IBM WebSphere® Enterprise Service Bus V6.1 – Lab exercise

WebSphere Enterprise Service Bus lab
Unmodeled faults

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 V6.1 installed
- Web Sphere Enterprise Server Bus V6.1 test environment installed

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 V6.1 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.

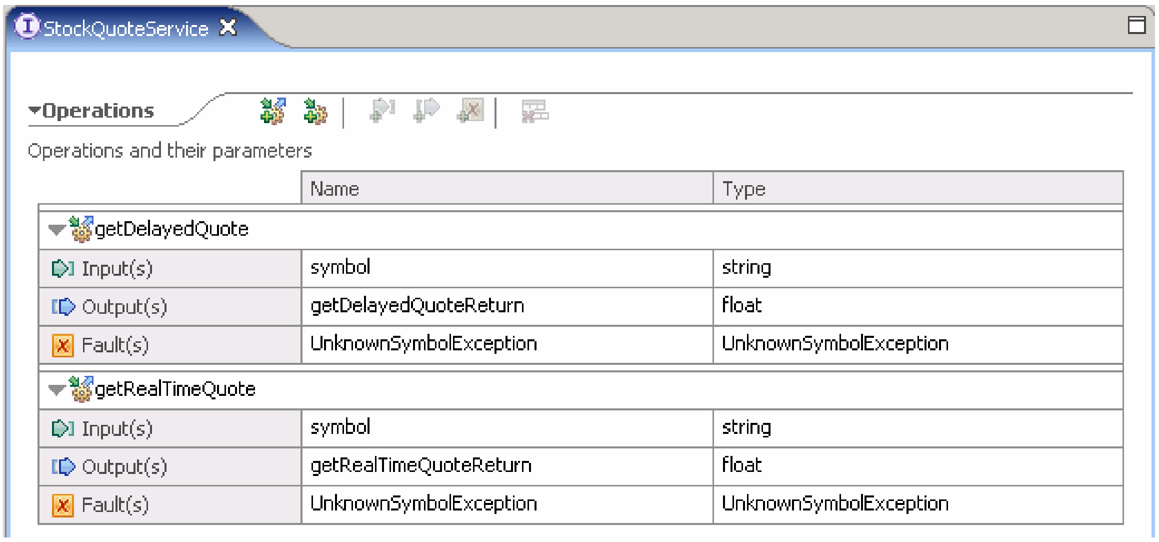
Introduction

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 creates a fault message if the stock symbol is not known. This particular fault message is modeled in the WSDL interface. To create an unmodeled fault, the mediation flow is run with a stock quote symbol that contains characters which are not valid, for example, #. @. !.

The Web service interface and business objects are shown below:

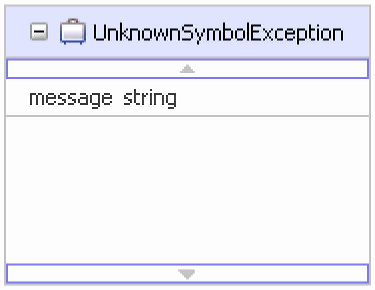
- The StockQuoteService provides these operations:
    - **getDelayedQuote** – accepts a stock symbol, and returns a cached stock quote, which expires in 10 minutes.
    - **getRealTimeQuote** – accepts a stock symbol, and returns an instance quote from Yahoo stock quote service.



Operations and their parameters

	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 can be inspected. In the case of the non-fault case, after printing out the SMO, the message is 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 is 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 might be Windows operating system specific. If you plan on running WebSphere Integration Developer on a Linux operating system you will need to issue the appropriate commands and use appropriate files for Linux. The directory locations are specified in the lab instructions using symbolic references, as follows:

Reference Variable	Windows Location	AIX®/UNIX® Location
<WID_HOME>	C:\Program Files\IBM\WID61	/opt/IBM/WID61
<ESB_PROFILE_HOME>	<WID_HOME>\pf\esb	<WID_HOME>/pf/esb
<LAB_FILES>	C:\Labfiles61\WESB\UnModeledFaults	/tmp/Labfiles61/WESB/UnModeledFaults
<WORKSPACE>	C:\Labfiles61\WESB\UnModeledFaults\workspace	/tmp/Labfiles61/WESB/UnModeledFaults/workspace

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:\LabFiles61\ should be replaced by C:/LabFiles61/

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:

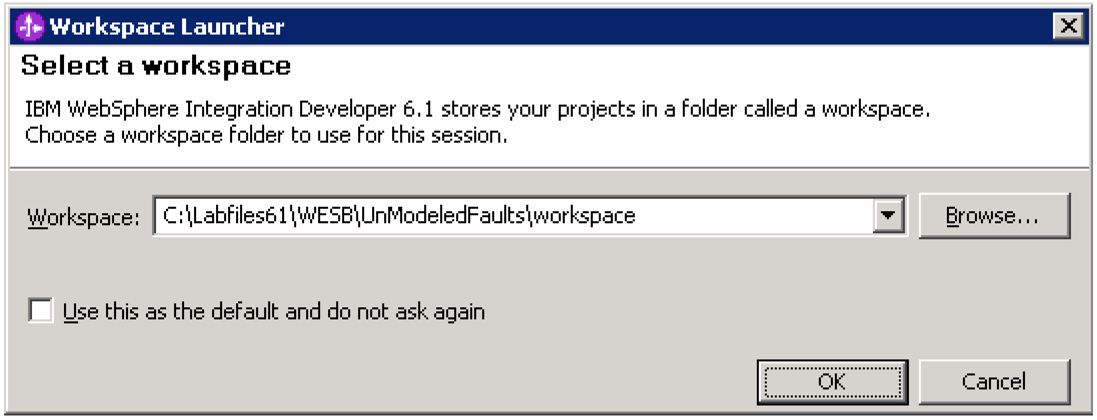
Reference variable	Example: Remote Windows test server location	Example: Remote z/OS® test server location	Input your values for the remote location of the test server
<SERVER_NAME>	server1	sssr011	
<WAS_HOME>	C:\Program Files\IBM\WebSphere\AppServer	/etc/sscell/AppServer	
<HOSTNAME>	localhost	mvsxxx.rtp.raleigh.ibm.com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

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

Part 1: 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 V6.1 with a workspace location of **<LAB_FILES>\WESB\UnModeledFaults\workspace**


Workspace Launcher

Select a workspace

IBM WebSphere Integration Developer 6.1 stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

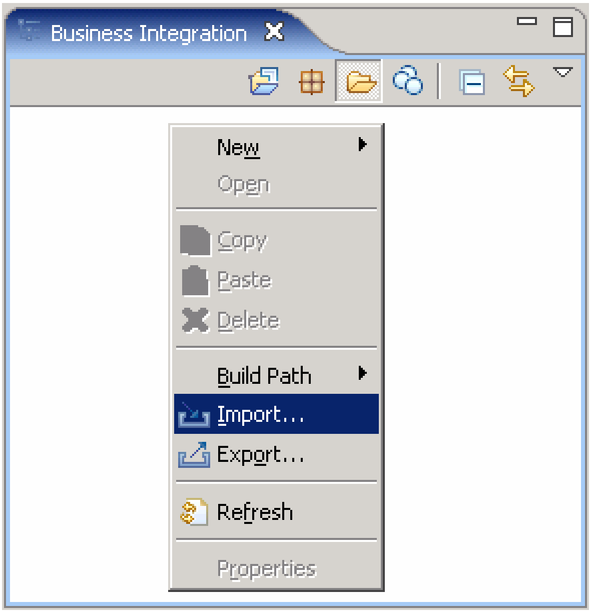
Workspace: C:\Labfiles61\WESB\UnModeledFaults\workspace

__ a. On the welcome screen, click the curved arrow at the top right to **Go to the business integration perspective** (), to close the Welcome window

____ 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)

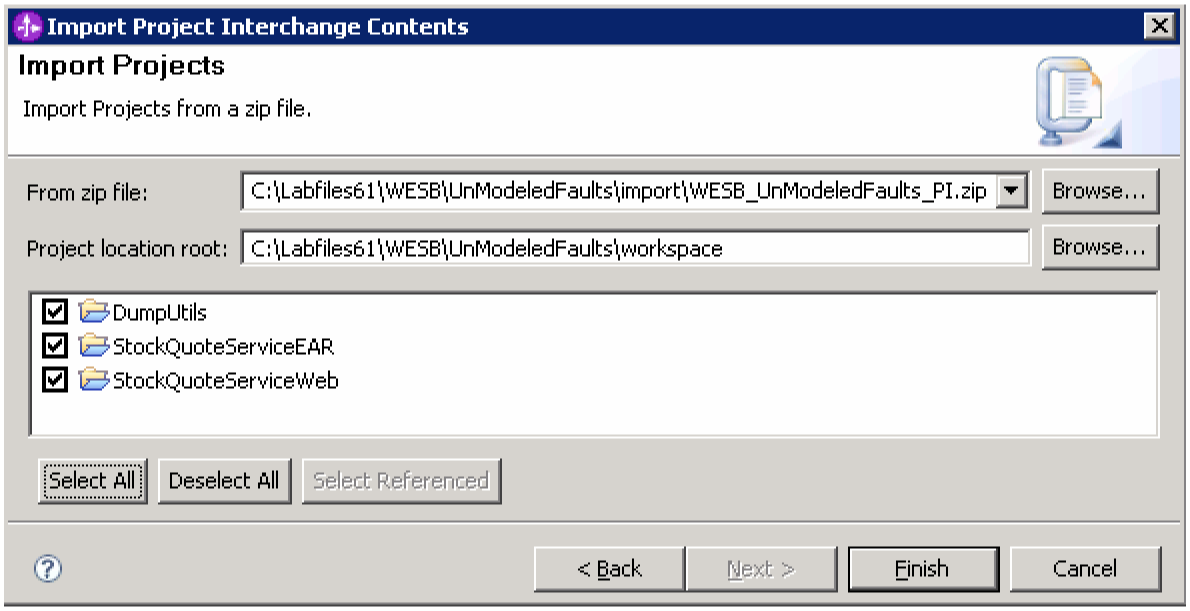
__ b. Select **Import** from the pop-up menu



__ c. From the **Import** dialog, expand '**Other**' and 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**


Import Project Interchange Contents

Import Projects

Import Projects from a zip file.

From zip file: C:\Labfiles61\WESB\UnModeledFaults\import\WESB_UnModeledFaults_PI.zip

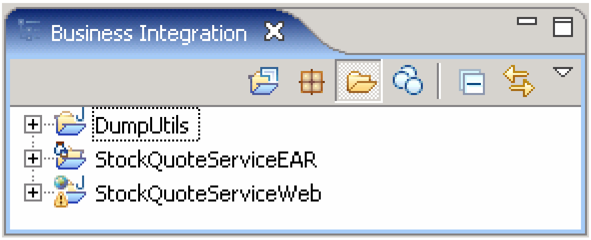
Project location root: C:\Labfiles61\WESB\UnModeledFaults\workspace

☑ DumpUtils
☑ StockQuoteServiceEAR
☑ StockQuoteServiceWeb

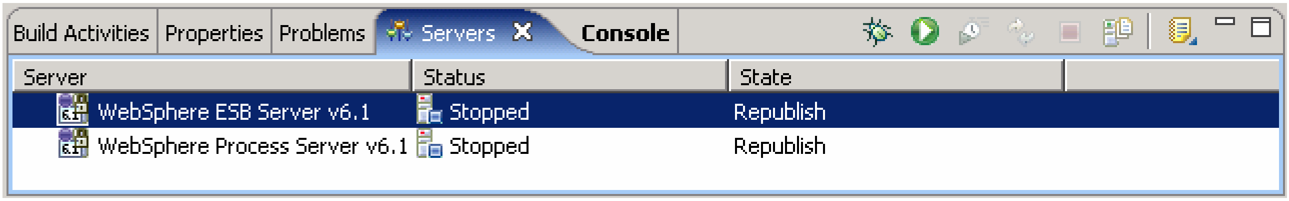
__ f. Click the **Select All** button to ensure all projects listed are selected

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

__ h. Verify that DumpUtils, StockQuoteServiceEAR and stockQuoteServiceWeb have been imported



____ 3. Verify that the '**WebSphere ESB Server v6.1**' is listed in the **Servers** view



Server	Status	State
WebSphere ESB Server v6.1	Stopped	Republish
WebSphere Process Server v6.1	Stopped	Republish

Part 2: Create mediation module

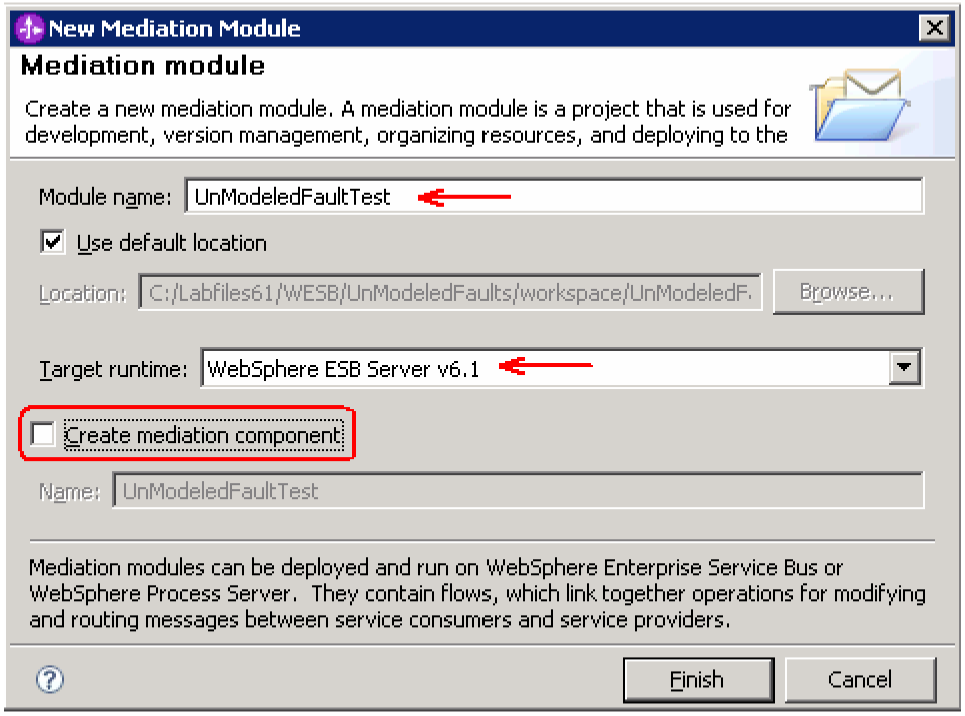
In this section of the lab, a new mediation module is created. There can only be one mediation module for each deployable project.

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

__ a. In the Business Integration view, right-click and select **New → Mediation Module** from the pop-up menu. The new Mediation Module wizard opens

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

__ c. Verify that the target runtime is the **WebSphere ESB Server v6.1** and **clear** the '**Create mediation flow component**' check 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

Module name: UnModeledFaultTest

☑ Use default location

Location: C:/Labfiles61/WESB/UnModeledFaults/workspace/UnModeledF.

Target runtime: WebSphere ESB Server v6.1

☐ Create mediation component

Name: UnModeledFaultTest

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 named **UnModeledFaultTest** is created. (You will create the mediation flow component later.)

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

__ a. In the Business Integration view, expand **UnModeledFaultTest** module you just created and double click '**Dependencies**' (Dependencies) to open the dependency editor

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

__ c. In the Java Project Selection dialog, select **DumpUtils** and click **OK**

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

__ e. 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 is used along with the operations it exposes.

____ 1. To import the Web service interface, complete these steps:

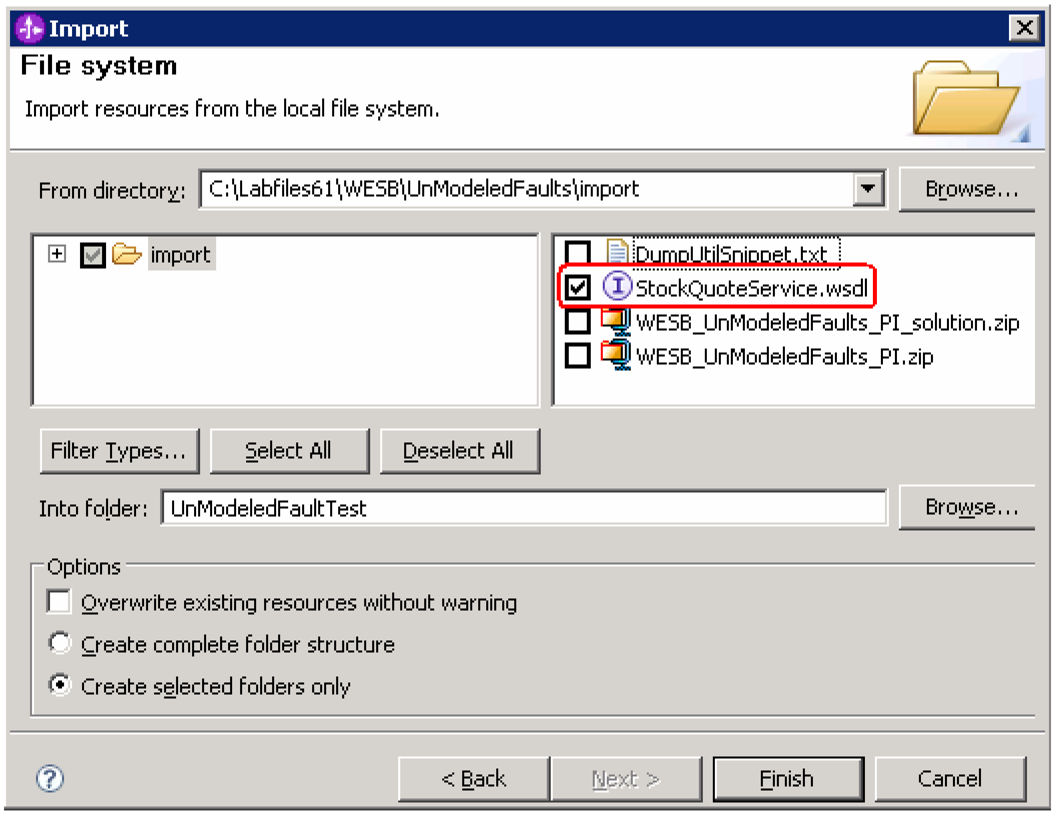
__ a. In the Business Integration view, right-click the **UnModeledFaultTest** mediation module and select **Import…** from the pop-up menu. The '**Import**' wizard opens

__ b. In the '**Import**' wizard, expand '**General**' and select **File System** and click **Next**

__ c. In the '**File system**' panel, click the **Browse** button for '**From Directory**' field, to file system

1) Select**<LAB_FILES>\import** and click **OK** on the '**Import from directory**' panel

2) Select the check box for **StockQuoteService.wsdl**


Import

File system

Import resources from the local file system.

From directory: C:\Labfiles61\WESB\UnModeledFaults\import

☑ import

☐ DumpUtilSnippet.txt
☑ StockQuoteService.wsdl
☐ WESB_UnModeledFaults_PI_solution.zip
☐ WESB_UnModeledFaults_PI.zip

Into folder: UnModeledFaultTest

Options
☐ Overwrite existing resources without warning
○ Create complete folder structure
● Create selected folders only

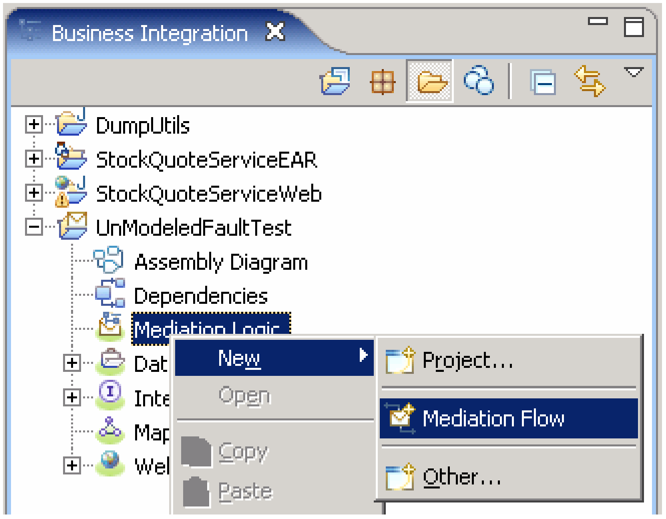
__ d. Click **Finish**.

Part 4: Create the mediation flow

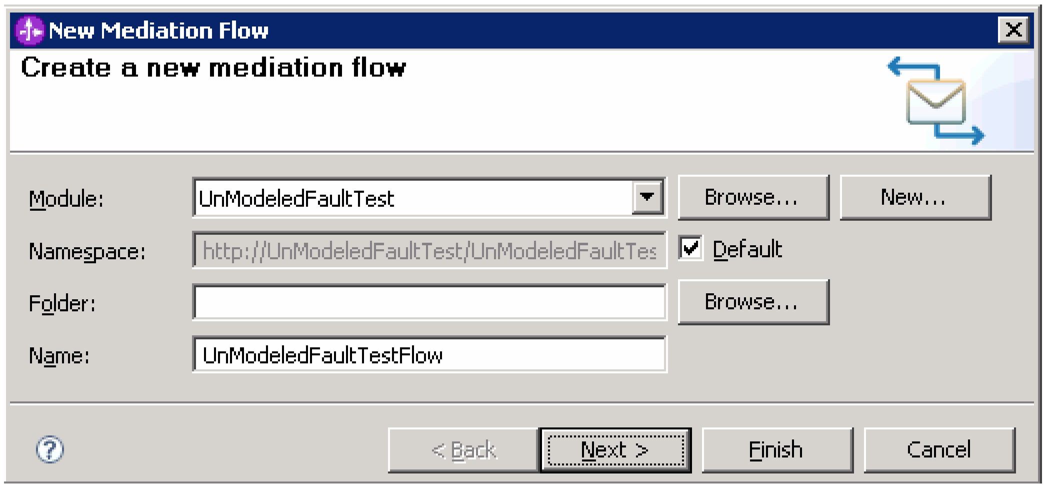
In this section of the lab, you will create the Mediation Flow. Once the operations are wired together, some mediation primitives are added for the new unmodeled fault flow in the mediation flow component. Custom Mediations are added to the various flows to dump the messages and inspect their contents.

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

__ a. In the Business Integration view, expand '**UnModeledFaultTest**' mediation module, right-click on '**Mediation Logic**' and select **New → Mediation Flow** from the pop-up menu



__ b. In the New Mediation Flow panel, enter the **Name** as **UnModeledFaultTestFlow**


New Mediation Flow

Create a new mediation flow

Module: UnModeledFaultTest

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

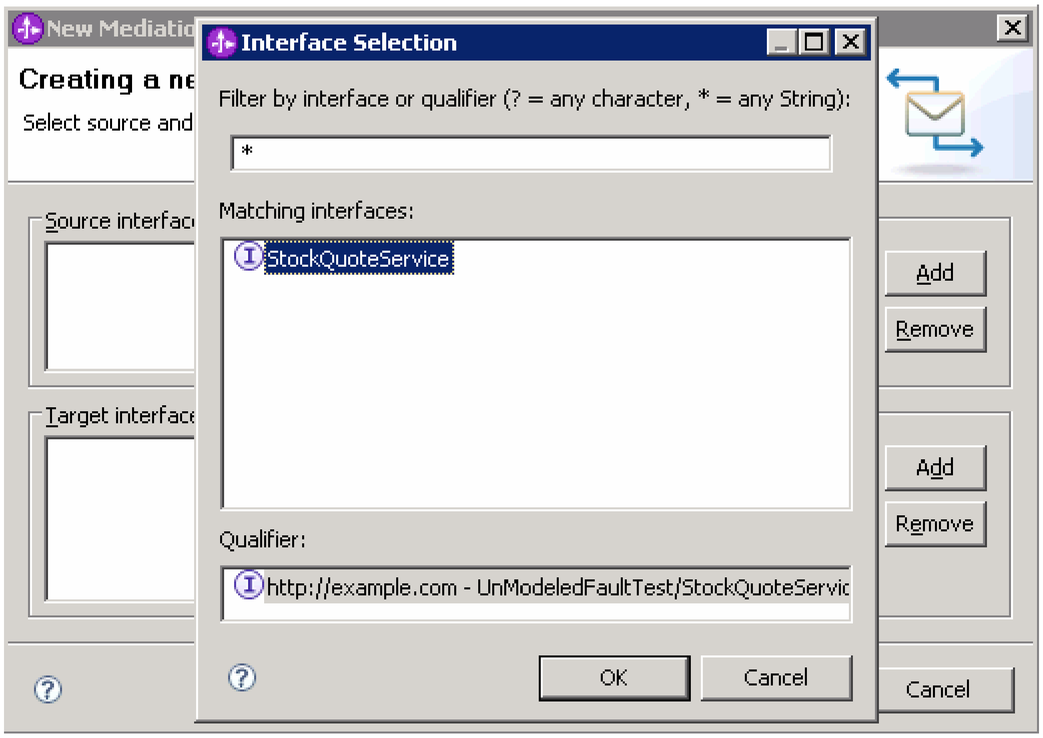
Folder:

Name: UnModeledFaultTestFlow

__ c. Click **Next**

__ d. In the '**Creating a new interface**' panel, add the source and target interfaces

1) Click the **Add** button next to the **Source interfaces** text area and select '**StockQuoteService**' from the **Interface Selection** pop-up window



Interface Selection

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

*

Matching interfaces:

StockQuoteService

Qualifier:

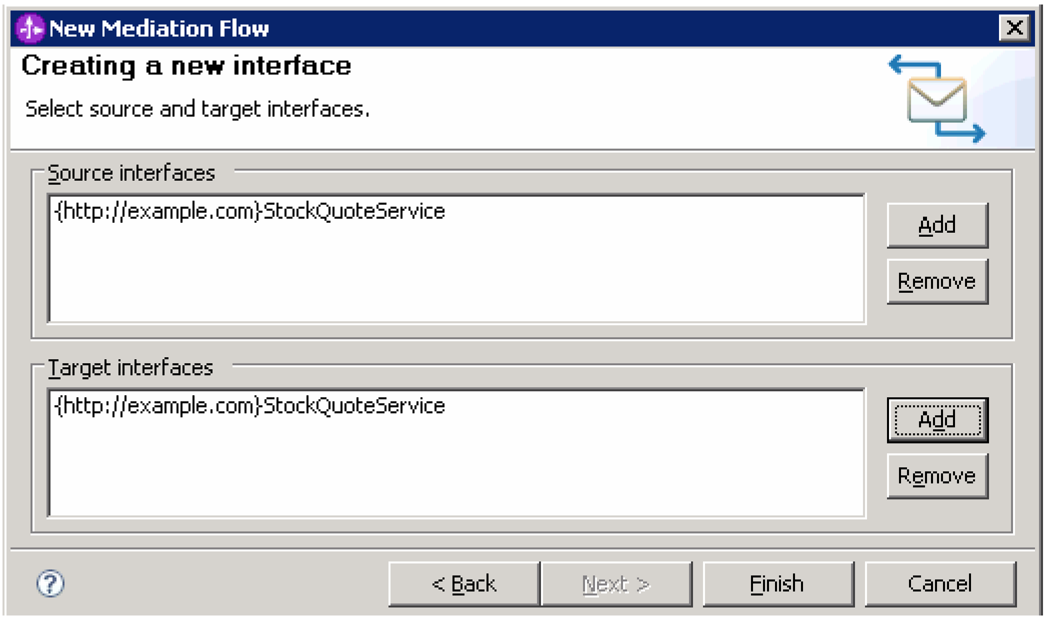
http://example.com - UnModeledFaultTest/StockQuoteServic

2) Click **OK** over the **Interface Selection** pop-up window

3) Similarly, click the **Add** button next to the **Target interfaces** text area and select ‘**StockQuoteService**’ from the **Interface Selection** pop-up panel

4) Click **OK** over the **Interface Selection** pop-up panel

5) The added source and target interfaces should look like in the picture shown below:


New Mediation Flow

Creating a new interface

Select source and target interfaces.

Source interfaces

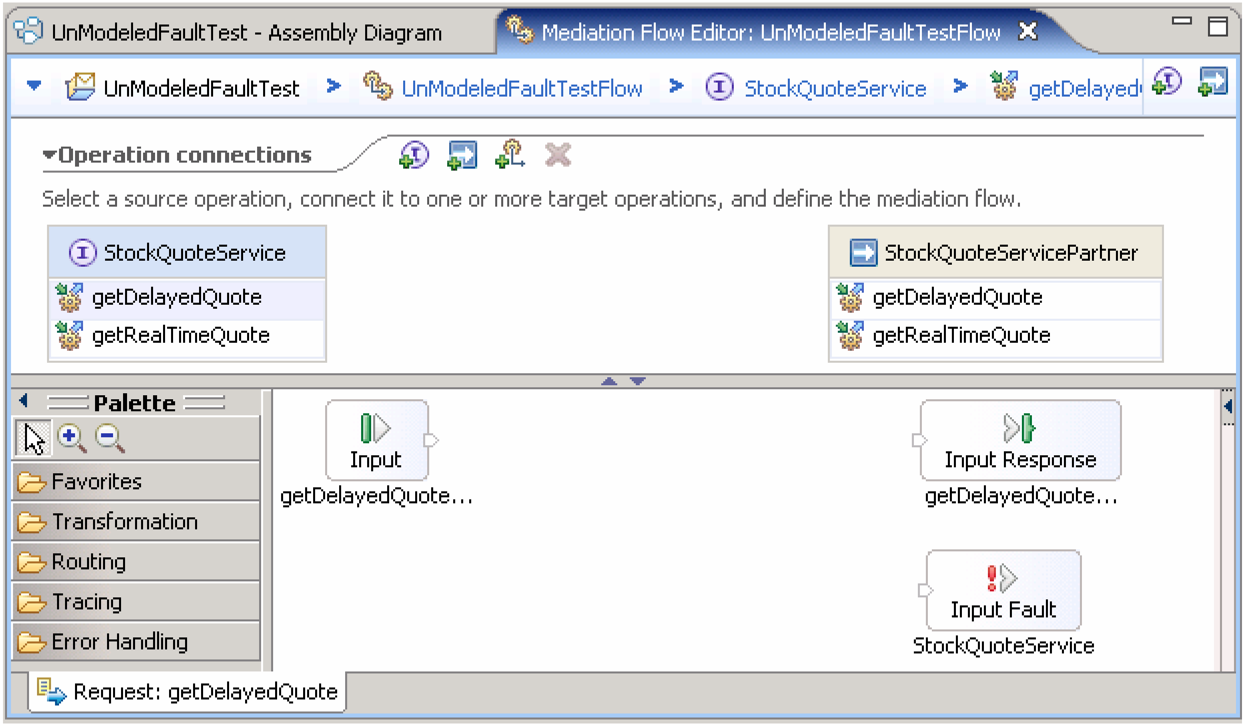
{http://example.com}StockQuoteService

Target interfaces

{http://example.com}StockQuoteService

6) Click **Finish**

__ e. Upon creation of the **UnModeledFaultTestFlow** Mediation Flow, it is opened in the Mediation Flow Editor as shown in the picture below:

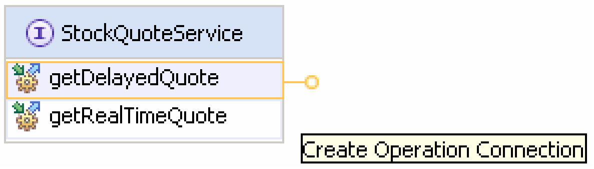


Create a request flow:

To create the request flow, complete these steps:

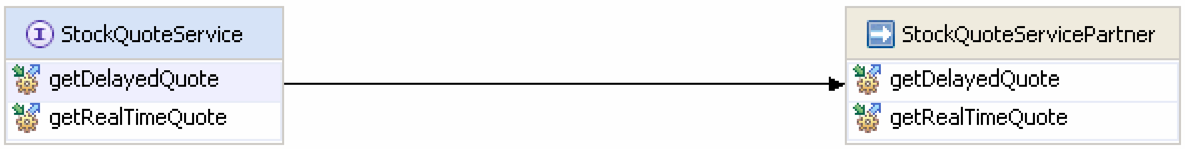
____ 2. In the 'UnModeledFaultTestFlow' mediation flow editor, connect the **source** to **target operations** in the Operation Connections view

__ a. Click anywhere on the source operation, **StockQuoteService/getDelayedQuote** on the left side of the Operation Connections view



__ b. Drag to the target operation, **StockQuoteServicePartner/getDelayedQuote** on the right side of the Operation Connections view and release the mouse click

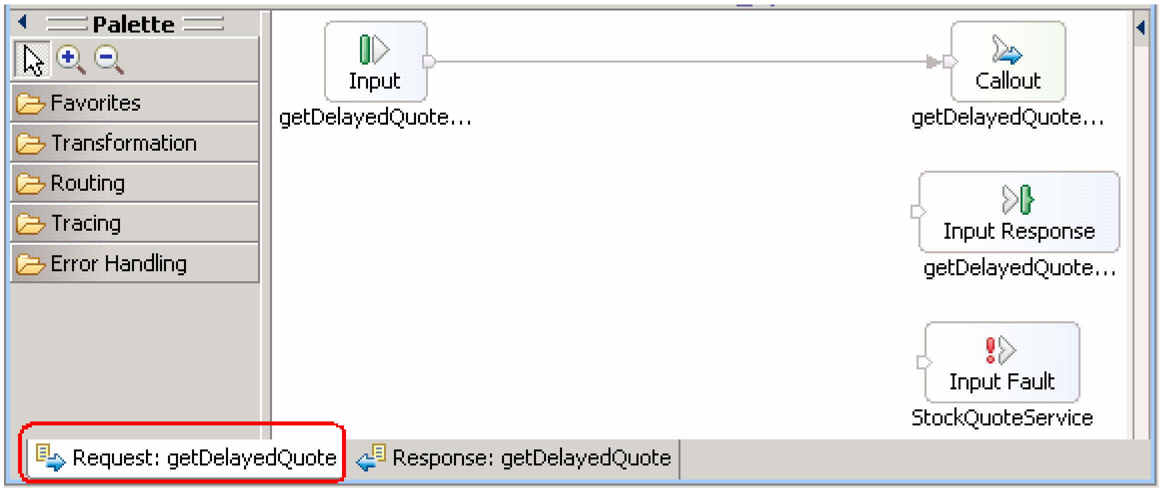
Alternative: Right-click on the source component, **StockQuoteService/getDelayedQuote** and select **Create an operation connection** and then click on the target operation, **StockQuoteServicePartner/getDelayedQuote**



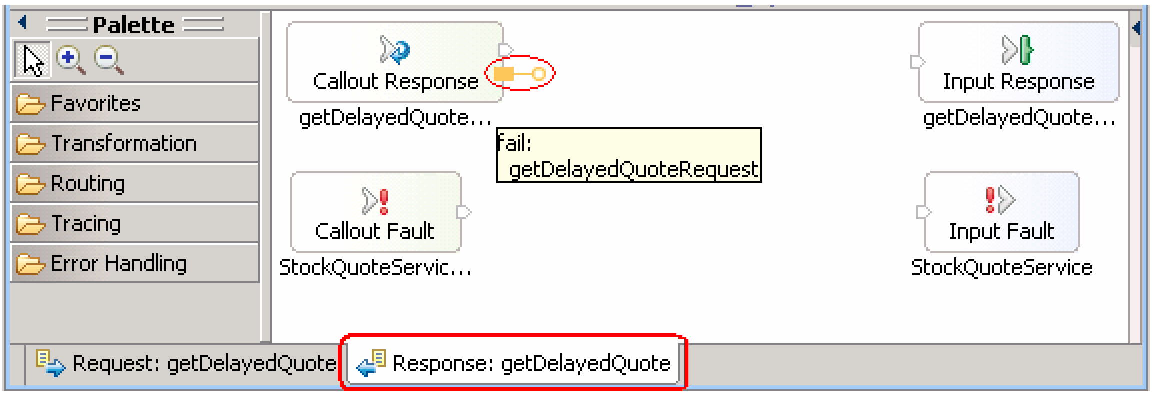
__ c. Click on the black arrow (wire) to view the Mediation Flow View and ensure the **Request** tab is selected to build the Request flow

____ 3. Drop a **Custom Mediation** primitive to the un modeled fault Response Flow canvas

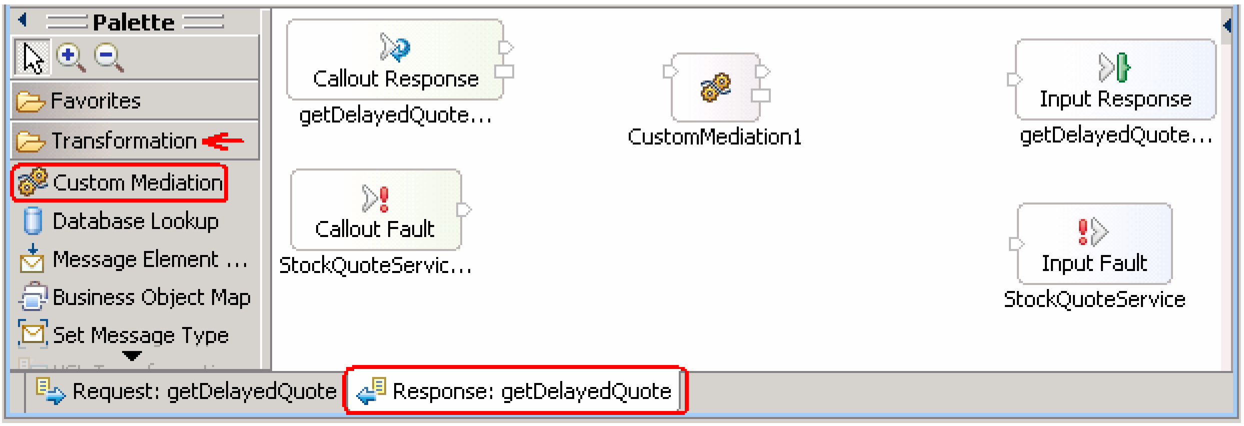
__ a. Connect the **getDelayedQuote Input** to the **getDelayedQuote Callout**. Click on the output terminal of the Input Request for getDelayedQuote and drag a wire to the input terminal of the getDelayedQuote Callout



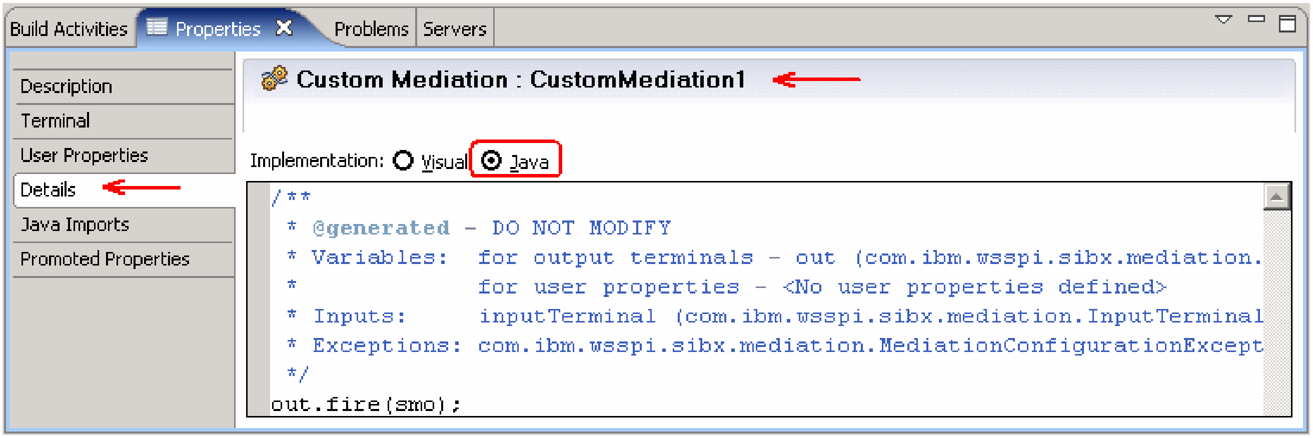
__ b. Select the **Response Tab** in the Mediation Flow diagram (middle view) to see details of the Response flows. Notice the new terminal that is used for the unmodeled fault, highlighted below with the red oval



__ c. In the **Mediation Flow Editor**(middle view), click the **Custom Mediation** icon (Custom Mediation) to select the Custom Mediation primitive from the transformation palette tray on the left side. Then drop it into the canvas between the Callout Response node and the Input Response Node



__ d. Select the **Details** tab in the '**Properties**' view (bottom window) of the **CustomMediation1**



Implementation: Visual [✓ Java]

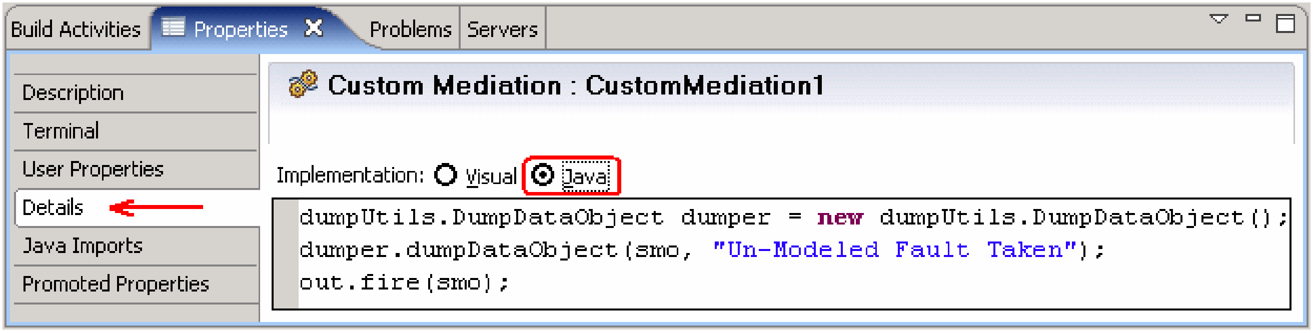
```
/**
 * @generated - DO NOT MODIFY
 * Variables:  for output terminals - out (com.ibm.wsspi.sibx.mediation.
 *             for user properties - <No user properties defined>
 * Inputs:     inputTerminal (com.ibm.wsspi.sibx.mediation.InputTerminal
 * Exceptions: com.ibm.wsspi.sibx.mediation.MediationConfigurationExcept
 */
out.fire(smo);
```

__ e. A 'DumpUtility' was included in the Project Interchange file first imported. Call it from the **CustomMediation1** to dump the contents of the Service Message Object (SMO). To do this, add following code to the '**Java**' implementation section of **CustomMediation1**:

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

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

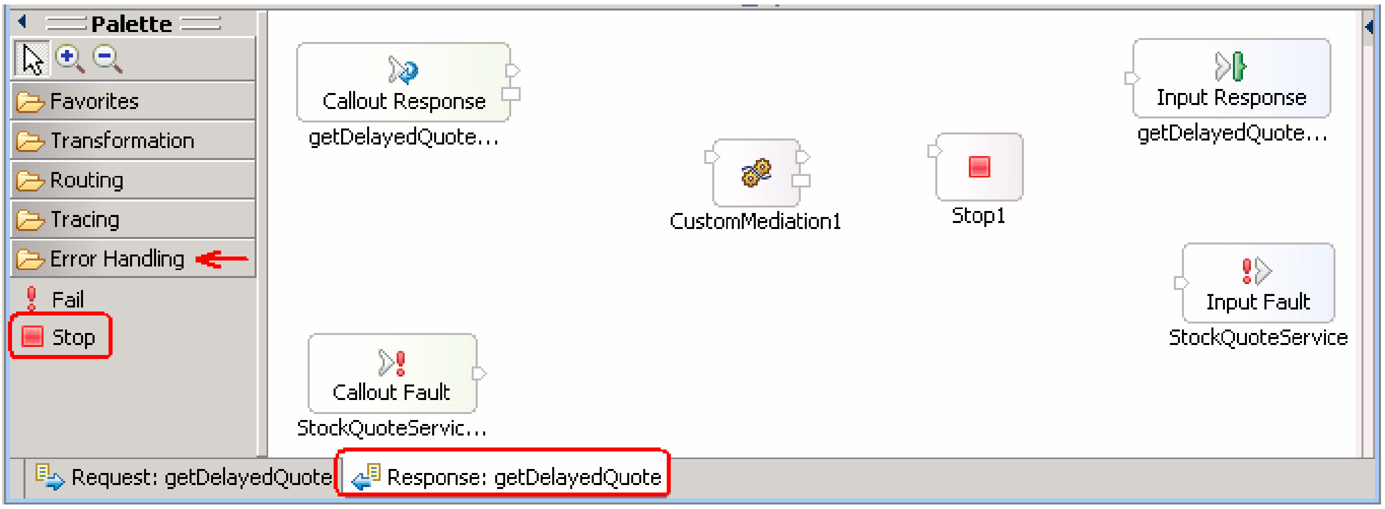
<LAB_FILES>\import\DumpUtilSnippet.txt



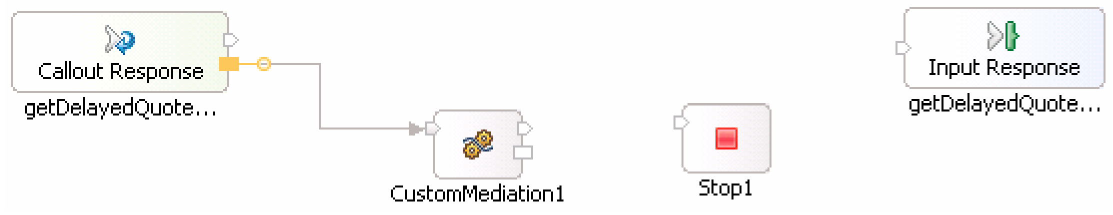
Implementation: Visual [✓ Java]

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

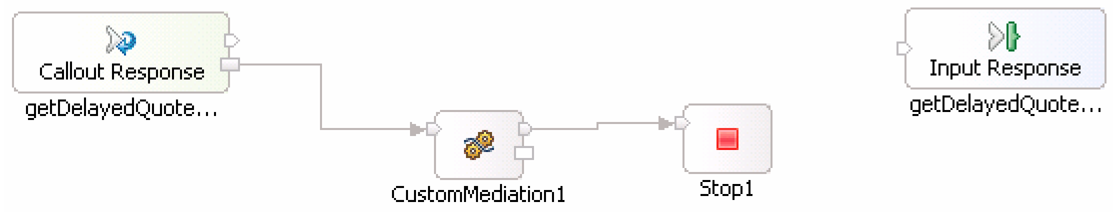
__ f. Drop a **Stop Mediation** primitive Response Flow canvas. Click the **Stop** icon (🟥 Stop) to select the Stop primitive from error handling palette tray on the left side and drop it into the right of **CustomMediation1** primitive as shown below:



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



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

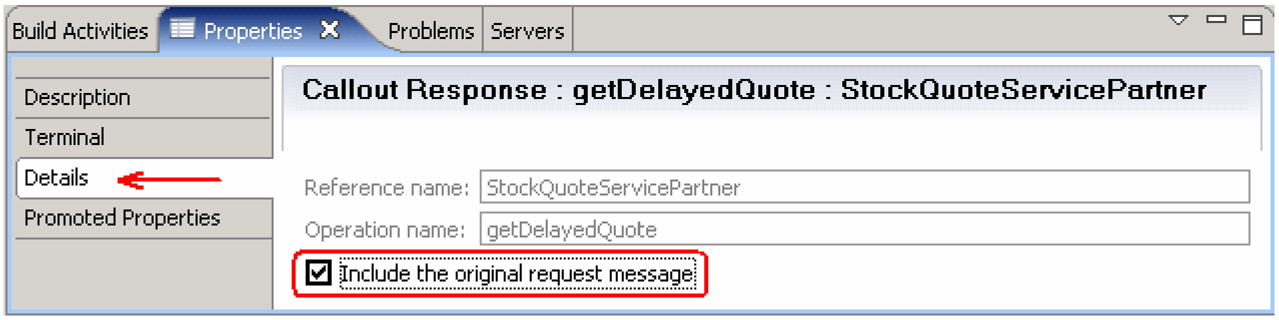


____ 4. Specify that the original request message should be included when an unmodeled fault is taken.

__ a. Select the **getDelayedQuote Callout Response**.

__ b. Select the **Details** tab in the properties view for **getDelayedQuote**.

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



Callout Response : getDelayedQuote : StockQuoteServicePartner

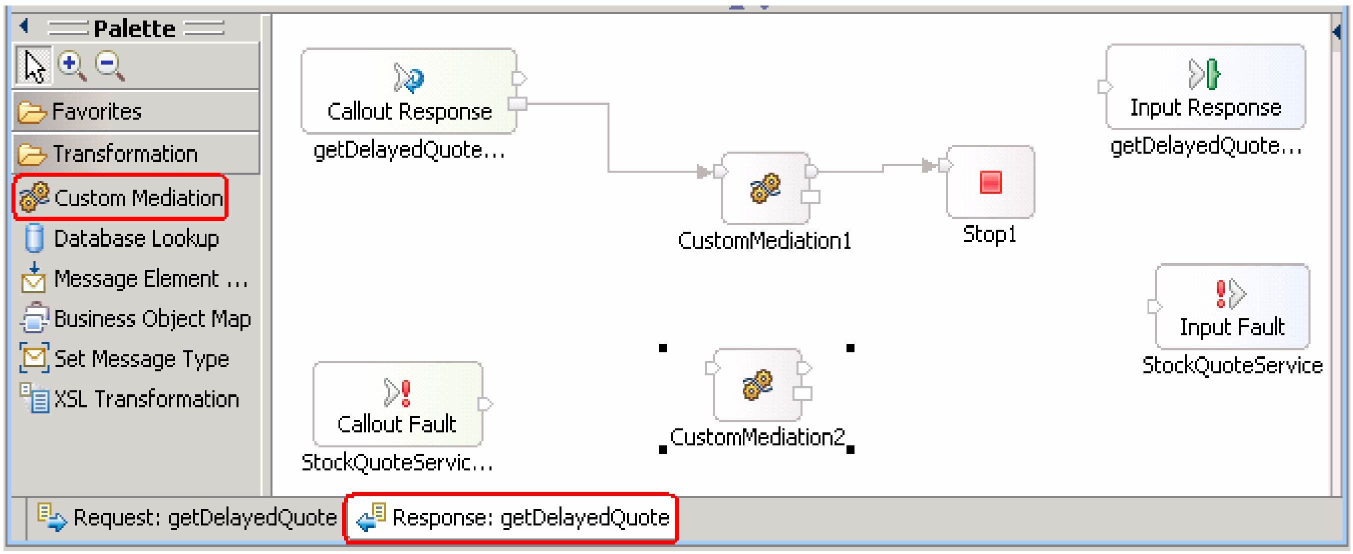
Reference name: StockQuoteServicePartner

Operation name: getDelayedQuote

☑ Include the original request message

____ 5. Add a Custom Mediation primitive to the modeled fault Response Flow canvas

__ a. Ensure the Response tab is selected. Click the **Custom Mediation** icon (Custom Mediation) in the transformation palette tray (left-hand side) and then click to the right of the **Callout Fault** terminal (the modeled fault) as shown below:



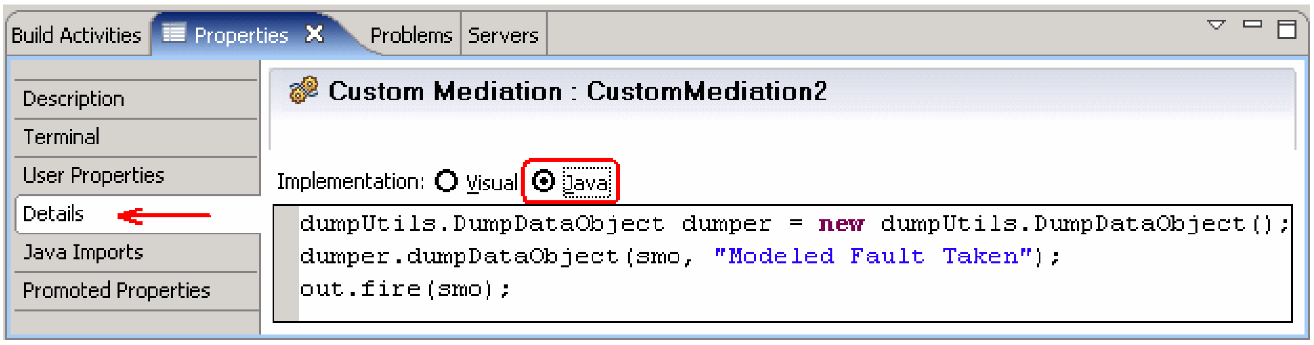
__ b. Select the **CustomMediation2** primitive and then the **Details** tab in the properties view.

__ c. Call the ‘DumpUtility’ from the **CustomMediation2** to dump the contents of the Service Message Object (SMO). To do this, copy and paste this Java snippet to the ‘**Java**’ **implementation** text area

```
dumpUtils.DumpDataObject dumper = new
dumpUtils.DumpDataObject();
dumper.dumpDataObject(smo, "Modeled Fault Taken");
out.fire(smo);
```

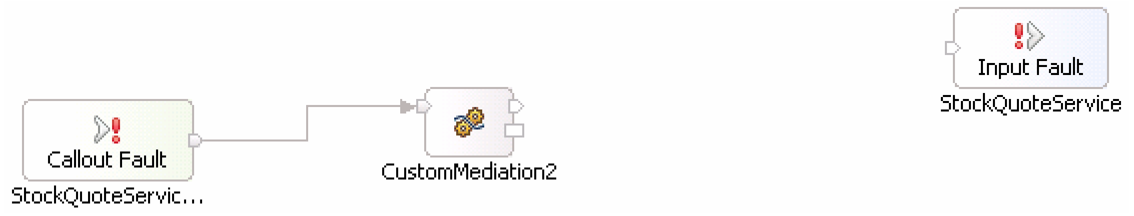
A generic form of this code that can be cut and pasted and edited is included in this file:

<LAB_FILES>\import\DumpUtilSnippet.txt

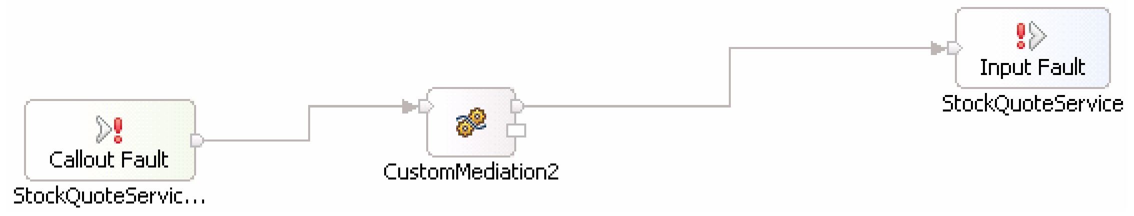


```
dumpUtils.DumpDataObject dumper = new dumpUtils.DumpDataObject();
dumper.dumpDataObject(smo, "Modeled Fault Taken");
out.fire(smo);
```

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

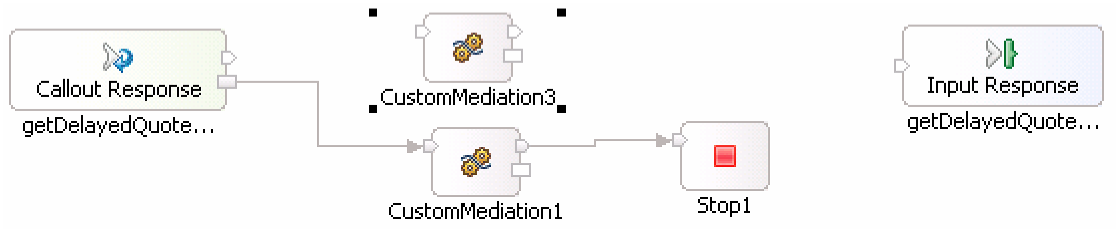


__ e. Click on the **output terminal** of the **CustomMediation2** primitive and drag a wire to the **input terminal** of the **StockQuoteService Input Fault**



____ 6. Add a Custom Mediation primitive to the non-fault Response Flow canvas

__ a. Ensure the Response tab is selected. Click the **Custom Mediation** icon (Custom Mediation) in the transformation palette tray (left-hand side) and then click to the right of the **Callout Response output terminal** as shown below:



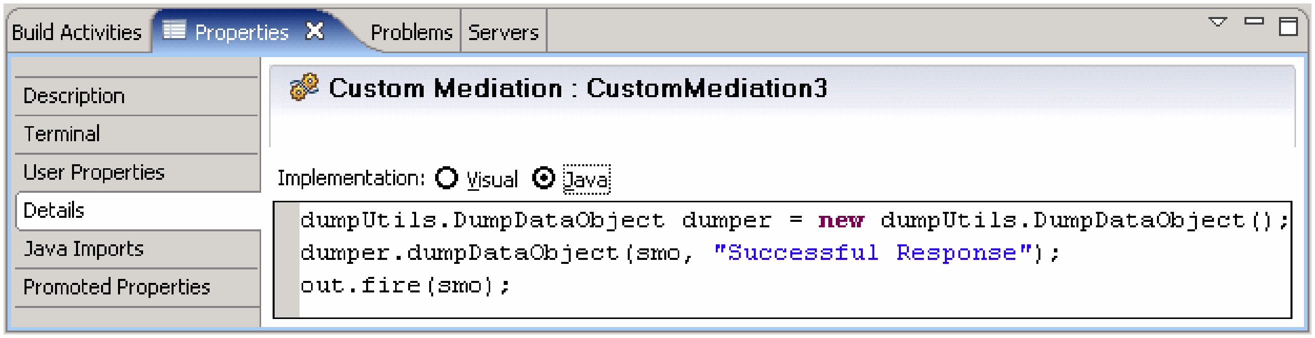
__ b. Select the **CustomMediation3** primitive and then select the **Details** tab in the properties view.

__ c. Call the 'DumpUtility' from the **CustomMediation3** to dump the contents of the Service Message Object (SMO). To do this, copy and paste this Java snippet to the '**Java**' implementation

```
dumpUtils.DumpDataObject dumper = new
dumpUtils.DumpDataObject();
dumper.dumpDataObject(smo, "Successful Response");
out.fire(smo);
```

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

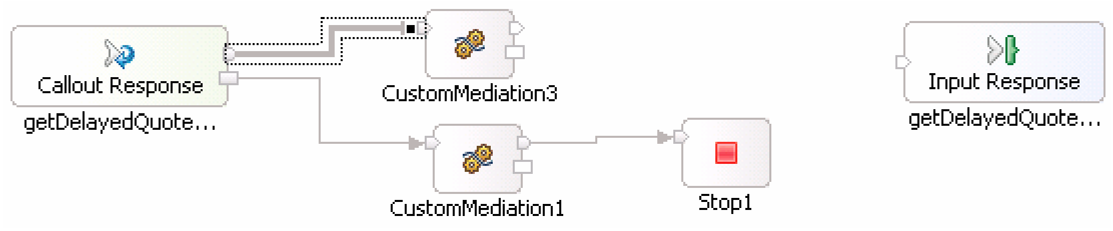
<LAB_FILES>\import\DumpUtilSnippet.txt



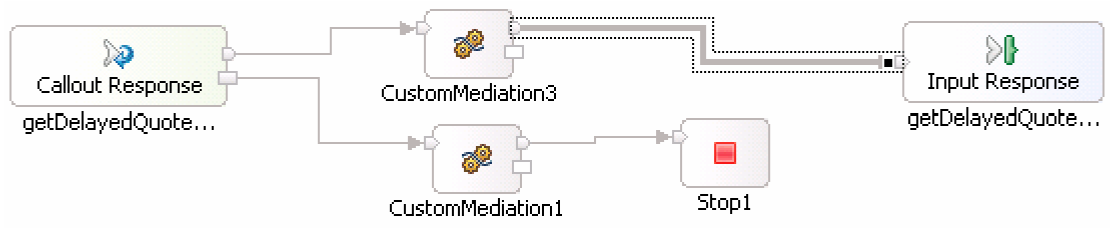
Implementation: Visual [✓ Java]

```
dumpUtils.DumpDataObject dumper = new dumpUtils.DumpDataObject();
dumper.dumpDataObject(smo, "Successful Response");
out.fire(smo);
```

__ 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 (**File → Save All** or **Crtl + Shift + S**)

Part 5: Assemble the module

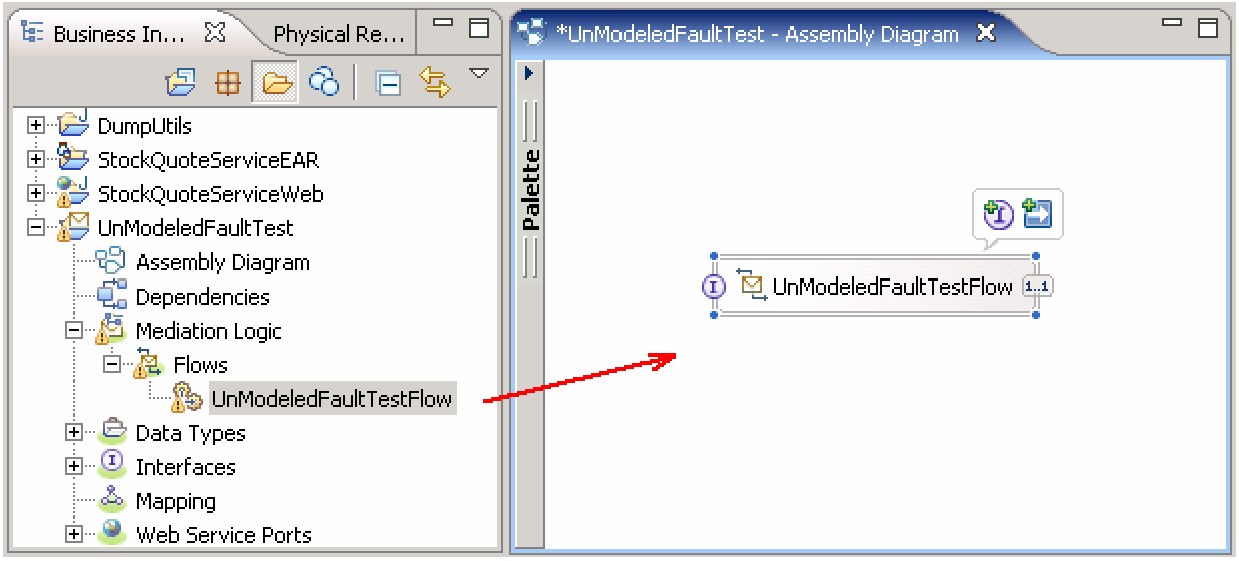
In this section of the lab, you will assemble the unModeledFaultTest Module, wiring an export to targets and define an SCA interface for the business process. As part of the assembly process, the appropriate deployment code will also be generated in preparation for running the business process as a service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

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

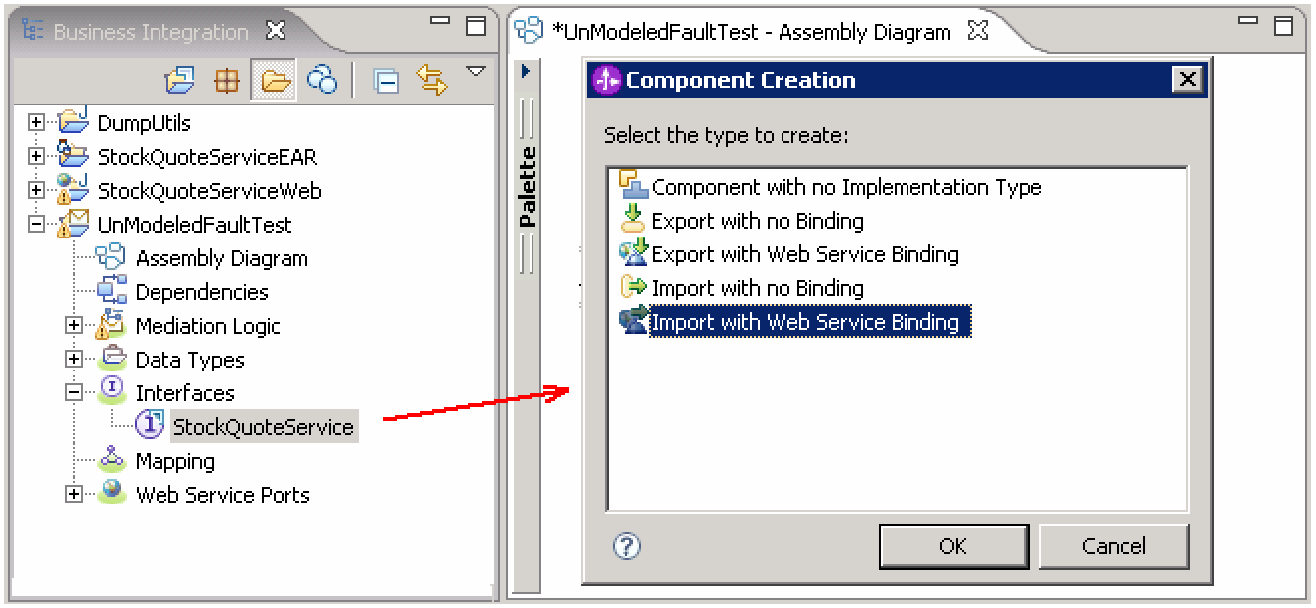
__ a. In the Business Integration view, expand the **UnModeledFaultTest** mediation module double-click the mediation module assembly (Assembly Diagram) to open it with the assembly editor

__ b. An empty assembly editor for the **UnModeledFaultTest** mediation module is opened

__ c. In the Business Integration view, expand the **UnModeledFaultTest → Mediation Logic → Flows**; select **UnModeledFaultTestFlow** mediation flow and then drag it over the assembly editor's canvas as shown below:



__ d. In the Business Integration view, expand the **UnModeledFaultTest → Interfaces**; select **StockQuoteService** an then drag it over the assembly editor's canvas



Component Creation

Select the type to create:

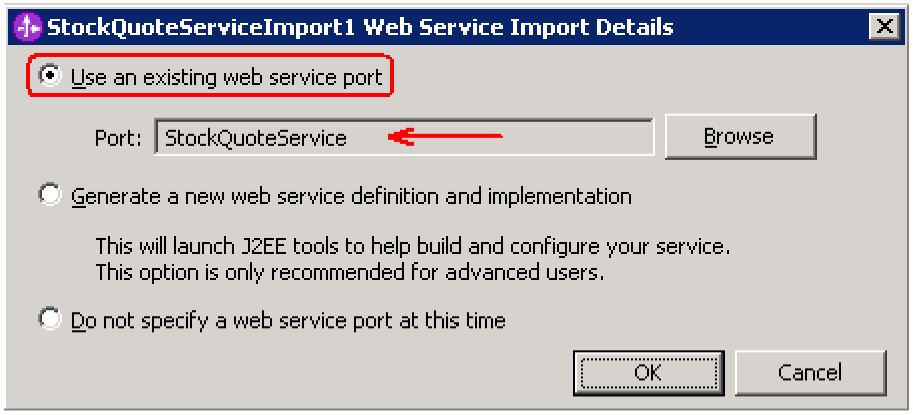
- Component with no Implementation Type
- Export with no Binding
- Export with Web Service Binding
- Import with no Binding
- Import with Web Service Binding

__ e. Select **Import with Web Service Binding** from the 'Component Creation' dialog and click **OK**

__ f. The '**Web Service Import Details**' panel pops up. Complete these steps:

1) Ensure the radio button for '**Use an existing web service port**' is selected

2) Click the **Browse** button for '**Port**' filed and select the '**StockQuoteService**' from the port selection panel



StockQuoteServiceImport1 Web Service Import Details

[✓ Use an existing web service port]

Port: StockQuoteService

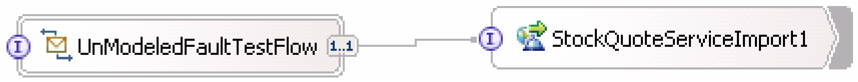
Generate a new web service definition and implementation

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

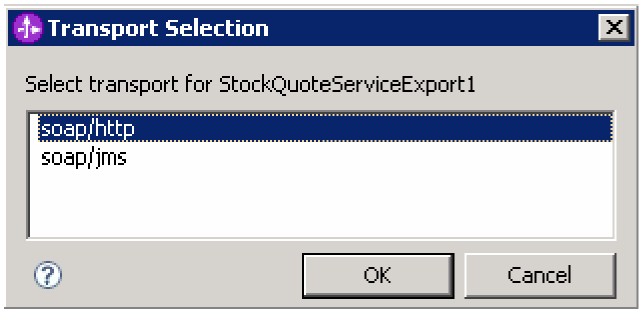
3) Click **OK**

__ g. Wire the **UnModeledFaultTest** reference to the **StockQuoteServiceImport1** as shown below:



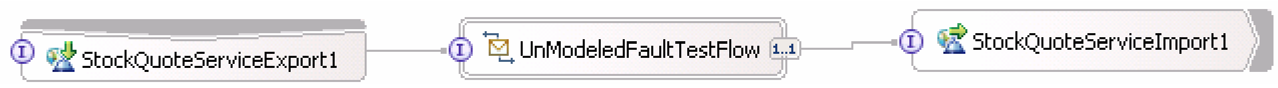
__ h. Right click **UnModeledFaultTestFlow** mediation flow and select **Generate Export → Web Services Binding** from the context menu

1) Select '**soap/http**' from the 'Transport Selection' panel



2) Click **OK**

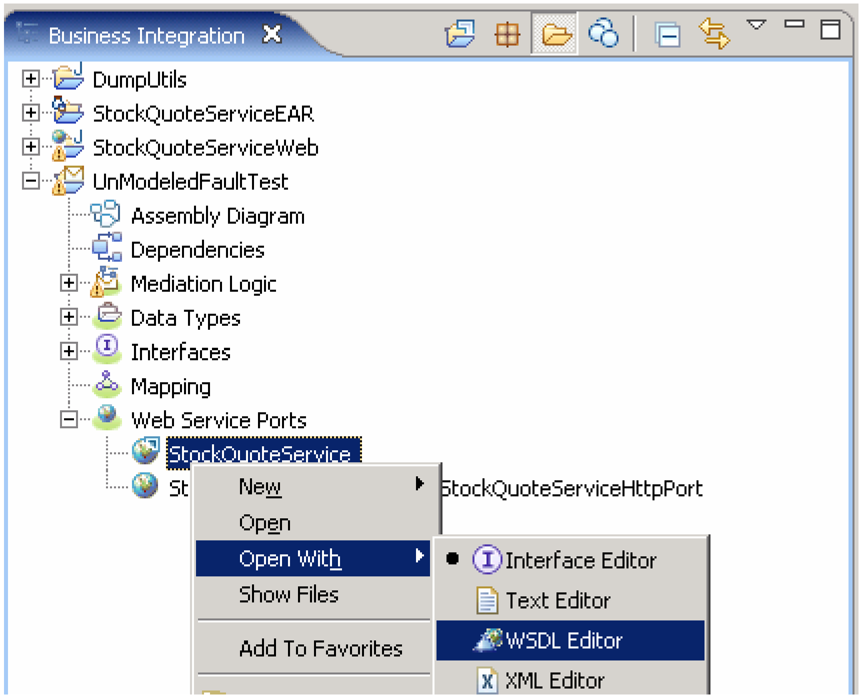
__ i. The final assembly diagram should look like in the picture below:



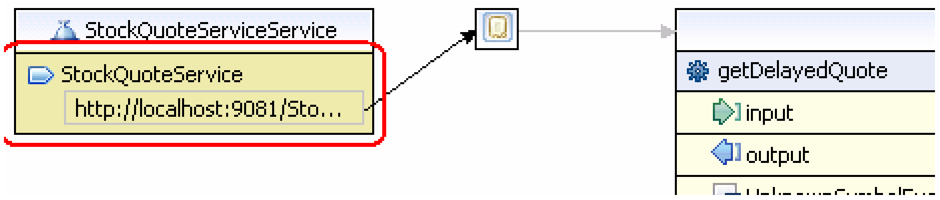
__ j. Save all work (**File → Save All** or **Crtl + Shift + S**)

____ 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.

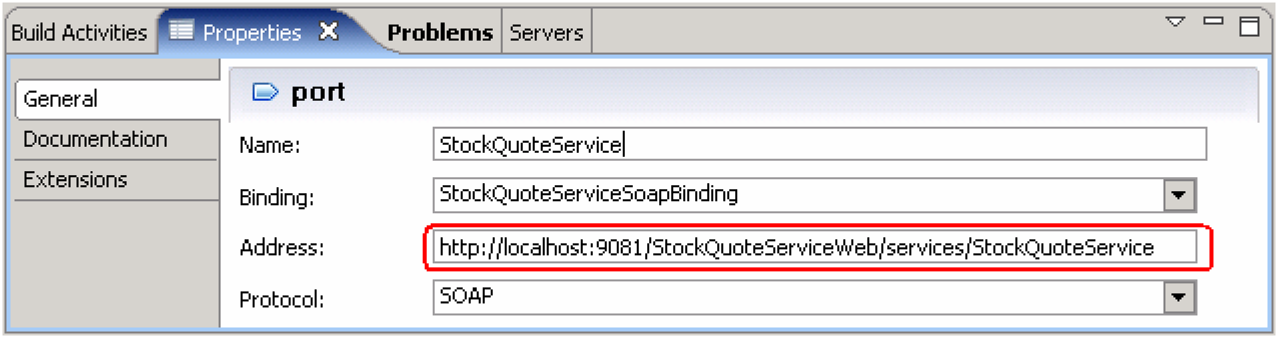
__ a. In the Business Integration view, expand '**UnModeledFaultTest → Web Service Ports**', right-click on **StockQuoteService** and select **Open With → WSDL Editor** from the context menu



1) In the WSDL editor, select the **StockQuoteService** and then the '**General**' tab under properties view



2) Update the address (Ex:- **<HOSTNAME>:<PORT>**) to match your environment.



Name: StockQuoteService

Binding: StockQuoteServiceSoapBinding

Address: http://localhost:9081/StockQuoteServiceWeb/services/StockQuoteService

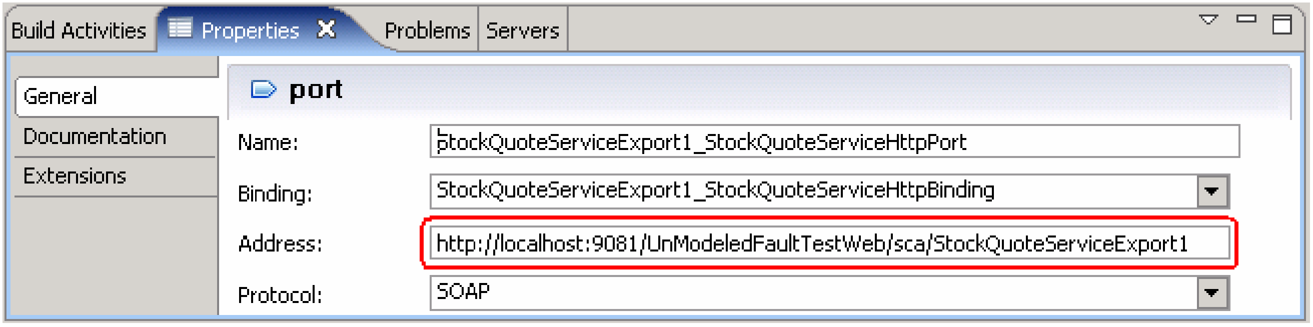
Protocol: SOAP

3) Press **Ctrl+S** to save. Close the WSDL editor.

__ b. In the Business Integration view, expand '**UnModeledFaultTest → Web Service Ports**', right-click on **StockQuoteServiceExport1_StockQuoteServiceHttpPort** and select **Open With → WSDL Editor** from the context menu

1) In the WSDL editor, select the **StockQuoteServiceExport1_StockQuoteServiceHttp** and then the '**General**' tab under properties view

2) Update the address (Ex:- **<HOSTNAME>:<PORT>**) to match your environment



Name: StockQuoteServiceExport1_StockQuoteServiceHttpPort

Binding: StockQuoteServiceExport1_StockQuoteServiceHttpBinding

Address: http://localhost:9081/UnModeledFaultTestWeb/sca/StockQuoteServiceExport1

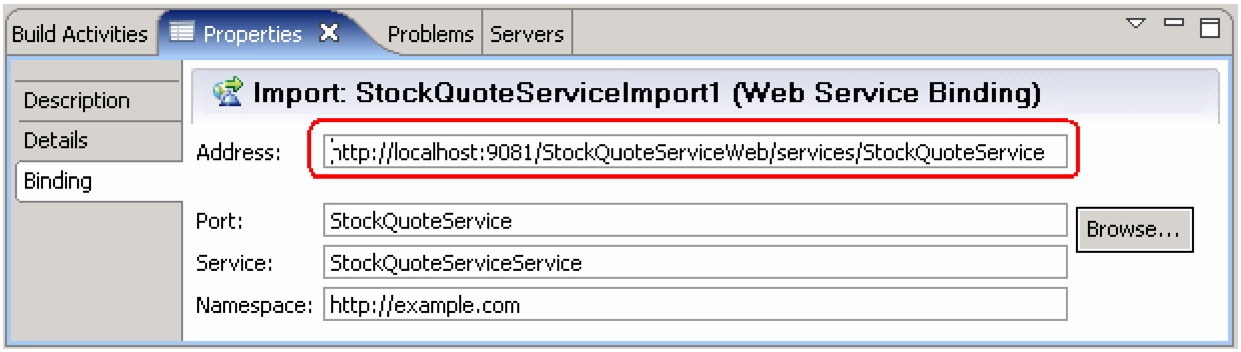
Protocol: SOAP

3) Press **Ctrl+S** to save. Close the WSDL editor.

__ c. In the Business Integration view, expand the **UnModeledFaultTest** mediation module double-click the mediation module assembly (Assembly Diagram) to open it with the assembly editor

1) Select the **StockQuoteServiceImport1** import on the Assembly Diagram and select the '**Bindings**' under the **Properties** view

2) Update the address (Ex:- **<HOSTNAME>:<PORT>**) to match your environment



Import: StockQuoteServiceImport1 (Web Service Binding)

Address: http://localhost:9081/StockQuoteServiceWeb/services/StockQuoteService

Port: StockQuoteService

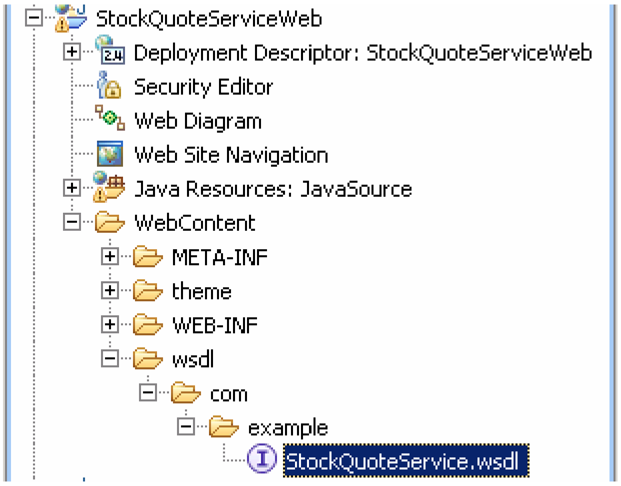
Service: StockQuoteServiceService

Namespace: http://example.com

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

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

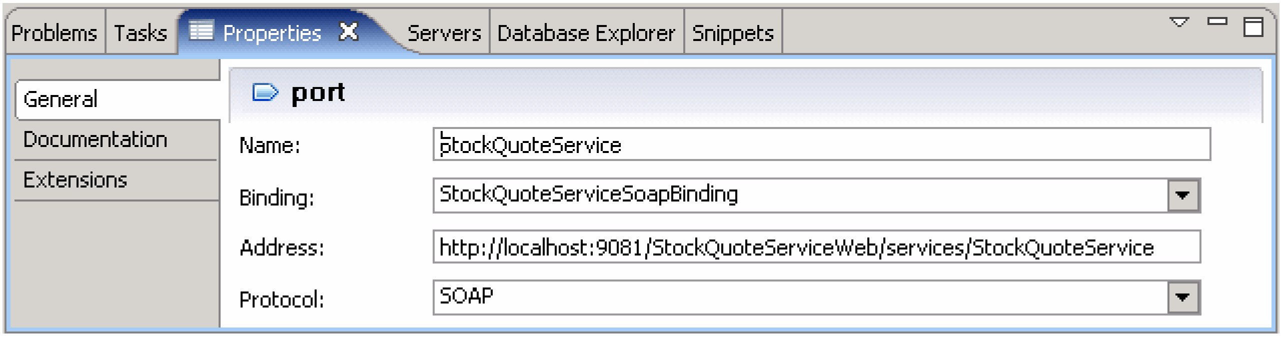
1) In the Project Explorer view, expand **StockQuoteServiceWeb → WebContent → wsdl → com → example**; right-click on **StockQuoteService.wsdl** and select **Open With → WSDL Editor** from the pop-up menu



```
StockQuoteServiceWeb
  Deployment Descriptor: StockQuoteServiceWeb
  Security Editor
  Web Diagram
  Web Site Navigation
  Java Resources: JavaSource
  WebContent
    META-INF
    theme
    WEB-INF
    wsdl
      com
        example
          StockQuoteService.wsdl
```

2) In the WSDL editor, select **StockQuoteService** and then the '**General**' tab under properties view

3) Update the address (Ex:- **<HOSTNAME>:<PORT>**) to match your environment



port

Name: StockQuoteService

Binding: StockQuoteServiceSoapBinding

Address: http://localhost:9081/StockQuoteServiceWeb/services/StockQuoteService

Protocol: SOAP

__ f. Press **Ctrl+S** to save. Close the WSDL editor.

Part 6: Add the modules to the server

In this part of the lab, the Mediation Module is published to the WebSphere ESB Server 6.1 test server.

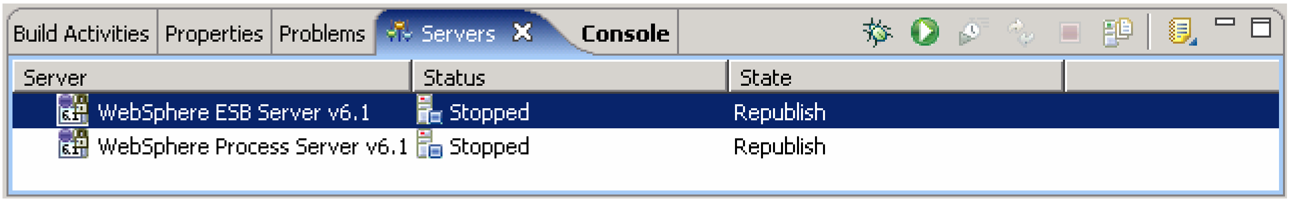
____ 1. Start the **WebSphere ESB Server** if not started and **add modules** to the server

If using a remote testing environment, follow the instructions in **Task: Adding remote server to WebSphere Integration Developer test environment** at the end of this document, to start the remote server.

If using a local ESB Integrated test environment, complete the steps below:

__ a. Open **Servers View**.

__ b. Select the '**WebSphere ESB Server v6.1**' and right-click to select "( ) **Start**" from the context menu



Server	Status	State
WebSphere ESB Server v6.1	Stopped	Republish
WebSphere Process Server v6.1	Stopped	Republish

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

____ 2. Add the **projects** to WebSphere ESB Server. In the servers view, right-click on **WebSphere ESB Server v6.1** and select '**Add and Remove Projects...**' from the context menu

Note: WebSphere ESB server that is being used is configured with an ESB profile that is part of the installation and not part of the workspace. Therefore, if there are any 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/uninstall those projects before adding these projects. This should avoid any potential errors.

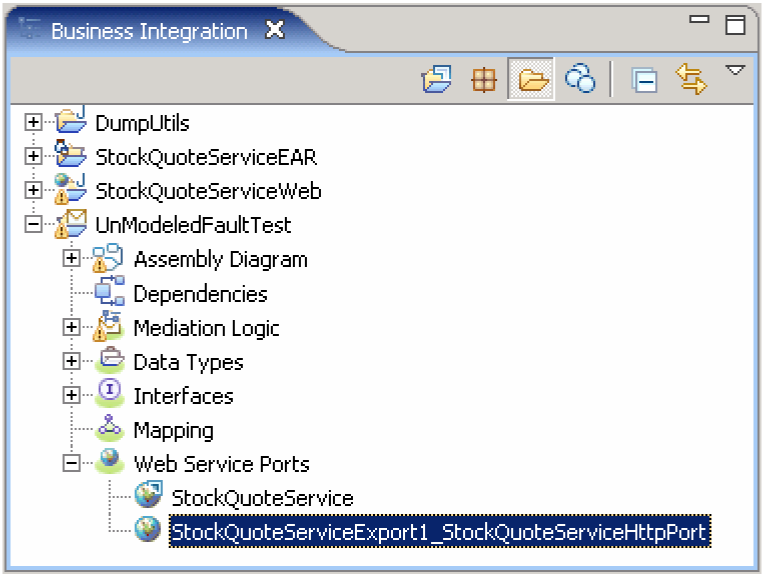
__ a. Click the **Add-All>>** button to move all projects to server

__ b. Click **Finish**

__ c. Wait for the deployment to finish.

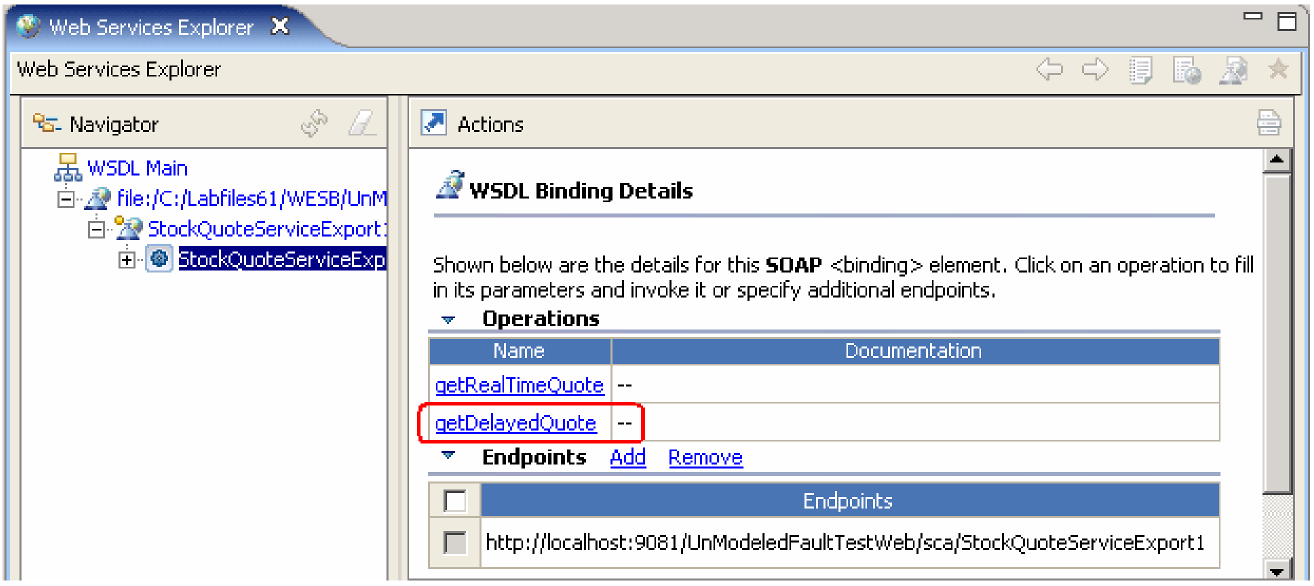
Part 7: Test the mediations using the Web services explorer

____ 1. In the Business Integration perspective, expand '**UnModeledFaultTest → Web Service Ports**'



____ 2. Right-click on ***StockQuoteServiceExport1_StockQuoteServiceHttpPort*** and then select '**Web Services → Test with Web Services Explorer**' from the context menu

____ 3. The Web Services Explorer opens the ***UnModeledFaultTest_StockQuoteServiceExport1.wsdl*** file as shown below:



WSDL Binding Details

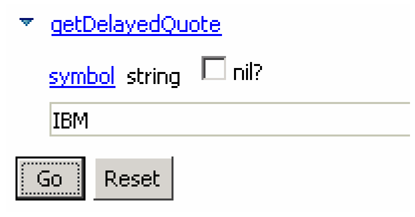
Shown below are the details for this **SOAP** <binding> element. Click on an operation to fill in its parameters and invoke it or specify additional endpoints.

Name	Documentation
getRealTimeQuote	--
getDelayedQuote	--

	Endpoints
☐	http://localhost:9081/UnModeledFaultTestWeb/sca/StockQuoteServiceExport1

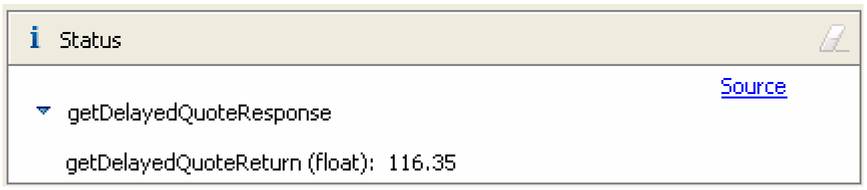
____ 4. Click the '**getDelayedQuote**' entry

__ a. Input **IBM** as the symbol



__ b. Click the '**Go**' button

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



```
getDelayedQuoteResponse
getDelayedQuoteReturn (float): 116.35
```

__ 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.

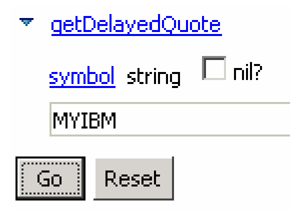
```
O ############ Start DataObject Dump ############
O User Supplied Comment = Successful Response
```

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

```
O |Property: body
O |----|### Start DO Dump ###
O |----|Type -> getDelayedQuoteResponse
O |----|Property: parameters
O |----|----|### Start DO Dump ###
O |----|----|Type -> getDelayedQuoteResponse_._type
O |----|----|Property: getDelayedQuoteReturn
O |----|----|    Type:  Float
O |----|----|    Value: 116.35
O |----|----|### End   DO Dump ###
O |----|### End   DO Dump ###
O |### End   DO Dump ###
O ############ End  DataObject Dump ############
```

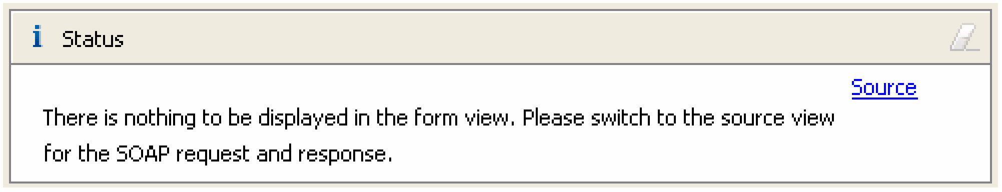
____ 5. Test the flow with an unknown symbol.

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



__ b. Click the '**Go**' button

__ 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.

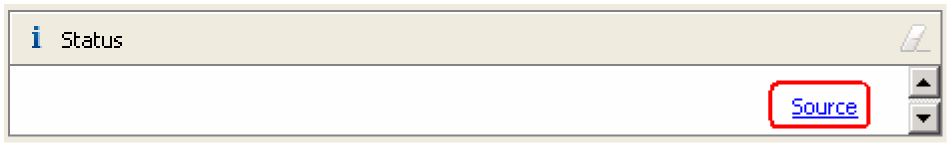
__ d. Look in the console log at the SMO that was dumped as part of the Custom Mediation. Verify that it went down the 'Modeled Fault' path

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

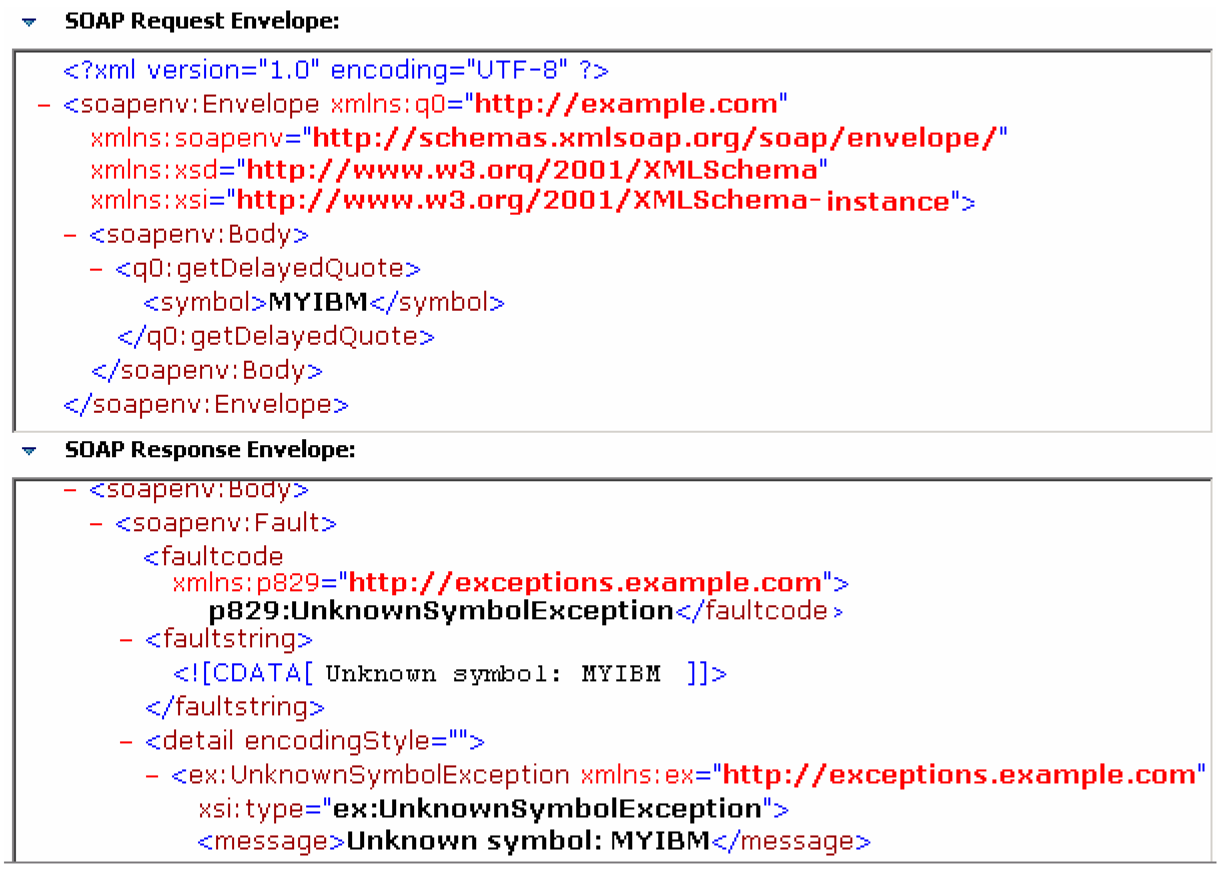
__ e. Inspect the rest of the SMO and see that the response returned the modeled UnknownSymbolException

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

```
  <?xml version="1.0" encoding="UTF-8" ?>
- <soapenv:Envelope xmlns:q0="http://example.com"
    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:Body>
    - <q0:getDelayedQuote>
        <symbol>MYIBM</symbol>
      </q0:getDelayedQuote>
    </soapenv:Body>
  </soapenv:Envelope>
```

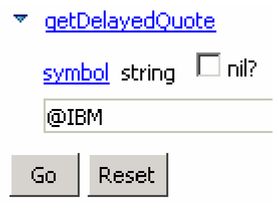
SOAP Response Envelope:

```
  - <soapenv:Body>
    - <soapenv:Fault>
        <faultcode
          xmlns:p829="http://exceptions.example.com">
            p829:UnknownSymbolException</faultcode>
      - <faultstring>
          <![CDATA[ Unknown symbol: MYIBM ]]>
        </faultstring>
      - <detail encodingStyle="">
        - <ex:UnknownSymbolException xmlns:ex="http://exceptions.example.com"
            xsi:type="ex:UnknownSymbolException">
            <message>Unknown symbol: MYIBM</message>
```

__ 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



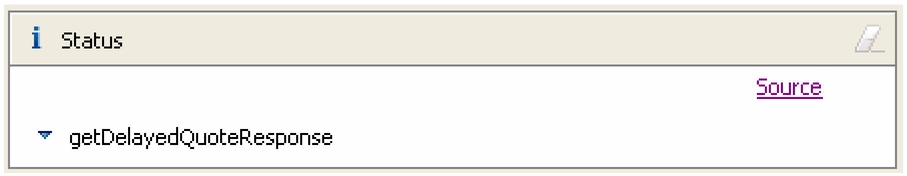
getDelayedQuote

symbol string ☐ nil?

@IBM

__ b. Click the '**Go**' button

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

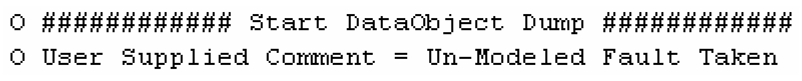


Status

Source

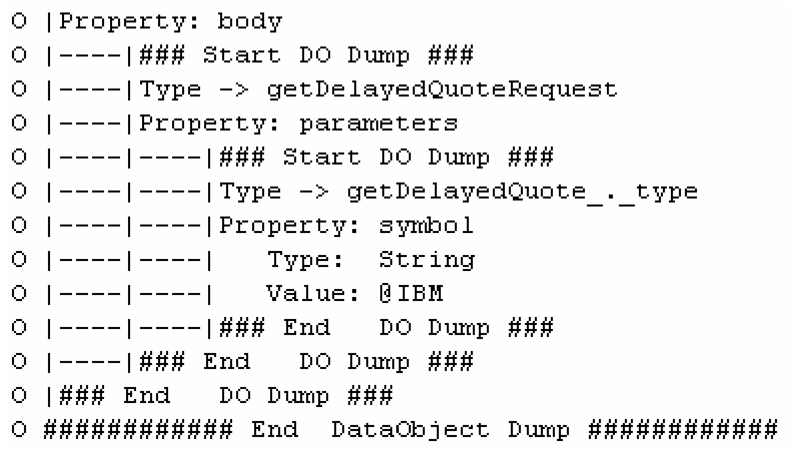
getDelayedQuoteResponse

__ d. 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



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

__ e. 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.



```
O |Property: body
O |----|### Start DO Dump ###
O |----|Type -> getDelayedQuoteRequest
O |----|Property: parameters
O |----|----|### Start DO Dump ###
O |----|----|Type -> getDelayedQuote_._type
O |----|----|Property: symbol
O |----|----|   Type:  String
O |----|----|   Value: @IBM
O |----|----|### End   DO Dump ###
O |----|### End   DO Dump ###
O |### End   DO Dump ###
O ############ End  DataObject 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 is 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>\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**

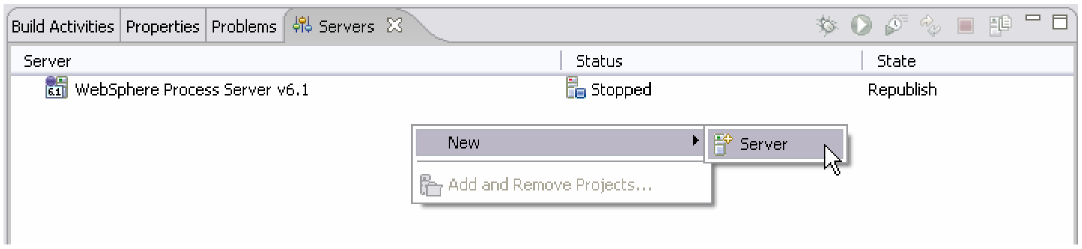
Task: Adding remote server to WebSphere Integration Developer test environment

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

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

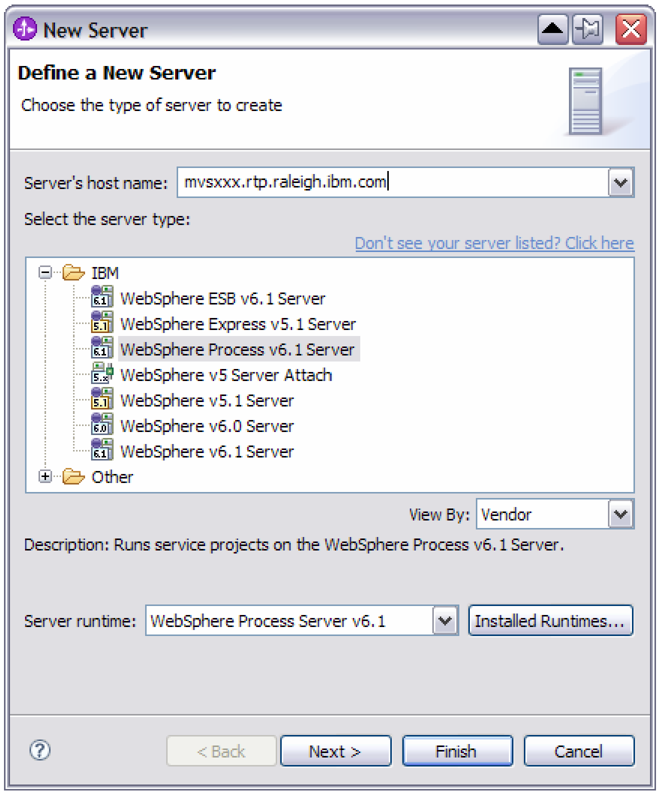
__ a. Right click on the background of the **Servers** view to access the pop-up menu.

__ b. Select **New > Server**



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

__ d. Ensure that the appropriate server type, '**WebSphere Process v6.1 Server**' or '**WebSphere ESB v6.1 Server**', is highlighted in the server type list


New Server
Define a New Server
Choose the type of server to create
Select the server type:
Don't see your server listed? Click here
IBM
WebSphere ESB v6.1 Server
WebSphere Express v5.1 Server
WebSphere Process v6.1 Server
WebSphere v5 Server Attach
WebSphere v5.1 Server
WebSphere v6.0 Server
WebSphere v6.1 Server
Other
Installed Runtimes...
< Back
Next >
Finish
Cancel

Server's host name: mvsxxx.rtp.raleigh.ibm.com

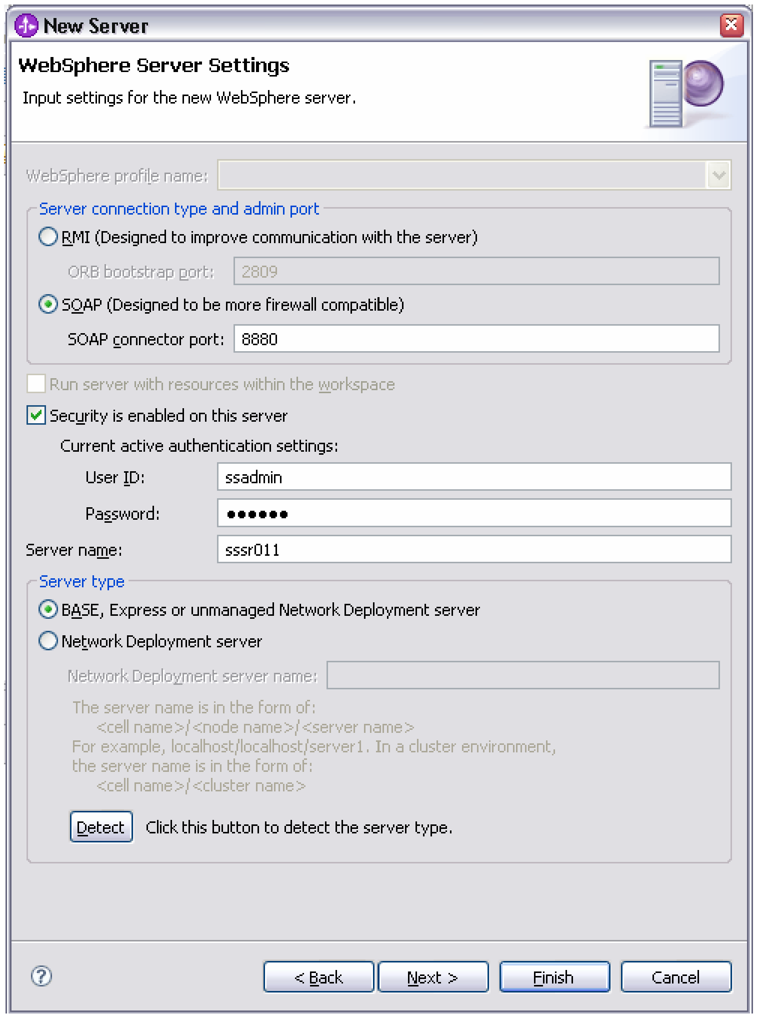
View By: Vendor

Description: Runs service projects on the WebSphere Process v6.1 Server.

Server runtime: WebSphere Process Server v6.1

__ e. Click **Next**.

__ f. On the WebSphere Server Settings page, leave the radio button for **SOAP** selected, changing the **SOAP connector port** to the correct setting (**<SOAP_PORT>**). If security is on in your server, check the box for **'Security is enabled on this server'** and input **<USERID>** for the user ID and **<PASSWORD>** for the password.


New Server

WebSphere Server Settings

Input settings for the new WebSphere server.

WebSphere profile name:

Server connection type and admin port

( ) RMI (Designed to improve communication with the server)

ORB bootstrap port: 2809

(●) SOAP (Designed to be 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: ssadmin

Password: ●●●●●●

Server name: sssr011

Server type

(●) BASE, Express or unmanaged Network Deployment server

( ) Network Deployment server

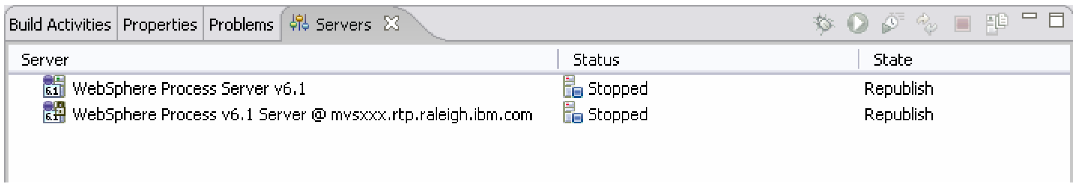
Network Deployment server name:

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

Detect Click this button to detect the server type.

__ g. Click **Finish**

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



Server	Status	State
WebSphere Process Server v6.1	Stopped	Republish
WebSphere Process v6.1 Server @ mvsxxx.rtp.raleigh.ibm.com	Stopped	Republish

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

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

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

User ID: **<USERID>**

Password: **<PASSWORD>**

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

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

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

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

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

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