder. This file, for your convenience, is created by providing the required inputs specified in the createEvents.bat. This file is used to generate the required E-mail events:

java -cp .\;.\mail.jar;.\activation.jar EmailEventCreatorNew "localhost" 25 1 "Emailuser2@aimcp101.austin.ibm.com" Emailuser1@aimcp101.austin.ibm.com Emailuser3@aimcp101.austin.ibm.com Emailuser4@aimcp101.austin.ibm.com "Test Email inbound Non-pass through" "C:\Labfiles70\EmailFiles\WbiAddress.xml" "C:\Labfiles70\EmailFiles\WbiPhone.xml" "false"

Note: This will generate 1 event E-mail from Emailuserser2@aimcp101.austin.ibm.com on the localhost mail server, to the inbox of Emailuserser1@aimcp101.austin.ibm.com with the file **WbiPhone.xml** as an attachment and the content of **WbiAddress.xml** file as the body of the E-mail. **You should change the fields according to your E-mail server settings and file locations.**

_____ 4. Run the batch file to generate email

__ a. Open a command prompt and change the directory to **<EMAILFILES>\CreateFixedEvents**

__ b. Enter **createEvents_sample_nonpassthrough.bat**

```
MAIL FROM:<Emailuser2@aimcp101.austin.ibm.com>
250 OK
RCPT TO:<Emailuser1@aimcp101.austin.ibm.com>
250 OK
RCPT TO:<Emailuser3@aimcp101.austin.ibm.com>
250 OK
RCPT TO:<Emailuser4@aimcp101.austin.ibm.com>
250 OK
DEBUG SMTP: Verified Addresses
DEBUG SMTP:    Emailuser1@aimcp101.austin.ibm.com
DEBUG SMTP:    Emailuser3@aimcp101.austin.ibm.com
DEBUG SMTP:    Emailuser4@aimcp101.austin.ibm.com
DATA
354 OK, send.
Message-ID: <1762786195.1201033529625.JavaMail.Administrator@localhost>
Date: Tue, 22 Jan 2008 14:25:29 -0600 (CST)
From: Emailuser2@aimcp101.austin.ibm.com
To: Emailuser1@aimcp101.austin.ibm.com
Subject: Test Email inbound Non-pass through
Cc: Emailuser3@aimcp101.austin.ibm.com
Mime-Version: 1.0
Content-Type: multipart/mixed;
        boundary="----=_Part_0_1720073107.1201033529500"

------=_Part_0_1720073107.1201033529500
Content-Type: text/xml; charset=us-ascii
Content-Transfer-Encoding: 7bit

<?xml version="1.0" encoding="UTF-8"?>
<wbiaddress:Wbiaddress xsi:type="wbiaddress:Wbiaddress" xmlns:xsi="http://www.w3
.org/2001/XMLSchema-instance" xmlns:wbiaddress="http://www.ibm.com/xmlns/prod/we
bsphere/j2ca/email/wbiaddress">
    <Addressid>98765</Addressid>
    <Addrline1>11501</Addrline1>
    <Addrline2>Burnet Rd</Addrline2>
    <Customercity>Austin</Customercity>
    <Customerstate>TX</Customerstate>
    <Customerzip>78758</Customerzip>
    <Customercountry>USA</Customercountry>
</wbiaddress:Wbiaddress>

------=_Part_0_1720073107.1201033529500
Content-Type: text/xml; name=sample.xml; charset=us-ascii
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename=sample.xml

<?xml version="1.0" encoding="UTF-8"?>
<wbiphone:Wbiphone xsi:type="wbiphone:Wbiphone" xmlns:xsi="http://www.w3.org/200
1/XMLSchema-instance" xmlns:wbiphone="http://www.ibm.com/xmlns/prod/websphere/j2
ca/email/wbiphone">
    <Phoneid>12345</Phoneid>
    <Phonenumber>5128380000</Phonenumber>
    <Phonetype>Office</Phonetype>
</wbiphone:Wbiphone>

------=_Part_0_1720073107.1201033529500--
```
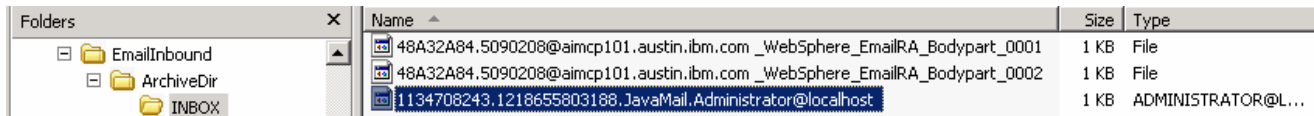
WBPMv70_AdapterEMailInbounLab.doc

__ c. You can see the E-mail in the inbox of Emailuser1, but it is deleted quickly by the adapter after polling.

____ 5.   Verify the results:

__ a. Because the adapter is running, the event E-mail will pass through the **emitEmailNonPS** method and you should see these messages in your **Server Logs** view (or SystemOut.log):

🗒 Log message	Dec 18, 2008 15:47:21.125	00000b10	WSVR0221I: Application started: EmailNonPSInboundModul...
☐ 🗒 Log message	Dec 18, 2008 15:50:20.328	00000623	Expected type:EmailBG
☐ 🗒 Log message	Dec 18, 2008 15:50:20.328	00000623	expectedType:http://www.ibm.com/xmlns/prod/webspher....
☐ 🗒 Log message	Dec 18, 2008 15:50:20.531	000005bf	****************ENDPOINT emitNonPSEmail*********...
☐ 🗒 Log message	Dec 18, 2008 15:50:20.531	000005bf	*************EMail Content************
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Address ID-----> 0
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Address Line1--> EGL Business Park
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Address Line2--> Kormangala
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Customer City-----> Bangalore
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Customer State-----> Karnataka
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Customer Zip-----> 560071
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Customer Country-----> INDIA
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	***ATTACHMENT NAME: sample.xml***
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	*************Attachment Content************
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Phone ID-----> 0
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Phone Number--> 9437843788
☐ 🗒 Log message	Dec 18, 2008 15:50:20.546	000005bf	Phont Type--> cell

__ b. Check the **<ARCHIVE>** which should contain a folder INBOX (created by the adapter) and if you open that **INBOX** folder, you will see a new file with the name ending in your host name.

Folders	×	Name ▲	Size	Type
⊟ 📁 EmailInbound		🗎 48A32A84.5090208@aimcp101.austin.ibm.com _WebSphere_EmailRA_Bodypart_0001	1 KB	File
⊟ 📁 ArchiveDir		🗎 48A32A84.5090208@aimcp101.austin.ibm.com _WebSphere_EmailRA_Bodypart_0002	1 KB	File
📁 INBOX		🗎 1134708243.1218655803188.JavaMail.Administrator@localhost	1 KB	ADMINISTRATOR@L...

__ c. You can open the file and verify the same content that the E-mail was created with.

____ 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 **EmailNonPSInboundModuleApp** 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 4: User defined (Fixed structure e-mail) scenario

User defined scenario is also a non-pass through but you can specify your own business object types using this option. In this part of the lab, you will configure **User defined scenario** using the new External Service option from the WebSphere Integration Developer and then test the configuration with different cases

4.1. Configure fixed structure using the external service wizard

In this part of the lab you will use this new External Service feature to create and configure the Function Selector, Data Binding and other required artifacts to test the inbound pass through scenario

____ 1. Create the module: **EmailFixedInboundModule**

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

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

____ 2. Import required business objects

   __ a. Expand **EmailFixedInboundModule** (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 **<EMAILFILES>** and click **OK**

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

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

   __ e. Ensure that the **EmailFixedInboundModule** is selected for Into folder

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

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

____ 3. Review the imported business objects:

__ a. Expand **EmailFixedInboundModule > 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:

__ c. Now, double-click **Order** and review the fields inside the object:

____ 4.   After reviewing, close the Customer and Order business objects 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**:

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

---

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 > EMail** and click **Next**.

____ 6.   On the Select an Adapter screen, select **IBM WebSphere Adapter for Email (IBM : 7.0.0.0) > CWYEM_EMail** and click **Next**

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

____ 7.   Service Configuration Properties:

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

__ b. Enter E-mail system connection information:

IBM WEBSPHERE ADAPTER 7.0 – LAB EXERCISE

1) For Staging directory, enter **${EMAIL_STAGING}**

Note: WebSphere Environment Variable '**EMAIL_STAGING**' should already be defined using the instructions provided in **Part 1, Step 5** of this lab.

---

2) Host name: **<EmailServer_HostName>** (or IP Address of the machine that has E-mail Server), for Ex: localhost

3) Port number: **110** (default, you should change it to the correct port number if your E-mail server is running on a different port)

4) Protocol: accept the default **POP3**

5) Not that the **Poll folder** is by default selected as **Inbox** and is not available

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

Connection properties

| Staging directory: | * ${EMAIL_STAGING} | Browse... |

E-mail system connection information

Host name:	localhost
Port number:	110
Protocol:	pop3
Poll folder or folders:	Inbox

__ 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 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.

__ e. Event persistence configuration:

1) Ensure that the **Auto create event table** is **checked**

2) Event recovery table name: **EmailFixedTable**

3) Event recovery data source (JNDI) name: **jdbc/Email**

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

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".

Advanced properties:

__ f. Archive folder: All the processed events are archived in this folder

1) Click **Browse…** next to it and select **<ARCHIVE>** and click **OK**

__ g. Failed event folder: All the failed events are archived in this folder

1) Click **Browse…** next to it and select **<FAILED>** and click **OK**

Note: You can also use WebSphere Environment variable, as you did for the pass through scenario, instead of providing the full path of the directories here.

____ 8.   For this lab, you are going to use security properties from the activation specification. So, select the radio button next to **Using security properties from the activation specification**

__ a. User name: **username using which you connect to your E-mail server**

Note: Use Emailuser1@aimcp101.austin.ibm.com, since it is configured as SMTP authentication user, if you followed the instructions in 'Install and configure e-mail server and client lab' to install and define users for hMailServer

__ b. Password: **password for the user to connect to your E-mail server**

Service properties
How do you want to specify the security credentials?

○ Using an existing JAAS alias (recommended)

A Java Authentication and Authorization Services (JAAS) alias is the preferred method.

J2C authentication data entry:

⊙ Using security properties from the activation specification

The properties will be stored as plain text; no encryption is used.

User name: Emailuser1@aimcp101.austin.ibm.com

Password: *********

○ Other

Use if no security is required or will be handled by the EIS system, or the RAR will be deployed on the server and security will be specified by the properties in the JNDI lookup name.

Function Selector Configuration:

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

---

Note: Follow the steps in **Appendix: Define Function selector** to manually define a function selector.

Function selector options: Use default function selector 'EmailFunctionSelector'   ▼

Function selector: Not defined   Select...

_____ 10.   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...

Data binding configuration:

__ 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 **EmailFixedStructureDataBinding**

__ d. Click **Next**

____ 11.   From **Data Binding Properties** screen, Define **mailContent**:

1) Click **Add…** next to the table. Add/Edit window is opened

2) Note that the **E-mail part** is **mailContent**

3) Click **Browse…** next to **Business object type**. Data Type Selection window is opened

a) Select **Customer** from the **Matching data types** list and click **OK**

4) For **Mime type**, select **text/xml** from the drop down list

5) For **Encoding**, accept the default **UTF-8** (Optionally, you can click Select and then select any encoding from the available list)

Data Handler Configuration:

__ b. Click **Select** next to **Data handler configuration**. A Binding Resource Configuration window is opened for you to define the data handler

__ c. Select **XML > UTF8XMLDataHandler** listed under Use existing data format transformation from the list

__ d. Click **Finish**

__ e. You are now done with defining the data handler and back to Add/Edit properties screen. The Data handler configuration name, **UTF8XMLDataHandler** is populated in this screen

__ f. Click **Finish** from the Add/Edit properties screen

__ g. You are back to the Data binding properties screen with a new entry. You have so far defined the Business object type for the mailContent which is populated in this window:

_____ 12. Define **attachment** part:

__ a. Click **Add…** next to the 'Data binding properties' table. This will open an Add/Edit window

__ b. Note that the **E-mail part** is **attachment1**

__ c. Click **Browse…** next to **Business object type**. Data Type Selection window is opened

1) Select **Order** from the Matching data types list and click **OK**

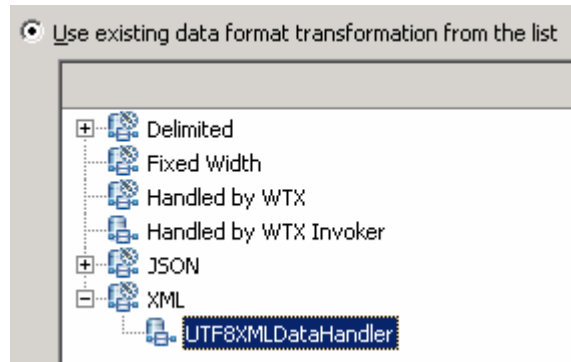__ d. For **Mime type**, select **text/xml** from the drop down list

__ e. For **Encoding**, accept the default **UTF-8** (Optionally, you can click Select and then select any encoding from the available list)

__ f. Click **Select…** next to **Data handler configuration**

1) From Data Handler Configuration window, select **XML > UTF8XMLDataHandler** under 'Use existing data format transformation from the list' and click **Finish**

__ g. The data handler is populated in Add/Edit window:

Binding type:	DataHandler ▼
E-mail part:	→ attachment1
Business object type:	* Order {http://www.ibm.c... Browse... New...
Mime type:	* text/xml ← ▼
Encoding:	UTF-8 ← Select...
Data handler configuration:*	UTF8XMLDataHandler ← Select...
Data binding configuration:	Not defined Select...

__ h. Click **Finish** from the Add/Edit window

__ i. You are back to the Binding Resource Configuration window and the defined Business object type and the Email-part are displayed in the table format in this screen:

Select DataBinding if you want to use a data binding developed for earlier versions of the adapter.

Data binding properties:

	Binding type	E-mail part	Business object type	Mime type	Encoding	Data handler configuration	Data binding configu	
	DataHandler	attachment1	{http://www.ibm.com/x...	text/xml	UTF-8	{http://EmailFixedInboun...		Add...
	DataHandler	mailContent	{http://www.ibm.com/x...	text/xml	UTF-8	{http://EmailFixedInboun...		Edit...

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

__ k. Note that the selected module is **EmailFixedInboundModule**

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

Module:	EmailFixedInboundModule ▼ Browse... New...
Namespace:	http://EmailFixedInboundModule ☑ Default
Folder:	Browse...
Name:	EmailFixedDB

2) Click **Finish**

__ l. Now the **EmailFixedDB** should be displayed for Data binding configuration

Data format options:	Use a data format configuration for all operations ← ▼
Data format:	* EmailFixedDB ← Select...

____ 13. Check the box next to **Change logging properties for wizard** to view the output location of the log file and the logging level. You can change the logging level using the drop down menu

__ a. Click **Next**

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

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

- Generic e-mail business object

- Generic e-mail business object with business graph

- User defined e-mail business object

In this part of the lab, you will use the last option, **User defined email business object**.

Create emitFixedEmail Operation:

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

__ a. For **The data type for the operation input**, select **User defined email business object** from the drop down list
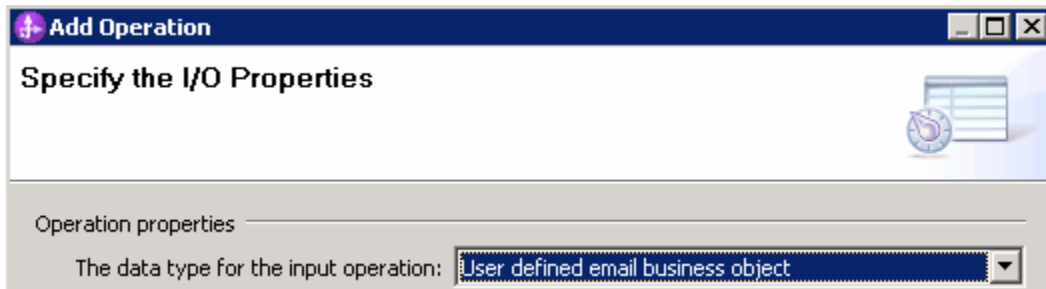
__ b. Click **Next**

The Data type for input is 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**. You can define your own data type here.

__ c. For Operation name, enter **emitFixedEmail**

  Define Input type:

__ d. Click **New…** next to **Input type**. New Business Object is opened

__ e. Ensure that the selected Module is **EmailFixedInboundModule** an click **Next**

__ f. For **Business object name**, enter **MyCustomEmail** (you can enter any name here)

__ g. **Select** the box next to **Generate a business graph for the business object**

Define **mailContent**:

__ h. Click **Add…** next to the 'Properties for e-mail type' table. This will open an Add/Edit window

  1) Note that the **E-mail part** is **mailContent**

  2) Click **Browse…** next to **Business object type**. Data Type Selection window is opened

  3) Select **Customer** from the Matching data types list and click **OK**

__ i. Now, back to Add/Edit window. You have so far defined the Business object type for the mailContent which is populated in this window:

E-mail part:	mailContent ▼	
Business object type:	* Customer {http://www.ibm	Browse... New...
Default attachment name:		

___ j. Click **Finish** from the Add/Edit window

___ k. You are back to the New Business Object window and the defined Business object type and the Email-part are displayed in the table format in this screen:

Business object name: MyCustomEmail

Business object namespace: http://www.ibm.com/xmlns/prod/websphere/j2ca/email/fixedstructureemail

☑ Generate a business graph for the business object

Properties for e-mail type:

	E-mail part	Business object type	Default attachment n	
	mailContent	{http://www.ibm.com/xmlns/prod/websphere/j2ca/email/customer}Customer		Add...
				Edit...
				Remove

⑦   < Back   Next >   Finish   Cancel

Define **attachment** part:

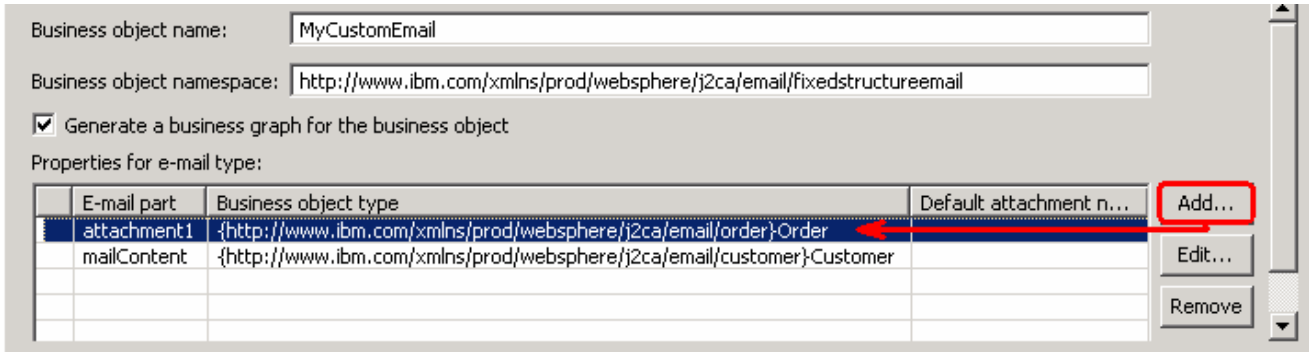___ l. Click **Add…** next to the 'Properties for e-mail type' table. This will open an Add/Edit window

1) Note that the **E-mail part** is **attachment1**

2) Click **Browse…** next to Business object type. Data Type Selection window is opened

3) Select **Order** from the Matching data types list and click **OK**

___ m. Now, back to Add/Edit window. You have so far defined the Business object type for the attachment1 which is populated in this window:

E-mail part:	attachment1 ▼	
Business object type:	* Order {http://www.ibm.co	Browse... New...
Default attachment name:		

___ n. Click **Finish** from the Add/Edit window

___ o. You are back to New Business Object window and the defined Business object type and the Email-part are displayed in the table format in this screen:
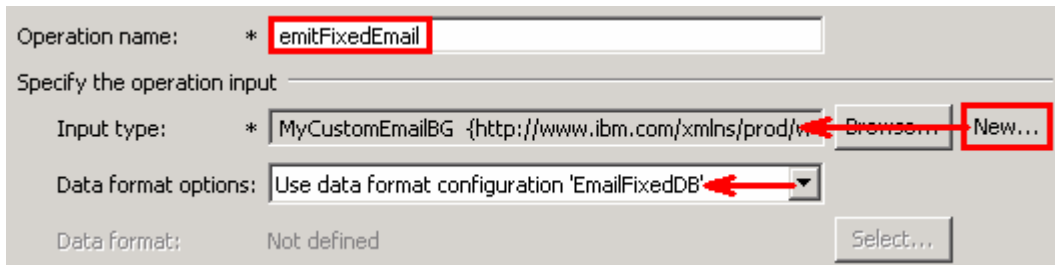
IBM WEBSPHERE ADAPTER 7.0 – LAB EXERCISE

	E-mail part	Business object type	Default attachment n...	Add...

Business object name: MyCustomEmail

Business object namespace: http://www.ibm.com/xmlns/prod/websphere/j2ca/email/fixedstructureemail

☑ Generate a business graph for the business object

Properties for e-mail type:

	E-mail part	Business object type	Default attachment n...	Add...
	attachment1	{http://www.ibm.com/xmlns/prod/websphere/j2ca/email/order}Order		
	mailContent	{http://www.ibm.com/xmlns/prod/websphere/j2ca/email/customer}Customer		Edit...
				Remove

__ p. Click **Finish**

You are now back to the Add operation window and the Input type you have defined in the previous steps is displayed here. Next step is selecting Data binding configuration.
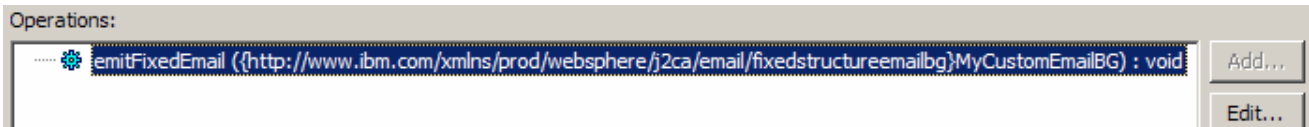
__ q. Accept the default selection, **Use data format configuration 'EmailFixedDB'**, for **Data format options**
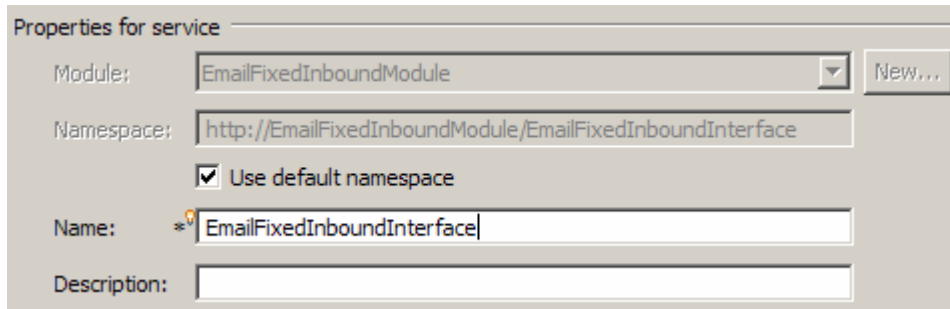
__ r. Add Operation window should have the below values:

Operation name: * emitFixedEmail

Specify the operation input

Input type: * MyCustomEmailBG {http://www.ibm.com/xmlns/prod/w   Browse   New...

Data format options: Use data format configuration 'EmailFixedDB'

Data format: Not defined   Select...

__ s. Click **Finish** from the Operation screen

____ 15.   The defined operation, **emitFixedEmail**, is populated under Operations list

Operations:

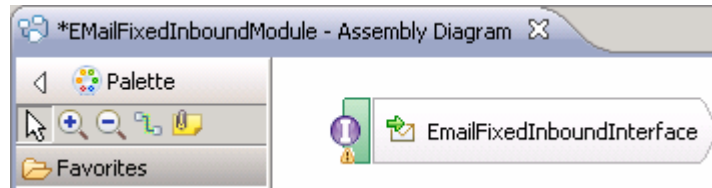⚙ emitFixedEmail ({http://www.ibm.com/xmlns/prod/websphere/j2ca/email/fixedstructureemailbg}MyCustomEmailBG) : void   Add...

Edit...

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

____ 16.   From Generate Service screen, enter **EmailFixedInboundInterface** for Name:

Properties for service

Module: EmailFixedInboundModule   New...

Namespace: http://EmailFixedInboundModule/EmailFixedInboundInterface

☑ Use default namespace

Name: * EmailFixedInboundInterface

Description:

__ a. Click **Finish**

____ 17. The Assembly diagram for EmailFixedInboundModule is opened with an export component, EmailFixedInboundInterface:


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

4.2. Add Java component

In this part of the lab, you will add a Java component and then wire the component to the existing export interface. Then you will continue to add a simple print Java code to the added Java component.

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

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

____ 3.   Wire the EmailFixedInboundInterface to the Component1

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

   __ b. Click **EmailFixedInboundInterface** 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 EmailFixedInboundModule will look like this:
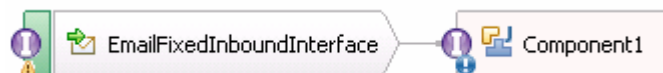
____ 4.   Generate implementation for Component1

   __ 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 **emitEmailPS** that needs to be implemented and add this code under that method:

```
System.out.println("************ENDPOINT
emitFixedEmail*************");
  DataObject customemail =
  emitFixedEmailInput.getDataObject("MyCustomEmail");
        System.out.println("*********Email content*********");
        DataObject customer = customemail.getDataObject("mailContent");
        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);
System.out.println("**********Attachment data**********");
DataObject order = customemail.getDataObject("attachment1");
String OrderNumber = order.getString("OrderNumber");
System.out.println("ORDER NUMBER-----> "+OrderNumber);
String OrderType = order.getString("OrderType");
System.out.println("ORDER TYPE--> "+OrderType);
String Quantity = order.getString("Quantity");
System.out.println("QUANTITY-----> "+Quantity);
String Price = order.getString("Price");

System.out.println("PRICE----> "+Price);
```

Note: The Java code is also available at **<EMAILFILES>\ FixedInbondJavaCode.txt**

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

__ e. Save (**Ctrl + S**) and close Assembly diagram: EmailFixedInboundModule

4.3. Test user defined scenario

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

____ 1. Start your e-mail server (if not started already)

__ a. Select **Start > hMailServer >hMailServerAdministrator**

__ b. From the hMailServer Administrator – Connect window, ensure that **localhost** is selected and click **Connect**

__ c. hMailServer Administrator window is opened and the Current status should show **Running**

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

__ c. You will now see the **EmailFixedInboundModuleApp** 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. Review and modify the batch file to generate required event E-mails

__ a. From windows explorer, browse to **<EMAILFILES\CreateFixedEvents**

Note: You can find CreateUserDefinedEvents.zip at <EMAILRAR_FILE>\Samples also which shipped with WebSphere Integration Developer. You will need to extract it and make modifications to the batch file inside the archive file. For your convenience, you can use the files under **<EMAILFILES>\CreateFixedEvents** which are already modified for this lab.

__ b. Review the batch file, createEvents.bat, under CreateFixedEvents folder:

Structure of **createEvents.bat** file:

<host> <port> <number of events to be created> <from> <to> <cc> <bcc> <subject> <content> <path for event file name> <Boolean for pass through/non pass through>

Where, Path for event file name - is the path on the local folder where the event file attachment has been placed.

Boolean for pass-through/non-pass-through – has true or false values. **True** indicates pass-through and **False** indicates non-pass-through.

__ c. Review **createEvents_sample_fixed.bat** file under CreateFixedEvents folder. This file, for your convenience, is created by providing the required inputs specified in the createEvents.bat. This is used to generate the required E-mail events:

java -cp .\;.\mail.jar;.\activation.jar EmailEventCreatorNew "localhost" 25 1 "Emailuser2@aimcp101.austin.ibm.com" Emailuser1@aimcp101.austin.ibm.com Emailuser3@aimcp101.austin.ibm.com Emailuser4@aimcp101.austin.ibm.com "Test Email

inbound Fixed" "C:\Labfiles70\EmailFiles\Customer.xml" "C:\Labfiles70\EmailFiles\Order.xml"
"false"

Note: This will generate 1 event e-mail from Emailuserser2@aimcp101.austin.ibm.com on the localhost mail server, to the inbox of Emailuserser1@aimcp101.austin.ibm.com with the file **Order.xml** as an attachment and the content of **Customer.xml** as the body of the e-mail. **You should change the fields according to your e-mail server settings and file locations.**

_____ 4.   Run the batch file to generate E-mail

__ a. Open a command prompt and change the directory to **<EMAILFILES>\CreateFixedEvents**

__ b. Enter **createEvents_sample_fixed.bat**

__ c. You can see the E-mail in the inbox of Emailuser1, but it is deleted quickly by the adapter after polling.

_____ 5.   Verify the results:

__ a. Because the adapter is running, the event E-mail will pass through the **emitFixedEmail** method and you should see these messages in your **Server Logs** view (or SystemOut.log):

☑ 📄 Log message	Dec 18, 2008 16:15:12.906	00000b10	WSVR0221I: Application started: EmailFixedInboundModule...
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	************ENDPOINT emitFixedEmail***************
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	**********Email content**********
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	NAME-----> IBM
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	ADDRESS--> 11501 Burnet Rd
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	CITY-----> Austin
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	STATE----> TX
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	**********Attachment data**********
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	ORDER NUMBER-----> ABC12345
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	ORDER TYPE--> BULK
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	QUANTITY-----> 500
☐ 📄 Log message	Dec 18, 2008 16:15:33.078	00000623	PRICE----> 26.59

__ b. Check the **ArchiveDir** which should contain a folder INBOX (created by the adapter) and if you open that INBOX folder, you will see a new file with the name ending in your host name

Folders	✕	Name ▲	Size	Type
⊟ 📁 EmailInbound	▲	📧 48A32A84.5090208@aimcp101.austin.ibm.com _WebSphere_EmailRA_Bodypart_0001	1 KB	File
⊟ 📁 ArchiveDir		📧 48A32A84.5090208@aimcp101.austin.ibm.com _WebSphere_EmailRA_Bodypart_0002	1 KB	File
📁 INBOX		📧 1134708243.1218655803188.JavaMail.Administrator@localhost	1 KB	ADMINISTRATOR@L...
📁 FailedEventDir		📧 1134952562.1218656887594.JavaMail.Administrator@localhost	1 KB	ADMINISTRATOR@L...

_____ 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 **EmailFixedInboundModuleApp** 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 solution Project Interchange

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

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

    __ c. Select **Other → Project Interchange** in the **Import** dialog and click **Next**
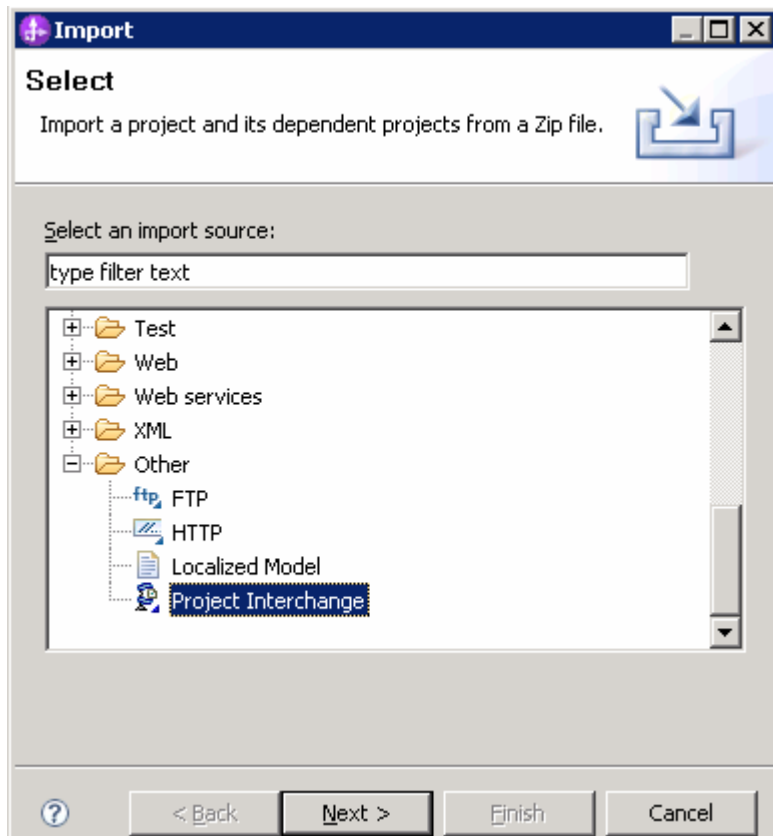
    __ d. For the **From zip file**, click on the **Browse** button and select the **EmailInbound_PI.zip** in the **<LAB_FILES>>\ EmailInbound\solution** directory

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

__ f. Click the **Select All** button. This will select all modules: **CWYEM_EMail, EmailFixedInboundModule, EmailNonPSInboundModule,** and **EmailPSInboundModule**
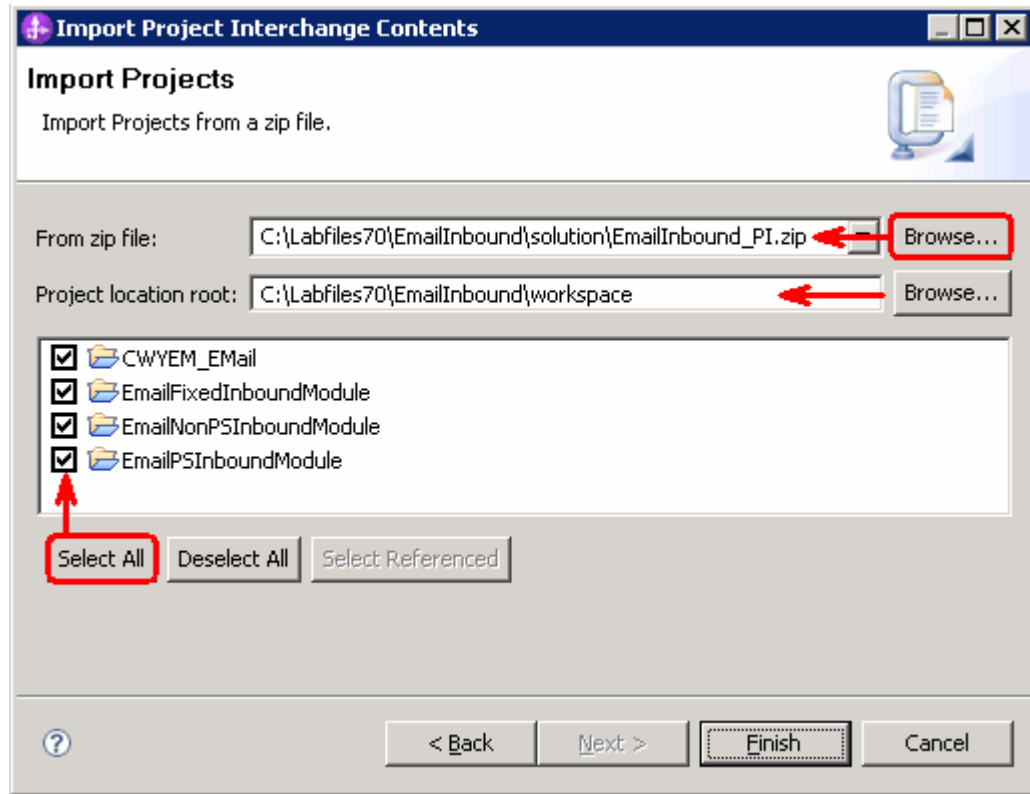
__ g. Click **Finish**

____ 3.   Test inbound pass through scenario: Continue with **Part 2.3** of this lab

____ 4.   Test inbound non pass through scenario: Continue with **Part 3.3** of this lab

____ 5.   Test inbound user defined (fixed structure e-mail) scenario: Continue with **Part 4.3** of this lab

What you did in this exercise

In this lab, you started with importing the E-mail Adapter RAR file into your WebSphere Integration Developer new workspace. Then, used your WebSphere Process Server administrative console and configured it to create the Data source and data base required to complete this lab.

You made use of the External Service wizard available in WebSphere Integration Developer to specify Activation Spec Properties and Resource Adapter Properties which, after deploying onto the server, will generate Business Objects and other artifacts.

 In the end you deployed and then tested the adapter application for pass-through, non-pass through and user defined (fixed structure) scenarios.

Appendix: Define function selector

In this part of the lab, you will define a function selector.

____ 1. Start External Service Wizard from the file menu or by dropping the adapter onto the canvas as shown in previous steps

____ 2. Under Service properties, for **Function selector options**, select **Use default function selector configuration** from the drop down list

____ 3. Click **New** next to **Function selector**. The Resource Configuration window is opened.

__ a. Ensure that the selected module is **EmailPSInboundModule**

__ b. For Name, enter any string. For Ex: **PSFS**

Module:	EmailPSInboundModule New...
Namespace:	http://EmailPSInboundModule
	☑ Use default namespace
Folder:	Browse...
Name:	* PSFS

__ c. Click **Next**

____ 4. In the next Function Selector Configuration window, select **EmailFunctionSelector** listed under 'Use existing function selector from the list'

Function Selector Configuration   _ □ ✕

Select Function Selector

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

◉ Us**e** existing function selector from the list

    ⊟ EmailFunctionSelector

__ a. Click **Finish**

____ 5. You will now be back to External Service window and the function selector created here is populated:

|	
Function selector options:	Use a function selector configuration ◄ ▼
Function selector:	* EmailFunctionSelector ◄ Select...

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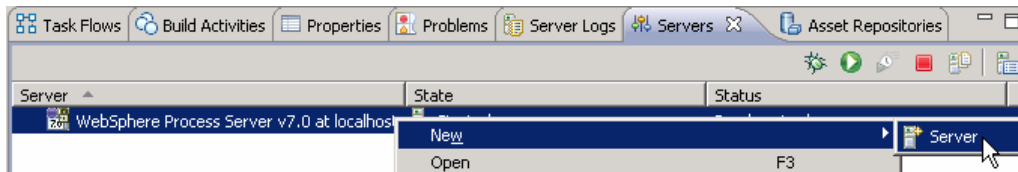
__ c. In the New Server dialog, specify the remote server's host name, **<HOSTNAME>**.

__ 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
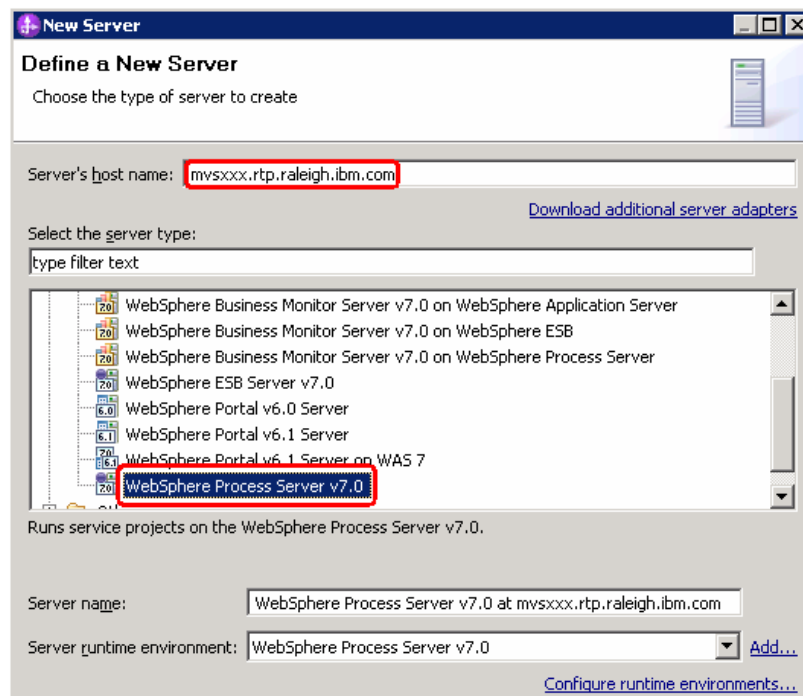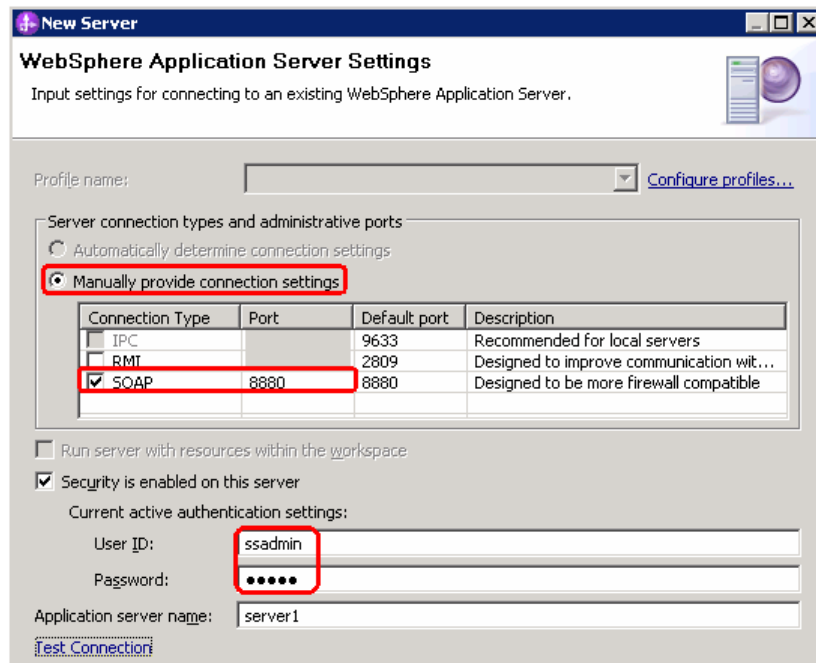

__ e. Click **Next**

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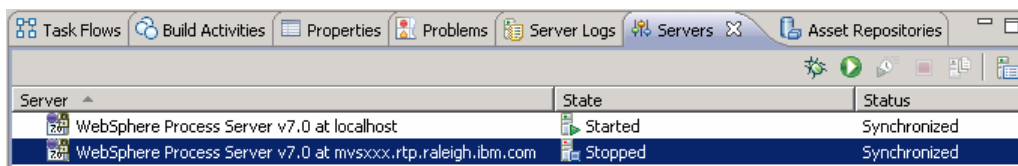
____ 2. Start the remote server if it is not already started. WebSphere Integration Developer does not support starting remote servers from the Server view.

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

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

User ID : **<USERID>**

Password : **<PASSWORD>**

WBPMv70_AdapterEMailInbounLab.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
```