

# An IBM Proof of Technology

## SOA (Service Oriented Architecture)

IBM Johannesburg 8<sup>th</sup> – 10<sup>th</sup> July 2008

# Lab Exercises Part 1

PoT.WebSphere.07.4.031.02

© Copyright International Business Machines Corporation 2007, 2008. All rights reserved.

US Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.



Created for **TechWorks**





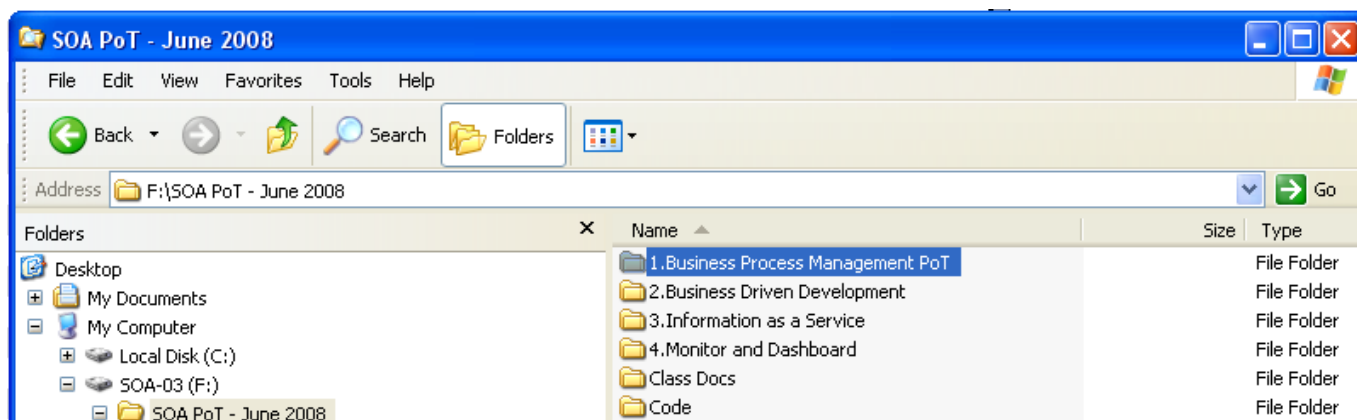
# Contents

<b>LAB PREPARATION .....</b>	<b>4</b>
<b>INTRODUCTION .....</b>	<b>6</b>
<b>LAB 1      PROMOTE STANDARDS AND INITIATE GOVERNANCE.....</b>	<b>8</b>
1.1      OPEN THE WEBSPHERE SERVICE REGISTRY AND REPOSITORY CONSOLE .....	8
1.2      SWITCH TO A DIFFERENT PROFILE .....	9
1.3      DEFINE HOW SERVICES SHOULD BE CLASSIFIED .....	11
1.4      LOAD THE STANDARD DATA OBJECT DEFINITIONS .....	13
1.5      DEFINE A NEW BUSINESS PROCESS AND APPLY GOVERNANCE .....	15
<b>LAB 2      MODEL THE BUSINESS PROCESS .....</b>	<b>19</b>
2.1      START THE WEBSPHERE BUSINESS MODELER .....	20
2.2      DEFINE THE BUSINESS MODEL .....	21
2.3      CONNECT TASKS TO DEFINE PROCESS FLOW .....	24
2.4      EXPORT THE BUSINESS MODEL AS A BPEL PROCESS.....	39
<b>LAB 3      APPLY GOVERNANCE TO THE PROCESS LIFECYCLE.....</b>	<b>42</b>
3.1      TRANSITION SIMPLEACCOUNTVERIFICATION PROCESS TO ASSEMBLE STATE .....	42
<b>LAB 4      ASSEMBLE AND DEPLOY THE PROCESS.....</b>	<b>52</b>
4.1      START THE WEBSPHERE INTEGRATION DEVELOPER.....	52
4.2      IMPORT THE BUSINESS MODEL INTO THE WEBSPHERE INTEGRATION DEVELOPER .....	54
4.3      ASSEMBLE THE BUSINESS PROCESS.....	55
4.4      IMPLEMENT THE TASKS OF THE BUSINESS PROCESS .....	60
4.5      DEPLOY AND TEST THE BUSINESS PROCESS .....	67
4.6      CLEANUP.....	72
<b>LAB 5      PUBLISH SERVICE INTERFACE FOR REUSE .....</b>	<b>73</b>
5.1      PUBLISH THE CREDIT REPORT SERVICE INTERFACE .....	73
<b>LAB 6      EXPLORE A WEB SERVICE ENDPOINT .....</b>	<b>77</b>
6.1      IMPORT THE COMPLETED WEB SERVICE .....	77
6.2      EXPLORE THE WEB SERVICE ARTIFACTS .....	79
6.3      PUBLISH THE WEB SERVICE .....	81
6.4      CHECK THE WEBSPHERE SERVICE REGISTRY AND REPOSITORY.....	82
<b>LAB 7      CHANGE SERVICE IMPLEMENTATION.....</b>	<b>85</b>
7.1      IMPORT THE WEB SERVICE WSDL FROM THE WEBSPHERE SERVICE REGISTRY AND REPOSITORY.....	85
7.2      CHANGE THE CREDIT REPORT TASK TO A WEB SERVICE IMPLEMENTATION.....	87
7.3      RETEST THE SIMPLEACCOUNTVERIFICATION PROCESS .....	90
7.4      CLEANUP.....	93
<b>LAB 8      EXTERNALIZE BUSINESS RULES .....</b>	<b>94</b>
8.1      CHANGE THE CREDIT RISK ASSESSMENT TASK TO A BUSINESS RULE .....	94
8.2      DEFINE THE BUSINESS RULE SET .....	96
8.3      SAVE AND VERIFY THE BUSINESS RULE .....	102
8.4      RETEST THE SIMPLEACCOUNTVERIFICATION PROCESS .....	102
8.5      MODIFY THE CREDIT ASSESSMENT BUSINESS RULE AT RUNTIME .....	105
8.6      RESET THE CREDIT RISK ASSESSMENT BUSINESS RULE.....	109
<b>LAB 9      MODIFY THE PROCESS CHOREOGRAPHY .....</b>	<b>111</b>
9.1      DEFINE THE INTERFACE FOR THE NEW 'GENERATE ACCEPTANCE' TASK .....	111
9.2      ADD THE GENERATEACCEPTANCE TASK TO THE SIMPLEACCOUNTVERIFICATION BPEL PROCESS.....	113
9.3      ADD THE IMPLEMENTATION FOR THE NEW GENERATE ACCEPTANCE TASK .....	120
<b>LAB 10     INCORPORATE HUMAN TASKS .....</b>	<b>129</b>
10.1     CREATE A HUMAN TASK TO START THE PROCESS .....	129
10.2     CONVERT THE PRICING AND APPROVAL JAVA COMPONENT TO A HUMAN TASK.....	131
10.3     GENERATE A WEB CLIENT FOR THE START PROCESS HUMAN TASK .....	132
10.4     GENERATE A WEB CLIENT FOR THE PRICING AND APPROVAL HUMAN TASK .....	134

10.5	RETEST THE SIMPLEACCOUNTVERIFICATION PROCESS .....	135
10.6	USE THE HUMAN TASK CLIENT TO START THE PROCESS .....	136
10.7	VIEW THE PROCESS STATE .....	137
10.8	USE THE HUMAN TASK CLIENT TO PERFORM PRICING AND APPROVAL .....	139
10.9	CLEANUP .....	140
<b>LAB 11</b>	<b>ADD NEW SERVICE PROVIDERS .....</b>	<b>141</b>
11.1	PUBLISH THE WSDLs FOR THE EXTERNAL CREDIT REPORT WEB SERVICES .....	141
<b>LAB 12</b>	<b>PROVIDE FLEXIBILITY WITH AN ENTERPRISE SERVICE BUS .....</b>	<b>144</b>
12.1	IMPORT THE CREDIT REPORT WEB SERVICES .....	144
12.2	IMPORT WSDLs FROM WEBSHERE SERVICE REGISTRY AND REPOSITORY .....	146
12.3	CREATE A MEDIATION MODULE .....	147
12.4	IMPLEMENT THE MEDIATION FLOW .....	148
12.5	DEFINE THE ROUTING LOGIC TO THE EXTERNAL EQUINOX CREDIT REPORT SERVICE .....	153
12.6	COMPLETE THE CREDITREPORTSERVICEROUTER ASSEMBLY DIAGRAM .....	157
12.7	RE-ASSEMBLE THE SIMPLEACCOUNTVERIFICATION TO USE THE CREDITREPORTSERVICEROUTER .....	161
12.8	RETEST THE SIMPLEACCOUNTVERIFICATION PROCESS .....	163
12.9	CLEANUP .....	166
<b>LAB 13</b>	<b>ENABLE DYNAMIC SERVICE INVOCATION .....</b>	<b>167</b>
13.1	ADD CUSTOM PROPERTIES TO THE WEB SERVICES .....	167
13.2	SPECIFY THE CLASSIFICATION OF THE WEB SERVICES .....	169
13.3	CREATE THE DYNAMIC CREDIT REPORT SERVICE MEDIATION MODULE .....	170
13.4	IMPLEMENT THE MEDIATION FLOW .....	171
13.5	DEFINE THE SERVICE ENDPOINT LOOKUP CRITERIA .....	173
13.6	COMPLETE THE DYNAMICCREDITREPORTSERVICE ASSEMBLY DIAGRAM .....	177
13.7	RE-ASSEMBLE THE SIMPLEACCOUNTVERIFICATION PROCESS TO USE THE DYNAMICCREDITREPORTSERVICE MODULE .....	179
13.8	RETEST THE SIMPLEACCOUNTVERIFICATION PROCESS .....	180
13.9	CHANGE THE CLASSIFICATION TO CHANGE THE SERVICE SELECTION .....	182
13.10	RETEST THE SIMPLEACCOUNTVERIFICATION PROCESS .....	185
13.11	DYNAMICALLY ADD TWO NEW CREDIT REPORT SERVICES AND HAVE THESE SERVICES AVAILABLE FOR THE PROCESS AT RUNTIME .....	186
13.12	RETEST THE SIMPLEACCOUNTVERIFICATION PROCESS .....	190
13.13	CLEANUP .....	194
<b>CONCLUSION</b>	<b>.....</b>	<b>195</b>
APPENDIX A.	LOADING LAB SOLUTIONS .....	196
APPENDIX B.	IBM TECHWORKS .....	202
APPENDIX C.	NOTICES .....	203
APPENDIX D.	TRADEMARKS AND COPYRIGHTS .....	205

# WebSphere Business Process Management Enabled by SOA

Uses VMWare Image 1



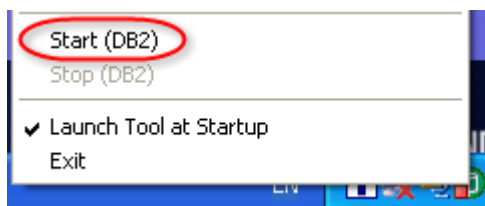
Start the “1.Business Process Management PoT” image, this will be used for days 1 & 2 of the workshop

## Lab Preparation

1. In the lower right corner of your screen, verify that the Database Manager is currently running by checking that the DB2 icon does not have a red mark.



If not running, then right-click on the DB2<sup>®</sup> icon. From the popup menu, select **Start (DB2)**.



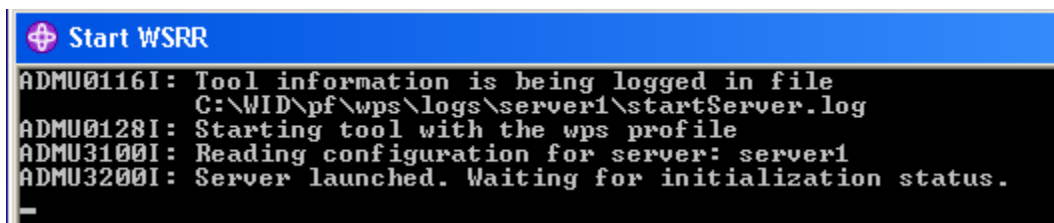
\_\_2. From the desktop, open the **PoT Lab Shortcuts** folder.



\_\_3. Double-click on the **Start Servers** icon in the folder.



A Command Prompt window will appear.



```

Start WSRR
ADMU0116I: Tool information is being logged in file
           C:\WID\pf\wps\logs\server1\startServer.log
ADMU0128I: Starting tool with the wps profile
ADMU3100I: Reading configuration for server: server1
ADMU3200I: Server launched. Waiting for initialization status.
  
```

This will start the required server for the labs. The Command Prompt window will automatically close when the server has started.

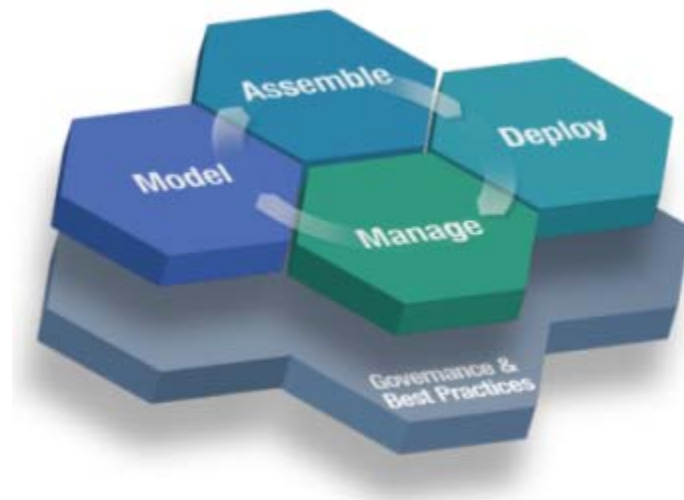
## Introduction

In this lab exercise, you will be implementing a business process from the ground up for verifying accounts. This business process will be called SimpleAccountVerification, and will be used to process requests to open accounts. This business process will consist of activities such as performing credit checks, risk assessments, approvals, and pricing plan selections.

### The main objectives include:

- Business agility
- Development flexibility
- Runtime dynamicity
- Easier integration
- Governance
- Reuse of assets
- Risk Reduction

The IBM® SOA Foundation and the WebSphere® Business Process Management software portfolio will be used to achieve these goals. The focus will be on the Model, Assemble, Deploy, and Manage phases of the SOA lifecycle.



You will use the WebSphere Business Modeler to define the business model, and then WebSphere Integration Developer for application assembly. The WebSphere Process Server built into the WebSphere Integration Developer will be used for deployment and testing. You will also use the WebSphere Service Registry and Repository for governance, service metadata and reuse.

Once the business process has been implemented and deployed, you will also perform several iterative changes to the process to illustrate how quickly this can be accomplished. This type of business flexibility is one of the primary benefits of SOA.





### Important!

If you encounter problems during the course of this lab, please call the attention of the class instructor or any of the lab assistants.

## Scenario




Your organization has several core business processes that need to be significantly improved. These processes are composed of business activities which are currently manual tasks, or implemented using standalone systems. These processes are currently not implemented end-to-end, which significantly limits business flexibility.

In this lab exercises, you have been assigned to integrate a very simple business process for handling account applications. This will be called the **SimpleAccountVerification** process. This is your organization's first SOA project, so we will keep it simple and incremental. This will involve a very rudimentary process of determining whether a request to open an account will be approved or rejected. This generic example can apply to loan applications, revolving credit applications, or other similar scenarios.

As you go through the different labs, you will be playing different roles, such as a business analyst, architect, integration developer, and web service developer.

## Icons

The following symbols appear in this document at places where additional guidance is available.

Icon	Purpose	Explanation
	Important!	This symbol calls attention to a particular step or command. For example, it might alert you to type a command carefully because it is case sensitive.
	Information	This symbol indicates information that might not be necessary to complete a step, but is helpful or good to know.
	Troubleshooting	This symbol indicates that you can fix a specific problem by completing the associated troubleshooting information.

## Lab 1 Promote Standards and Initiate Governance

### Goals:

- **Ensure success of SOA projects**
  - **Initiate lifecycle governance**
  - **Reuse assets and promote standards**
- **Align business with IT**
  - **Define classifications**

### Roles: Enterprise Architect, Administrator

You are a member of the enterprise architecture team tasked with enforcing enterprise-level standards, such as common service interfaces and business object definitions. This assumes that the necessary SOA analysis and design activities have already been performed, and these efforts have produced artifacts capturing standard definitions. In this lab, you will publish several of these artifacts.

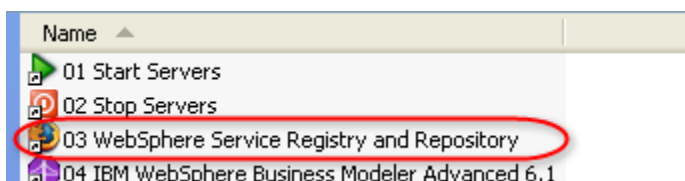
First, you'll publish an XML Schema Definition (XSD) file containing the following standard data objects:

Customer	Basic customer information
CustomerApplication	Application submitted by the customer as part of the loan request

Next, you'll load the approved interface for Credit Report web services. Then, you'll create a standard classification of Credit Report services to indicate whether a Credit Report service is internal or external.

### 1.1 Open the WebSphere Service Registry and Repository console

- \_\_\_1. Switch to the **PoT Lab Shortcuts** folder. Double-click on the **WebSphere Service Registry and Repository** link. Reopen the folder from the desktop if needed.



The default web browser appears showing the WebSphere Service Registry and Repository Console login screen.



### Hint

The URL for the WebSphere Service Registry and Repository is <http://localhost:9080/ServiceRegistry>.

- \_\_2. If a Security Alert warning appears, click on **Yes** to proceed.



### Troubleshooting

If the WebSphere Service Registry and Repository console does not appear, or you encounter web page errors, wait for the Command Prompt window started earlier to automatically close. This will indicate that the servers have started.

## 1.2 Switch to a different profile



### What next?

The WebSphere Service Registry and Repository allows you to set up different profiles for specific needs. Only one profile can be active. In the next steps, you will switch from the default profile to the sample governance profile.

- \_\_3. From the Perspective drop-down list, select **Configuration**, and then click on the **Go** button beside it.



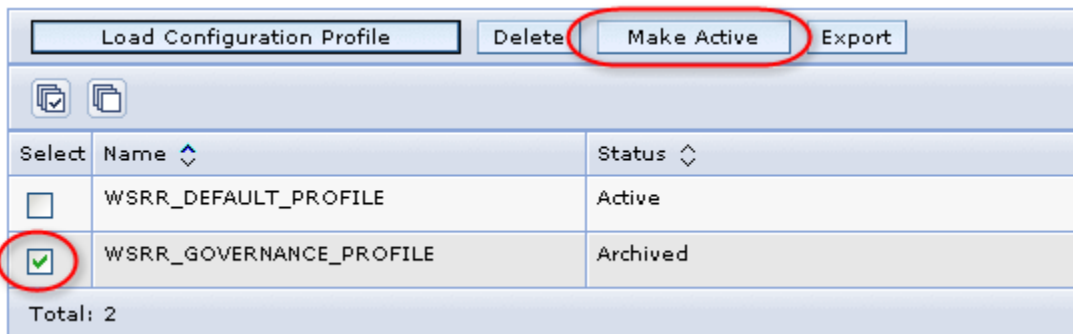
### Hint

Profiles contain their own set of perspectives. Notice that the list only shows three perspectives to choose from. These are the perspectives available in the currently active profile. Aside from being able to create your own profile, you can also create your own perspectives for the different roles in your organization, such as for an Administrator, Business Analyst, Architect, Developer, and User.


\_\_4. Expand the **Manage Configuration Profiles** section. Click on **Configuration Profiles**.



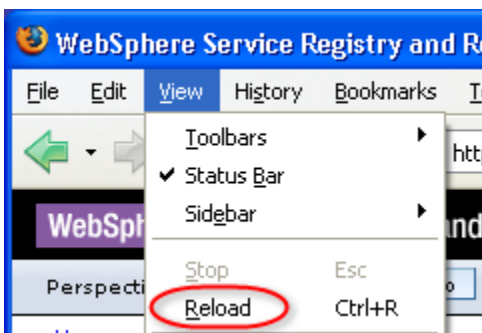
\_\_5. Select **WSRR\_GOVERNANCE\_PROFILE**. Click on **Make Active**.



**Hint**

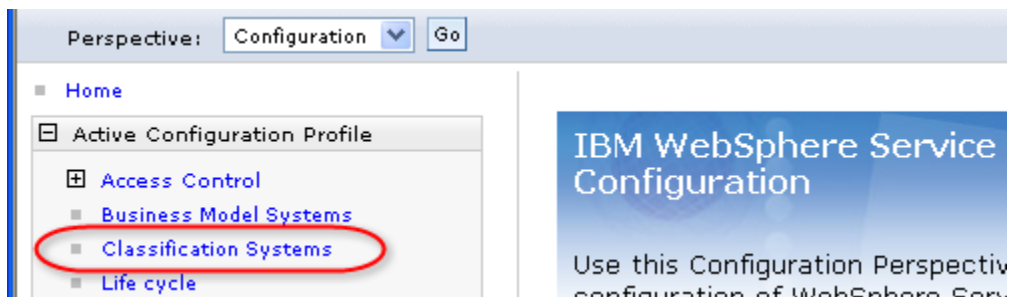
 You have switched to a sample profile (WSRR\_GOVERNANCE\_PROFILE) that is provided with the product to help you get started quickly. The default profile is intended for those who want to start from the ground up. The sample Governance profile contains a well-defined service model, with sample definitions of templates, lifecycles, validators, classifications, roles and perspectives.

\_\_6. Refresh the browser so that the new active profile will be reflected in the web page.

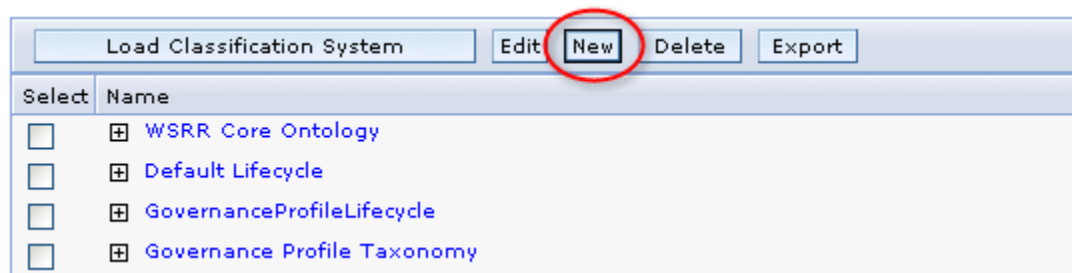


### 1.3 Define how services should be classified

\_\_7. Expand the **Active Configuration Profile** section. Click on **Classification Systems**.



\_\_8. Click on **New**.

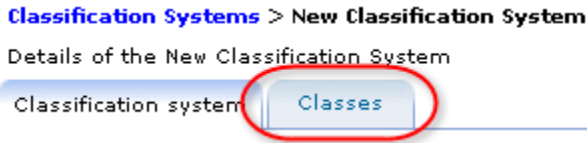


\_\_9. Enter the following values:

Field	Value
URI	http://www.mycompany.com/Taxonomy
Name	Credit Report Services Visibility

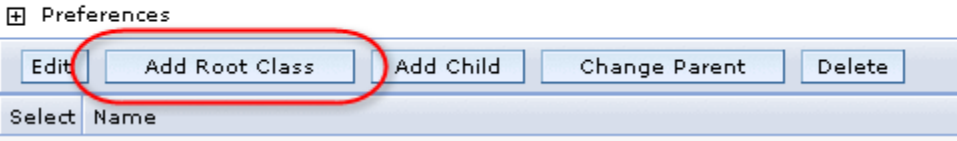


\_\_10. Click on the **Classes** tab.



\_\_11. Click on **Add Root Class**.

Select a class to Add a child, change its parent, Delete or Edit it. Add a Root Class using the 'Add Root Class' Button.



\_\_12. Enter the following values. Just append **Internal** to the Class ID.

Field	Value
Class ID	http://www.yourcompany.com/Taxonomy# <b>Internal</b>
Name	Internal

Enter the details for the new root class.

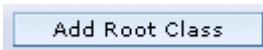
Class ID

Name

Comment

\_\_13. Click on **OK**.

\_\_14. Click on **Add Root Class** again.

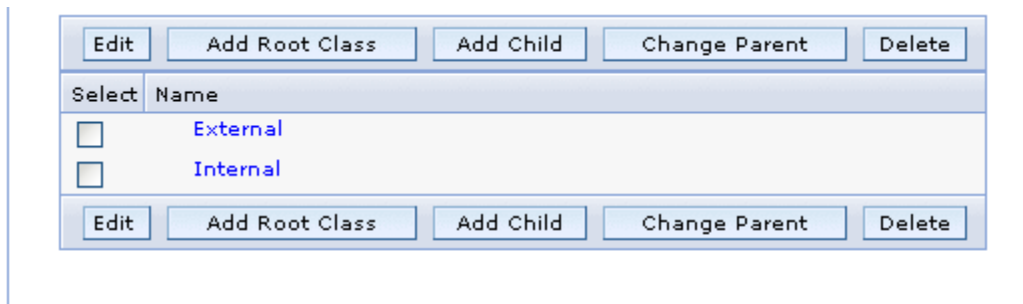



\_\_15. Enter the following values. Just append **External** to the Class ID.

Field	Value
Class ID	http://www.yourcompany.com/Taxonomy#External
Name	External

\_\_16. Click on **OK**.

\_\_17. Click on **Save and Commit**.


**What did you just do?**

A new classification has been defined. The Credit Report services that will be loaded in later labs can be classified as either an Internal or External service.

## 1.4 Load the standard data object definitions



**What next?**

Your organization has decided to use standard data objects for Customer and CustomerApplication information.

\_\_18. From the Perspective drop-down list, select **GP Architect**. Click on the **Go** button.



Notice that the new active profile now provides more perspectives for specific roles.

- \_\_19. Scroll to the bottom of the web page. Click the **Browse** button and select **C:\PoT\BPMSOA\Businessitems.xsd**.

**Load Documents** ?

This facility enables you to load one or more documents, with the option to save them as a group. Specify a file to load, select a document type and, optionally, enter a description and a version.

**Path to the Document**


Local file system

Specify path

C:\PoT\BPMSOA\Businessitems.xsd

Remote file location

Specify URL

 The file Businessitems.xsd contains the standard definition of the Customer and CustomerApplication business objects.

- \_\_20. For the Document type, select **XSD**. Click on **OK**.

Document type

XSD

Enter document description:

Enter document version:

- \_\_21. Click on **Finish**.

**Documents to be Loaded**

When all required documents are listed below select either 'Finish' to complete th want to refer to these documents as a document group.

**Businessitems.xsd** (ready to load) | [Remove](#) | [Replace](#) |



A message will appear indicating that the document was loaded successfully.

#### Documents Loaded Successfully

The following documents have been loaded into the repository:

Name	Description	Namespace
<a href="#">Businessitems.xsd</a>		http://Businessitems



#### What just happened?

The Businessitems.xsd has been successfully loaded in the WebSphere Service Registry and Repository. The standard Customer and CustomerApplication data objects defined in the Businessitems.xsd file are now ready to be reused by other projects.

## 1.5 Define a new business process and apply governance

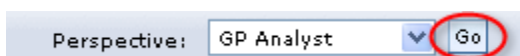


#### What next?

The goal of these labs is to develop and deploy a business process called **SimpleAccountVerification**. Before you start modeling and assembling this process, you will first need to define the SimpleAccountVerification process in the WebSphere Service Registry and Repository. This is to let other business units or teams in your organization know about your plans so they can consider reuse and set up subscriptions and notifications. This will also allow governance to be applied to the lifecycle of the process to ensure that certain requirements are satisfied before the process can move through the different phases of its lifecycle.

### Role: Business Analyst

- \_\_22. Let's now assume the role of a business analyst. Select the **GP Analyst** perspective, and then click on **Go**.



- \_\_23. Expand the **Business Metadata** and **Business Service View** sections. Click on **Business Processes**.



\_\_24. Click on **New**.

**Business Processes**

This is the collection of business processes present in the registry.

⊕ Preferences

\_\_25. For the Name, specify **SimpleAccountVerification**. Click on **OK**.

Entity Properties	Entity Metadata
<p>Name</p> <input type="text" value="SimpleAccountVerification"/>	<ul style="list-style-type: none"> <li>Classifications</li> </ul>
<p>Description</p> <input type="text"/>	
<p>Version</p> <input type="text"/>	
<p>Business owner</p> <input type="text"/>	
<p>Cost per Transaction</p> <input type="text"/>	

\_\_26. Click on **SimpleAccountVerification**.

Select	Name	Graph	Description	Version
<input type="checkbox"/>	SimpleAccountVerification			

Total: 1

\_\_27. Click on the **Governance** tab.

**Business Processes > SimpleAccountVerification**

Details of the SimpleAccountVerification Business Process.

Details | Impact Analysis | **Governance**

Entity Properties	Entity Metadata
<p>Name</p> <input type="text" value="SimpleAccountVerification"/>	<ul style="list-style-type: none"> <li>Graphical View</li> <li>Properties</li> </ul>

\_\_28. Select **InitiateInterfaceLifecycle**. Click on **Govern**.

**Governance Status**


This object is not currently governed. Choose the initial click the button to make it governed.

Initial state transitions

InitiateInterfaceLifecycle


Govern

**Hint**

 You actually selected the wrong lifecycle type. However, there is already a governance validation mechanism in place to ensure that the correct lifecycle is selected. This will result in an error message.

\_\_29. Expand the governance validation message by clicking the + sign.

Messages

 A problem occurred while attempting to make the object governable. Reason: GSR1427E: The Governance Policy Validator has encountered one problem.

GSR1427E: The Governance Policy Validator has encountered one problem.

GSR1400E: AnyOfAssertion: ServiceInterfaceLifecycleEntityAssertion: ServiceInterface Lifecycle must be on a Service Binding, ServiceInterface, Message Schema or XSDDocument. None of the assertions contained within the AnyOfAssertion passed

\_\_30. Click on the web browser **Back** button.



\_\_31. Select **InitiateServiceLifecycle**. Click on **Govern**.

**Governance Status**

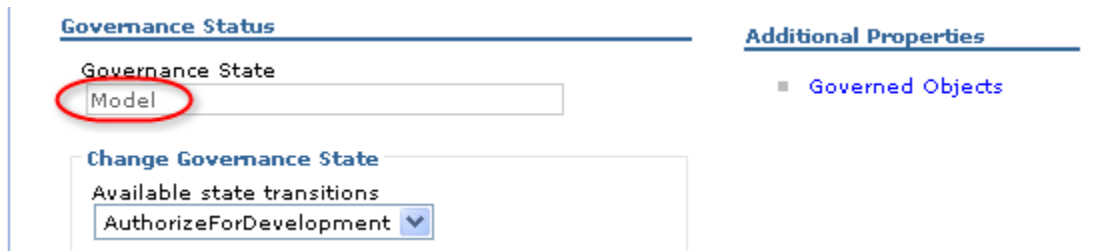
This object is not currently governed. Choose the initial and click the button to make it governed.

Initial state transitions

InitiateServiceLifecycle

Govern

Notice that the SimpleAccountVerification process is now in the Model state.



**Governance Status**

Governance State  
Model

**Change Governance State**

Available state transitions  
AuthorizeForDevelopment

**Additional Properties**

- Governed Objects



What are the different states in each lifecycle? The next steps will illustrate this.

\_\_32. Expand **Governance Lifecycle View**. Expand **Governed Entities**. Expand **Service Lifecycle**.



[-] Governance Lifecycle View

- [-] Governed Entities
  - Governed - No Lifecycle
  - [-] Service Lifecycle
    - Model
    - Assemble
    - Deploy
    - Manage
    - Retired
  - [-] Interface Lifecycle
  - [-] Service Version Lifecycle
  - [-] Policy Lifecycle

This shows the different states or phases in the Service Lifecycle. You can also view the other Lifecycles that have been defined in this sample profile.



**What next?**

You are now ready to model the SimpleAccountVerification process.



**Please wait for the next lecture before proceeding to the next lab.**

## Lab 2 Model the Business Process

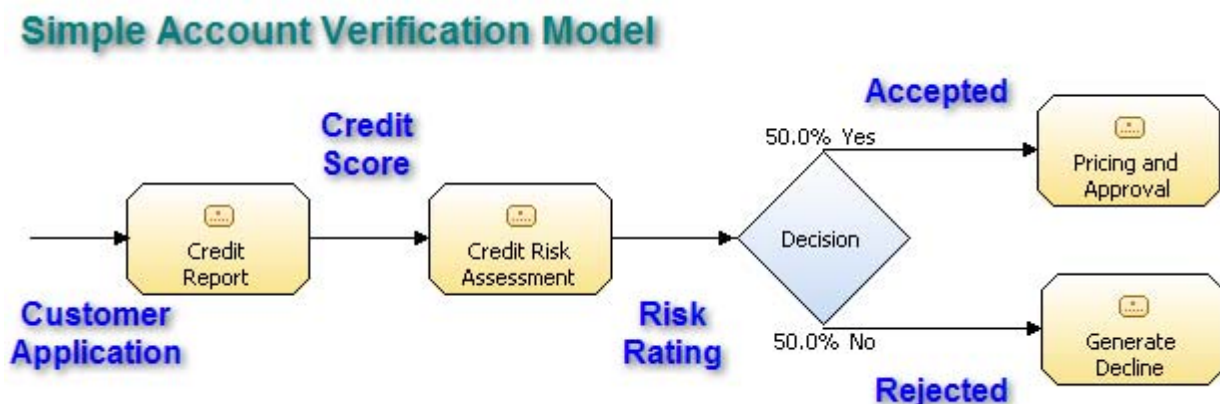
### Goals:

- **Align IT to business processes**
  - **Initiate business-driven development**
  - **IT organized as business tasks and services**
- **Capture more precise business models**
- **Lower business costs**
  - **Run simulations before allocating time and resources**

### Role: Business Analyst

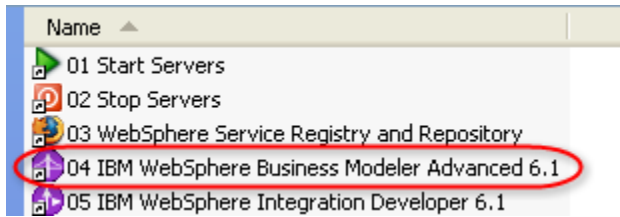
In this Model phase, you will assume the role of a Business Analyst. You have been given the responsibility of documenting an existing business process which handles account verifications. This Account Verification process is part of a larger Account Open process which allows customers to submit requests to open an account. The customer submits the application form with the required information, and then a credit report is performed to determine the applicant's credit score. A credit risk assessment is then completed based on the credit score, which will result in a risk rating of either HIGH or LOW. A request with a high risk rating will automatically be rejected. A request with a low risk rating will be approved and a pricing plan will be determined.

The simple model you will build is illustrated in the following screenshot:



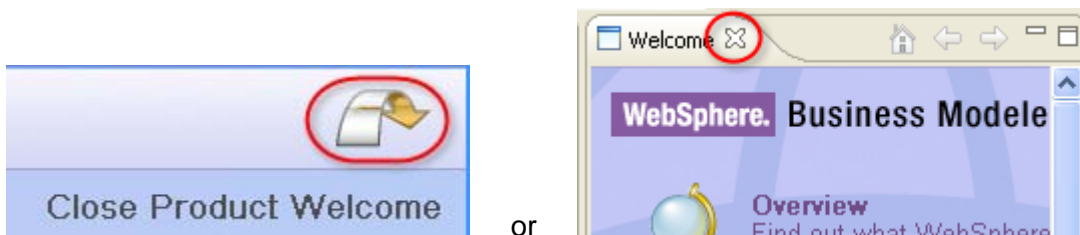
## 2.1 Start the WebSphere Business Modeler

1. Switch to the **PoT Lab Shortcuts** folder. Double-click on **04 IBM WebSphere Business Modeler Advanced 6.1**. Reopen the folder from the desktop if needed.



The WebSphere Business Modeler will appear.

2. Close the Product Welcome view.



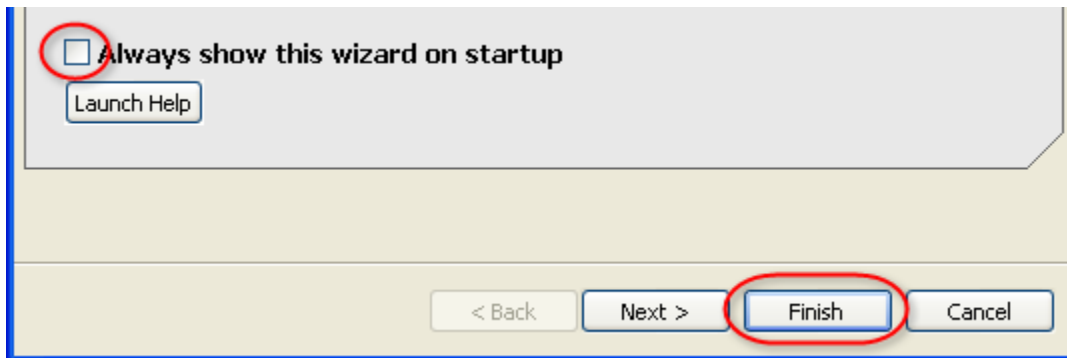
3. On the **Quickstart Wizard**, specify the following.

Project Name	SimpleAccountVerification
Process Name	SimpleAccountVerification

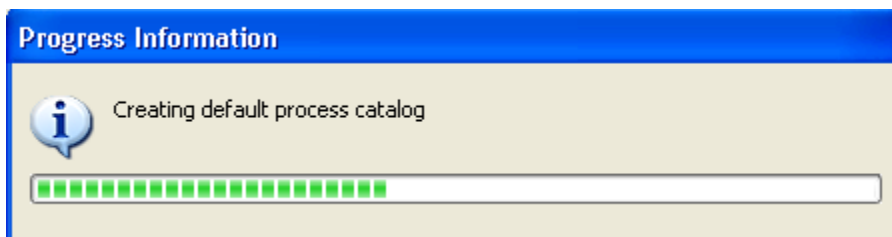
Use the default for the **Process catalog name** field.

<b>Project name</b>	SimpleAccountVerification
<b>Process catalog name</b>	Processes
<b>Business process name</b>	SimpleAccountVerification

\_\_4. Uncheck the **Always show this wizard on startup** option. Click on **Finish**.



\_\_5. Wait for the progress indicator to disappear.

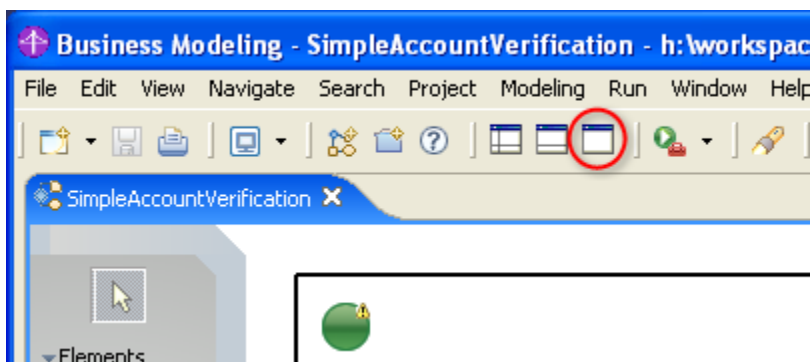


\_\_6. Maximize the **WebSphere Business Modeler** window if needed.



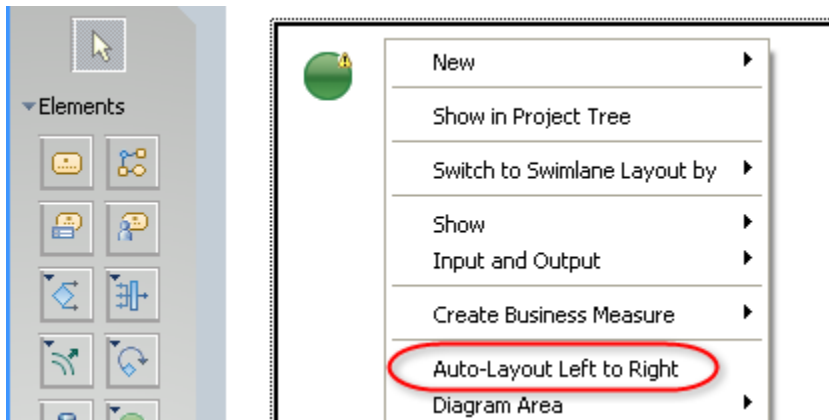
## 2.2 Define the business model

\_\_7. From the toolbar, click on the **Apply 1-pane layout** button.



This will simplify the layout and hide the advanced features and views for now.


\_\_8. In the canvas or white-space area, **right-click** and then select **Auto-Layout Left to Right**.



This will compress all the default elements and make them easier to find. By default, the business model will initially have **Process start** and **Process stop** elements positioned in the corners of the canvas, but should now be visible.

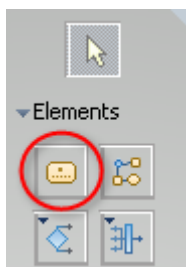
\_\_9. Select the two default process elements and press the **Delete** key. You will not need these elements for the business model you are creating.





**Troubleshooting**  
You can undo any incorrect actions by either pressing **Ctrl-z**, or selecting **Edit->Undo** from the menu bar.

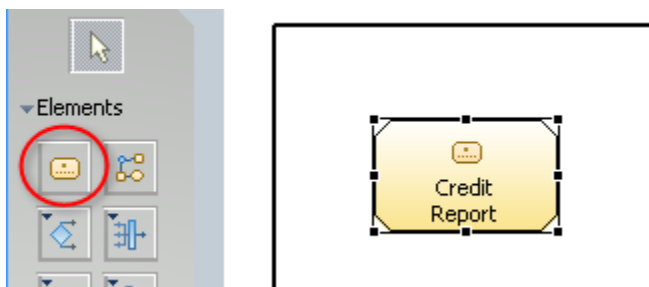
\_\_10. From the palette on the left, select the element **Create local task**.




This element will represent a business task in the business process you are diagramming. Keep in mind that this element typically should not represent programs, applications, databases, or any other software component, but rather high-level activities composing a business process.

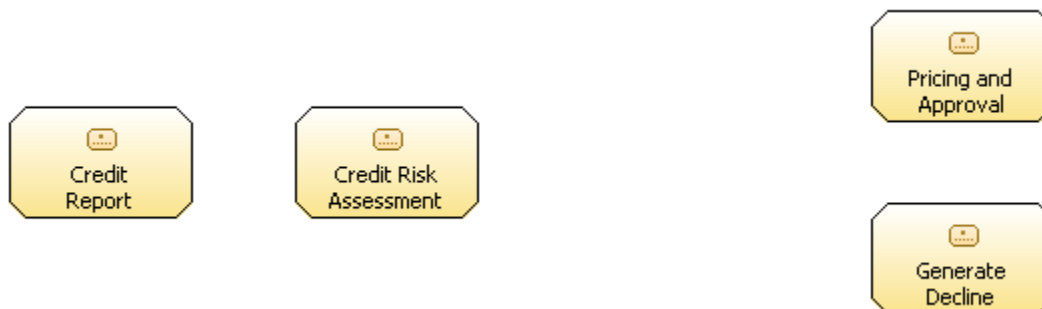


- \_\_11. Click on the canvas to drop the local task element. By default, the name of the element should be in edit mode. Label the element as **Credit Report**. (You do not need to drag and drop.)



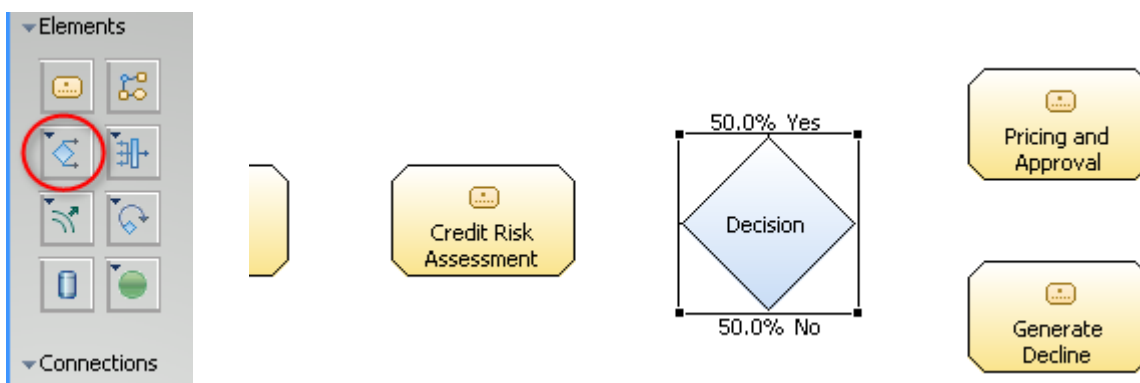
**Tip**  
 To rename the element, just highlight the element, wait 1-2 seconds, then click on the element again. The label will be in edit mode.

- \_\_12. Select three more local task elements and drop them into the canvas. Rename these new elements as **Credit Risk Assessment**, **Pricing and Approval** and **Generate Decline**. Position the elements similar to the screenshot.

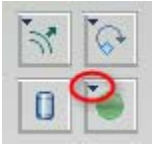


These four elements will represent the four tasks in our simple account verification scenario. These tasks will later be implemented as invocations to applications or external systems.

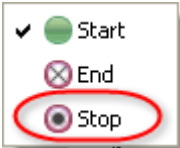
- \_\_13. From the palette, select the element **Create simple decision** and drop it in the middle of the three new elements.



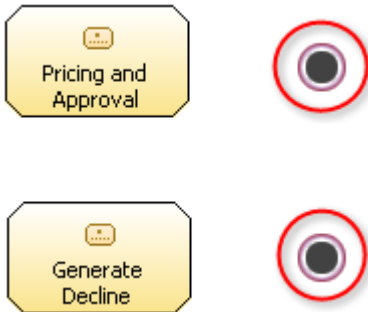
\_\_14. From the palette, click on the small arrow on the element **Create start** to display a submenu of elements.



\_\_15. Select the element **Stop**.



\_\_16. Drop to the right of both the **Pricing and Approval** element and the **Generate Decline** element.



\_\_17. Click on **Ctrl+s** to save your current progress.

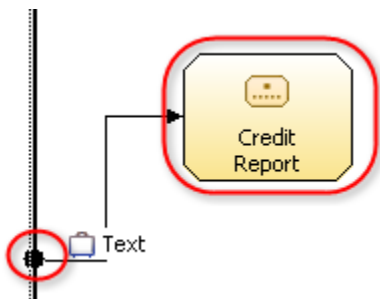
### 2.3 Connect tasks to define process flow

\_\_18. From the palette, click the **Connections** button.



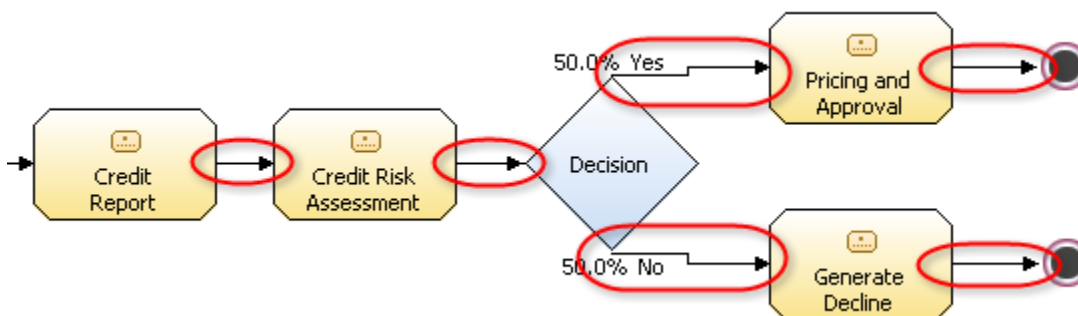
This will allow you to start defining the flow through the different tasks to represent the business process flow.

\_\_19. Click on the left border of the canvas, and then click on the **Credit Report** task.



This connection indicates that the **Credit Report** task will be invoked first and will be passed some text information as a parameter.

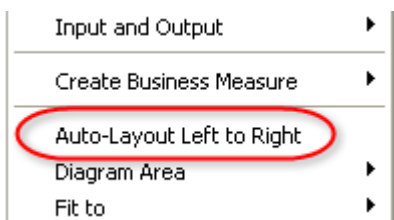
\_\_20. Complete the following connections:



\_\_21. From the palette, click on the **Select** button to switch out of connection mode.



\_\_22. Right-click on an empty space in the canvas. From the popup menu, select **Auto-Layout Left to Right**.



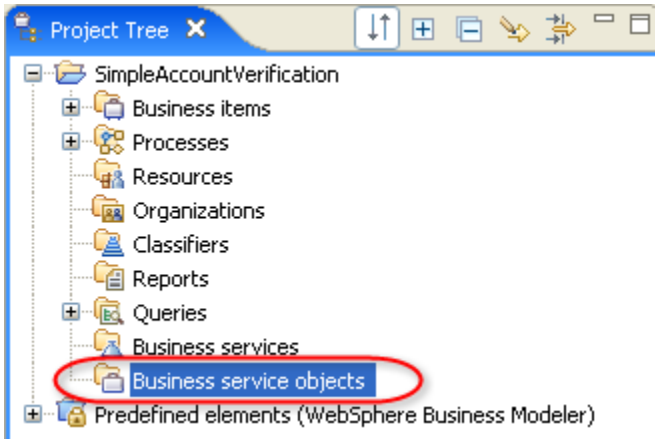
The wiring of all the business tasks to define the process flow is now complete. You will now configure the business item that will be used by the business tasks.

\_\_23. Click on **Ctrl+s** to save your current progress.

\_\_24. Close the **SimpleAccountVerification** Model editor.



\_\_25. From the **Project Tree** view, expand the elements in the **Project** list until you can select **SimpleAccountVerification -> Business service objects**.

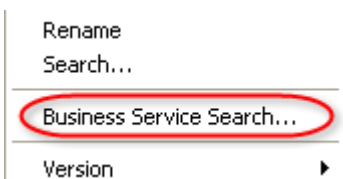


#### What next?

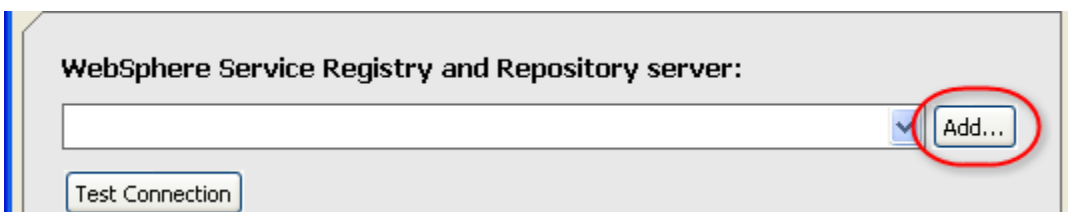


The next step is to reuse the standard business objects defined in the Businessitems.xsd file previously loaded into the WebSphere Service Registry and Repository. In this SimpleAccountVerification process, the customer information required for verification will be stored in the standard business object CustomerApplication. This CustomerApplication object definition will be retrieved from the WebSphere Service Registry and Repository and stored in the Business service objects folder highlighted above.

\_\_26. From the Project Tree, right-click on the **Business service objects** folder. From the popup menu, select **Business Service Search**.



\_\_27. Click on **Add**.




\_\_28. Specify **wrrpot** in the Name field. Accept the defaults for the other fields.



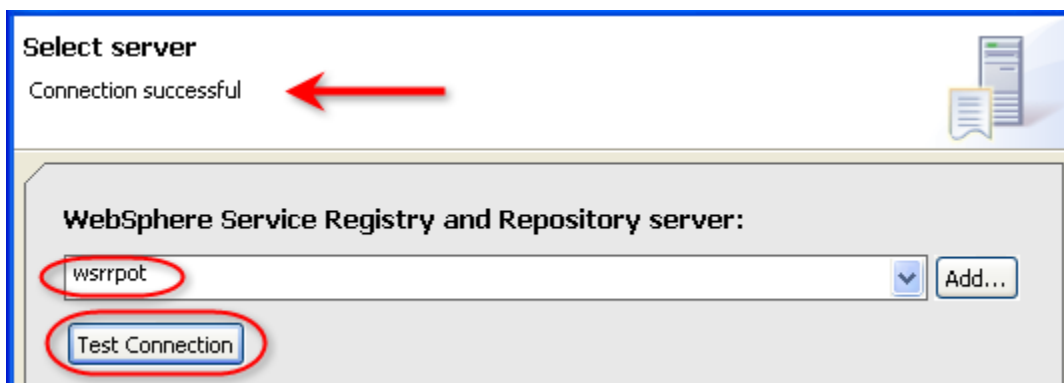
\_\_29. Click on **OK**.

**What just happened?**



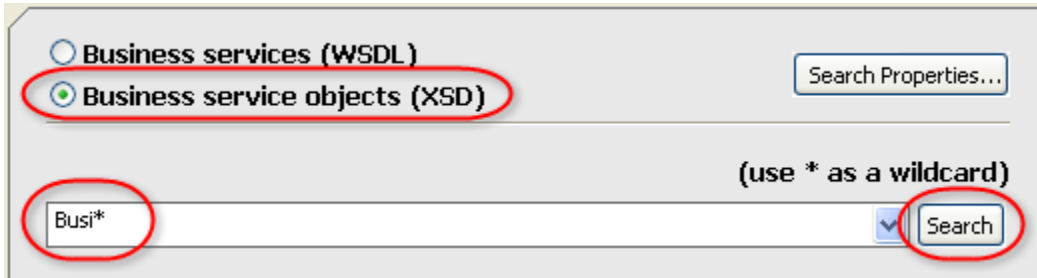
You created a connection to the WebSphere Service Registry and Repository server, and specified an alias name of **wrrpot**. This will allow you to find and publish artifacts to the server from within this development environment.

\_\_30. Select **wrrpot**, and then click on **Test Connection**. Verify that a “Connection successful” message is displayed.

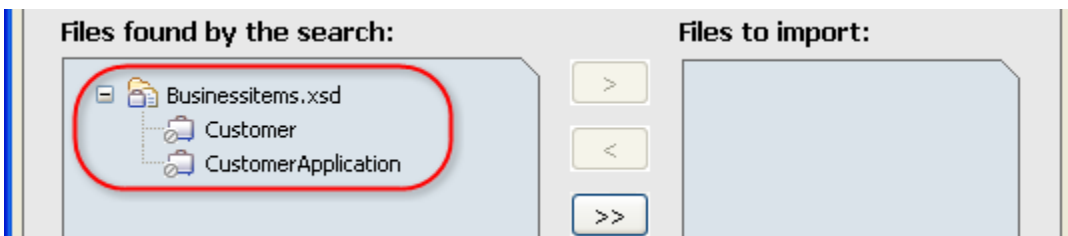


\_\_31. Click on **Next**.

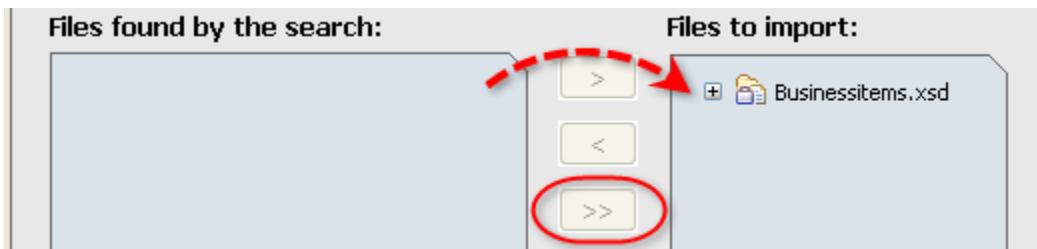
\_\_32. Specify **Busi\*** as the Search criteria (case-sensitive). Select the **Business service objects (XSD)** option, and then click on **Search**.



\_\_33. The file **Businessitems.xsd** is found. Expand **Businessitems.xsd**.

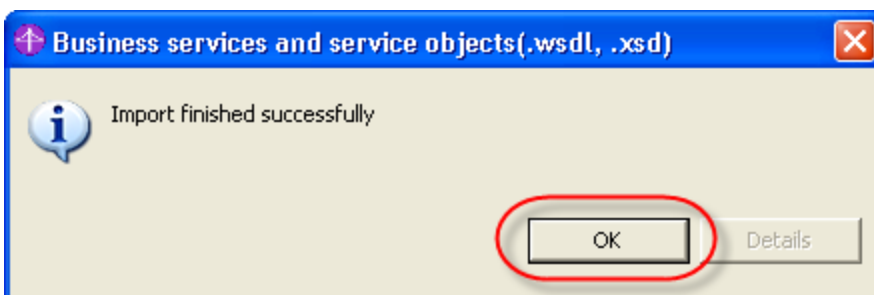


\_\_34. Select the All button  to move **Businessitems.xsd** to the **Files to import** column.



\_\_35. Click on **Finish**.

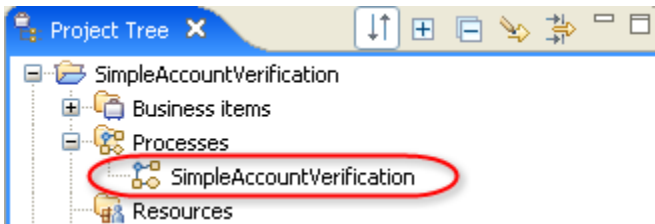
\_\_36. From the Import status information window, click on **OK**.



**What just happened?**

The Businessitems.xsd has been imported, and the CustomerApplication and Customer objects can now be reused in the SimpleAccountVerification business model.

\_\_37. From the Project Tree view, double-click on **SimpleAccountVerification -> Processes -> SimpleAccountVerification**.

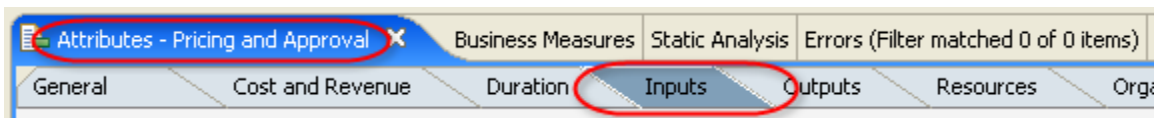


This will open the business model you created earlier.

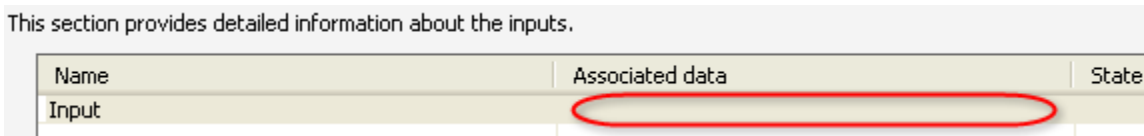
\_\_38. Select the **Pricing and Approval** element.



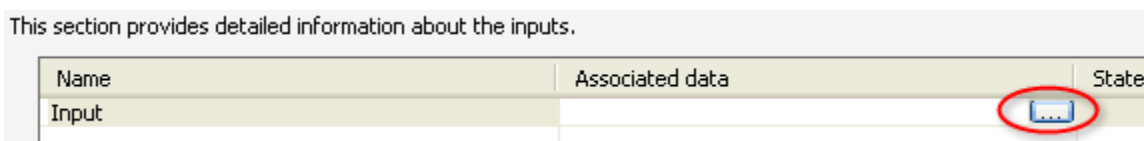
\_\_39. On the Properties area at the bottom section of the screen, select the **Attributes - Pricing and Approval** tab, and then select the **Inputs** tab.



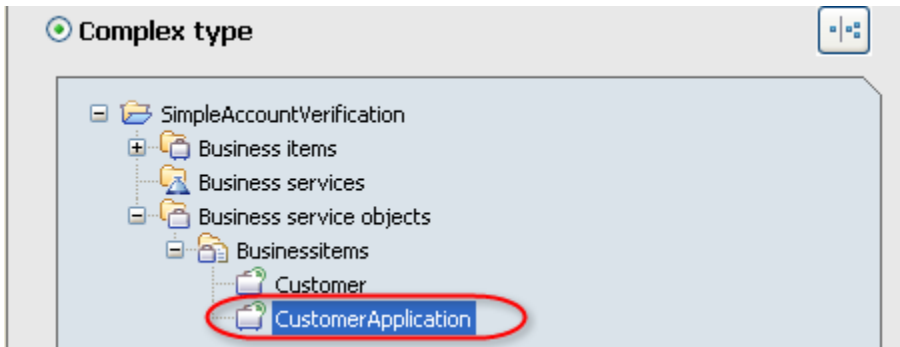
\_\_40. Click on the **Associated data** column.



\_\_41. Click on the **Browse** button. 



\_\_42. Expand **SimpleAccountVerification** to select **CustomerApplication**.



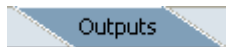
\_\_43. Click on **OK**.

\_\_44. Verify that the Associated data is now set to **CustomerApplication**.

This section provides detailed information about the inputs.

Name	Associated data	State
Input	CustomerApplication	

\_\_45. Select the **Outputs** tab.



\_\_46. Click on the **Associated data** column.

This section provides detailed information about the outputs.

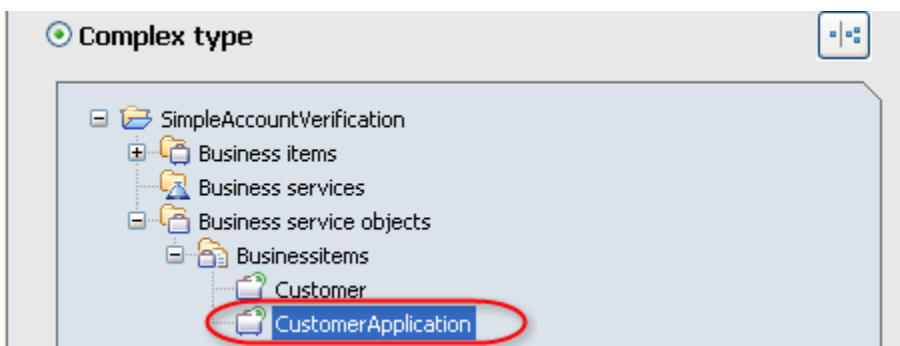
Name	Associated data	State
Output		

\_\_47. Click on the **Browse** button.

This section provides detailed information about the outputs.

Name	Associated data	State
Output		...

\_\_48. Expand **SimpleAccountVerification** to select **CustomerApplication**.





\_\_49. Click on **OK**.

\_\_50. Verify that the Associated data is now set to **CustomerApplication**.

This section provides detailed information about the outputs.

Name	Associated data	State
Output	CustomerApplication	



**What just happened?**

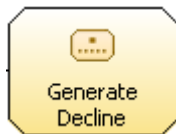
You just specified that the CustomerApplication object will be used for both the input and output of the **Pricing and Approval** task.



**What's next?**

You will now also specify the CustomerApplication object as the input and output for the **Generate Decline** task.

\_\_51. Select the **Generate Decline** element.



\_\_52. On the Properties area at the bottom section of the screen, select the **Attributes - Generate Decline** tab, and then select the **Inputs** tab.

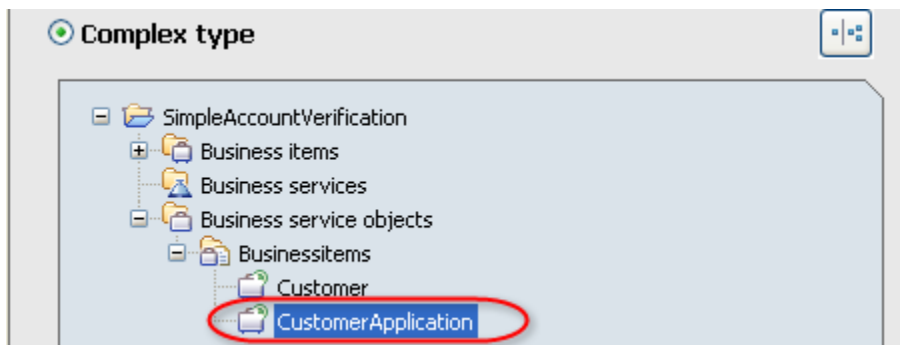
\_\_53. Click on the **Associated data** column.

This section provides detailed information about the inputs.

Name	Associated data	State
Input		

\_\_54. Click on the **Browse** button. 

\_\_55. Expand **SimpleAccountVerification** to select **CustomerApplication**.



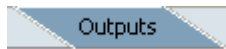
\_\_56. Click on **OK**.

\_\_57. Verify that the Associated data is now set to **CustomerApplication**.

This section provides detailed information about the inputs.

Name	Associated data	State
Input	CustomerApplication	

\_\_58. Select the **Outputs** tab.



\_\_59. Click on the **Associated data** column.

\_\_60. Click on the **Browse** button. 

\_\_61. Expand **SimpleAccountVerification** to select **CustomerApplication**.

\_\_62. Click on **OK**.

\_\_63. Verify that the Associated data is now set to **CustomerApplication**.

This section provides detailed information about the outputs.

Name	Associated data	State
Output	CustomerApplication	

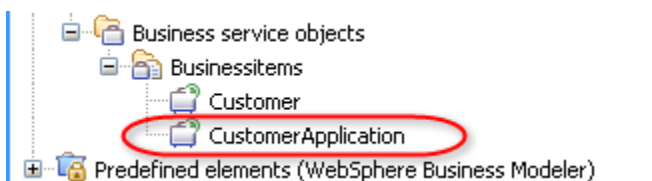


**What just happened?**

You also specified the CustomerApplication object as the input and output for the **Generate Decline** task.

\_\_64. Click on **Ctrl+s** to save your current progress.

\_\_65. From the Project Tree view, expand the **Business service objects** folder. Double-click on **CustomerApplication**.

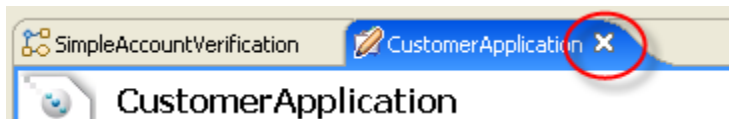


This will open the **CustomerApplication** business object view.

Name	Type
applicationID	Text
applicationDate	Date
pricing	Text
requestedLimit	Decimal (single-precision)
+ customer	Customer
creditScore	Integer
creditReportRequired	Boolean
applicationDecision	Boolean
creditRiskAssessment	Text

This is the data structure of the **CustomerApplication** business object. As you can see, it also contains a nested **Customer** business object.

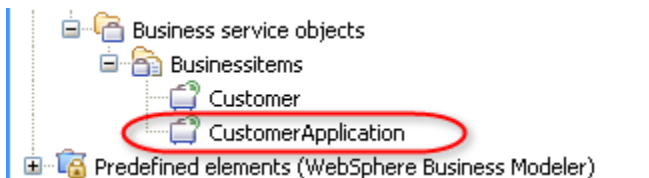
\_\_66. Close the **CustomerApplication** business object view.



#### What's next?

You will now use the drag and drop approach for specifying the input and output objects of the other tasks.

\_\_67. From the Project Tree view, select **CustomerApplication**.

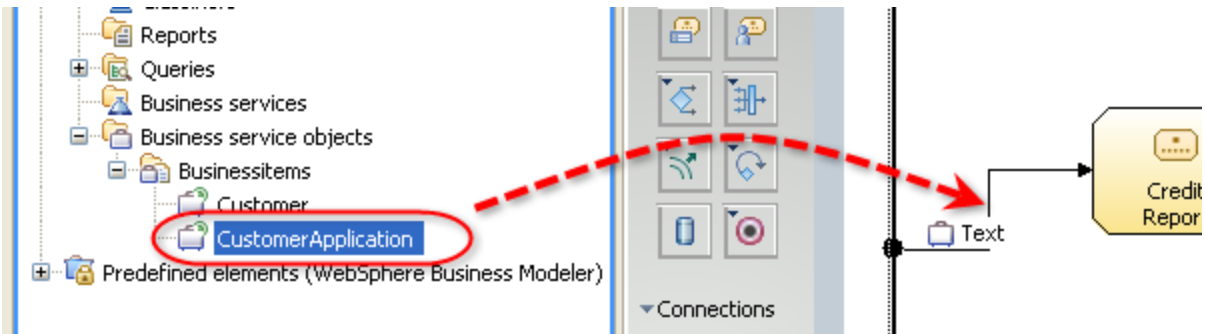


\_\_68. Drag and drop the **CustomerApplication** business object on the connection between the left border of the canvas and the element **Credit Report**.

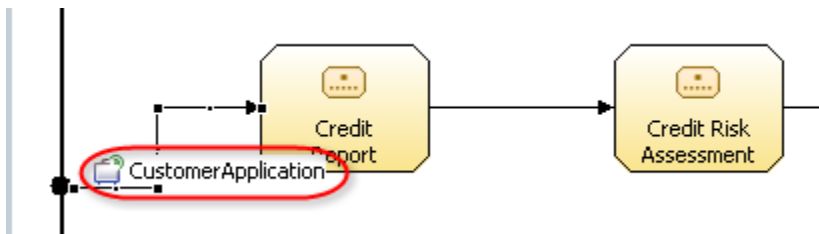


#### Tip

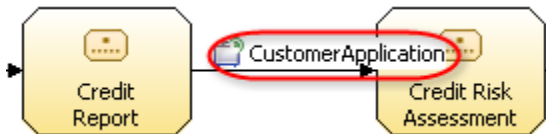
A dotted box appears when the object is ready to be dropped on the connection.



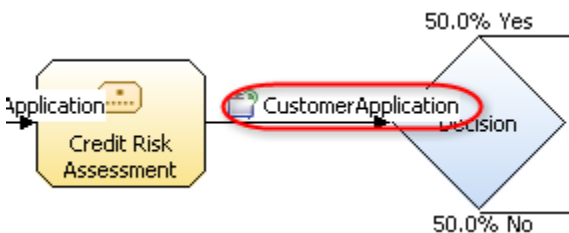
This drag and drop action sets up the business item **CustomerApplication** as the input to the activity **Credit Report**. The connection should now look similar to this:



\_\_69. Drag and drop the **CustomerApplication** business object on the connection between the Credit Report task and the Credit Risk Assessment task.




\_\_70. Drag and drop the **CustomerApplication** business object on the connection between the Credit Risk Assessment task and the Decision element.

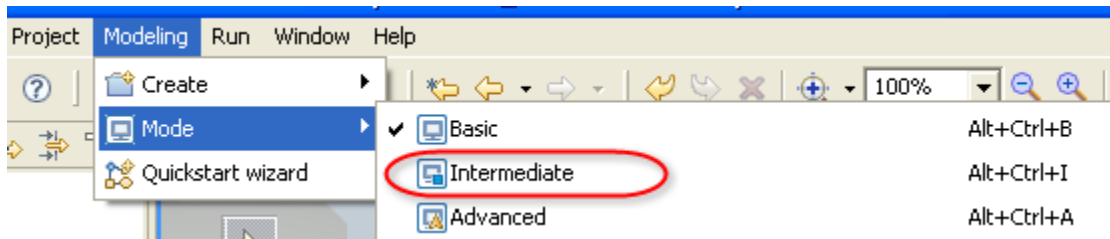


\_\_71. Click on **Ctrl+s** to save your current progress.


**What's next?**

 You will now specify the logic for the decision element to determine whether the request will be approved or rejected.

\_\_72. From the menu bar, select **Modeling->Mode->Intermediate**.

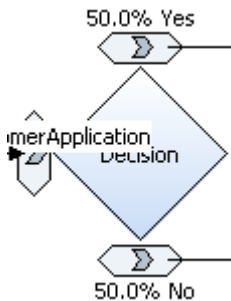


**Tip**

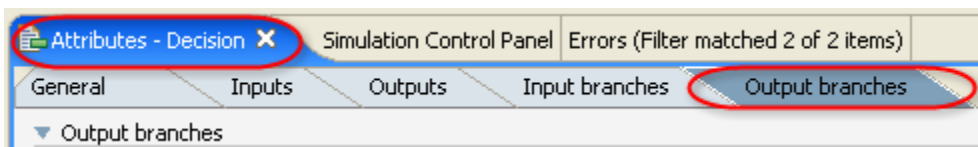


This mode will provide more advanced tooling capabilities. Specifically, this will allow you to provide logic to the Decision element. This will also display more information on the business model diagram.

\_\_73. From the business model editor, select the **Decision** element.



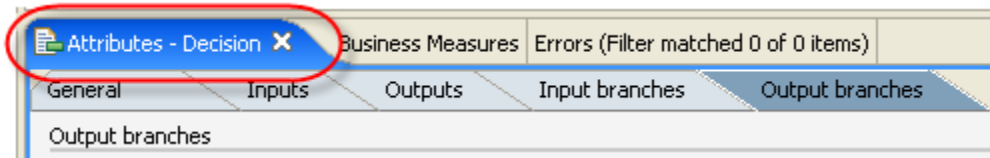
\_\_74. On the **Properties** area at the bottom section of the screen, click on the **Attributes - Decision** tab, and then the **Output branches** tab.



\_\_75. Select the row for the **Yes** output branch.

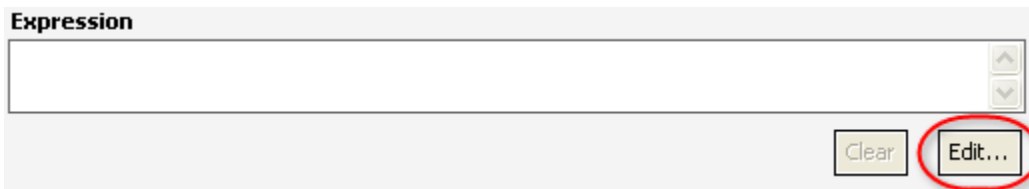
Name	Contents
Yes	Output
No	Output:2

\_\_76. Double-click on the **Attributes - Decision** view titlebar to maximize the view.



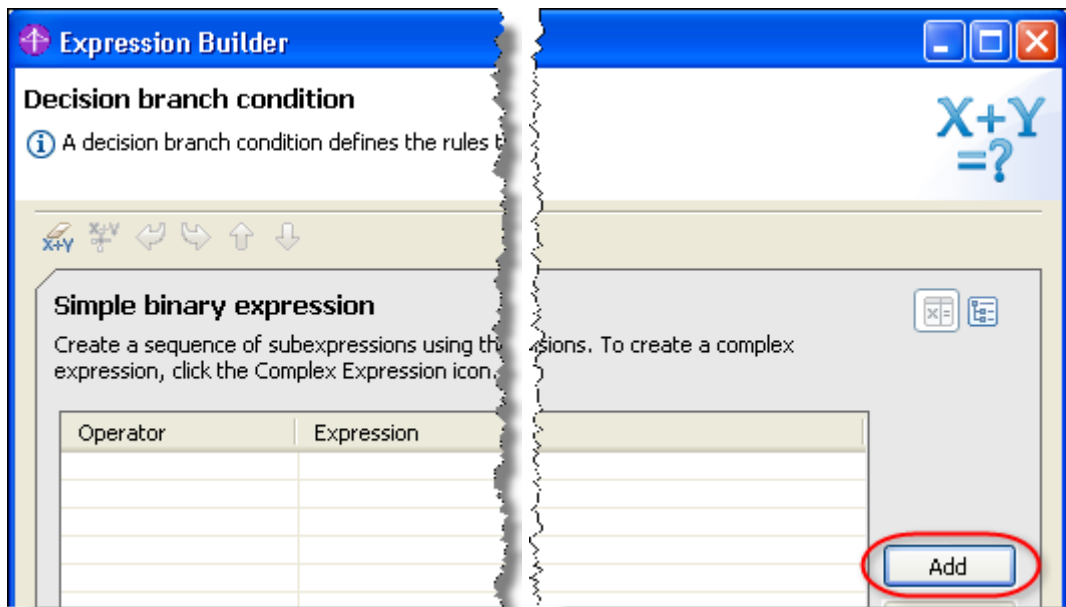
**i Tip**  
 You can maximize any view by double-clicking on its titlebar. Double-clicking again on the titlebar will restore the view to its original size.

\_\_77. On the lower-right corner of the view, click on **Edit**. Scroll down if necessary.

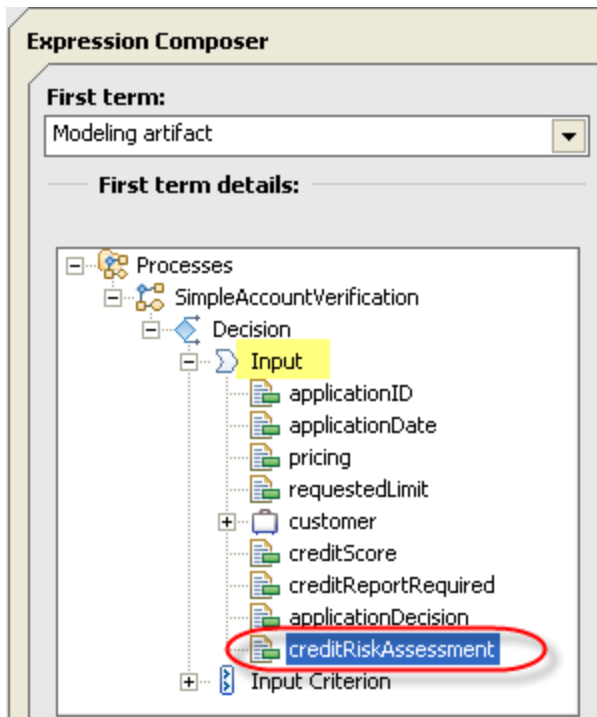


The Expression Builder window will appear.

\_\_78. Click on **Add**.



- \_\_79. In the **Expression Composer** section, expand the **First term details** tree list, and then select **creditRiskAssessment**. Ensure that you are selecting **creditRiskAssessment** from the **Input** parent branch, and not the **Input Criterion** branch.

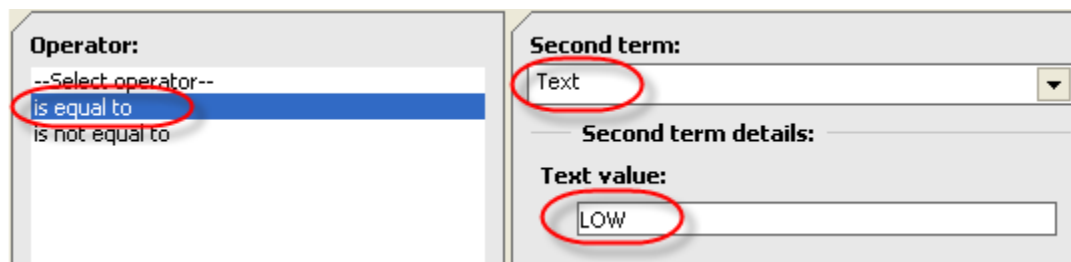


**Important!**

Ensure that you are selecting **creditRiskAssessment** from the **Input** parent branch, and not the **Input Criterion** branch.

- \_\_80. Make the following selections:

Operator	is equal to
Second term	Text
Text value	LOW



\_\_81. Click on **Apply**. 

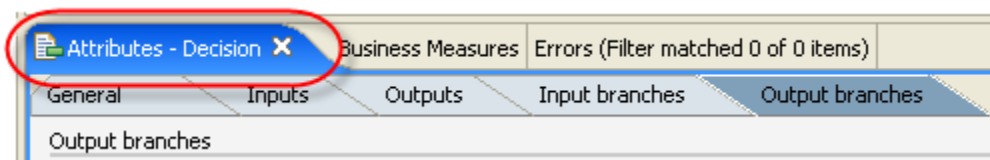
\_\_82. Ensure that the **Expression text** contains the following expression.

```
Expression text
'Processes.SimpleAccountVerification.Decision.Input.creditRiskAssessment' is equal to "LOW"
```

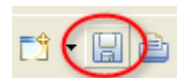
\_\_83. Click on **OK**.

This will also automatically create an inverse expression for the **No** output branch.

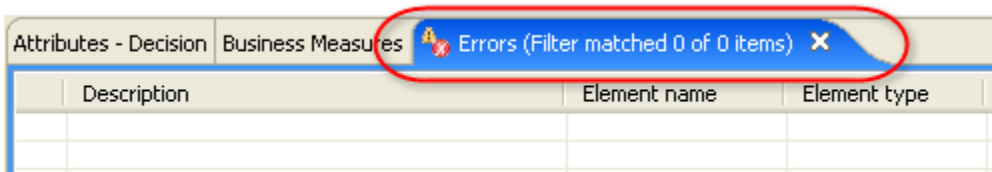
\_\_84. Double-click on the **Attributes - Decision** view titlebar to restore the view to its original size.



\_\_85. Press **Ctrl+s** or click on the **Save** button on the toolbar.



\_\_86. Click on the **Errors** tab and ensure that there are no errors listed.



**What just happened?**



You have just completed modeling a simple account verification process. For the sake of simplicity, you just used the default names for the task operations and the input and output variables. The operation names will typically default to **InputCriterion**. Also by default, input variables will typically be named **input1**, and output variables will be named **output1**.

**What's next?**



The business model will now be exported to the file system as a BPEL version of the model. In the next lab, this BPEL model will be imported into the WebSphere Integration Developer for assembly.



**Troubleshooting**

If the business model was not completed successfully, or was not exported successfully, then you can still proceed to the next lab. A solution version of the model is available if needed.



## 2.4 Export the business model as a BPEL process

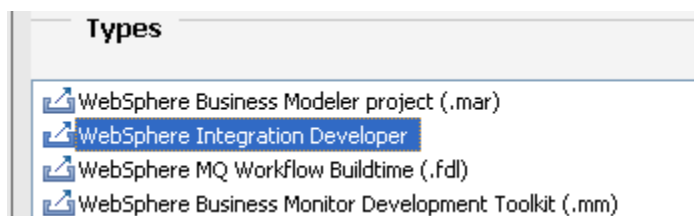
\_\_87. From the menu bar, select **File -> Export**.

\_\_88. From the **Export** window, expand **WebSphere Business Modeler**. Select **WebSphere Business Modeler Export**.



\_\_89. Click on **Next**.

\_\_90. From the next window, select **WebSphere Integration Developer**.



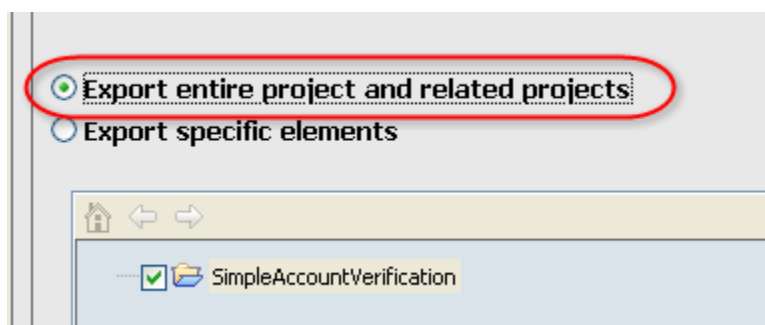
Selecting **WebSphere Integration Developer** as the export type will export the business model as a BPEL process and stored in a Project Interchange file.

\_\_91. Click on **Next**.

\_\_92. Specify a Target directory of **c:\lab**.

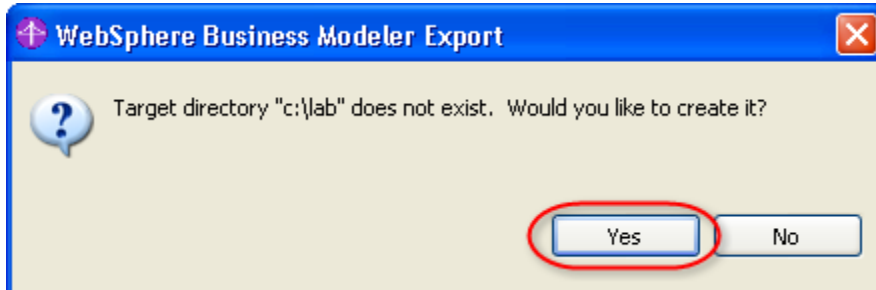


\_\_93. Select the option **Export entire project and related projects**.

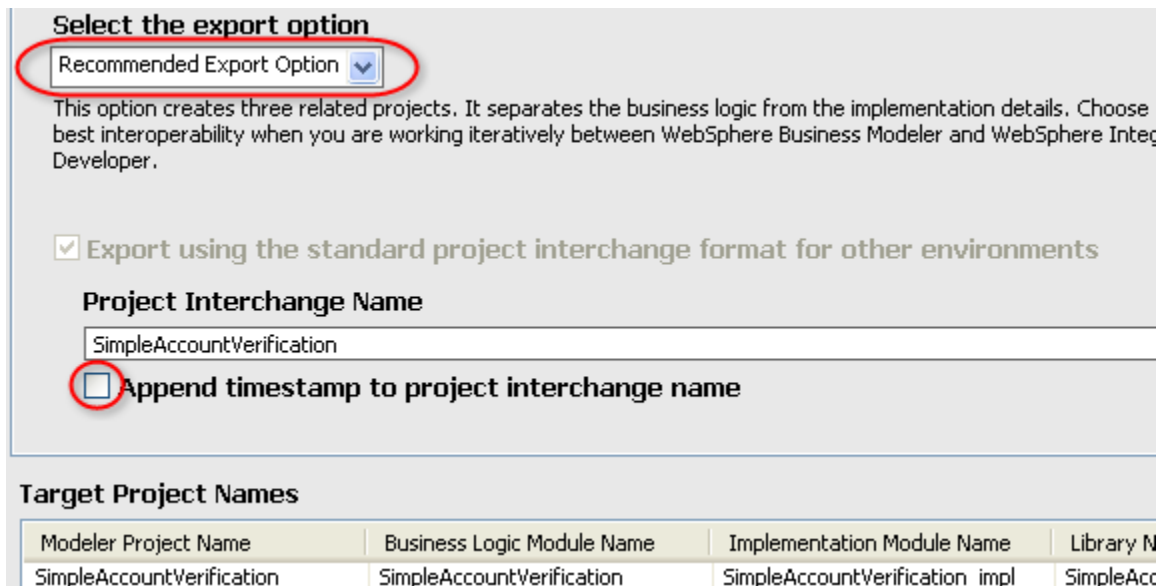


\_\_94. Click on **Next**.

\_\_95. If a window appears asking if you would like to create the target directory, click on **Yes**.

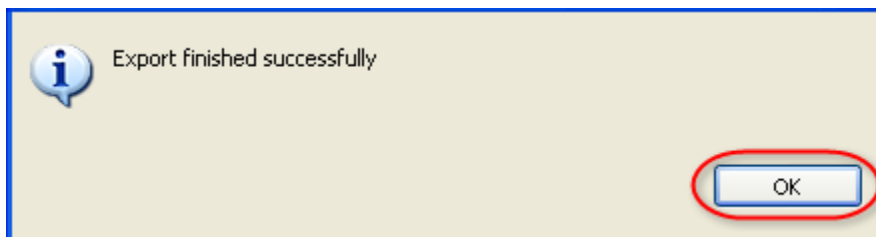


\_\_96. Ensure that the **Recommended Export Option** is selected. Uncheck the **Append timestamp** option.



\_\_97. Click on **Finish**.

\_\_98. From the **Export finished** window, click on **OK**.



**What just happened?**

The business model has now been exported into the file system as a BPEL process contained in a Project Interchange file. It is ready for import into the WebSphere Integration Developer.

\_\_99. From the main menu, select **File -> Exit** to close the WebSphere Business Modeler.

The “**Model**” phase of our SOA project is now complete.



**Please wait for the next lecture before proceeding to the next lab.**

## Lab 3 Apply Governance to the Process Lifecycle

### Goals:

- **Ensure success through lifecycle governance**
  - **Transition from *model* phase to *assemble* phase**

**Role: Architect or Project Manager**


The business model for the SimpleAccountVerification process has been completed and is ready for implementation and assembly. However, you made the process governable in an earlier lab because you need the proper validation and control mechanisms to ensure the success of this SOA project. This involves ensuring that certain requirements are met before the SimpleAccountVerification process can transition from the model phase to the assembly phase. You will now attempt to transition the process to the assembly phase and see how governance can be applied to the process lifecycle.

### 3.1 Transition SimpleAccountVerification Process to Assemble State

- \_\_1. Switch to the web browser showing the **WebSphere Registry and Repository** console.

**Troubleshooting**

If the WebSphere Service Registry and Repository console was closed, just reopen using the link in the PoT Lab Shortcuts folder.



Name
01 Start Servers
02 Stop Servers
03 WebSphere Service Registry and Repository
04 IBM WebSphere Business Modeler Advanced 6.1

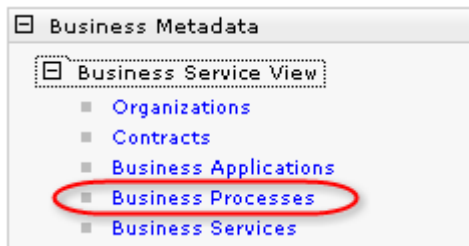
You can also open a web browser and go to <http://localhost:9080/ServiceRegistry>.

- \_\_2. Select **GP Architect** from the Perspective drop-down list. Click on the **Go** button.



The screenshot shows the WebSphere Service Registry and Repository console. The 'Perspective' dropdown menu is open, showing a list of perspectives: Administrator, Configuration, GP Administrator, GP Analyst, **GP Architect** (highlighted with a red circle), and GP Developer. The 'Go' button is visible next to the dropdown. The main content area displays 'IBM WebSphere Service Registry Configuration'.

- \_\_3. Expand the **Business Metadata** and **Business Service View** sections. Click on **Business Processes**.




- \_\_4. Click on **SimpleAccountVerification**.

Select	Name	Graph	Description	Version
<input type="checkbox"/>	SimpleAccountVerification			
Total: 1				

- \_\_5. Click on the **Governance** tab.



**What's next?**

 You will now transition the SimpleAccountVerification process from the “Model” phase to the “Assemble” or implementation phase. You will then need to satisfy certain requirements before the governance validators will allow you to perform the transition.

- \_\_6. Click on **Transition**.

Governance State

**Change Governance State**

Available state transitions

A governance validation error message appears.

\_\_7. Expand the governance validation message by clicking the + sign.

Messages


**A problem occurred while attempting to transition the state of the object. Reason: GSR1428E: The Governance Policy Validator has encountered 2 problems.**

GSR1428E: The Governance Policy Validator has encountered 2 problems.

GSR1420E: RelationshipAssertion: OwingOrganizationAssertion: **Owning Organization must be specified for Entity.** There are not enough target objects on this relationship. The minimum is 1

GSR1424E: Classification Assertion: BusinessDomainAssertion: **Entity must be assigned to a Business Domain.** Entity was not classified correctly to match query /WSRR/BaseObject/[classifiedByAnyOf ('http://www.ibm.com/xmlns/prod/serviceregistry/6/1/GovernanceProfileTaxonomy#BusinessDomain')]


**What just happened?**



A Governance Validator exists as part of this sample profile to ensure that the following requirements are satisfied before the SimpleAccountVerification process will be authorized for development and transitioned to the Assemble state:

- Owning organization must be specified
- Must be assigned to a Business Domain classification

\_\_8. Click on **Organizations**.



A screenshot of a tree view under 'Business Metadata'. The tree structure is: Business Metadata > Business Service View > Organizations (highlighted with a red circle) > Contracts > Business Applications > Business Processes.

\_\_9. Click on **New**.

\_\_10. Specify a Name of **Finance**. Click on **OK**.

**Entity Properties**

---

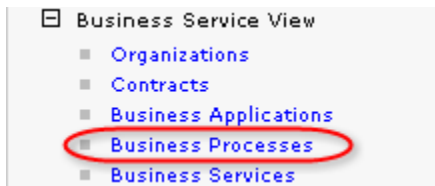
Name

Description

Contact Name

Contact Email address

\_\_11. Click on **Business Processes**.




\_\_12. Click on **SimpleAccountVerification**.

Select	Name	Graph	Description	Version
<input type="checkbox"/>	SimpleAccountVerification			

Total: 1

**What's next?**



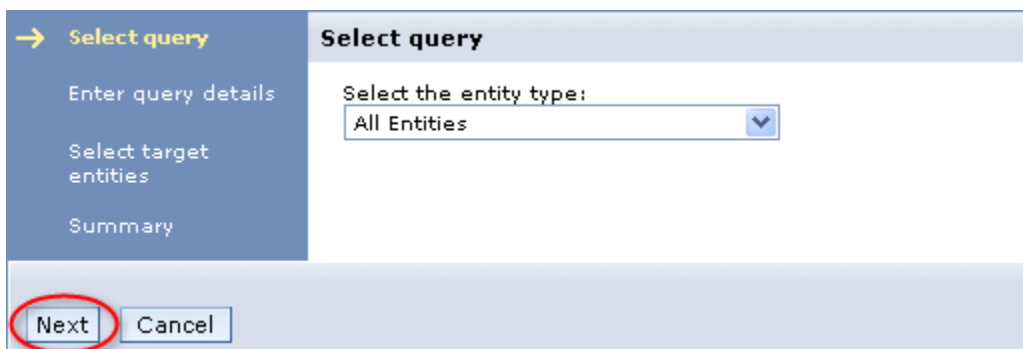
You just defined the Finance department as an organization in WebSphere Service Registry and Repository. The next step is to specify that the Finance department owns the SimpleAccountVerification process by defining a relationship. To define a relationship, you search for the target of the relationship with the SimpleAccountVerification process.

\_\_13. Click on **Owning Organization**.



\_\_14. Click on **Add**.

\_\_15. Click on **Next**.



\_\_16. Specify **Fin\*** as the search criteria. Click on **Next**.

\_\_17. Select **Finance**. Click on **Next**.

Select	Name	Description	Object Type
<input checked="" type="checkbox"/>	Finance		concept
Total:1			

\_\_18. Click on **Finish**.

Another governance validation message appears indicating that the Finance Organization must first be governed before it can be specified as the Owning Organization of the process.

Messages

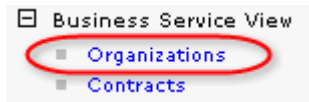
An error occurred attempting to add targets to a relationship named "wsrrgp\_owningOrganization" on the entity.

GSR1427E: The Governance Policy Validator has encountered one problem.

GSR1424E: Classification Assertion: GovernedOrganizationAssertion: **Owning Organization must be Governed** before being added. Entity was not classified correctly to match query /WSRR/BaseObject/[classifiedByAllOf ('http://www.ibm.com/xmlns/prod/serviceregistry/6/1/GovernanceProfileLifecycle#State')]



\_\_19. Click on **Organizations**.



\_\_20. Click on **Finance**.

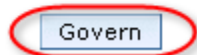
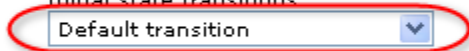
\_\_21. Click on the **Governance** tab. 

\_\_22. Select the **Default transition**. Click on **Govern**.

#### Governance Status

This object is not currently governed. Choose the initial state transition for this object and click the button to make it governed.

Initial state transitions

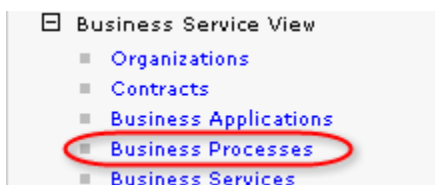


The Governance State should now be **LifecycleSelection**. This means that the next transition step is to select the proper governance lifecycle profile, but this will not be needed for this lab.

Governance State

LifecycleSelection

\_\_23. Click on **Business Processes**.



\_\_24. Click on **SimpleAccountVerification**.

Select	Name	Graph	Description	Version
<input type="checkbox"/>	SimpleAccountVerification			
Total: 1				

\_\_25. Click on **Owning Organization**.

**Entity Relationships**

- Documentation
- **Owning Organization**
- Implementation module

\_\_26. Click on **Add**.

\_\_27. Click on **Next**.

→ **Select query**

Enter query details

Select target entities

Summary

**Select query**

Select the entity type:

All Entities

**Next**

Cancel

\_\_28. Specify **Fin\*** as the search criteria. Click on **Next**.

Select query

→ **Enter query details**

Select target entities

Summary

**Enter query details**

Enter details for the query. Empty fields are not used in the query.

Query: All Entities

Use all of the following (AND)

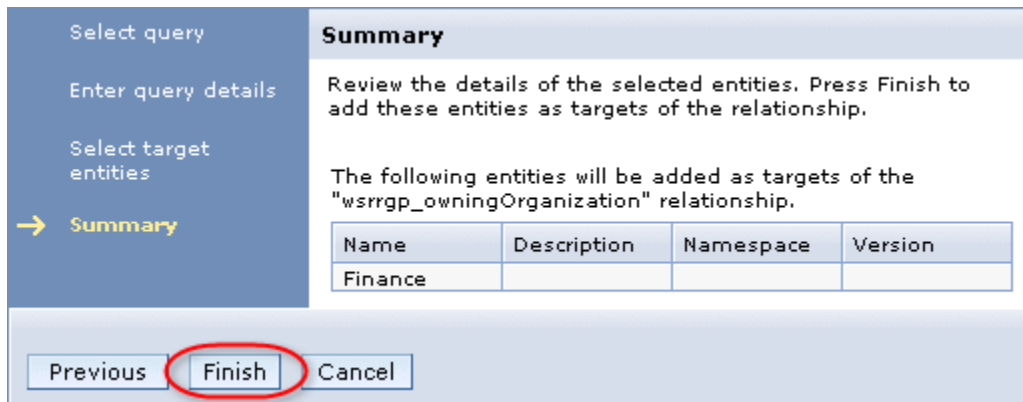
Name

Fin\*

\_\_29. Select **Finance**. Click on **Next**.

Select	Name	Description	Object Type
<input checked="" type="checkbox"/>	Finance		concept
Total:1			

\_\_30. Click on **Finish**.



Select query

Enter query details

Select target entities

→ **Summary**

**Summary**


Review the details of the selected entities. Press Finish to add these entities as targets of the relationship.

The following entities will be added as targets of the "wsrrgp\_owningOrganization" relationship.

Name	Description	Namespace	Version
Finance			

Previous **Finish** Cancel

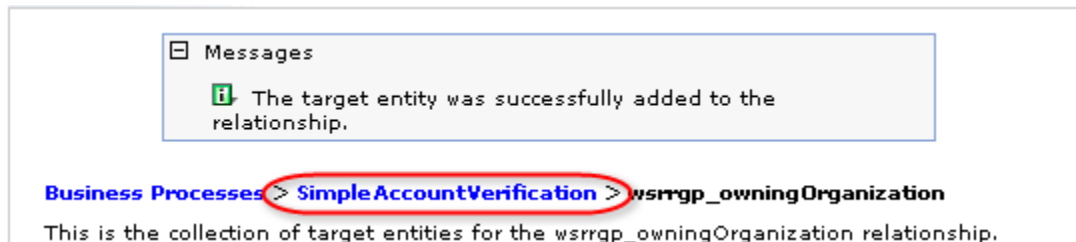
You should now be able to complete the creation of the relationship.



**What just happened?**

The Finance department has been specified as the Owing Organization of the process. You have now satisfied the first governance requirement for authorizing the development of the process.

\_\_31. Click on the **SimpleAccountVerification** link.



Messages

The target entity was successfully added to the relationship.

**Business Processes** > **SimpleAccountVerification** > wsrrgp\_owningOrganization

This is the collection of target entities for the wsrrgp\_owningOrganization relationship.

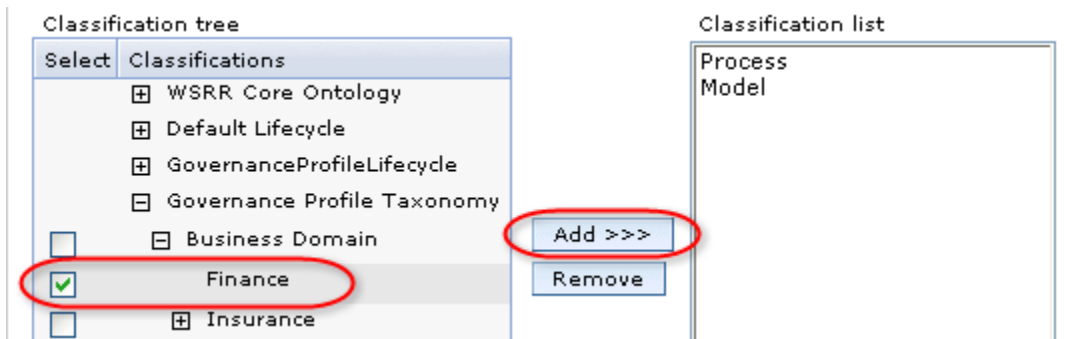
\_\_32. Click on **Classifications**.

Entity Metadata

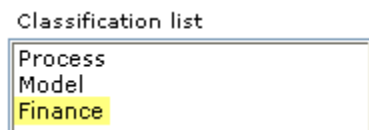
- Graphical View
- Properties
- Custom Relationships
- **Classifications**

\_\_33. Expand **Governance Profile Taxonomy** and **Business Domain**.

\_\_34. Select **Finance**. Click on **Add**.



Finance has been added to the classification list.



\_\_35. Click on **OK**.

**What just happened?**

The process has been classified as being part of the Finance Business Domain. You have now satisfied the second governance requirement for authorizing the development of the process. You should now be able to transition the process to the Assemble state.

\_\_36. Click on the **Governance** tab. Governance

\_\_37. Click on **Transition**.



The SimpleAccountVerification process is now in the Assemble phase.





**What's next?**

The process has been authorized for development. You will now import the process defined in the WebSphere Business Modeler into the WebSphere Integration Developer for implementation and assembly.



**Please wait for the next lecture before proceeding to the next lab.**

## Lab 4 Assemble and Deploy the Process

### Goals:

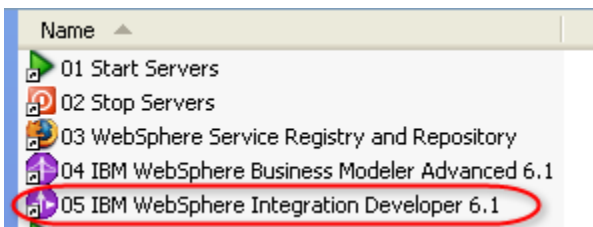
- **Assemble service components to form business process**
- **Lower business costs and achieve IT flexibility**
  - **Apply the building-block approach**
  - **Integrate using modular service components**
  - **Loosely-couple service components for flexibility**

### Role: Integration Developer

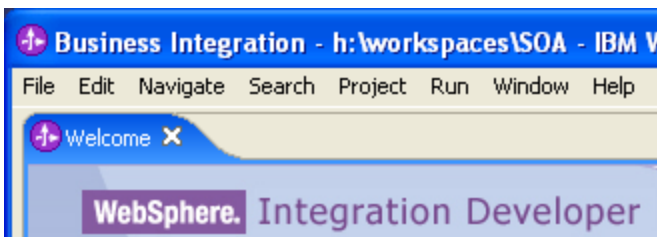
You will now assume the role of an Integration Developer who is responsible for assembling the different service components to compose the business process. The next steps will illustrate how loose-coupling and the building-block approach can be applied to make integration at the enterprise scale significantly easier.

### 4.1 Start the WebSphere Integration Developer

- \_\_\_1. Switch to the **PoT Lab Shortcuts** folder. Double-click on **05 IBM WebSphere Integration Developer 6.1**. Reopen the folder from the desktop if needed.



The **WebSphere Integration Developer** Welcome view should appear by default.



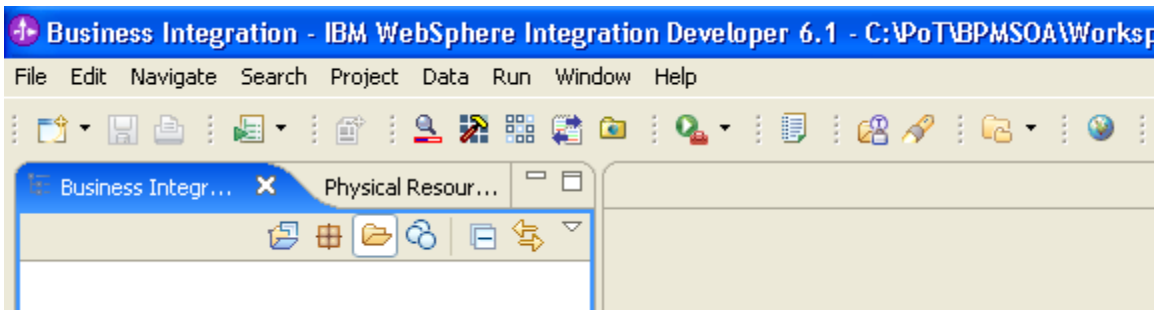
\_\_2. Maximize the **WebSphere Integration Developer** window.



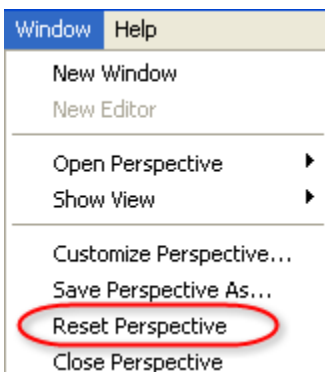
\_\_3. Click on the **Workbench** icon to close the Welcome view.



The Business Integration perspective now appears.



\_\_4. From the main menu, select **Window -> Reset Perspective**.



**Troubleshooting**

The perspective can be reset to display all the default views in their default locations. This will help in situations where a needed view cannot be found.

\_\_5. From the Reset Perspective confirmation window, click on **OK**.



**What's next?**

You will now import the business model that was created in the previous lab. The business model was exported as a BPEL process.

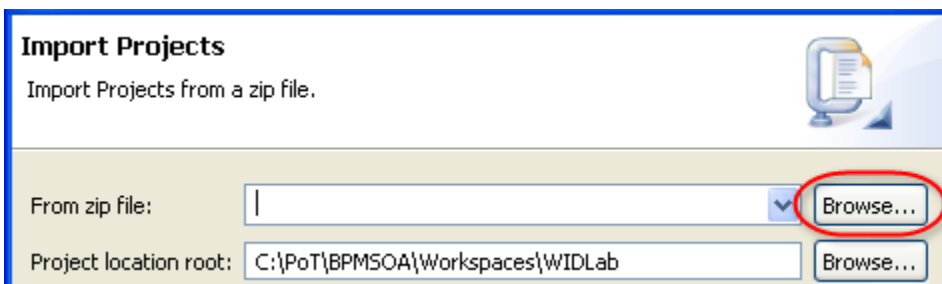
## 4.2 Import the business model into the WebSphere Integration Developer

\_\_6. From the main menu, select **File -> Import**.

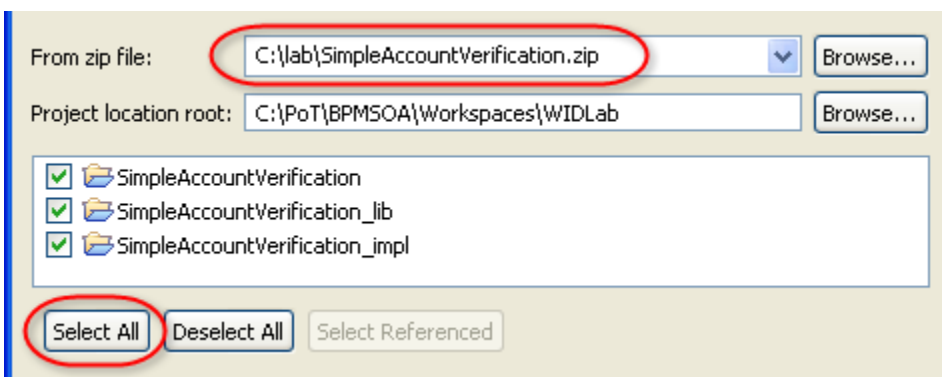
\_\_7. Expand the **Other** category. Select **Project Interchange** as the import source. Click on **Next**.



\_\_8. Click the **Browse** button for the **From zip file** field.



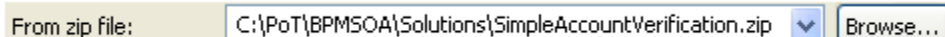
\_\_9. Select the file **C:\lab\SimpleAccountVerification.zip**. Click on **Select All**.



### Troubleshooting

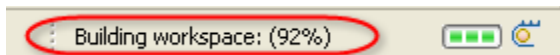


If the file **C:\lab\SimpleAccountVerification.zip** does not exist because the previous lab was not completed successfully, then you can use the file **C:\PoT\BPMSOA\Solutions\SimpleAccountVerification.zip**.






- \_\_10. Click on **Finish**.
- \_\_11. In the lower right corner of the window, if a progress indicator appears, wait for it to disappear before proceeding.



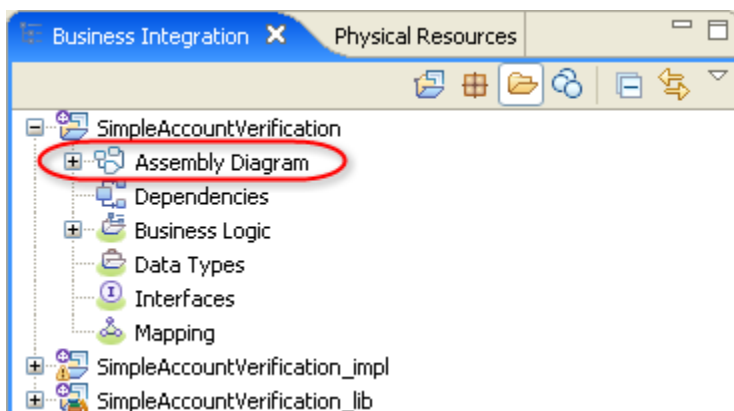
**What's next?**



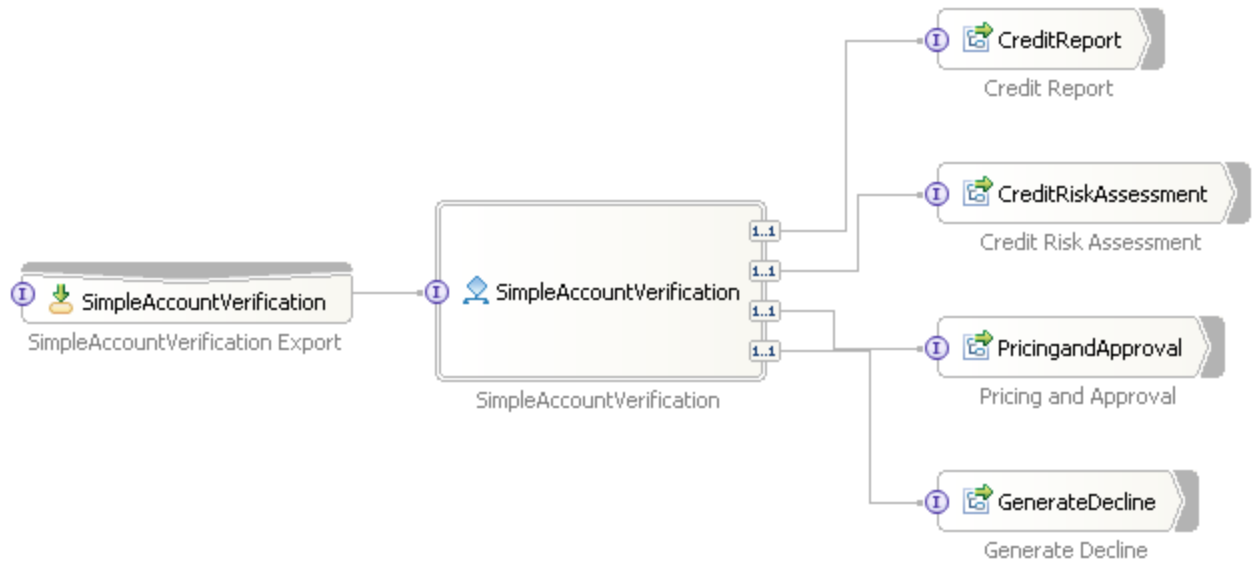
The imported BPEL process still needs to be completed or assembled. The tasks or activities in the process still need to be implemented or wired to their corresponding service endpoints.

### 4.3 Assemble the business process

- \_\_12. From the Business Integration view, in the SimpleAccountVerification module, double-click on **Assembly Diagram**.



This will open the Assembly Diagram editor for the main module. Refer to the screenshot below. The main module contains the BPEL process. You will also notice two other modules in the Business Integration view. You will work with these modules later.



**Hint**

Why are there three modules? You can actually just work with one module, which will simplify development. However, three modules provide a clearer separation and better flexibility. Changes later on will have a smaller impact. Here are brief descriptions of each module:




- SimpleAccountVerification - main module which contains the BPEL process and the overall assembly diagram.
- SimpleAccountVerification\_impl - implementation module which contains the implementation of the tasks defined in the BPEL process, as well as other related components.
- SimpleAccountVerification\_lib - library module which contains the interface definitions, business object definitions, maps, and other non-executable artifacts that need to be shared across different modules.

By separating the implementation module from the main BPEL module, changes made to the task or service implementations will not require a redeployment of the main module. By using a separate library module, it becomes easier to distribute and enforce standard definitions, such as a standard credit report service interface, or a standard customer object.

\_\_13. Double-click on the **Assembly Diagram** editor tab to maximize the view.



**Hint**



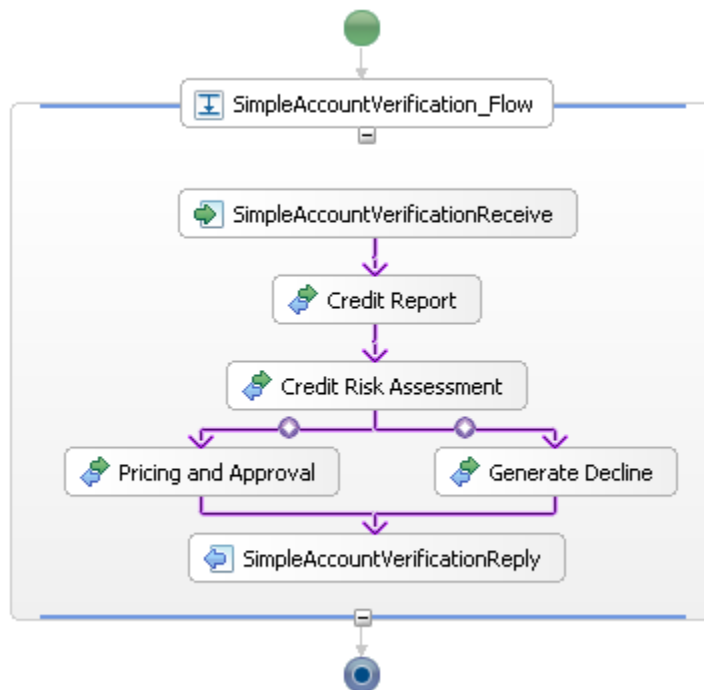
Double-clicking on the tab again will restore the view back to its original size. For now keep the view maximized so you'll have more screen space to work with. You can also double-click on any view titlebar or tab to maximize or restore the size.

\_\_14. From the **Assembly Diagram** editor, double-click on the **SimpleAccountVerification** process.



The SimpleAccountVerification process editor appears.

\_\_15. Review the **SimpleAccountVerification** process.

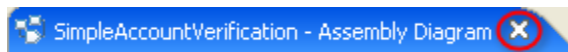


This is the BPEL version of the business model defined using the WebSphere Business Modeler. BPEL will be explained later on. Because this was exported from the WebSphere Business Modeler, it is already complete and does not require further development.

- \_\_16. Close the **SimpleAccountVerification** process editor.  
The **Assembly Diagram** editor will reappear.



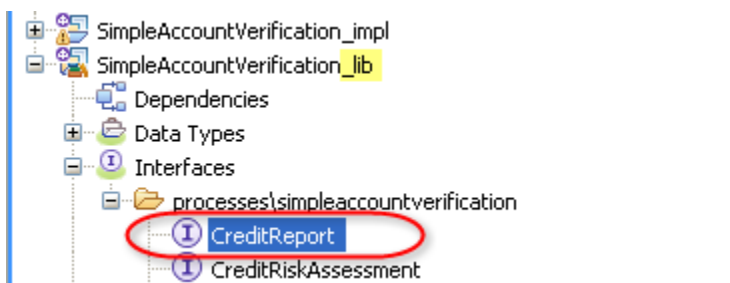
- \_\_17. Close the Assembly Diagram editor.



**What's next?**

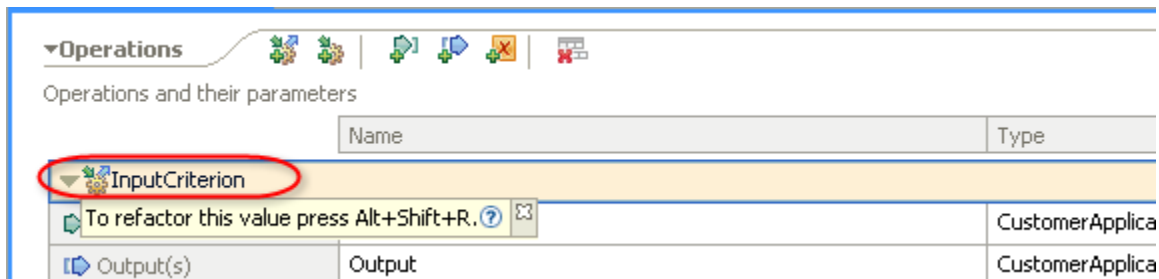
You'll make a simple change to the process imported from the WebSphere Business Modeler. Specifically, you'll rename the operation name for the Credit Report Service to something more meaningful.

- \_\_18. From the Business Integration view, expand **SimpleAccountVerification\_lib**. Expand **Interfaces** to expose CreditReport.
- \_\_19. Double-click on **Credit Report**.



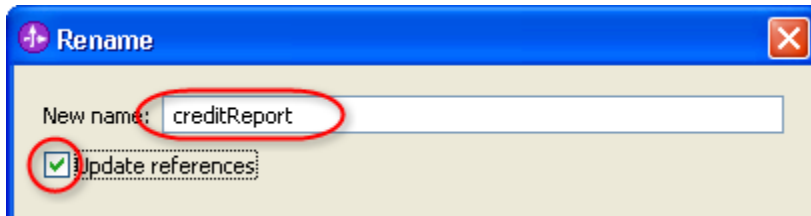
This will open the Interface Editor for the CreditReport service.

- \_\_20. Place the cursor on any part of the operation name **InputCriterion**.



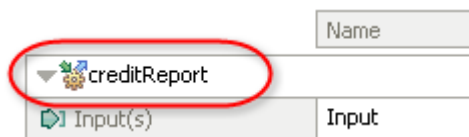
\_\_21. Press **Alt+Shift+R** to refactor/rename the operation name.

\_\_22. Change the operation name to **creditReport**. Select **Update references**.



\_\_23. Click on **OK**.

The operation name should now be creditReport.



#### Hint



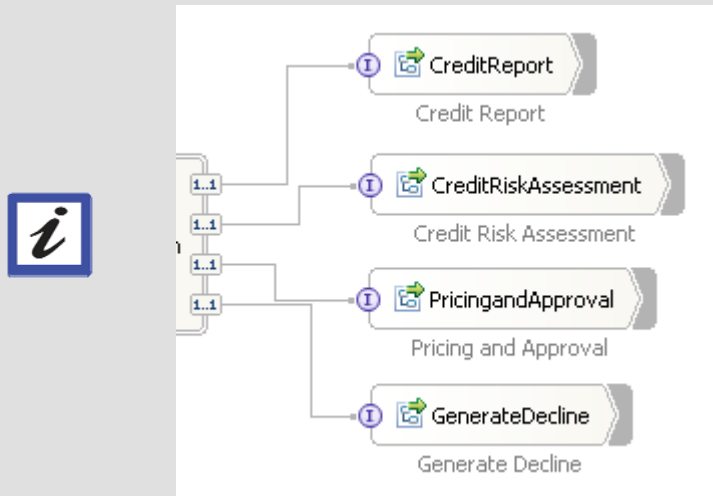
Renaming through the refactoring capabilities of this development environment will also rename all references to the name in other components. This will synchronize your changes and avoid introducing errors. Refactoring in general is an essential capability with this type of integration development.

\_\_24. Close the CreditReport Interface Editor.



**What's next?**

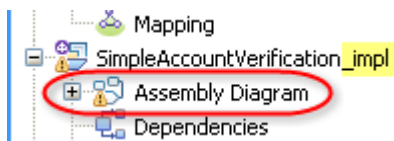
At this point, the tasks of the business process have been defined, as well as the flow through these tasks. However, the tasks have not yet been implemented. The next step is to implement the four tasks of our simple process.



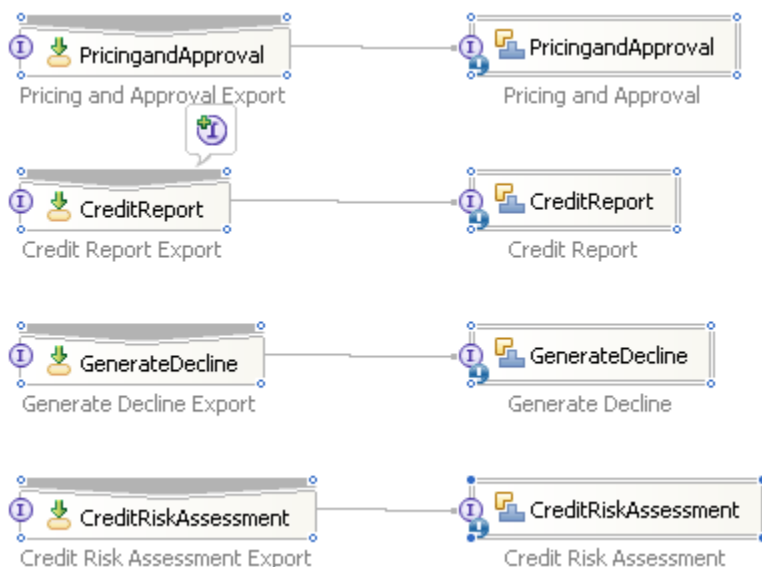
These tasks will initially be implemented using Java™ code because the required back-end systems are still under development. The Java code will serve as stubs or placeholders until the actual systems become available.

**4.4 Implement the tasks of the business process**

- \_\_25. From the Business Integration view, double-click on **SimpleAccountVerification\_impl -> Assembly Diagram**.



This will open the Assembly Diagram editor for this module.



#### Hint

This Assembly Diagram editor is where you will implement the four tasks of the SimpleAccountVerification process. The components on the left side are the Export components. The components on the right side are the Import components.

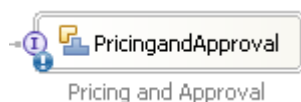


Export components allow a module to offer a service to others. Exports define interactions between modules and service requesters. Specifically, Exports define how the service implementations or endpoints can be accessed by others.

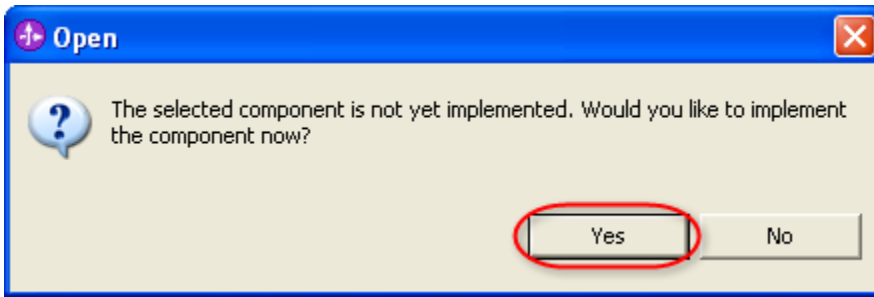
Import components allow a module to access external services as if they were local. Imports define interactions between modules and service providers. Specifically, Imports represent the implementations or service endpoints.

This Assembly Diagram editor is where you will implement the four tasks of the SimpleAccountVerification process.

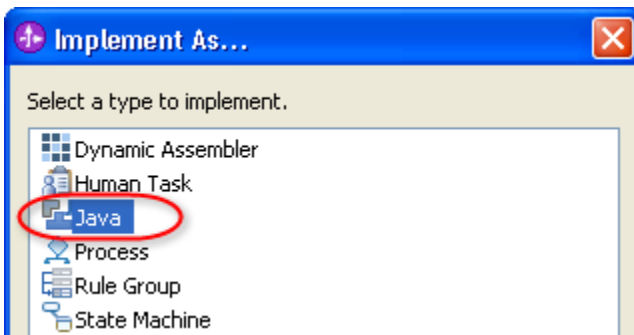
- \_\_26. Double-click on the **PricingandApproval** component.



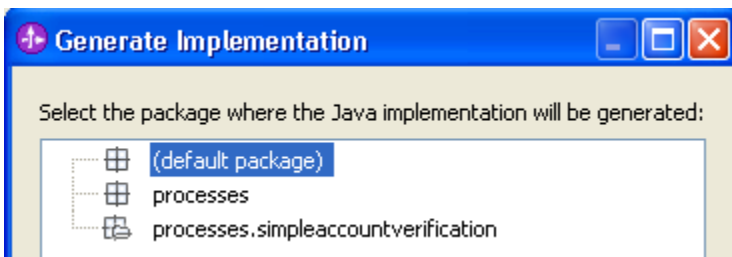
\_\_27. From the **Open** confirmation window, click on **Yes**.



\_\_28. Select **Java** as the implementation type. Click on **OK**.



\_\_29. From the **Generate Implementation** window, accept the default selection, and then click on **OK**.



The Java editor appears for the implementation of the **PricingandApproval** component.

```
*SimpleAccountVerification_impl - Assembly Diagram  PricingandApprovalImpl.java x
package sca.component.java.impl;

import commonj.sdo.DataObject;

public class PricingandApprovalImpl {
    /**
```



\_\_30. Scroll down to the bottom of the Java code. Select the highlighted text below and **delete**.

```

public DataObject InputCriterion(DataObject input) {
    //TODO Needs to be implemented.
    return null;
}

```

**Delete highlighted**

\_\_31. Type **pricapp** in the line of code as shown below. Press **Ctrl+spacebar**.

```

public DataObject InputCriterion(DataObject input) {
    pricapp
}

```

**Ctrl-Spacebar**



**Hint**

The Ctrl+spacebar key combination activates a special feature called Code-Assist. This will add code snippets based on predefined code templates.

A code snippet appears.

```

public DataObject InputCriterion(DataObject input) {
    /**
     * CODE STUB
     * Placeholder for production code later
     * Code only for initial development and testing
     */
    System.out.println(">>>>> Request is approved!.");
    return input;
}

```

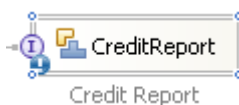


**Hint**

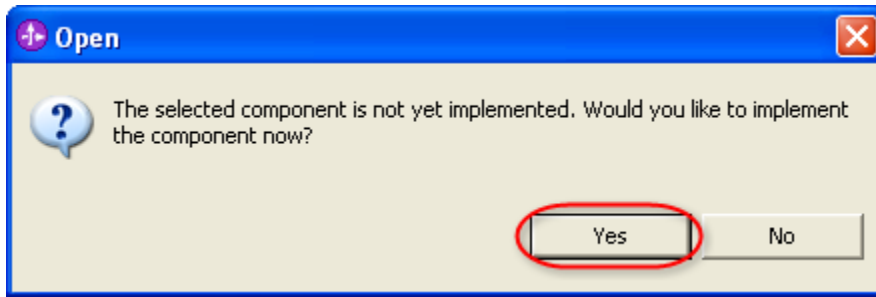
This code will simply output a message to the console indicating that the request was approved.

\_\_32. Press **Ctrl+s** to save the Java code. Close the **Java editor**.

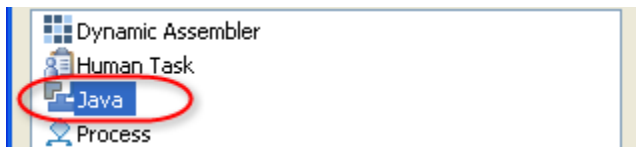
\_\_33. From the Assembly Diagram editor, double-click on the **Credit Report** component.



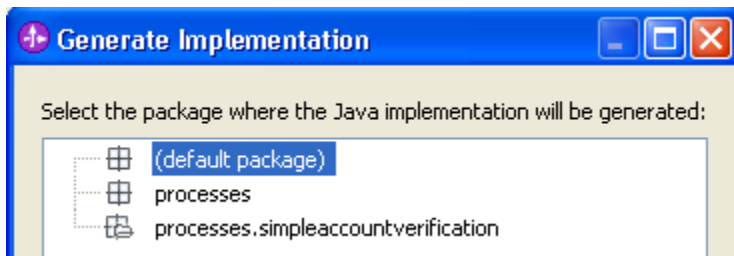
\_\_34. From the **Open** confirmation window, click on **Yes**.



\_\_35. Select **Java** as the implementation type. Click on **OK**.

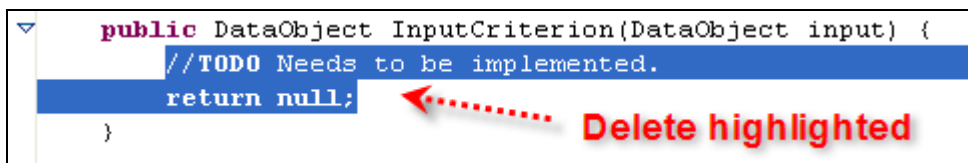


\_\_36. From the **Generate Implementation** window, accept the default selection, and then click on **OK**.

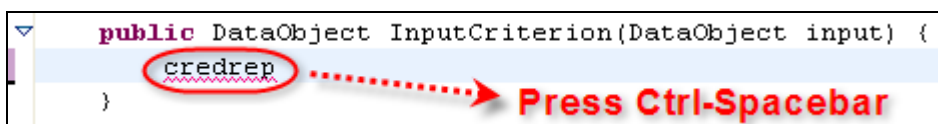


The Java editor appears for the implementation of the **CreditReport** component.

\_\_37. Scroll down to the bottom of the Java code. Select the highlighted text below and **delete**.



\_\_38. Type **credrep** in the line of code as shown below. Press **Ctrl+spacebar**.





\_\_45. Type **gendec** in the line of code as shown below. Press **Ctrl+spacebar**.

```
public DataObject InputCriterion(DataObject input) {
    gendec
}
```

A code snippet appears.

```
public DataObject InputCriterion(DataObject input) {
    /**
     * CODE STUB
     * Placeholder for production code later
     * Code only for initial development and testing
     */
    System.out.println(">>>> Request is denied!.");
    return input;
}
```



**Hint**

This code will simply output a message to the console indicating that the request was denied.

\_\_46. Press **Ctrl+s** to save the Java code. Close the **Java editor**.

\_\_47. From the Assembly Diagram editor, double-click on the **Credit Risk Assessment** component.



\_\_48. From the **Open** confirmation window, click on **Yes**.

\_\_49. From the **Implement As** window, select **Java** from the list, and then click on **OK**.

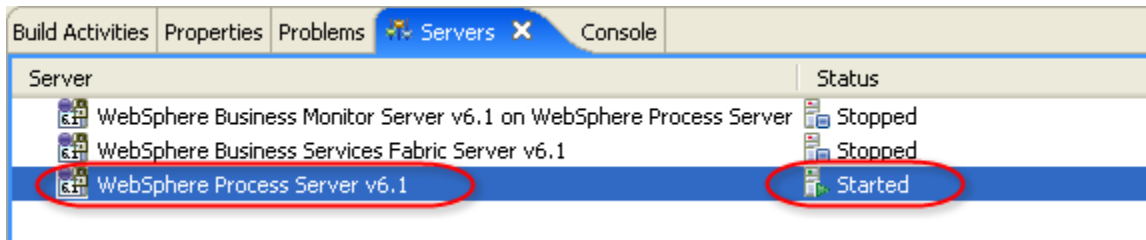
\_\_50. From the **Generate Implementation** window, accept the default selection, and then click on **OK**.

\_\_51. Scroll down to the bottom of the Java code. Select the highlighted text below and **delete**.

```
public DataObject InputCriterion(DataObject input) {
    //TODO Needs to be implemented.
    return null;
}
```



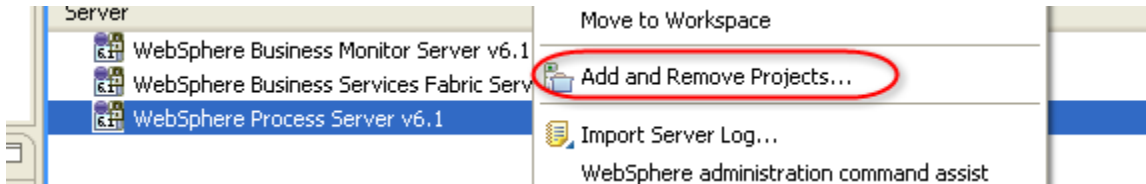
- \_\_56. Click on the **Servers** tab. Verify that the **WebSphere Process Server v6.1** test server is already started.



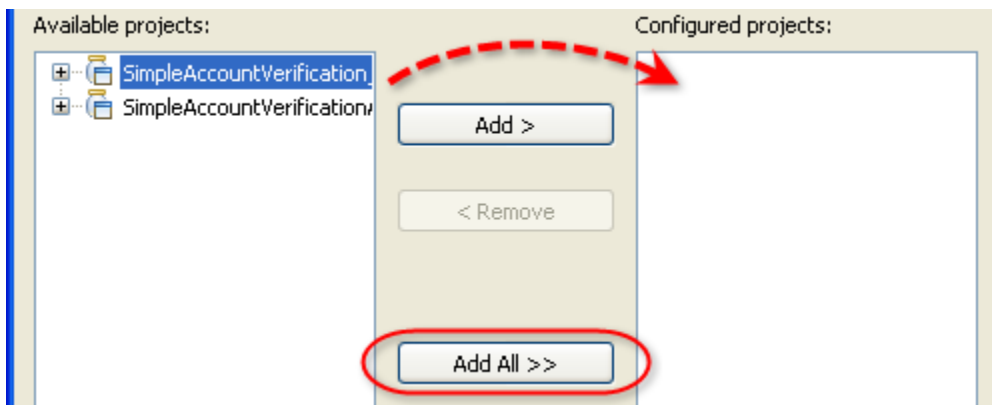
**Hint**

This test server built into the WebSphere Integration Developer is a full instance of the WebSphere Process Server.

- \_\_57. Right-click on the **WebSphere Process Server v6.1** test server. From the popup menu, select **Add and Remove Projects**.



- \_\_58. Click on **Add All**.



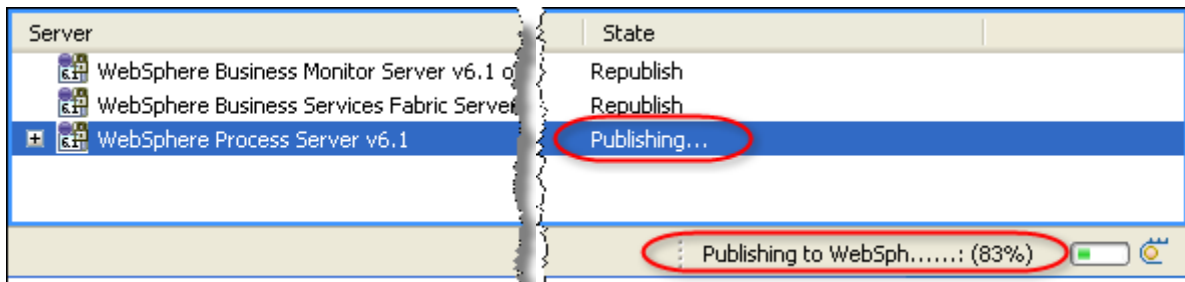
- \_\_59. Click on **Finish**.



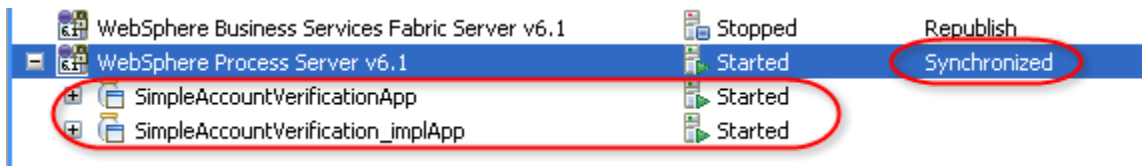
**What just happened?**

The SimpleAccountVerification process and all its related components were deployed to the WebSphere Process Server for testing. Specifically, an Enterprise Application module (EAR), EJB module, and other needed Web and Client modules were generated for the SimpleAccountVerification projects and deployed to the test server.

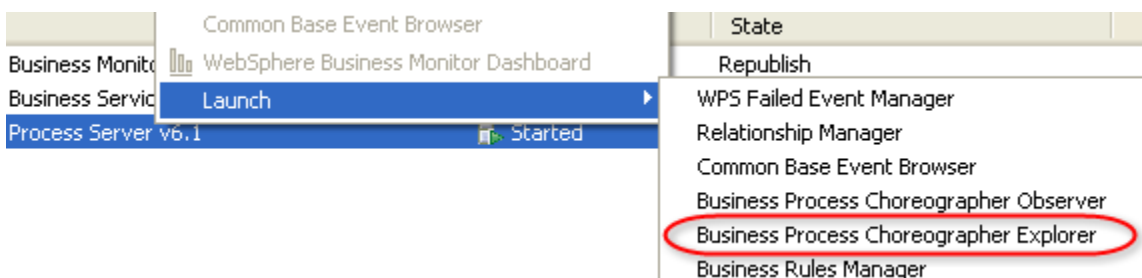
\_\_60. The server state will change to **Publishing**. You will also see a progress indicator at the bottom.



\_\_61. Wait for the progress indicator at the bottom to disappear. From the Servers view, check that the server state has changed to **Synchronized**. Expand WebSphere Process Server v6.1 and check that the SimpleAccountVerification projects have **Started**.



\_\_62. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Launch -> Business Process Choreographer Explorer**.



The Business Process Choreographer Explorer appears.

#### Hint

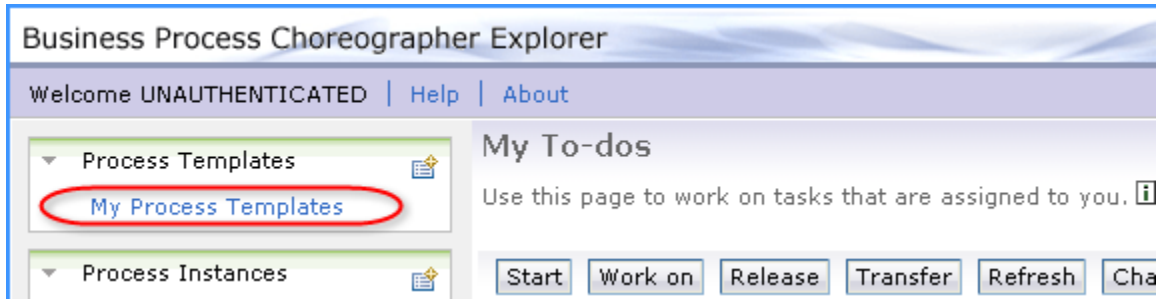


The Business Process Choreographer Explorer is a built-in web application that provides a way to manage business processes and work on items associated with human tasks. You will use this to test the SimpleAccountVerification process.

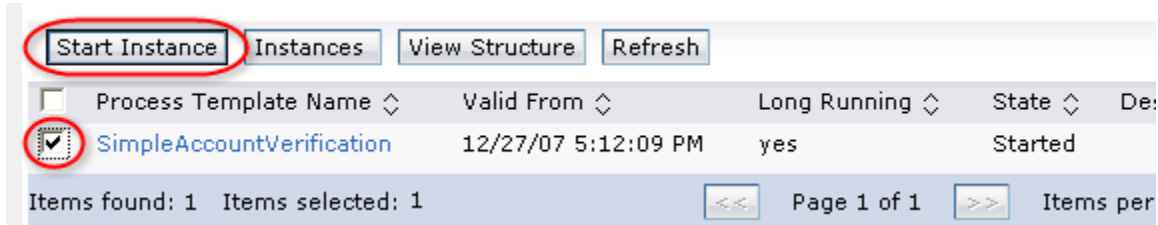
The Business Process Choreographer Explorer can also be started using a standard web browser at <https://localhost:9443/bpc>.

\_\_63. If a Security Alert window appears, select **Yes** to proceed.

\_\_64. Click on the **My Process Templates** link to display a list of processes which can be started.

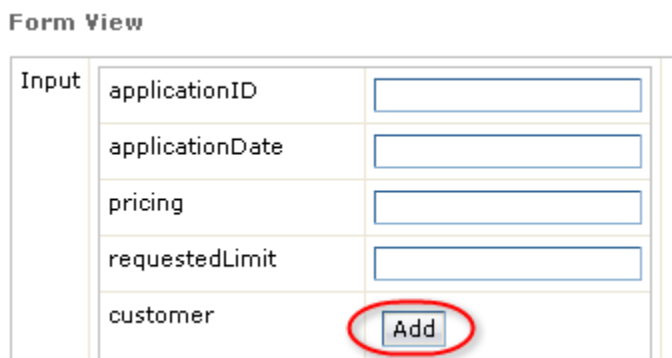


\_\_65. Place a checkmark beside **SimpleAccountVerification**. Click on **Start Instance**.



This will start the SimpleAccountVerification process, and display the default input page.

\_\_66. From the default input page, click on **Add**.







## 4.6 Cleanup

- \_\_70. When you're done testing the business process, close the Business Process Choreographer Explorer.



The “**Deploy**” phase of our SOA project is now complete.



**Please continue to the next lab.**

## Lab 5 Publish Service Interface for Reuse

### Goals:

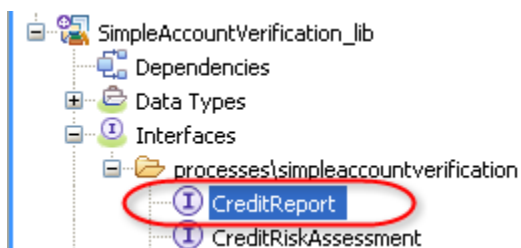
- Encourage reuse at the enterprise level
- Promote standards

### Role: Integration Developer

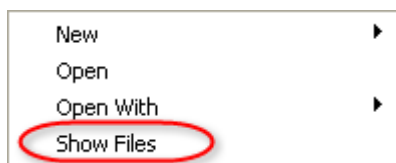
In an earlier lab, the WebSphere Business Modeler was used to model the SimpleAccountVerification business process, and then generate a BPEL process based on this model. This BPEL process was assembled and implemented using the WebSphere Integration Developer. One of the tasks in the SimpleAccountVerification process invokes a Credit Report service. The interface of the Credit Report service was based on the business model from the WebSphere Business Modeler, and will now become the standard interface for all credit report services used in the enterprise. In order to promote this interface as an enterprise standard and enable reuse, it will be published to the WebSphere Service Registry and Repository.

### 5.1 Publish the Credit Report Service Interface

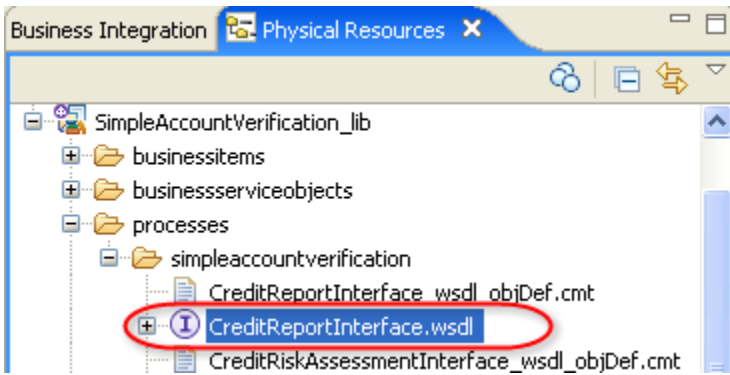
1. Select and right-click on the **CreditReport** Interface in the SimpleAccountVerification\_lib module.




2. From the popup menu, select **Show Files**.



This will switch to the Physical Resources view to display the actual file in the file system.



**Hint**

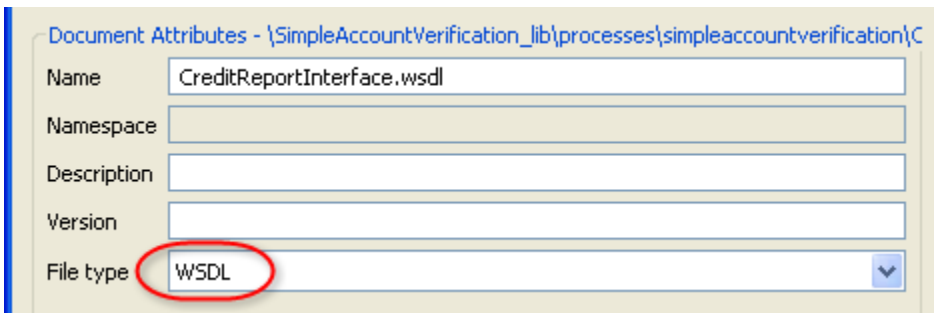


A service interface in the Business Integration view is actually a standard WSDL file in the file system. We need to select the WSDL file in order to publish it to the WebSphere Service Registry and Repository.

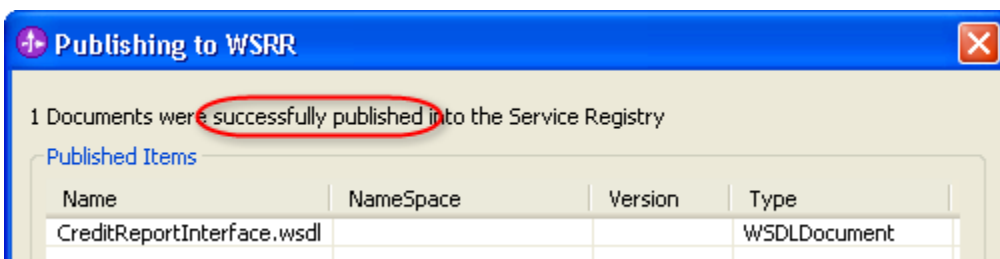
\_\_3. Right-click on **CreditReportInterface.wsdl**. From the popup menu, select **Service Registry -> Publish Document(s)**.



\_\_4. Check that the selected File type is **WSDL**. Click on **Finish**.




\_\_5. Verify that the WSDL file was successfully published. Click on **OK**.



**Role: Architect**

**What's next?**



You will now switch back to the role of an architect and review the published interface file. In a more realistic situation, governance will be applied and the proper authorization procedures will be enforced before the interface becomes an enterprise standard. However, to save time in this lab, you will just review the interface file.

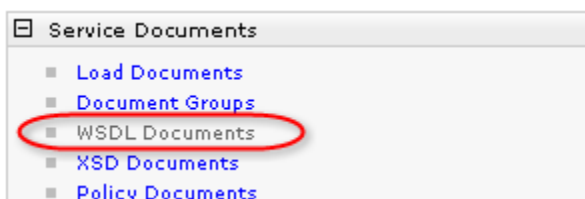
\_\_6. Switch to the web browser showing the **WebSphere Service Registry and Repository** console.

**Troubleshooting**




If the WebSphere Service Registry and Repository console was closed, just start a web browser and go to <http://localhost:9080/ServiceRegistry>.


\_\_7. Expand the **Service Documents** section. Click on **WSDL Documents**.



\_\_8. Verify that **CreditReportInterface.wsdl** appears on the list.

Select	Name	Graph	Description	Namespace
<input type="checkbox"/>	CreditReportInterface.wsdl			http://SimpleAccountVerification/Processes/
Total: 1				

**What just happened?**



You verified that the CreditReportInterface.wsdl file uploaded from the WebSphere Integration Developer is now contained in the WebSphere Service Registry and Repository. The WSDL file can now be reused as the standard interface for both internal and external Credit Report services. In a realistic environment, the Credit Report Interface should also be properly governed, but that will be skipped in this lab.



**Please wait for the next lecture before proceeding to the next lab.**

## Lab 6 Explore a Web Service Endpoint

### Goals:

- **Explore the Internal Credit Report Web Service**

**Role: Web Service Developer**

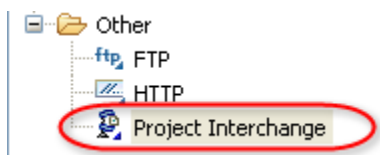
Let's first make several assumptions:

- You are a web services developer for the Finance department.
- You are working on a different machine from where the SimpleAccountVerification process is being assembled.
- Your primary role is application and web services development.
- For this particular project, you do not need to be concerned with BPEL process development. That has been assigned to another member of your development team.
- This lab is not focused on the deep technical aspects of web services development. The focus is more on how web services fit into SOA and BPEL projects. In line with this, you will assume that you've already completed the web service needed by the BPEL process.

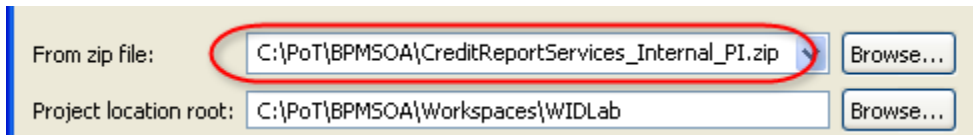
One of the tasks of the SimpleAccountVerification process is to request a credit report. This CreditReport service is currently implemented as Java code. The decision now is to change the implementation to invoke a web service. You were assigned to provide an internal credit report web service, which must be based on a standard service interface published to the WebSphere Service Registry and Repository. You've worked diligently, and you've now completed the Internal Credit Report Web Service, and it is ready for use. The next step is to publish the web service to the WebSphere Service Registry and Repository so that it can be invoked by the SimpleAccountVerification process.

### 6.1 Import the completed web service

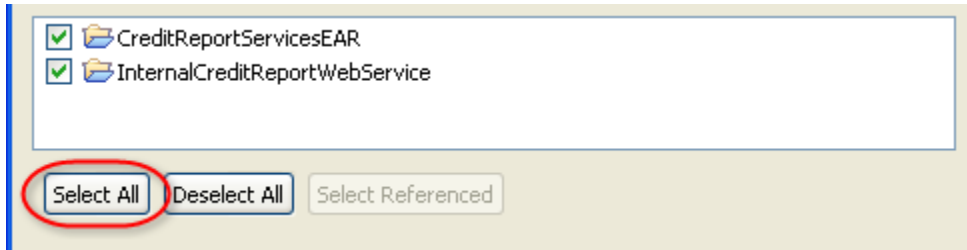
- \_\_1. Switch to the **WebSphere Integration Developer**.
- \_\_2. From the main menu, select **File -> Import**.
- \_\_3. Expand the **Other** category. Select **Project Interchange** as the import source. Click on **Next**.



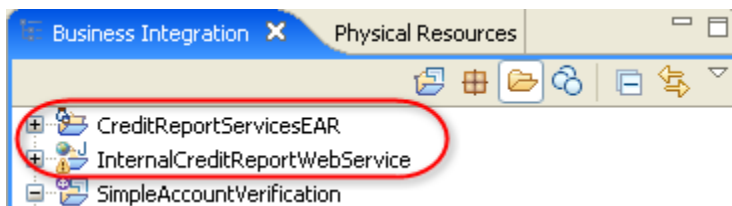
- \_\_4. Click the **Browse** button for the **From zip file** field. Select the file **C:\PoT\BPMSOA\CreditReportServices\_Internal\_PI.zip**.



- \_\_5. Click on **Select All**. Click on **Finish**.



You will now see two more modules in the Business Integration view.



**What just happened?**

Instead of developing the web service from the ground up, you imported two modules which contain the completed Credit Report web service. This was done to save time. At this point, just assume that you've just finished the development of the web service.



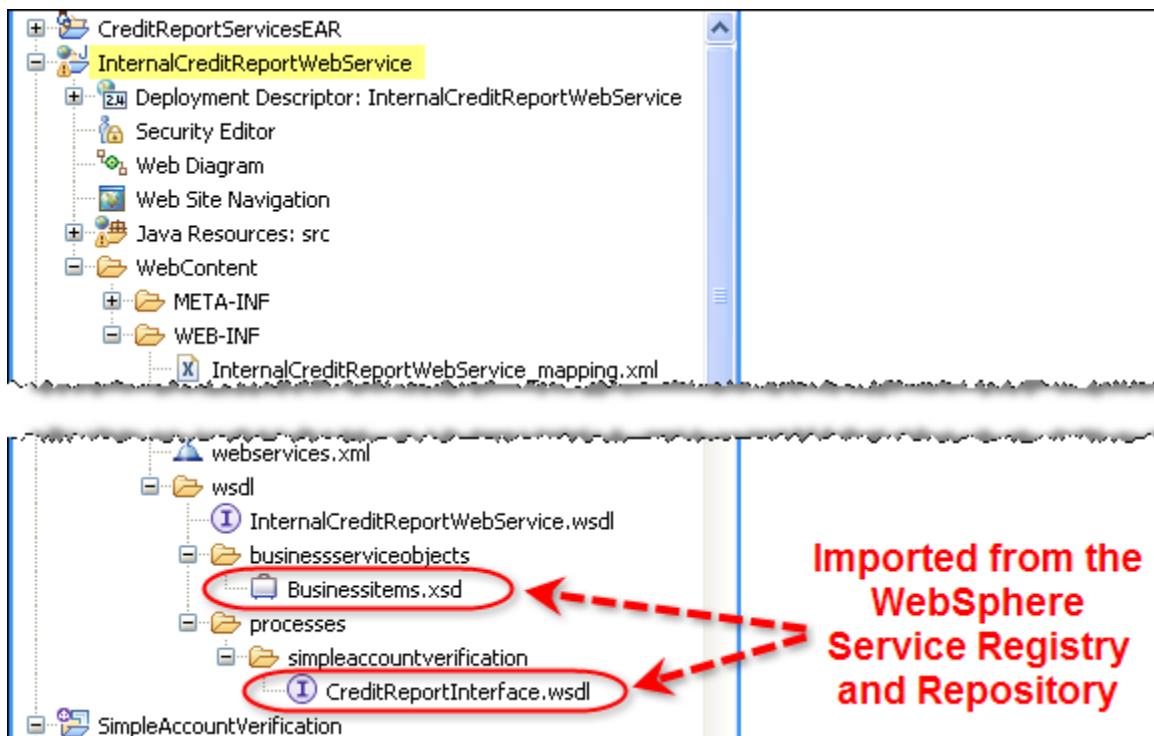
**What's next?**

You will briefly explore the completed Credit Report web service, and then publish the web service to the WebSphere Service Registry and Repository for reuse.



## 6.2 Explore the web service artifacts

- \_\_6. Expand the **InternalCreditReportWebService** module. Expand **WebContent** -> **WEB-INF**. Expand **wsdl** until you find the files **Businessitems.xsd** and **CreditReportInterface.wsdl**.



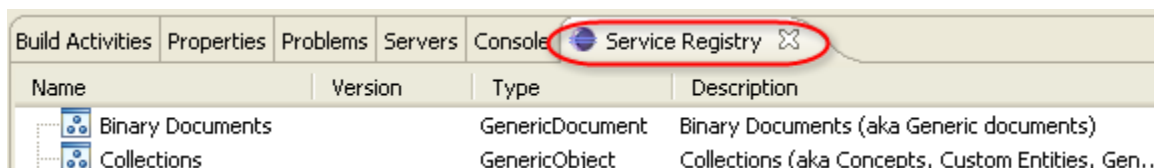
The **Businessitems.xsd** and **CreditReportInterface.wsdl** files were imported from the WebSphere Service Registry and Repository at the start of the development of the web service. The standard Credit Report service interface defined in the WSDL file, as well as the standard CustomerApplication object defined in the XSD file, were used in the web service.



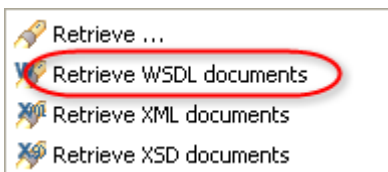
### What's next?

You will now explore how the **Businessitems.xsd** and **CreditReportInterface.wsdl** files were imported from the WebSphere Service Registry and Repository

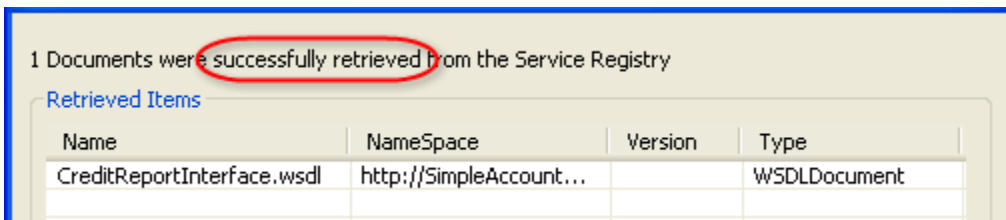
- \_\_7. Switch to the **Service Registry** view.



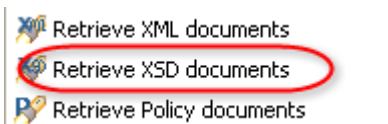
- \_\_8. Right-click anywhere inside the Service Registry view. From the popup menu, select **Retrieve WSDL documents**.



A window appears indicating that the CreditReportInterface.wsdl file was successfully retrieved.

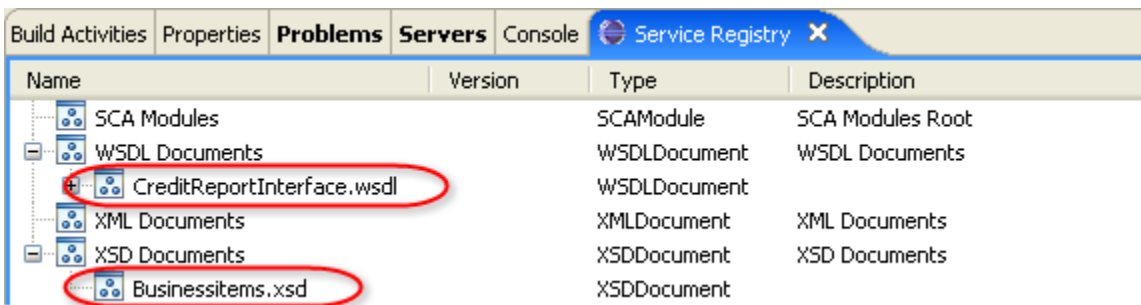


- \_\_9. Click on **OK**.
- \_\_10. Right-click again anywhere inside the Service Registry view. From the popup menu, select **Retrieve XSD documents**.




A window appears indicating that the Businessitems.xsd file was successfully retrieved.

- \_\_11. Click on **OK**.
- \_\_12. Switch to the Service Registry view. You will now see the CreditReportInterface.wsdl and Businessitems.xsd files retrieved from the WebSphere Service Registry and Repository.



From here these files were imported into the InternalCreditReportWebService module so that the Internal Credit Report web service can be developed based on the standard interface defined in the WSDL, as well as use the standard data objects defined in the XSD.

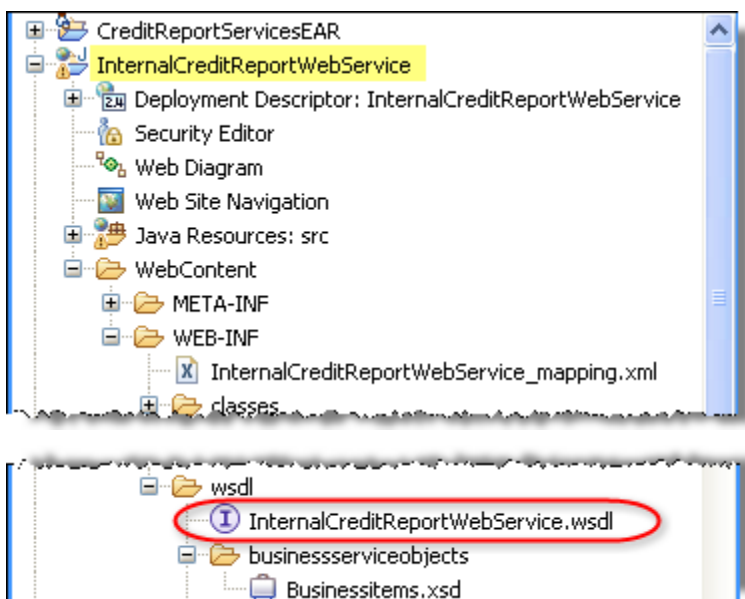
**What's next?**



You will now publish the web service to the WebSphere Service Registry and Repository. The assumption is that the web service developer is using a different machine from the developer working on the BPEL process. You are publishing the web service from this machine so that it can be imported into the machine of the integration developer.

### 6.3 Publish the Web Service

- \_\_13. Switch to the **Business Integration** view.
- \_\_14. Expand InternalCreditReportWebService. Right-click on **InternalCreditReportWebService.wsdl**.



- \_\_15. From the popup menu, select **Service Registry -> Publish Document(s)**.



\_\_16. Ensure that the selected File type is **WSDL**. Click on **Finish**.

Document Attributes - \InternalCreditReportWebService\WebContent\WEB-INF\wsdl\InternalCreditReportWebServ

Name: InternalCreditReportWebService.wsdl

Namespace:

Description:

Version:

File type: WSDL

\_\_17. A window will appear indicating that the document was successfully published. Click on **OK**.

## 6.4 Check the WebSphere Service Registry and Repository

\_\_18. Switch to the web browser showing the **WebSphere Service Registry and Repository** console.

**Troubleshooting**

If the WebSphere Service Registry and Repository console was closed, just start a web browser and go to <http://localhost:9080/ServiceRegistry>.

\_\_19. Select **GP Developer** from the perspective list. Click on **Go**.

Perspective: GP Developer Go

\_\_20. Expand the **Service Documents** section. Click on **WSDL Documents**.

- Service Documents
  - Load Documents
  - Document Groups
  - WSDL Documents
  - XSD Documents
  - Policy Documents

\_\_21. Verify that **InternalCreditReportWebService.wsdl** appears on the list. (Do not click)

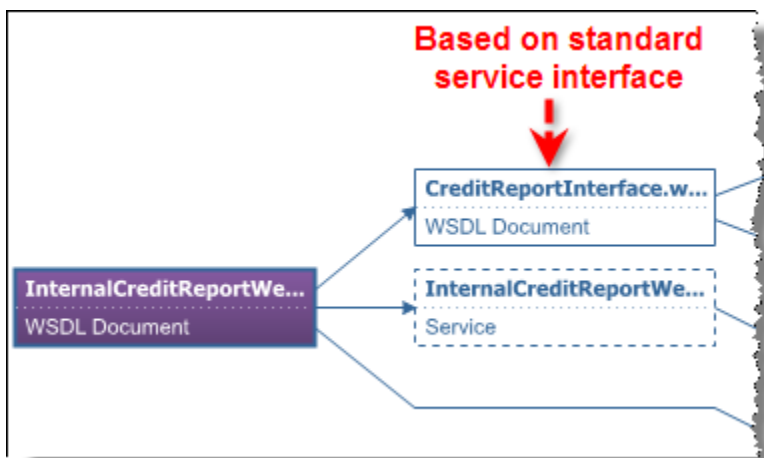
Select	Name	Graph	Description	Namespace
<input type="checkbox"/>	CreditReportInterface.wsdl			http://SimpleAccountVerification/
<input type="checkbox"/>	InternalCreditReportWebService.wsdl			http://Processes/SimpleAccountV
Total: 2				

\_\_22. Click on the **Graph** button for the InternalCreditReportWebService.wsdl.



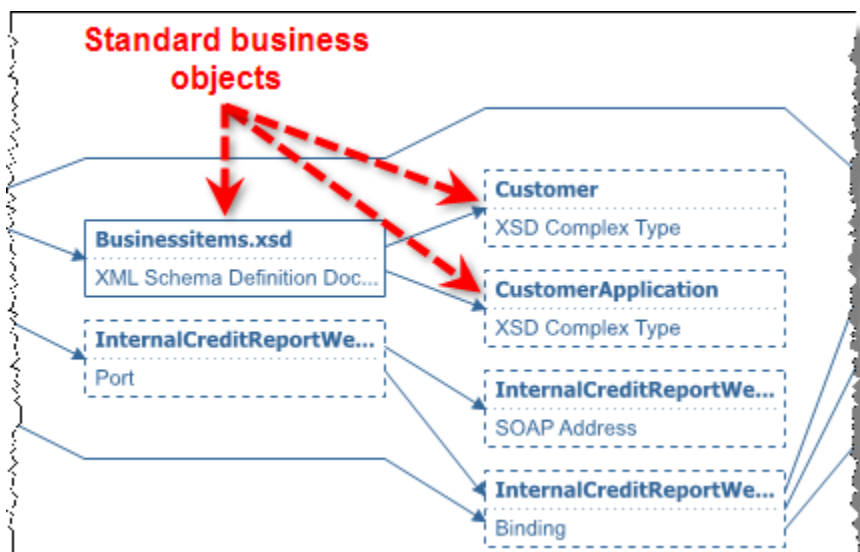
After a few moments, a graphical view appears showing the different relationships involving InternalCreditReportWebService.wsdl.

\_\_23. Maximize the web browser. Review the graph.



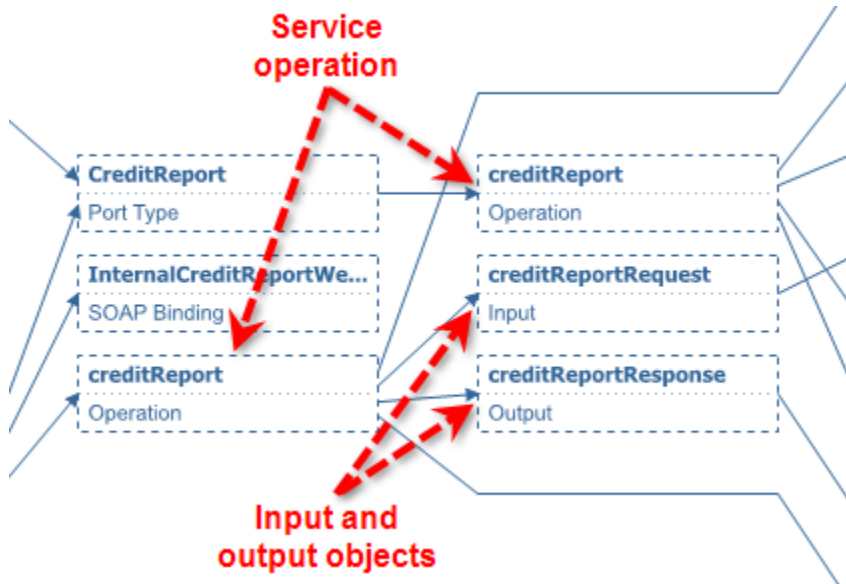
Notice that the web service is based on the standard interface.

\_\_24. Partially scroll to the right of the diagram.



The web service also uses the standard business objects.

\_\_25. Scroll to the right of the diagram again.



The graph also shows the service operation and input/output objects.



**What's next?**

The InternalCreditReportWebService.wsdl file can now be reused and imported into the development machine where the SimpleAccountVerification process is being developed.



**Please continue to the next lab.**

## Lab 7 Change Service Implementation

### Goals:

- Respond to changing business needs
- Take advantage of IT flexibility provided by infrastructure
- Quickly change implementation of business tasks

### Role: Integration Developer

Let's first make several assumptions:

- You are back to being an integration developer.
- Another member of your development team has just completed the Internal Credit Report web service, and has made it available for use.
- You are working on a different machine from where the Internal Credit Report web service was developed.

To illustrate the IT flexibility needed for business flexibility, you will change the implementation of one of the tasks without major effort. By applying the loose-coupling and building block approach, switching to a different application or system becomes easier and quicker. This is true even if the new system involves a different technology, programming language, and runtime environment. This approach allows business processes to be more responsive to changes in their business environment, such as being able to quickly adapt to customer demands, exploit new market opportunities, or react to competitive threats.

In our simple scenario, let's assume that the Credit Report task now needs to invoke an existing system exposed as a web service. You will now change the implementation type of this task from Java code to a web service invocation. The web service is really just an example in this exercise. Instead of a web service, you can just as easily switch the Credit Report task to a human task, business rule, business process, messaging-based application, back-end system exposed through adapters, etc.

### 7.1 Import the web service WSDL from the WebSphere Service Registry and Repository

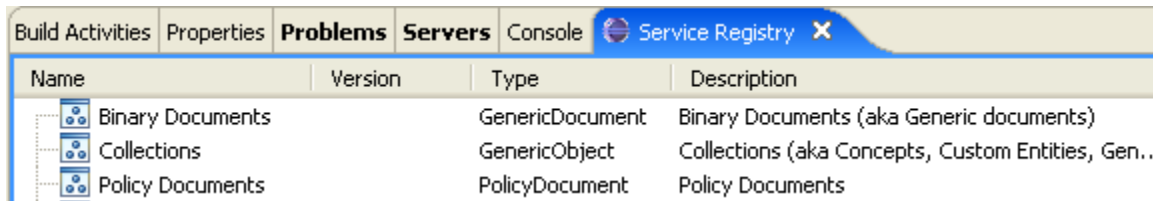
1. Switch to the **WebSphere Integration Developer**.



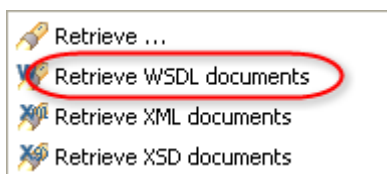
#### What's next?

You will need to import the WSDL file for the Internal Credit Report web service so that it can be used by the process. This WSDL file was published earlier by the web service developer to the WebSphere Service Registry and Repository.

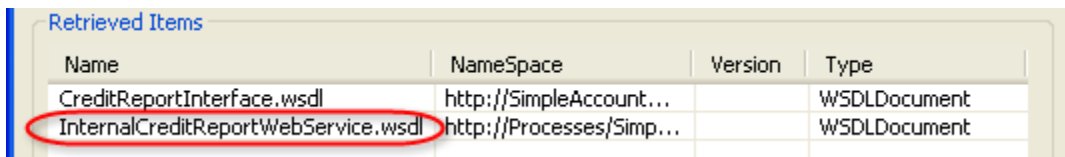
\_\_2. Switch to the **Service Registry** view.



\_\_3. Right-click anywhere inside the Service Registry view. From the popup menu, select **Retrieve WSDL documents**.



A window appears indicating that the InternalCreditReportWebService.wsdl file was successfully retrieved.

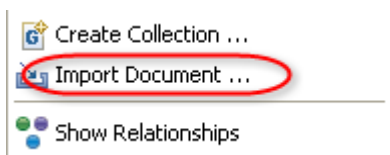


\_\_4. Click on **OK**.

\_\_5. Switch to the Service Registry view. Expand **WSDL Documents**. Right-click on **InternalCreditReportWebService.wsdl**.

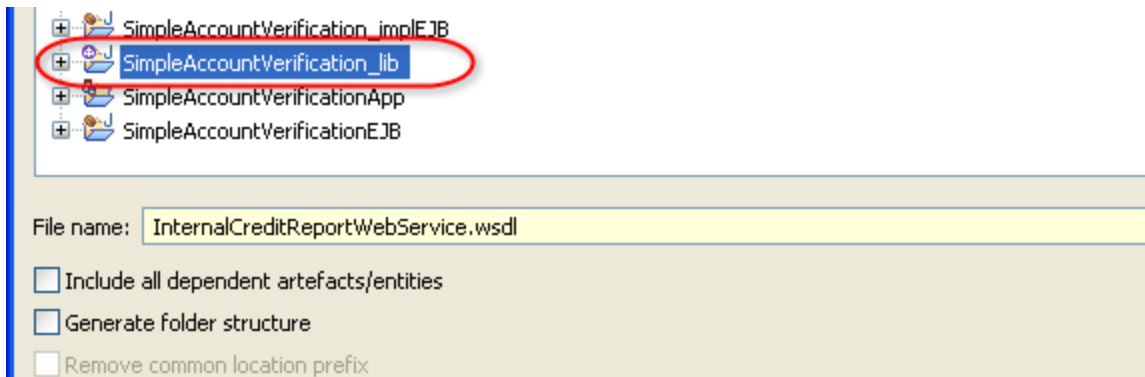


\_\_6. From the popup menu, select **Import Document**.

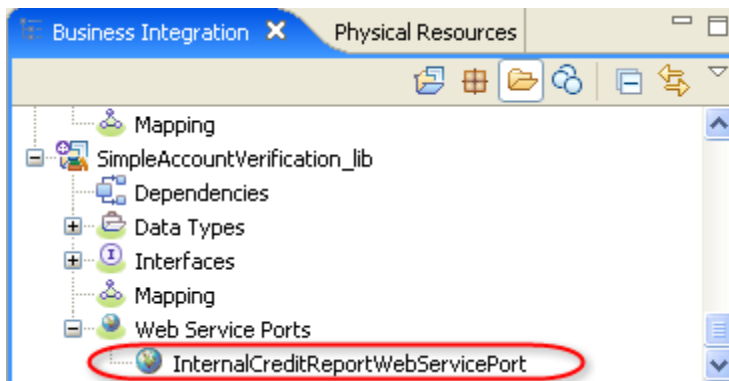


\_\_7. Select **SimpleAccountVerification\_lib**. Click on **Finish**.





- \_\_8. Switch to the Business Integration view. Verify that the InternalCreditReportWebService now appears under Web Service Ports, and is ready for use.



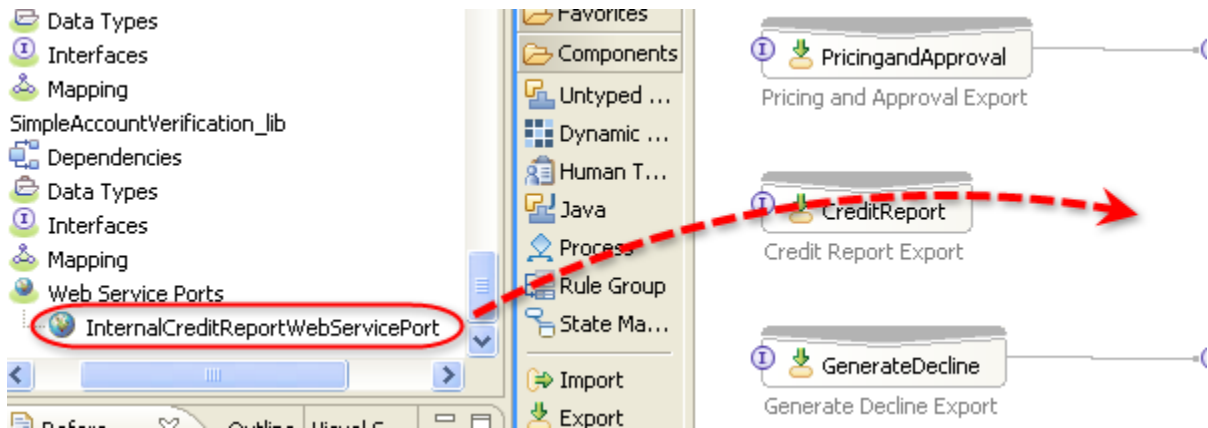
**What just happened?**  
 You imported the InternalCreditReportWebService.wsdl file published by the web service developer and this web service can now be invoked by the BPEL process.

## 7.2 Change the Credit Report task to a Web Service Implementation

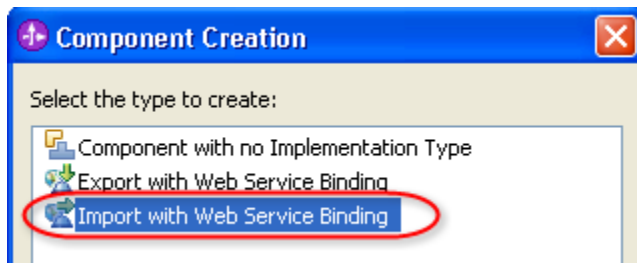
- \_\_9. From the **Assembly Diagram** editor, select the **CreditReport** component on the right. Press the **delete** key.



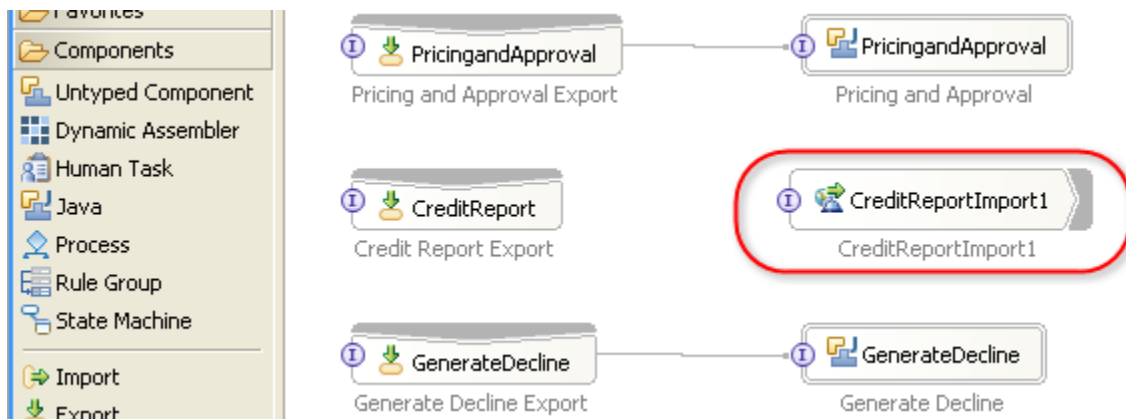
- \_\_10. Using the left mouse button, drag and drop **InternalCreditReportWebServicePort** into an empty space in the **Assembly Diagram** editor.



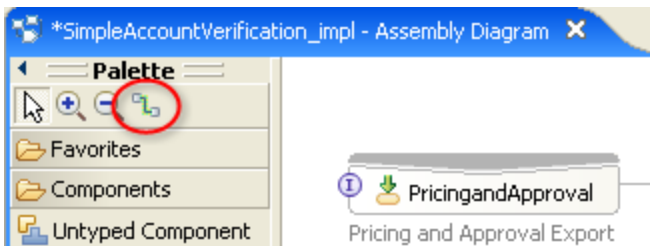
- \_\_11. Select **Import with Web Service Binding**. Click on **OK**.



A **CreditReport** Web Service component will appear in the **Assembly Diagram** editor.



- \_\_12. From the palette on the left, select the **Wire** icon.




This will switch the editor to 'wire' mode, which will make it easier to connect components in the Assembly Diagram editor.

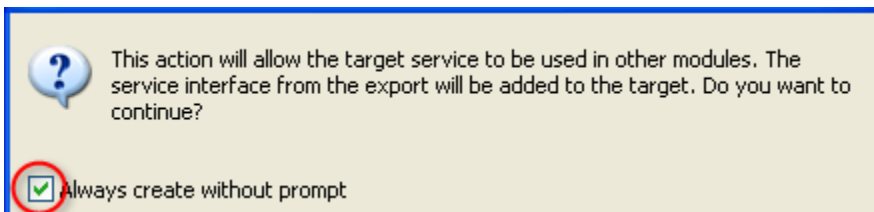
- \_\_13. Left-click on the CreditReport export component. This will start a connection (blue line).



**Hint**

 An export component defines how the service implementation can be accessed by others. Specifically in the diagram above, the Credit Report Import on the right represents the implementation or service endpoint. The Credit Report Export on the left defines how other external components can access the Credit Report Service.

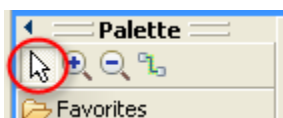
- \_\_14. Click on the **CreditReportImport1** component.
- \_\_15. A prompt window will appear. Select the **Always create without prompt** option. Click on **OK**.



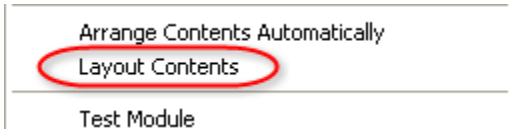
This will complete the connection.



- \_\_16. From the Palette, click on the **Selection Tool** icon to switch back to selection mode.



- \_\_17. Right-click on an empty space in the **Assembly Diagram** editor. From the popup menu, select **Layout Contents**.



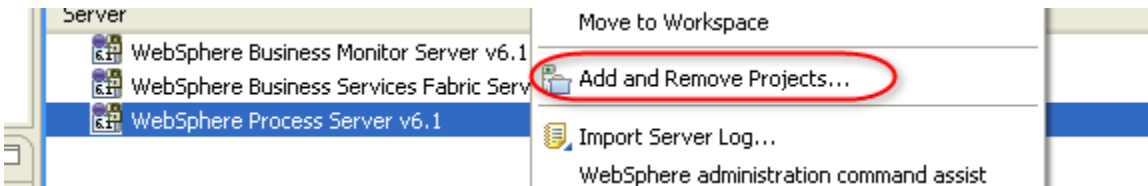
This will rearrange the **CreditReportImport1** component for a cleaner layout.

- \_\_18. Press **Ctrl+s** to save the modified assembly. Do not close the Assembly Diagram editor.

### 7.3 Retest the SimpleAccountVerification Process

- \_\_19. Switch to the **Servers** view.

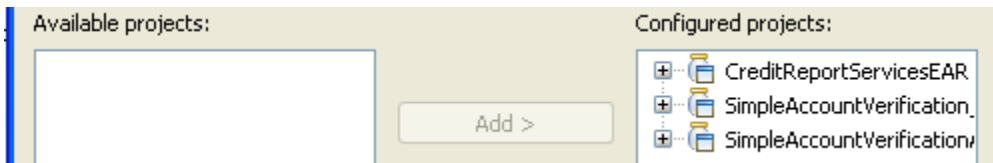
- \_\_20. Right-click on the **WebSphere Process Server v6.1**. From the popup menu, select **Add and Remove Projects**.



- \_\_21. From the **Add and Remove Projects** window, click on **Add All**.

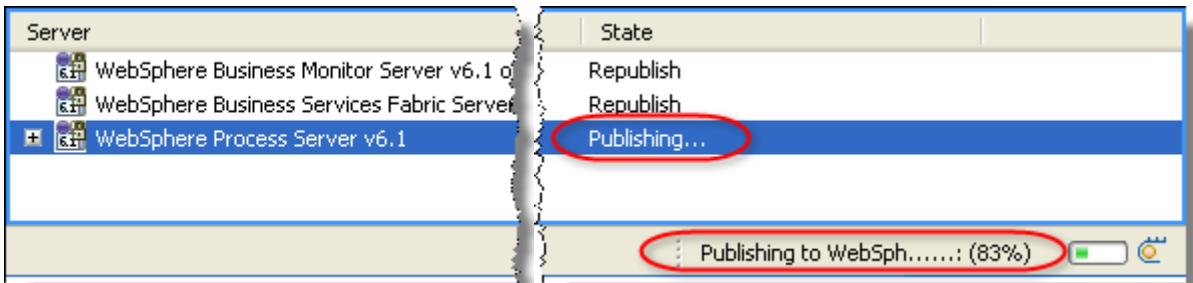


This will add the project to the list of projects that will be deployed to the test server.

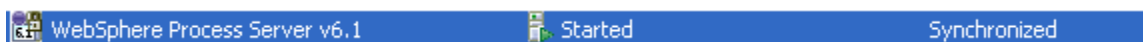


- \_\_22. Click on **Finish**.

- \_\_23. The server state will change to **Publishing**. You will also see a progress indicator at the bottom.

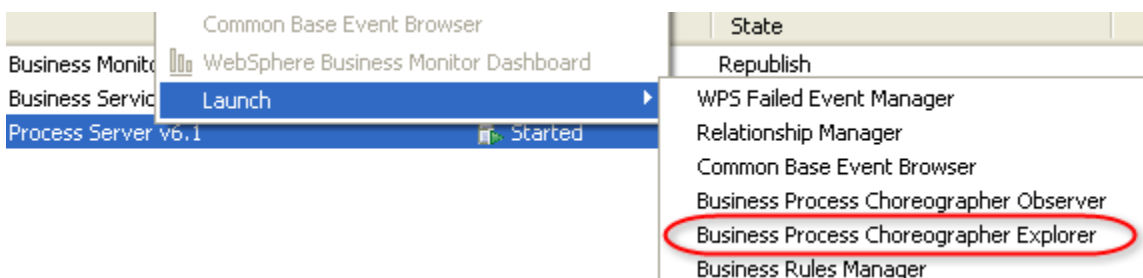


- \_\_24. Wait for the progress indicator at the bottom to disappear. From the Servers view, check that the server state has changed to **Synchronized**.



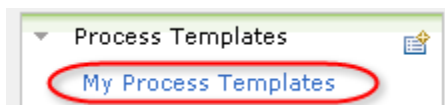
- \_\_25. From the Servers view, expand **WebSphere Process Server v6.1**.

- \_\_26. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Launch -> Business Process Choreographer Explorer**.

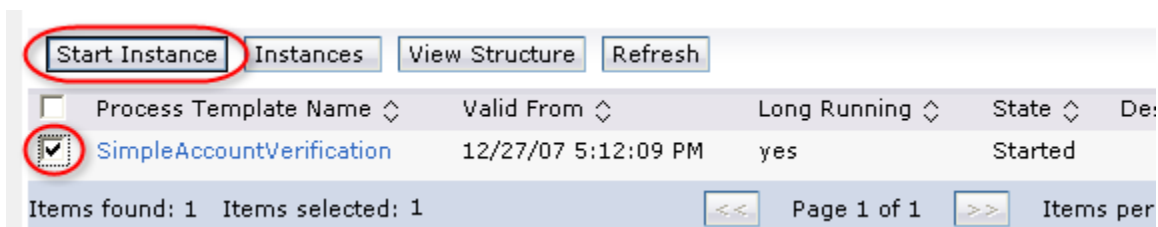


The Business Process Choreographer Explorer appears.

- \_\_27. Click on the **My Process Templates** link to display a list of processes which can be started.



- \_\_28. Place a checkmark beside **SimpleAccountVerification**. Click on **Start Instance**.



This will start the SimpleAccountVerification process, and display the default input page.

\_\_29. From the default input page, click on **Add**.

**Form View**

Input	applicationID	<input type="text"/>
	applicationDate	<input type="text"/>
	pricing	<input type="text"/>
	requestedLimit	<input type="text"/>
	customer	<input type="button" value="Add"/>

\_\_30. For the **customerID** field, type **123**.

requestedLimit	<input type="text"/>	
customer	customerID	<input type="text" value="123"/>
	companyName	<input type="text"/>
	customerAddress	<input type="text"/>
	customerCity	<input type="text"/>
	customerStateProvince	<input type="text"/>
	customerCountry	<input type="text"/>
	customerPostalCode	<input type="text"/>
	customerTaxID	<input type="text"/>



**Hint**

A Customer ID of “123” will result in a low credit score and a denied request. Any other Customer ID value will result in an approved request.

\_\_31. Click on **Submit**.

\_\_32. After a few moments, the SimpleAccountVerification process will generate messages to the **Console** view indicating that the request is denied.



## Lab 8 Externalize Business Rules

### Goals:

- **Achieve business agility**
- **Allow changes to business rules at runtime**
  - **No recoding, retesting, and redeploying**

### Role: Integration Developer

Changes in your business environment often involve the need to adjust business variables such as discount rates, fees, or product pricing based on promotions, time of year, etc. It might also involve modifications to business algorithms or decision tables. Business processes need to be able to quickly adapt to these changes, without time, cost, and resource-intensive development and maintenance work.

The Business Rules capability of the WebSphere Process Server can provide that type of business flexibility, where business variables and formulas can be loosely-coupled or externalized from the business processes. This feature enables quick and easy modifications to business rules without recoding or redeployment.

The Business Rules feature of the WebSphere Process Server also adheres to the building-block approach, where these business rules are modularized and treated just like any other service component. This will be illustrated in this lab where you will quickly change the Credit Risk Assessment task from Java code to Business Rules.

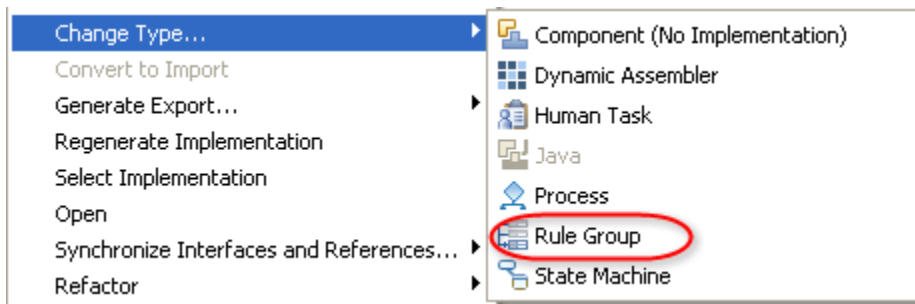
### 8.1 Change the Credit Risk Assessment Task to a Business Rule

- \_\_\_1. From the **Assembly Diagram** editor, right-click on the **CreditRiskAssessment** component on the right side.





\_\_2. From the popup menu, select **Change Type -> Rule Group**.

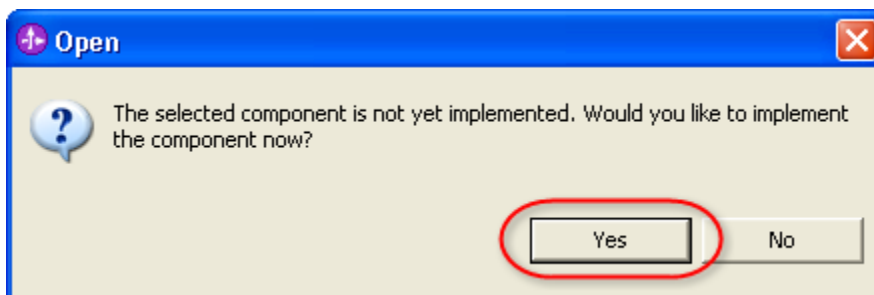


This will change the implementation type from Java to a Rule Group. The logic to determine whether the risk assessment rating is HIGH or LOW, is currently in Java code. It will now be replaced by a Business Rule for better flexibility.

\_\_3. Double-click on the **CreditRiskAssessment** component.

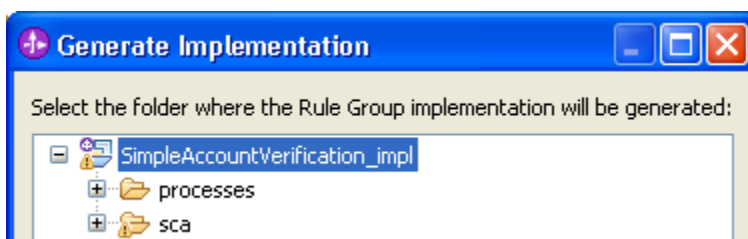


\_\_4. From the prompt window, click on **Yes**.



An empty Business Rule Group will be created. The next step is to define the specific rules to determine if the credit risk assessment rating should either be HIGH or LOW based on the credit score.

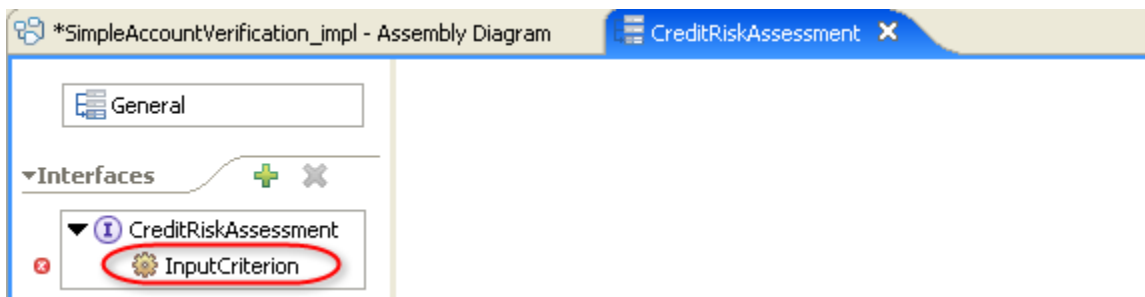
\_\_5. From the **Generate Implementation** window, accept the default selection, and then click on **OK**.



The Business Rule Group editor appears.

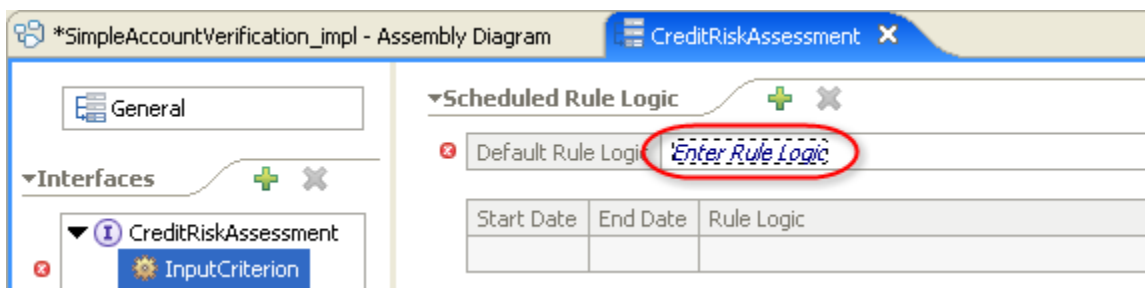
## 8.2 Define the Business Rule Set

\_\_6. From the **Rule Group** editor, click on the **InputCriterion** Interface.



A Rule Group contains related Business Rules. In other words, a Rule Group is a logical grouping of related Rules.

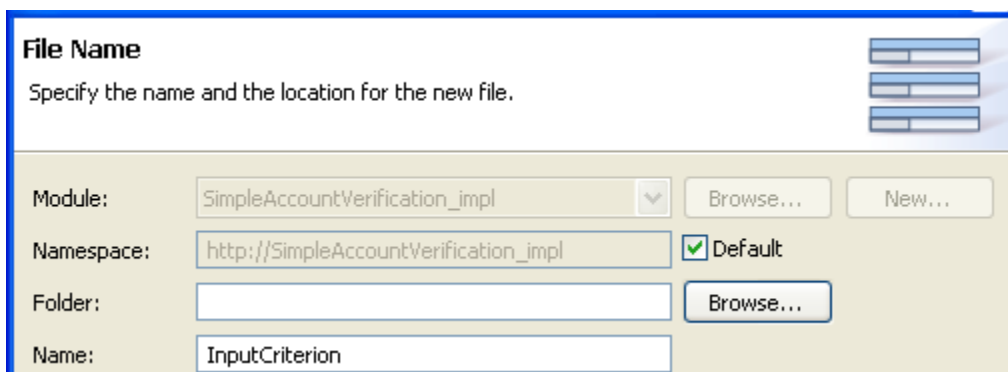
\_\_7. Click on the **Enter Rule Logic** link.



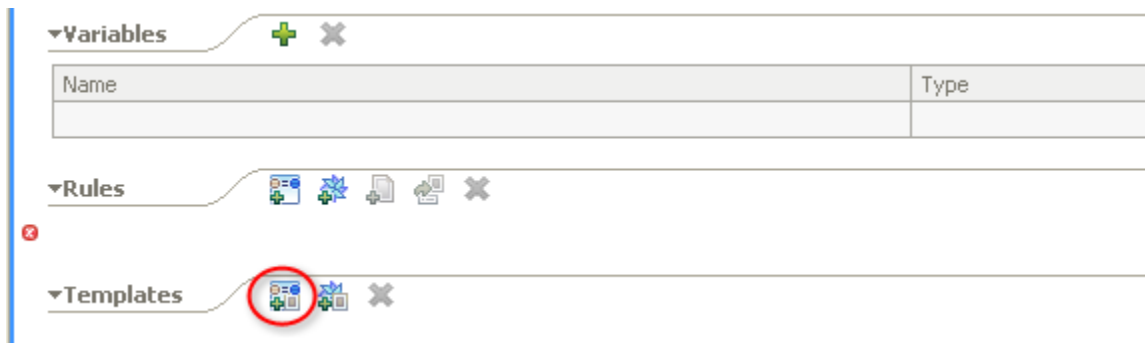
\_\_8. Select **New rule set** from the popup list.



\_\_9. From the **New Rule Set** window, accept the defaults, and then click on **Finish**.



\_\_10. From the Rule Set editor, click on the **Add If Then Template** icon in the Templates section.

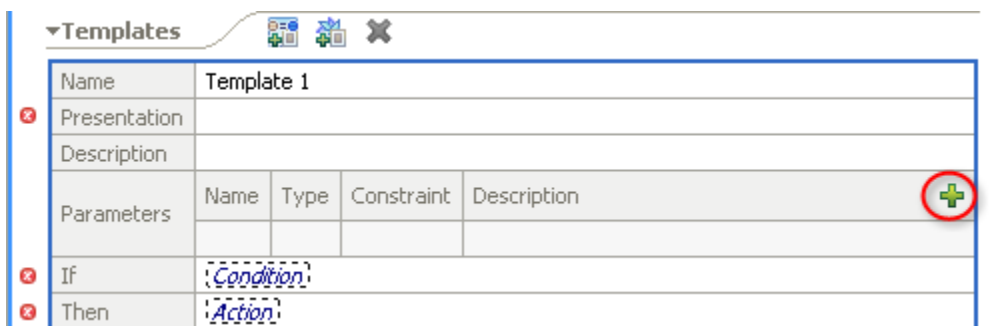


### What's next?



The next step is to define a business rule where the risk assessment will be rated “HIGH” if the credit score from the CreditReport web service is less than a certain threshold. In the previous Java implementation, the threshold was fixed at 500. In this business rule, it will be a variable that can be changed at runtime.

\_\_11. In the **Templates** section, click on the **Add Template Parameter** icon.



\_\_12. Change the Parameter Name from **param1** to **threshold**.

Parameters	Name	Type	Constraint	Description
	threshold	Select Type	None	
If	{Condition}			
Then	{Action}			

\_\_13. Click on the **Select Type** link.

Parameters	Name	Type	Constraint	Description
	threshold	Select Type	None	
If	{Condition}			
Then	{Action}			

\_\_14. From the popup list, select **int**.

Parameters	Name	Type	Constraint	Descri
	threshold	Select Type <span style="color:red">✖</span>	None	
If	<a href="#">Condition</a>	<ul style="list-style-type: none"> <li> string</li> <li> int</li> <li> double</li> <li> boolean</li> </ul>		
Then	<a href="#">Action</a>			

Take note of the red 'x' marks (✖) highlighted below. This indicates that errors exist in those fields. That is to be expected at this point because the proper values still have to be specified.

	Name	Template 1			
<span style="color:red">✖</span>	Presentation				
	Description				
Parameters	Name	Type	Constraint	Description	
	threshold	int	None		
<span style="color:red">✖</span>	If	<a href="#">Condition</a>			
<span style="color:red">✖</span>	Then	<a href="#">Action</a>			

\_\_15. Click on the **Condition** link in the **If** field.

Parameters	Name	Type	Constraint	Description
	threshold	int	None	
<span style="color:red">✖</span>	If	<a href="#">Condition</a>		
<span style="color:red">✖</span>	Then	<a href="#">Action</a>		

\_\_16. Ignore the popup list and just type the following into the **If** field:

**Input.creditScore<threshold**

Parameters	Name	Type	Constraint	Description
	threshold	int	None	
	If	Input.creditScore<threshold		
<span style="color:red">✖</span>	Then	<a href="#">Action</a>		

The red 'x' mark (✖) beside the **If** field should disappear if the Condition value was specified correctly.

\_\_17. Click on the **Action** link in the **Then** field.

Parameters	Name	Type	Constraint	Description
	threshold	int	None	
	If	Input.creditScore<threshold		
<span style="color:red">✖</span>	Then	<a href="#">Action</a>		

\_\_18. Ignore the popup list and type the following into the **Then** field:

**Output.creditRiskAssessment="HIGH"**

Parameters	Name	Type	Constraint	Description
	threshold	int	None	
If	Input.creditScore<threshold			
Then	Output.creditRiskAssessment="HIGH"			

The red 'x' mark (✖) beside the **Then** field should disappear if the Condition value was specified correctly.

\_\_19. Type the following into the **Presentation** field:

**If credit score is less than {threshold} then risk assessment is HIGH**

Name	Template 1
Presentation	If credit score is less than {threshold} then risk assessment is HIGH
Description	

#### Hint

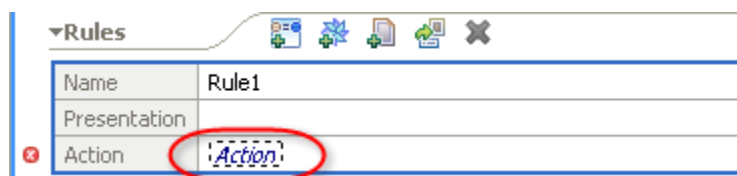


The text specified in the Presentation field will be displayed in the Business Rules Manager user interface. The Business Rules Manager will allow you to dynamically change the parameter values in the template rule while the process is running. In this case, the threshold parameter can be changed at runtime to affect how the risk assessment rating is determined, without having to modify, retest, and redeploy the process. This will be illustrated later in the lab.

\_\_20. In the **Rules** section, click on the **Add Action Rule** icon.

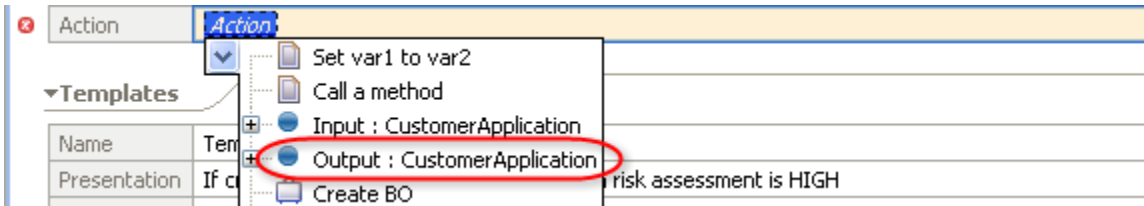


\_\_21. Click on the **Action** link in the **Action** field.

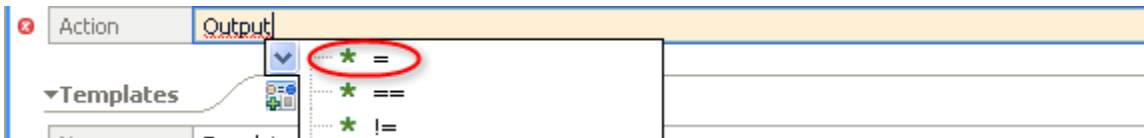


In the previous steps, you ignored the popup list which appeared when a field link was clicked to keep the steps simple. However, the popup lists can also be very useful in supplying the proper values for the fields. Let's try it in the next step.

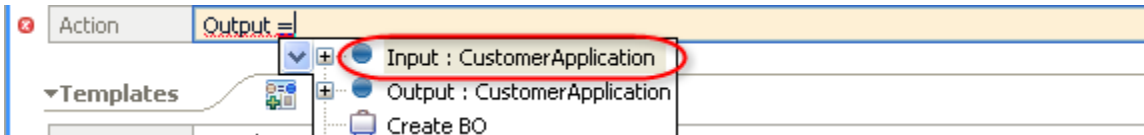
- \_\_22. From the popup list, select **Output : CustomerApplication**. (**Output** is a property from the Rule Set Interface, and **CustomerApplication** is the property type.)



- \_\_23. From the popup list, select the '=' operation.



- \_\_24. From the popup list, select **Input : CustomerApplication**.



The Action field should now look similar to this :

Name	Rule1
Presentation	
Action	Output = Input



**Hint**

This will copy the contents of the Input variable to the Output variable. The Input variable is a parameter passed into this Rule Set when invoked. The Output variable will be passed back to the component invoking this Rule when execution is complete.

- \_\_25. While the cursor is still in the **Action** field, press the **Enter** key. This will add another **Action** link.

- \_\_26. Ignore the popup list again and type the following into the second line of the **Action** field:  
**Output.creditRiskAssessment="LOW"**


Name	Rule1
Presentation	
Action	Output = Input Output.creditRiskAssessment="LOW"

- \_\_27. Click on the **Add Template Rule** icon in the **Rules** section.

▼Rules 

Name	Rule1
Presentation	
Action	Output = Input Output.creditRiskAssessment="LOW"

- \_\_28. From the popup list, select **Template 1**.

▼Rules 

Name	Rule1
Presentation	
Action	Output = Input Output.creditRiskAssessment="LOW"

Template 1

This will create an empty Rule based on the Rule Template created earlier.


- \_\_29. Specify a value of **500** for the **threshold** variable in the **Presentation** field.

Name	Rule2
Template	Template 1
Presentation	If credit score is less than <b>500</b> then risk assessment is HIGH

The Business Rule is now complete. This Rule Set basically recreates the logic in the previous Java implementation where the risk assessment will be HIGH if the credit score is less than the current threshold level of 500. The significant difference is that this was done without any programming, and the threshold value can be changed dynamically at runtime.



**Important!**

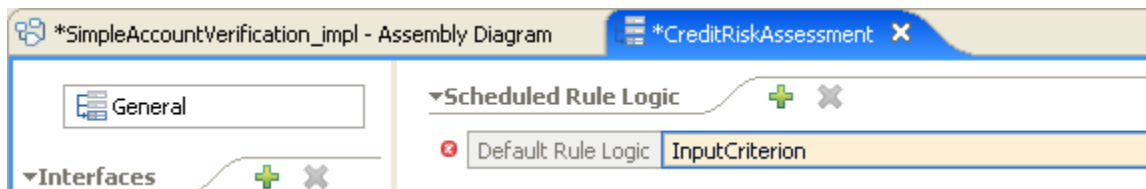
There should no longer be any red 'x' marks (  ).

### 8.3 Save and Verify the Business Rule

\_\_30. Press **Ctrl+s** to save the **Rule Set**.

\_\_31. Close the **InputCriterion** Rule Set editor. 

Focus should return to the **CreditRiskAssesment** Business Rule Group.




\_\_32. Press **Ctrl+s** to save the Rule Group.

\_\_33. Close the **CreditRiskAssesment** Rule Group editor. 

Focus should return to the **Assembly Diagram** editor.

\_\_34. From the **Assembly Diagram** editor, press **Ctrl+s** to save the modified assembly. Do not close the Assembly Diagram editor.

\_\_35. Switch to the **Problems** view. Verify that no errors exist (messages with a red 'x' mark ) . Warnings messages are expected.

### 8.4 Retest the SimpleAccountVerification Process

\_\_36. Switch to the **Servers** view.

The WebSphere Process Server should now have a state of **Republish**.

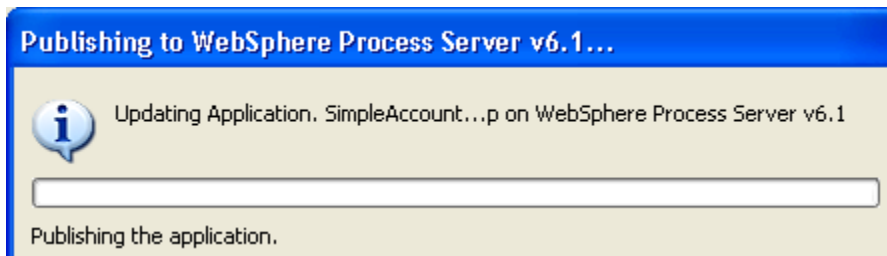
Server	Status	State
WebSphere Business Monitor Server v6.1 on W	Stopped	Republish
WebSphere Process Server v6.1	Started	Republish
SimpleAccountVerificationApp	Started	

\_\_37. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Publish**.





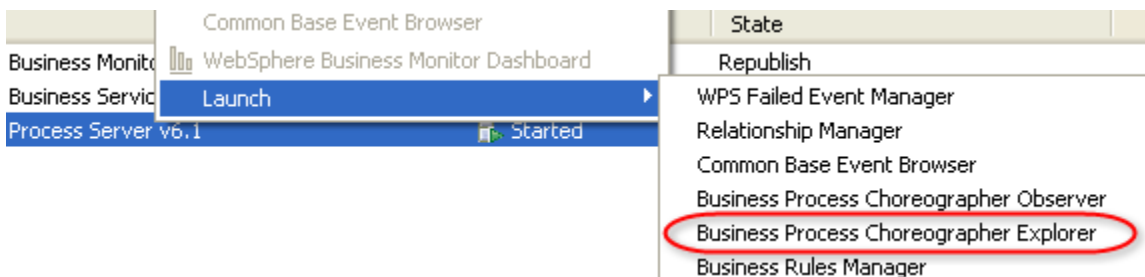
\_\_38. Wait for the progress indicator window to disappear.



\_\_39. Verify that the state of the WebSphere Process Server is now **Synchronized**.

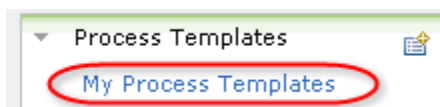
Server	Status	State
WebSphere Business Monitor Server v6.1 on W	Stopped	Republish
WebSphere Process Server v6.1	Started	<b>Synchronized</b>
SimpleAccountVerificationApp	Started	

\_\_40. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Launch -> Business Process Choreographer Explorer**.

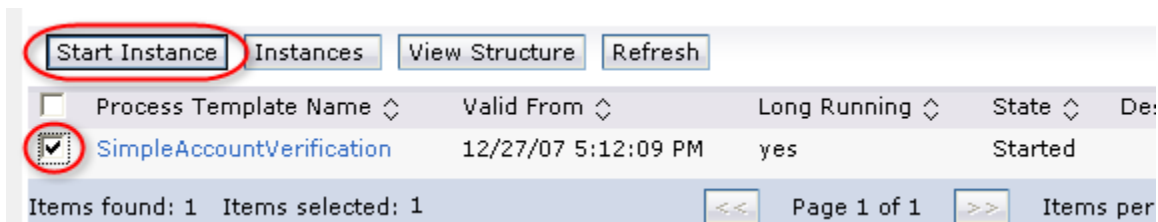


The Business Process Choreographer Explorer appears.

\_\_41. Click on the **My Process Templates** link to display a list of processes which can be started.



\_\_42. Place a checkmark beside **SimpleAccountVerification**. Click on **Start Instance**.



This will start the SimpleAccountVerification process, and display the default input page.

\_\_43. From the default input page, click on **Add**.

**Form View**

Input	applicationID	<input type="text"/>
	applicationDate	<input type="text"/>
	pricing	<input type="text"/>
	requestedLimit	<input type="text"/>
	customer	<input type="button" value="Add"/>

\_\_44. For the **customerID** field, type **123**.

customer	customerID	<input type="text" value="123"/>
	companyName	<input type="text"/>



**Hint**

A Customer ID of “123” will result in a low credit score and a denied request. Any other Customer ID value will result in an approved request.

\_\_45. Click on **Submit**.

After a few moments, the SimpleAccountVerification process will generate messages to the **Console** view.

\_\_46. Verify that the request is denied.

```

○ *****
○ ** Call Internal Credit Report Service **
○ *****
○ >>>> Credit Score = 449
○ *****
○ >>>> Request is denied!.

```

A Customer ID of “**123**” again resulted in a low credit score and a denied request. The difference in this case is that a Business Rule was used instead of Java code. The Business Rule determined that the credit risk assessment was **LOW** because the credit score was below the current threshold of **500**.



**What's next?**

Because the credit risk assessment is now implemented as a Business Rule, the next few steps will illustrate how this provides greater business agility.

\_\_47. Close the Business Process Choreographer Explorer.



## 8.5 Modify the Credit Assessment Business Rule at Runtime

\_\_48. Click on a web browser icon in the **Quick Launch** bar.



\_\_49. From the web browser window, open the URL <http://localhost:9080/br> .  
The Business Rules Manager user interface appears.



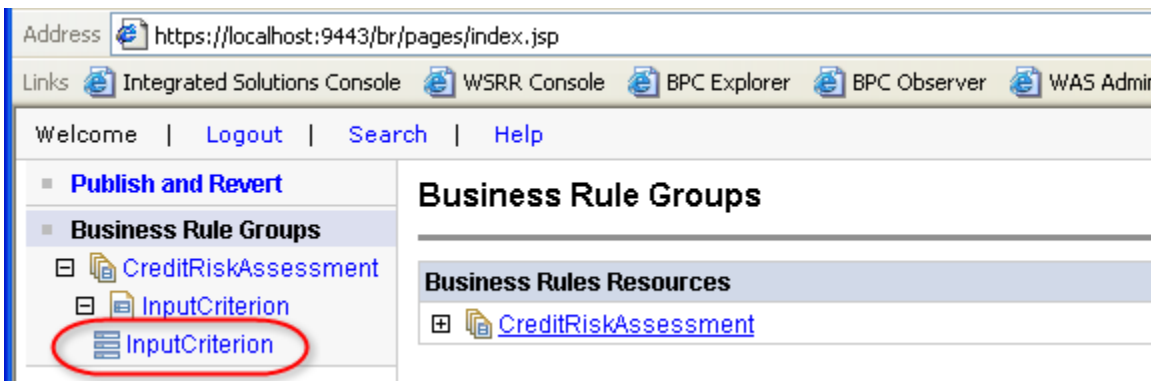
**Hint**

The Business Rules Manager UI can also be used inside the WebSphere Integration Developer. Just follow these steps:


- Switch to the Servers view. Right-click on WebSphere Process Server v6.1.
- From the popup menu, select Launch -> Business Rules Manager.

Using a standard web browser, an authorized person such as a manager or business analyst, can dynamically change the behavior of the business process at runtime by modifying the externalized business rules.

\_\_50. From the **Rule Books** tree view on the left, click on the lowest **InputCriterion**.



**Hint**



The logon window did not appear because security is disabled in the test server. In a secure production environment, users will be required to log in and business rules can only be accessed by authorized users.

\_\_51. From the **InputCriterion - Rule Set** pane, click on **Edit**.

**InputCriterion - Rule Set**

Back **Edit** Copy

---

**General Information**


<b>Last Published</b>	Jan 1, 2008 18:36 (Local Time)
<b>Description</b>	

---

**Rules**

Display Name	Rule
Rule2	If credit score is less than <b>500</b> then risk assessment is HIGH

**What's next?**



This is the rule set for the Credit Risk Assessment Business Rule. The current threshold of 500 will be lowered so that the request of Customer ID 123 will be accepted.

\_\_52. From the Rule set editor, change the threshold value to **300**.

**Rules**

New Rule from Template

Name	Display Name	Rule
Rule2	Rule2	If credit score is less than <b>300</b> then risk assessment is HIGH

\_\_53. Click on **Save**.

\_\_54. Click on the **Publish and Revert** link.

Welcome | Logout | Search | Help

**Publish and Revert**

Business Rule Groups

CreditRiskAssessment

**InputCriterion - Rule Set**

Back Edit Copy

Messages: "InputCriterion" has been published. You may publish the changes.

A list appears showing Business Rules Resources that were changed. You can either revert the changes back to its original state, or publish the changes.

\_\_55. Ensure that **InputCriterion** is selected.

Select business rule groups and rule schedules to publish.

↓	Resources	Status	Description
	<a href="#">CreditRiskAssessment</a>	Original	
<input checked="" type="checkbox"/>	<a href="#">InputCriterion</a>	Original	
	<a href="#">InputCriterion</a>	Local Change	

\_\_56. Click on **Publish**.

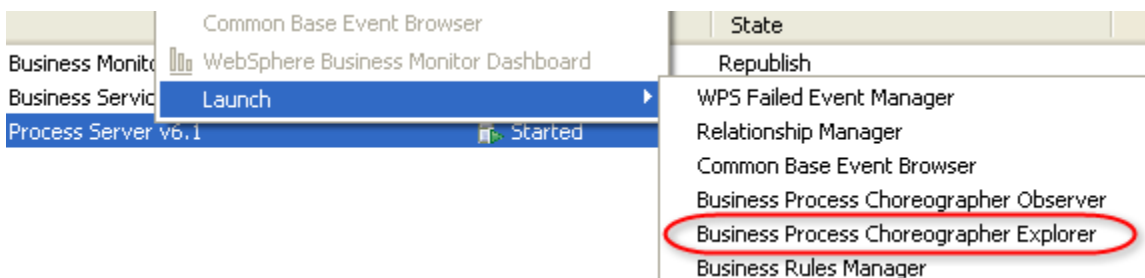


The **Messages** box should indicate that the rule change was published successfully.

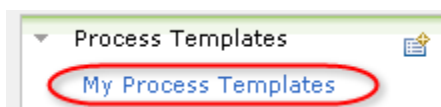
Messages:

\_\_57. Switch to the **WebSphere Integration Developer**.

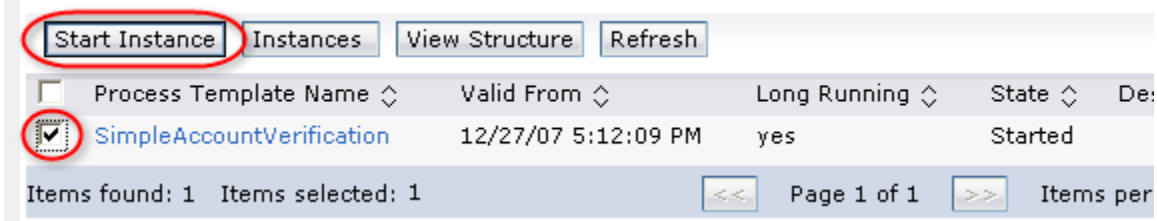
\_\_58. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Launch -> Business Process Choreographer Explorer**.



\_\_59. Click on the **My Process Templates** link to display a list of processes which can be started.



\_\_60. Place a checkmark beside **SimpleAccountVerification**. Click on **Start Instance**.



\_\_61. From the default input page, click on **Add**.

\_\_62. For the **customerID** field, type **123**.

customer	customerID	<input type="text" value="123"/>
	companyName	<input type="text"/>

\_\_63. Click on **Submit**.


\_\_64. After a few moments, the SimpleAccountVerification process will generate messages to the **Console** view indicating that the request is now approved.

\_\_65. Verify that the request has been approved.

```

O *****
O ** Call Internal Credit Report Service **
O *****
O >>>> Credit Score = 449
O *****
O >>>> Request is approved!.
    
```

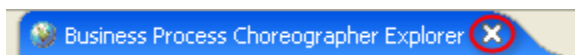
A Customer ID of “123” now resulted in an accepted request because the threshold level in the Credit Risk Assessment Business Rule was lowered to 300. Significant cost reductions can result from this type of business and IT flexibility.



**What's next?**  
The threshold variable of the business rule will be reset to its original value of 500.

## 8.6 Reset the Credit Risk Assessment Business Rule

\_\_66. Close the Business Process Choreographer Explorer.



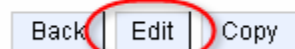
\_\_67. Switch to the web browser with the Business Rules Manager web page.

\_\_68. From the **Rule Books** tree view on the left, click on the lowest **InputCriterion**.



\_\_69. From the **InputCriterion - Rule Set** pane, click on **Edit**.

### InputCriterion - Rule Set



\_\_70. From the Rule Set editor, change the threshold value to **500**.

Name	Display Name	Rule
Rule2	Rule2	If credit score is less than 500 then risk assessment is HIGH

\_\_71. Click on **Save**.

\_\_72. Click on the **Publish and Revert** link.



A list appears showing Business Rules Resources that were changed. You can either revert the changes back to its original state, or publish the changes.

\_\_73. Ensure that **InputCriterion** is selected.

Select business rule groups and rule schedules to publish.

↓	Resources	Status	Description
	<a href="#">CreditRiskAssessment</a>	Original	
<input checked="" type="checkbox"/>	<a href="#">InputCriterion</a>	Original	
	<a href="#">InputCriterion</a>	Local Change	

\_\_74. Click on **Publish**.

**Publish and Revert**

Back **Publish** Messages:

The **Messages** box should indicate that the rule change was published successfully.

Messages:

\_\_75. Close the web browser with the **Business Rules Manager** page.



**Please wait for the next lecture before proceeding to the next lab.**



## Lab 9 Modify the Process Choreography

### Goals:

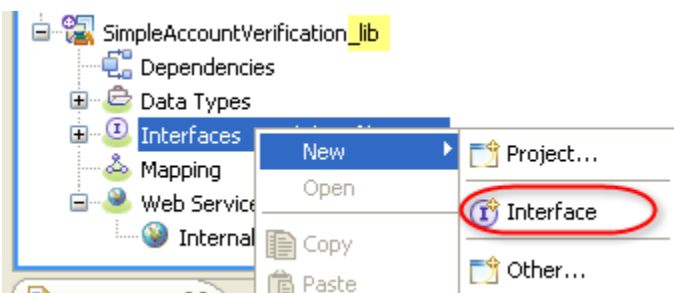
- Add a task to the BPEL process

#### Role: Integration Developer

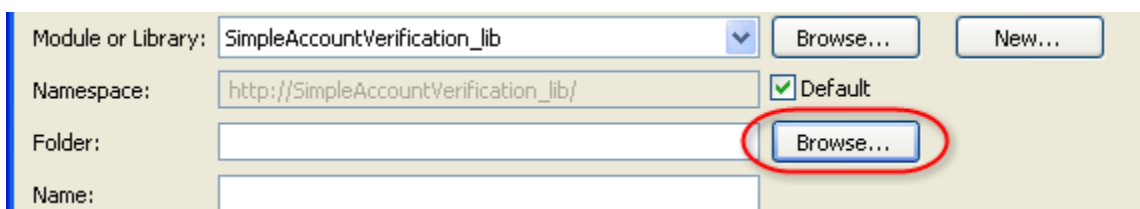
The SimpleAccountVerification BPEL process will be modified to add a new task called Generate Acceptance. The Generate Acceptance task is being added simply to display output from the Pricing and Approval human task. This will also help introduce BPEL development.

### 9.1 Define the Interface for the new 'Generate Acceptance' task

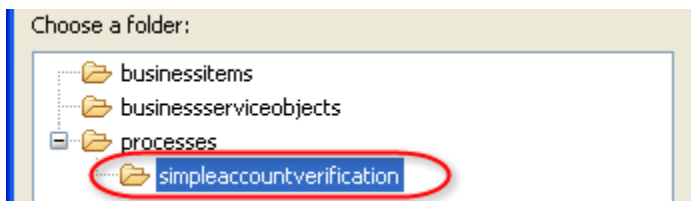
1. From the **Business Integration** tree view, in the SimpleAccountVerification\_lib module, right-click on **Interfaces**. From the popup menu, select **New -> Interface**.



2. Click on the **Browse** button for the Folder field.



3. Select the **simpleaccountverification** folder. Click on **OK**.



\_\_4. Specify **GenerateAcceptance** for the Name. Click on **Finish**.

Module or Library: SimpleAccountVerification\_lib [Browse... New...]  
 Namespace: http://SimpleAccountVerification\_lib/processes/simplea [Default]  
 Folder: processes/simpleaccountverification [Browse...]  
 Name: **GenerateAcceptance**

\_\_5. From the Interface editor, click the **Add Request Response Operation** icon.

SimpleAccountVerification\_impl - Assembly Diagram | GenerateAcceptance

Operations and their parameters

Name	Type
generateAcceptance	
Input(s)	input1
Output(s)	output1

\_\_6. Specify **generateAcceptance** as the operation name.

Name	Type
<b>generateAcceptance</b>	
Input(s)	input1
Output(s)	output1

\_\_7. Click on the **string** link for input1.

Name	Type
generateAcceptance	
Input(s)	input1 <a href="#">string</a>
Output(s)	output1

**What's next?**

When the generateAcceptance operation is invoked, a parameter variable named input1 will be passed. By default, the parameter input1 is defined as a string type. This will need to be changed to use the CustomerApplication object type instead.

\_\_8. From the popup list, select **CustomerApplication**.

string

- time
- Businessitem1 http://
- Customer http://
- CustomerApplication** http://

\_\_9. Change also the type of **output1** to **CustomerApplication**.

Name	Type
input1	CustomerApplication
output1	CustomerApplication



**What's next?**

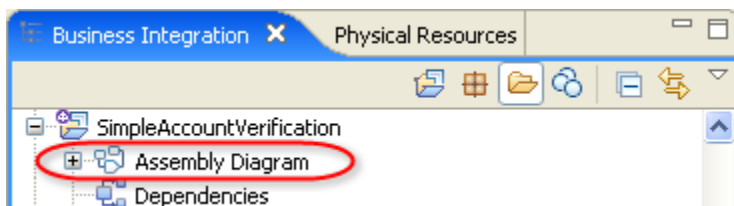
During execution, when the generateAcceptance operation has completed, a result variable named output1 will be passed back to the calling component.

\_\_10. Press **Ctrl+s** to save the new Interface.

\_\_11. Close the **GenerateAcceptance** Interface editor.

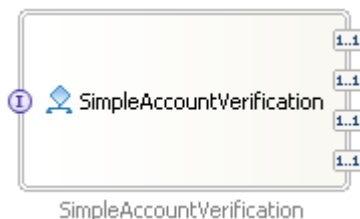
## 9.2 Add the GenerateAcceptance Task to the SimpleAccountVerification BPEL Process

\_\_12. In the SimpleAccountVerification main module, double-click on **Assembly Diagram**.

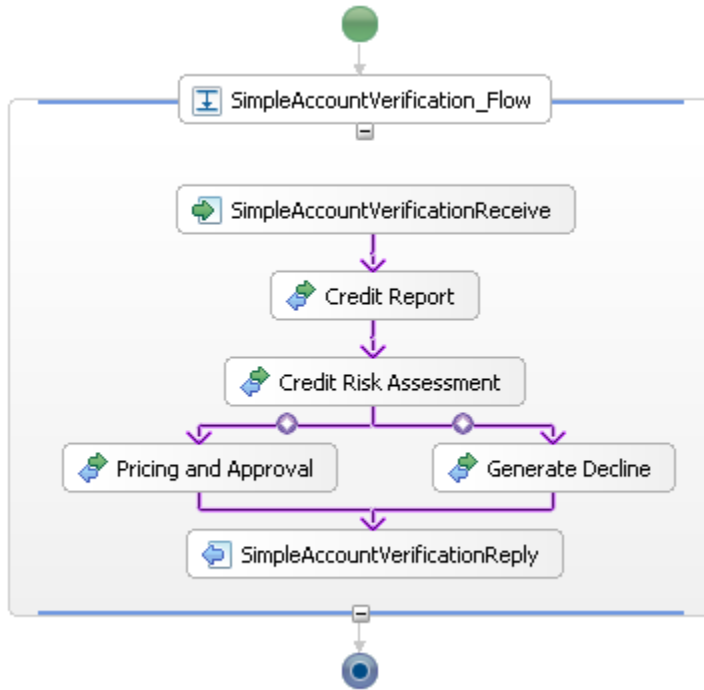


This will open the main assembly diagram containing the BPEL process.

\_\_13. From the **Assembly Diagram** editor, double-click on the **SimpleAccountVerification** component.

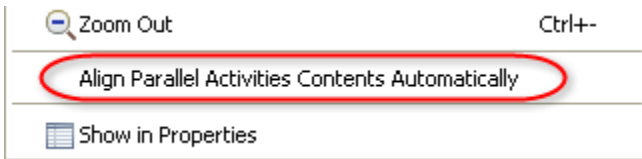


The **SimpleAccountVerification** BPEL editor will appear.



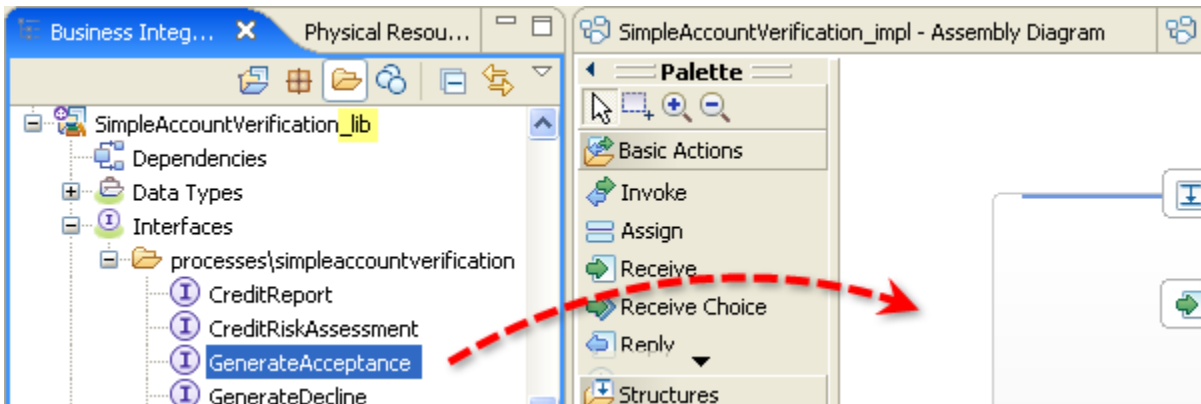
This is the BPEL process based on the **SimpleAccountVerification** business model developed earlier using the WebSphere Business Modeler.

- \_\_14. Right-click on an empty space in the BPEL editor. From the popup menu, select **Align Parallel Activities Contents Automatically**.

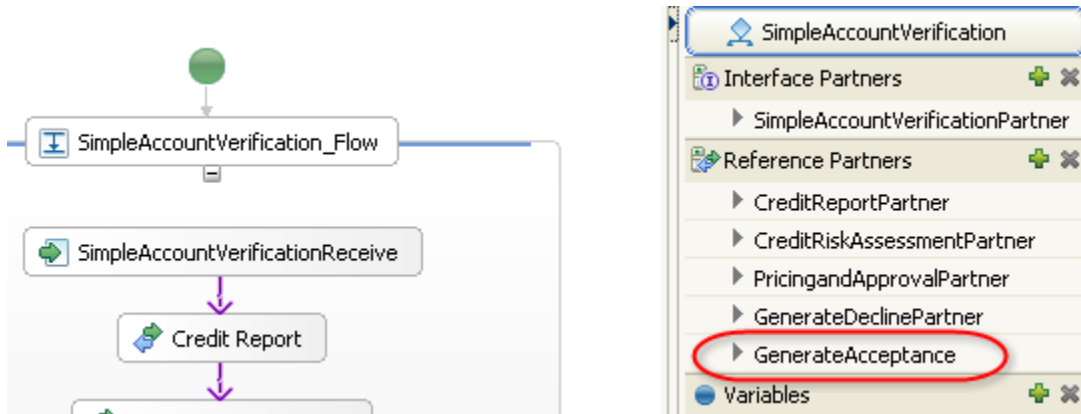


This option will make the editor automatically adjust the BPEL diagram to the optimum layout.

- \_\_15. From the SimpleAccountVerification\_lib module, and using the left mouse button, drag and drop the **GenerateAcceptance** Interface to an empty space in the BPEL editor.



- \_\_16. Verify that the **GenerateAcceptance** Interface was added to the list of Reference Partners. It can now be used by any task or activity in the BPEL process.

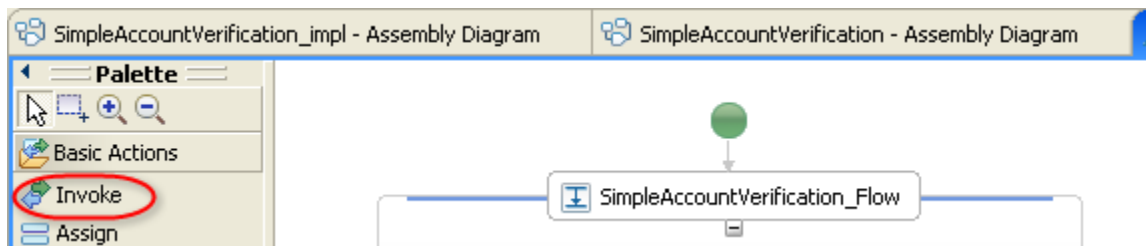


**Hint**



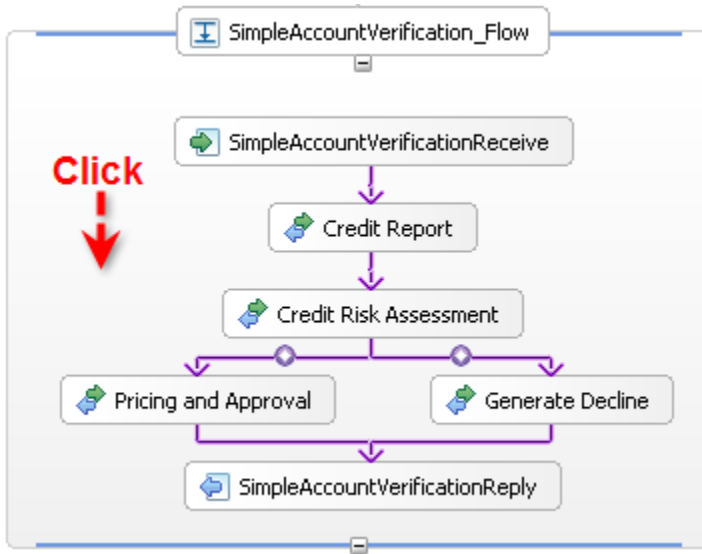
A Reference Partner is a soft link to the actual service endpoint. This is how the WebSphere Integration Developer enables loose-coupling between the BPEL process and the implementation of its activities. The developer is not forced to specify that the Credit Report task is a web service or some specific implementation type. This is done outside of the BPEL process and in the assembly diagram. This provides the flexibility needed to easily make changes.

- \_\_17. From the **Activity Palette** of the BPEL editor, click on the **Invoke** icon.

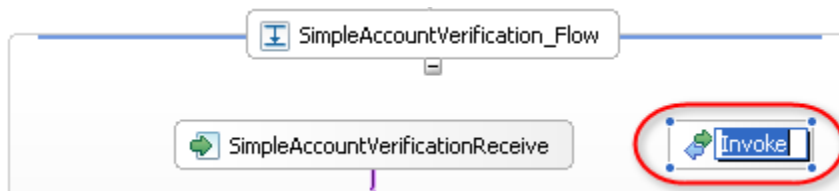


This will load the cursor with the **Invoke** activity and can then be dropped on BPEL diagram.

\_\_18. Click on an empty space inside the SimpleAccountVerification flow.



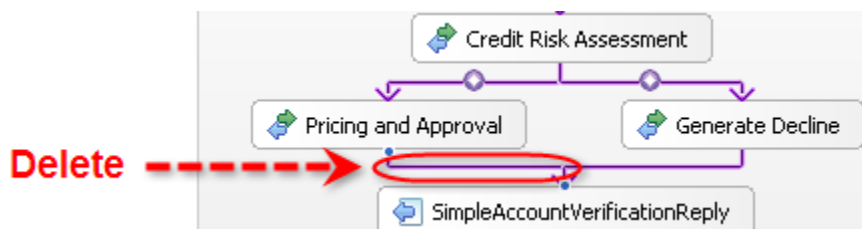
This will add a new Invoke activity to the BPEL diagram.

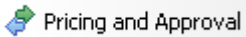


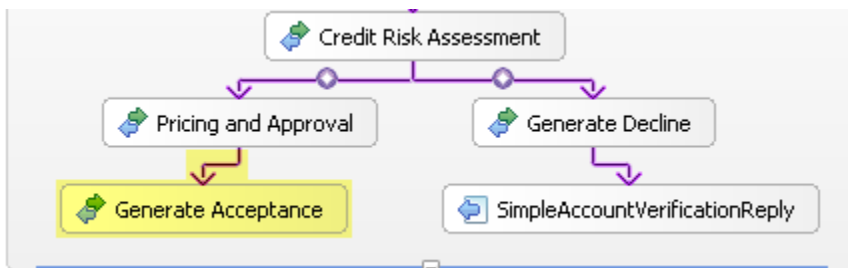
\_\_19. Change the **name** of the new Invoke activity to **Generate Acceptance**.

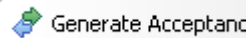


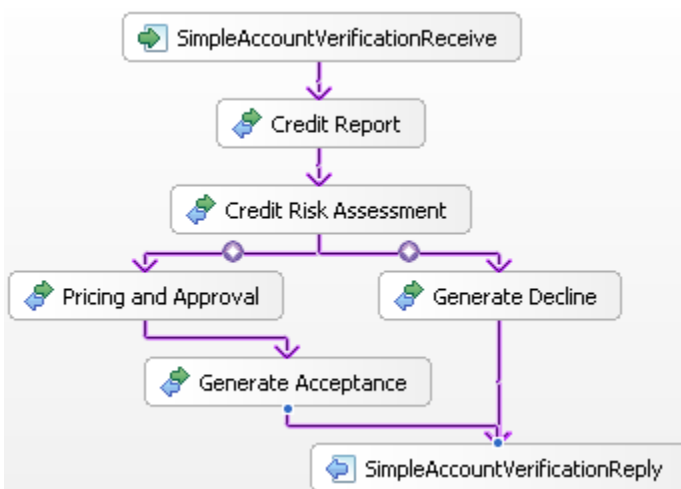
\_\_20. Select the link between the **Pricing and Approval** activity and the **SimpleAccountVerificationReply** activity. Press the **Delete** key.



- \_\_21. Right-click on the **Pricing and Approval** activity. 
- \_\_22. From the popup menu, select **Add Link**. This will start a connection.
- \_\_23. Click on the **Generate Acceptance** activity to complete the connection.



- \_\_24. Right-click on the **Generate Acceptance** activity. 
- \_\_25. From the popup menu, select **Add Link**.
- \_\_26. Click on the **SimpleAccountVerificationReply** activity to complete the connection.
- \_\_27. Verify that the SimpleAccountVerification process looks similar to the screenshot below.

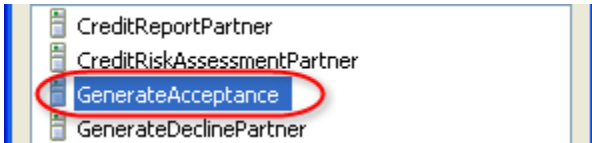


**Troubleshooting**

If needed, press Ctrl+z or select Edit -> Undo to undo your changes.

\_\_28. Right-click on the **Generate Acceptance** activity. From the popup menu, select **Set Partner**.

\_\_29. From the Select a Partner window, select **GenerateAcceptance**. Click on **OK**.



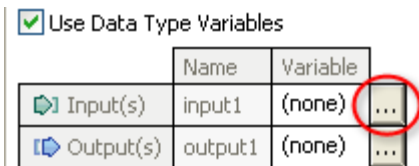
\_\_30. Right-click again on the **Generate Acceptance** activity.

\_\_31. From the popup menu, select **Show in Properties**. This will display the **Properties** view for the Generate Acceptance activity in the bottom of the screen.

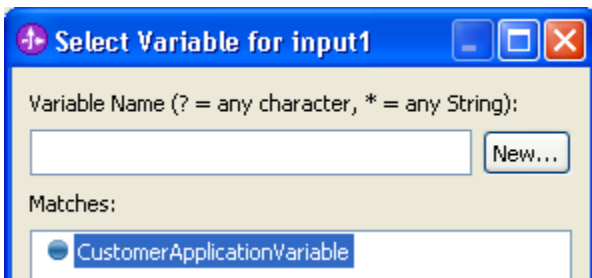
\_\_32. Click on the **Details** tab.



\_\_33. Click on the **More** (...) button for the **Input** field.



\_\_34. Ensure that **CustomerApplicationVariable** is selected. Click on **OK**.



This will specify the CustomerApplicationVariable for the Input.

\_\_35. Perform the same steps to also specify **CustomerApplicationVariable** for the **Output**.



Use Data Type Variables

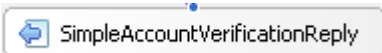
	Name	Variable	
Input(s)	input1	CustomerApplicationVariable	...
Output(s)	output1	CustomerApplicationVariable	...



### What just happened?

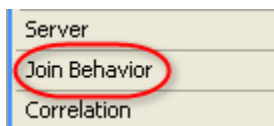
You configured the GenerateAcceptance activity to use the CustomerApplication business object for both input and output.

\_\_36. Right-click on the SimpleAccountVerificationReply activity.

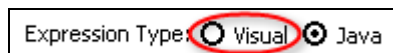


\_\_37. From the popup menu, select **Show in Properties**.

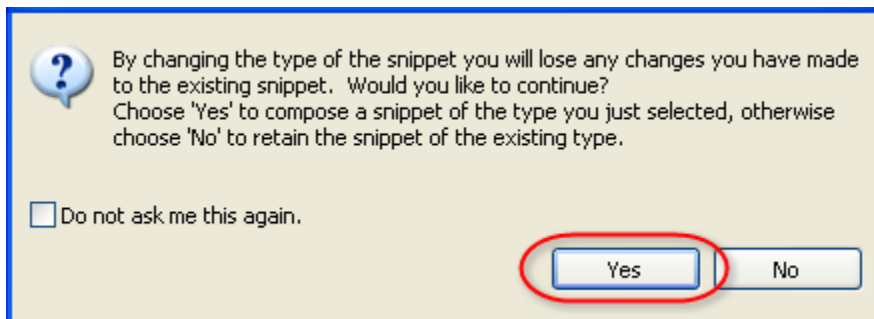
\_\_38. From the **Properties** view, select the **Join** tab.



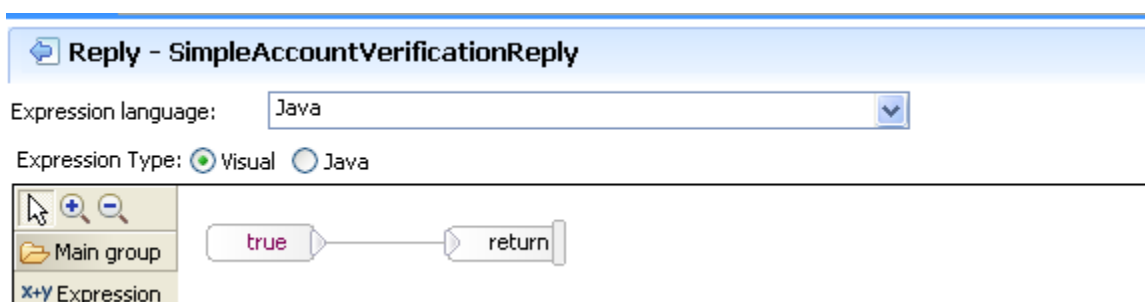
\_\_39. Select the **Visual** option for the **Expression Type**.



\_\_40. From the **Question** window, click on **Yes**.



The **Visual Snippet Editor** appears which allows you to implement logic or expressions without having to know Java programming.





**Hint**

The default expression in the Visual Snippet Editor simply indicates that a boolean value of true will be returned to the caller of the SimpleAccountVerification process. At this point, no changes are necessary.

\_\_41. Press **Ctrl+s** to save the changes to the BPEL process.

\_\_42. Close the **SimpleAccountVerification** BPEL editor.



**Hint**

Notice that as you were developing the BPEL process, the actual implementations for the tasks in the process were never specified. You only associated the tasks to references to external implementations. This provides the loose-coupling characteristic needed for effective integration. The actual implementations are defined externally using the Assembly Diagram editor. This approach also provides a level of abstraction to components such as the BPEL process for greater IT flexibility.

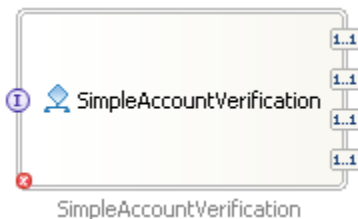


**What just happened?**

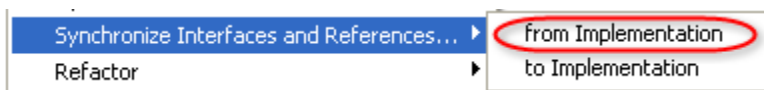
The Generate Acceptance activity was added to the flow of the SimpleAccountVerification process. However this activity has not yet been linked to its implementation. Only a soft link was specified. The hard link will be specified next.

### 9.3 Add the implementation for the new Generate Acceptance task

\_\_43. From the **Assembly Diagram** editor, right-click on the **SimpleAccountVerification** component.




\_\_44. From the popup menu, select **Synchronize Interfaces and References -> from Implementation**.



\_\_45. From the confirmation window, click **Yes**.


**What just happened?**



This changed the assembly diagram to reflect the new task added to the BPEL process. A new reference point was added to the right of the BPEL process component for the Generate Acceptance activity. This also fixed the mismatch error and removed the red 'x' mark (✖) on the SimpleAccountVerification component.

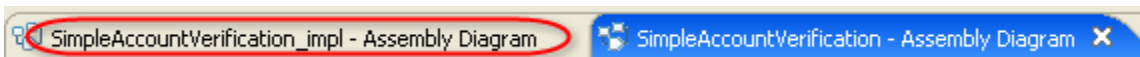


**What's next?**

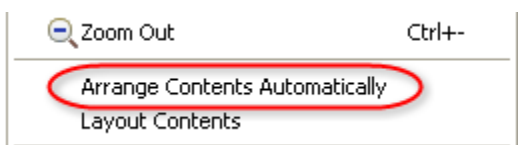


The next step is to create an implementation for the Generate Acceptance activity. This will be done in the implementation module. You will then export this implementation to the BPEL process assembly diagram in the main module. The exported implementation will then be linked to the Generate Acceptance reference point.

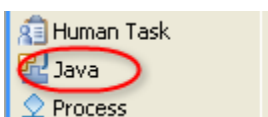
\_\_46. Switch to the Assembly Diagram editor for the implementation module.



\_\_47. Right-click on an empty space in the canvas. From the popup menu, select **Arrange Contents Automatically**.



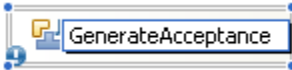
\_\_48. From the Palette on the left, select the **Java** component.



- \_\_49. Click on an empty space in the canvas to drop the Java component.  
The Java component appears in the canvas.



- \_\_50. Single-click on the new **Java** component to edit the name. Change the name to **GenerateAcceptance**.



**Hint**

To rename the Java component, you can also right-click on the component, and then select Rename from the popup menu.

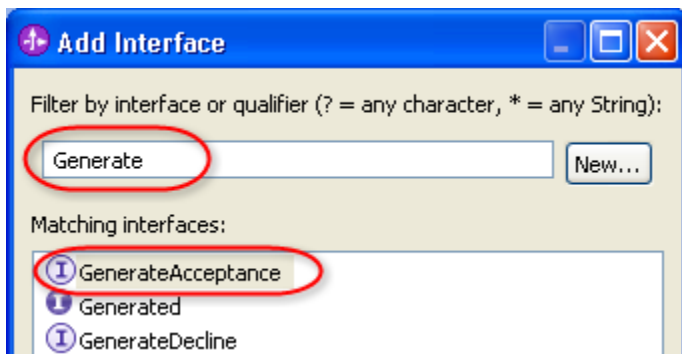
- \_\_51. Click on the **Add Interface** icon displayed above the component.




**Troubleshooting**

If the Add Interface icon is not displayed, right-click on the GenerateAcceptance component, and then select Add -> Interface from the popup menu.

- \_\_52. Specify the text **Generate** as the Filter criteria, and then select **GenerateAcceptance** from the matching interfaces list.



\_\_53. Click on **OK**.



**What just happened?**

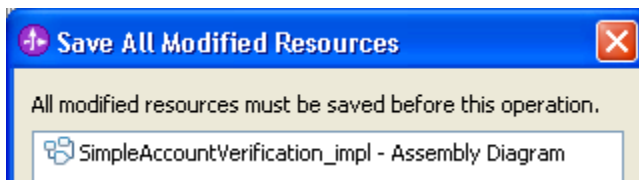
The newly added Java component can now be invoked using the operations and input/output objects defined in the GenerateAcceptance interface. Code will also be generated based on this added interface.

\_\_54. Right-click on the **GenerateAcceptance** Java component.

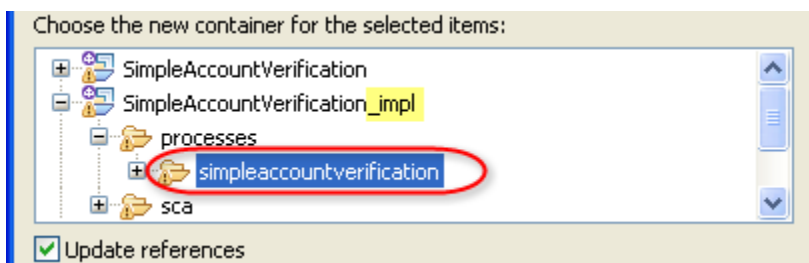


\_\_55. From the popup menu, select **Refactor -> Move**.

\_\_56. From the prompt window, click on **OK** to save the assembly diagram before refactoring.



\_\_57. From the container list, expand **SimpleAccountVerification\_impl** to select **simpleaccountverification**.

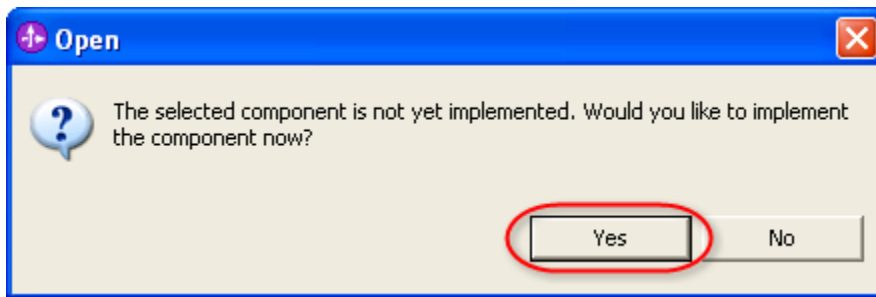


\_\_58. Click on **OK** to indicate that the GenerateAcceptance Java component will be created in the selected folder.

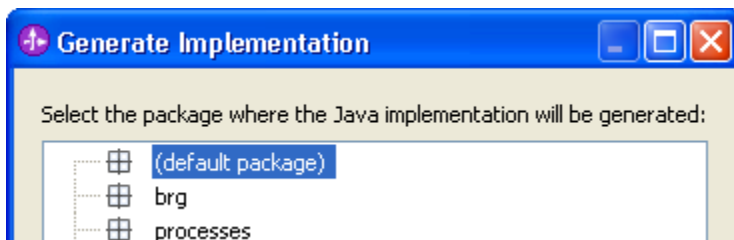
\_\_59. Double-click on the **GenerateAcceptance** component.



\_\_60. From the **Open** confirmation window, click on **Yes**.

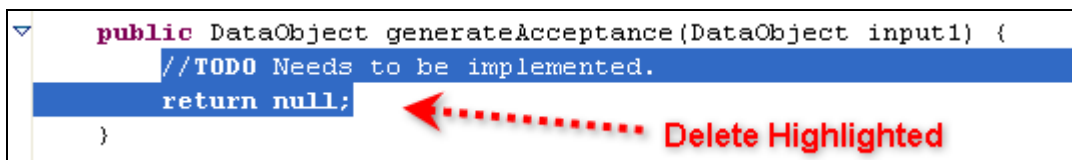


\_\_61. From the **Generate Implementation** window, accept the default selection, and then click on **OK**.



The Java editor appears for the implementation of the **GenerateAcceptance** component.

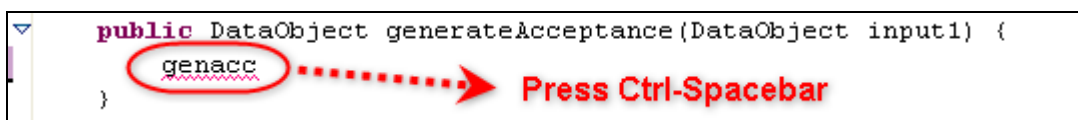
\_\_62. Scroll down to the bottom of the Java code. Select the highlighted text below and **delete**.



**Troubleshooting**

If you don't find the code displayed above, then the steps which should have added the GenerateAcceptance interface might have been skipped.

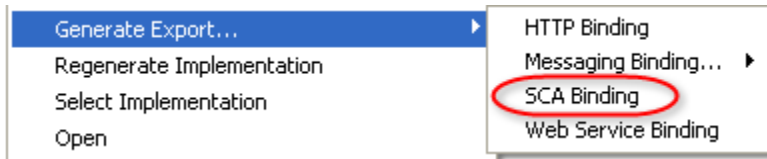
\_\_63. Type **genacc** in the line of code as shown below. Press **Ctrl+spacebar**.



The **Ctrl+spacebar** key combination activates a special feature called Code-Assist. This will add the required code snippet for you.



\_\_66. From the popup menu, select **Generate Export -> SCA Binding**.



**Hint**  
 An SCA Binding will allow other modules, specifically the main BPEL module, to access the Generate Acceptance service component.

An Export component will be generated.

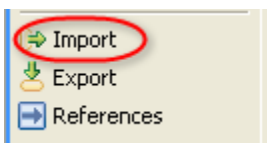


\_\_67. Press **Ctrl+s** to save the changes.

\_\_68. Switch to the Assembly Diagram for the main module.



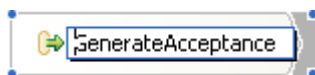
\_\_69. From the Palette, select the **Import** component.



\_\_70. Click on an empty space in the canvas to drop the Import component.



\_\_71. Single-click on the Import component to edit the name. Change the name to **GenerateAcceptance**.





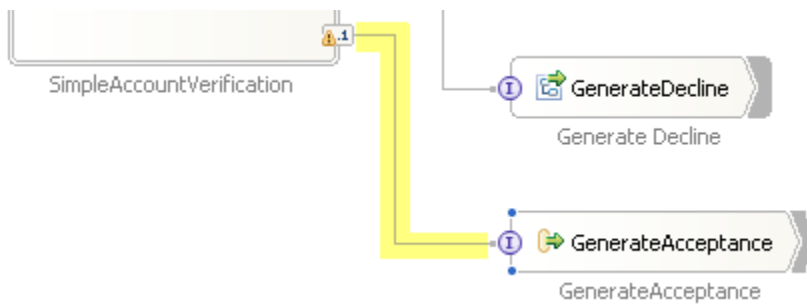
\_\_72. From the palette, click on the **Wire** button.



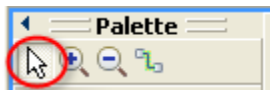
\_\_73. Click on the new **GenerateAcceptance** reference point on the **SimpleAccountVerification** process component. This will start a connection or wiring.



\_\_74. Click on the **GenerateAcceptance** import component. This will complete the new connection.



\_\_75. From the palette, click on the **Selection Tool** button.



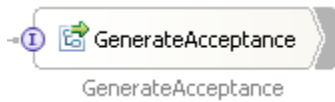
\_\_76. Right-click on the **GenerateAcceptance** component.



\_\_77. From the popup menu, select **Generate Binding -> SCA Binding**.

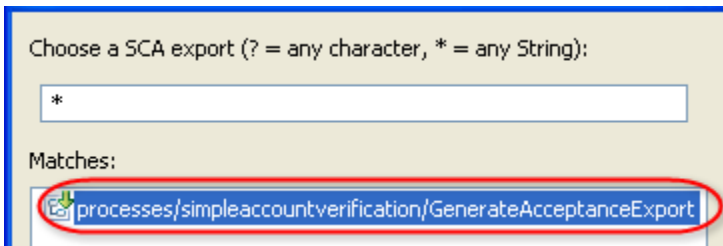


\_\_78. Right-click again on the **GenerateAcceptance** component.




\_\_79. From the popup menu, select **Select Service to Import**.

\_\_80. Select **GenerateAcceptanceExport** from the list. Click on **OK**.



\_\_81. Press **Ctrl+s** to save the assembly diagram.

The warning symbol () on the reference point should now disappear.



**What just happened?**

The implementation of the GenerateAcceptance service was created in the implementation module. You then imported the GenerateAcceptance service into the main module, and linked it to the Generate Acceptance activity in the BPEL process.



**Please continue to the next lab.**

## Lab 10 Incorporate Human Tasks

### Goals:

- **Integrate business processes end-to-end**
- **Include human tasks with automated business processes**
  - **Assign the right work to the right people at the right time**
  - **Human task must be repeatable, reusable, auditable**

### Role: Integration Developer

Business processes often involve human intervention, such as a loan officer reviewing loan applications, or a manager approving employee reimbursements. There will always be situations where systems or programs cannot automatically make decisions based on the available information. A business process might also involve a trivial task such as a secretary manually sending fax requests to print promotion materials. Because the goal is to implement the entire process end-to-end, then manual tasks need to be included with the automated tasks. With IBM's SOA Foundation and the WebSphere Business Process Management software portfolio, human activities are fully supported in business processes. In fact, the human task capabilities in the WebSphere Process Server are also designed for loose-coupling, effective reuse, and the building-block approach.

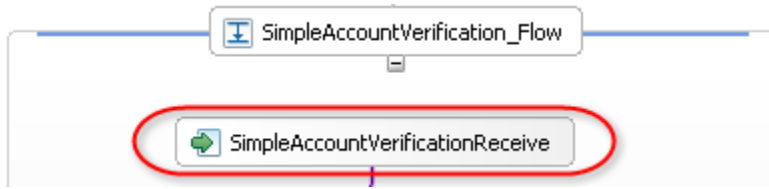
In this section, the implementation type for the Pricing and Approval task will be converted from Java to a human task. It has been decided that this final approval task now requires manual intervention. You will also allow the BPEL process to be invoked by a human task.

### 10.1 Create a Human Task to start the process

- \_\_1. Double-click on the **SimpleAccountVerification** process.

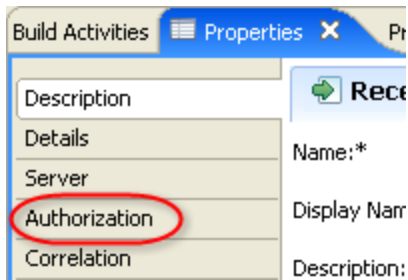


- \_\_2. Right-click on the **SimpleAccountVerificationReceive** activity.

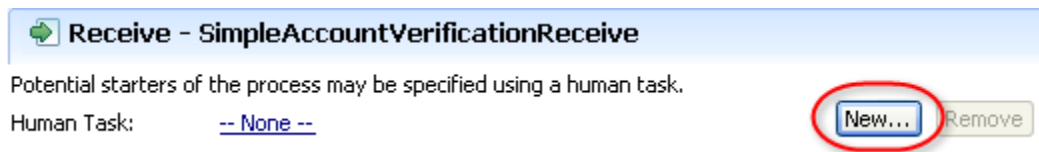


\_\_3. From the popup menu, select **Show in Properties**.

\_\_4. From the Properties view, click on the **Authorization** tab.

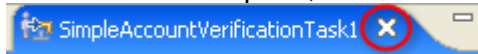


\_\_5. Click on **New**.




The Human Task editor appears.

\_\_6. Close the **Human Task** editor. At this point, we do not need to make any changes to the Human Task configuration.



**What just happened?**

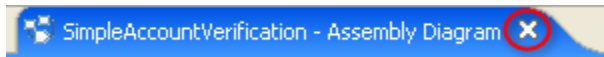
 A Human Task called SimpleAccountVerificationTask1 was defined for the Receive component in the BPEL process. This will allow the BPEL process to be invoked by a Human Task.

\_\_7. From the SimpleAccountVerification BPEL editor, press **Ctrl+s** to save your work.

\_\_8. Close the **SimpleAccountVerification** BPEL editor.

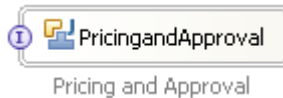


\_\_9. Close the Assembly Diagram editor for the main module with the BPEL process.

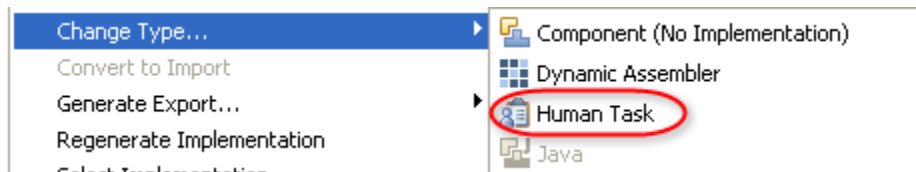


## 10.2 Convert the Pricing and Approval Java component to a Human Task

\_\_10. From the **SimpleAccountVerification\_impl** assembly diagram editor, right-click on the **PricingandApproval** component on the right-side.



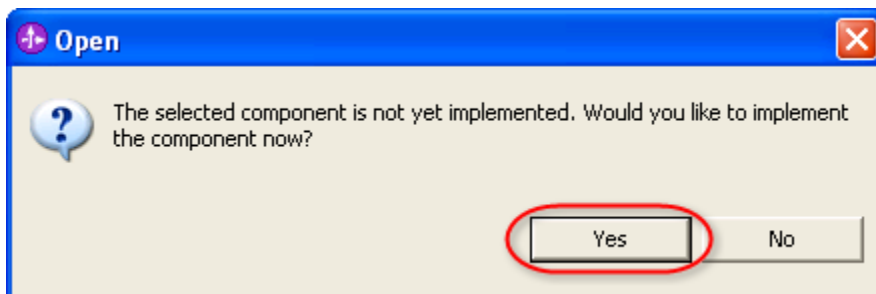
\_\_11. From the popup menu, select **Change Type -> Human Task**.



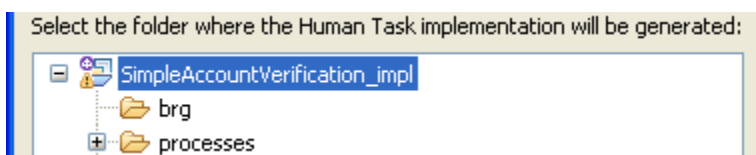
\_\_12. Double-click on the **PricingandApproval** component, which is now a Human Task.



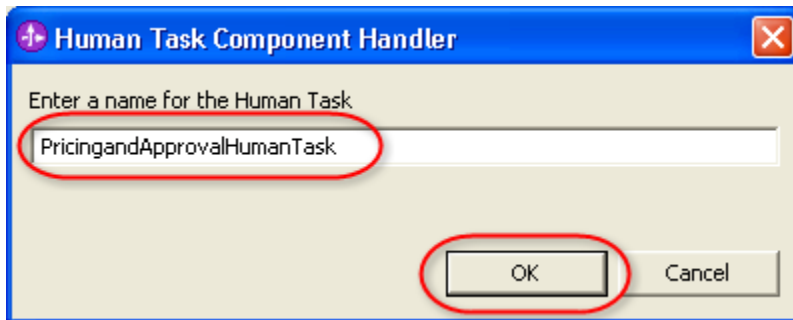
\_\_13. From the **Open** confirmation window, click on **Yes**.



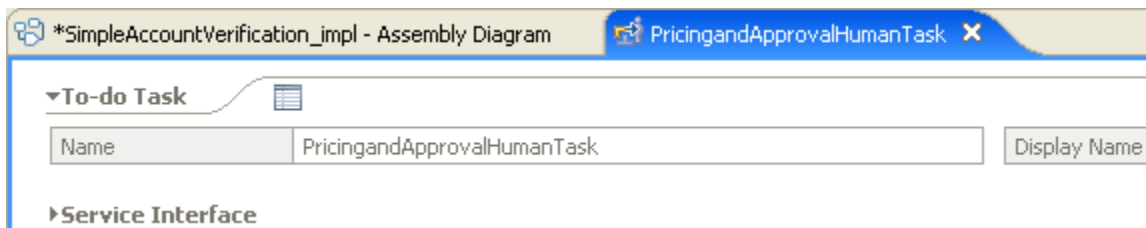
\_\_14. From the Generate Implementation window, ensure that **SimpleAccountVerification\_impl** is selected. Click on **OK**.



- \_\_15. From the Human Task Component Handler window, change the name to **PricingandApprovalHumanTask**. Click on **OK**.




The Human Task editor will appear.



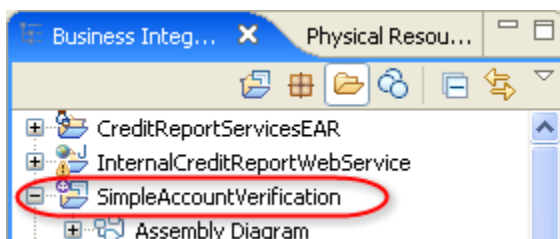
- \_\_16. Close the Human Task editor. At this point, we do not need to make any changes to the Human Task configuration.
- \_\_17. From the Assembly Diagram editor, press **Ctrl+s** to save your work.

**What's next?**

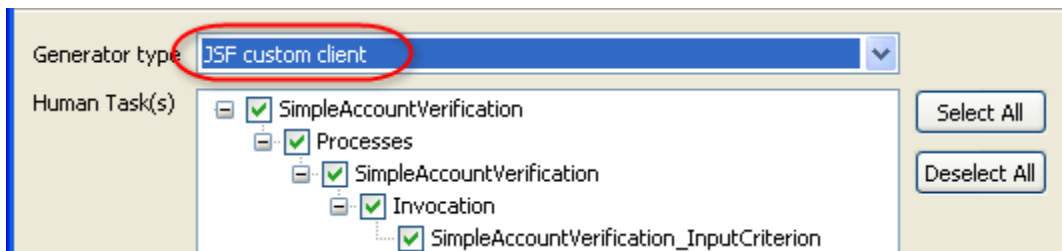
 You will now create the user interfaces or clients for the two human tasks you just defined. One web client for the human task to start the process, and another web client for the Pricing and Approval human task.

### 10.3 Generate a web client for the Start Process Human Task


- \_\_18. From the Business Integration tree view, right-click on **SimpleAccountVerification**.



- \_\_19. From the popup menu, select **Generate User Interfaces**.
- \_\_20. From the User Interface Wizard for Human Tasks window, select **JSF custom client** as the client generator type.



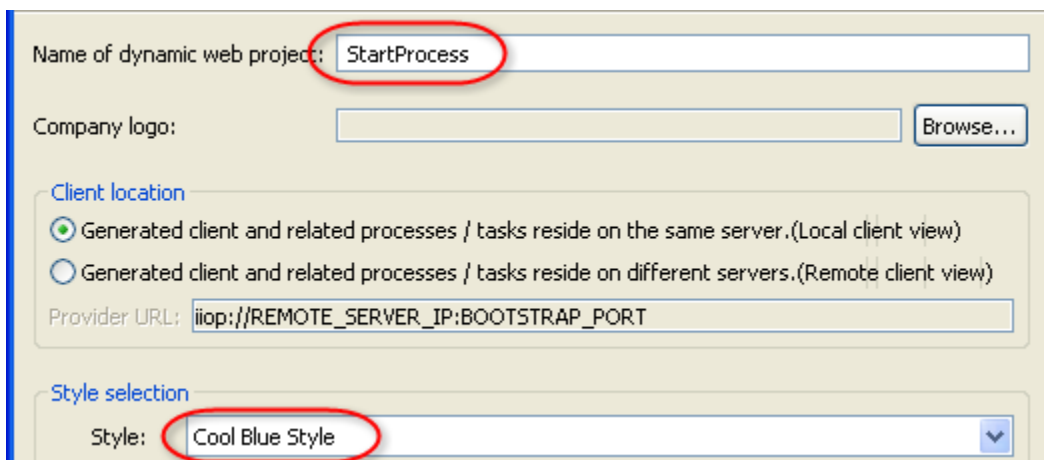
**Hint**



If you noticed, there are also other client generator types available. You can also choose to generate a Lotus® Forms client or a portlet. For this lab, we will use a Java Server Faces (JSF) client for the human tasks.

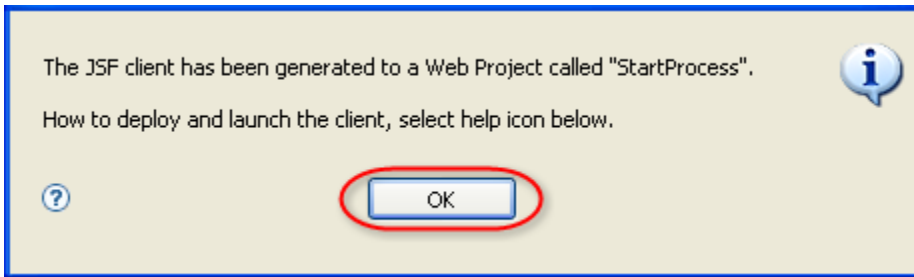
- \_\_21. Click on **Next**.
- \_\_22. On the next window, specify the following.

Name of dynamic web project: **StartProcess**  
 Style selection: **Cool Blue Style**




- \_\_23. Click on **Finish**. Wait for the client generation operation to complete.

\_\_24. From the **Client generation completed** information window, click on **OK**.



**What just happened?**



A standard web page was generated to allow a person to start the SimpleAccountVerification process. The web page and all its related artifacts will be packaged into its own J2EE project so that it can be deployed as a separate web application.

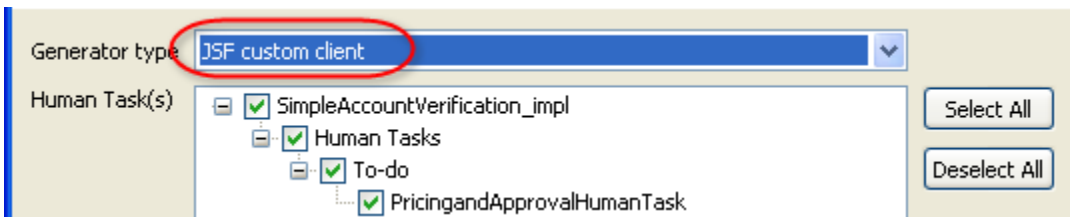
## 10.4 Generate a web client for the Pricing and Approval Human Task

\_\_25. From the Business Integration tree view, right-click on **SimpleAccountVerification\_impl**.



\_\_26. From the popup menu, select **Generate User Interfaces**.

\_\_27. From the User Interface Wizard for Human Tasks window, select **JSF custom client** as the client generator type.



\_\_28. Click on **Next**.

\_\_29. On the next window, specify the following.


Name of dynamic web project: **PricingandApproval**  
 Style selection: **Cool Blue Style**

\_\_30. Click on **Finish**. Wait for the client generation operation to complete.



\_\_31. From the **Client generation completed** information window, click on **OK**.

**What just happened?**



A standard web page was generated to allow a person to perform the Pricing and Approval task. This web client will allow the person to claim and then complete the related work item. When completed, control will transfer back to the BPEL process.


## 10.5 Retest the SimpleAccountVerification process

\_\_32. Switch to the **Servers** view.

\_\_33. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Add and Remove projects**.

\_\_34. From the **Add and Remove Projects** window, click on **Add All**.

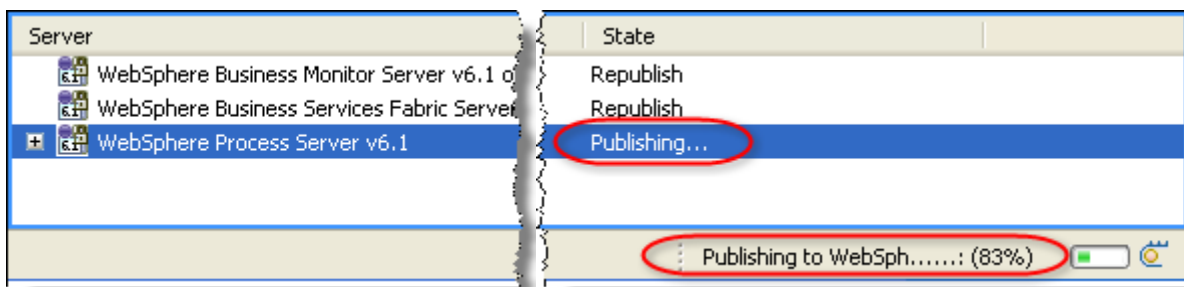
**Hint**



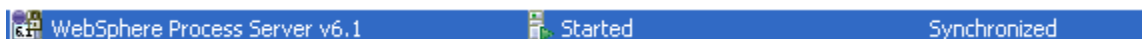
Take note of the two new projects: PricingandApprovalEAR and StartProcessEAR. These contain the generated Human Task web interfaces.

\_\_35. Click on **Finish**.


\_\_36. The server state will change to **Publishing**. You will also see a progress indicator at the bottom.



\_\_37. Wait for the progress indicator at the bottom to disappear. From the Servers view, check that the server state has changed to **Synchronized**.



**What's next?**



Its time to test the two new human tasks and the added Generate Acceptance activity. In the previous labs, you used the Business Process Choreographer explorer to start and test the process. This time a human task will be used to start the process. You will also use a human task to perform the Pricing and Approval activity instead of code.

## 10.6 Use the Human Task client to start the process

\_\_38. Click on a web browser icon in the **Quick Launch** bar.

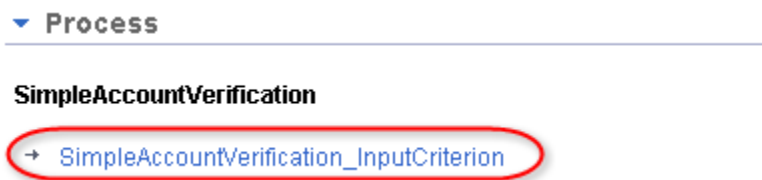


\_\_39. From the web browser, open the URL <http://localhost:9080/StartProcess>.

\_\_40. From the Business User Client web page, click on **New**.



\_\_41. Click on **SimpleAccountVerification\_InputCriterion**.



\_\_42. For the **customerID** field, type **111**, and then click on **Create**.

customerID:

### Hint



If we specified a customer ID of 123, the process flow would follow the 'denied' execution path. However, you need to specify any customer ID other than 123 so that the request will be approved, and the flow will go through the 'approved' path which includes the modified Pricing And Approval human task and the new Generate Acceptance activity.

\_\_43. Switch to the **WebSphere Integration Developer**. Switch to the **Console** view.

\_\_44. Verify that a credit score of 501 is displayed, and that no decision has been made yet on whether the request is approved or denied.

```

O *****
O ** Call Internal Credit Report Service **
O *****
O >>>>> Credit Score = 501
O *****
  
```

**What's next?**



When the Credit Score is displayed in the console, the execution of the process has paused. This indicates that the process has reached the Pricing and Approval task and is waiting for a response from a 'human'. In the next steps, you will assume the role of a Loan Officer or Credit Manager, and perform the Pricing and Approval task.

While the process is in a suspended state waiting for the human task to complete, you will also use some of the features of the WebSphere Integration Developer to analyze the execution flow and the process state.

## 10.7 View the Process State

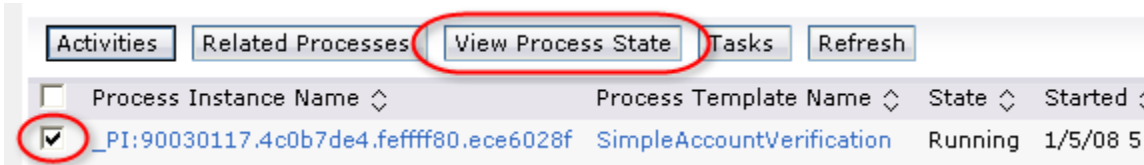
\_\_45. Switch to the **Servers** view.

\_\_46. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Launch -> Business Process Choreographer Explorer**.

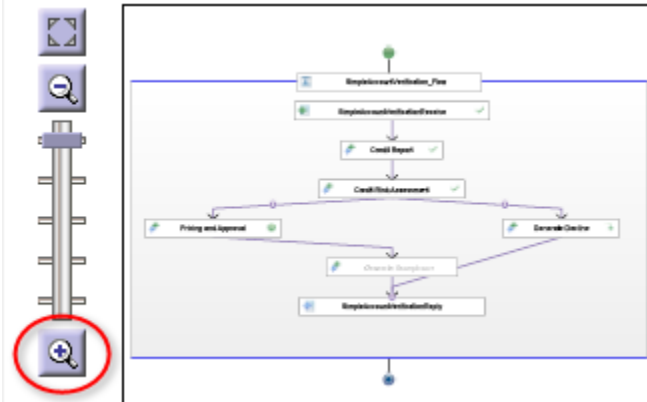
\_\_47. Click on the **Started By Me** link.



\_\_48. Select the **SimpleAccountVerification** process template name. Click on **View Process State**.

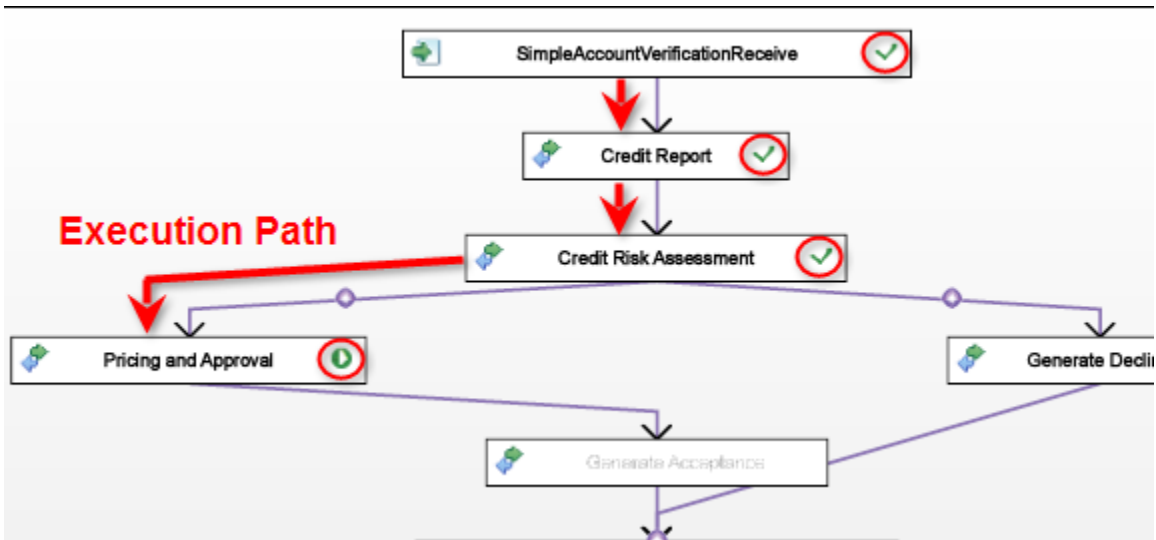


\_\_49. Click on the zoom **in** button two times.



\_\_50. Hold down the **Alt** key and then use the cursor to grab and move the diagram to center it.

\_\_51. Notice the icons representing the state of each activity (🟢, ✓, ⬇️). This will indicate the execution path.




**Hint**


The icon 🟢 indicates that the activity is currently running. Specifically it shows that the Pricing and Approval activity waiting for the human task to complete.

The icon ✓ indicates that the activity was invoked and finished.

The icon ⬇️ indicates that the activity was skipped.



For a complete list, click on the  icon.

Use this page to view a graphical image of the process instance .  
Hold down the ALT key and drag the mouse to move the.

**Hint**

The activity state is also displayed on the view when the cursor is placed on the activity.

Selected Activity:	Credit Report
State:	Finished
Activated:	1/5/08 5:21 PM

**Hint**

You can right-click on the Scalable Vector Graphics (SVG) diagram and select **Copy SVG**. You can then do a **Paste Special** in WordPad or other editor to copy and paste the process state diagram.

## 10.8 Use the Human Task client to perform Pricing and Approval

- \_\_52. When you're finished viewing the process state diagram, switch to the **web browser** you used to start the process.
- \_\_53. Open the URL <http://localhost:9080/PricingandApproval>.
- \_\_54. Click on the **Open** link.



IBM Business User Client

My ToDo's

- HOME
- My ToDo's
- **Open**
- Claimed

Business User Client

My ToDo's

- Open

- \_\_55. Click on the **PricingandApproval** task.

Task Name	Description	First Activated	On
→ <b>PricingandApproval</b>		1/5/08 5:21:40 PM	UP

Items found: 1 Page 1 of 1 Items per page: 20

**Hint**

You were not required to log in with a user ID and password because security is currently disabled. This will typically not be the case in production environments. Based on the security profile or credentials of the user, the list should only display tasks which have been assigned to the user.



## Lab 11 Add New Service Providers

### Goals:

- Ensure success through governance
- Promote standards and reuse
  - Load WSDLs of approved external Credit Report services to the WebSphere Service Registry and Repository

### Role: Architect or Project Manager

Aside from the Internal Credit Report service, a decision has been made to also start using external Credit Report services, such as those provided by Equinox, Experian, and Trans Union. These credit report agencies have supplied the WSDL files needed to access their external web services, and will now be published to the WebSphere Service Registry and Repository. The assumption is that these external Credit Report services will be reviewed and approved for use within the organization.

### 11.1 Publish the WSDLs for the external Credit Report web services

- \_\_1. Switch to the web browser showing the WebSphere Registry and Repository console.



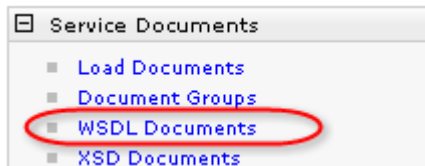
#### Troubleshooting

If the WebSphere Service Registry and Repository console was closed, just start a web browser and go to <http://localhost:9080/ServiceRegistry>.

- \_\_2. If needed, select **GP Architect** from the Perspective drop-down list. Click on the **Go** button.



\_\_3. Expand the **Service Documents** section. Click on **WSDL Documents**.



You should see the two WSDL files loaded earlier.

Select	Name	Graph	Description	Namespace
<input type="checkbox"/>	CreditReportInterface.wsdl			http://SimpleAccountVerification/
<input type="checkbox"/>	InternalCreditReportWebService.wsdl			http://Processes/SimpleAccountV

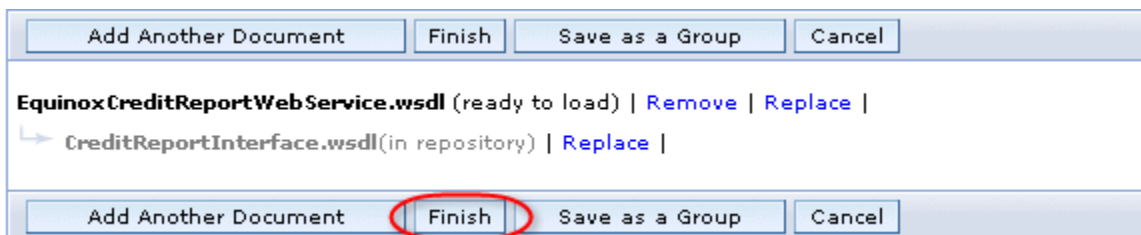
\_\_4. Click on **Load Documents**.

\_\_5. Click on **Browse** to select **C:\PoT\BPMSOA\EquinoxCreditReportWebService.wsdl**.



\_\_6. Click on **OK**.

\_\_7. Click on **Finish**.



A message appears that the document was loaded successfully.

**Documents Loaded Successfully**

The following documents have been loaded into the repository:

Name	Description	Namespace
EquinoxCreditReportWebService.wsdl		http://Processes/SimpleAccountVerification/Credit



**What just happened?**

The WSDL file from the Equinox credit report company was loaded in the WebSphere Service Registry and Repository. This WSDL will be reviewed and approved for use within the company. This web service will serve as one of the service endpoints for the Credit Report task in the SimpleAccountVerification process.





Please continue to the next lab.

## Lab 12 Provide Flexibility with an Enterprise Service Bus

### Goals:

- Provide a layer between services for loose-coupling and flexibility
- Use an ESB between service consumers and providers
  - Between BPEL process and Credit Report web services


### Role: Integration Developer

The WebSphere Enterprise Service Bus provides flexibility at the service connectivity tier. Currently, the SimpleAccountVerification process invokes a specific and fixed web service for the Credit Report activity. However, the process now needs to invoke several Credit Report web services depending on specific criteria, such as the requested credit limit. The use of mediation modules will allow the connectivity and routing logic to be extracted from the main process. This is loose-coupling taken to another level, or the connectivity level. It is another layer of abstraction where the main process does not need to know which service endpoint will be invoked to perform the requested task. The process simply needs to invoke a generic and abstract service, and the WebSphere Enterprise Service Bus can handle the routing, transformation, and mediation of the request to the actual service endpoints.

### 12.1 Import the Credit Report Web Services

\_\_1. Switch to the **WebSphere Integration Developer**.

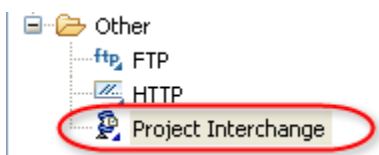
**What's next?**



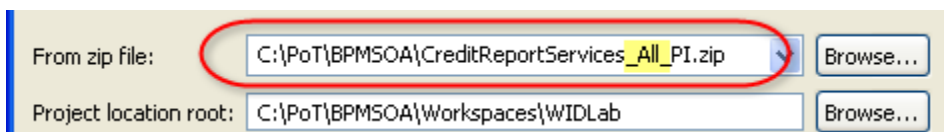
In a more realistic development environment, the external Credit Report web services will probably be accessed remotely and will not need to be imported into this development machine. However for this lab, you will need to import the web services so that these can be deployed before it can be invoked by the BPEL process.

\_\_2. From the main menu, select **File -> Import**.

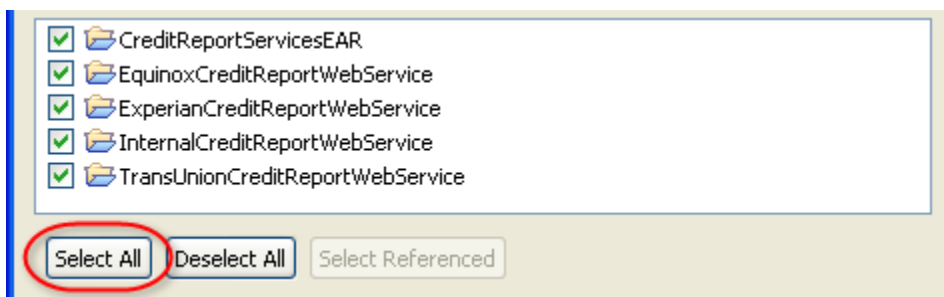
\_\_3. Expand the **Other** category. Select **Project Interchange** as the import source. Click on **Next**.



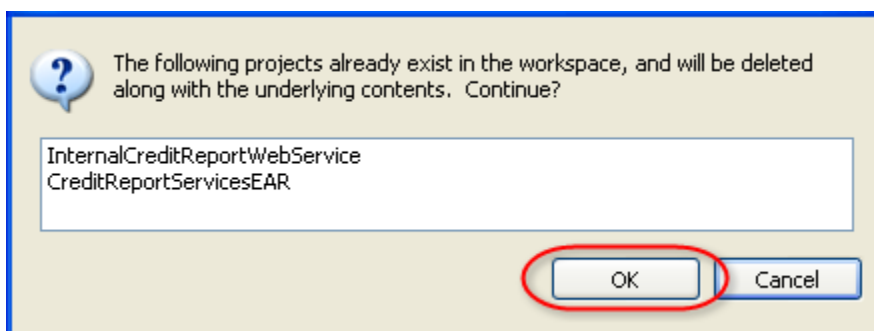
- \_\_4. Use the **Browse** button for the **From zip file** field to select **C:\PoT\BPMSOA\CreditReportServices\_All\_PI.zip**.




- \_\_5. Click on **Select All**. Click on **Finish**.



- \_\_6. From the confirmation window, click on **OK** to overwrite existing projects.



**What just happened?**



Several new external credit report web services were imported. The Equinox Credit Report web service is what will be used in this lab. The other web services will be used in later labs.

- \_\_7. Switch to the **Servers** view. Expand **WebSphere Process Server v6.1**. Right-click on **CreditReportServicesEAR**. From the popup menu, select **Restart CreditReportServicesEAR**.



**What's next?**

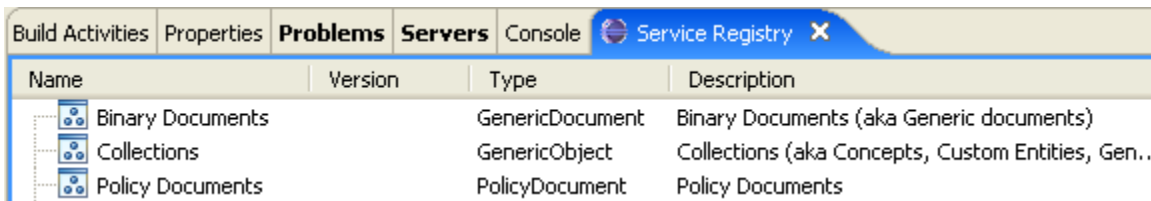


Instead of directly invoking a fixed web service (Internal Credit Report), you will modify the process to invoke a variable Credit Report web service using an Enterprise Service Bus. Specifically you will create a Mediation module to dynamically invoke either the Internal Credit Report or the external Equinox Credit Report web service depending on the following:

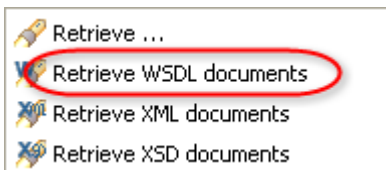
- Internal Credit Report - requested credit limit is less than 5000
- Equinox Credit Report - requested credit limit is 5000 or greater

## 12.2 Import WSDLs from WebSphere Service Registry and Repository

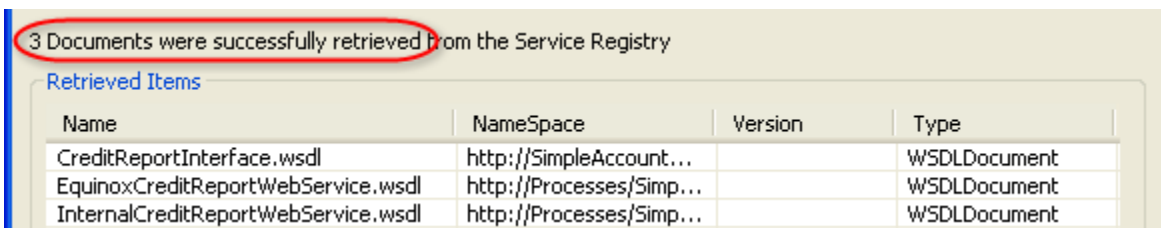
\_\_8. Switch to the **Service Registry** view.



\_\_9. Right-click anywhere inside the Service Registry view. From the popup menu, select **Retrieve WSDL documents**.



A window appears indicating that the documents (WSDL files) were successfully retrieved.

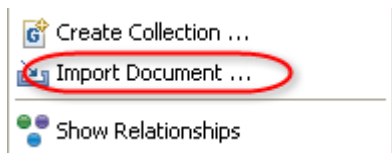


\_\_10. Click on **OK**.

\_\_11. Expand **WSDL Documents**. Right-click on **EquinoxCreditReportWebService.wsdl**.



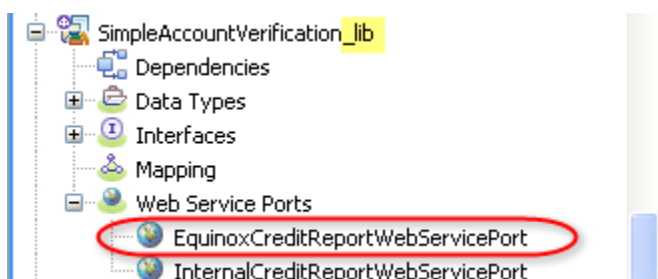
\_\_12. From the popup menu, select **Import Document**.



\_\_13. Select **SimpleAccountVerification\_lib**. Click on **Finish**.



\_\_14. From the Business Integration view, verify that there are now two web services under the Web Service Ports of the SimpleAccountVerification\_lib module.

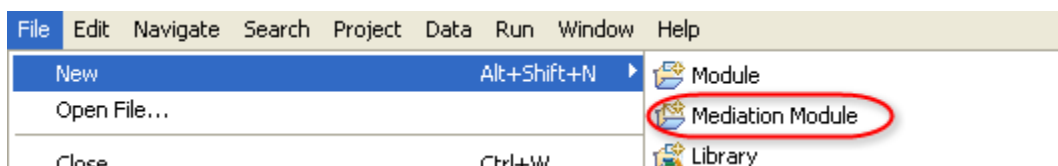


**What just happened?**

You imported the WSDLs of the three external Credit Report web services from the WebSphere Service Registry and Repository.

### 12.3 Create a Mediation Module

\_\_15. From the main menu, select **File -> New -> Mediation Module**.



- \_\_16. Specify **CreditReportServiceRouter** for the Module Name. For the Target Runtime, select **WebSphere Process Server v6.1**.

Module name:

Use default location

Location:

Target runtime:

Create mediation component

Name:

- \_\_17. Click on **Next**.

- \_\_18. Select **SimpleAccountVerification\_lib**. Click on **Finish**.

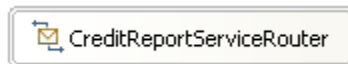
**Select required libraries**

Select libraries containing re-usable resources such as interfaces, to be used by this module.

Libraries:

- SimpleAccountVerification\_lib

An Assembly Diagram editor appears containing a Mediation Flow component called CreditReportServiceRouter.



**Hint**



Mediation is a way of mediating or intervening dynamically between services. A mediation flow implements a mediation. Mediation flows intercept and modify messages that are passed between existing services (providers) and clients (requesters) that want to use those services. A mediation flow provides functions such as message logging, data transformation and routing.

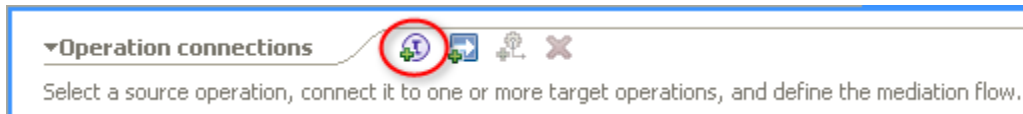
## 12.4 Implement the Mediation Flow

- \_\_19. Double-click on the CreditReportServiceRouter component.

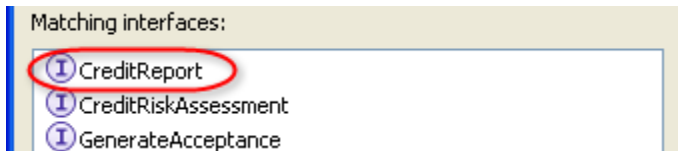


A Mediation Flow Editor will appear.

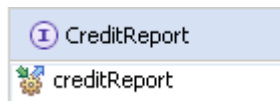
\_\_20. Click on the **Add Interface** button.



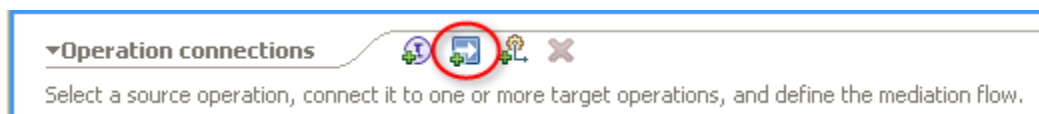
\_\_21. Select **CreditReport** from the list. Click on **OK**.



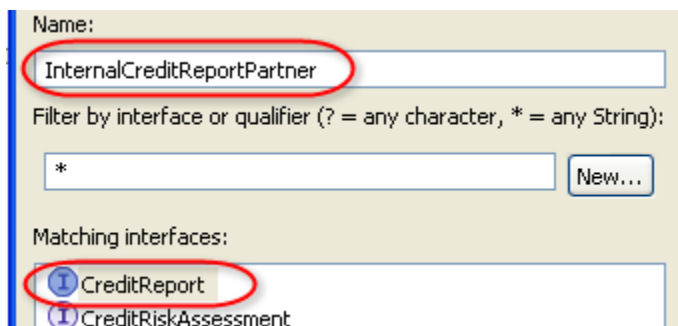
A CreditReport Interface component will appear in the Flow editor.



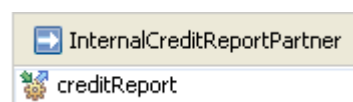
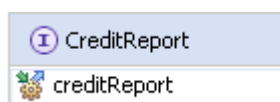
\_\_22. Click on the **Add Reference** button.



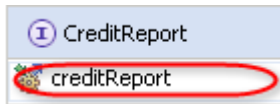
\_\_23. Change the Name to **InternalCreditReportPartner**. Select **CreditReport** from the list. Click on **OK**.



You should now see the CreditReport Interface and InternalCreditReportPartner Reference components.



\_\_24. Right-click on the **creditReport** operation of the CreditReport Interface on the left.

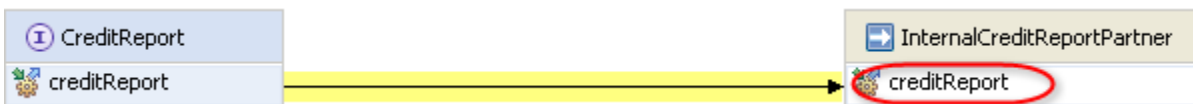


\_\_25. From the popup menu, select **Create Operation Connection**.

This will start a connection.



\_\_26. Click on the **creditReport** operation of InternalCreditReportPartner to complete the connection.

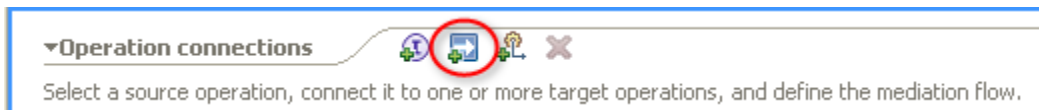


**What just happened?**

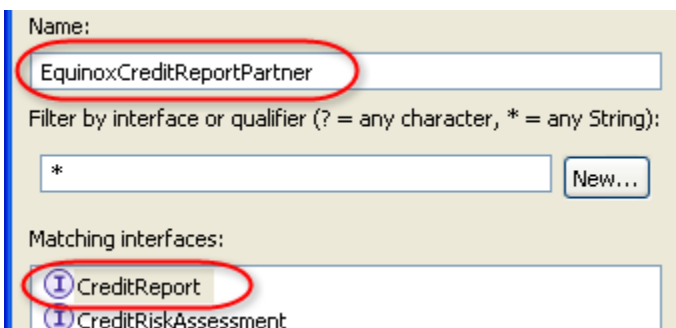


You created a connection to indicate that when the creditReport operation of the CreditReportServiceRouter is invoked, the request will be passed to the creditReport operation of the partner or service provider. Later you will define the Internal Credit Report web service as the service provider or endpoint.

\_\_27. Click on the **Add Reference** button again to add a second reference partner.

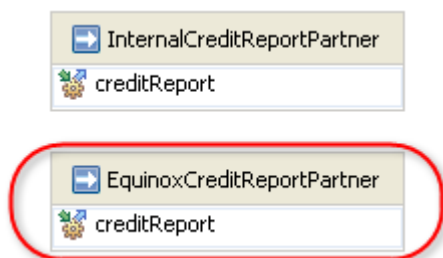


\_\_28. Change the Name to **EquinoxCreditReportPartner**. Select **CreditReport** from the list. Click on **OK**.

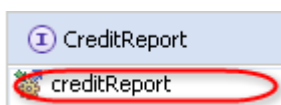




A second reference partner will appear below the InternalCreditReportPartner reference.

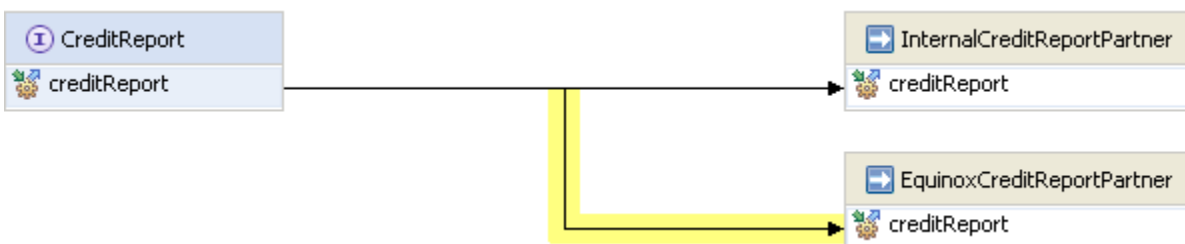


\_\_29. Right-click on the **creditReport** operation of the CreditReport Interface on the left.




\_\_30. From the popup menu, select **Create Operation Connection**.

\_\_31. Click on the **creditReport** operation of EquinoxCreditReportPartner to complete the connection.

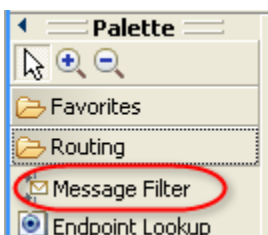


**What just happened?**




With a second connection, you are now indicating that when the creditReport operation of the CreditReportServiceRouter is invoked, the request will be passed to either the Internal Credit Report web service or the Equinox Credit Report web service. You will set up this routing logic in the next steps.

\_\_32. From the Palette, expand the **Routing** folder. Select the **Message Filter** primitive.



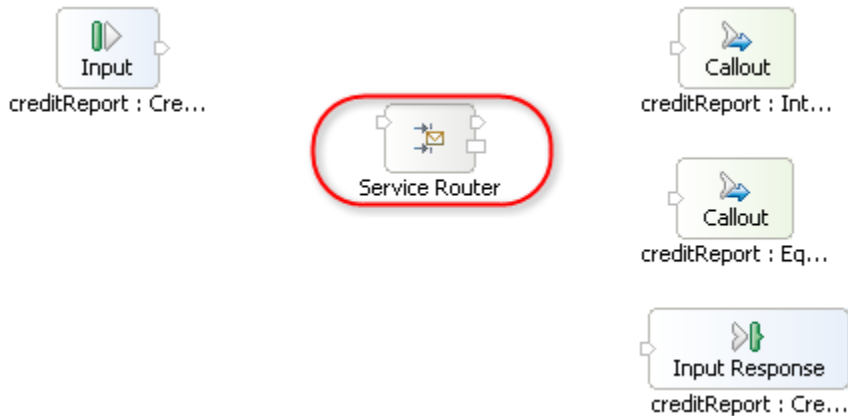
**Hint**

The lower area of the Mediation Flow Editor with the Palette is the Request/Response flow editor. The tabs at the bottom of the flow editor allows you to switch between the Request and Response editors.



You are currently working with the Request flow editor.

\_\_33. Click to drop the **Message Filter** into the Request flow editor in the area indicated by the following screenshot. Rename **MessageFilter1** to **Service Router**.



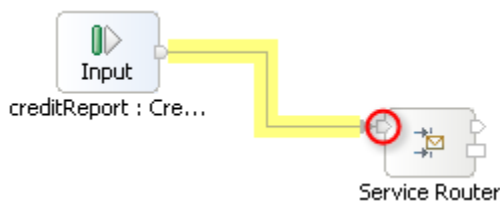
\_\_34. Move the mouse on top of the **out** terminal for the **Input** node.



This will expose the yellow connector.

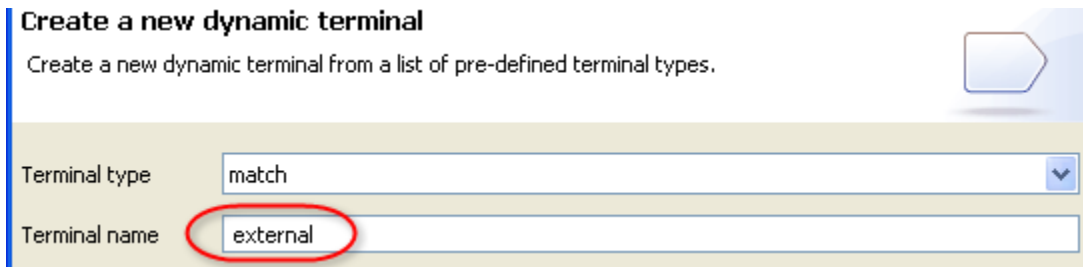


\_\_35. Connect the yellow connector to the **in** terminal of **ServiceRouter**.



\_\_36. Right-click on **ServiceRouter**. From the popup menu, select **Add Output Terminal**.

\_\_37. From the New Dynamic Terminal window, specify **external** as the **Terminal name**.



**Create a new dynamic terminal**  
Create a new dynamic terminal from a list of pre-defined terminal types.

Terminal type: match

Terminal name: external

\_\_38. Click on **OK**.

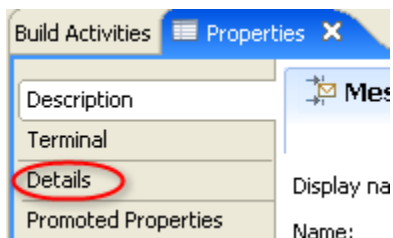
This will add a second terminal to **ServiceRouter**.



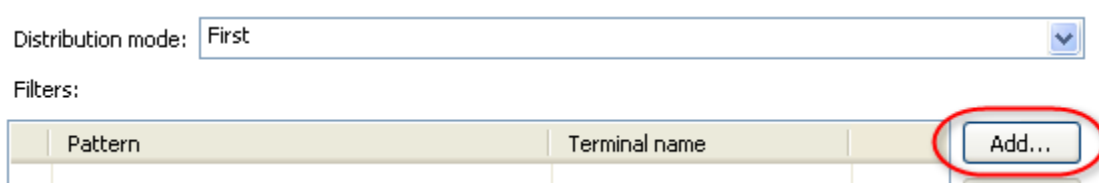
## 12.5 Define the Routing Logic to the external Equinox Credit Report Service

\_\_39. Right-click **ServiceRouter**. From the popup menu, select **Show in Properties**.

\_\_40. Click on the **Details** tab.



\_\_41. Click on **Add**.

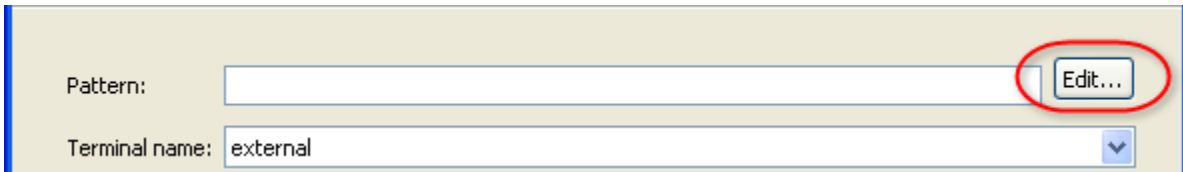


Distribution mode: First

Filters:

Pattern	Terminal name	Add...

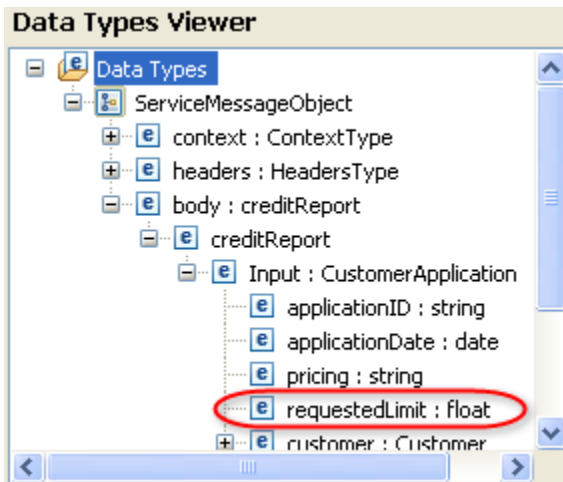
\_\_42. From the **Add/Edit** window, click on **Edit**.



**What's next?**

You will define the criteria to determine when the request will be routed to the **external** terminal. The external terminal will then be linked to the Equinox Credit Report service.

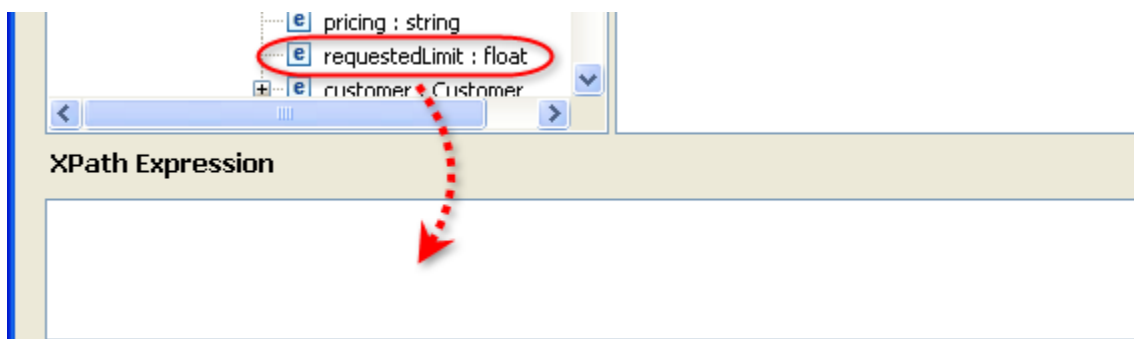
\_\_43. From the XPath Expression Builder window, expand Data Types to select the **requestedLimit** attribute.



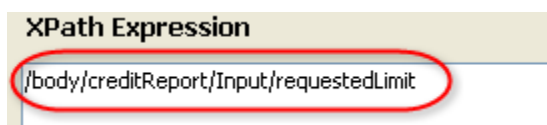
**Hint**

These are the attributes of the CustomerApplication business object which will be passed into this CreditReportServiceRouter Mediation Module.

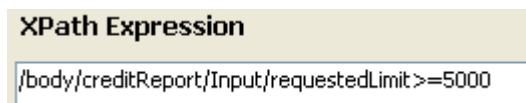
\_\_44. Drag and drop the **requestedLimit** attribute to the XPath Expression source viewer.



The XPath Expression viewer should now contain the expression **/body/creditReport/Input/requestedLimit**.



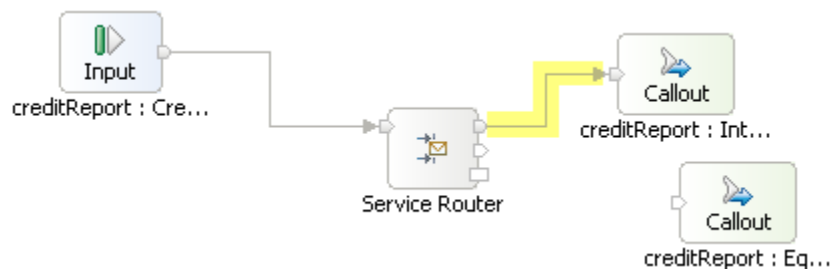
\_\_45. Add the text “>=5000” to the end of the XPath expression. The full expression should be “/body/creditReport/Input/requestedLimit>=5000”.



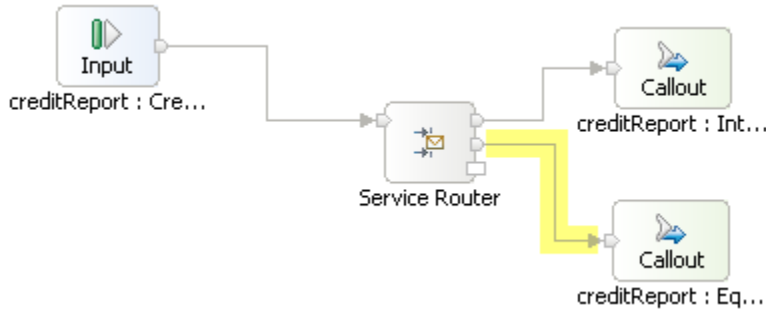
\_\_46. Click on **Finish**.

\_\_47. From the **Add/Edit** window, click on **Finish**.

\_\_48. From the Request Flow editor, connect the ServiceRouter **out** terminal (upper) to the topmost **Callout** node.



\_\_49. Connect the **ServiceRouter premium** terminal (middle) to the second **InputCriterion : CreditReportPartner1 Callout** node.



**What just happened?**



The routing logic was defined so that any request submitted with a requestedLimit of 5000 or more will be forwarded to the external Equinox Credit Report service. A value of less than 5000 will be routed to the Internal Credit Report service.

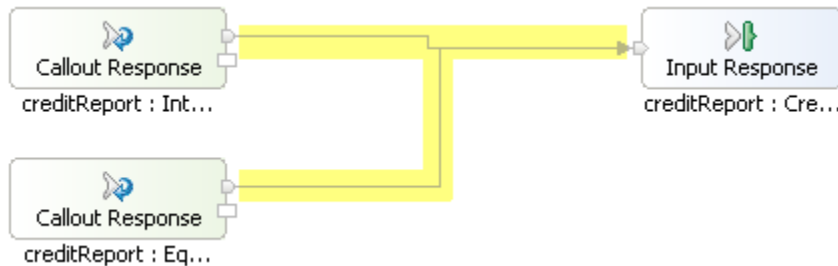
The Request Flow is now complete. The next step is to complete the logic for the Response Flow.

\_\_50. From the Flow editor, click on the **Response: creditReport** tab at the bottom.



This will switch to the Response Flow Editor.

\_\_51. Connect the **out** terminals of both **Callout Response** nodes to the **in** terminal of the **Input Response** node.



**Hint**

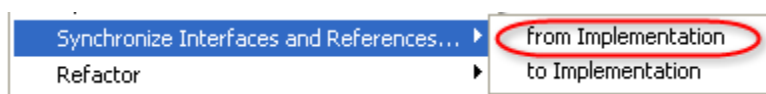
There is no special routing logic needed for the Response flow. Output will simply be passed back to the calling component.

- \_\_52. Press **Ctrl+s** to save the Mediation Flow.
- \_\_53. Close the Mediation Flow Editor.  
Focus should return to the Assembly Diagram editor for **CreditReportServiceRouter**.

## 12.6 Complete the CreditReportServiceRouter assembly diagram


- \_\_54. Right-click on **CreditReportServiceRouter**. 

- \_\_55. From the popup menu, select **Synchronize Interfaces and References -> from Implementation**.




- \_\_56. From the confirmation window, click **Yes**.

**What happened?**

 When you implemented the CreditReportServiceRouter using the Mediation Flow editor, you added two references, the InternalCreditReportPartner and the EquinoxCreditReportPartner. The CreditReportServiceRouter component in the assembly diagram did not reflect this yet, so it needed to be refreshed, or synchronized with its current implementation. After the component was synchronized, the red 'x' icon was cleared.

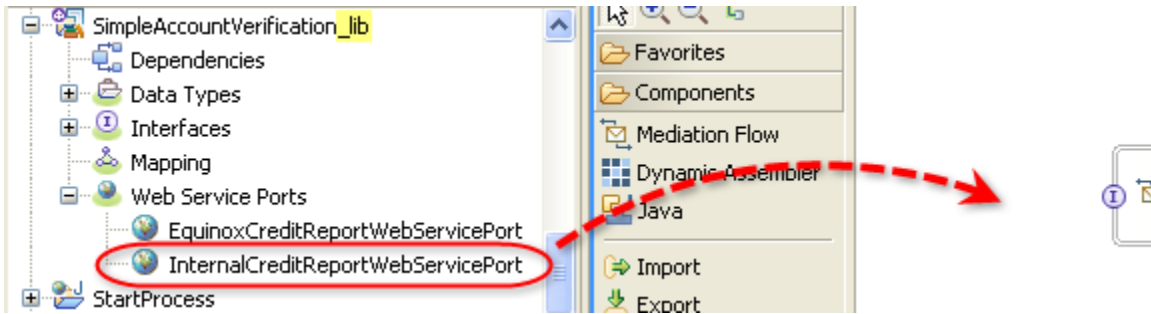
**What's next?**

 You will now link the references specified in the CreditReportServiceRouter mediation flow to its corresponding web services.

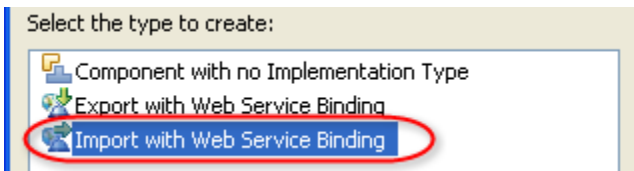
- \_\_57. Right-click on an empty space in the canvas. From the popup menu, select **Arrange Contents Automatically**.



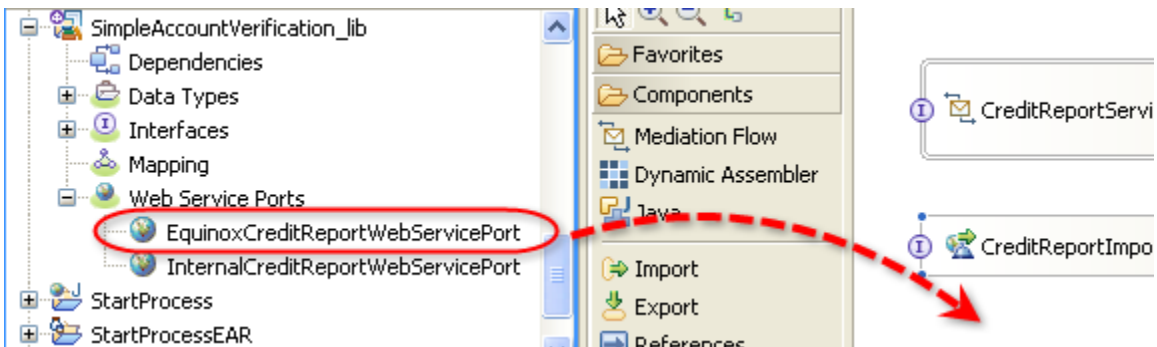
\_\_58. From the Business Integration view, drag and drop the **InternalCreditReportWebServicePort** to the assembly diagram.



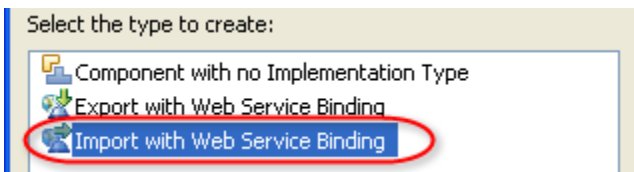
\_\_59. From the Component Creation window, select **Import with Web Service Binding**. Click on **OK**.



\_\_60. From the Business Integration view, drag and drop the **EquinoxCreditReportWebServicePort** to the assembly diagram.



\_\_61. From the Component Creation window, select **Import with Web Service Binding**. Click on **OK**.



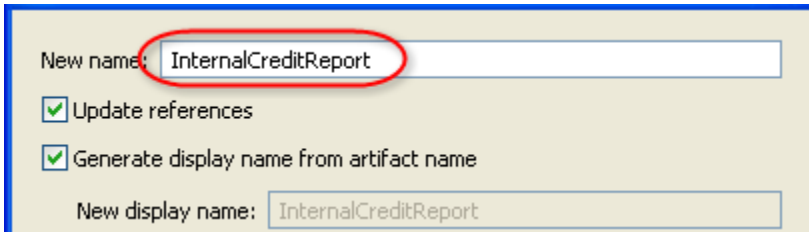
\_\_62. Press **Ctrl+s** to save your current progress.

\_\_63. Right-click on **CreditReportImport1**.





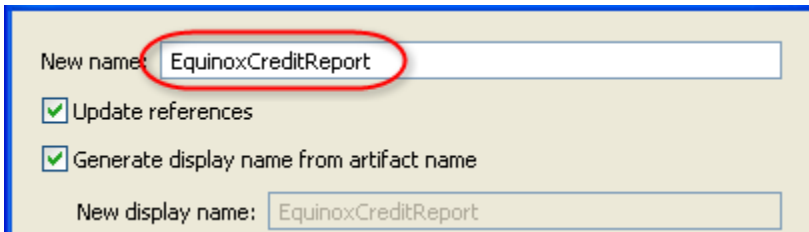
- \_\_64. From the popup menu, select **Refactor -> Rename**.
- \_\_65. Change the name to **InternalCreditReport**. Click on **OK**.



- \_\_66. Right-click on **CreditReportImport2**.



- \_\_67. From the popup menu, select **Refactor -> Rename**.
- \_\_68. Change the name to **EquinoxCreditReport**. Click on **OK**.



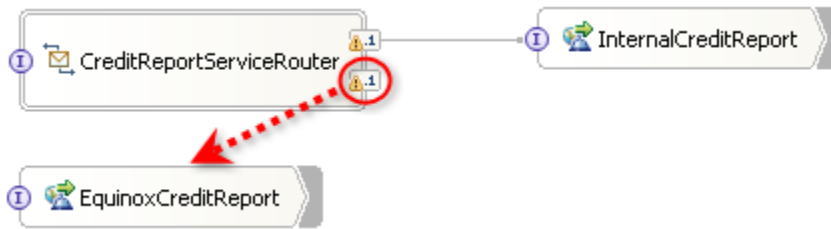
- \_\_69. From the Palette, click on the Wire icon.



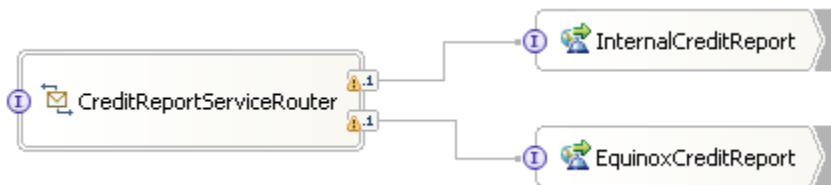
- \_\_70. Connect the upper reference point of the CreditReportServiceRouter to **InternalCreditReport**.



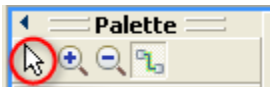
\_\_71. Connect the lower reference point of the CreditReportServiceRouter to **EquinoxCreditReport**.



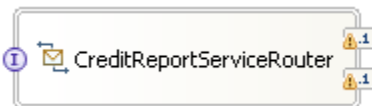
The layout will automatically be rearranged.



\_\_72. From the Palette, click on the **Selection Tool** icon.



\_\_73. Right-click on **CreditReportServiceRouter**.





\_\_74. From the popup menu, select **Generate Export -> SCA Binding**.

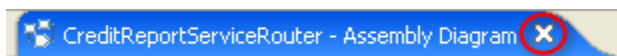


\_\_75. Press **Ctrl+s** to save the CreditReportServiceRouter Assembly Diagram.

**What happened?**

 All the components have now been assembled for the CreditReportServiceRouter. This mediation flow has been set up so that requests for a Credit Report will be forwarded to either the Internal or the Equinox Credit Report web service depending on the requested credit limit.

- \_\_76. Switch to the **Problems** view. Verify that no errors exist (messages with a red 'x' mark ). Warnings and information messages are expected.
- \_\_77. Close the Assembly Diagram editor for CreditReportServiceRouter.



Focus should return to the SimpleAccountVerification\_impl Assembly Diagram editor.

#### What's next?



Instead of only invoking the Internal Credit Report web service, you will now use the WebSphere Enterprise Service Bus. The static web service component will be replaced by the CreditReportServiceRouter mediation component so that requests can be dynamically routed to either the Internal Credit Report web service or the Equinox Credit Report web service.

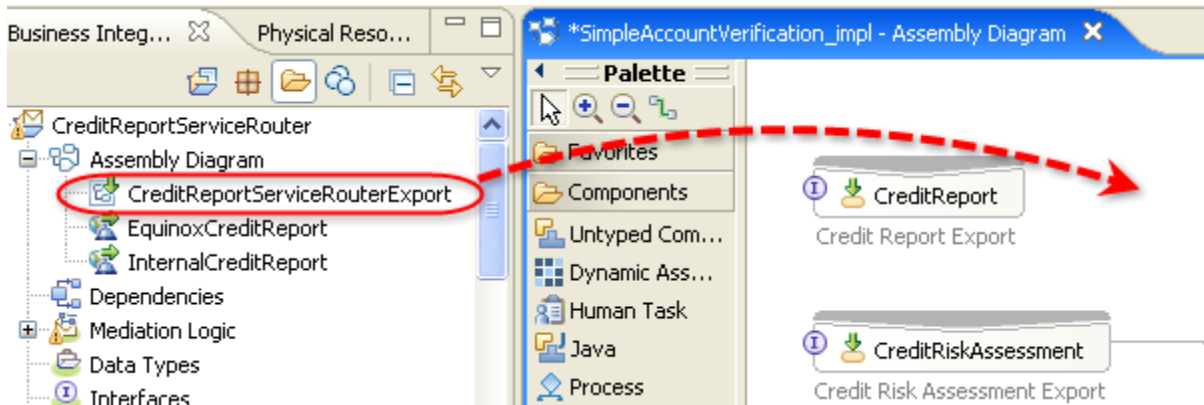
## 12.7 Re-assemble the SimpleAccountVerification to use the CreditReportServiceRouter

- \_\_78. From the **Assembly Diagram** editor, **delete** the **CreditReportImport1** component on the right side.

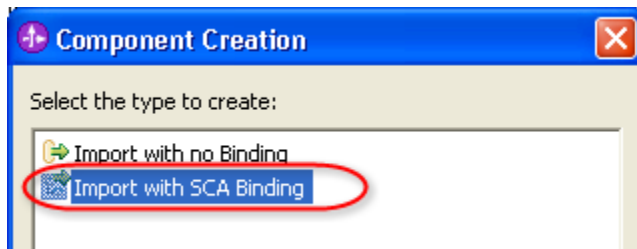


This will be replaced by the **CreditReportServiceRouter** mediation component.

- \_\_79. From the Business Integration view, expand **CreditReportServiceRouter** -> **Assembly Diagram**. Select **CreditReportServiceRouterExport**. Drag and drop to an empty space in the Assembly Diagram editor.



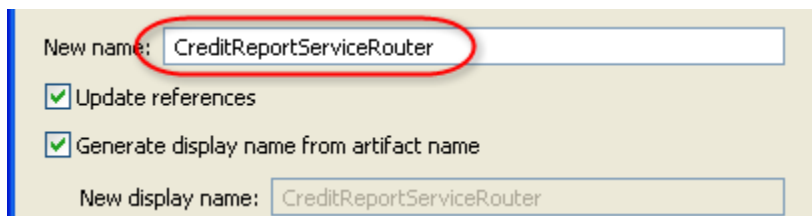
- \_\_80. From the **Component Creation** window, select **Import with SCA Binding**. Click on **OK**.



- \_\_81. Right-click on **Import1**. From the popup menu, select **Refactor** -> **Rename**.

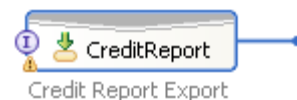
- \_\_82. Click on **OK** to save the assembly diagram before refactoring.

- \_\_83. Change the name of **Import1** to **CreditReportServiceRouter**. Click on **OK**.



- \_\_84. From the Palette, click on the **Wire** icon to switch to connection mode. 

- \_\_85. Click on **CreditReport** to start a connection.




\_\_86. Click on **CreditReportServiceRouter** to complete the connection.



\_\_87. From the Palette, click on the **Selection Tool** icon. 

\_\_88. Press **Ctrl+s** to save the changes.

\_\_89. Switch to the **Problems** view. Verify that no errors exist (messages with a red 'x' mark ). Warnings and information messages are expected.

## 12.8 Retest the SimpleAccountVerification Process

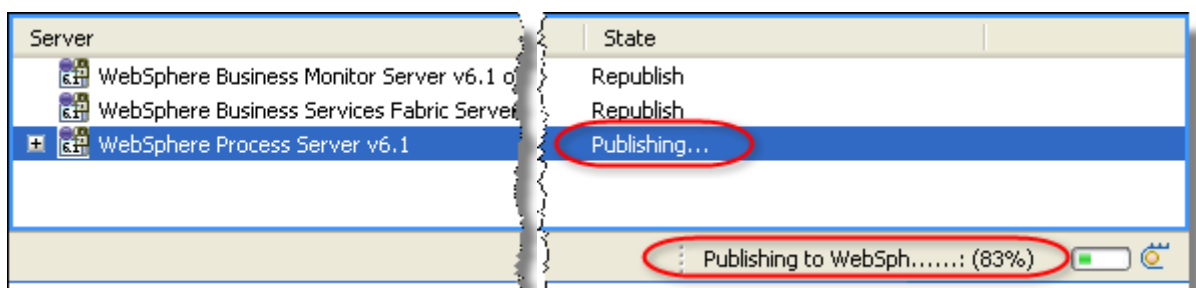
\_\_90. Switch to the **Servers** view.

\_\_91. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Add and Remove projects**.

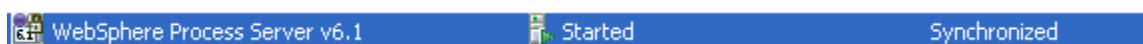
\_\_92. From the **Add and Remove Projects** window, click on **Add All**.

\_\_93. Click on **Finish**.

\_\_94. The server state will change to **Publishing**. You will also see a progress indicator at the bottom.



\_\_95. Wait for the progress indicator at the bottom to disappear. From the Servers view, check that the server state has changed to **Synchronized**.



### What's next?




You will now retest the SimpleAccountVerification process. This time, either the Internal Credit Report or the external Equinox Credit Report web service will be dynamically invoked depending on the following:

- Internal Credit Report - requested credit limit is less than 5000
- Equinox Credit Report - requested credit limit is 5000 or greater

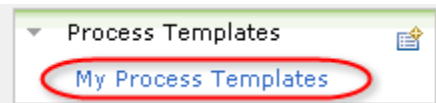
\_\_96. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Launch -> Business Process Choreographer Explorer**.

**Hint**

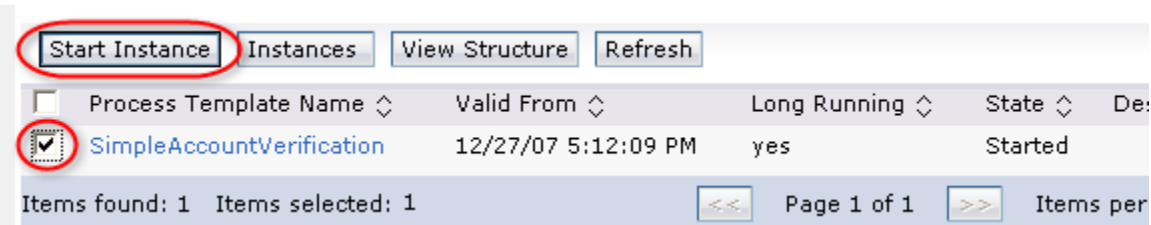


You now have another way of starting the process. You can also use a web browser (<http://localhost:9080/StartProcess>), instead of using the Business Process Choreographer Explorer.

\_\_97. Click on the **My Process Templates** link to display a list of processes which can be started.



\_\_98. Place a checkmark beside **SimpleAccountVerification**. Click on **Start Instance**.



\_\_99. From the input page, specify a requestedLimit of **1000** or any amount below 5000. Click on **Add**.

requestedLimit

customer


\_\_100. For the **customerID** field, type **123**.

customer

customerID

companyName

**Hint**



You specified a Customer ID of "123" so that the process will follow the 'denied' execution path and bypass the human task. This will simplify the test. Specifying a different Customer ID will go through the 'accepted' execution path and will require a response from the Pricing and Approval human task. At this point, you are only testing to determine which web service will be invoked.

\_\_101. Click on **Submit**.

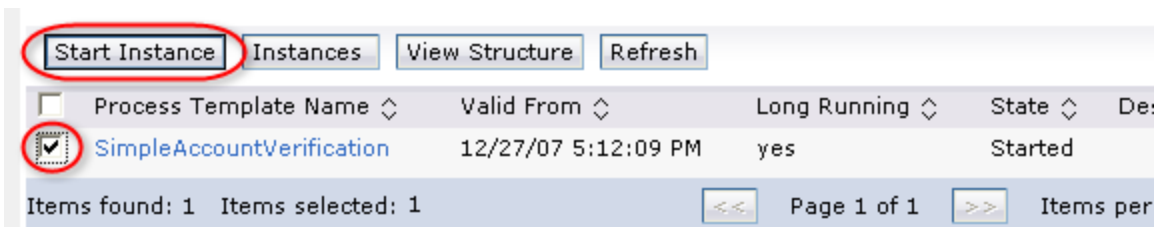
\_\_102. After a few moments, the process will generate messages to the **Console**.

\_\_103. Verify that the Internal Credit Report web service was invoked.

```

○ *****
○ ** Call Internal Credit Report Service **
○ *****
  
```

\_\_104. Restart the **SimpleAccountVerification** process again.



\_\_105. This time, specify a requestedLimit of **6000** or any amount greater than 5000. Click on **Add**.

requestedLimit	<input type="text" value="6000"/>
customer	<input type="button" value="Add"/>

\_\_106. For the **customerID** field, type **123**.

customer	customerID	<input type="text" value="123"/>
	companyName	<input type="text"/>

\_\_107. Click on **Submit**.

\_\_108. After a few moments, the process will generate messages to the **Console**.

\_\_109. Verify that the Equinox Credit Report web service was invoked.

```

○ *****
○ ** Call Equinox Credit Report Service **
○ *****
  
```



**What just happened?**

You have successfully implemented an Enterprise Service Bus mediation component to enable dynamic routing of Credit Report requests.

## 12.9 Cleanup

\_\_110. When you're done testing the process, close the Business Process Choreographer Explorer.



**Please continue to the next lab.**



## Lab 13 Enable Dynamic Service Invocation

### Goals:

- **Enhance process agility**
  - **Dynamically determine service endpoints at runtime**
  - **Select endpoints based on classifications and service metadata**
  - **Handle variable number of service endpoints**

### Role: Integration Developer / Architect

This lab will illustrate another level of flexibility through dynamic service invocations. This will showcase how service endpoints can be determined at runtime based on classifications, service metadata, and custom properties. In the current implementation, the ESB Mediation routes requests to either the InternalCreditReport web service or the external EquinoxCreditReport web service depending on the requested credit limit. Both web services are known by the ESB Mediation module. However, for added flexibility, the requirement now involves routing requests to web services that are only determined at runtime. The number of web services involved can also vary at runtime.

For this lab, you will modify the implementation of the mediation component to enable dynamic lookups of available services at runtime from the WebSphere Service Registry and Repository. Specifically, the process should only search for Credit Report web services at runtime that are classified for “**Production**” and have a status of “**available**”. New web services can also be added and made available to the process at runtime. Obsolete web services can dynamically be removed as well.

### 13.1 Add custom properties to the web services

#### Role: Architect

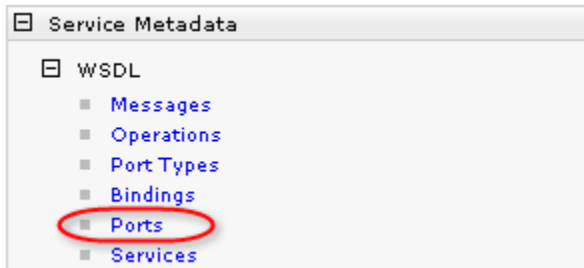
- \_\_\_1. Switch to the web browser showing the **WebSphere Service Registry and Repository** console.



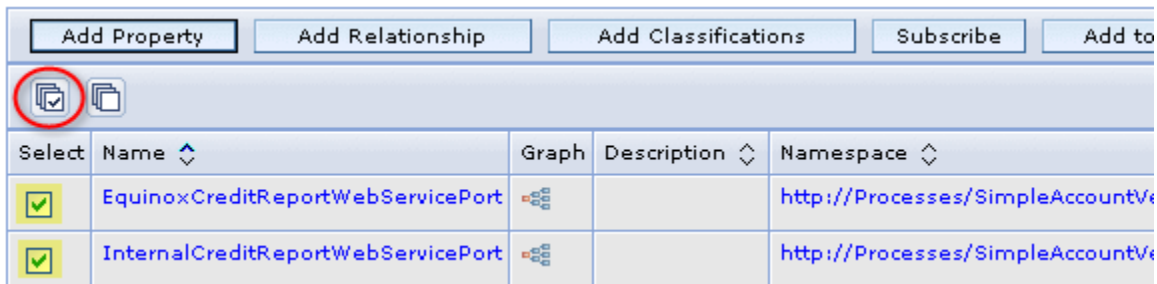
#### Troubleshooting

If the WebSphere Service Registry and Repository console was closed, just start a web browser and go to <http://localhost:9080/ServiceRegistry>.

\_\_2. Expand the **Service Metadata** -> **WSDL**. Click on **Ports**.



\_\_3. Click on the **Select all items** button.



\_\_4. Click on the **Add Property** button.

\_\_5. Specify a property name of **Status**. Specify a value of **available**.

**General Properties**

Name

Value



\_\_6. Click on **OK**.

**What just happened?**

The Status property will be used at runtime to determine which web service to use. In the lab scenario, only services with an "available" status will be used.

## 13.2 Specify the classification of the web services

\_\_7. Select **InternalCreditReportWebServicePort**.

Select	Name	Graph	Description	Namespace
<input type="checkbox"/>	EquinoxCreditReportWebServicePort			http://Processes/SimpleAccountV
<input checked="" type="checkbox"/>	InternalCreditReportWebServicePort			http://Processes/SimpleAccountV

\_\_8. Click on the **Add Classifications** button.

Add Classifications

\_\_9. Expand **Governance Profile Taxonomy -> Environment**. Select the **Production** classification. Click on **Add**.

Classification tree

Select	Classifications
<input type="checkbox"/>	WSRR Core Ontology
<input type="checkbox"/>	Default Lifecycle
<input type="checkbox"/>	GovernanceProfileLifecycle
<input type="checkbox"/>	Governance Profile Taxonomy
<input type="checkbox"/>	Business Domain
<input type="checkbox"/>	Environment
<input type="checkbox"/>	Development
<input checked="" type="checkbox"/>	Production
<input type="checkbox"/>	Test

Classification list

Add >>>

Remove

\_\_10. Click on **Production** in the Classification list.

Classification list

Production

\_\_11. In the Details section below, notice the URI. This will be used later as a lookup criteria when you implement the mediation flow.

Name	URI
Production	.../GovernanceProfileTaxonomy#Production

\_\_12. Click on **OK**.

OK



**What just happened?**

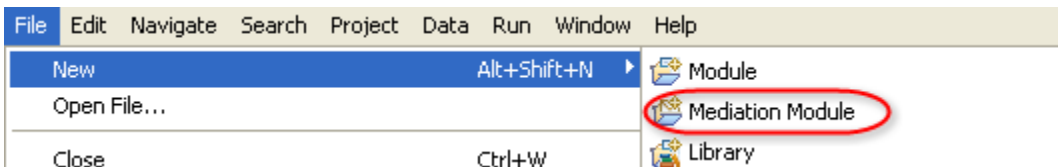
The Environment classification will be used at runtime to determine which web service to use. In the lab scenario, only services classified as “Production” ready will be used.

### 13.3 Create the Dynamic Credit Report Service Mediation Module

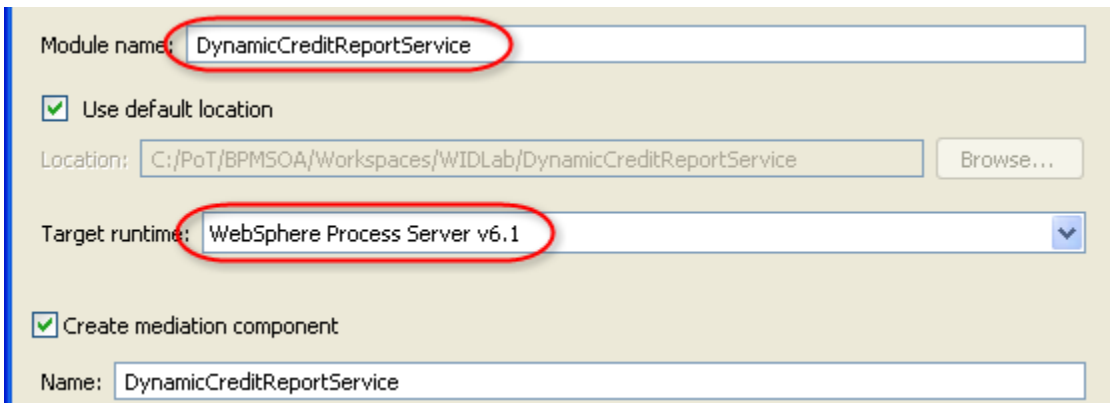
**Role: Integration Developer**

\_\_13. Switch back to the **WebSphere Integration Developer**.

\_\_14. From the main menu, select **File -> New -> Mediation Module**.

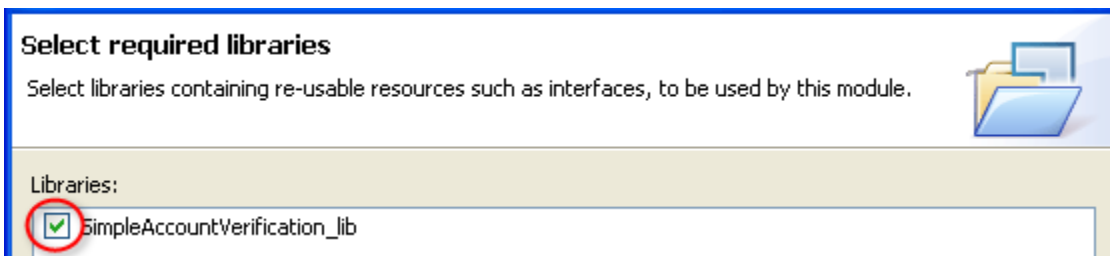


\_\_15. Specify **DynamicCreditReportService** for the Module Name. For the Target Runtime, select **WebSphere Process Server v6.1**.

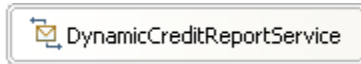


\_\_16. Click on **Next**.

\_\_17. Select **SimpleAccountVerification\_lib**. Click on **Finish**.

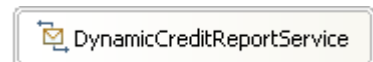


An Assembly Diagram editor appears containing a Mediation Flow component called DynamicCreditReportService.

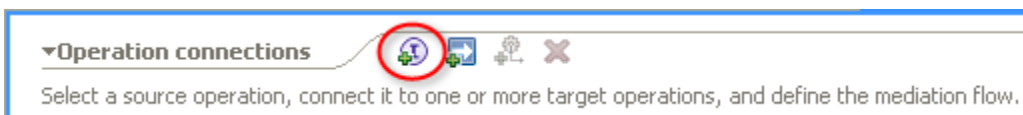


### 13.4 Implement the Mediation Flow

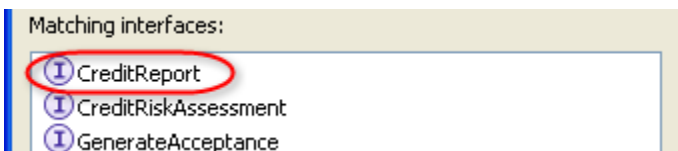
- \_\_18. Double-click on the **DynamicCreditReportService** component.  
A Mediation Flow Editor will appear.



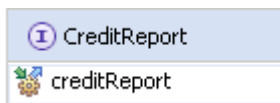
- \_\_19. Click on the **Add Interface** button.



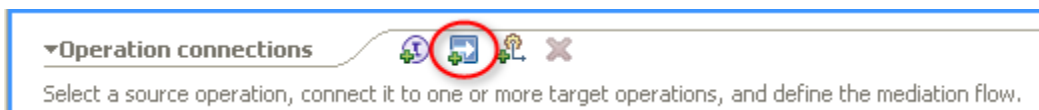
- \_\_20. Select **CreditReport** from the list. Click on **OK**.



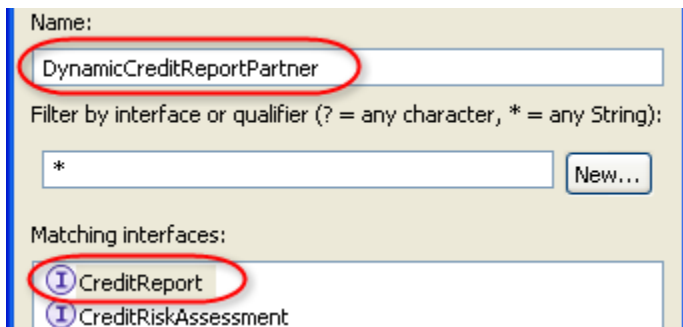
A CreditReport Interface component will appear in the Flow editor.



- \_\_21. Click on the **Add Reference** button.



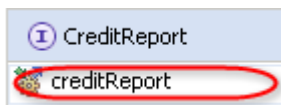
- \_\_22. Change the Name to **DynamicCreditReportPartner**. Select **CreditReport** from the list. Click on **OK**.



You should now see the CreditReport Interface and DynamicCreditReportPartner Reference components.

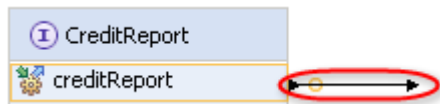


- \_\_23. Right-click on the **creditReport** operation of the CreditReport Interface on the left.



- \_\_24. From the popup menu, select **Create Operation Connection**.

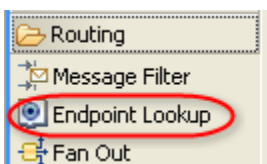
This will start a connection.



- \_\_25. Click on the **creditReport** operation of DynamicCreditReportPartner to complete the connection.



- \_\_26. From the Palette, expand the **Routing** folder. Select the **Endpoint Lookup** primitive.



\_\_27. Click to drop the **Endpoint Lookup** in the area indicated by the following screenshot. Rename **EndpointLookup1** to **DynamicServiceRouter**.



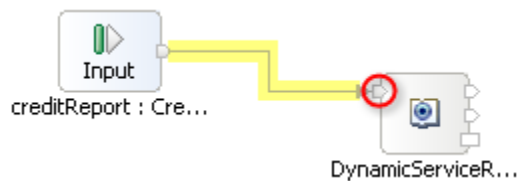
\_\_28. Move the mouse on top of the **out** terminal for the **Input** node.



This will expose the yellow connector.



\_\_29. Connect the yellow connector to the **in** terminal of **DynamicServiceRouter**.



### 13.5 Define the service endpoint lookup criteria

\_\_30. Right-click on the **DynamicServiceRouter** primitive.



\_\_31. From the popup menu, select **Show in Properties**.

\_\_32. Click on the **Details** tab.

\_\_33. For the Name, use the **Browse** button to select **CreditReport**.

Name:

Namespace:


Version:

\_\_34. For the **Registry Name**, specify **wsrpot**. For the match policy, select **Return all matching endpoints and set alternate routing targets**.

Registry Name:

Match Policy:


**Hint**



The Registry Name “wsrpot” is the alias of a WebSphere Service Registry and Repository server that was previously defined as part of the WebSphere Process Server configuration.

\_\_35. Click on the **Advanced** tab.

**What’s next?**



You will now be specifying the criteria to determine how services will be selected from the WebSphere Service Registry and Repository. The first criteria will be based on classifications. For this lab, only Credit Report web services with a classification of “Production” will be selected. To add classifications as a lookup criteria, you will use the URI of the classification.

\_\_36. Switch to the **PoT Lab Shortcuts** folder. Double-click on the **Misc.txt** link. Reopen the folder from the desktop if needed.

\_\_37. From the Misc.txt editor, copy the following text (Ctrl+c or Ctrl+Ins):

*http://www.ibm.com/xmlns/prod/serviceregistry/6/1/GovernanceProfileTaxonomy#Production*

Production Classification URI:  
<http://www.ibm.com/xmlns/prod/serviceregistry/6/1/>

...Screenshot continued...

...Screenshot continued

[/6/1/GovernanceProfileTaxonomy#Production](#)

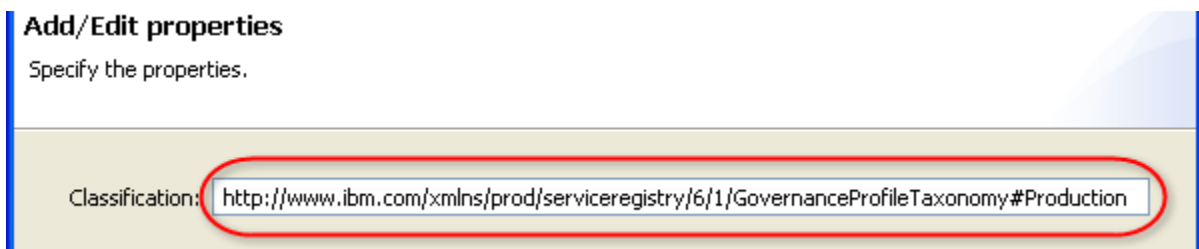


\_\_38. Switch to the **WebSphere Integration Developer**.

\_\_39. For Classifications, click on the **Add** button.




\_\_40. Paste the copied text into the Classification field.



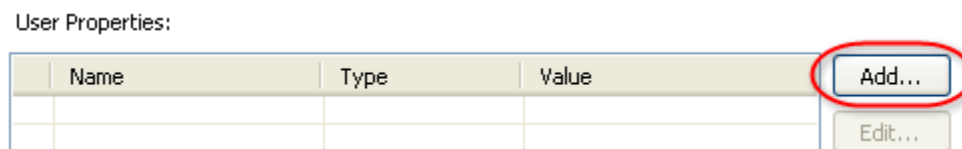
\_\_41. Click on **Finish**.

**What's next?**

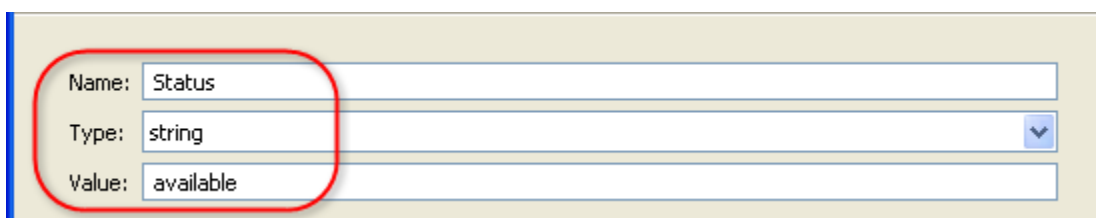


For the second lookup criteria, you will use the “Status” property you added earlier to the Credit Report web services in the WebSphere Service Registry and Repository. Specifically, you will specify that only web services with a Status of “available” will be selected.

\_\_42. Scroll down to the User Properties, and then click on the **Add** button.



\_\_43. For the Name, specify **Status**. For the Type, select **string**. For the Value, specify **available**.

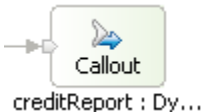


\_\_44. Click on **Finish**.

\_\_45. Connect the **out** terminal of DynamicServiceRouter to the **in** terminal of CreditReportPartner.



\_\_46. Right-click on the **Callout** node.



\_\_47. From the popup menu, select **Show in Properties**.

\_\_48. Click on the **Retry** tab.

\_\_49. For the **Retry on** field, specify **Any fault**. For the **Retry count**, specify **30**. For the **Retry delay**, specify **10**.

Retry on:

Retry count:

Retry delay (seconds):

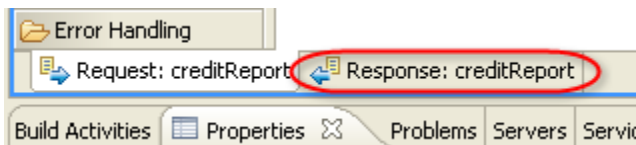
Try alternate endpoints

**Hint**



If the intended Credit Report web services fail, and there are also no alternate web services that can be used, then the settings above will allow the WebSphere Enterprise Bus to keep retrying up to 30 times in 10 second intervals. This helps improve process reliability.

\_\_50. Click on the **Response: InputCriterion** tab to switch to the Response Flow editor.



\_\_51. Connect the **out** terminal of Callout Response to the **in** terminal of Input Response.

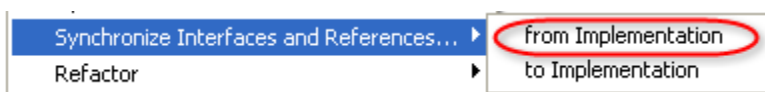


- \_\_52. Press **Ctrl+s** to save the Mediation Flow.
- \_\_53. Close the Mediation Flow Editor.  
Focus should return to the Assembly Diagram editor for **DynamicCreditReportService**.

### 13.6 Complete the DynamicCreditReportService assembly diagram

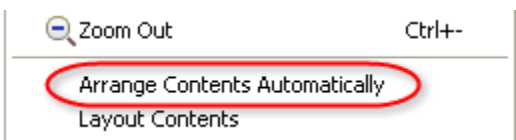
- \_\_54. Right-click on **DynamicCreditReportService**. 

- \_\_55. From the popup menu, select **Synchronize Interfaces and References -> from Implementation**.



- \_\_56. From the confirmation window, click **Yes**.

- \_\_57. Right-click on an empty space in the canvas. From the popup menu, select **Arrange Contents Automatically** if it has not yet been enabled (no check mark).



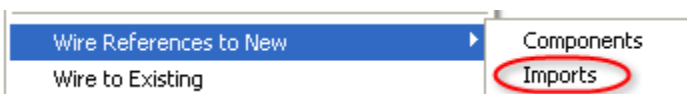
#### What's next?



In the previous ESB lab, you used a drag-and-drop approach to completing the CreditReportServiceRouter assembly diagram. You dragged the needed web service ports from the Business Integration view to the assembly diagram editor. For this lab, you will use a different approach to completing this assembly diagram. You will instead choose the components using the popup menu.

- \_\_58. Right-click on **DynamicCreditReportService**. 

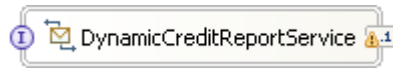
- \_\_59. From the popup menu, select **Wire References to New -> Imports**.



The DynamicCreditReportPartner component is created and wired to the DynamicCreditReportService.



\_\_60. Right-click on **DynamicCreditReportService**.



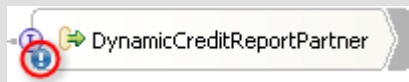
\_\_61. From the popup menu, select **Generate Export -> SCA Binding**.



\_\_62. Press **Ctrl+s** to save the DynamicCreditReportService Assembly Diagram.

**Hint**

Notice the warning icon on the DynamicCreditReportPartner.



This is because only an interface was specified. No web service port or implementation was bound to the component. However, this is by design.

A web service interface, such as the Credit Report Interface, contains the abstract definition of the service operations and associated inputs/outputs. A web service port represents the concrete implementation of the web service, such as the Equinox Credit Report Service. You will not specify a web service port at this time because you only need to use the interface. This will allow the module to dynamically determine which web service port to use at runtime.

**What happened?**



All the components have now been assembled for the DynamicCreditReportService. This mediation flow has been set up so that requests for a Credit Report will be forwarded to either the Internal or the Equinox Credit Report web service depending on the classification and status of the services. These can be changed at runtime.

\_\_63. Switch to the **Problems** view. Verify that no errors exist (messages with a red 'x' mark ✖). Warnings and information messages are expected.

\_\_64. Close the Assembly Diagram editor for DynamicCreditReportService.

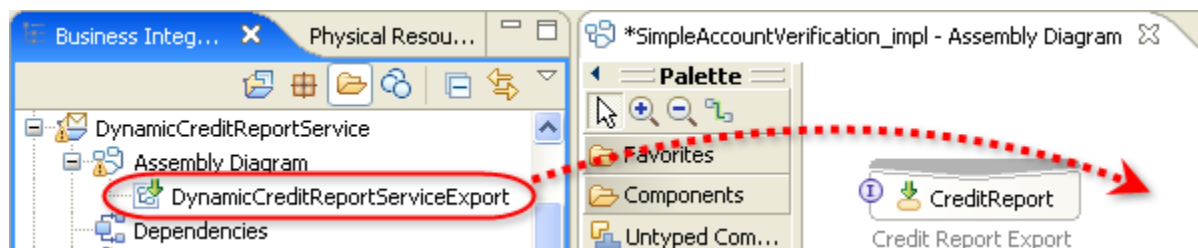
## 13.7 Re-assemble the SimpleAccountVerification Process to use the DynamicCreditReportService Module

- \_\_65. From the Assembly Diagram editor for SimpleAccountVerification, delete the **CreditReportServiceRouter** component.

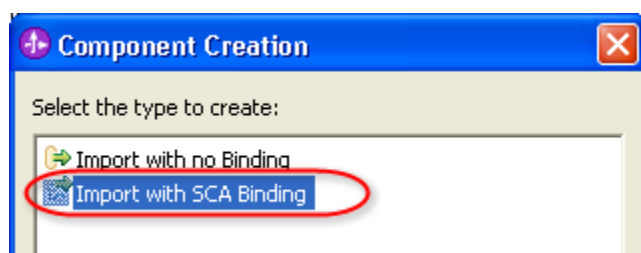


This will be replaced by the **DynamicCreditReportService** mediation component.

- \_\_66. From the Business Integration view, expand **DynamicCreditReportService -> Assembly Diagram**. Select **DynamicCreditReportServiceExport**. Drag and drop to an empty space in the Assembly Diagram editor.



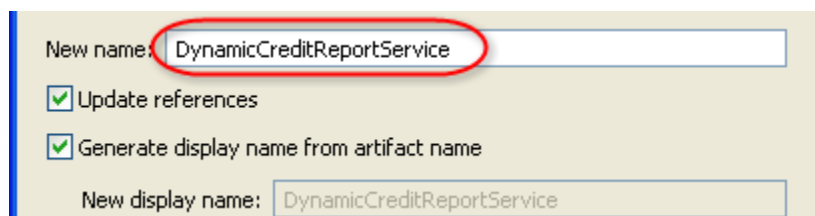
- \_\_67. From the **Component Creation** window, select **Import with SCA Binding**. Click on **OK**.



- \_\_68. Right-click on **Import1**. From the popup menu, select **Refactor -> Rename**.

- \_\_69. Click on **OK** to save the assembly diagram before refactoring.

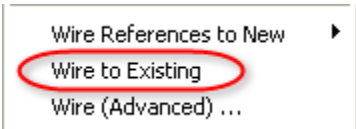
- \_\_70. Change the name of **Import1** to **DynamicCreditReportService**. Click on **OK**.



\_\_71. Right-click on **CreditReport**.



\_\_72. From the popup menu, select **Wire to Existing**.



This will create a connection between CreditReport and DynamicCreditReportService.



\_\_73. Press **Ctrl+s** to save the changes.

\_\_74. Switch to the **Problems** view. Verify that no errors exist (messages with a red 'x' mark ✖). Warnings and information messages are expected.

### 13.8 Retest the SimpleAccountVerification Process

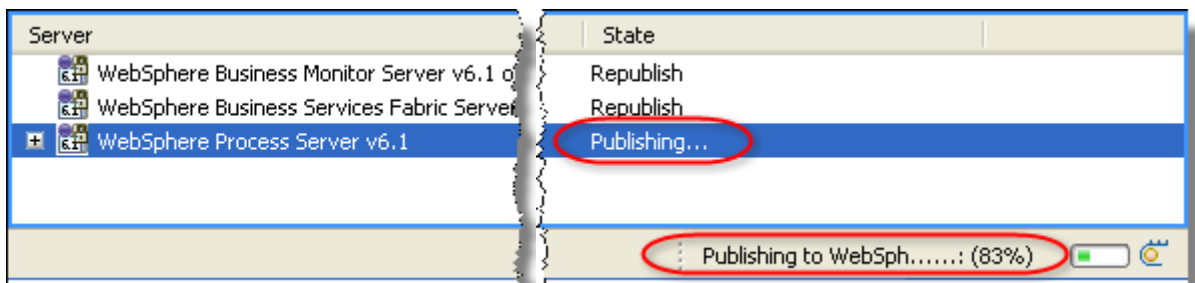
\_\_75. Switch to the **Servers** view.

\_\_76. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Add and Remove projects**.

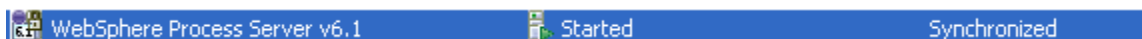
\_\_77. From the **Add and Remove Projects** window, click on **Add All**.

\_\_78. Click on **Finish**.

\_\_79. The server state will change to **Publishing**. You will also see a progress indicator at the bottom.



- \_\_80. Wait for the progress indicator at the bottom to disappear. From the Servers view, check that the server state has changed to **Synchronized**.



**What's next?**

You will now retest the process with its dynamic service invocation capability. Earlier you set the status of both the Internal and Equinox web services as “available”. However, only the Internal Credit Report web service was classified as “Production” approved.

Web Service	Status	Classification
Internal	available	Production
Equinox	available	

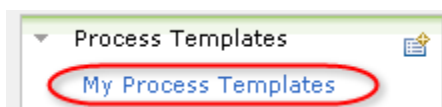
Only web services with both a status of “available” and a classification of “Production” will be selected by the SimpleAccountVerification process. For the next test, only the Internal web service will be selected.

- \_\_81. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Launch -> Business Process Choreographer Explorer**.

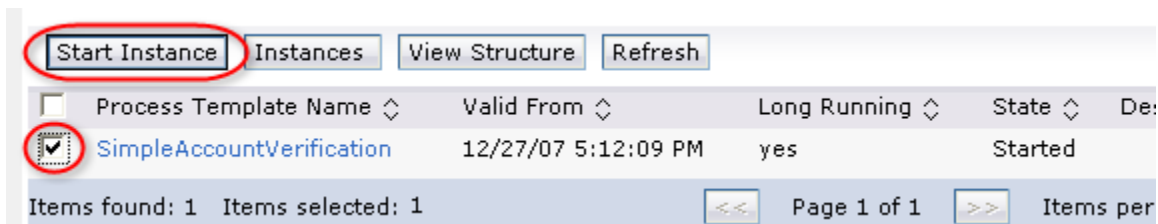
**Hint**

You can also start the process from a web browser (<http://localhost:9080/StartProcess>), instead of using the Business Process Choreographer Explorer.

- \_\_82. Click on the **My Process Templates** link to display a list of processes which can be started.



- \_\_83. Place a checkmark beside **SimpleAccountVerification**. Click on **Start Instance**.




\_\_84. Click on **Add**.

\_\_85. For the **customerID** field, type **123**.

customer	customerID	123
	companyName	

**Hint**



You specified a Customer ID of “123” so that the process will follow the ‘denied’ execution path and bypass the human task. At this point, you are only testing to determine which web service will be invoked.

\_\_86. Click on **Submit**.


\_\_87. After a few moments, the process will generate messages to the **Console**.

\_\_88. Verify that the Internal Credit Report web service was invoked.

```


○ *****
○ ** Call Internal Credit Report Service **
○ *****
    
```

**What just happened?**



The SimpleAccountVerification process used the Internal Credit Report web service because it is the only service with a “Production” classification.

**What’s next?**



You will now assume the role of an architect and change the classification of the web services to see how this dynamically changes which web service is invoked by the process at runtime.

## 13.9 Change the classification to change the service selection

**Role: Architect**

\_\_89. Switch to the web browser with the **WebSphere Service Registry and Repository** console.

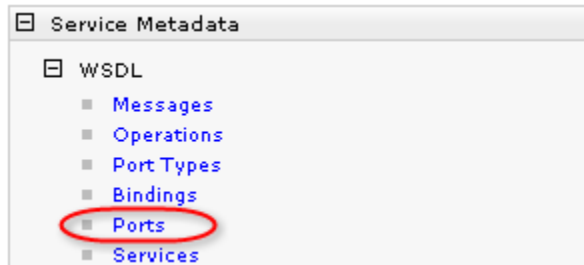
**Troubleshooting**





If the WebSphere Service Registry and Repository console was closed, just start a web browser and go to <http://localhost:9080/ServiceRegistry>.



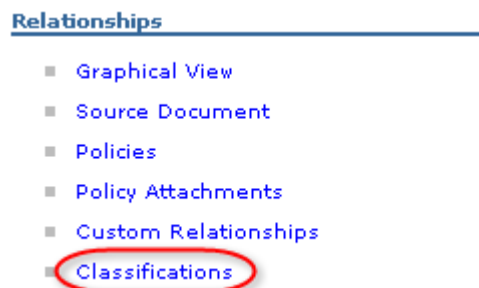
\_\_90. Expand the **Service Metadata** -> **WSDL**. Click on **Ports**.



\_\_91. Click on **InternalCreditReportWebServicePort**.

Select	Name	Graph	Description	Namespace
<input type="checkbox"/>	EquinoxCreditReportWebServicePort			http://Processes/SimpleAccountV...
<input type="checkbox"/>	<b>InternalCreditReportWebServicePort</b>			http://Processes/SimpleAccountV...

\_\_92. Click on **Classifications**.

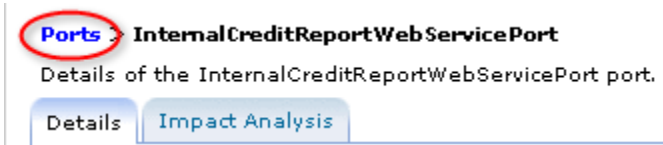


\_\_93. Select **Production** from the Classification list. Click on **Remove**.



\_\_94. Click on **OK**. 

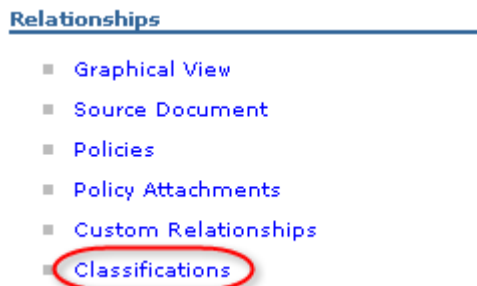
\_\_95. Click on the **Ports** link at the top of the page.



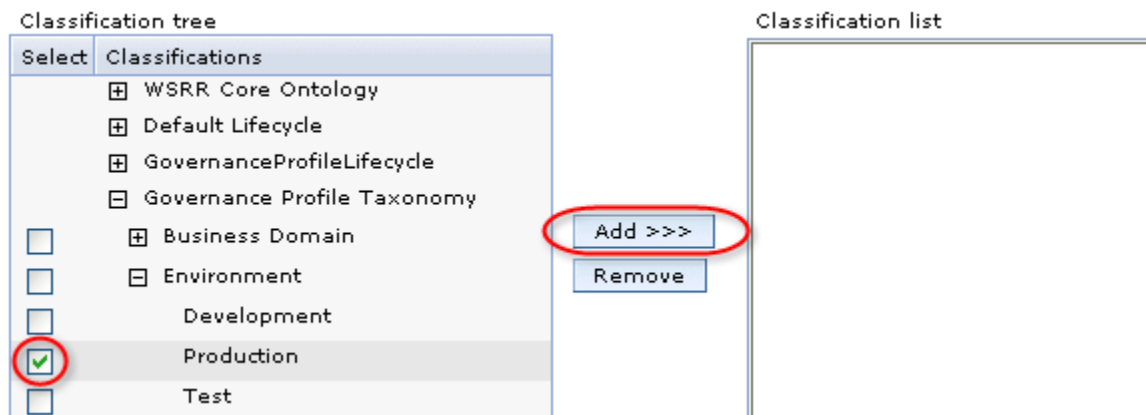
\_\_96. Click on **EquinoxCreditReportWebServicePort**.

Select	Name	Graph	Description	Namespace
<input type="checkbox"/>	EquinoxCreditReportWebServicePort			http://Processes/SimpleAccountV...
<input type="checkbox"/>	InternalCreditReportWebServicePort			http://Processes/SimpleAccountV...

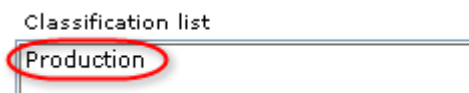
\_\_97. Click on **Classifications**.



\_\_98. Select the **Production** classification. Click on **Add**.




\_\_99. Verify that **Production** in the Classification list.



\_\_100. Click on **OK**. 

**What just happened?**

You removed the “Production” classification from the Internal Credit Report web service and added it to the Equinox web service.

	Web Service	Status	Classification
	Internal	available	
	Equinox	available	Production

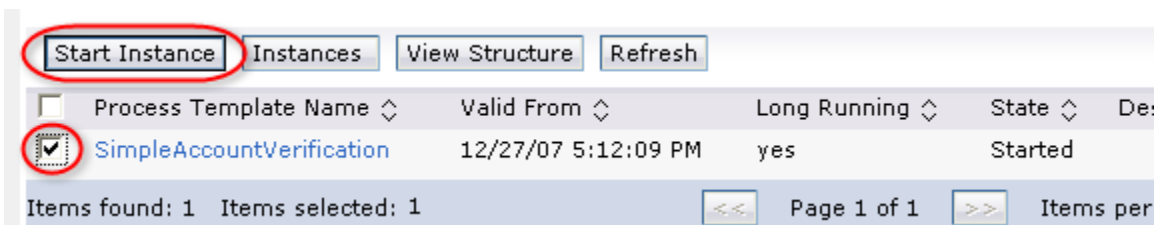
Only web services with both a status of “available” and a classification of “Production” will be selected by the SimpleAccountVerification process. For the next test, only Equinox will be selected.

## 13.10 Retest the SimpleAccountVerification Process

**Role: Integration Developer**

\_\_101. Switch to the **WebSphere Integration Developer**.

\_\_102. Place a checkmark beside **SimpleAccountVerification**. Click on **Start Instance**.



The screenshot shows the 'Start Instance' button circled in red. Below it is a table of process templates. The 'SimpleAccountVerification' row has a checkmark in the selection column. The table columns are: Process Template Name, Valid From, Long Running, State, and De:.

Process Template Name	Valid From	Long Running	State	De:
SimpleAccountVerification	12/27/07 5:12:09 PM	yes	Started	


Items found: 1 Items selected: 1

\_\_103. Click on **Add**.

\_\_104. For the **customerID** field, type **123**.

customer	customerID	<input type="text" value="123"/>
	companyName	<input type="text"/>

**Hint**

 You specified a Customer ID of “123” so that the process will follow the ‘denied’ execution path and bypass the human task. At this point, you are only testing to determine which web service will be invoked.


\_\_105. Click on **Submit**. 


\_\_106. After a few moments, the process will generate messages to the **Console**.

\_\_107. Verify that the Equinox Credit Report web service was invoked.

```

○ *****
○ ** Call Equinox Credit Report Service **
○ *****
  
```

 **What just happened?**  
 The SimpleAccountVerification process used the Equinox Credit Report web service because it is the only service with a “Production” classification.

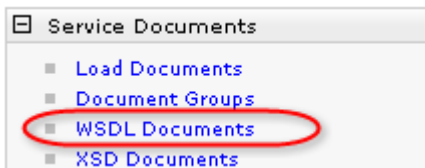
 **What’s next?**  
 You will again assume the role of an architect and add two new Credit Report web services, one from Experian and the other from Trans Union. Both of these new services will have a classification of “Production” and a status of “available”. Finally, you will change the status of the Equinox web service to “offline”. The process should only select either the Experian or the Trans Union web services.

### 13.11 Dynamically add two new Credit Report services and have these services available for the process at runtime

**Role: Architect**

\_\_108. Switch to the web browser with the **WebSphere Service Registry and Repository** console.

\_\_109. Expand the **Service Documents** section. Click on **WSDL Documents**.



\_\_110. Click on **Load Documents**. 

\_\_111. Click on **Browse** to select **C:\PoT\BPMSOA\ExperianCreditReportWebService.wsdl**.

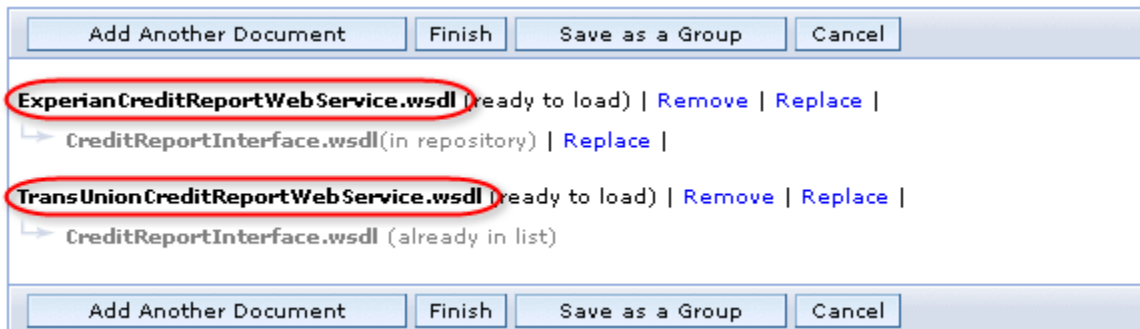
\_\_112. Click on **OK**.

\_\_113. Click on **Add Another Document**. 

\_\_114. Click on **Browse** to select **C:\PoT\BPMSOA\TransUnionCreditReportWebService.wsdl**.

\_\_115. Click on **OK**.

\_\_116. Verify that the WSDL files for the two new external credit report services appear on the list.



\_\_117. Click on **Finish**. 

A message appears that the documents were loaded successfully.

#### **Documents Loaded Successfully**

The following documents have been loaded into the repository:

Name	Description	Namespace
<a href="#">ExperianCreditReportWebService.wsdl</a>		http://Processes/SimpleAccountVerification/Cr
<a href="#">TransUnionCreditReportWebService.wsdl</a>		http://Processes/SimpleAccountVerification/Cr

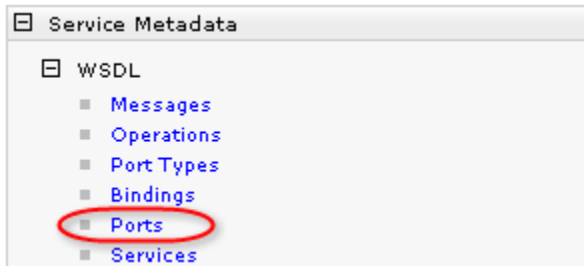
#### **What just happened?**

The WSDL files from the two new external credit report agencies (Experian, Trans Union) were loaded in the WebSphere Service Registry and Repository. These WSDLs will be reviewed and approved for use within the company. These web services will serve as additional service endpoints for the Credit Report task in the SimpleAccountVerification process.



This illustrates how a dynamic service invocation approach will allow new service providers to participate in business processes at runtime. New web services can be used by the process without code modifications and redeployments.

\_\_118. Expand the **Service Metadata** -> **WSDL**. Click on **Ports**.



\_\_119. Click on **EquinoxCreditReportWebServicePort**.

\_\_120. Click on **Properties**.



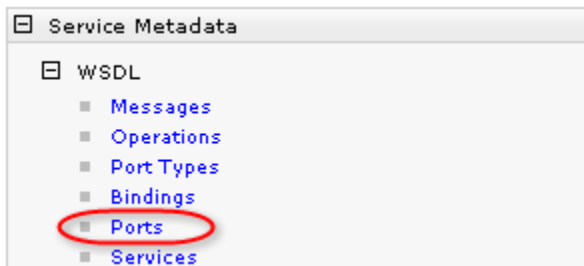
\_\_121. Click on **Status**.

\_\_122. Change the value to **offline**.







\_\_123. Click on **OK**.

\_\_124. From the **Service Metadata** section, click on **Ports** again.



\_\_125. Select **ExperianCreditReportWebServicePort** and **TransUnionCreditReportWebServicePort**.

Select	Name ▾	Graph	Description ▾	Namespace ▾
<input type="checkbox"/>	EquinoxCreditReportWebServicePort			http://Processes/SimpleAccou
<input checked="" type="checkbox"/>	ExperianCreditReportWebServicePort			http://Processes/SimpleAccou
<input type="checkbox"/>	InternalCreditReportWebServicePort			http://Processes/SimpleAccou
<input checked="" type="checkbox"/>	TransUnionCreditReportWebServicePort			http://Processes/SimpleAccou


\_\_126. Click on the **Add Property** button. 

\_\_127. Specify a property name of **Status**. Specify a value of **available**.





**General Properties**

Name

Value

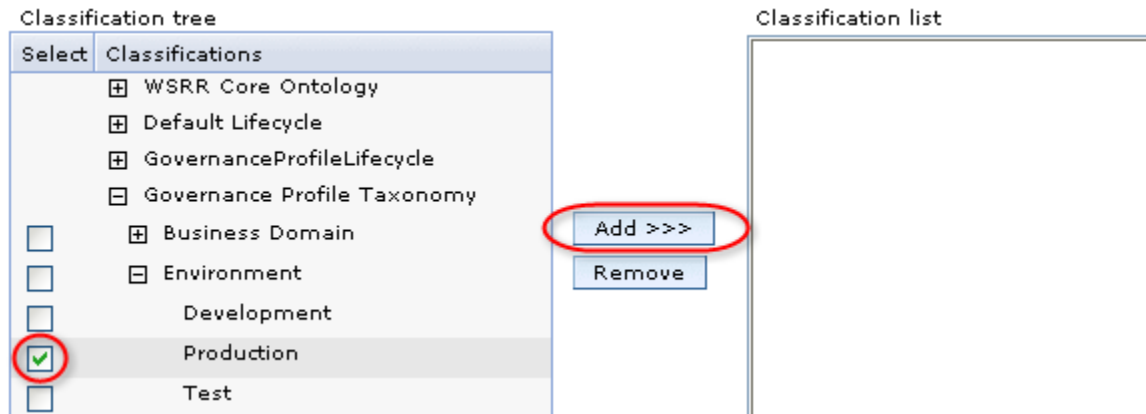
\_\_128. Click on **OK**. 

\_\_129. Select **ExperianCreditReportWebServicePort** and **TransUnionCreditReportWebServicePort** again.

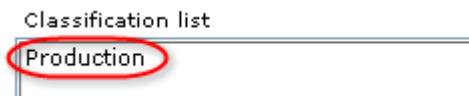
Select	Name ▾	Graph	Description ▾	Namespace ▾
<input type="checkbox"/>	EquinoxCreditReportWebServicePort			http://Processes/SimpleAccou
<input checked="" type="checkbox"/>	ExperianCreditReportWebServicePort			http://Processes/SimpleAccou
<input type="checkbox"/>	InternalCreditReportWebServicePort			http://Processes/SimpleAccou
<input checked="" type="checkbox"/>	TransUnionCreditReportWebServicePort			http://Processes/SimpleAccou

\_\_130. Click on the **Add Classifications** button. 

\_\_131. Select the **Production** classification. Click on **Add**.




\_\_132. Verify that **Production** in the Classification list.



\_\_133. Click on **OK**. 

**What just happened?**

You added two new Credit Report web services (Experian, TransUnion), added classifications, and set the status properties. Here is a summary of the state of each Credit Report web service:

	Web Service	Status	Classification
	Internal	available	
	Equinox		Production
	Experian	available	Production
	TransUnion	available	Production

Only web services with both a status of “available” and a classification of “Production” will be selected by the SimpleAccountVerification process. For the next test, only Experian or TransUnion can be selected.

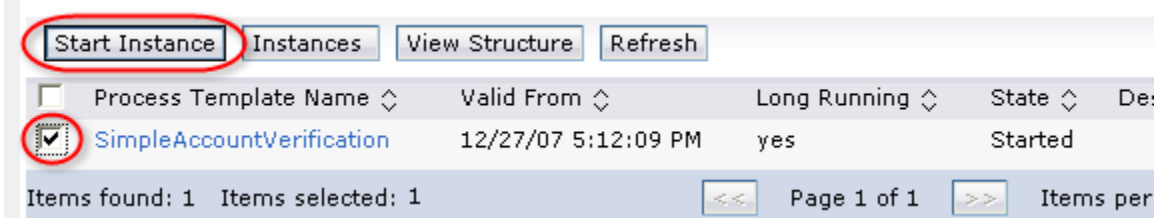
## 13.12 Retest the SimpleAccountVerification Process

**Role: Integration Developer**

\_\_134. Switch to the **WebSphere Integration Developer**.



\_\_135. Place a checkmark beside **SimpleAccountVerification**. Click on **Start Instance**.




\_\_136. Click on **Add**.

\_\_137. For the **customerID** field, type **123**.

customer	customerID	<input type="text" value="123"/>
	companyName	<input type="text"/>

**Hint**

 You specified a Customer ID of “123” so that the process will follow the ‘denied’ execution path and bypass the human task. At this point, you are only testing to determine which web service will be invoked.

\_\_138. Click on **Submit**.

\_\_139. After a few moments, the process will generate messages to the **Console**.


\_\_140. Verify that either the Experian ...

```

○ *****
○ ** Call Experian Credit Report Service **
○ *****
... or TransUnion Credit Report web service was invoked.
○ *****
○ ** Call TransUnion Credit Report Service **
○ *****


```

**What just happened?**

 The SimpleAccountVerification process selected either the TransUnion or Experian Credit Report web service because these are the only web services with a classification of “Production” and a status of “available”

Because two services can be selected, if one web service fails, then the process will select the other web service that is available and production ready. This helps improve process reliability.

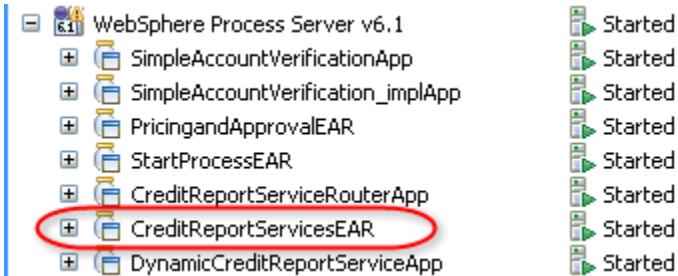
**What's next?**



There was another option that you set earlier which also helps improve process reliability. This was the retry option in the mediation flow. You will now test that retry capability. You will shut down all Credit Report web services, and although this will generate exceptions, the mediation component will keep retrying up to 30 times in 10 second intervals.

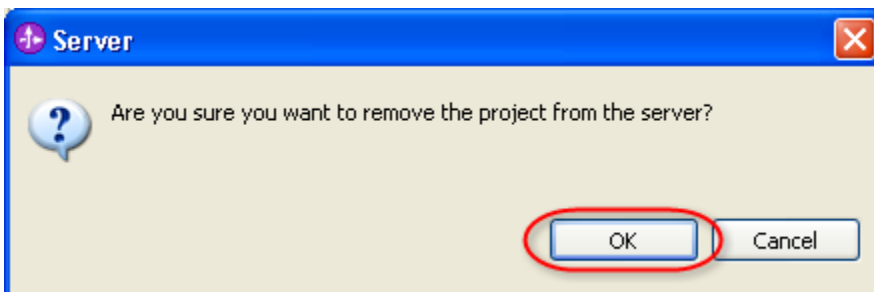
\_\_141. Switch to the **Servers** view.

\_\_142. Expand **WebSphere Process Server v6.1**. Right-click on **CreditReportServicesEAR**.



\_\_143. From the popup menu, select **Remove**.

\_\_144. From the confirmation window, click on **OK**.



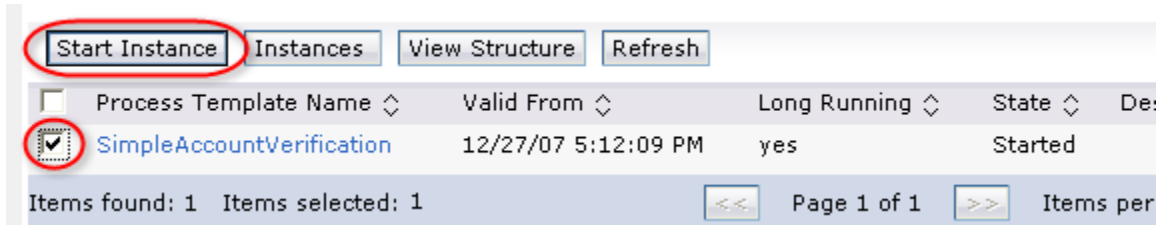
\_\_145. Wait for the progress indicator at the bottom to disappear before proceeding.

**What just happened?**



You shut down all the Credit Report web services by removing these from the server.

\_\_146. Place a checkmark beside **SimpleAccountVerification**. Click on **Start Instance**.



\_\_147. Click on **Add**.

\_\_148. For the **customerID** field, type **123**.

customer	customerID	123
	companyName	

\_\_149. Click on **Submit**.

\_\_150. After a few moments, the process will generate messages to the **Console**. This time you will see repeating exceptions because the WebSphere Enterprise Service Bus is continuously retrying.

```
[1/9/08 19:19:50:843 EST] 0000004d PivotHandlerW W com.ibm.ws.webservice
WebServicesFault
faultCode: HTTP
faultString: ( 404 ) Not Found
faultActor: http://ProcIntg:9080
faultDetail:
    null: WSWS3192E: Error: return code: ( 404 ) Not Found
```



#### What's next?

With the retry option configured for 30 attempts and 10 seconds between each attempt, you have approximately 5 minutes to restart the Credit Report web services.

\_\_151. Switch to the **Servers** view.

\_\_152. Right-click on **WebSphere Process Server v6.1**. From the popup menu, select **Add and Remove projects**.

\_\_153. From the **Add and Remove Projects** window, click on **Add All**.

\_\_154. Click on **Finish**.

After a few moments, you will see messages in the Console view indicating that the process was able to invoke either the Experian ...

```
○ *****  
○ ** Call Experian Credit Report Service **  
○ *****
```

... or TransUnion Credit Report web service.

```
○ *****  
○ ** Call TransUnion Credit Report Service **  
○ *****
```



**What just happened?**

When an invocation of a web service failed, the WebSphere Enterprise Service Bus continued to try and access the web service based on the configured retry count and delay. This retry capability will significantly improve process and service reliability.

### 13.13 Cleanup

\_\_155. When you're finished testing, close the **Business Process Choreography Explorer**.

\_\_156. Close the **Misc.txt** editor.

### Congratulations!



**You have successfully completed all the labs.**

---

## Conclusion

Hopefully these labs helped illustrate the significant benefits which can be achieved using IBM's SOA Foundation and the WebSphere Business Process Management software portfolio.



## Appendix A. Loading Lab Solutions

In case you encounter problems, or just wish to skip ahead to the solution of a specific part in the lab, then the completed projects are also available. The solutions can be imported as Project Interchange files located in the following directory:

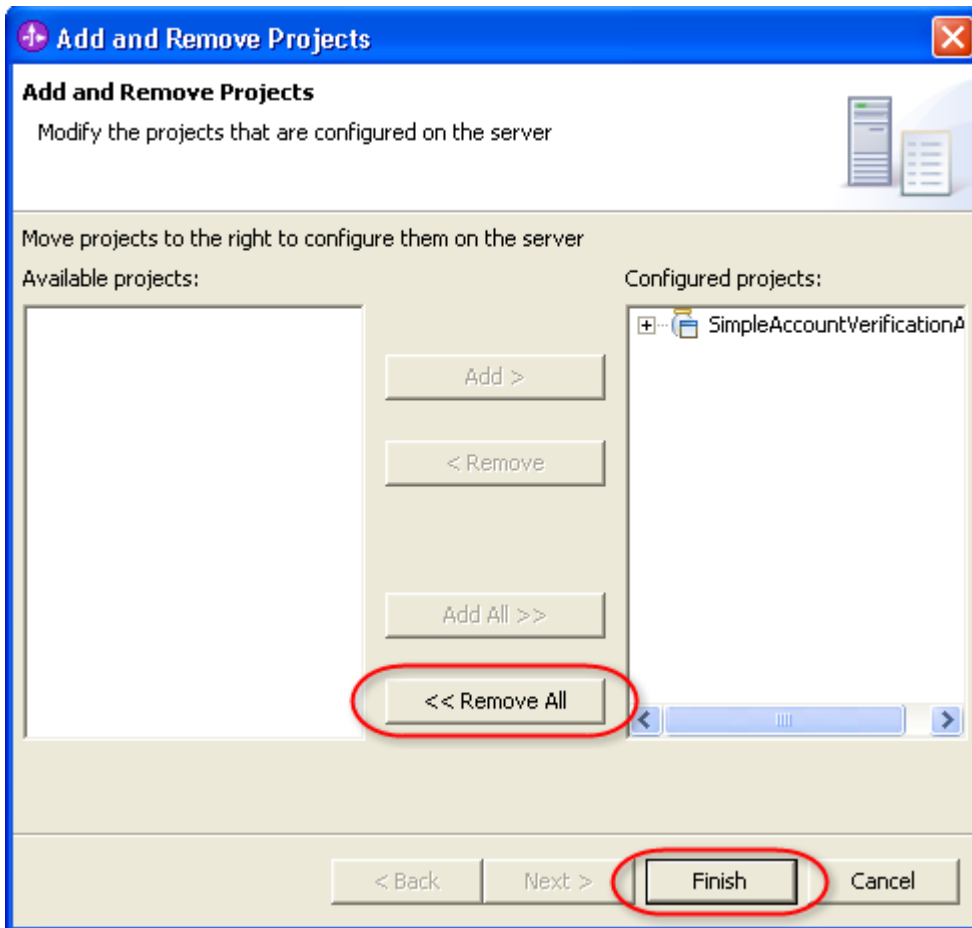
**C:\PoT\BPMSOA\Solutions**

Below is a list of the solution files which can be found in the directory above.

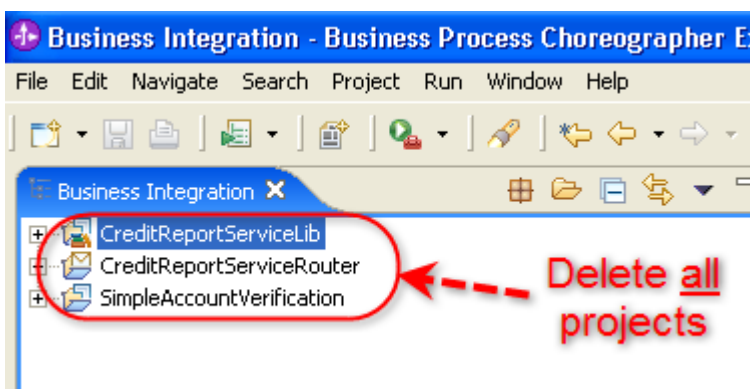
ModelerLabCompleted.mar	Lab ? fully completed (WebSphere Business Modeler)
JavaLabCompleted_Pi.zip	Lab ? fully completed
WebServiceLabCompleted_Pi.zip	Lab ? fully completed
BusinessRulesLabCompleted_Pi.zip	Lab ? fully completed
HumanTaskLabCompleted_Pi.zip	Lab ? fully completed
ESBLabCompleted_Pi.zip	Lab ? fully completed
DynamicServiceLabCompleted_Pi.zip	Lab ? fully completed

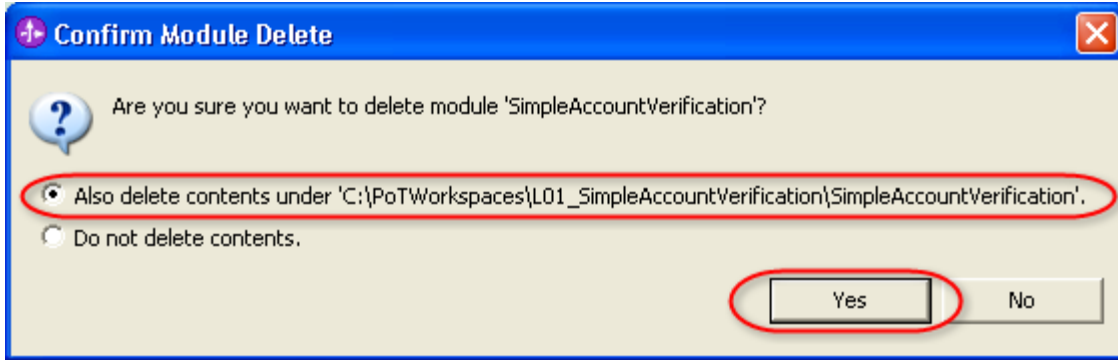
### Steps for loading a solution file into the WebSphere Integration Developer:

1. Ask for assistance from the instructors or lab assistants if necessary.
2. Remove all applications currently deployed to the test server. (Refer to the following screenshot)

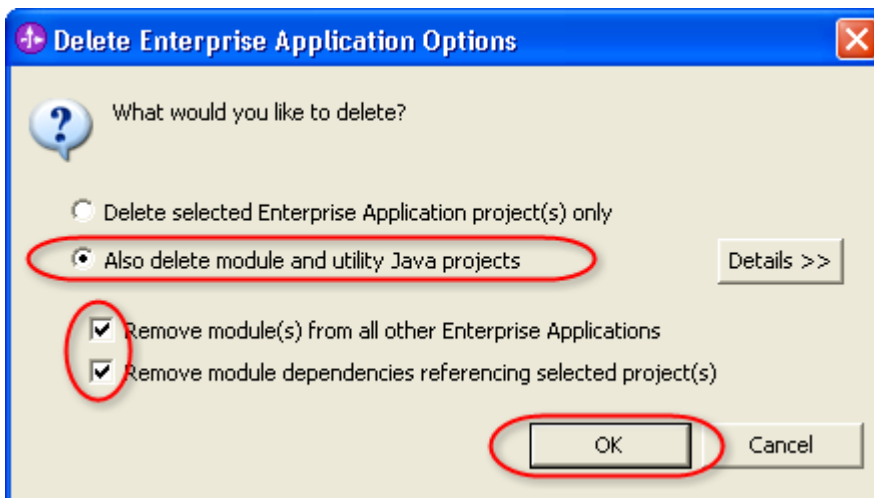
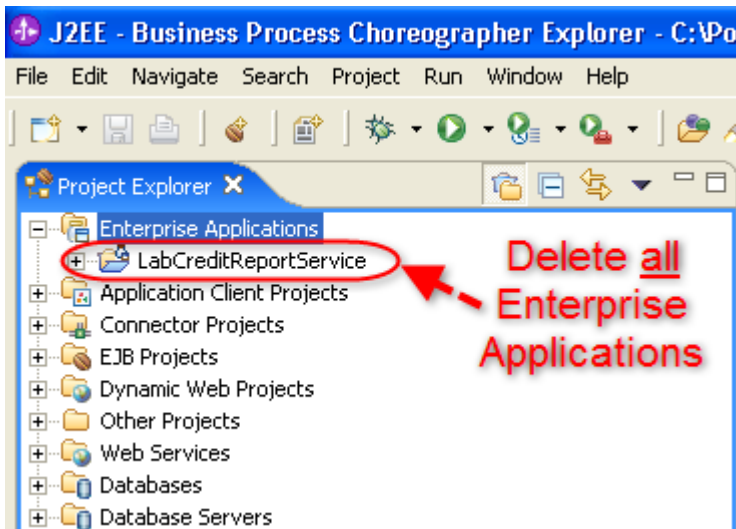


3. Delete all projects in the **Business Integration** view.

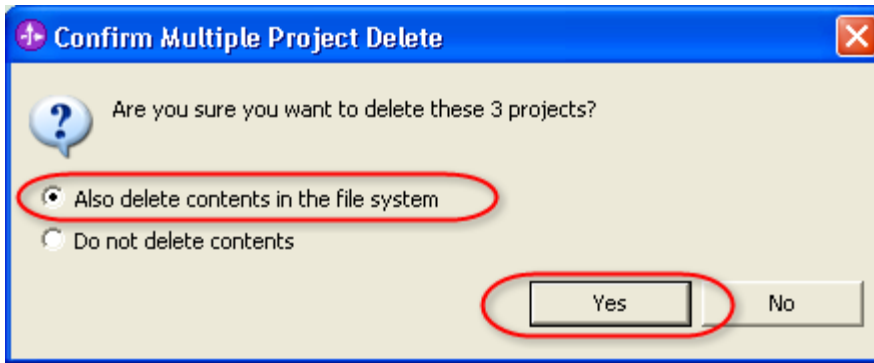




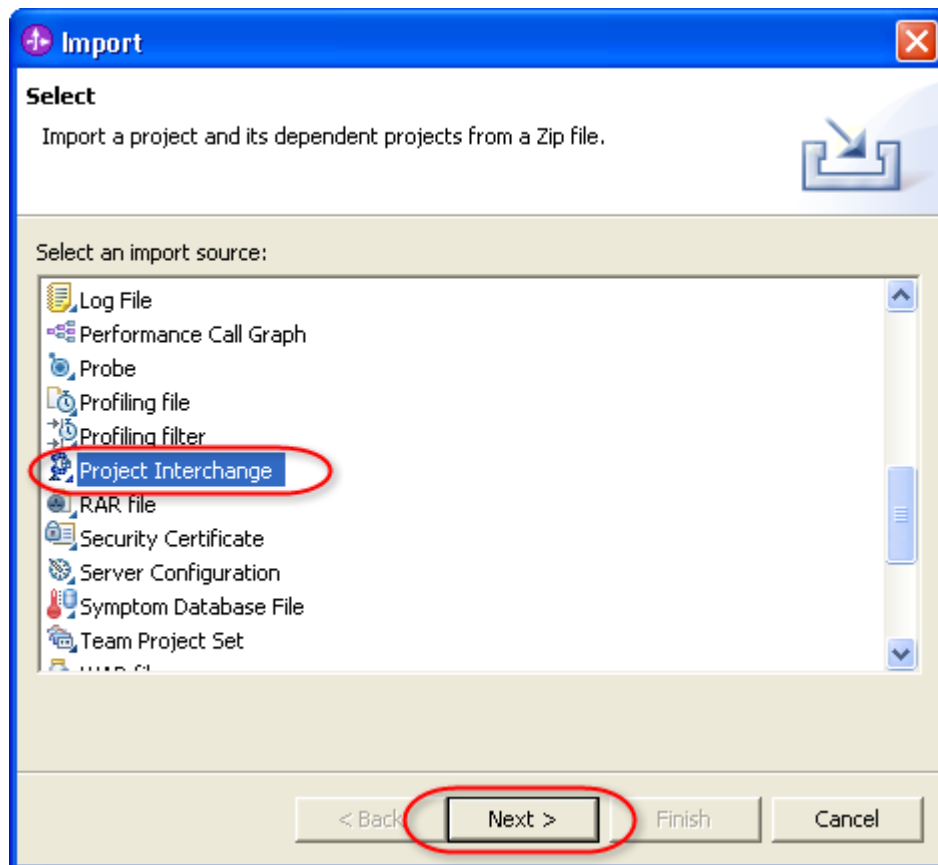
4. Switch to the **J2EE** perspective. Delete all enterprise applications.

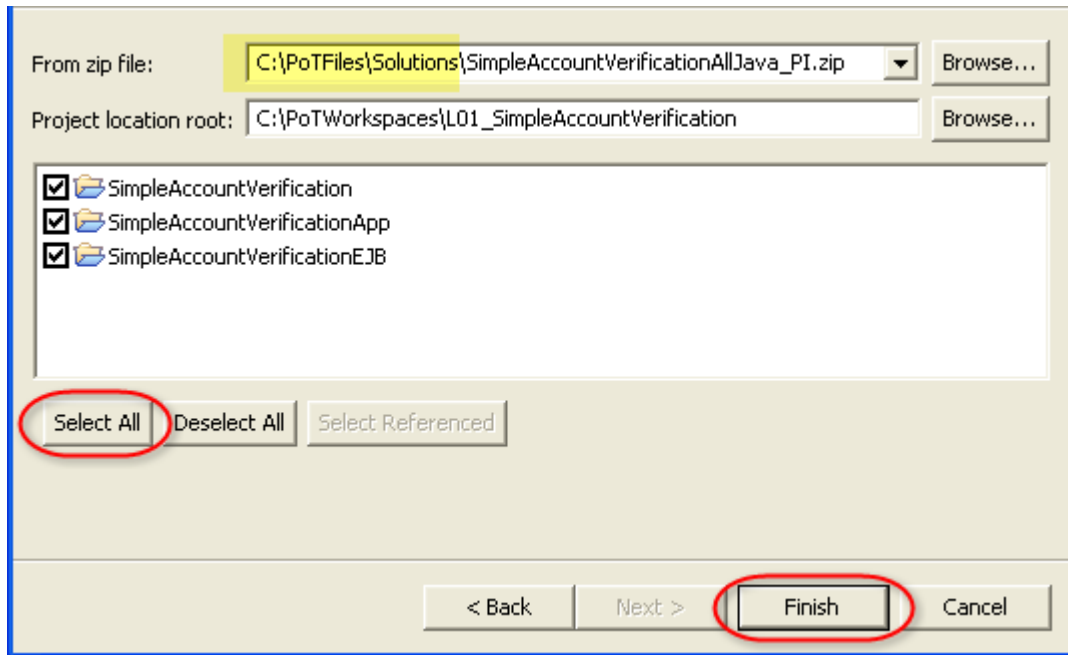




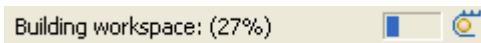


5. Switch back to the **Business Integration** perspective.
6. Import the desired Project Interchange file. Refer to the solution file table described above.

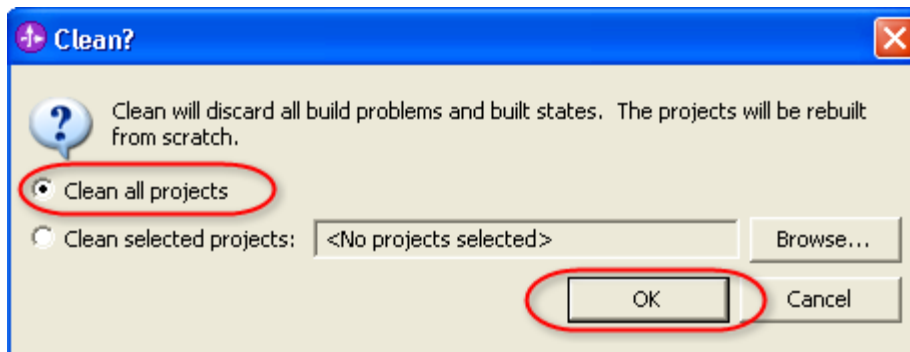
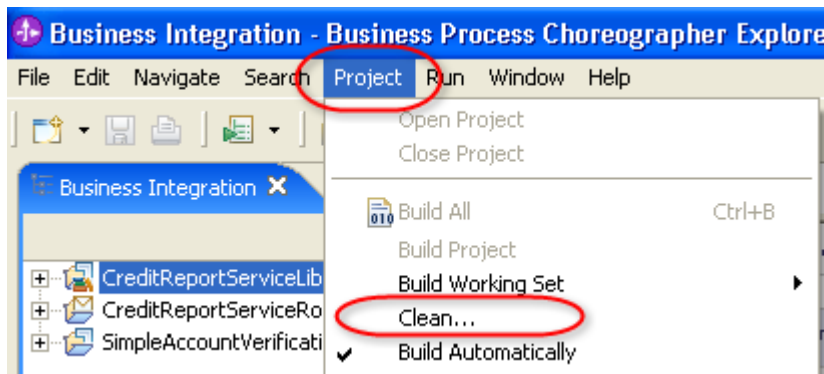




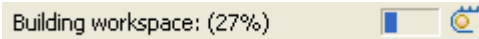
7. If a progress indicator appears, wait until it is complete.



8. Perform a Project Clean-all.



9. If a progress indicator appears, wait until it is complete.



10. Ensure that there are no errors listed in the Problems view. If errors exist, delete the following files and perform another Project Clean-all (repeat step 8):

SimpleAccountVerificationApp  
SimpleAccountVerificationEJB  
SimpleAccountVerificationWeb

Do not delete SimpleAccountVerification.

11. Start the test server if not yet started.
12. Add all projects to the test server.
13. Proceed to the appropriate section of the lab for testing the process.

---

## Appendix B. IBM TechWorks

TechWorks is a worldwide team serving IBM software sales, technical, and development professionals and through these organizations our customers and partners. The TechWorks organization works to:

- Provide implementation, execution, and expertise for brand and general technical sales strategies and objectives
- Provide deep technical and product knowledge directly to customers and partners
- Create material to teach our field teams, our partners, and our customers about what IBM products can do
- Operate world-class, high-quality facilities that support and enhance the interactions between our field technical sales team and our clients and partners

For more information about IBM TechWorks, talk with your IBM sales representative.

## Appendix C. Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
 IBM Corporation  
 North Castle Drive  
 Armonk, NY 10504-1785  
 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation  
 Licensing  
 2-31 Roppongi 3-chome, Minato-ku  
 Tokyo 106-0032, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

---

## Appendix D. Trademarks and copyrights

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	IBM logo	Cool Blue	DB2	Lotus	WebSphere
System i	iSeries	AIX	CICS	ClearCase	Cloudscape
DataPower	FileNet	i5/OS	Informix	IMS	pSeries
Rational	z/OS	zSeries			

Adobe, Acrobat, Portable Document Format (PDF), and PostScript are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both. See Java Guidelines

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product and service names may be trademarks or service marks of others.