



# **SupportPac LA71: IBM Operational Decision Manager Integration for WebSphere Process Server**

## **Getting started with IBM Business Process Manager**

### **Task 6 – Integration Designer assembles a BPEL Process containing an SCA Decision Component**

# Copyright

## Copyright notice

© Copyright IBM Corp. 1987, 2013

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

---

## Trademarks

IBM, the IBM logo, ibm.com, WebSphere, and ILOG are trademarks or registered trademarks of International Business Machines Corp., in many jurisdictions worldwide.

Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at *Copyright and trademark information*.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

---

## Overview

With the business rules created, the next step is to utilize this business logic in a business processes. In this task you will create the business process that calls the business rules using the SCA Managed Decision Components created in task 5.

Readers who want to skip the exercise should delete the existing **BPELProcessDecisionModule** projects(make sure to disassociate it from Process Center first) and import the task 6 solution into IID from

**[SupportPac LA71 Path]\BPMTutorial\task6\BPelProcessDecisionModule.zip.**

You can continue at Step 8 to test the Process in the Process Server.

### Step 1. Define the Process Interface

The business process requires an interface which can be called to start the BPEL process. In this step you will create a new interface using the existing Business Objects.

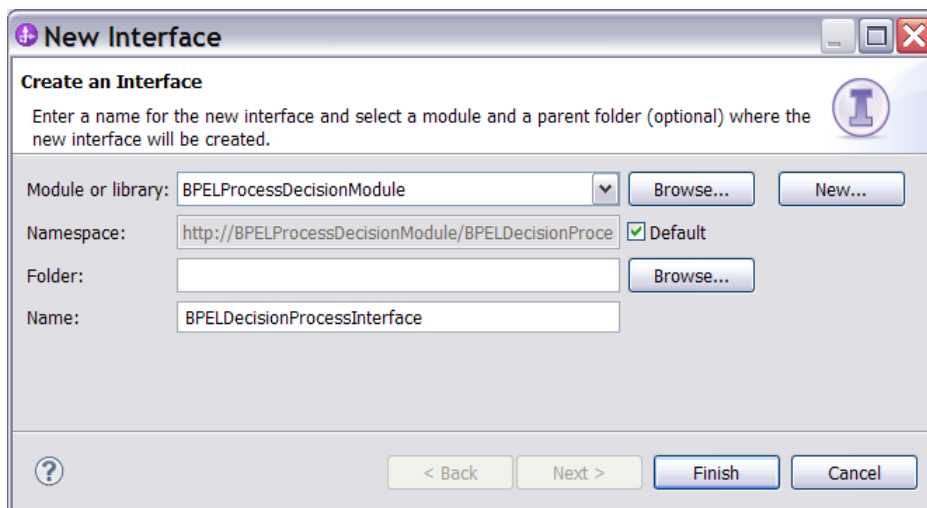
Open Integration Designer and select the **BPelProcessDecisionModule** project.

Select the **Interfaces** folder and right click.

Select **New > Interface**.

Make sure the Module is defined as **BPelProcessDecisionModule**

Define the interface name as **BPelDecisionProcessInterface**



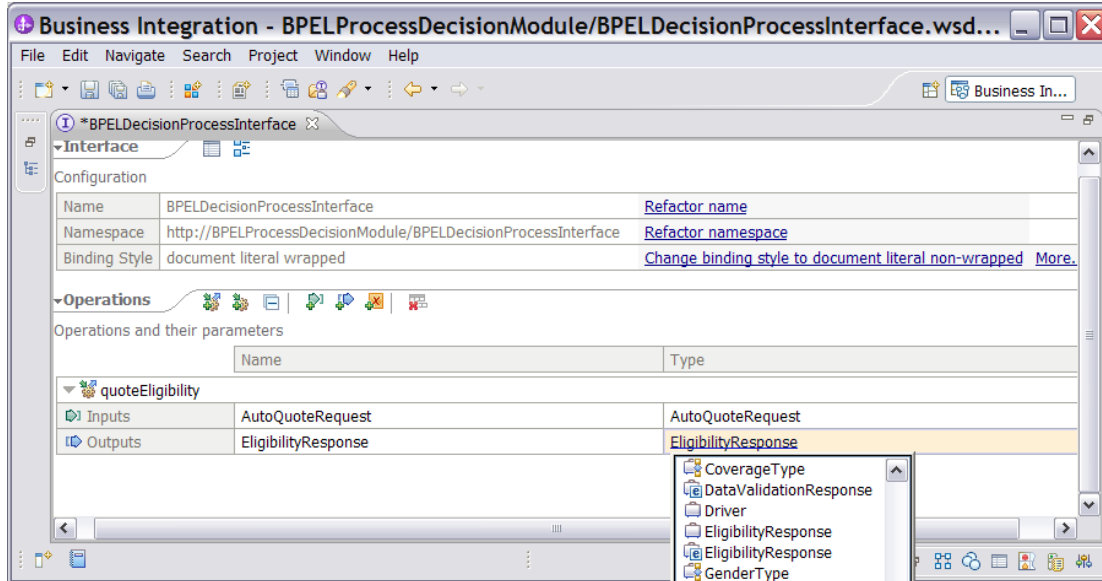
Click **Finish**.

In the **BPelDecisionProcessInterface** tab that opens :

Click the **Add Request Response operation** icon and rename the resulting operation to **quoteEligibility**.

Rename the operation input to **AutoQuoteRequest** and select the existing **AutoQuoteRequest** type.

Rename the operation output to **EligibilityResponse** and select the existing **EligibilityResponse** type.



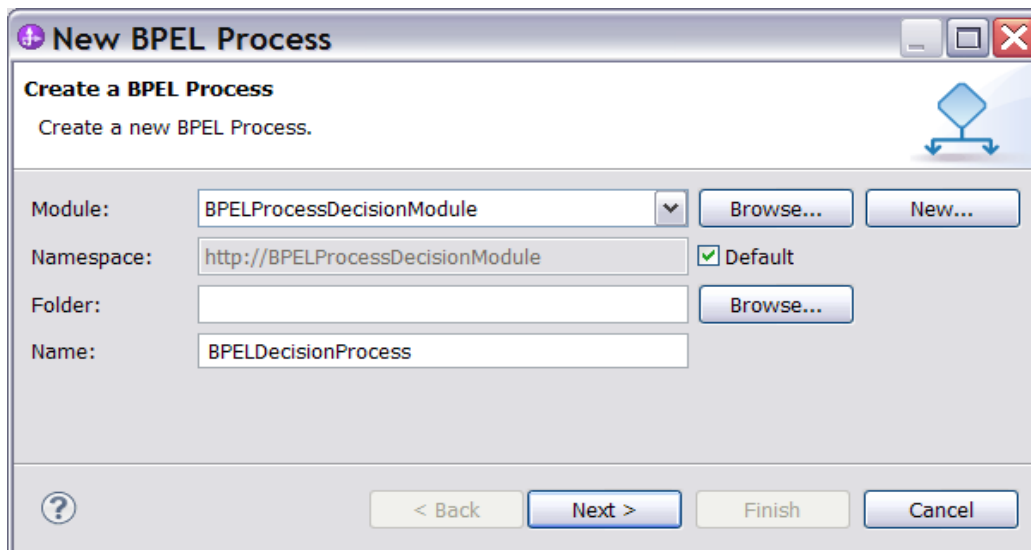
Save the updated **BPELDecisionProcessInterface** interface.

## Step 2. Create the Business Process Implementation for Quote Eligibility

In this step you will create a new BPEL process to that will be started from this interface.

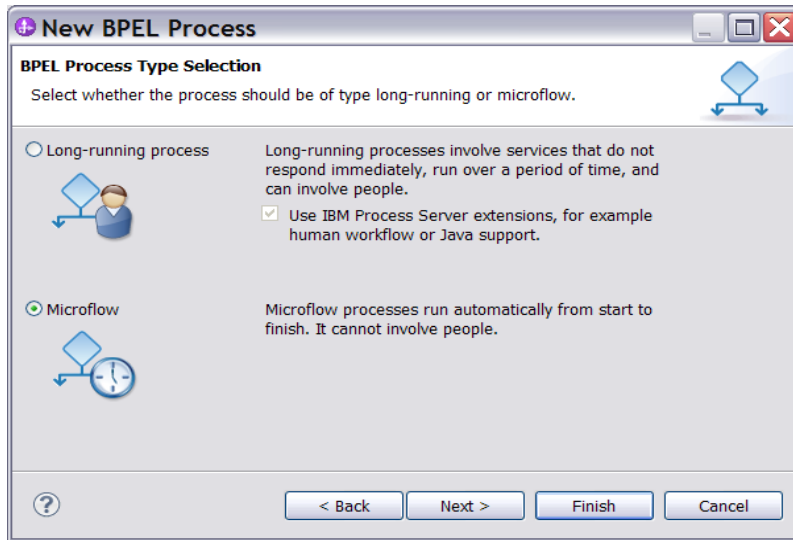
Select **File > New > BPEL Process**.

In the Create a BPEL Process screen, make sure the **BPELProcessDecisionModule** is selected. For the name, enter **BPELDecisionProcess**.

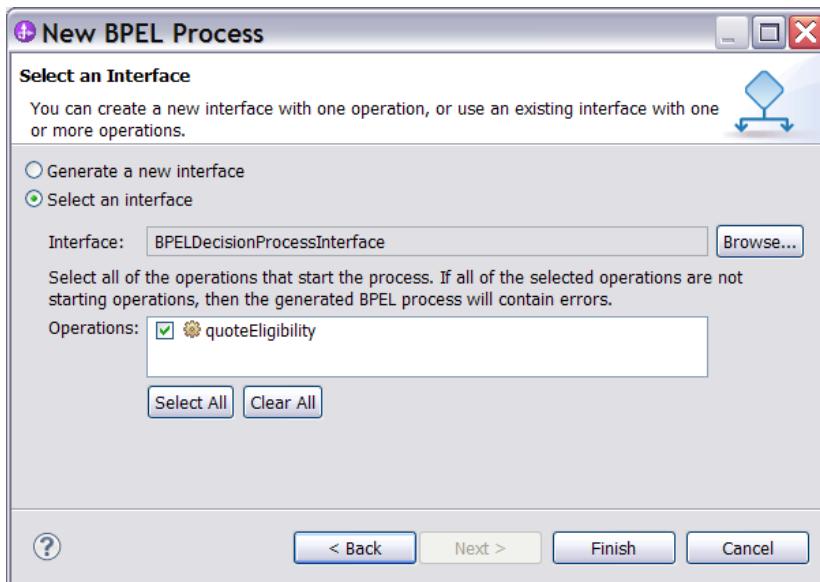


Click **Next**.

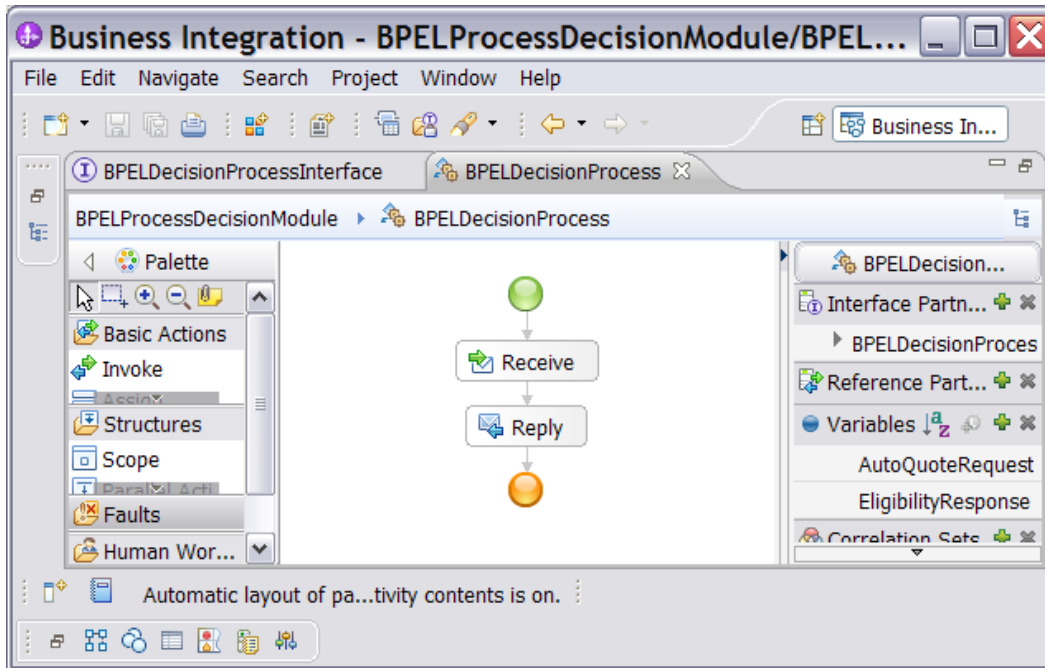
On the **Business Process Type Selection** screen, select **Microflow**.



Click **Next**. In the **Select an Interface** screen,  
 Select the radiobutton: **Select an interface**.  
 Click **Browse...** and select the **BPELDecisionProcessInterface** you have just created.  
 Ensure the **quoteEligibility** operation is selected and click **Finish**.



The BPEL process is created and displayed in the process editor.



### Step 3. Add logic for checking eligibility before confirming the quote.

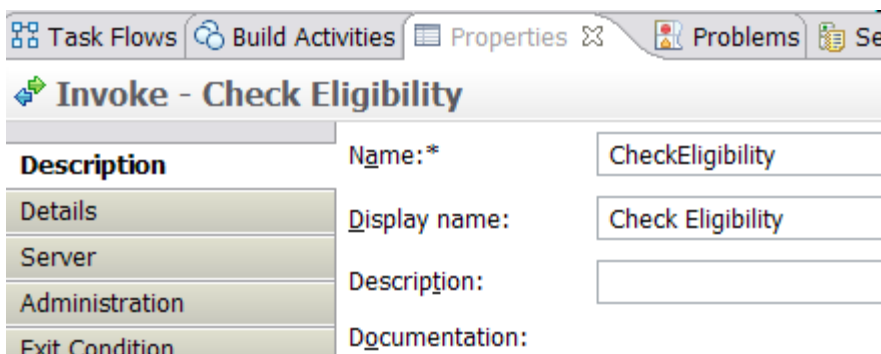
In the process editor, from the Palette under Basic Actions, select **Invoke**.  
Click between the **Receive** and **Reply** activities to insert the **Invoke** activity.

Select the **Invoke** activity and select the **Properties** view.

Change the Name to **CheckEligibility**.

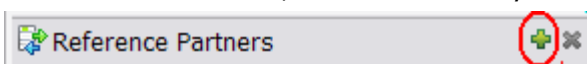
Change the Display name to **Check Eligibility**.

This node will make the call to the ManagedDecision CheckEligibility decision to determine if the individual is eligible for a insurance.

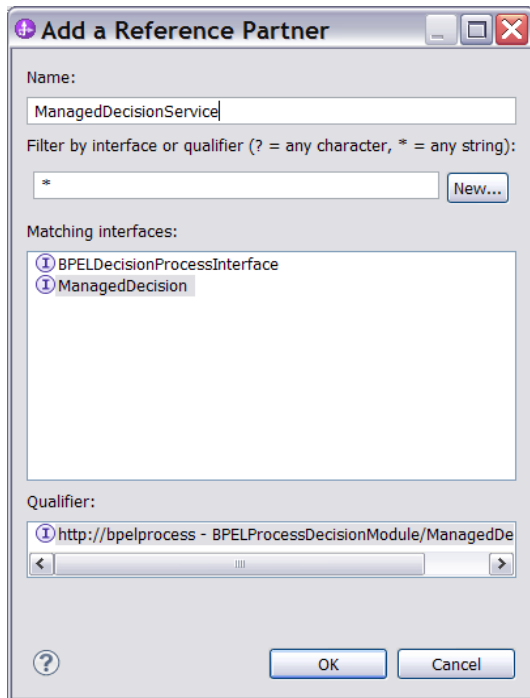


The Decision Service is called through a Reference Partner that must now be created.

In the Process Editor, select the add symbol for Reference Partners.



In the **Add a Reference Partner** screen  
For the name type **ManagedDecisionService**  
In the **Matching interface** field select **ManagedDecision**



Click **OK**.

Select the Check Eligibility node again and in the Properties view, select the **Details** tab.

For the Partner, select **Browse...** and select **ManagedDecisionService** (the partner you have just created)  
Click **OK**.

Change the Operation to be **Eligibility**.

Name	Type	Read from Variable
AutoQuoteRequest	AutoQuoteRequest	(none)
EligibilityResponse	EligibilityResponse	(none)

Name	Type	Store into Variable
EligibilityExecutionResult	EligibilityExecutionResult	(none)

You now need to map the decision service parameters to process variables.  
 For the AutoQuoteRequest Input, under the Read from Variable column click **(none)**.  
 Select **AutoQuoteRequest** – the parameter input to the process when starting it.  
 For the EligibilityResponse Input, under the Read from Variable column click **(none)**.  
 Select **EligibilityResponse** – the parameter returned from the process when starting it. This parameter will need to be initialized before passing to the decision service.

Name	Type	Read from Variable
AutoQuoteRequest	AutoQuoteRequest	AutoQuoteRequest
EligibilityResponse	EligibilityResponse	(none)

For the **EligibilityExecutionResult** Output, under the Read from Variable column, click **(none)**.  
 Click **New...** and for the Variable Name accept the value **EligibilityExecutionResult**.

Variable Name

EligibilityExecutionResult

OK Cancel

Click **OK**.  
 Save the Invoke activity and check the properties again.



**Invoke - Check Eligibility**

Description: Partner:\* ManagedDecisionService [Browse...](#)

Details: Interface:\* [ManagedDecision](#)

Operation:\* Eligibility

☒ Use data type variables mapping

	Name	Type	Read from Variable
Inputs	AutoQuoteRequest	AutoQuoteRequest	AutoQuoteRequest
	EligibilityResponse	EligibilityResponse	EligibilityResponse
	Name	Type	Store into Variable
Outputs	EligibilityExecutionResult	EligibilityExecutionResult	EligibilityExecutionResult

This should remove any error from the process and result in a flow including the invoke of the decision.

#### Step 4. Initialize the decision parameters.

In the process editor, from the Palette under Basic Actions, select **Assign** and click between the **Receive** and **Check Eligibility** activities.

Rename the node to **Initialize**. This will be used to ensure that the EligibilityResponse variable is initialized.

Select the **Initialize node** and select Details inside the Properties tab.

**Business Integration**

File Edit Navigate Search Project Window Help

Task Flows Build Activities Properties Problems Server Logs Servers

**Assign - Assign**

Details

Server

Exit Condition

Environment

Assign From: String

Assign To: EligibilityResponse MainMessage

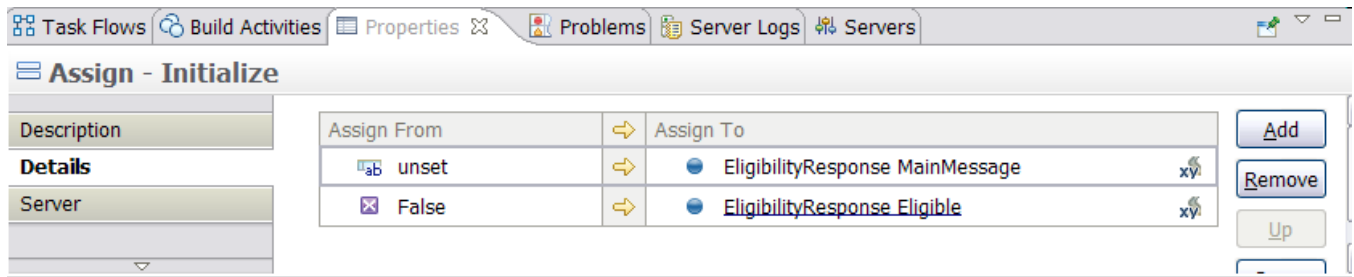
Add Remove Up Down

In the **Select From** field select a **String** and enter the value **"unset"**.

In the **Select To** select **EligibilityResponse**, open the twistie and select **MainMessage**.

Click the **Add** Button.

Repeat this to set the **EligibilityResponse.Eligible** to **false**.

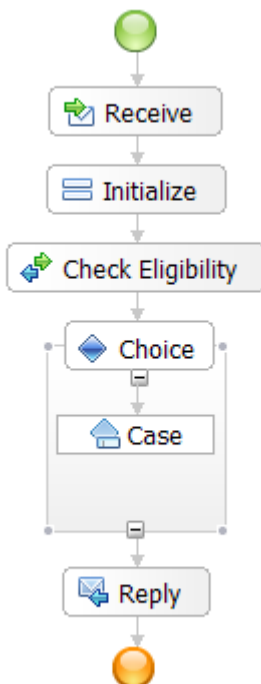


This has now initialized the EligibilityResponse variable so it can be used as an in/out parameter to the decision service.

### Step 5. Use the managed decision to influence the process flow.

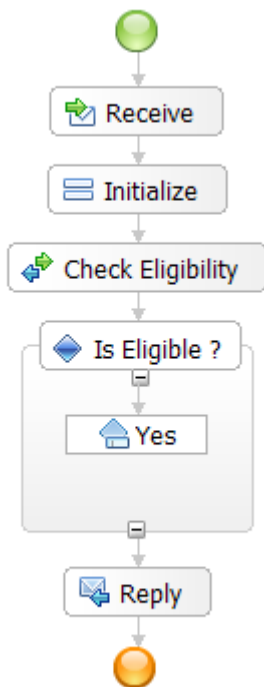
Combining a call to an Invoke (representing a decision) followed by a Choice node is a common design pattern which reduces the amount of hard-coded logic. The decision is reused from the BRMS but the "Choice" enforces the decision in the context of the process. The logic can be captured in business rules with the result checked by the Choice and different cases represent different paths to handle the result.

In the process editor, from the Palette under **Structures**, select **Choice**. Click between the **Check Eligibility** and **Reply** activities. A Choice is inserted into the flow.



Select the **Choice** node and in the Properties view change the Name to **Is Eligible?**. A Case is created by default.

Select the Case and in the Properties view change the Display name to **Yes**.

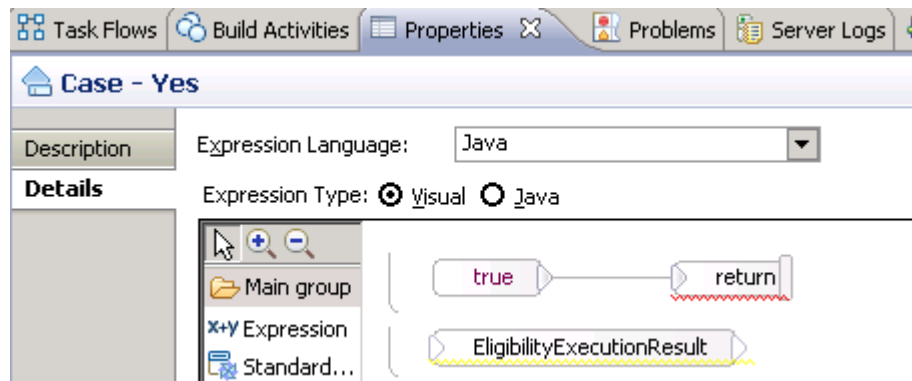


The “Yes” condition is executed if the result from the Check Eligibility invoke is true.

To specify the condition, select the Details tab.

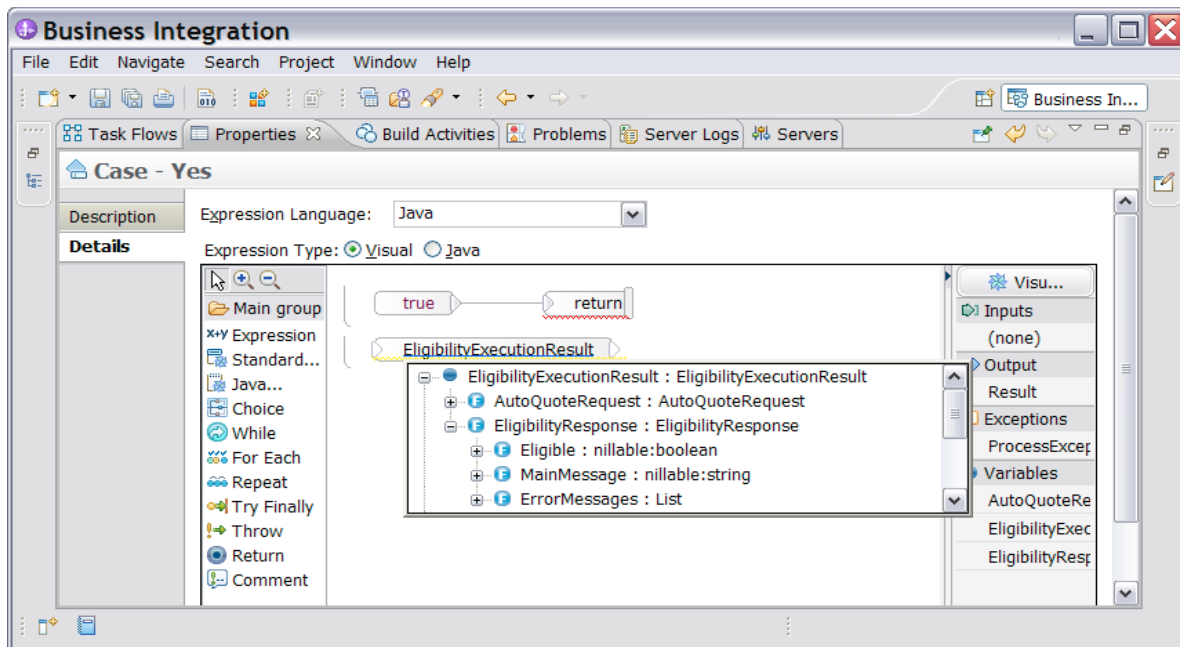
For the Expression Language select **Java**.

From the list of Variables, drag and drop **EligibilityExecutionResult** onto the case editor.

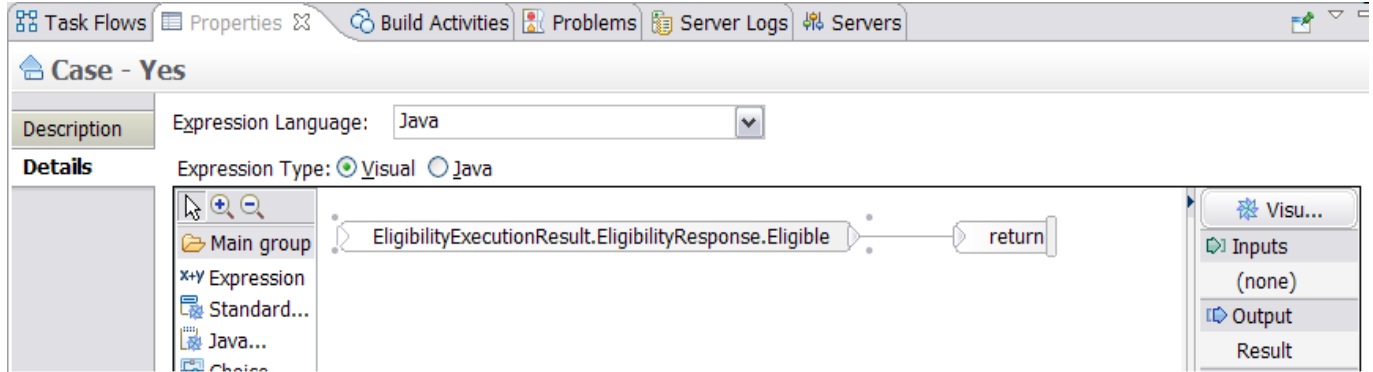


Click **EligibilityExecutionResult** to see the attributes for the variable.

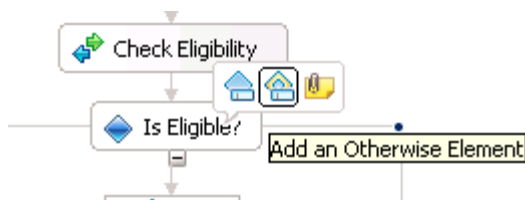
Expand **EligibilityExecutionResult > EligibilityResponse > Eligible** by clicking the '+' symbols, and then click on **Eligible**.



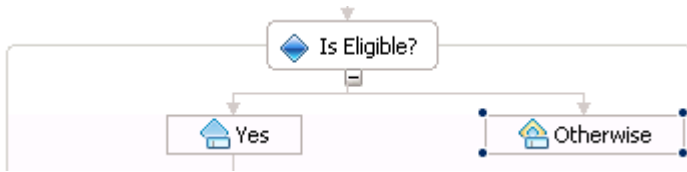
Right-click the **true** value and select **Delete** to remove this value from the case editor. Wire the **EligibilityExecutionResult.EligibilityResponse.Eligible** variable to the **return** by selecting the variable and then dragging the line which is extending to its right onto the **return**. Note that the wire will not operate until the fully qualified value is used.



Besides the Yes case, a case is needed to handle a customer that is not eligible. Click the **Is Eligible?** choice node.



Select the **Otherwise** icon. An otherwise case is added.



You have now created two cases that the process will follow dependent on the decision made by the BRMS as to whether the customer is eligible for insurance or not.

## Step 6. Define Process Response Messages based on decision cases.

In this step you will provide the process logic for each of the decision cases.

A confirmation needs to be returned to the customer according to whether they were eligible or not. For this process, a snippet will be used to specify a confirmation message.

From the Palette, select **Snippet** from the list of Basic Actions.

Click beneath the **Yes** case node.

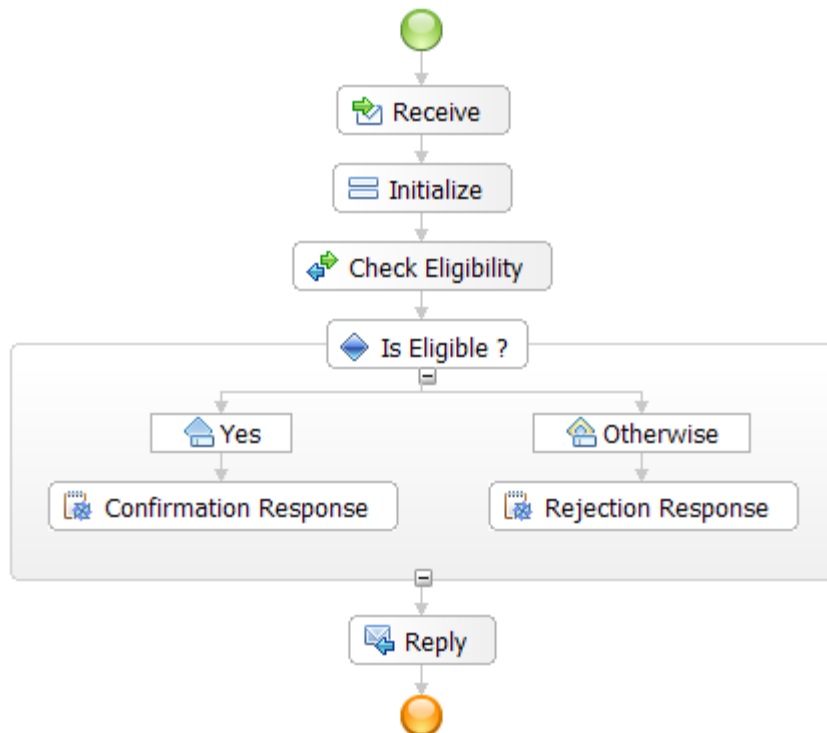
Change the Display name to **Confirmation Response**.

From the Palette, select **Snippet** from the list of Basic Actions.

Click beneath the Otherwise case node.

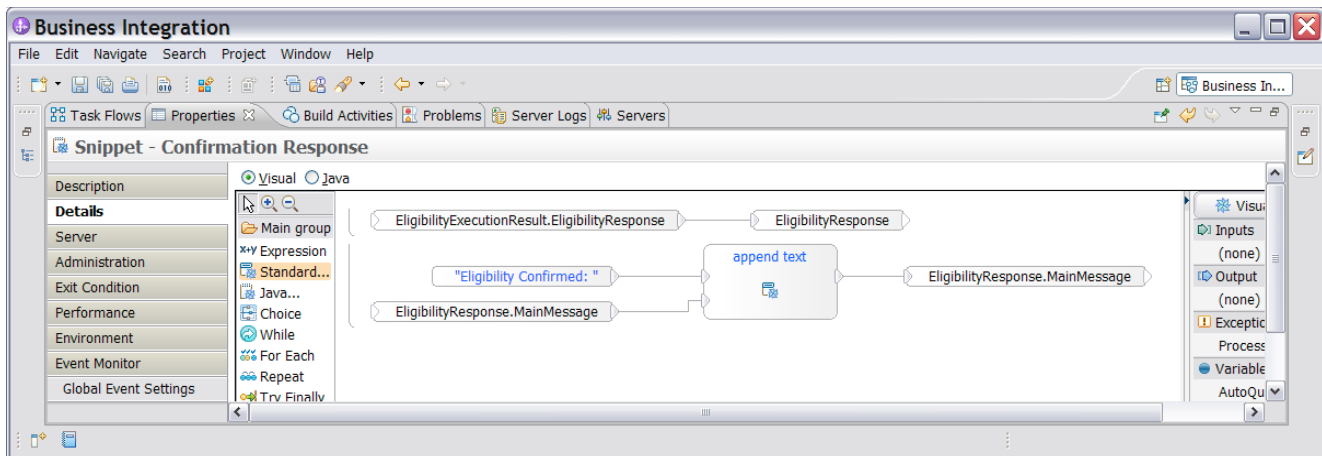
Change the Display name to **Rejection Response**.

Click the **Save** button.



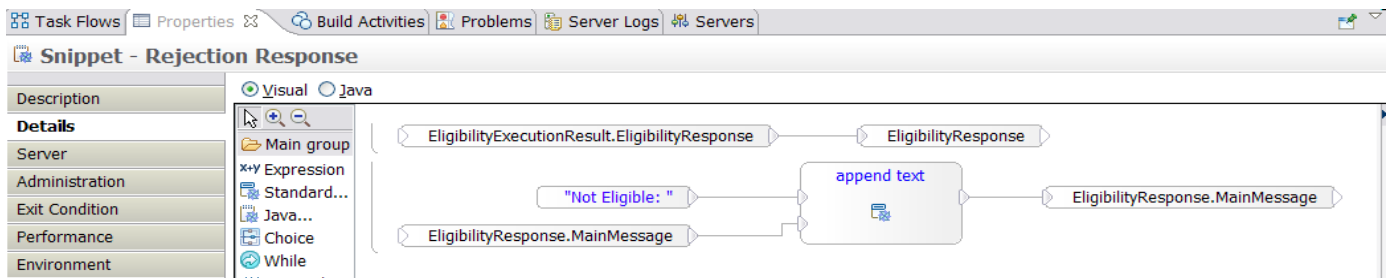


Wire the output of the append text snippet to this new **EligibilityResponse.MainMessage** by selecting the append text snippet and then dragging the line which is extending to its right onto the new **EligibilityResponse.MainMessage**.



Now you need to repeat this for the Rejection Response. Before you leave the Confirmation Response select and copy all the visual snippets.

In the Process Editor select the Rejection Response node and open the Properties view Details tab. Paste the copy buffer into the window. Select the **"Eligibility Confirmed: "** String, click **String** in the popup, and set its value to **"Not Eligible: "**

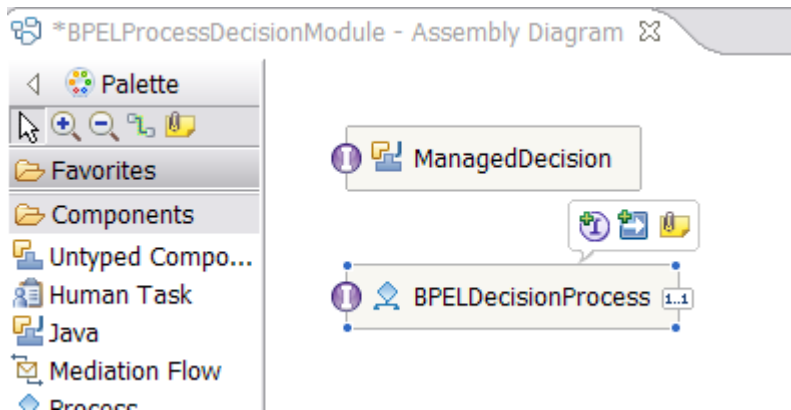


Save your work. You have now completed the implementation of the process.

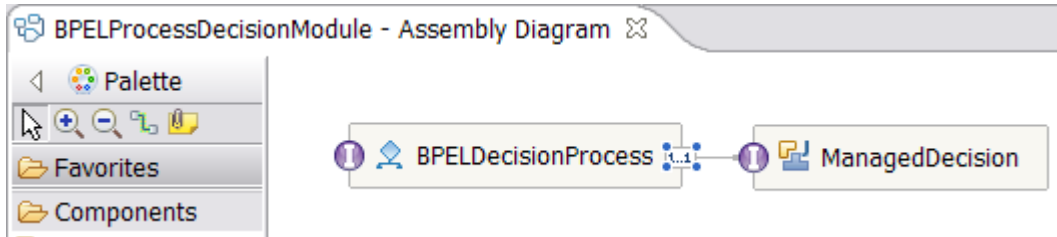
You have now created two cases that the process will follow dependent on the decision made by the BRMS as to whether the customer is eligible for insurance or not.

## Step 7. Assembling the BPM Application.

In this step you will assemble the full process solution before deploying and testing it. Open the **Assembly Diagram** to open the assembly editor if it is not already opened. In the Business Integration view, under **BPELProcessDecisionModule > Integration Logic > BPEL Processes**, select the **BPELDecisionProcess** process. Drag and drop the **BPELDecisionProcess** process onto the assembly editor.



Connect the reference on the **BPELDecisionProcess** to the **ManagedDecision** component by selecting the **BPELDecisionProcess** node and then dragging its connector extending to the right from the reference to the interface of the component.



Save the assembly editor. Check for any errors in the Problems view and resolve. Your application is complete and ready for a full test of the process and rules.

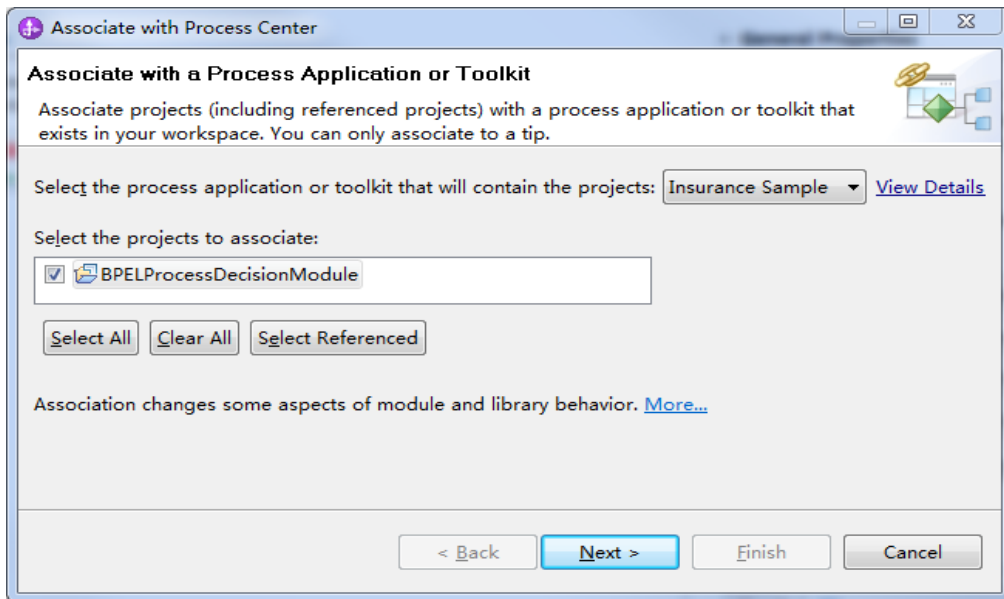
## Step 8. Deploy and test the BPEL Process.

Just as the business rules, the business process can be deployed as an application to the Business Process Manager. Unlike the business rules, deployment is simply installing the application on the server. Once deployed, the process can be tested using the Test Component facility, Business Process Choreographer Explorer, or Business Space. You will test the process and the rules using the Test Component.

The process needs to be associated with a process application before publishing to the process center for test.

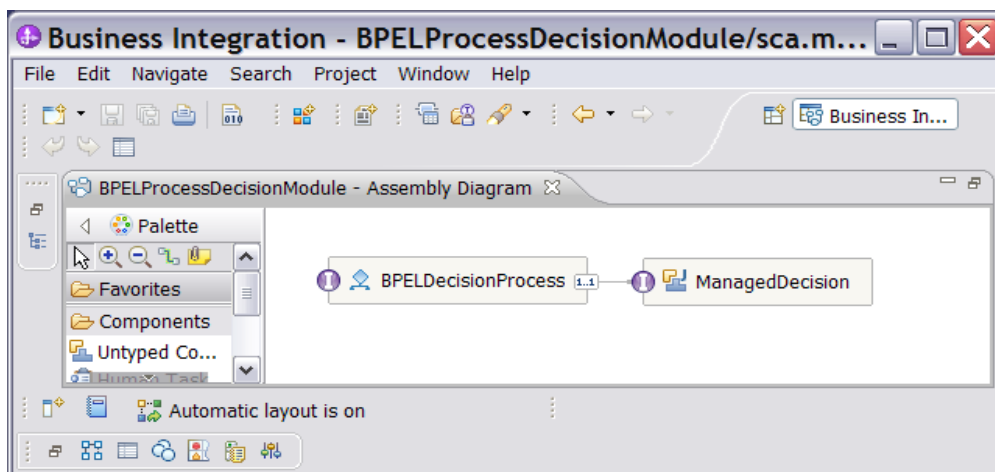
In the Business Integration view, right click your BPEL process module and select "Associate with Process Center".





Open the Assembly Diagram.

Select the **BPELDecisionProcess**, right click and select **Test Component**.



In the Integration Test Client Value Editor, select the **NumberOfAccidents** property and set it to 5.

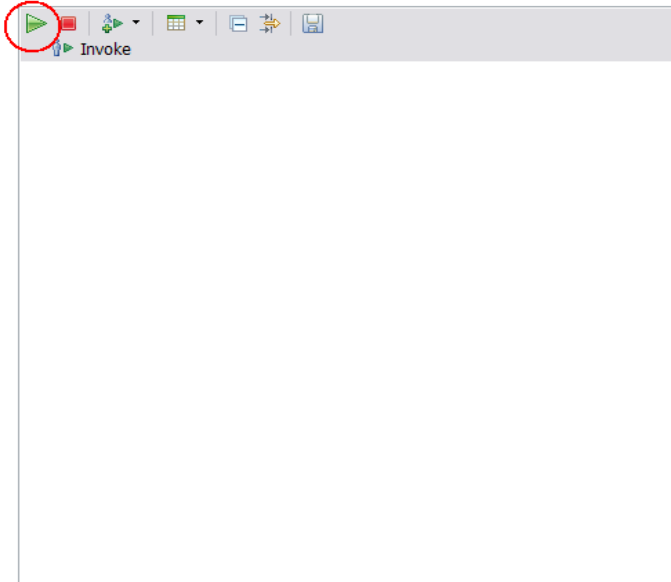
The rules should indicate that the quote is not eligible.

Click the continue icon to start the Test Client.

## Integration Test Client: BPELProcessDecisionModule\_Test

### Events

This area displays the events in a test trace. Select an event to display its properties in the General Properties and Detailed Properties sections. [More...](#)



### Detailed Properties

Specify the component, interface, operation, and input parameter values for the Invoke event, and then click the Continue icon in the Events area to run the test. [More...](#)

**Configuration:** Default Module Test

**Module:** BPELProcessDecisionModule

**Component:** BPELDecisionProcess

**Interface:** BPELDecisionProcessInterface

**Operation:** quoteEligibility

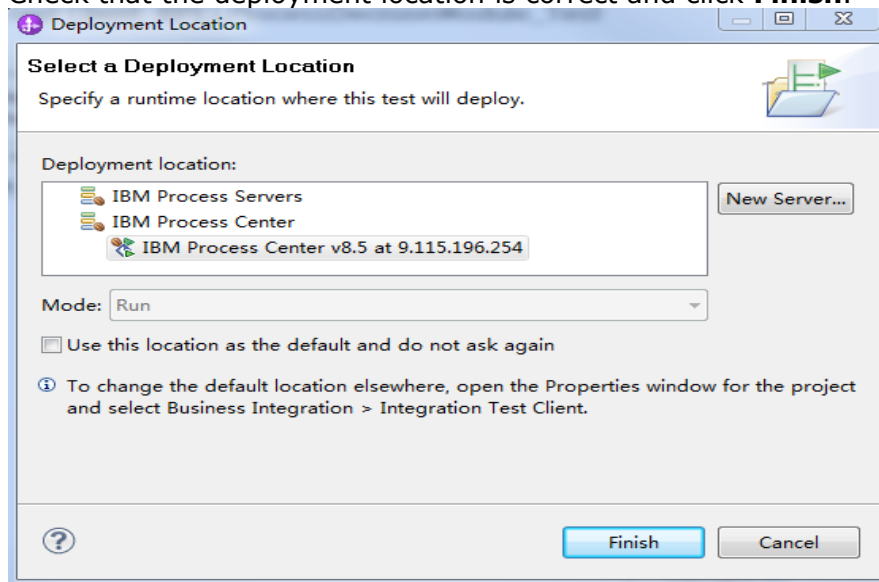
#### Initial request parameters:

☒ Value editor ☐ XML editor

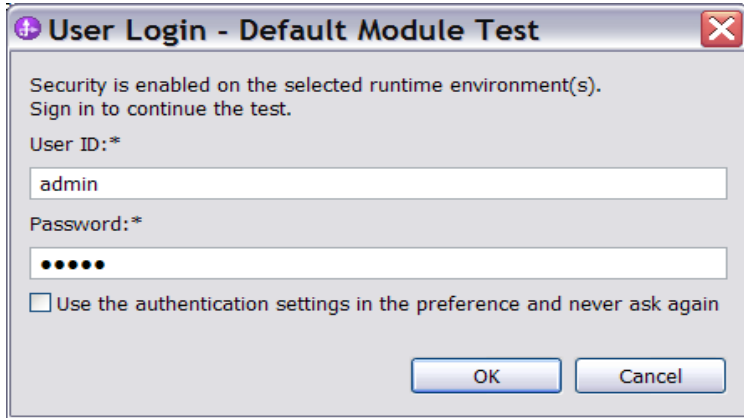
Name	Type	Value
VehicleVandalizedOrStolen	boolean	false
LicenseSuspendedOrRevoked	boolean	false
DUI	boolean	false
NumberOfAccidents	integer	5
NumberOfTrafficTickets	integer	0
Vehicle	Vehicle	
VehicleIdentificationNumber	string	

To edit values, start typing or press F2.

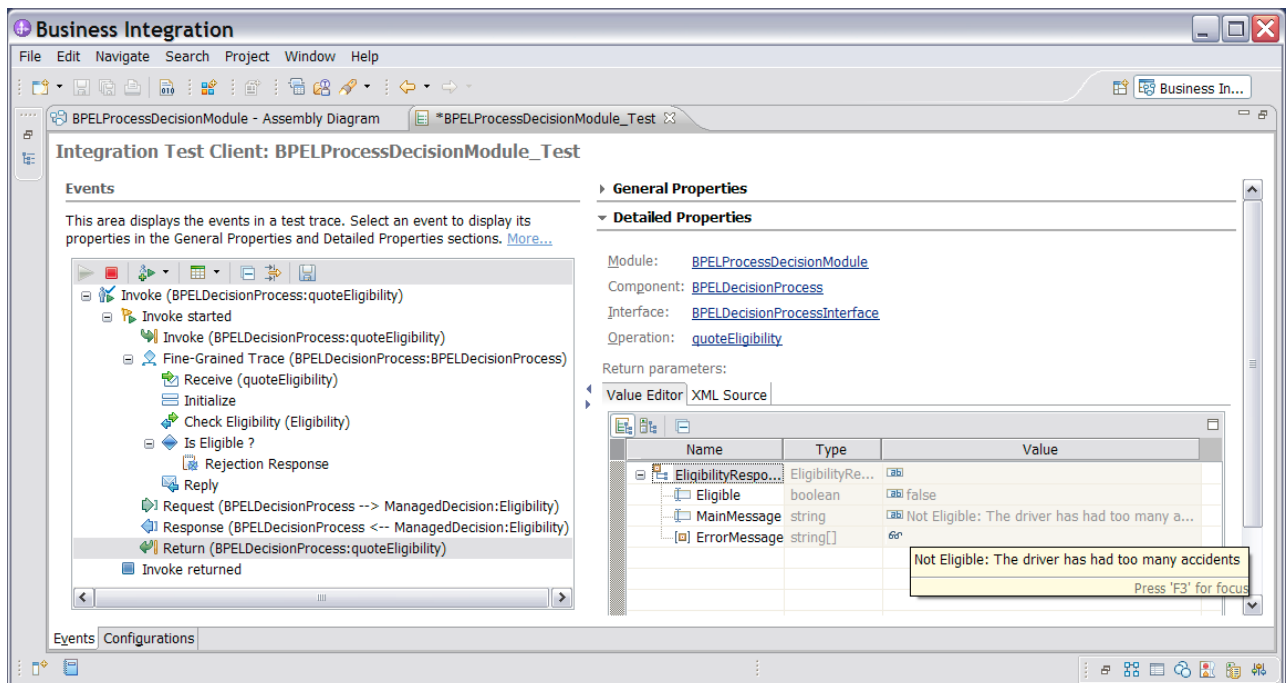
Check that the deployment location is correct and click **Finish**.



Check that the User Login characteristics are correct and click OK.



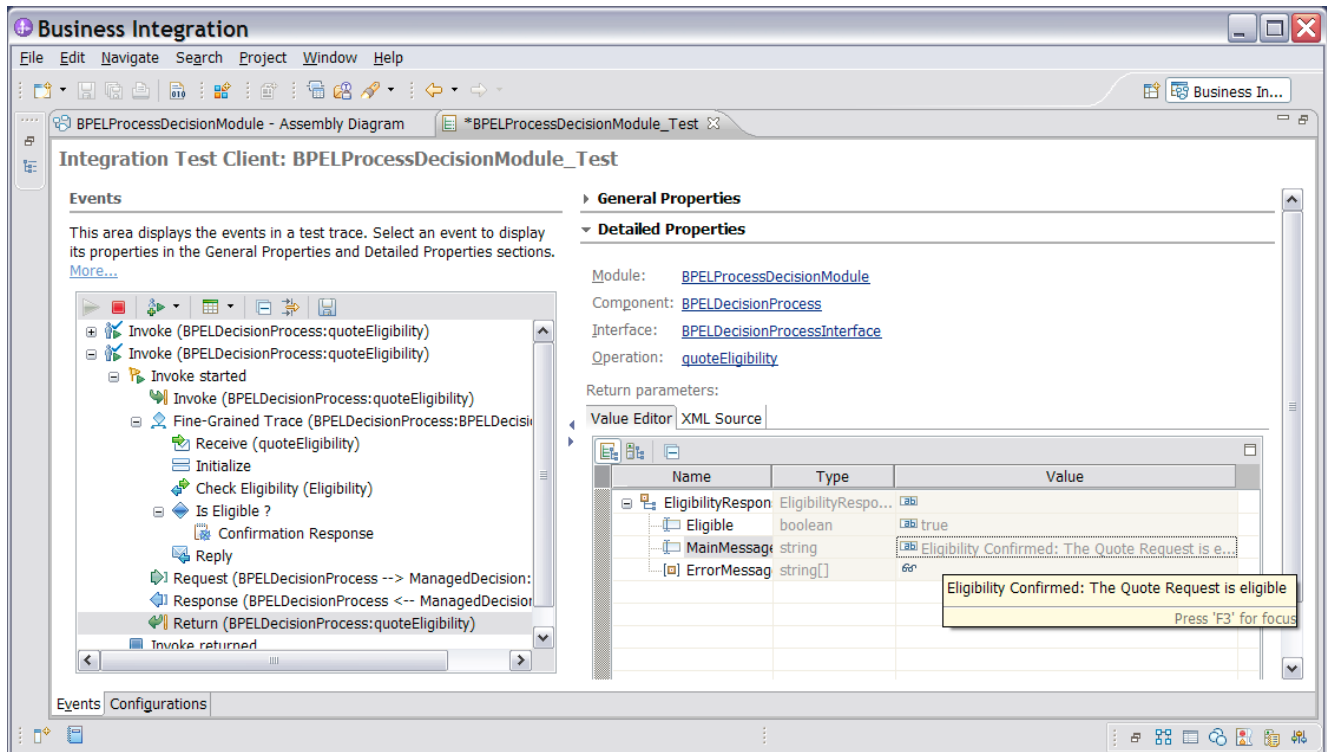
The process runs to completion showing the fine grained trace in the events panel.  
The **EligibilityResponse** from the process shows that the **Rejection Response** has correctly interpreted the response from the decision service



Click the Invoke icon and set the value of 2 for the number of accidents. Try running the Integration Client again by clicking the continue icon.



This time the Eligible branch of the flow should be followed.



In this task you have developed and tested a BPEL Process that uses a BRMS Managed Decision Component.