IBM WebSphere® Enterprise Service Bus v 6.0.2 – Lab exercise

WebSphere Enterprise Service Bus lab – Unmodeled faults

What this exercise is about 1

Lab requirements 1

What you should be able to do 2

Introduction 2

Exercise instructions 3

Part 1: Set up development environment 5

Part 2: Create mediation module 9

Part 3: Import Web service interface 12

Part 4: Create the mediation flow 16

Part 5: Assemble the module 26

Part 6: Add the modules to the server 35

Part 7: Test the mediations with the Web Services Explorer 37

What you did in this exercise 45

Solution Instructions 46

Task: Adding remote server to WebSphere Integration Developer test environment 47

What this exercise is about

The objective of this lab is to provide you with an understanding of how to mediate an unmodeled fault within a mediation flow. Unmodeled faults occur for faults that are not defined in the WSDL interface. This lab will show you how to handle any faults that are not explicitly defined in the WSDL interface. You will create a mediation module to handle them and mediate them within a mediation flow component.

Lab requirements

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

- Web Sphere Integration Developer V6.0.2 installed
- Web Sphere Enterprise Server Bus V6.0.2 test environment installed
- Sample code in the directory C:\LabFiles602 (Windows) or /tmp/LabFiles602 (Linux and AIX)

What you should be able to do

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

- Import the project interchange file into the Web Sphere Integration Developer V6.0.2 development environment
- Create and edit a mediation module and mediation flow
- Wire the new fail terminals in the Callout Response nodes to mediation primitives
- Test the unmodeled fault using the Web Services Explorer.

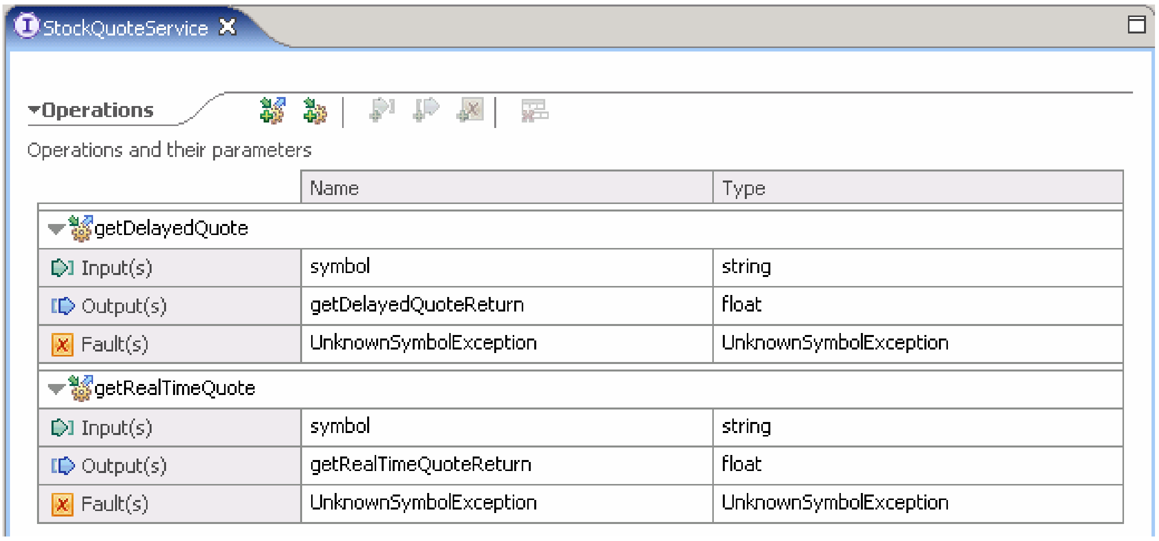
Introduction

This lab demonstrates the ability to mediate modeled and unmodeled faults within a mediation flow. It focuses particularly on the ability to route the two different types of faults (modeled and unmodeled) to the appropriate flow within the mediation flow component.

In this lab, the mediation flow will call an existing Web service through a Web services import binding. The Web service returns stock prices for a particular stock symbol and throws a fault message if the stock symbol is not known. This particular fault message will be modeled in the WSDL interface. To create an unmodeled fault, the mediation flow will be run with a stock quote symbol that contains invalid characters, for example, #. @. !.

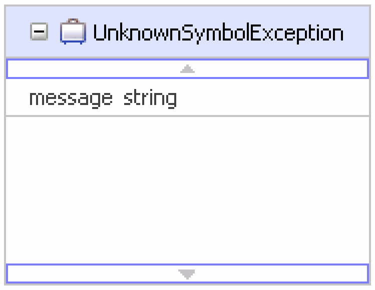
The Web service interface and business objects are shown below:

- The StockQuoteService provides these operations:
    - **getDelayedQuote** – accepts a stock symbol, and returns a cached stock quote, which expires in 10 minutes.
    - **getRealTimeQuote** – accepts a stock symbol, and returns an instance quote from Yahoo stock quote service.



	Name	Type
getDelayedQuote		
Input(s)	symbol	string
Output(s)	getDelayedQuoteReturn	float
Fault(s)	UnknownSymbolException	UnknownSymbolException
getRealTimeQuote		
Input(s)	symbol	string
Output(s)	getRealTimeQuoteReturn	float
Fault(s)	UnknownSymbolException	UnknownSymbolException

- In the event when an unknown symbol is received, an **UnknownSymbolException** business exception is returned as a WSDL fault



- A valid stock symbol must begin with a character from ***a*** through ***z*** or ***A*** through ***Z***, inclusive. Otherwise, a **BadSymbolException** is thrown by the operation. However, this exception is not defined in the Web service interface.

You will create a mediation flow that adds a custom mediation to each of the response flows in order to dump the Service Message Object (SMO) that is passed in so that each one may be inspected. In the case of the non-fault case, after printing out the SMO, the message will be passed back as the input response.

In the case of the modeled fault, the flow will pass the message back to the callout fault node. Finally, in the case of the unmodeled fault flow, the message will be silently stopped by way of a Stop primitive. Another option that might be used is to use the Event Emitter primitive rather than the Custom Mediation primitive used here.

Exercise instructions

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

Reference Variable	Windows Location	AIX®/UNIX® Location
<WID_HOME>	E:\WID602	
<LAB_FILES>	C:\LabFiles602\	/tmp/Labfiles602/
<TEMP>	C:\temp	/tmp

Windows users' note: When directory locations are passed as parameters to a Java™ program such as EJBdeploy or wsadmin, it is necessary to replace the backslashes with forward slashes to follow the Java convention. For example, C:\LabFiles602\ would be replaced by C:/LabFiles602/

Note that the previous table is relative to where you are running WebSphere Integration Developer. This table relates variables to where you are running the remote test environment:

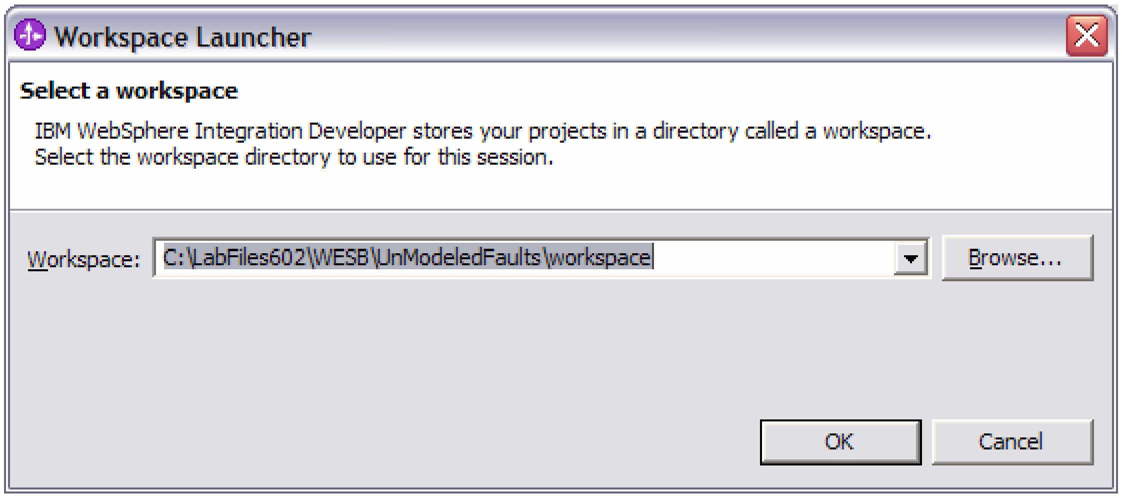
Reference Variable	**Example: Remote Windows test server location**	**Example: Remote z/OS test server location**	**Input your values for the remote location of the test server**
<SERVER_NAME>	server1	cl1sr01	
<WAS_HOME>	C:\Program Files\IBM\WebSphere\AppServer	/etc/cl1cell/AppServerNode1	
<HOSTNAME>	Localhost	mvsxxx.rtp.raleigh.ibm.com	
<BOOTSTRAP_PORT>	2809	2809	
<TELNET_PORT>	N/A	1023	
<PROFILE_NAME>	AppSrv01	default	
<PORT>	9080	9138	
<USERID>	N/A	cl1admin	
<PASSWORD>	N/A	fr1day	

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

Part 1: Set up development environment

In this section of the lab, all the projects that are part of the **WESB_UnModelledFaults_PI.zip** project interchange file are imported into a new workspace. There are three projects that are imported.

____ 1. Start WebSphere Integration Developer V6.0.2 with a workspace location of **<LAB_FILES>\WESB\UnModeledFaults\workspace** (that is, C:\LabFiles602\WESB\UnModeledFaults\workspace)

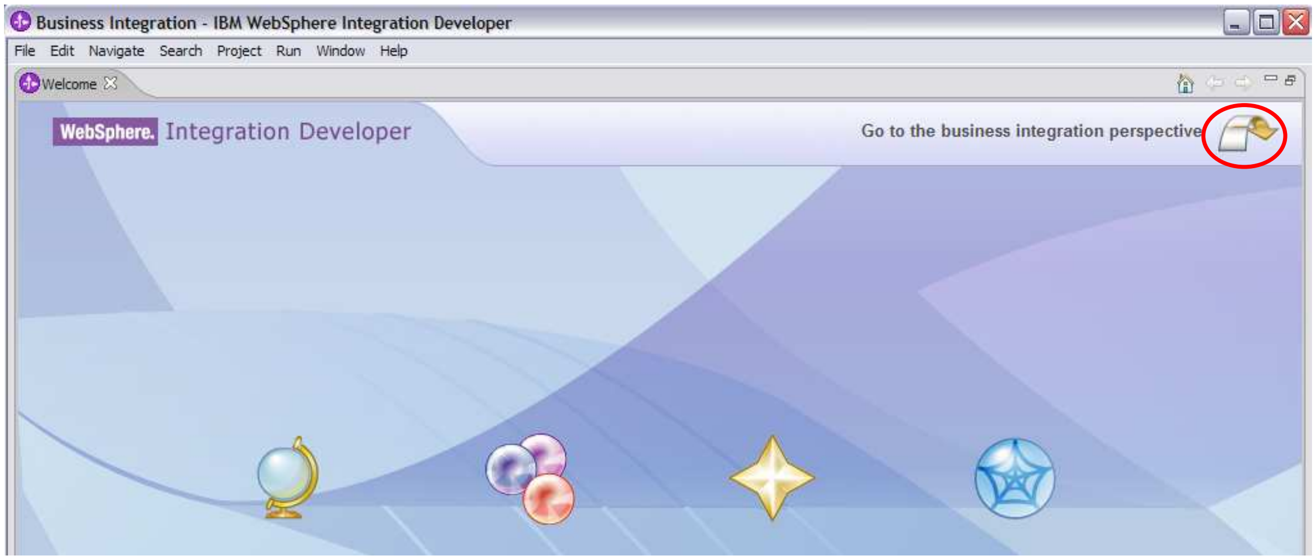


Select a workspace

IBM WebSphere Integration Developer stores your projects in a directory called a workspace.
Select the workspace directory to use for this session.

Workspace: C:\LabFiles602\WESB\UnModeledFaults\workspace

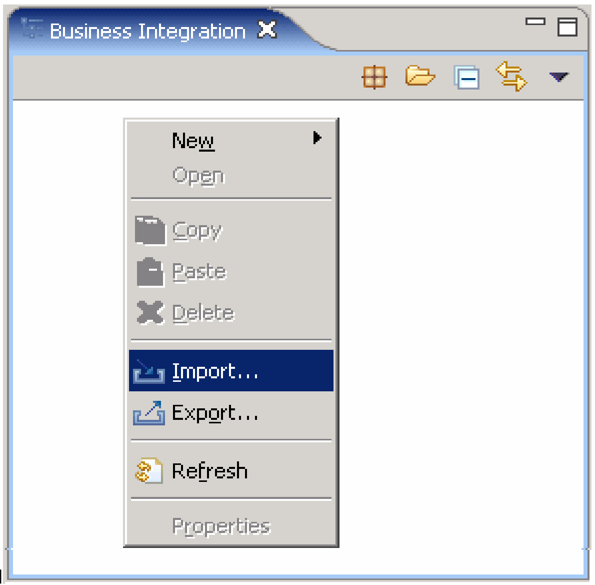
__ a. When WebSphere Integration Developer V6.0.2 opens, close the **Welcome page** by clicking on the curved arrow at the top right to "**Go to the business integration perspective**


Business Integration - IBM WebSphere Integration Developer
File
Edit
Navigate
Search
Project
Run
Window
Help
Welcome
WebSphere. Integration Developer
Go to the business integration perspective

__ b. Ensure you are in the **Business Integration** perspective.

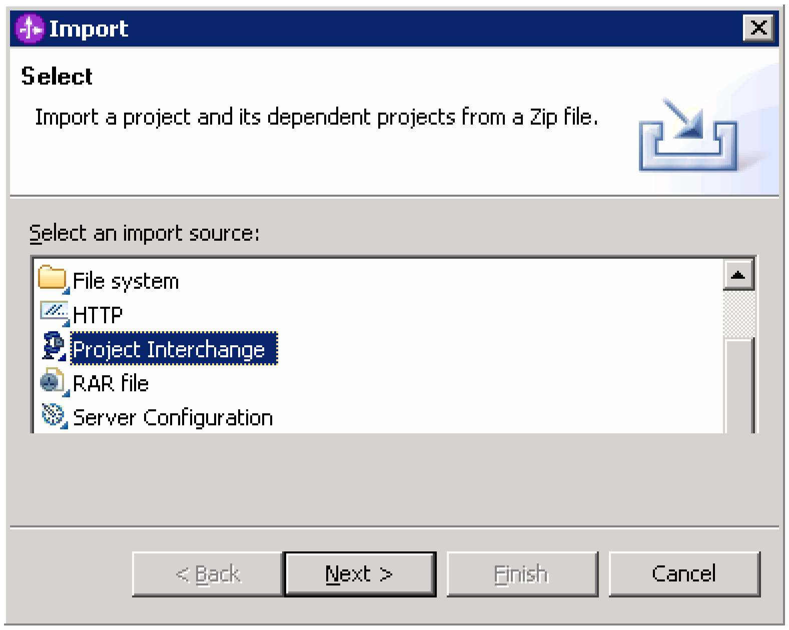
____ 2. Import the Project Interchange file, **WESB_UnModeledFaults_PI.zip**, into the development environment

__ a. Right-click inside **Business Integration View** (top left view in the Business Integration Perspective)



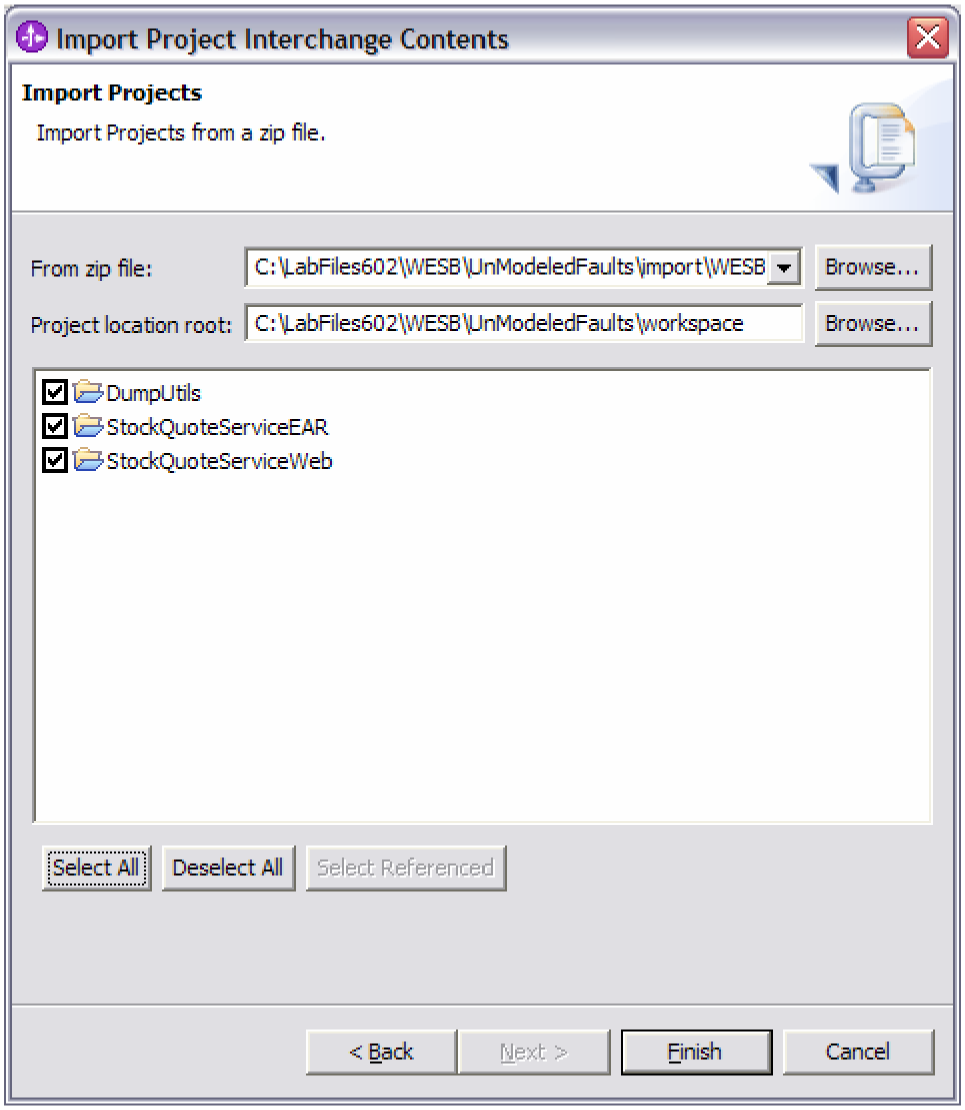
__ b. Select **Import** from the context menu

__ c. From the **Import** dialog, select **Project Interchange** from the list



__ d. Click **Next**

__ e. Click the **Browse** button for "**From zip file"** to navigate for the Project interchange file, **WESB_UnModeledFaults_PI.zip,** located in the **<LAB_FILES>\WESB\UnModeledFaults\import** directory



From zip file: C:\LabFiles602\WESB\UnModeledFaults\import\WESB

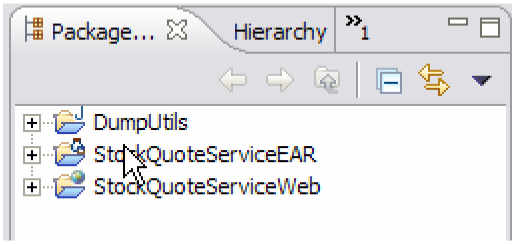
Project location root: C:\LabFiles602\WESB\UnModeledFaults\workspace

- [x] DumpUtils
- [x] StockQuoteServiceEAR
- [x] StockQuoteServiceWeb

__ f. Click the **Select All** button to ensure all projects listed are selected

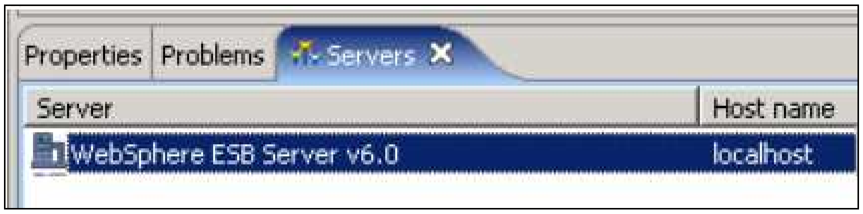
__ g. Click the **Finish** button (projects will be imported and auto-build will run)

__ h. Verify that DumpUtils, StockQuoteServiceEAR and stockQuoteServiceWeb have been imported by switching to the Java perspective (**Window > Open Perspective > Java**):



__ i. Switch back to **Business Integration perspective**

____ 3. Verify that the WebSphere ESB Server v6.0 is listed in the **Servers** view



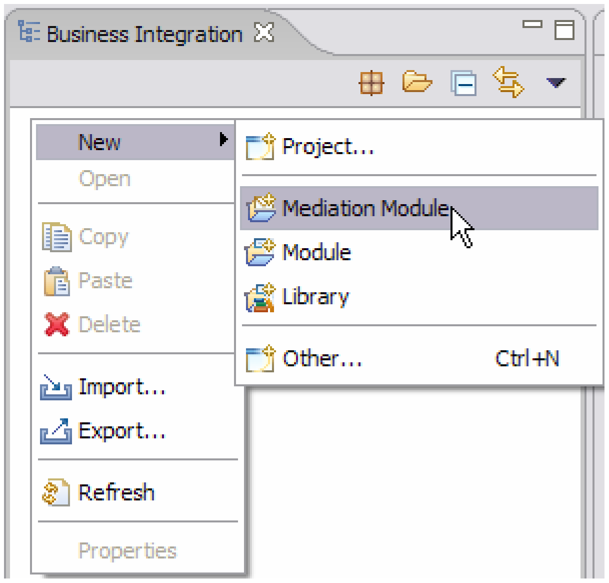
Server	Host name
WebSphere ESB Server v6.0	localhost

Part 2: Create mediation module

In this section of the lab, a new mediation module is created. There can only be one mediation module for each deployable project.

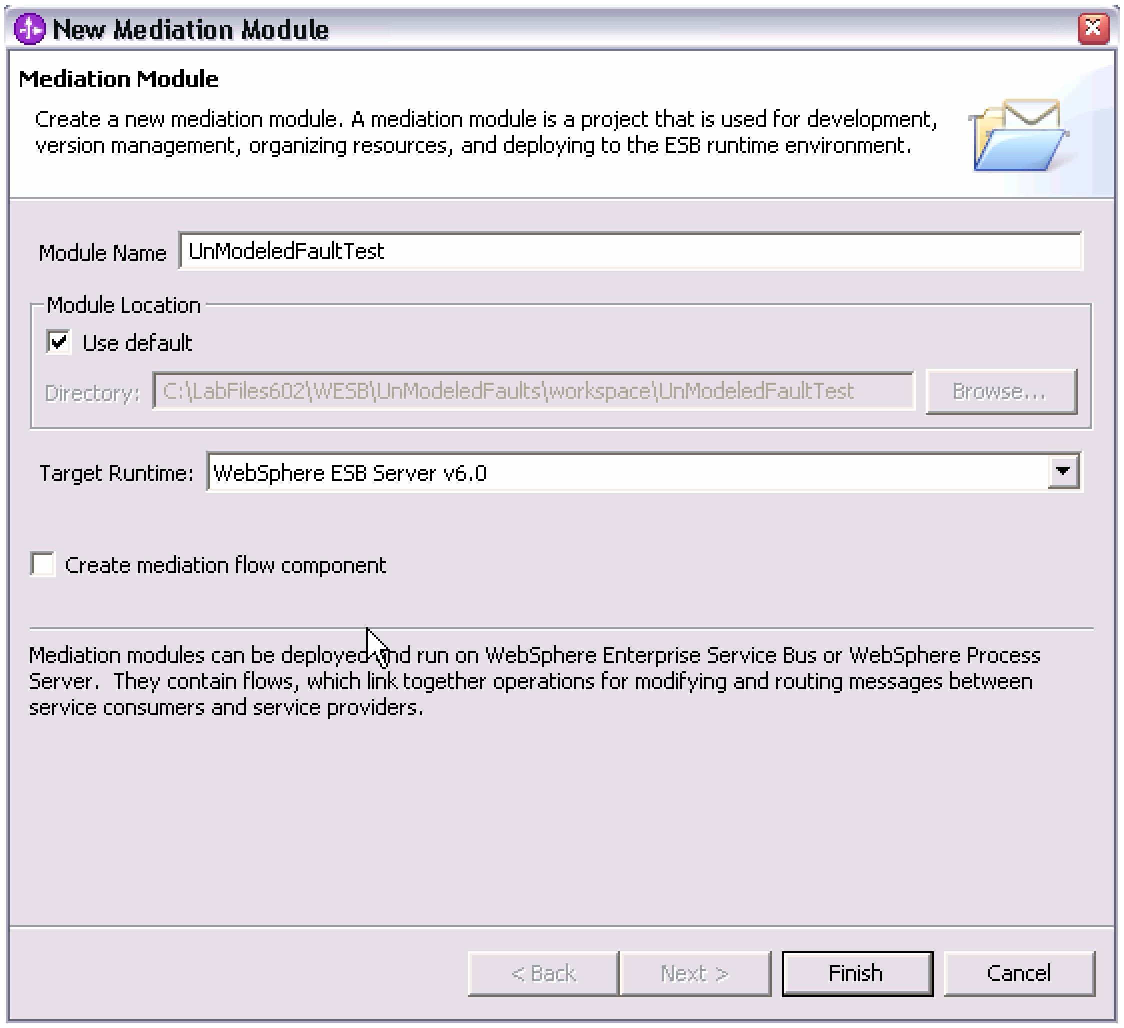
____ 1. To create the mediation module, complete these steps:

__ a. In the Business Integration view, right-click to see the context menu and select **New > Mediation Module**. The new Mediation Module window opens



__ b. In the New Mediation Module window, type the **Module Name** as **UnModeledFaultTest**

__ c. Verify that the target runtime is the WebSphere ESB Server v6.0 and **uncheck** the "**Create mediation flow component**" box.


New Mediation Module

Mediation Module

Create a new mediation module. A mediation module is a project that is used for development, version management, organizing resources, and deploying to the ESB runtime environment.

Module Name: UnModeledFaultTest

Module Location

[✓] Use default

Directory: C:\LabFiles602\WESB\UnModeledFaults\workspace\UnModeledFaultTest

Target Runtime: WebSphere ESB Server v6.0

[ ] Create mediation flow component

Mediation modules can be deployed and run on WebSphere Enterprise Service Bus or WebSphere Process Server. They contain flows, which link together operations for modifying and routing messages between service consumers and service providers.

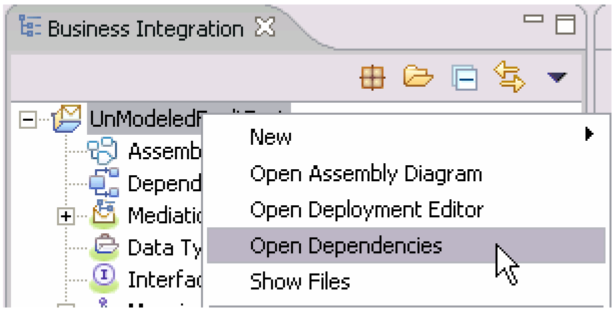
__ d. Click **Finish**

__ e. A mediation module called **UnModeledFaultTest** is created. The mediation flow component will be created later.

____ 2. The Dump Utility that will be used needs to be added as a dependency to the mediation module.

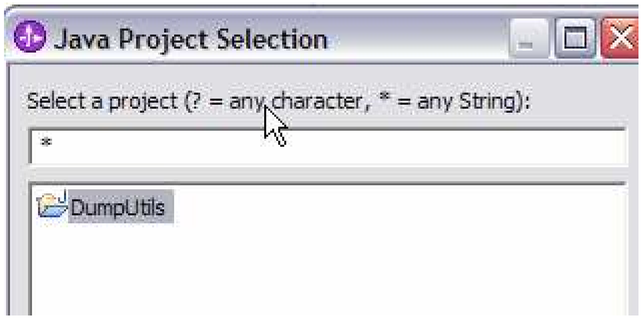
__ a. In the Business Integration view, right click on the **UnModeledFaultTest** module you just created to access the pop-up menu.

__ b. Select **Open Dependencies.**



__ c. In the Dependencies Editor, expand the Java section and click the **Add…** button.

__ d. In the Java Project Selection dialog, select **DumpUtils**



__ e. Click **OK**.

__ f. Press **Ctrl+S** to save the dependencies for this module.

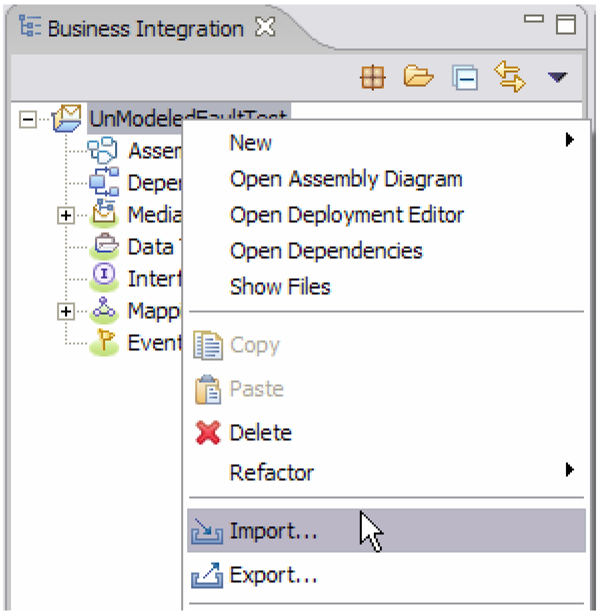
__ g. Close the Dependencies editor.

Part 3: Import Web service interface

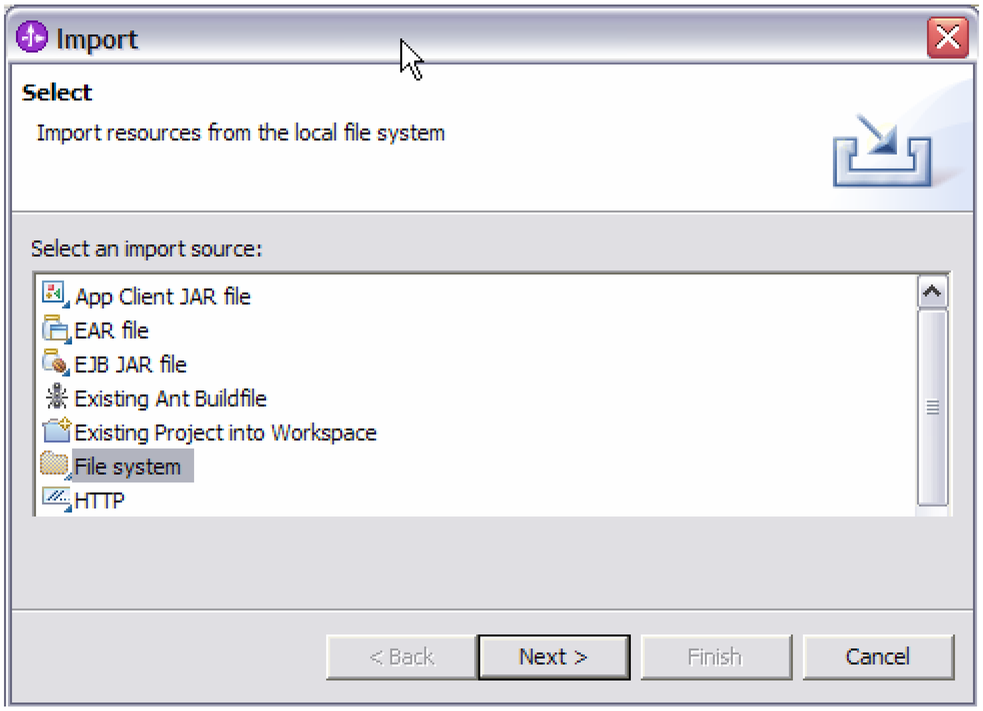
In this section of the lab, the interface to the StockQuoteService is imported into the UnModeledFaultTest Mediation Module Project. It is in the form of a .wsdl file that defines the service that will be used along with the operations it exposes.

____ 1. To import the Web Service Interface, complete these steps:

__ a. Right-click the **UnModeledFaultTest** folder in the Business Integration view, select **Import…**

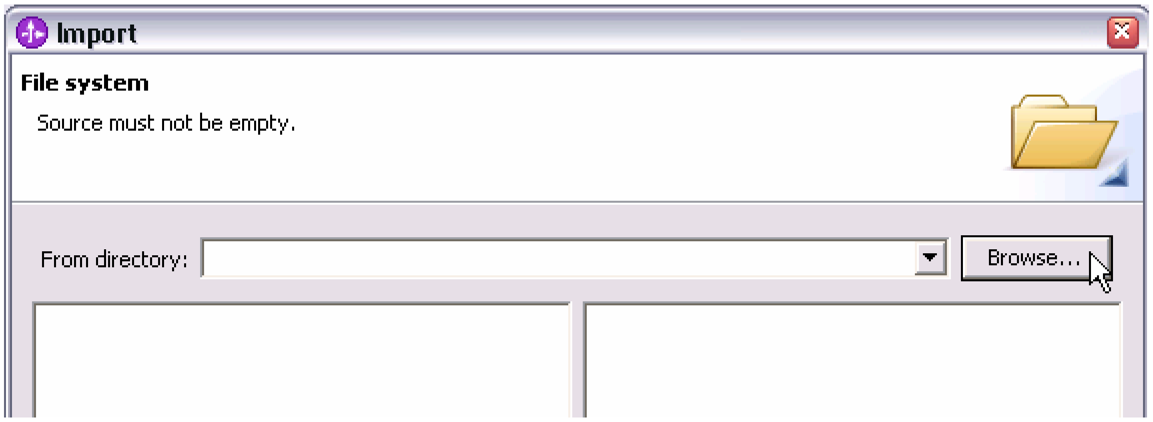


__ b. Select **File System**

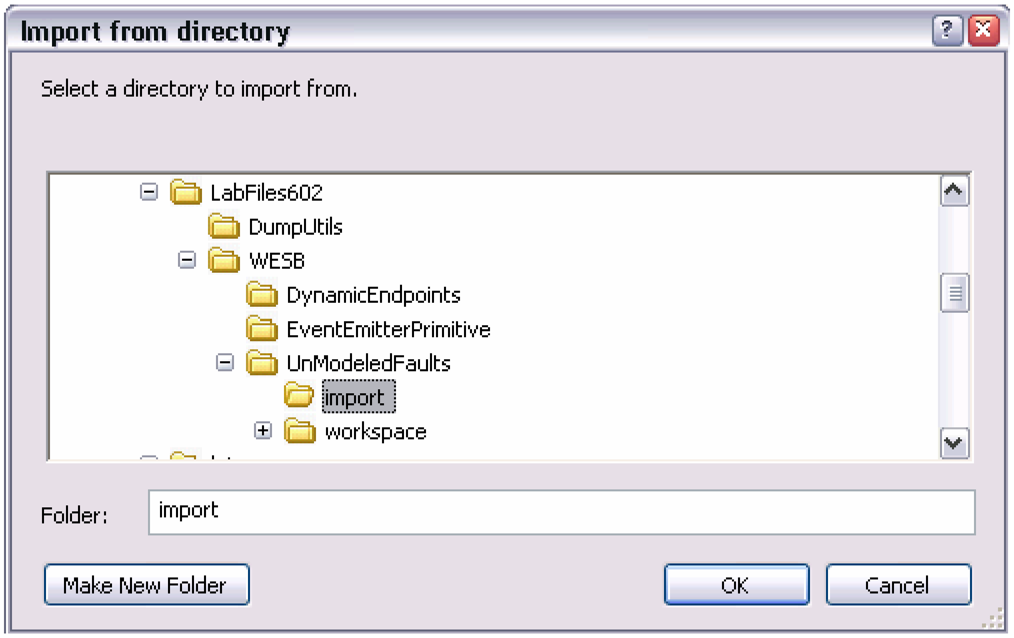


__ c. Click **Next**

__ d. On the **File system** window, on the **From Directory** field, click **Browse**

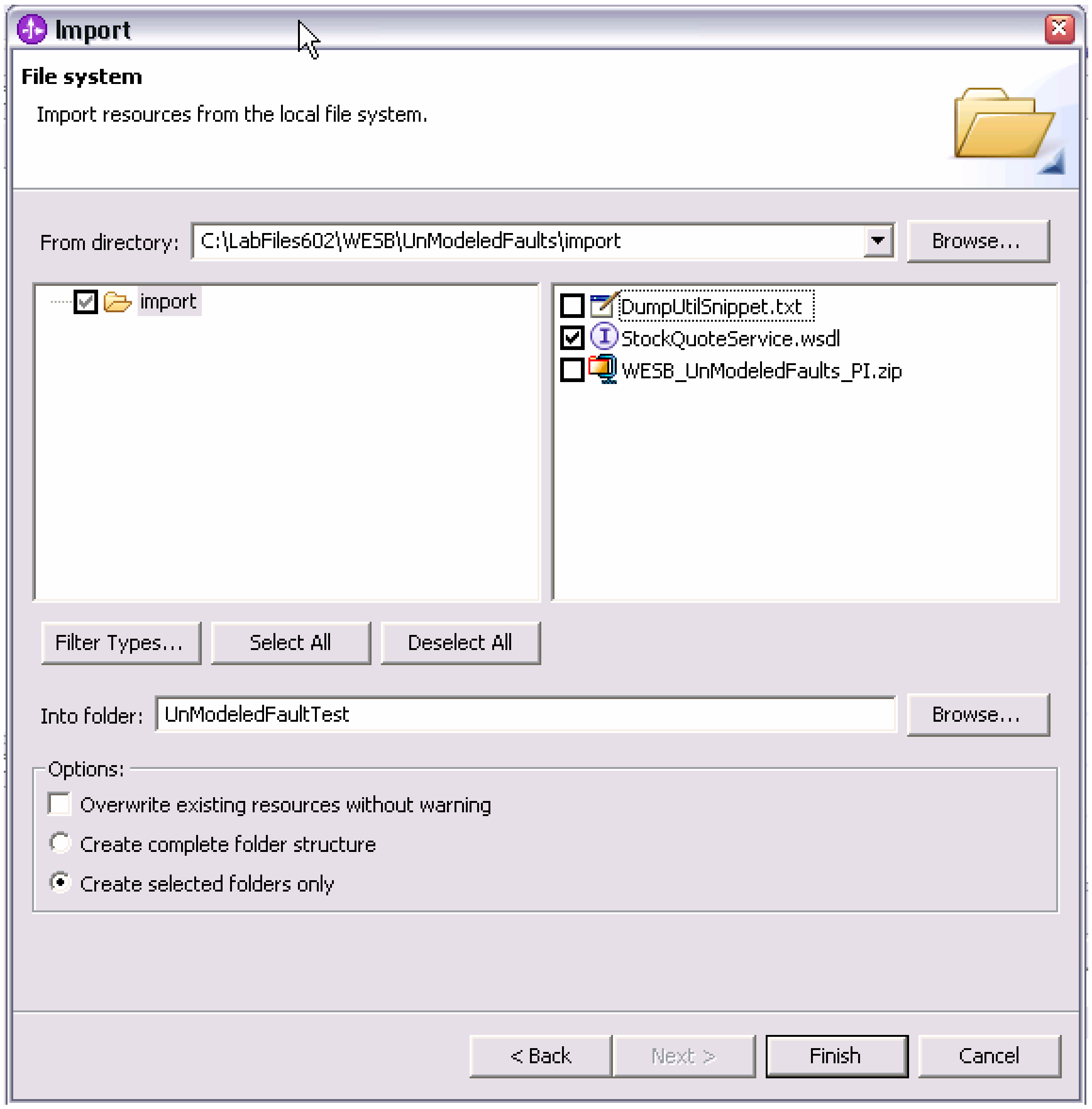


__ e. Select**<LAB_FILES>\WESB\UnModeledFaults\import**



__ f. Click **OK**

__ g. Click the check box for **StockQuoteService.wsdl** and verify the Into Folder contains **UnModeledFaultTest**.


Import

File system

Import resources from the local file system.

From directory: C:\LabFiles602\WESB\UnModeledFaults\import

[✓] import

[ ] DumpUtilSnippet.txt
[✓] StockQuoteService.wsdl
[ ] WESB_UnModeledFaults_PI.zip

Into folder: UnModeledFaultTest

Options:
[ ] Overwrite existing resources without warning
( ) Create complete folder structure
(●) Create selected folders only

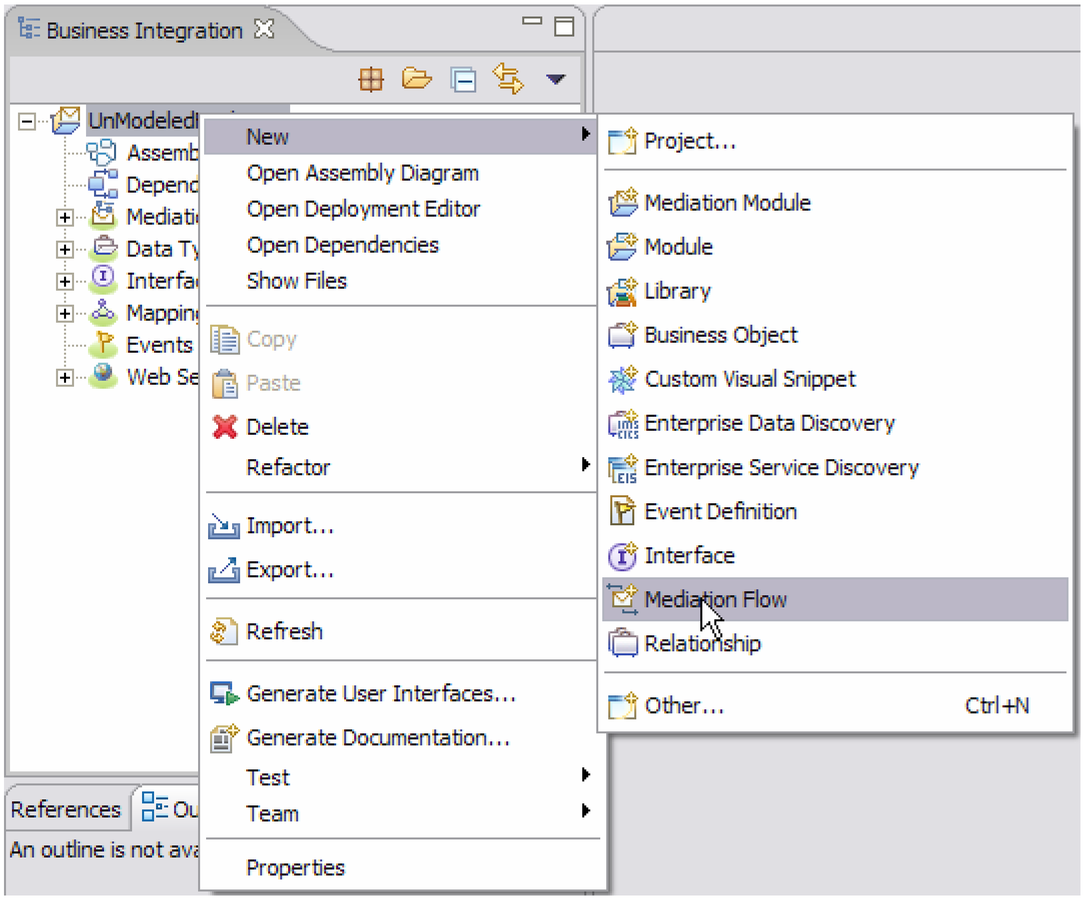
__ h. Click **Finish**.

Part 4: Create the mediation flow

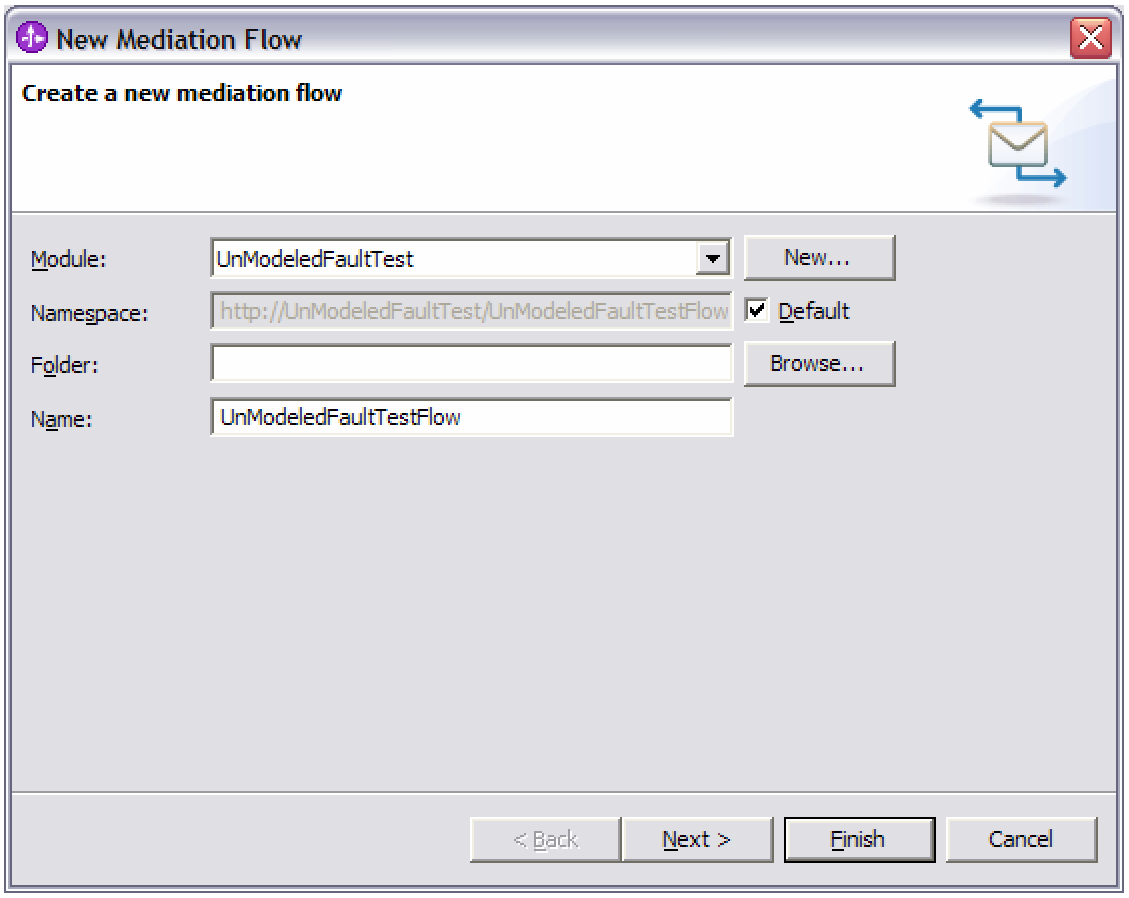
In this section of the lab, the Mediation Flow will be created. Once the operations are wired together, some mediation primitives will be added for the new unmodeled fault flow in the mediation flow component. Custom Mediations will be added to the various flows to dump the messages and inspect their contents.

____ 1. To create the mediation flow, complete these steps:

__ a. Right-click the **UnModeledFaultTest** folder in the Business Integration view, select **New > Mediation Flow**



__ b. In the New Mediation Flow window, type the **Name** as **UnModeledFaultTestFlow**



Create a new mediation flow

Module: UnModeledFaultTest

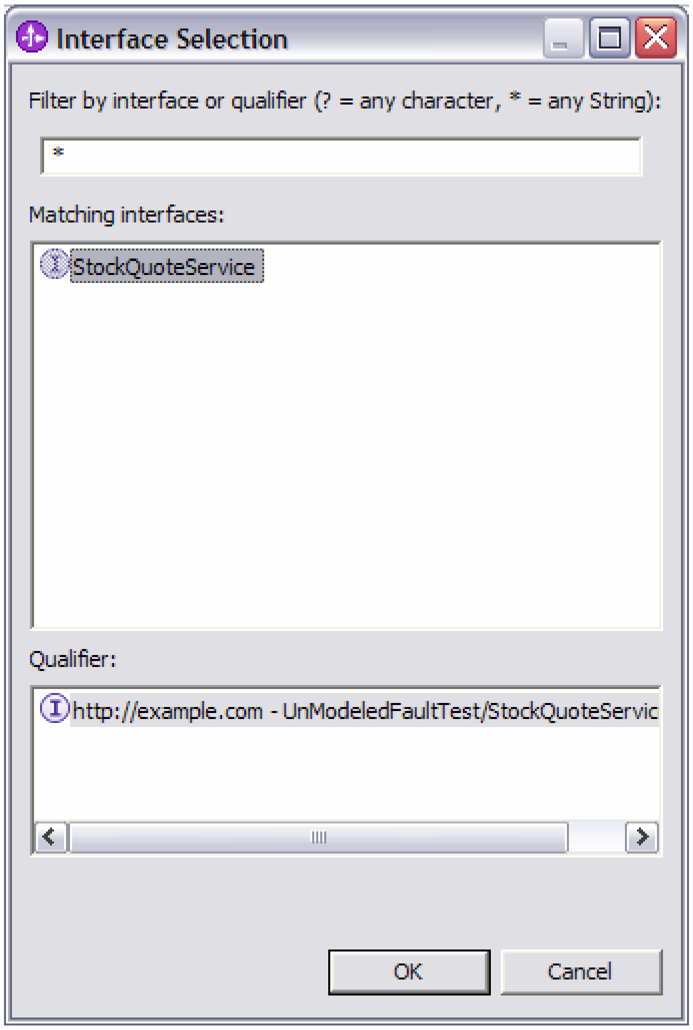
Namespace: http://UnModeledFaultTest/UnModeledFaultTestFlow [✓ Default]

Folder:

Name: UnModeledFaultTestFlow

__ c. Click **Next**

__ d. On the Creating a New Interface window, click **Add** for **both *Source*** and ***Target*** Interfaces and select **StockQuoteService** for each.


Interface Selection

Filter by interface or qualifier (? = any character, * = any String):

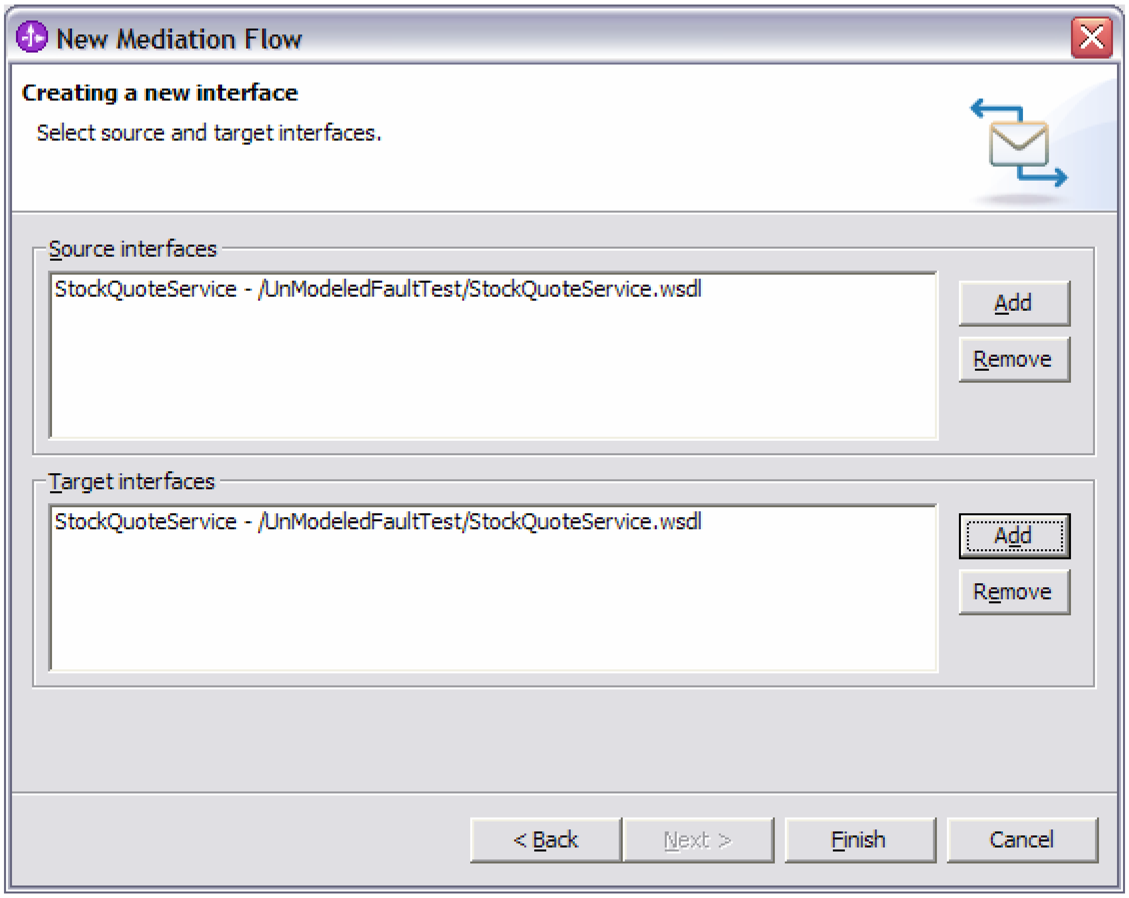
*

Matching interfaces:

StockQuoteService

Qualifier:

http://example.com - UnModeledFaultTest/StockQuoteServic


New Mediation Flow

Creating a new interface

Select source and target interfaces.

Source interfaces

StockQuoteService - /UnModeledFaultTest/StockQuoteService.wsdl

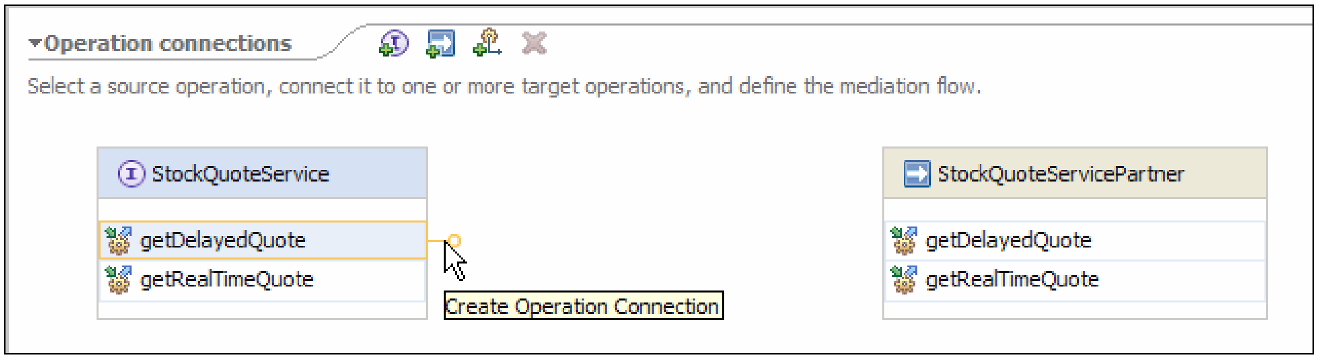
Target interfaces

StockQuoteService - /UnModeledFaultTest/StockQuoteService.wsdl

__ e. Click **Finish**

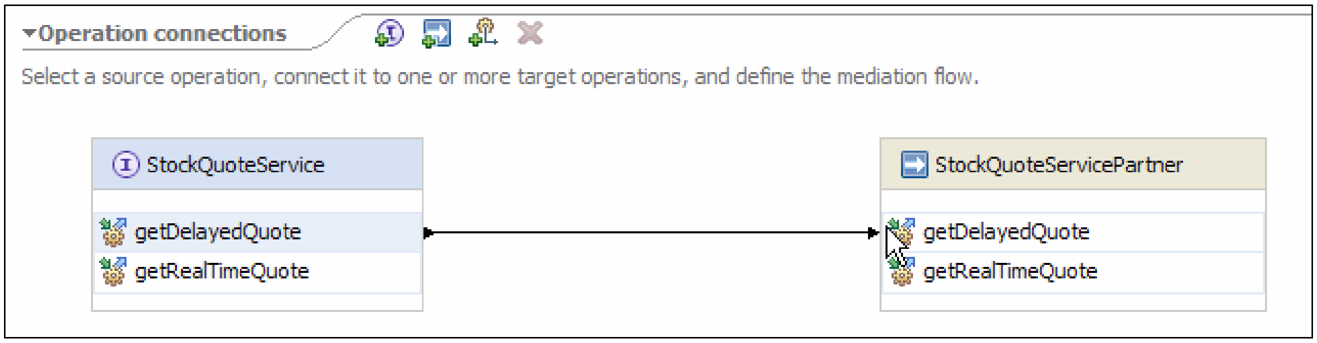
____ 2. The Mediation Flow Module was created and the Mediation Flow Editor opened. To connect the operations, complete these steps:

__ a. Hover next to the **getDelayedQuote** under **StockQuoteService** in the Operation connections window and grab the yellow wire


Operation connections

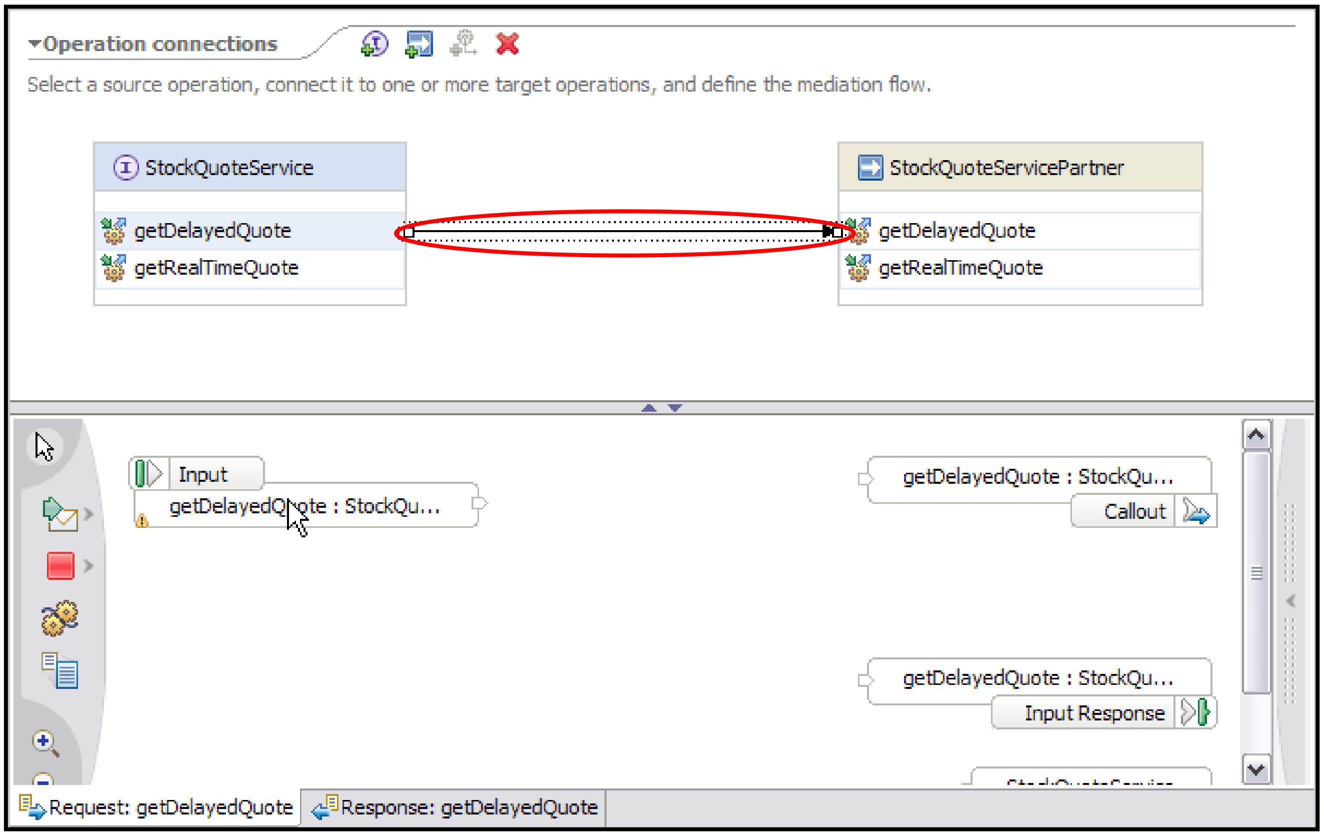
Select a source operation, connect it to one or more target operations, and define the mediation flow.

__ b. Drag the wire to the **getDelayedQuote** under **StockQuoteServicePartner** on the right-hand side

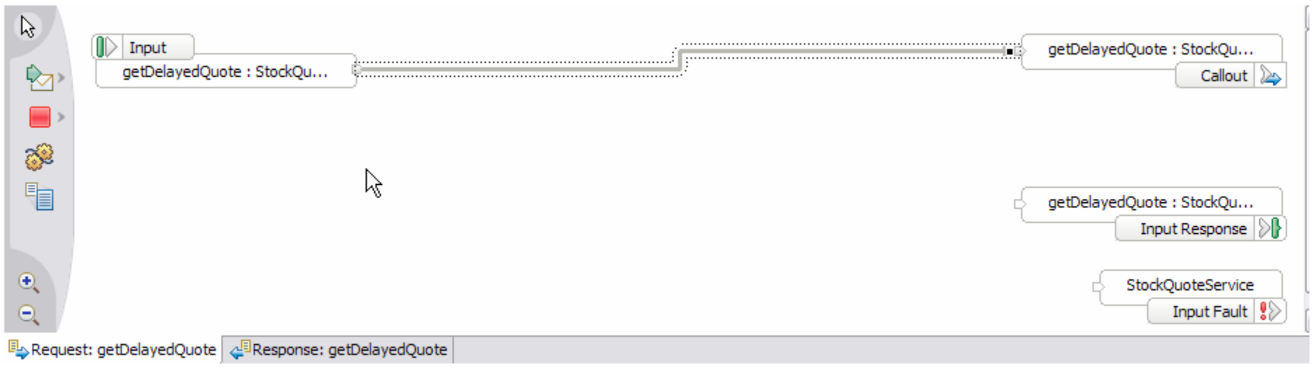


____ 3. Add a custom mediation primitive to the unmodeled fault Response Flow

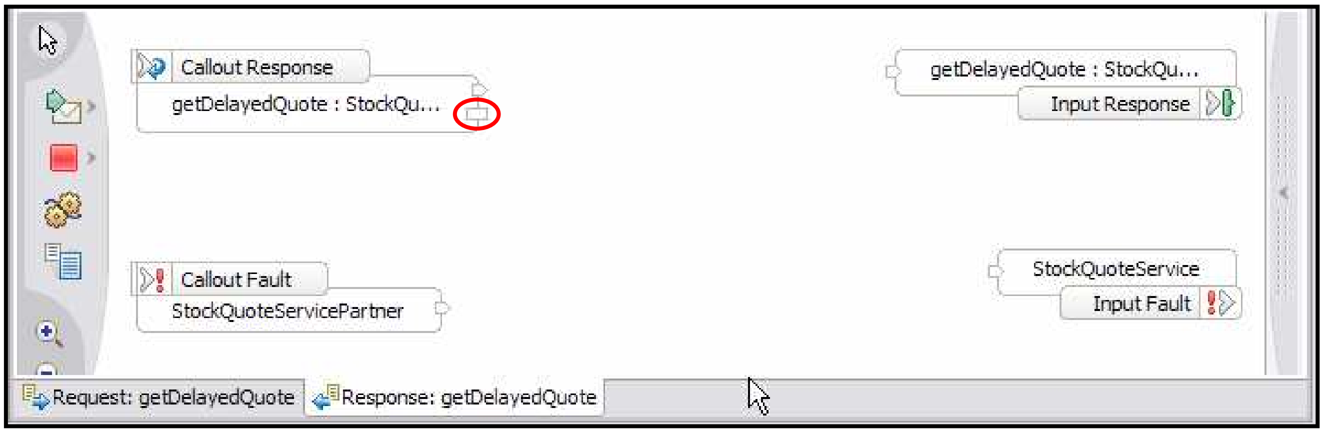
__ a. Click on the **black line** for the **getDelayedQuote** section of the **StockQuoteService** Interface to make the Mediation Flow diagram populate with information (if not already up).


Operation connections
Select a source operation, connect it to one or more target operations, and define the mediation flow.
StockQuoteService
getDelayedQuote
getRealTimeQuote
StockQuoteServicePartner
getDelayedQuote
getRealTimeQuote
Input
getDelayedQuote : StockQu...
getDelayedQuote : StockQu...
Callout
getDelayedQuote : StockQu...
Input Response
Request: getDelayedQuote
Response: getDelayedQuote

__ b. Connect the **getDelayedQuote Input** to the **getDelayedQuote Callout**. Click on the output terminal of the Input Request for getDelayedQuote and drag a wire to the input terminal of the getDelayedQuote Callout

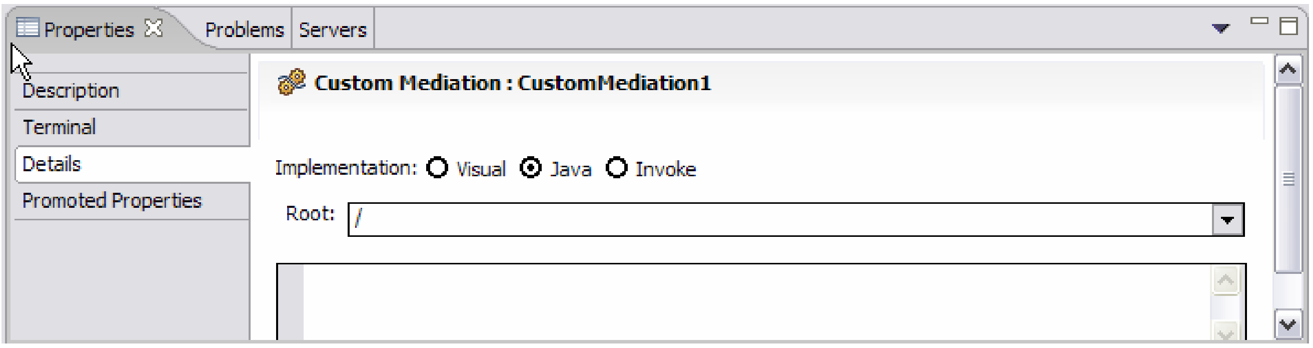

Input
getDelayedQuote : StockQu...
getDelayedQuote : StockQu...
Callout
getDelayedQuote : StockQu...
Input Response
StockQuoteService
Input Fault
Request: getDelayedQuote
Response: getDelayedQuote

__ c. Click on the **Response Tab** in the Mediation Flow diagram to see details of the Response flows. Notice the new terminal that is used for the unmodeled fault, highlighted below with the red oval.



__ d. Click the **Custom Mediation** icon ( ) in the mediation flow palette (left-hand side) and then click to the right of the unmodeled fault terminal. The **CustomMediation1** will appear.

__ e. Click on the **Details** tab in the properties view (bottom window) of the **CustomMediation1**. If not already showing in the bottom pane, right-click on **CustomMediation1** and select **Show in Properties**.



Custom Mediation : CustomMediation1

Implementation: Visual [✓ Java] Invoke

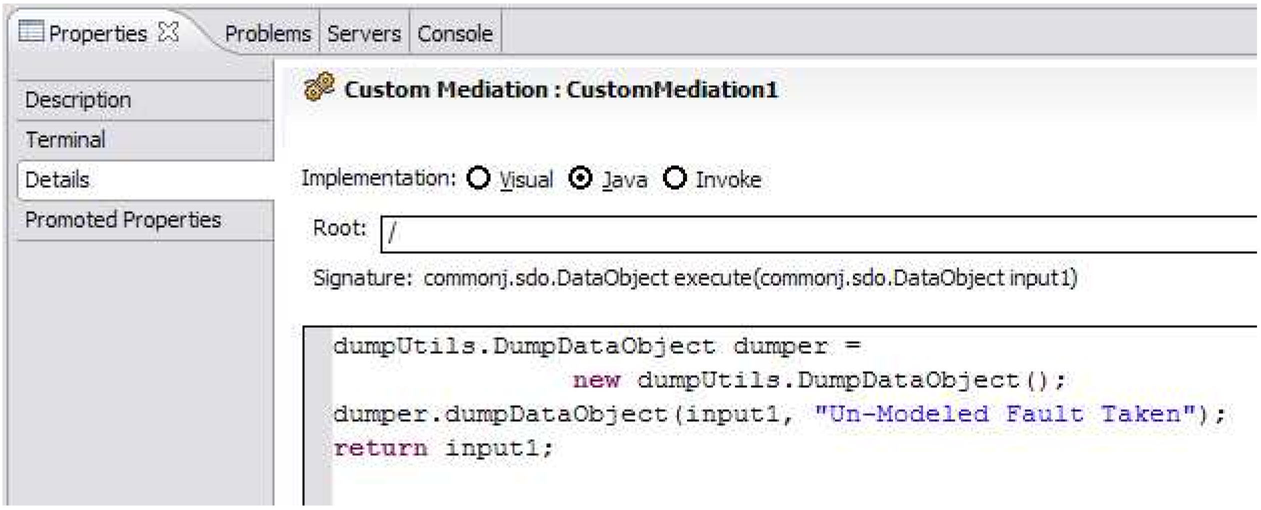
Root: /

__ f. A DumpUtility was included in the Project Interchange File first imported. Call that from the **CustomMediation1** to dump the contents of the Service Message Object (SMO). To do this, add this code to the **Details** section of **CustomMediation1**:

```
dumpUtils.DumpDataObject dumper =
                new dumpUtils.DumpDataObject();
dumper.dumpDataObject(input1, "Un-Modeled Fault Taken");
```

```
return input1;
```

A generic form of this code that can be cut and pasted and edited is included in this file: **<LAB_FILES>\WESB\UnModeledFaults\import\DumpUtilSnippet.txt**



Custom Mediation : CustomMediation1

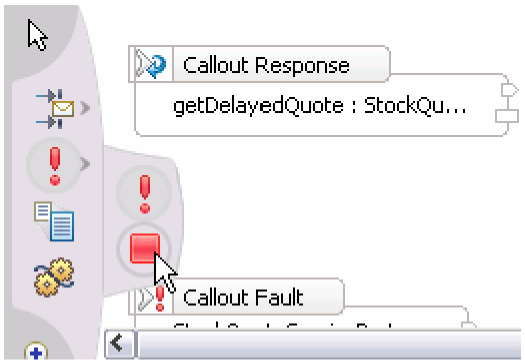
Implementation: Visual [✓ Java] Invoke

Root: /

Signature: commonj.sdo.DataObject execute(commonj.sdo.DataObject input1)

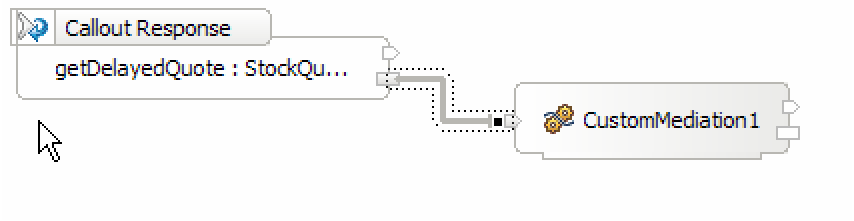
```
dumpUtils.DumpDataObject dumper =
                new dumpUtils.DumpDataObject();
dumper.dumpDataObject(input1, "Un-Modeled Fault Taken");
return input1;
```

__ g. In the middle window, click the **Stop** mediation icon ( ) in the mediation flow palette (left-hand side). You may have to click on the **Fail** mediation icon ( ) to find it as shown in the graphic below.

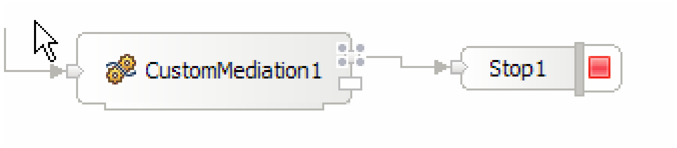


__ h. Click to the right of **CustomMediation1**. The **Stop1** primitive will appear.

__ i. Click on the **fail terminal** of the **Callout Response** for **getDelayedQuote** and drag a wire to the **input terminal** of the **CustomMediation1** primitive.



__ j. Wire the **output terminal** of **CustomMediation1** to the **input terminal** of the **Stop1** primitive

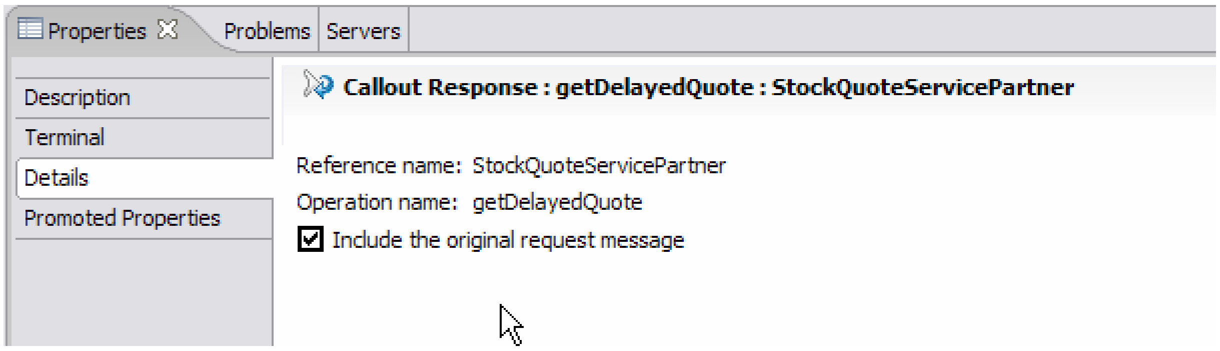


____ 4. Specify that the original request message should be included when an unmodeled fault is taken.

__ a. Click on the **getDelayedQuote Callout Response**.

__ b. Click on the **Details** tab in the properties view for **getDelayedQuote**. If not already showing in the bottom pane, right-click on **getDelayedQuote Callout Response** and select **Show in Properties**

__ c. Click the '**Include the original request Message**' box to indicate the complete message should be propagated to the fail terminal in the event of a failure

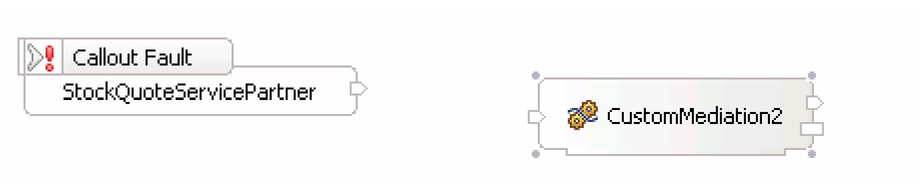


Callout Response : getDelayedQuote : StockQuoteServicePartner

Reference name: StockQuoteServicePartner
Operation name: getDelayedQuote
[✓] Include the original request message

____ 5. Add a custom mediation primitive to the modeled fault Response Flow

__ a. Still in the Response tab, click the **Custom Mediation** icon ( ) in the mediation flow palette (left-hand side) and then click to the right of the **Callout Fault** terminal (the modeled fault). The **CustomMediation2** will appear.



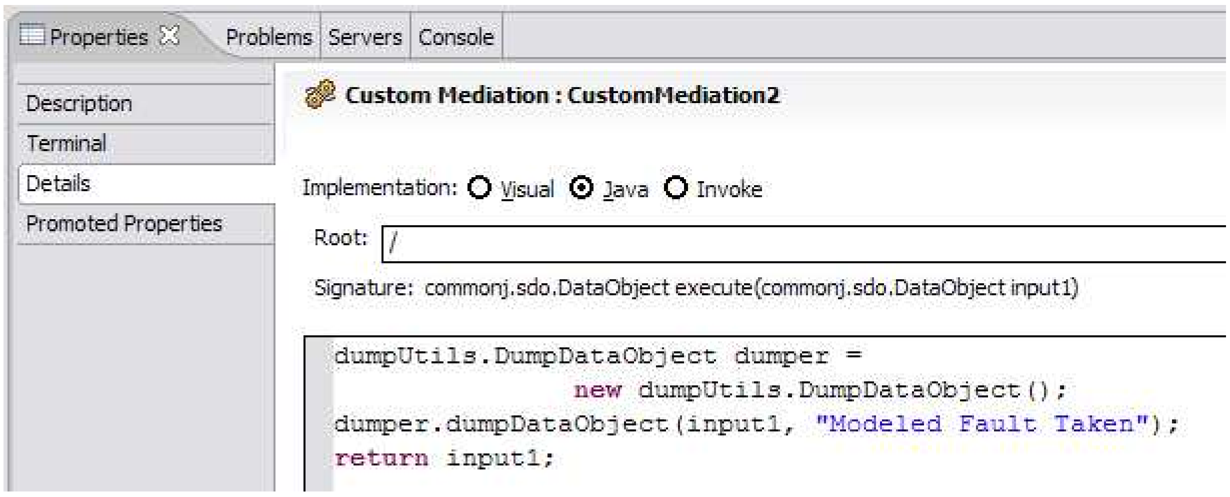
__ b. Click on the **Details** tab in the properties view **CustomMediation2**. If not already showing in the bottom pane, right-click on **CustomMediation2** and select **Show in Properties**.

__ c. Call the Dump Utility from the **CustomMediation2** to dump the contents of the Service Message Object (SMO). To do this, add this code to the **Details** section of the Custom Mediation:

```
dumpUtils.DumpDataObject dumper =
               new dumpUtils.DumpDataObject();
dumper.dumpDataObject(input1, "Modeled Fault Taken");
return input1;
```

A generic form of this code that can be cut and pasted and edited is included in this file:

<LAB_FILES>\WESB\UnModeledFaults\import\DumpUtilSnippet.txt



Custom Mediation : CustomMediation2

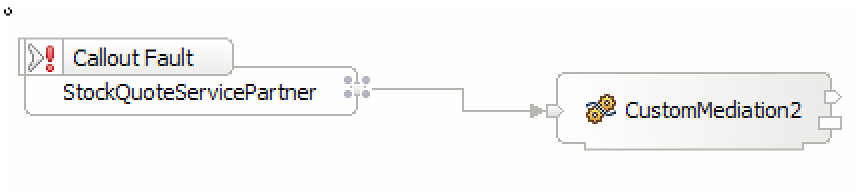
Implementation: Visual [✓ Java] Invoke

Root: /

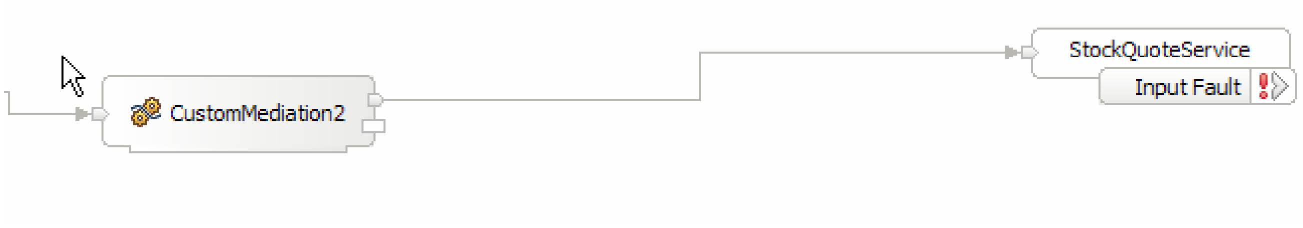
Signature: commonj.sdo.DataObject execute(commonj.sdo.DataObject input1)

```
dumpUtils.DumpDataObject dumper =
                new dumpUtils.DumpDataObject();
dumper.dumpDataObject(input1, "Modeled Fault Taken");
return input1;
```

__ d. Click on the **output terminal** of the **Callout Fault** of **StockQuoteServicePartner** and drag a wire to the **input terminal** of the **CustomMediation2** primitive

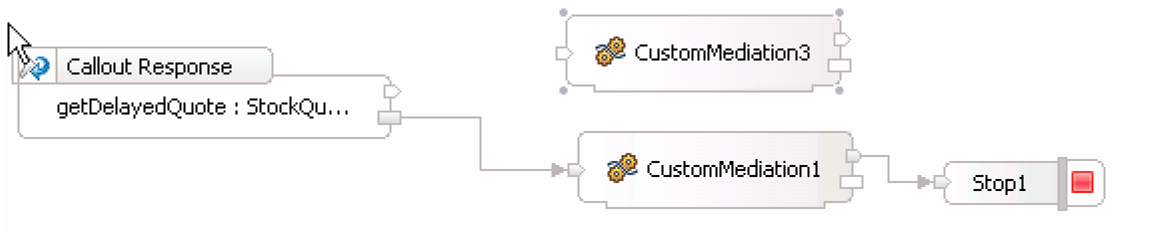


__ e. Click on the **output terminal** of the **CustomMediation2** primitive and drag a wire to the **input terminal** of the **StockQuoteService Input Fault**



____ 6. Add a custom mediation primitive to the non-fault Response Flow

__ a. Still in the Response tab, click the **Custom Mediation** icon ( ) in the mediation flow palette (left-hand side) and then click to the right of the **Callout Response output terminal**. The **CustomMediation3** will appear.



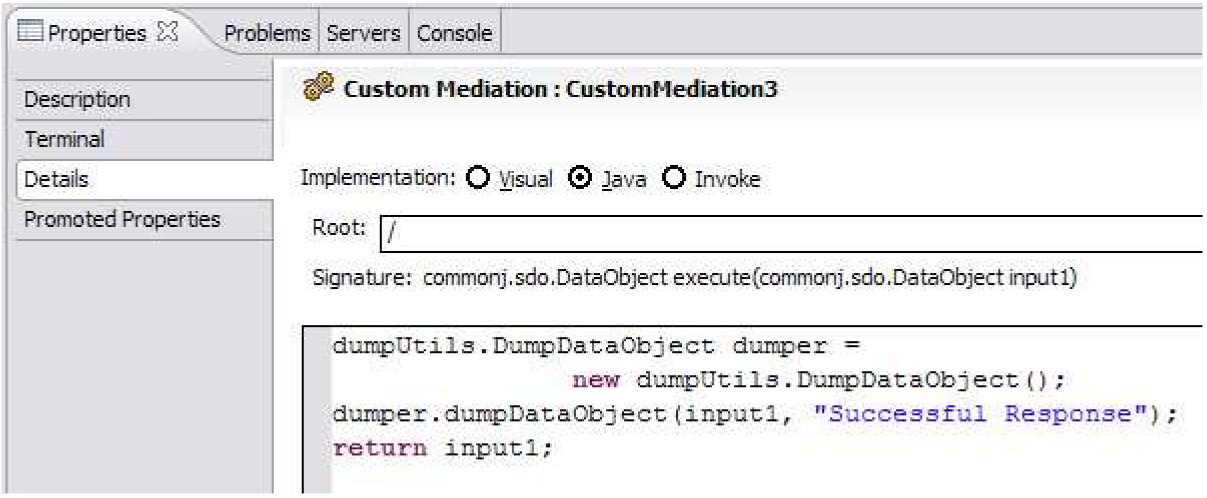
__ b. Click on the **Details** tab in the properties view **CustomMediation3**. If not already showing in the bottom pane, right-click on **CustomMediation3** and select **Show in Properties**.

__ c. Call the Dump Utility from the **CustomMediation3** to dump the contents of the Service Message Object (SMO). To do this, add this code to the **Details** section of the Custom Mediation:

```
dumpUtils.DumpDataObject dumper =
               new dumpUtils.DumpDataObject();
dumper.dumpDataObject(input1, "Successful Response");
return input1;
```

A generic form of this code that can be cut and pasted and edited is included in this file:

<LAB_FILES>\WESB\UnModeledFaults\import\DumpUtilSnippet.txt



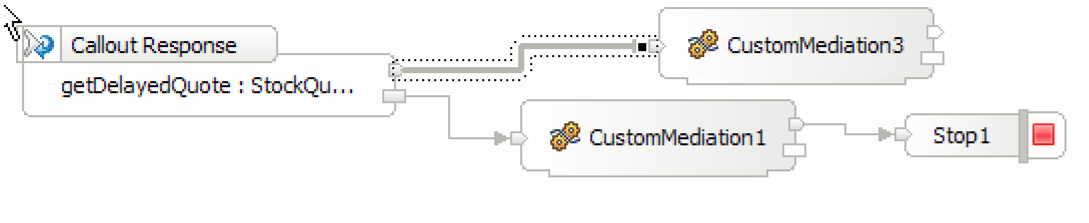
Implementation: Visual [✓ Java] Invoke

Root: /

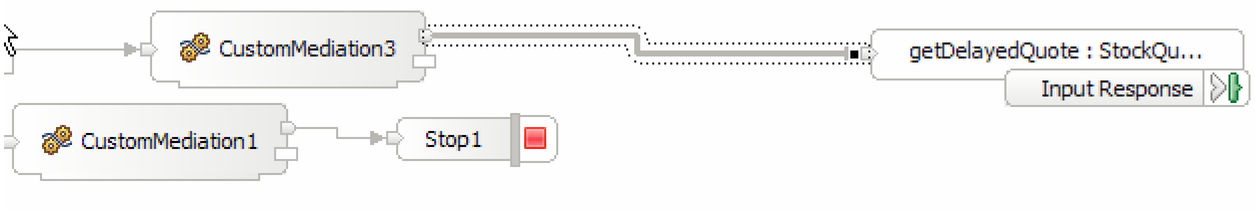
Signature: commonj.sdo.DataObject execute(commonj.sdo.DataObject input1)

```
dumpUtils.DumpDataObject dumper =
                new dumpUtils.DumpDataObject();
dumper.dumpDataObject(input1, "Successful Response");
return input1;
```

__ d. Click on the **output terminal** of the **Callout Response** of **getDelayedQuote** and drag a wire to the **input terminal** of the **CustomMediation3** primitive



__ e. Click on the **output terminal** of the **CustomMediation3** primitive and drag a wire to the **input terminal** of the **getDelayedQuote Input Response**



__ f. Save all work by **File > Save All** or **Crtl + Shift + S**

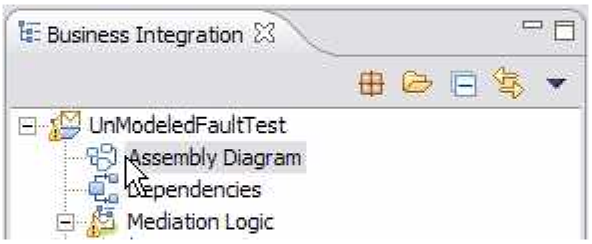
Part 5: Assemble the module

In this section of the lab, you will assemble the unModeledFaultTest Module, wiring an export to targets and define an SCA interface for the business process. As part of the assembly process, the appropriate deployment code will also be generated in preparation for running the business process as a service component on WebSphere Process Server

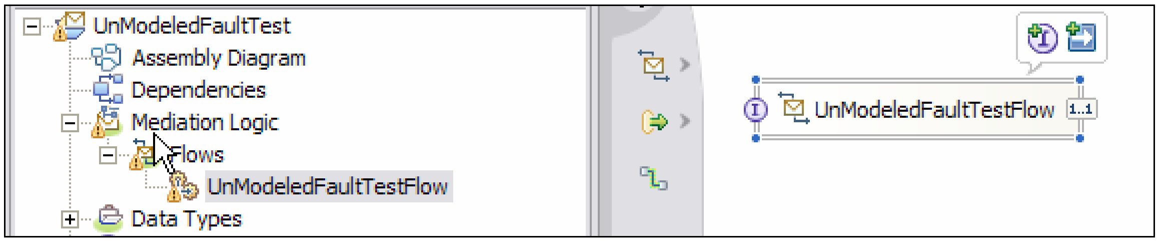
____ 1. The Mediation Flow Component needs to be added to the assembly diagram and connected to the Web Service it calls.

__ a. Expand the **UnModeledFaultTest** module by clicking on the '+'.

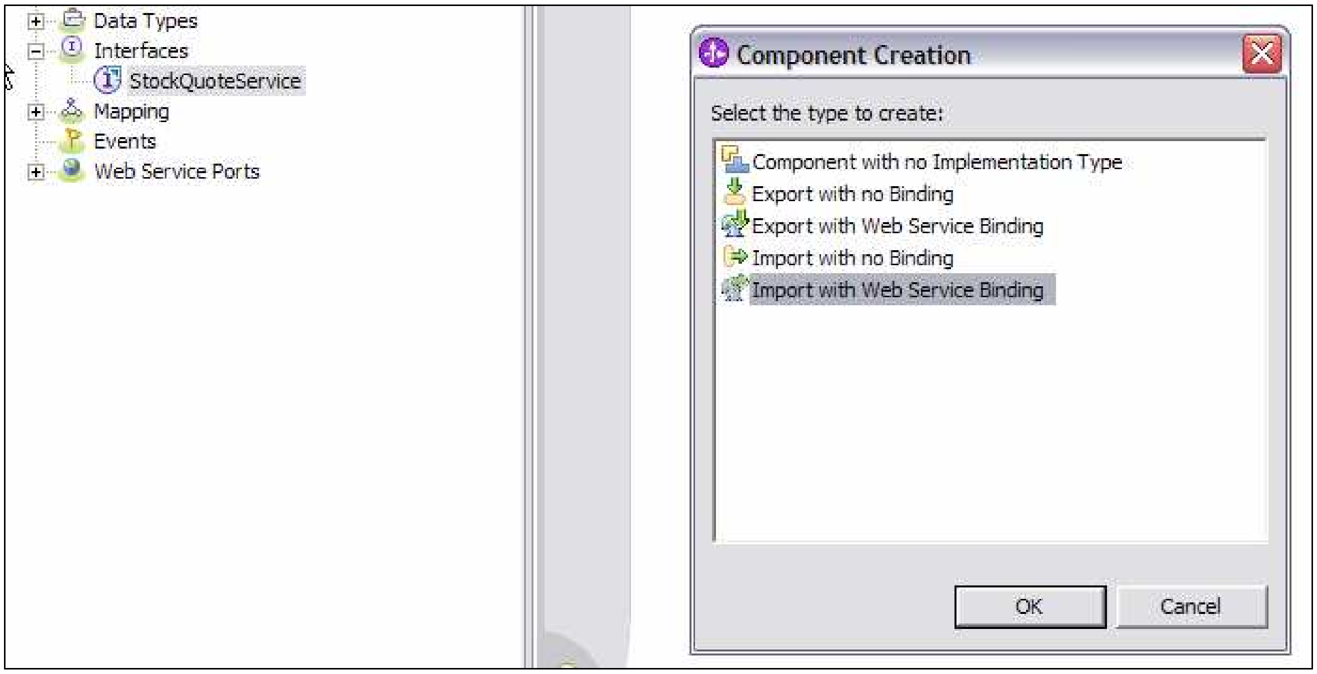
__ b. Double-click on the Assembly Diagram option directly underneath the **UnModeledFaultTest** Module to open the Assembly editor



__ c. Drag the **UnModeledFaultTestFlow** component on to the canvas



__ d. Drag the **StockQuoteService** onto the canvas and select **Import with Web Service Binding**.



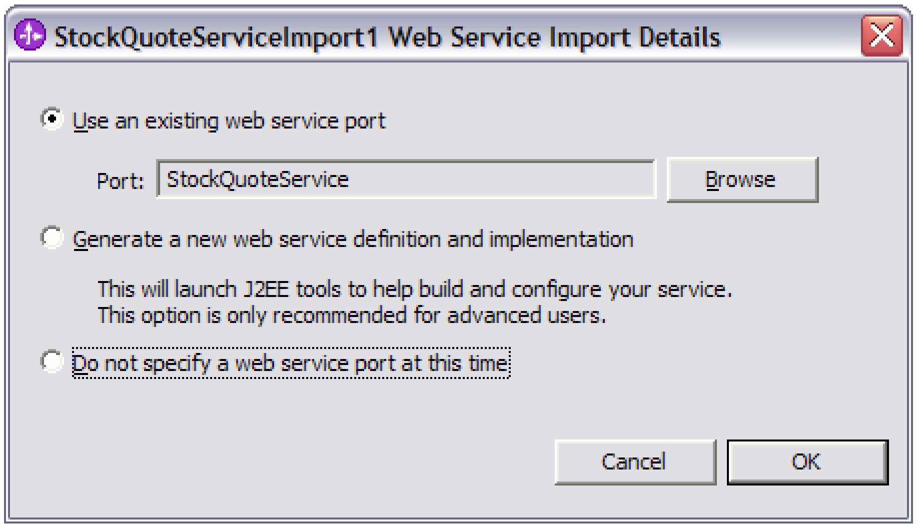
Component Creation

Select the type to create:

- Component with no Implementation Type
- Export with no Binding
- Export with Web Service Binding
- Import with no Binding
- Import with Web Service Binding

__ e. Click **OK**

__ f. Select to Use an existing Web service port. Browse to select **StockQuoteService**



StockQuoteServiceImport1 Web Service Import Details

(●) Use an existing web service port

Port: StockQuoteService

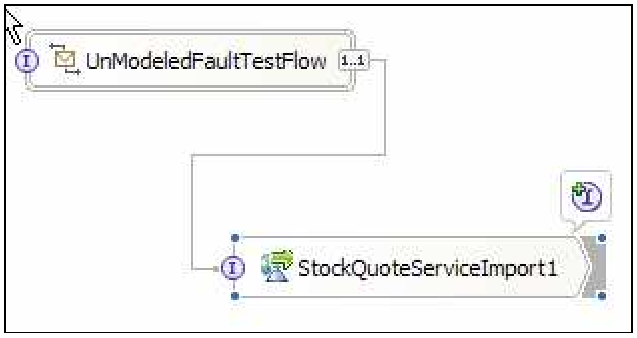
( ) Generate a new web service definition and implementation

This will launch J2EE tools to help build and configure your service.
This option is only recommended for advanced users.

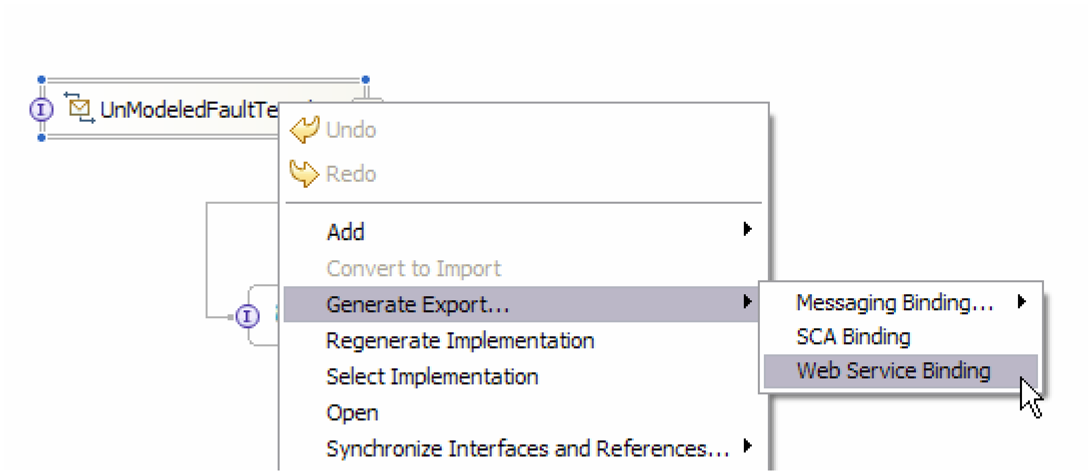
( ) Do not specify a web service port at this time

__ g. Click **OK**

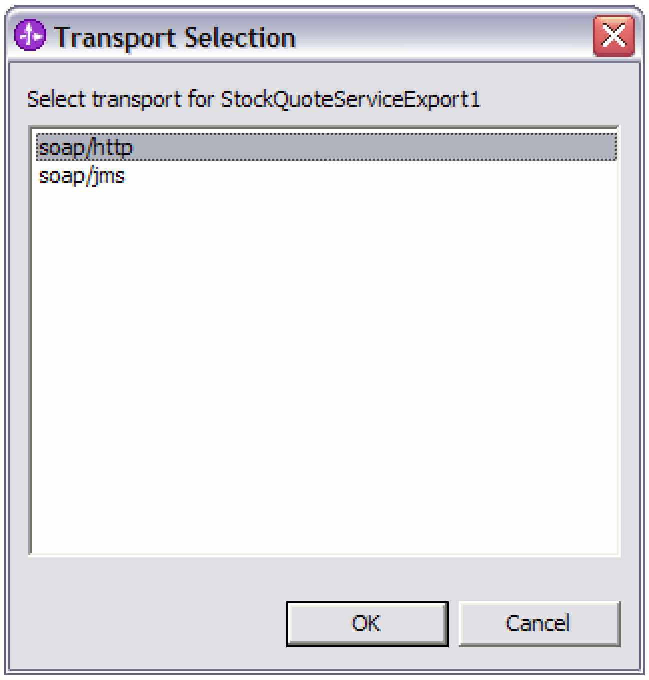
__ h. Wire the **UnModeledFaultTest** reference to the **StockQuoteServiceImport1**



__ i. Right click **UnModeledFaultTestFlow** and select **Generate Export > Web Services Binding**



__ j. Select **soap/http** when prompted for the transport

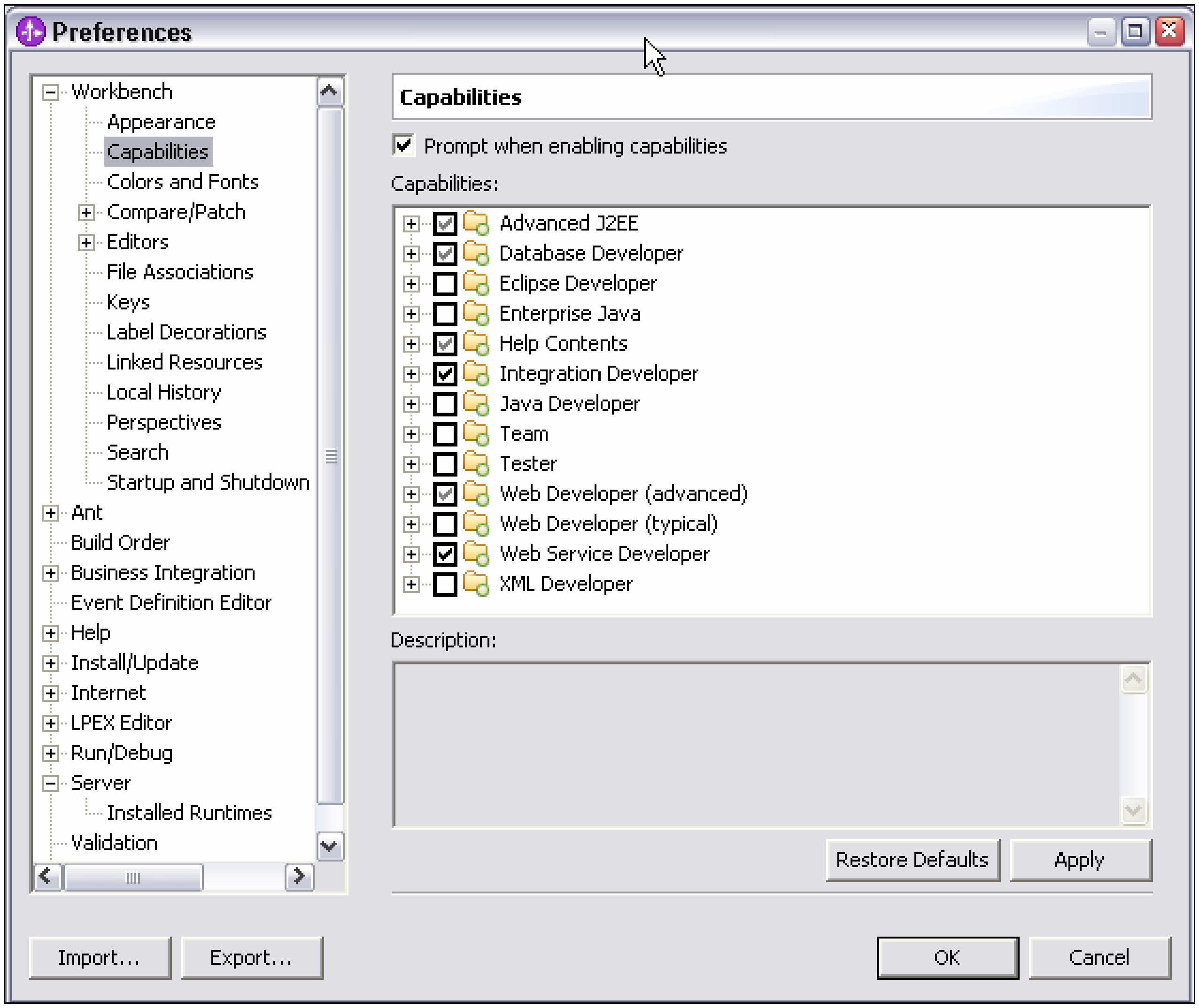


__ k. Click **OK**

__ l. Save all work by **File > Save All** or **Crtl + Shift + S**

____ 2. **OPTIONAL:** If using a host or port other than localhost:9080 on your test system, complete these steps to modify the destinations that are assumed in the imported code.

__ a. Under **Window → Preferences → Workbench → Capabilities**, select **Web Service Developer** and click **Apply**



Capabilities

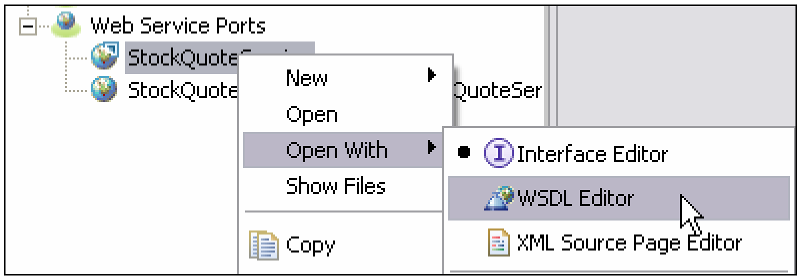
[✓] Prompt when enabling capabilities

Capabilities:

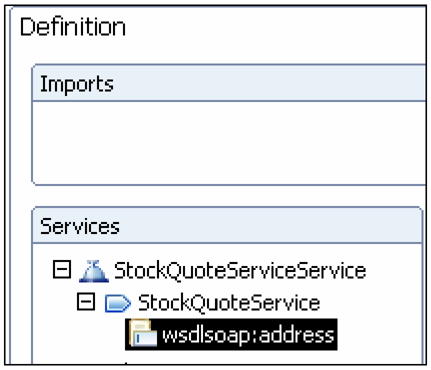
✓	Advanced J2EE
✓	Database Developer
☐	Eclipse Developer
☐	Enterprise Java
✓	Help Contents
✓	Integration Developer
☐	Java Developer
☐	Team
☐	Tester
✓	Web Developer (advanced)
☐	Web Developer (typical)
✓	Web Service Developer
☐	XML Developer

Description:

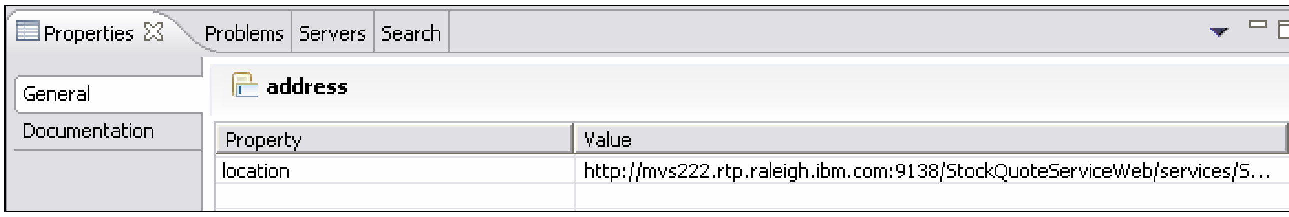
__ b. In the Business Integration view, right-click on **StockQuoteService** under Web Service Ports and select **Open With → WSDL Editor**



__ c. Expand the **StockQuoteServiceService** on the right until you see **wsdlsoap:address** and select that.



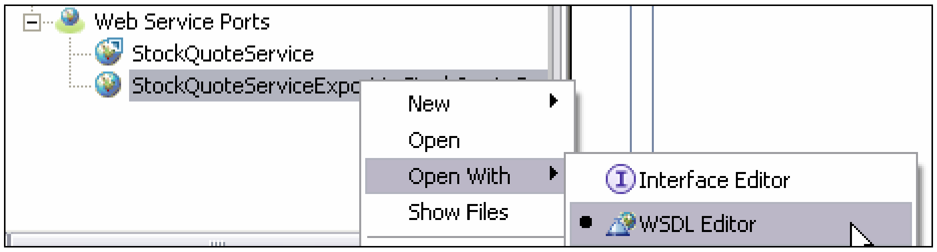
__ d. In the Properties view at the bottom, change the localhost:9080 to match the **<HOSTNAME>:<PORT>** of your test server.



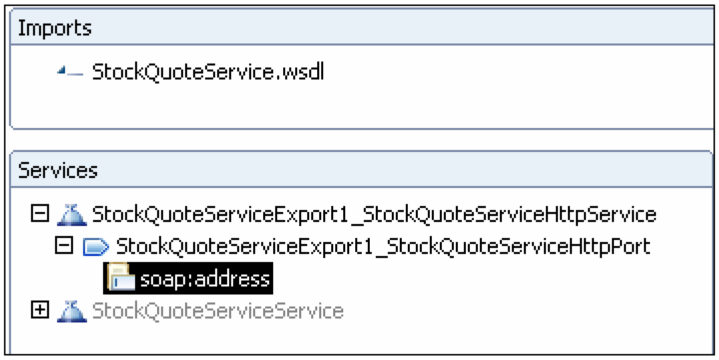
Property	Value
location	http://mvs222.rtp.raleigh.ibm.com:9138/StockQuoteServiceWeb/services/S...

__ e. Press **Ctrl+S** to save the wsdl file. Close it.

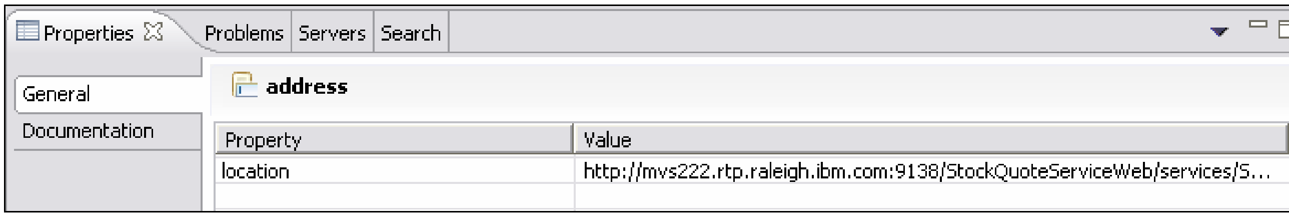
__ f. In the Business Integration view, right-click on **StockQuoteServiceExport1_StockQuoteServiceHttpPort** under Web Service Ports and select **Open With → WSDL Editor**.



__ g. Expand the **StockQuoteServiceExport1_StockQuoteServiceHttpService** on the right until you see **soap:address** and select that.


Imports
StockQuoteService.wsdl
Services
StockQuoteServiceExport1_StockQuoteServiceHttpService
StockQuoteServiceExport1_StockQuoteServiceHttpPort
soap:address
StockQuoteServiceService

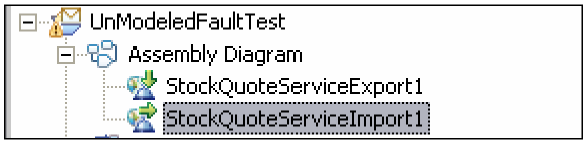
__ h. In the Properties view at the bottom, change the localhost:9080 to match the **<HOSTNAME>:<PORT>** of your test server.



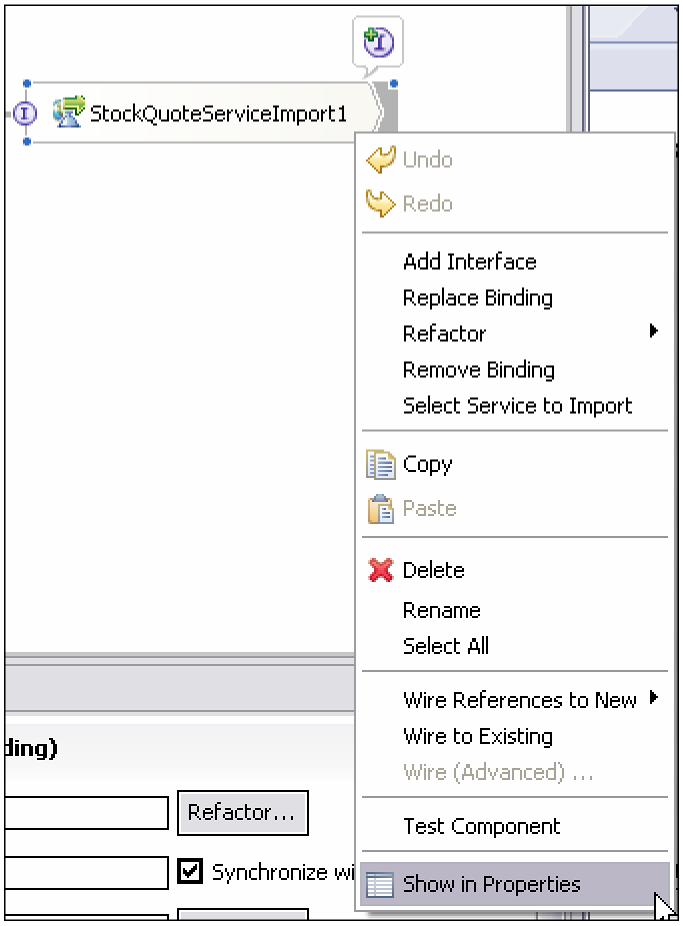
Property	Value
location	http://mvs222.rtp.raleigh.ibm.com:9138/StockQuoteServiceWeb/services/S...

__ i. Press **Ctrl+S** to save the wsdl file. Close it.

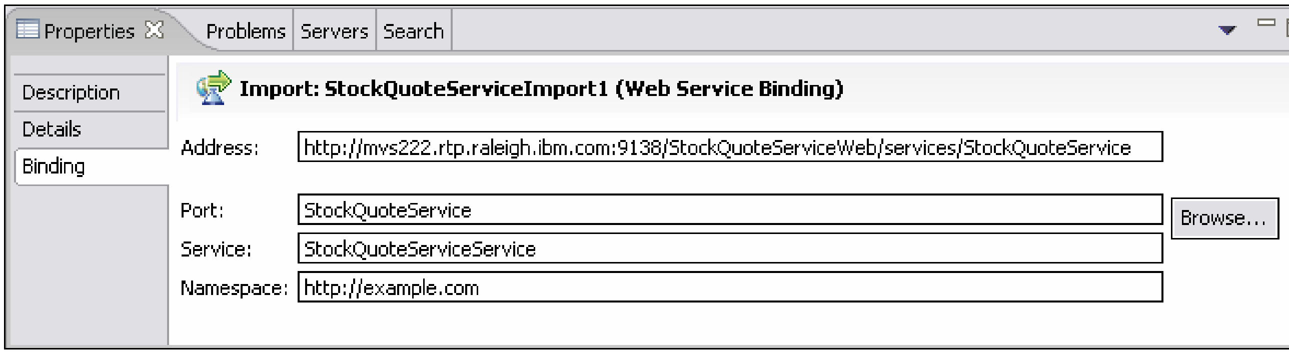
__ j. In the Business Integration view, open the **Assembly Diagram** under **UnModeledFaultTest** by double-clicking on **StockQuoteServiceImport1**



__ k. Right-click on the **StockQuoteServiceImport1** box on the Assembly Diagram and click on **Show in Properties**



__ l. In the **Binding tab** in the **Properties** at the bottom of the screen, change the localhost:9080 to match the **<HOSTNAME>:<PORT>** of your test server



Properties | Problems | Servers | Search

Description | Details | Binding

Import: StockQuoteServiceImport1 (Web Service Binding)

Address: http://mvs222.rtp.raleigh.ibm.com:9138/StockQuoteServiceWeb/services/StockQuoteService

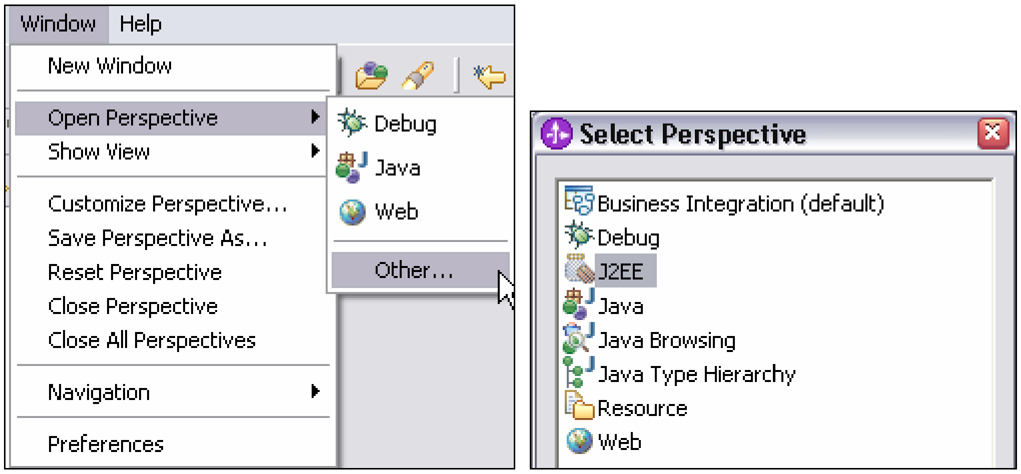
Port: StockQuoteService

Service: StockQuoteServiceService

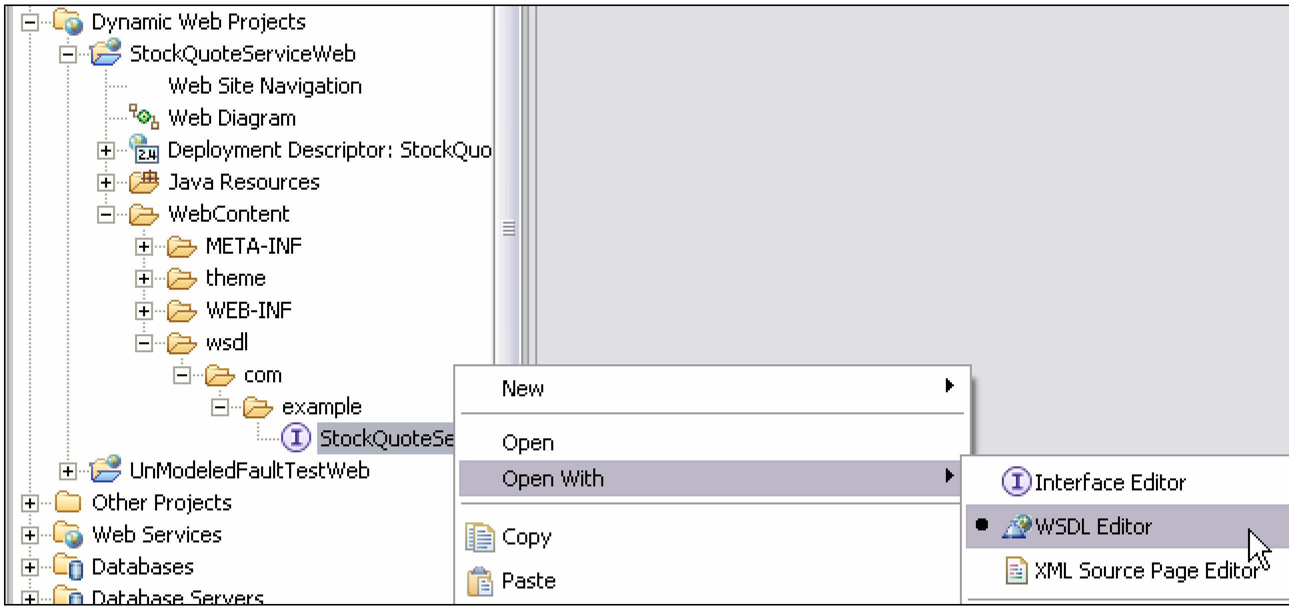
Namespace: http://example.com

__ m. Press **Ctrl+S** to save the assembly diagram. Close it.

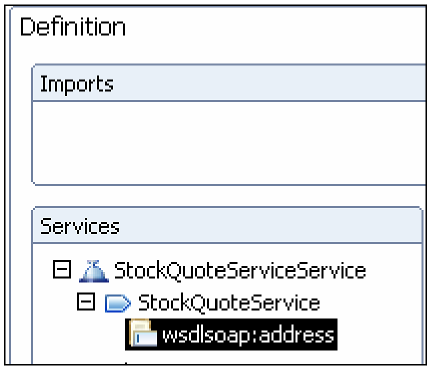
__ n. Open the J2EE Perspective (**Window → Open Perspective → Other…→ J2EE**). Click on **OK**.



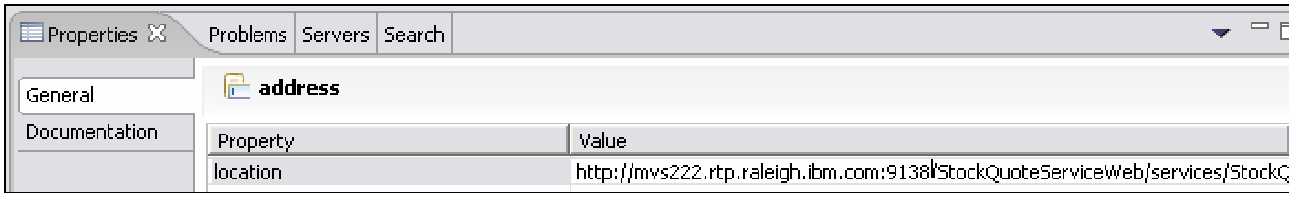
__ o. In the Project Explorer view, right-click on **StockQuoteService.wsdl** under **Dynamic Web Projects → StockQuoteServiceWeb → WebContent → wsdl → com → example** and select **Open With → WSDL Editor**.



__ p. Expand the **StockQuoteServiceService** on the right until you see **wsdlsoap:address** and select that.



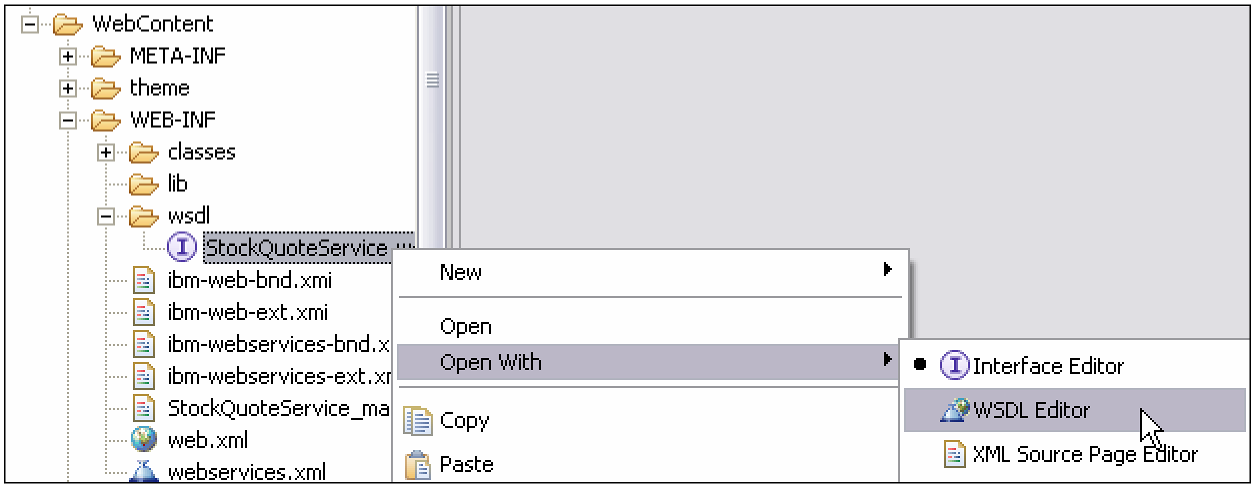
__ q. In the Properties view at the bottom, change the localhost:9080 to match the **<HOSTNAME>:<PORT>** of your test server.



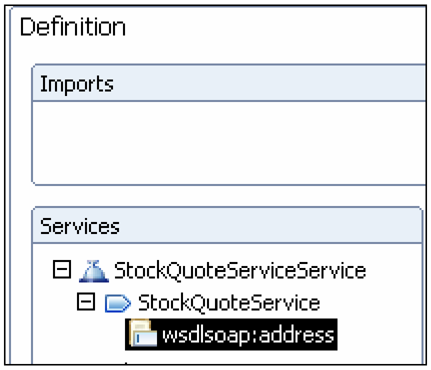
Property	Value
location	http://mvs222.rtp.raleigh.ibm.com:9138/StockQuoteServiceWeb/services/StockQ

__ r. Press **Ctrl+S** to save the wsdl file. Close it.

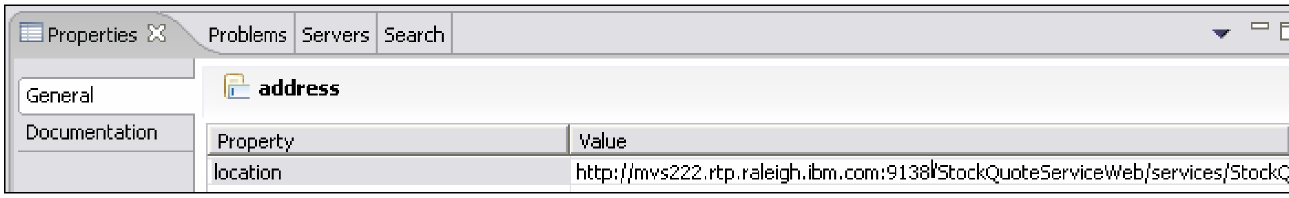
__ s. In the Project Explorer view, right-click on **StockQuoteService.wsdl** under **Dynamic Web Projects → StockQuoteServiceWeb → WebContent → WEB-INF → wsdl** and select **Open With → WSDL Editor**.



__ t. Expand the **StockQuoteServiceService** on the right until you see **wsdlsoap:address** and select that.



__ u. In the Properties view at the bottom, change the localhost:9080 to match the **<HOSTNAME>:<PORT>** of your test server.



Property	Value
location	http://mvs222.rtp.raleigh.ibm.com:9138/StockQuoteServiceWeb/services/StockQ

__ v. Press **Ctrl+S** to save the wsdl file. Close it.

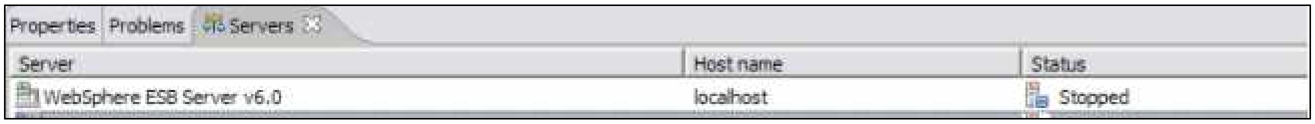
Part 6: Add the modules to the server

In this section, you will prepare the WebSphere ESB Server to run the mediation to test the unmodeled fault.

____ 1. Start WebSphere ESB Server

__ a. Open **Servers View**.

__ b. Highlight **WebSphere ESB Server v6.0** and click **Start button** ( ).

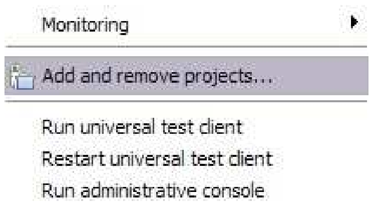


Server	Host name	Status
WebSphere ESB Server v6.0	localhost	Stopped

__ c. This will take some time. Wait for the server to start and the `Server server1 open for e-business` message.

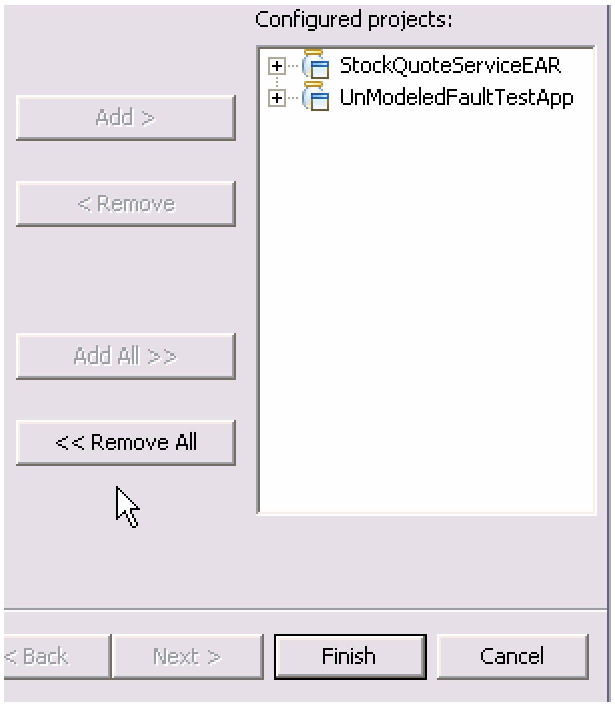
____ 2. Add **projects** to the WebSphere ESB Server (once it has started, not before)

__ a. In Servers view, right-click on WebSphere ESB Server v6.0 and select "Add and Remove Projects..."

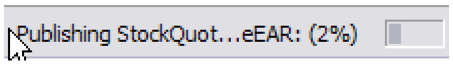

Monitoring
Add and remove projects...
Run universal test client
Restart universal test client
Run administrative console

NOTE: Note that the WebSphere server you are using is configured with a WebSphere ESB Server profile that is part of the installation and not part of the workspace. Therefore, if you have projects deployed to the server from a different workspace, there may be some naming conflicts or other problems. If this occurs, open the administrative console and stop and uninstall those projects before adding these projects. That should avoid any potential errors.

__ b. Click **Add-All>>** button to move all projects to server and click **Finish** button.



__ c. Wait for the deployment to finish. While the project is deploying you will see something like the following in the lower right corner of WebSphere Integration Developer.

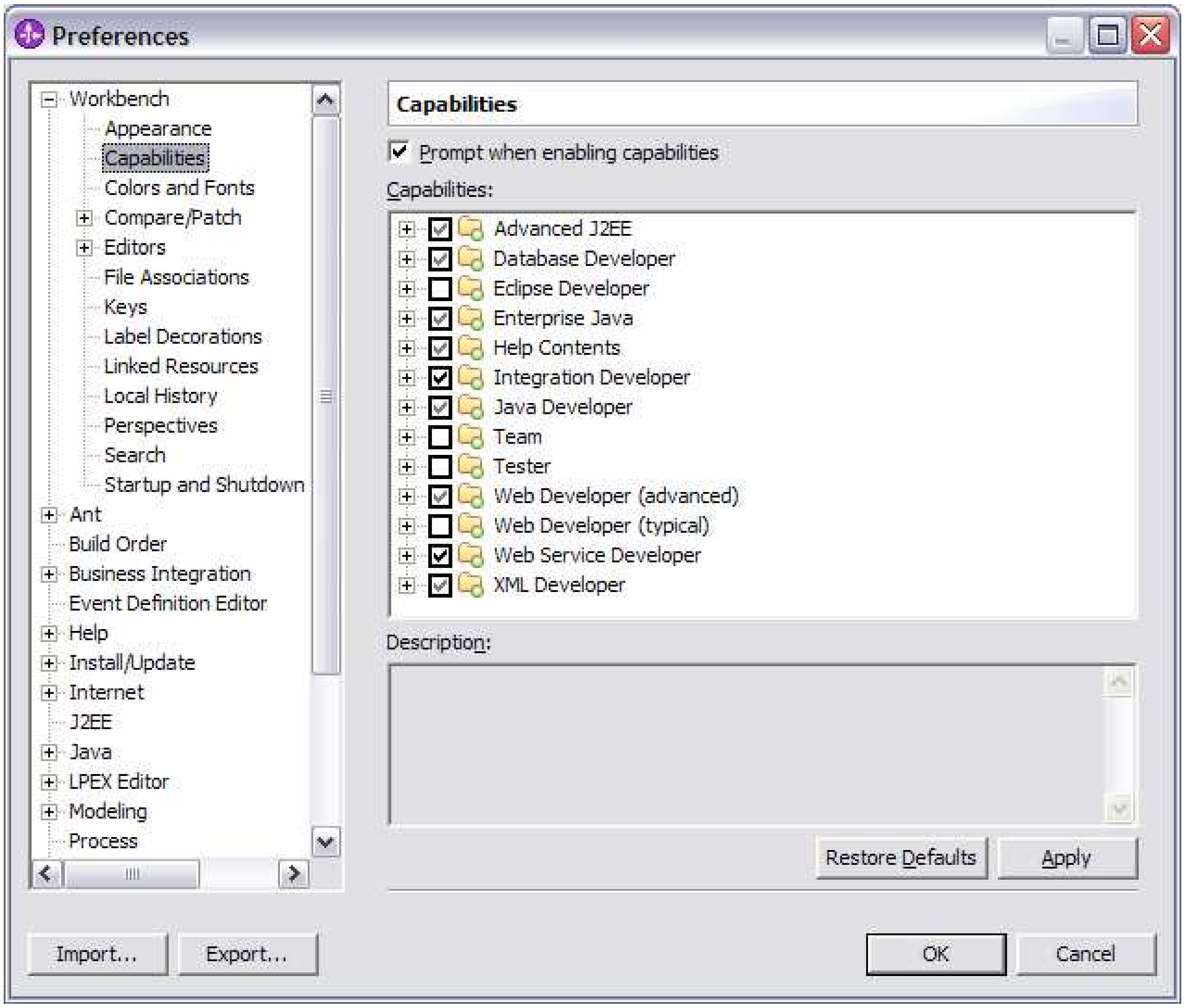


Part 7: Test the mediations with the Web Services Explorer

____ 1. Enable the Web Service Developer in WebSphere Integration Developer. You only need to do this once.

__ a. Select **Window > Preferences** from the menu bar in WebSphere Integration Developer.

__ b. Open **Work bench** > **Capabilities** and select **Web Service Developer**



☑ Prompt when enabling capabilities

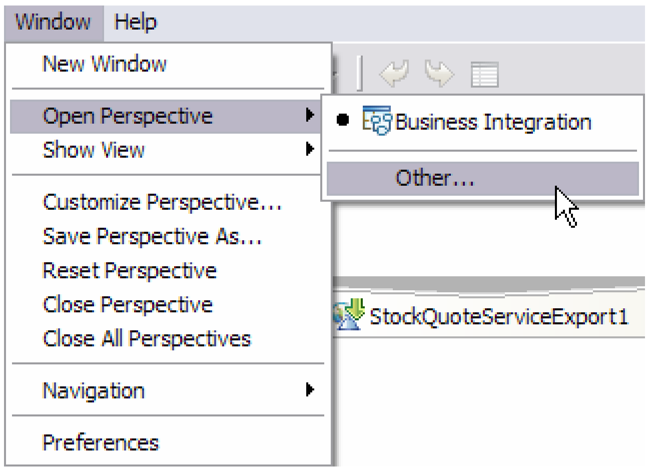
Capabilities:

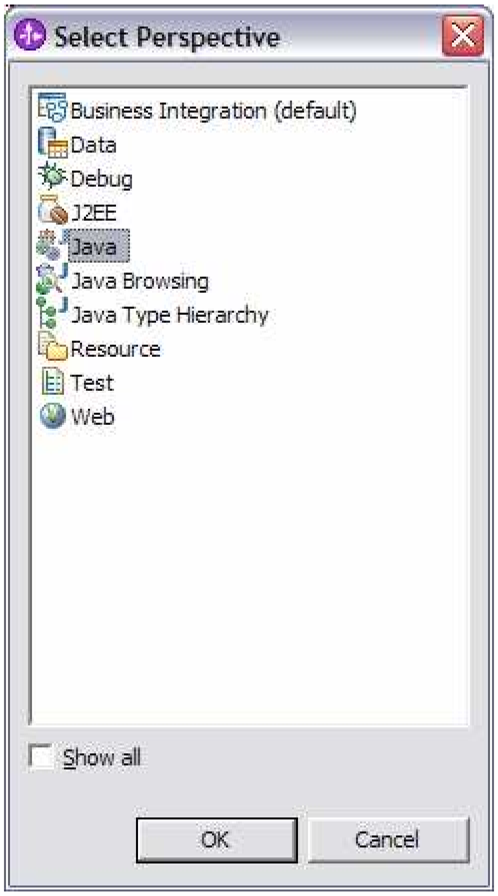
- ☑ Advanced J2EE
- ☑ Database Developer
- ☐ Eclipse Developer
- ☑ Enterprise Java
- ☑ Help Contents
- ☑ Integration Developer
- ☑ Java Developer
- ☐ Team
- ☐ Tester
- ☑ Web Developer (advanced)
- ☐ Web Developer (typical)
- ☑ Web Service Developer
- ☑ XML Developer

__ c. Click **OK**

____ 2. Open the Java perspective in WebSphere Integration Developer

__ a. Select **Window > Open Perspective > Other > Java** from the menu





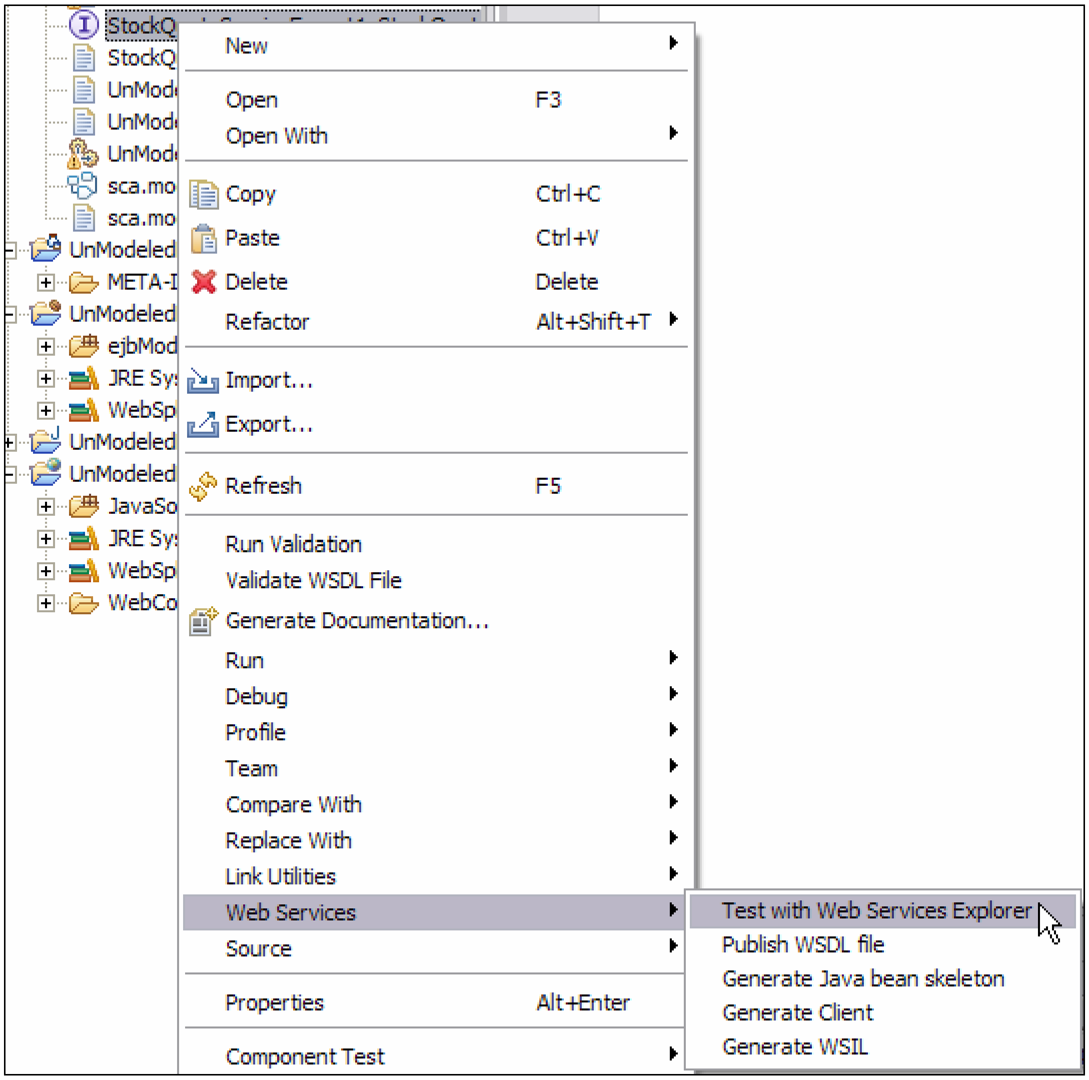
__ b. Click **OK**

____ 3. Test the mediation flows using the Web Services Explorer

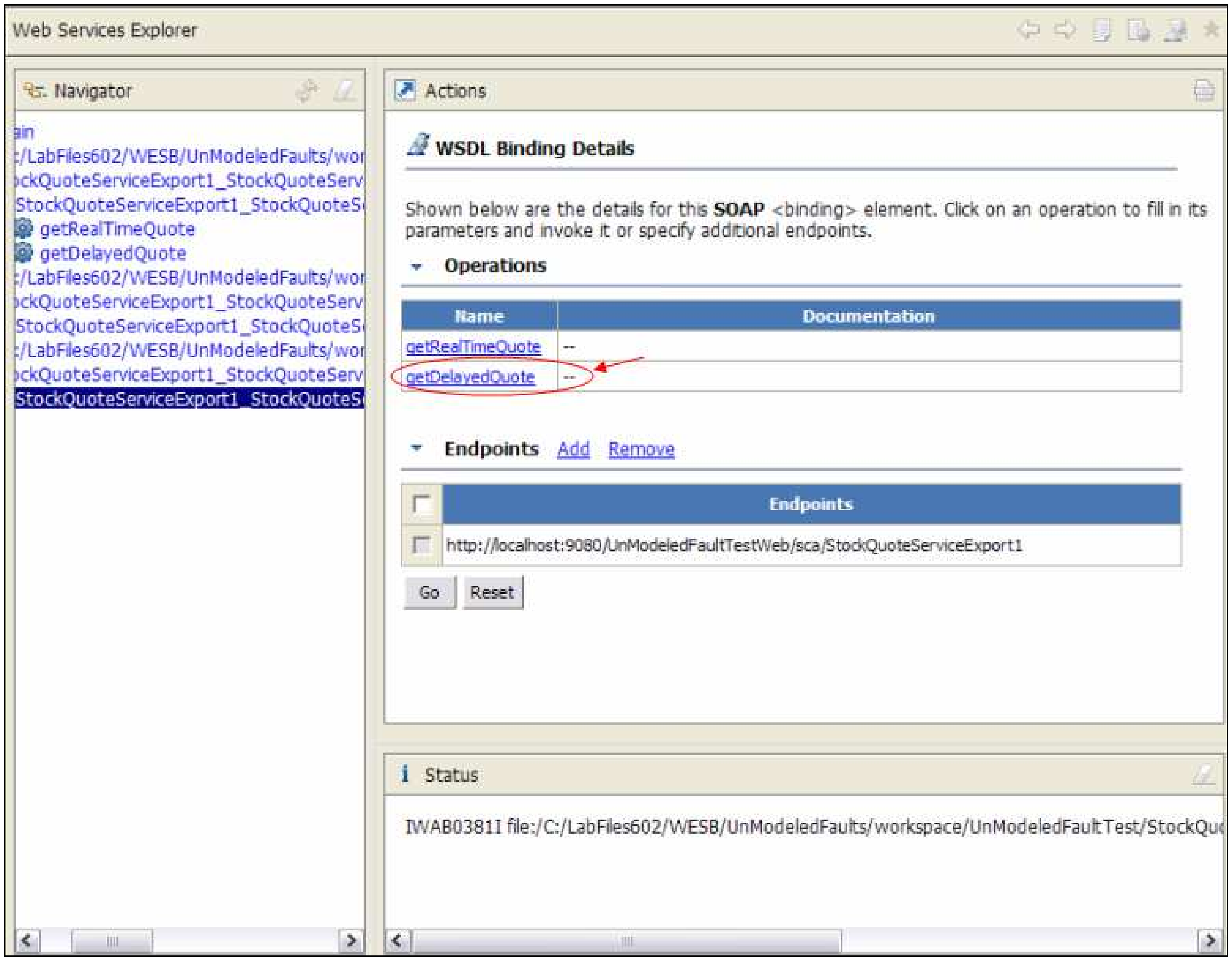
__ a. Expand the **UnModeledFaultTest** project

__ b. Right click the **StockQuoteServiceExport1_StockQuoteServiceHttp_Service > Web Services > Test with Web Services Explorer**

NOTE: Web Services Developer capability will need to be turned on to se the "Test with Web Services Explorer" menu option. Go to Window > Preferences > Workbench > Capabilities to turn on menu options or see above instructions.



__ c. The Web Services Explorer is opened in the main window. Click on **GetDelayedQuote** in the **Actions** window. Ignore any warnings you may see on the problems tab at this time.



ain
:/LabFiles602/WESB/UnModeledFaults/wor
ockQuoteServiceExport1_StockQuoteServ
StockQuoteServiceExport1_StockQuoteS
getRealTimeQuote
getDelayedQuote
:/LabFiles602/WESB/UnModeledFaults/wor
ockQuoteServiceExport1_StockQuoteServ
StockQuoteServiceExport1_StockQuoteS
:/LabFiles602/WESB/UnModeledFaults/wor
ockQuoteServiceExport1_StockQuoteServ
StockQuoteServiceExport1_StockQuoteS

WSDL Binding Details

Shown below are the details for this **SOAP** <binding> element. Click on an operation to fill in its parameters and invoke it or specify additional endpoints.

Operations

Name	Documentation
getRealTimeQuote	--
getDelayedQuote	--

Endpoints Add Remove

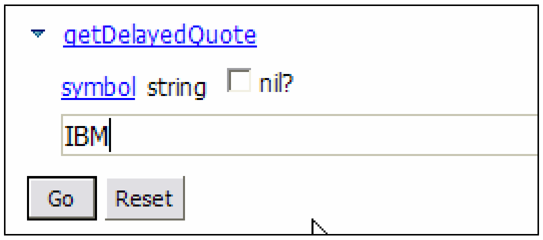
	Endpoints
☐	http://localhost:9080/UnModeledFaultTestWeb/sca/StockQuoteServiceExport1

Go Reset

IWAB0381I file:/C:/LabFiles602/WESB/UnModeledFaults/workspace/UnModeledFaultTest/StockQu

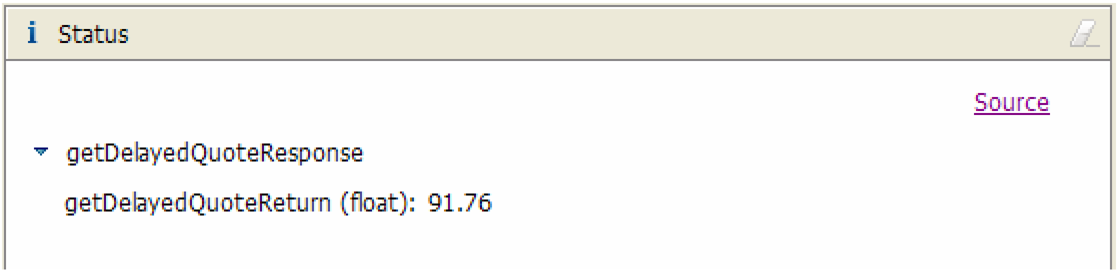
____ 4. Test the flow with valid data

__ a. Input **IBM** as the symbol you want to get a quote for



__ b. Click **Go**

__ c. Verify that the result is shown in the Status section of the Web Services Explorer. Your stock quote value may differ.

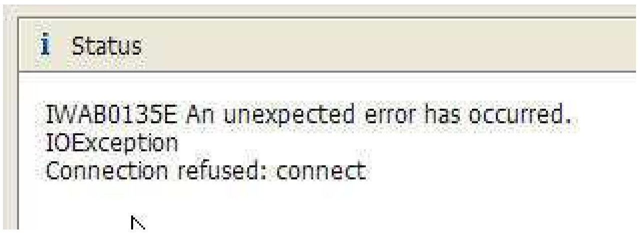

Status

Source

getDelayedQuoteResponse

getDelayedQuoteReturn (float): 91.76

NOTE: Note: If you receive this error in the status area after clicking the GO button….

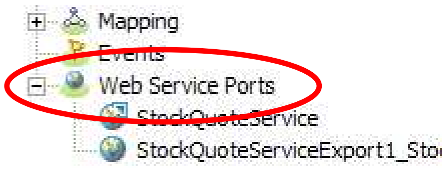

Status

```
IWAB0135E An unexpected error has occurred.
IOException
Connection refused: connect
```

The Web Service Explorer is not accessing the ESB port, 9081. To update the Web service port, follow these instructions.

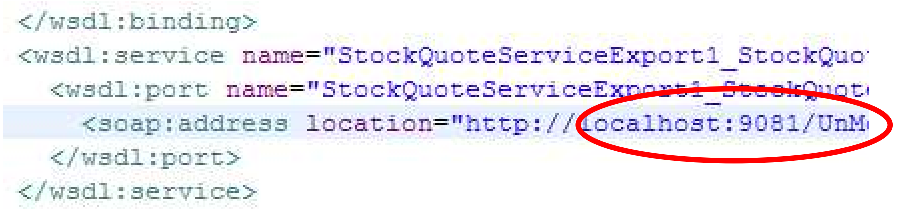
1. Navigate to the Business Integration View and open the mediation module with the Web service

2. Open Web Service Ports and right click the Web service that is failing.



3. Navigate to Open With > XML Source Page Editor.

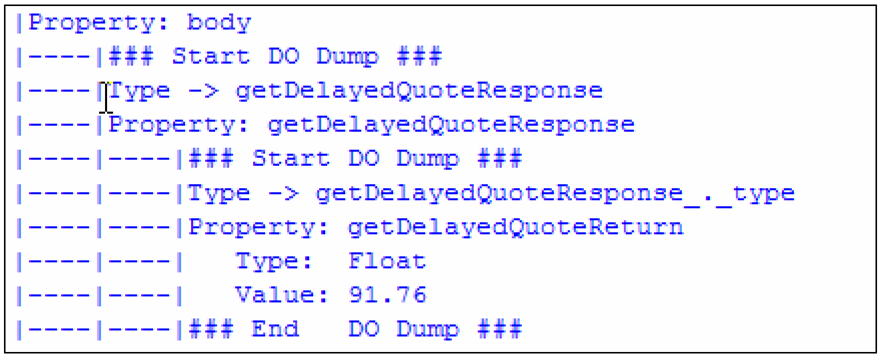
4. At the bottom of the WSDL, there is a <soap:address location. Change the port in localhost:<port> to 9081.


```
</wsdl:binding>
<wsdl:service name="StockQuoteServiceExport1_StockQuo
  <wsdl:port name="StockQuoteServiceExport1_StockQuot
    <soap:address location="http://localhost:9081/UnM
  </wsdl:port>
</wsdl:service>
```

__ d. Look in the console log at the SMO that was dumped as part of the Custom Mediation. Verify that it went down the 'Successful Response' path. You should see the following at the beginning of the message.

```
############ Start DataObject Dump ############
User Supplied Comment = Successful Response
```

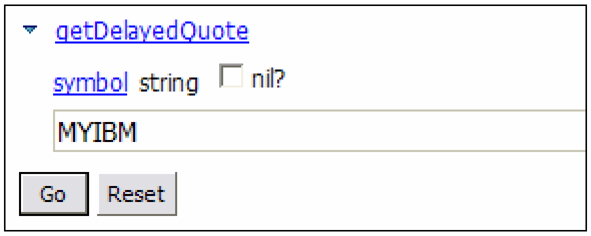
__ e. Inspect the rest of the SMO and see that the response had the stock value returned.



```
|Property: body
|----|### Start DO Dump ###
|----|Type -> getDelayedQuoteResponse
|----|Property: getDelayedQuoteResponse
|----|----|### Start DO Dump ###
|----|----|Type -> getDelayedQuoteResponse_._type
|----|----|Property: getDelayedQuoteReturn
|----|----|    Type:  Float
|----|----|    Value: 91.76
|----|----|### End   DO Dump ###
```

____ 5. Test the flow with an unknown symbol.

__ a. Input **MYIBM** as the symbol you want to get a quote for



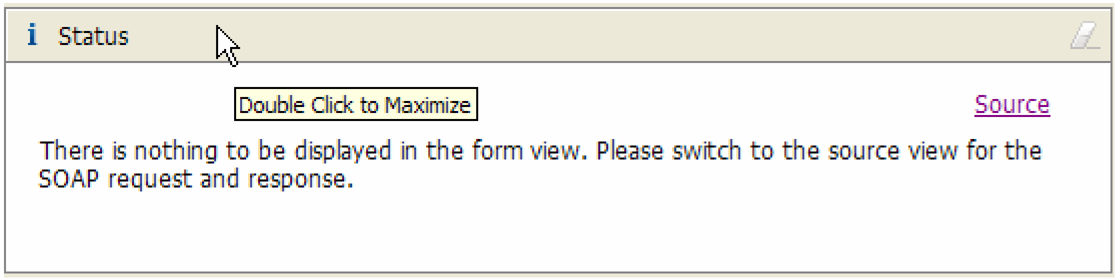
getDelayedQuote

symbol string nil?

MYIBM

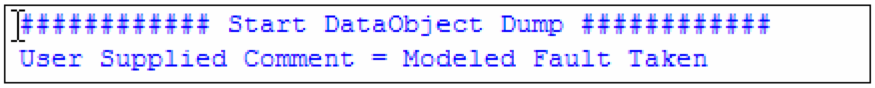
__ b. Click **Go**

__ c. The status window should inform you there is nothing to display:



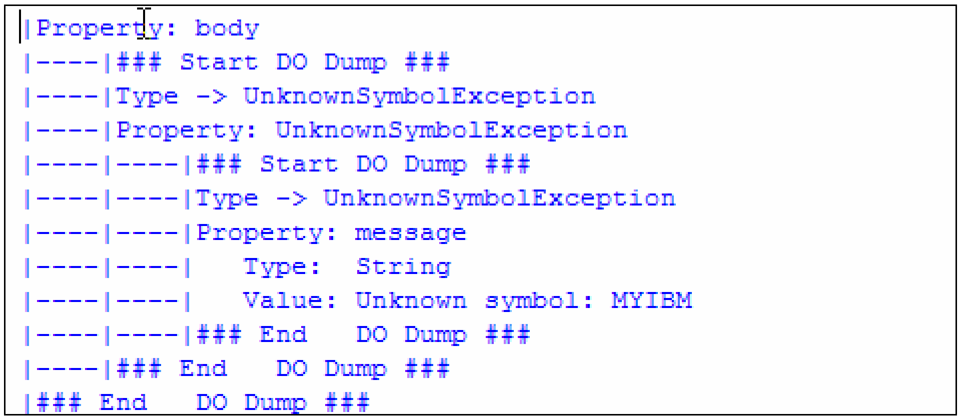
There is nothing to be displayed in the form view. Please switch to the source view for the SOAP request and response.

__ d. Look in the console log at the SMO that was dumped as part of the Custom Mediation. Verify that it went down the 'Modeled Fault' path



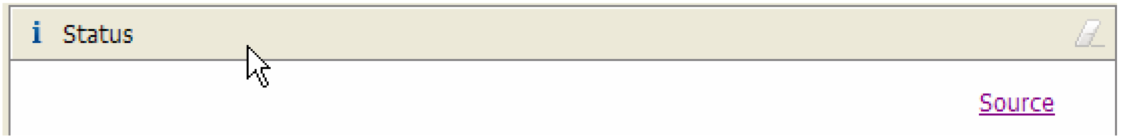
```
############ Start DataObject Dump ############
User Supplied Comment = Modeled Fault Taken
```

__ e. Inspect the rest of the SMO and see that the response returned the modeled UnknownSymbolException

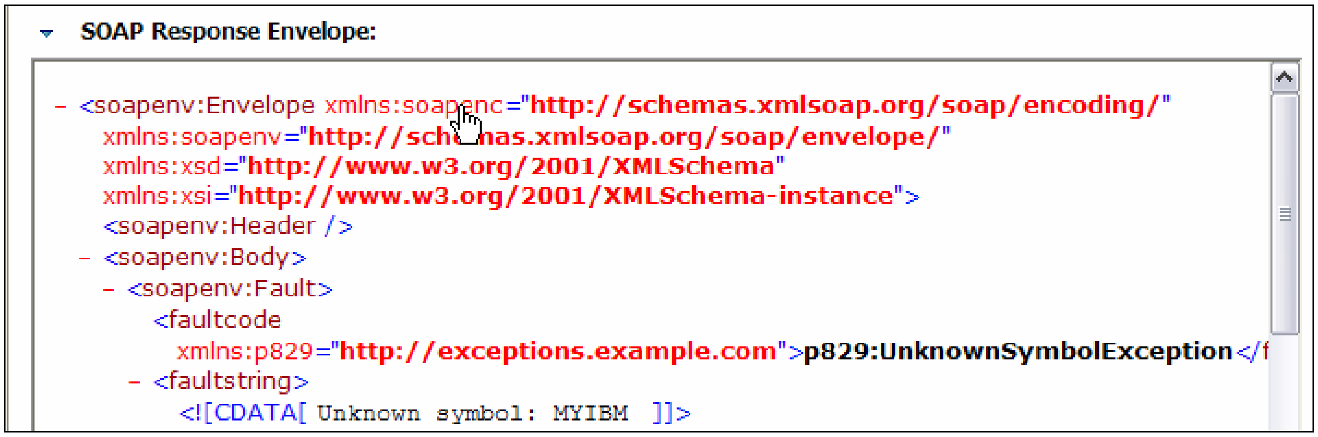


```
|Property: body
|----|### Start DO Dump ###
|----|Type -> UnknownSymbolException
|----|Property: UnknownSymbolException
|----|----|### Start DO Dump ###
|----|----|Type -> UnknownSymbolException
|----|----|Property: message
|----|----|    Type:  String
|----|----|    Value: Unknown symbol: MYIBM
|----|----|### End   DO Dump ###
|----|### End   DO Dump ###
|### End   DO Dump ###
```

__ f. The soap envelope is also available in the status window (bottom window) by clicking on the Source link. Click on **Source**



__ g. Maximize the Status window by double-clicking on the Status bar and inspect the Soap Response envelope. The modeled UnknownSymbolException fault should be seen.



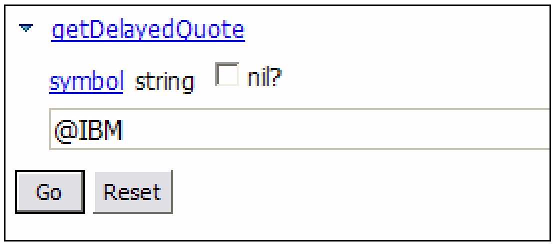
SOAP Response Envelope:

```
- <soapenv:Envelope xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
    xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Header />
  - <soapenv:Body>
    - <soapenv:Fault>
        <faultcode
          xmlns:p829="http://exceptions.example.com">p829:UnknownSymbolException</f
      - <faultstring>
          <![CDATA[ Unknown symbol: MYIBM  ]]>
```

__ h. Double-click on the Status window again to restore the Web Services Explorer window

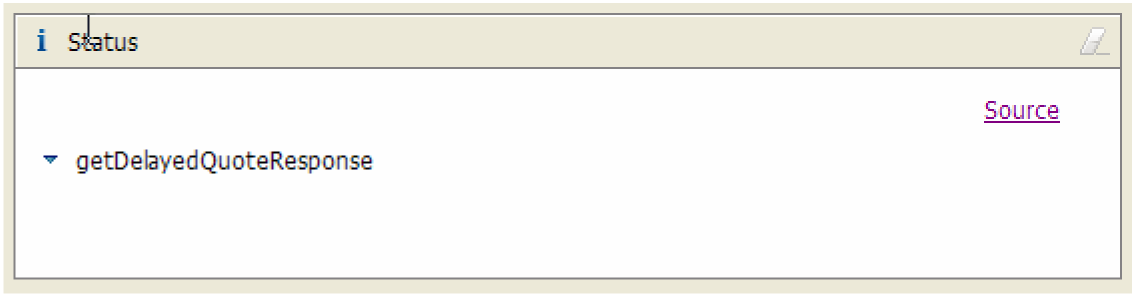
____ 6. Test the flow with a symbol containing invalid characters to test the unmodeled fault case

__ a. Input **@IBM** as the symbol you want to get a quote for

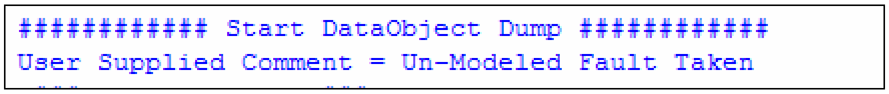


__ b. Click **Go**

__ c. The status window should have nothing in it:

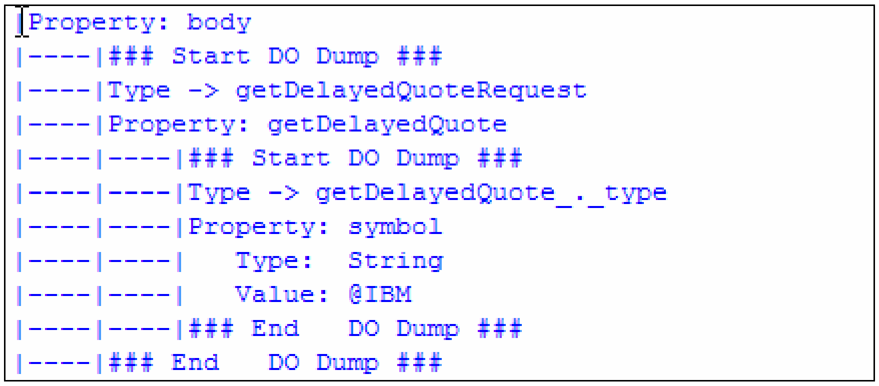


Look in the console log at the SMO that was dumped as part of the Custom Mediation. You will see a "BadSymbolException". In addition, verify that it went down the 'Unmodeled Fault' path



```
############ Start DataObject Dump ############
User Supplied Comment = Un-Modeled Fault Taken
```

__ d. Inspect the rest of the SMO and verify that the entire message was included in the response. You should see that the body has the invalid symbol you entered.



```
Property: body
|----|### Start DO Dump ###
|----|Type -> getDelayedQuoteRequest
|----|Property: getDelayedQuote
|----|----|### Start DO Dump ###
|----|----|Type -> getDelayedQuote_._type
|----|----|Property: symbol
|----|----|   Type:  String
|----|----|   Value: @IBM
|----|----|### End   DO Dump ###
|----|### End   DO Dump ###
```

What you did in this exercise

In this lab, you saw how to handle an unmodeled fault within a mediation flow. External services are not typically change controlled by their users and thus may change without the user of the service being aware. You saw how unmodeled faults (ones not specified in the WSDL interface) can be handled in the mediation flow when this occurs. You used the new unmodeled fault terminal in order to call a Custom Mediation that printed out the Service Message Object and then Stopped. Another option would be to use an Event Emitter to log the fault.

Solution Instructions

____ 1. Import **Solution** Project Interchange file.

__ a. With a blank workspace in WebSphere Integration Developer, Go to **File → Import → Project Interchange**

__ b. Click on top Browse button and navigate to **<LAB_FILES>\WESB\UnModeledFaults\import\WESB_UnModeledFaults_PI_solution.zip**

__ c. Click **Finish** button

____ 2. **OPTIONAL**: If testing on a remote system, complete **Step 2** in **Part 5: Assemble the module**

____ 3. Start with **Part 6: Add the modules to the server**

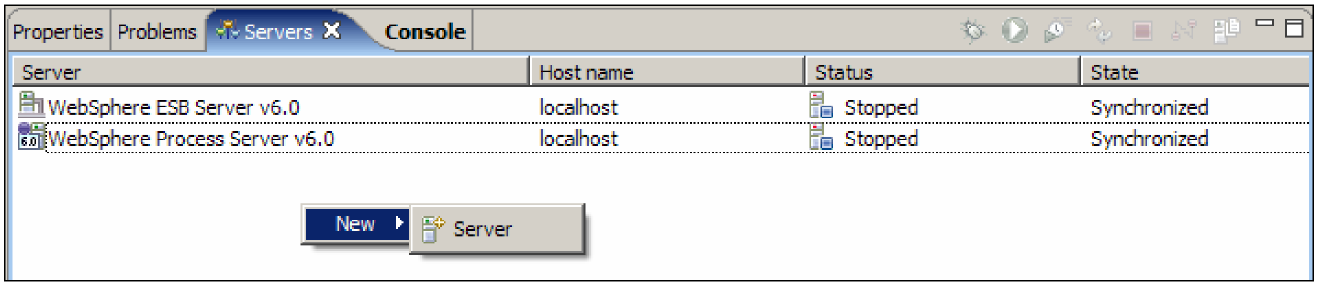
Task: Adding remote server to WebSphere Integration Developer test environment

This task describes how to add a remote server to the WebSphere Integration Developer Test environment. The sample will use a z/OS machine.

____ 1. Create a new remote server

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

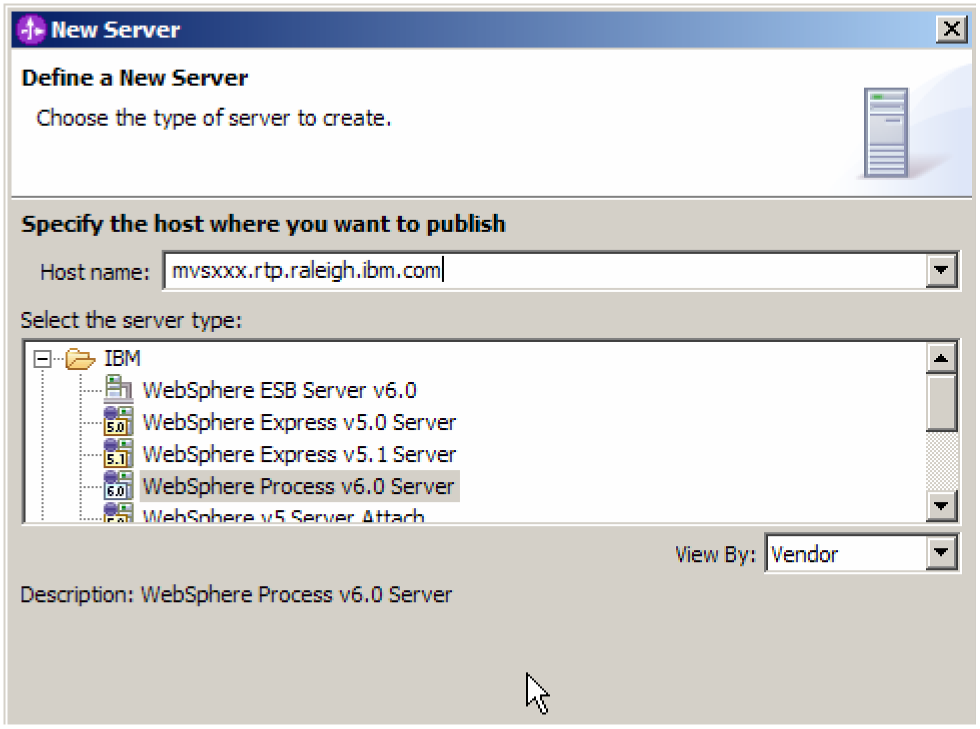
__ b. Select **New > Server**



Server	Host name	Status	State
WebSphere ESB Server v6.0	localhost	Stopped	Synchronized
WebSphere Process Server v6.0	localhost	Stopped	Synchronized

__ c. Specify host name of the remote server, **<HOSTNAME>**

__ d. Ensure that '**WebSphere Process v6.0 Server**' is highlighted in the server type list


New Server

Define a New Server

Choose the type of server to create.

Specify the host where you want to publish

Host name: mvsxxx.rtp.raleigh.ibm.com

Select the server type:

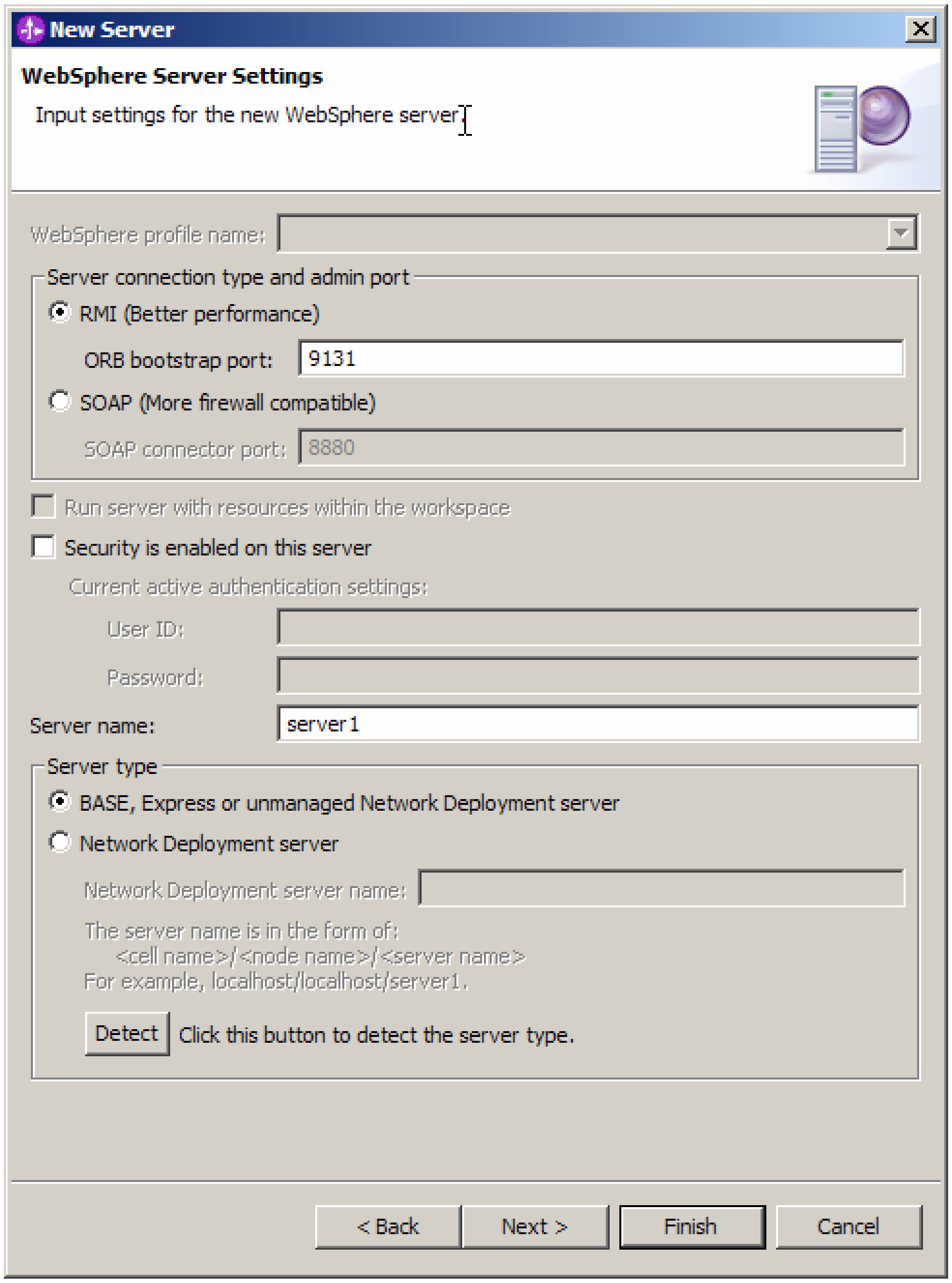
- IBM
  - WebSphere ESB Server v6.0
  - WebSphere Express v5.0 Server
  - WebSphere Express v5.1 Server
  - WebSphere Process v6.0 Server
  - WebSphere v5 Server Attach

View By: Vendor

Description: WebSphere Process v6.0 Server

__ e. Click **Next**

__ f. On the WebSphere Server Settings page, select the radio button for **RMI** and change the ORB bootstrap port to the correct setting (**<BOOTSTRAP_PORT>**)



WebSphere Server Settings

Input settings for the new WebSphere server.

WebSphere profile name:

Server connection type and admin port

● RMI (Better performance)

ORB bootstrap port: 9131

○ SOAP (More firewall compatible)

SOAP connector port: 8880

☐ Run server with resources within the workspace

☐ Security is enabled on this server

Current active authentication settings:

User ID:

Password:

Server name: server1

Server type

● BASE, Express or unmanaged Network Deployment server

○ Network Deployment server

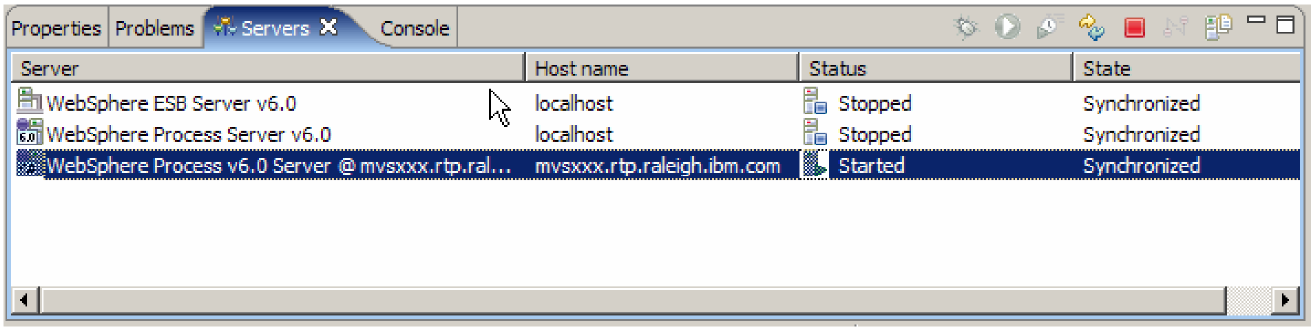
Network Deployment server name:

The server name is in the form of:
<cell name>/<node name>/<server name>
For example, localhost/localhost/server1.

Detect Click this button to detect the server type.

__ g. Click **Finish**

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



Server	Host name	Status	State
WebSphere ESB Server v6.0	localhost	Stopped	Synchronized
WebSphere Process Server v6.0	localhost	Stopped	Synchronized
WebSphere Process v6.0 Server @ mvsxxx.rtp.ral...	mvsxxx.rtp.raleigh.ibm.com	Started	Synchronized

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

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

‘**telnet <HOSTNAME> <TELNET_PORT>**’

userid: **<USERID>**

pwd: **<PASSWORD>**

__ b. Navigate to the bin directory for the profile being used:

cd <WAS_HOME>/profiles/<PROFILE_NAME>/bin

__ c. Run the command file to start the server: **./startServer.sh <SERVER_NAME>**

__ d. Wait for status message indicating server has started:

```
ADMU3200I: Server launched. Waiting for initialization status.

ADMU3000I: Server cl1sr01 open for e-business; process id is 0000012000000002
```

This page is left intentionally blank.