Informatica® ActiveVOS
9.2.4.6

3. Designer Sample Orchestration Project

Informatica ActiveVOS 3. Designer Sample Orchestration Project
9.2.4.6
March 2020

Publication Date: 2023-11-08

Table of Contents

CHAPTER 1

Starting the ActiveVOS Server

This chapter includes the following topic:

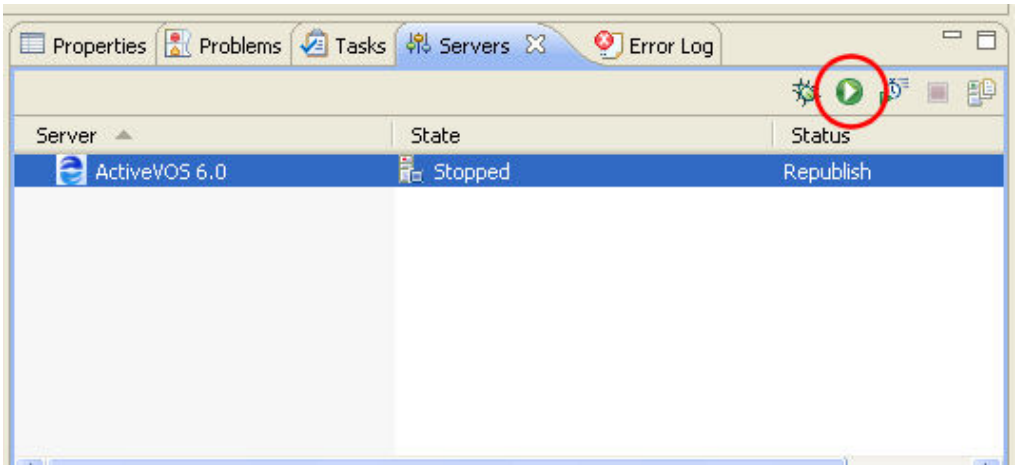
Starting the ActiveVOS Server

The Process Server consists of the Process Server engine running under Apache Tomcat 7.0.33. Tomcat is the servlet container that is used in the official reference implementation for the Java Servlet and JavaServer Pages technologies.

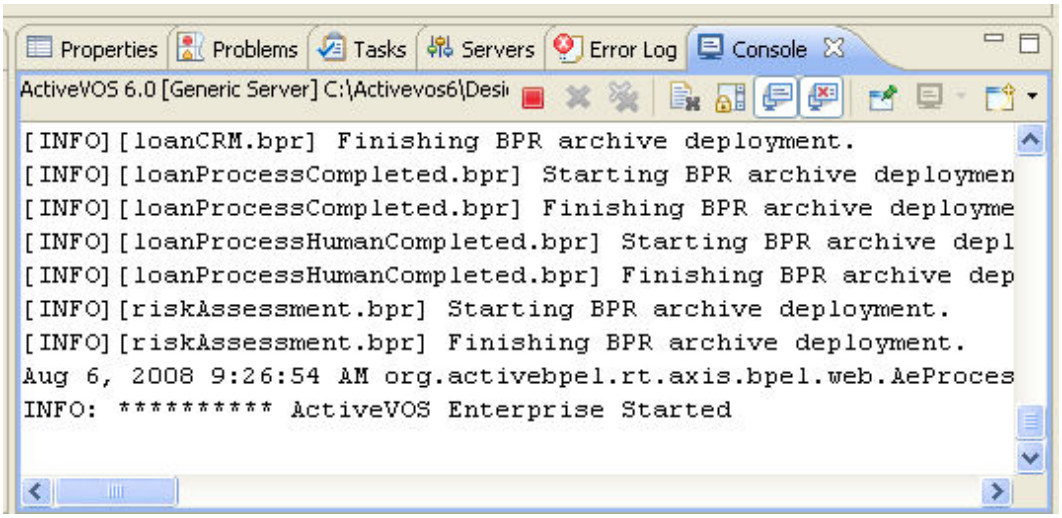
Starting the Process Server:

Note: If the Process Server already exists in the list of Servers, skip to step 4 to start it.

1. Select the Servers view in the lower right of the workspace.
2. Right-mouse click within the Servers view and select New > Server.
3. In the Server type list, select Process Server, and select Finish.
4. Select the Start the Server button, as shown in the example.



As the server starts up, you see start up tasks scroll in the Console. Several files are deployed to the embedded server each time you start it. Many of these files are for BPEL for People activities that you may want to create for your next project. When the server is started, the message indicates this, as shown.


Properties
Problems
Tasks
Servers
Error Log
Console
ActiveVOS 6.0 [Generic Server] C:\Activevos6\Desi
[INFO] [loanCRM.bpr] Finishing BPR archive deployment.
[INFO] [loanProcessCompleted.bpr] Starting BPR archive deploymen
[INFO] [loanProcessCompleted.bpr] Finishing BPR archive deployme
[INFO] [loanProcessHumanCompleted.bpr] Starting BPR archive depl
[INFO] [loanProcessHumanCompleted.bpr] Finishing BPR archive dep
[INFO] [riskAssessment.bpr] Starting BPR archive deployment.
[INFO] [riskAssessment.bpr] Finishing BPR archive deployment.
Aug 6, 2008 9:26:54 AM org.activebpel.rt.axis.bpel.web.AeProces
INFO: ********** ActiveVOS Enterprise Started

CHAPTER 2

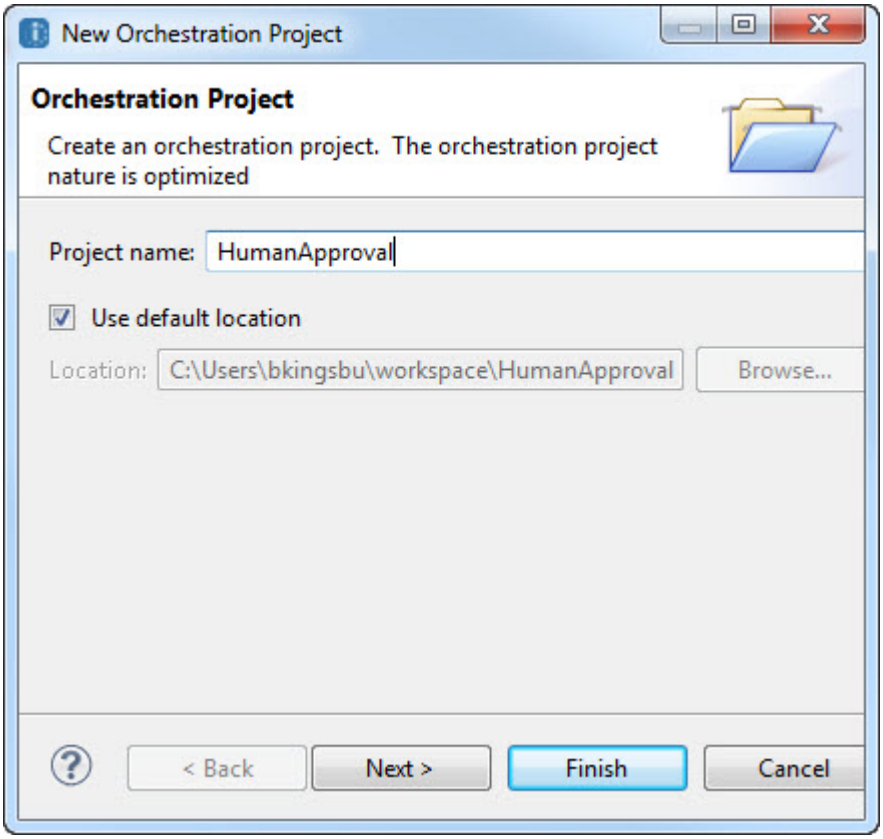
Human Approval Sample

This chapter includes the following topics:

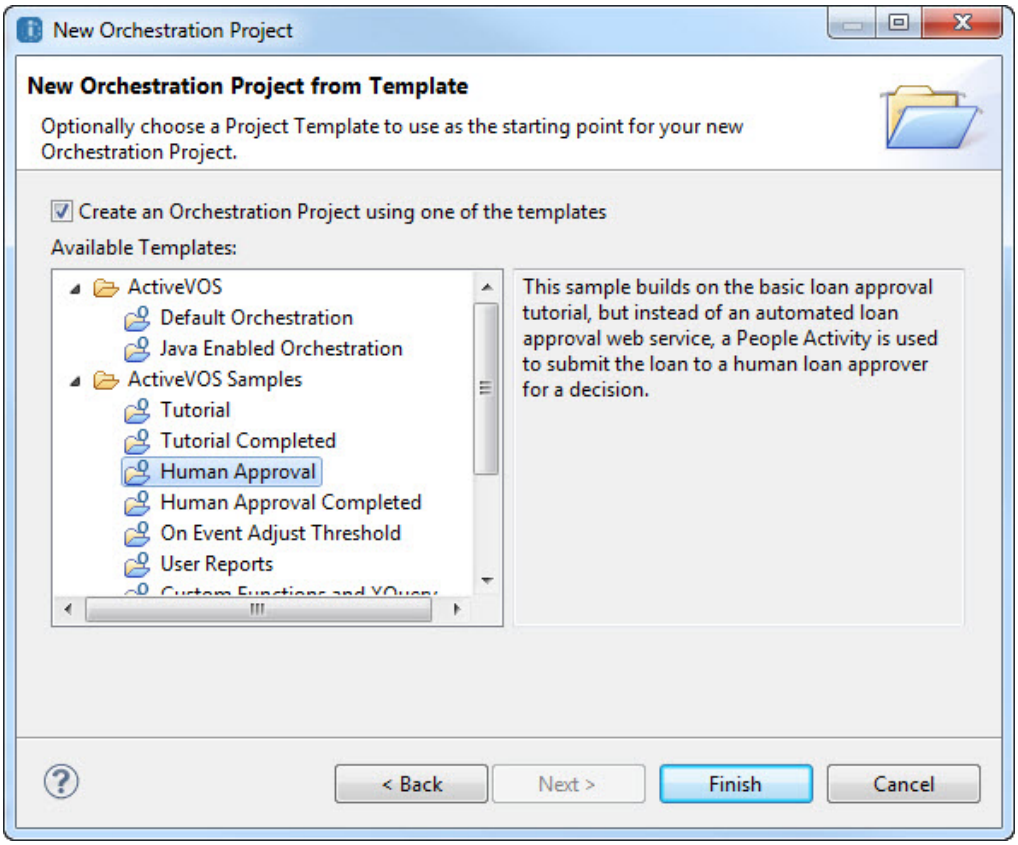
Getting Started

To start, you are going to create a new orchestration project using the Human Approval sample template.

1. In Process Developer, select **File > New > Orchestration Project** from the main menu.



2. Enter `HumanApproval` as the project name and click **Next**.
3. From the list of Available Templates, select Human Approval.



4. Click **Finish**.

Preparing the Process for Human Interaction

The process that is explained here is based on the Loan Approval process that is built when you complete the Process Developer Tutorial. During this part, you will:

- Replace the Web service that determines if a loan is approved or rejected with a people activity so that a person can make this decision.
- Update the logic to return an immediate acknowledgment to the loan requester if this path of the process is taken.
- Send a message to a customer service representative after the people activity completes.

Step 1. Remove the Invoke Loan Approver Activity

Since the decision to approve or reject a loan is now handled by a person, you need to update the loan approval process by removing the call to the loan approver Web service. This call will be replaced by a people activity.

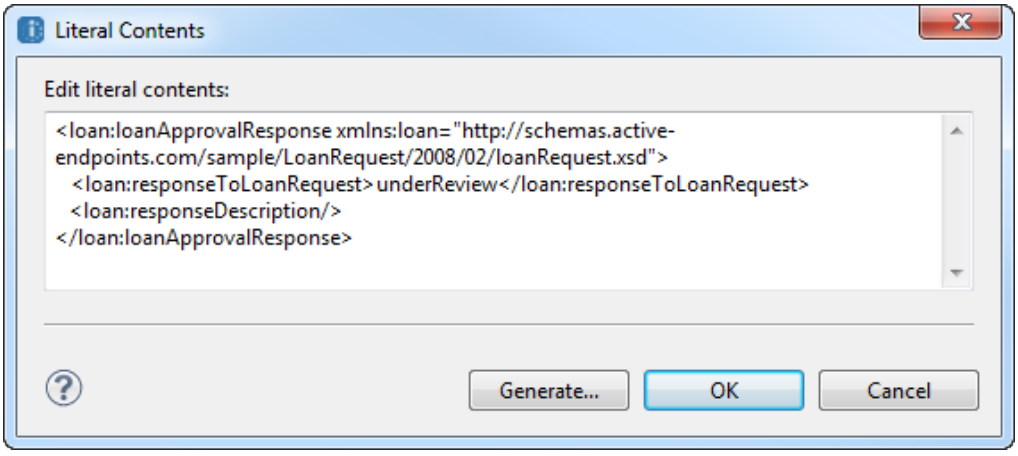
1. From the Project Explorer, expand the `bpel` folder and open the `loanProcessHuman.bpel` process.
2. Select the `Invoke Loan Approver` activity on the canvas, right-click, and select **Delete**.
3. In the Outline view, expand the Participant Partner Links section, right-click on the `LoanApproval` node, and delete it. This removes the LoanApproval participant from the Participants view.

Step 2. Update the Reply Sent to the Loan Requester

Before invoking the people activity, it makes sense to reply back to the loan requestor, informing the requestor that the loan will be reviewed. Do this by updating the reply activity that follows the Invoke Loan Approver activity just deleted.

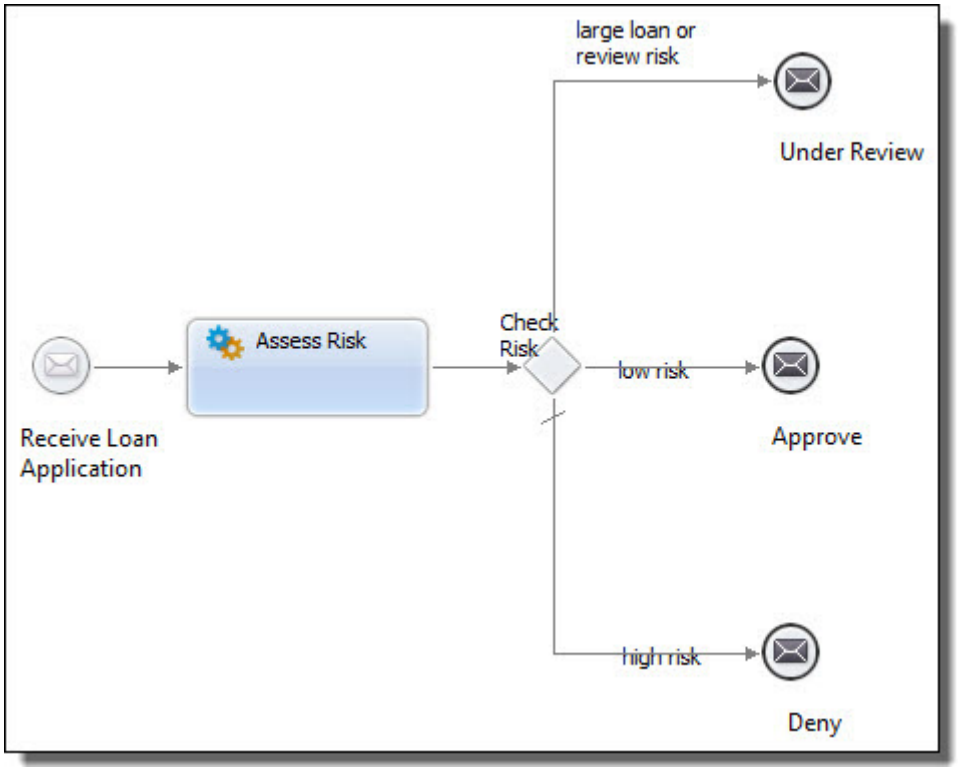
Next, format a message indicating that the loan is now under review. Do this by defining the data properties associated with the Reply activity. This implicitly builds an Assign statement that initializes the message variable.

1. Select the Return Approval Response reply activity and rename it to `Under Review`.
2. Select the `Under Review` activity.
3. In the Properties view for this activity, select the Data tab.
4. Change the Assignment from Single Variable to XPaths.
5. Press the **Add** button to create a new XPath copy expression.
6. Edit the XPath expression as follows:
   - In the first column headed E/L (for *Expression/Literal*), select Literal.
   - In the From column, select the ... button (the dialog button) to open the **Literal Contents** dialog.
   - Select **Generate**, and then **Finish** to accept the default values to create the `loan:loanApprovalResponse` element.
   - Change the values in the generated XML as follows. The result is shown in the following illustration.
     - Change the value in the `<loan:responseToLoanRequest>` element to `underReview`.
     - Remove the value of `string` in `<loan:responseDescription>`.
     - Because it is not needed, you can delete the optional `<loan:rejectionReason>...</loan:rejectionReason>` element.



7. Because you are allocating a literal value for the entire content of the `approval` variable, there's no need to specify a value for the To Path part of the assignment.
8. **Delete** the `approver` variable from the Variables section in the Outline view. It is no longer needed.

Your process should look like this:



Adding Human Interaction

After an acknowledgment is sent to the loan requester, add the People activity to the process.

Step 1. Add the LoanApproval Human Task Participant and Task Owners and Administrators

To create the human interaction for approving a loan, use WS-BPEL For People and WS-Human Task features.

A group called *loanreps* can claim and work on the tasks, while a group called loanmgrs would be administrators--they can escalate and reassign tasks. In BPEL, these are called a *Logical People Group*. They are added in the Participants view under Human Task Participants.

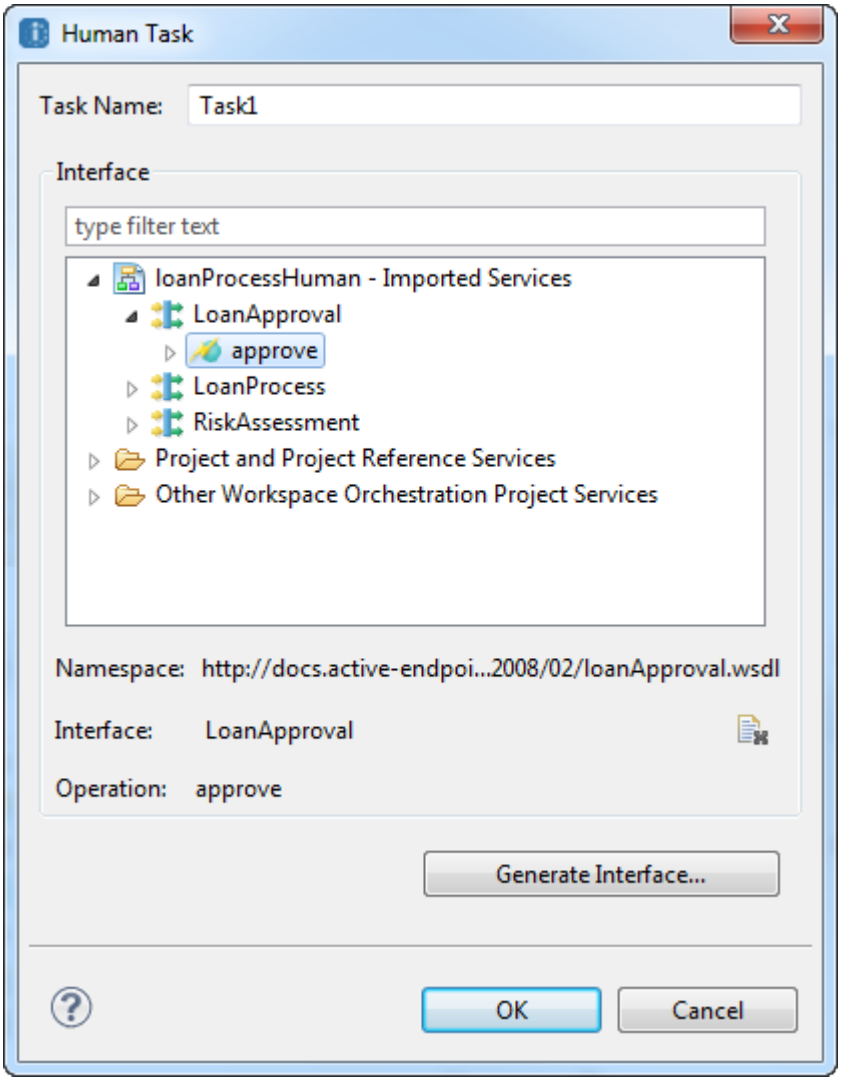
Logical People Group definitions must be defined for both loanreps and loanmgrs, and then they aree associated with the proper role for the task.

1. In the Participants view, right click Human Task Participants, and select **New Human Task Participant**. You will need to add a new participant role named `Role1` to the Participants view.
2. Select *Role1*. Edit the value for Name from *Role1* to `loanreps`. Later, when defining a deployment descriptor for the process, you will associate this Logical People Group name with an actual set of users that the configured Identity Service will resolve at runtime.
3. Repeat Steps 1 through 2 to add a second Logical People Group, this time naming it `loanmgrs`.

Step 2. Create the loan approval Human Task

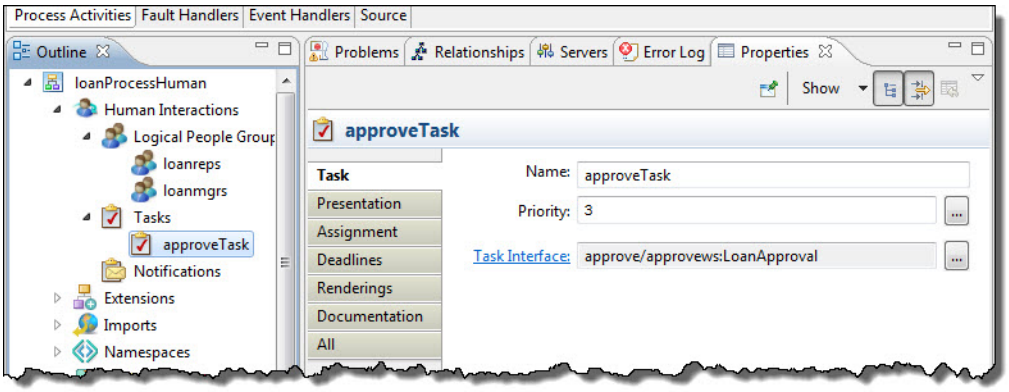
Now that you've declared the Logical People Groups, you can define its tasks. Start with WSDL defining the interactions. You can reuse the LoanApproval WSDL that was removed from the process earlier, but this time it will provide definitions for interaction with a person instead of a service.

1. In the Participants view, right click the *loanreps* Logical People Group, and select **New Human Task...**. The **Human Task** dialog opens, showing available port types. Remember that in WS-BPEL For People, people activities are just another type of service interaction, and they are declared in WSDL.
2. Expand the *loanProcessHuman - Imported Services* then the *LoanApproval* to display the *approve* operation, select it and press OK.

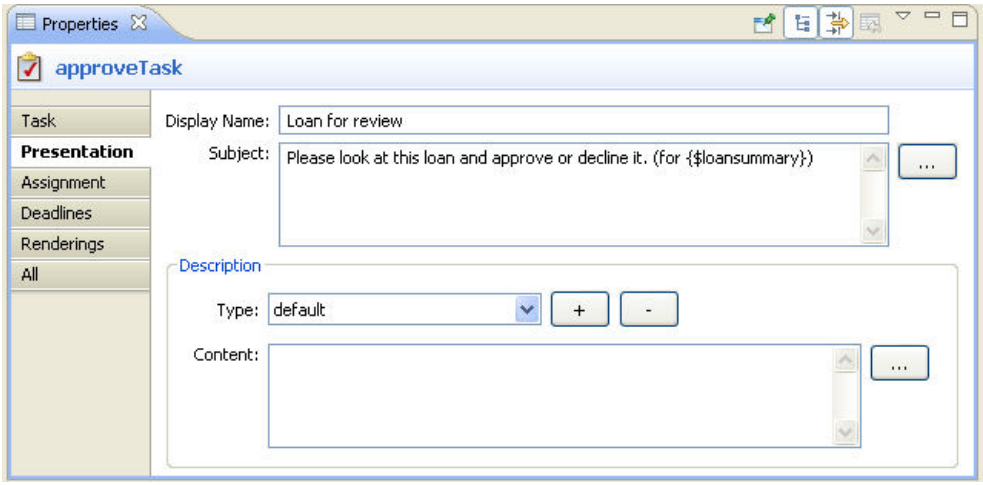


A new task will appear in the Participants view, under *loanreps*, called *Task1*.

3. Select `Task1` so that is displayed in the *Properties* view. Edit the value of the **Name** property to `approveTask`.
4. Set the Priority to 3. Your task properties should look similar to the following:



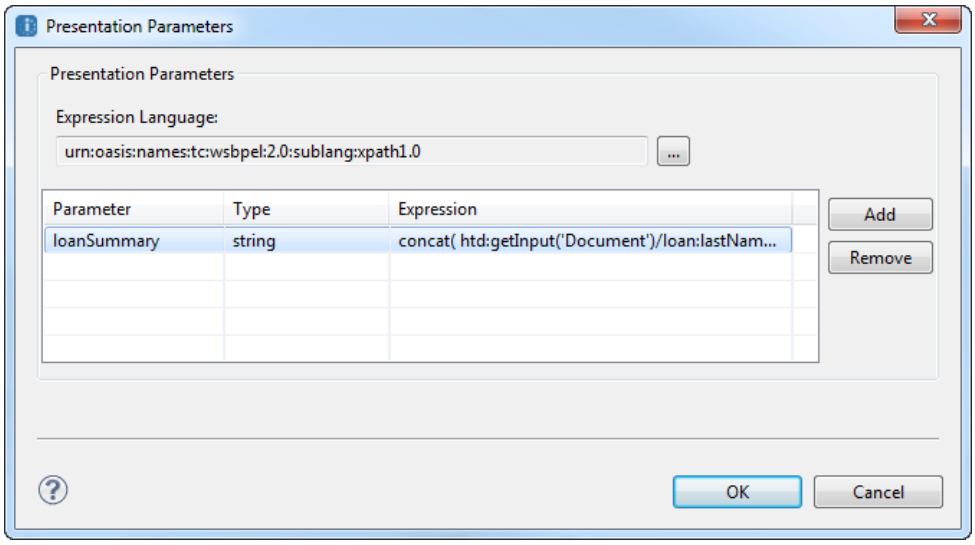
5. Select the Presentation side tab in the Properties view and set the following properties:
   - *Display Name*: `Loan for review`
   - *Subject*: `Please look at this loan and approve or decline it. (for {$loansummary})`
   - *Type*: `default`



6. Still at the Presentation tab, define the `loansummary` substitution variable that was just entered in the Subject area:
   a. Press the ellipsis button at the right of the panel.
   b. In the **Presentation Parameters** dialog that appears, press Add to add a new parameter. A new parameter called `param1` whose type is `string` will appear.
   c. Click on the *Parameter* name to make it editable, and enter `loansummary`.
   d. Leave the Type as `string`.
   e. Click on the expression to make it editable, and enter the following value:

```
concat(htd:getInput('Document')/loan:lastName , ", ",
       htd:getInput('Document')/loan:firstName , "; $",
       htd:getInput('Document')/loan:amountRequested )
```

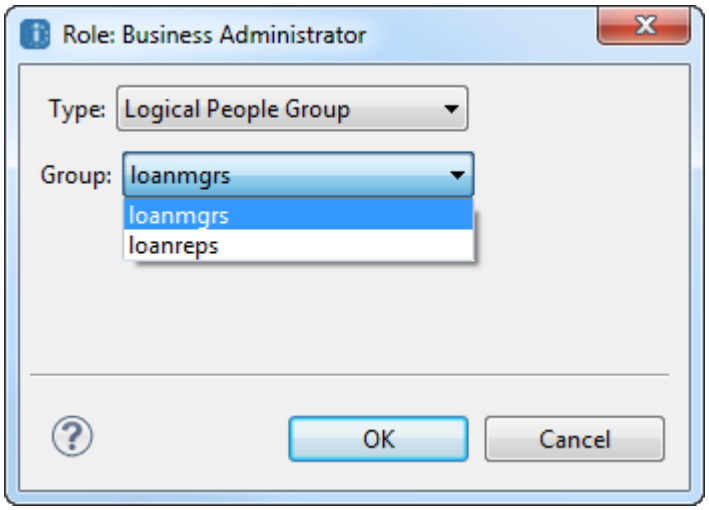
   Your dialog should look similar to this:



Step 3. Complete Binding the Task Owners and Administrators to the Task

The Potential Owners Role was automatically set to *loanreps* when you created the task. You now need to assign other users to some of the other task roles. Do this by assigning the `loanmgrs` Logical People Group to the Business Administrators role, and by assigning the `Loan Process` literal to the Task Initiator role.

1. Select the *Assignment* tab and set the value for *Business Administrator* to `loanmgrs` as follows:
   a. Click in the Value column for Role Name Business Administrator. An ellipsis button appears. Click it to display the Role dialog.
   b. In the **Role** dialog, select *Logical People Group* in the Type picklist.
   c. In the Group picklist, select *loanmgrs*, and press **OK**.

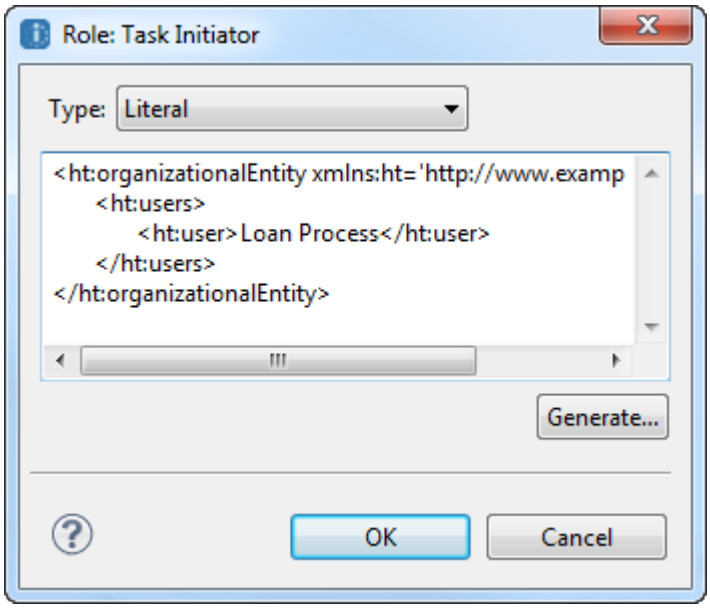


2. Still at the Assignment tab, set a value for *Task initiator* as follows:
   a. Click in the Value column for Role Name Task Initiator. An ellipsis button appears. Click it to display the Role dialog.
   b. In the **Role** dialog, select *Literal* in the Type picklist.

c. Select **Generate** to generate XML for a user.

d. Edit the value of the `ht:user` element to say `Loan Process`.

e. Your dialog should look like this:



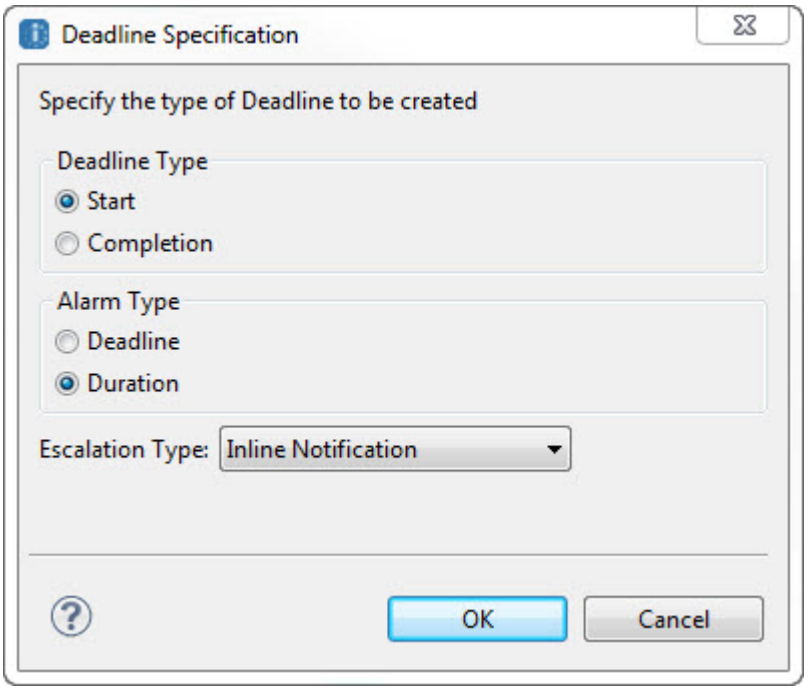
Step 4. Setting Task Deadlines

You can specify one or more deadlines by which a task must be started and also one or more deadlines by which a task must be completed. If a start or completion date is missed, you can trigger one or more escalation actions. For example, if you set a start the task deadline for two hours after a task is created and the deadline is missed, you can send a notification to the task owner. Here are the deadlines you will set:

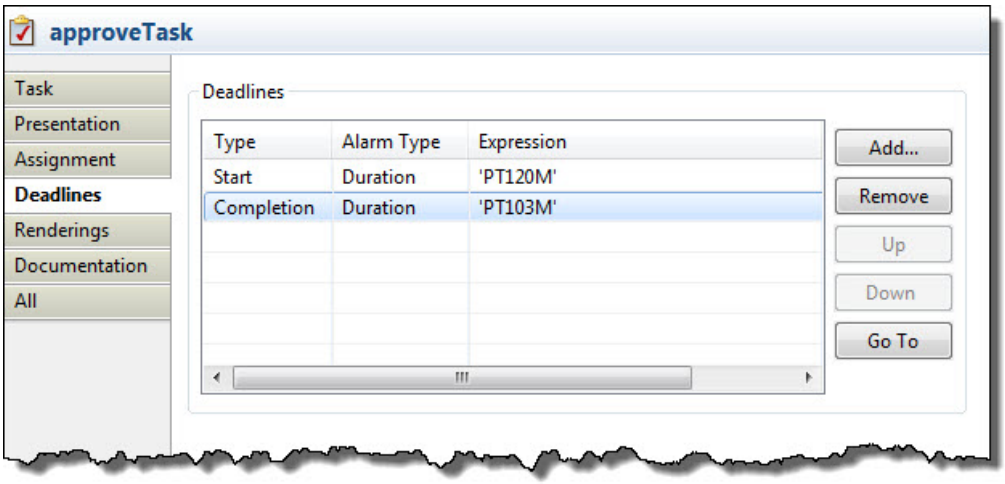
- A start deadline that is triggered if the task is not claimed and started by a loanrep within 2 hours.
- A completion deadline that is triggered if the task s not completed by a loanrep within 130 minutes.

Step 4a. Setting the Start Deadline

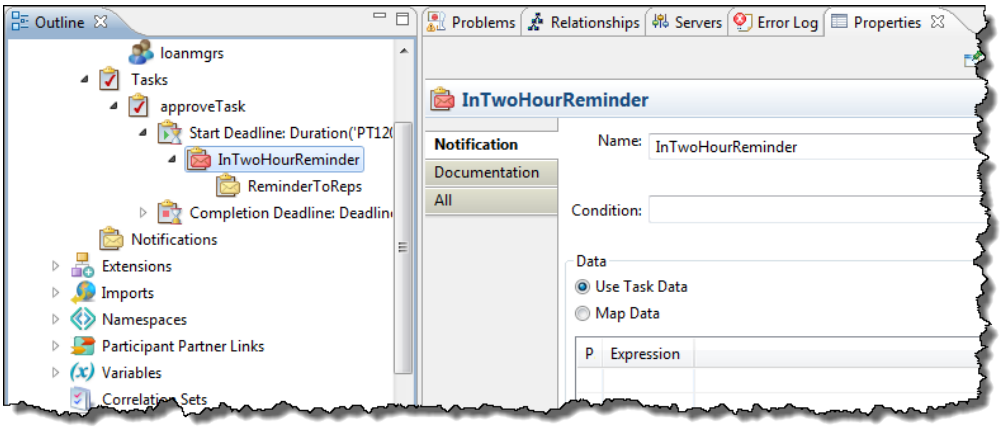
1. From either Participants or Outline view, select the approveTask node and select the Deadlines side tab in the Properties view.
2. Click the **Add** button and set the properties for *Deadline Type* to *Start*, the *Alarm Type* to *Duration* and the *Escalation Type* to *Inline Notification*.



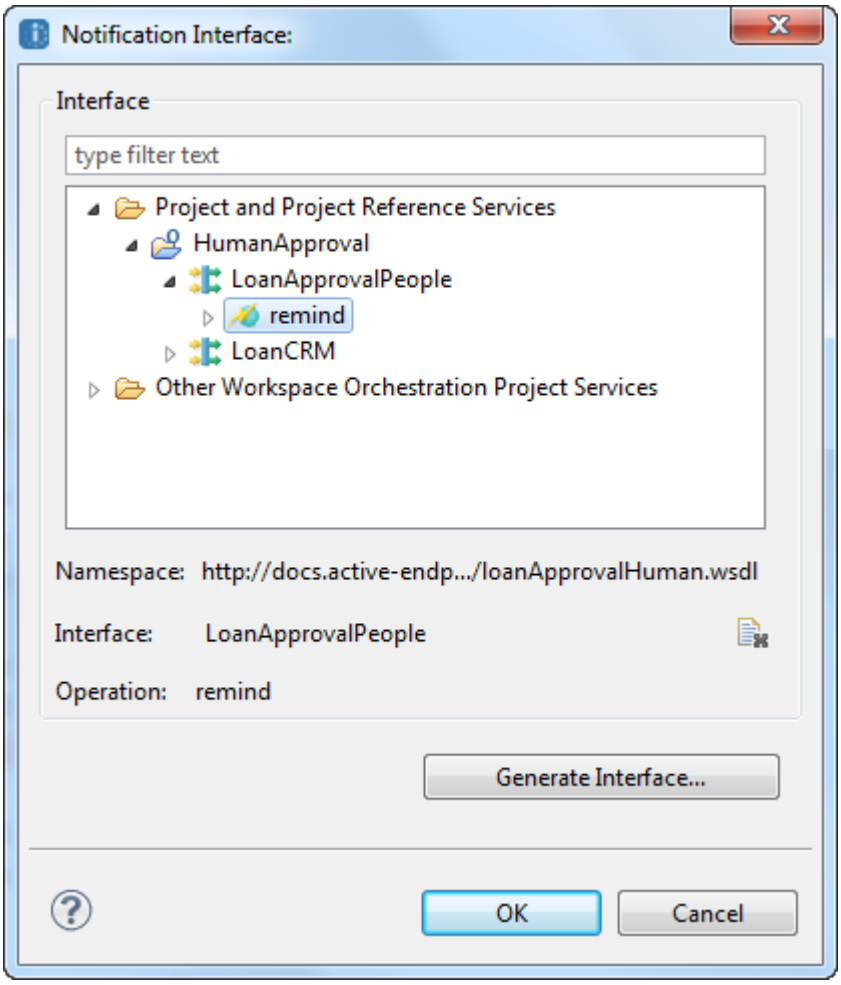
3. Press **OK**.
4. Click in the blank value under the *Expression* column and press the ellipses button (...) and set the duration expression to the following: `'PT120M'`. Include the single quotes in what you type.
5. Repeat steps 1-4 and create a completion deadline with a Duration Alarm Type for 10 minutes later (`'PT130M'`).
6. The two task deadlines appear in the *Properties* view:



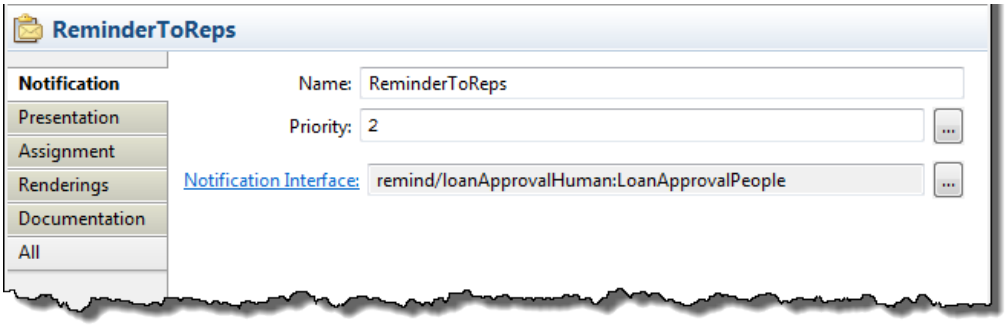
7. In the Outline view, select the Start Deadline and select the Escalations side tab in the Properties view.
8. Select the Inline Notification labeled InlineNofication1 and press the **Go To** button.
9. In the Properties view, select the Notification tab and change the name to `TwoHourReminder`.



10. In the Outline view, expand the `TwoHourReminder` node and select the child node labeled `N1`. This is the where you define the escalation action that is triggered if the deadline is reached.
11. In the Properties view, select the Notification tab and set the following properties:
    - *Name*: `ReminderToReps`
    - *Priority*: `2`
12. Press the ellipsis button to the right of the Notification Interface field.
13. In the **Notification Interface** dialog that appears, navigate to the LoanApprovalPeople port type, and select the remind operation.

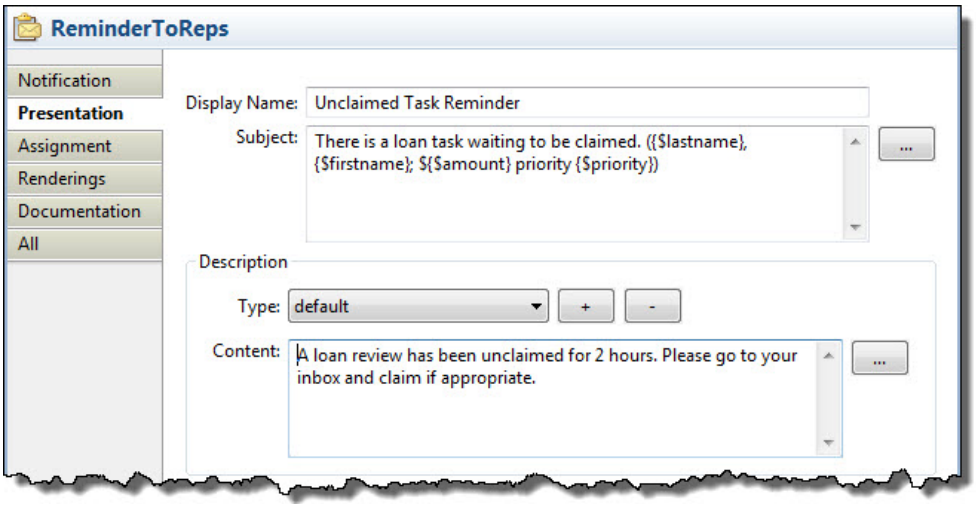


The remind operation is a one-way operation that notifies loanreps that a loan application has waited two hours for review. The Notification tab should look similar to the following:



14. In the Properties view, select the Presentation tab and set the following properties:
    - *Display Name*: `Unclaimed Task reminder`
    - *Subject*: `There is a loan task waiting to be claimed. ({$lastname}, {$firstname}; ${$amount} priority {$priority})`
    - *Description Type*: `default`

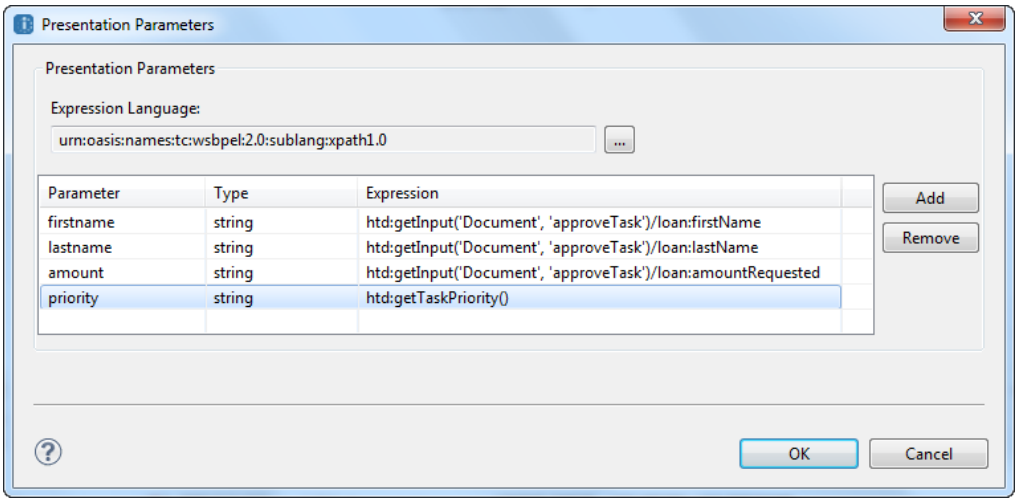
- *Description Content*: `A loan review has been unclaimed for 2 hours. Please go to your inbox and claim if appropriate.`



15. In the Properties view, select the Presentation tab, press the ellipsis to the right of the *Subject* field, and set the following presentation parameters:

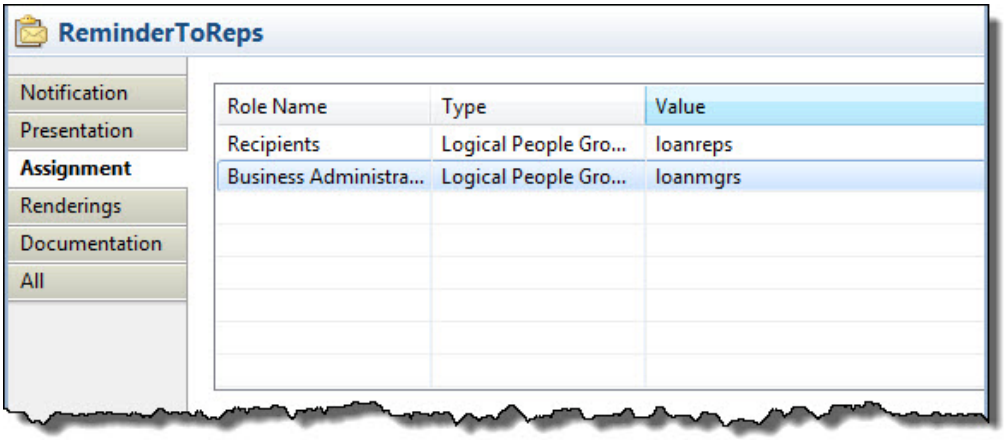
Parameter	Type	Expression
`firstname`	`string`	`htd:getInput('Document', 'approveTask')/loan:firstName`
`lastname`	`string`	`htd:getInput('Document', 'approveTask')/loan:lastName`
`amount`	`string`	`htd:getInput('Document', 'approveTask')/loan:amountRequested`
`priority`	`string`	`htd:getTaskPriority()`

Your completed dialog should look similar to this:



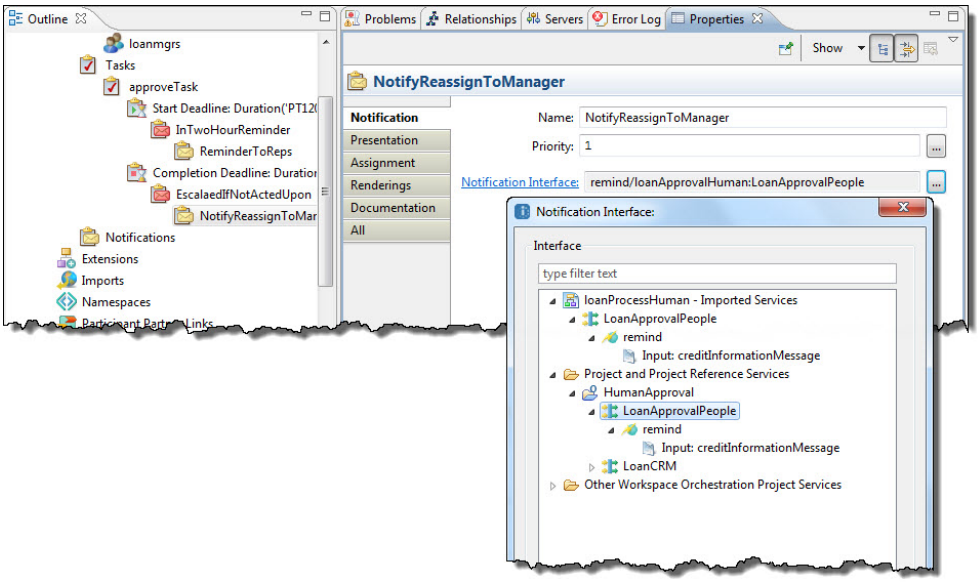
16. Press **OK**.

17. In the Properties view, select the Assignment tab, click in the Value field, and set the Recipients to loanreps and Business Administrator to loanmgrs.



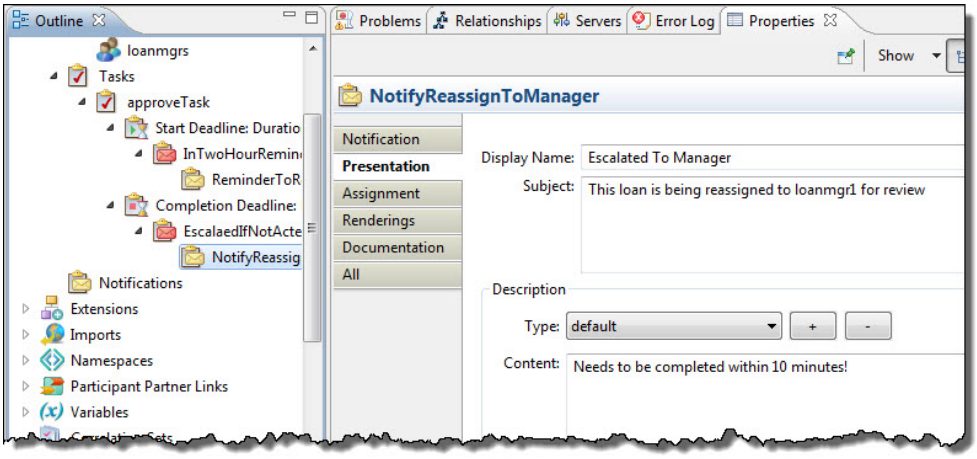
Step 4b. Setting the Completion Deadline

1. In the Outline view select the Completion Deadline and press the Escalation tab in the Properties view.
2. Select the *Inline Notification* labeled InlineNofication1 and press the **Go To** button.
3. In the Properties view, select the Notification tab and change the name to `EscalateIfNotActedUpon`.
4. In the Outline view, expand the `EscalateIfNotActedUpon` node and select the child node labeled `N1`.
5. In the Properties view select the Notification tab and set the following properties:
   - *Name*: `NotifyReassignToManager`
   - *Priority*: `1`
   - *Notification Interface*: `remind/loanhuman:LoanApprovalPeople`

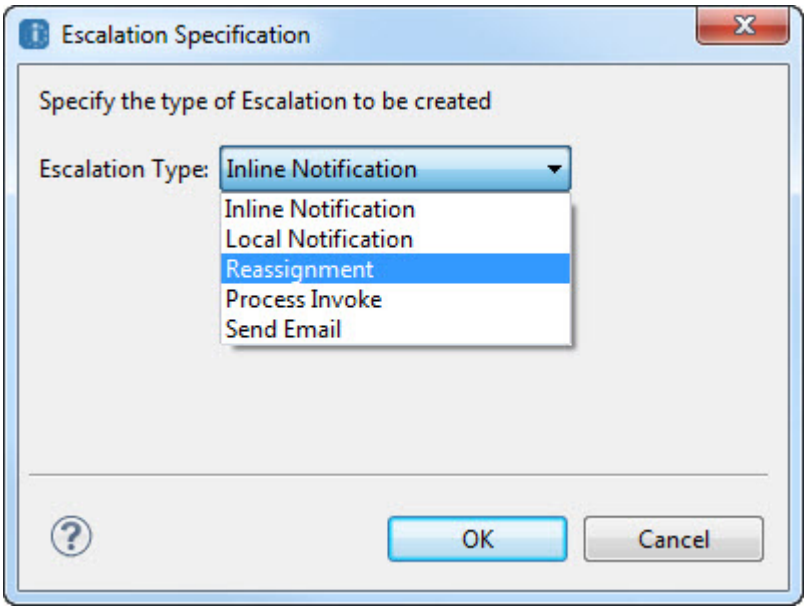


6. In the Properties view select the Presentation tab and set the following properties:
   - *Display Name*: `Escalated To Manager`
   - *Subject*: `This loan is being reassigned to loanmgr1 for review`

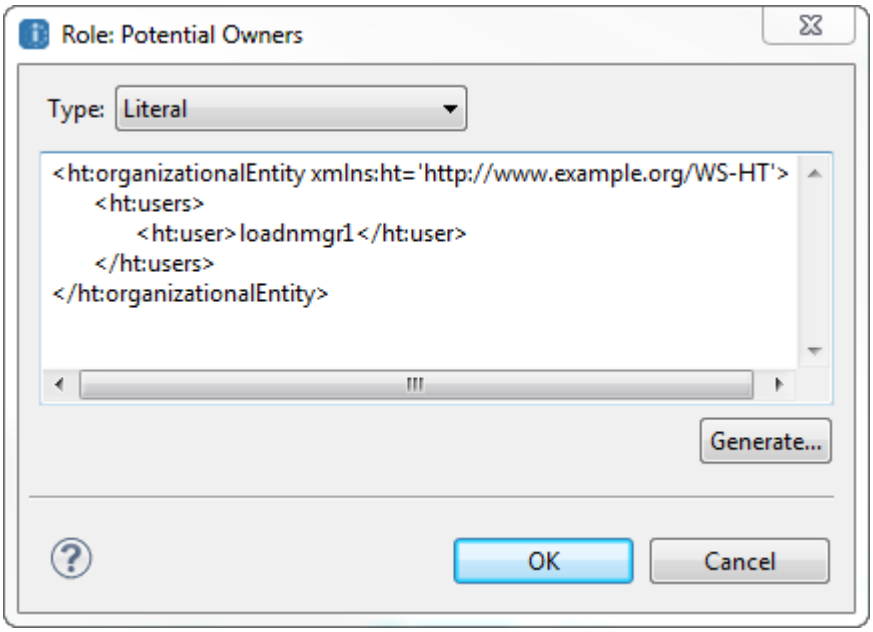
- *Description Type*: `default`
- *Description Content*: `Needs to be completed within 10 minutes!`



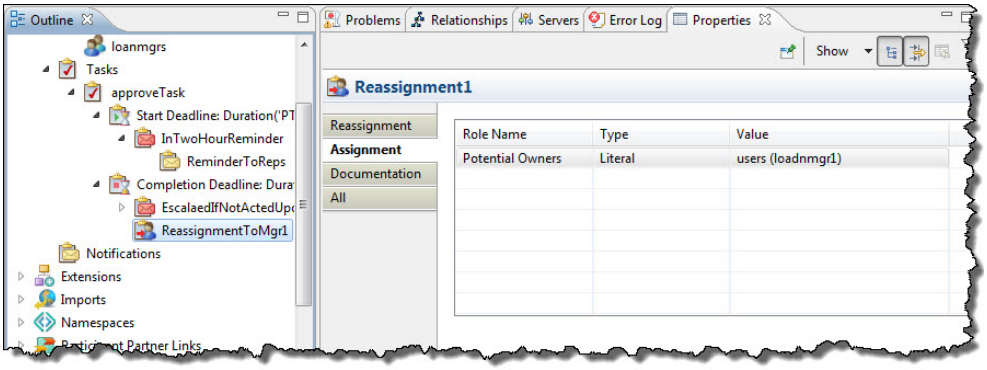
7. In the Properties view select the Assignment tab and set the Recipients to loanreps and Business Administrator to loanmgrs, as you did in a previous step for Reminder to Reps.
8. In the Outline view select the Completion Deadline and press the Escalation side tab in the Properties view.
9. Press the **Add** button and specify Reassignment for the *Escalation Type*:



10. Select the Reassignment labeled*Reassignment1* and press the **Go To** button.
11. In the Properties view select the Reassignment side tab and change the name to `ReassignToMgr1`.
12. In the Properties view select the *Assignment*side tab and set the properties by pressing the ellipses (...) button in the *Value* column:
    - Select Literal for the Type.
    - Press the **Generate...** button
    - Change the value in `<ht:user>` from `Some User` to `loanmgr1`



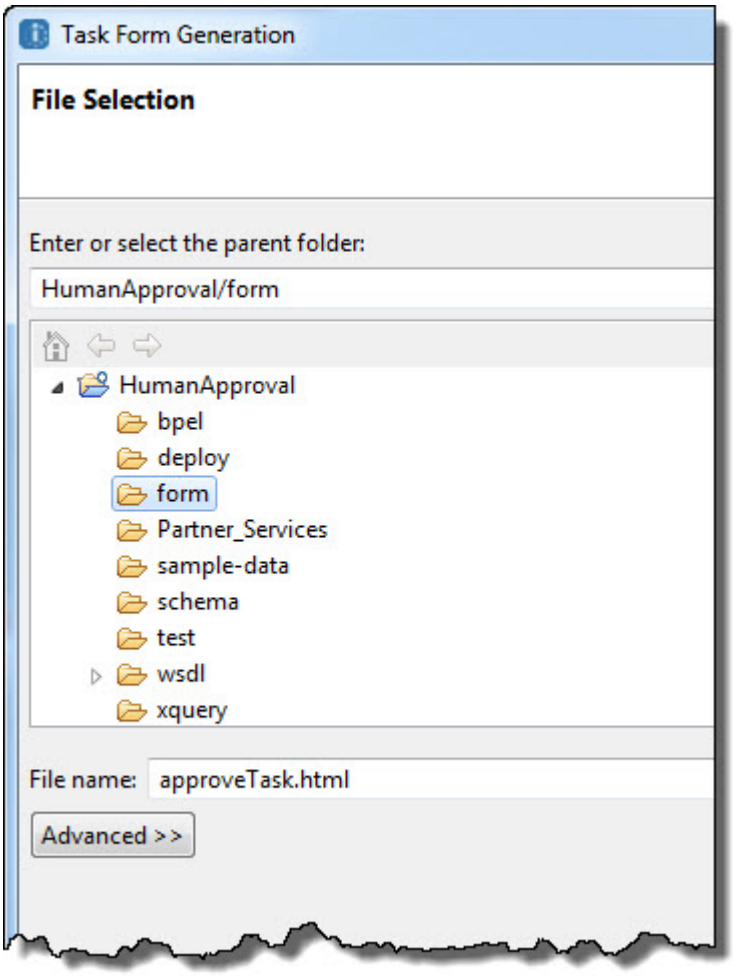
- Press **OK**.
- Your Properties should look like the following:



Step 5. Setting Task Renderings

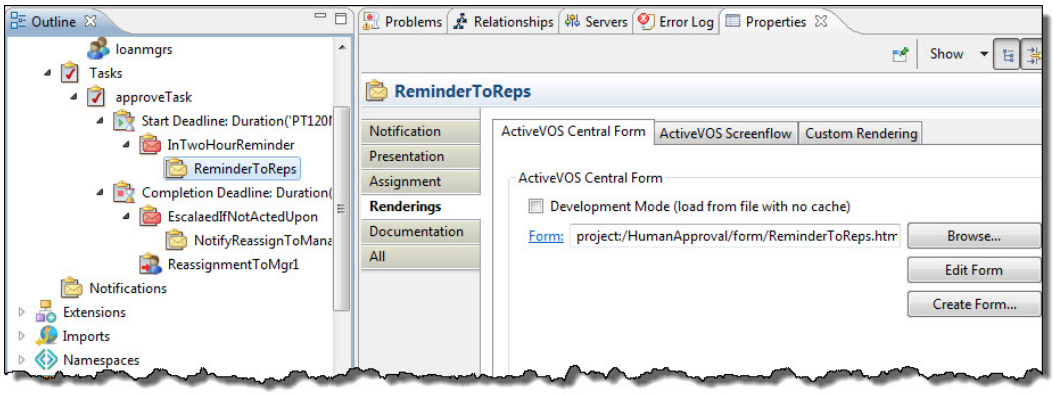
Process Developer can create an HTML Form (or an XSL file or a custom rendering) that will render the task information in the Process Central Task forms. You can edit this form to customize how task information is presented to the user in the Process Central web application.

1. In the Outline view select the *approveTask*.
2. In the Properties view select the Renderings tab.
3. The **Create Form** button presents a dialog were you can specify the `project` folder and name for the form.
4. You will create the form in the `form` folder and use the default name of `approveTask.html`.



After you select **OK**, the Form editor opens.

5. You can see a preview of the form in the top half of the editor. Close the form.
6. Create Process Central forms for the task notifications by expanding the approveTask in the Outline view and trhen creating new Process Central Forms for the two notifications: *ReminderToReps* and *NotifyReassignToManager* (highlighted in the following figure).

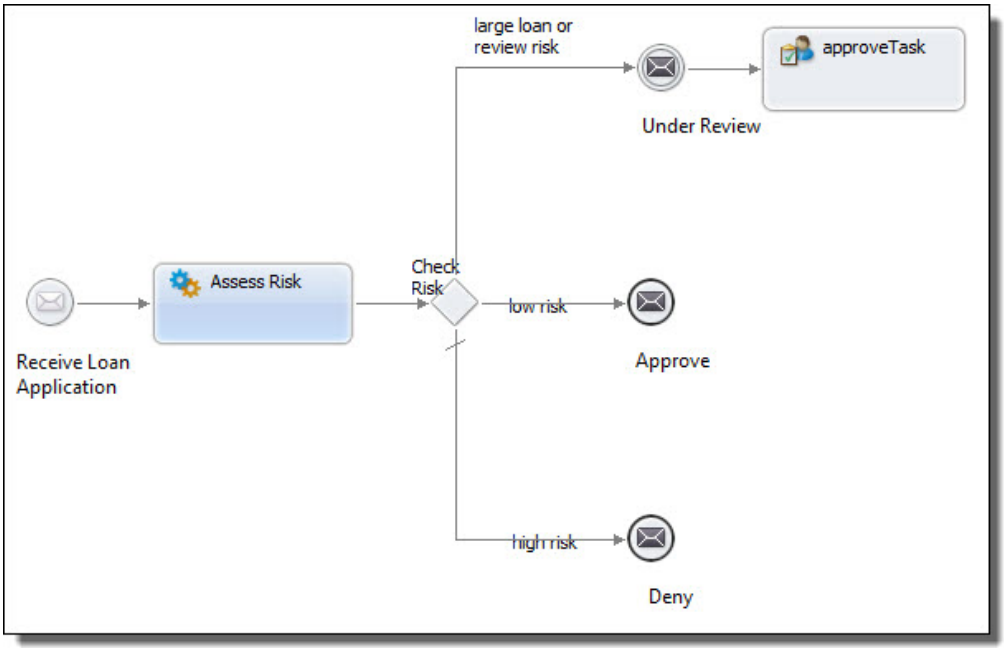


That's it. The task is now complete with escalations, notifications, and rendering information. All that remains is to create a People Activity based on the task, and to place it in the process.

Step 6. Creating the People Activity

Now that you've defined the human task, the last step is to create a people activity in the process that uses the task. Although you've already defined most of the details of the human interaction through setting properties associated with the task, the people activity has a few properties of its own as well, and you need to set them as well.

1. In the Participants view, select the approveTask. Drag it from the Participants view to a position after the `Under Review` activity:



2. Select the people activity, and the associated Properties view. Edit the properties for the activity as follows:
   a. On the People Activity tab, change the name to `ReviewLoan`. Select a color, such as orange.
   b. On the Assignment tab, no changes are required. Note that the role assignment information that was entered during task definition has been propagated to the people activity automatically. It can be overridden here if desired.
   c. The Schedule tab allows you to control when the activity is executed, and how long it is allowed to take. These parameters are separate from the escalation deadlines that are part of the task definition, and all are taken into account at runtime. Set an Expiration expression for a task duration of `'PT180M'`, the maximum allowed duration for the people activity.
   d. On the *Input* tab:
      - In the *Assignment* picklist, select *Single Variable*.
      - In the *Variable* picklist, select *creditInformation*.
   e. On the Output tab:
      - In the *Assignment* picklist, select Single Variable.
      - In the *Variable* picklist, declare a new output variable as follows:
        Select **New Variable**... and assign the following values:

        - *Variable Name*: `approve-task`

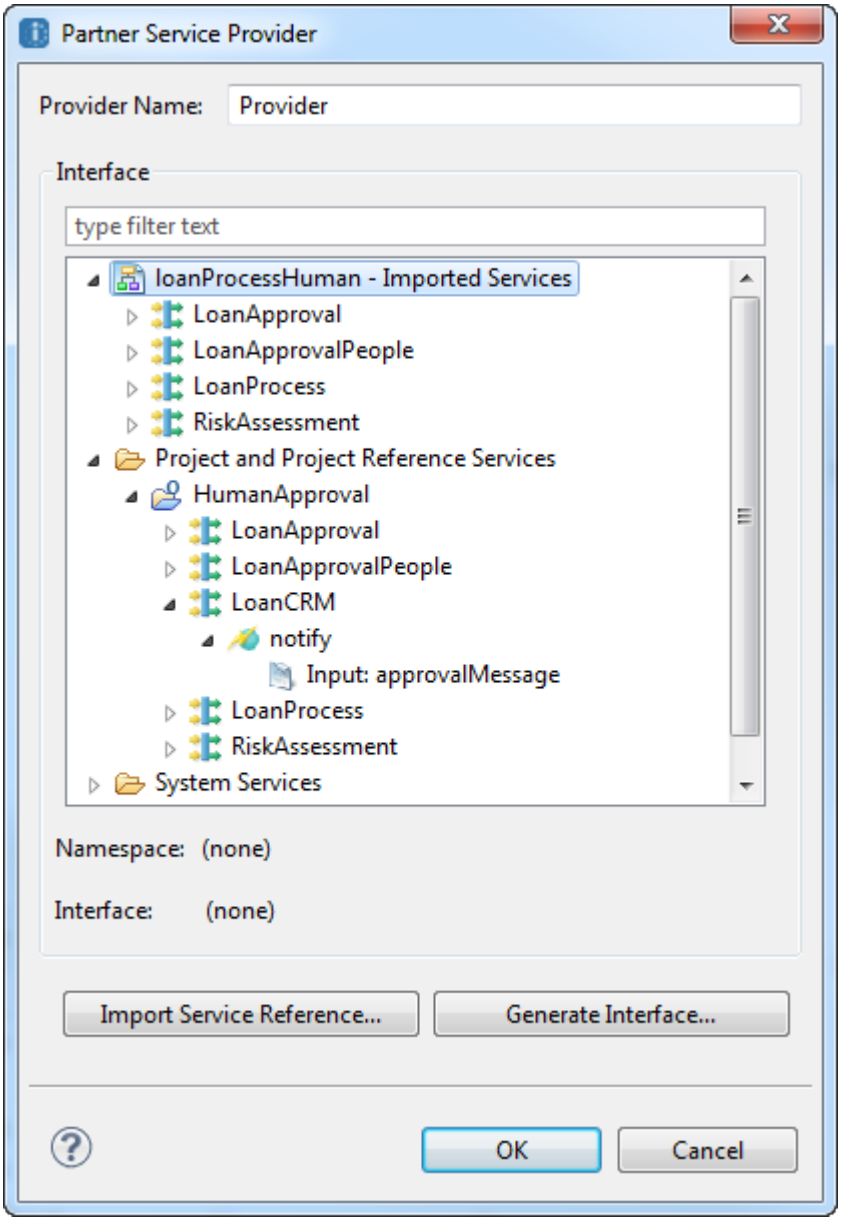
- *Scope*: Accept the default value of Process
- *Variable Type*: Accept the default value of Element

You have now completed the definition of both a human task for loan application review, and a people activity using that task. You've added the people activity to the process at the appropriate location. In the final part of the process, you will notify the Customer Service Department of the loan approval decision, so that they can inform the customer.

Completing the Human Review

In this section, you'll update the loan approval process by invoking a Web service that will notify the Customer Service Department of the loan's status.

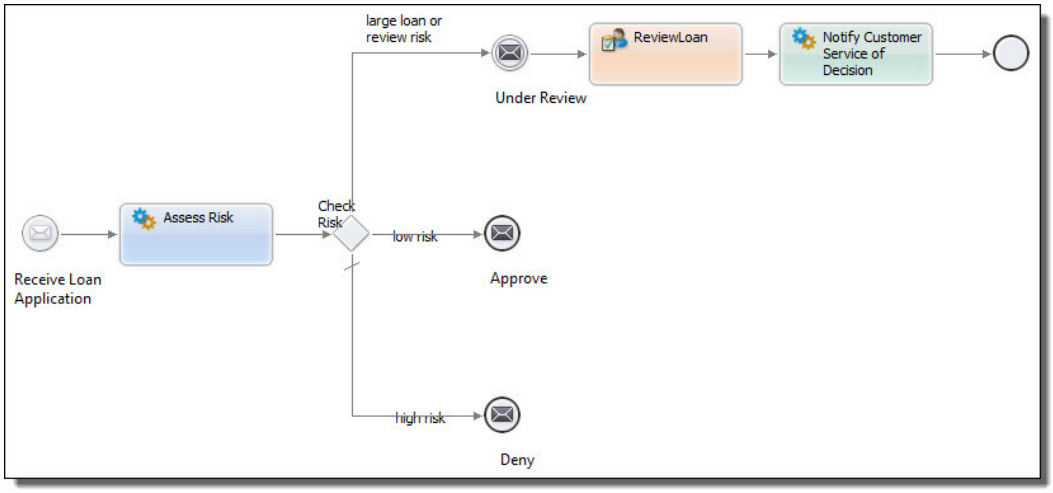
1. In the Participants view, right click Partner Service Providers to create a new Partner Service Provider. Navigate to the notify operation of the LoanCRM port type, select it, and press **OK**.



2. The LoanCRM port type is added to the Participants view with the default name `Provider`. Highlight this entry in the Participants view, and change the Name value to `CustomerService`.

3. You are now ready to add the invoke to the process.
In the Participants view, expand the CustomerService provider and select the *notify* operation. Drag it to the canvas and drop it just after `ReviewLoan`. A new invoke activity will appear, with the default name `notify`.

4. Edit the properties of the invoke as follows:
   a. On the Invoke tab, name the activity `Notify Customer Service Of Decision`. Add a color, such as green.
   b. At the Input tab, select Single Variable, and select approve-task.
5. From the Throw Event palette, add a None event after the invoke.
6. Save your process. Your process should now look similar to this:



Updating Fault Handling

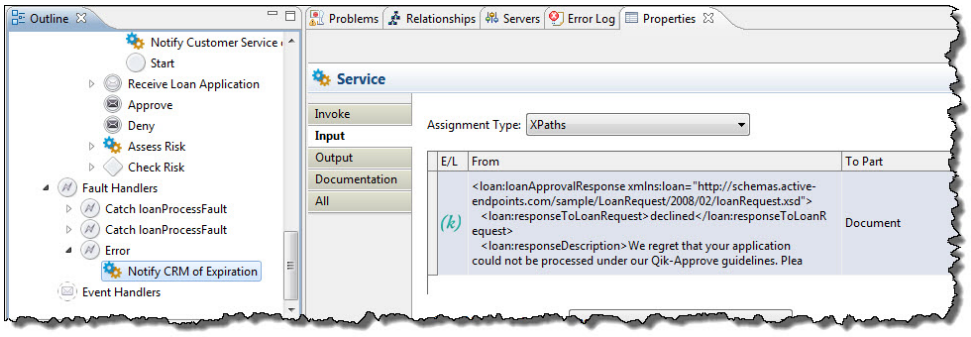
In this section, you will update the fault handling of the original loan approval process to handle an expired human task.

1. Select the Fault Handlers tab.
2. From the Catch Event drawer of the Palette, drag an Error activity and drop it on the Fault Handlers canvas.
3. In the Properties view, click the ... button and set the following property:
   Fault Name: `b4p:taskExpired` (you can use the ellipses... button and select this)
4. From the Participants view, in the Partner Service Provider**s** section, expand the CustomerService provider and drag the notify operation inside the new Catch activity.
   a. In the Properties view on its Invoke side tab, set the Activity Name to *Notify CRM Of Expiration*.
   b. At the Input tab, select XPaths
   c. Choose Literal for the E/L type and use the following:

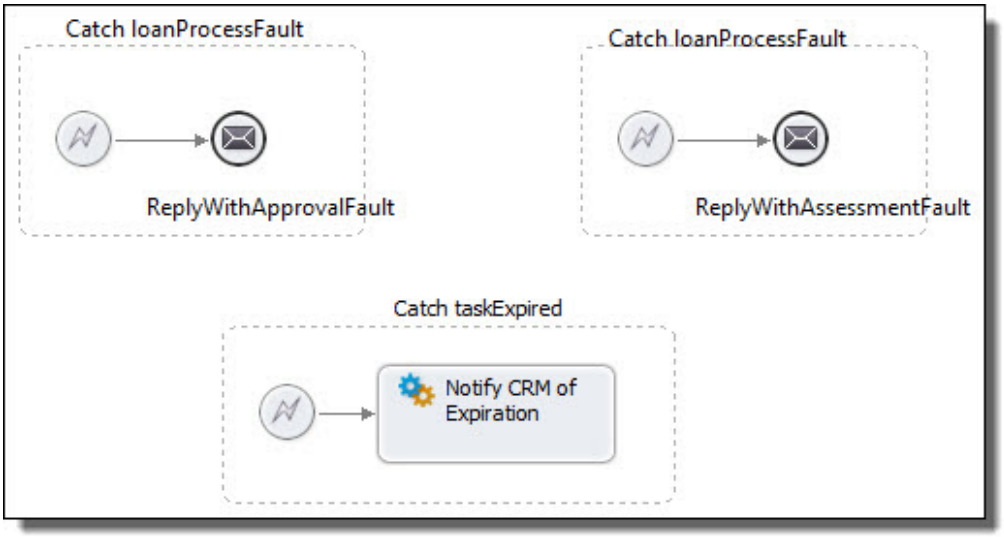
```
<loan:loanApprovalResponse xmlns:loan=
        "http://schemas.active-endpoints.com/sample/LoanRequest/2008/02/
loanRequest.xsd">
   <loan:responseToLoanRequest>declined</loan:responseToLoanRequest>
   <loan:responseDescription>We regret that your application
    could not be processed under our Qik-Approve guidelines. Please
    reapply in person at one of our offices.</loan:responseDescription>
   <loan:rejectionReason>
      <loan:reason>infoRequired</loan:reason>
```

```
        <loan:description>Loan underwriting could not be completed
      in the recommended time window.</loan:description>
     </loan:rejectionReason>
  </loan:loanApprovalResponse>
```

d. *To Part*: Document



5. You can rename the *error* text to `Catch taskExpired`. Your fault handler should now look similar to this:



Simulating the Process

Now that the process is complete, you should simulate its execution before deploying and running it on the server. Detailed instructions for simulating the Loan Process are described in *Tutorial Part 8, Simulating the Process* (located elsewhere in this help) so they are not repeated here. Once you are satisfied that your process behaves as expected, you can deploy the process and run it on the Process Server.

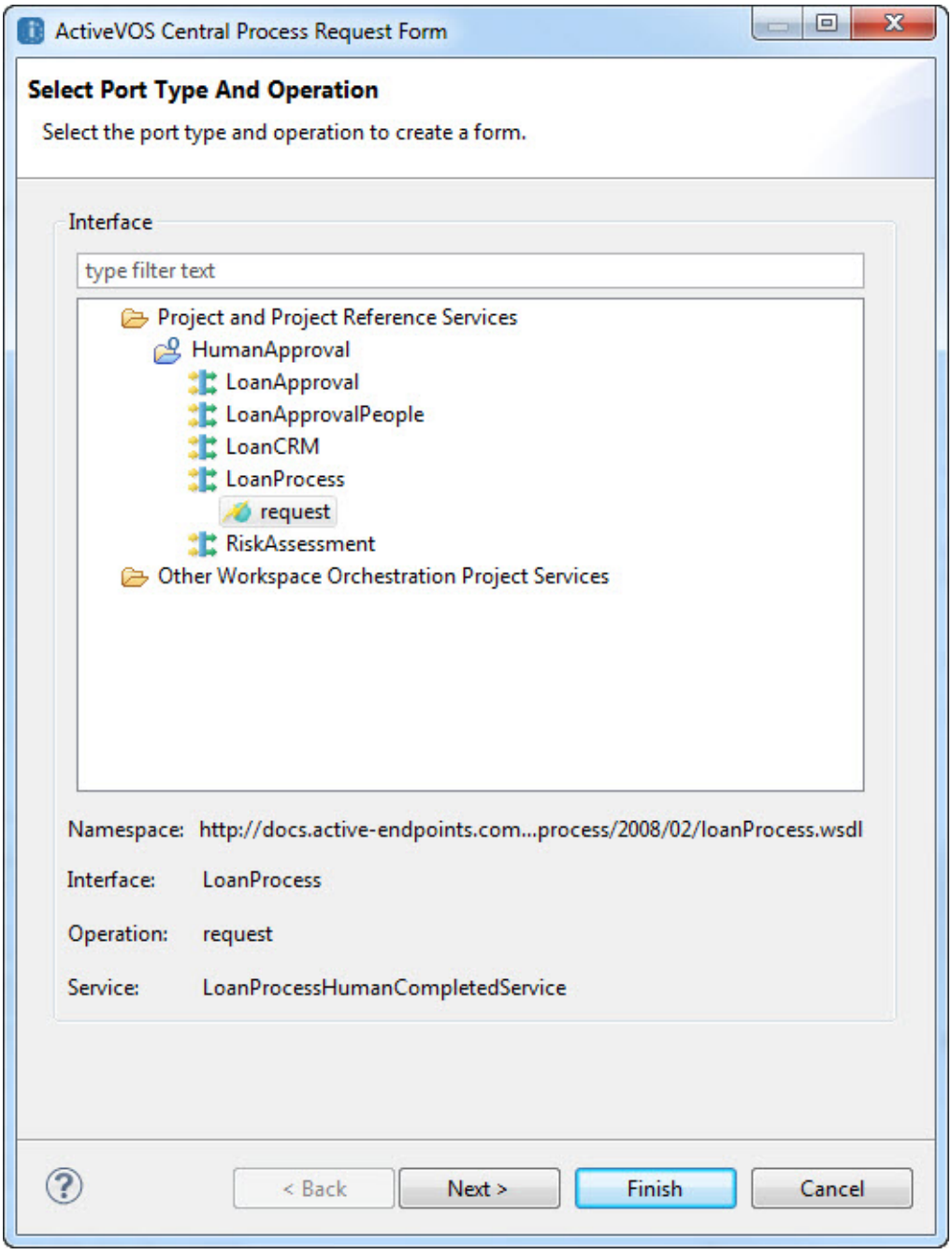
Creating a Request Form

Before deploying the process, you must create a request form that you can use to invoke the process on the server. Two steps are required here:

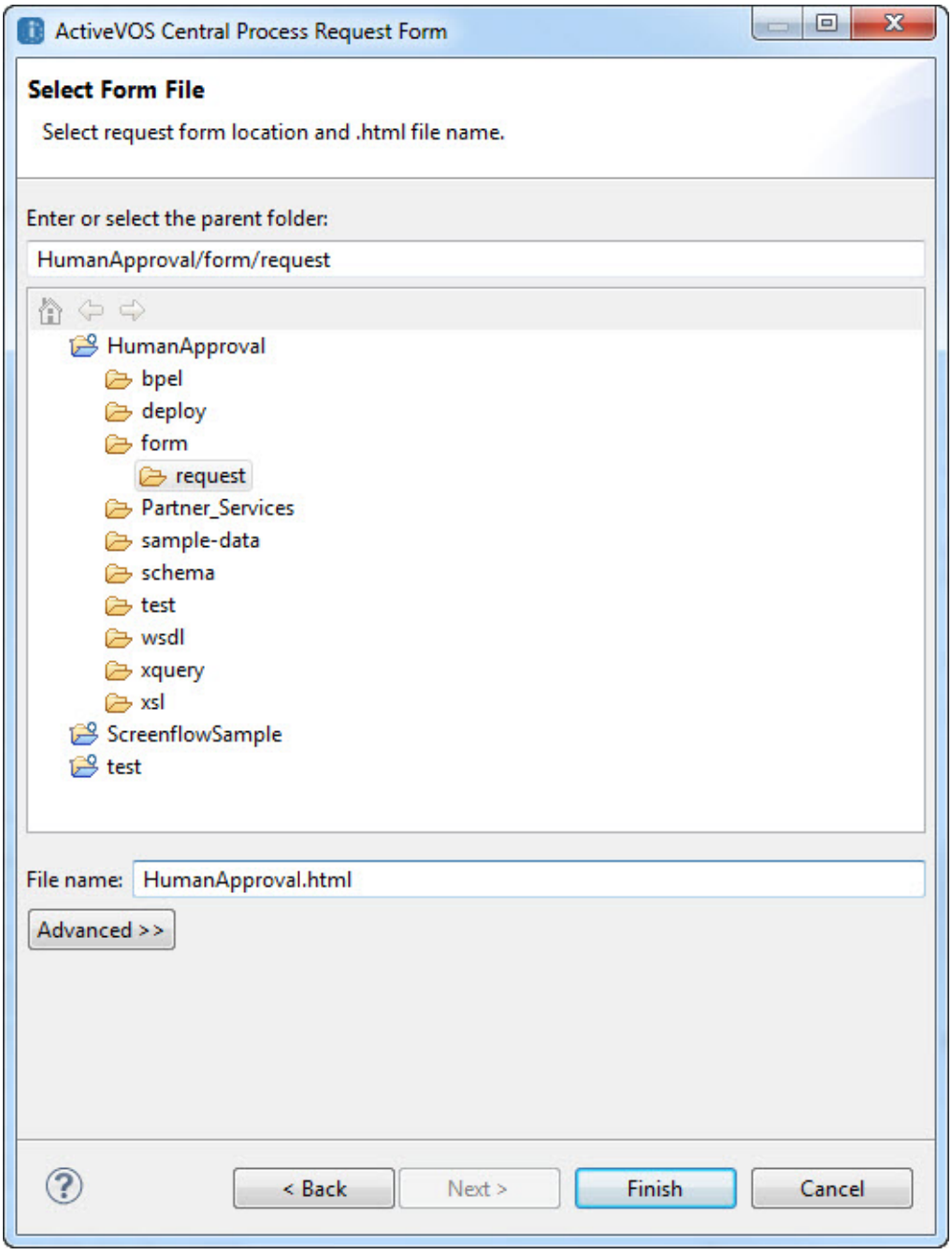
1. Create the Request form.
2. Create a Process Central Configuration (`.avcconfig`) file that defines the structure and contents of the application.

Step 1: Create the Request Form

1. In the Project Explorer, create a sub-folder under the `form` folder called `request`.
2. In the Project Explorer, right-click on the `form` folder and select **New > Process Request Form**.
3. Expand the LoanProcess port type and select the request operation as shown here:



4. Press **Next**.
5. Name the form `HumanApproval.html`.



6. Press **Finish**.

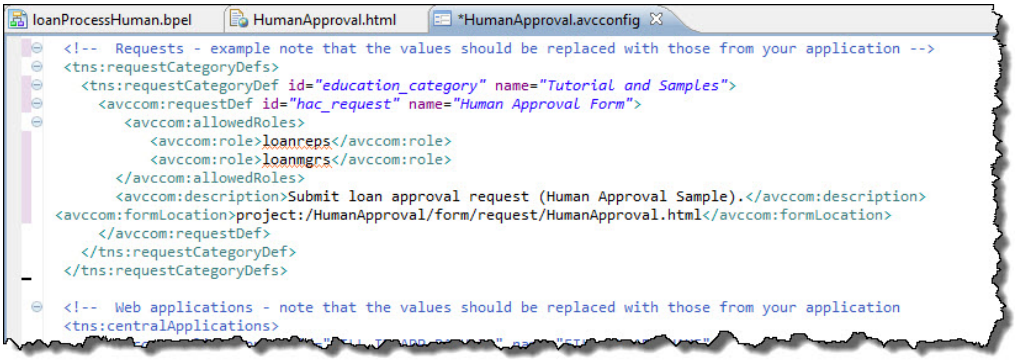
Step 2: Create the Process Central Configuration

1. In the Project Explorer, right-click on the `deploy` folder and select **New > Central Configuration**. Enter a value of `HumanApproval.avcconfig` for the file name and press **Finish**.
2. In the Source tab, scroll down to the Requests section and replace the `<tns:requestCategoryDefs>` with this code:

```
<!-- Requests - example note that the values should be replaced with those from your
application -->
  <tns:requestCategoryDefs>
```

```
    <tns:requestCategoryDef id="education_category" name="Tutorial and Samples">
     <avccom:requestDef id="hac_request" name="Human Approval  Form">
        <avccom:allowedRoles>
           <avccom:role>loanreps</avccom:role>
           <avccom:role>loanmgrs</avccom:role>
        </avccom:allowedRoles>
        <avccom:description>Submit loan approval request (Human Approval Sample).</
avccom:description>
        <avccom:formLocation>project:/HumanApproval/form/request/HumanApproval.html</
avccom:formLocation>
     </avccom:requestDef>
    </tns:requestCategoryDef>
  </tns:requestCategoryDefs>
```

3. This will deploy the `HumanApproval.html` file to Process Central and allow access to anyone with the loanreps or loanmgrs role. The file should look similar to this:



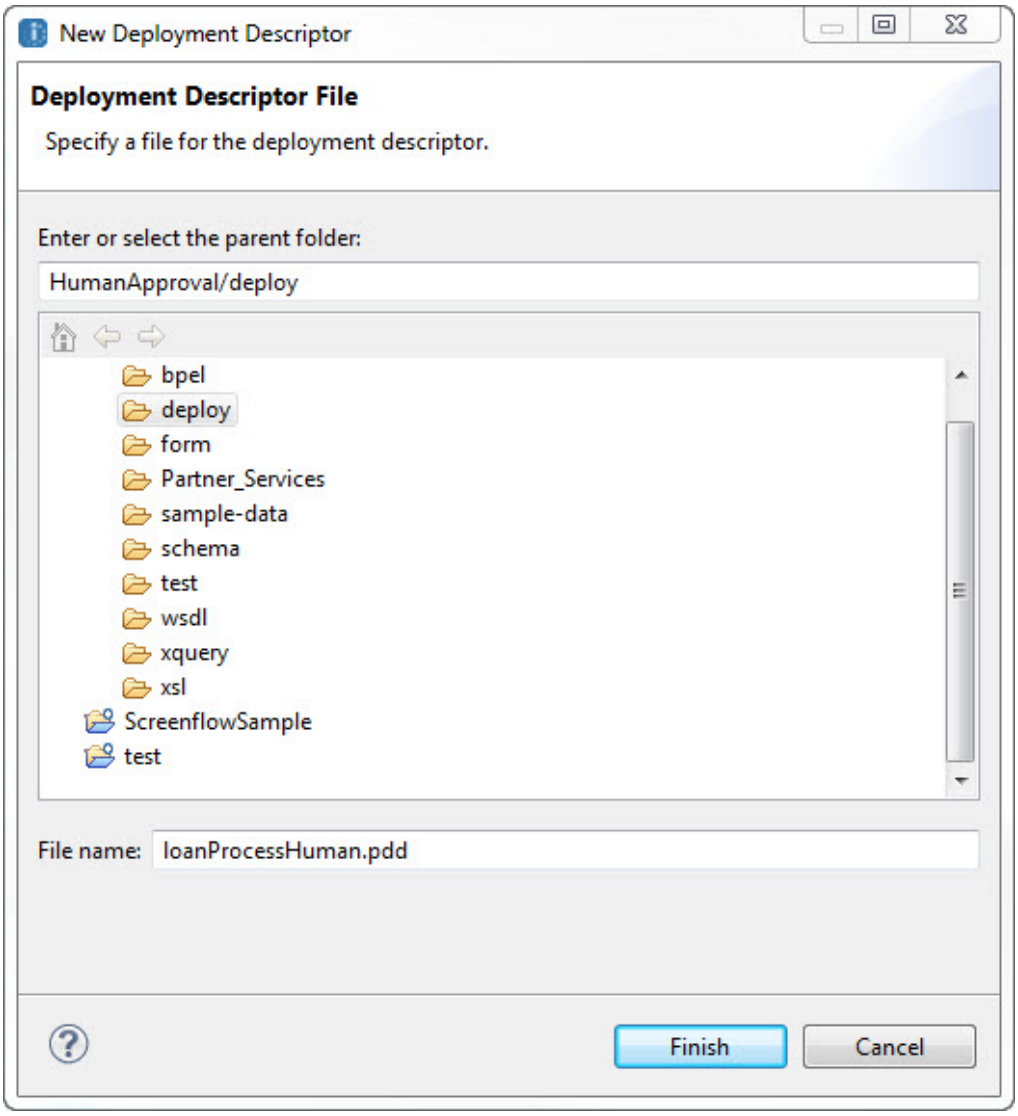
4. Save the file.
5. The request form and the Process Central configuration file can now be deployed. Instructions for deploying can be found in the Human Approval Completed Orchestration Process template.

Preparing for Deployment

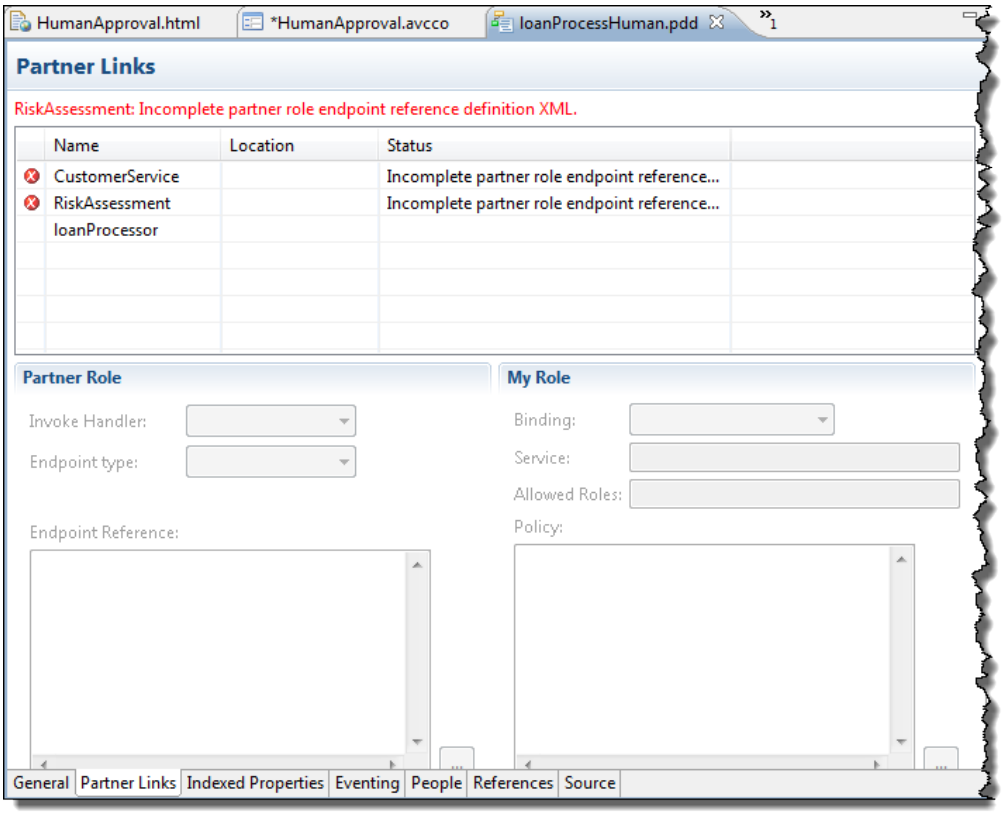
In this section, you'll prepare the process for deployment by creating the Process Deployment Descriptor (`.pdd`) file. For more information, see *Deploying the Loan Process* and *Deploying the Partner Processes*.

A Process Deployment Descriptor (`.pdd`) file describes the relationship between the partner links defined in the BPEL file and the implementation that interacts with partner endpoints. You create a `.pdd` file to add address information about your endpoint references. The `.pdd` file is an integral part of the deployment package for the process.

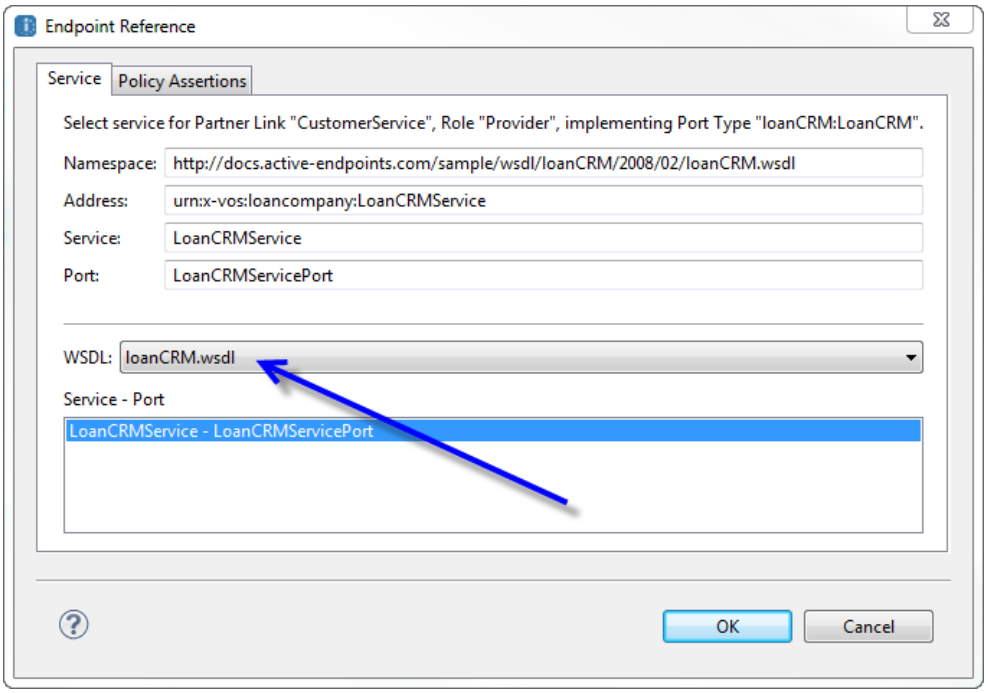
1. In the Project Explorer, right-click on the *loanProcessHuman.bpel* process and select **New > Deployment Descriptor** to open the **New Deployment Descriptor** dialog.
2. Select the `deploy` folder, and press **Finish**, as shown.



3. The `loanProcessHuman.pdd` file opens in the PDD editor The properties of the General tab are displayed.
4. Accept the defaults on the General tab and press the Partner Links tab to define the bindings to the web services called by the process.

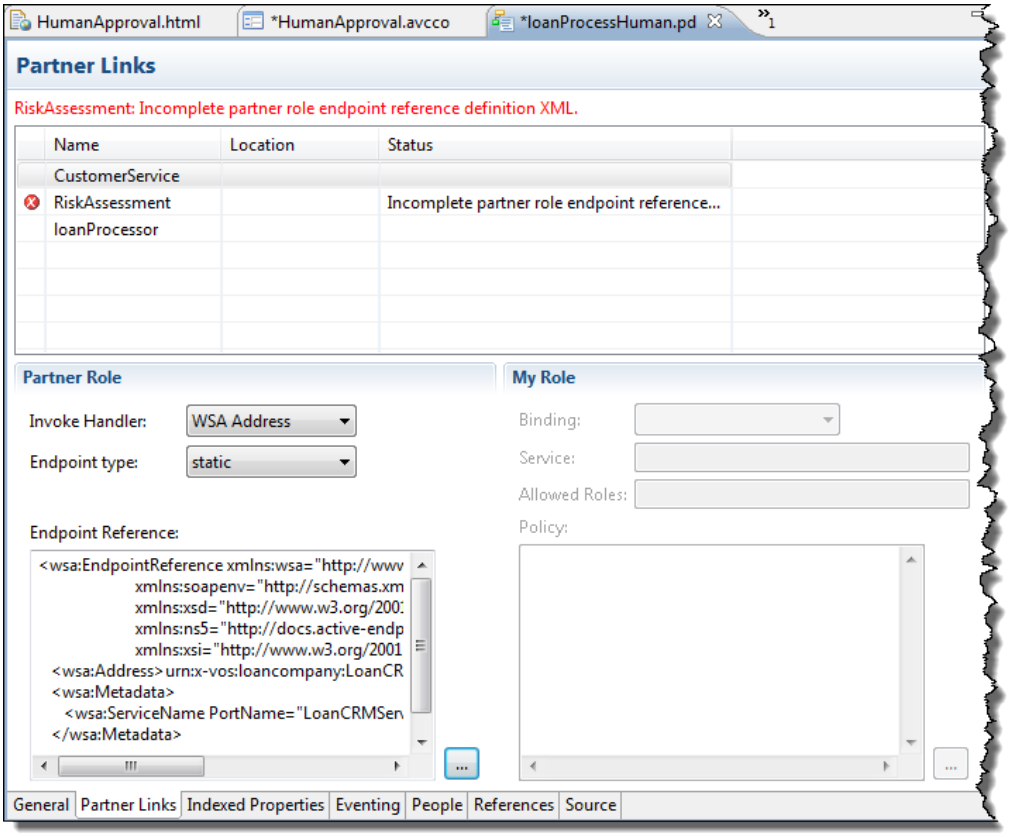


5. Set the runtime bindings for the CustomerService partner link by clicking on the CustomerService row and set the following properties:
   - *Invoke Handler*: **WSA Address**
   - *Endpoint type*: **static**
   - Press the ellipses (...) button next to the textarea labeled Endpoint Reference to open the **Endpoint Reference** dialog
   - Select the `loanCRM.wsdl` file from the WSDL picklist.

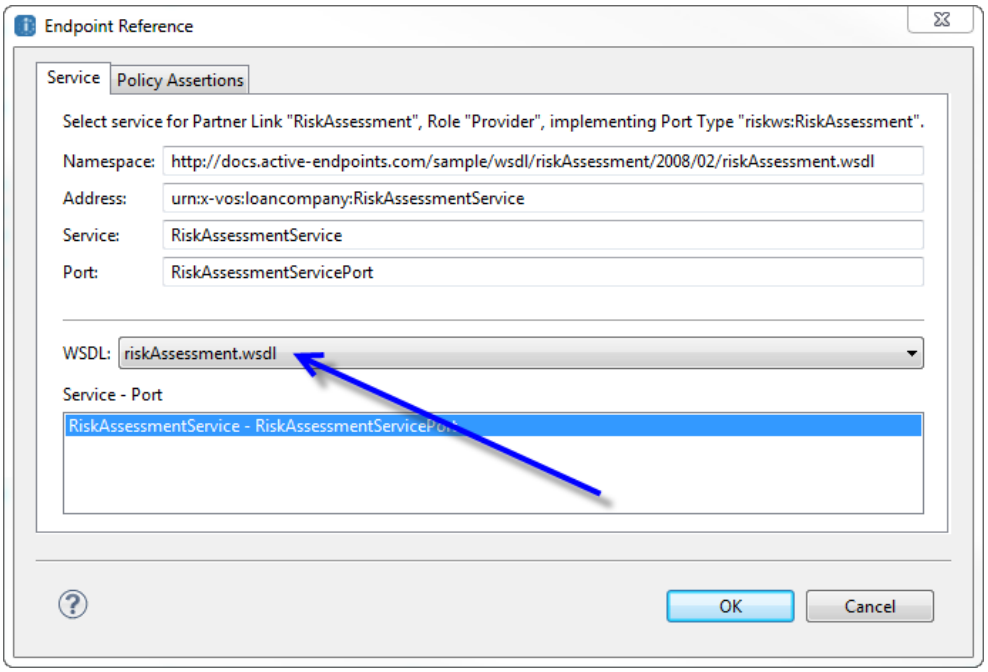


- Press **OK**.

Notice that the Endpoint Reference information is now filled in using the values contained in the `loanCRM`.wsdl file:



6. Set the runtime binding information for the RiskAssessment partner link by selecting the RiskAssessment row and set the following properties:
   - *Invoke Handler*: WSA Address
   - *Endpoint type*: static
   - Press the ellipses (...) button next to the textarea labeled Endpoint Reference to open the **Endpoint Reference** dialog
   - Select the `riskAssessment.wsdl` file from the WSDL picklist.

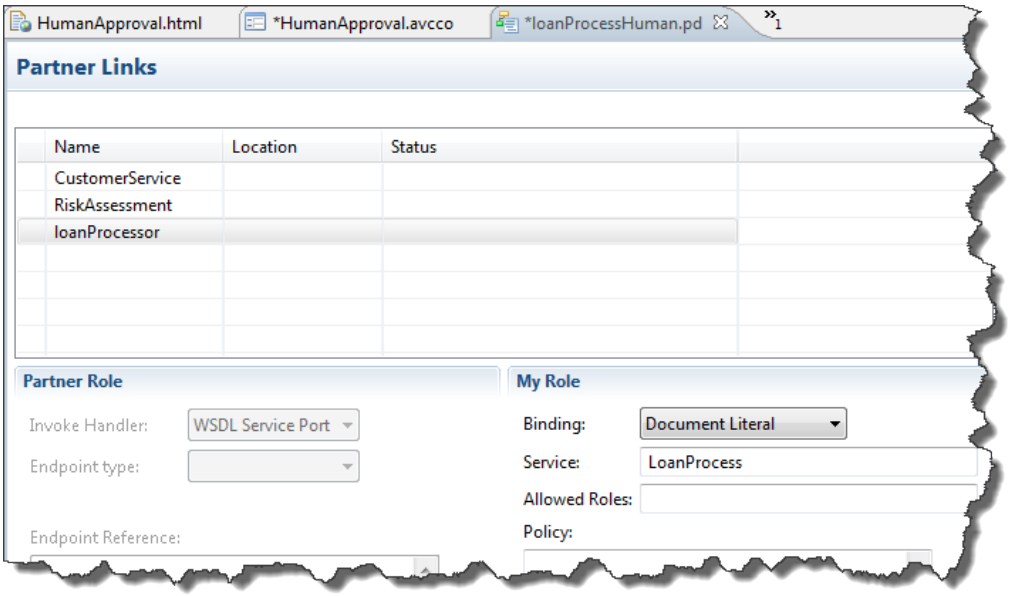


   - Press **OK**.

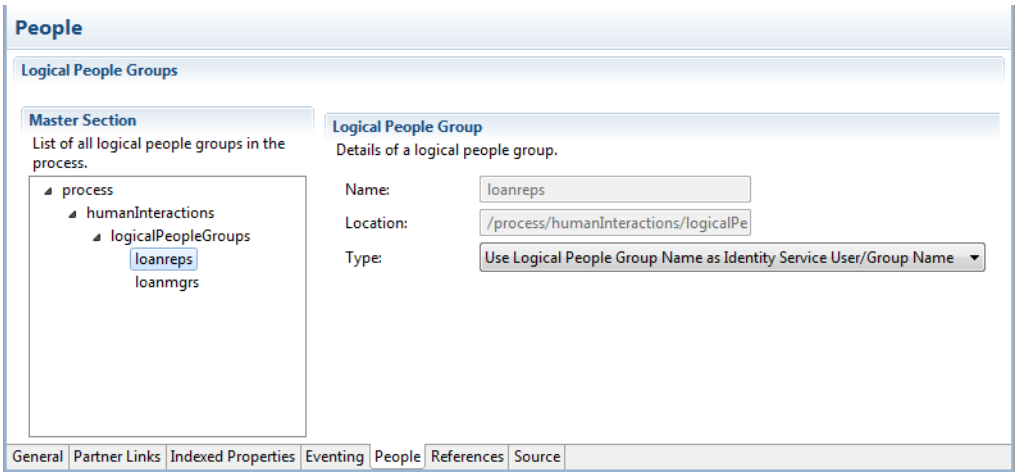
   Notice that the Endpoint Reference information is now filled in using the values contained in the `riskAssessment.wsdl` file.

7. Next set the runtime binding information for the LoanProcess partner link by selecting the LoanProcess row and setting the following properties:
   - *Binding*: Document Literal
   - *Service*: LoanProcess Note that this name must match the Service name selected in the `HumanApproval.html` request form, described in *Creating a Request Form* above.

   Notice that the Endpoint Reference information is now complete:



8. The final step is to bind the Logical People Groups to actual server roles. Select the People tab and notice that the Logical People Groups are already assigned to names in the `tomcat-users.xml` file for the Process Developer embedded server Identity Service. Because Identity Service group names are used in the People activity, they are already selected here.



9. Save the file.

Running the Process on the Server

Your process is now ready to be deployed. Deployment is the act of publishing your BPEL process to the Process Server where it can run.

CHAPTER 3

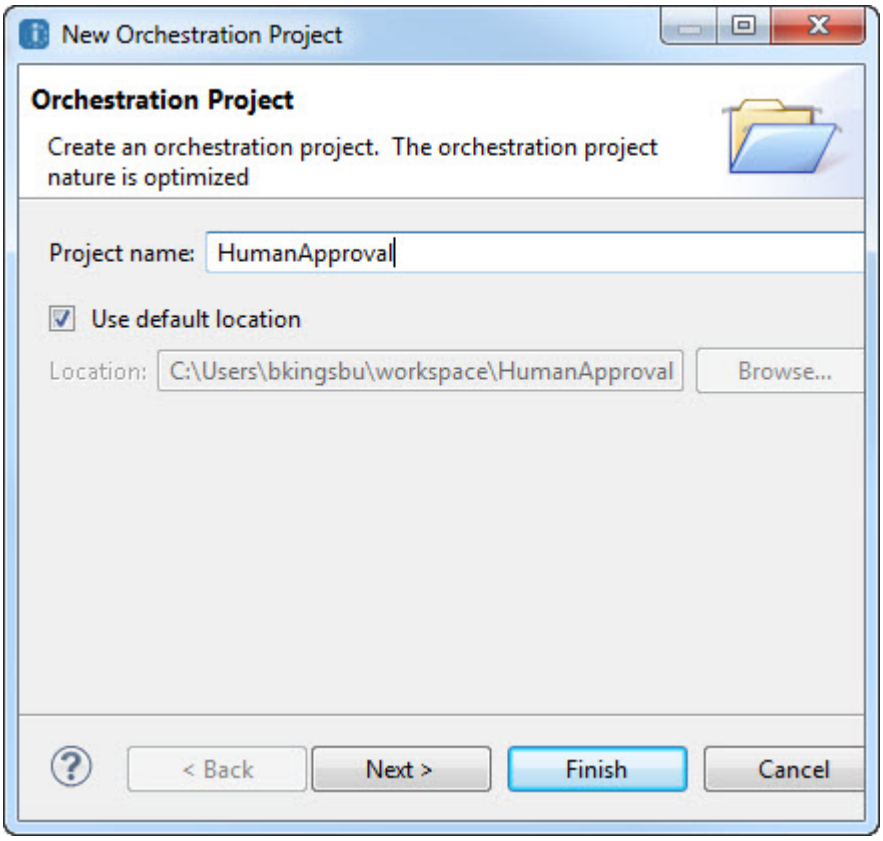
Human Approval Completed

This chapter includes the following topics:

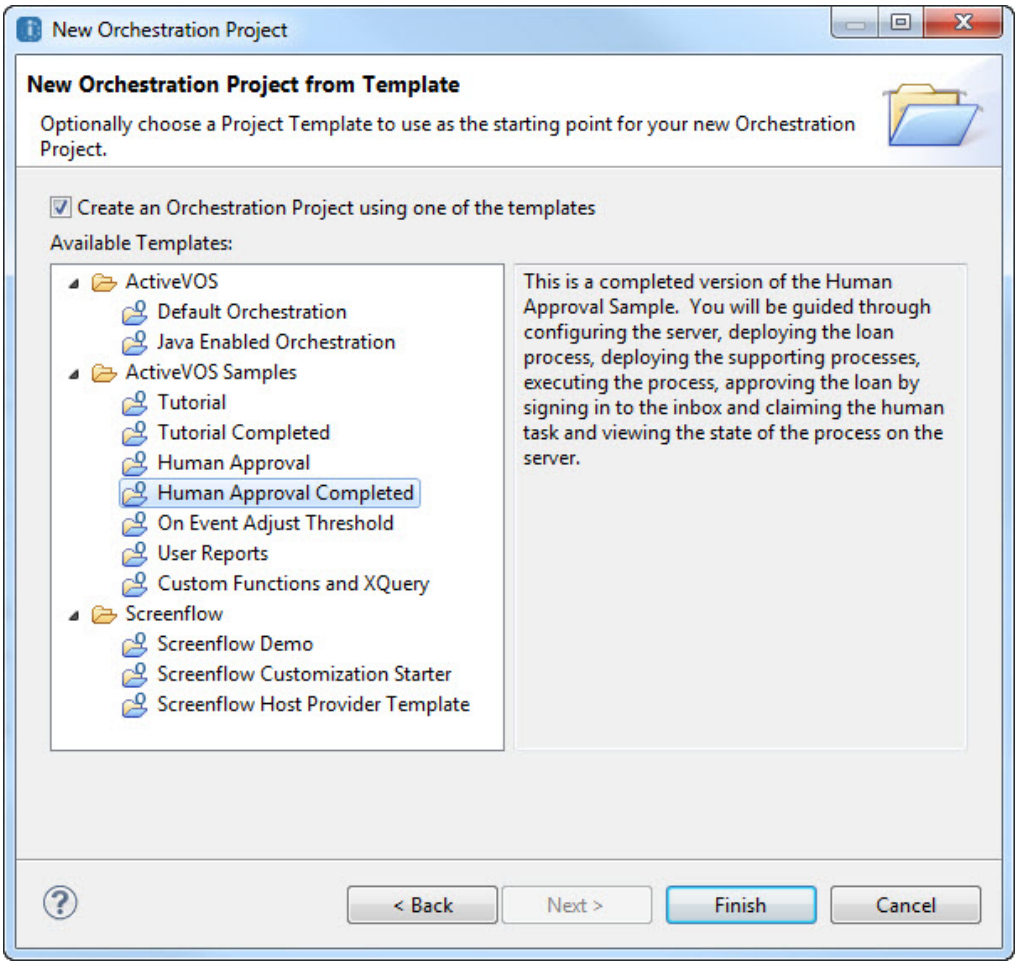
Getting Started with the Completed Project

To start, you will create a new orchestration project based on the *Human Approval Completed* sample template. By using this template, you will create a new orchestration project that includes a complete loan approval process, including human interactions.

1. Open the Process Developer Designer and select **File > New > Orchestration Project** from the main menu.
2. Type `HumanApprovalCompleted` for the Project name and press **Next**.



3. Select *Human Approval Completed* from the list of samples to use the Human Approval Completed template.



4. Press **Finish**.

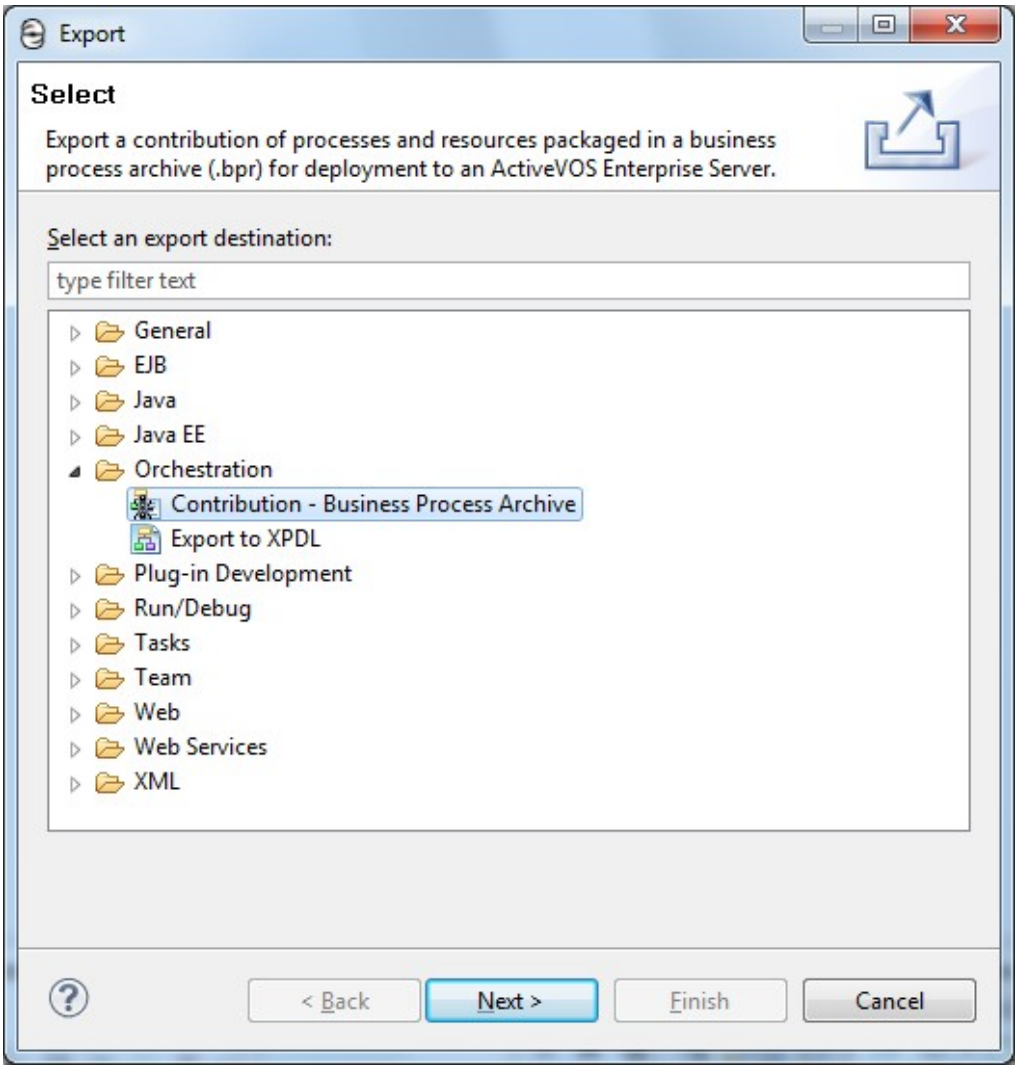
Starting the Process Developer Embedded Server

Before you can deploy the Loan Process and the supporting partner processes to the Process Developer Server, you must start it. Instructions for starting the server are in *Starting the Process Developer Server* located elsewhere in this help. After starting the server, continue to the next topic.

Deploying the Loan Process

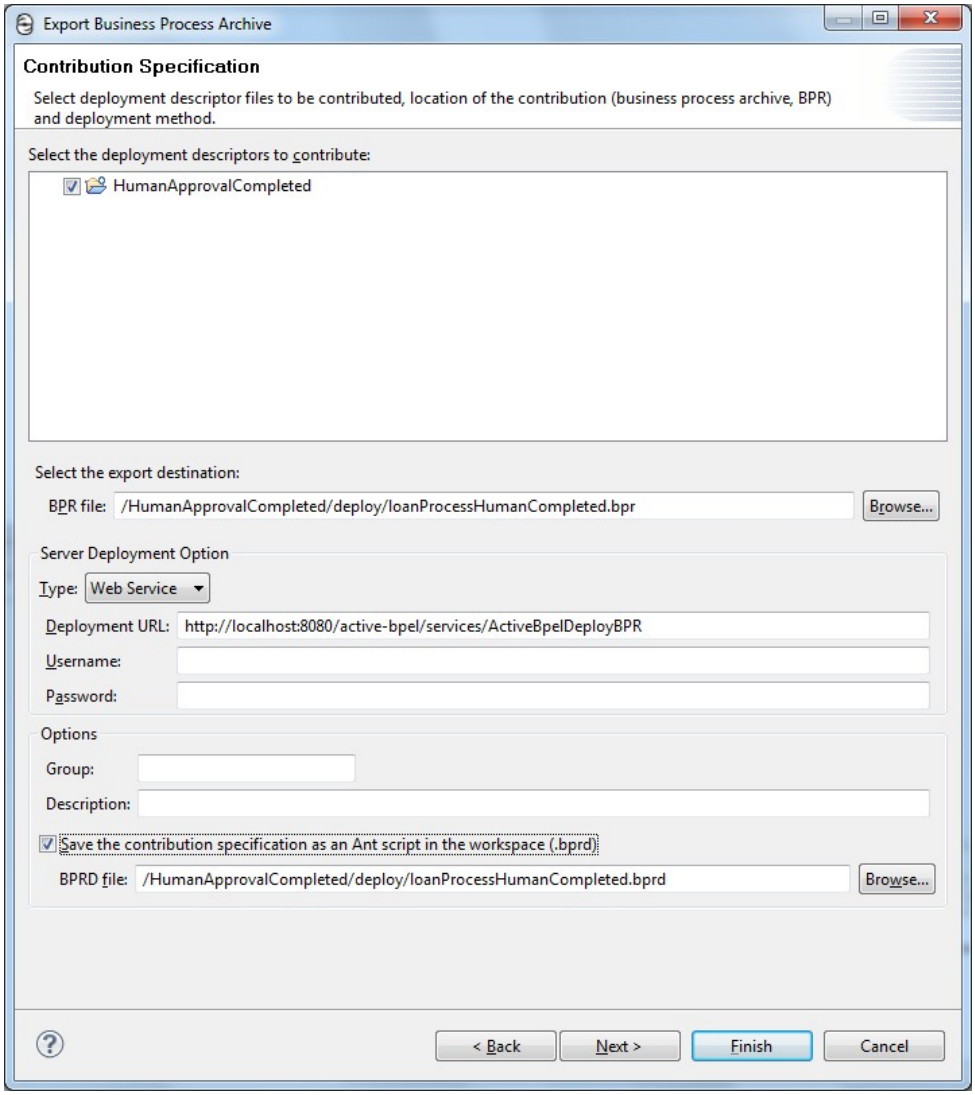
To deploy the Loan Process, you need to create a Business Process Archive (`.bpr`) file by exporting the Process Deployment Descriptor (`.pdd`) file. These files are packaged in the `deploy` folder of this sample along with another file containing a `.bprd` extension.

1. In the Navigator view, right-click on the *HumanApprovalCompleted* project and select **Export**.
2. Expand the Orchestration folder and select *Contribution - Business Process Archive*.

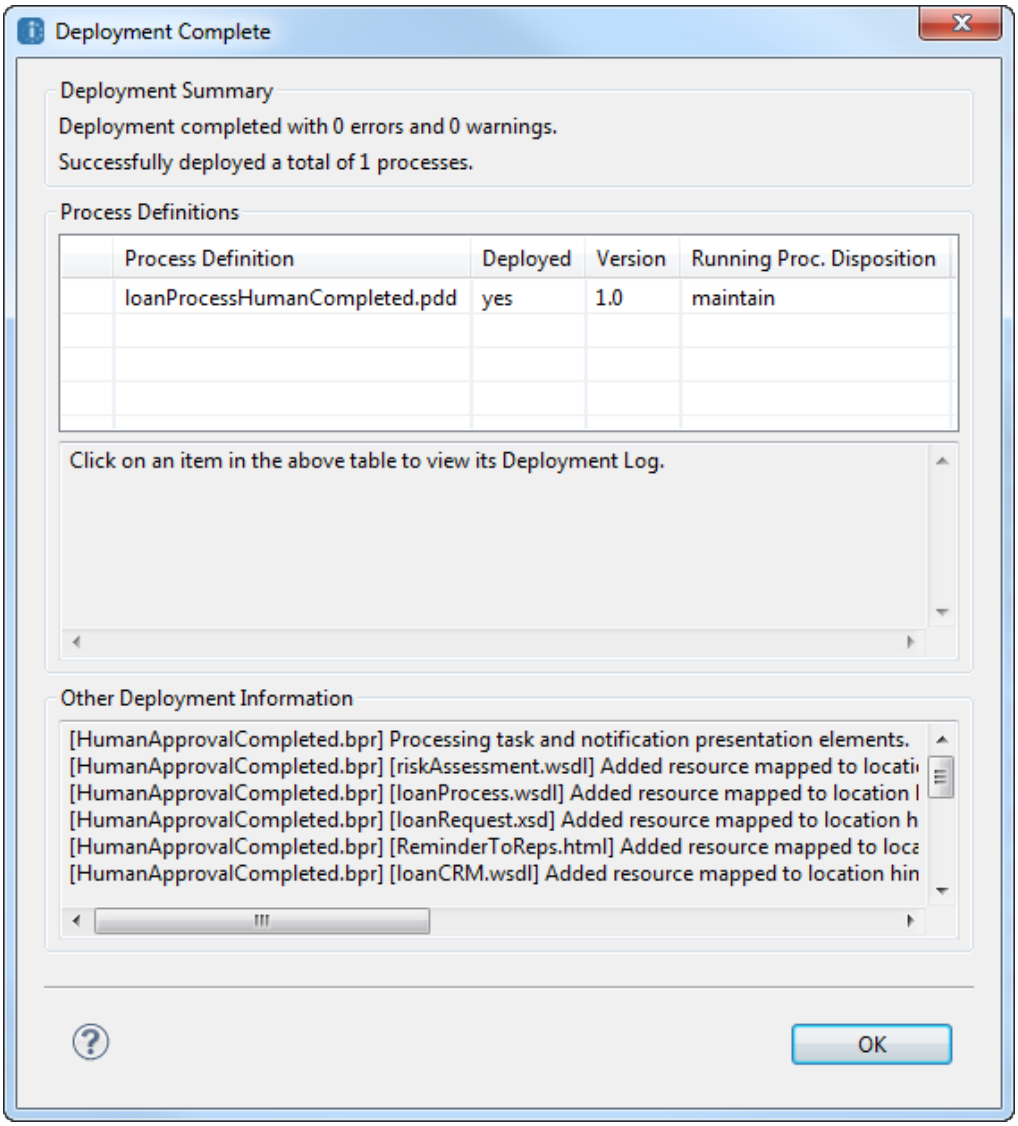


3. Press **Next**.
4. Set the following properties:
   a. Make sure that the `loanProcessHumanCompleted.pdd` file is selected.
   b. Press the **Browse** button and navigate to the `deploy` folder of your project and specify a file named `loanProcessHumanCompleted.bpr`. **Note**: the file is overwritten if it already exists.
   c. Select *Web Service* from the *Type* picklist.
   d. Create a `.bprd` file by checking *Save the contribution specification as an Ant script in the workspace (.bprd)* checkbox near the bottom of the dialog.

e. Navigate to the `deploy` folder of your project and specify a file named `loanProcessHumanCompleted.bprd`--if the file already exists, you will be asked if it is ok to replace it. The `.bprd` file will have all the choices that you specify during the deployment and makes re-deploying the process a one-step process. You will see how to deploy using the `.bprd` file in the next topic when you deploy the partner processes.
Your dialog should look similar to this:



5. Press the **Finish** button.
A dialog indicating the status of the deployment is displayed:



6. Press **OK**.

The Loan Process is now deployed.

Deploying the Partner Processes

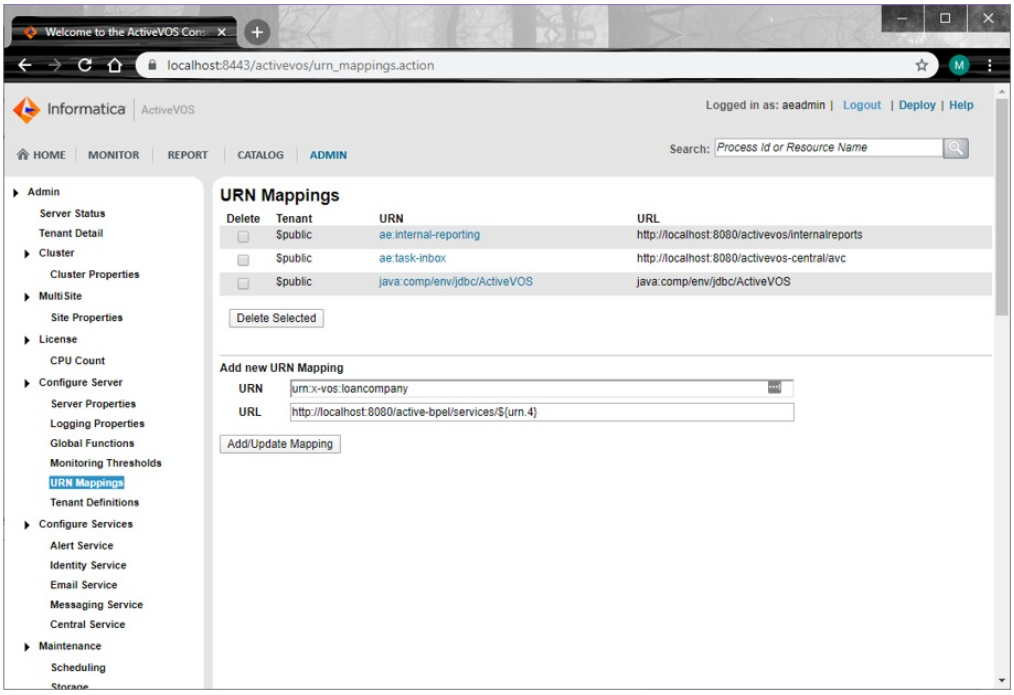
You will now need to to deploy the two partner processes that the Loan Process uses to process a loan. The BPR files for the partner services are included in the `Partner Services` folder of the *HumanApprovalCompleted* project. You will use the Process Console to deploy the `loanCRM.bpr` and `riskAssessment.bpr` BPR files. Instructions for doing this are located in `Tutorial: Part 10`.

Running the Loan Process on the Server

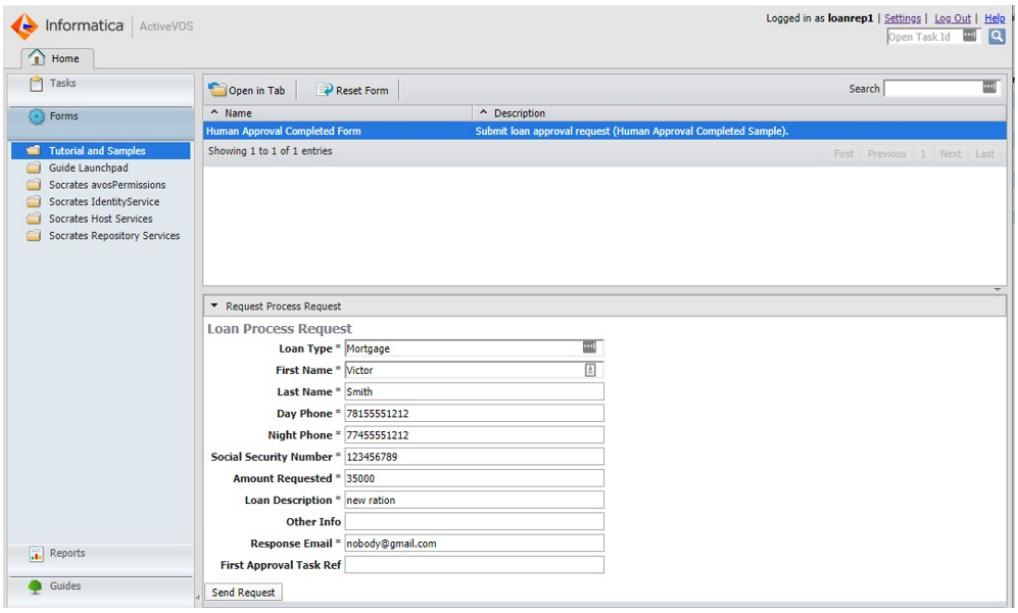
Before you can run the process, you need to ensure that the URNs that are used in the Process Deployment Descriptor (`.pdd`) file that defined the partner services are mapped to the actual URLs where those services were deployed to. Afterward, you can run a simple Ant script to run the loan process.

1. Open the Process Console, normally located at `http://localhost:8080/activevos/` and navigate to **Admin > Configure Server > URN Mappings** page and set (or verify) the following URN Mapping:
   URN: `urn:x-vos:loancompany`

   URL: `http://localhost:8080/active-bpel/services/${urn.4}`

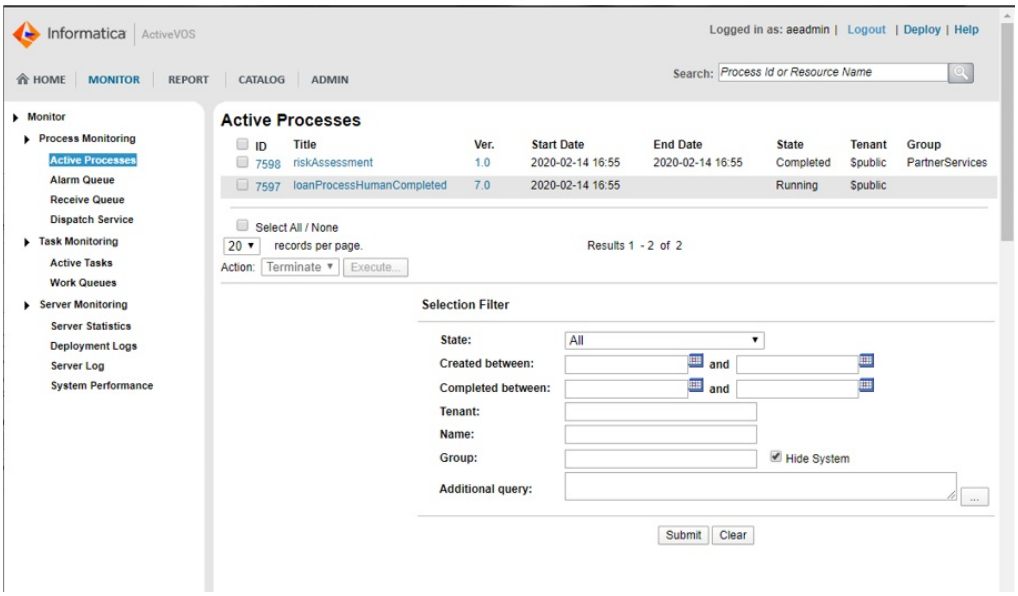


2. Enter the following link in a web browser: `http://localhost:8080/activevos-central/`.
3. Sign in using the following credentials:
   - *Username*: `loanrep1`
   - *Password*: `loanrep1`
4. In the left side navigation pane, choose the Tutorial and Samples section from the Requests tab and then select the Human Approval Completed Form.
5. Enter details as shown below and submit the form by pressing the **Send Request** button:

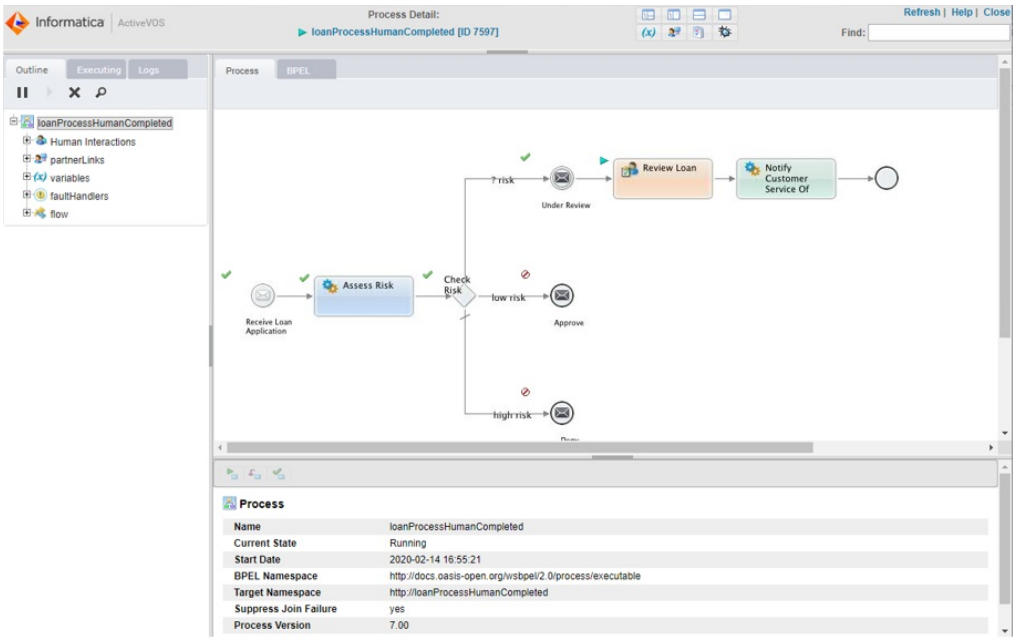


6. The request is submitted to the server and the response is written into the Process Output section:

7. To see the status of the running process, enter the following link in a web browser: `http://localhost:8080/activevos/`.
8. Select **HOME > Active Processes** from the Process console:



9. Press the `loanProcessHumanCompleted` process to view the details of the process. As the following figure shows, the process is waiting on the ReviewLoan People activity:



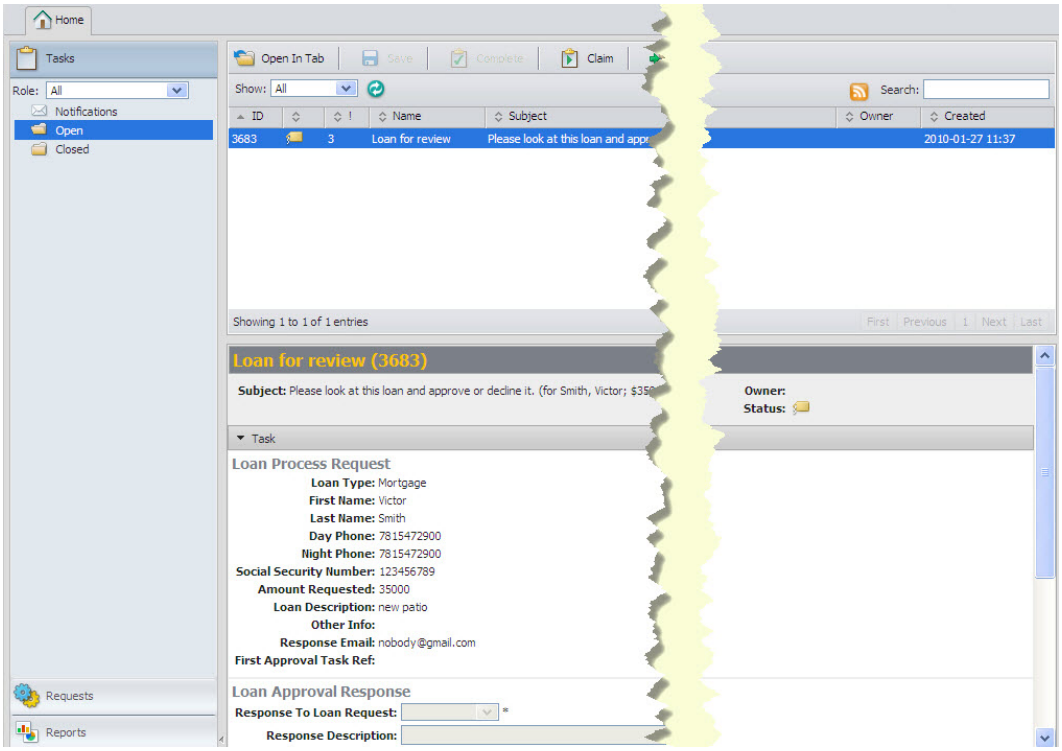
10. The process will wait in this state until the People activity is completed or expires.

Claiming Updating and Completing the Human Task

The loan that was requested requires human intervention to complete due to the amount of the loan. To complete the loan process a human loan representative must claim and complete the task.

Claim and Start the task

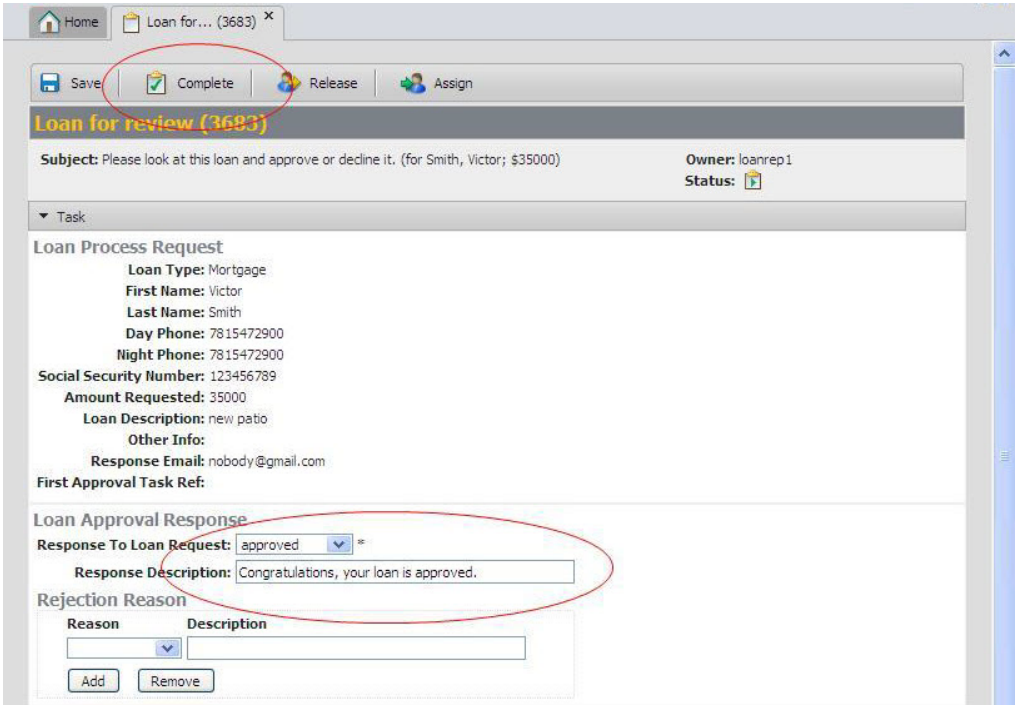
In Process Central, select the Tasks tab. A new task was added to the list of unclaimed, open tasks for the user loanrep1 as the screen shot below shows. If the task is not displayed press the **Refresh** button. Start this task by selecting the row and pressing the **Claim** button and then the **Open In Tab** button:



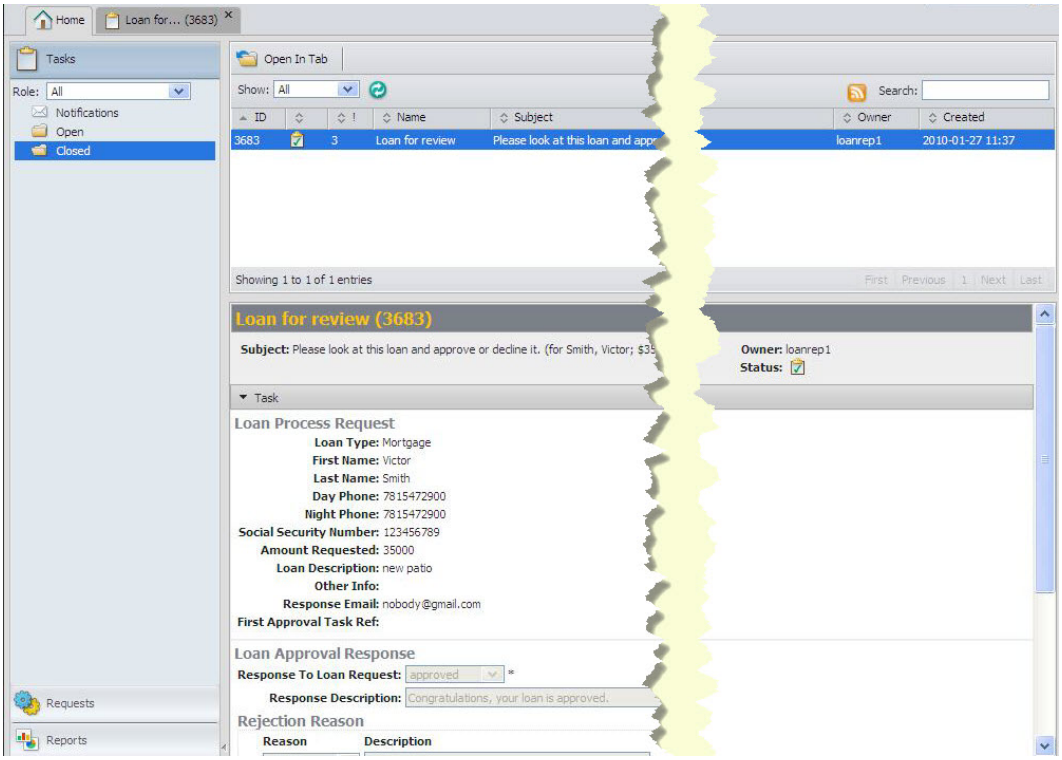
Complete the task

1. Change the value of the ResponseToLoanRequest picklist to approved.
2. Enter some text for the ResponseDescription. **Note**: You must enter a string that is greater than 10 characters.

3. Press the **Complete** button to complete the task.



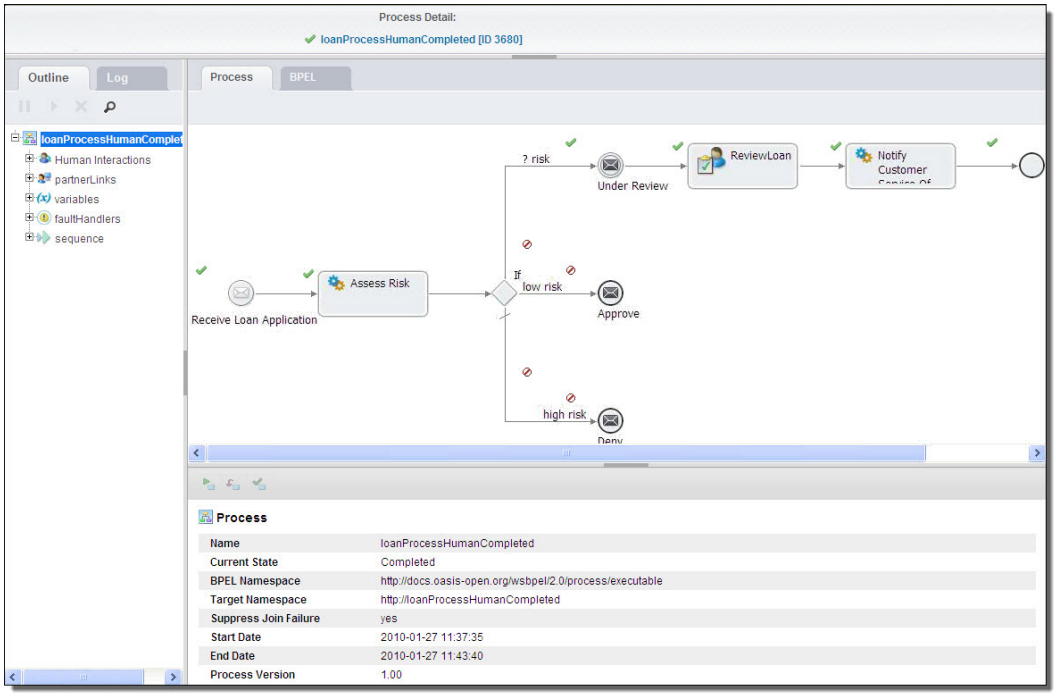
Go back to the Task list and verify that the task is now completed and moved to the list of Closed tasks:



Review the Completed Loan Process

Once the human task completes, the `loanProcessHumanCompleted` process will continue on and complete. You can check this by refreshing the `loanProcessHumanCompleted` process from the list of Active Processes or from the process details

Press the **Refresh** button to see the current state of the process. As the figure below shows, the `ReviewLoan` People activity has completed. All other activities in the process have also completed, leaving the process in the Completed state.



CHAPTER 4

User Reports

This chapter includes the following topic:

User Reports Sample

This sample has reporting samples and information that will help you design and deploy your own reports. The first step is to create the report using the reporting module in Process Developer is built on the robust BIRT reporting engine. That engine has eclipse plug-in support, which Process Developer includes for you. These plugins contain a rich editing environment for the construction of new reports. Once you have created a report, you can then use a Business Process Archive (BPR file) to deploy it to the server as a resource.

After deployment, the report is available on the Report page of the console and optionally on the Reports tab of Process Central if you specify this using a Process Central configuration (`.avcconfig`) file. This sample's documentation supplies information about using reports with Process Developer, but only helps supplement the reporting module's (BIRT) documentation. Please refer to the BIRT documentation for more details on advanced reporting features.

This topic discusses the following:

- Getting Started
- Sample Reports
- Deploying Reports
- Process Data Model
- Process Data Model Entity Relation Diagram Subset
- Human Task Data Model

Getting Started

You can add a new report to any project. It is good practice to separate the reporting files from other files in your project. In this example, the supplied reports are stored in the `report` folder. When you use **File > New** and choose to create a new report, always use the Process Developer Template. The template has a script, which when the report is deployed, automatically detects the Data Source connection information by using the supplied `AeBirtContext` class.

If you are interested in this script, open a report that uses the template, select the Data Source, and then click on the *Script* tab in the report editor. The Data Source while in Process Developer does not use the server settings; instead, it is predefined to point to your embedded Process Server's server derby database. This database has a limitation that only one application can connect to it at a time, so if you are creating reports using derby make sure you stop the embedded Process Server first.

Note: If you need to change the default, edit the data source properties to point to the database in which you store your data.

Once you have created a new report, the next step is to define a data set. A data set is the SQL and parameters that return the results from the database. After creating the SELECT statement, you can simply drag the data set onto the canvas and a simple table is created for you. Now you can go ahead and make any of the formatting changes for your report that you need.

Charts and other reporting types are also available. To use these features, you will need to consult the BIRT documentation. A few chart samples are included to give you a sense of the capabilities. Please note that these reports were designed and tested against the embedded server's derby database. You may need to make modifications to the SQL Data Set for each report if you want to use them with another supported database.

Sample Reports

Report	Description
IndexedPropertyValues	Displays all the indexed values in the system grouped by property name. Additionally it can show an individual property's values using the filter.
ProcessDistribution	Displays a pie chart of the distribution of process executions by process deployment (plan I, process name). Additionally, it has a listing of the counts.
OpenTaskRoles	Displays the users and groups and their roles for open tasks.
ProcessResponseTimes	Displays a chart showing process response times.
ProcessStatesByDate	Displays a chart showing process states by date.
ProcessStatesByProcess	Displays a chart showing process states by process name.
ProcessList	Displays a list of running processes similar to the Console's Active Process List. Clicking on a Process ID link opens the process view in a new tab. **Note**: This sample also shows how to create links to the console process detail view page.

Deploying Reports

Reports are easily deployed simply by selecting them or the folder where they reside using the context menu to select **Export**. From within the displayed dialog, select the **Orchestration > Business Process Archive** export option. Select a location for your BPR file. Typically, you will also choose Web service deployment with the location of your server (the default is localhost on port 8080). Also, make sure to check the *Replace existing resources* checkbox or the system may not overwrite the previous report deployments unless you have set that as the default when setting server properties.

You can also choose to save the deployment script, which is an ant script that can deploy your reports again simply by selecting **Execute** from the context menu. Note the sample project has a saved deployment script in the deploy folder that you can use to automatically deploy the sample reports. Selecting the **Next** button brings you to a page where you will see your reports in the additional resources list. Choose a *Process Group* name for both your entries, for example `Sample Reports`.

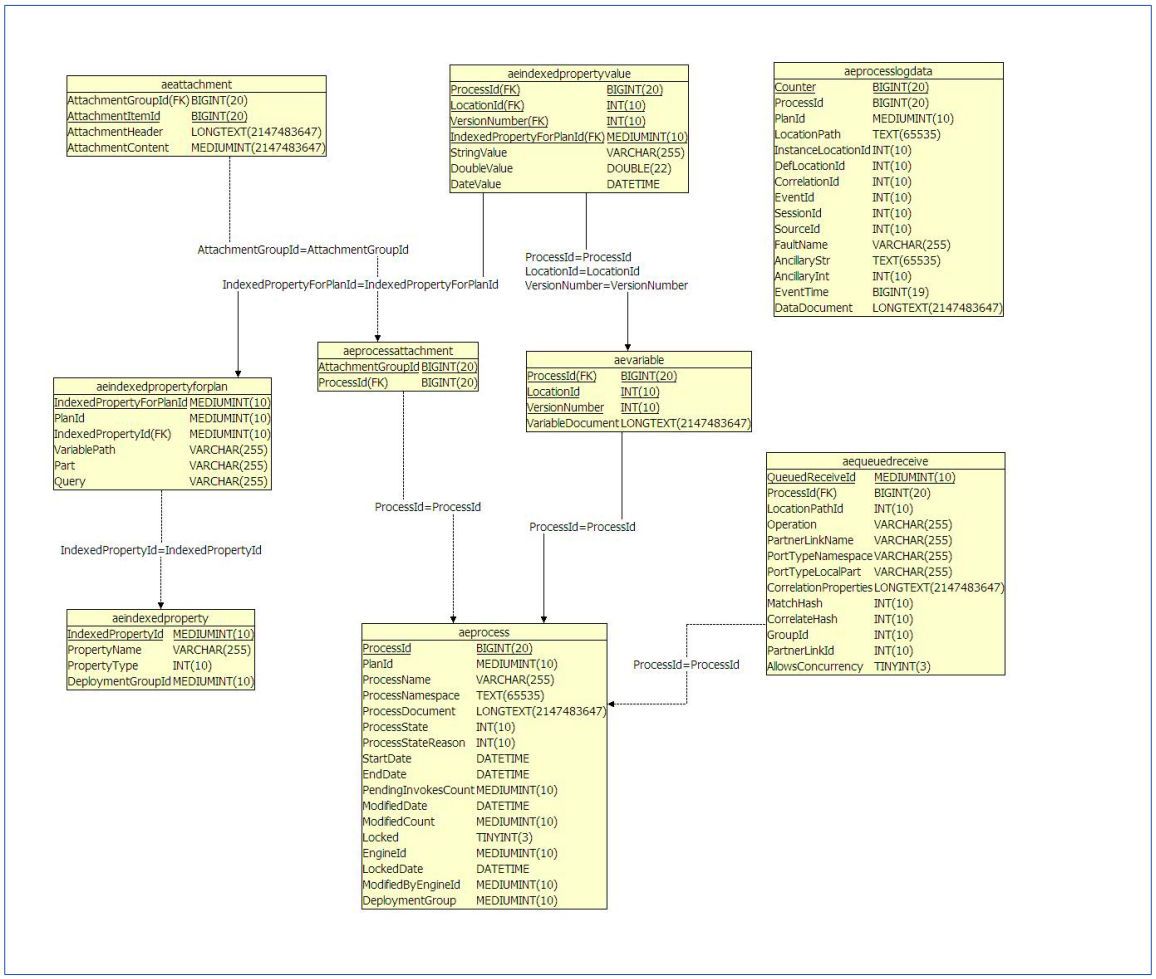
Process Data Model

Table 1. Table 1 - Process Developer Process Tables

Table	Description	Notes
AeUserProcessView(AeProcess)	Holds the main state of a process instance. The primary state of the process (for example, *running*, *started*, *completed*, and so on) is stored in the ProcessState column.	Actual BPEL language information and related services are stored in a related table, called AePlan.
AeVariable	This is the set of variable data associated with the process.	
AeIndexedPropertyValue	During deployment, you can choose to index fields from variables in your process. This table stores those values, which are typically your key performance indicators (KPI).	In order to find the property name associated with the value data, you need to link to AeIndexedPropertyForPlan (plan is the deployment information/table for a process definition) and the AeIndexedProperty tables.
AeProcessLogData	Contains the logging information associated with a process instance.	This is used to do root cause analysis (show process state at a point in time) of issues. It also contains useful reporting information.
AeProcessAttachment	Contains any attachments to variables in the process.	The actual attachment data is stored in the AeAttachment table as a BLOB.
AeAttachment	Stores the BLOB of an attachment.	
Others ...		

Process Data Model Entity Relation Diagram Subset

Table 2. Figure 1--ER Diagram for some of the process centric tables.



Human Task Data Model

In order to create human task-centric reports, you will need a basic understanding of the human task data model. The data model is of course centered on the task, which is in the `AeB4PTask` table. The primary key of this table is the `ProcessId`. Other tables have keys that relate back to the `ProcessId` in this table. As mentioned in the process section of the data model, a system process actually handles the task state management using the `AeB4PTask` table and the `AeProcess` table.

Table 2 contains a short list and descriptions of task tables for which you may have monitoring or other interests. Figure 2 contains a portion of the full data model that is centered on the `AeB4PTask` table.

Table 3. Table 2 - Human Task Tables

Table	Description	Notes
`AeB4PTask`	Holds the main state of a task instance. The primary state of the task (for example, *unclaimed*, *claimed*, *started*, *completed*, and so on) is stored in the State column see Table 4 below The `TaskType` column indicates whether the row contains a task or notification, (0=task, 1=notification).	Much of the task state interaction is accomplished through processes. Tochange anything in a task, you must use the WS-HT Service API. Most task access information is stored in `CLOB` columns in this table as organization entities. However, for quicker access when logging in, it is is also separated into its own table (`AeB4pTaskACL`).
`AeB4pTaskACL`	This table contains the access control list for the task. Type is zero for user and 1 for group. See Table 3 below for generic human role column values.	This table controls access to task information when using the WS-HT API. It controls what is visible to a user who is logged into the Process Central inbox, or another client application.
`AeB4PTaskPa` `AeB4PTaskEventDetail`	Contains all events that change the state of the task.	This is used to produce the history of a task.
`AeB4PTaskAttachments`	Contains the attachments associated with a task.	This is used to store attachments that are associated with a task and are typically available in the task detail display.
Others ...		

Table 4. Table 3--AeB4PTaskACL: GenericHumanRole Column

Role	Value	Description
Initiator	0	Task Initiator
Stakeholder	1	Task Stakeholder
Potential Owner	2	Potential Owner
Actual Owner	3	Actual Owner
Excluded Owner	4	An excluded owner
Business Administrator	5	Business Administrator
Notification recipient	6	Recipient of a notification task type

Table 5. Table 4--AeB4PTask: State Column

State	Value
Deferred	0

Unclaimed	1
Claimed	2
Started	3
Completed	4
Skipped	5
System Error	6
Faulted	7
Exited	8
Suspended	9

Table 6. Figure 2--Human Task ER Diagram Subset