# BTSHOL12: Integrating Business Rules

Objectives

In this lab, you will learn how to enable applications to use Business Rule Engine.

After completing this lab, you will be able to:

* Define business rules.
* Call business rules from within an orchestration.
* Build and deploy the business rules project.
* Start and test the business rules

Scenario

In this lab we are re-using the same scenario from Lab 11, *Enabling Business Activity Monitoring*. This lab was a typical Order Management business process, with a manual approval step.

In this lab, you will complement this lab by automating the approval step providing any of the two scenarios:

* + The total order amount is less than $100
  + The customer has previously made orders for more than $2000.

If any of these two scenarios is true, the order is automatically approved, and does not require any manual step.

The BizTalk Business Rules Engine provides the tools to abstract business rules from orchestration code and to modify those business rules without stopping the orchestration process. For this reason, Northwind Traders plans to move most of its business logic to the Business Rules Engine toolset.

Estimated time to complete this lab   
(Exercise 1-4):   
45 minutes

Optional:  
Exercise 5-8

In this lab, you will create and deploy a rule set and see how to execute those rules from within an orchestration.

User Name: **Administrator**

Password: **pass@word1**

Exercise 1  
Familiarize yourself with the solution

Through the rest of this lab you using a simple order approve solution. This is the same scenario as Lab11.

|  |  |
| --- | --- |
| Tasks | Detailed steps |
| Configure the starting lab environment. | 1. In the Windows Explorer, browse to **C:\Labs\Lab 12\Start** and open **Lab12.sln**. 2. The solution appears in the Solution Explorer and the. Double-click the **OrderApprovalProcess.odx** to open the order process orchestration.      1. The **Receive Order** shape receives an order message 2. The **Send For Approval** shape initiates the **OrderIdCorrelation** correlation set on the Order/OrderId, and sends the message out for approval. 3. The **Receive Approval** shape is following the correlation set, and receives the approved (or declined) Order. 4. The **Approved** branch checks if the Order is approved: ApprovedOrder.Approved == true 5. The Approved/Denied Order is sent out. |

Exercise 2  
Defining a Business Policy

In this exercise you will take on the role of a *Business Analyst* and will complete the creation of a Business Policy which will be used to automate the approval step. If the criteria’s of the policy are met, the order will automatically be approved. If not, the order will continue through a manual approval.

|  |  |
| --- | --- |
| Tasks | Detailed steps |
| Defining the Policy   1. Start the Business Rule Composer. | 1. Press **windows button**, open **Business Rule Composer**. 2. If the **Open Rule Store** dialog box appears, accept the default options and click **OK** to connect to the database. |
| 1. Create a new policy   *A policy is a collection of one or more rules.* | 1. In the Policy Explorer, right-click **Policies**, and then click **Add New Policy**. 2. Type **OrderApprovalPolicy** as the policy name. |
| 1. Configure the Facts Explorer.   *Setting up the Facts Explorer allows you to drag and drop arguments to create rules* | 1. In the **Facts Explorer**, click the **XML Schemas** tab. 2. Right-click the Schemas node, then click Browse, and then open **C:\Labs\Lab 12\Start\Lab12\Order.xsd**.   The Order schema appears in the Facts Explorer.   1. Click the **Order.xsd** root node, and in the Properties window, verify the **Document Type** value is **Lab12.Order** or change it so that it is. |
| 1. Add rules to the policy.   *Rules in a policy fire in the order of their priority. If rules have the same priority the order is random.* | 1. Right-click **Version 1.0 (not saved)**, and select **Add New Rule**. Set the name of the new rule to *[1] DefaultApproval.* 2. In the **Properties** window for the rule, set **Priority** to **1**.   You do this to control the order that rules fire in. Rules will fire in order from highest to lowest propriety.   1. In the Conditions Editor (right pane), right-click **Conditions**, select **Predicates->Equals**. 2. For the left and right hand side arguments, *argument1* and *argument2*, enter a **1** in both places. This will make the rule always evaluate to true. |
| 1. Setting the Action   *If the condition created evaluates to true, the action defined in the Action pane will be performed.* | 1. In the **Action** pane (below the **Condition** Pane), drag the *Order.Approved* from the XML Schema tab in the **Facts Explorer**. 2. Click the “<enter a value>” and set the value to *False*.   You do this to establish a default value before evaluating the “real” rule. |
| 1. Add rules to the policy. | 1. Right-click **Version 1.0 (not saved)**, and select “Add New Rule”. Set the name of the new rule to *[2] EvaluateApproval.*   We will leave the priority as the default 0 value for this rule.   1. In the Conditions Editor (right pane), right-click **Conditions**, select “Predicates->LessThan”. 2. In the Facts Explorer, drag the *TotalSum* from the Order in the XML Schemas tab, to the left side of the predicate (*argument1*). 3. Set *argument2* to 100. |
| 1. Setting the Action   *If the condition created evaluates to true, the action defined in the Action pane will be performed.* | 1. In the Action pane (below the Condition Pane), drag the *Order.Approved* from the XML Schema tab in the Facts Explorer. 2. Click the “<enter a value>” and set the value to True.   *The Policy should look similar to this:*  Untitled |

Exercise 3  
Test and Publish the policy to BizTalk

In this exercise we will test the policy using different inputs and deploy it to BizTalk.

|  |  |
| --- | --- |
| Tasks | Detailed steps |
| Test the policy | 1. In the Policy Explorer, right-click the *Version 1.0 (not saved)* node, and select *Test Policy*. Click *Yes,* to save the policy. 2. In the *Select Facts* dialog, select Lab12.Order, and click the “Add instance” button. Browse to *C:\Labs\Lab 12\Start* and select the *Order.xml* file.   Untitled   1. Open *C:\Labs\Lab 12\Start\Order.xml* using notepad, and examine the content. 2. Click the *Test* button, to start the test. |
| Analyze the result | 1. In the result pane on the right, **scroll to the top.**     You will not see the evaluation of the DefaultApproval rule since it contains no facts.   1. The result should evaluate to “*False*” as the screen shoot above. 2. If you have closed the Order.xml, open it again using notepad (*C:\Labs\Lab 12\Start\Order.xml).* 3. Set the *TotalSum* to *50*, and save the document.   <TotalSum>50</TotalSum>   1. Re-test the policy, and examine the result.   The test should evaluate to True, as the Order amount is less than 100.   1. Right-click *Version 1.0* in the Policy Explorer, and select *Publish.* Then select it again, and select *Deploy.* |

Exercise 4  
Calling the rule from a BizTalk orchestration

In this exercise we will use the Call Rules shape to call the policy from an orchestration. Then we will deploy and test the orchestration.

|  |  |
| --- | --- |
| Tasks | Detailed steps |
| Update the orchestration | 1. In Visual Studio 2015, open the OrderApprovalProcess.odx from the Solution Explorer. 2. From the tool box, drag a *Call Rules* shape and drop it after the “Receive Order” shape. Set the name to “Call Order Approval Policy”. 3. Click the exclamation mark, and click the *No Policy specified,* to open the *CallRules policy configuration* dialog. 4. In the *CallRules policy configuration* dialog, select the *OrderApprovalPolicy* from the drop-down list. 5. Click the grid below to add *OrderIn* as a policy parameter. Click *Ok.* 6. Drag a *Decision* shape below the CallRules shape, and set the name to *Auto Approval.* 7. Click *Rule\_1*, and set the Name to *“Yes”.* Click the exclamation mark to set the expression. 8. In the *BizTalk Expression Editor* set the expression to:   OrderIn.Approved == true   1. Add a *Message Assignment* to the left branch (“Yes”). 2. Click the *ConstructMessage\_1* shape and set the *Message Constructed* property to “Approved Order”. 3. Double click the *MessageAssignment\_1* shape and set the expression to:   ApprovedOrder = OrderIn;   1. Drag the *Send For Approval* and *Receive Approval* shapes to the **right** decision branch.   If the policy evaluates to False, the process should proceed with manual approval. |
| Deploy the orchestration | 1. In the Solution Explorer, right-click the *Lab12* project and select *Deploy*. Make sure the solution was deployed successfully. |
| Configure Bindings | 1. Open *BizTalk Administration Console*, expand the *Applications* node and right-click the *Lab12* application, and select *Import-> Bindings.* Type *C:\Labs\Lab 12\Start \Lab12\Lab12.BindingInfo.xml* and click *Open.* 2. Check to make sure that the orchestrations port bindings are correct. |
| Start and test the orchestration | 1. Right-click the *Lab12* Application and select *Start*. 2. Open Windows Explorer, and expand the folder to *C:\Labs\Work\Lab 12\FileDrop.* 3. Copy the *Order\_50.Xml* to the *New Order* folder*.*   This message should evaluate to True, and the file should be automatically approved.   1. Copy the *Order\_500.Xml* to the *New Order* folder*.*   This message should evaluate to False, and should require manual approval through the Waiting for approval folder. |

Exercise 5  
Using a database fact

In this exercise you will create a new version of the Business Rule that adds the logic of calling a database that holds the customers previous purchases and takes that into account for the approval.

|  |  |
| --- | --- |
| Tasks | Detailed steps |
| Updating the Policy   1. Database Facts   *Setting up the Facts Explorer allows you to drag and drop arguments to create rules* | 1. In the **Business Rule Composer**, and **Facts Explorer**, click the **Databases** tab. 2. Right-click **Servers**, and then click **Browse**. 3. Click **OK** to display a list of SQL databases. |
| 1. The policy   *A policy is a collection of one or more rules.* | 1. In the Policy Explorer, expand **Policies**, **OrderApprovalPolicy**, and right-click **Version 1.0 - Deployed**, and select “**Copy**”. 2. Right-click **OrderApprovalPolicy**, and select “**Paste Policy Version**”. |
| 1. Add rules to the policy. | 1. Select the *[2] EvaluateApproval* rule and in the Conditions Editor (right pane), right-click **Conditions**, select to“Add logical OR” 2. Right-click the “OR” node, and select “Add logical AND”. 3. Right-click the “AND” node, select “Predicates->Equal”. 4. In the Facts Explorer, click the *Databases* tab. Browse down to the *SalesDetail* database. Expand the *CustomerSalesStatistics* table, and drag the *CustomerName* column to *argument1*(left side of the *is equal to* predicate) 5. In the Facts Explorer, click the *XML Schemas* tab. Drag the CustomerName element from the Order schema to *argument2.* 6. In the Conditions Editor (right pane), right-click the “AND” node, select “Predicates->GreaterThan”. 7. In the Facts Explorer, drag the *SalesDetails/ CustomerSalesStatistics/TotalOrdersToDate* column from the Database fact to *argument1.* 8. In the Conditions Editor click the argument2, and set it to “2000”.   So far we have set up a query similar to:  SELECT \*  FROM CustomerSalesStatistics  WHERE CustomerName = ?  AND TotalOrdersToDate = ?  *Your policy should look similar to this now:*    *If you were to write these conditions using C#, it would have been similar to:* |

Exercise 6  
Test and Publish the new policy version to BizTalk

In this exercise we test the new version of the policy in the Business Rules Composter to see that it does what we want and then deploy and test the new version of the business rule to BizTalk and see how it automatically gets used.

|  |  |
| --- | --- |
| Tasks | Detailed steps |
| Test the policy | 1. In the Policy Explorer, right-click the *Version 1.1 (not saved)* node, and select *Test Policy*. If asked, click *Yes* to save the policy. 2. In the *Select Facts* dialog, select Lab12.Order, and click the “Add instance” button. Browse to *C:\Labs\Lab 12\Start* and select the *Order.xml* file if it is not selected for you already. 3. Continue by selecting the *SalesDetail: CustomerSalesStatistics (Data Connection)* node, and click the “Add instance” button. Click *Ok* in the *Connect to SQL Server* dialog, and browse to the *SalesDetail* database. Select the *CustomerSalesStatistics* table*.*      1. If you have closed the Order.xml, open it again using notepad (*C:\Labs\Lab 12\Start\Order.xml).* 2. Set the *TotalSum* element in the Order.xml file to *500*, and save the document.   <TotalSum>500</TotalSum>   1. Click the *Test* button, to start the test. |
| Analyze the result | 1. In the result pane on the right, **scroll to the top.**      1. The result should evaluate to “*True*” as the screen shoot above. 2. Press *windows button->* click *SQL Server Management Studio*. Expand the *SalesDetails,* and right-click the *dbo.CustomerSalesStatistics* table. Select “Edit top 200 rows”. 3. Set the *TotalOrderToDate* to **1500** for the *Fabrikam* customer (First Row). 4. Re-test the policy, and examine the result.   The test should evaluate to False, as the Customer (Fabrikam) haven’t had enough order value to date.   1. If you have closed the Order.xml, open it again using notepad (*C:\Labs\Lab 12\Start\Order.xml).* 2. Set the *TotalSum* to *50*, and save the document.   <TotalSum>50</TotalSum>   1. Re-test the policy, and examine the result.   The test should evaluate to True, as the Order amount is still less than 100.  Read more about the Policy Test Trace Output at http://msdn.microsoft.com/en-us/library/aa561493.aspx.   1. Right-click *Version 1.1* in the Policy Explorer, and select *Publish.* Then select it again, and select *Deploy.* |

Exercise 7  
Calling a rule with a database connection from a BizTalk orchestration

In this exercise we will update the orchestration with the implementation necessary to call a Business Rule that uses a database.

|  |  |
| --- | --- |
| Tasks | Detailed steps |
| Update the orchestration | 1. In Visual Studio 2015, open the Lab12.sln if you have closed it. 2. On the Lab12 project, select to Add Reference. 3. Add references to the System.Data and System.Transactions under the .Framework tab. 4. Add a reference to the Microsoft.RuleEngine.dll in the C:\Program Files (x86)\Common Files\Microsoft BizTalk folder by locating it using the Browse tab.   You will need these to be able to supply the business rule with a database connection.   1. Open the OrderApprovalProcess.odx orchestration. 2. Now select the Orchestration surface so that you can set the orchestration properties, and set the transaction type of the orchestration to be Long Running.   Untitled   1. Next create a Scope shape above the Call Rules shape. Configure it to have a Transaction Scope set to Atomic.   We need this since the classes we are using are not serializable. Such classes can only be used from within an atomic transactional scope.  Untitled   1. Before we can configure the Call Rules shape we must prepare a database connection that we can pass as a parameter (as an input fact). In the Orchestration View, beneath the atomic Scope shape, create two variables: 2. **sqlConn** as a **System.Data.SqlClient.SqlConnection** 3. **dataConn** as a **Microsoft.RulesEngine.DataConnection**   Untitled   1. In the Scope shape, create a new Expression shape. In the expression shape instantiate the two variables using the code below:   sqlConn = new System.Data.SqlClient.SqlConnection("Initial Catalog= SalesDetails;Data Source=(local);Integrated Security=SSPI;");  dataConn = new Microsoft.RuleEngine.DataConnection("SalesDetails", "CustomerSalesStatistics", sqlConn);   1. Move the Call Rules shape and place it inside the atomic scope shape. 2. Configure the Call Rules shape with the OrderIn message and the dataConn object.   Untitled  *Verify that your orchestration looks similar to this:* |
| Deploy the orchestration | 1. In the **Solution Explorer**, right-click the *Lab12* project and select *Deploy*. Make sure the solution was deployed successfully. 2. In the **BizTalk Server Administration Console**, **Restart the host instance**. |
| Test the orchestration | 1. Open Windows Explorer, and expand the folder to *C:\Labs\Work\Lab 12\FileDrop.* 2. Copy the *Order\_50.Xml* to the *New Order* folder*.*   This message should evaluate to True, and the file should be automatically approved.   1. Copy the *Order\_500.Xml* to the *New Order* folder*.*   This message should evaluate to False, and should require manual approval through the Waiting for approval folder.   1. Copy the *Order\_1000.Xml* to the *New Order* folder*.*   This message should evaluate to True, and the file should be automatically approved.  *Do you understand the result? If not, discuss with another course participant or ask your instructor.* |

Exercise 8  
Deploy a new version of the rule

Given time, should you be waiting for the rest of the class to complete the mandatory exercises, use the Business Rules Composer to update (for example) the static value compared to the TotalOrdersToDate field to another number. Publish and Deploy the new version of the policy and see how it affects the outcome of the above tests in the way you would have expected. No changes need to be done to the orchestration and no host instance restart is necessary.