





Sample: Loan Application

Note

Before using this information and the product it supports, read the information in Notices at the end of this book.

First Edition (December 2006)

© Copyright International Business Machines Corporation 2006. All rights reserved.

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

Contents

Chapter 1. Introduction 1-1

Chapter 2. Working with the ready-made sample 2-1

Importing the ready-made sample	2-1
Running the ready-made sample	2-1

Chapter 3. Preparing to build the application. 3-1

Creating the loan application module	3-1
--	-----

Chapter 4. Implementing the assembly diagram 4-1

Viewing LoanApplicationModule in the assembly diagram	4-1
Adding components and stand-alone references	4-1
Adding the business process	4-2
Adding the stand-alone references	4-3
Adding the remaining components	4-3

Chapter 5. Creating the business objects and interfaces. 5-1

Creating business objects	5-1
Creating a business object for an applicant's contact information	5-2
Creating a business object for an applicant.	5-2
Creating a business object for a loan application	5-3
Creating a business object for an applicant's credit information	5-3
Creating a business object for human task information	5-3
Creating interfaces.	5-4
Creating mainProcessInterface	5-5
Creating LoanLimitsRuleInterface.	5-5
Creating CreditCheckInterface	5-6
Creating FollowUpPHTInterface.	5-6
Creating CompleteLoanHTInterface	5-6
Creating ProcessAppPHTInterface	5-7

Chapter 6. Adding interfaces to the components and wiring them. 6-1

Adding interfaces to the components	6-1
Adding the interface to the mainProcess component	6-1
Adding the interface to the CreditCheck component	6-1
Adding the remaining interfaces to the components	6-1
Making the components communicate	6-2
Wiring the process to the rule group.	6-2
Wiring stand-alone references to the process	6-2
Wiring the remaining components	6-2

Chapter 7. Implementing the business process 7-1

Generating the mainProcess component starting implementation.	7-2
Defining variables	7-3
Defining the HumanTaskData variable	7-3
Defining the automatic approval variables.	7-3
Defining the credit check variables	7-3
Preparing the loan request for processing	7-4
Assigning input to the credit check request variable	7-4
Assigning the other variables	7-4
Invoking a service to check credit.	7-5
Defining a case for approved loan requests	7-6
Creating the choice	7-6
Creating the condition snippet of a case	7-6
Acknowledging the good credit score	7-7
Creating a path for approved loan requests	7-7
Invoking the service to check automatic approval	7-7
Creating the choice	7-8
Creating the condition snippet of a case	7-8
Creating a path for automatically approved loan requests	7-8
Assigning an automatic response to the applicant	7-9
Assigning a task to the bank employee	7-9
Invoking the human task	7-9
Creating a path for manually approved loan requests	7-10
Creating the otherwise case	7-10
Assigning an automatic response to the applicant	7-10
Assigning a task to the bank employee	7-10
Invoking the human task	7-11
Creating a path for rejected loan requests.	7-11
Assigning an automatic response to the applicant.	7-11
Assigning a task to the bank employee	7-12
Invoking the human task	7-12

Chapter 8. Implementing business rules, a Java component, and human tasks 8-1

Creating and using business rules	8-1
Generating the rule group	8-1
Adding a rule set to check an applicant's credit rating	8-1
Adding the autoapproval rule set.	8-2
Implementing the CreditCheck Java Component	8-3
Implementing the human tasks	8-3
Implementing the FollowUpDeclinedApplication human task	8-3
Implementing the ProcessTheApplication and CompleteTheLoan human tasks	8-4

Chapter 9. Testing the loan application	9-1
Starting the application	9-1
Importing the EJBs.	9-5
Creating the web project.	9-5
Importing the EJBs.	9-6
Importing the JSPs.	9-6
Invoking the loan application with a JSP	9-7

Testing the loan application using the JSPs.	9-7
--	-----

Chapter 10. Summary	10-1
--------------------------------------	-------------

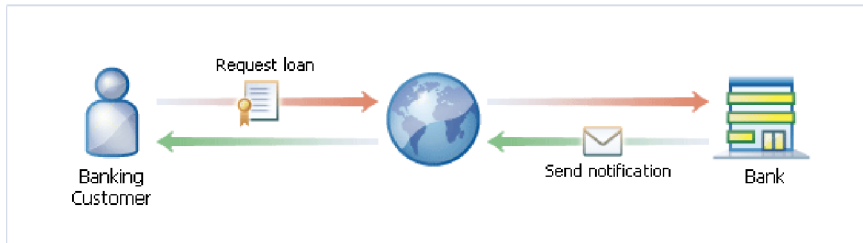
Notices	A-1
--------------------------	------------

Chapter 1. Introduction

The loan application that you create will be developed on the workbench using a top-down approach.

In order to receive a loan request through a web interface, it is necessary to create several artifacts to process the application. You need data objects that store information about the loan request, like the loan amount and name of the customer. To process the information and send a reply to the customer about the status of the request, you need to create a business process. To check the credit rating of the applicant and decide if you will approve or reject the request, you need to implement a business rule. And finally to process the request to a bank employee that executes the loan request, you need to create human tasks.

Overall Scenario



All of those artifacts are then "packed" as components and wired together using interfaces. The last step before you can use the application is to deploy it on a server.

To create your application, you will use the following seven editors:

Name	Icon	Purpose
assembly editor		Allows top-down development by creating and connecting components to form the integrated loan application.
business process editor		Helps you create the business logic for evaluating the loan application based on the applicant's credit rating and the business rules of the bank.
visual snippets editor		Helps you evaluate credit check conditions in the business process visually, and without writing any Java code.
interface editor		Helps you create interfaces that specify the means of accessing and returning output of components to other components in the assembly diagram.
business rules editor		Help you create the rules that define the bank's loan policies, such as when an application is approved or rejected.
rule group editor		Helps you select a business rule to execute based on the date or other criteria.
business object editor		Creates containers for business objects to represent the entities in the scenario. These include the customer or applicant, loan application, contact information for the applicant, and the data passed to bank administrators.

The following tasks will guide you through the steps and explain how to create the loan application.

Chapter 2. Working with the ready-made sample

You can import the ready-made application instead of building it yourself. This will help you get an overview of the artifacts you will create if you plan to build the application yourself. This section provides instructions on how to run the imported application in the WebSphere® Process Server Integrated Test Environment.

The following topics provide the step-by-step instructions on how to work with the ready-made sample.

Importing the ready-made sample

Import the completed module and resources required to run, test, and debug the loan application sample.

Click the following link to launch the import wizard:

Import the complete ready-made sample

In the wizard, click **Finish** to complete the import. The **LoanApplicationModule** opens in the Business Integration view.

Running the ready-made sample

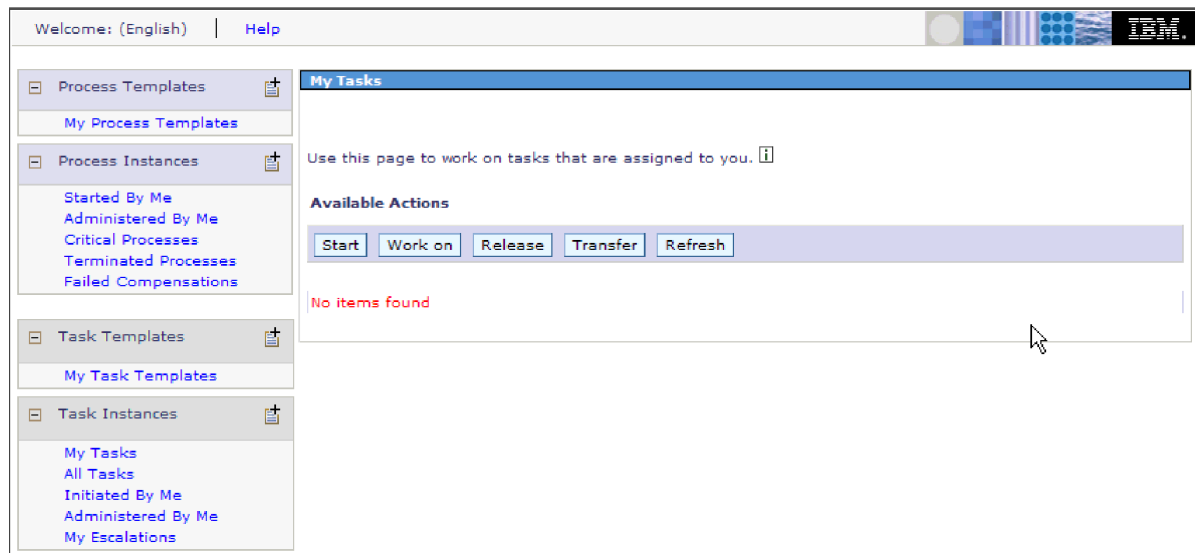
After you have imported the required resources, you can run the application in the test environment.

Note: You can also test the application using JSPs. For more information about testing an application using JSPs, see the instructions in “Invoking the loan application with a JSP” on page 9-7.

You will use the Business Process Choreographer (BPC) Explorer web client to run, test, and debug your application.

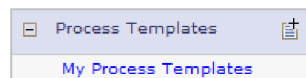
To run and test the loan application, follow these steps:

1. Add the application to the server.
 - a. In the Business Integration perspective, go to the **Servers** view.
 - b. Right-click **WebSphere Process Server**.
 - c. From the pop-up menu, select **Add and remove projects**. The Add and Remove Projects window opens.
 - d. In the navigation tree, click **LoanApplicationModuleApp**.
 - e. Click **Add** and then click **Finish**. It will take several minutes for the server to start and publish the loan application. Wait until “Application started: ...” is displayed in the console window.
2. Launch the Business Process Choreographer Explorer.
 - a. In the **Servers** view, right-click **WebSphere™ Process Server v6.0**.
 - b. From the pop-up menu, select **Launch → Business Process Choreographer Explorer**. The Business Process Choreographer Explorer opens and shows all tasks that are assigned to you. Currently there are no available tasks, so the “No items found” message is displayed.

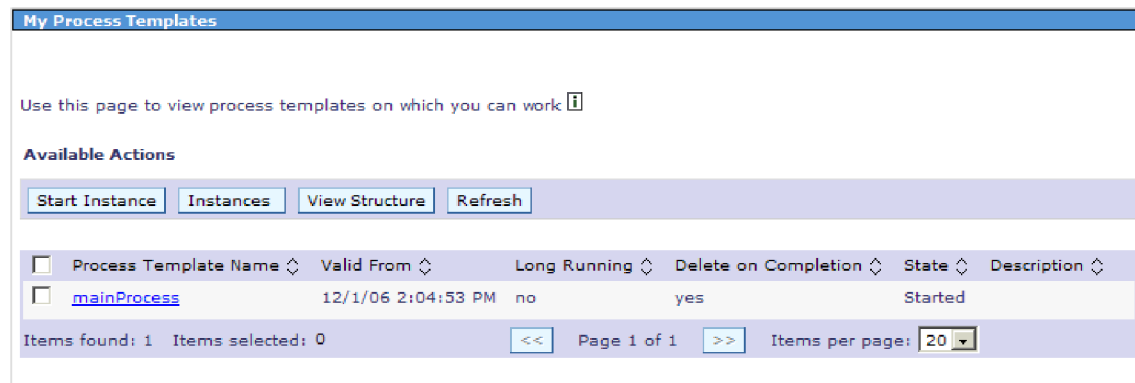


3. Invoke the application.

- a. To view a list of all process templates published to the server, click the **My Process Templates** label.



- b. To select the mainProcess template and start the loan application process, click the **mainProcess** label.



- c. To view all operations available to start the process, click the **Operations** tab. applyOnline is the only operation in the loan application that is available to start the process.
- d. Click the **applyOnline** label. A web-based front end opens where you can test your application.

Process Template

Use this page to view information about a process template [?](#)

Available Actions

[Instances](#) [View Structure](#)

Process Template Description

[Process Template Name](#) mainProcess
[Documentation](#)
[Description](#)

[Details](#) **Operations** [Process Instances](#) [Custom Properties](#) [Query Properties](#)

Operation Name	Activity Name	Description
applyOnline	Receive	

Items found: 1 [<<](#) Page 1 of 1 [>>](#) Items per page: 20

- e. Click the + button beside applicant to display the input fields where you will insert your test values.
- f. In the **name** field, type John Smith.
- g. In the **email** field, type jsmith@jsmith.com.
- h. In the **taxPayerId** field, type 1234567.

Note: If you want a non-random value for the credit score to test the approval cases, add "888" to the end of taxPayerId.

- i. In the **loanamount** field, type 60000.

Process Input Message

Use this page to provide the input that is needed to start an instance of a business process. [?](#)

Available Actions

[Submit](#)

Process Template Name mainProcess
Operation applyOnline
Process Input Message applicationInformation

applicant								
<table border="1"> <tr> <td>name</td> <td>John Smith</td> </tr> <tr> <td>email</td> <td>jsmith@smith.com</td> </tr> <tr> <td>taxPayerId</td> <td>1234567</td> </tr> <tr> <td></td> <td>-</td> </tr> </table>	name	John Smith	email	jsmith@smith.com	taxPayerId	1234567		-
name	John Smith							
email	jsmith@smith.com							
taxPayerId	1234567							
	-							
loanamount 60000								

Before you submit the request, be aware that the following rules apply for the applyOnline operation:

Loan request result	Credit rating*	Loan amount
automatically approved	>= 750	<= \$50 000
manually approved	>= 750	> \$50 000

Loan request result	Credit rating*	Loan amount
declined	< 750	N/A

* The credit rating is determined randomly rather than being pulled from a database or other file. If taxPayerId ends with "888" then the credit rating is always > 750.

- j. Click **Submit**. This loan application request is sent to the process for evaluation and a response is displayed, like in the image below:

Process Output Message

Use this page to view the results of a business process that you started. [i]

Process Template Name

mainProcess

Operation

applyOnline

Process Input Message

applicationInformation

applicant

name

John Smith

email

jsmith@smith.com

taxPayerId

1234567

loanamount

60000.0

Process Output Message

response

Thank you for your application. You will be contacted shortly to verify your information..

The message you receive concerning your application depends on your credit rating, and because of that it is determined randomly. Therefore, the message you receive will not always match the one displayed in the image above.

Response	Meaning	Human Task
"Thank you for your application. You will be contacted shortly to verify your information."	declined	FollowUpDeclinedApp
"Your application has been approved and is awaiting completion"	automatically approved	CompleteTheLoan
"Your application has been received and is under review."	manually approved	ProcessTheApplication

4. The loan request is submitted to the bank, and a bank employee will receive the data to follow up on the request. Now, you will switch into the role of the bank employee, who wants to work on the tasks sent to him or her.

- a. Click the **My Tasks** label. A list of tasks assigned to you is displayed.

Task Instances

My Tasks

All Tasks


Initiated By Me

Administered By Me

My Escalations

- b. Click the **FollowUpDeclinedApp** label in case your request is declined (see table above), which is the task assigned to you by the bank customer when they submitted the loan request.

My Tasks

Use this page to work on tasks that are assigned to you. 

Available Actions


Start Work on Release Transfer Refresh

<input type="checkbox"/> Task Name	State	Kind	Owner	Originator	Escalated	Suspended	Activated	Last Modified	Ex
<input type="checkbox"/> FollowUpDeclinedApp	Ready	Participating		UNAUTHENTICATED	no	no		12/4/06 5:39:45 PM	12/4/06 5:39:45 PM

Items found: 1 Items selected: 0 << Page 1 of 1 >> Items per page: 20

- c. Click the **Work On** button.
- d. Click the **Complete** button to complete the application follow-up.

Task Message

Use this page to provide the data required to complete the task. 

Available Actions

Complete Save Release Cancel

Task Name
Task Input Message

FollowUpDeclinedApp

TaskInformation

instruction	This declined application requires follow up..		
application	applicant	name	John Smith
		email	jsmith@smith.com
		taxPayerId	1234567
	loanamount	60000.0	

Task Output Message

No data available

Chapter 3. Preparing to build the application

The first step toward building the loan application is creating a module that will contain the application.

The following topics provide the step-by-step instructions on how to build the loan application:


Prerequisites

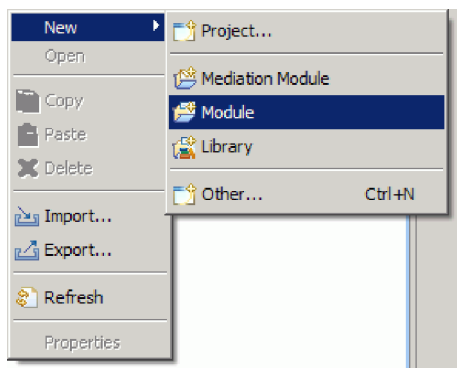
If you previously ran the ready-made application, delete that module and its contents before you proceed to build the application. Otherwise, components previously generated in the ready-made sample will still exist and will cause problems while building the application.

Creating the loan application module

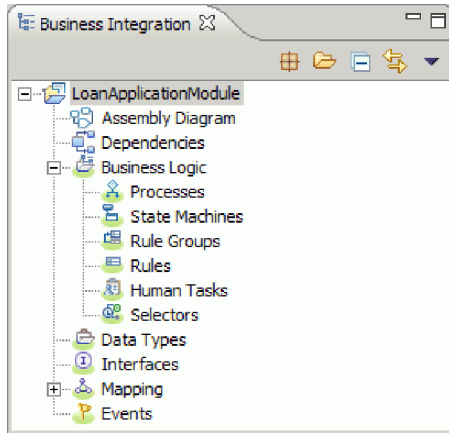
All the resources for this sample are stored in LoanApplicationModule. A module is a WebSphere Integration Developer project that is used for development, version management, organizing business service resources, and deploying to the WebSphere Process Server.

You will perform most of your work on the loan application sample from the Business Integration perspective.

1. Switch to the Business Integration perspective in the workbench. If the Business Integration perspective is not open, then you can open it as follows:
 - a. Click , which is the **Open Perspective** button.
 - b. Select the **Business Integration** perspective. To see a complete list of perspectives, select **Other** from the menu, and check the **Show all** check box.
2. Create a new module.
 - a. In the Business Integration view, right-click and select **New** → **Module**. The New Module Wizard opens.



- b. In the **Module Name** field, type LoanApplicationModule and click **Finish**. The structure for your new module is now created and available in the navigation tree of the Business Integration view. This is where you will find all the artifacts you develop for the loan application.
3. Browse the module structure:



You will interact with the following artifacts in the module structure:

- **Assembly Diagram:** You will model your application in the assembly diagram by creating services and wiring them together.
- **Business Logic:** Every artifact relating to the business and its policies is stored here. You will create a business process, a rule group containing rules, the rules that spell out the bank's loan policies, and human tasks that handle manual business functions.
- **Data Types:** Here you will find the business objects you create.
- **Interfaces:** Here you will find the interfaces you create to define the inputs and outputs of the services you see in the assembly diagram.

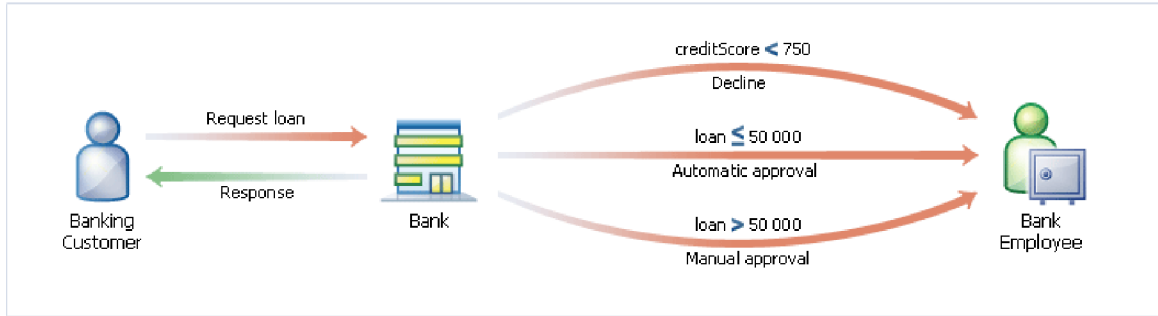
Next you will begin implementing the assembly diagram by adding the skeleton of services to your application.

Chapter 4. Implementing the assembly diagram

Create the fundamental elements of your first Service-Oriented Architecture (SOA) application.

The following image describes the business logic behind loan application:

Module Assembly



The left-side of this diagram shows the interaction between the banking customer and the bank. He requests a loan and receives a response. The banking customer is also referred to as the "loan applicant".

The right-side reveals how some of the bank policies affect the status of a loan request before it is reviewed by a bank employee.

You will create the components that allow this interaction in the assembly diagram.

Viewing LoanApplicationModule in the assembly diagram

In the assembly diagram, you will create components and wire them together to create the loan application.

Components are reusable business services, each with its own separate implementation.

If it is not already opened, open the LoanApplicationModule assembly diagram with the assembly editor by following these steps:

1. In the navigation tree of the Business Integration view, expand **LoanApplicationModule** and you will see several artifacts nested within it, the first of which is **Assembly Diagram**.
2. Double-click **Assembly Diagram** and the assembly editor will open.








Note: You can also see previously created artifacts by double-clicking their navigation tree item in the Business Integration view.

Next you will begin to compose the application by adding components to the assembly diagram.

Adding components and stand-alone references

To provide services in the application, you will add components in the assembly diagram. Each component has a specific task within the application and an interface to communicate with other components. To allow access to the business process from components outside of the module, you will add stand-alone references to the module.

Here is an overview of what you will be adding to the assembly diagram:



Component Type	Name	Purpose
Business Process 	mainProcess	You will create this central component to contain the business logic that evaluates the loan requests received by your application. mainProcess is a service that will access the other business services in the loan application, such as the rule group and human tasks, to accomplish this goal. Through a series of checks and choices that you create, the business process will direct the loan request to either approved or rejected.
Stand-alone References 	Stand-alone References	You will add Stand-alone References to use existing JavaServer Pages (JSPs) and Enterprise JavaBeans (EJBs) to interact with the components that you create in the loan application module. This will allow you to quickly test the loan application from a Web browser.
Rule Group 	LoanLimits	You will create business rules to determine, for example, what the minimum credit rating required to approve a loan is, or what the maximum loan request amount to be auto-approved is. These rules will be contained in the rule group.
Human Task 	FollowUpDeclinedApp	You will create a human task to delegate the responsibility of following up on a rejected loan to a bank employee, perhaps to provide references to credit counselling services. This action cannot be completed automatically, and requires the skills or authority of a person.
Human Task 	ProcessTheApplication	Another human task that you will create to handle the manual processing of a loan application.
Human Task 	CompleteTheLoan	Another human task that you will create to acknowledge the auto approval of a loan application.
Java 	CreditCheck	You will create this Java component to use an existing Java code implementation that generates a random value for an applicant's credit rating.

The following tasks show you how to add the components and stand-alone references that the loan application requires.

Adding the business process

The mainProcess component provides a business process that executes the business logic, including receiving the loan request, checking the loan amount, sending a reply and delegating the request to a bank employee.



To add the process component, follow these steps:

1. In the assembly editor, click , which is the **Component (with no implementation type)** icon, and then click , which is the **Process** icon. The icon displayed in the palette will always be the most recently selected from that category.
2. Click the canvas. **Component1** is added. The exclamation mark in the lower-left corner indicates that the implementation for this component has not yet been created. You will create the implementation for each component later on.
3. Rename Component1 to mainProcess by clicking the component and typing over the highlighted text.

Adding the stand-alone references






Stand-alone references stand alone, which means that their implementation resides outside of the module, and they allow existing applications outside of your module to invoke the services of the components contained in the module. In this sample, the bank customer needs a web interface to process its data to the bank. To do that, you need a standalone-reference to implement this interface.

To add stand-alone references so that you can invoke mainProcess from a JSP later, follow these steps:

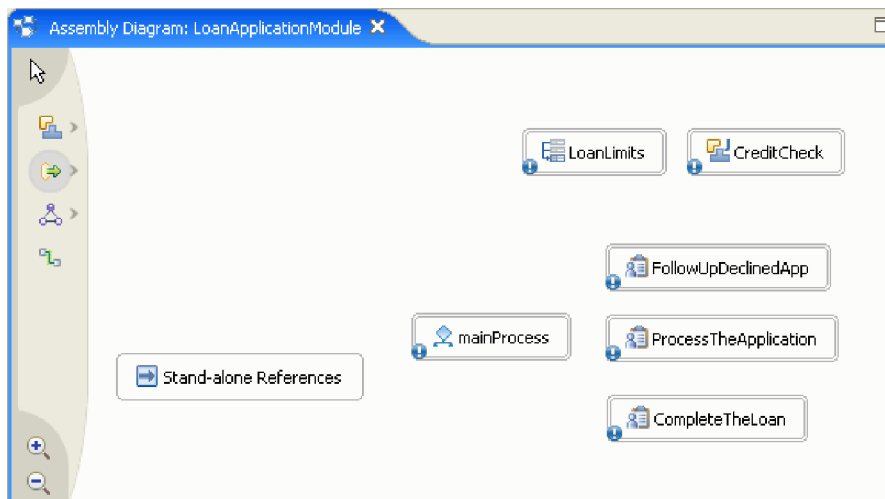
1. Switch to the assembly editor.
2. Click , which is the **Import** icon, and then click , which is the **Stand-alone References** icon.
3. Click the canvas. The **Stand-alone References** figure is added.

Adding the remaining components

1. Add the remaining components to the assembly diagram using the following information:

Component Type	Name
Rule Group 	LoanLimits
Human Task 	FollowUpDeclinedApp
Human Task 	ProcessTheApplication
Human Task 	CompleteTheLoan
Java 	CreditCheck

2. Click **File** → **Save**. The Assembly Diagram should now look similar to this:

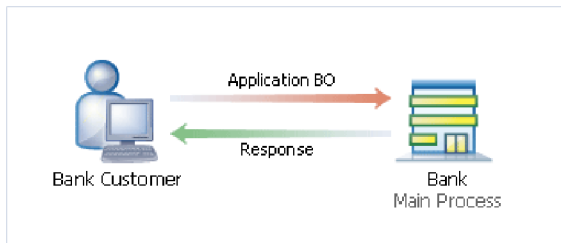


Next you will create the objects and interfaces that facilitate communication between each of the components that you just added to the assembly diagram.

Chapter 5. Creating the business objects and interfaces

You will define the information units needed to request and approve loan applications, and create the interfaces that will allow you to integrate your loan application components together.

Business Objects



In the image above you can see how to use business objects. A bank customer sends his request by providing information about the loan amount and his name. This information is transferred via the ApplicationBO to the bank. In the following topic you will create all the needed business objects for your loan application.

Interfaces

After you have created the business objects, you need a channel to pass the business object through from one component to another. These channels are called interfaces and they define what data flows from one component to another (input) and in the opposite direction (output).

Creating business objects

You will create business objects to hold data passed between components in the application.

For example, if a bank customer requests a loan, they will need to provide details such as their e-mail and ID. These attributes are stored, with names like email and taxPayerId, in the business objects that you create for your loan application, along with their data types. The business objects act as the currency in your application, and are transferred through interfaces to other components.

Here is an overview of the business objects you will create:


Name	Purpose
ContactBO	You will create this business object to hold contact details of the loan applicant after they have entered their data online.
ApplicantBO	This business object is a specialized ContactBO that also holds the tax payer identification of the applicant.
ApplicationBO	This business object represents the loan application submitted by the applicant online.
CreditBO	This business object contains the credit information for the applicant.
HumanTaskBO	This business object is sent to the bank. It includes the application and instructions for a bank employee to follow when evaluating that particular loan.

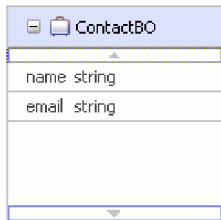
The following tasks show you how to create the business objects required for the loan application.

Creating a business object for an applicant's contact information

ContactBO is a business object that contains contact information, both name and e-mail, for a loan applicant.

To create ContactBO, follow these steps:

1. In the Business Integration view, right-click **LoanApplicationModule** and click **New** → **Business Object**.
2. In the **Name** field, type ContactBO. Leave the **Inherit from** field as <none> and click **Finish**. The editor opens for ContactBO.
3. Click  , which is the **Add an attribute to a business object** button. An attribute named attribute1 is created.
4. Rename attribute1 to name by clicking the figure and typing over the highlighted text. Keep the default type of string.
5. Following the steps above, add another attribute named email of type string.
6. Click **File** → **Save** and then close the business object editor.

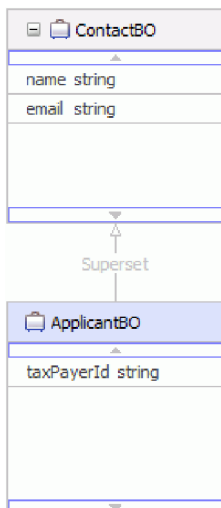


Creating a business object for an applicant

ApplicantBO is a business object that represents a loan applicant in the process. This business object only needs to have one attribute created, which is for the tax payer identification of an applicant. The name and email attributes are inherited by ApplicantBO to reduce duplication, and it therefore treats those attributes as if they were part of ApplicantBO.

To create ApplicantBO, follow these steps:

1. Create a new business object named ApplicantBO that inherits from ContactBO, and then complete the business object to match the following:

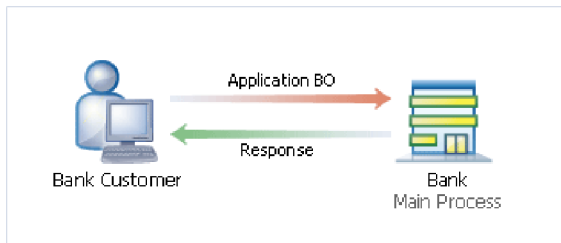


2. Click **File** → **Save** and then close the business object editor.

Creating a business object for a loan application

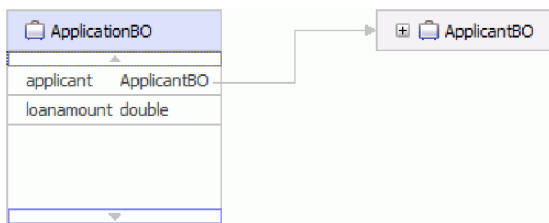
ApplicationBO is a container for the loan application data passed through the business process. This business object has all of the applicant's information stored in its applicant attribute, and it has the amount of the loan in the loanamount attribute.

ApplicationBO is a complex business object since it uses another business object, in this case ApplicantBO, as the type for one of its attributes.



To create ApplicationBO, follow these steps:

1. Create a new business object named ApplicationBO, and then complete the business object to match the following:



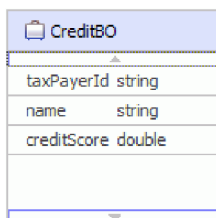
2. Click **File** → **Save** and then close the business object editor.

Creating a business object for an applicant's credit information

CreditBO is a business object that holds the loan applicant's credit information. It has identifying attributes of the applicant, such as name and taxPayerId, but also an additional attribute that holds the applicant's credit score. This credit score will be obtained from the CreditCheck component.

To create CreditBO, follow these steps:

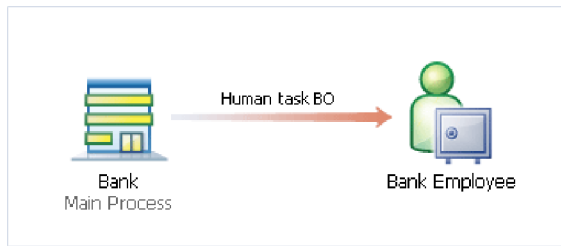
1. Create a new business object named CreditBO, and then complete the business object to match the following:



2. Click **File** → **Save** and then close the business object editor.

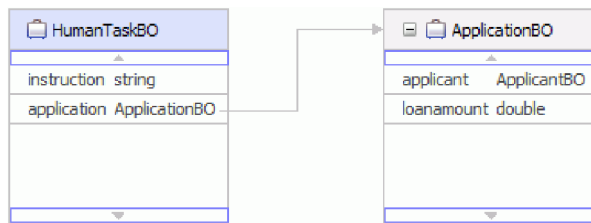
Creating a business object for human task information

HumanTaskBO is a business object that is passed to a bank employee, containing the application in an ApplicationBO, and a string of instructions for the bank employee to follow in order to complete the loan application process.



To create HumanTaskBO, follow these steps:

1. Create a new business object named HumanTaskBO, and then complete the business object to match the following:



2. Click **File** → **Save** and then close the business object editor.

Next you will use the business objects as inputs and outputs in the interfaces that you create for each component.

Creating interfaces

You create the interfaces to specify what data can be passed between the loan application components.

Interfaces provide the ability to process information received from other components or services. For example, the bank customer processes information to the bank (input) and receives a reply (output). The following table shows you the interfaces you will create.

Name	Component	Purpose
mainProcessInterface	mainProcess	Operation: applyOnline Input: customer provides information for this loan request to the bank (applicationInformation) Output: customer receives a reply from the bank about his loan request (response)
LoanLimitsRuleInterface	LoanLimits	Operation: checkCredit Input: customer information is sent from the process to check the customer's credit rating against the business rules (applicantInformation) Output: process receives a reply indicating whether the credit rating was high enough to accept the loan request (response) Operation: autoapproval Input: loan request is sent from the process to check the loan amount against the business rules (applicationInformation) Output: process receives a reply indicating whether the loan amount was low enough to automatically approve the request (response)


Name	Component	Purpose
CreditCheckInterface	CreditCheck	Operation: checkCredit Input: tax payer identification is sent from the rule group to get a generated credit score (taxPayerId) Output: rule group receives a reply with the generated credit score of the applicant (creditScore)
FollowUpHTInterface	FollowUpDeclinedApp	Operation: FollowUpDeclinedApp Input: customer information and instructions are sent from the process to the bank employee for following up on declined loans (TaskInformation)
CompleteLoanHTInterface	CompleteTheLoan	Operation: CompleteTheLoan Input: customer information and instructions are sent from the process to the bank employee for reviewing automatically approved loans. (TaskInformation).
ProcessAppHTInterface	ProcessTheApplication	Operation: ProcessTheApplication Input: customer information and instructions are sent from the process to the bank employee for manually approving loans.


To create the interfaces for each component and the required operations, follow the tasks outlined below.

Creating mainProcessInterface

mainProcessInterface is the connection between the stand-alone references and the mainProcess component. It has the applyOnline operation which receives the loan request and responds with a message based on the status of the loan application.

To create mainProcessInterface, which will be added to the mainProcess component, follow these steps:

1. In the Business Integration view, right-click **LoanApplicationModule** and click **New** → **Interface**.
2. In the **Name** field, type mainProcessInterface and then click **Finish**. The editor opens for mainProcessInterface.
3. Click , which is the **Add Request Response Operation** button. This operation type provides an input and output to process information in both directions. operation1 is added.
4. Rename operation1 to applyOnline by typing over the highlighted text.
5. Rename input1 to applicationInformation. Click on the default type of string and, from the drop-down box, change the type to ApplicationBO.
6. Rename output1 to response. Keep the default type of string.
7. Click **File** → **Save** and then close the interface editor.

	Name	Type
▼ 	applyOnline	
Input(s)	applicationInformation	ApplicationBO
Output(s)	response	string

Creating LoanLimitsRuleInterface

LoanLimitsRuleInterface is the connection between the mainProcess component and the LoanLimits rule group. LoanLimitsRuleInterface has the checkCredit and autoapproval operations. The checkCredit

operation is involved in determining an applicant's credit score. The autoapproval operation takes an application and indicates whether it should be approved outright.

To create LoanLimitsRuleInterface, follow the steps that you used to create mainProcessInterface and add the following operations:

1.

	Name	Type
▼	checkCredit	
Input(s)	applicantInformation	ApplicantBO
Output(s)	response	boolean
▼	autoapproval	
Input(s)	applicationInformation	ApplicationBO
Output(s)	response	boolean

2. Click **File** → **Save** and then close the interface editor.

Creating CreditCheckInterface

CreditCheckInterface is the connection between the LoanLimits rule group and the CreditCheck component. CreditCheckInterface has the checkCredit operation, which returns the credit score of the applicant, identified by his or her taxpayerId.

To create CreditCheckInterface, add the following operation:

1.


	Name	Type
▼	checkCredit	
Input(s)	taxpayerId	string
Output(s)	creditScore	CreditBO

2. Click **File** → **Save** and then close the interface editor.

Creating FollowUpHTInterface

FollowUpHTInterface is the connection between the mainProcess component and the FollowUpDeclinedApp human task. FollowUpHTInterface has the FollowUpDeclinedApp operation. You use this interface for when a loan application needs to be reviewed after it is declined.

To create FollowUpHTInterface, which has a different type of operation than the previous interfaces, follow these steps:

1. Create an interface from the Business Integration view as you did for the previous interfaces.
2. Click , which is the **Add One-Way Operation** button. operation1 is added.
3. Complete the operation as shown below.

	Name	Type
▼	FollowUpDeclinedApp	
Input(s)	TaskInformation	HumanTaskBO

4. Click **File** → **Save** and then close the interface editor.

Creating CompleteLoanHTInterface

CompleteLoanHTInterface is the connection between the mainProcess component and the CompleteTheLoan human task. CompleteLoanHTInterface has the CompleteTheLoan operation. You use this interface when a loan application needs to be reviewed after it is completed.

To create CompleteLoanHTInterface, add the following operation:

1.

	Name	Type
▼ CompleteTheLoan		
Input(s)	TaskInformation	HumanTaskBO

2. Click **File** → **Save** and then close the interface editor.

Creating ProcessAppHTInterface

ProcessAppHTInterface is the connection between the mainProcess component and the ProcessTheApplication human task. ProcessAppHTInterface is an interface with the ProcessTheApplication operation. You use this interface when an application needs to be manually processed and approved.

To create ProcessAppHTInterface, add the following operation:

1.

	Name	Type
▼ ProcessTheApplication		
Input(s)	TaskInformation	HumanTaskBO

2. Click **File** → **Save** and then close the interface editor.

Next you will add the interfaces you just created to components in the assembly diagram to expose their business functions, and then wire the components together so that they can communicate.

Chapter 6. Adding interfaces to the components and wiring them

Now you have created the components, business objects, and interfaces. You have also related the business objects to the interfaces. The next step is for you to bind the interfaces to the components and wire them together, so that the components can communicate with each other.


Adding interfaces to the components

Now you will wire all created components together using the related interfaces. To accomplish this, you will add one or more interfaces to the component and then wire the related components together in the assembly editor.

The following tasks show you how to add the interfaces to the components:


Adding the interface to the mainProcess component

To add mainProcessInterface to the mainProcess component, follow these steps:

1. Switch to the assembly editor.
2. Click the **mainProcess** component.
3. Click , which is the **Add Interface** button. The Add Interface window opens.
4. From the list, select mainProcessInterface and click **OK**.





Adding the interface to the CreditCheck component

To add CreditCheckInterface to the CreditCheck component:

1. In the assembly editor, click the **CreditCheck** component.
2. Click , which is the **Add Interface** button. The Add Interface window opens and you will see a list in the window with Java and Web Services Description Language (WSDL) interfaces that are available for you to use in your applications. WSDL is the default interface type in WebSphere Integration Developer, used for describing web services.
3. Only the WSDL interfaces will be used in the loan application, so you may reduce the number of choices by clicking **Show WSDL**. From the reduced list, select CreditCheckInterface and click **OK**.

Adding the remaining interfaces to the components

To add LoanLimitsRuleInterface to the LoanLimits component, and interfaces to the human tasks:

Component	Interface Added
LoanLimits 	LoanLimitsRuleInterface
FollowUpDeclinedApp 	FollowUpHTInterface
ProcessTheApplication 	ProcessAppHTInterface
CompleteTheLoan 	CompleteLoanHTInterface

Now that you have interfaces bound to each component, you will wire the components together.

Making the components communicate

To enable components to communicate with each other through operations defined in interfaces, you wire components in the assembly diagram.

By connecting a source and a target component in the assembly diagram, the source can access the services of the target. The source components call other components via partner references shown on the right side of the component, and through operations defined by the interfaces on the targets that you added in the previous section.

For example, in order for the process to receive an ApplicationBO as defined in the applyOnline operation of mainProcessInterface, it needs to be connected with Stand-alone References so that operation is available to be invoked when a bank customer requests a loan from the bank's Web site.

The tasks below show you how to wire all the components in the assembly diagram.

Wiring the process to the rule group

You will wire mainProcess to the rule group so that the bank rules can be applied to an ApplicantBO, and then a boolean response received will indicate if a loan was rejected, or if it was manually or automatically approved.

To wire the mainProcess component to the LoanLimits rule group, follow these steps:

1. Position the mouse over the border of the mainProcess component until a yellow handle is displayed.
2. Click the yellow handle and drag it to connect with the LoanLimits component. The Add Wire dialog box is displayed.
3. To create a new partner reference on mainProcess, click **OK**. A wire connects the two components through the partner reference on mainProcess and the interface of LoanLimits.

Wiring stand-alone references to the process

To access the services of the application through a JSP, you connect the stand-alone references with the business process. Through this communication, an ApplicationBO is sent to the process to be checked, and a string response is received back to the JSP.

To connect stand-alone references to mainProcess, follow these steps:

1. Connect the Stand-alone References to the mainProcess component. The Add Wire dialog box is displayed.
2. To create a matching reference on Stand-alone References, click **OK**.
3. Another Add Wire dialog box will appear asking if you would like to convert WSDL interfaces to Java interfaces. Because you will test the loan application with JSPs and they are Java-based, it will be faster to invoke the module through Java interfaces rather than convert them first from WSDL interfaces. Click **Yes**. The new wire is created.

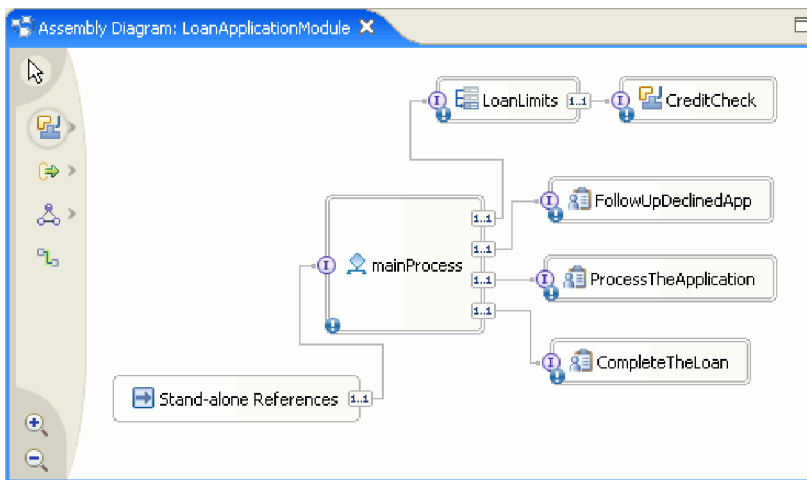
Wiring the remaining components

Now that you know how to use the wiring tool, connect the following components together in the assembly diagram:

1.

Source	Target	Communication
LoanLimits	CreditCheck	Operation: checkCredit Input: taxpayerId (string) Output: creditScore (CreditBO)
mainProcess	FollowUpDeclinedApp	Operation: FollowUpDeclinedApp Input: TaskInformation (HumanTaskBO)
mainProcess	ProcessTheApplication	Operation: ProcessTheApplication Input: TaskInformation (HumanTaskBO)
mainProcess	CompleteTheLoan	Operation: CompleteTheLoan Input: TaskInformation (HumanTaskBO)

2. Click **File** → **Save**. The assembly diagram should now look like this:



3. Close the assembly editor.

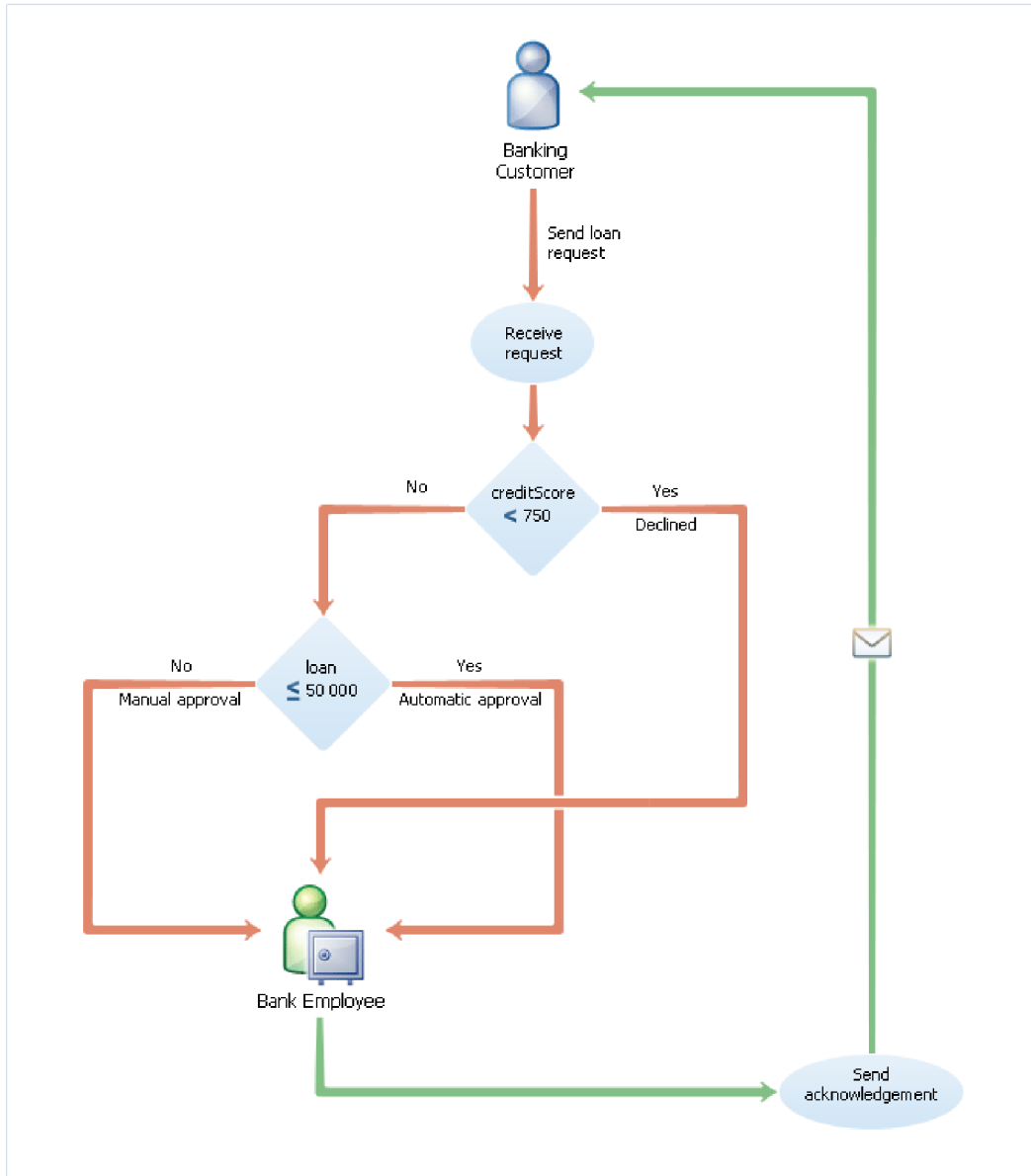
Now that each component has been wired together, and the operations are complete, you will apply the business logic by implementing the process.

Chapter 7. Implementing the business process

Implement the process that receives and checks the loan request.

The following image describes how you will implement the business process for the loan application:

Business Process



After the loan request is received, the applicant's credit score will be checked against a business rule of the bank. Credit score is randomly determined per customer, depending on the taxPayerId. An applicant with a taxPayerId that ends with "888" is always trusted, and is therefore assigned a credit score of 888. The business rule states that any application submitted by an applicant who has a credit score of 750 or less, will be rejected.

If the applicant's credit score is above 750, there will be another evaluation, this time against the loan amount indicated in the application. If the loan amount is less than \$50 000, the process is set up to

automatically approve that loan. However, if the loan amount exceeds \$50 000, it will need to be manually approved by a bank employee. In every case, a bank employee is involved in the process, even if the application was automatically approved.

To create the process that will handle the implementation of the loan application business logic as outlined above, complete the following tasks.

Generating the mainProcess component starting implementation

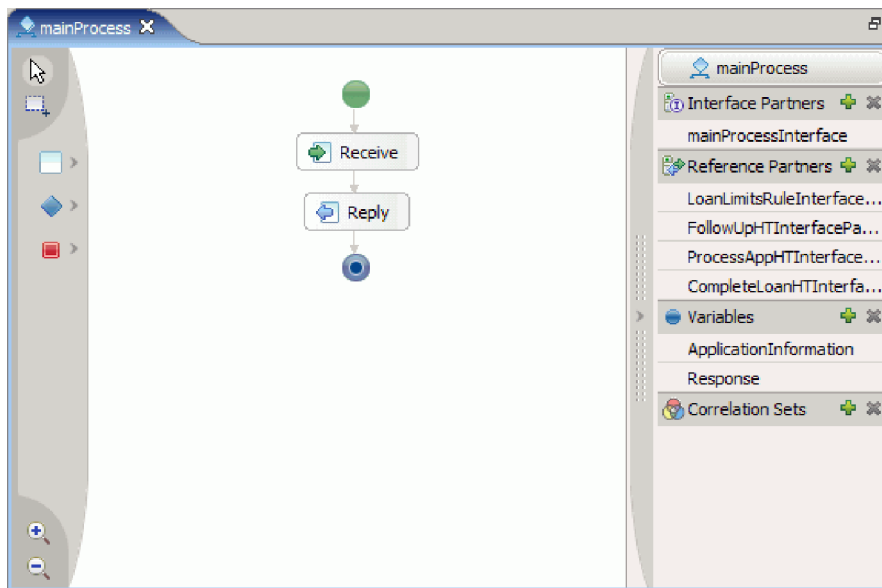
The mainProcess component contains the business logic for the loan application.

You will use the process editor to create and edit the loan application business process, represented visually as a series of activities, and modeled using Business Process Execution Language (BPEL). This will include implementing the following steps in the loan application process:

- Receiving the loan request
- Performing a check of the applicant's credit rating (< 750)
- Performing an automatic approval request (<= \$50,000)
- Processing the request with a bank employee
- Sending an acknowledgment to the bank customer

To generate the mainProcess component, which is the process in this application, follow these steps:

1. Right-click the **mainProcess** component.
2. From the pop-up menu, select **Generate Implementation**. The Generate Implementation window opens.
3. In the navigation tree, click **LoanApplicationModule** and click **OK**. The process editor opens with a basic business process.



4. Click **File** → **Save**.

Recall that the mainProcess component has an interface, mainProcessInterface, with a single operation named applyOnline. The business process displayed was generated based on this operation.

The applyOnline operation defines one input and one output. The input is mapped to the **Receive** activity. When mainProcessInterface transfers the data for the bank customer's loan request, it arrives at the **Receive** activity and is stored in the request variable, **ApplicationInformation**. The output is mapped

to the **Reply** activity. The response message stored in the **Response** variable is transferred back to the bank customer using the **Reply** activity, informing the customer about the status of the loan request.

Next you will create the variables to hold the business object data used in the process.

Defining variables

Create the variables used to store the data that is exchanged between the mainProcess component and other components it communicates with.


For example, to transfer information about the loan request to a bank employee, you need to create a HumanTaskData variable. You will use this variable for the interface between the mainProcess and the human task, implemented in separate components.

To define the variables used in the mainProcess implementation, follow the tasks below.

Defining the HumanTaskData variable

To pass the instructions on how to handle each loan application to a bank employee, we store those instructions in a variable, HumanTaskData, of type HumanTaskBO.

To define the HumanTaskData variable, follow these steps:

1. Click  , which is the **Add Variable** button. Variable is added to the Variables list.
2. Rename Variable to HumanTaskData by typing over the highlighted text.
3. In the **Properties** view, click **Details**.
4. Click **Browse**. The Data Type Selection window opens.
5. From the list, select HumanTaskBO and click **OK**.
6. Click **File** → **Save**.

Defining the automatic approval variables

You use the AutoApprovalRequest variable for the input when invoking the autoapproval operation, and store the response in the AutoApprovalResponse, for use in the process.

1. To create the automatic approval variables, define them as follows:

Name	Type	Data Type
AutoApprovalRequest	Data Type	ApplicationBO
AutoApprovalResponse	Data Type	boolean

2. Click **File** → **Save**.

Defining the credit check variables

You use the CreditCheckRequest variable for the input when invoking the checkCredit operation, and store the response in the CreditCheckResponse variable, for use in the process.

1. To create the credit check variables, define them as follows:

Name	Type	Data Type
CreditCheckRequest	Data Type	ApplicantBO
CreditCheckResponse	Data Type	boolean

2. Click **File** → **Save**.

Now that you have all the variables defined, you will begin assigning them initial values to provide input for operations invoked from the business process.

Preparing the loan request for processing



You will take the received ApplicationBO and assign its attributes to variables, so you can invoke operations from within the process, using the data stored in the variables as input.

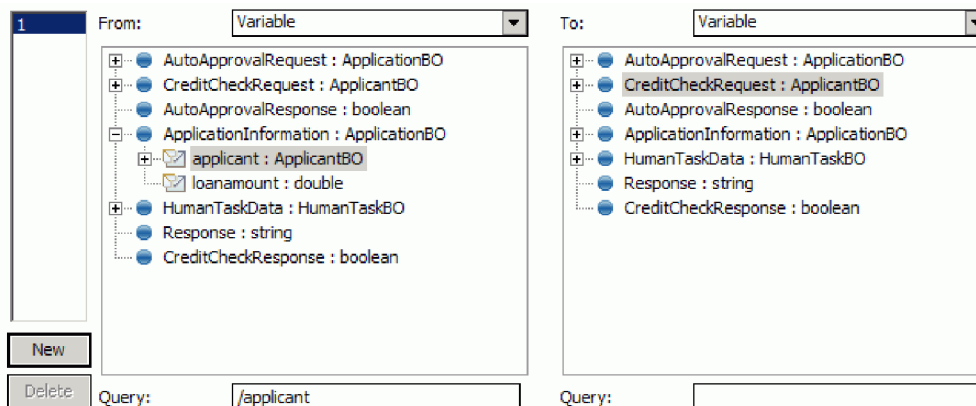
To copy the received data into variables, follow the tasks below.

Assigning input to the credit check request variable

You need to initialize the CreditCheckRequest variable with an applicant's information from an ApplicantBO, so that the creditCheck operation can be invoked.

To copy the applicant information to the CreditCheckRequest variable, follow these steps:

1. Switch to the process editor.
2. Click , which is the **Empty Action** icon and then click , which is the **Assign** icon.
3. Click the connection between **Receive** and **Reply**. The Assign activity was added.
4. Rename Assign to CopyInput by typing over the highlighted text.
5. In the Properties view, click **Details**.
6. In the **From** list, select Variable.
7. In the navigation tree, expand **ApplicationInformation : ApplicationBO** → **applicant : ApplicantBO**. The **Query** field is set to /applicant
8. In the **To** list, select Variable.
9. In the navigation tree, expand **CreditCheckRequest : ApplicantBO**. The **Query** field is empty. You should see the following assignment:
- 10.



11. Click **File** → **Save**.

Assigning the other variables

You will assign the remaining variables with initial values. You can assign multiple variables using a single Assign activity.

There are two variables that we need to assign:

- **HumanTaskData:** This variable needs to be initialized with the loan application from an ApplicationBO, so that the process can send instructions to the bank employees.
- **AutoApprovalRequest:** This variable needs to be initialized with the applicant's information from an ApplicationBO, so that the autoapproval operation can be invoked.

To assign the variables, follow these steps:

1. Click the **New** button. A second assignment is created and added to the number list. You can use that list to jump between each assignment.
2. Assign the new variable, as follows:

From	From Selection	From Query
Variable	ApplicationInformation : ApplicationBO	(none)

To	To Selection	To Query
Variable	HumanTaskData : HumanTaskBO → application : ApplicationBO	/application

3. Click the **New** button again. A third assignment is created and added to the number list.
4. Assign the new variable, as follows:

From	From Selection	From Query
Variable	ApplicationInformation : ApplicationBO	(none)

To	To Selection	To Query
Variable	AutoApprovalRequest : ApplicationBO	(none)

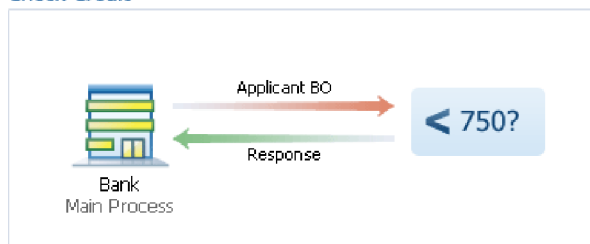
5. Click **File** → **Save**.

Now that you have initialized the variables with the correct business objects, you will begin to invoke other components from within the process.




Invoking a service to check credit


Define an activity in the process that will call an operation to check if an applicant is eligible for a loan.

Check Credit



To add the invoke activity to the CreditCheck, follow these steps:

1. Switch to the process editor.
2. Click , which is the **Assign** icon, and then click , which is the **Invoke** icon.
3. Click the connection between **CopyInput** and **Reply**. The **Invoke** activity is added.
4. Rename Invoke to CheckCredit.
5. In the Properties view, click **Details**.
6. Click **Browse**. The Select a Partner window opens.
7. From the list, select LoanLimitsRuleInterfacePartner and click **OK**. checkCredit will be selected as the **Operation**.
8. Check the **Use Data Type Variables** checkbox.
9. Click the first  button. The Select Variable for applicantInformation window opens.

10. From the list, select `CreditCheckRequest` and click **OK**.
11. Click the second  button. The Select Variable for response window opens.
12. From the list, select `CreditCheckResponse` and click **OK**.
13. Click **File** → **Save**.

You will now take the reply from the invoke activity and use it to direct the application down one of two paths in the process, to either loan approval or loan declination.

Defining a case for approved loan requests


Define a choice activity with a case to divide the process into separate paths. The loan request will follow one of two paths, either for loan approval or loan declination, based on the result of a boolean expression.

To add a case to a choice activity, follow the tasks outlined below:

Creating the choice

You will add a choice to split the process into one path for approving the loan application and another path for declining the loan application.

To add the choice, follow these steps:

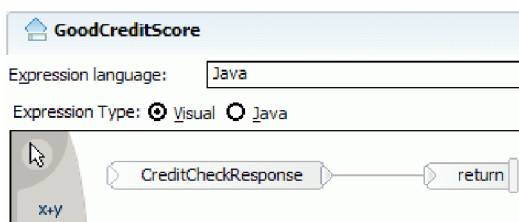
1. In the process editor, click  , which is the **Choice** icon.
2. Click the connection between **CheckCredit** and **Reply**. The **Choice** activity is added.
3. Rename Choice to `ScoreEvaluation` by typing over the highlighted text.
4. Click **File** → **Save**.

Creating the condition snippet of a case

You need to create a condition statement that will evaluate the applicant's credit score and then choose which path in the process to follow. You can create the condition statement using visual snippets or using Java code.

To create the condition visual snippet, follow these steps:

1. In the process editor, click the **Case** label.
2. In the Properties view, click **Description**.
3. In the **Display Name** field, type `GoodCreditScore`.
4. In the Properties view, click **Details**.
5. From the **Expression language** list, select `Java`.
6. For the **Expression Type**, select `Visual`.
7. In the canvas, click on **true** and replace it with `CreditCheckResponse`. It should still be connected to **return** and look like the following image:





8. Click **File** → **Save**.

Acknowledging the good credit score

After you add the GoodCreditScore case, you need to create an activity that acknowledges the receipt of a good credit score and sends out an automatic response to the loan applicant.

To add the AcknowledgeReceipt activity, follow these steps:

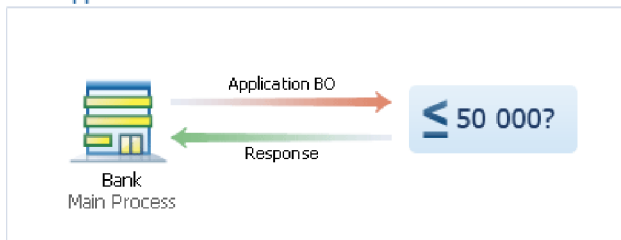
1. In the process editor, click , which is the **Invoke** icon, and then click , which is the **Assign** icon.
2. Click the **GoodCreditScore** label. The Assign activity is added.
3. Rename Assign to AcknowledgeReceipt.
4. In the Properties view, click **Details**.
5. From the **From** list, select Fixed Value.
6. In the **Description** field, type Your application has been received and is being processed..
7. From the **To** list, select Variable.
8. In the navigation tree, select **Response : string**. The **Query** field is empty.
9. Click **File** → **Save**.

Next you will create another set of activities in the process to decide if the loan request will be automatically or manually approved.

Creating a path for approved loan requests

You will create a path in the process for approved loan requests to follow, after confirming that the loan applicant has a good credit rating. You will then continue the path with another division in the process, this time creating one path for automatic approval of a loan and another for manual approval.

Auto Approval







To create a path in the process for approved loan requests and checking for automatic approval, follow the tasks below:

Invoking the service to check automatic approval

Invoke the LoanLimits component to evaluate if the loan request is eligible for automatic approval.

To add an activity to invoke the service, follow these steps:


1. In the mainProcess editor, click , which is the **Assign** icon and then click , which is the **Invoke** icon.
2. Click below the **AcknowledgeReceipt** activity. The Invoke activity is added.
3. Rename Invoke to CheckAutoApproval.
4. In the **Properties** view, click **Details**.
5. Click **Browse**. The Select a Partner window opens.
6. From the list, select LoanLimitsRuleInterfacePartner and click **OK**.
7. From the **Operation** list, select autoapproval.
8. Check the **Use Data Type Variables** checkbox.

9. Click the first  button. The Select Variable for applicationInformation window opens.
10. From the list, select `AutoApprovalRequest` and click **OK**.
11. Click the second  button. The Select Variable for response window opens.
12. From the list, select `AutoApprovalResponse` and click **OK**.
13. Click **File** → **Save**.

Creating the choice

You will add a choice to split the process into one path for approving the loan request automatically and another path for approving the loan request manually.

To add the choice, follow these steps:

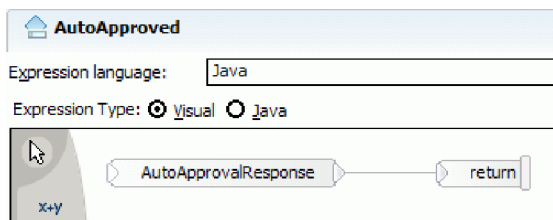
1. In the mainProcess editor, click , which is the **Choice** icon.
2. Click below **CheckAutoApproval**. The **Choice** activity is added.
3. Rename Choice to `AutoApprovalTest` by typing over the highlighted text.
4. Click **File** → **Save**.

Creating the condition snippet of a case

You need to create a condition statement that will examine the loan amount in the application and then choose which path in the process to follow. You can create the condition statement using visual snippets or using Java code.

To create the condition visual snippet, follow these steps:

1. In the process editor, click the **Case** label.
2. In the Properties view, click **Description**.
3. In the **Display Name** field, type `AutoApproved`.
4. In the Properties view, click **Details**.
5. From the **Expression language** list, select `Java`.
6. For the **Expression Type**, select `Visual`.
7. In the canvas, click on **true** and replace it with `AutoApprovalResponse`. It should still be connected to **return** and look like the following image:



8. Click **File** → **Save**.

Now that you have completed the choice and invoke activities, you will create the next two paths needed for the loan approval.

Creating a path for automatically approved loan requests



You will create a path that handles the automatic approval of an applicant's loan if they have a good credit rating.

To create a path for the automatically approved loan requests, follow the tasks below.

Assigning an automatic response to the applicant

Assign a message to the Response variable that is sent automatically back to the applicant, indicating that their application was automatically approved.


To assign the response, follow these steps:

1. In the process editor, click , which is the **Invoke** icon and then click  which is the **Assign** icon.
2. Click the **AutoApproved** label. The Assign activity is added.
3. Rename Assign to Approved.
4. In the Properties view, click **Details**.
5. In the **From** list, select Fixed Value.
6. In the **Description** field, type Your application has been approved and is awaiting completion..
7. In the **To** list, select Variable.
8. In the navigation tree, select **Response : string**. The **Query** field is empty.
9. Click **File** → **Save**.

Assigning a task to the bank employee

Assign an instruction to the HumanTaskData variable which is then sent to the bank employee.




To assign the instruction follow these steps:

1. In the process editor, click , which is the Assign icon.
2. Click below the **Approved** figure . The Assign activity is added.
3. Rename Assign to HumanCompletion.
4. In the Properties view, click **Details**.
5. In the **From** list, select Fixed Value.
6. In the **Description** field, type This auto approved application needs to be completed..
7. In the **To** list, select Variable.
8. In the navigation tree, expand **HumanTaskData : HumanTaskBO** → **instruction : string**. The **Query** field is set to /instruction.
9. Click **File** → **Save**.

Invoking the human task

You will invoke the CompleteTheLoan human task component from the process so that the bank employee can finish the loan.

To create the activity that will invoke the human task, follow these steps:

1. In the process editor, click , which is the **Assign** icon and then click , which is the **Invoke** icon.
2. Click below the **HumanCompletion** activity. The Invoke activity is added.
3. Rename Invoke to CompleteTheLoan.
4. In the Properties view, click **Details**.
5. Click **Browse**. The Select a Partner window opens.
6. In the list, select CompleteLoanHTInterfacePartner and click **OK**. CompleteTheLoan will be selected as the **Operation**
7. Check the **Use Data Type Variables** checkbox.
8. Click the  button. The Select Variable for Task Information window opens.
9. Select **HumanTaskData** from the list and click **OK**.
10. Click **File** → **Save**.

Now that you have implemented the automatic approval path, you will do the same for the manual approval path.

Creating a path for manually approved loan requests


You need to create a path to handle when the applicant's credit score is good but automatic loan approval is not possible.

To create the path for manually approved requests, follow the tasks below:

Creating the otherwise case

You need to start a new case for the manually approved requests.



To create the otherwise case, follow these steps:

1. In the process editor, hover over the **AutoApprovalTest** activity.
2. From the pop-up, click , which is the **Add Otherwise** button.
3. Click **File** → **Save**.

Assigning an automatic response to the applicant

Assign a message to the Response variable that is sent automatically back to the applicant, indicating that their application needs to be manually approved.


To assign the response, follow these steps:

1. In the process editor, click , which is the **Invoke** icon and then click  which is the **Assign** icon.
2. Click the **Otherwise** label. The Assign activity is added.
3. Rename Assign to ManualApproval.
4. In the **Properties** view, click **Details**.
5. In the **From** list, select Fixed Value.
6. In the **Description** field, type Your application has been received and is under review..
7. In the **To** list, select Variable.
8. In the navigation tree, select **Response : string**. The **Query** field is empty.
9. Click **File** → **Save**.

Assigning a task to the bank employee

Assign an instruction to the HumanTaskData variable which is then sent to the bank employee.




To assign the instruction, follow these steps:

1. In the process editor, click , which is the **Assign** icon.
2. Click below the **ManualApproval** activity. The Assign activity is added.
3. Rename Assign to HumanApproval.
4. In the **Properties** view, click **Details**.
5. In the **From** list, select Fixed Value.
6. In the **Description** field, type This application requires manual approval..
7. In the **To** list, select Variable.
8. In the navigation tree, expand **HumanTaskData : HumanTaskBO** → **instruction : string**. The **Query** field is set to /instruction.
9. Click **File** → **Save**.

Invoking the human task

You will invoke the ProcessTheApplication human task component so that the bank employee can process the loan.

To create the activity that will invoke the human task, follow these steps:

1. In the process editor, click , which is the **Assign** icon and then click , which is the **Invoke** icon.
2. Click below the **HumanApproval** activity. The Invoke activity was added.
3. Rename Invoke to ProcessApplication.
4. In the **Properties** view, click **Details**.
5. Click **Browse**. The Select a Partner window opens.
6. In the list, select ProcessAppHTInterfacePartner and click **OK**. ProcessTheApplication is selected as the **Operation**.
7. Check the **Use Data Type Variables** checkbox.
8. Click the  label. The Select Variable for TaskInformation window opens.
9. Select **HumanTaskData** from the list and click **OK**.
10. Click **File** → **Save**.

Now that you have a path to process approved loan requests, you will add an additional path for declined requests.

Creating a path for rejected loan requests




Some loan applicants will have a bad credit score and their applications will need to follow a separate path in the business process. You will now create the path in the process to decline the application.

To complete the path in the process that handles declined applications, follow the tasks below.

Assigning an automatic response to the applicant

Assign a message to the Response variable that is sent automatically back to the applicant, indicating that their application was declined.


To assign the response, follow these steps:

1. In the process editor, hover over the **ScoreEvaluation** activity.
2. From the pop-up, click , which is the **Add Otherwise** button.
3. Click , which is the **Invoke** icon, and then click , which is the **Assign** icon.
4. Click the **Otherwise** label. The Assign activity is added.
5. Rename Assign to Declined.
6. In the **Properties** view, click **Details**.
7. In the **From** list, select Fixed Value.
8. In the **Description** field, type Thank you for your application. You will be contacted shortly to verify your information..
9. In the **To** list, select Variable.
10. In the navigation tree, select **Response : string**. The **Query** field is empty.
11. Click **File** → **Save**.

Assigning a task to the bank employee

Assign an instruction to the HumanTaskData variable which is then sent to the bank employee.




To assign the instruction, follow these steps:

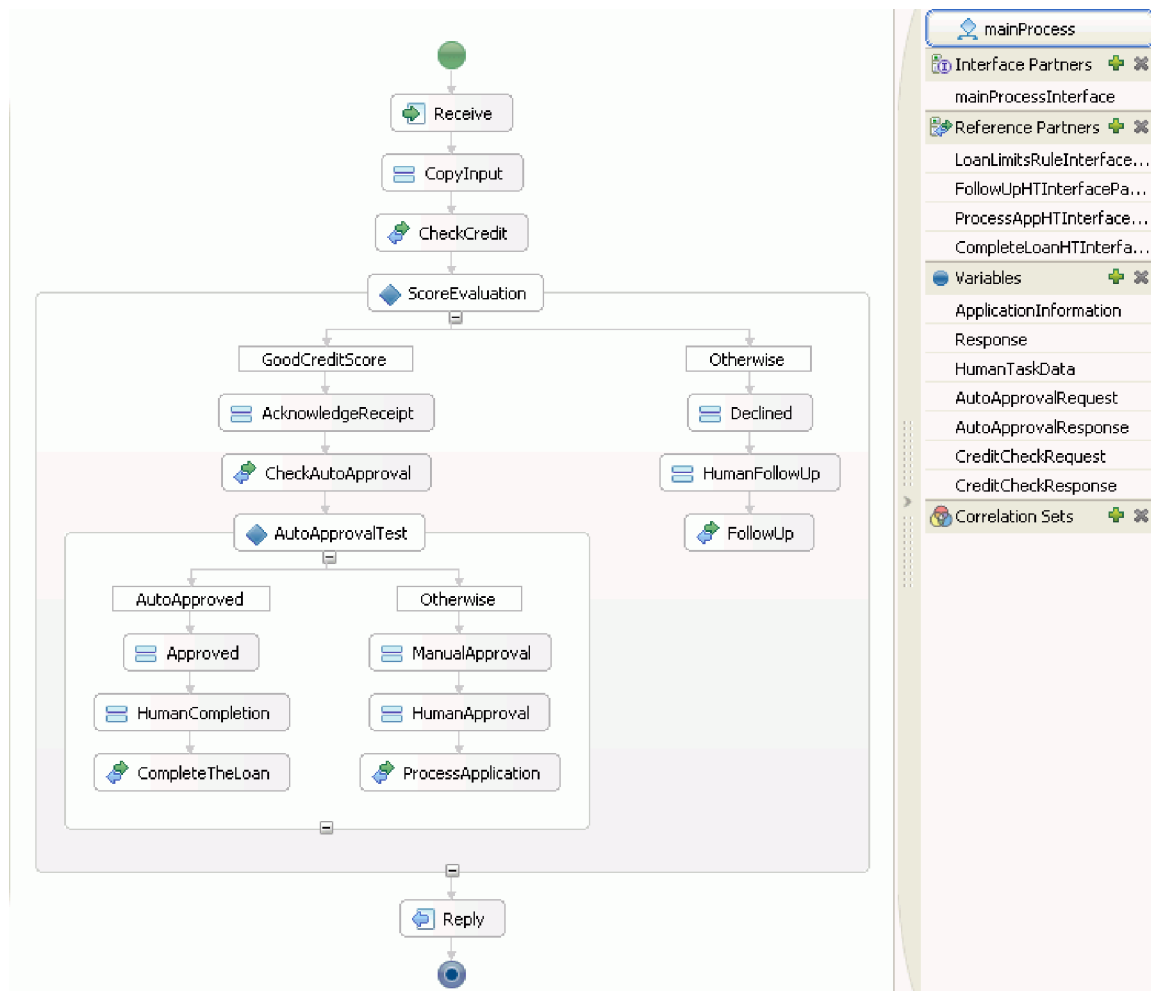
1. In the process editor, click  , which is the **Assign** icon.
2. Click below the **Declined** activity. The Assign activity is added.
3. Rename Assign to HumanFollowUp.
4. In the **Properties** view, click **Details**.
5. In the **From** list, select Fixed Value.
6. In the **Description** field, type This declined application requires follow up..
7. In the **To** list, select Variable.
8. In the navigation tree, expand **HumanTaskData : HumanTaskBO** → **instruction:string**. The **Query** field is set to /instruction.
9. Click **File** → **Save**.

Invoking the human task

You will invoke the FollowUpDeclinedApp human task component so that the bank employee can review the declined request.

To create the activity that will invoke the human task, follow these steps:

1. In the process editor, click  , which is the **Assign** icon, and then  , which is the **Invoke** icon .
2. Click below the **HumanFollowUp** activity. The Invoke activity is added.
3. Rename Invoke to FollowUp.
4. In the **Properties** view, click **Details**.
5. Click **Browse**. The Select a Partner window opens.
6. From the list, select FollowUpHTInterfacePartner and click **OK**. FollowUpDeclinedApp is selected as the **Operation**
7. Check the **Use Data Type Variables** checkbox.
8. Click the  label. The Select Variable for Task Information window opens.
9. Click **HumanTaskData** and click **OK**.
10. The process should now look like this:



11. Click **File** → **Save** and then close the process editor.

Now that you have completed the process implementation, you will need to fully implement the bank rules, credit checking, and the human tasks.

Chapter 8. Implementing business rules, a Java™ component, and human tasks

Implement the conditions to approve a loan and the needed interaction by people involved.

Creating and using business rules

Create a group of rules that the bank will use to enforce its policy for loan applications. You will use these rules to send a response to the business process after the `LoanLimits` rule group is invoked, indicating if an applicant's credit rating is good enough to have the loan approved, or if a loan amount qualifies for automatic approval.

For example, the bank will want to specify that all loan applicants must have a minimum credit rating of 750 to get a loan request approved. This is achieved by creating a rule set with an if-then rule that says "If the applicant has a credit rating below 750, return a response to the invoker indicating that the loan request should be rejected". In this case, the invoker is the `CreditCheck` activity in `mainProcess`, and the boolean response that it receives is used in the `ScoreEvaluation` activity to decide which path in the process to follow.

To implement the `LoanLimits` business group and its business rules, follow the tasks below.

Generating the rule group

The rule group you will create is a group of rule sets that accept input from operations in the `LoanLimitsRuleInterface`, and provide responses based on the rules you create.

To generate the `LoanLimits` rule group implementation, follow these steps:

1. Switch to the assembly editor.
2. Right-click the **LoanLimits** component.
3. From the pop-up menu, select **Generate Implementation**. The Generate Implementation window opens.
4. In the navigation tree, click **LoanApplicationModule** and click **OK**. The rule group editor opens for `LoanLimits`


Adding a rule set to check an applicant's credit rating





Create a set of rules for deciding if the applicant is eligible for loan from the bank.

You will create the following rules:

- Rule1: This default rule returns the response as false, indicating that the loan application has been declined.
- Rule2: This rule invokes the `checkCredit` operation to obtain a credit score for the applicant.
- Rule3: This rule takes the credit score stored in a variable from Rule2 and compares that against a value of 750. If the credit score of the applicant is 750 or higher, then the response is true and loan application will be approved.

To add the `checkCredit` rule set and the rules, follow these steps:

1. Click  , which is the icon beside the **checkCredit** label.
2. Click [Enter Destination](#) , which is the Enter Destination text figure.
3. Select **New Ruleset**. The New Rule Set window opens.
4. Click **Finish**. The business rules editor opens for `checkCredit`.

5. Click  , which is the **Add Variable** button. var1 is added.
6. In the **Select Type** field, select CreditB0.
7. Click  , which is the **Add Action Rule** button. Rule1 is added.
8. In the **Action** field, type response = false.
9. Click  , which is the **Add Action Rule** button. Rule2 is added.
10. Click the **Action** text figure.
11. Select **Invoke** from the list.
12. Click the **Select Partner Link** text figure and select CreditCheckInterfacePartner
13. Click the **Select Operation** text figure and select checkCredit.
14. Click the **Input** field. In the navigation tree, select **applicantInformation : ApplicantBO → taxPayerId : nillable:string**.
15. In the **Output** field, type var1.
16. Click  , which is the **Add If-Then Rule** button. Rule3 is added.
17. In the **If** field, type var1.creditScore >= 750.
18. In the **Then** field, type response = true.
19. Click **File** → **Save** and close the rule set editor.
20. Switch to the rule group editor.

Adding the autoapproval rule set


Create a set of rules to decide if the loan request will be automatically approved.

You will create the following rules:

- Rule1: This default rule returns the response as false, indicating that the loan application must be manually approved.
- Rule2: This if-then rule checks the loan amount against a value of 50 000. If the loan amount is less than this value, the response is true and the loan is automatically approved.

As you did for checkCredit, add the definition for the autoapproval rule set.

1. Give Rule1 the following values:

Rule Type	Action
Action Rule 	response = false

2. Give Rule2 the following values:

Rule Type	If	Then
If-Then Rule 	applicationInformation.loanamount <= 50000	response = true

3. Click **File** → **Save** and close the rule set editor.

Now that you have defined the business rules implementation, you will implement the CreditCheck component, which gets a credit rating for each applicant.

Implementing the CreditCheck Java™ Component

The CreditCheck component will have a Java implementation that generates a credit score using the taxPayerId of an applicant. Instead of editing the Java that is being generated and creating the implementation yourself, you will take a shortcut by importing the Java implementation.

To implement the CreditCheck Java™ component, follow these steps:

1. Switch to the assembly editor.
2. Right-click the **CreditCheck** component.
3. From the pop-up menu, select **Generate Implementation**. The Generate Implementation window opens.
4. Click **OK**. The Java editor opens.
5. Close CreditCheckImpl.java.
6. Select **File** → **Import**. The Import window opens.
7. Click **File System**.
8. Click **Next**.
9. In the **From directory** list, type <WSInstallDir>\wstools\eclipse\plugins\com.ibm.wbit.samples.content_6.0.2\scenario\parts\loanapplication.
10. Expand the folder structure of **loanapplication** and deselect the **web** folder.
11. Select the check box beside **CreditCheckImpl.java**.
12. In the **Into folder** field, type LoanApplicationModule\sca\component\java\impl. The **File system** field is set to Import resources from the local file system..
13. Select **Create selected folders only**.
14. Click **Finish**. If you are prompted to overwrite the previous file, click **OK** to confirm that you want to do so.
15. Click **File** → **Save**.

Now that the implementation for checking credit has been imported, you will implement the human tasks invoked at the end of the business process.

Implementing the human tasks

Human tasks are activities in the business process that people must perform, such as contacting a customer after an application is declined or when the loan amount is very large.

You will use human tasks in the loan application to confirm the approval or declination of each loan request that is entered into the business process.

These tasks show you how to implement human tasks in the assembly diagram:

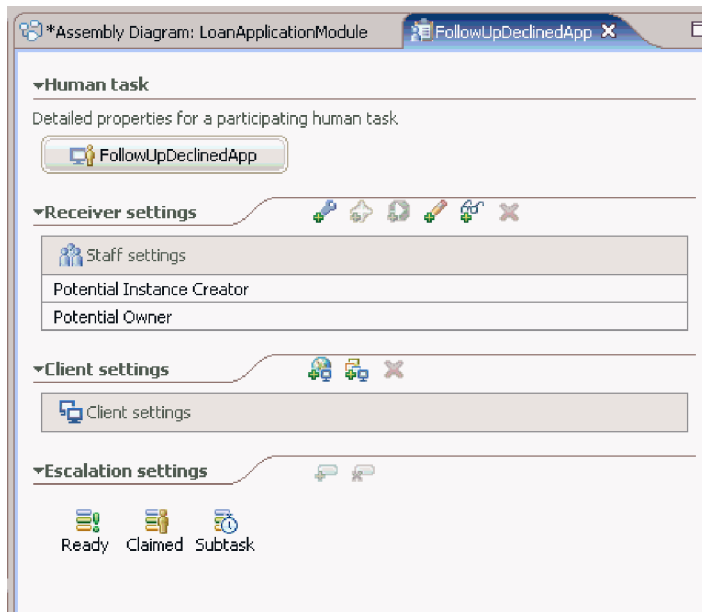
Implementing the FollowUpDeclinedApplication human task

This human task receives a HumanTaskBO that includes an application and instructions from the process when a loan request is declined. The bank employee that receives this task will then follow up with that particular loan.

To implement the FollowUpDeclinedApplication human task, follow these steps:

1. Switch to the assembly editor.
2. Right-click the **FollowUpDeclinedApp** component.
3. From the pop-up menu, select **Generate Implementation**. The Generate Implementation window opens.

4. In the navigation tree, click `LoanApplicationModule` and then **OK**. The Human Task Component Handler window opens.
5. Retain `FollowUpDeclinedApp` in the **Enter a name for the Human Task** field.
6. Click **OK**. The human task editor opens.





7. Close the human task editor and in the assembly editor click **File** → **Save**.

Implementing the `ProcessTheApplication` and `CompleteTheLoan` human tasks

Implement the remaining two human tasks.

1. Following the steps you used to create the `FollowUpDeclinedApplication` human task, define these human tasks:

Component	Human Task Name	Purpose
ProcessTheApplication 	ProcessTheApplication	This human task follows a loan request that must be manually approved by a bank employee.
CompleteTheLoan 	CompleteTheLoan	This human task follows the automatic approval of a loan request, and must be completed by a bank employee.

2. Switch to the assembly editor and click **File** → **Save**.

Now that you have implemented the human tasks, the loan application is ready to be run and tested.

Chapter 9. Testing the loan application

Test the loan application in the testing environment and from a Web page.

There are two ways to test the loan application:

- **Business Process Choreography Explorer:** This is a Web-based user interface for testing business processes. It allows you to start and execute your loan application by inserting your loan request and processing the approval.
- **Imported JavaServer Pages (JSPs):** This is the Web interface that a bank customer would see and interact with if they went to the bank's Website to apply for a loan. JSPs can be used to customize your own user interface for the application you just built.

Starting the application

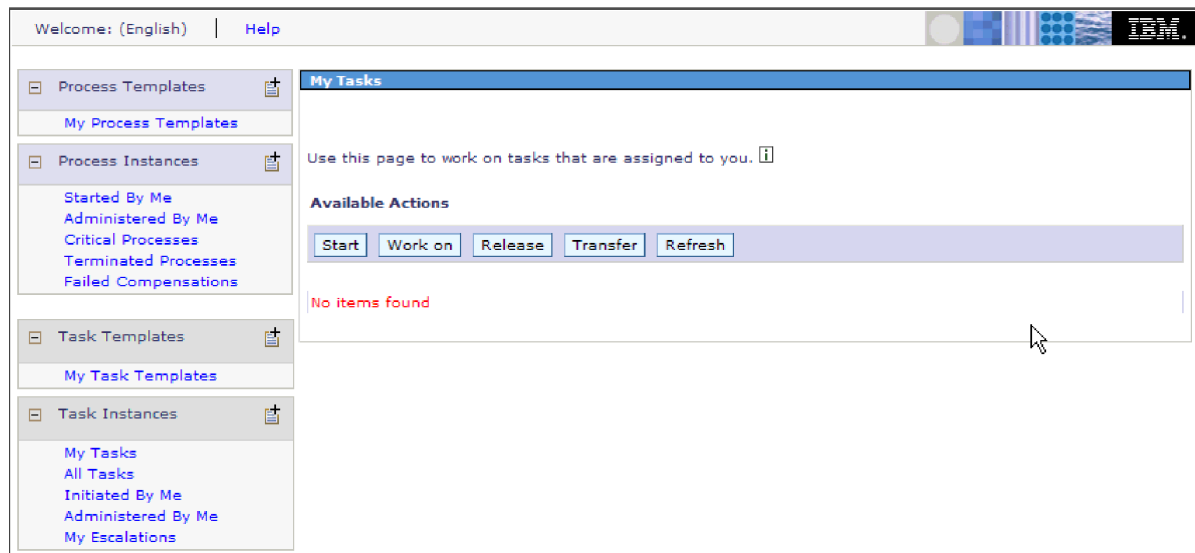
After you build the loan application, you will run it in the WebSphere® Process Server Integrated Test Environment.

Note: You can also test the application using JSPs. For more information about testing an application using JSPs, see the instructions in "Importing the EJBs" on page 9-5.

You will use the Business Process Choreographer (BPC) Explorer web client to run, test, and debug your application.

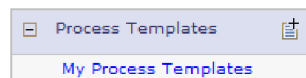
To run and test the loan application, follow these steps:

1. Add the application to the server.
 - a. In the Business Integration perspective, go to the **Servers** view.
 - b. Right-click **WebSphere Process Server**.
 - c. From the pop-up menu, select **Add and remove projects**. The Add and Remove Projects window opens.
 - d. In the navigation tree, click **LoanApplicationModuleApp**.
 - e. Click **Add** and then click **Finish**. It will take several minutes for the server to start and publish the loan application. Wait until "Application started: ..." is displayed in the console window.
2. Launch the Business Process Choreographer Explorer.
 - a. In the **Servers** view, right-click **WebSphere™ Process Server v6.0**.
 - b. From the pop-up menu, select **Launch → Business Process Choreographer Explorer**. The Business Process Choreographer Explorer opens and shows all tasks that are assigned to you. Currently there are no available tasks, so the "No items found" message is displayed.

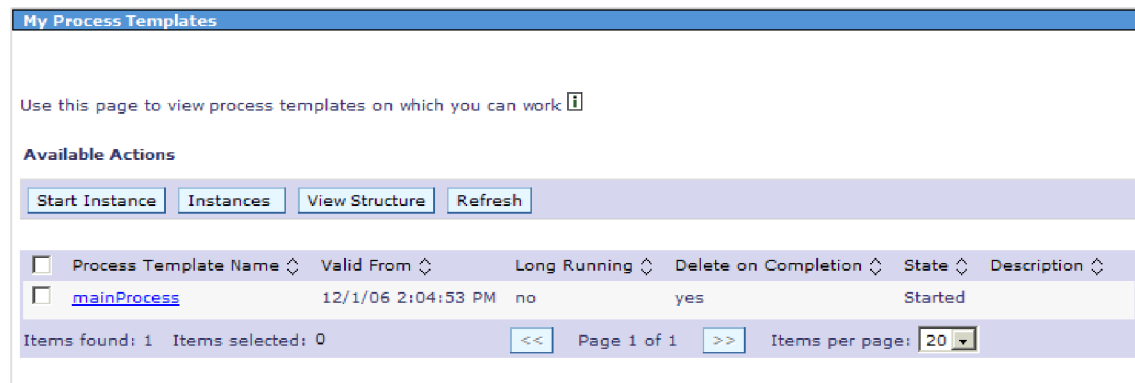


3. Invoke the application.

- a. To view a list of all process templates published to the server, click the **My Process Templates** label.



- b. To select the mainProcess template and start the loan application process, click the **mainProcess** label.



- c. To view all operations available to start the process, click the **Operations** tab. applyOnline is the only operation in the loan application that is available to start the process.
- d. Click the **applyOnline** label. A web-based front end opens where you can test your application.

Process Template

Use this page to view information about a process template [?](#)

Available Actions

[Instances](#) [View Structure](#)

Process Template Description

[Process Template Name](#) mainProcess
[Documentation](#)
[Description](#)

[Details](#) **Operations** [Process Instances](#) [Custom Properties](#) [Query Properties](#)

Operation Name	Activity Name	Description
applyOnline	Receive	

Items found: 1 << Page 1 of 1 >> Items per page: 20

- e. Click the + button beside applicant to display the input fields where you will insert your test values.
- f. In the **name** field, type John Smith.
- g. In the **email** field, type jsmith@jsmith.com.
- h. In the **taxPayerId** field, type 1234567.

Note: If you want a non-random value for the credit score to test the approval cases, add "888" to the end of taxPayerId.

- i. In the **loanamount** field, type 60000.

Process Input Message

Use this page to provide the input that is needed to start an instance of a business process. [?](#)

Available Actions

[Submit](#)

[Process Template Name](#) mainProcess
[Operation](#) applyOnline
[Process Input Message](#)

applicationInformation	applicant	
	name	<input type="text" value="John Smith"/>
	email	<input type="text" value="jsmith@smith.com"/>
	taxPayerId	<input type="text" value="1234567"/>
		<input type="text" value="-"/>
	loanamount	<input type="text" value="60000"/>

Before you submit the request, be aware that the following rules apply for the applyOnline operation:

Loan request result	Credit rating*	Loan amount
automatically approved	>= 750	<= \$50 000
manually approved	>= 750	> \$50 000

Loan request result	Credit rating*	Loan amount
declined	< 750	N/A

* The credit rating is determined randomly rather than being pulled from a database or other file.

- j. Click **Submit**. This loan application request is sent to the process for evaluation and a response is displayed, like in the image below:

Use this page to view the results of a business process that you started. [i]

Process Template Name: mainProcess
Operation: applyOnline
Process Input Message: applicationInformation

applicant	name	John Smith
	email	jsmith@smith.com
	taxPayerId	1234567
loanamount	60000.0	

Process Output Message: response

Thank you for your application. You will be contacted shortly to verify your information..

The message you receive concerning your application depends on your credit rating, and because of that it is determined randomly. Therefore, the message you receive will not always match the one displayed in the image above.

Response	Meaning	Human Task
"Thank you for your application. You will be contacted shortly to verify your information."	declined	FollowUpDeclinedApp
"Your application has been approved and is awaiting completion"	automatically approved	CompleteTheLoan
"Your application has been received and is under review."	manually approved	ProcessTheApplication

4. The loan request is submitted to the bank, and a bank employee will receive the data to follow up on the request. Now, you will switch into the role of the bank employee, who wants to work on the tasks sent to him or her.

- a. Click the **My Tasks** label. A list of tasks assigned to you is displayed.

Task Instances

- My Tasks
- All Tasks
- Initiated By Me
- Administered By Me
- My Escalations

- b. Click the **FollowUpDeclinedApp** label in case your request is declined (see table above), which is the task assigned to you by the bank customer when they submitted the loan request.

My Tasks

Use this page to work on tasks that are assigned to you. ⓘ

Available Actions

Start Work on Release Transfer Refresh

<input type="checkbox"/>	Task Name	State	Kind	Owner	Originator	Escalated	Suspended	Activated	Last Modified	Ex
<input type="checkbox"/>	FollowUpDeclinedApp	Ready	Participating		UNAUTHENTICATED	no	no		12/4/06 5:39:45 PM	12/4/06 5:39:45 PM

Items found: 1 Items selected: 0 << Page 1 of 1 >> Items per page: 20

- c. Click the **Work On** button.
- d. Click the **Complete** button to complete the application follow-up.

Task Message

Use this page to provide the data required to complete the task. ⓘ

Available Actions

Complete Save Release Cancel

Task Name
Task Input Message

FollowUpDeclinedApp

TaskInformation	instruction	This declined application requires follow up..	
application	applicant	name	John Smith
		email	jsmith@smith.com
		taxPayerId	1234567
	loanamount	60000.0	

Task Output Message No data available

Importing the EJBs

Enterprise JavaBeans (EJBs) are reusable software components that are useful for executing tasks. You will be using the EJBs to start the loan application through JavaServer Pages (JSPs).

To import the EJBs, follow these tasks:

Creating the web project

To test the application through a JSP interface, you need to create a web project.

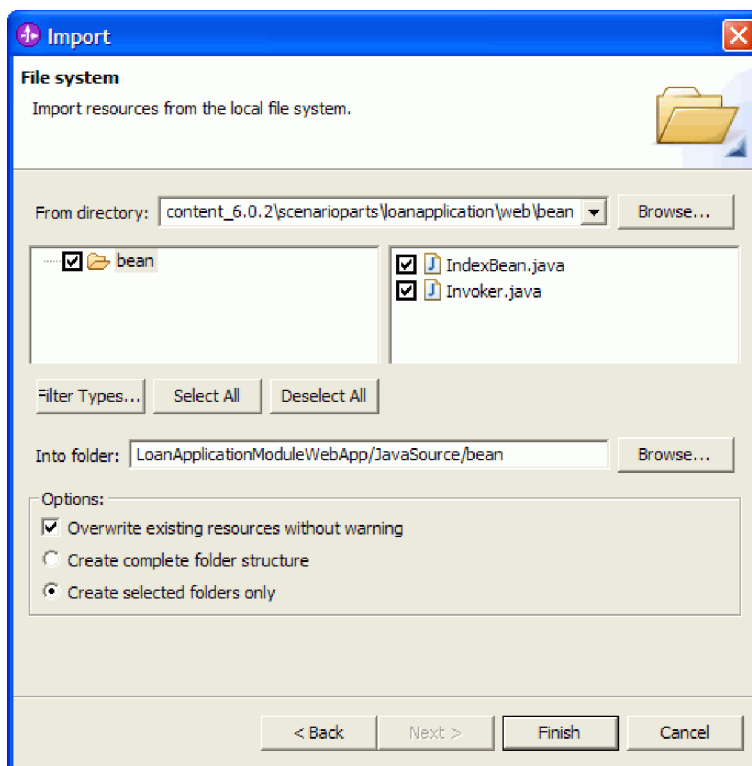
To create the web project, follow these steps:

1. Click **File** → **New** → **Other** → **Web** → **Dynamic Web Project**. If you don't see the **Web** folder, click the **Show all Wizards** checkbox. In the Confirm Enablement dialog, check the **Always enable capabilities and don't ask me again** checkbox and then click **OK**.
2. Click **Next** and in the **Name** field type `LoanApplicationModuleWebApp`.
3. Click **Show Advanced**.
4. In the **EAR project** field, select `LoanApplicationModuleApp` and click **Next**.
5. In the **Available dependant JARs** table, click the checkboxes beside `LoanApplicationModule.jar` and `LoanApplicationModuleEJB.jar`
6. Click **Finish**. In the Confirm Perspective Search dialog, click **No**.
7. Click **File** → **Save**.

Importing the EJBs

To import EJBs to test the application, follow these steps:

1. From the Business Integration view, right-click LoanApplicationModule and select **Import**. The Import window opens.
2. Select **File system** and then **Next**.
3. In the **From directory** list, type <WSInstallDir>\wstools\eclipse\plugins\com.ibm.wbit.samples.content_6.0.2\scenarioparts\loanapplication\web\bean where <WSInstallDir> is the product's installation directory.
4. In the navigation tree, select the **bean** folder and click the check box beside it to select both **IndexBean.java** and **Invoker.java**.
5. In the **Into folder** field, type LoanApplicationModuleWebApp\JavaSource\bean and select **Create selected folders only**.
6. Check the **Overwrite existing resources without warning** checkbox and click **Finish**.



7. Click **File** → **Save**.

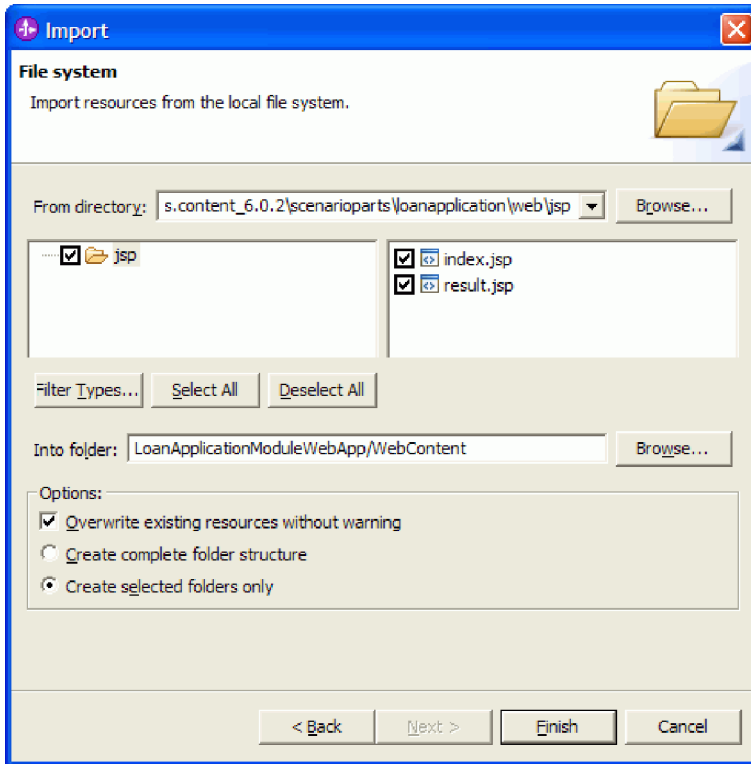
Importing the JSPs

You will use JavaServer Pages (JSPs) to invoke the EJBs and obtain user input to test the loan application.

JSPs act as extensions to HTML or XML tags by implementing dynamic content within static content, such as web pages. To import the JSPs to test the application, follow these steps:

1. In the menu, select **File** → **Import**. The Import window opens.
2. Click **File system** and then **Next**.
3. In the **From directory** list, type <WSInstallDir>\wstools\eclipse\plugins\com.ibm.wbit.samples.content_6.0.2\scenarioparts\loanapplication\web\jsp where <WSInstallDir> is the product's install directory.
4. In the navigation tree, select the check box beside **jsp** to select both **index.jsp** and **result.jsp**.

5. In the **Into folder** field, you will see LoanApplicationModuleWebApp\WebContent.
6. Select **Create selected folders only**.
7. Check the **Overwrite existing resources without warning** checkbox and click **Finish**.



8. Click **File** → **Save**.

Invoking the loan application with a JSP

Start the loan application on a server using the JSP as an interface.

Follow these instructions to invoke the JSP to test the application:

1. Switch to the **Web** perspective. Click the **Open a perspective** button in the workbench, click **Other**, then select **Web** from the list opened in the Select perspective window and click **OK**.
2. In the Project Explorer view, expand **Dynamic Web Projects** → **LoanApplicationModuleWebApp** → **WebContent** → **index.jsp**.
3. Right-click **index.jsp** and in the pop-up menu click **Run** → **1 Run on Server**. The Server Selection window opens.
4. Click **Choose an existing server**.
5. In the navigation tree, click **localhost** → **WebSphere Process Server v6.0**.
6. Click **Finish**. The server will be automatically started and the module will be added to it. You can follow these actions in the **Console** view. Wait until the progress indicator in the bottom right corner stops and the Web Browser opens with the Your Bank Loan Application window.
7. Click **File** → **Save**.

Testing the loan application using the JSPs

To test the loan application, you will interact with it through a Web browser using JSPs.

To test the loan application, you use the imported JSPs and EJBs because the JSPs provide an input interface for you to enter the loan application data. The CreditCheck and LoanLimits components direct the loan application through the process with their implementation. Depending on the user's credit score and the amount of the loan, the process will generate three possible responses. Explore the responses by entering the following data:

1. **Use Case 1:** Bank customer inserts any loan amount value, but their credit score happens to be below 750 (it is randomly chosen). This application will be declined.

Your Name: John Smith

Email: john@smith.com

Taxpayer Number: 12345

Loan Amount: 5000 (or any other valid amount)

Response: "Thank you for your application. You will be contacted shortly to verify your information."

Click **Back to the previous page** in the navigation bar of the Web Browser and then **Refresh the current page** to display the Your Bank Loan Application window with empty input fields. The result of this request is determined on a random basis due to the credit score evaluation.

2. **Use Case 2:** Bank customer inserts a loan amount value of \$50 000 or lower, and their credit score is 750 or higher. This application will be automatically approved.

Your Name: John Smith

Email: john@smith.com

Taxpayer Number: 888

Loan Amount: 1000

Response: "Your application has been approved and is awaiting completion."

Return to the previous page and refresh it as you did before to display the Your Bank Loan Application window with empty input fields. The **Taxpayer Number** 888 is not random, and will always result in an approval.

3. **Use Case 3:** Bank customer inserts a loan amount value greater than \$50 000, and their credit score is 750 or higher. This application will be manually approved.

Your Name: John Smith

Email: john@smith.com

Taxpayer Number: 888

Loan Amount: 65000

Response: "Your application has been received and is under review."

Return to the previous page and refresh it as you did before to display the Your Bank Loan Application window with empty input fields. The **Taxpayer Number** 888 is not random, and will always result in an approval.

Chapter 10. Summary

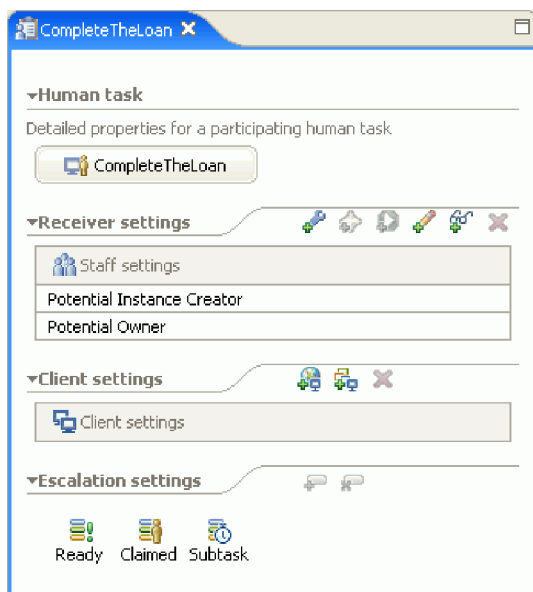
You have now completed the Loan Application sample and you can review what you have learned.

This sample showed you the top-down development of a business application, from module creation through to testing.

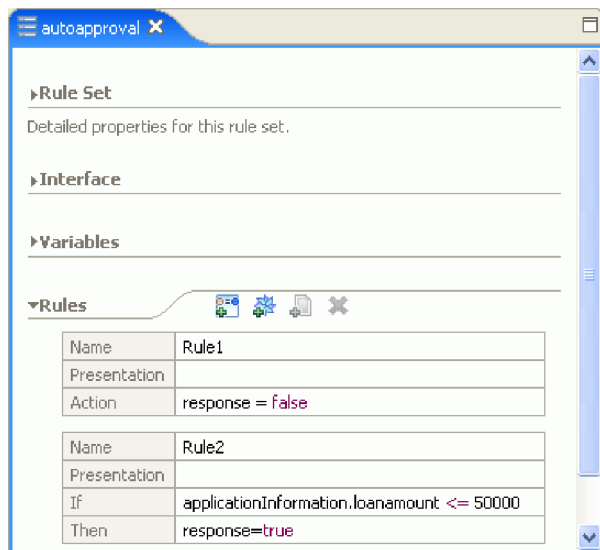
Lessons learned

By completing this sample, you learned to do following tasks:

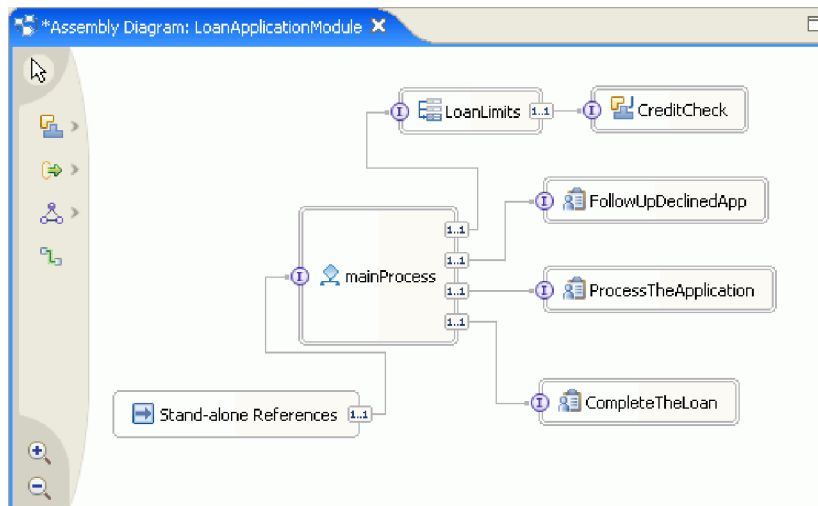
- Creating a module to contain the application
- Creating business objects with attributes and variables to store data
- Creating interfaces to define operations and adding them to components
- Generating implementations for the business process and for other components
- Running the application in the WebSphere Process Server V6 Integrated Test Environment
- Testing the application using EJBs and JSPs
- Creating human tasks with the human task editor:



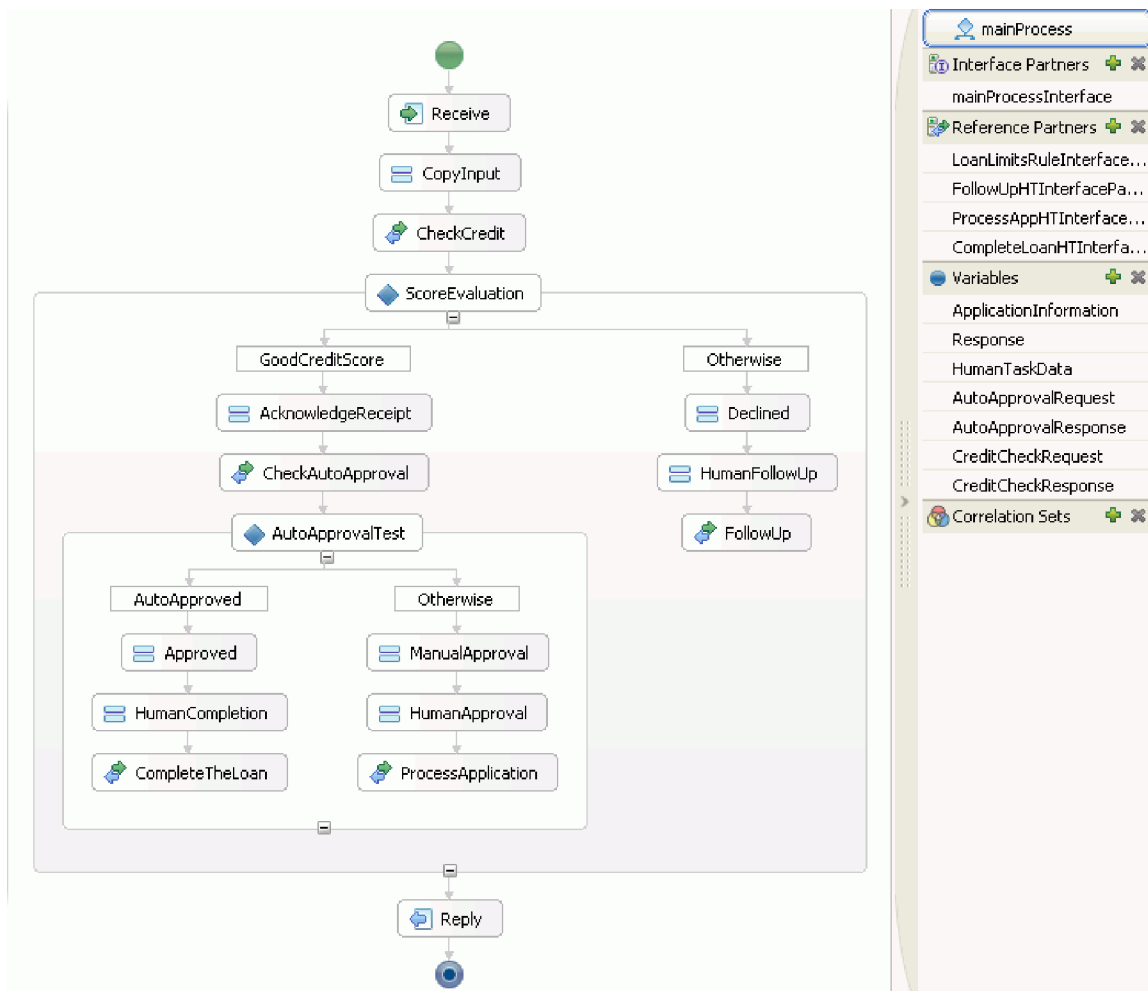
- Creating business rules with the business rules editor:



- Using the assembly diagram to create components, add stand-alone references, and wire them together:



- Using the process editor to organize a business process with separate tasks in sequence:



Notices

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

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this documentation in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM® product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this documentation. The furnishing of this documentation does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR CONDITIONS OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

*Intellectual Property Dept. for WebSphere Integration Developer
IBM Canada Ltd.
8200 Warden Avenue
Markham, Ontario L6G 1C7
Canada*

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this documentation and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples may include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2000, 2006. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Programming interface information

Programming interface information is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

See <http://www.ibm.com/legal/copytrade.shtml>.



Printed in USA