Queensland University of Technology

Oracle Business Process Management

Tutorial Lab Project (INN696-1, Semester 1/2014)

Loan Assessment Process

Student: Hoang Huy Nguyen (Student No.: 8651523)
Supervisor: Professor Marcello La Rosa

Contents

1 INTRODUCTION 5

2 BACKGROUND OF ORACLE BPM 7

3 INSTALLATION OF ORACLE BPM 17

4 NAVIGATE JDEVELOPER ENVIRONMENT 21

5 CREATE A NEW BPM APPLICATION PROJECT 28

6 MODEL THE PROCESS 31

6.1 ADD PROCESS LANES 32
6.2 ADD MAIN ACTIVITIES AND SEQUENCE FLOWS 35
6.3 ADD ANCILLARY ACTIVITIES 39

7 IMPLEMENT THE PROCESS 41

7.1 INITIAL SET-UP 41
7.1.1 Server Connection 41
7.1.2 Email Notification 42
7.2 IMPLEMENT ORGANIZATION 49
7.3 IMPLEMENT DATA OBJECTS 51
7.4 IMPLEMENT HUMAN TASKS 60
7.4.1 Enter Application Form 61
7.4.2 Assess Credit History 86
7.4.3 Appraise Property 89
7.4.4 Assess Eligibility 91
7.4.5 Prepare and Send Acceptance Pack 92
7.4.6 Prepare and Send Home Insurance Quote 96
7.4.7 Verify Repayment Agreement 98
7.4.8 Take Final Decision 100
7.5 IMPLEMENT SCRIPT TASKS 102
7.5.1 Initialize Data 102
7.5.2 Update Rejected/Eligible status 104
7.5.3 Update Approval/Rejected loan status 106
7.6 IMPLEMENT SERVICE TASKS 107
7.7 IMPLEMENT BUSINESS RULES 114
7.8 IMPLEMENT TIMER 118
7.9 IMPLEMENT GATEWAYS 119
7.10 IMPLEMENT EMAILS 121
7.10.1 Reject Email – Illegibility 121
7.10.2 Cancel Application Email 123
7.10.3 Approval Email 124
7.10.4 Rejection Email 124

8 DEPLOY THE PROCESS 124

9 RUN THE PROCESS 130

9.1 PROCESS USER 130
9.2 ADMINISTRATOR 141

10 REFERENCES 143

11 APPENDIX 144

11.1 FURTHER IMPROVEMENTS 144
11.2 INITIALIZATION DATA 145
11.3 GENERATE PDF CODE 146
11.3.1 DownloadFileBean.java 146

11.3.2 PDFGeneratorBean.java 150
11.4 COMPLETENESS CHECKING CODE 152
11.5 COMMON ERRORS 153

ABBREVIATIONS

ADF	Application Development Framework
BAM	Business Activity Monitoring
BPM	Business Process Management
BPMN	Business Process Model and Notation
EJB	Enterprise Java Beans
JDK	Java Development Kit
JEE	Java Enterprise Edition
JSF	Java Server Faces
MDS	Metadata Services
OSB	Oracle Service Bus
SCA	Service Component Architecture
SOA	Service Oriented Architecture
UMS	User Messaging Service

1 Introduction

This project is to deploy Oracle BPM environment on a lab environment and demonstrate business process implementation for a Loan Application Assessment process. This process is based on *Fundamentals of Business Process Management* book by Dumas, M., La Rosa, M., Mendling, J., Reijers, H.A. The output is also a complete tutorial of the implementation details which can be used by future students who are interested in this technology.

Below is a summary of the business process.

The process is initiated when a loan applicant submits an application form. A loan officer will receive the application form and check for its completeness, either manually or automatically with a system automatic check. If the application is not complete, it is sent back to the applicant for supplement. In case the application is not completed by the applicant within 5 days, the system will terminate the process. Once it is complete, it will be forwarded to a financial officer and a property appraiser for assessment. The financial officer checks credit history of the applicant while the property appraiser evaluates the value of the property. The financial officer gives the application a credit grade (A, AA, B, BB, etc.) which will be translated into a rating mark by the system. Once both financial and property assessment are done, the application is transferred to a loan officer who will assess its overall eligibility. If it is not eligible, the process sends email notification to the applicant and stops there. If it is marked as eligible, the loan officer will then prepare an application pack which includes a repayment agreement with detailed terms/conditions and send to the applicant. In the next step, if the application includes a request for home insurance, it will be forwarded to an insurance sales representative (ISR) who will provide a quote for the home insurance. Once the loan officer receives the repayment agreement returned from the applicant (via email, hard-copy) and the home insurance quote from the ISR (if applicable), the loan officer will verify the repayment agreement and forward all information to another loan officer for the final decision. In case the repayment agreement verification is pending for more than 14 days, the system will terminate the process. In the final step, the loan officer makes a decision to approve or reject the application. In either cases, an email notification will be sent to the applicant for their information and the process stops.

In this tutorial, we will implement the above process with Oracle SOA/BPM using key components such as Oracle BPMN, BPEL, Business Rules, Human Workflow, and Oracle ADF. System installation for Oracle SOA/BPM server and Oracle JDeveloper must be in place (installation is guided in this paper).

It is easier for readers who want to exercise this tutorial if they are equipped with basic knowledge and skills of BPMN, BPEL, Java programming, XML, SOA, Web services and Java servlet/JSF technologies.

The tutorial is structured into the following sections.

Background of Oracle BPM	Introduces underlying Oracle BPM concepts and technologies used in this tutorial
Install Oracle SOA/BPM	Outlines main installation steps along with shared experience to ensure a smooth system set up. Readers should still refer to Oracle installation document for technical instructions
Navigate Oracle JDeveloper environment	Explains essential design/development editors used in Oracle JDeveloper environment facilitate readers with this complex environment
Create a new BPM application project	Step by step guide on how to open a BPM application project from scratch

Model the Process	Step by step guide on how to model the business process scenario in BPMN notation with Oracle BPM studio
Implement the Process	Step by step instructions on how to implement the process with Human task, rules, server script, timer event, web forms, etc.
Deploy the Process	Provides guide on how to deploy the process to SOA/BPM servers
Run the Process	Demonstrates a process run as a process user and track the process trace as a process administrator

In addition, the Appendix section contains source codes available to be pasted during the tutorial, suggestion for technical enhancement after this tutorial, and a list of common errors likely encountered by newcomers to this massive technologies.

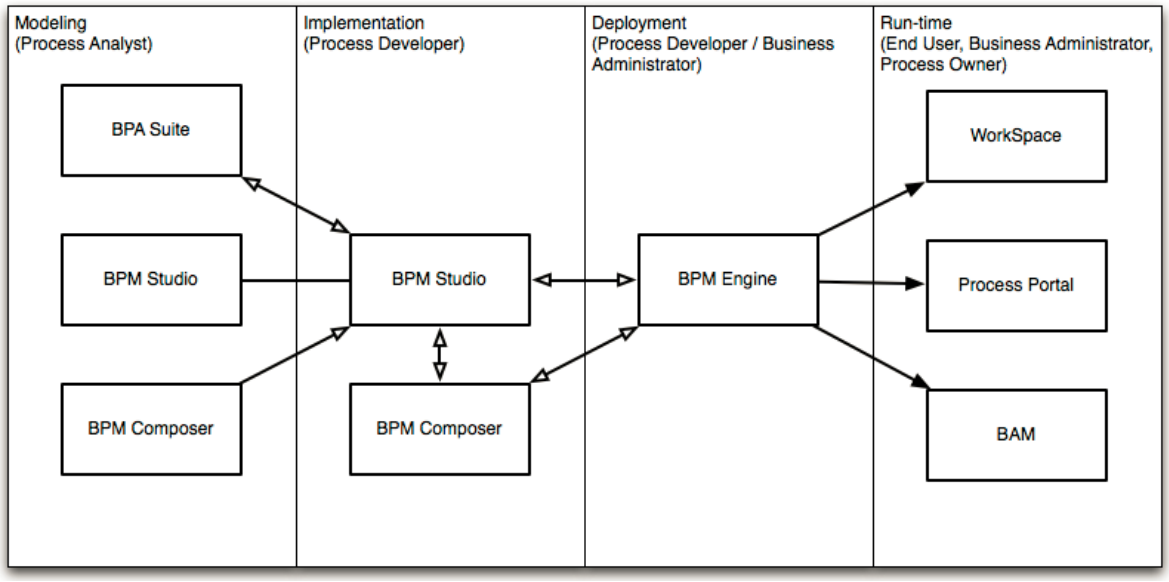
2 Background of Oracle BPM

This section is to equip learners with key components of an Oracle BPM application and architectural concepts. This knowledge is essential for understanding the tutorial practice in section 6 onwards which is only dedicated to procedural steps.

Oracle offers a comprehensive BPM environments based on Service Component Architecture. We can participate into this environment with different roles: business analyst, BPM developer, system administrator or process user. In this tutorial, we will involve in all of these roles through our hands-on practice; therefore this section would attempt to cover a quite complete overview picture.

Oracle BPM Application Development Lifecycle
(http://docs.oracle.com/cd/E23943_01/user.1111/e15175/bpmug_intro_bpm_suite.htm)

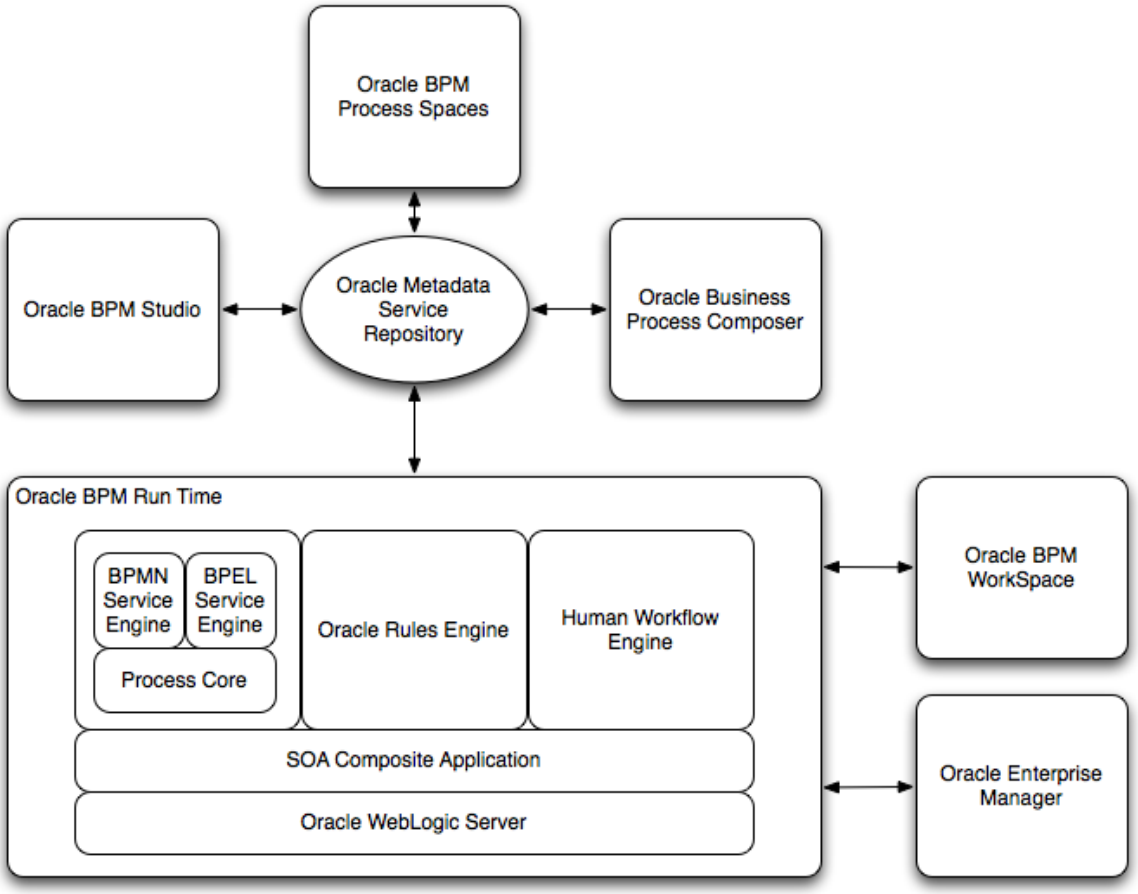
The stages of the development life cycle of an Oracle BPM application are outlined below.



Oracle BPM Architecture

(http://docs.oracle.com/cd/E23943_01/user.1111/e15175/bpmug_intro_bpm_suite.htm)

The core of Oracle BPM product is Oracle BPM Run-Time as shown below.



Oracle BPM Runtime

Oracle BPM Runtime or "service infrastructure" provides the internal message transport infrastructure for connecting components and enabling data flow. The service infrastructure is responsible for routing messages along the wire connections between services, service components, and references.

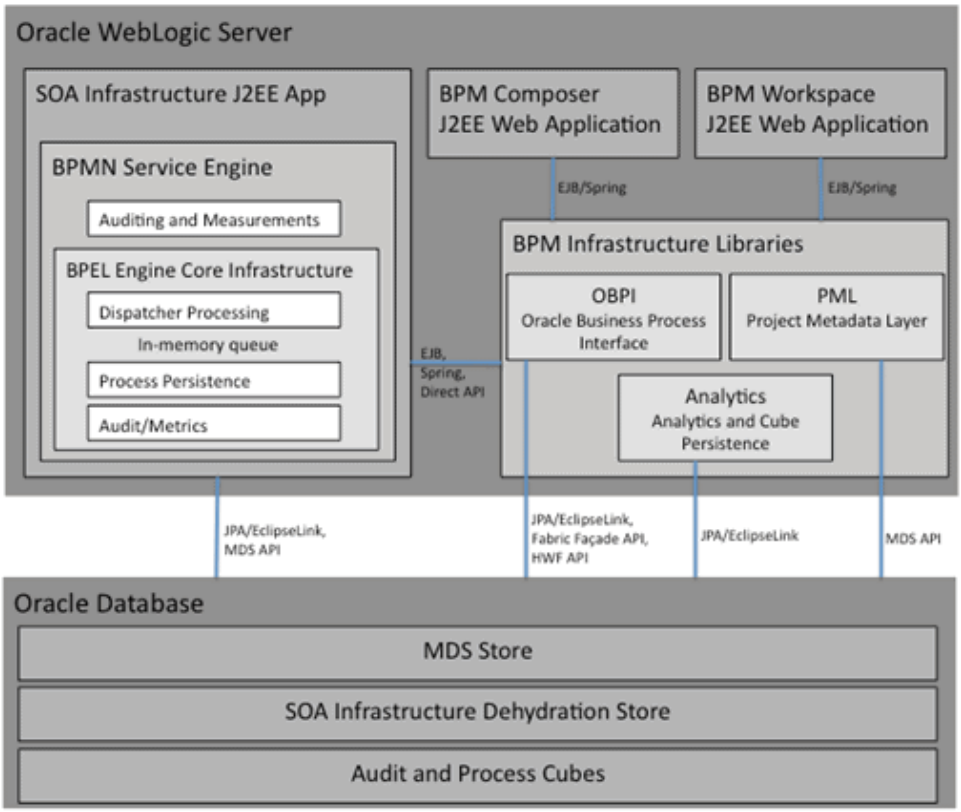
BPEL Service Engine

For process orchestration of synchronous and asynchronous BPEL processes

BPMN Service Engine

For creating and modeling business processes using Business Process Management Notation and Modeling (BPMN). The BPEL, BPMN and Process Core (provided shared services for both BPMN and BPEL engines) are commonly called as BPM engine.

Since we are going to work extensively with BPMN Engine, let's take a deeper look into the figure below.



Notably, according to Oracle documentation, BPMN Service Engine is actually an extension of the existing BPEL Service Engine and as such it leverages the core infrastructure of the BPEL. The BPMN Service Engine leverages JPA/EclipseLink to store/recover the state of a process instance in the SOA Infrastructure dehydration store maintained by a database and to persist audit records that are created in the course of running a process. MDS APIs are used to retrieve metadata Information about the BPMN Process Model and other BPM project artifacts like the Business Catalog.

Business Rules Engine

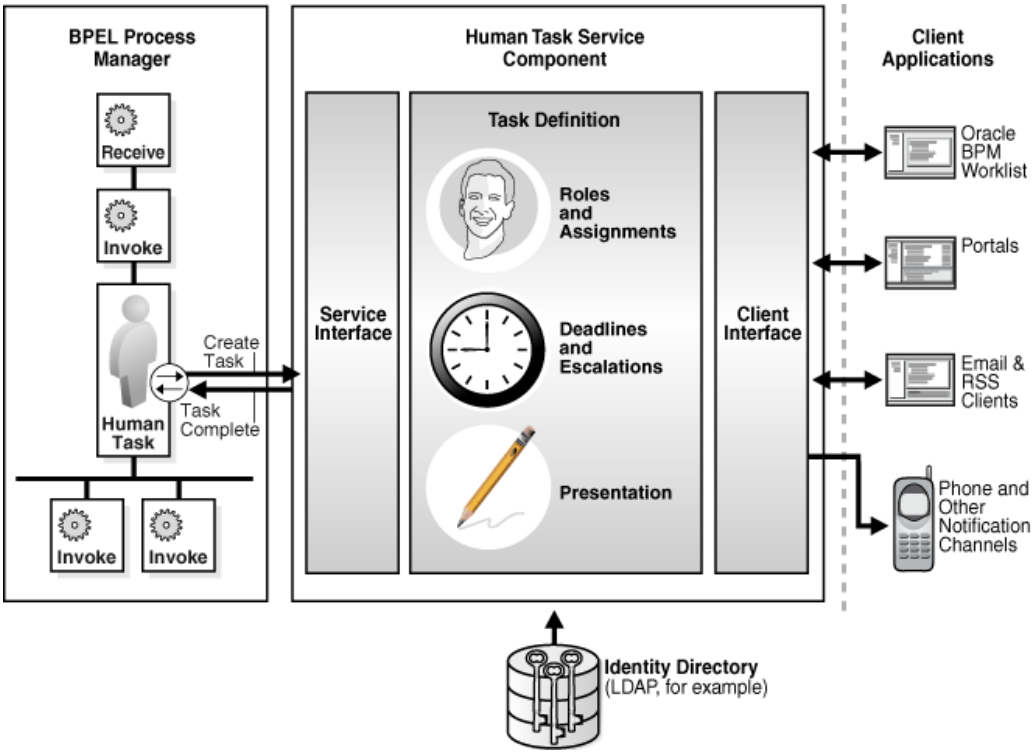
For making a decision or for processing based on business rules.

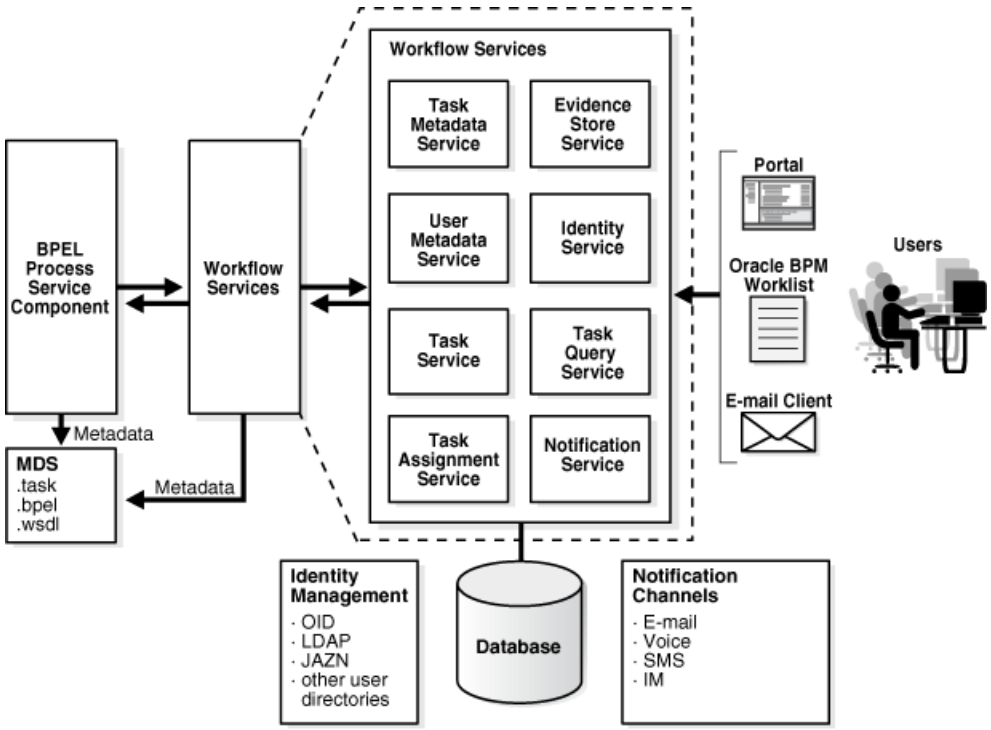
Human Workflow Engine

For modeling a human task (for example, manual order approval) that describes the tasks for users or groups to perform as part of an end-to-end business process flow.

This engine is also used extensively in this tutorial. The details of its services are shown below.

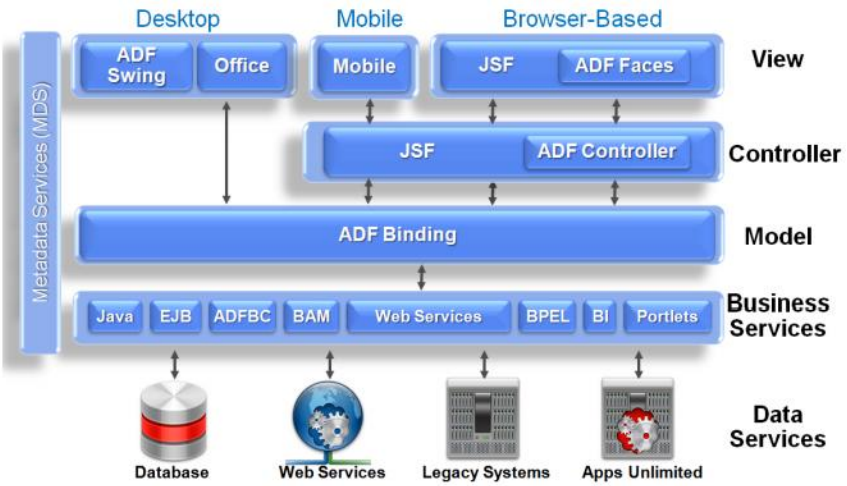
We can see that the Human Task Workflow Engine are a major set of SOA components to manage a wide range of human related services, including routing, roles and permission, security, timers, interaction with user interface, and notification.





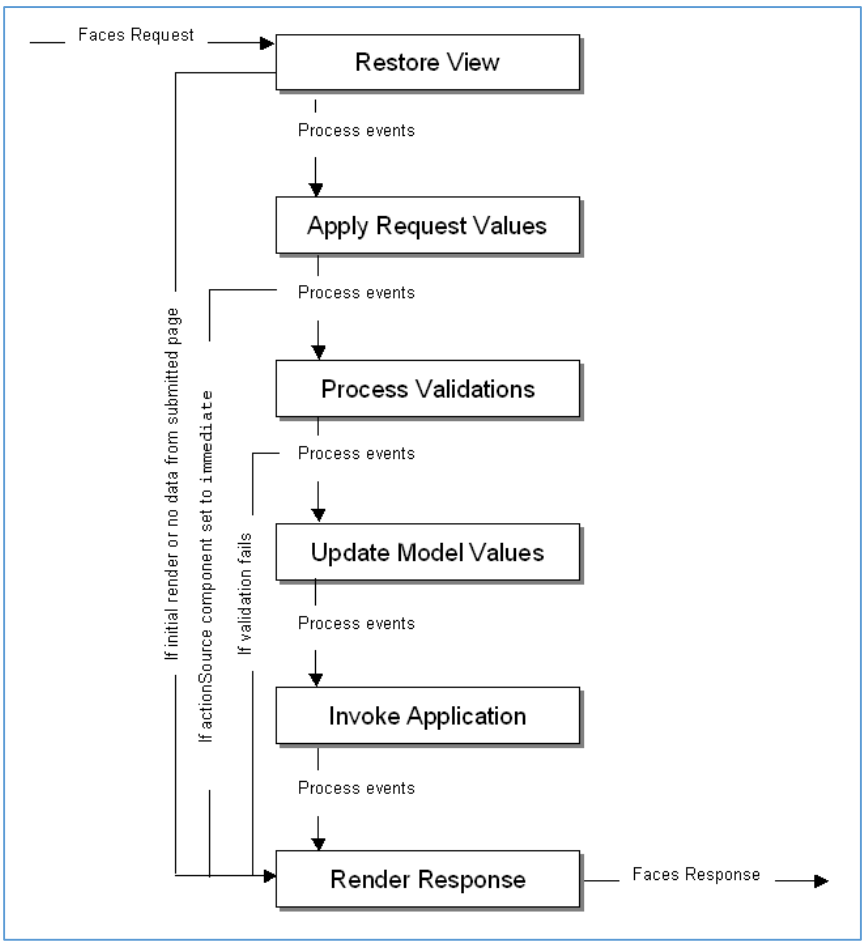
User Interface and Data Control

Another important aspect of BPM application is the interaction with users which is implemented by Oracle ADF. Oracle ADF runs on top of Java Server Faces (JSF) which abstracts HTML and Java Servlet technologies with web components.



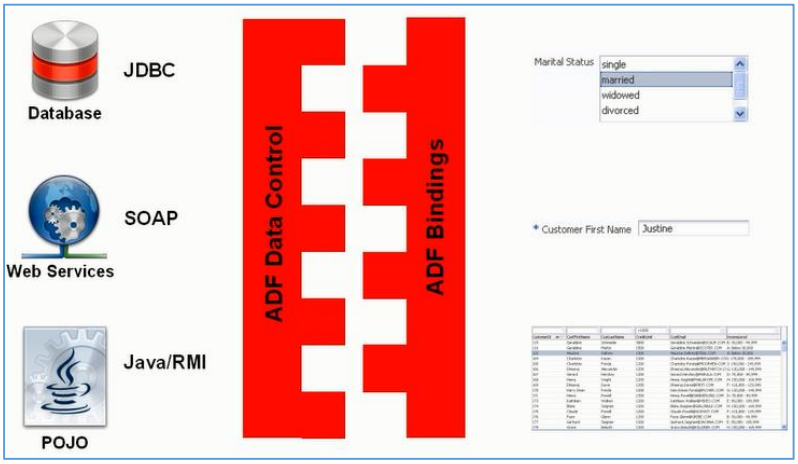
In this tutorial, we are going to implement a number of web pages in Oracle ADF. The most common functions of Oracle ADF will be introduced here to facilitate our practice in the tutorial. Many other features of this complex framework will not be covered though since it beyond the scale of this tutorial. The tutorial user should refer to other materials available for Oracle ADF for a more solid understanding.

The figure below depicts main phases of ADF Face Request. The key point to us is the "**Invoke Application**" phase where we can add our custom codes. We will demonstrate this feature for PDF Generation.



Oracle ADF added the following new features to JSF:

- A wide range of new Ajax-based UI faces (components)
- Data bindings
- Data controls



The prominent contribution of Oracle ADF is the ADF Model layer which includes Data Control and Bindings as shown above. Other mechanism is primarily based on JSF infrastructure. The main idea of data control and data bindings is to simplify the process of binding UI elements with various data source in a declarative manner. Developers don't need to bother with details of various data sources but only work with a common facilities provided by the Data Controls editor in Oracle JDeveloper. We will work with this facility during the human task implementation.

For BPM technology, Oracle provides a special data control called BPM data control which interacts with both BPMN and Human Workflow Engine. It hides the complexity in dealing with these two engines and ease the development efforts.

However, it is noted that Oracle ADF is a huge and complex technology stack on top of JSF, which requires a relatively steep learning curve.

Other technologies

In addition, Oracle provides other BPM related environments but not in scope of this tutorial:

- Business Activity Monitoring
- Business Process Composer
- Oracle Service Bus
- Oracle WebCenter.

What is Oracle BPM Application?

Now, we'll narrow down to the key understanding of Oracle BPM application or process oriented application in terms of Oracle technologies.

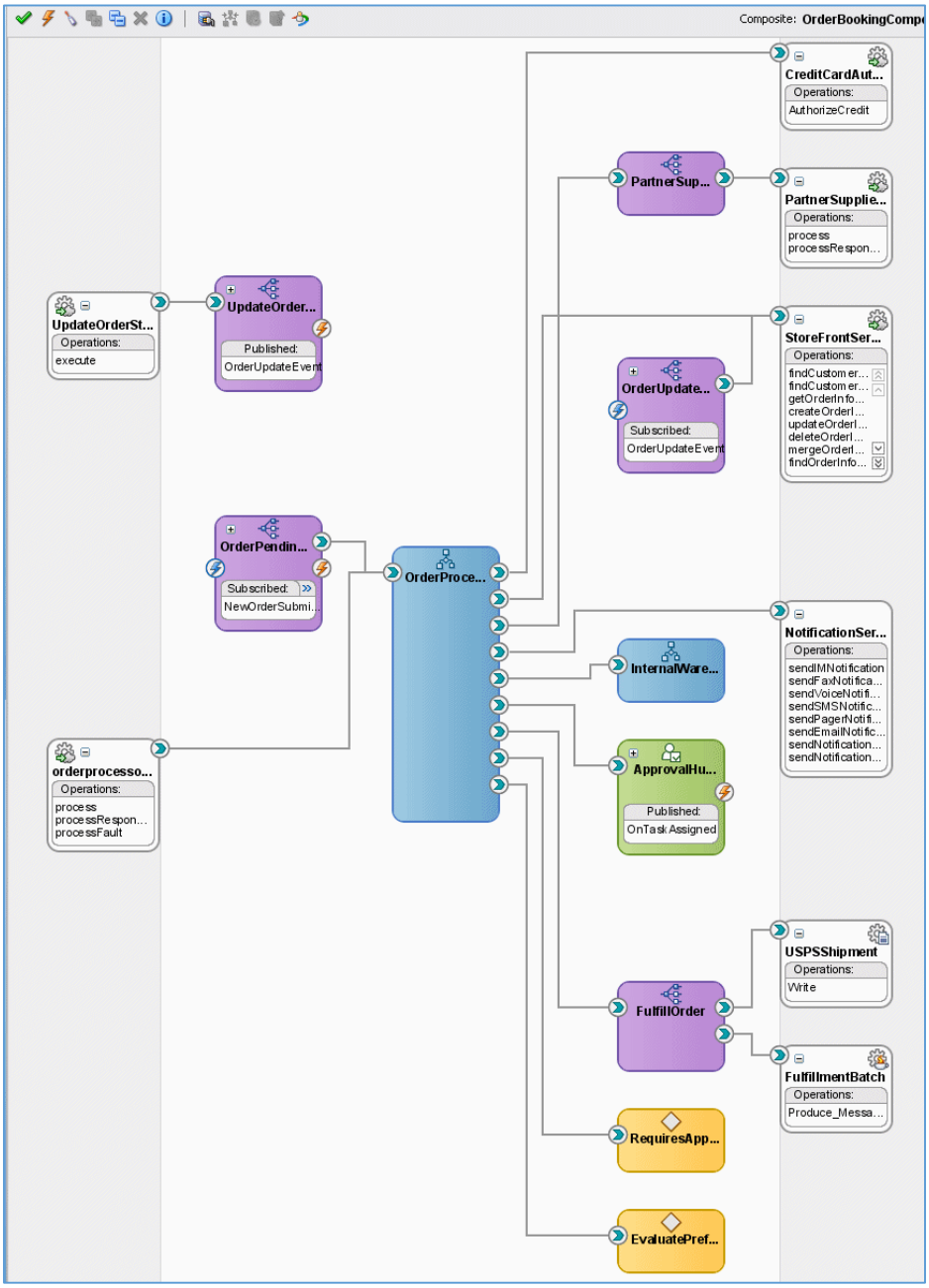
Two technologies implemented long before BPMN engine are SOA and BPEL. A BPM project is the basic unit of deployment to the BPM runtime. A BPM project is comprised of service components (such as BPMN Process, Business Rules, and Human Workflow) and references organized as a SOA Composite, organizational data (such as Roles and Business Calendars), and business indicator metadata and dashboard data. Components are targeted to service engines during deployment while services and references are enabled using the binding components. The metadata for organizational data, business indicators and dashboards is persisted and evaluated by appropriate components at runtime. At runtime, messages are received by the binding component or the BPM Workspace and are then routed to the appropriate service engine(s) by the Service Infrastructure.

The key understanding must be noted here is that every BPM application is actually a SOA Composite application in Oracle BPM implementation.

SOA composite applications such as those shown in the figure below consist of service components. Note that each component is web application created by Oracle using JSP Servlet, EJB and JMS etc. These service components are installed as JEE application (http://docs.oracle.com/cd/E28280_01/admin.1111/e10226/soasuite_intro.htm#CEGCCAEB).

These components are assembled into a single SOA composite application. They are the basic building blocks of SOA composite applications to implement a part of the overall business logic of the SOA composite application.

The figure below shows an example of a SOA composite application with various components: BPMN process, BPEL process, Human Tasks, Decision service (business rules) and Bindings.



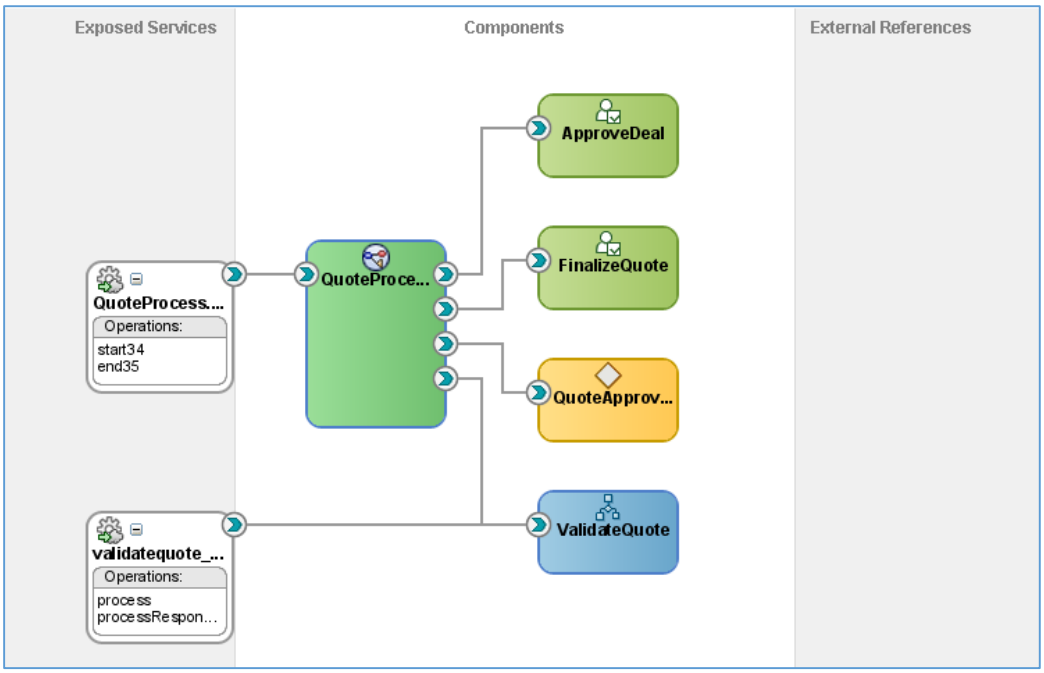
An Oracle SOA composite application can consist of a variety of service components, binding components, and services.

- BPEL processes
- BPMN processes (if Oracle BPM Suite is installed)
- Human workflows
- Oracle Mediator
- Decision services (Oracle Business Rules)
- Spring
- JCA adapters
- HTTP binding
- EJB service
- Direct binding service
- Oracle Application Development Framework (ADF) Business Component service
- Oracle BAM
- Oracle B2B
- Oracle Healthcare
- Business events
- Oracle User Messaging Service

Note that Oracle SOA Composite Application complies with Service Component Architecture standard. Therefore, those interested can read more details in the SCA specification at http://www.oasis-opencsa.org/sca.

Thus, from a process user perspective, the application is an automation of a process model. However, from implementation insight, an Oracle BPM application is eventually a SOA Composite application with added components such as: BPMN Engine, BPM Workspace, BPM Process Composer. (http://docs.oracle.com/cd/E25054_01/doc.1111/e15176/soa_composite_bpmpd.htm)

For example, the figure below shows a composite view of a BPMN diagram in Oracle JDeveloper. In our tutorial, we will demonstrate a similar composite view of BPMN diagram. It means that BPMN is an composite component which makes calls to other composite components through web services.



Thus, implementing a BPM process in Oracle BPM environment is actually creating an SOA composite application with BPMN process as a main component. It also shows that Oracle has a different approach

from some other BPMS vendors (such as IBM and Bizaghi) when Oracle exposes many technical components in an overall SOA composite picture for developers to select and build a BPM oriented application, rather than package a dedicated solution for BPM application approach.

How an Oracle BPM application is executed at run-time

When we run a BPM project, the SOA engine creates a SOA composite instance which is identified by a unique instance ID. The SOA composite instance contains an instance of each of the components defined in the SOA composite. For example, if our SOA composite defines a BPMN process and a human Task, then the SOA composite instance contains a BPMN process instance and a human task instance

When the Oracle SOA Service Infrastructure application starts, it initializes all the service engines (BPMN, BPEL, Human Task, etc.) and loads the composites from the MDS repository. If the composite contains any BPMN processes, it targets those individual components to the BPMN service engine, similarly for BPEL, Human Task and others. Once the process is loaded and its BPM specific metadata persisted in the database, the system is available to receive requests. (According to http://docs.oracle.com/cd/E14571_01/core.1111/e10106/ha_soa.htm)

A detailed startup and shutdown lifecycle of BPMN process is as follows:

1. Start BPM Server.
2. Start BPMN service engine.
3. BPM project composites are loaded from the MDS repository by the SOA Service Infrastructure.
4. BPMN components are dispatched to the BPMN service engine to be loaded.
5. The BPM project metadata, such as organization data and audit/measurement metadata, is persisted in the infrastructure database.
6. Composite binding components are activated.
7. The BPMN engine services requests (more details below)
8. The shutdown signal is received by the SOA Service Infrastructure.
9. The SOA Service Infrastructure starts unloading composites.
10. Composite binding components are disabled.
11. BPMN components are dispatched to the BPMN engine to be unloaded.
12. The BPMN service engine shuts down.

Step 7 is elaborated as follows:

1. There are two ways to initiate a BPMN process instance, either using Initiate Task via BPM Workspace or from a Web Service client. In the former, a user would log into BPM Workspace and click on a visible process link that they are granted to start a process instance.

2. The BPMN Service Engine starts executing the process. The process activity may initiate different types of service components included in the process execution: BPEL process, web services, human task, etc.

3. If it is a BPEL process, the process will transfer control to BPEL process engine.

4. If it is a User Task, the BPMN Service Engine creates a user task and transfer control to Human Workflow Engine. After the human workflow is complete, control is passed back the BPMN engine. Any required data objects are passed back to the user task. Notably human tasks are independent from BPMN processes. Therefore, if we terminate a BPMN process while it runs

a user task, the associated human tasks keeps running independently. Normally human task will interact with human users via web pages, mobile devices or desktop application. Oracle ADF is used to implement this interaction. ADF is an extension of Java Server Faces which in turn runs on java servlet technology. At this step, a HTTP request for ADF Face resource will be sent to web server which initiates ADF request processing lifecycle.

5. If it is a server script, it will be executed directly in BPMN engine

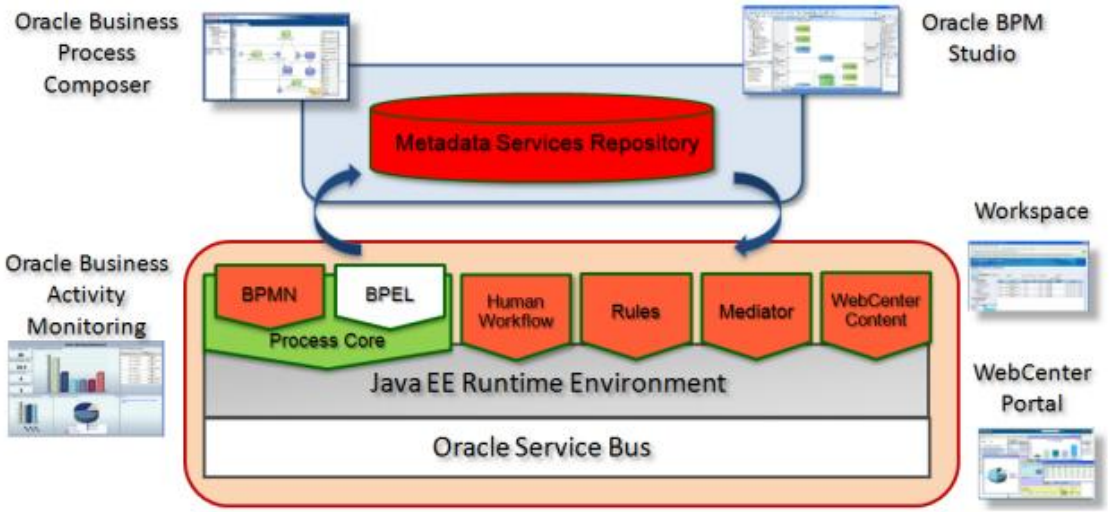
6. If it is a web service or adapter, BPMN engine will call to SOA Infrastructure to execute external web service.

7. If it is a business rule, BPMN engine will transfer control to Business Rules Engine.

8. If it is an email task, BPMN engine will transfer it to Human Workflow Engine.

Tooling view

The Oracle tools provided to design and implement a BPM application are diversified, as shown below.



Oracle BPM Studio

This component is implemented in Oracle JDeveloper and used as integrated environment for modelling business processes (BPM analyst) as well as implementing the process flow (BPM developer).

Oracle Business Process Composer

This is a lightweight tool which allows business analysts or power users to customize the process within their permissions without having to redevelop or redeploy the process.

Oracle BPM Workspace

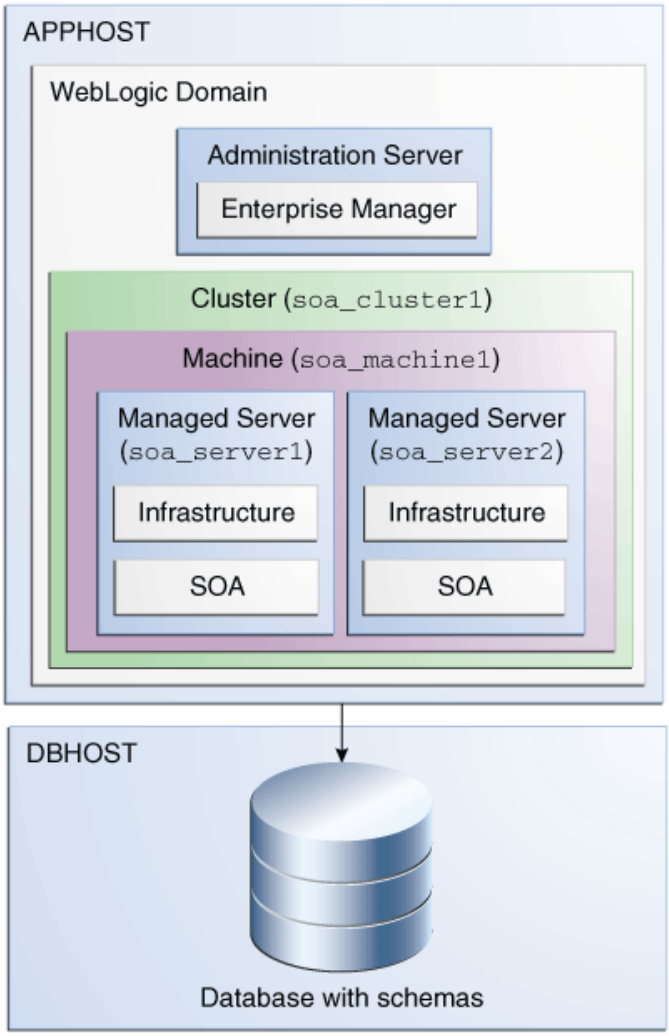
This is a collaborative work list for process users and owners to run a process instance. They can also monitor the process performance, track an instance progress.

Oracle Enterprise Manager

This is an administrative console for administrator to manage SOA Infrastructure and various SOA composite components, including BPMN engine, BPEL engine, Human Workflow engine, Business Rules, Mediator engine and notification.

Deployment Architecture

The deployment of Oracle BPM on lab environment follows the topology shown below which covers the key products including Oracle BPM, BAM, MDS and Oracle Service Bus.



The installation guideline in section 3 will explain further details of the logical organization of Oracle BPM deployment topology above.

3 Installation of Oracle BPM

This section outlines main installation steps for Oracle BPM/SOA version 11.1.1.7.0. It won't contain all step-by-step instructions but readers should follow Oracle online installation document at the following link:
http://www.oracle.com/technetwork/middleware/soasuite/overview/quickstartguidesoasuite11gr1ps4-459545.pdf

It is noted that Oracle installation manual is well-explained. This section thus only highlights critical steps and some important preparations based on practical experience to ensure a smooth process with less frustration.

Note that there may be multiple versions of Oracle SOA/BPM Suite available for download, but this paper is for Oracle SOA 11g and the version number is 11.1.1.7.

Hardware configuration

Oracle BPM/SOA is a large-scale installation which consume huge capacity of hard disk, RAM and CPU. My personal experience showed that

Software requirements

We have installed Oracle BPM successfully on the systems

- Windows 7 Professional or Windows 2008 Server
- Oracle Database 11g Release 2
- Internet Explorer 8 and 9
- Java Development Kit 7 64-bit server or IBM JRockit 64-bit server

There may be some confusion in terms of Oracle BPM Suite and Oracle SOA Suite packages for download. Oracle BPM Suite is actually Oracle SOA Suite with added BPM components. Oracle does not publish on their web site a separate product for BPM Suite 11g, but they provide SOA Suite 11g package and some BPM-related components to run on the SOA platform. Therefore, the installation process includes SOA Suite installation as a major set up work and some BPM components.

Access privileges for installation

The logged in user for Oracle SOA Suite installation should be the default Administrator of Windows, otherwise there may be numerous error messages relating to lack of system rights. This is because the installation script requires some critical system privileges in it chain of commands. Although you can login as a user with granted Administrator rights (belongs to Administrators group) and/or trigger the installation script with "Run as administrator" option, the chain of commands called by the script may lose this access identity during its execution and encounter privilege lacking errors during the process.

Again, it is vital to run Oracle Weblogic and Oracle SOA Suite installation script by logging in Windows as default **Administrator** user. It does not mean we have to always log in as Administrator user to be able to start Oracle BPM/SOA server after installation. We can do it with a different user and "Run as administrator" option.

Installation Parameters

It is a good practice to keep a note of all parameters and options set throughout the installation process. You will definitely need some of them later on or get frustrated because of not having them in hands. For example, they should include: installation folder for every component, port number, version number, domain name, server name in use, chosen username and password, service name, system ID, database name, and many others.

Now, once we are aware of the pre-installation notice, the following sections will explain the main installation steps to help you feel more confident with Oracle installation manual. You should relate the steps here with the architecture background presented in section 2.

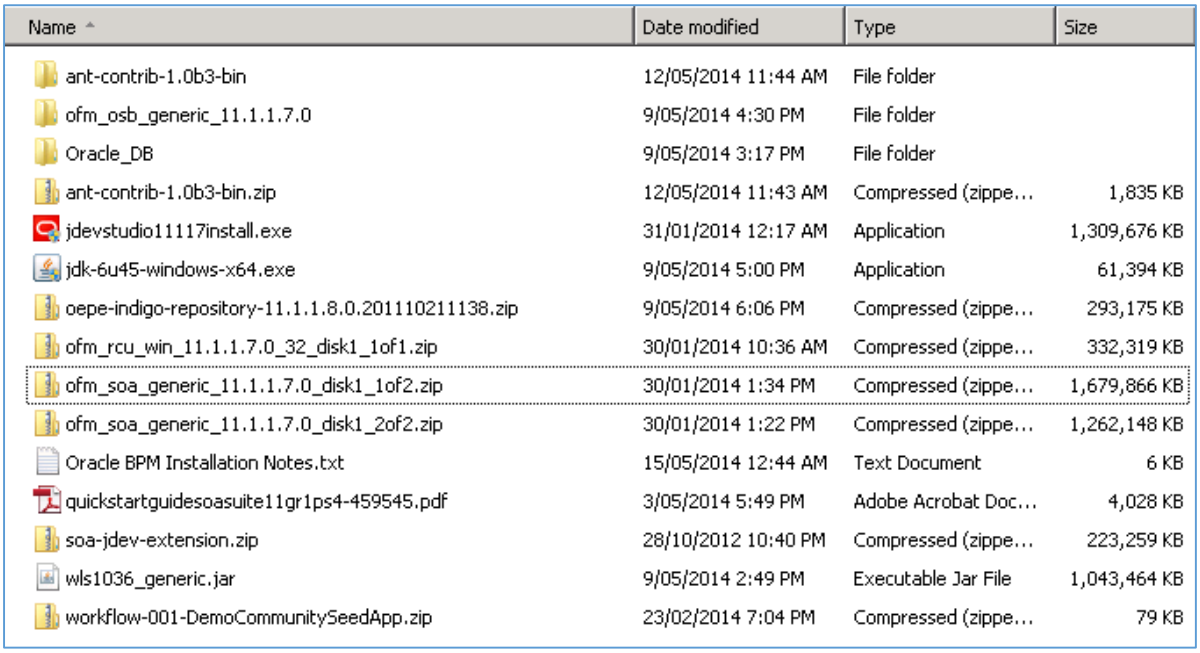
Step 1: Download all necessary packages for installation

Oracle SOA Suite can be downloaded from the link below.

http://www.oracle.com/technetwork/middleware/soasuite/downloads/soasuite11gdownload-2210918.html

JDK and IBM JRocket can be download from Oracle JDK web site.

The downloaded files can be saved into one folder as shown below.



Name	Date modified	Type	Size
ant-contrib-1.0b3-bin	12/05/2014 11:44 AM	File folder	
ofm_osb_generic_11.1.1.7.0	9/05/2014 4:30 PM	File folder	
Oracle_DB	9/05/2014 3:17 PM	File folder	
ant-contrib-1.0b3-bin.zip	12/05/2014 11:43 AM	Compressed (zippe...	1,835 KB
jdevstudio11117install.exe	31/01/2014 12:17 AM	Application	1,309,676 KB
jdk-6u45-windows-x64.exe	9/05/2014 5:00 PM	Application	61,394 KB
oepe-indigo-repository-11.1.1.8.0.201110211138.zip	9/05/2014 6:06 PM	Compressed (zippe...	293,175 KB
ofm_rcu_win_11.1.1.7.0_32_disk1_1of1.zip	30/01/2014 10:36 AM	Compressed (zippe...	332,319 KB
ofm_soa_generic_11.1.1.7.0_disk1_1of2.zip	30/01/2014 1:34 PM	Compressed (zippe...	1,679,866 KB
ofm_soa_generic_11.1.1.7.0_disk1_2of2.zip	30/01/2014 1:22 PM	Compressed (zippe...	1,262,148 KB
Oracle BPM Installation Notes.txt	15/05/2014 12:44 AM	Text Document	6 KB
quickstartguidesoasuite11gr1ps4-459545.pdf	3/05/2014 5:49 PM	Adobe Acrobat Doc...	4,028 KB
soa-jdev-extension.zip	28/10/2012 10:40 PM	Compressed (zippe...	223,259 KB
wls1036_generic.jar	9/05/2014 2:49 PM	Executable Jar File	1,043,464 KB
workflow-001-DemoCommunitySeedApp.zip	23/02/2014 7:04 PM	Compressed (zippe...	79 KB

It is important to make sure all packages to install are of the same version. For example, the above components have the same version number as 11.1.1.7. Incompatible components might not work with each other, for example JDeveloper 11.1.1.6 may have issues to run smoothly with SOA 11.1.1.7.

The key packages in the installation folder are summarized in the following table.

	Installation	**Installation files**
1	SOA Suite	Ofm_soa_generic_11.1.17.0_disk1_1of2.zip Ofm_soa_generic_11.1.17.0_disk1_2of2.zip
2	Repository Creation Utility	Ofm_rcu_win_11.1.1.7.0_32_disk1_1of1.zip
3	JDeveloper	jdevstudio11117install.exe
4	Oracle Weblogic Server	wls1036_generic.jar
5	Oracle OSB (optional for this tutorial)	Ofm_osb_generic_11.1.1.7.0
6	JDK Server edition 64 bit	Jdk-6u45-windowx-x64.exe
7	Oracle Database Server 11g R2	Oracle DB
8	Demo Community (to create demo users)	Workflow-001-DemoCommunitySeedApp.zip

These components will be explained in the next steps.

Step 2: Install Oracle Database 11g R2

Oracle Database server is needed to store schema for Oracle BPM/SOA, meaning they are server application with their own database. Thus they need a DBMS for data management.

Follow standard installation for Oracle Database Server.

Note that the character set for database should be chosen as Unicode UTF8 since it is required by Oracle SOA/BPM schema.

Step 3: Install JDK

The JDK should be 64-bit and server edition to be able to accelerate the performance of BPM/SOA engines by taking advantage of more than 4GB RAM space.

It is recommended to install Oracle JDK 6 or IBM JRockit, both should be 64-bit server edition. There might be compatibility issues with different Java versions.

After JDK installation, make sure you can set up the default java home on your system to the new java installation folder because the default call “java” in installation script may point to a different java installation on your system.

Step 4: Install Weblogic Server

Oracle SOA/BPM is JEE application to be run within a JEE container. Oracle Weblogic server is used as a JEE container.

The installation process for Oracle Weblogic server is normally standard and smooth.

Step 5: Install Database Schema

Note that Oracle SOA/BPM is actually a JEE server application and it uses database to store it design-time and run-time data. The data can be process instances, process runtime data, configuration parameters, audit data, business activity monitoring data, etc.

In particular, it is noted that a MDS database must be installed which is used as a common data store for process data in Oracle SOA platform.

The Repository Creation Utility (RCU) is used for database installation. A database server must have been installed and a database server instance has been started before running this step. Also make sure that you can provide the right database connection parameters when asked: server name, instance id, username.

Step 6: Install Oracle SOA

Once the Weblogic server and database tables have been in place, we now can deploy Oracle SOA as a JEE application to Weblogic Server in this step. It is a large-scale server application including SOA Infrastructure, BPEL engine, BPMN engine, Human Workflow Engine, Business Rules Engine, etc., including the console applications used to administer these engines (Enterprise Manager).

This is often referred to as BPM run-time architecture.

Step 7: Create SOA Domain

Referring to the deployment architecture in section 2, Oracle SOA manages its infrastructure logically with host, domain, clusters and managed server concepts. This step is to generate these administration data in the management database of Oracle SOA. There is no software to install in this step but only configuration data to create on the basis of web logic server, database server and SOA server application.

You select a specific way you want SOA server to organize BPM/SOA applications in the future: by domain, cluster, standalone server or managed servers, etc.

Step 8: Install JDeveloper

After the SOA infrastructure has been established, now it’s time to install Oracle JDeveloper as a design and development environment.

Noted that JDeveloper is offered by Oracle as a sole tool for all types of Java-based development projects, not only BPM application.

For BPM application development, you should add some design components: SOA Composite editor and BPMN editor. Otherwise, you will not find out where to design the BPMN process model and links to key process-related concepts: business objects, human resources, human tasks, rules, etc.

Step 9: Create Demo Users

This step is to create some demo users for our tutorial. Oracle BPM allows us to define logical roles in a process. At run-time, these roles are translated to real users on the system by connecting to a LDAP server provided with Oracle Weblogic Server. This role-user mapping can be created in the process at design-time.

In order to have users available for the design-time mapping, Oracle provides a script to add a list of users of a fictitious company to Weblogic Server.

The script can be found at the following link.

http://www.oracle.com/webfolder/technetwork/tutorials/obe/fmw/obpm/11g/r1/install/files/DemoCommunitySeedApp.zip]DemoCommunitySeedApp.zip

You download the script and follow the instructions to install users.

Management console and tools

After successful installation, you can access Oracle SOA/BPM consoles and tools as follows.

Oracle Enterprise Manager

http://localhost:7001/em

Oracle BPM Workspace

http://localhost:8001/bpm/workspace

Oracle BPM Composer

http://localhost:8001/bpm/composer

Oracle Business Activity Monitoring

http://localhost:9001/OracleBAM/

Oracle Service Bus

http://localhost:7001/sbconsole

Oracle Admin Server Weblogic Administration

http://wflab02.qut.edu.au:7001/console

Oracle WebLogic for each server: run from the Enterprise Manager console.

4 Navigate JDeveloper environment

Oracle JDeveloper is designed as an integrated development environment for various types of projects: Java desktop, web, mobile, BPM project, etc. It is claimed as a strength of Oracle BPM environment when different roles (business analysts, developers, designers and others) can use one tool only for their duties. However, due to its richness, it turns out to be highly complicated tool to quickly grasp and easily become overwhelming to a new user.

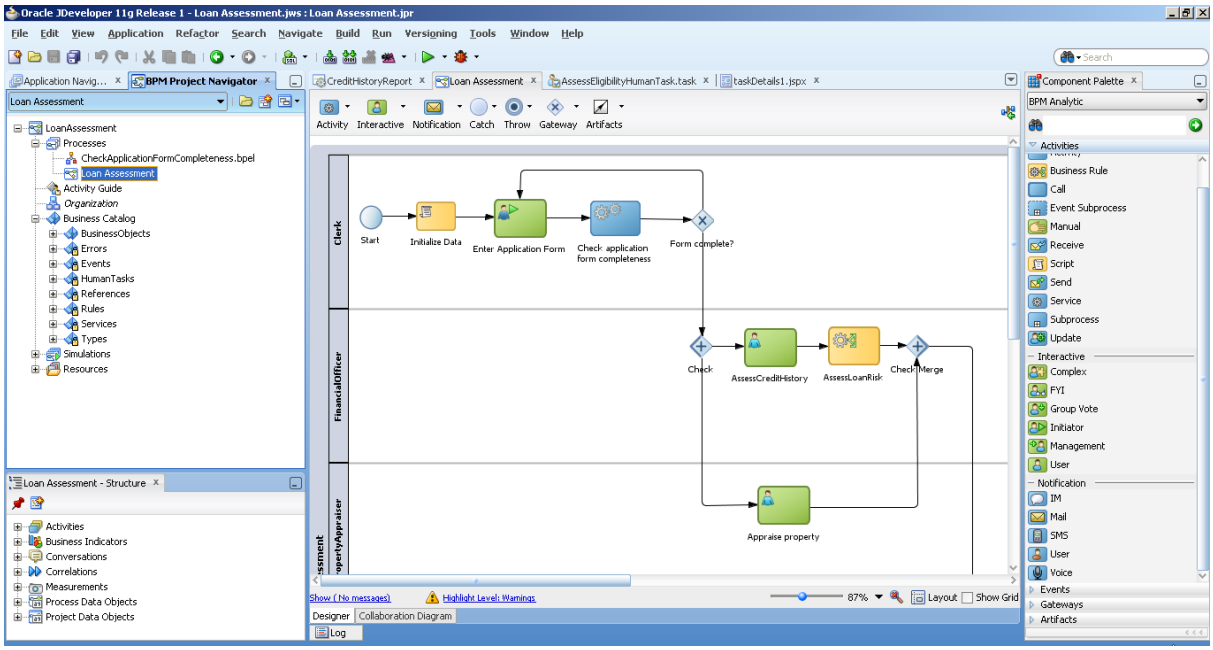
This section explains main elements of this environment so as to help you get acquainted to this tool. The essential skill is to understand the meaning of every structural element and where to find them when you need to.

First of all, readers of Oracle BPM documentation may be confused with many design editors such as BPM Studio, SOA Composite Editor, BPEL Process editor and so on; however, they will not be able to

find any of them as a software. Actually they are parts of Oracle JDeveloper installation and exist in one tool only, not multiple tools.

If you open Oracle JDeveloper now, you can see it contains all those design editors as interface components.

The main screen of Oracle JDeveloper is shown below. Note that there are docking panes on this user interface. You can turn on/off these panes through the View menu.



We'll focus on the main design panes as listed below in this section. Other supporting panes will be explained in details during the tutorial.

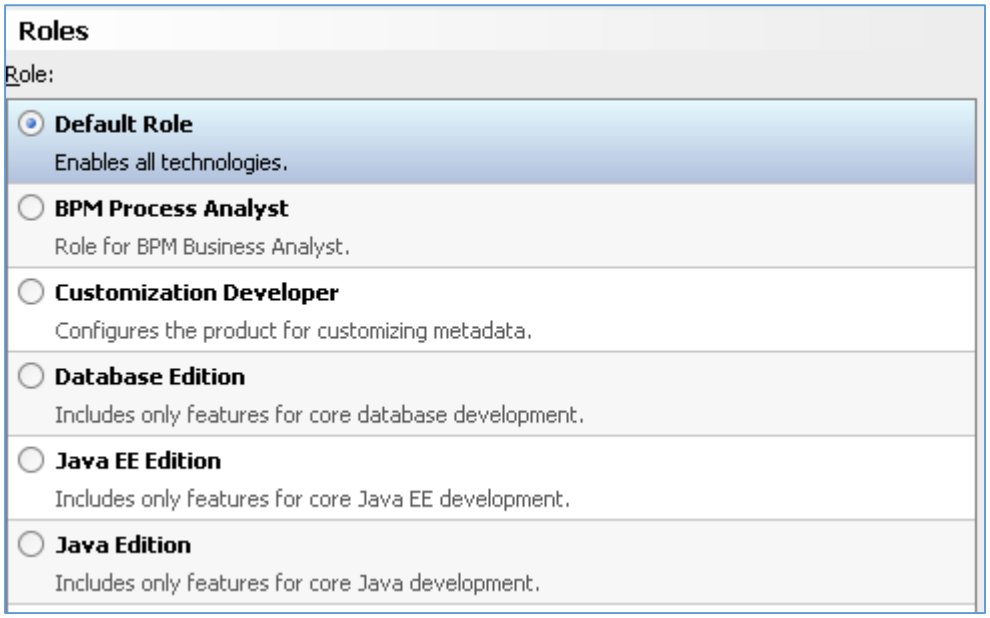
- BPM Project Navigator
- Application Navigator
- BPMN Editor
- BPEL Editor
- ADF Page Editor
- Component Palette
- Property Inspector

BPM Project Navigator and Application Navigator are two main explorers to open the design artefacts which will be shown in other design panes.

The design panes above are arranged by default into docking windows as displayed in the figure above. The window has basic facilities for a normal IDE: moving, floating, docking, pin, unpin, minimize, maximize. You can spend some time to familiarize yourself with these window features. You can also use View menu to find and unhide a particular pane.

At times you may encounter that a pane is invisible though you have selected to show it through the View menu. In that case you can restore the main screen to its standard arrangement by selecting Window menu > Reset Windows to Factory Settings. The whole JDeveloper window will be reset and the pane can be navigable again through the View menu.

The interface is configurable for different user role to simplify the number of accessible design panes. In case you cannot find a particular pane, the reason can be it is not relevant for the chosen role. You can select to make all panes visible in Oracle JDeveloper by going to Tools menu > Preferences > Roles and select "Default Role".



BPM Project Navigator

BPM Project Navigator contains logical elements of a BPM project such as process, organization, business catalog, simulations. Therefore, you should open this pane whenever you need to work with these BPM concepts.

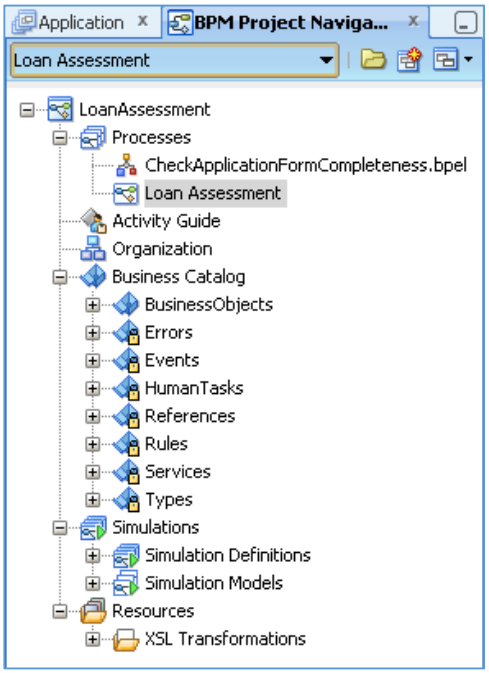
It is organized in a tree view with the top node is the project name. Pre-defined sub-nodes represent concepts of a BPM project.

The main node is Processes which allows us to create BPMN and BPEL processes.

The Organization is for people perspective of the process.

Business Catalog contains most of the elements of a process as seen in the figure. When you design a process model, these elements will be arranged into these folders.

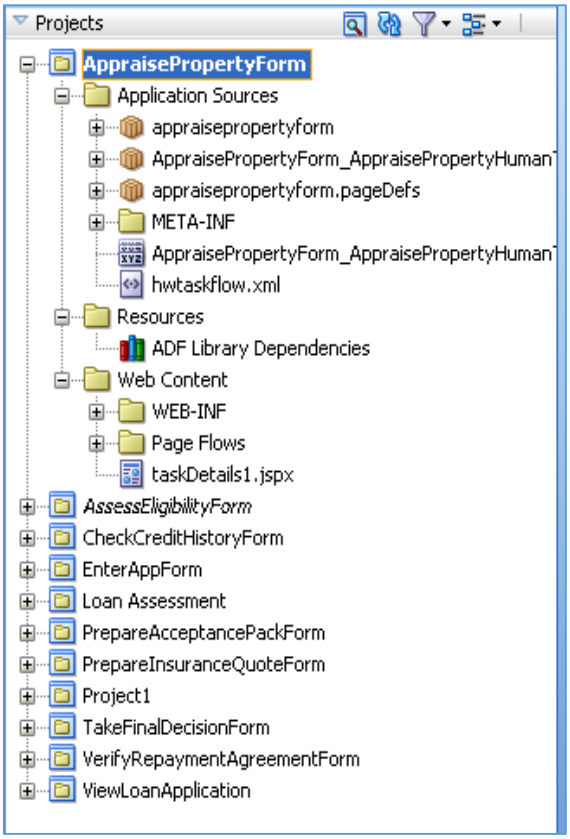
Activity Guide, Resources and Simulations: not used in this tutorial.



Application Navigator

A BPM application is structured as a group of projects, in which there is a BPM project with BPM resources such as BPMN process model file, BPEL model file, business rule model file, human task model file, etc., and several Web projects, each containing web page implementation for human tasks in the BPM project. Application Navigator is thus a tool to navigate and edit each project in the BPM application.

As outlined in the figure on the right, Loan Assessment is the BPM project whereas others are Oracle ADF projects which implement the human tasks in the Loan Assessment process.



The BPM project has structure as:

- Application Resources
- BPM Content
- SOA Content

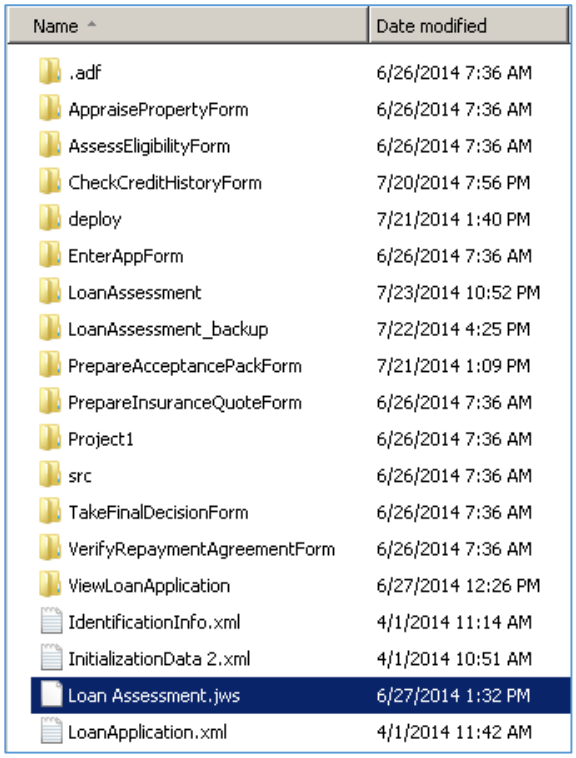
Every Oracle ADF project has three main folders:

- Application Resources
- Resources
- Web Content

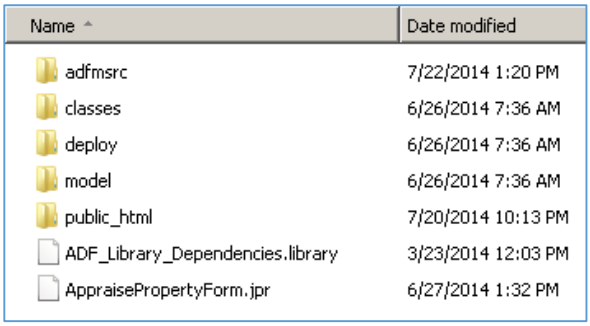
Application Navigator shows physical files in a project rather than a logical organization. If you want to have a deeper look at the file format, implementation details or specification level, you should explore the project with Application Navigator.

If we look at files level, the application folder is shown in the following picture, and it is the same structure as shown in the Application Navigator. The application master file has .jws extension at the root folder. There is one folder for BPM project and one folder for each Oracle ADF project. A project master file has .jpr extension.

It is not recommended to edit the project files directly (they are all text files) since most of them are auto-generated and managed via Oracle JDeveloper. Modifying them outside of Oracle JDeveloper may break the integrity of the application or project and cause unstable errors.



Name	Date modified
.adf	6/26/2014 7:36 AM
AppraisePropertyForm	6/26/2014 7:36 AM
AssessEligibilityForm	6/26/2014 7:36 AM
CheckCreditHistoryForm	7/20/2014 7:56 PM
deploy	7/21/2014 1:40 PM
EnterAppForm	6/26/2014 7:36 AM
LoanAssessment	7/23/2014 10:52 PM
LoanAssessment_backup	7/22/2014 4:25 PM
PrepareAcceptancePackForm	7/21/2014 1:09 PM
PrepareInsuranceQuoteForm	6/26/2014 7:36 AM
Project1	6/26/2014 7:36 AM
src	6/26/2014 7:36 AM
TakeFinalDecisionForm	6/26/2014 7:36 AM
VerifyRepaymentAgreementForm	6/26/2014 7:36 AM
ViewLoanApplication	6/27/2014 12:26 PM
IdentificationInfo.xml	4/1/2014 11:14 AM
InitializationData 2.xml	4/1/2014 10:51 AM
Loan Assessment.jws	6/27/2014 1:32 PM
LoanApplication.xml	4/1/2014 11:42 AM



Name	Date modified
adfmsrc	7/22/2014 1:20 PM
classes	6/26/2014 7:36 AM
deploy	6/26/2014 7:36 AM
model	6/26/2014 7:36 AM
public_html	7/20/2014 10:13 PM
ADF_Library_Dependencies.library	3/23/2014 12:03 PM
AppraisePropertyForm.jpr	6/27/2014 1:32 PM

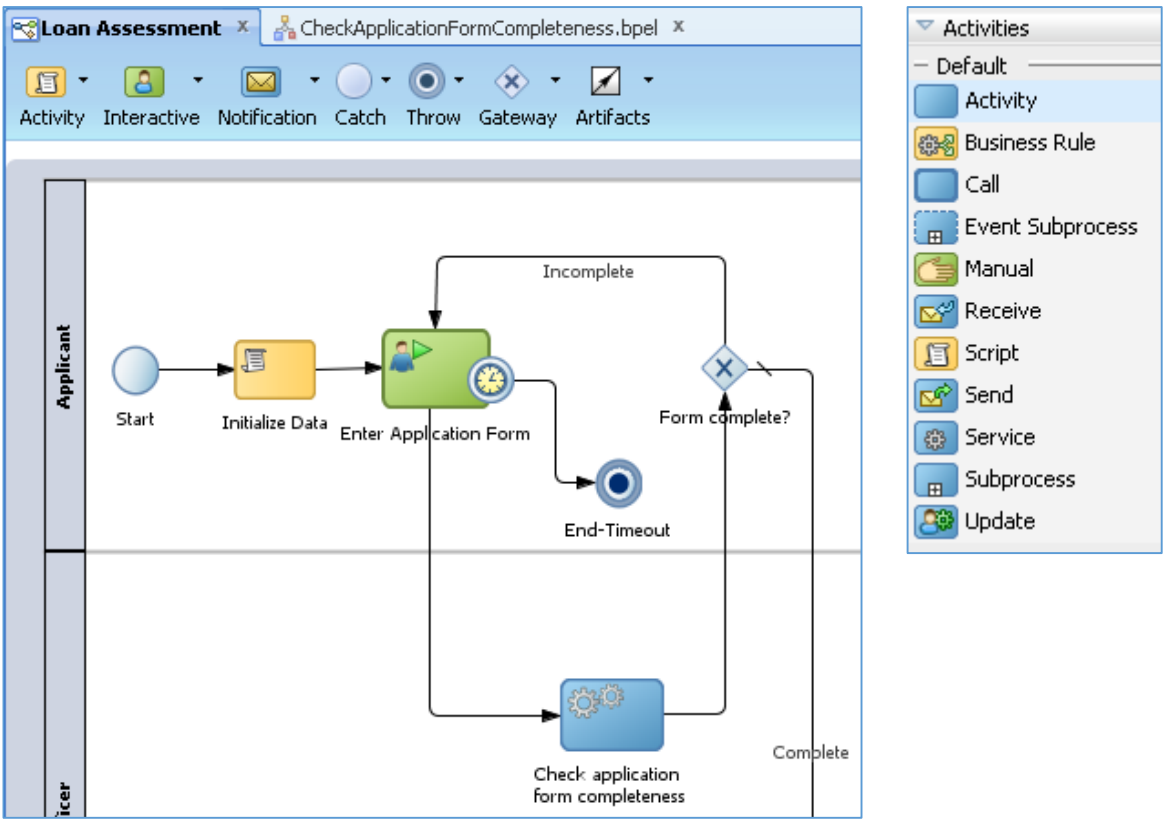
Design Editor, Structure, Component Palette and Property Inspector

There are three main design editors used for a BPM application: BPMN editor, BPEL editor and Oracle ADF editor. Editor is used to open and edit a design (default location is in the center of the main screen). Each design, either BPMN process model or BPEL process model or ADF page, has a structure which can be viewed via the Structure pane (default location is at the bottom left corner of the main screen). Each design type is also accompanied with a set of design objects shown in the Component Palette (default location is on the right of the main screen). Each design object has a set of associated properties shown in the Property Inspector pane (default location is on the right of the main screen).

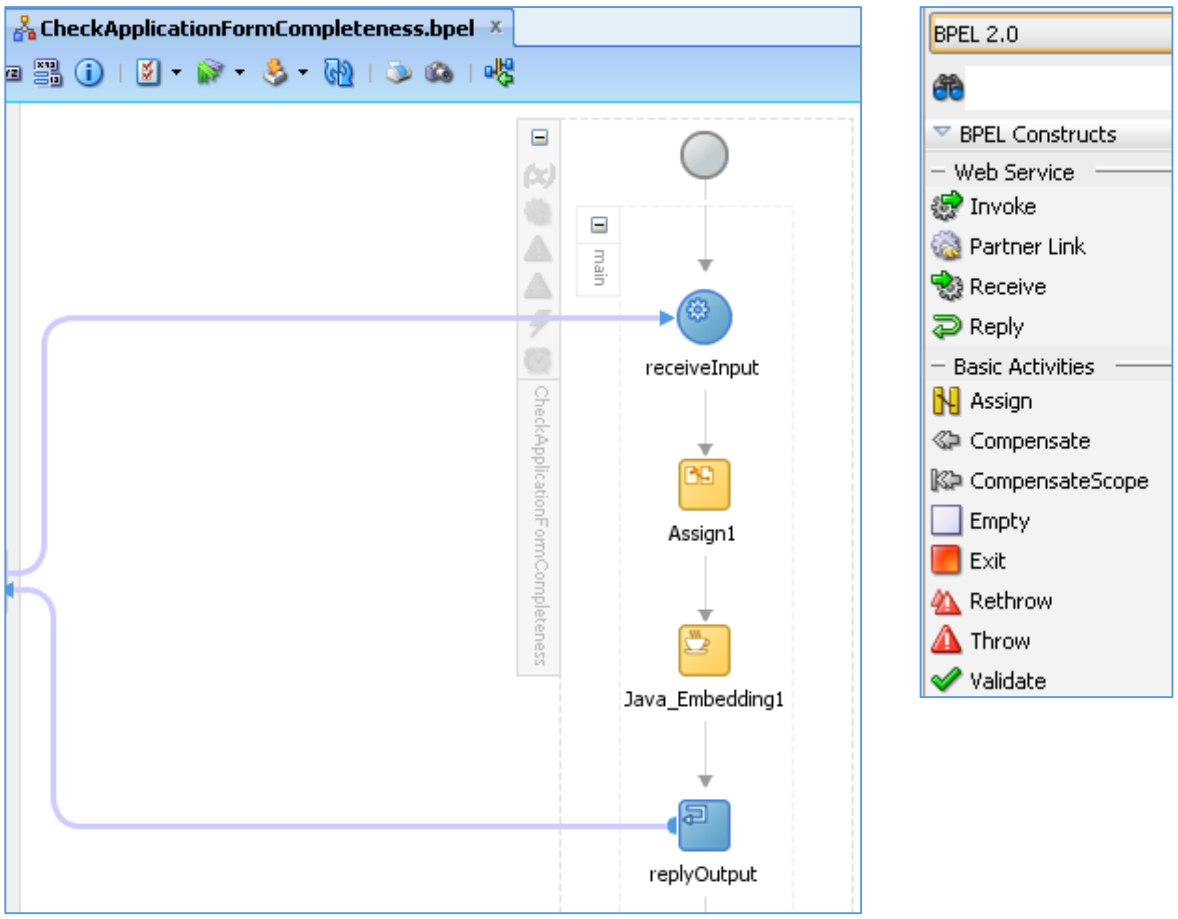
The design editor has a visual display of its elements. When we open a design editor, the Component Palette will open the corresponding component set for that editor. When we click on a component in the design editor, the Property Inspector will show properties of that component.

The figure below shows some examples of the three main editors for BPMN, BPEL and Oracle ADF.

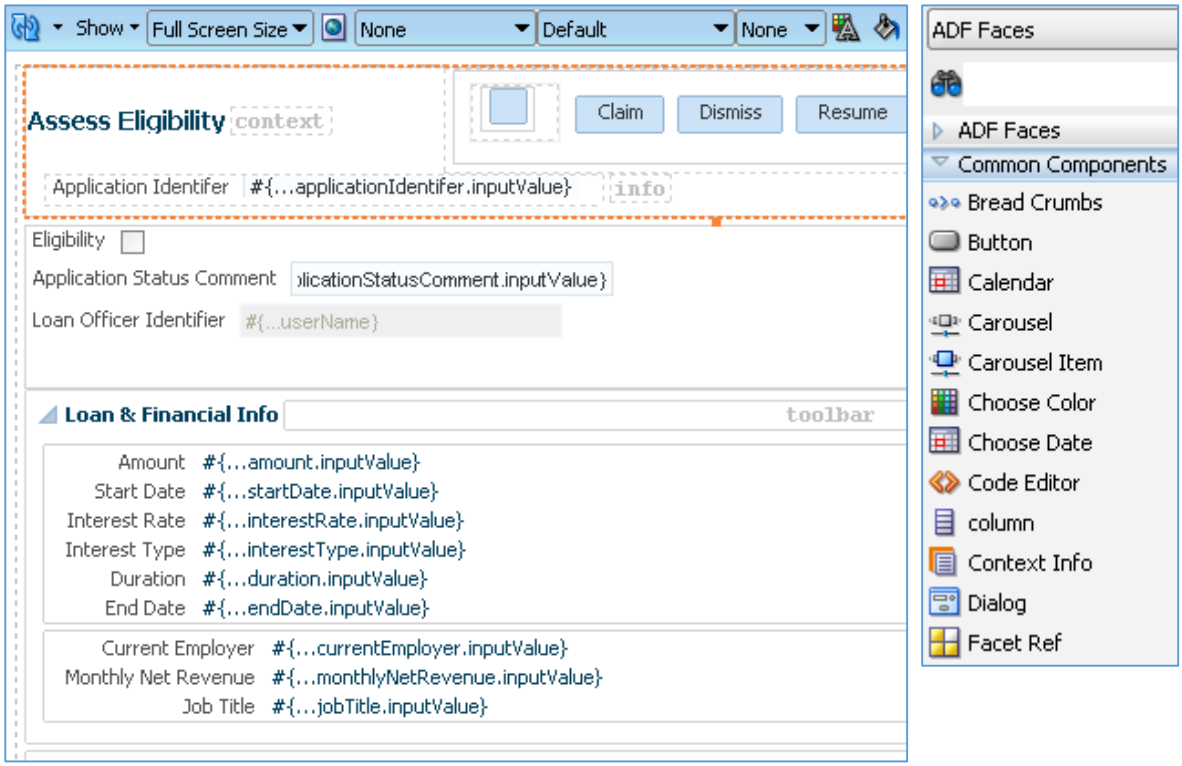
BPMN Editor


Loan Assessment
CheckApplicationFormCompleteness.bpel
Activity
Interactive
Notification
Catch
Throw
Gateway
Artifacts
Applicant
Start
Initialize Data
Enter Application Form
Incomplete
Form complete?
End-Timeout
Check application form completeness
Complete
Activities
Default
Activity
Business Rule
Call
Event Subprocess
Manual
Receive
Script
Send
Service
Subprocess
Update

BPEL Editor



ADF Editor



Working with a Design Editor

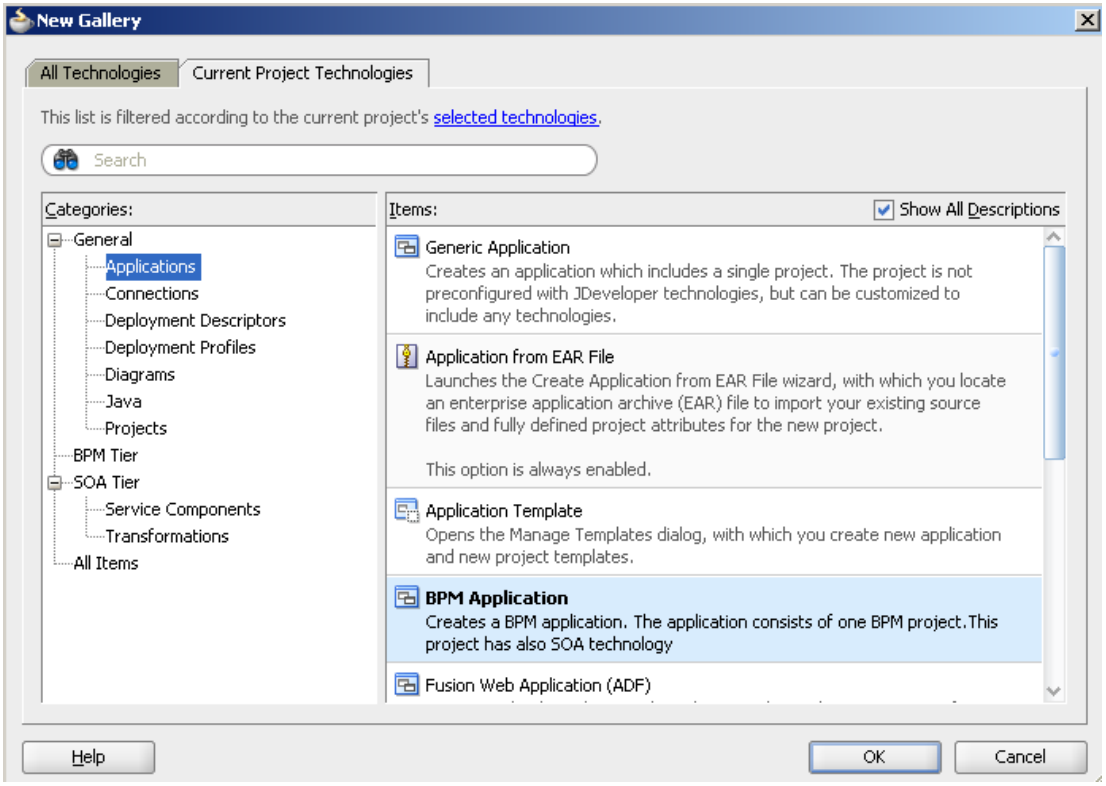
All design editors have common design features:

- Support visual view
- Support drag and drop from the Component Palette
- Support Undo and Redo (shortcut key is Control-Z)
- Support a context menu with a right click on a design object
- Support a context menu (right click)

5 Create a new BPM application project

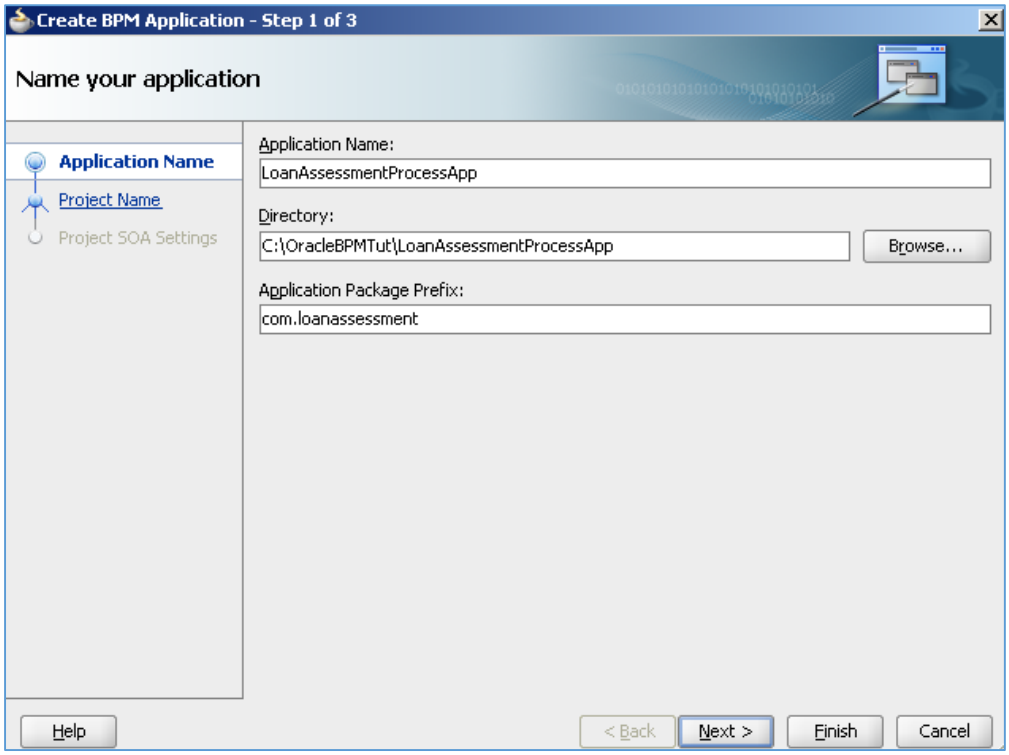
When logging into Oracle JDeveloper, remember to select “All Roles” so that we have all privileges in the system for project activities in the following instructions.

To open a new project for Loan Assessment process, click File > New, select General > Applications > BPM Applications. Click OK.



In Application Name window, type Project Name: “**LoanAssessmentProcessApp**”. Select a folder for the application. Choose an application package prefix, e.g. “com.loanassessment”

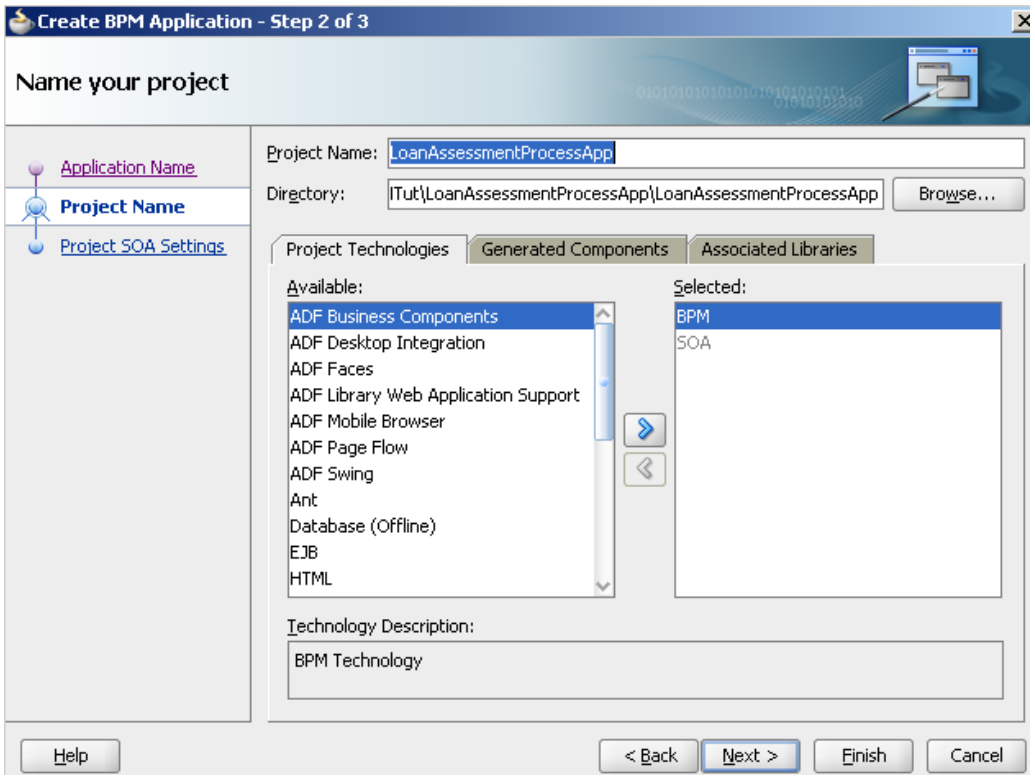
Click Next.



In Project Name window, enter a name for the project ("**LoanAssessmentProcessApp**"). Note that a folder is selected for the project which is a subfolder of the application folder. This is a good practice because an application can have multiple projects, each for different components, particularly for ADF components. We'll create more projects for this application later on.

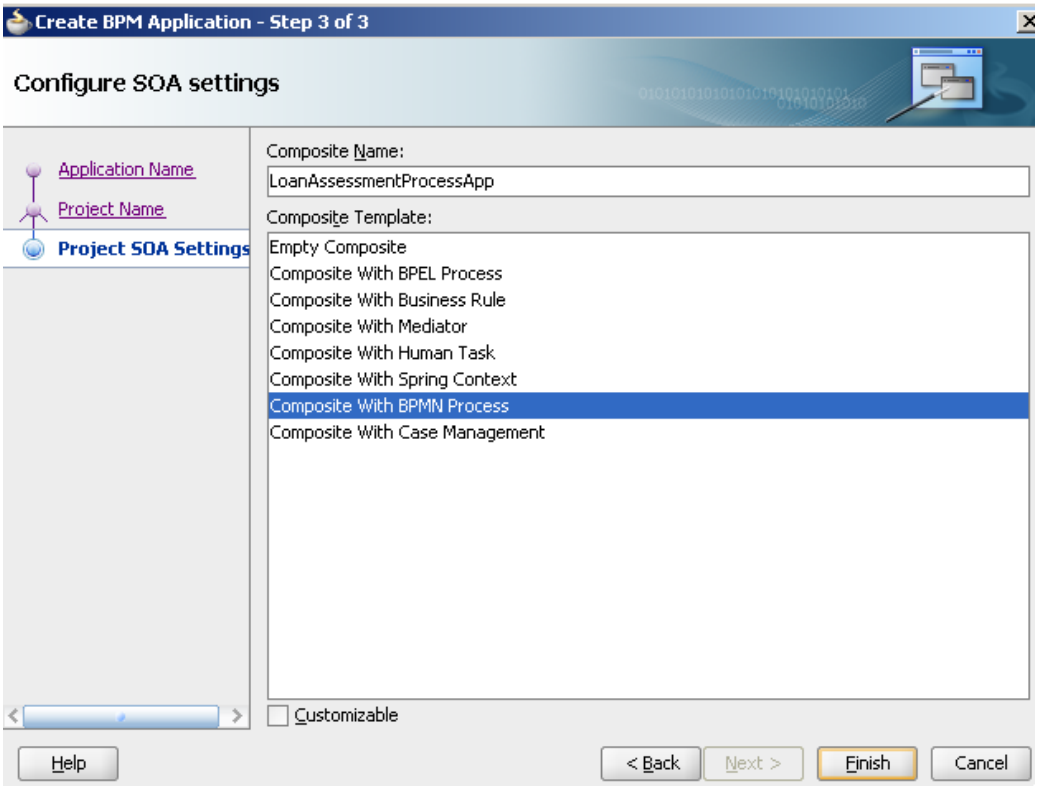
Predefined technologies such as BPM and SOA have been selected by default for the project.

Click Next.

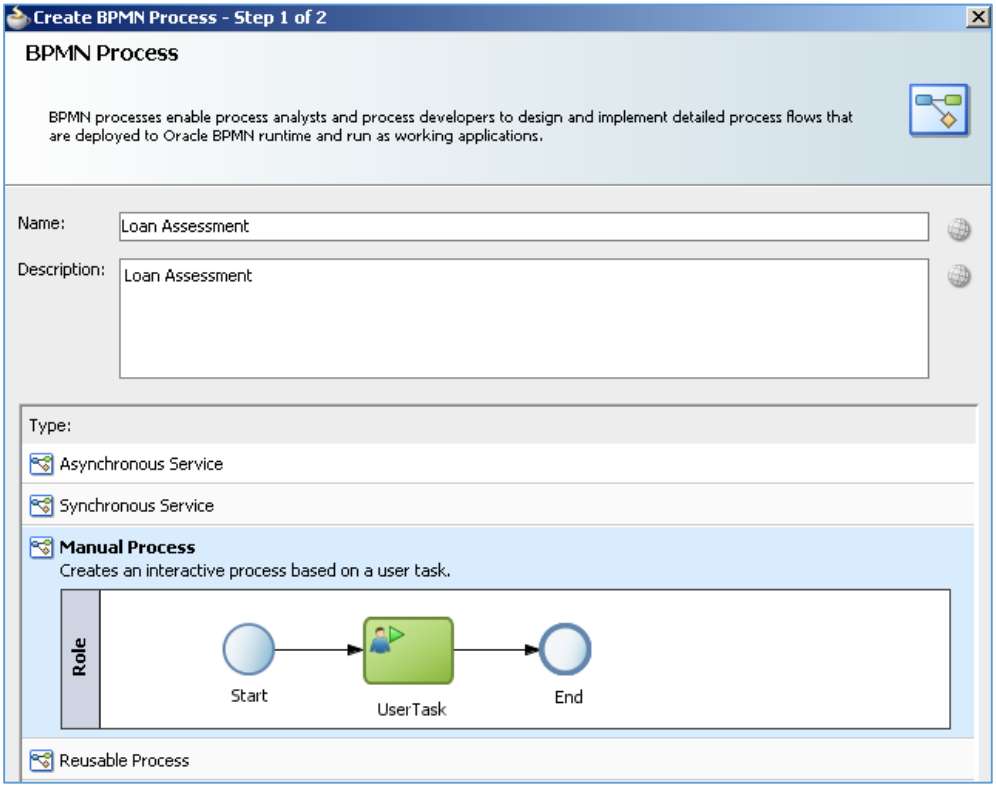


In Project SOA Settings window, select “Composite with BPMN process” option. This means we are going to design BPMN process and create BPMN-based process application as a composite application.

Click Finish.

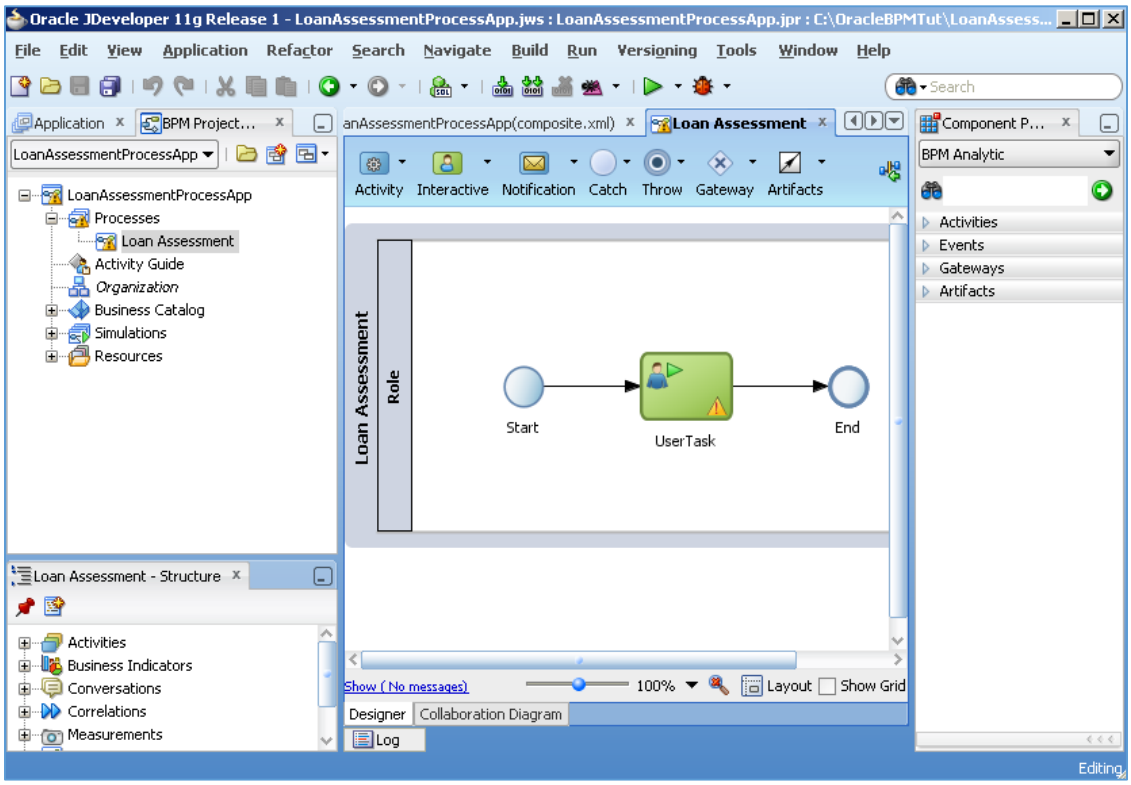


A Wizard will display to select style of process we want to create first. Choose “Manual Process” since our process will have many interactive activities. Then, enter a name the process: “Loan Assessment”.



Click Finish in the next window with default values (Sampling points are defined ways of collecting data for process analytics, not the focus of this tutorial).

A new project will be created and shown in the design environment. It has a lane called “Loan Assessment” and a default role called “Role”. There is a start event, end event and the first activity called “UserTask”.



In project folder, a hierarchy of files are generated as shown. File “.jws” is the project file.

Name	Date modified	Type	Size
.adf	6/26/2014 12:14 PM	File folder	
LoanAssessmentProcessApp	6/26/2014 12:20 PM	File folder	
src	6/26/2014 12:14 PM	File folder	
LoanAssessmentProcessApp.jws	6/26/2014 12:14 PM	JWS File	2 KB

6 Model the Process

Starting from the initial process model, we can drag and drop modelling widgets from Component Palette to the BPMN editor. Oracle BPM has support for BPMN 2.0 notation along with additional Oracle specific elements such as Script, Email notification, Service call, and Business rule.

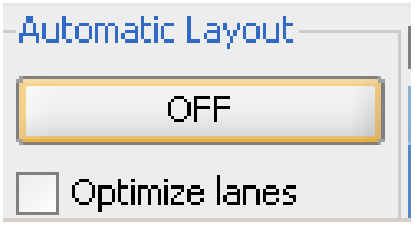
The primary stages of modelling a process in BPMN are as follows.

- First, we add process lanes (also called roles) since they can be quickly identified from the process scenario and help establishing initial structure for the model
- Then, we add main process activities which should cover the process scenario as much as possible
- Then, we add ancillary activities to implement some specialized process requirements which are not yet covered in previous stage.

Turn off Automatic Layout

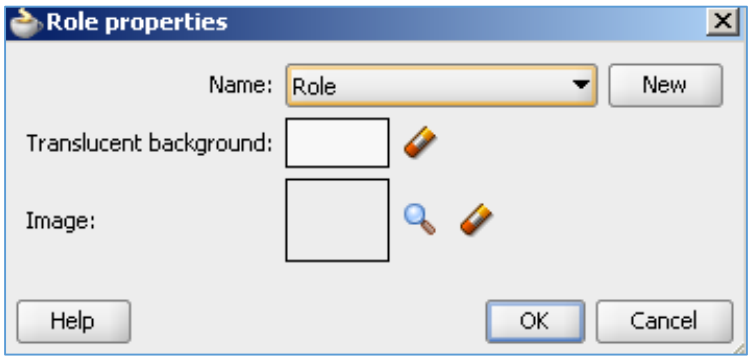
You should turn off the automatic layout setting of the BPMN editor, otherwise it may inadvertently change the layout of the model out of your control and take much time on re-layouting the model again and again.

In the BPMN editor, click on the Layout button at the bottom area. If the setting in the popup is ON then turn it to OFF.



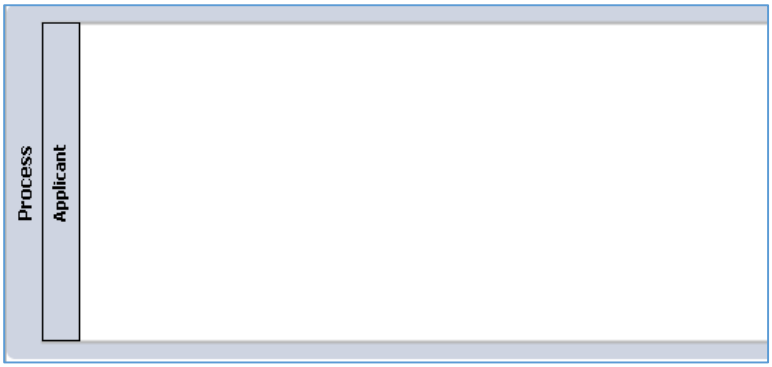
6.1 Add process lanes

1. Open the initial process model in BPMN editor (in BPM Project Navigator, expand the project and Processes, double click on “Loan Assessment” process).
2. Select the start event, the UserTask activity and stop event. Press Delete button to delete them all. We clean them so that we focus on lanes at the moment. We’ll add activities later.
3. Right click on the current Role in the model called “Role”, select Properties. In the Role properties, select New.



In the Role popup, type “Applicant” and click OK.

The Applicant role will be created and replace the role “Role” in the model.

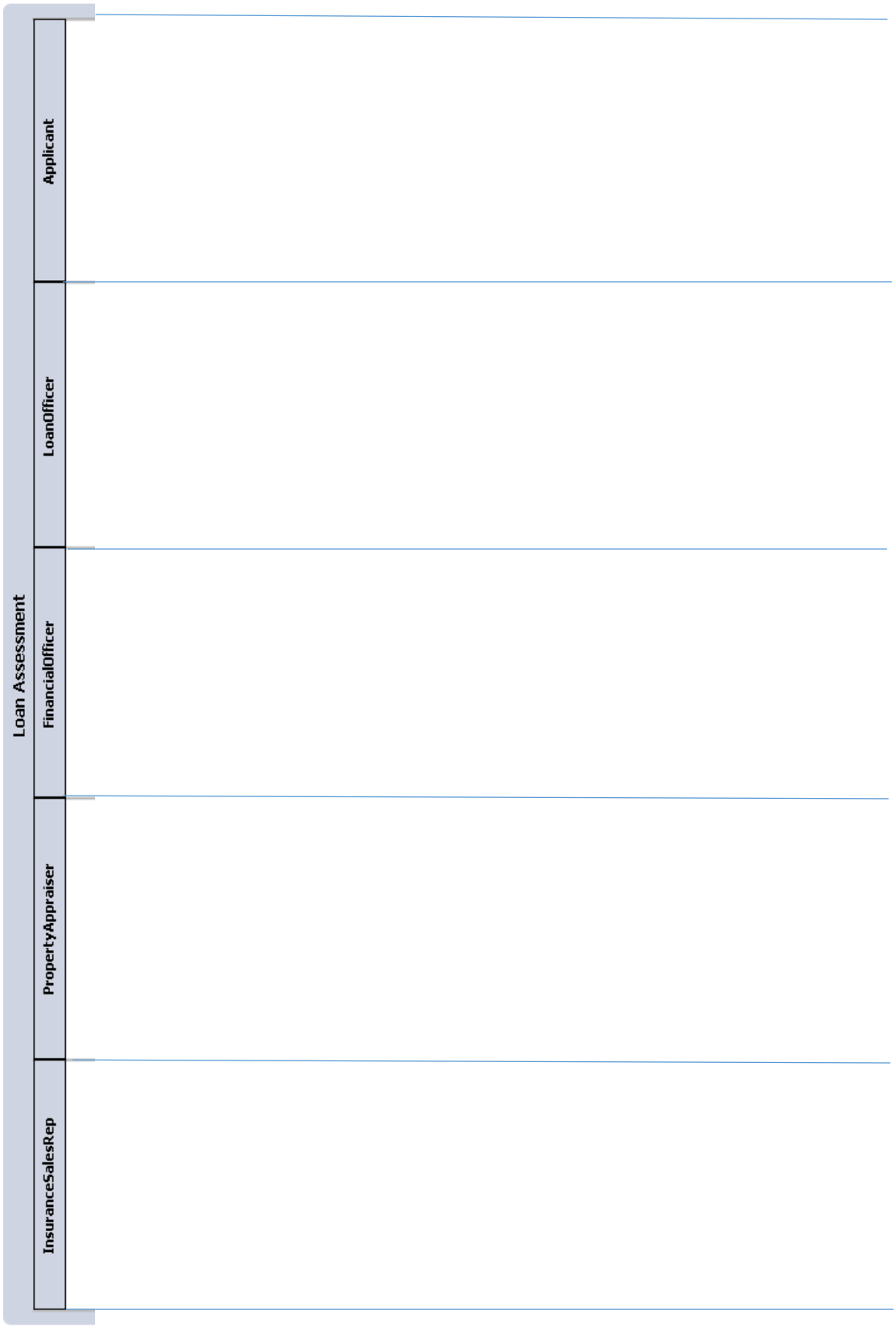


4. Right click on the empty space of the process model, select “Add Role” in the menu. In the Role properties, select New.

Type “Loan Officer” and click OK. A Loan Officer role is created and added below the Applicant role in the model.

5. Repeat the same steps to create Financial Officer, Property Appraiser and Insurance Sales Rep role. Note that the role name does not allow white space. You can drag and drop the role on each other to adjust the order of them.

   The new process model with all roles (lanes) should look similar to the figure below.


Loan Assessment
Applicant
LoanOfficer
FinancialOfficer
PropertyAppraiser
InsuranceSalesRep

6.2 Add main activities and sequence flows

Open the process model in BPMN editor.

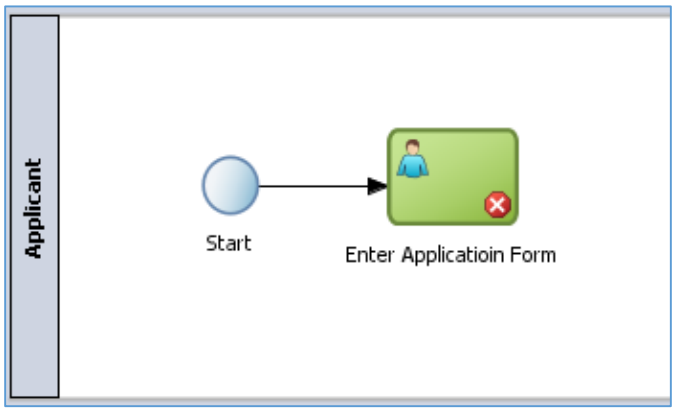
Add "Start" event

1. In Component Palette, expand Events section.
2. Drag "None" component from "Start Events" section onto the Applicant lane.
3. In the "Properties" pop-up, enter "Start" in the Name field, then click OK.
4. A Start event will be added to the Applicant lane. Ignore the red cross sign at the moment.

Add "Enter Application Form" activity

1. In Component Palette, expand Activities, drag a User activity from Interactive section onto the empty space of Applicant lane.
2. In the Properties – UserTask1 pop-up, enter a value for the Name field: "Enter Application Form". Click OK.
3. "Enter Application Form" activity is created and shown in the Applicant lane.
4. Drag Start event and "Enter Application Form" activity to adjust them to be horizontally aligned and from left to right order, respectively.
5. In Component Palette, expand Artifacts. Click on Sequence Flow to select it and then move and hover your mouse over the Start event. The mouse icon changes to a plus sign. Click the mouse on the Start event and drag it onto "Enter Application Form" activity and drop there. A sequence flow will connect the Start event to "Enter Application Form".

   Note that you can always click on the sequence flow, hold and move it to adjust the connection line.

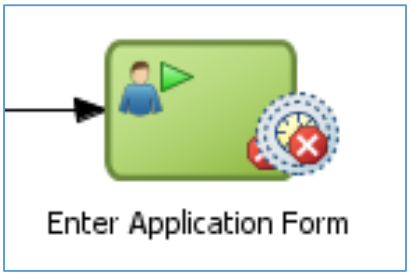


Add Timer event for "Enter Application Form" activity

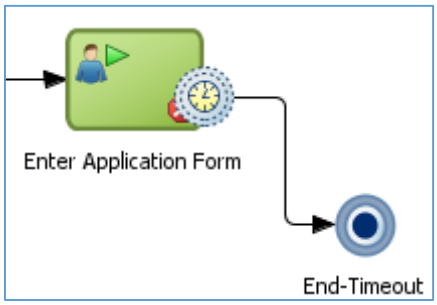
The scenario is if the "Enter Application Form" lasts for more than 5 days, the system will terminate that step and the process stops.

Modelling steps:

1. In Component Palette, expand Events
2. Drag Timer from "Catch Events" section and drop it on the edge of the "Enter Application Form" activity.
3. In the "Properties – BoundaryEvent" popup, enter "TimeOut" in the Name field then click OK.
4. The timer will attach to the edge of the activity shape as shown below.



5. Drag a Terminate event from Component Palette, End Events section to the model on Applicant lane.
6. In the “Properties – End” popup, enter “End – Timeout” in the Name field. Click OK.
7. A Terminate event will be added to the model
8. Drag a Sequence Flow to connect the Timer event of Enter Application Form activity to the Terminate event. Note that you can always click on the sequence flow, hold and move it to adjust the connection line.



Add a Service Call for checking form completeness

The scenario is the system will check the completeness of the application form automatically and returns a result whether the form is complete or not.

Modelling steps:

1. Drag a Service activity component from Component Palette, Default section to the Loan Officer lane but locate it after the Enter Application Form activity.
2. In the “Properties – ServiceTask” popup, enter “Check application form completeness” in the Name field. Click OK.
3. A Service Call activity will be added to the model.
4. Connect the Enter Application Form activity to the Check Application Form Completeness with a sequence flow.

Add an Exclusive Gateway for form completeness decision

The scenario is the process checks the result of “Check Application Form Completeness” activity. If the form is complete, it proceeds to the next activity. Otherwise, the process returns the application form to the Applicant to supplement the missing or invalid information.

Modelling steps:

1. Drag an Exclusive Gateway from Component Palette, Gateways section to the Applicant lane in alignment with the Enter Application Form activity but after the Check Application Form Completeness activity.
2. In the “Properties – ExclusiveGateway” popup, enter “Complete?” in the Name field. Then click OK.
3. An Exclusive gateway will be added to the model.

4. Connect Check Application Form Completeness activity to the “Complete?” gateway with a sequence flow.
5. Connect “Complete” gateway back to Enter Application Form activity with a sequence flow.

Add a Parallel gateway for directing process flow to two activities at the same time

The scenario is if the form is checked as complete, the system will forward the form to Financial Officer and Property Appraiser at the same time for assessment.

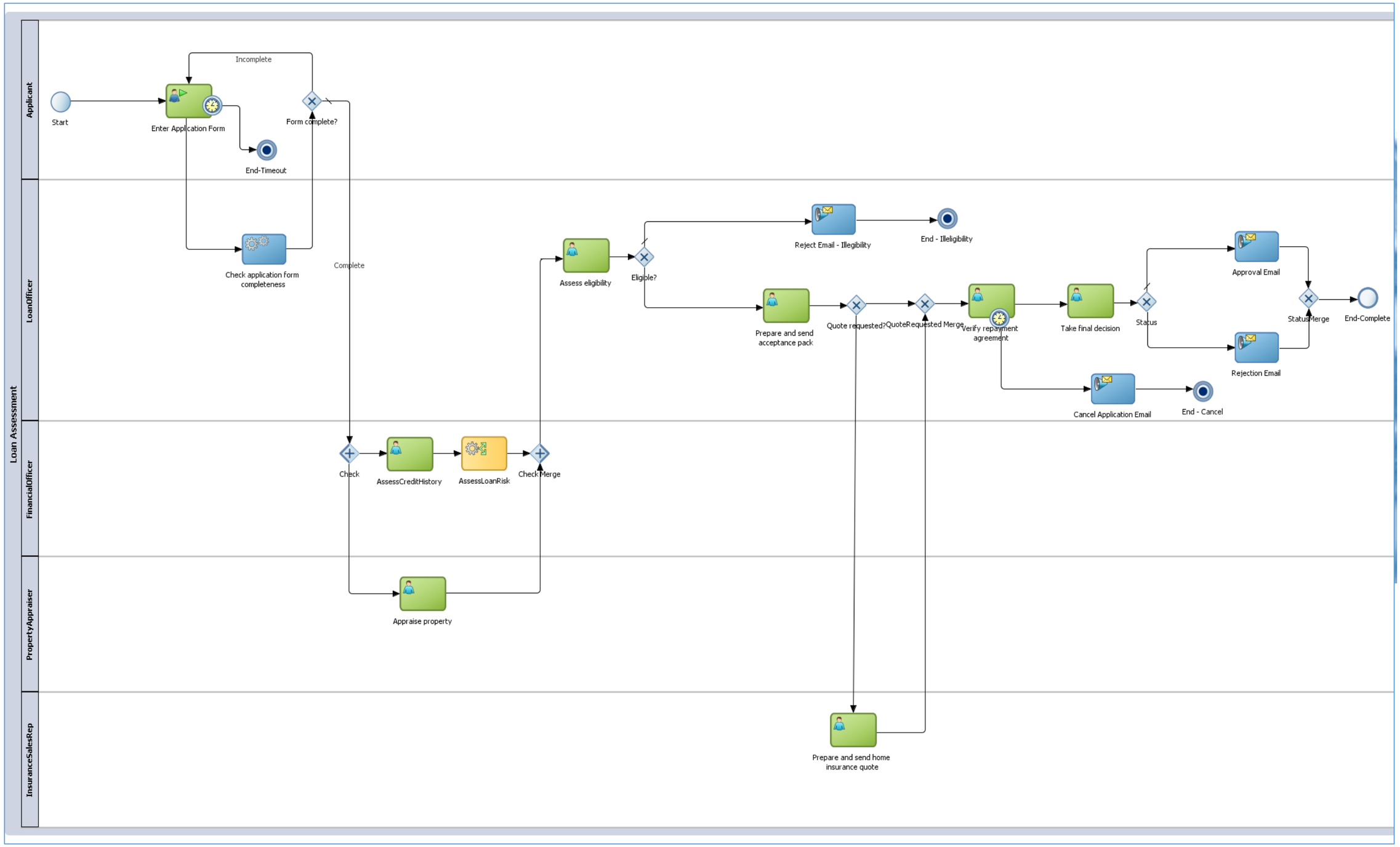
Modelling steps:

1. Drag a Parallel Gateway from Component Palette, Gateways section to the Financial Officer lane and place it after the “Complete?” gateway horizontally
2. In the “Properties – ParalellGateway” popup, enter “Check” in the Name field. Then click OK.
3. A Parallel gateway will be added to the model.
4. Connect “Complete?” gateway to the “Check” parallel gateway with a sequence flow.

At this point, you have been quite familiar with different types of components, Component Palette and sequence flow. Let’s apply the same steps for the remaining process scenarios. The process model at the end of this stage should be modelled and connected similarly to the following figure.

Below is remaining components listed in the order of their presence in the model.

- Assess Credit History activity, User activity component.
- Appraise Property activity, User activity component
- Assess Loan Risk, Business Rule component
- Parallel gateway for merging parallel flows from Assess Credit History activity and Assess Property activity.
- Assess Eligibility activity, User activity component
- Eligible? Exclusive gateway
- Reject Email – Illegibility activity, Mail activity component
- Terminate event after Reject Email – Illegibility activity, Terminate event component
- Prepare and send acceptance pack activity, User activity component
- Quote Requested? Exclusive gateway
- Prepare and send home insurance quote, User activity component
- Quote Request Merge exclusive gateway
- Verify Repayment Agreement activity, User activity component
- TimeOut timer for Verify Repayment Agreement activity
- Cancel Application Email activity, Mail activity component
- Terminate event after Cancel Application Email activity, Terminate event component
- Take Final Decision activity, User activity component
- Status exclusive gateway to check Approval status after Take Final Decision activity
- Approval Email notification, Mail activity component
- Rejection Email notification, Mail activity component
- StatusMerge exclusive gateway
- End – Complete event, “None” event component.


Loan Assessment
Applicant
LoanOfficer
FinancialOfficer
PropertyAppraiser
InsuranceSalesRep
Start
Enter Application Form
Incomplete
Form complete?
End-Timeout
Check application form completeness
Complete
Assess eligibility
Eligible?
Reject Email - Illegibility
End - Illeligibility
Prepare and send acceptance pack
Quote requested?
QuoteRequested Merge
Verify repayment agreement
Take final decision
Status
Approval Email
Rejection Email
StatusMerge
End-Complete
Cancel Application Email
End - Cancel
Check
AssessCreditHistory
AssessLoanRisk
CheckMerge
Appraise property
Prepare and send home insurance quote

6.3 Add ancillary activities

In the next modelling steps, we will add some implementation specific activities in accordance with the process scenarios.

Initialize process data

The requirement here is Oracle BPM engine cannot initialize the process data automatically since our process data is complex with compound data objects (data object contains another data objects and so on). We have to do it explicitly in the process. Intuitively we may encounter this requirement only after we have implemented data objects for the process (explained in the succeeding section). However, for business process with potentially complex data objects, this issue can be anticipated beforehand and thus the in-charged business analyst can perform this modelling action.

Modelling steps:

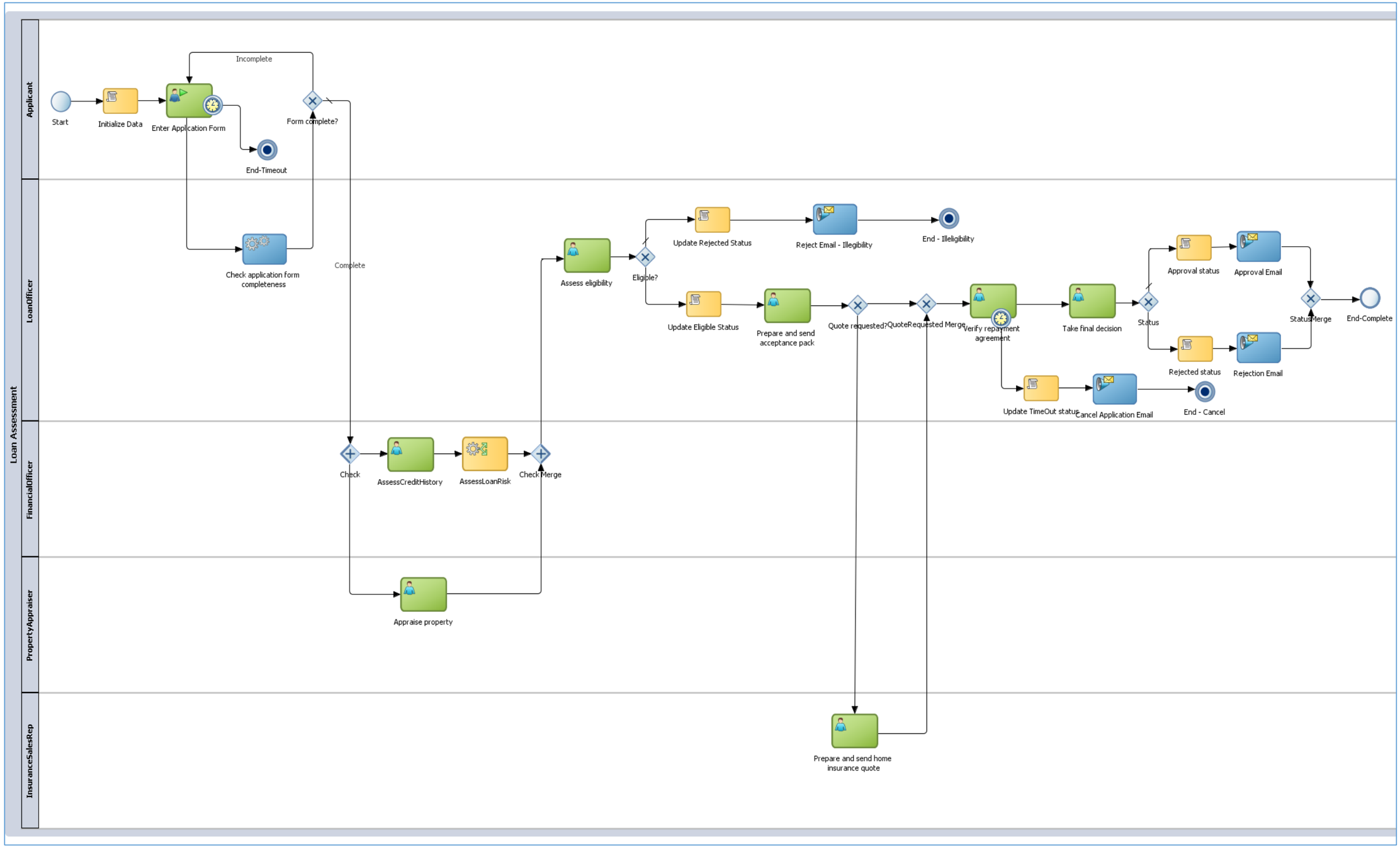
1. Drag a Script activity component from Component Palette, Activities section to Applicant lane, drop it on the sequence flow between the Start event and the Enter Application Form activity.
2. In “Properties – ScriptTask” popup, enter “Initialize Data” in the Name field. Then click OK.
3. A script activity will be added to the model with sequence flow is reconnected from Start event to the Initialize Data activity and from that activity to Enter Application Form activity.

Update application status

The scenario is the application status will be updated after some particular activities, such as after Assess Eligibility activity or Take Final Decision activity.

Modelling steps: we use a Script activity component and follow the same steps as those for Initialize Data activity.

The process model at the end of this stage should look like the figure below. This step also completes process modelling activities and we can move on to process implementation in the next section.


Loan Assessment
Applicant
LoanOfficer
FinancialOfficer
PropertyAppraiser
InsuranceSalesRep
Start
Initialize Data
Enter Application Form
Incomplete
Form complete?
End-Timeout
Check application form completeness
Complete
Assess eligibility
Eligible?
Update Rejected Status
Reject Email - Illegibility
End - Illeligibility
Update Eligible Status
Prepare and send acceptance pack
Quote requested?
QuoteRequested Merge
Verify repayment agreement
Take final decision
Status
Approval status
Approval Email
Rejected status
Rejection Email
StatusMerge
End-Complete
Update TimeOut status
Cancel Application Email
End - Cancel
Check
AssessCreditHistory
AssessLoanRisk
Check Merge
Appraise property
Prepare and send home insurance quote

7 Implement the Process

In this section, we will add implementation details for all modelling components completed in preceding section.

7.1 Initial set-up

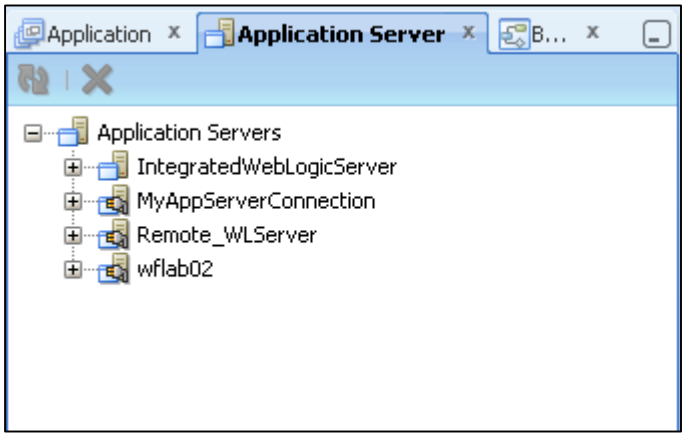
Some configurations need to be set up prior to implementation activities. You might need assistance from your administrator for system parameters (username, password, port number, etc.).

7.1.1 Server Connection

This is to set up a connection to SOA/BPM server. There are implementation steps we need to get information from the server such as list of users, database tables. It is also used during process deployment.

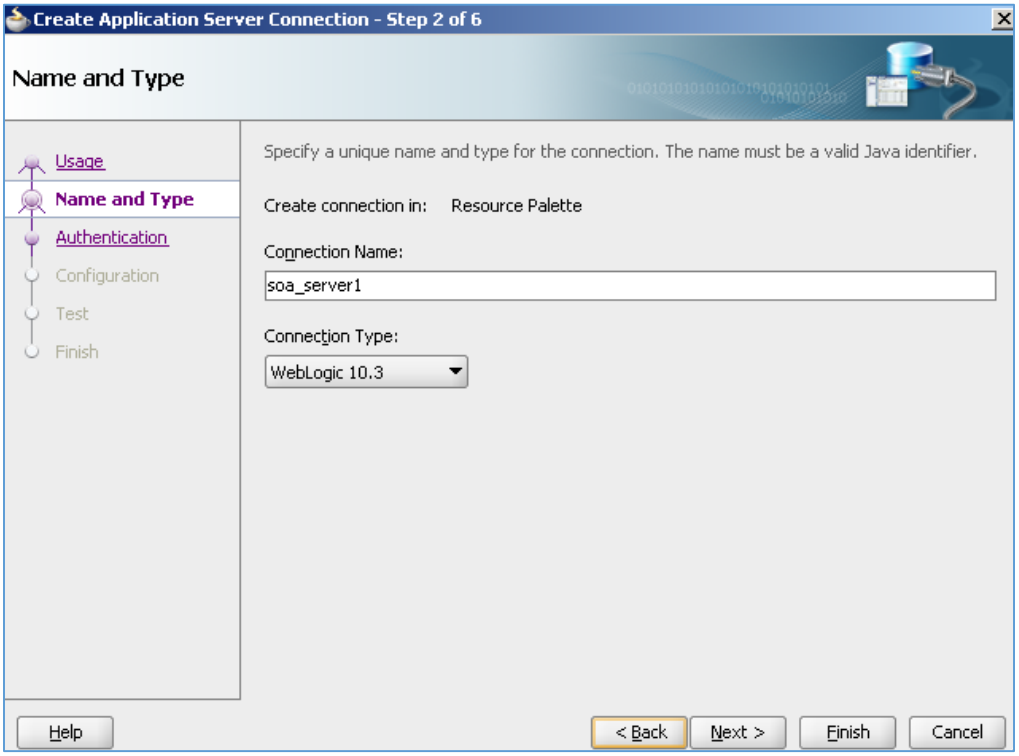
Click on View menu > Application Server Navigator.

Right click on Application Servers, choose “New Application Server…” to open a Create Application Server connection dialog.



Click Next in the first screen (with default value is Standalone Server which is this tutorial installation).

Type any name without spaces in Connection Name.



Enter system information including username, password, server host name, port, domain name.

Click Test Connection.

If the test does not succeed, one of the cause may be firewall block connection to the server.

7.1.2 Email Notification

In order to implement Mail activity, we need to configure the User Messaging Service (UMS) on Weblogic Server so that it can send email. One way is to install a mail server and set up connection between UMS to the mail server. Another way is we can configure UMS to connect to Google mail server as a relay server to send email. In this section we'll use the second way for simplicity.

It is required to have Internet connection to send email from Oracle SOA server to an external email. Make sure that the connection is available on the server.

The instruction below is from an Internet resource at http://www.soatutor.com/2012/10/setting-up-email-notification-in-oracle.html.

There are four steps to configure email notification.

Step 1: Import certificates from gmail and add it to your server trust store
Step 2: Configure email driver properties
Step 3: Enable notification mode
Step 4: Testing the configuration

Step 1: Import certificates from Gmail and add it to your server trust store

Any email server uses two protocols to send/receive messages. SMTP for sending mails. Either POP3 or IMAP for receiving mails. Gmail uses IMAP for retrieving mails.

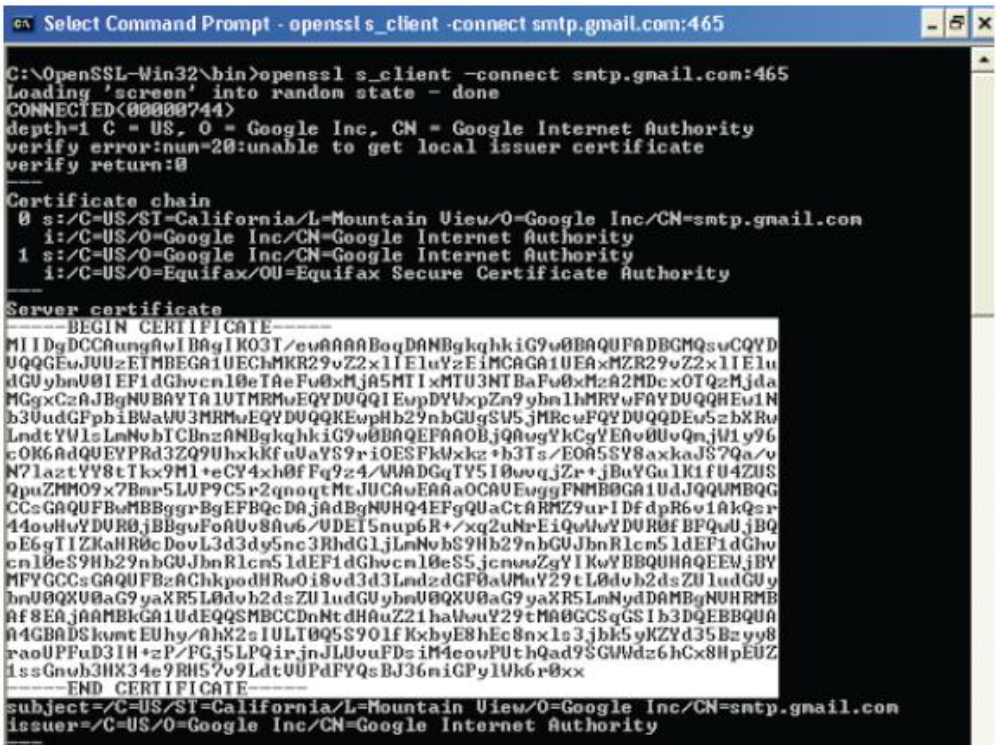
So, you need to get both SMTP and IMAP certificates connecting to the Gmail server in order to send/receive mails to/from your inbox.

You can download the certificates using an open source software called openssl. First, you need to download and install it.

Open command prompt and cd to openssl_install_folder/bin. Give the below command to view the SMTP certificate.

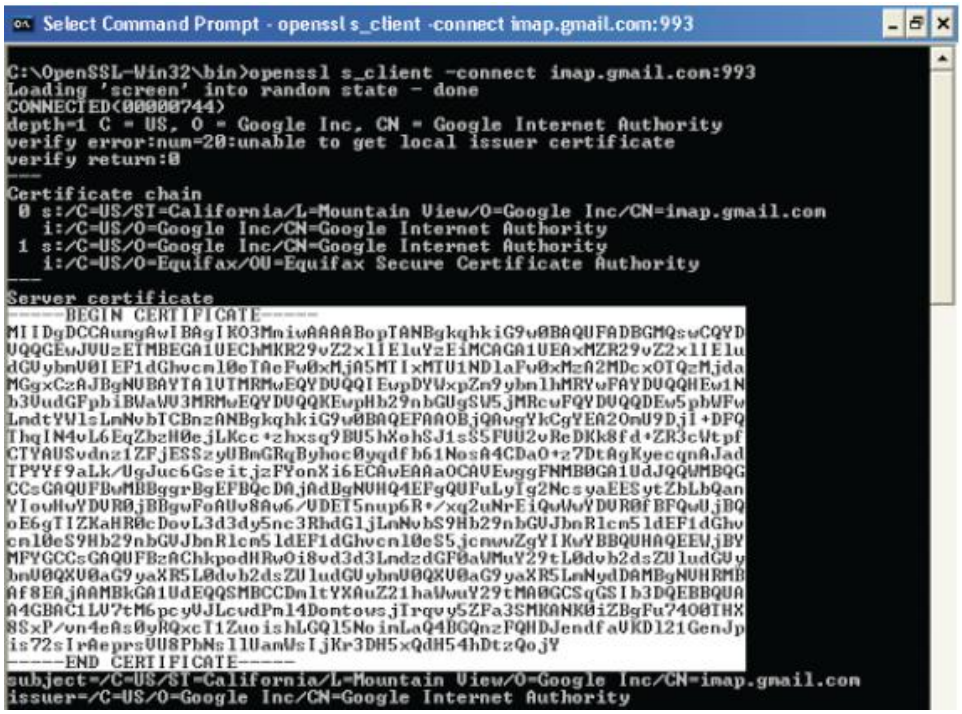
```
openssl s_client -connect smtp.gmail.com:465
```

Copy the code highlighted in the picture above and paste it to a file and name it smtp_gmail.cert



Similarly, issue the follwing command to view the IMAP certificate

```
openssl s_client -connect imap.gmail.com:993
```



Now that you have both smtp and imap certificates with you, you need to import these to your server trust store

For this, open command prompt and navigate to %JAVA_HOME%/bin. Issue the following command

```
keytool -import -alias smtp.gmail.com -keystore trusted-certificates.jks -
file <location of smtp_gmail.txt>
```

```
keytool -import -alias imap.gmail.com -keystore trusted-certificates.jks –
file <location of imap_gmail.txt>
```

Once you are done with importing the trust certificates to the keystore using the keytool, you need to tell the managed server(soa_server1) that there is a user defined trust store from which it has to look for keystore.

This will be done by editing the %MIDDLEWARE_HOME%\user_projects\domains\soa_domain\bin\setDomainEnv.cmd file

Search for *-Djavax.net.ssl.trustStore* and replace the value with the the trusted-certificates.jks file path that was generated by the keytool command. Also you need to edit -Djavax.net.ssl.trustStorePassword (if not available, create one). So finally your entries should look somewhat similar to this

```
-Djavax.net.ssl.trustStore=D:\oracle\Middleware\jdk160_29\bin\trusted-
certificates.jks -Djavax.net.ssl.trustStorePassword=welcome1
```

Once you are done with this edit, one step is pending, where you will tell the managed server that a custom keystore is setup and has to be considered.

This is done by opening the Admin Console(<adminHost>:<adminPort>/console --> Environments --> Servers --> click on soa_server1)

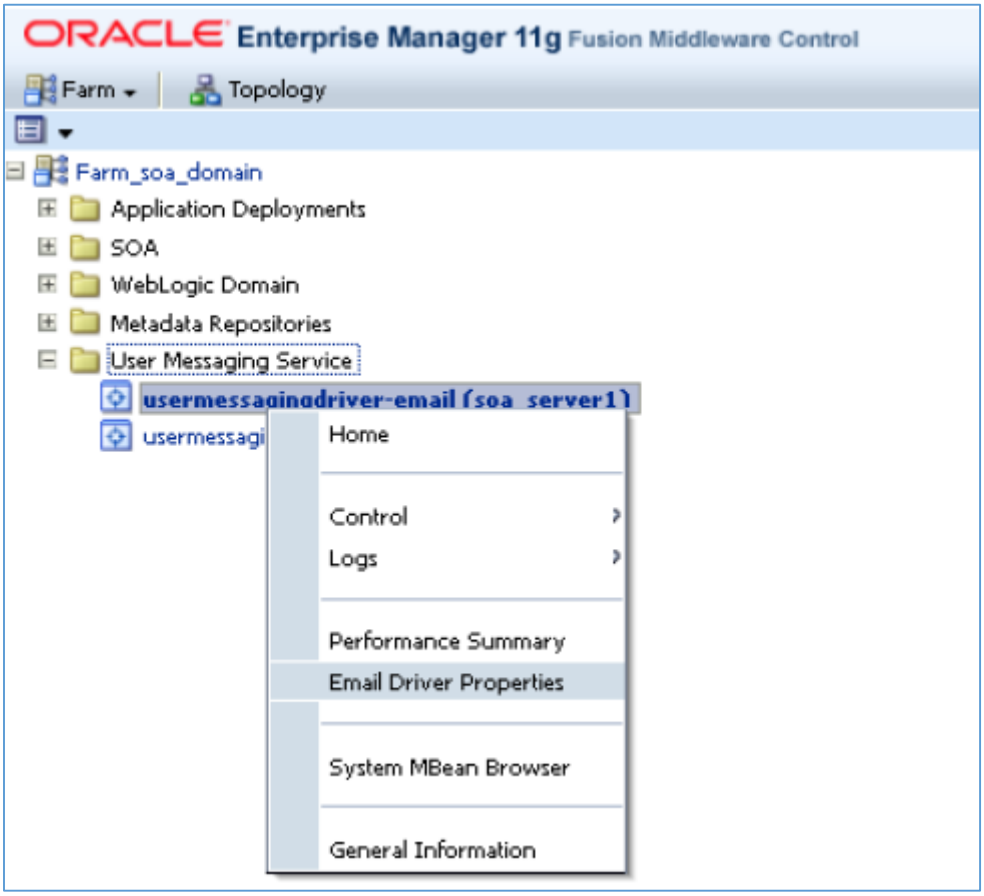
Click on Keystores, and change the Keystores to "Custom Identity and Java Standard Trust".

That finishes the certificates configuration.

Step 2: Configure email driver properties

This step configures email driver properties like email server details, incoming/outgoing email, passwords, etc.

For this, open EM, traverse as shown to open Email Driver Properties Screen.



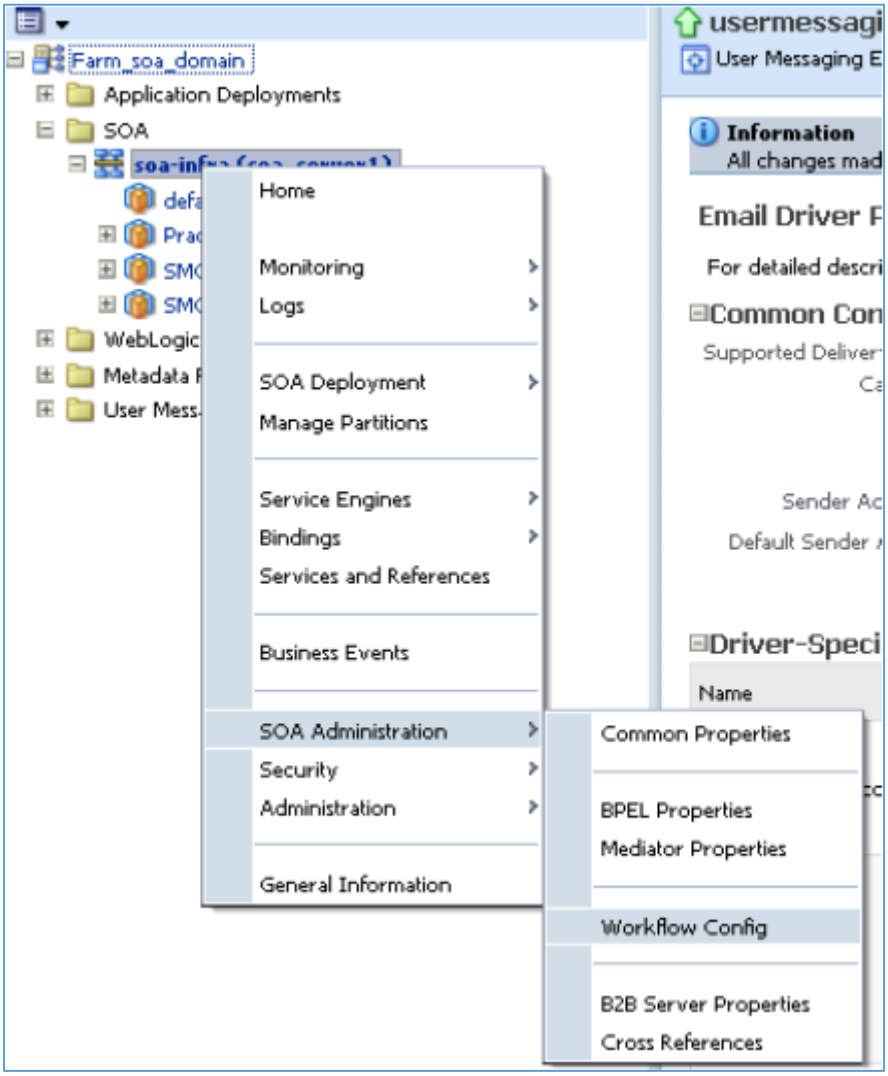
Configure the below mentioned properties

Property	Value	Comments
----------	**FOR SENDING MAIL**	----------
OutgoingMailServer	smtp.gmail.com	As mentioned, SMTP is used to send mails
OutgoingMailServerPort	465	This is the port gmail uses for SMTP
OutgoingMailServerSecurity	SSL	
OutgoingDefaultFromAddr	<mydefaultmailId@gmail.com>	The default FROM address (if one is not provided in the outgoing message).
OutgoingUsername	<mymailId@gmail.com>	
OutgoingPassword	Password of the gmail id	Option to be selected is "Use Cleartext Password"
---------similarly for----	**FOR INCOMING MAIL**	----------
IncomingMailServer	imap.gmail.com	As mentioned, IMAP is used to receive mails
IncomingMailServerPort	993	This is the port gmail uses for IMAP
MailAccessProtocol	IMAP	
IncomingMailServerSSL	No need to check this option	
IncomingMailIDs		
IncomingUserIDs		
IncomingUserPasswords		

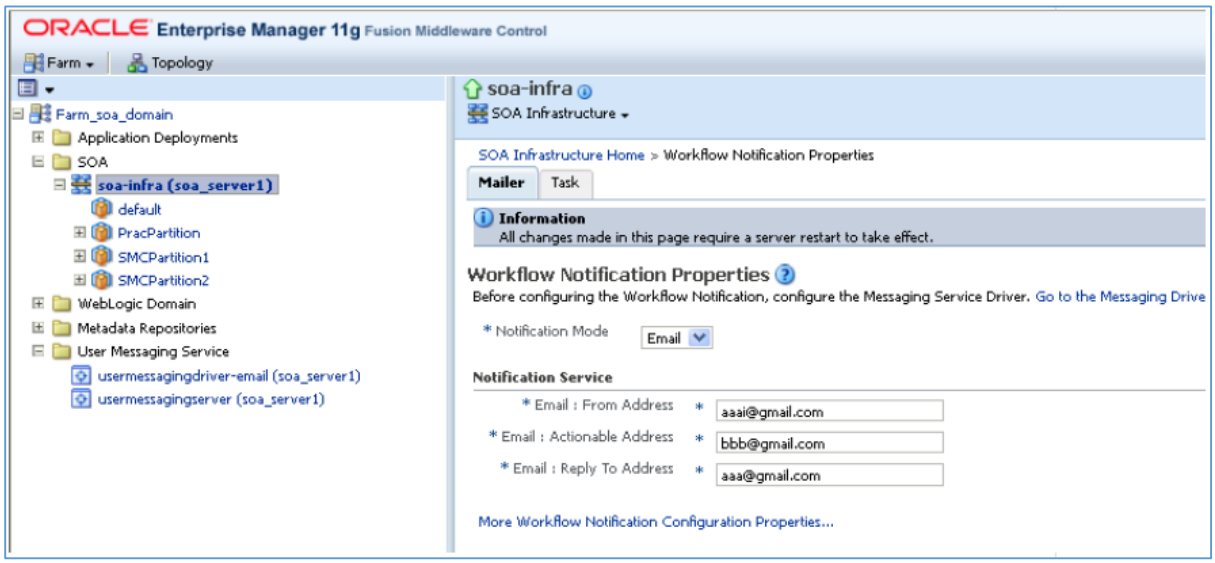
Step 3: Enable notification mode

This step lets the server know what mode to use for notifications. Since that we've configured email notification above, we'll enable EMAIL notification mode.

Traverse to *WorkFlow Config* in EM as shown in picture.



Setup the required values.



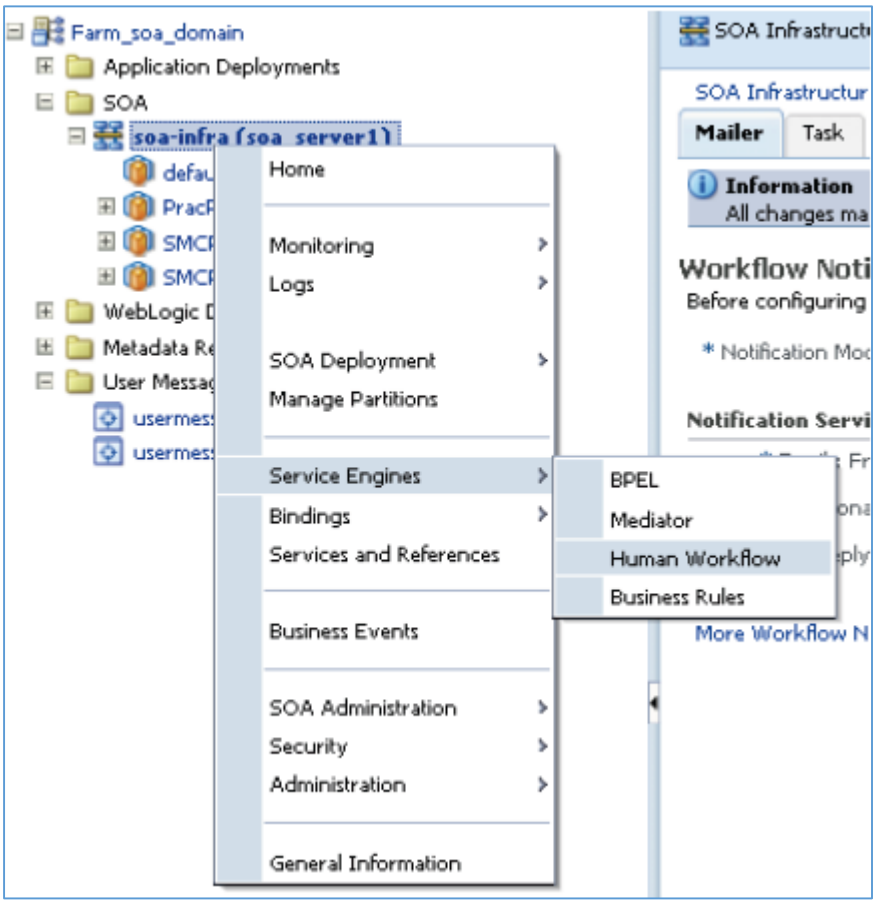
Restart the Admin (for step1) & Managed Server (for steps 2&3)

This sets up the required configurations.

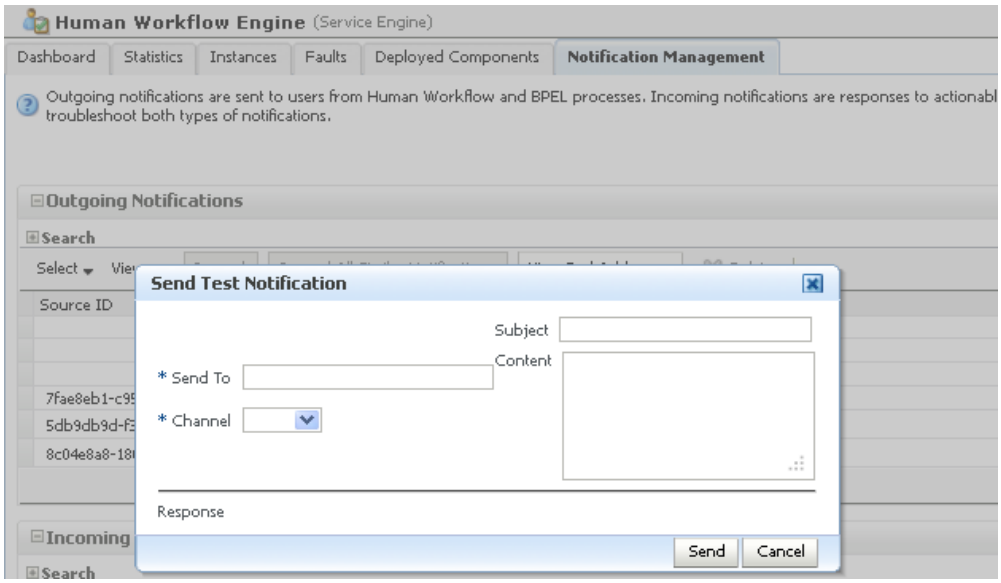
Step 4: Testing the configuration

The last thing that you need to do is test it.

Navigate to *Human WorkFlow* as shown



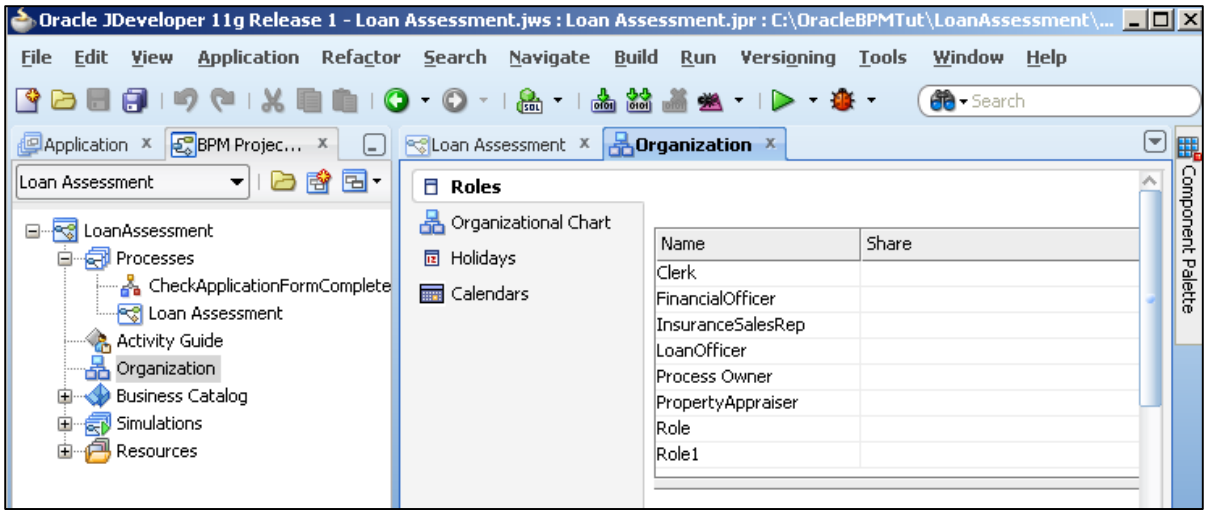
Notification Management --> Send Test Notification --> give details in the popup and check the mail.

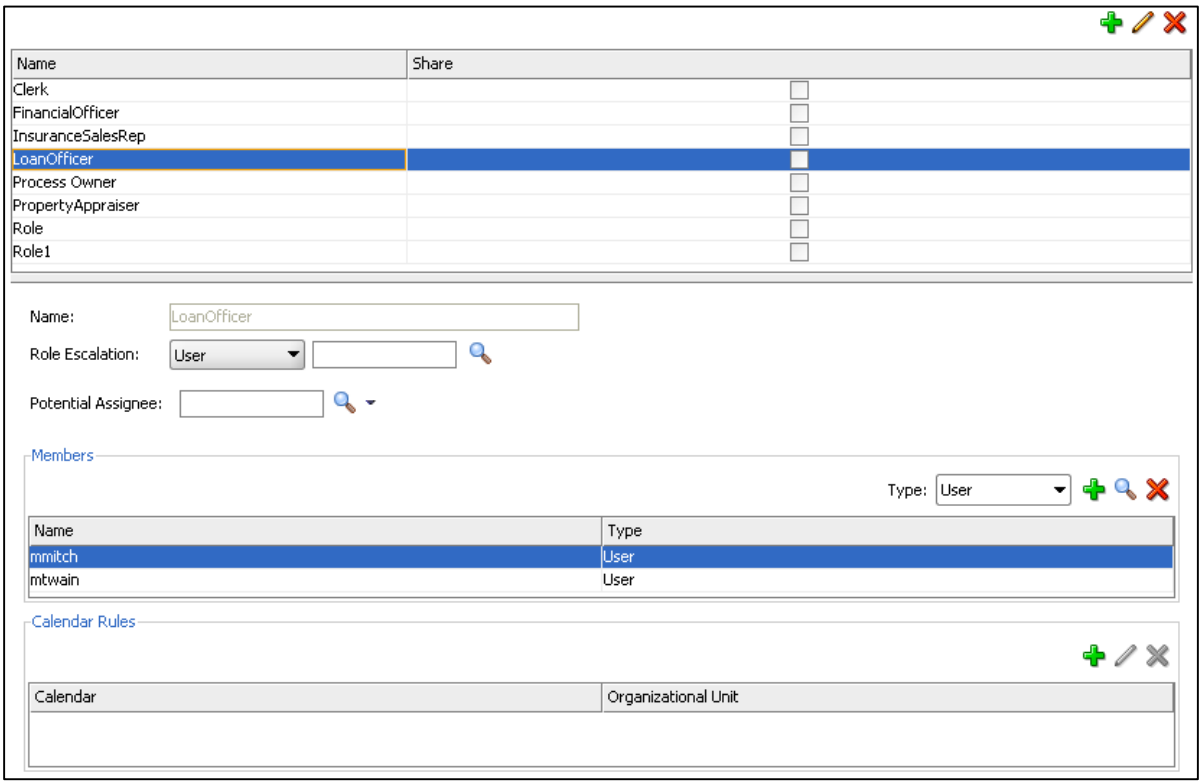


7.2 Implement Organization

In terms of human resources to execute interactive tasks, Oracle BPM provides a mechanism to map from design-time role model to run-time user model.

In BPM Project Navigator, click on Organization. The list of roles defined in the process is shown as follows. This environment allows us to add real users to each role and define availability hours (calendar) for users.



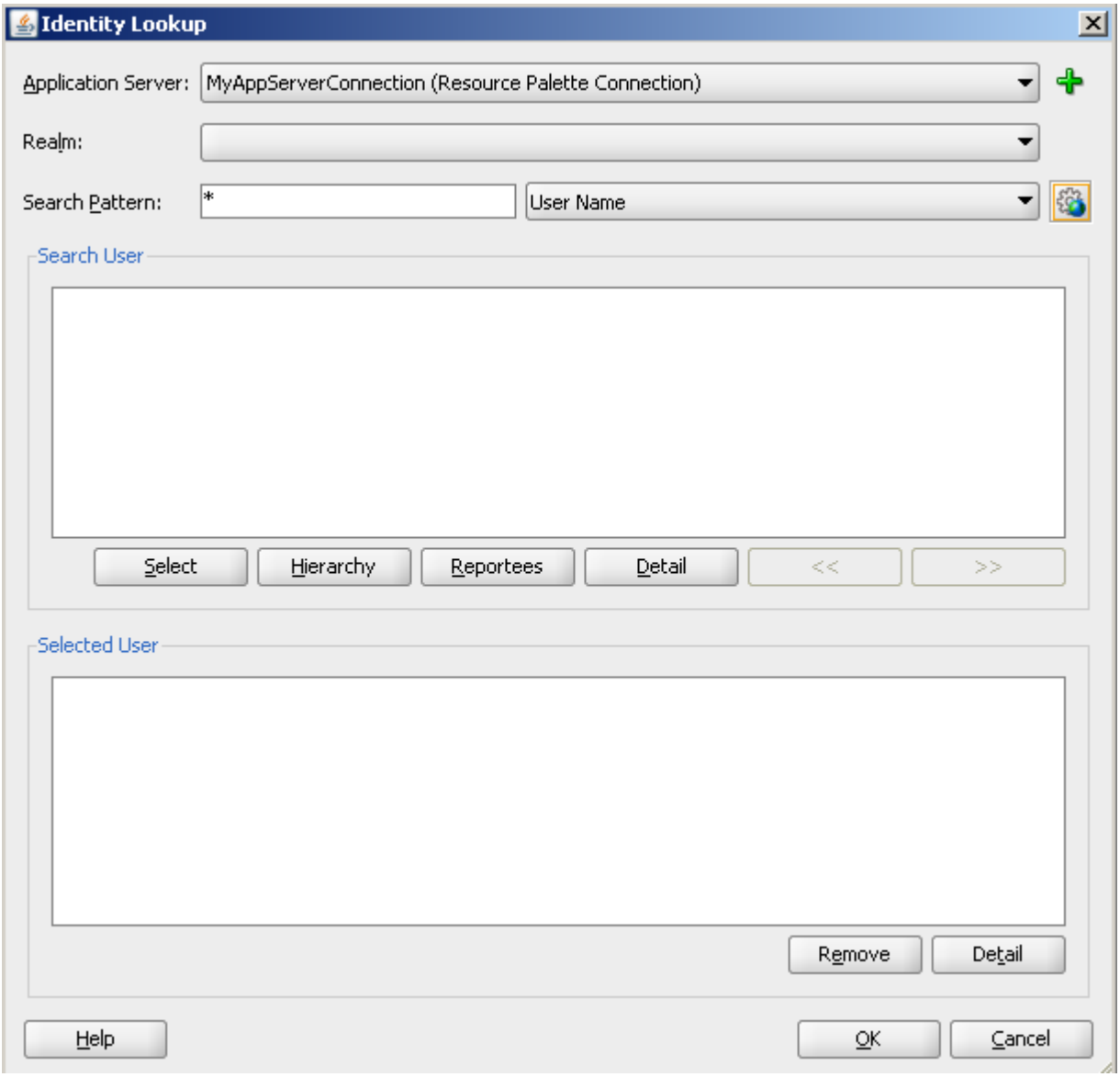


Each real user is granted a user ID in BPMS system by system administrator to access and execute human tasks on the system.

The screen below demonstrates how to map system users to design-time roles in BPMN diagram. The purpose is at run-time, these users will be routed to different activities according to routing rules. We'll define these rules later in the Human task configuration.

Add users to design roles

For this tutorial, a list of users have been created in the system. Click on plus sign button in member panel will open an "Identity Lookup" window.



Select server connection as created in Section 7.1.1. Enter search keyword or search all.

List of users will be listed. These users include those demo users we have installed in section 3, step 9. Select users you want to add to the role. Repeat for each role.

Selected users for the tutorial are as follows:

Role	User
Applicant	jcooper
Financial Officer	cdickens
Loan Officer	mmitch, mtwain
Property Appraiser	rsteven
Process Owner	wfaulk
Insurance Sales Representative	fkafka

Note: we need to have at least two Loan Officer users because there is a requirement of separation of duties. The last step “Take Final Decision” must be done by a Loan Officer who is different from the Loan Officer who performed previous steps.

7.3 Implement Data Objects

In BPM Project Navigator, navigate to Business Catalog > BusinessObjects. This function allows us to add data object types. Once these types are defined, we can add as many data objects as we want to the process.

Thus, there are two main steps for implementing data objects:

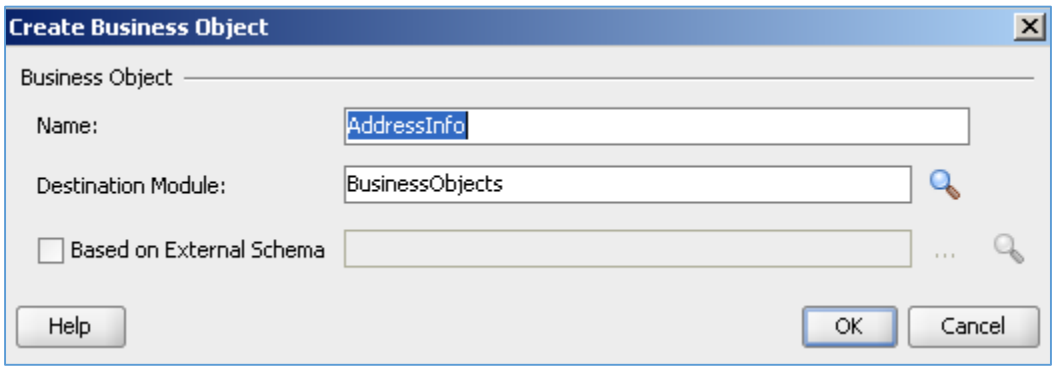
Step 1: Define Business Objects
Step 2: Create Data Objects

Step 1: Define Business Objects

In BPM Project Navigator, expand Business Catalog > BusinessObjects.

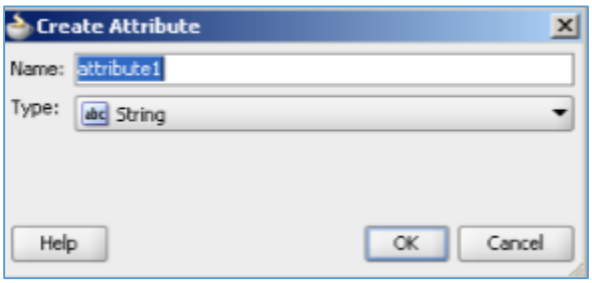
Right click on BusinessObjects, select New > Business Objects.

In the “Create Business Object” popup, provide business object name (without spaces), e.g. AddressInfo. Click OK.

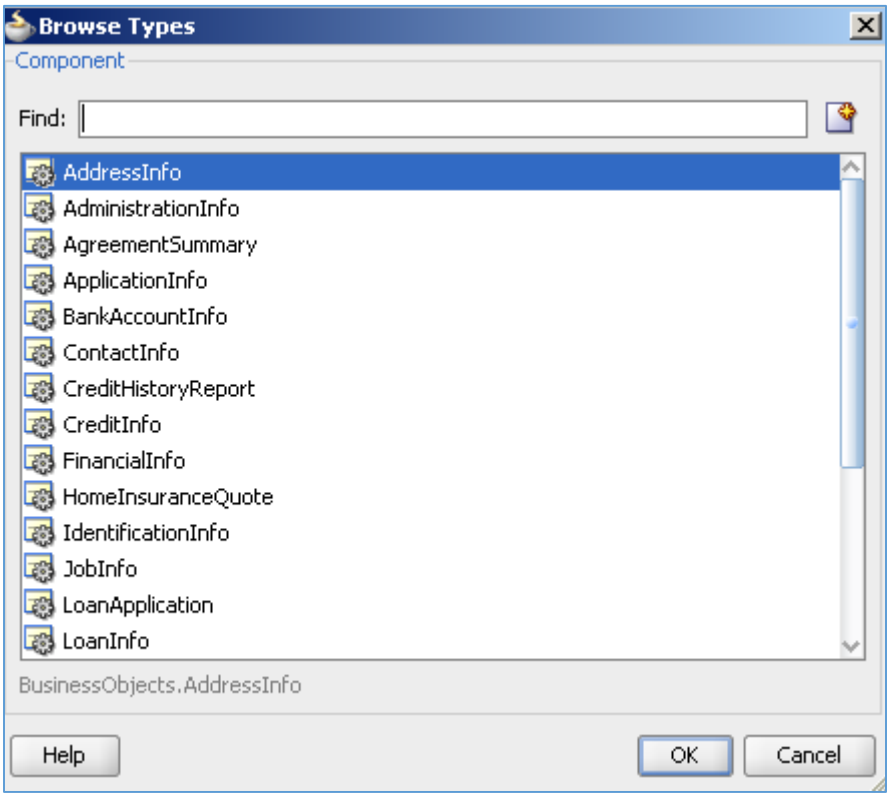


The business object will be created and shown in Business Catalog > BusinessObjects. Its detailed information is also shown as below (or double-click to open it if it is not shown).

Click on the green plus sign button to add a new attribute.

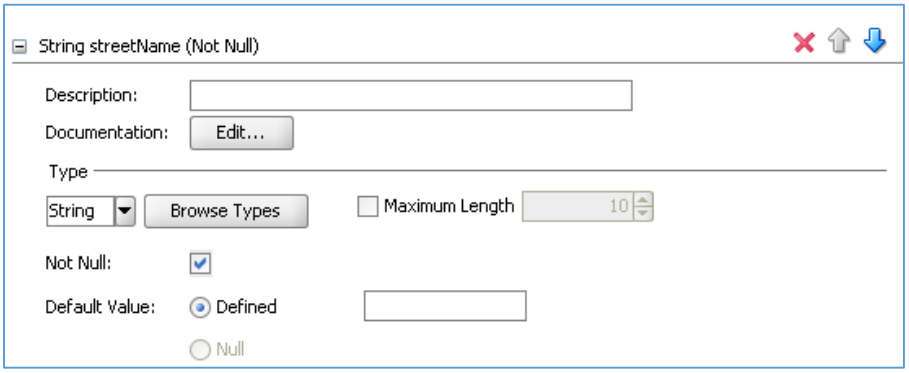


In the “Create Attribute” popup, select a name and type of the attribute. If the attribute is not primitive types (e.g. integer, string), we can select <Component> and <Array>. For Component or Array, we can select further a business object as base element as shown below. We need to select from XSD files which contain type definition for the object.



Click OK in the Create Attribute popup when you have filled in the Name and Type field.

The attribute will be added and shown in the detailed information of the business object.



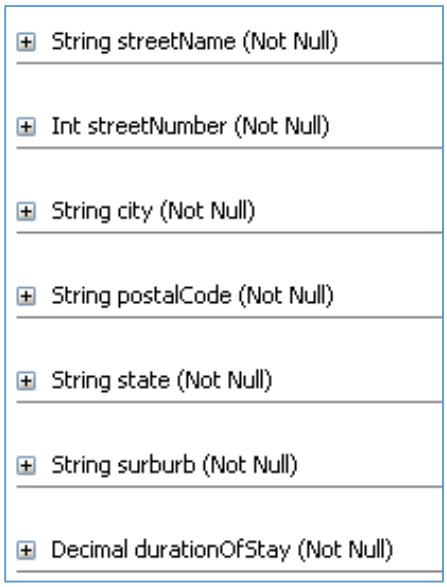
You can expand it to see more attribute properties. You can set some property values: type, maximum length, not null, default value, etc. Click Save on the menu to save the changes.

Repeat the same procedures above to add other attributes to the business object.

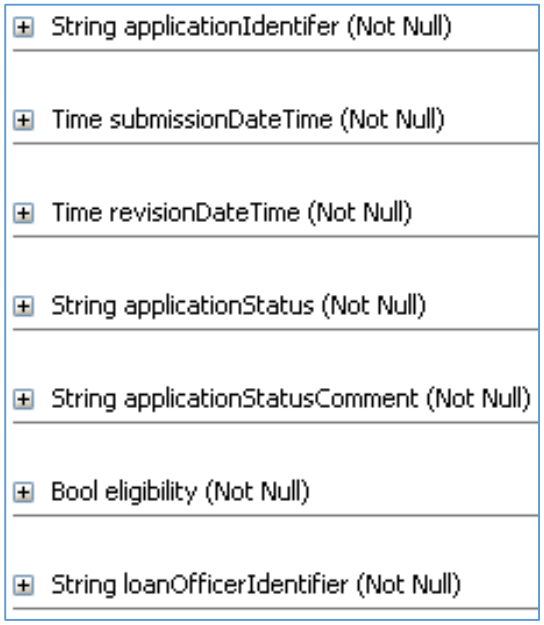
That's the procedure. Now, let's create all business objects needed for the process based on the process scenario.

Below is list of all business objects for this tutorial. They are listed with simple business objects first (object with primitive attributes), then compound business objects, (object with attributes can be another business object).

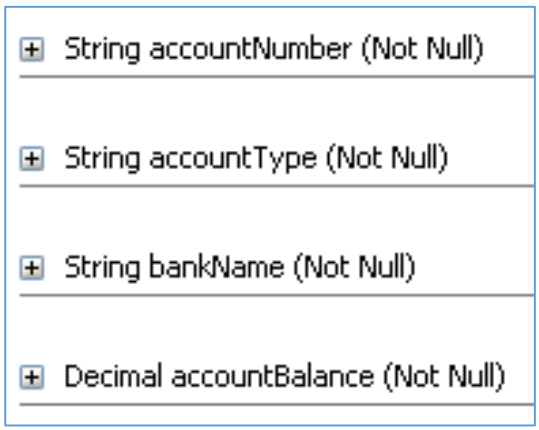
AddressInfo



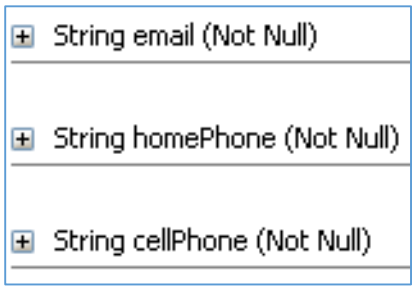
AdministrationInfo



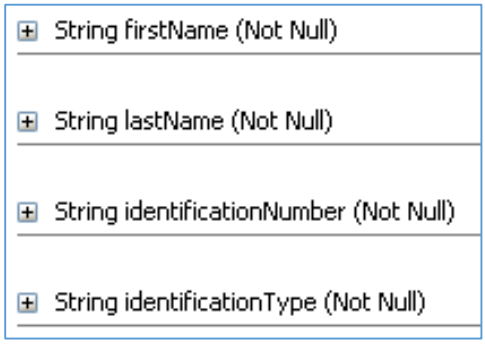
BankInfo



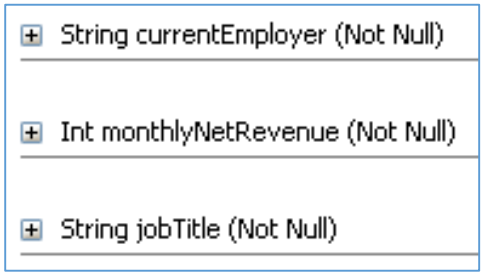
ContactInfo



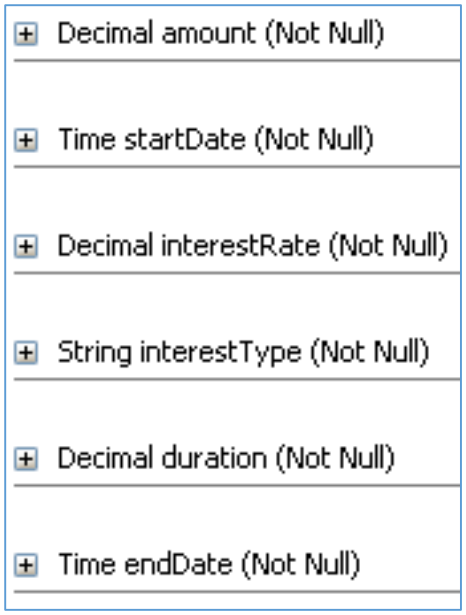
IdentificationInfo



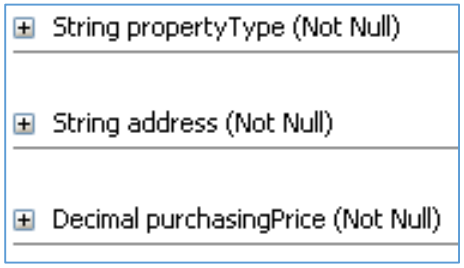
JobInfo



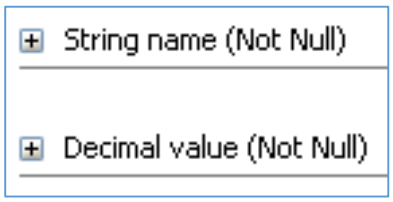
LoanInfo



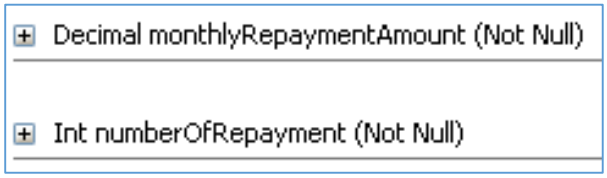
PropertyInfo



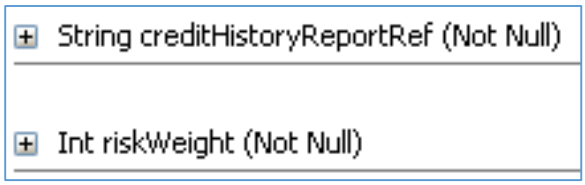
SurroundingPropertyInfo



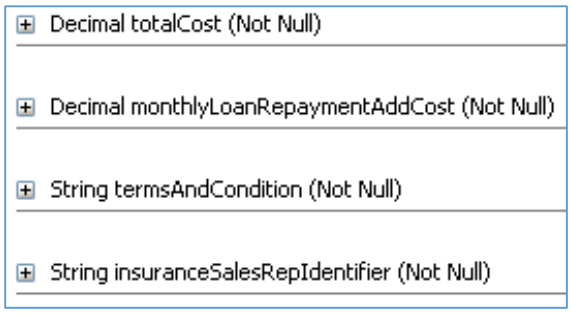
RepaymentAgreement



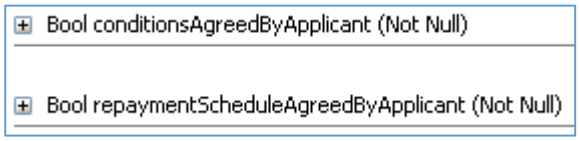
RiskAssessment



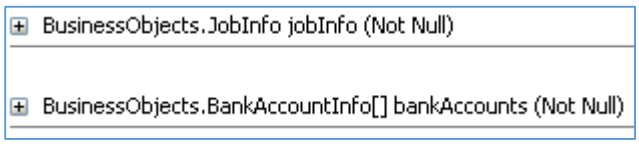
HomeInsuranceQuote



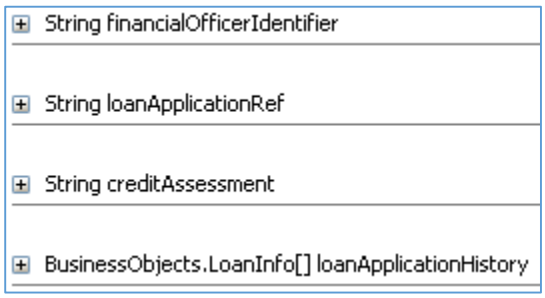
AgreementSummary



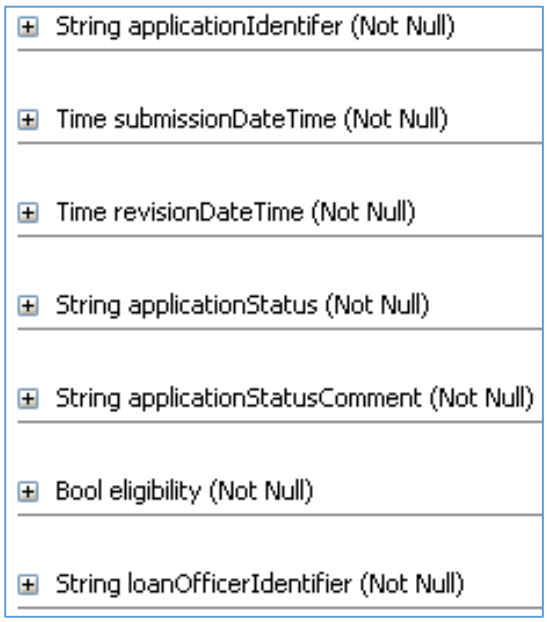
FinancialInfo



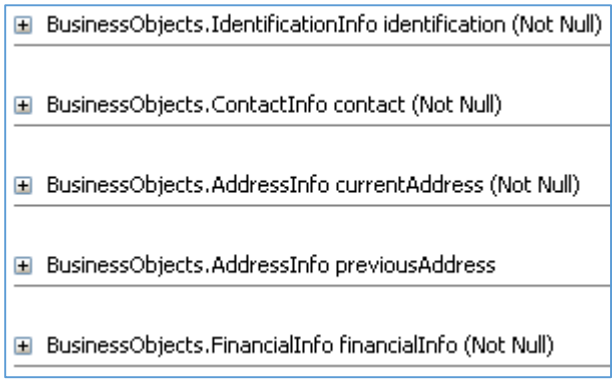
CreditHistoryReport



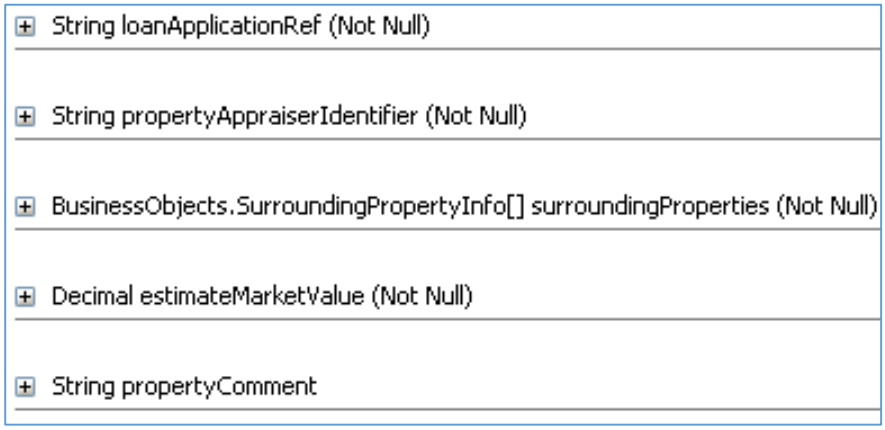
AdministrationInfo



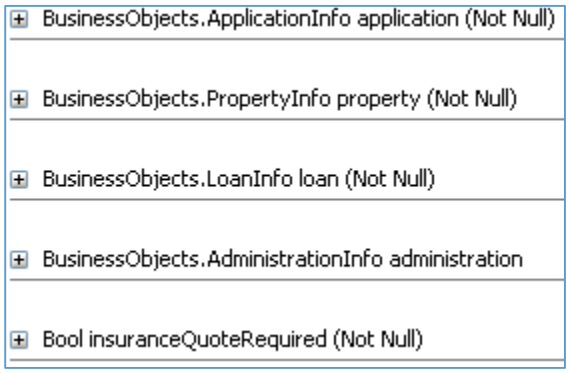
ApplicationInfo



PropertyAppraiserInfo



LoanApplicationInfo

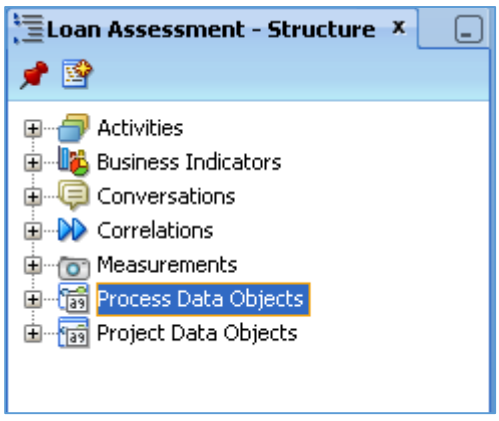


We have finishes the business objects definition, i.e. the data object types. We can now create data objects for our process.

Step 2: Create Data Objects

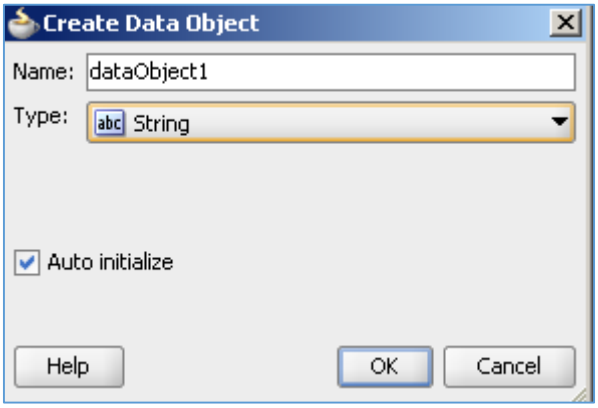
In BPM Project Navigator, navigate to Processes > Loan Assessment process. Click on Loan Assessment process.

Note that the Structure pane shows the structure of this process as follows.

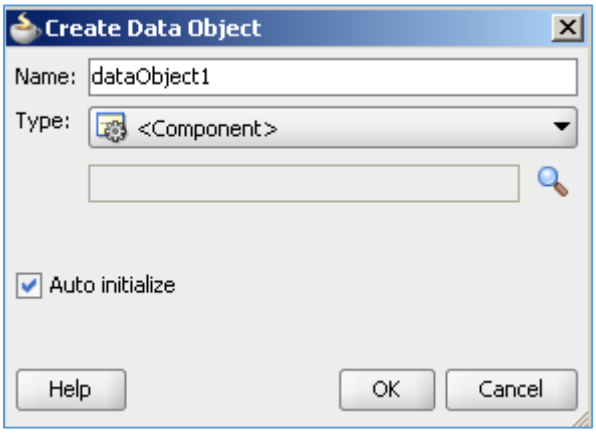


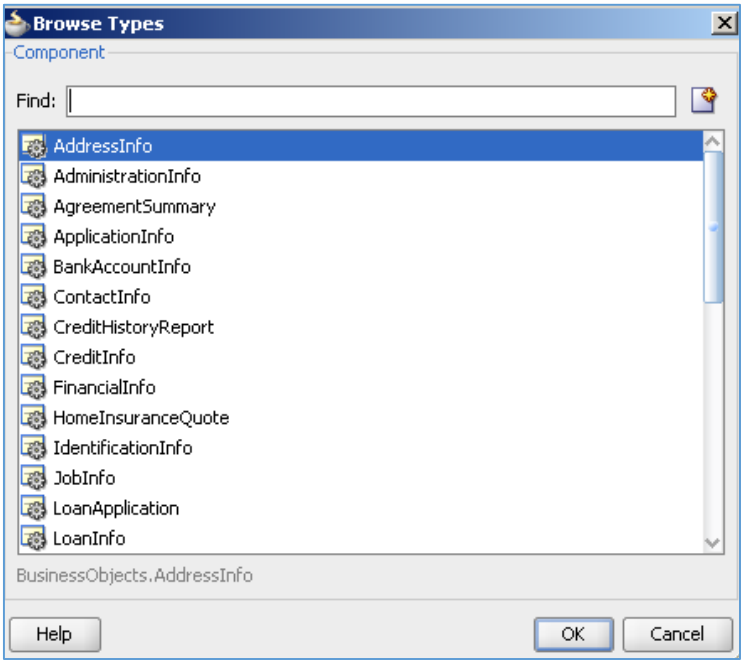
Right click on Process Data Objects, select New to create a new data object for the process.

In the Create Data Object popup which looks the same as the Create Attribute for business object, we can enter value for Name and Type field.



If it is a data object of the type defined above, select Component in Type field, then click on the Search button to select from list of Business Objects we have created in previous step. Once done, click OK in the Create Data Object popup.





Let's follow the above procedures to create the following process data objects.

Data Object Name	Type	How to use it
loanApplication	LoanApplication	Used to contain loan application form
propertyAppraisal	PropertyAppraisal	Used to contain property appraisal information
repaymentAgreement	RepaymentAgreement	Used to contain repayment agreement
homeInsuranceQuote	HomeInsuranceQuote	Used to contain Home Insurance Quote if applicable
agreementSummary	AgreementSummary	Used to contain the information of Agreement Summary between the Bank and the Applicant
riskAssessment	RiskAssessment	Used to contain Risk Assessment by the Financial Officer
creditHistoryReport	CreditHistoryReport	Used to contain Credit History report information
formCompleteCheckOutcome	Boolean	Used to contain the result of form completeness check done by the system
quoteRequestCheckOutcome	Booelan	Used to indicate whether the Applicant request any home insurance in the submitted application form
eligibilityCheckOutcome	Boolean	Used to contain the result of eligibility check done by the Loan Officer
applicationApprovalOutcome	Boolean	Used to contain the result of final decision done by the Loan Officer

7.4 Implement Human Tasks

Human Task is an implementation of every user task in the BPMN diagram. In this section, we will create human task for every user task.

Human Task is normally associated with a kind of human interaction, for example, via desktop forms, web pages, or mobile screens. In this tutorial we use web forms.

Firstly, we need to design how the human interaction takes place, including the look and feel of the web page. Oracle defines the following properties of a human task.

- Title: title of the task to be displayed in the task list
- Priority: priority of the task
- Outcome: is it a submit, or approval/reject, or OK, Yes/No, Accept
- Owner: owner of the task
- Data: data used by the task to display to users or for computation.
- User Assignment: who can access to perform the task

The above properties are encapsulated in a human task object defined by Oracle BPM. At run-time it is managed by Oracle Human Workflow Engine.

User interface design, including layout and look and feel, is part of the human task implementation. Oracle BPM is accompanied with Oracle ADF as a user interface framework. Oracle ADF is a huge library enabling developers to implement not only web pages but also desktop forms and mobile interface.

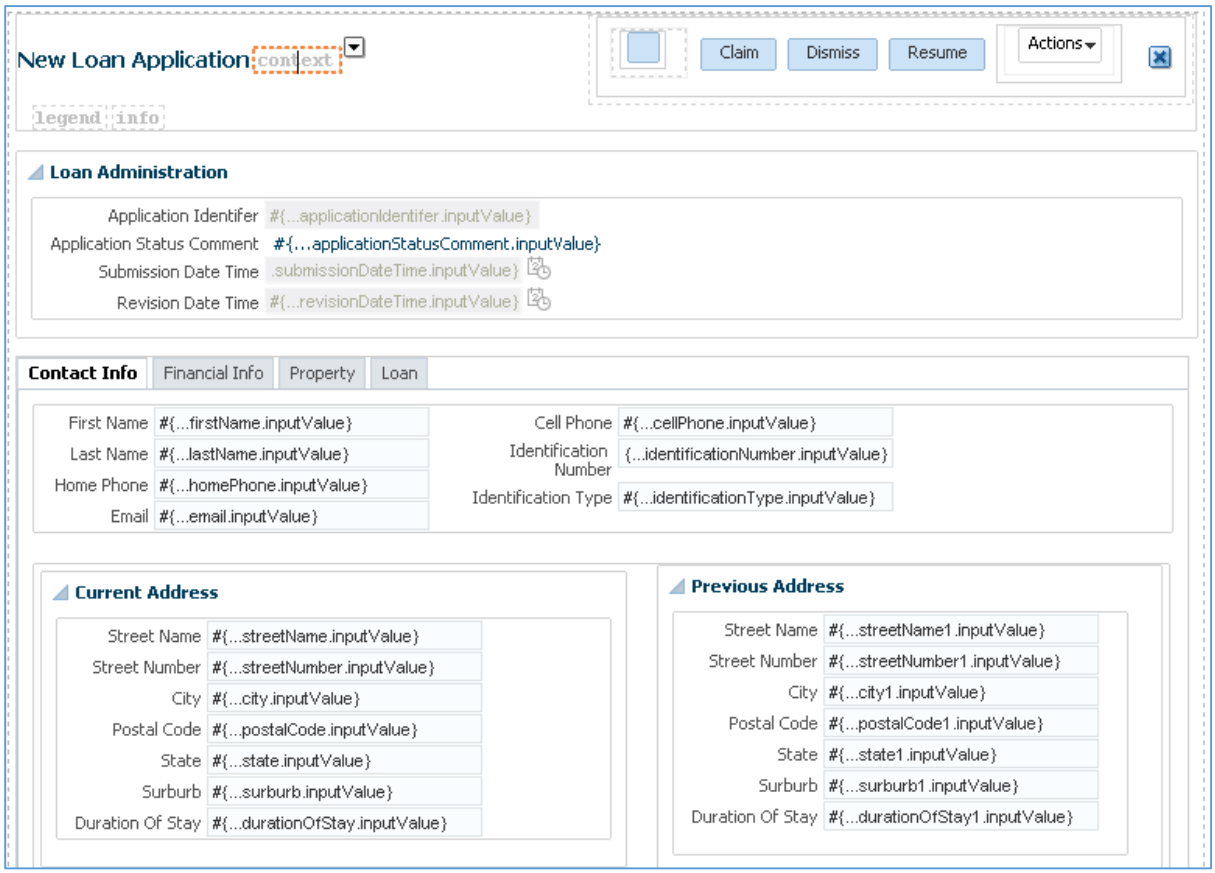
In addition, we also need to design how data is exchanged between the web page and the data source (database, XML, etc.). In this respect, Oracle provides support with Data Control and Binding framework, including implementation library and design-time manipulation. We'll design using the provided facilities without having to implement things from scratch.

The following section demonstrates the process of implementing the "Enter Application Form" task in details. Then, you can do similarly for other forms.

7.4.1 Enter Application Form

The web form we want to create is shown. It has the following features:

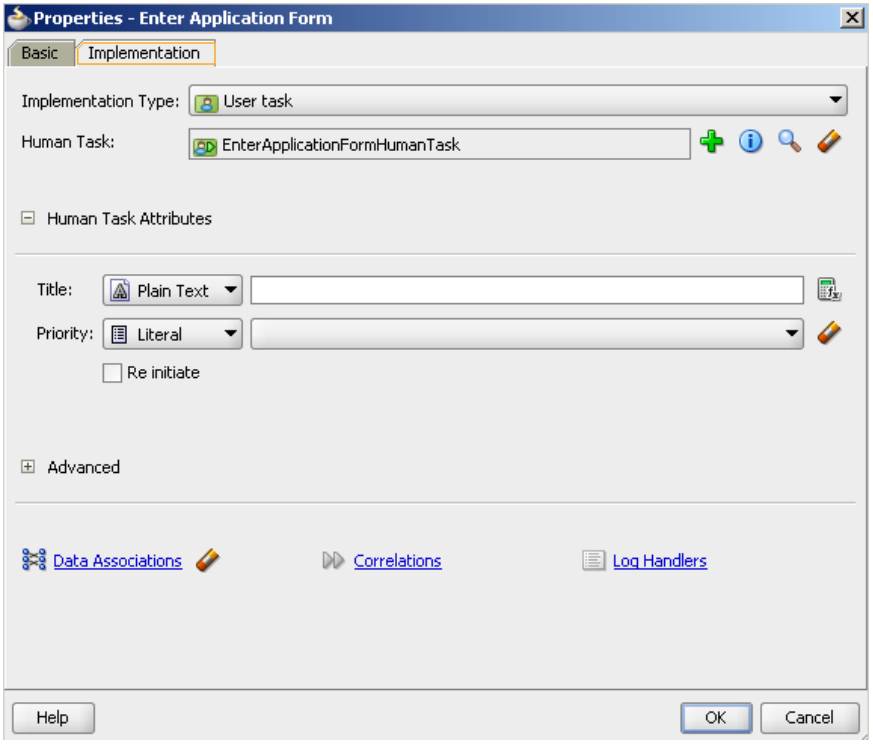
- A title and range of process-related buttons are at the top of the page.
- The Loan Administration information is shown next, consisting of application identifier, application status comment, Submission date time, revision date time.
- The main content is consisted of tabbed pages
    - The first tab shows Contact information with personal information items at the top. The current and previous address are shown under in two blocks
    - The second tab shows Financial Information with Employment information first and List of bank accounts.
    - The third tab shows Property information
    - The fourth tab shows Loan information. The insurance quote required or not is a check box in this tab.



Implement Human Task object

First, we'll set some configurations for Enter Application Form human task.

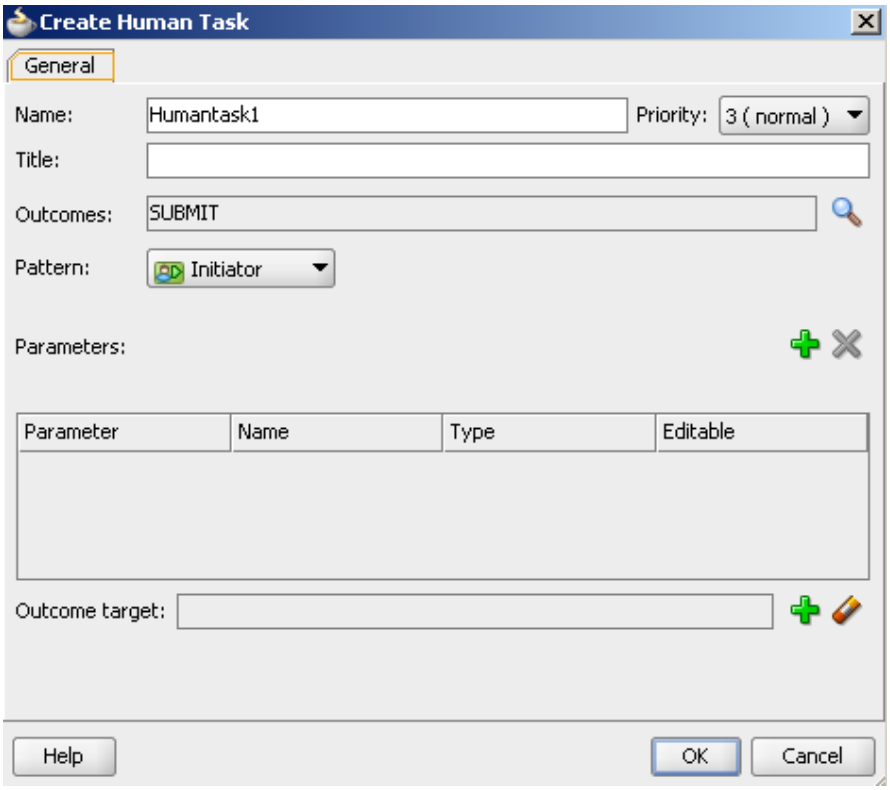
Right click on Enter Application Form user task, select Properties > Implementation



Click the green plus sign button to add a new Human Task.

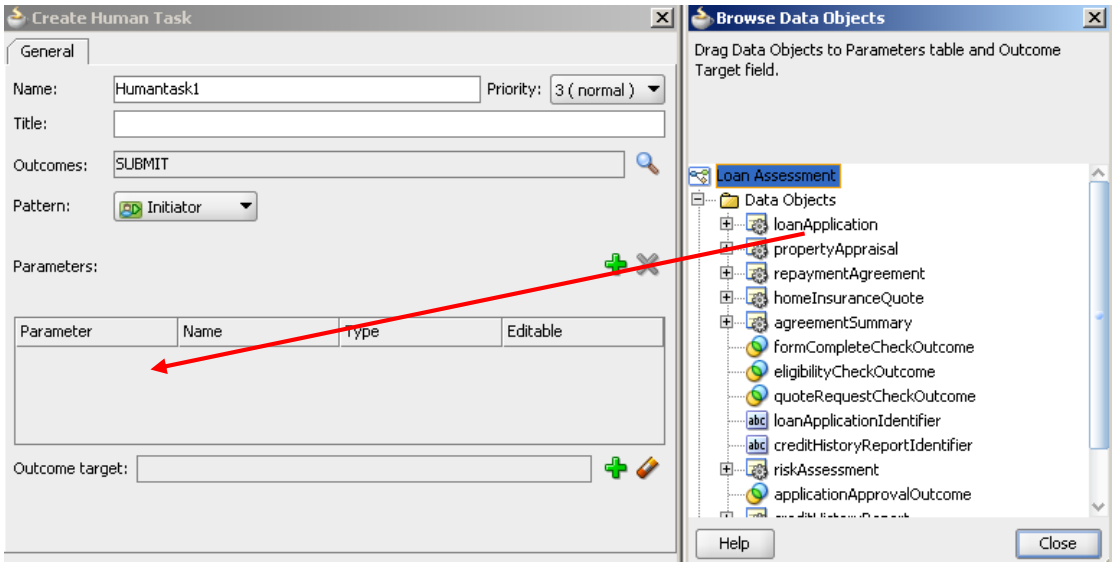
In the pop-up window:

- Enter name and title of the task, leave
- Leave priority as default
- Select an outcome. Here select SUBMIT since this is the first form that user will submit.
- Patterns: select Initiator since this is the first form user will initiate in the process.



Next, we will add data variables for the task and then assign (map) the process variables to task variables and vice versa. This has to be done because a task can only access task variables, not process variables. The only way of data exchange between tasks is through process variables. For each task, it is common to assign process variables to task variables before the task is run and then assign task variables back to the process variables after the task run has finished.

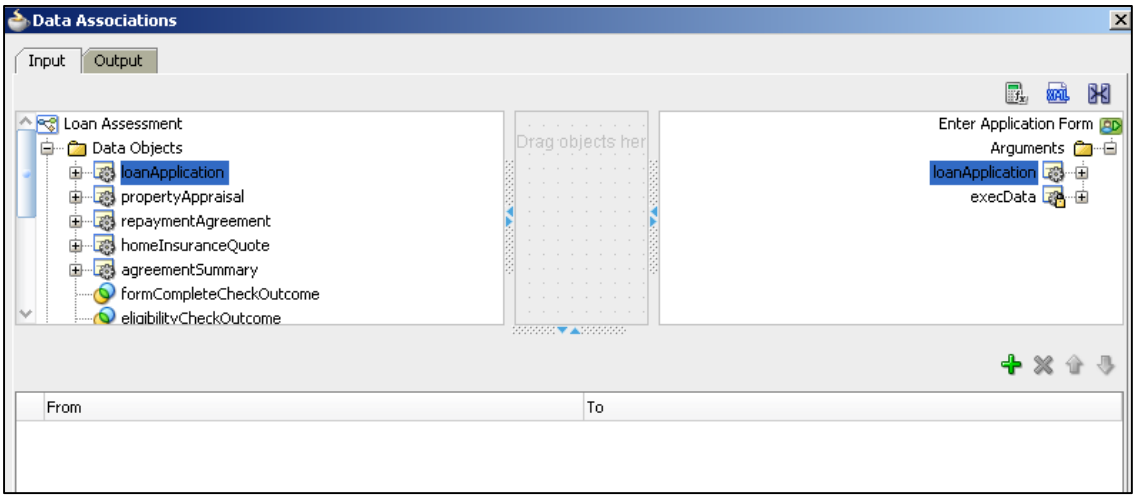
In Create Human Task window, the Parameters section lists defined task variables. Click the green plus sign button to add a new task variable. It opens a Browse Data Objects window which shows list of data object types which were defined as business objects. It allows us to create new task variable from these data object types.



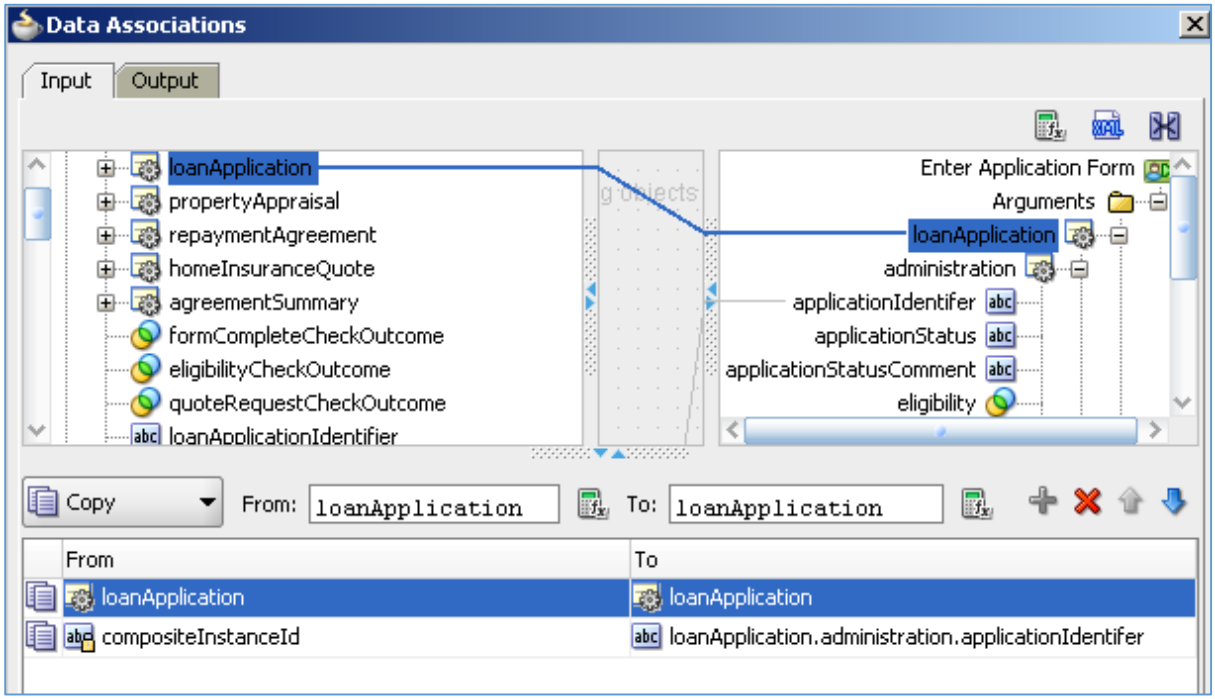
Drag **loanApplication** variable on to the Parameters table, select Editable. We need to add a Loan Application variable here because all information shown on the form is contained in this data object type. If we want to show other data items in the target form, we need to add a variable of an object type that contains the relevant information in the same “drag and drop” manner.

Select OK in Create Human Task window and back to Properties window.

Click Data Association to open Data Association window. There are Input and Output mapping tab. In the Input tab, the process variables are on the left-handed pane and the task variables are on the right-handed pane, and vice versa in the Output tab.

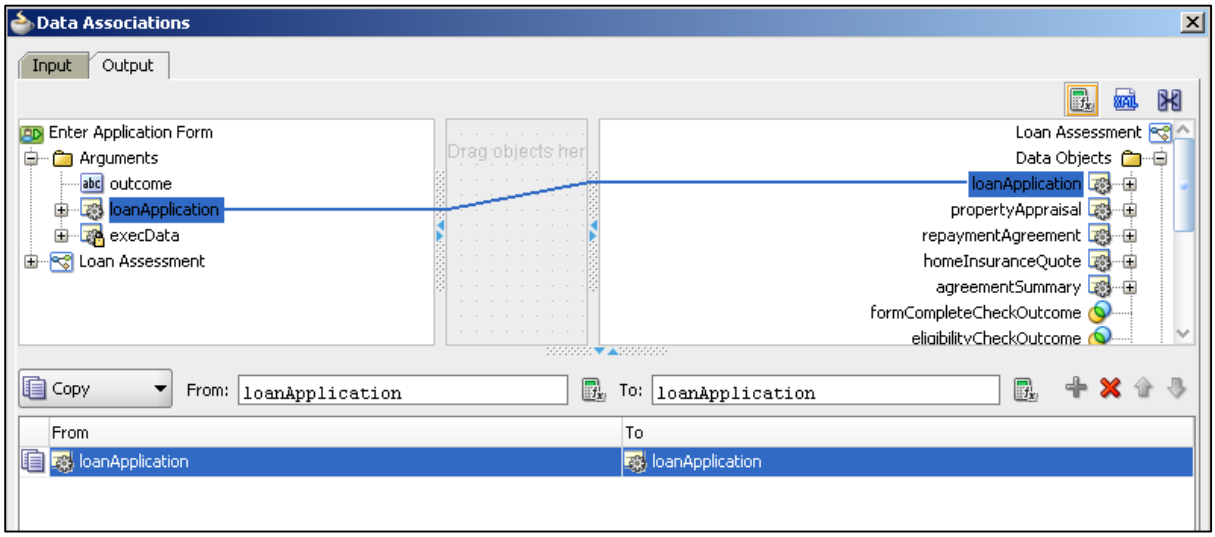


To map (mean assigning) from a process variable to task variable, drag a process variable and drop it over the corresponding task variable. A line will be drawn connecting two variables to indicate the mapping, as shown below.



Note: the mapping here shows that the System Composite Instance ID will be used as the Application Identifier number.

Perform mapping in the Output table as shown below. It means the loan application after being filled in by the user on the web form will be transferred and written back to the process variable.



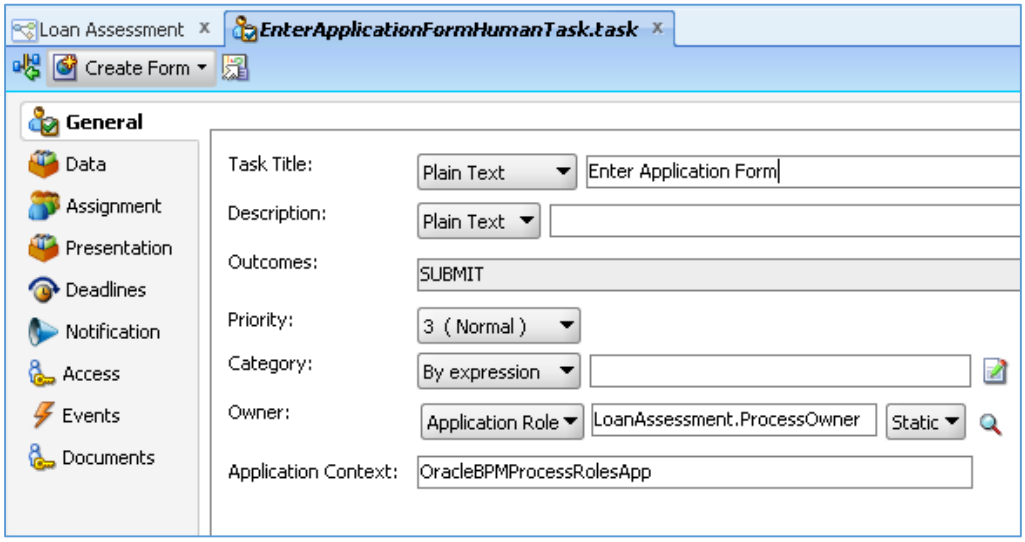
Then select OK button in the Data Association window to finish data mapping.

Select OK in the Properties window to finish human task implementation.

After Data Configuration, we can open a Human Task configuration to view more details.

In BPMN diagram, right click on Enter Application Form activity, select Open Human Task. The human task details are shown with different sections: General, Data, Assignment, Presentation, etc. as follows.

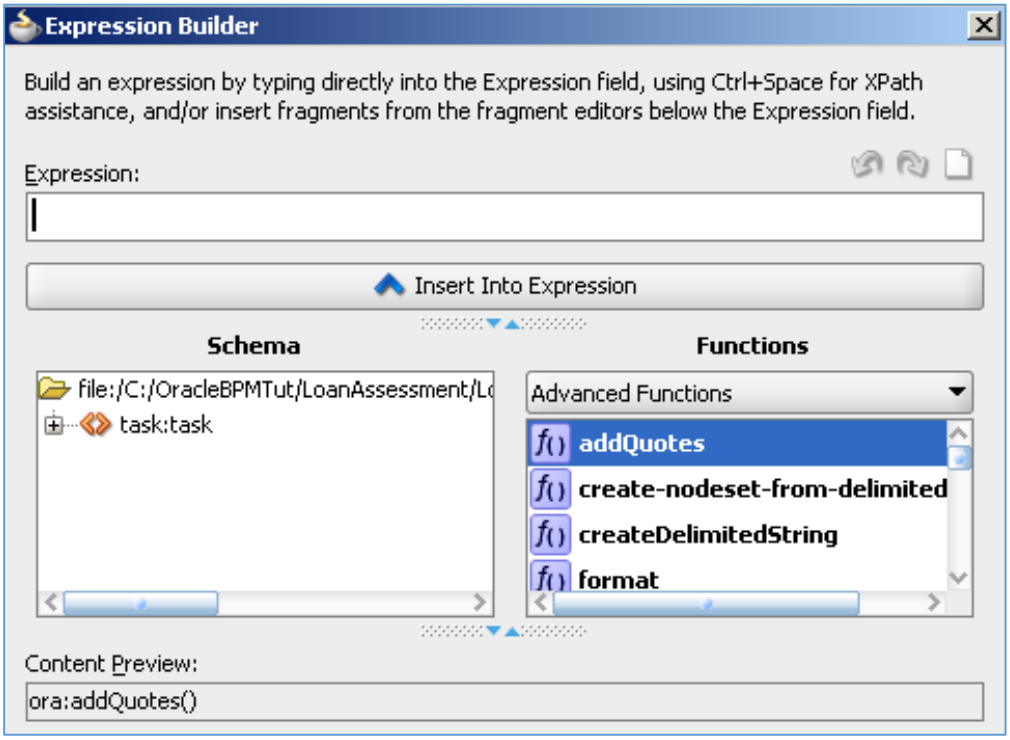
In Data section, the Loan Application variable has been added in previous step.



We'll change how the title of this task is displayed in the General section and keep other sections below the Data section unchanged with default settings. The task title appears in the task list of a process user which can have many other tasks. Therefore, a task title should show the name of the task and the loan application identifier so that a process user can quickly know without opening it.

Click on General section. Click on Plain Text drop-down box. It'll show Plain Text, Text and XPath, Translation. Click on Text and XPath. Then click on the Edit button on the right of the title text field (the button tooltip is "Build title from XPath…").

The Expression Builder pops up as below.



Copy and paste the following text into the Expression box.

```
concat('Enter Loan Application - ID =',
/task:task/task:payload/ns2:LoanApplication/ns2:administration/ns0:applicat
ionIdentifer)
```

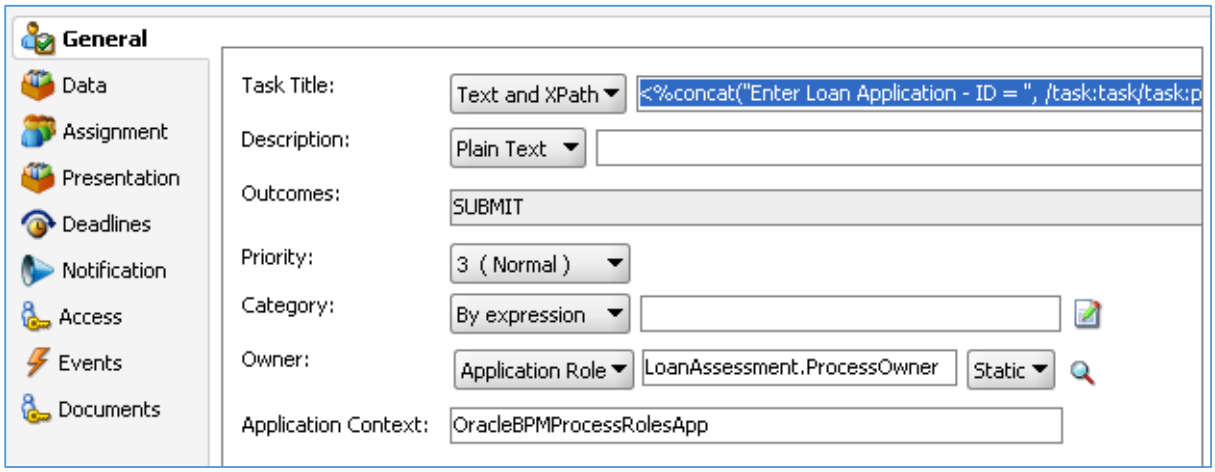
You can create the above expression by playing a little bit with the Expression Builder: select drop-down button under Functions, scroll down and select String Functions, double-click on “concat” to add it to the Expression box. This function will connect all parameters into one string. As the first parameter, type “Enter Loan Application – ID = “ between the two round brackets (including two quote characters).

Type a colon as parameter separator.

To add a second parameter which should be the loan application identifier, we need to select it from the task information schema which Oracle BPM created for this task. This schema is displayed in a tree view under Schema. In this schema, you should note that task:payload node contains the task variable (Loan Application in this case) whereas other nodes are task metadata automatically added by Oracle, i.e. title, creator, owner, priority, etc. Expand task:payload and select LoanApplication > administration > applicationIdentifier. Double-click to add it as second parameter of concat function, after the colon. It is in form of an XPath).

Select OK in the Expression Folder.

The Task Tittle in the General tab will be updated.



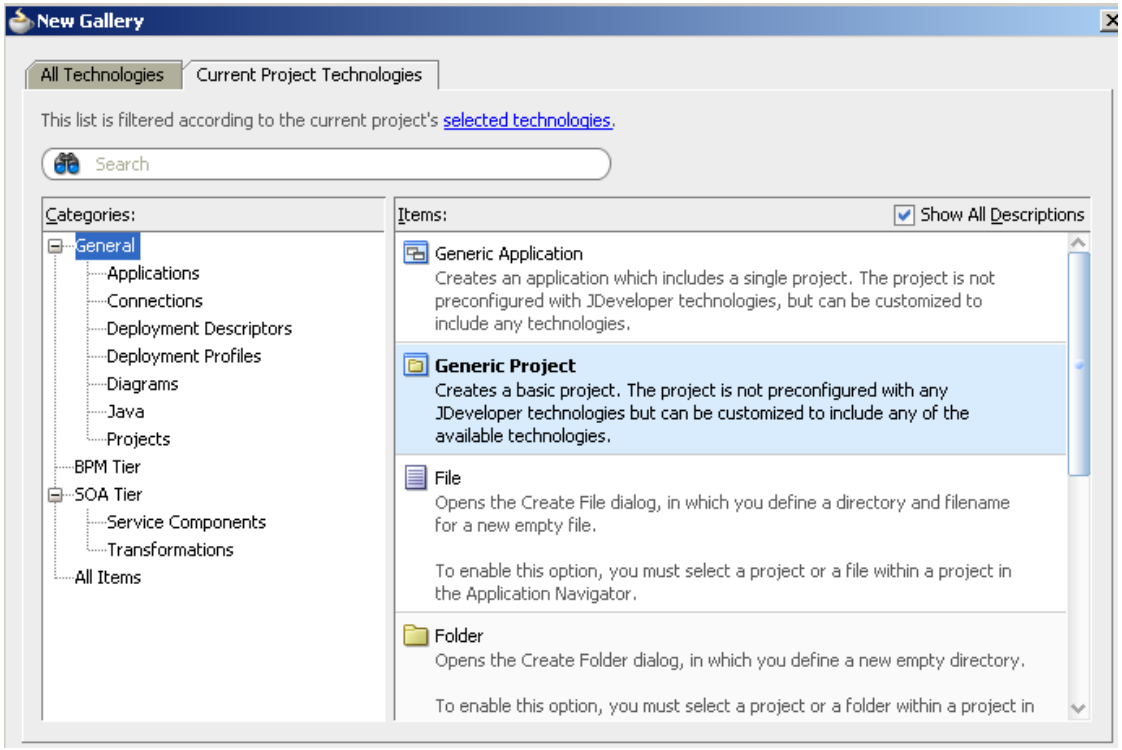
Implement Web Form

In this stage, we’ll create a web page form for user to enter loan application.

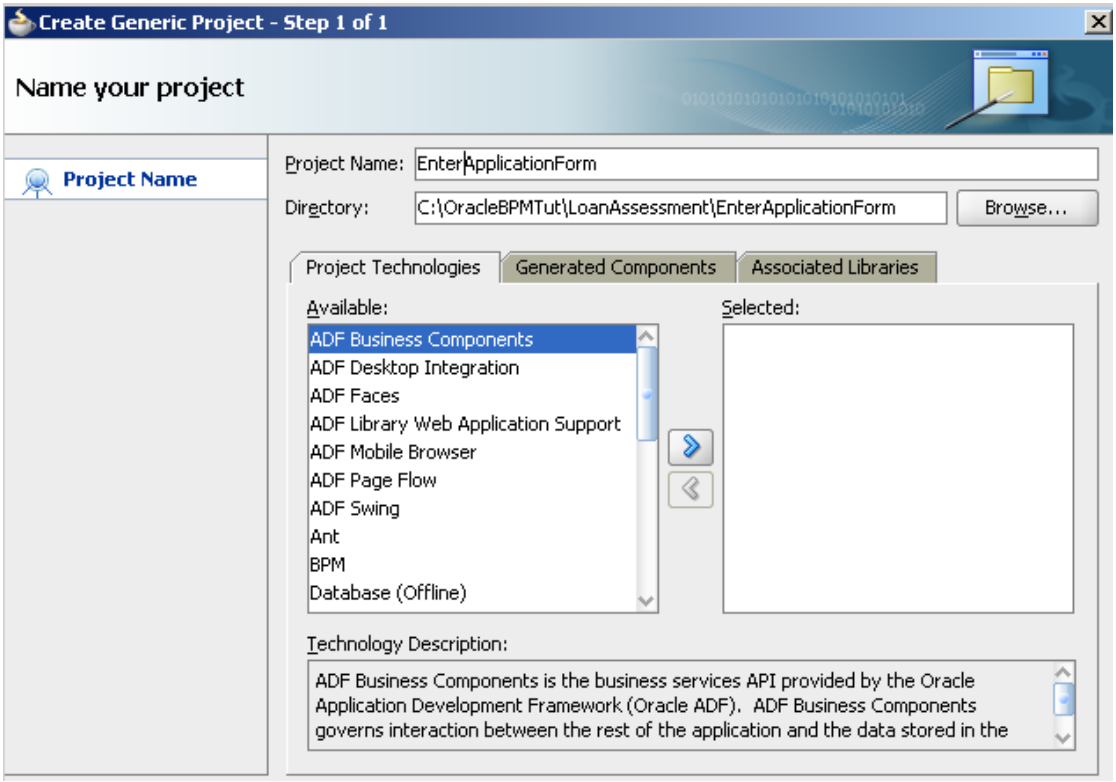
As mentioned in section 2, Oracle leverages Oracle ADF as a framework for user interface of Oracle BPM. So, you’ll create an Oracle ADF project and connect it with the human task as configured in Oracle BPM project above. There is a way to automatically generate ADF form from a human task (there is a button Create Form in an opening human task window). However, we will create an ADF from manually which enables us to design and customize the form layout.

In an opening Oracle BPM application, select File menu > New.

In the New Galllery window, select Generic Project and click OK.



In Create Generic Project window, enter “EnterApplicationForm” as project name. Leave the Directory as default which is a subfolder with the same project name will be created under the Oracle BPM application folder. Then click on the Finish button.

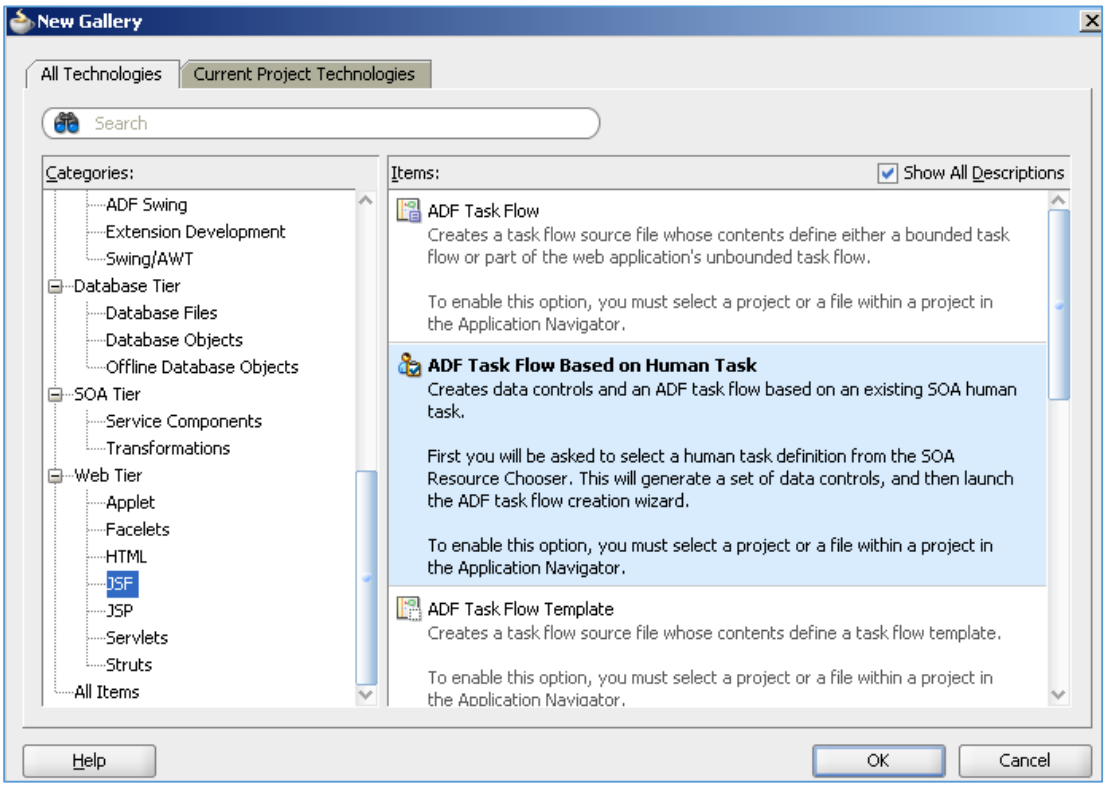


A new project will be created and added as part of the Oracle BPM application: a new subfolder is created under the application folder, a new folder is shown in the Application Navigator.

In the next step, we'll add an ADF form

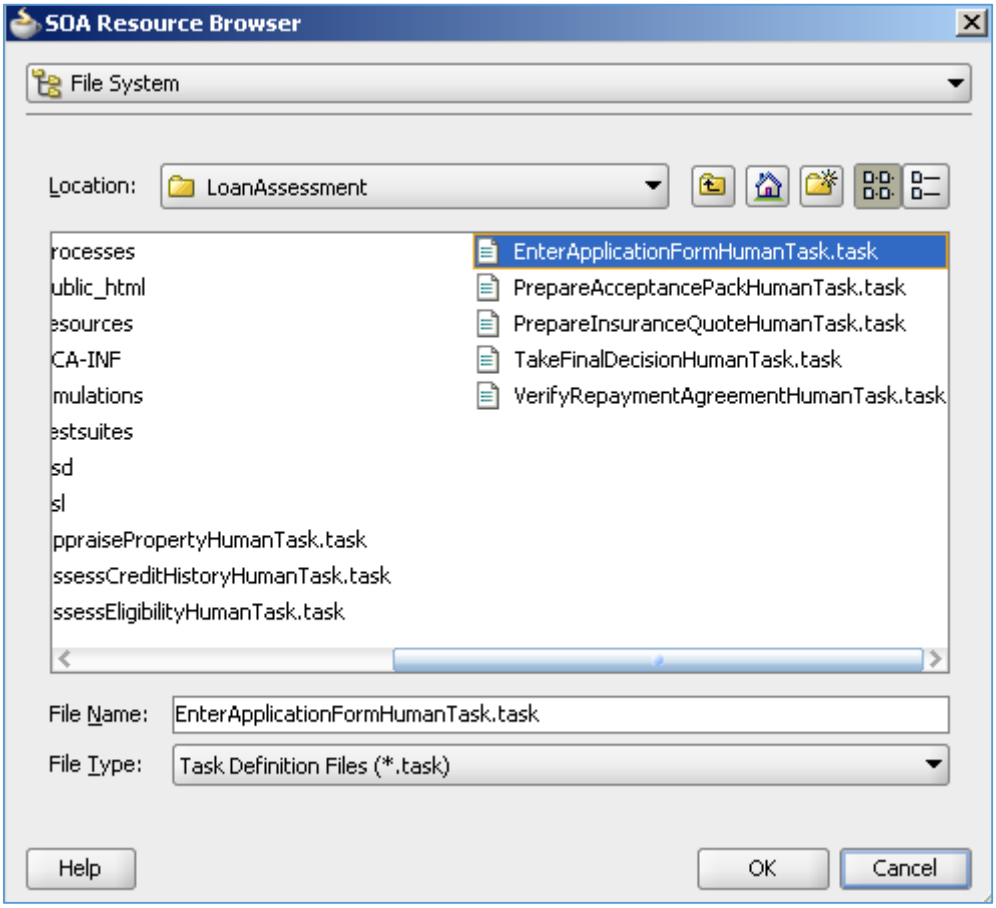
Right-click on the project folder in the Application Navigator and select New…In the New Gallery window, scroll down and select Web Tier > JSF. Select ADF Task Flow based on Human Task in the right pane. Task Flow is a new feature provided by Oracle ADF to JSF. It allows to design wizard-like web page with Back/Next button (task flow). In special case a task flow has only one page. In this tutorial we only design one page for every web form.

Then click on OK button.

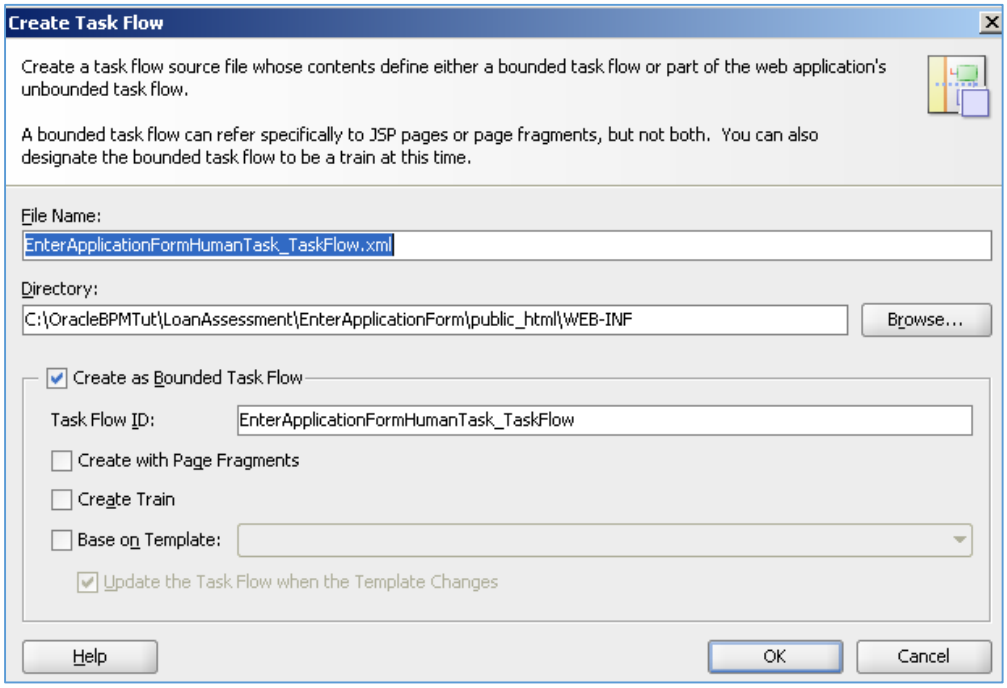


The SOA Resource Browser window appears and asks for the human task file associated with every human task of the BPMN diagram. The default location being opened is the ADF project folder but the human task file is located in the BPM project folder under the BPM application folder. Use Location button in the window, navigate to LoanAssessment\LoanAssessment folder (this is the BPM project folder).

Select **EnterApplicationFormHumanTask.task** file which contains the content of the Enter Application Form human task in the BPMN diagram. Then click on OK button.

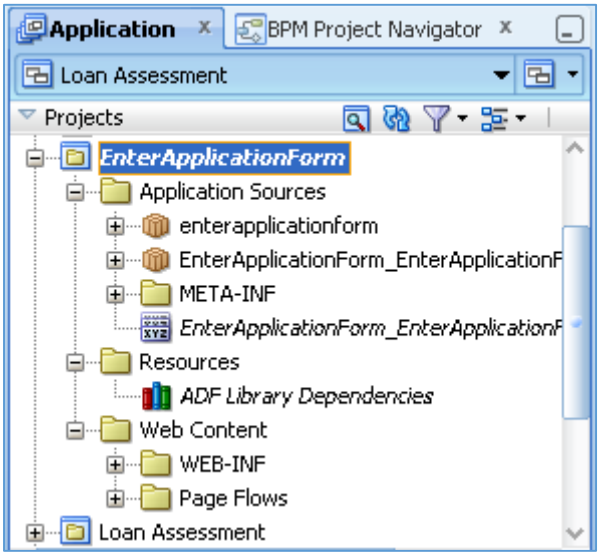


In the Create Task Flow window, accept default values and click on OK button.

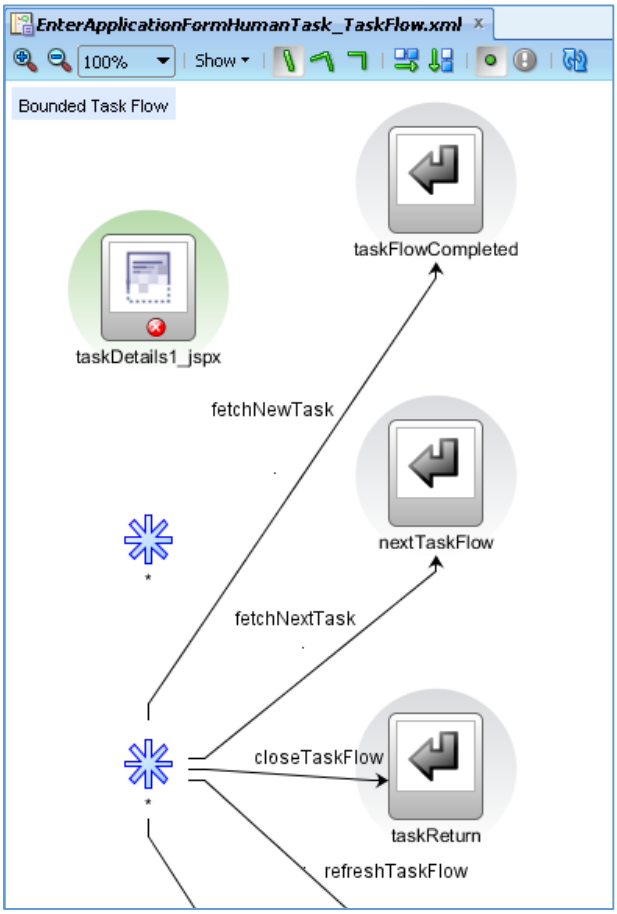


As a result, you can see some change effects on the EnterApplicationForm project.

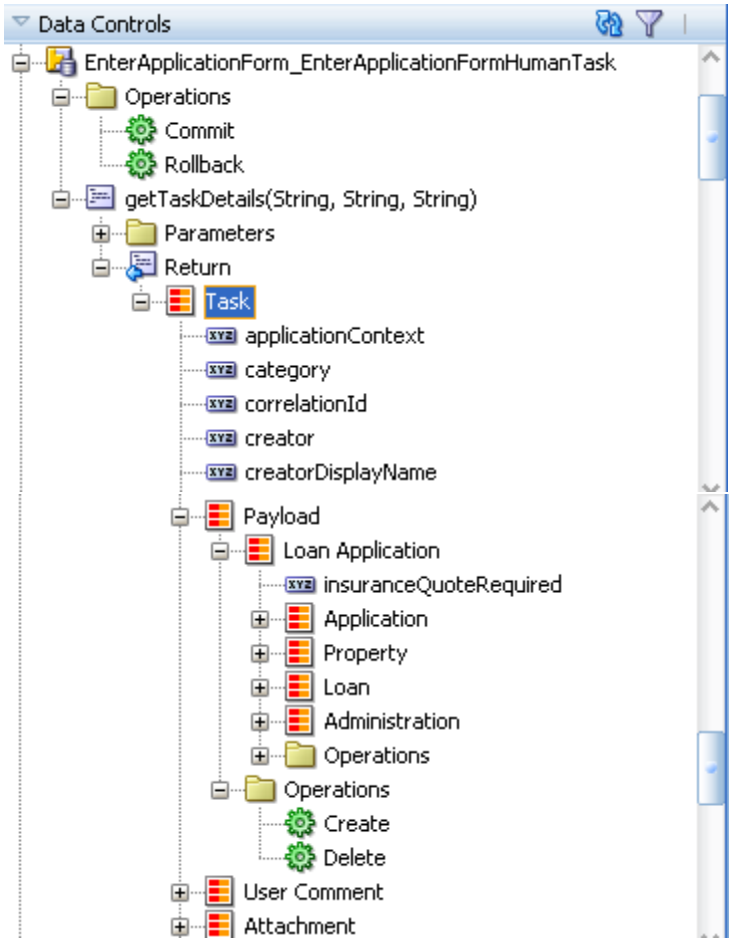
Many folders and files have been added automatically to the **EnterApplicationForm** project as shown below. It is also a typical structure of an Oracle ADF project which is basically similar to a java web project with some ADF-specific components.



The task flow design is opened in the design editor.



In addition, under the Data Controls pane, there is a new folder added: **EnterApplicationForm_EnterApplicationFormHumanTask**, as shown below.



Remember the Oracle ADF introduction in section 2, this is the Data Control implemented by Oracle ADF to connect the user interface with the data source (called data binding). Oracle ADF provides different types of data control for different data sources such as database, EJB, files, etc. This data control is called BPM data control which implements data binding with the human task data. Oracle has provided this facility with underlying codes in place, what we need to do is to declare the bindings between this data control and the web form in accordance with our requirements.

One data control is created for each Oracle ADF project.

Once the ADF project has been created with data control connected with the human task in place, we can now design the task flow and the web form.

In Application Navigator, expand the **EnterApplicationForm** project, go to Web Content > Page Flows. Double click on **EnterApplicationFormHumanTask_TaskFlow** to open it if it is not shown in the design editor.

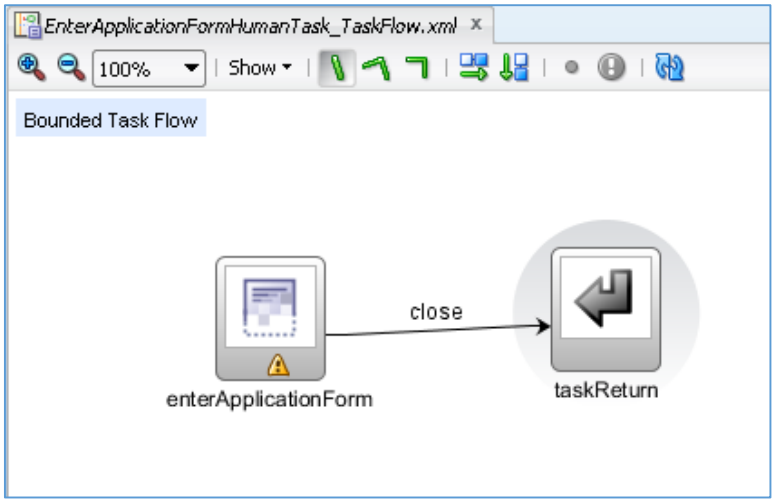
Since we don't use the full task flow features, let's delete all objects in the Task Flow except **taskReturn**, including all star-shaped objects.

From the Component Palette on the right, drag a View object to the design editor. Type "**enterApplicationForm**" into the highlighted name under the shape.

Next, click on Control Flow Case object in the Component Palette, then click on the **enterApplicationForm** view object and connect it to **taskReturn** object (the mouse will change to a plus sign while you are doing).

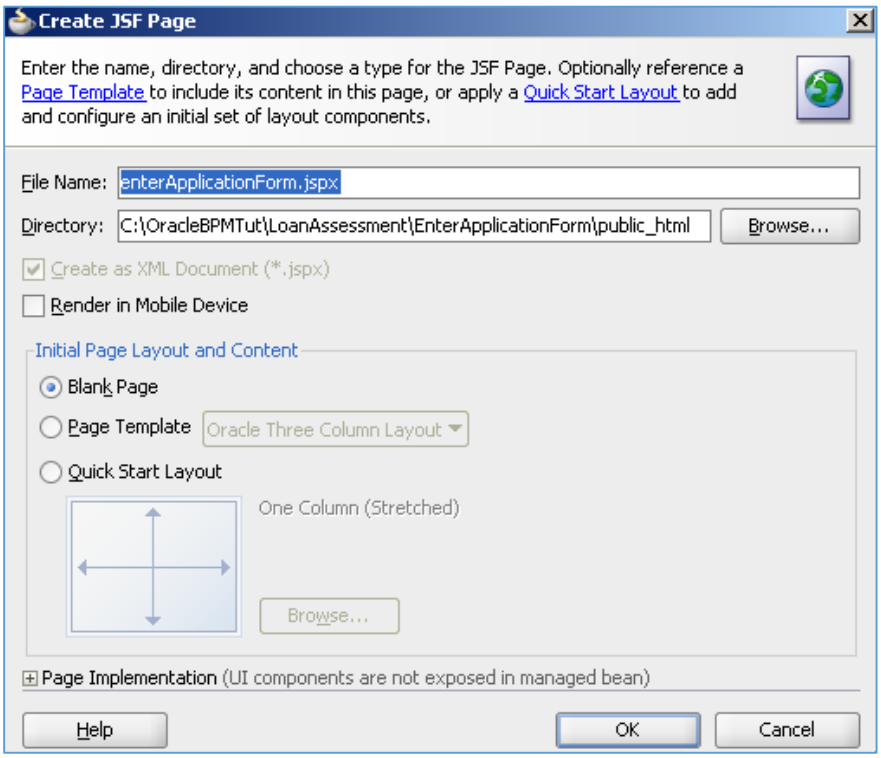
Name the connection flow as “close”.

The final task flow should look like below.



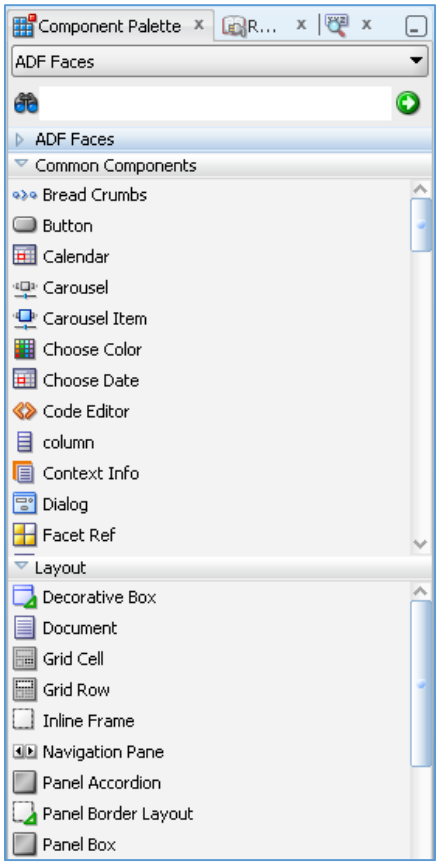
Now, double click on the **enterApplicationForm** view object in the task flow to create a JSF page for that view object.

A Create JSF Page will pop up with default values for a JSF page. Accept default values and click on OK button. We want to create a blank page first so that we will design the page by ourselves.



Wait a few minutes for the Designer to initialize. Then a blank page displays in the designer as shown below. A corresponding file is also added to the project under Web Content folder: **enterApplicationForm.jspx**.

While the designer is showing, open the Component Palette, you can see it contains a range of many interface elements for the page.



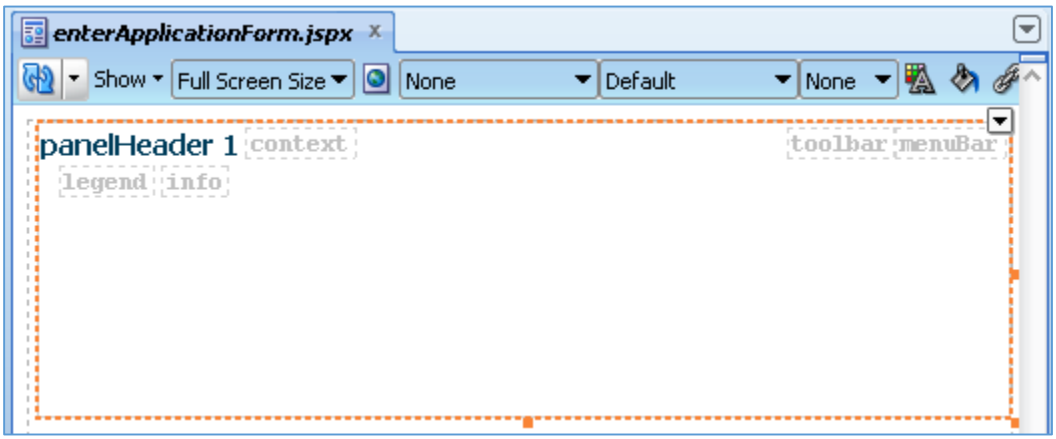
Now, keep the **Designer**, the **EnterApplicationForm_EnterApplicationFormHumanTask** data control and the **Component Palette** open. We mainly move around these three panes to design the web form.

Our target web form is the one shown at the beginning of this section. We'll add form items for every section and use panel to lay them out.

Note that when we are dragging and dropping fields on to the page, JDeveloper automatically records the binding data which represent the binding between form fields and data source.

Page Header

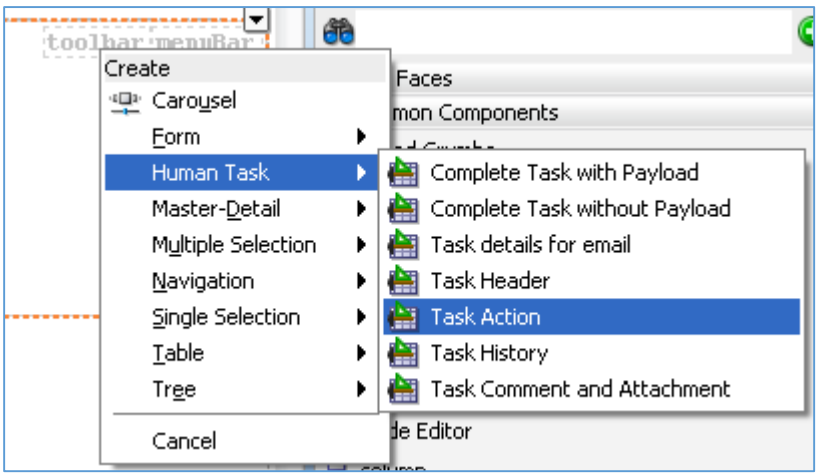
Now, select **Panel Header** in the Component Palette, under Layout section. Drag and drop Panel Header to the top of the Designer space. The designer looks like below. Note that there is a **toolbar** placeholder which allows us to design toolbar buttons there.



Note that you can always use Control-Z button to revert the designer to previous state if you make any mistake. The Undo function works well for the designer space.

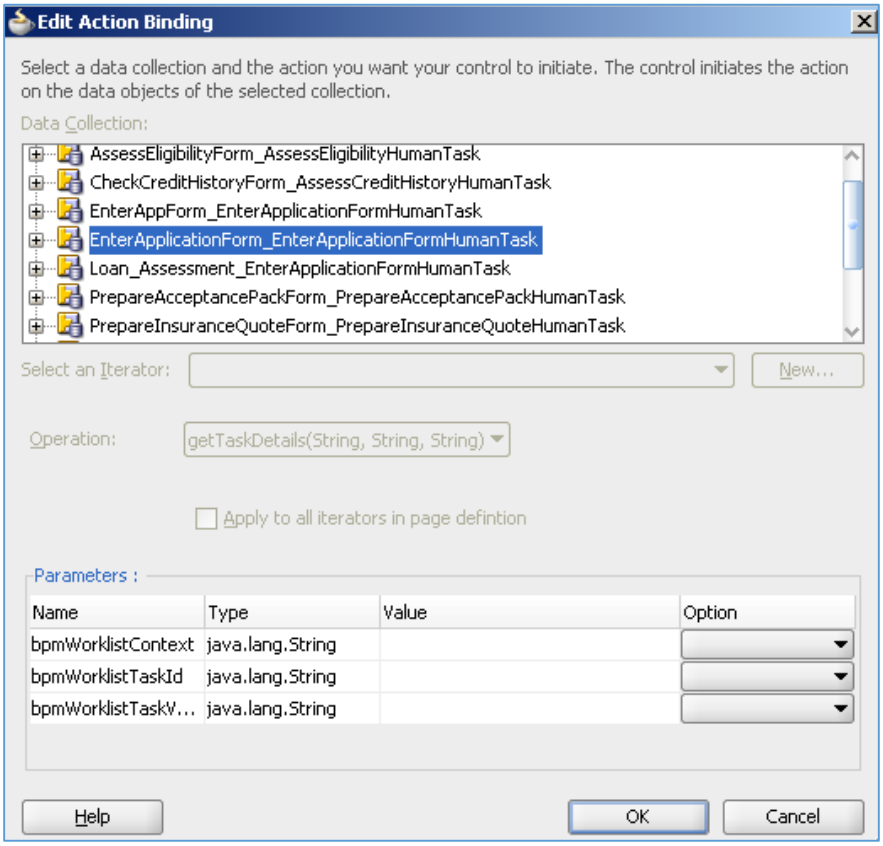
Now, switch to Data Controls pane, select the Task node under getTaskDetails > Return. Drag the Task node and drop it onto the toolbar placeholder in the panel header above.

A context menu will display right below.

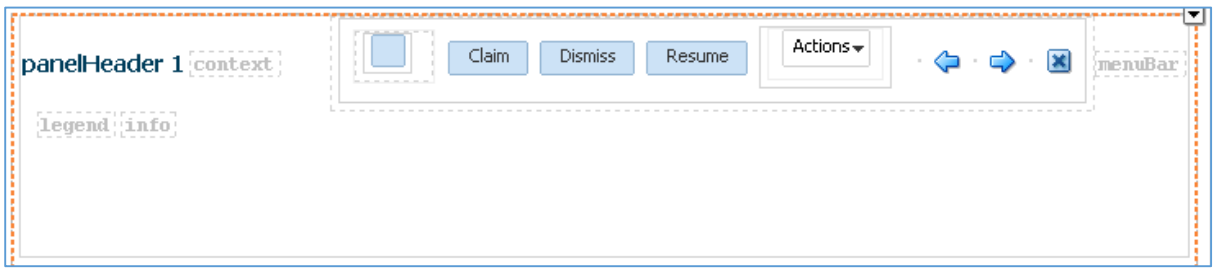


Select Human Task > Task Action in the context menu.

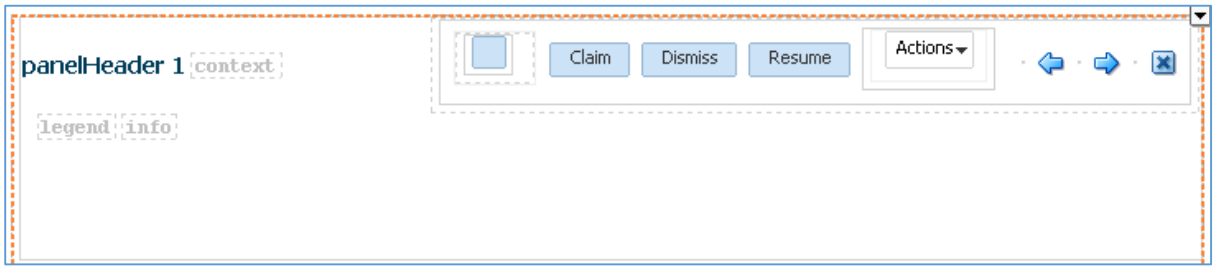
Click OK two times in the Edit Action Binding popup.



The page header will look like this with buttons added to the toolbar.

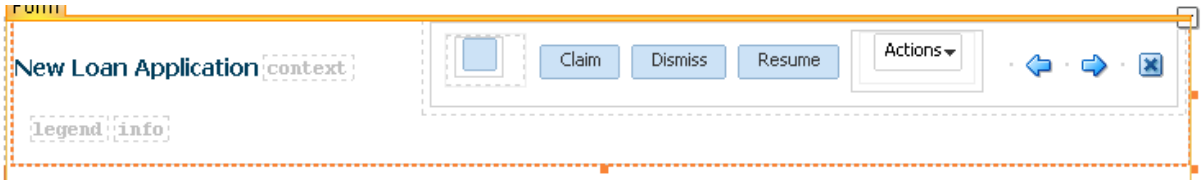


Select the menuBar and press Delete button to delete it.



Click the mouse on the **panelHeader 1** box. When it becomes editable, change the text to “New Loan Application”.

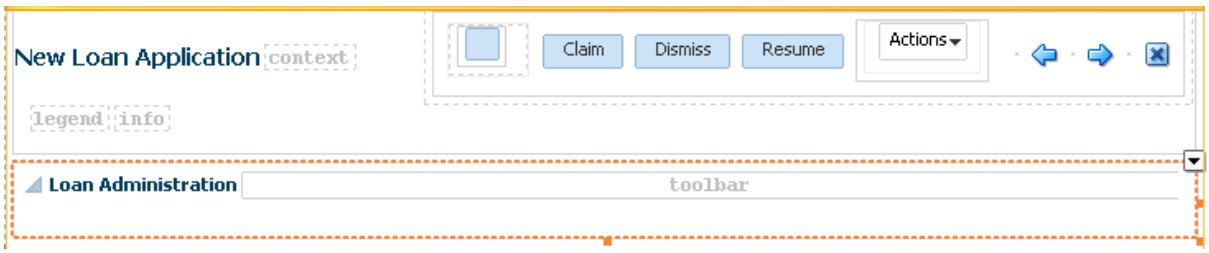
Click on the border of the page header, then click on the handle on the border to adjust the height of the panel to the size you want.



Loan Administration Panel

Select a Panel Box from Component Palette under Layout section. Drag and drop it on the designer under the page header panel.

Edit PanelBox1 to “Loan Administration”.

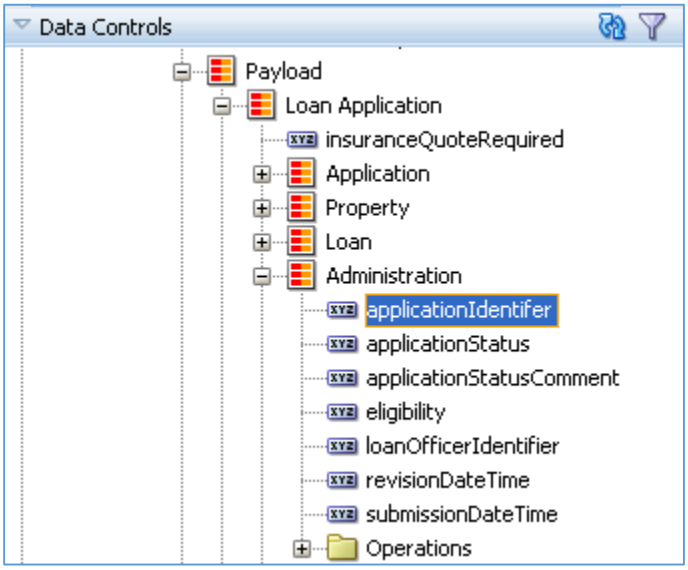


Drag and drop a Panel Form Layout from Component Palette on the Loan Administration panel. Panel Form Layout is used to contain form fields.

Select the footer and delete it.

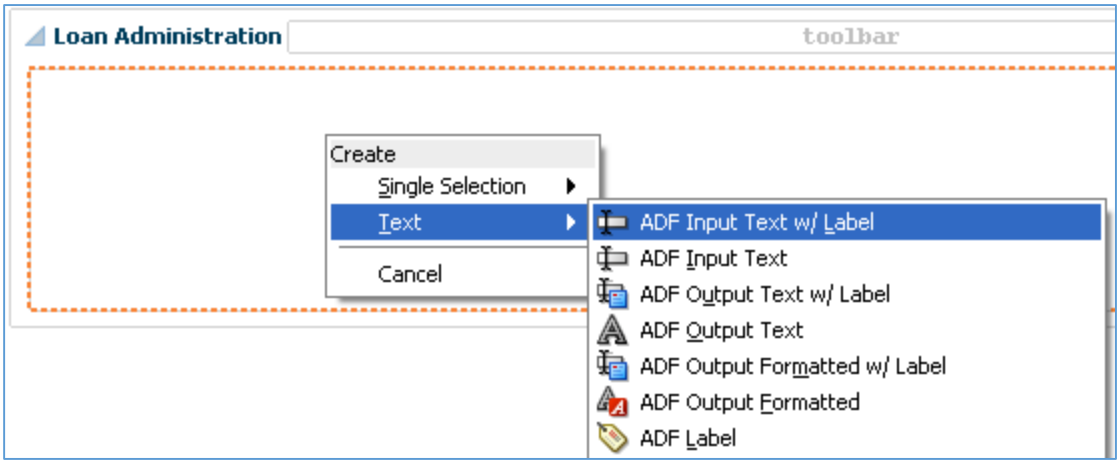
Use the form layout handle to adjust its height.

Now, move to the Data Controls pane, expand Task > Payload > Loan Application > Administration, as shown below.

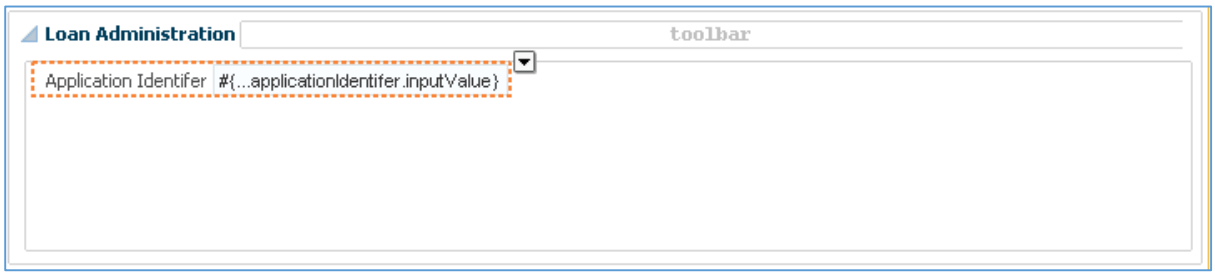


Drag **applicationIdentifier** and drop it on to the Panel Form Layout (the panel will show the border when it gets focus).

A context menu will appear. Select Text > ADF Input Text w/ Label.



The panel will look like this.

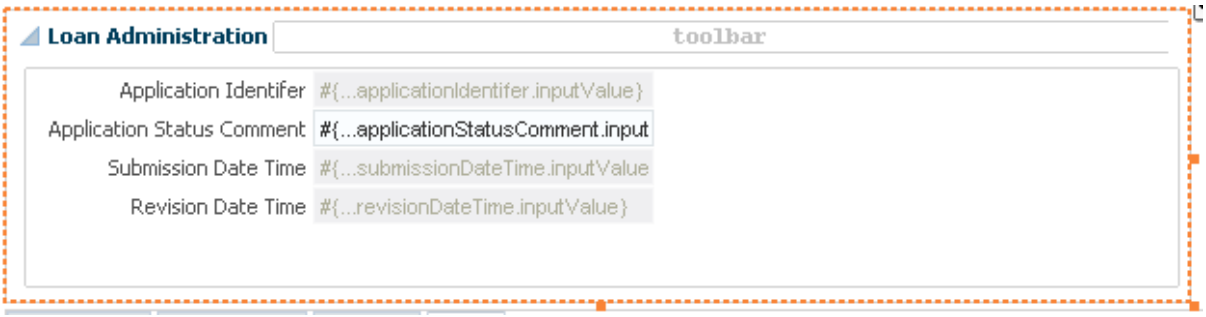


Now, with the application identifier field selected, open Inspector pane (on the right next to the Component Palette). It displays all existing properties of the field.

Scroll down to Behavior section, set **Disabled** property value to true. The field background color will be set to grey. At run-time, this field is displayed with grey background color and not editable.

Do the same with applicationStatusComment, submissionDateTime, revisionDateTime. Drag and drop it on to the Loan Administration panel, under the Application Identifier field.

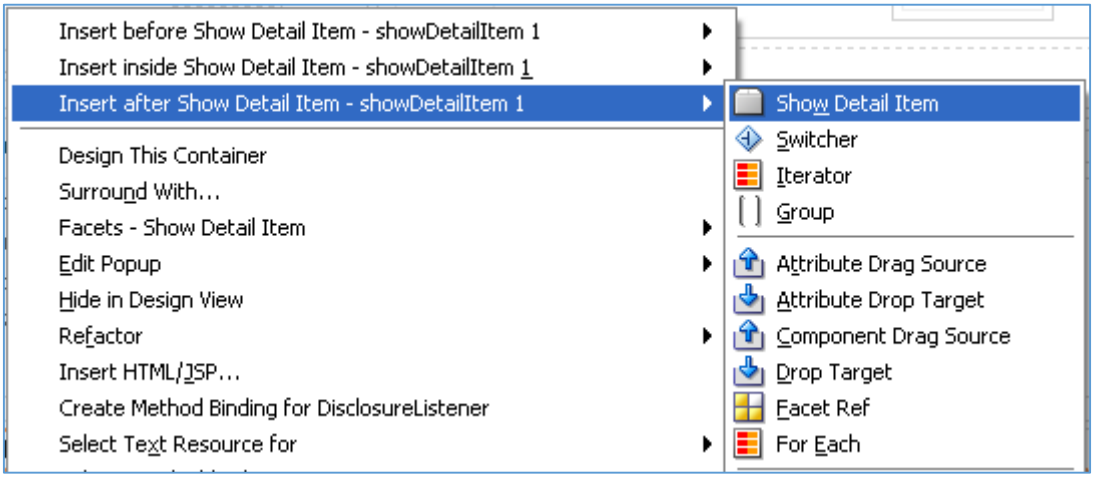
Then set the Disabled property to true for Submission DateTime and Revision DateTime field.



Tabbed Panel

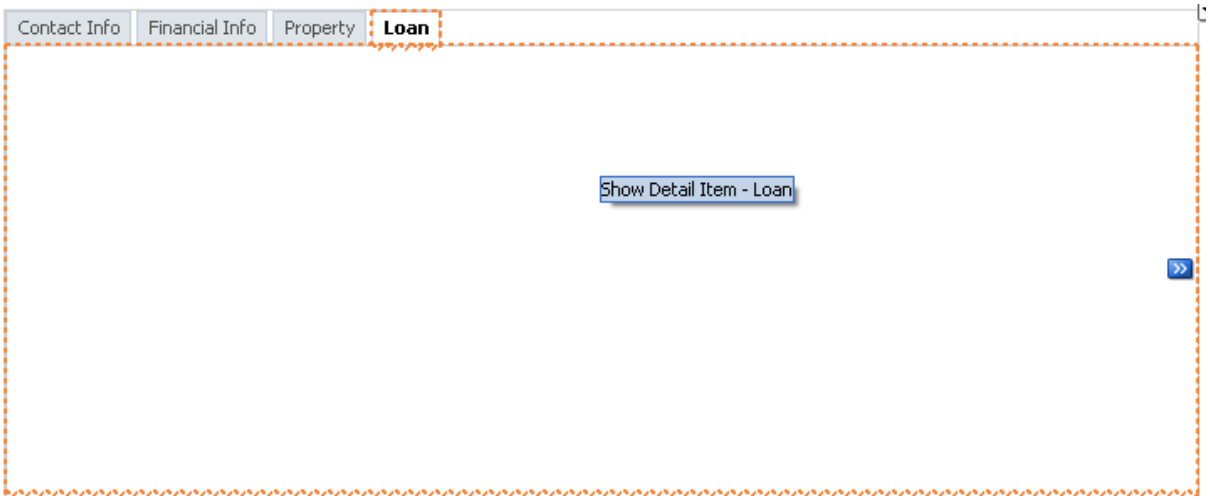
Drag and drop a Panel Tabbed from Component Palette to the designer. Use the handle on the border to adjust its width. It should look like below.

Right click on the ShowDetailItem 1, select Insert after… > Show Detail Item to add another tabbed page.



Add two more tabbed page similarly to the above.

Click on the title of every tabbed page until it is editable. Change them to: Contact Info, Financial Info, Property, Loan.

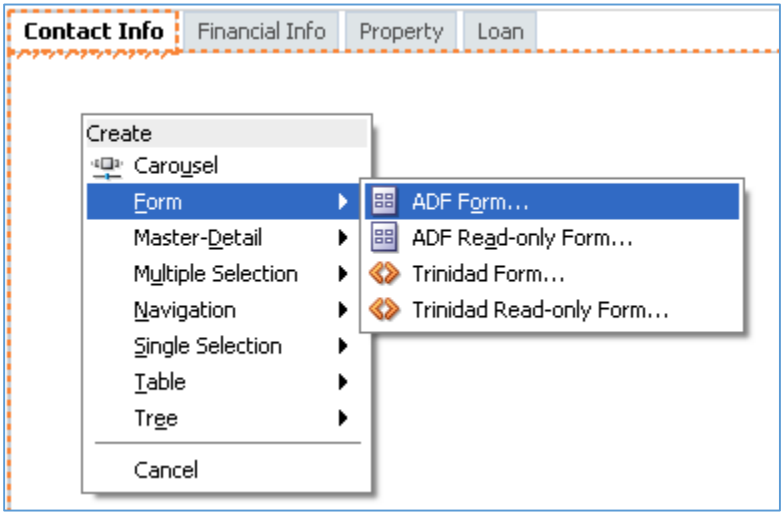


Contact Info Tabbed Page

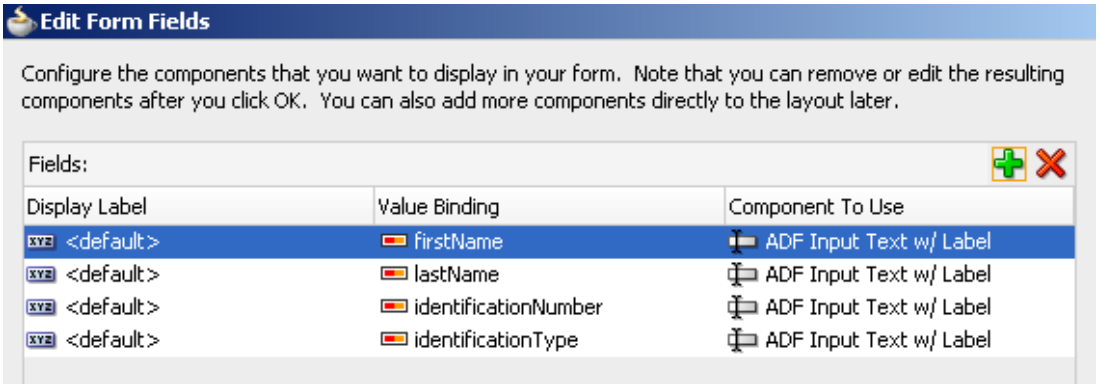
Click on Contact Info tab to open it in the designer.

From Data Controls pane, select Identification node in Payload > Loan Application > Application. Drag and drop it on the Contact Info tab page.

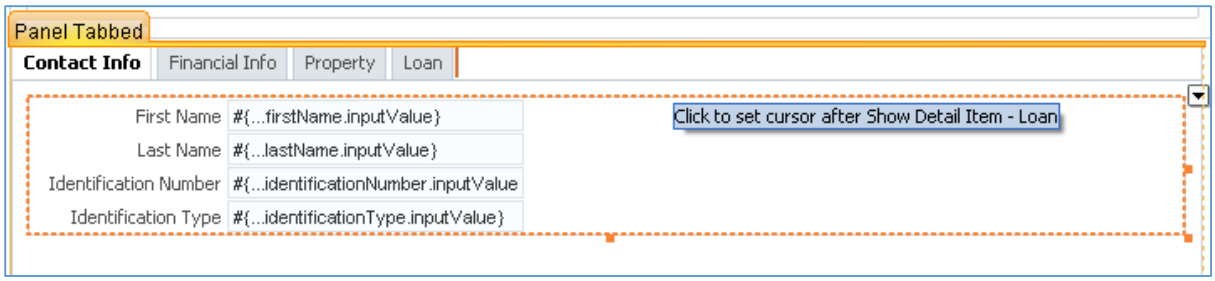
Select Form > ADF Form… in the context menu.



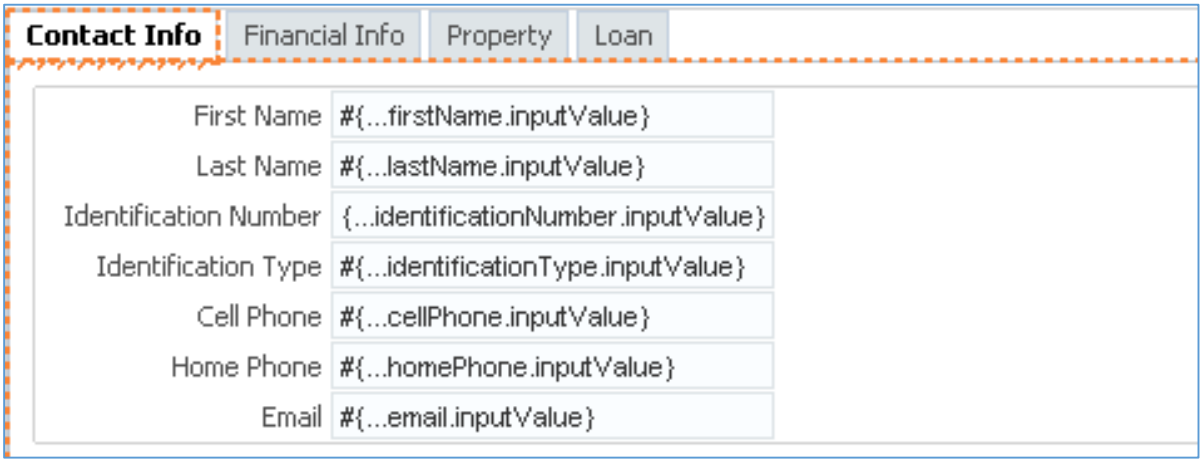
Select OK in the Edit Form Fields.



Contact Info tabbed page is updated as below. Note that a Panel Form Layout is automatically added to group these fields in one form panel.

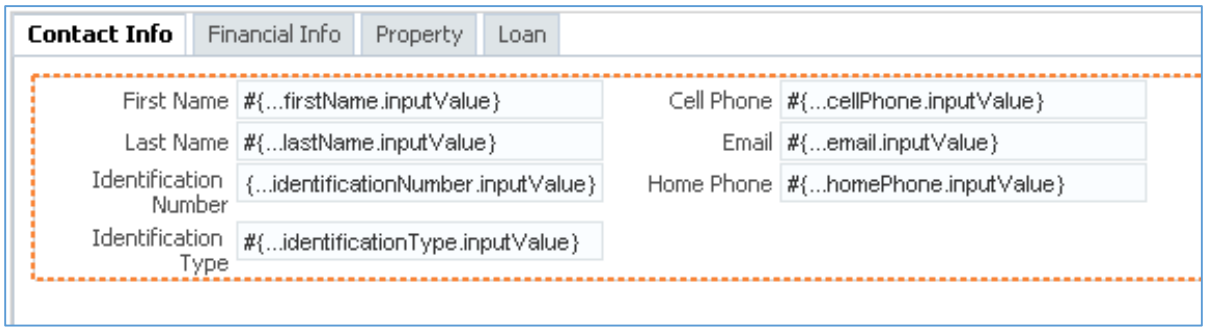


Now, drag and drop individual fields from Data Controls to the above form panel. These fields are under Loan Application > Application > Contact, they are cellphone, homePhone, email.



You can adjust the layout of form fields by arranging them in rows and columns. Select the Form Layout Panel containing these fields (you can select the element right on the Structure pane rather than trying to clicking on it on the form).

Open Inspector which will shows properties of the Form Layout Panel. Enter 3 for MaxColumns and 4 for Rows. The fields will be arranged accordingly.

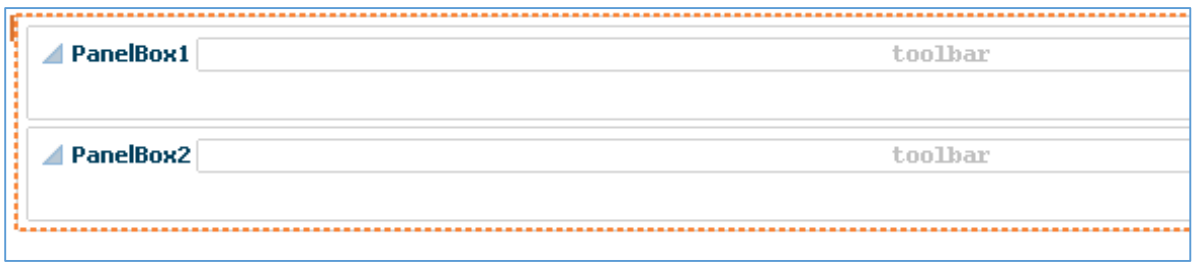


Current and Previous Address Panel

Drag and drop a Panel Group Layout from Component Palette on the Contact Info tabbed page, under the Form Layout Panel above.

Adjust its size to occupy the page width.

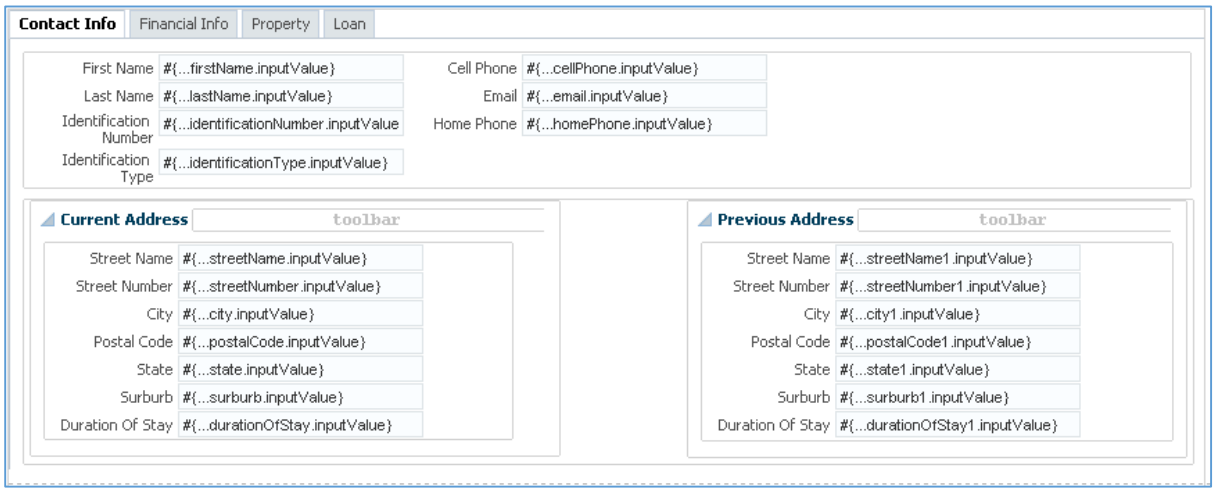
Then, drag and drop two Panel Box on the Group Layout panel.



Select the Group Layout panel. In the Inspector pane, set the Layout property from to **horizontal**. The two box panel will be arranged horizontally like below.

Edit the title of two box panels to: Current Address and Previous Address.

Next, drag and drop Current Address node from Data Controls pane on to the Current Address box panel, and Previous Address node on to the Previous Address box panel.



Note that in more complex form with fields and panels, some fields or panel borders may be hidden partially. You can fine tune the edge and size of panel by clicking on this button.

The panel will be displayed alone so that you can see its full border and then able to grab the handle on the border to adjust its size.

You have done the design of Contact Info tabbed page.

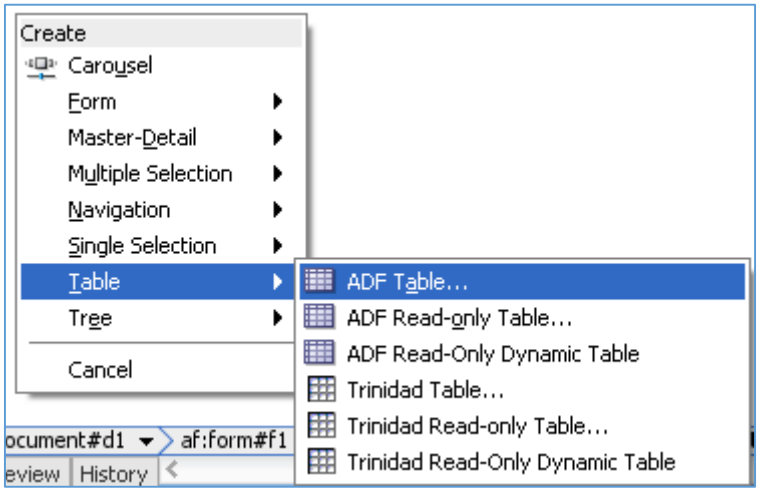
Financial Info tabbed Page

You can apply the same techniques above for this page.

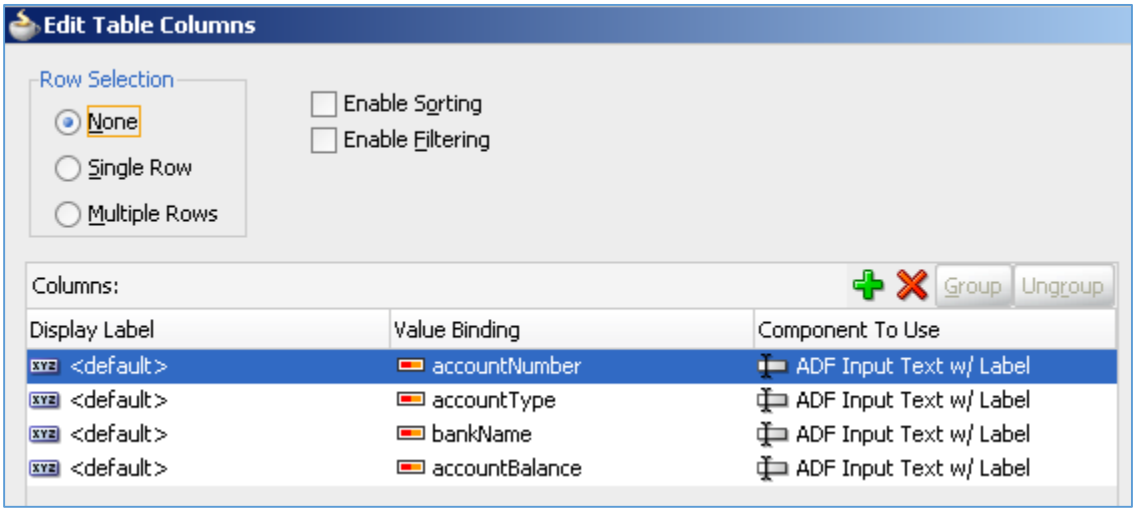
The only difference here is a table layout for bank account listing.

Drag Bank Accounts from Data Controls pane (under Application > Financial Info) and drop it on Financial Info tabbed page.

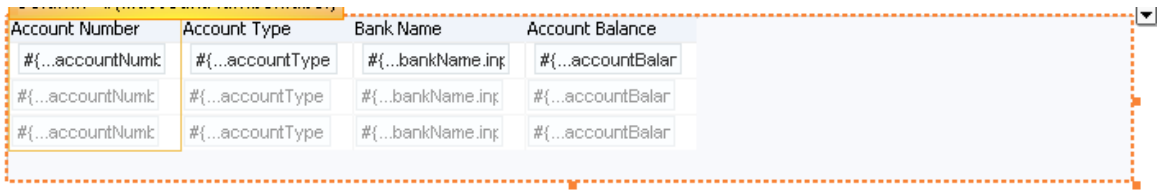
Select Table > ADF Table… in the context menu.



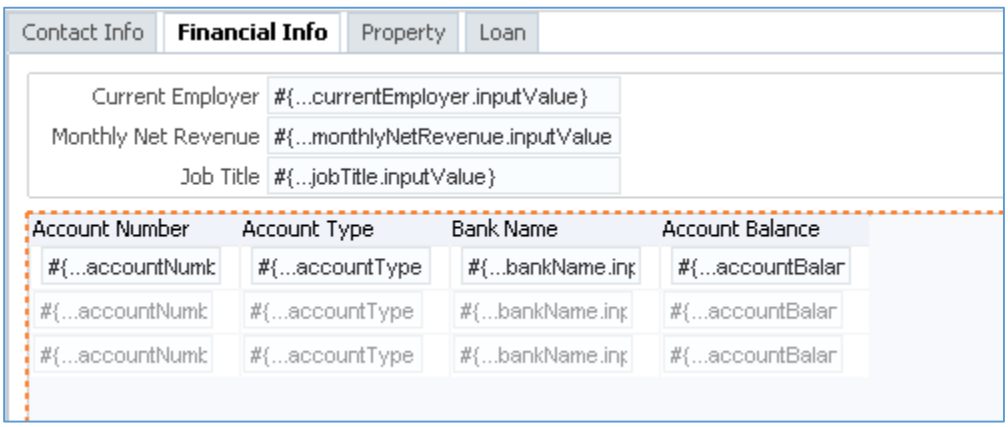
Select OK in the Edit Table Columns.



Then adjust the size of the table.



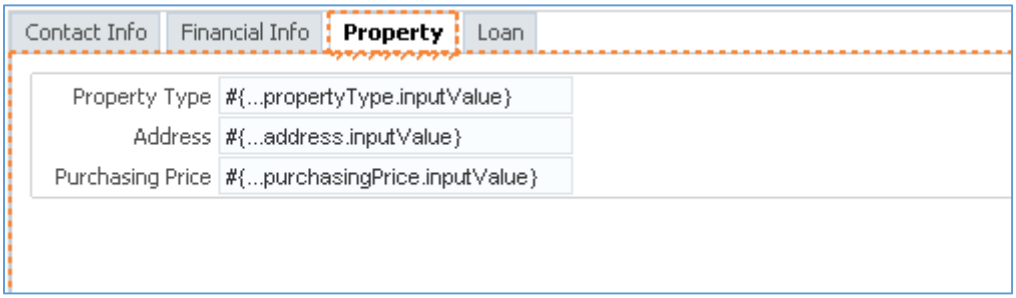
The complete tab should look like below.



Property Tabbed Page

Apply the above techniques with no difference to design this form.

The complete tab should look like below.

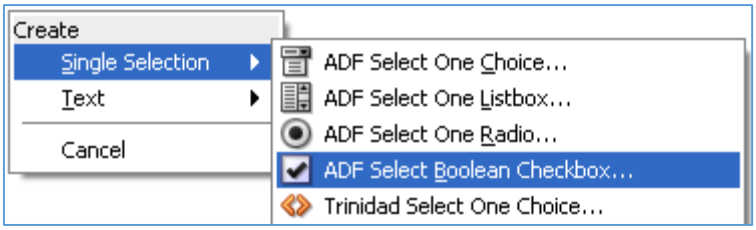


Loan Tabbed Page

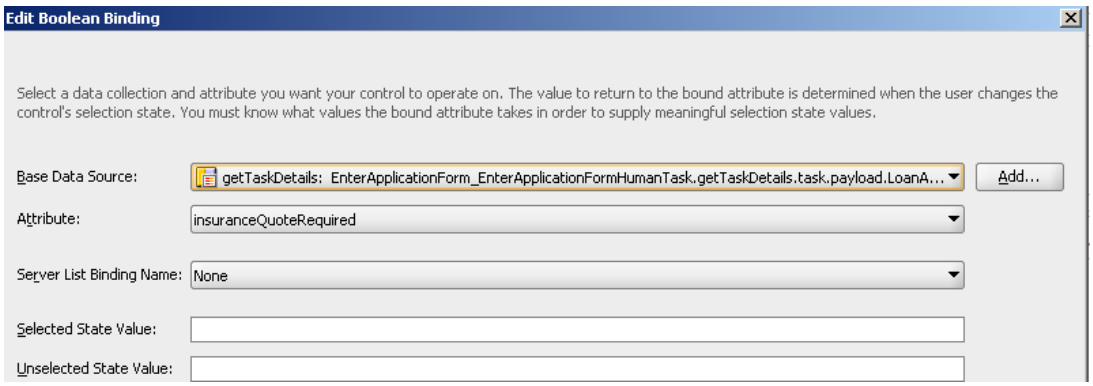
All fields should be designed the same as previous page.

The only exception is the checkbox for Insurance Quote if it is requested.

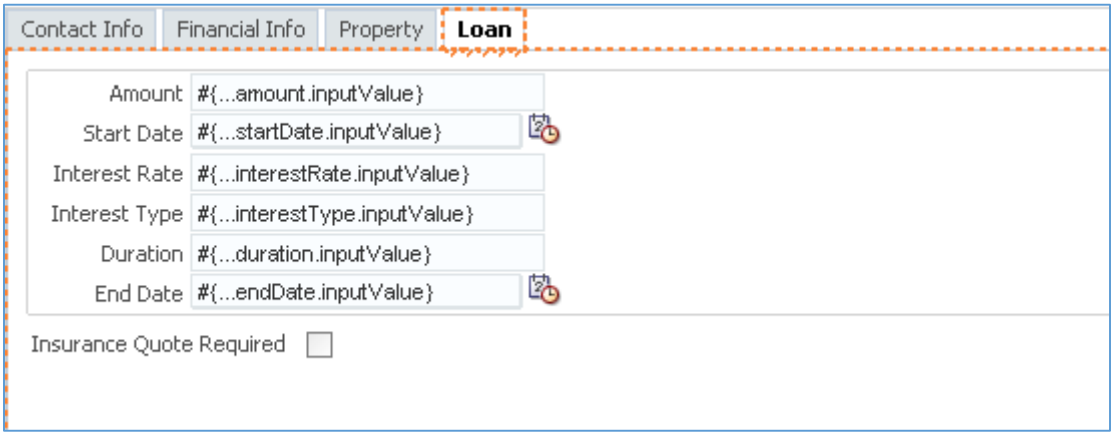
When you drag and drop the **insuranceQuoteRequired** node on the page, select Single Selection > ADF Select Boolean Checkbox from the context menu.



In the Edit Boolean Binding, enter **true** for Selected State Value and **false** for Unselected State Value. Then click OK.

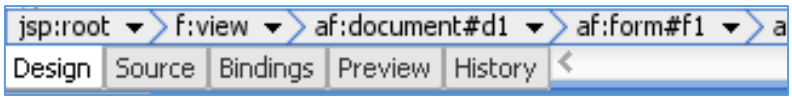


The complete page should look like below.

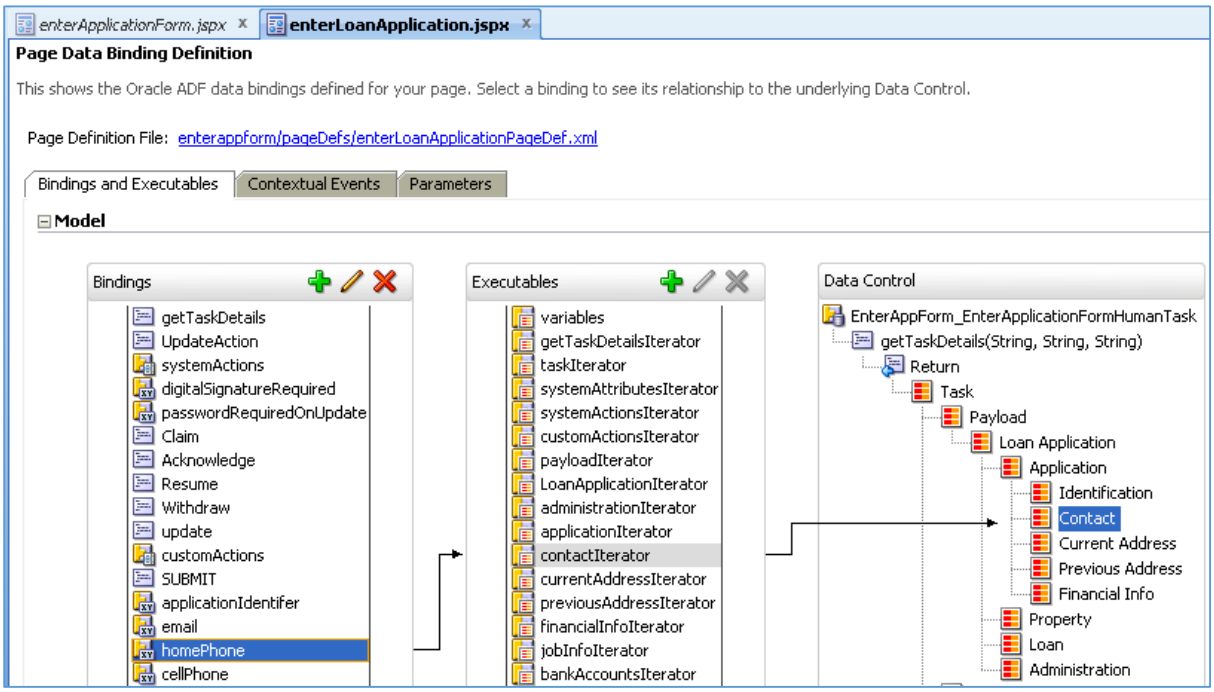


At this stage, we have finished the design of Enter Application Form and the binding between form fields and the data source.

If you want to view more details of binding data, click on Bindings at the bottom of the designer.



Click on one of the Bindings field, e.g. **homePhone**. The binding is shown as follows. This is actually a visualization of the binding configuration file named **enterApplicationFormPageDef.xml** located under EnterApplicationForm\enterapplicationform.pageDef folder.



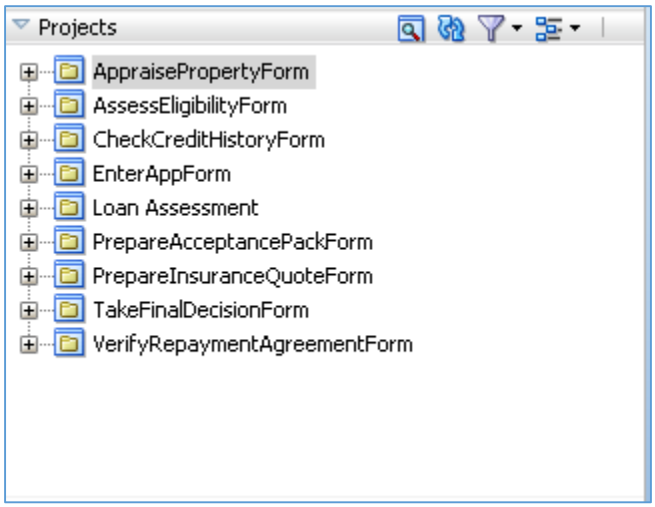
So, we have done various implementation steps for the first human task, including setting human task object (title, data mapping, users, etc.) and details of Oracle ADF form design. Hope that you have grasped considerable skills with these implementation details.

The implementation of other forms follows the same techniques as those presented for Enter Application Form page. Therefore, in the subsequent sections we won't be going through step-by-step

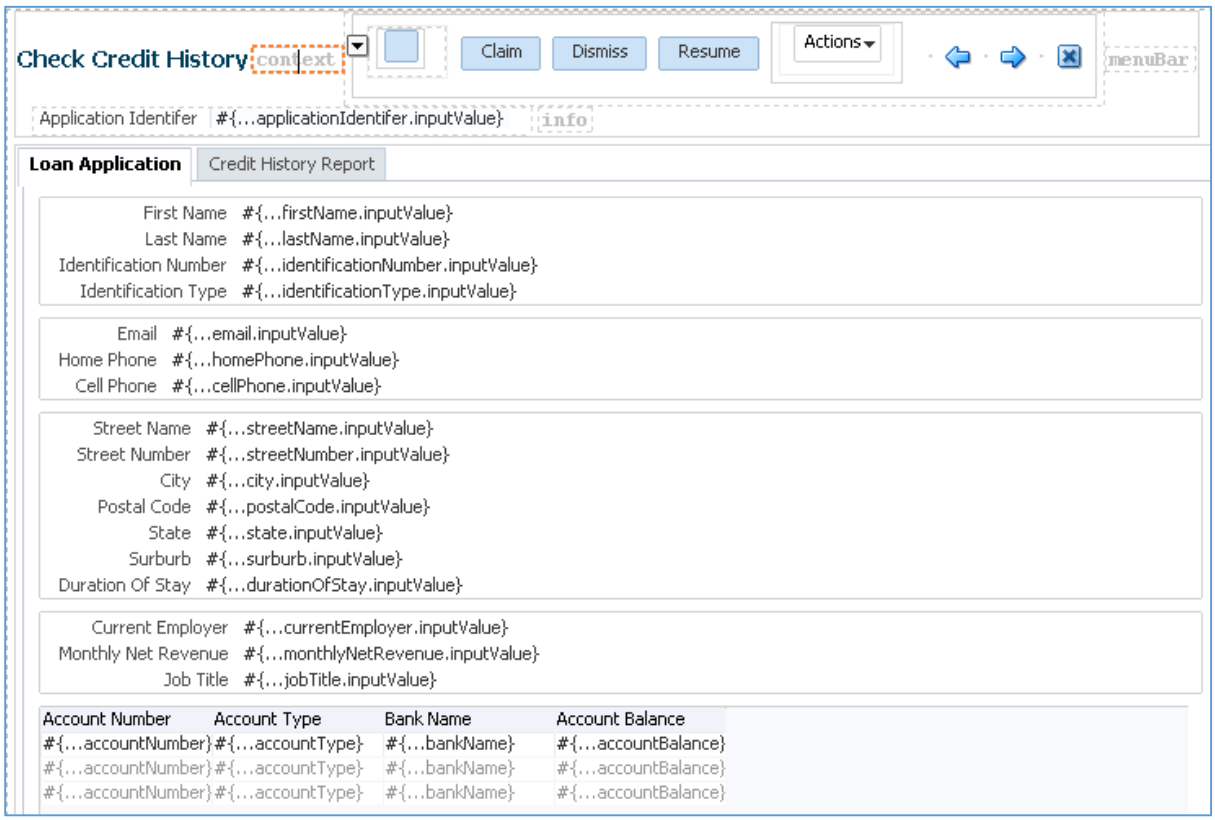
instructions but only some complete page and task configuration. You are encouraged to apply the same guidelines as described for Enter Application Form page to achieve those results.

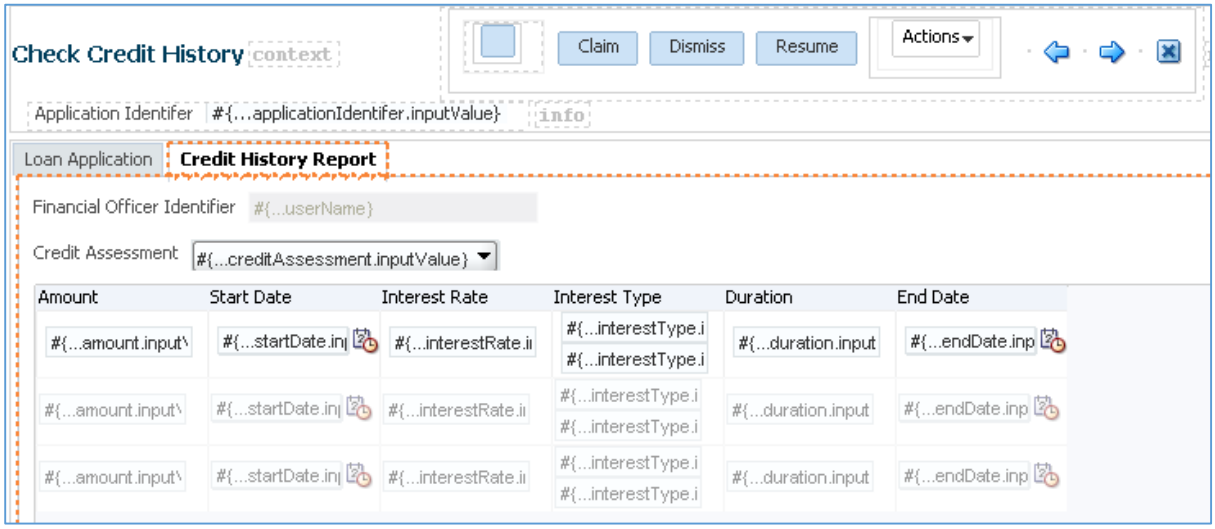
However, there are still some specific features of some forms, such as logged in username display, custom button, and PDF generation. Don't worry. They will be explained in details to implement.

There will be totally 9 projects created in the Application Navigator: 1 BPM project (Loan Assessment) and 8 human task implementation project corresponding to 8 user tasks in our BPMN process model.

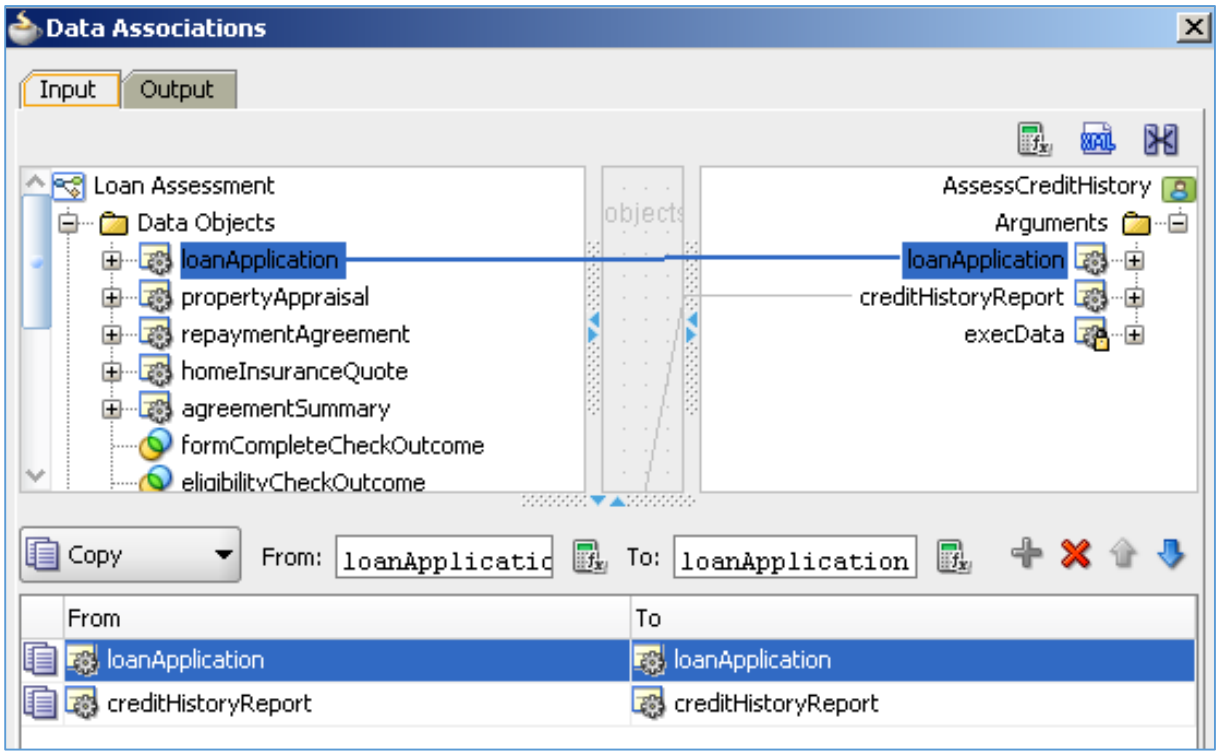


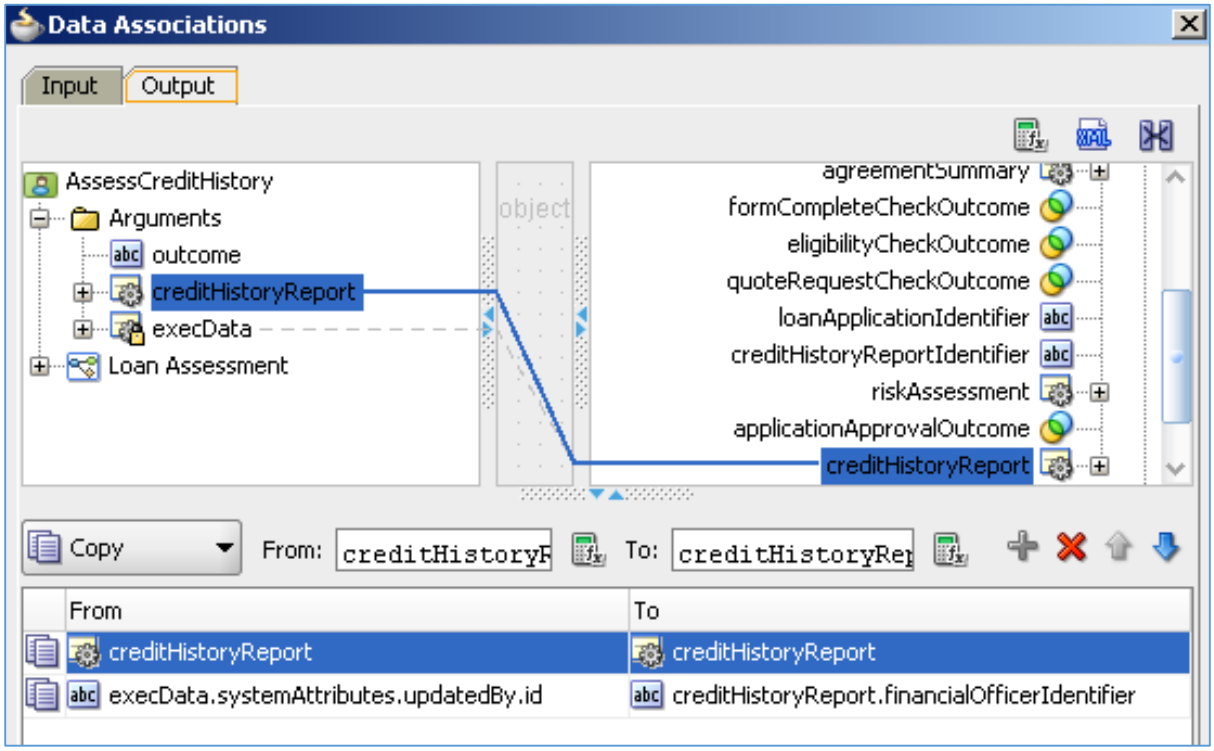
7.4.2 Assess Credit History





In order to show the logged in username as initial value in the Financial Officer Identifier field, please do the following. Select the Financial Officer Identifier field, in the Inspector pane, enter this string value in the Value property: **#{securityContext.userName}**

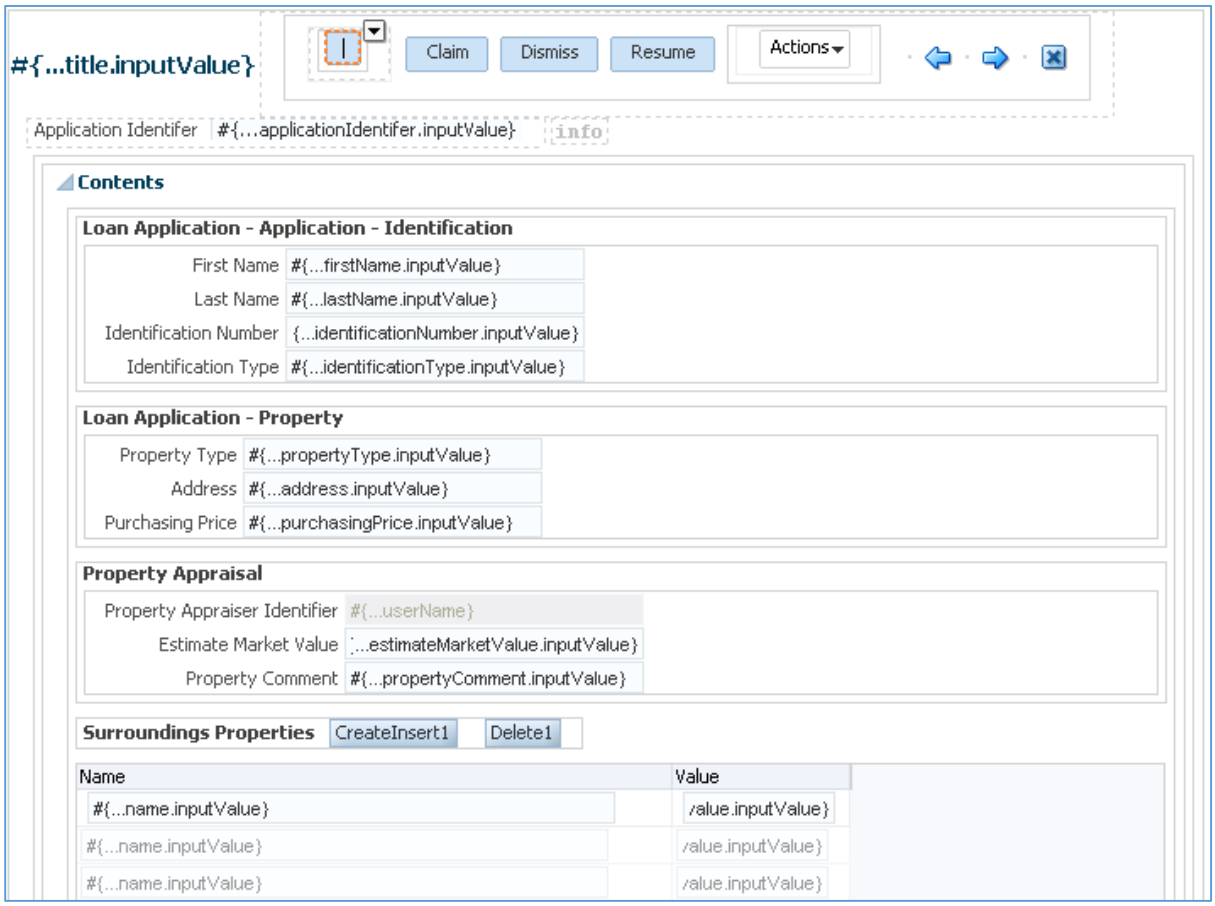




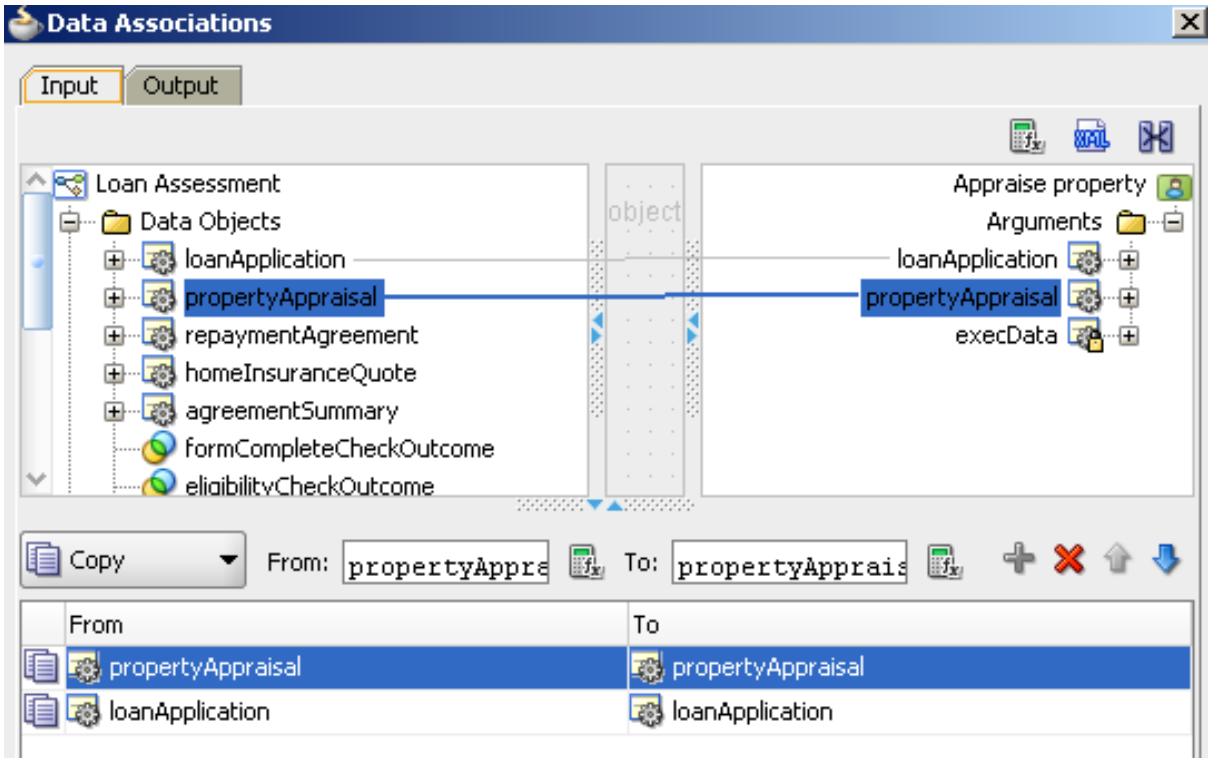
Note: the second mapping from execData.systemAttribues.updatedBy.id is to record the logged in username to the process variable. In this case it is Financial Officer.

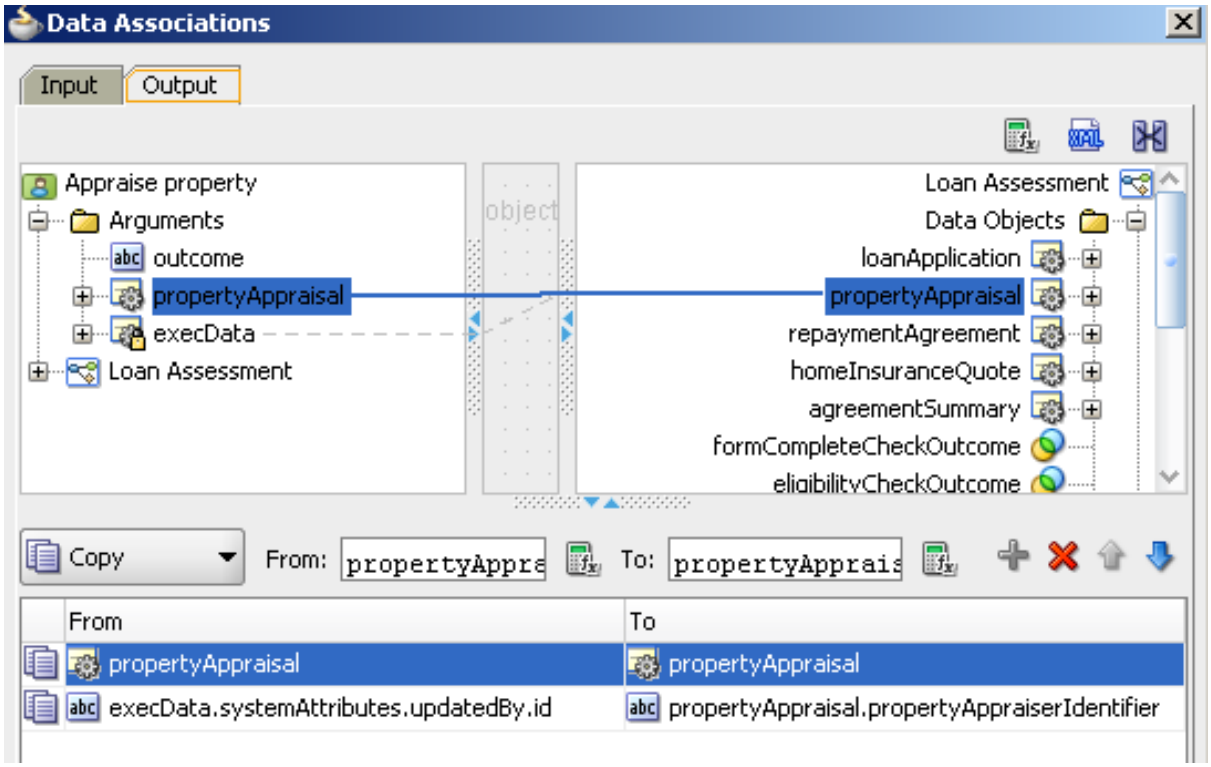
7.4.3 Appraise Property

-

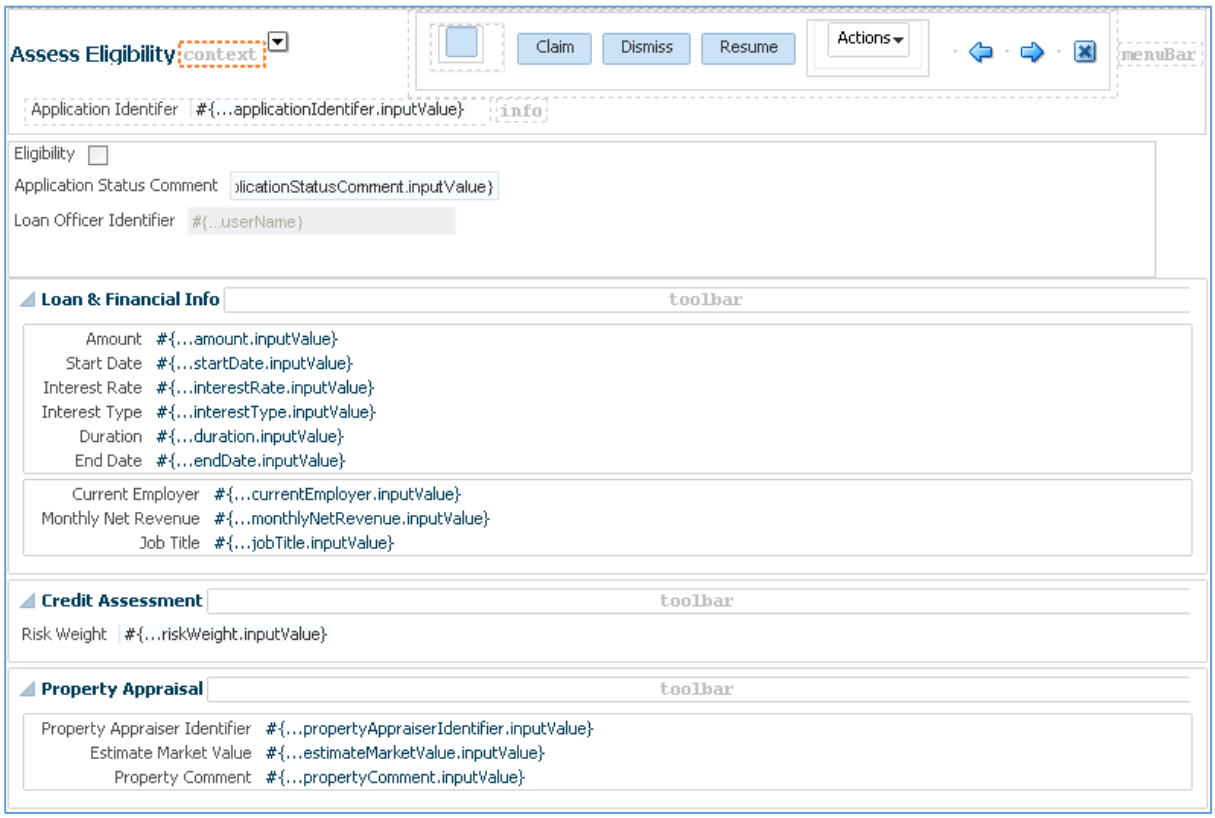


In order to show the logged in username as initial value in the Property Appraiser Identifier field, please do the following. Select the Property Appraiser Identifier field, in the Inspector pane, enter this value in the Value property: **#{securityContext.userName}**

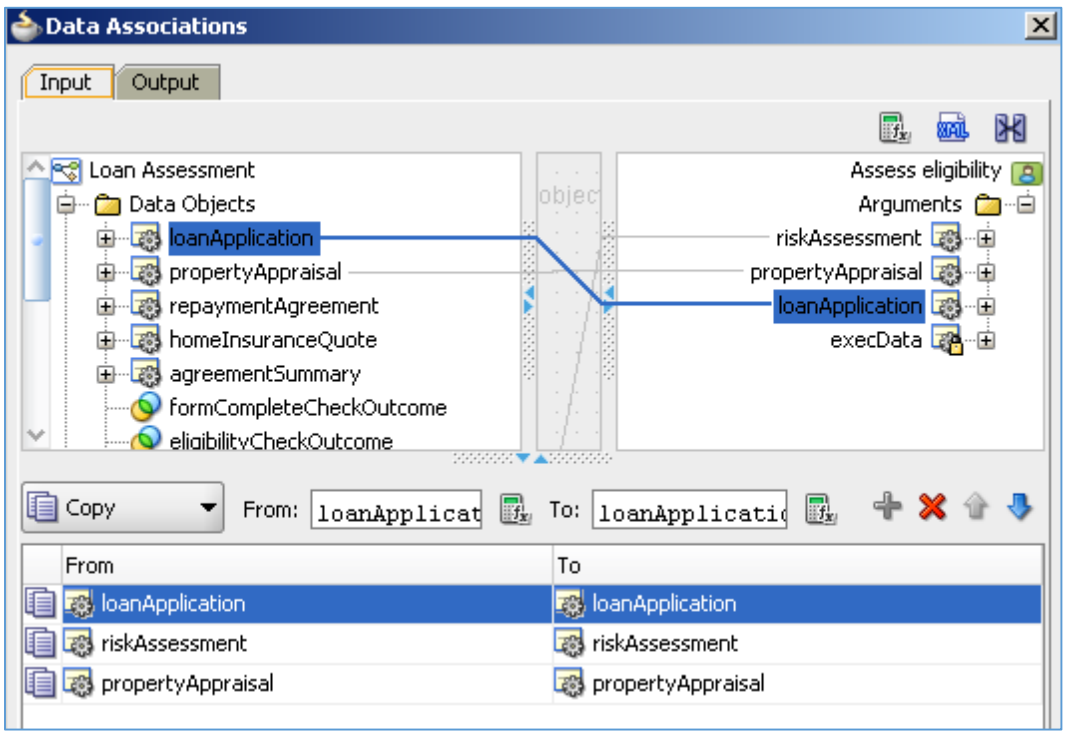

Data Associations
Input
Output
Loan Assessment
Data Objects
loanApplication
propertyAppraisal
repaymentAgreement
homeInsuranceQuote
agreementSummary
formCompleteCheckOutcome
eligibilityCheckOutcome
object
Appraise property
Arguments
loanApplication
propertyAppraisal
execData
Copy
From:
propertyAppra
To:
propertyApprais
From
To
propertyAppraisal
propertyAppraisal
loanApplication
loanApplication

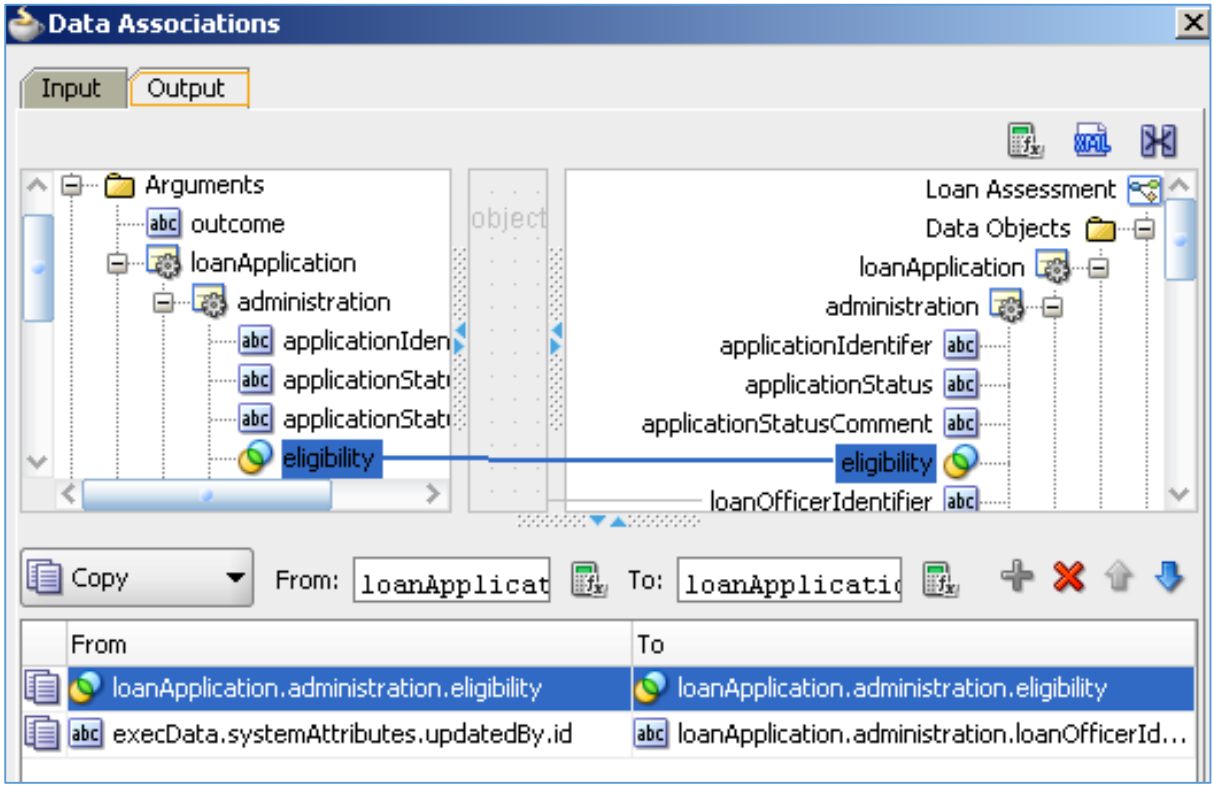

Data Associations
Input
Output
Appraise property
Arguments
outcome
propertyAppraisal
execData
Loan Assessment
object
Loan Assessment
Data Objects
loanApplication
propertyAppraisal
repaymentAgreement
homeInsuranceQuote
agreementSummary
formCompleteCheckOutcome
eligibilityCheckOutcome
Copy
From:
propertyAppra
To:
propertyApprais
From
To
propertyAppraisal
propertyAppraisal
execData.systemAttributes.updatedBy.id
propertyAppraisal.propertyAppraiserIdentifier

7.4.4 Assess Eligibility

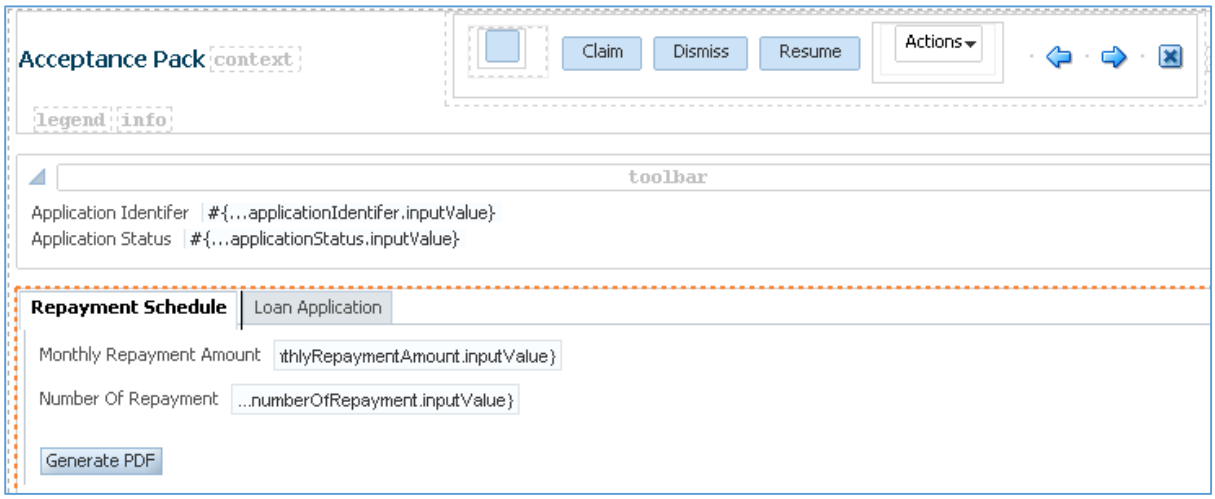


In order to show the logged in username as initial value in the Loan Officer Identifier field, please do the following. Select the Loan Officer Identifier field, in the Inspector pane, enter this value in the Value property: **#{securityContext.userName}**



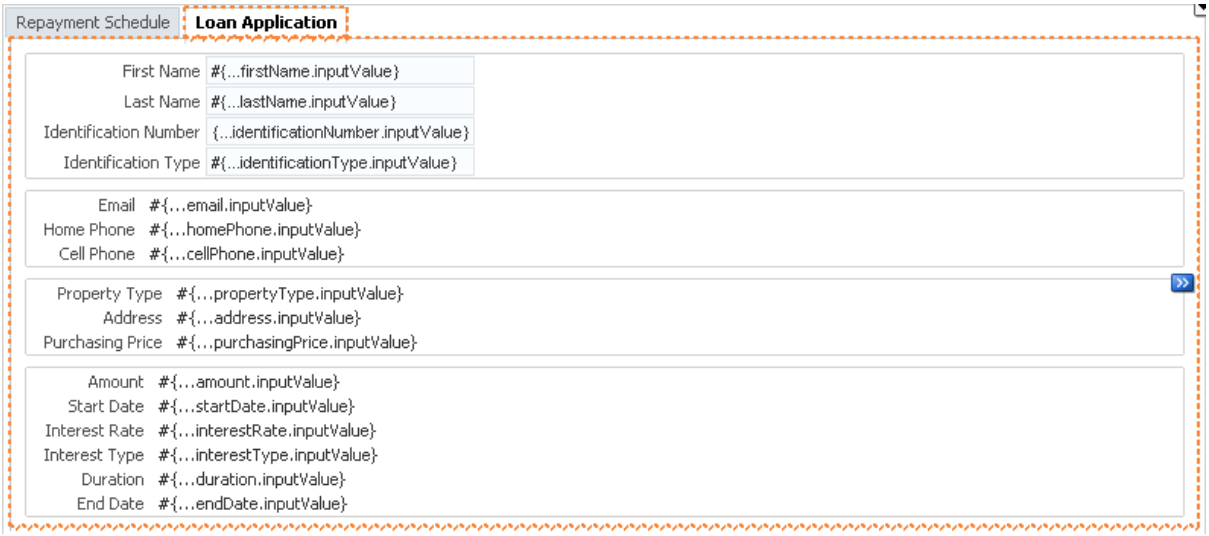


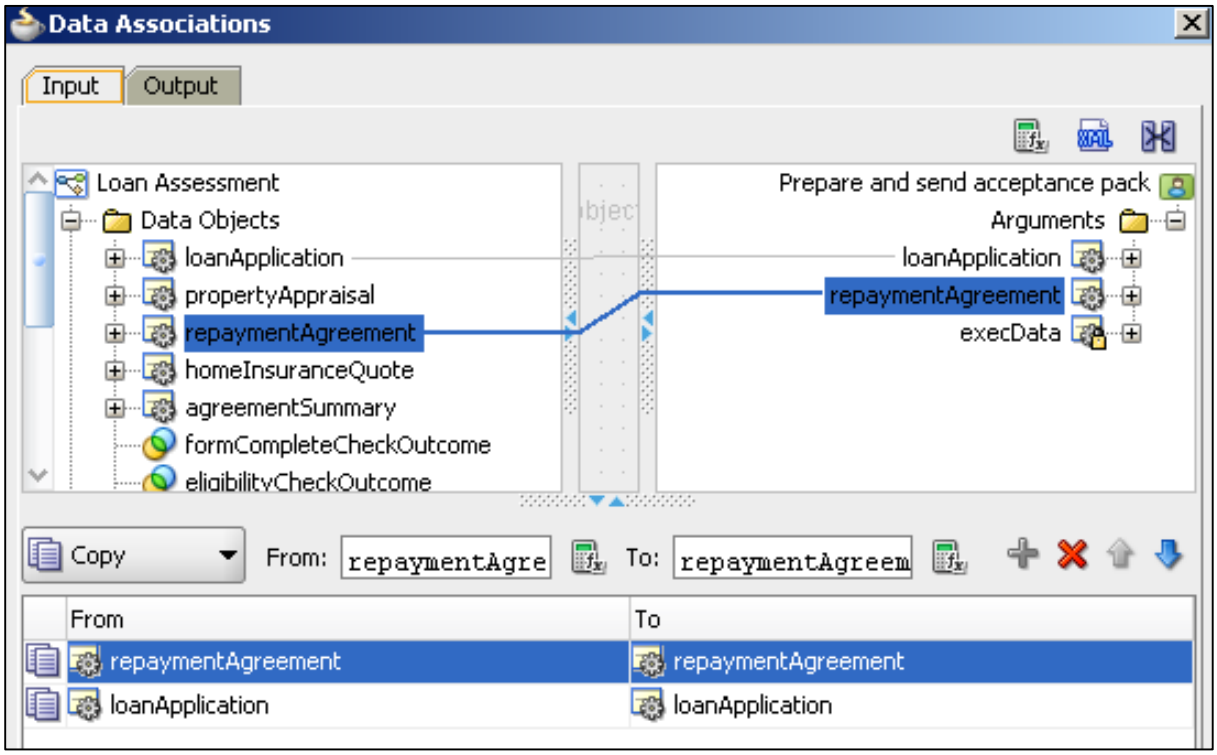
7.4.5 Prepare and Send Acceptance Pack

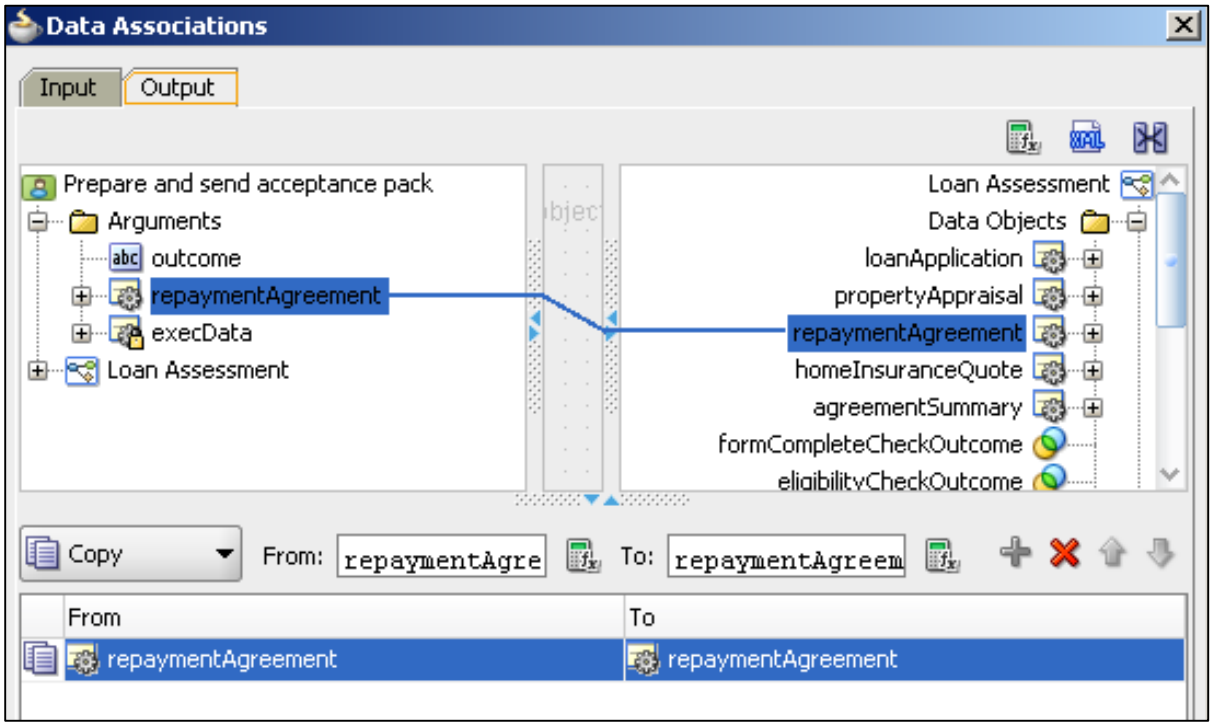


To add a custom button to the Repayment Schedule tabbed page, from Component Palette, Common Components section, drag and drop a Button component to Repayment Schedule page.

Right click the button and select “Go to Properties” to open the Inspector pane. Enter “Generate PDF” in the Text property and enter. The button label is updated to “Generate PDF” as shown above.


Repayment Schedule
Loan Application
First Name #{...firstName.inputValue}
Last Name #{...lastName.inputValue}
Identification Number {...identificationNumber.inputValue}
Identification Type #{...identificationType.inputValue}
Email #{...email.inputValue}
Home Phone #{...homePhone.inputValue}
Cell Phone #{...cellPhone.inputValue}
Property Type #{...propertyType.inputValue}
Address #{...address.inputValue}
Purchasing Price #{...purchasingPrice.inputValue}
Amount #{...amount.inputValue}
Start Date #{...startDate.inputValue}
Interest Rate #{...interestRate.inputValue}
Interest Type #{...interestType.inputValue}
Duration #{...duration.inputValue}
End Date #{...endDate.inputValue}


Data Associations
Input
Output
Loan Assessment
Data Objects
loanApplication
propertyAppraisal
repaymentAgreement
homeInsuranceQuote
agreementSummary
formCompleteCheckOutcome
eligibilityCheckOutcome
Prepare and send acceptance pack
Arguments
loanApplication
repaymentAgreement
execData
Copy
From: repaymentAgre
To: repaymentAgreem
From
To
repaymentAgreement
repaymentAgreement
loanApplication
loanApplication



Implement PDF Generation

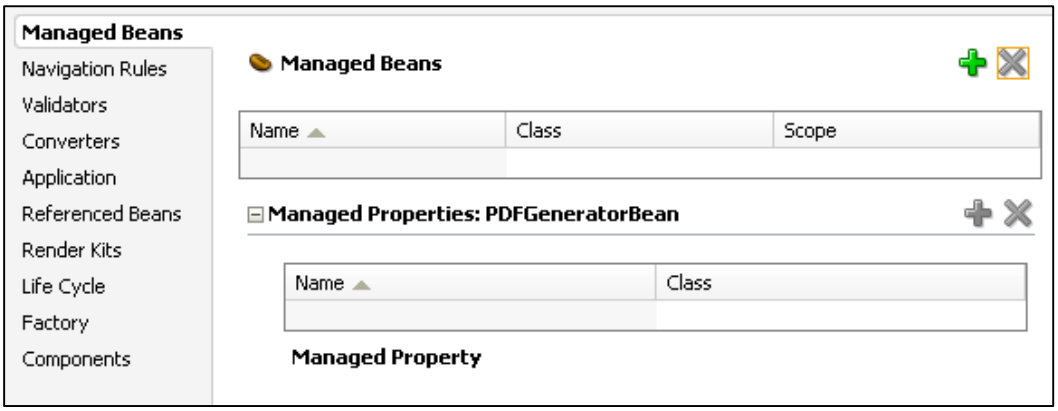
Firstly, we need to add managed beans to the project.

We'll use iTextPDF, a free library to generate PDF from java code. So we need to download this library and add it to the project first.

Download this library from http://itextpdf.com. Unzip the download file and copy the itextpdf-x.x.x.jar to a lib folder under PrepareAcceptancePackForm project folder.

In Application Navigator, right click on PrepareAcceptancePackForm project and select Project Properties. In the Project Properties window, select Libraries and Classpath. Select Add JAR/Library and add the above jar file in the lib subfolder.

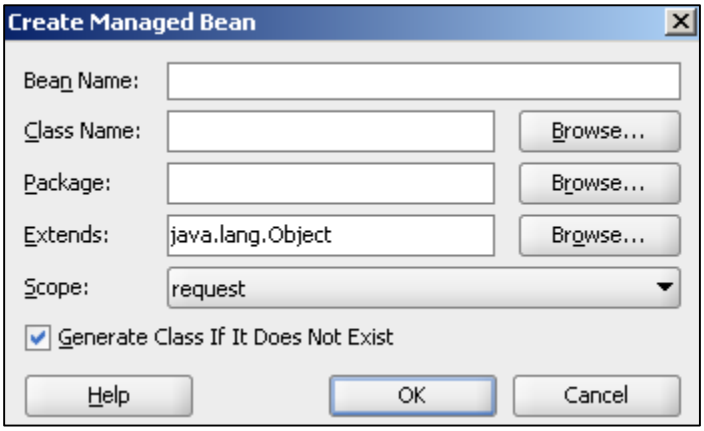
Next, we need to add some managed beans to the project. In Application Navigator, open **faces-config.xml** located in Web Content\WEB-INF folder. Select Overview tab at the bottom.



Click on the green Plus sign button to add a new managed bean. A Create Managed Bean is shown below. Enter the following values and click on OK button.

- Bean Name: DownloadFileBean

- Class Name: DownloadFileBean
- Package: support
- Extends: java.lang.Object
- Scope: request
- General Class If It Does Not Exist: checked.



A DownloadFileBean.java file will be created and located in Application Sources \ support package.

Do similarly to create a second managed bean with name and class name are the same: PDFGeneratorBean.

Finally, we'll write codes for PDF generation.

Open **DownloadFileBean.java**. Copy the source code in Appendix 11.3.1 and override the existing source code of DownloadFilebean.java.

Open **PDFGeneratorBean.java**. Copy the source code in Appendix 11.3.2 and override the existing source code of PDFGeneratorBean.java.

You can explore the source code in the Appendix to understand the detailed logic of PDF creation.

At this point, you have finished creating two managed beans in the project. Now we need to bind the button click event to the managed bean method.

From Component Palette, drag and drop a File Download Action Listener component onto the Generate PDF button. If you open the Structure pane, you can see there is a node called **af:fileDownloadActionListener** added under the **af:commandButton – Generate PDF**.

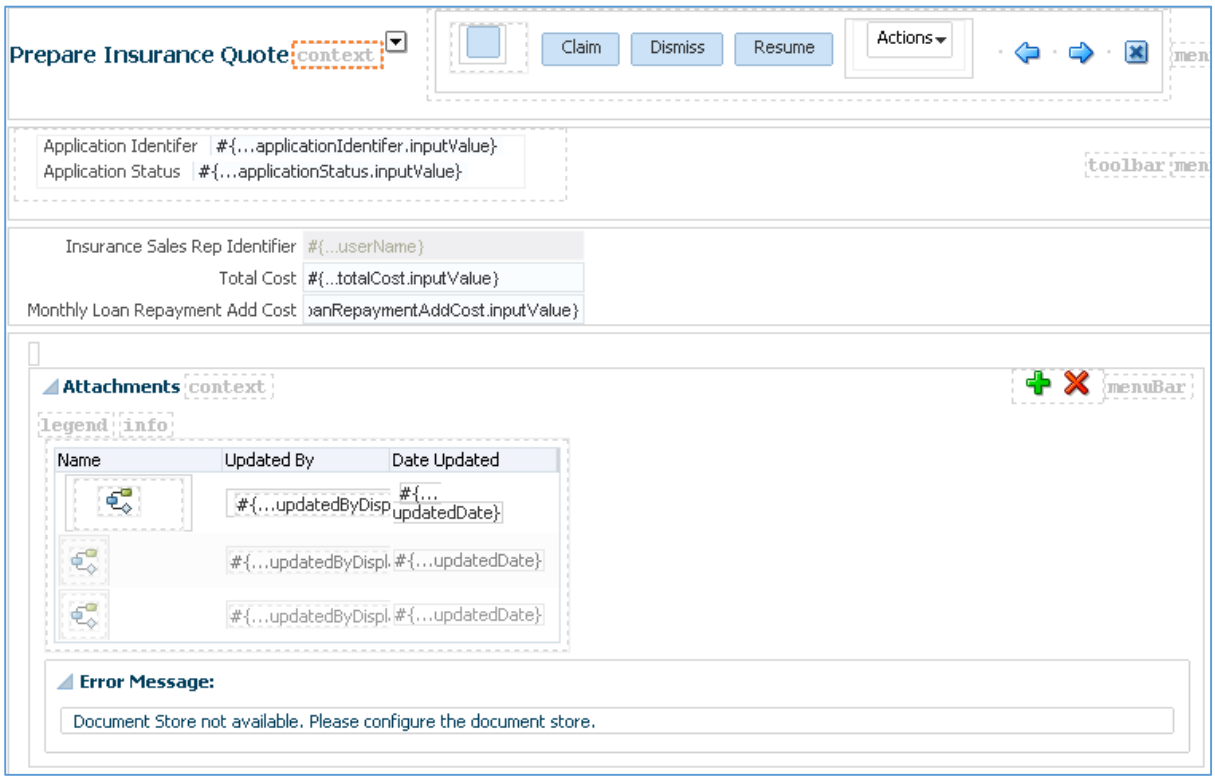
Click on **af:fileDownloadActionListener** node in the Structure pane (this component is not visible in the form). Then open Inspector pane to view its properties.

Enter the following property values for **af:fileDownloadActionListener** component.

- Filename: test.pdf
- Method: #{DownloadFileBean.generatePDF}

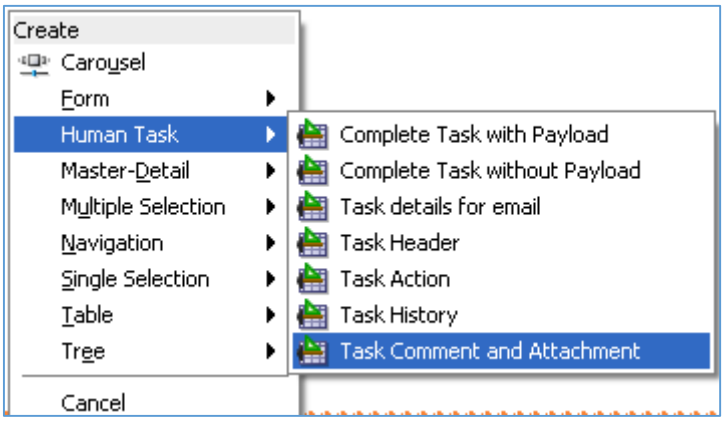
We've finished the implementation for PDF generation function.

7.4.6 Prepare and Send Home Insurance Quote

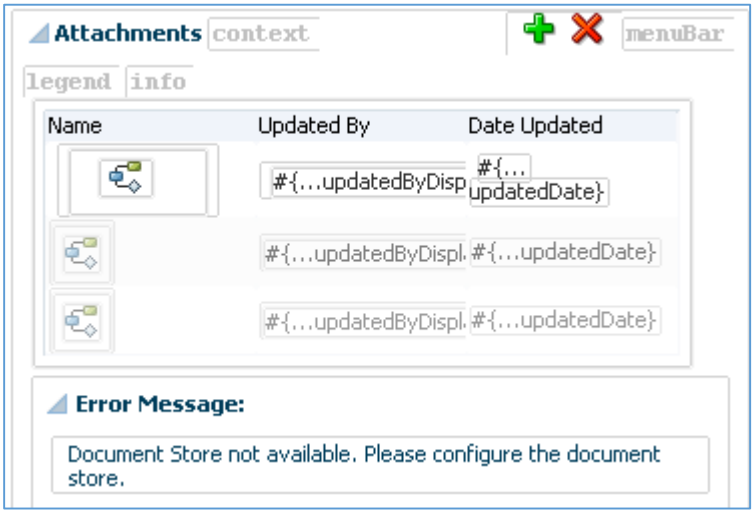


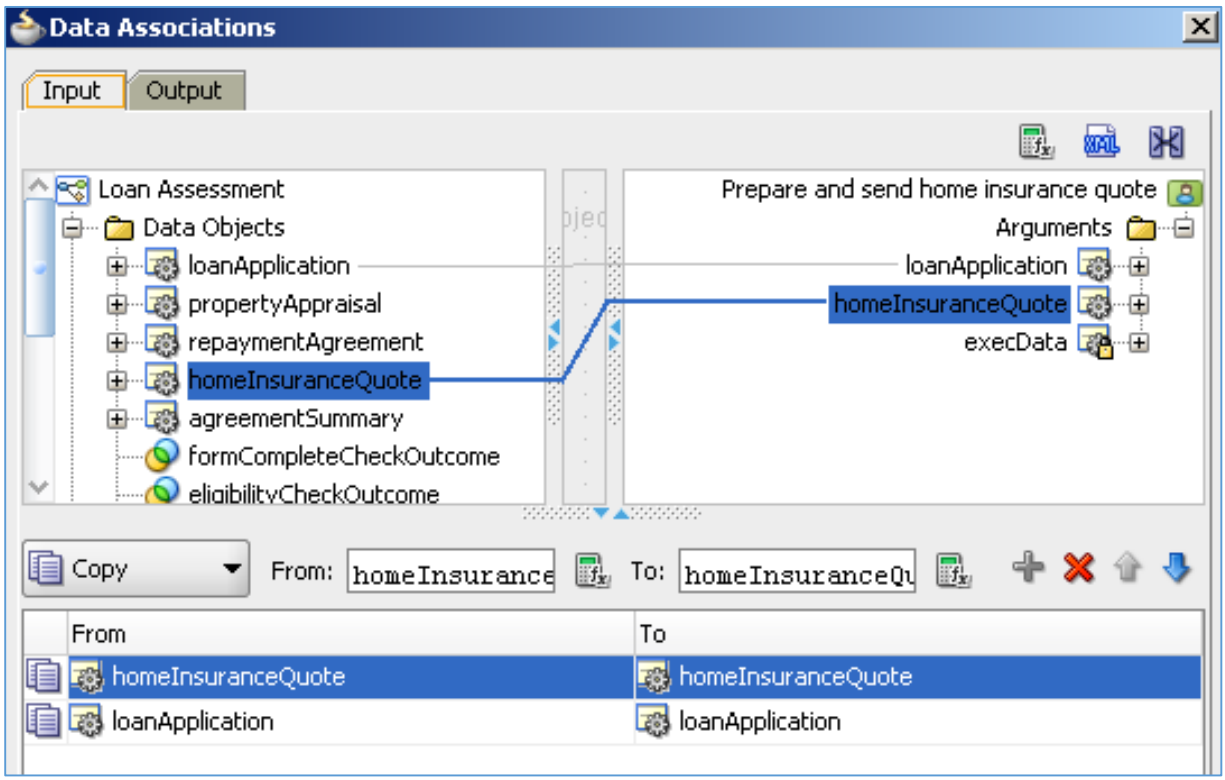
In order to show the logged in username as initial value in the Insurance Sales Rep Identifider field, please do the following. Select the Insurance Sales Rep Identifider field, in the Inspector pane, enter this value in the Value property: **#{securityContext.userName}**

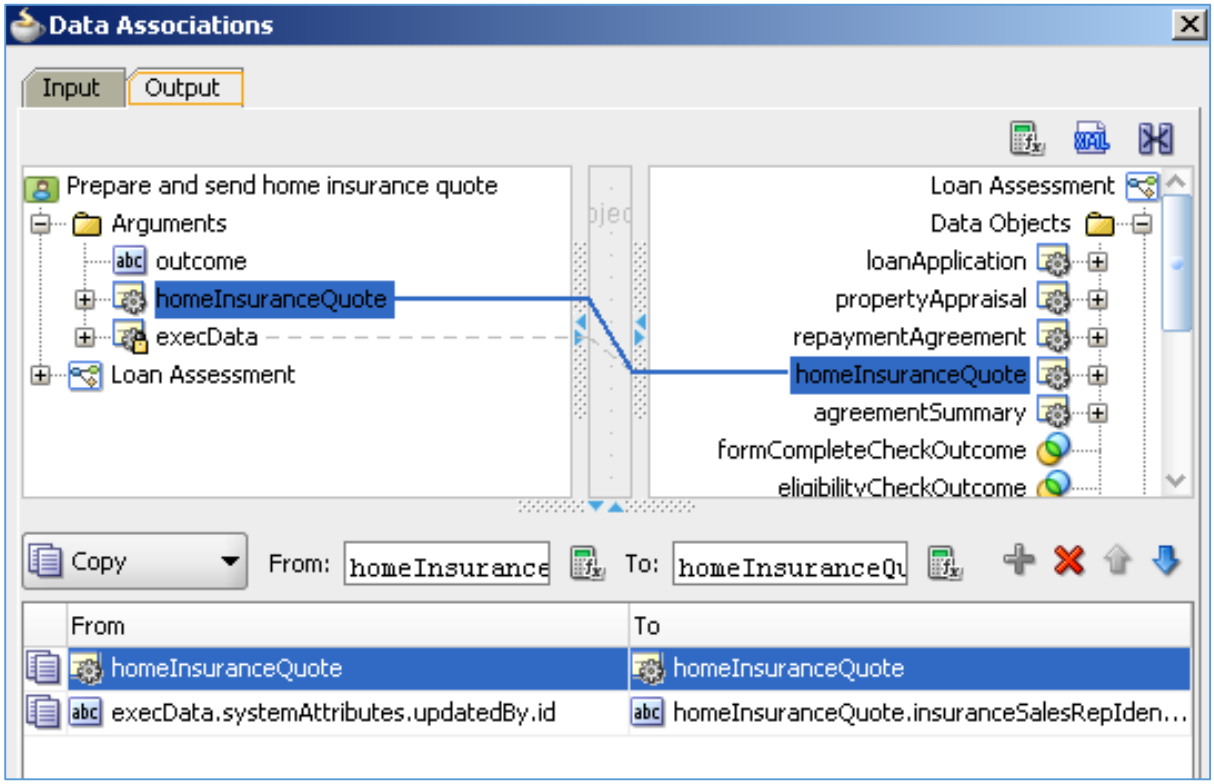
To add Attachments fields as shown above, drag Task node from Data Controls pane to the page position where you want to place it. Select Human Task > Task Comment and Attachment as shown below.



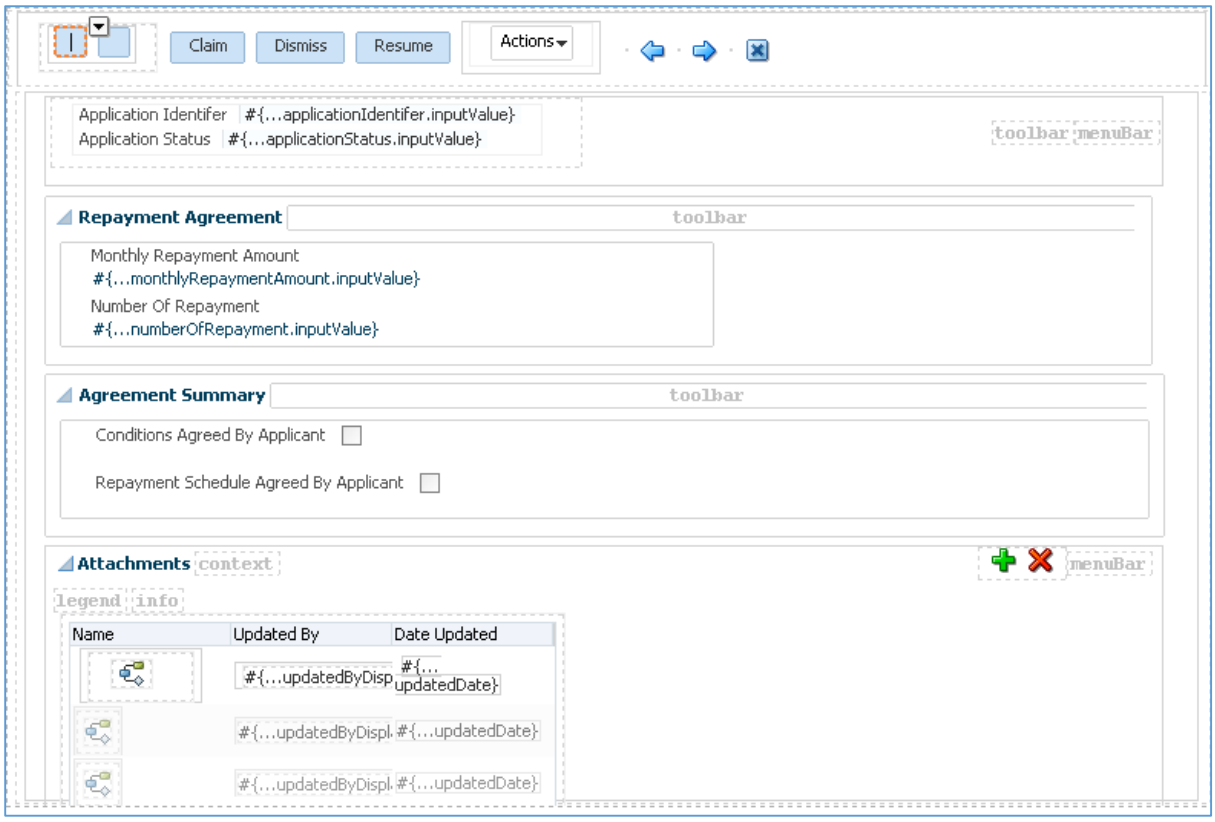
Two blocks including Comments and Attachments will be added to the page. Delete the Comments block and keep the Attachments block. So you have added an Attachments function which enables file upload on the form.

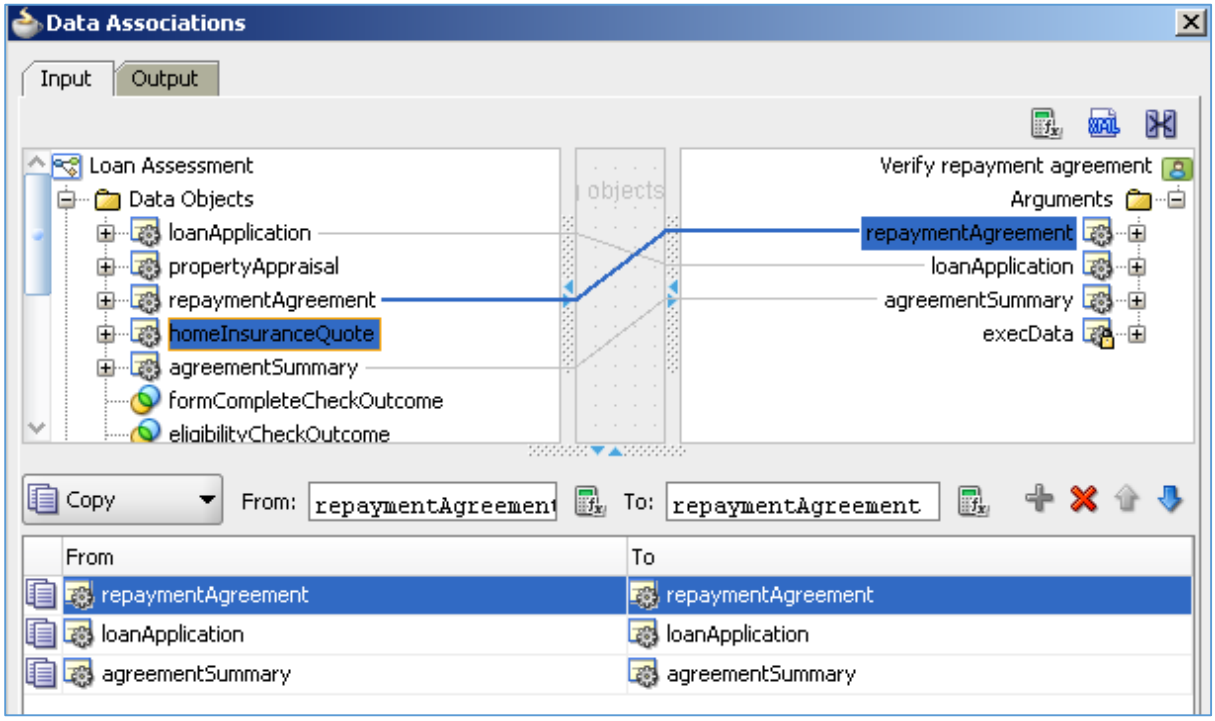

Attachments
context
menuBar
legend
info
Name
Updated By
Date Updated
#{...updatedByDisp
#{...updatedDate}
#{...updatedByDispl
#{...updatedDate}
#{...updatedByDispl
#{...updatedDate}
Error Message:
Document Store not available. Please configure the document store.

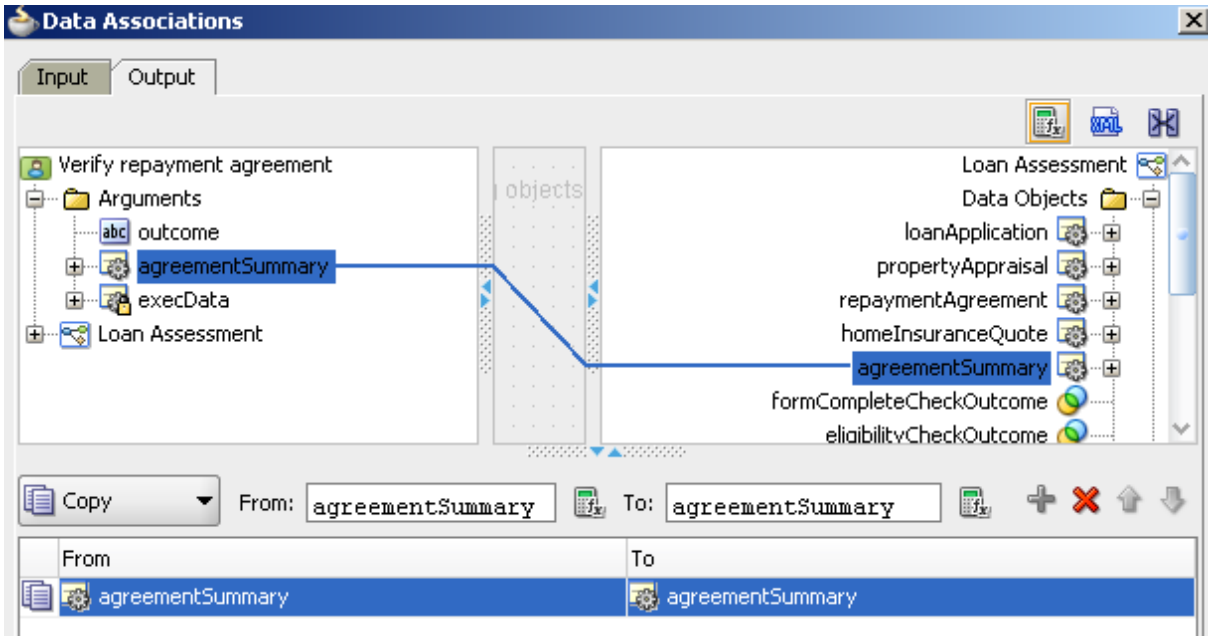

Data Associations
Input
Output
Loan Assessment
Data Objects
loanApplication
propertyAppraisal
repaymentAgreement
homeInsuranceQuote
agreementSummary
formCompleteCheckOutcome
eligibilityCheckOutcome
Prepare and send home insurance quote
Arguments
loanApplication
homeInsuranceQuote
execData
Copy
From:
homeInsurance
To:
homeInsuranceQu
From
To
homeInsuranceQuote
homeInsuranceQuote
loanApplication
loanApplication



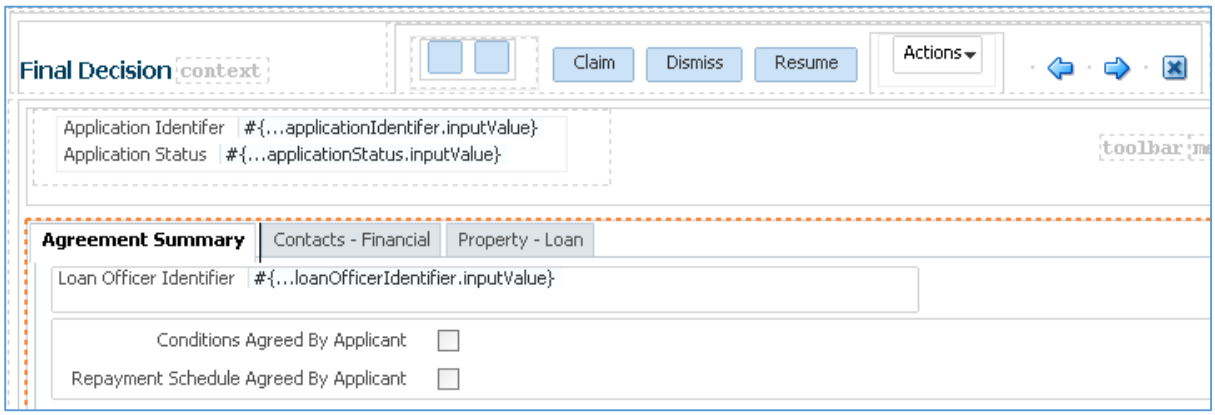
7.4.7 Verify Repayment Agreement

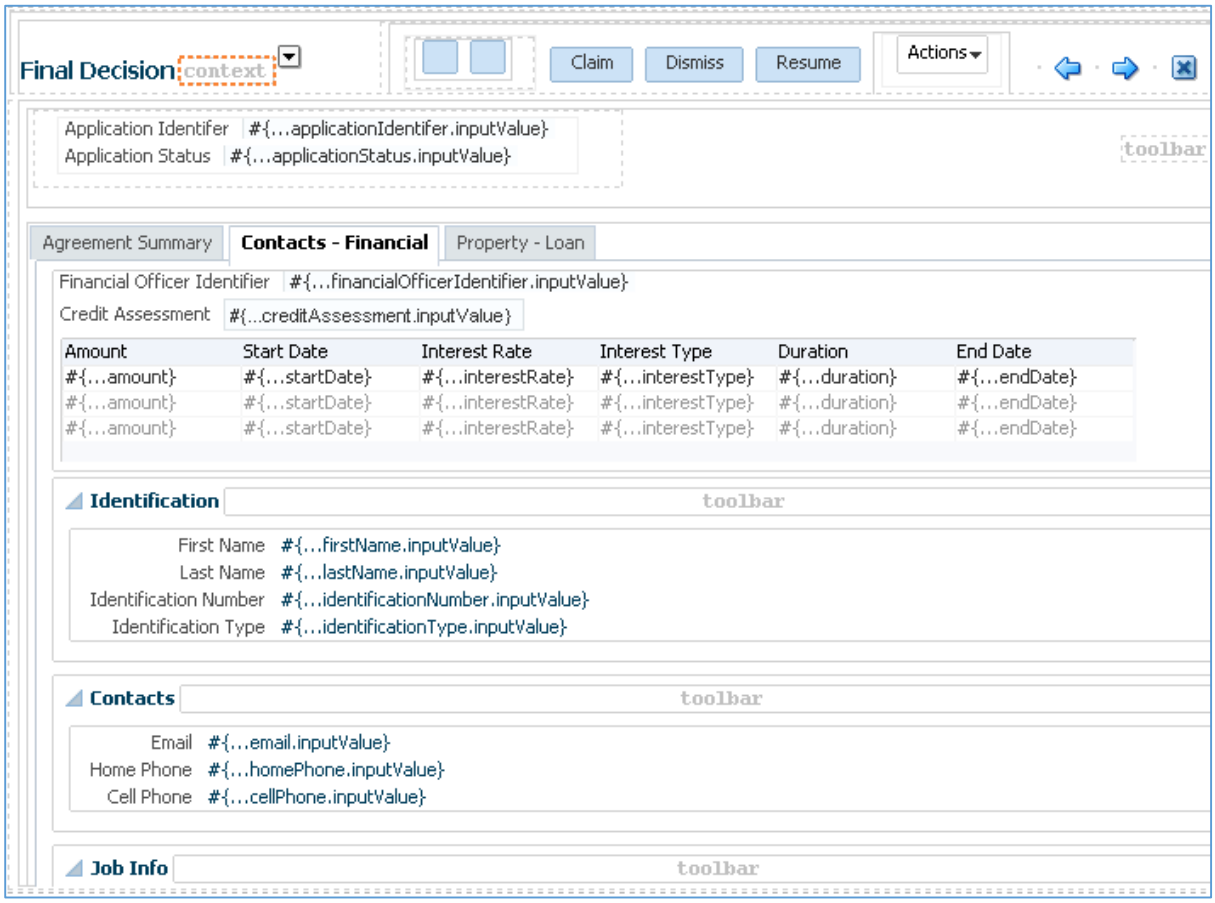


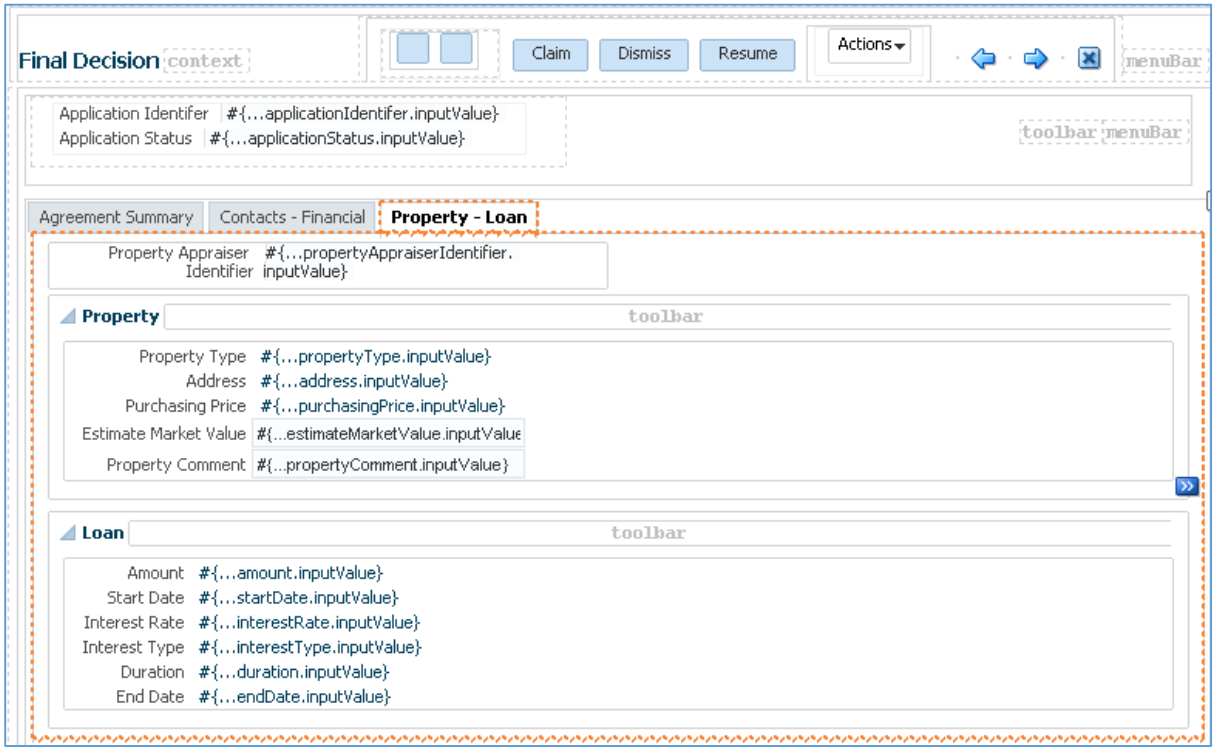

Data Associations
Input
Output
Loan Assessment
Data Objects
loanApplication
propertyAppraisal
repaymentAgreement
homeInsuranceQuote
agreementSummary
formCompleteCheckOutcome
eligibilityCheckOutcome
objects
Verify repayment agreement
Arguments
repaymentAgreement
loanApplication
agreementSummary
execData
Copy
From: repaymentAgreement
To: repaymentAgreement
From
To
repaymentAgreement
repaymentAgreement
loanApplication
loanApplication
agreementSummary
agreementSummary

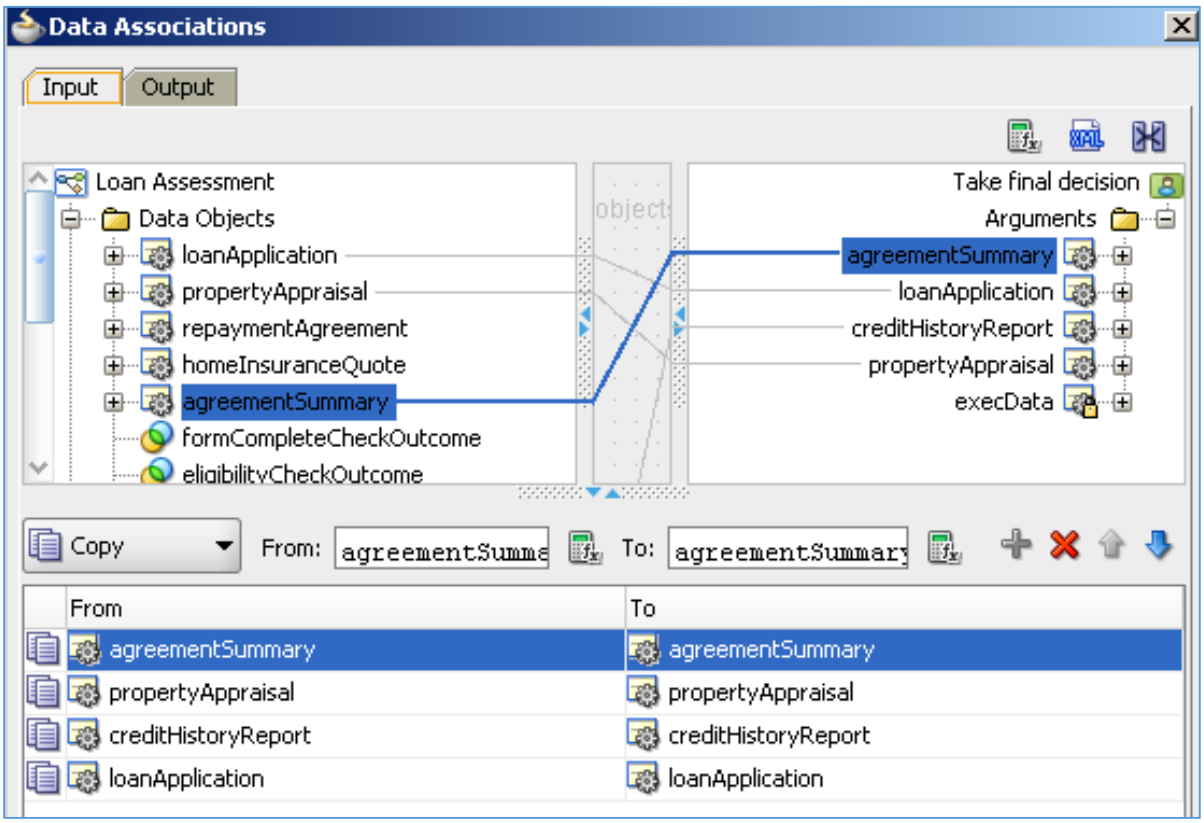

Data Associations
Input
Output
Verify repayment agreement
Arguments
outcome
agreementSummary
execData
Loan Assessment
objects
Loan Assessment
Data Objects
loanApplication
propertyAppraisal
repaymentAgreement
homeInsuranceQuote
agreementSummary
formCompleteCheckOutcome
eligibilityCheckOutcome
Copy
From: agreementSummary
To: agreementSummary
From
To
agreementSummary
agreementSummary

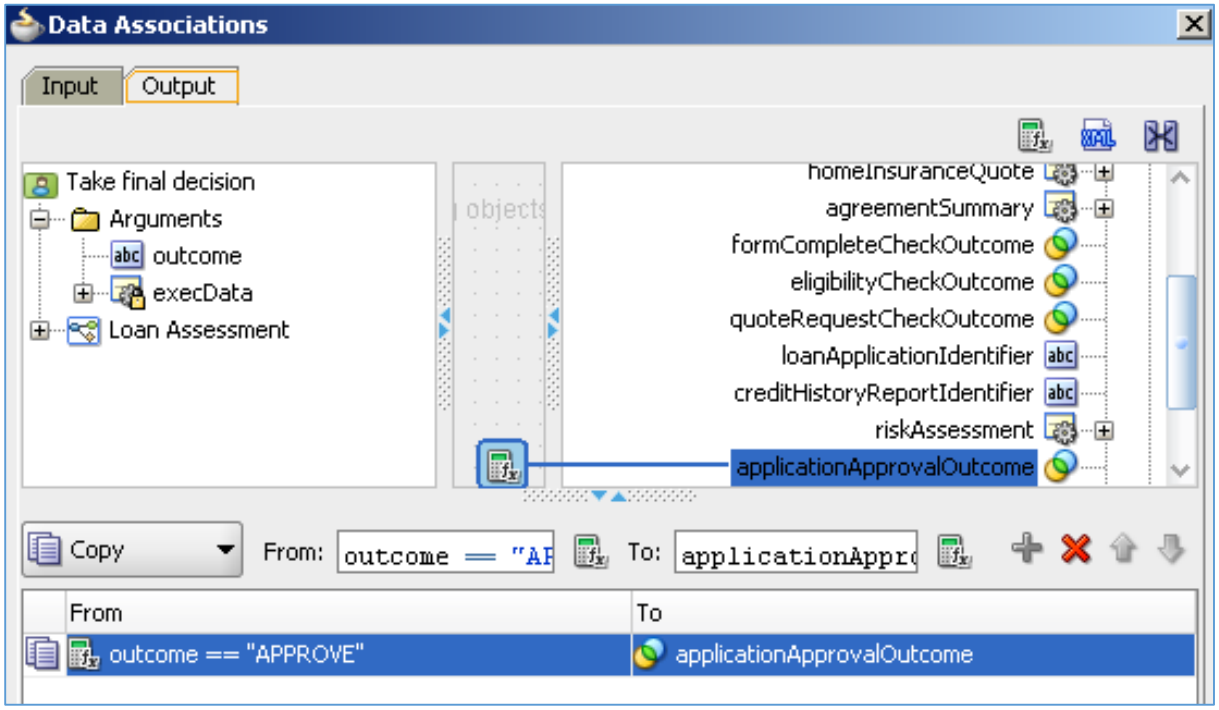
7.4.8 Take Final Decision






Final Decision
context
Claim
Dismiss
Resume
Actions
menuBar
Application Identifer #{...applicationIdentifer.inputValue}
Application Status #{...applicationStatus.inputValue}
toolbar
menuBar
Agreement Summary
Contacts - Financial
Property - Loan
Property Appraiser Identifier #{...propertyAppraiserIdentifier.inputValue}
Property
toolbar
Property Type #{...propertyType.inputValue}
Address #{...address.inputValue}
Purchasing Price #{...purchasingPrice.inputValue}
Estimate Market Value #{...estimateMarketValue.inputValue
Property Comment #{...propertyComment.inputValue}
Loan
toolbar
Amount #{...amount.inputValue}
Start Date #{...startDate.inputValue}
Interest Rate #{...interestRate.inputValue}
Interest Type #{...interestType.inputValue}
Duration #{...duration.inputValue}
End Date #{...endDate.inputValue}


Data Associations
Input
Output
Loan Assessment
Data Objects
loanApplication
propertyAppraisal
repaymentAgreement
homeInsuranceQuote
agreementSummary
formCompleteCheckOutcome
eligibilityCheckOutcome
object
Take final decision
Arguments
agreementSummary
loanApplication
creditHistoryReport
propertyAppraisal
execData
Copy
From: agreementSumma
To: agreementSummary
From
To
agreementSummary
agreementSummary
propertyAppraisal
propertyAppraisal
creditHistoryReport
creditHistoryReport
loanApplication
loanApplication



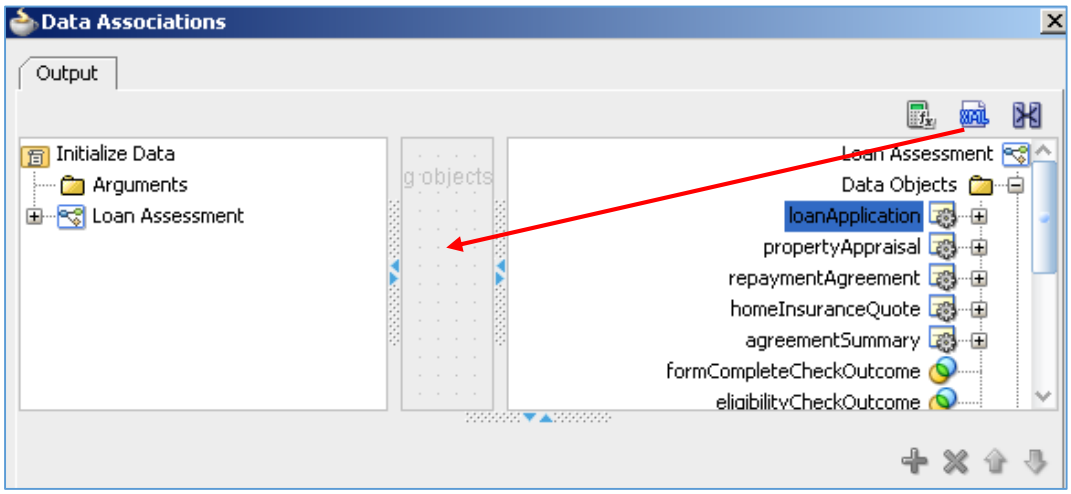
7.5 Implement Script Tasks

7.5.1 Initialize Data

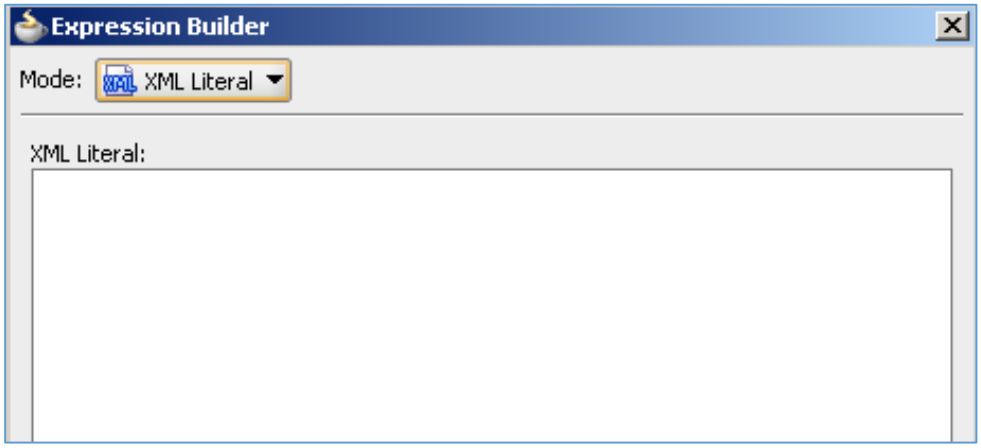
The first form is Loan Application Form which input is Loan Application object. This is a large compound object containing other compound objects (multiple levels). Oracle BPM cannot initialize this complex object and we have to do it explicitly and manually.

Double click on Initialize Data task. In the Properties window, open Implementation tab. Click on Data Associations.

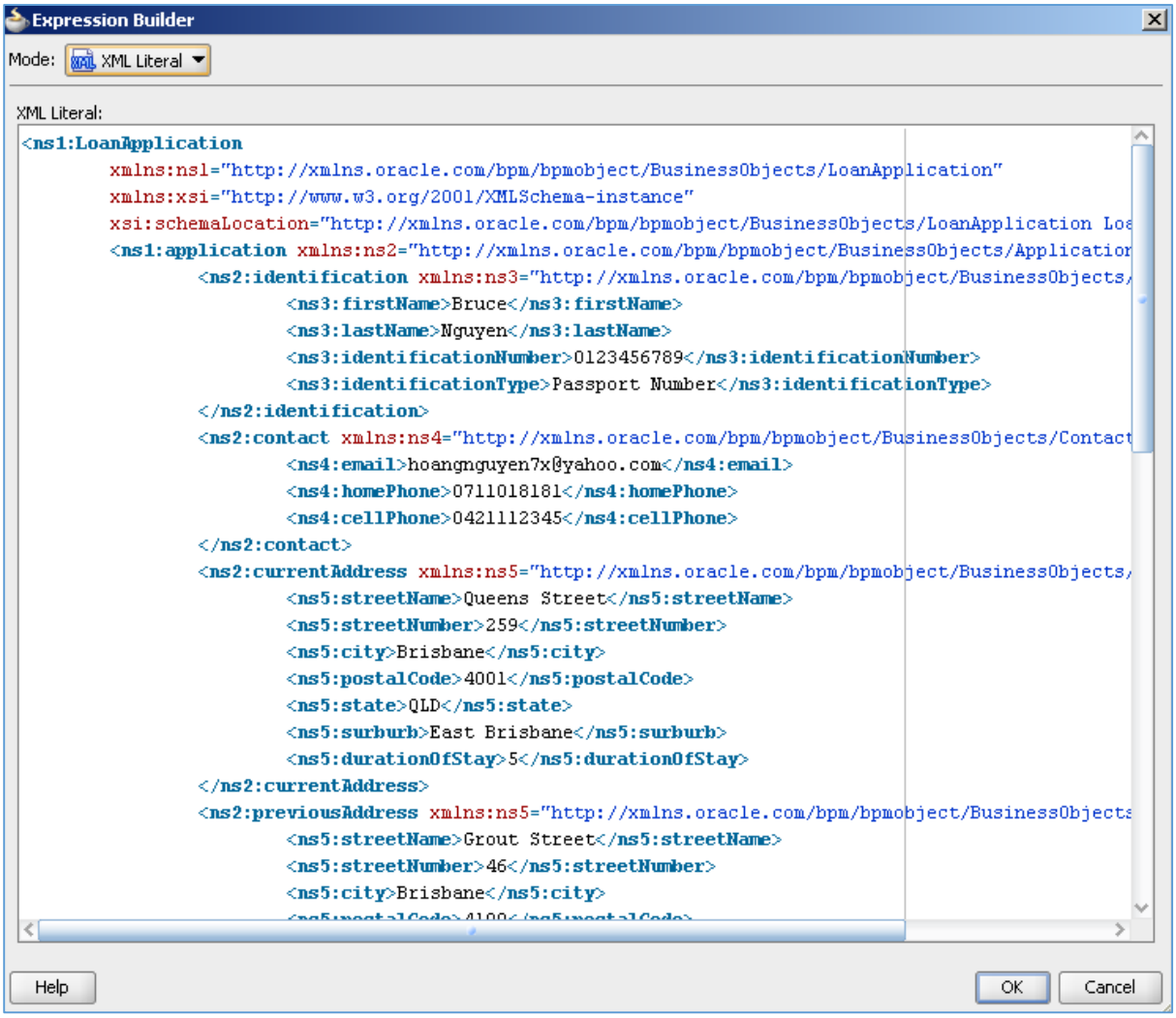
In the Data Associations window (with Output tab only), select an XML Literal symbol and drag and drop it on to the space as shown below.



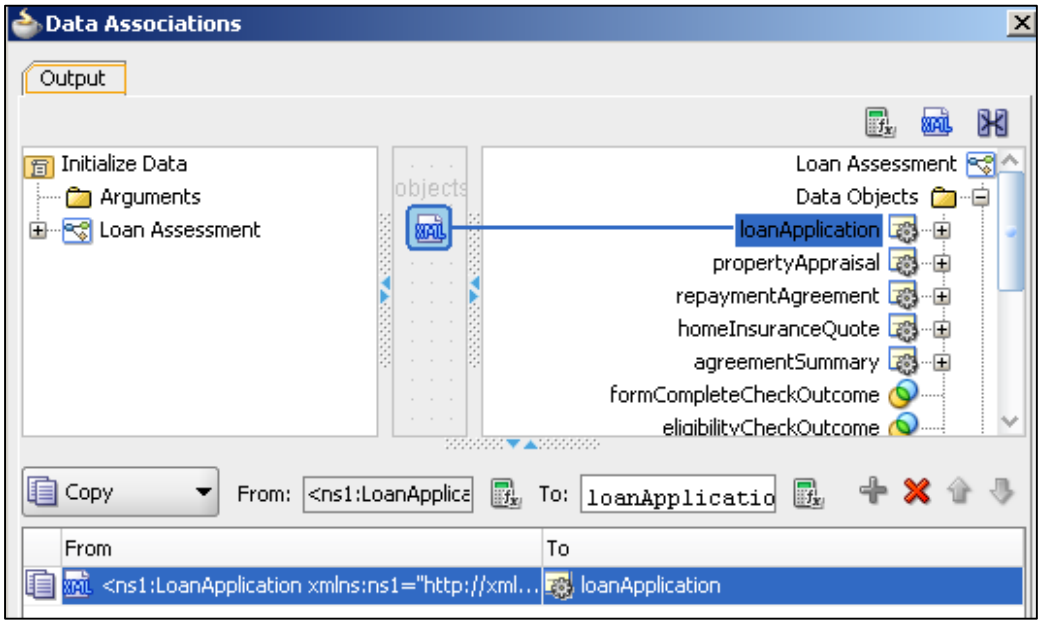
An Expression Builder pops up as below.



Copy the Initialization Data in Appendix 11.2 and paste it into the Expression Builder. Then click on the OK button.



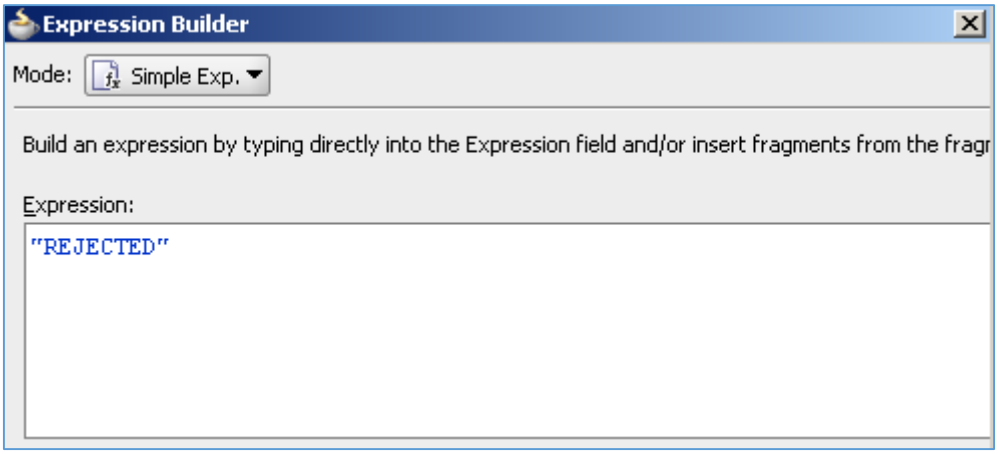
In Data Associations window, drag a mapping line from XML Literals symbol to loanApplication process variable. The complete Data Associations window looks like below.

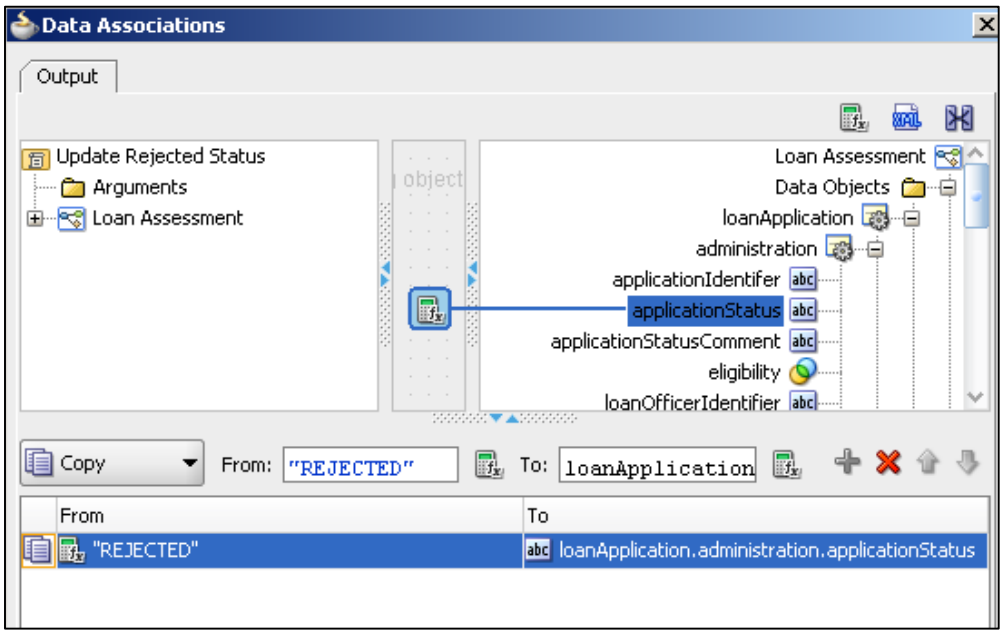


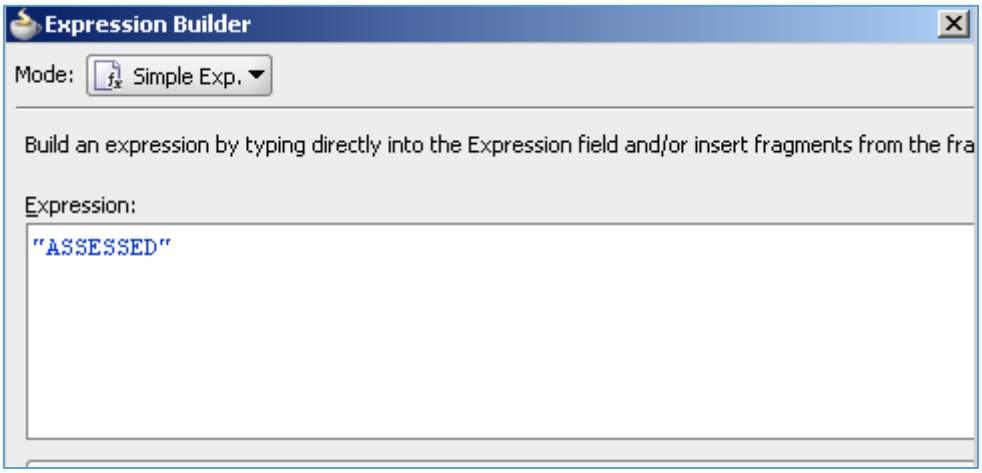
That we have finished the Initialize Data task to assign at run-time initial values to loanApplication variable, so that the first Enter Application Form will be displayed properly.

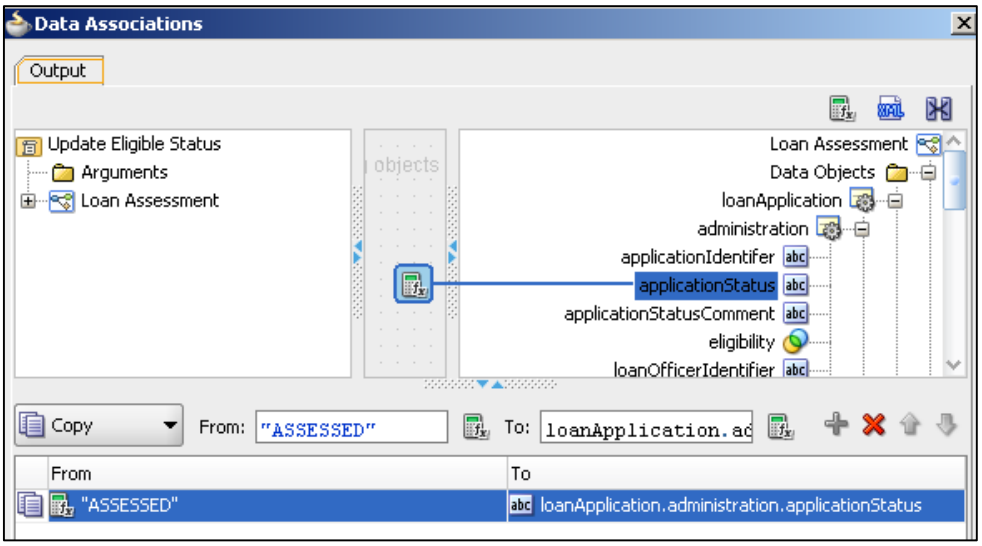
7.5.2 Update Rejected/Eligible status

Perform similarly as described above for the Update Rejected Status task and Update Eligible Task.



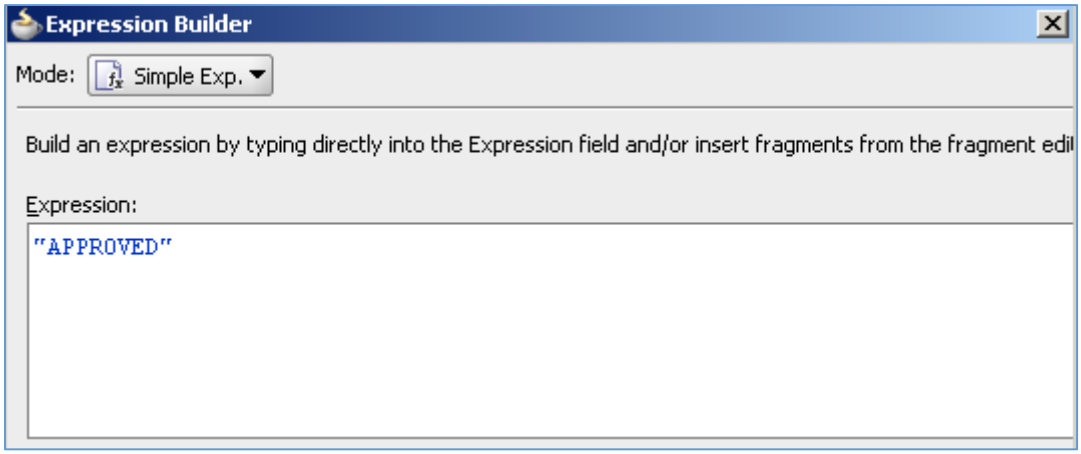

Data Associations
Output
Update Rejected Status
Arguments
Loan Assessment
object
Loan Assessment
Data Objects
loanApplication
administration
applicationIdentifer
applicationStatus
applicationStatusComment
eligibility
loanOfficerIdentifier
Copy
From: "REJECTED"
To: loanApplication
From
To
"REJECTED"
loanApplication.administration.applicationStatus

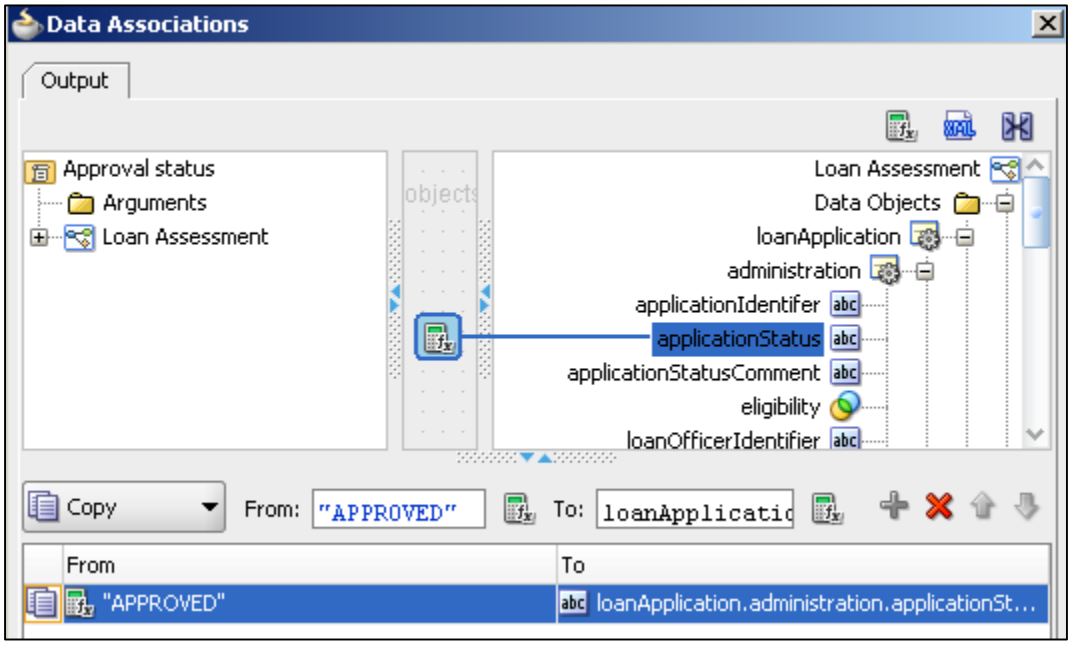

Expression Builder
Mode: Simple Exp.
Build an expression by typing directly into the Expression field and/or insert fragments from the fra
Expression:
"ASSESSED"

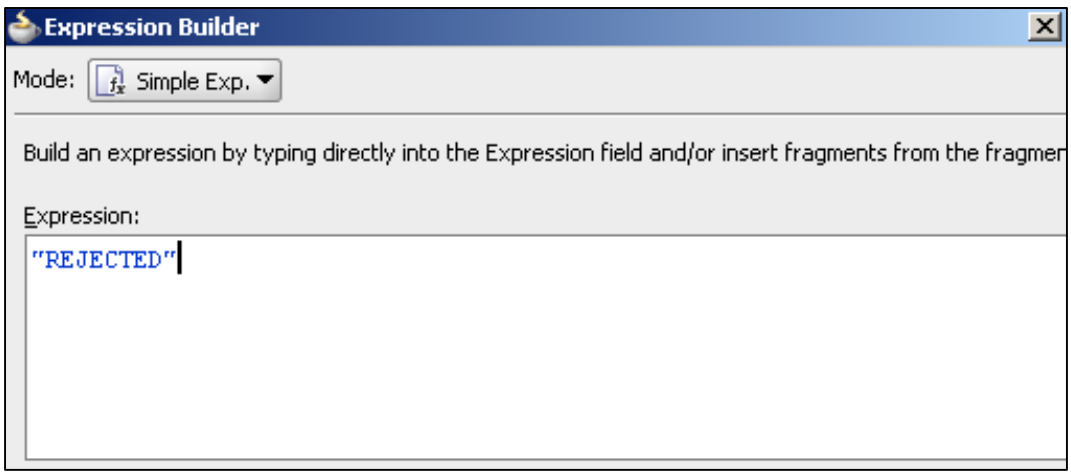

Data Associations
Output
Update Eligible Status
Arguments
Loan Assessment
objects
Loan Assessment
Data Objects
loanApplication
administration
applicationIdentifer
applicationStatus
applicationStatusComment
eligibility
loanOfficerIdentifier
Copy
From: "ASSESSED"
To: loanApplication.ac
From
To
"ASSESSED"
loanApplication.administration.applicationStatus

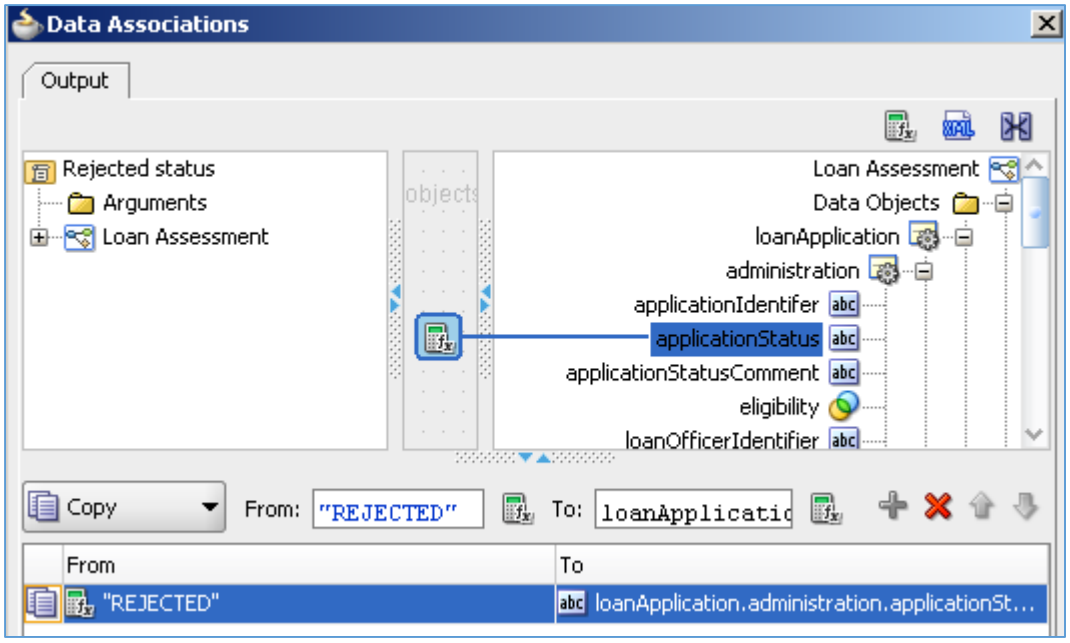
7.5.3 Update Approval/Rejected loan status

Perform similarly for Approval status task and Rejected status task.





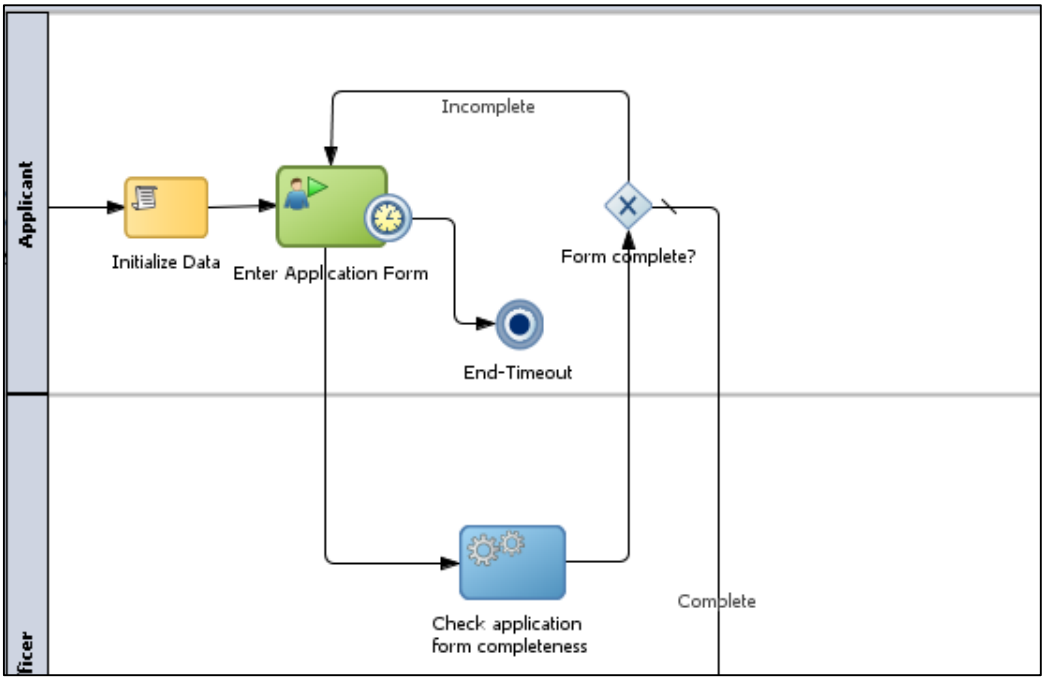




7.6 Implement Service Tasks

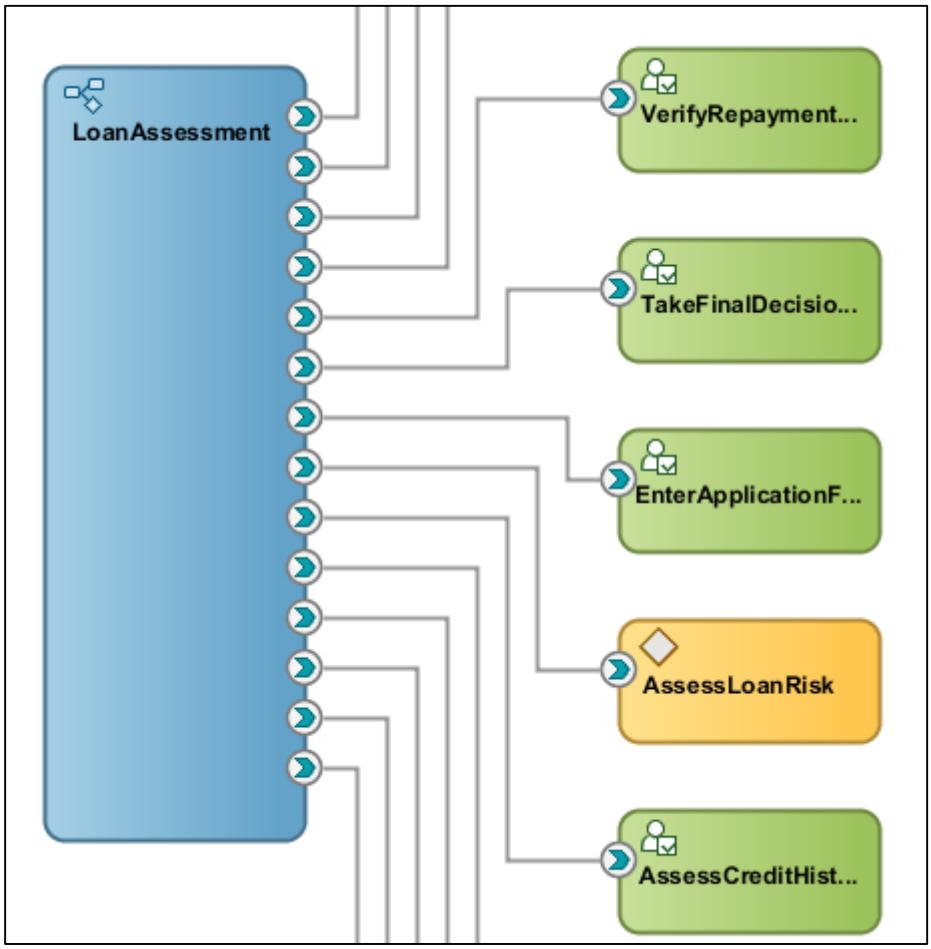
We have one service task called “Check application form completeness”. According to process requirement, this task is automated by the system. So it must be implemented through custom Java code to check application form completeness. This is done via Java Embedding object of BPEL process which allows to insert custom Java code in BPM.

Open the BPMN diagram completed in section 6. Part of the process model is shown below with “Check application form completeness” task to be implemented.



To add BPEL process, we need to switch to the SOA Composite view of this process. On the header bar of BPMN Editor, click on the button located in the top right corner.

The Composite view of BPMN process will be shown as follows.



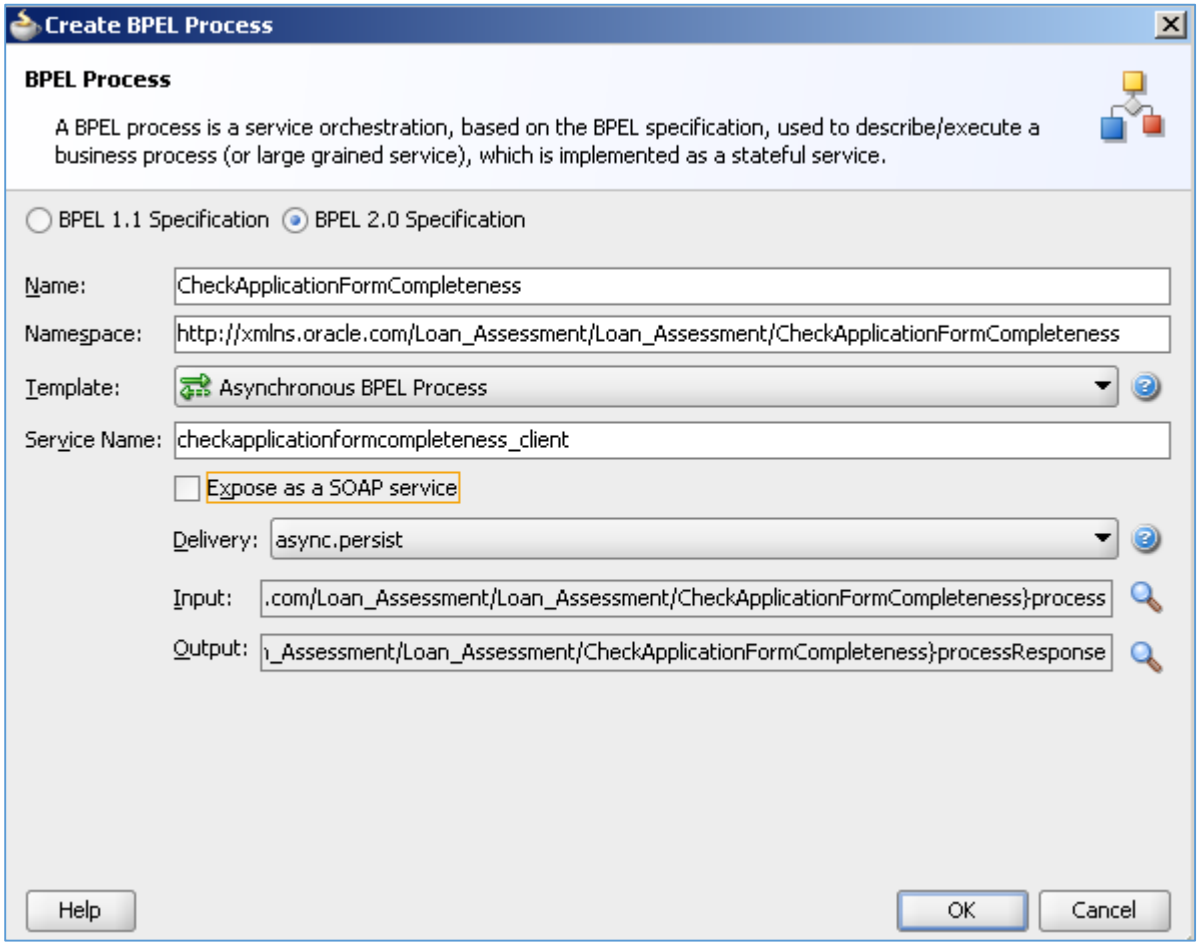
Note that the BPMN process is represented as a largest symbol here (Loan Assessment). It is connected with human tasks and one business rule task. This is a visualization of SOA Composite application complying SCA architecture. It shows clearly BPM application is in principle an SOA Composite application.

Now, we need to add a BPEL process into this composite application. From the Component Palette, drag and drop a BPEL Process element on to the composite editor.

A “Create BPEL Process” window appears as follows. Enter the following values in the window:

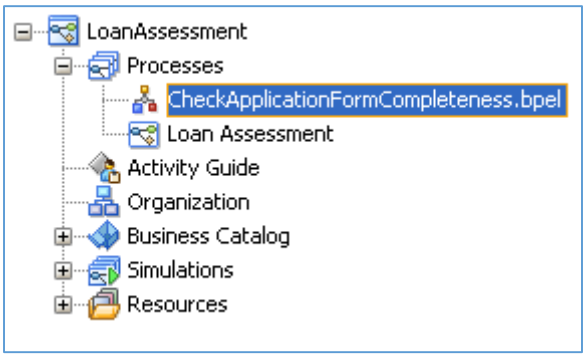
- BPEL 2.0 Specification: selected
- Name: CheckApplicationFormCompleteness
- Template: Asynchronous BPEL Process
- Expose as a SOAP service: unchecked

Then, click on the OK button.

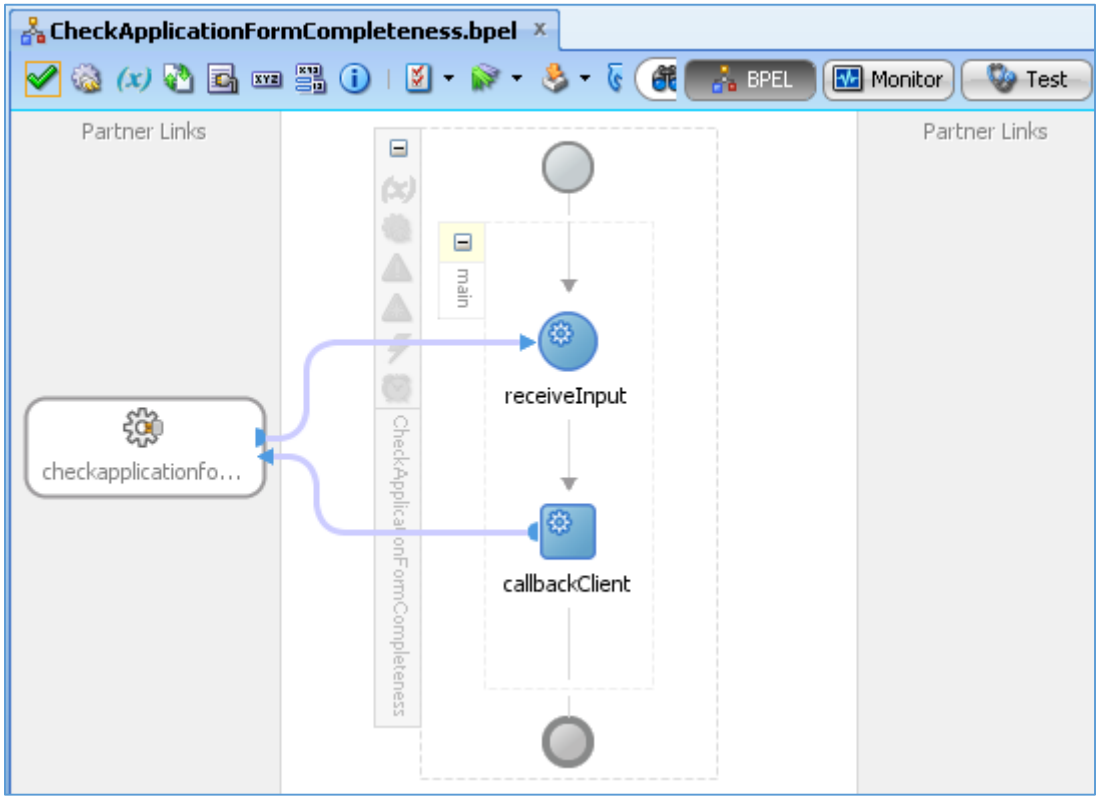


A new object is added to the composite application which represents the BPEL process.

In addition, a BPEL process is also added under the Processes node in the BPM Project Navigator.

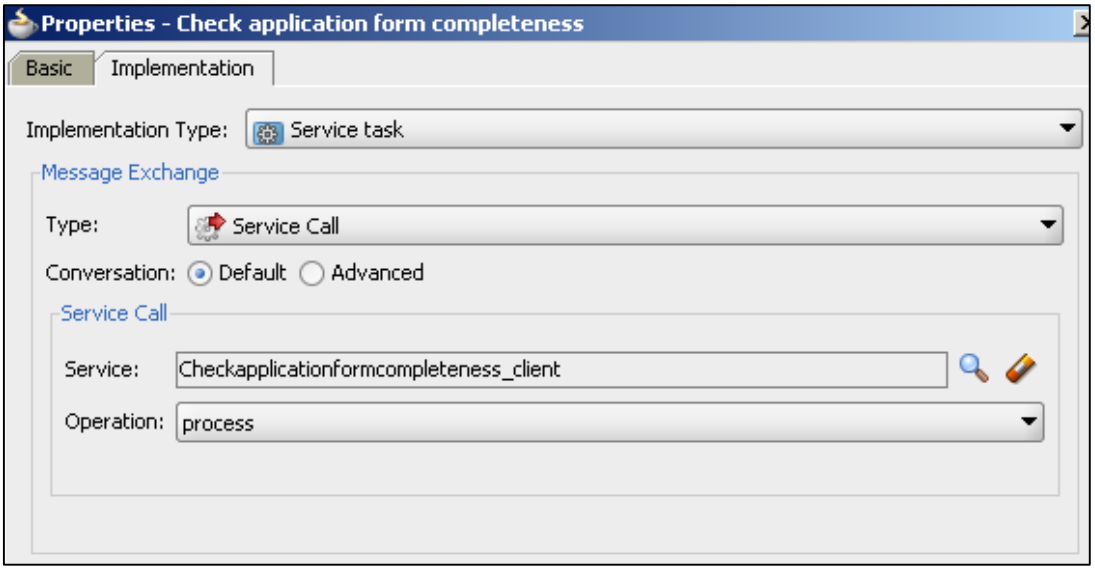


If you double click on the **CheckApplicationFormCompleteness.bpel** to open it, you can view the BPEL process diagram as shown below. Note that it has a Partner Link named **Checkapplicationformcompleteness_client**.



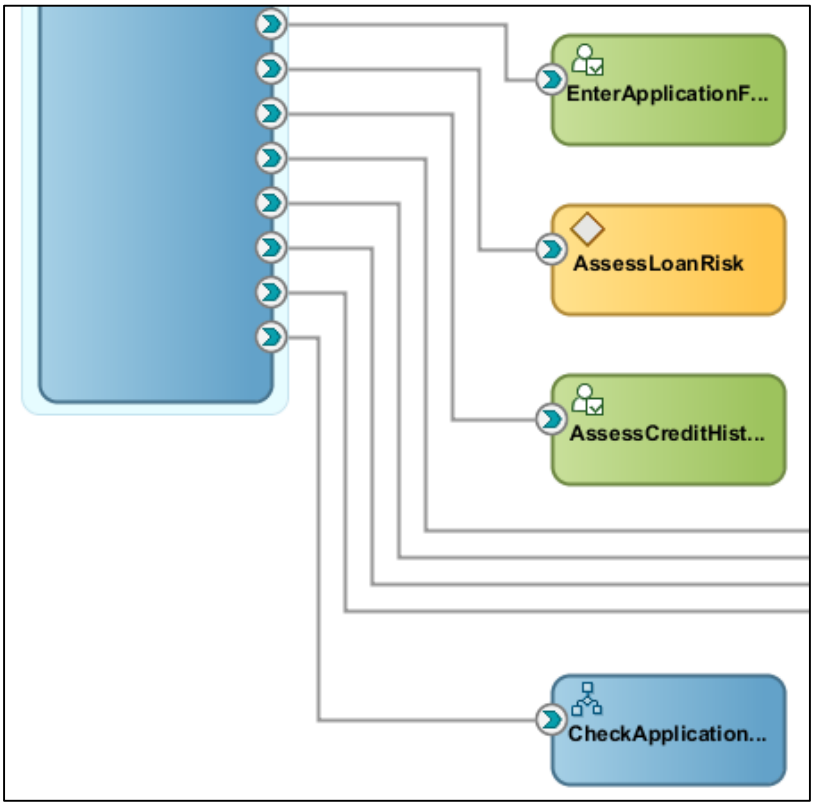
Now we will associate the Service Task to this BPEL process.

In BPMN diagram, double click on the “Check application form completeness” service task to open its Properties window. In Implementation tab, select Type as Service Call. Click on Search button of Service field to search for an existing service. Select **Checkapplicationformcompleteness_client** from the list and click on OK button.



That has associated the service task with the BPEL process as its implementation.

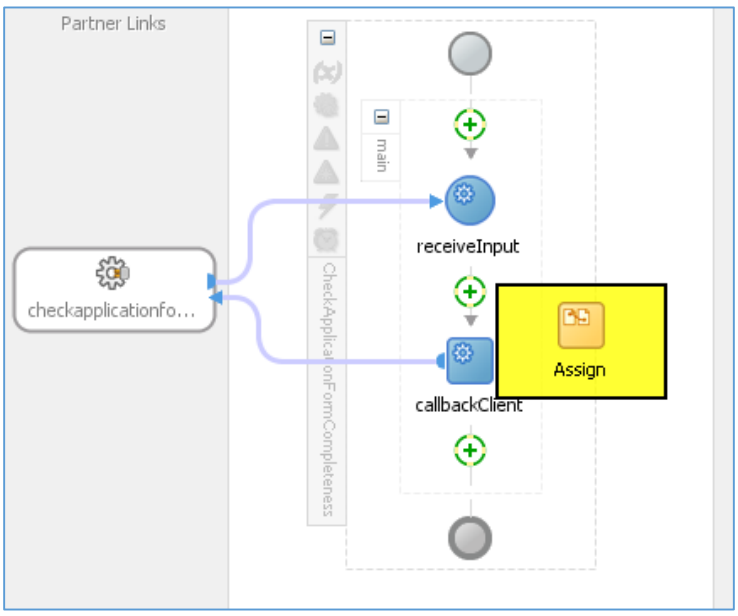
If you open the SOA Composite at the moment, you can see there is a connection line from the BPMN process component to the BPEL process component, as shown below.



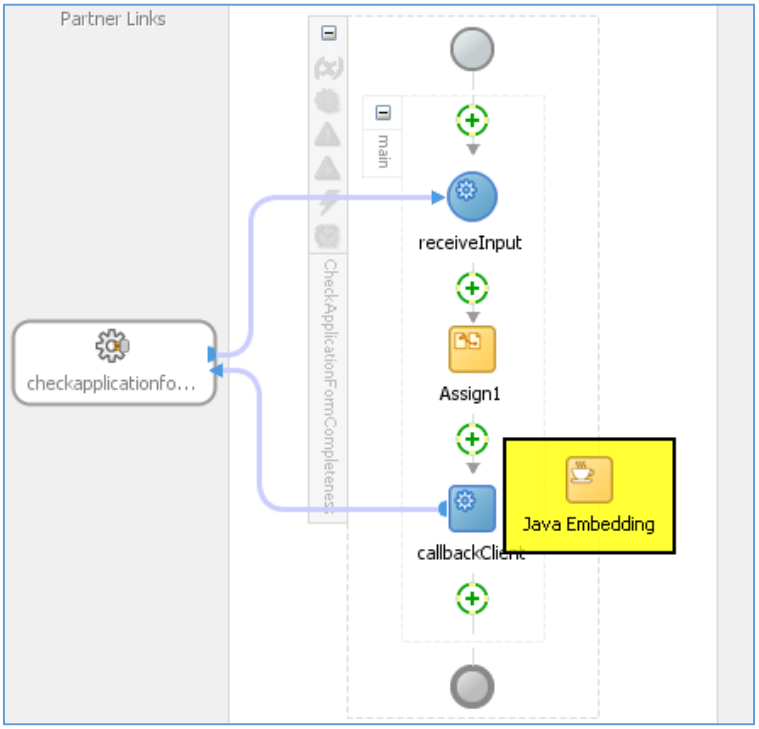
Now, we need to insert Java code into the BPEL process to implement the application form completeness checking logic.

Open **CheckApplicationFormCompleteness.bpel** process.

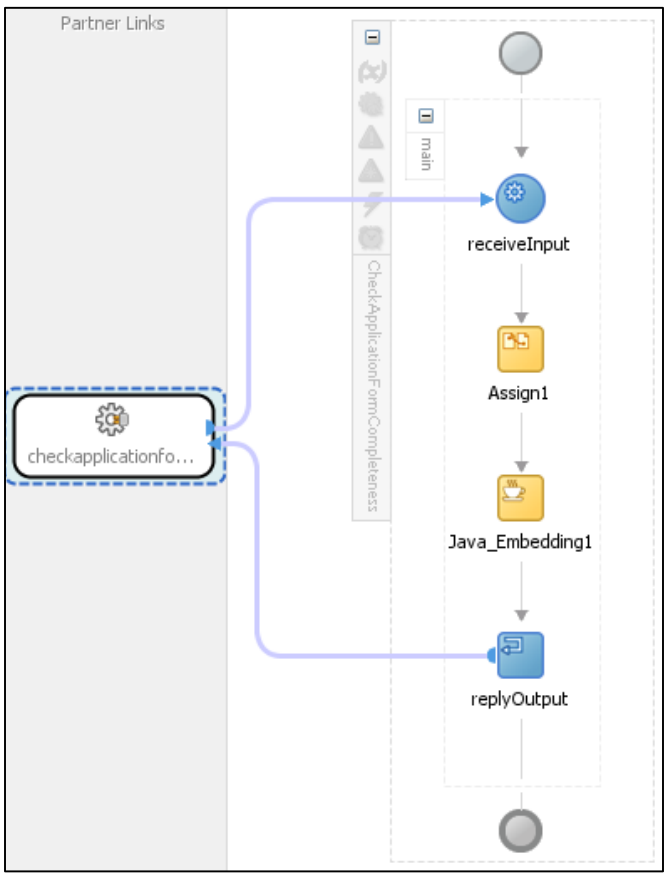
From the Component Palette, drag and drop an Assign activity (located in Basic Activities section) to the process between **receiveInput** and **callbackClient**, as shown below. There will be markers appear for you to drop the component on them (round plus sign).



Similarly, drag and drop a Java Embedding component (located in Oracle Extensions) under the **callbackClient** activity.

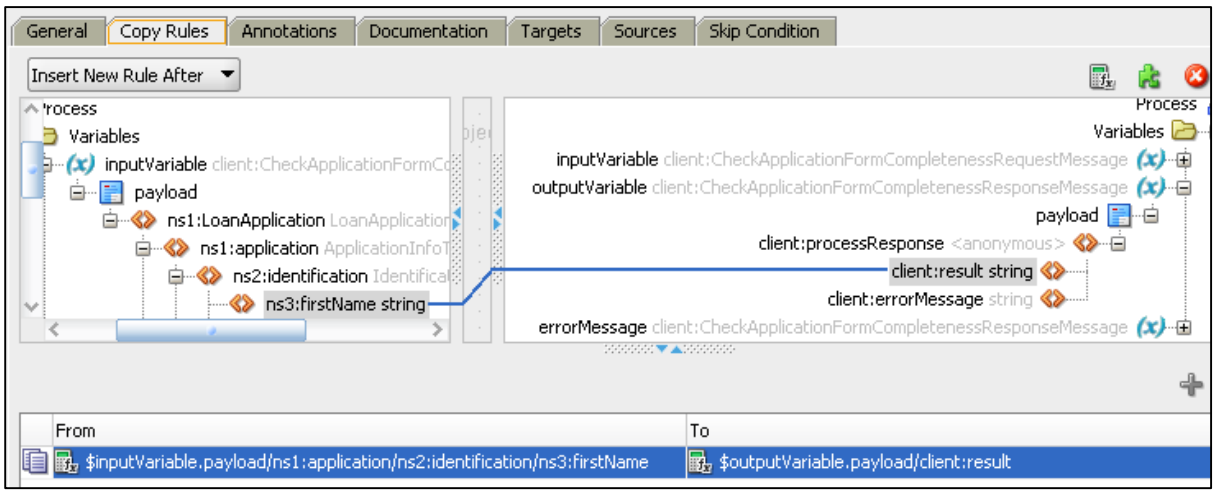


The new process should look like below.

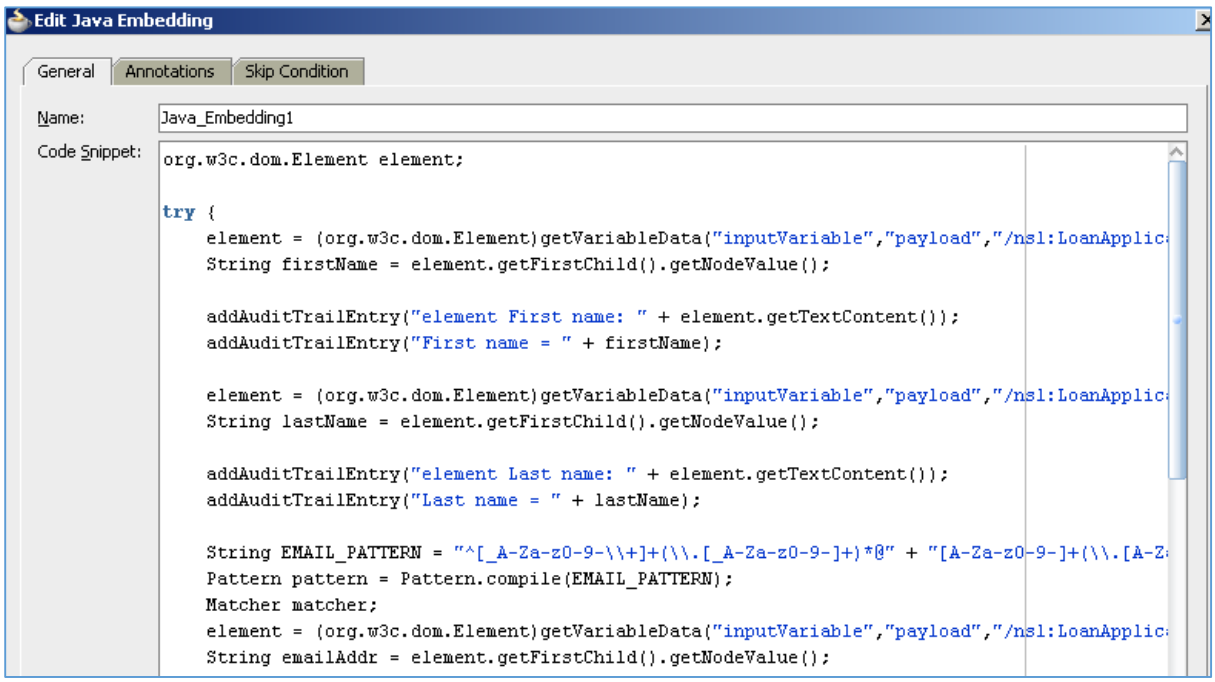


Double click on Assign activity. In the “Edit Assign” window, drag to connect **firstName** to the **resultString** under **processResponse** of **outputVariable**, as shown below. The reason of this

assignment is outputVariable must be initialized to any value before it is referenced in BPEL process. Here, we assign the first name in the input variable as a dummy value to the outputVariable.



Now, we implement the completeness checking code. Double click on the Java Embedding activity in the BPEL process diagram. Then, copy and paste the code in Appendix 11.4 to Code Snippet editor, as shown below.



There is still one small step to do, we need to insert declaration statement of the code libraries used. From the BPEL diagram, click on the Source tab at the bottom of the diagram editor to open its XML file.

Search for the import keyword and insert the following three lines under that.

```
<import location="org.w3c.dom.Element" importType="http://schemas.oracle.com/bpel/extension/java" />
<import location="java.util.regex.Matcher" importType="http://schemas.oracle.com/bpel/extension/java" />
<import location="java.util.regex.Pattern" importType="http://schemas.oracle.com/bpel/extension/java" />
```

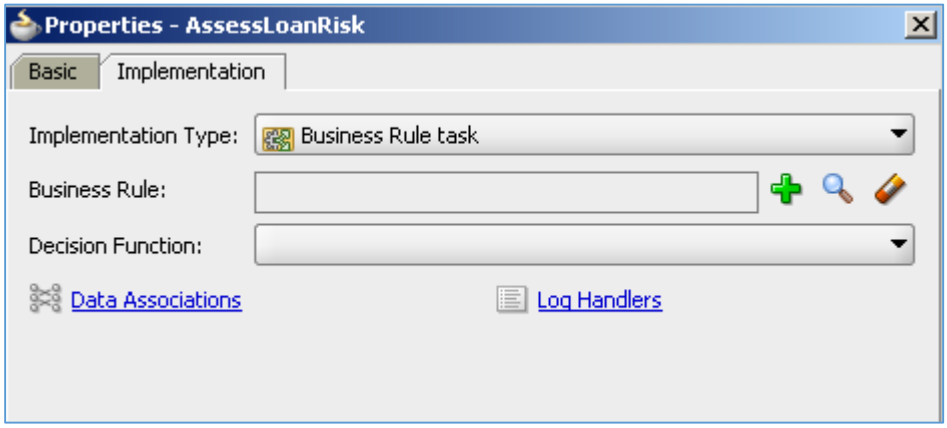
7.7 Implement Business Rules

Our process requirement is the system will translate the credit grading scheme in form of B, BB, BBB, A, AA, AAA to a credit rating point scheme from 0 to 100.

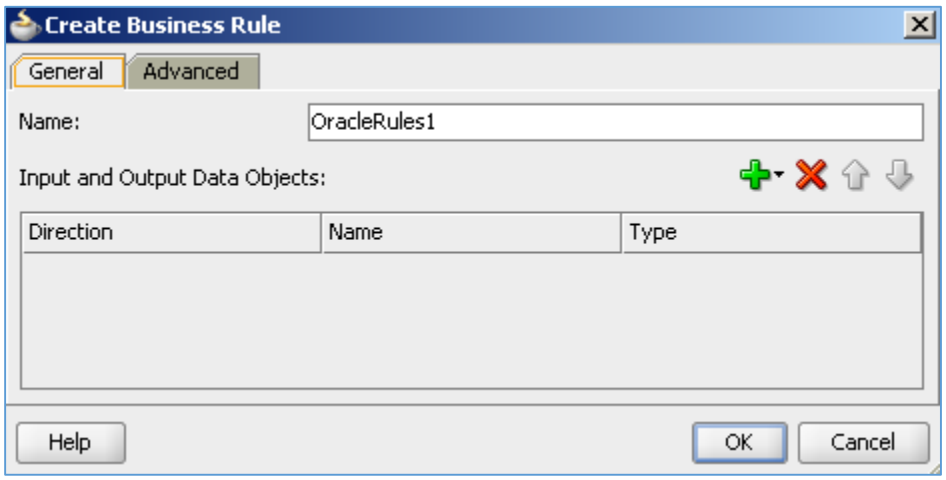
In order to be able to change this rule in the future easily, we can implement this requirement with business rule engine.

Open the BPMN diagram, double click on the AssessLoanRisk activity to open its Properties window.

Click on the green plus sign button.



In Create Business Rule popup, enter a rule name: AssessLoanRisk.



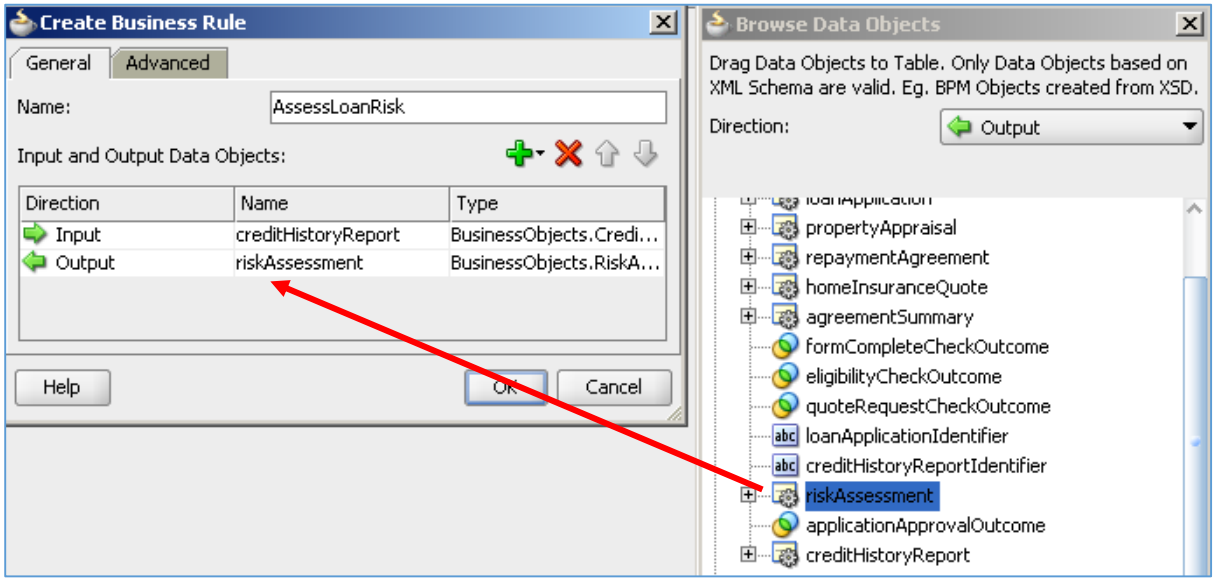
Click on the green plus sign button to add an Input data object. In the Browse Data Object popup next to the Create Business Rule window, list of process variables is displayed for selection.

Drag **creditHistoryReport** and drop it on the Create Business Rule window, in the area of input and output variable.

Do the same to add **riskAssessment** as an Output data object.

Then, click on Advanced tab and change the package name to LoanAssessment.

Then, click on OK button.

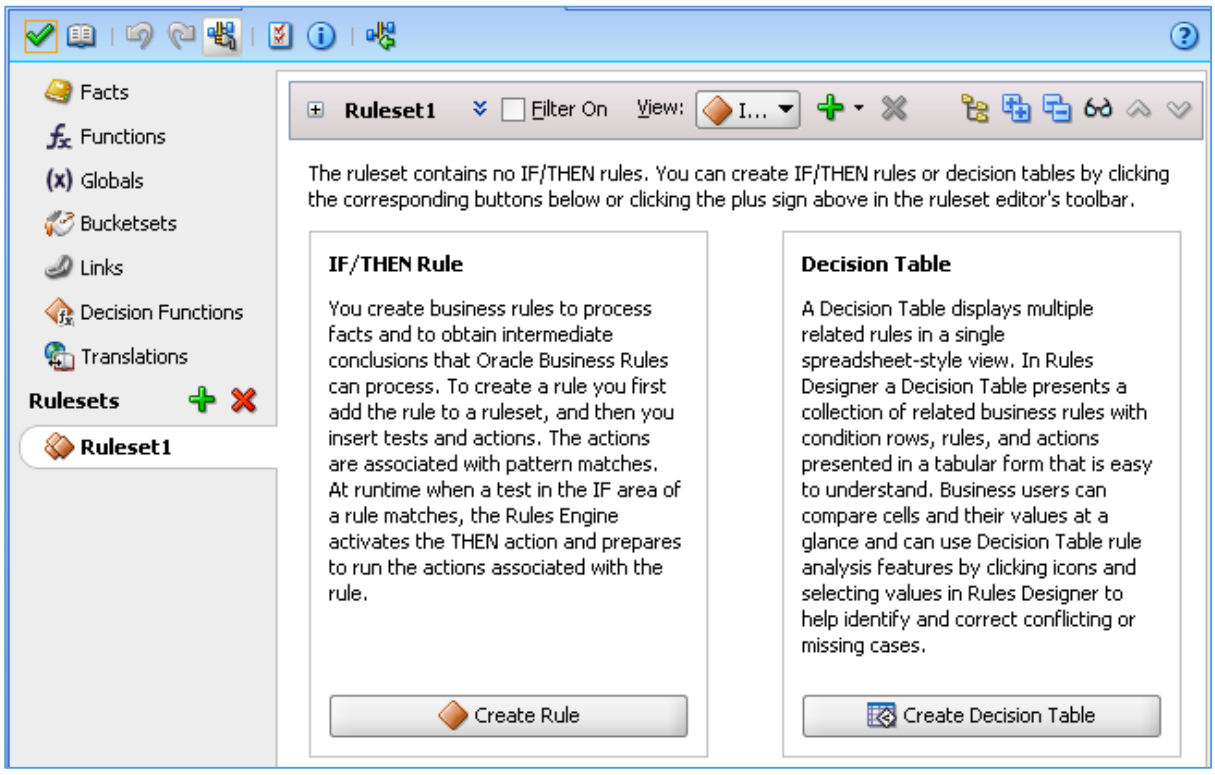


As a result, a new business rule is created and added to Business Catalog under Business Catalog > Rules > LoanAssessment (BPM Project Navigator).

Expand the rule in Business Catalog and double click to open it (or right click on the AssessLoanRisk activity and select Open Business Rule).

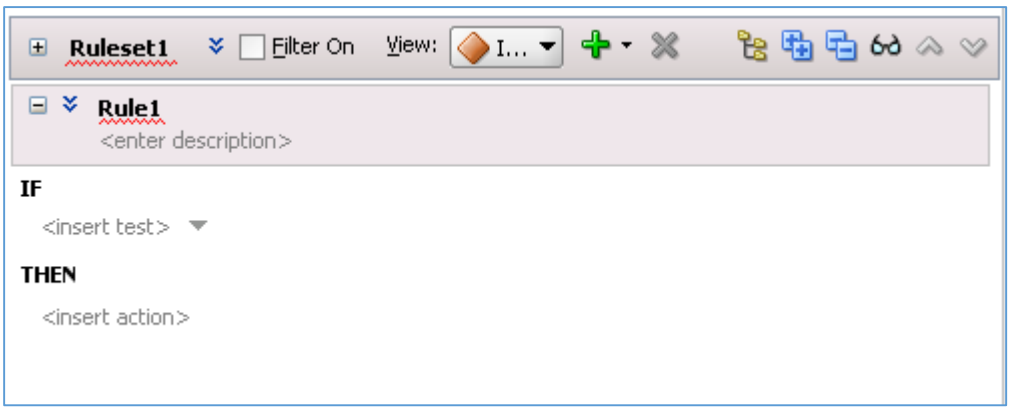
The Business Rule facility is shown below. We can now add more implementation details. The business rule implementation created here will be executed by a business rule engine of BPM runtime. There are two types of rules supported: IF/THEN and Decision Table.

We'll use IF/THEN rule in this case. Note that there is a default **Ruleset1** created. We'll add rules to this rule set.



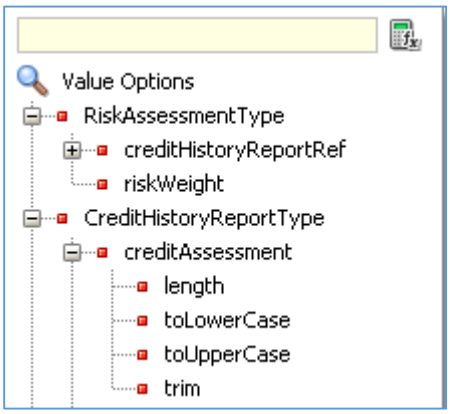
Click on Create Rule button in the IF/THEN rule area.

The Rule1 is shown below with empty IF and THEN clause. We need to implement these two clauses for Rule1.

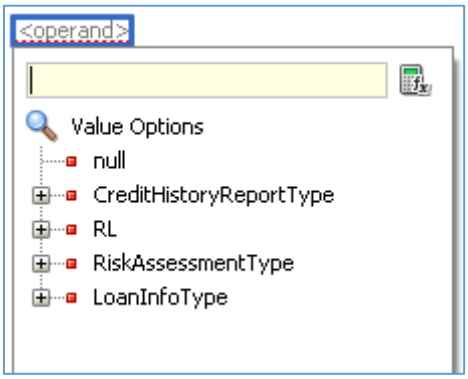


Click on the <insert test> in IF clause. It then shows a template: <operand> == <operand>

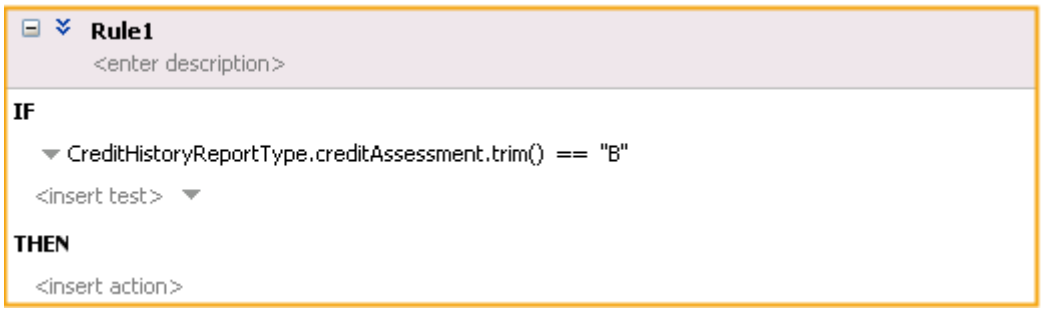
Click on the first operand. It shows a list of available variables to be selected from. Select **CreditHistoryReportType** > **creditAssessment** > **trim**.



Then, click on the second operand. Enter “B” in the box (including two quote characters) and press Enter.



The IF clause should look like below.



Now, click on <insert action> in the THEN clause. Select **assert new** in the drop-down box.

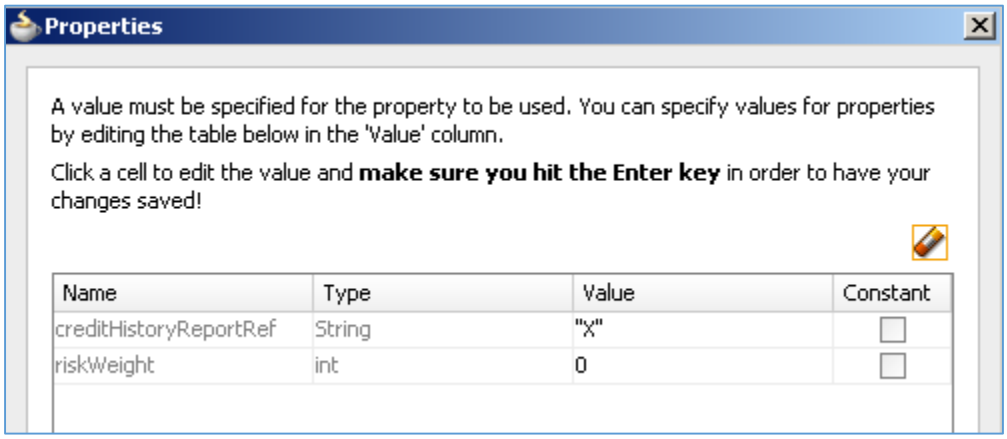
The THEN clause changes to assert new <target>.

Click on <target>. Select **RiskAssessmentType** in the drop-down box.

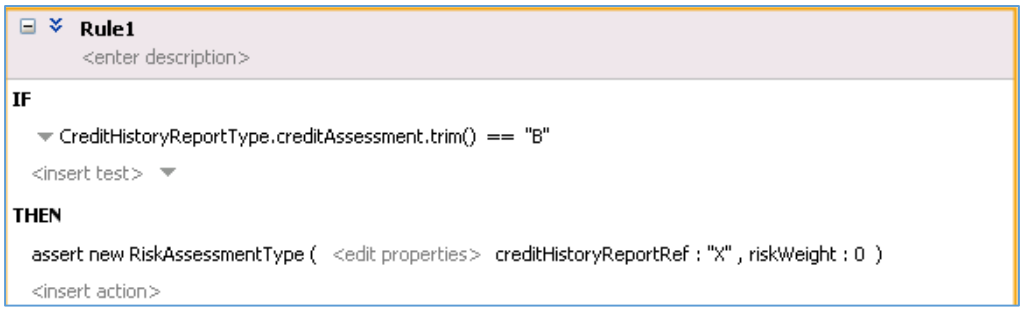
The THEN clause shows assert new RiskAssessmentType (<edit properties>).

Click on <edit properties>. Enter “X” for creditHistoryReportRef (this is unused value at the moment) and press Enter, and 0 for riskWeight and press Enter, as shown below.

Then click on OK button.



The Rule1 has been implemented as below. It is to translate B grade to rating value 0.

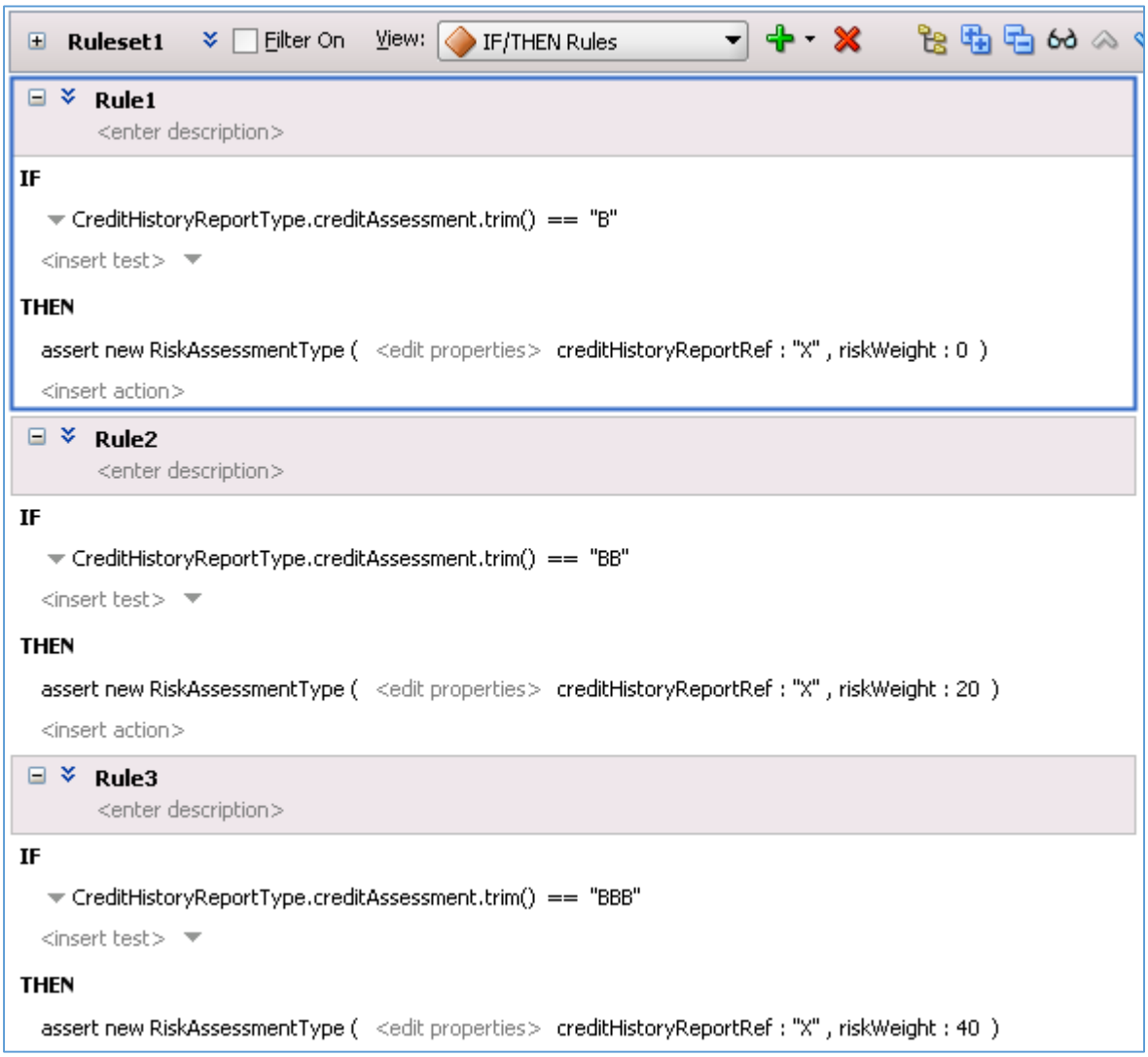


Follow the same procedures above, you can implement additional rules to translate other credit grades to numeric crating values.

Click on the drop-down button next to the green plus sign on the header of **Ruleset1**, then select Create Rule. Then follow the same procedures above.

The finished Ruleset1 should contain 6 rules, from Rule1 to Rule6.

Rule	Grade	Numeric value
Rule1	B	0
Rule2	BB	20
Rule3	BBB	40
Rule4	A	60
Rule5	AA	80
Rule6	AAA	100



7.8 Implement Timer

Our process requirement is the process will terminate if the process user haven't finished completing the application form within 5 days. In another case, the process also terminates if the Loan Officer waits more than 14 days for the customer feedback to complete the repayment agreement verification activity.

We have modelled this feature with Timer activity in section 6. We'll set implementation details in accordance with the above requirements.

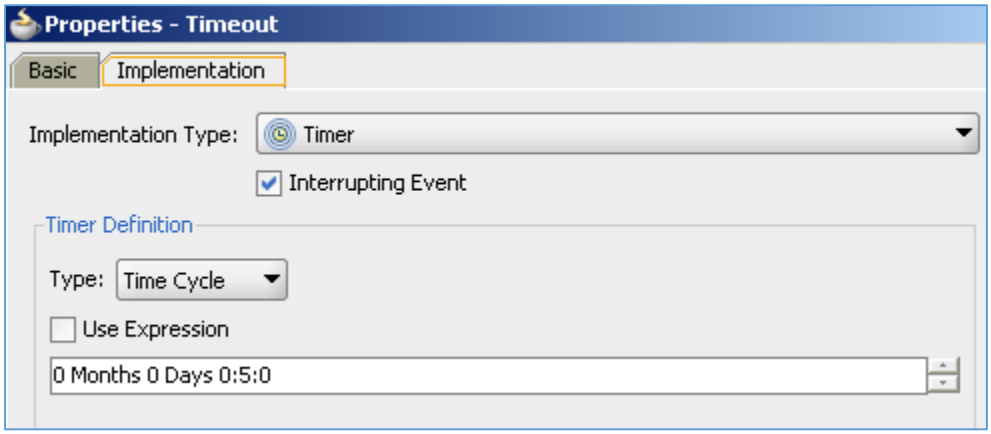
Timer for Enter Application Form

Double click on the Timer event attached to the edge of Enter Application Form activity.

In the Properties window, set the following values.

- Interrupting Event: checked
- Type: Time Cycle
- Time value: 5 days (for testing purpose, we set it to 5 minutes in the picture below but it should be set to 5 days in reality).

Then, click on OK button.



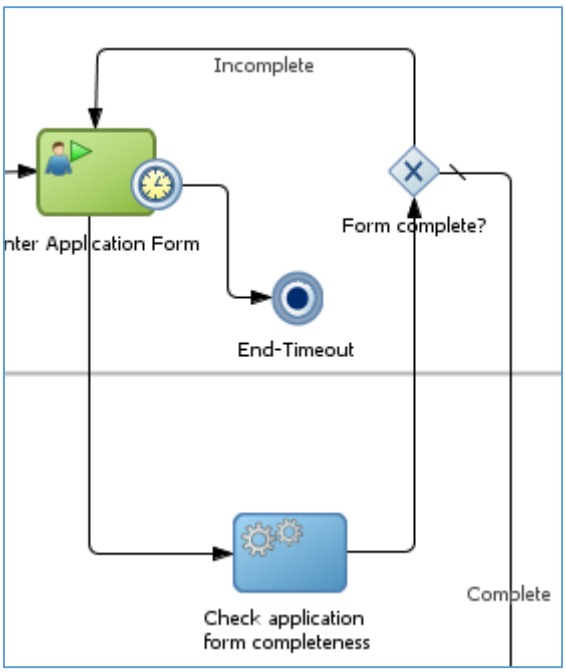
That finishes the implementation for timer event. It means when the timer reaches the end of its time cycle, it will interrupt the current activity and changes the process flow to the interrupting flow.

Apply the same procedures for timer event of Verify Repayment Agreement activity.

7.9 Implement Gateways

We also need to implement the conditional gateway where the flow is driven by the input data object value. The process model has four conditional gateways: assess form completeness, assess application eligibility, consider if insurance quote requested, and consider final decision.

Let's do it first for gateway after checking form completeness.



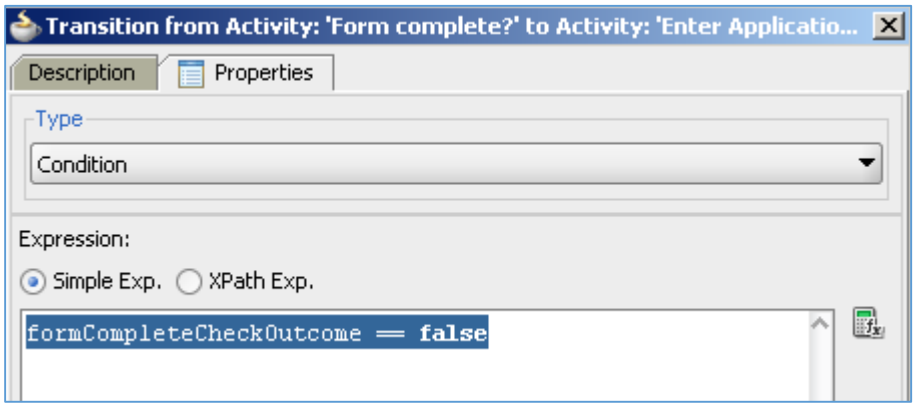
If you double click on two outflows from the gateway to view the Properties (Properties tab), you will see one is marked as default (unconditional) and one is conditional. We need to set a condition for the

conditional flow. When the condition is true, the process will follow the conditional flow; otherwise, it follows the unconditional flow.

Now, double click on the unconditional flow (the incomplete case), click on Properties tab in the popup. Enter this expression: formCompleteCheckOutcome == false.

The above expression is simply based on formCompleteCheckOutcome process variable. The value of this variable has been set in the “Check application form completeness” activity. Then, click on OK button.

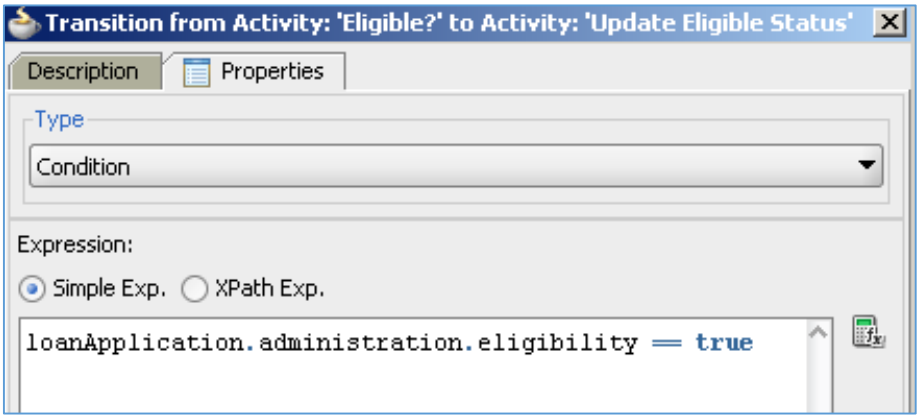
You can open the Expression Builder by clicking on the button on the right to enter more complex expression if need to.



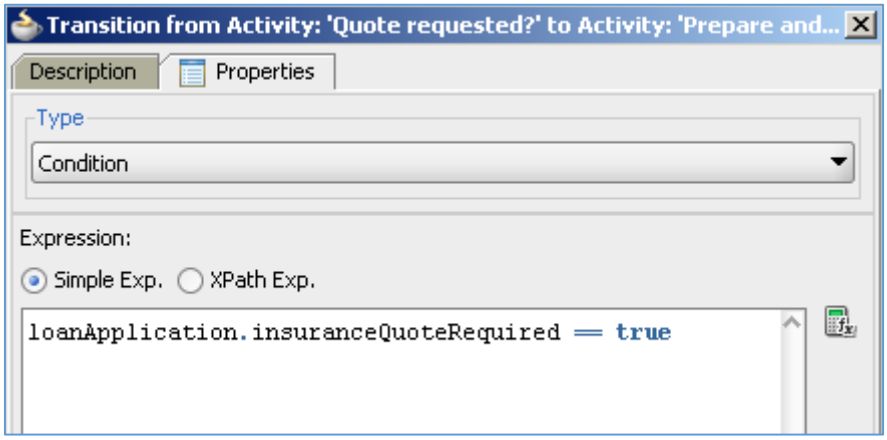
You have done implementing the flows at “Form Complete?” gateway.

Please do similarly for other gateways, following the screen shots below.

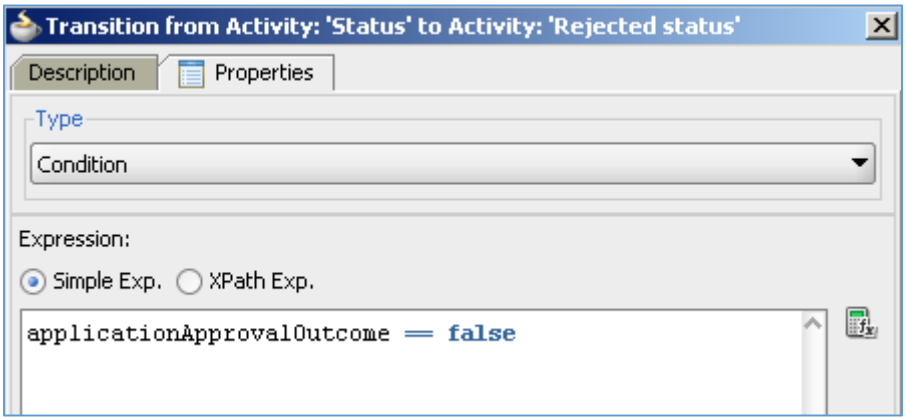
“Application Eligible” gateway – Eligible flow.



“Insurance Quote requested” gateway – Requested flow.



"Final Decision Status" gateway – Rejected flow.



7.10 Implement Emails

The process requirement is to notify applicant via emails about their application status.

We have added email task in the process model in section 6. We also have set up email notification function for the UMS service on the server to send email.

Now, we'll implement email notification for every email task, including recipient and mail content.

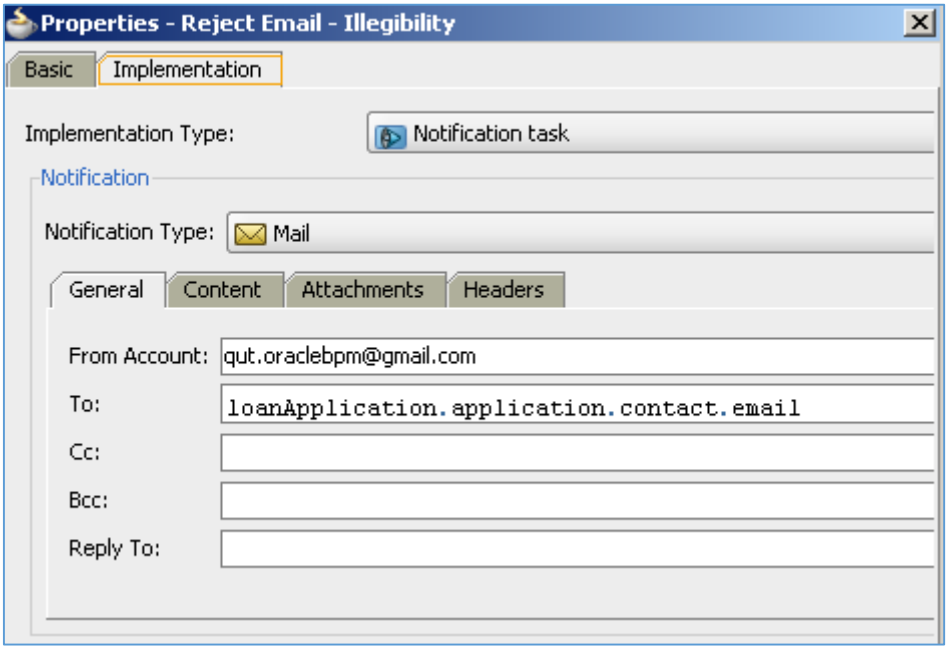
There are totally four email tasks in the process:

- Reject email due to illegibility
- Cancel Application email due to timeout at the verifying repayment agreement step
- Approval email after the "Take Final Decision" step.
- Rejection email after the "Take Final Decision" step.

We'll demonstrate procedures for "Reject email due to illegibility" case. You can then apply the same procedures for other cases.

7.10.1 Reject Email – Illegibility

Open BPMN diagram, double click on the "Reject Email – Illegibility" task. Click on Implementation tab.



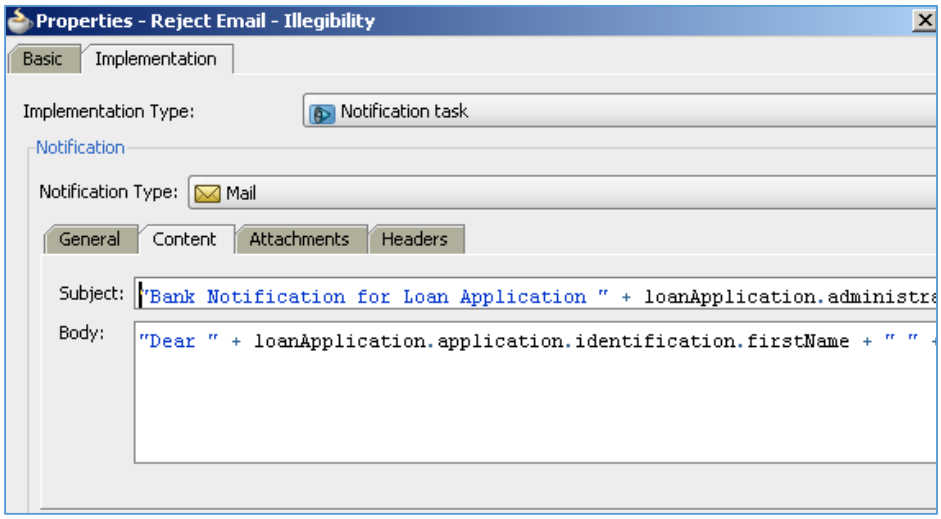
Enter the following information in the General tab.

- From Account: qut.oraclebpm@gmail.com
- To: click on the Expression Builder icon on the right. In the Expression Builder, enter the following information:
  - Mode: Simple Exp.
  - Expression: loanApplication.application.contact.email

  Click OK to close the Expression Builder.

  Note: qut.oraclebpm@gmail.com is a free Gmail account set up for this tutorial (remember we have set up Gmail server as a relay server to send email in section 7.1.2). You can log in this account on Gmail to check emails later (password: ironcar108).

In the Properties window, click on Content tab.



Open Expression Builder for Subject field, enter the following information:

- Mode: Simple Exp.
- Expression: (including quote characters).

```
"Bank Notification for Loan Application " +
loanApplication.administration.applicationIdentifer + " - Eligibility Notice"
```

Then, click OK.

Open Expression Builder for Body field, enter the following information:

- Mode: Simple Exp.
- Expression: (including quote characters).

```
"Dear " + loanApplication.application.identification.firstName + " " +
loanApplication.application.identification.lastName + ",<br><br>STATUS OF YOUR LOAN
APPLICATION<br><br>Application ID: " +
loanApplication.administration.applicationIdentifer + ".<br><br>Thank you for your interest
in our mortgage products.<br><br>We want to inform you that your application to our loan
product has expired <br> since our last contact with you. This is in accordance with our
policy of 5-day maximum pending period.<br><br>Your application submission date: " +
loanApplication.administration.submissionDateTime + "<br>Your application revision date:
" + loanApplication.administration.revisionDateTime + "<br><br>Kindly lodge a new
application once your information is available and/or write to us at
mortgage@bestbank.com for any clarification
required<br><br>Sincerely,<br><br>Mortgage Services"
```

Then, click OK.

Click OK in the Properties window. That finishes notification implementation for the “Reject Email – Illegibility” task.

Now, you should do the same for other notification tasks with the guided information below.

7.10.2 Cancel Application Email

From Account	Plain Text	qut.oraclebpm@gmail.com
To	Simple Exp.	loanApplication.application.contact.email
Subject	Simple Exp.	"Bank Notification for Loan Application " + loanApplication.administration.applicationIdentifer + " - Eligibility Notice"
Body	Simple Exp.	"Dear " + loanApplication.application.identification.firstName + " " + loanApplication.application.identification.lastName + ", STATUS OF YOUR LOAN APPLICATION Application ID: " + loanApplication.administration.applicationIdentifer + " Thank you for your interest in our mortgage products. Please be informed that your application above has expired since our last contact with you. This is in accordance with our policy of 5-day pending period. Your application submission date: " + loanApplication.administration.submissionDateTime + " Your application revision date: " + loanApplication.administration.revisionDateTime + " Please lodge a new application once you have enough information, or write to us at mortgage@bestbank.com if you need any clarification or assistance Sincerely, Mortgage Services"

7.10.3 Approval Email

From Account	Plain Text	qut.oraclebpm@gmail.com
To	Simple Exp.	loanApplication.application.contact.email
Subject	Simple Exp.	"Bank Notification for Loan Application " + loanApplication.administration.applicationIdentifer + " - Approval Notice"
Body	Simple Exp.	"Dear " + loanApplication.application.identification.firstName + " " + loanApplication.application.identification.lastName + ", STATUS OF YOUR LOAN APPLICATION Application ID: " + loanApplication.administration.applicationIdentifer + " Thank you for your interest in our mortgage products. We are glad to inform you that your application has been approved in accordance with our policy and based on the information you have submitted to us. We look forward to working with you in the next step. If we don't receive your response within 30 business days since this notification, your existing application will be filed and any new application must be lodged again afterwards. Sincerely, Mortgage Services"

7.10.4 Rejection Email

From Account	Plain Text	qut.oraclebpm@gmail.com
To	Simple Exp.	loanApplication.application.contact.email
Subject	Simple Exp.	"Bank Notification for Loan Application " + loanApplication.administration.applicationIdentifer + " - Rejection Notice"
Body	Simple Exp.	"Dear " + loanApplication.application.identification.firstName + " " + loanApplication.application.identification.lastName + ", STATUS OF YOUR LOAN APPLICATION Application ID: " + loanApplication.administration.applicationIdentifer + " Thank you for your interest in our mortgage products. We are unable to approve your application in our final review according to our policy and based on the information you have submitted to us. Should you need any consultation, please contact us at mortgage@bestbank.com. Sincerely, Mortgage Services"

8 Deploy the Process

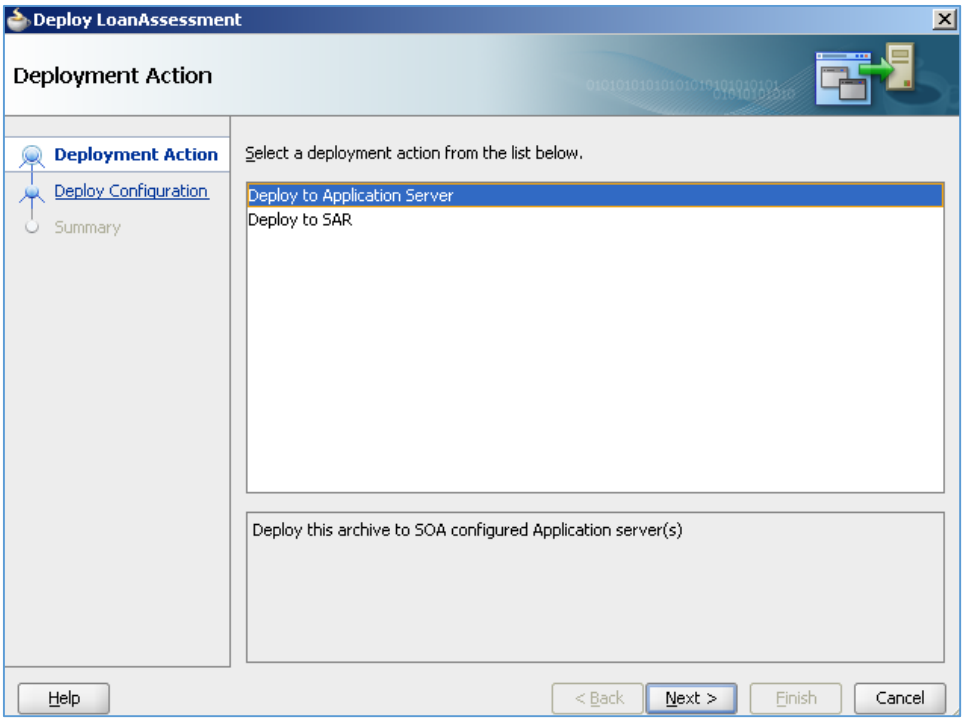
When you are up to this point, you have accomplished a lot of implementation work. Great job!

It is time to deploy your implemented process to SOA/BPM server. We can use the server connection set up at the beginning of section 6.

In Application Navigator, right click on the BPM project (Loan Assessment). In the pop-up menu, select Deploy > LoanAssessment…

A wizard will display and guide you through deployment process.

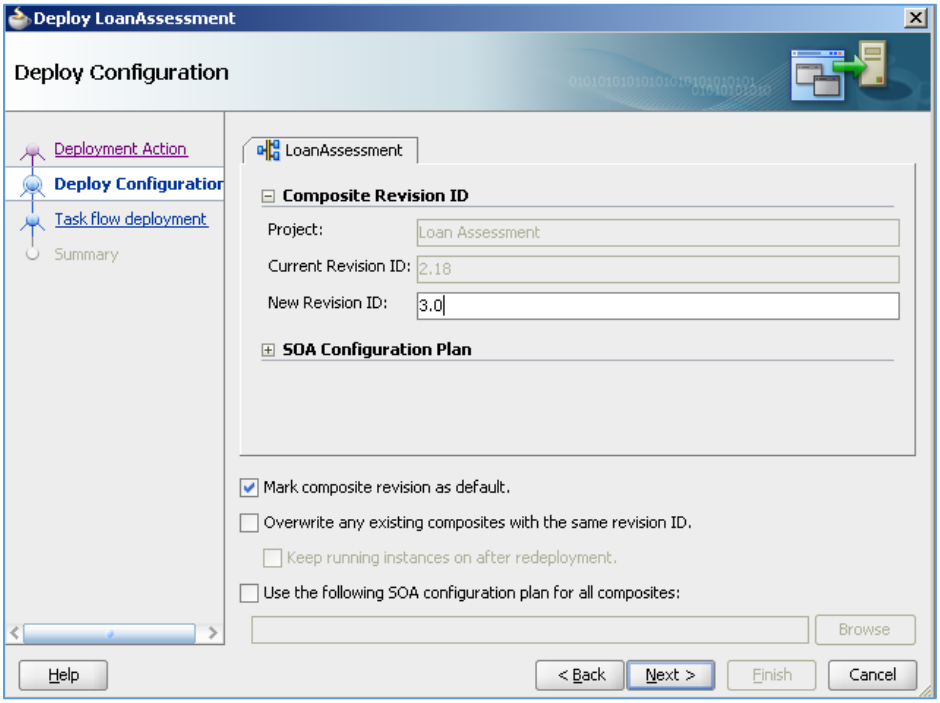
Select to Application Server in the first screen (Deployment Action) and click Next.



In Deployment Configuration screen:

- New Revision ID: enter a version number
- Mark composite revision as default: checked
- Overwrite any existing composites with the same revision ID: checked
- Keep running instances on after redeployment: checked

Click Next.

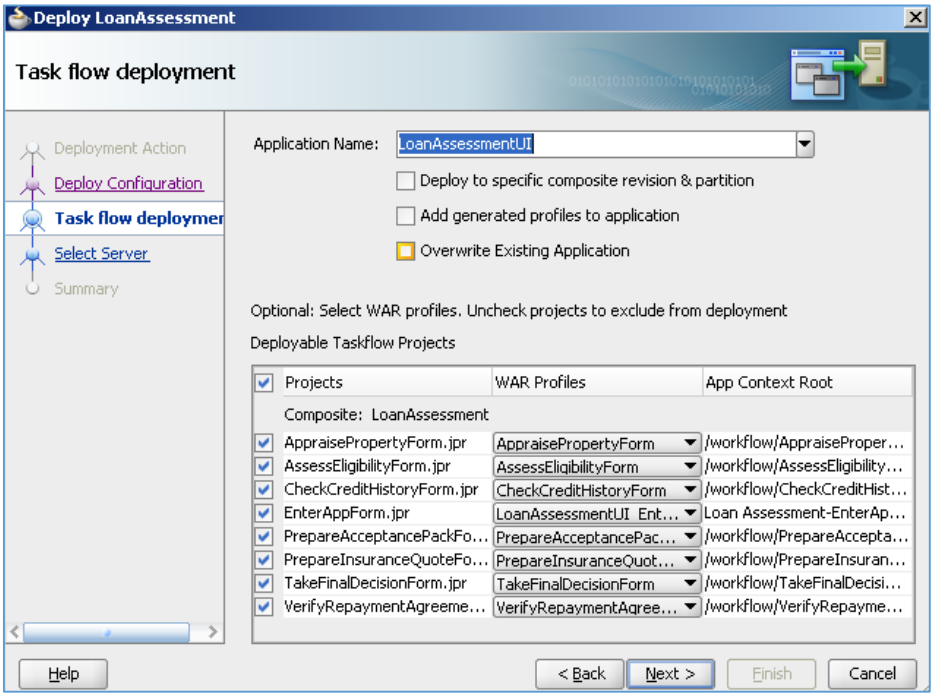


In Task Flow deployment screen:

- Application Name: enter a name for UI package, e.g. LoanAssessmentUI
- Overwrite Existing Application: checked
- Select all UI projects in the list below.

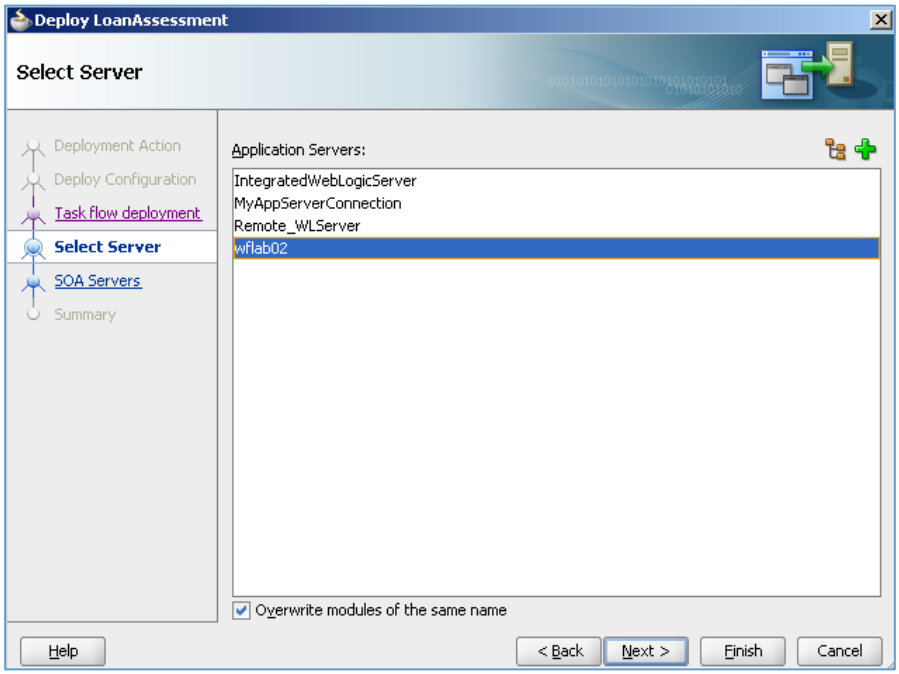
This step means all human task web form projects will be packaged in one package only called LoanAssessmentUI (.ear package) and deployed to the server.

Click Next.

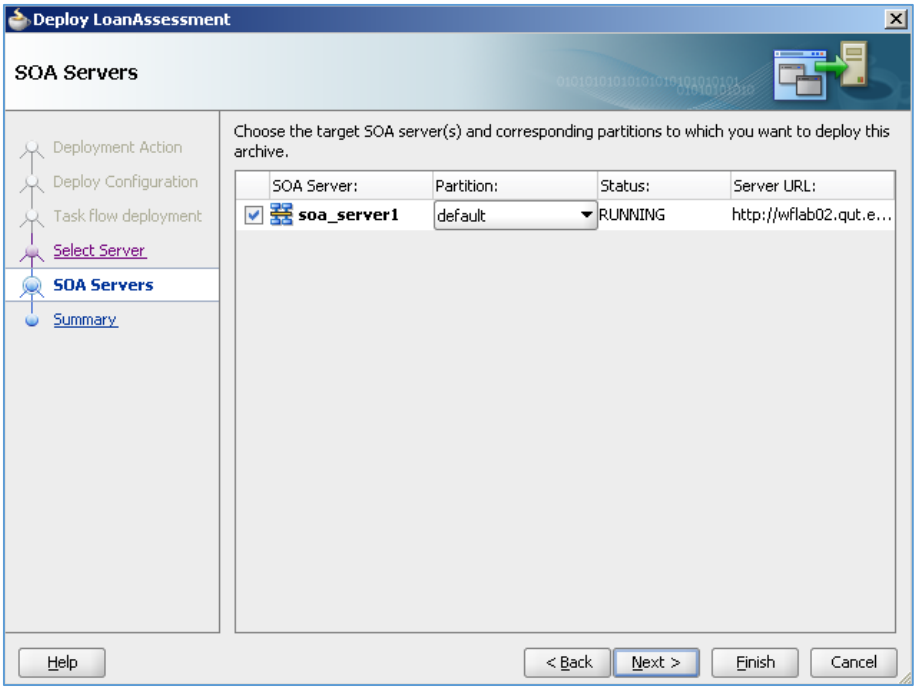


In Select Server screen, select the server name configured in section 7.1.1

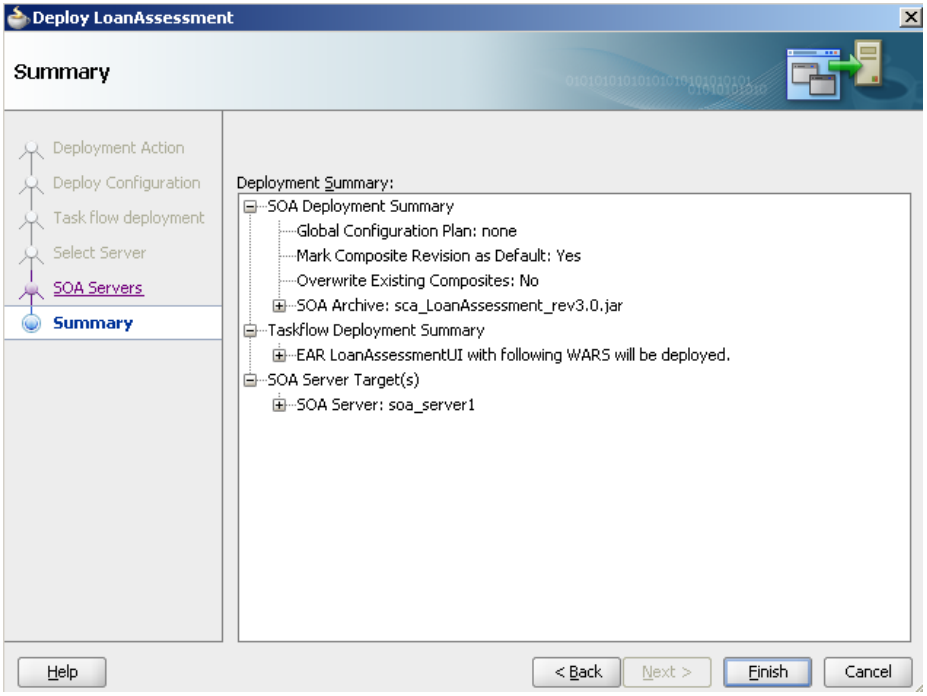
Click Next.



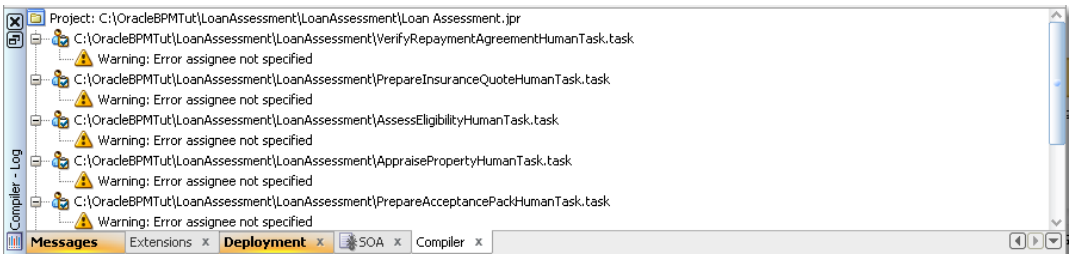
In SOA Servers, wait until all servers displayed. Select the running server you want to deploy to and click Next. Normally you only need to deploy to one SOA server, in case there might be other servers running (BAM, OSB servers). Then click Next.

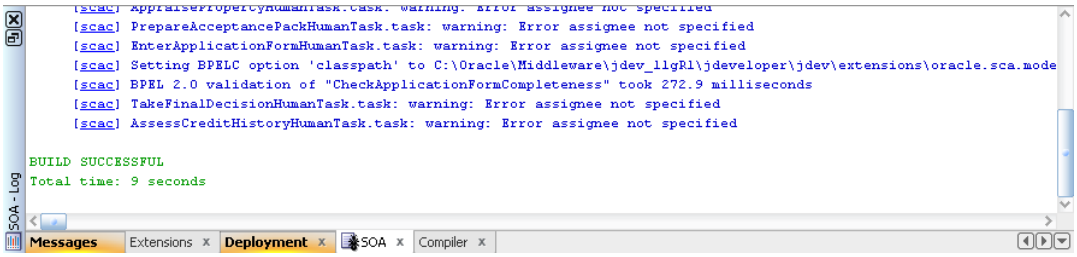


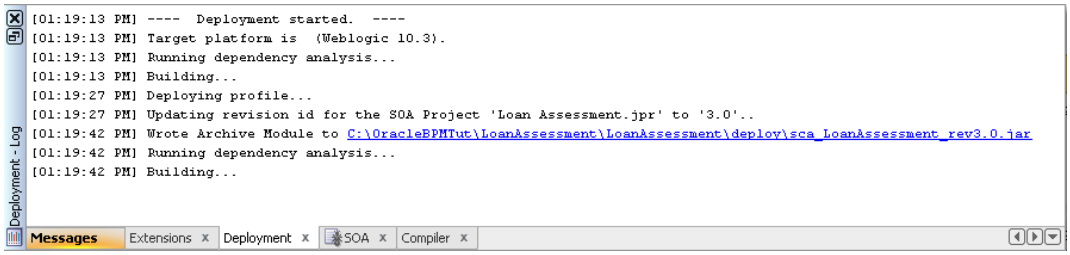
In the Summary, review all information again and click Finish to start deploying to the server.

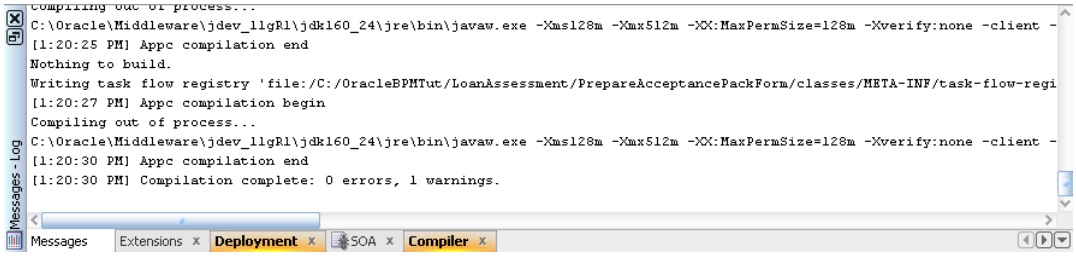


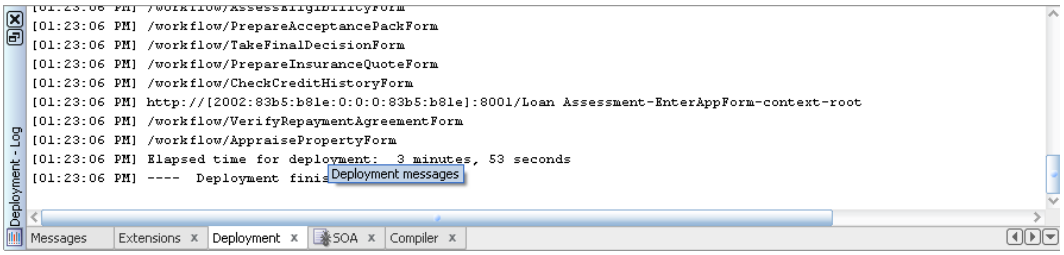
The deployment will take a while. You can watch the deployment process by clicking on the Log button at the bottom of JDeveloper screen. It shows several tabs: Messages, Deployment, SOA, Compiler. The order of deployment steps is: Compiler, SOA and then Deployment. The Messages tab shows detailed messages of these steps.





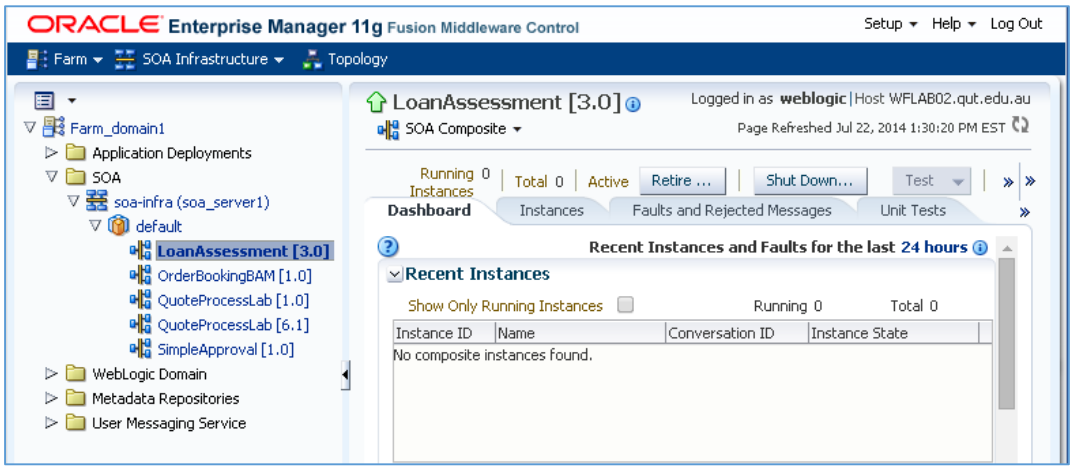


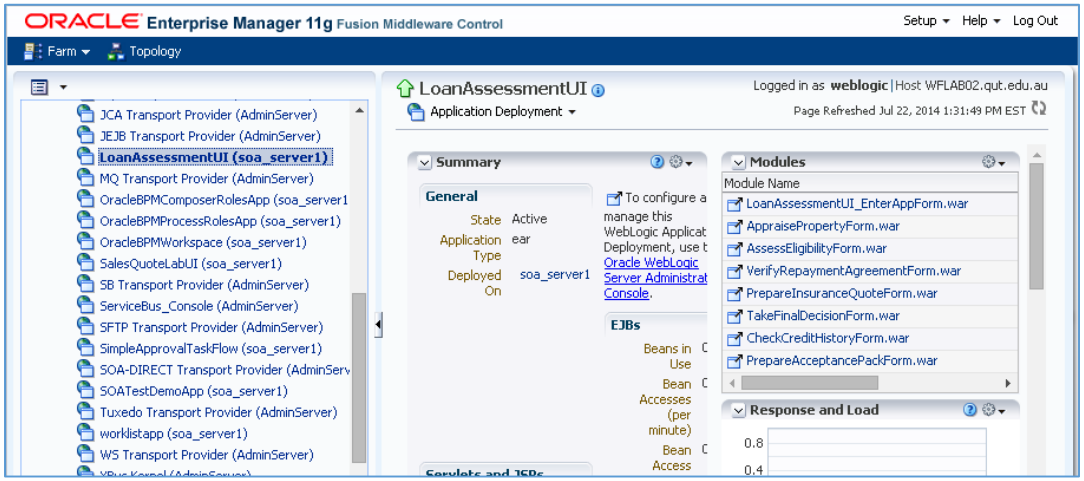




You should watch the Deployment tab until it shows “Deployment finished” without error messages.

On the server, you can login Enterprise Manager as Administrator and view the new deployment as shown in the figure below. There are two deployments: one for the process deployment (LoanAssessment_versionnumber) and one for the UI deployment (LoanAssessmentUI).





9 Run the process

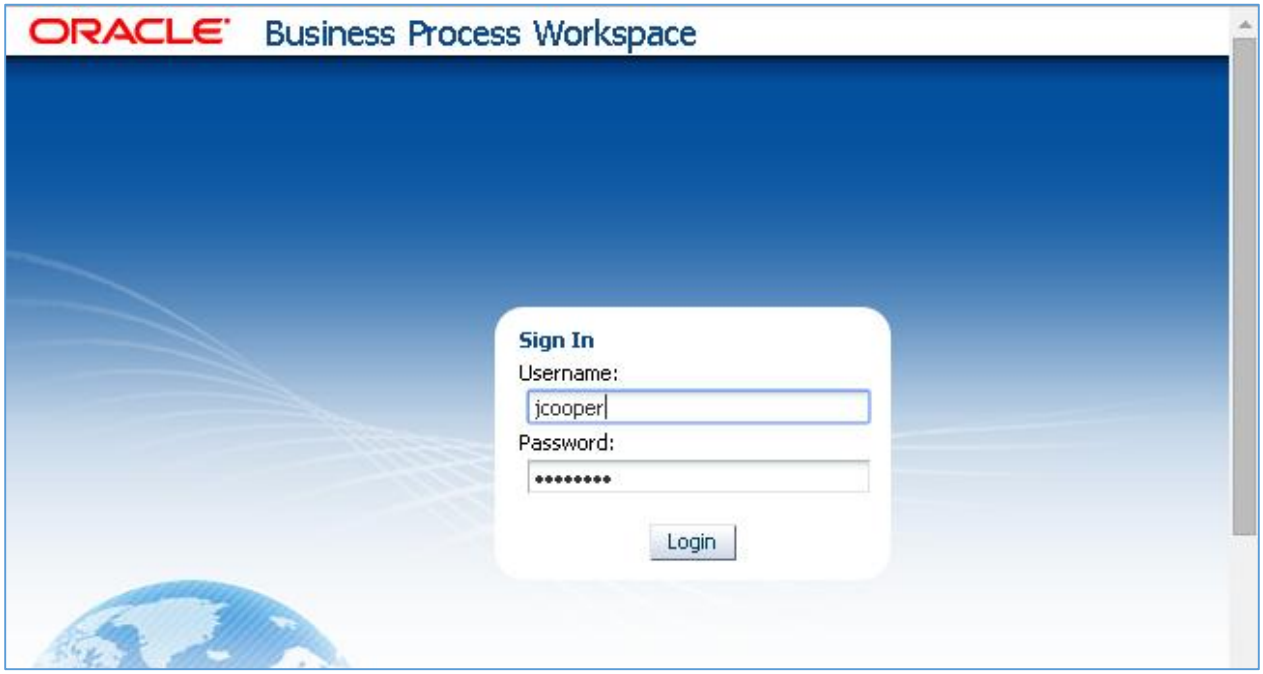
Once you have deployed the process successfully to the server, you can now run the process in Oracle BPM Workspace. This section will demonstrate the main scenario of the Loan Assessment process run.

Remember that you have implemented a process organization in section 7.2, as listed below again. Now, you should log in BPM Workspace with these example users to run the process step by step.

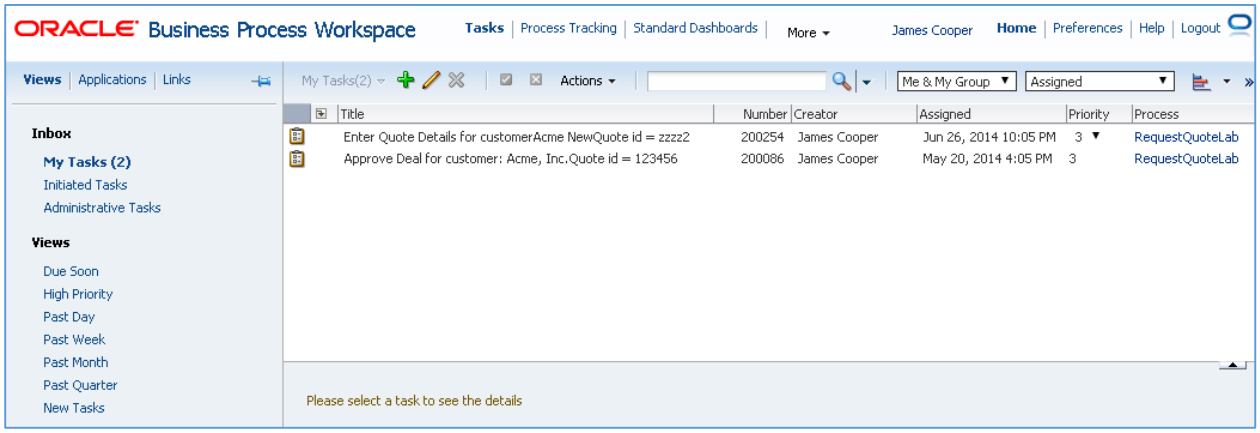
Role	User
Applicant	jcooper
Financial Officer	cdickens
Loan Officer	mmitch, mtwain
Property Appraiser	rsteven
Process Owner	wfaulk
Insurance Sales Representative	fkafka

9.1 Process user

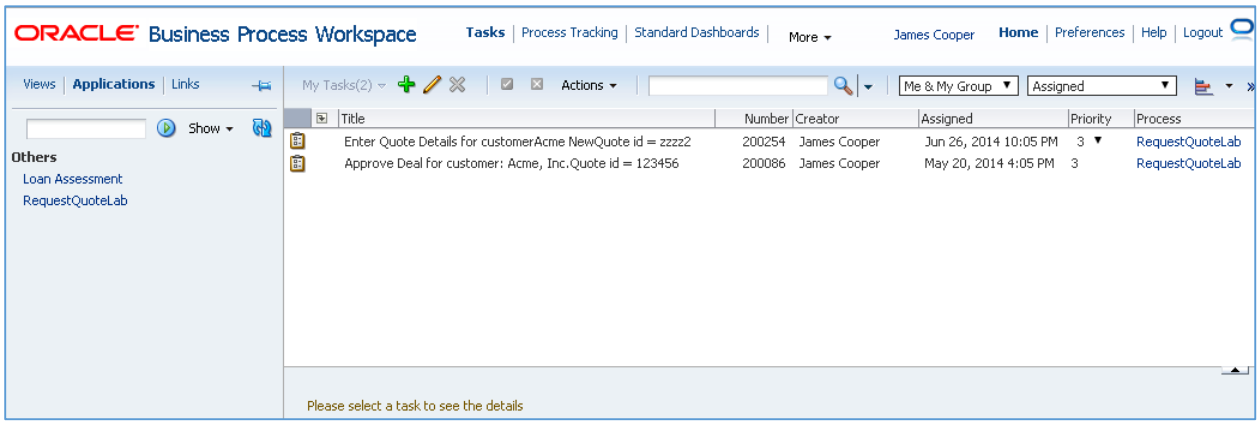
Open a web browser and type in a link to BPM Workspace according to your installation in section 3, for example, http://localhost:8001/bpm/workspace. Log into BPM Workspace as **jcooper** as an Applicant who is able to initiate a process instance.



The homepage of BPM workspace for jcooper is shown below.



Click on Application link on the left pane. It will show a list of process links available to be initiated.

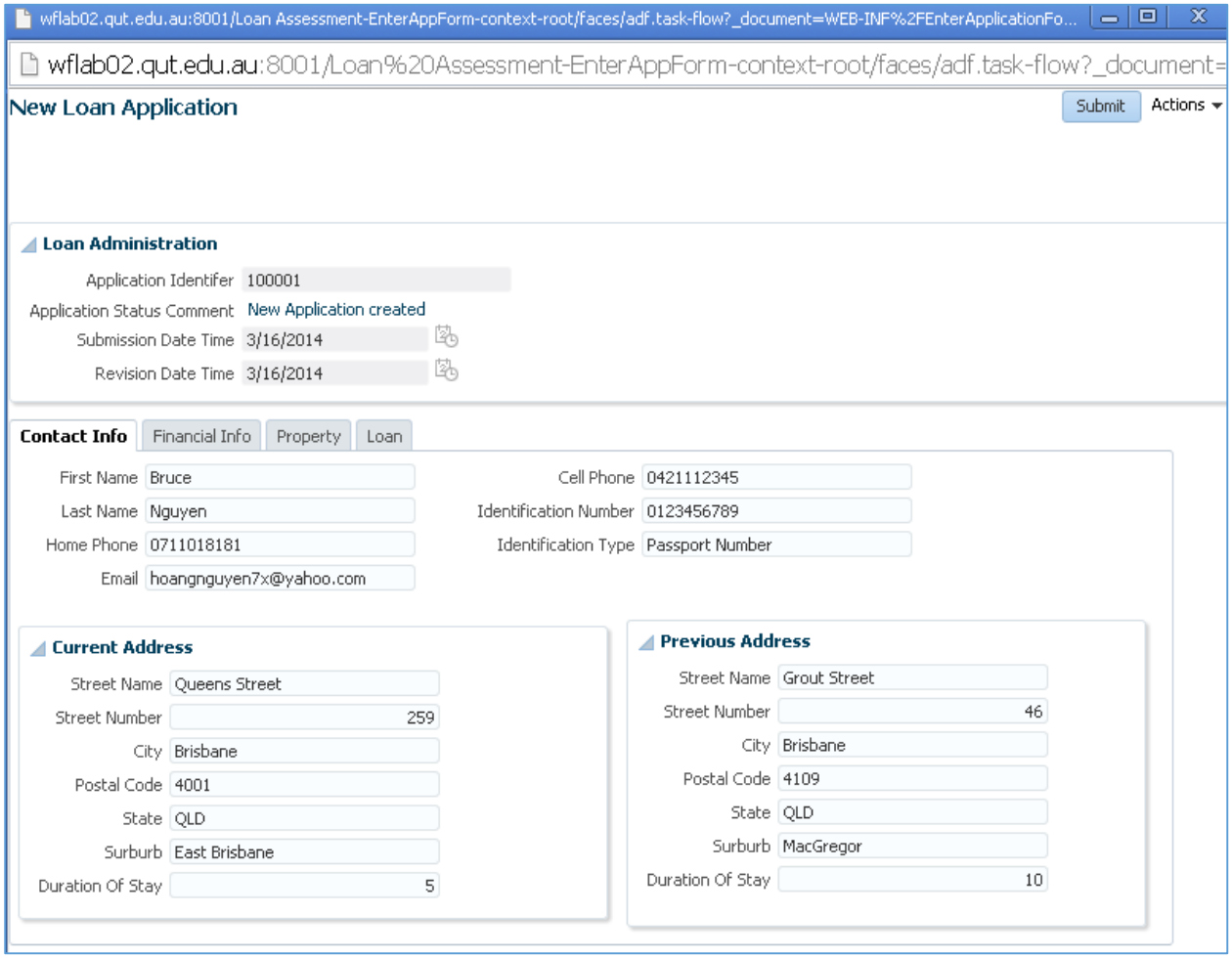


Click on the Loan Assessment link. It will open a new loan application form.

Fill information in the form. Note that the Home Insurance Quote Required option is located in the Loan tab. If you want to see the Home Insurance step done by the Insurance Sales Rep, you should check this option.

You need to provide at least First Name, Last Name and a valid email address. Note that after the form is submitted, if one of these three items are not valid (empty name or invalid email address), the step remains with **jcooper** user. The Application Status Comment will show the invalid information items.

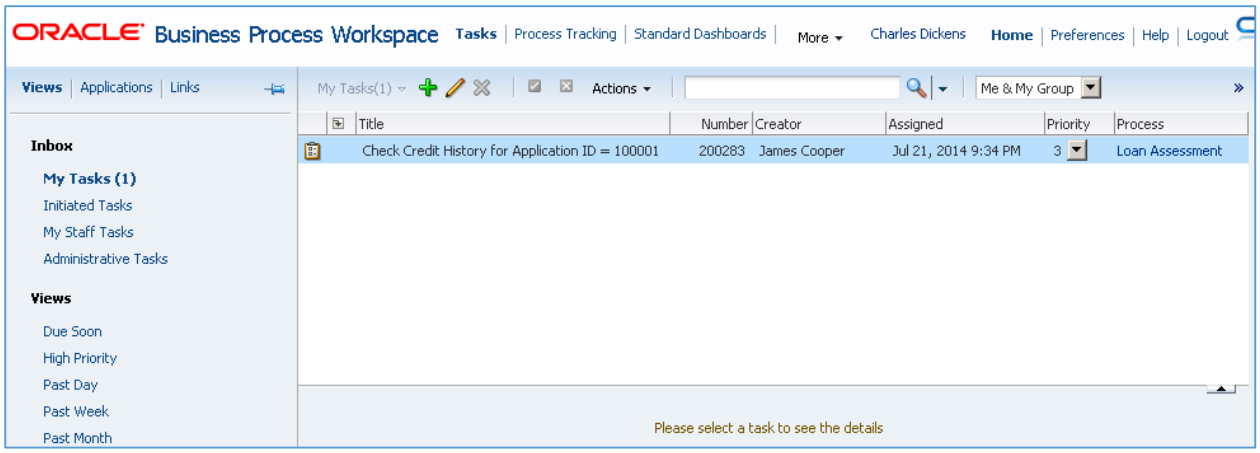
Click Submit when you are done with the form.



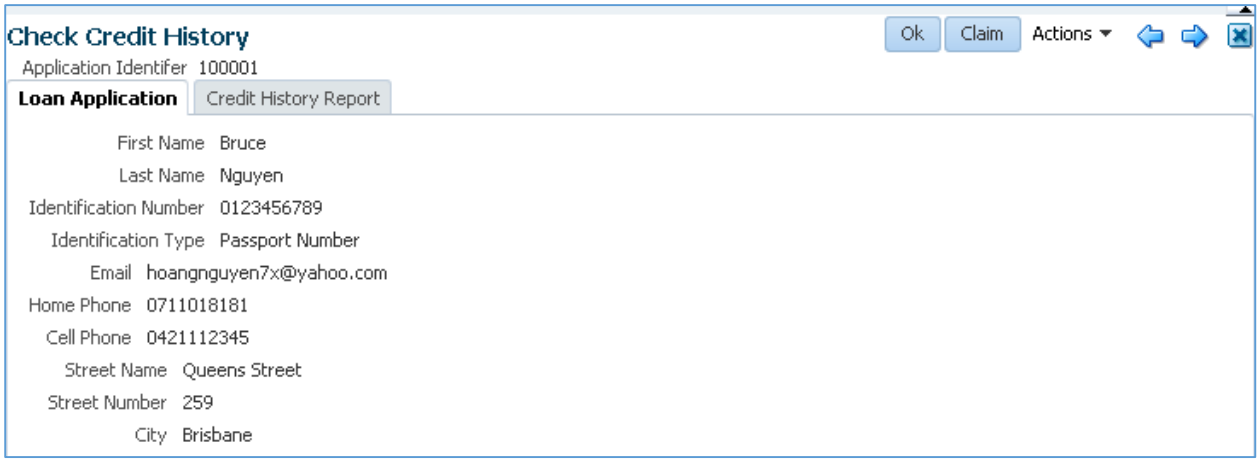
The process will proceed to the “Check application form completeness” activity. If these information items are checked as valid by the “Check application form completeness” activity, the process will forward to the next steps according to the process model.

Now, log out BPM Workspace and log in as **cdickens** (Financial Officer).

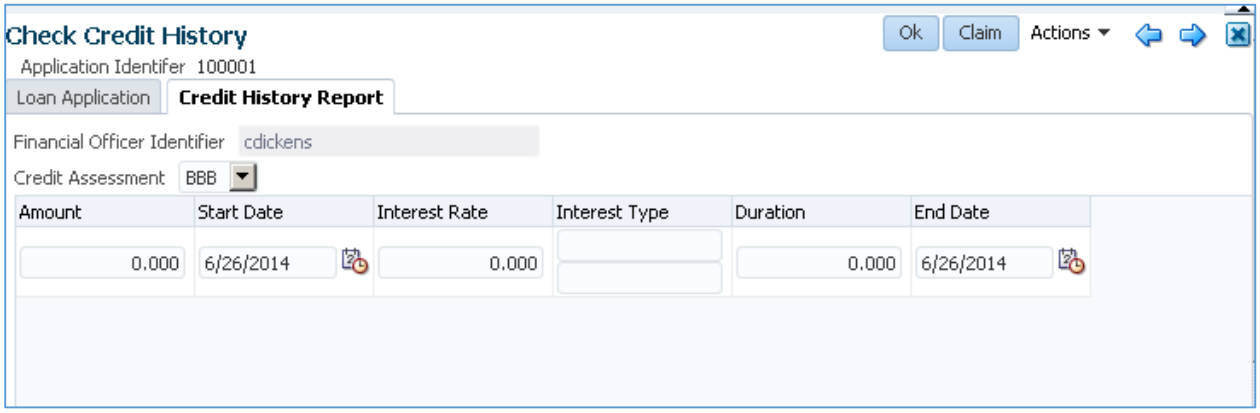
The homepage of cdickens shows there is one Check Credit History for Application ID = 100001



Click on the task title to open it in the pane below.

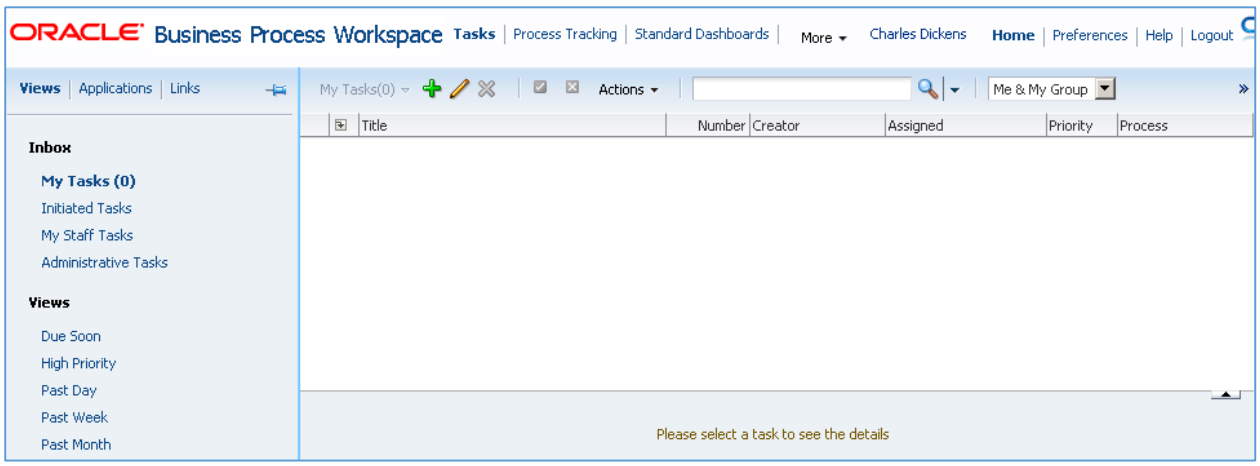


Click on Credit History Report tab. The Financial Officer Identifier has default value from the logged in username. Select a credit grade from the dropdown list. (Note: if you leave credit assessment empty and submit, there will be a run-time process error since the user validation is not complete in this tutorial, it is a point for enhancement).



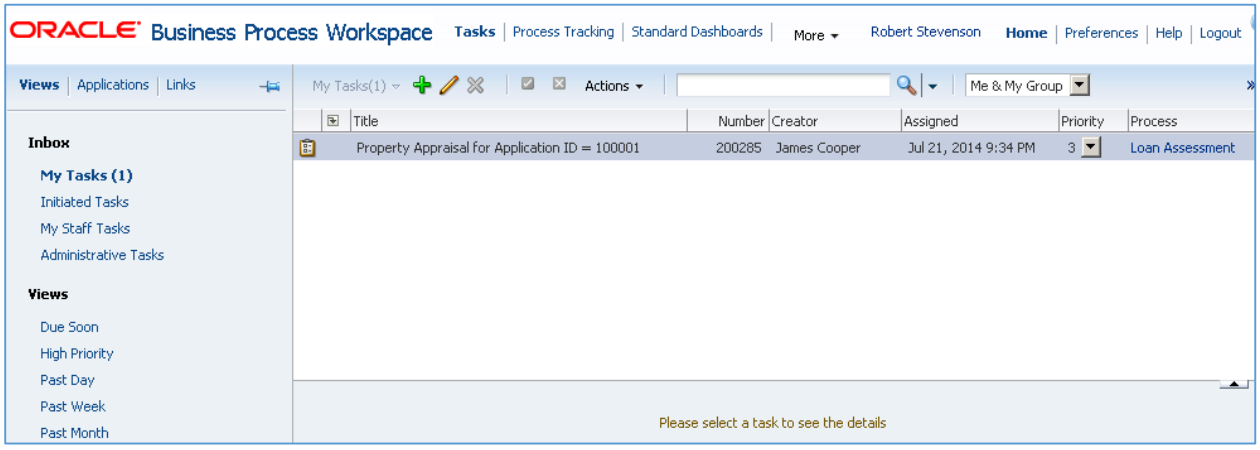
Then, click Ok.

The task will be submitted. If you click on My Tasks link again, the task will disappear in the list.



Now, log out BPM Workspace and log in as **rsteven** user (Property Appraiser).

The homepage below shows a new task on Loan Application ID = 100001 for rsteven to do.



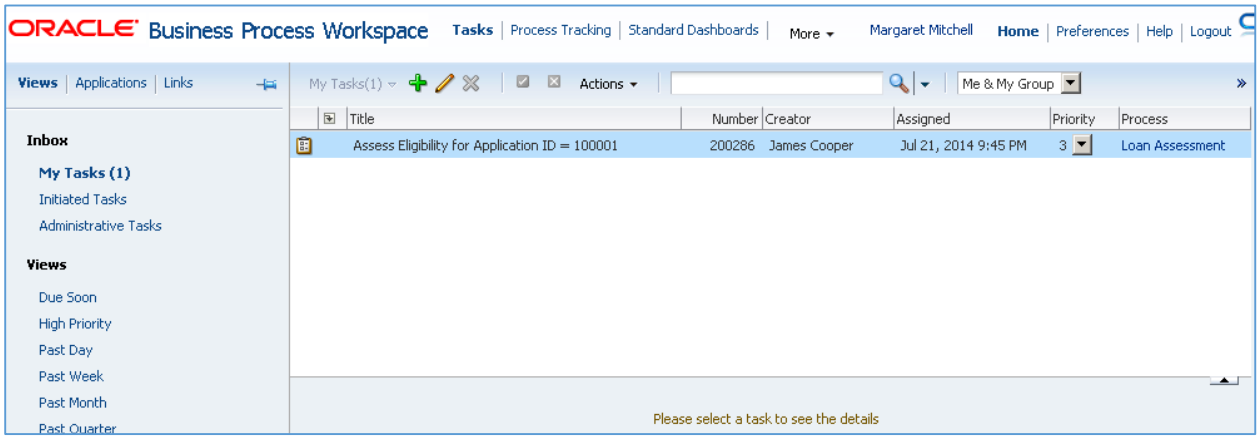
Click on the task to open it in the pane below. The Property Appraiser Identifier has been defaulted to the logged in username.

Enter Estimate Market Value and a comment. Then click Ok.

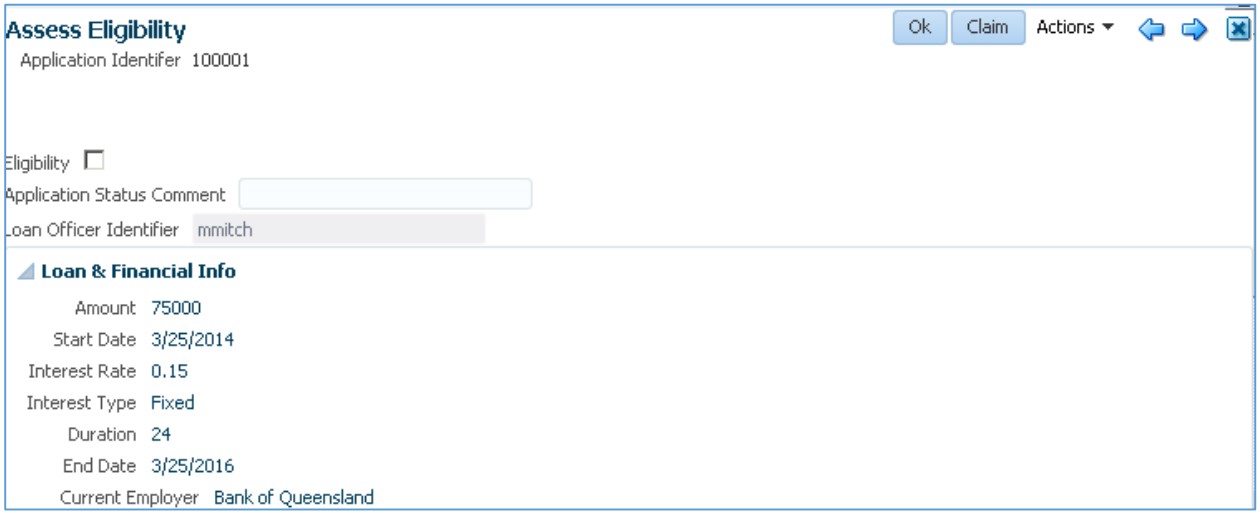
The task will be submitted to the server and disappear from the user task list.

In the next step, log out BPM Workspace and log in as **mmitch**.

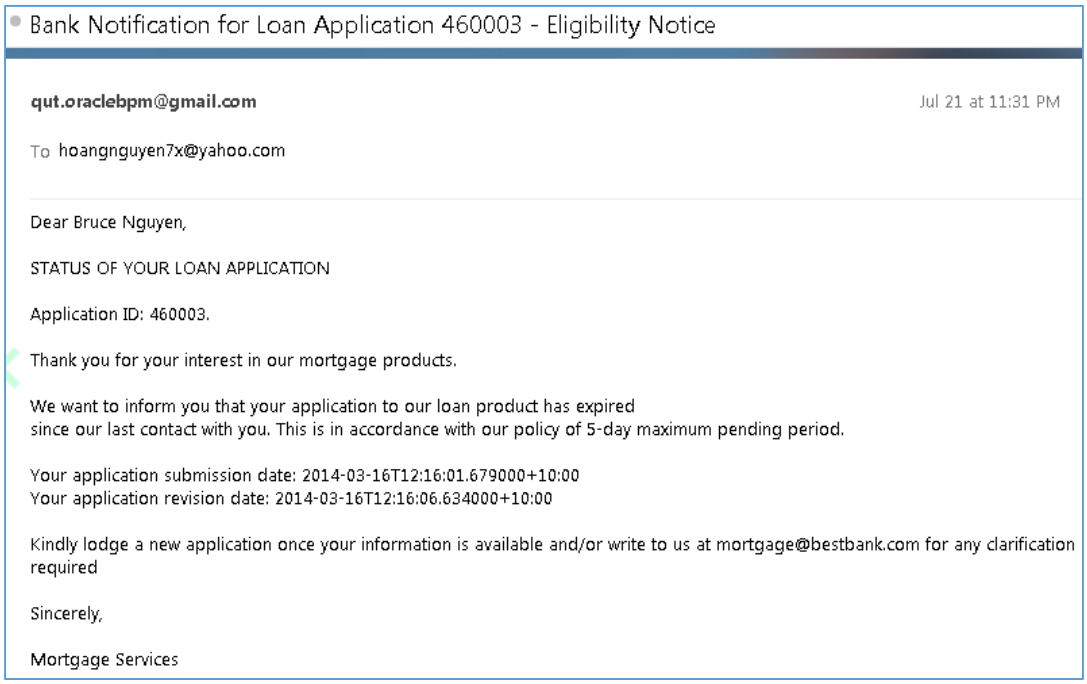
The homepage as shown below indicates there is an Assess Eligibility for Loan Application ID = 100001.



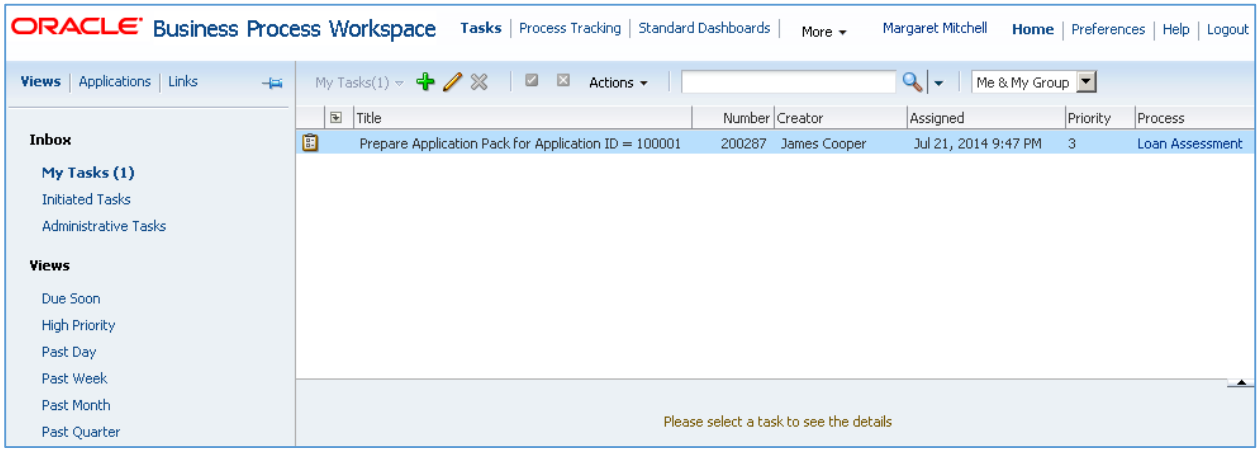
Click on the task to open it.



If the Eligibility box is not checked and the user clicks on Ok button, the process will stop at this point and an email will be sent to user in the following format.



In this main scenario, the user checks the Eligibility checkbox, add comments and click Ok. The task list will refresh and a following task appear in the list: “Prepare Application Pack”, still for the same **mmitch** user.



Click on the task to open it.

Note that the application status is now “ASSESSED” after its eligibility has been accepted in previous step.

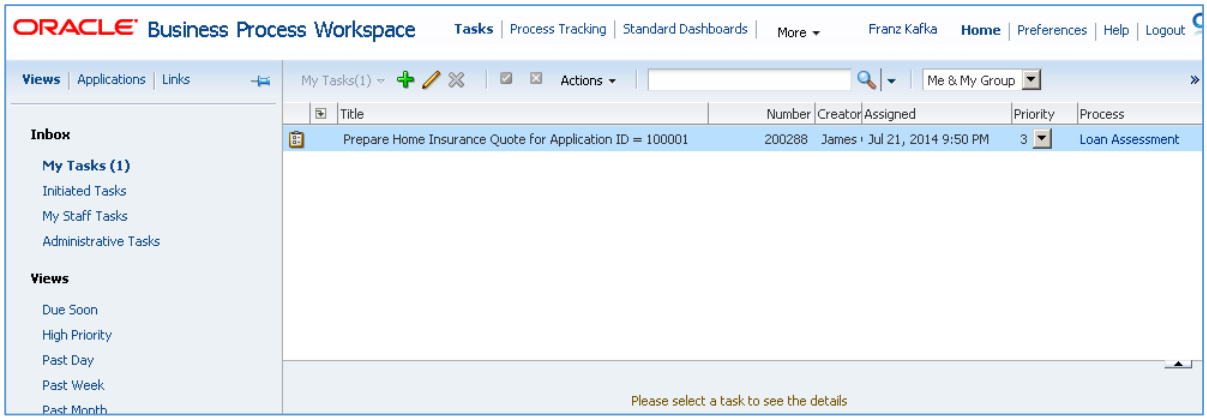
Enter Monthly Repayment Amount and Number of Repayments.

Now, you can click on the Generate PDF to open a repayment schedule in a pop-op window.

Click OK in the task form to complete it.

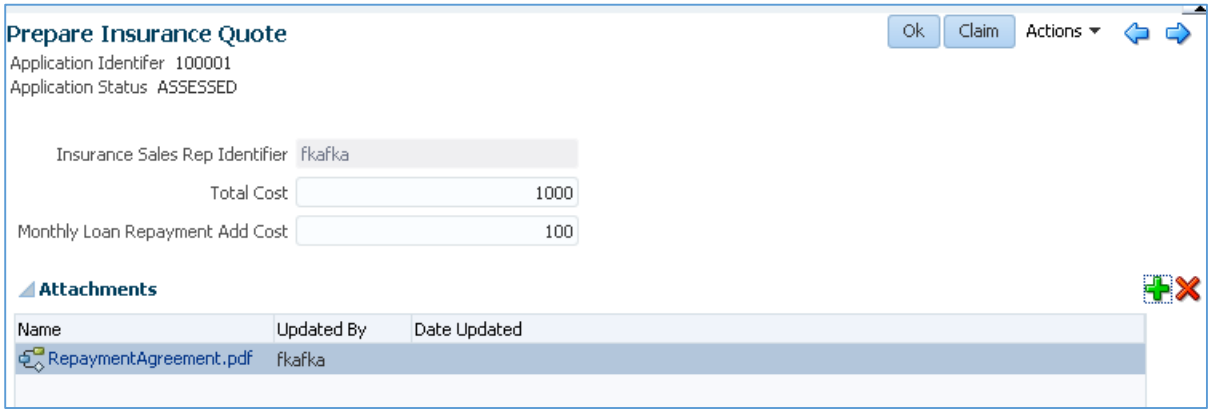
If the Home Insurance Quote Required option is selected in the Enter Application Form screen, the next step will be forwarded to the Insurance Sales Rep user.

Log out BPM workspace and log in as **fkafka** user (Insurance Sales Rep).

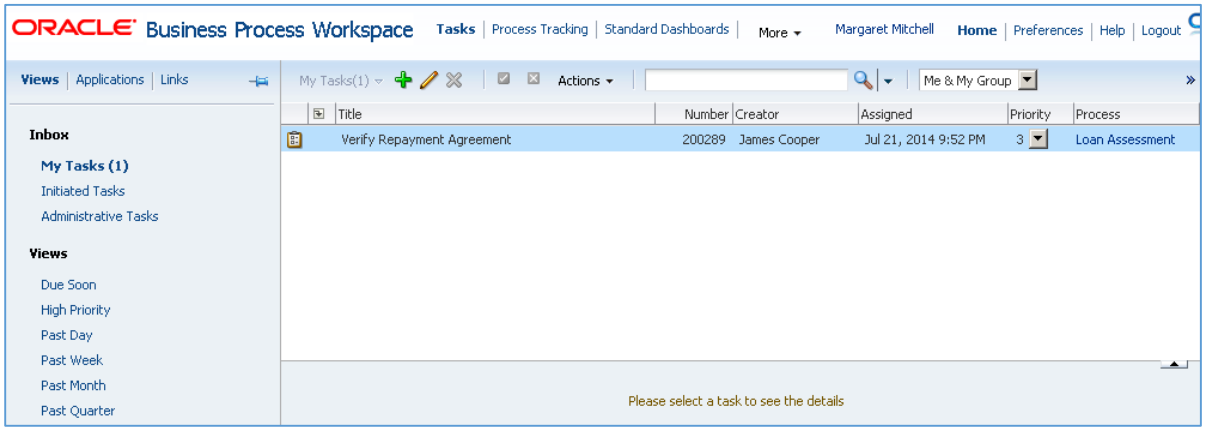


Click on the task to open it.

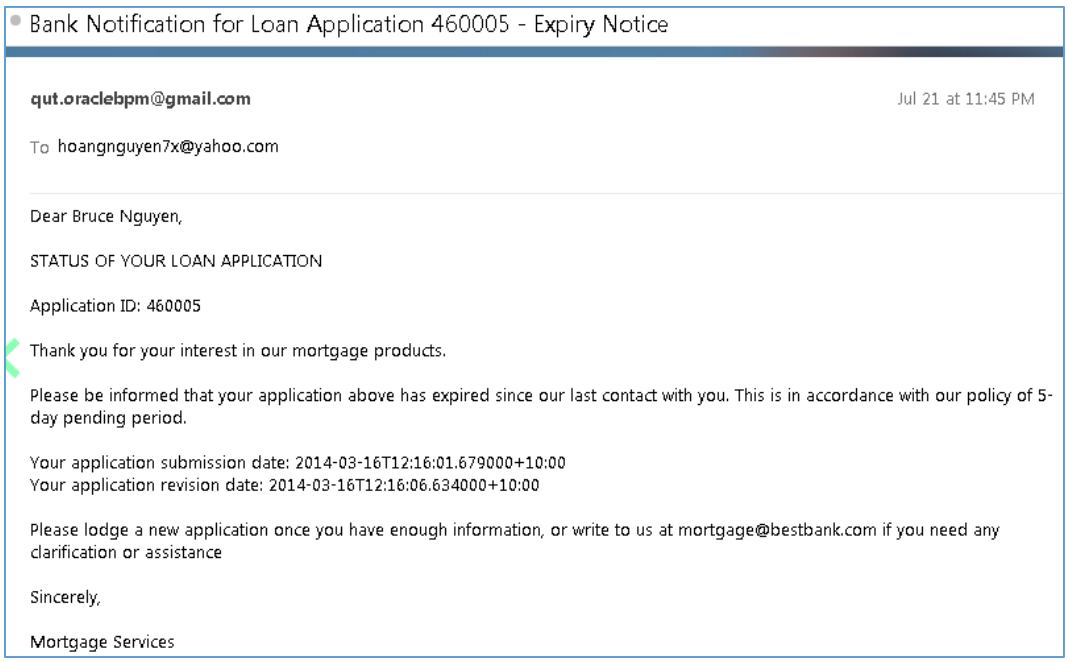
Enter a total cost and monthly added cost as shown below. Then click Ok.



Now, log out BPM Workspace and log in again as **mmitch** user. There is a task shown in the list to verify repayment agreement.



If the user does not open the task and submit it within 5 minutes (timer setting), the process will stop because of application timeout. An email will be sent to the applicant address in the following format.

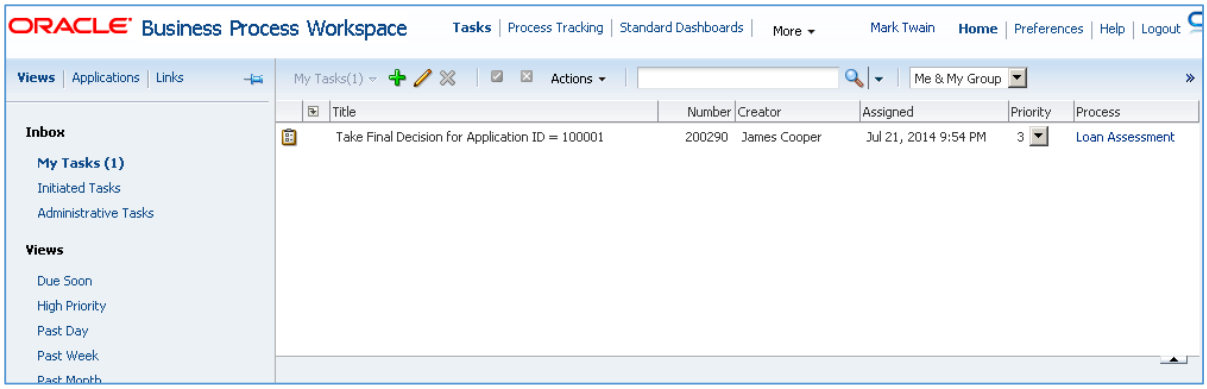


In this main scenario, the user clicks on the task to open it.

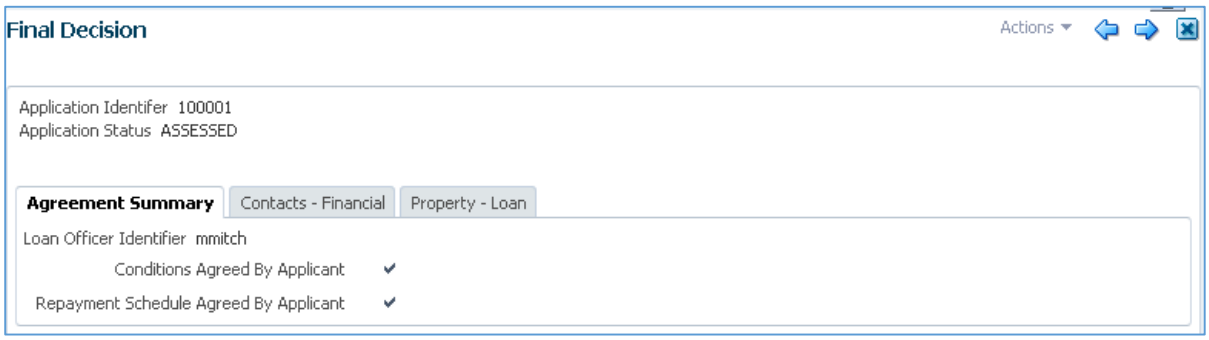
Confirm two conditions agreed by Applicant by clicking on the two check boxes.

Click Ok.

The task list will refresh and display a new task called “Take Final Decision” for loan application ID = 100001.



Click on the task to open it. Note that the current user (mmitch) cannot perform any actions on the task because it must be done by a different Loan Officer user.

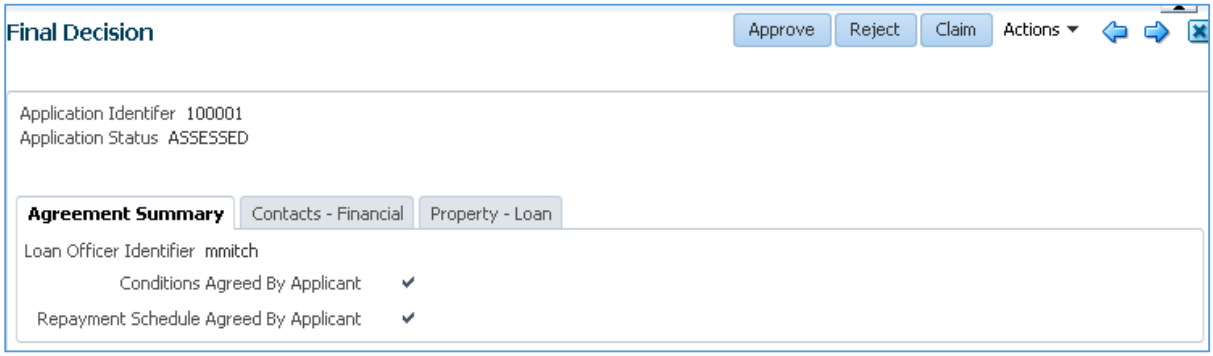


Log out BPM Workspace and log in as **mtwain** user (another Loan Officer).

There is a task in the task list to take final decision for application ID = 100001.

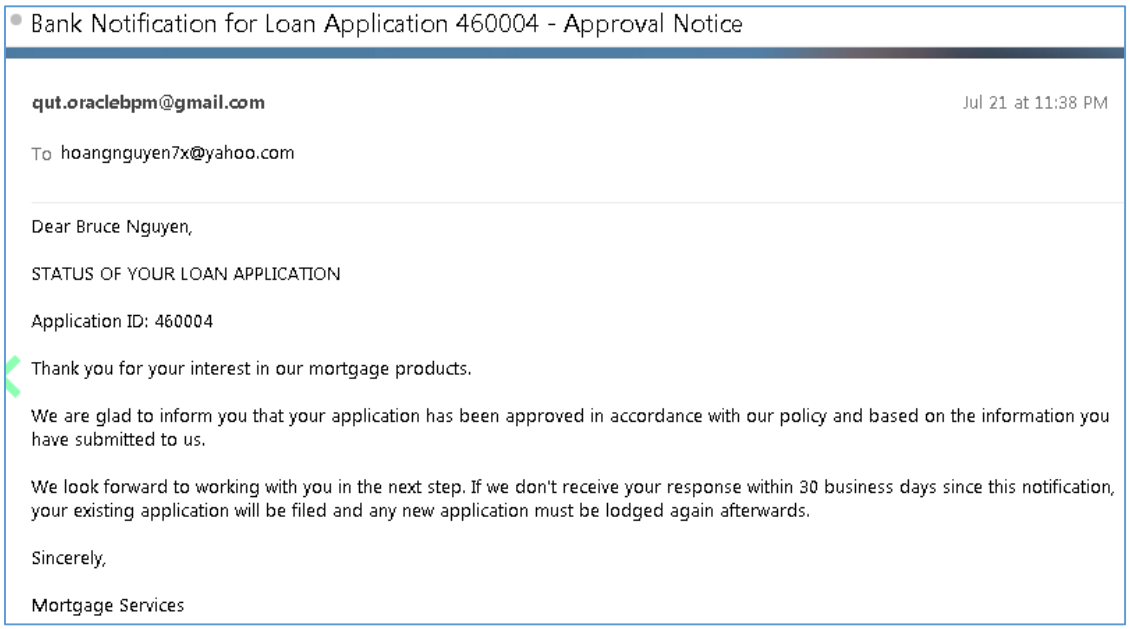
Click on the task to open it.

Click Approve or Reject.

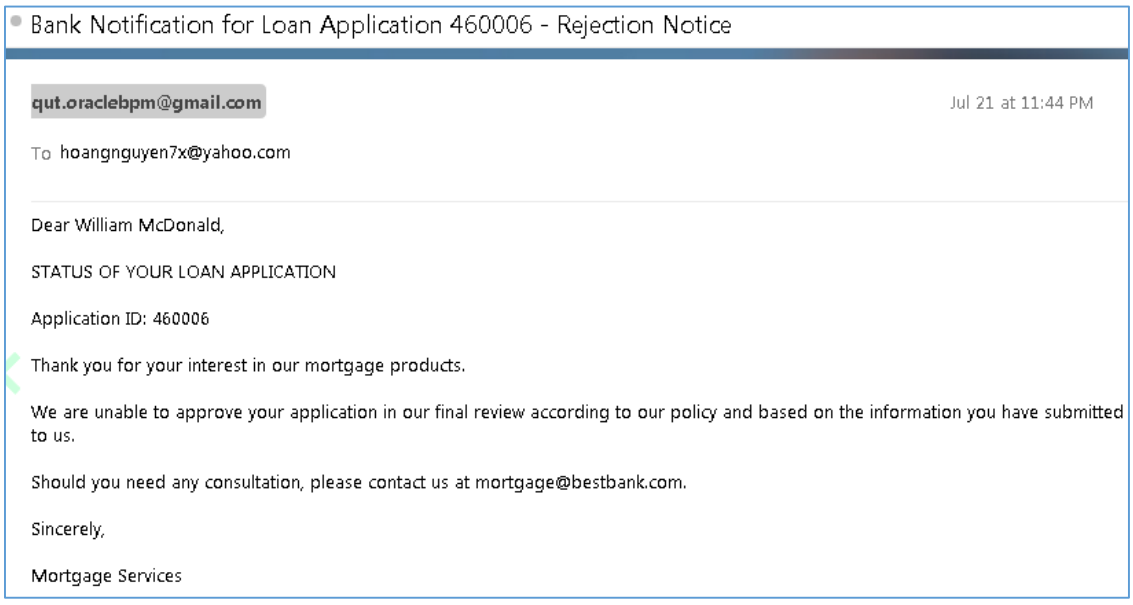


The task will be submitted.

If the application is approved, an approval email is sent to the applicant in the following format.



If the application is rejected, a rejection email is sent to the applicant in the following format.



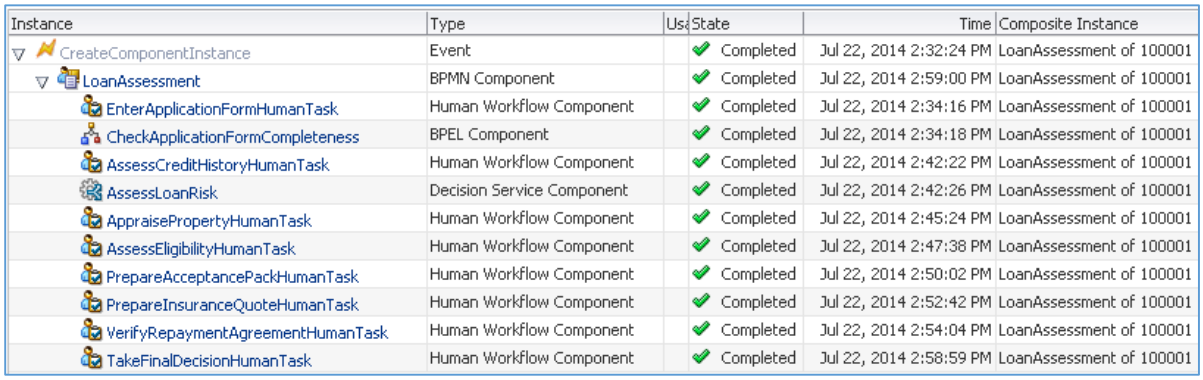
At this point, you have finished a main scenario of Loan Assessment process run.

9.2 Administrator

If you can access to the server as Administrator, you can view the whole trace of the process instance, as shown below.

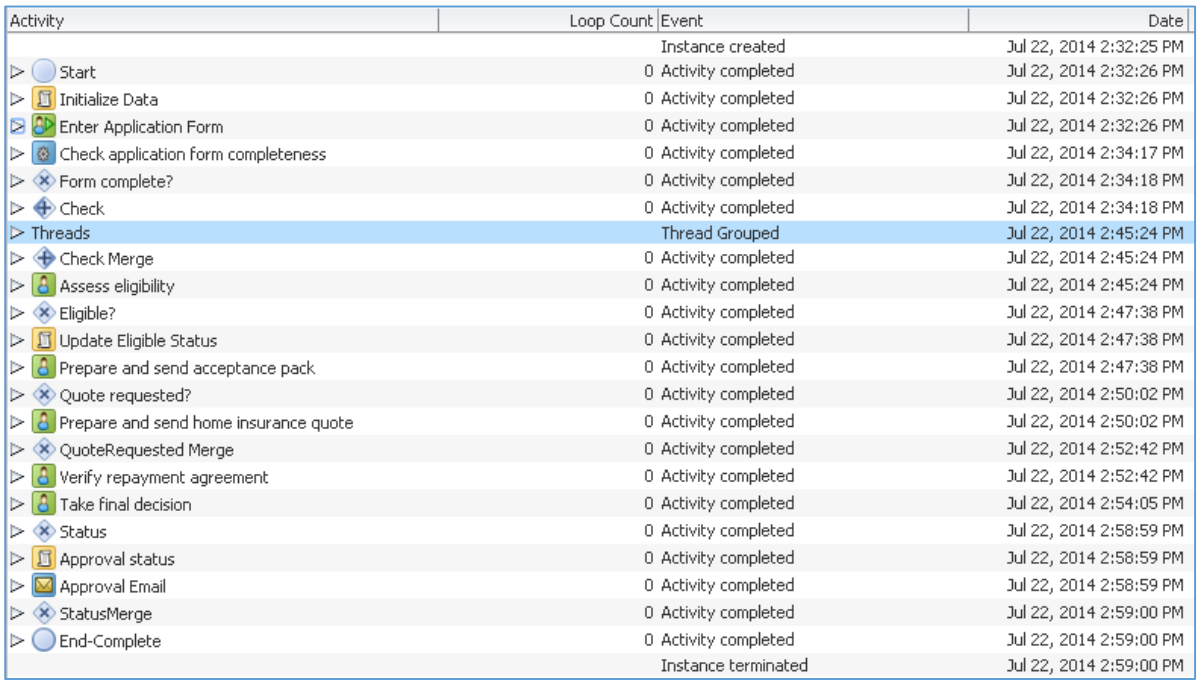
Log in Enterprise Manager as weblogic user (http://localhost:7001/em).

Click SOA > soa_infra > Loan Assessment > click on an Instance link in the list.



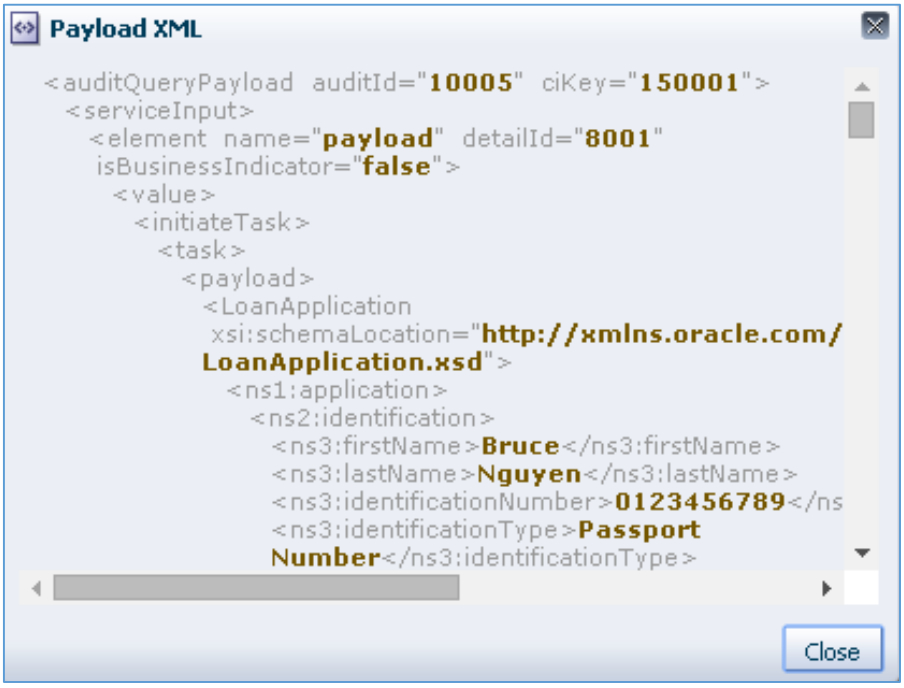
Instance	Type	Us	State	Time	Composite Instance
CreateComponentInstance	Event		Completed	Jul 22, 2014 2:32:24 PM	LoanAssessment of 100001
LoanAssessment	BPMN Component		Completed	Jul 22, 2014 2:59:00 PM	LoanAssessment of 100001
EnterApplicationFormHumanTask	Human Workflow Component		Completed	Jul 22, 2014 2:34:16 PM	LoanAssessment of 100001
CheckApplicationFormCompleteness	BPEL Component		Completed	Jul 22, 2014 2:34:18 PM	LoanAssessment of 100001
AssessCreditHistoryHumanTask	Human Workflow Component		Completed	Jul 22, 2014 2:42:22 PM	LoanAssessment of 100001
AssessLoanRisk	Decision Service Component		Completed	Jul 22, 2014 2:42:26 PM	LoanAssessment of 100001
AppraisePropertyHumanTask	Human Workflow Component		Completed	Jul 22, 2014 2:45:24 PM	LoanAssessment of 100001
AssessEligibilityHumanTask	Human Workflow Component		Completed	Jul 22, 2014 2:47:38 PM	LoanAssessment of 100001
PrepareAcceptancePackHumanTask	Human Workflow Component		Completed	Jul 22, 2014 2:50:02 PM	LoanAssessment of 100001
PrepareInsuranceQuoteHumanTask	Human Workflow Component		Completed	Jul 22, 2014 2:52:42 PM	LoanAssessment of 100001
VerifyRepaymentAgreementHumanTask	Human Workflow Component		Completed	Jul 22, 2014 2:54:04 PM	LoanAssessment of 100001
TakeFinalDecisionHumanTask	Human Workflow Component		Completed	Jul 22, 2014 2:58:59 PM	LoanAssessment of 100001

Click on the LoanAssessment link above, the process steps will be displayed in more details.



Activity	Loop Count	Event	Date
		Instance created	Jul 22, 2014 2:32:25 PM
Start	0	Activity completed	Jul 22, 2014 2:32:26 PM
Initialize Data	0	Activity completed	Jul 22, 2014 2:32:26 PM
Enter Application Form	0	Activity completed	Jul 22, 2014 2:32:26 PM
Check application form completeness	0	Activity completed	Jul 22, 2014 2:34:17 PM
Form complete?	0	Activity completed	Jul 22, 2014 2:34:18 PM
Check	0	Activity completed	Jul 22, 2014 2:34:18 PM
Threads		Thread Grouped	Jul 22, 2014 2:45:24 PM
Check Merge	0	Activity completed	Jul 22, 2014 2:45:24 PM
Assess eligibility	0	Activity completed	Jul 22, 2014 2:45:24 PM
Eligible?	0	Activity completed	Jul 22, 2014 2:47:38 PM
Update Eligible Status	0	Activity completed	Jul 22, 2014 2:47:38 PM
Prepare and send acceptance pack	0	Activity completed	Jul 22, 2014 2:47:38 PM
Quote requested?	0	Activity completed	Jul 22, 2014 2:50:02 PM
Prepare and send home insurance quote	0	Activity completed	Jul 22, 2014 2:50:02 PM
QuoteRequested Merge	0	Activity completed	Jul 22, 2014 2:52:42 PM
Verify repayment agreement	0	Activity completed	Jul 22, 2014 2:52:42 PM
Take final decision	0	Activity completed	Jul 22, 2014 2:54:05 PM
Status	0	Activity completed	Jul 22, 2014 2:58:59 PM
Approval status	0	Activity completed	Jul 22, 2014 2:58:59 PM
Approval Email	0	Activity completed	Jul 22, 2014 2:58:59 PM
StatusMerge	0	Activity completed	Jul 22, 2014 2:59:00 PM
End-Complete	0	Activity completed	Jul 22, 2014 2:59:00 PM
		Instance terminated	Jul 22, 2014 2:59:00 PM

You can even expand a step and click to view the activity payload in XML format as follows.


Payload XML
<auditQueryPayload auditId="10005" ciKey="150001">
<serviceInput>
<element name="payload" detailId="8001"
isBusinessIndicator="false">
<value>
<initiateTask>
<task>
<payload>
<LoanApplication
xsi:schemaLocation="http://xmlns.oracle.com/
LoanApplication.xsd">
<ns1:application>
<ns2:identification>
<ns3:firstName>Bruce</ns3:firstName>
<ns3:lastName>Nguyen</ns3:lastName>
<ns3:identificationNumber>0123456789</ns
<ns3:identificationType>Passport
Number</ns3:identificationType>
Close

10 References

Getting Started with Oracle BPM Suite 11gR1 – A Hands-On Tutorial – Heidi Buelow, Manoj Das, Manas Deb, Prasen Palvankar, Meera Srinivasan.

Getting Started with Oracle SOA Suite 11gR1 – A Hands-On Tutorial – Heidi Buelow, Manas Deb, Jayaram Kasi, Demed L'Her, Prasen Palvankar

Oracle Business Process Management Suite 11g Handbook – Manoj Das, Manas Deb, Mark Wilkins

Oracle® Fusion Middleware User's Guide for Oracle Business Process Management.

11 Appendix

11.1 Further Improvements

Within the timeframe of this project, there are definitely features to be enhanced for the process implemented in this tutorial. Some suggestions are listed below for interested students to develop.

- One single UI project for all Human Task Forms
- Implement Form Validation in Oracle ADF
- Put Applicant to a separate pool (different organization)
- Collaborative event messages
- Integrate with BAM to monitor and analyze process performance
- Add persistence with database adapter
- Use page template and view template to design ADF reusable interface

11.2 Initialization Data

This is initialized data for loan application object.

```
<ns1:LoanApplication
        xmlns:ns1="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/LoanApplication"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/LoanApplication LoanApplication.xsd">
        <ns1:application xmlns:ns2="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/ApplicationInfo">
                <ns2:identification xmlns:ns3="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/IdentificationInfo">
                        <ns3:firstName> </ns3:firstName>
                        <ns3:lastName> </ns3:lastName>
                        <ns3:identificationNumber></ns3:identificationNumber>
                        <ns3:identificationType>Passport Number</ns3:identificationType>
                </ns2:identification>
                <ns2:contact xmlns:ns4="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/ContactInfo">
                        <ns4:email> </ns4:email>
                        <ns4:homePhone></ns4:homePhone>
                        <ns4:cellPhone> </ns4:cellPhone>
                </ns2:contact>
                <ns2:currentAddress xmlns:ns5="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/AddressInfo">
                        <ns5:streetName> </ns5:streetName>
                        <ns5:streetNumber></ns5:streetNumber>
                        <ns5:city>Brisbane</ns5:city>
                        <ns5:postalCode></ns5:postalCode>
                        <ns5:state> </ns5:state>
                        <ns5:surburb> </ns5:surburb>
                        <ns5:durationOfStay></ns5:durationOfStay>
                </ns2:currentAddress>
                <ns2:previousAddress xmlns:ns5="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/AddressInfo">
                        <ns5:streetName> </ns5:streetName>
                        <ns5:streetNumber></ns5:streetNumber>
                        <ns5:city>Brisbane</ns5:city>
                        <ns5:postalCode></ns5:postalCode>
                        <ns5:state> </ns5:state>
                        <ns5:surburb> </ns5:surburb>
                        <ns5:durationOfStay></ns5:durationOfStay>
                </ns2:previousAddress>
                <ns2:financialInfo xmlns:ns6="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/FinancialInfo">
                        <ns6:jobInfo xmlns:ns7="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/JobInfo">
                                <ns7:currentEmployer> </ns7:currentEmployer>
                                <ns7:monthlyNetRevenue></ns7:monthlyNetRevenue>
                                <ns7:jobTitle> </ns7:jobTitle>
                        </ns6:jobInfo>
                        <ns6:bankAccounts xmlns:ns8="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/BankAccountInfo">
                                        <ns8:accountNumber></ns8:accountNumber>
                                        <ns8:accountType> </ns8:accountType>
                                        <ns8:bankName> </ns8:bankName>
                                        <ns8:accountBalance></ns8:accountBalance>
                        </ns6:bankAccounts>
                </ns2:financialInfo>
        </ns1:application>
        <ns1:property xmlns:ns9="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/PropertyInfo">
                <ns9:propertyType> </ns9:propertyType>
                <ns9:address></ns9:address>
                <ns9:purchasingPrice></ns9:purchasingPrice>
        </ns1:property>
        <ns1:loan xmlns:ns10="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/LoanInfo">
                <ns10:amount></ns10:amount>
                <ns10:startDate></ns10:startDate>
                <ns10:interestRate></ns10:interestRate>
                <ns10:interestType> </ns10:interestType>
                <ns10:duration></ns10:duration>
                <ns10:endDate></ns10:endDate>
        </ns1:loan>
        <ns1:administration xmlns:ns11="http://xmlns.oracle.com/bpm/bpmobject/BusinessObjects/AdministrationInfo">
                <ns11:applicationIdentifer> </ns11:applicationIdentifer>
                <ns11:submissionDateTime></ns11:submissionDateTime>
                <ns11:revisionDateTime></ns11:revisionDateTime>
                <ns11:applicationStatus> </ns11:applicationStatus>
                <ns11:applicationStatusComment> </ns11:applicationStatusComment>
                <ns11:eligibility> </ns11:eligibility>
                <ns11:loanOfficerIdentifier></ns11:loanOfficerIdentifier>
        </ns1:administration>
        <ns1:insuranceQuoteRequired>true</ns1:insuranceQuoteRequired>
</ns1:LoanApplication>
```

11.3 Generate PDF Code

11.3.1 DownloadFileBean.java

```
package support;

import com.itextpdf.text.BaseColor;
import com.itextpdf.text.Document;
import com.itextpdf.text.DocumentException;
import com.itextpdf.text.Font;
import com.itextpdf.text.Image;
import com.itextpdf.text.Paragraph;
import com.itextpdf.text.pdf.PdfWriter;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;

import java.io.IOException;

import java.net.MalformedURLException;
import java.net.URISyntaxException;
import java.net.URL;

import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.List;

import javax.faces.context.ExternalContext;
import javax.faces.context.FacesContext;
import javax.servlet.ServletContext;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import oracle.binding.AttributeBinding;
import oracle.binding.BindingContainer;
import oracle.adf.model.BindingContext;



public class DownloadFileBean {

    public DownloadFileBean() {

    }

    private static String FILE = "c:/temp/RepaymentAgreement.pdf";

    private static Font catFont =

        new Font(Font.FontFamily.TIMES_ROMAN, 18, Font.BOLD);

    private static Font redFont =

        new Font(Font.FontFamily.TIMES_ROMAN, 12, Font.NORMAL, BaseColor.RED);

    private static Font subFont =

        new Font(Font.FontFamily.TIMES_ROMAN, 16, Font.BOLD);

    private static Font smallBold =

        new Font(Font.FontFamily.TIMES_ROMAN, 12, Font.BOLD);


    private BindingContainer bindings;


    public void generatePDF(FacesContext facesContext,

                            java.io.OutputStream outputStream) {
```

```
    this.generatePDFFile(facesContext, outputStream); // Generate PDF File
    this.downloadPDF(facesContext, outputStream); // Download PDF File

}

private void downloadPDF(FacesContext facesContext,
                         java.io.OutputStream outputStream) {

    try {
        facesContext = facesContext.getCurrentInstance();
        ServletContext context =
            (ServletContext)facesContext.getExternalContext().getContext();
        ExternalContext ctx = facesContext.getExternalContext();
        HttpServletResponse res = (HttpServletResponse)ctx.getResponse();

        res.setContentType("application/pdf");
        System.out.println(context.getRealPath("/"));
        File file = new File(FILE);
        FileInputStream fdownload;
        byte[] b;
        fdownload = new FileInputStream(file);
        int n;
        while ((n = fdownload.available()) > 0) {
            b = new byte[n];
            int result = fdownload.read(b);
            outputStream.write(b, 0, b.length);
            if (result == -1)
                break;
        }
        outputStream.flush();
    } catch (Exception e) {
        e.printStackTrace();
    }

}

private void generatePDFFile(FacesContext facesContext,
                             java.io.OutputStream outputStream) {
```

```
try {
    System.out.println("In Generate PDF.............");
    Document document = new Document();
    PdfWriter.getInstance(document, new FileOutputStream(FILE));
    document.open();
    addMetaData(document);
    addLogo(document);
    addTitlePage(document);
    addContent(document);
    document.close();
    System.out.println("End of PDF.....................");



    facesContext = facesContext.getCurrentInstance();
    ServletContext context =
        (ServletContext)facesContext.getExternalContext().getContext();



    System.out.println(context.getRealPath("/"));


    File file = new File(FILE);


    FileInputStream fdownload;
    byte[] b;


    System.out.println(file.getCanonicalPath());
    System.out.println(file.getAbsolutePath());
    fdownload = new FileInputStream(file);


    int n;
    while ((n = fdownload.available()) > 0) {
        b = new byte[n];
        int result = fdownload.read(b);
        outputStream.write(b, 0, b.length);
        if (result == -1)
            break;
    }
```

```
            outputStream.flush();


        } catch (Exception e) {

            e.printStackTrace();

        }

        //     /   return null;

        // Add event code here...

    }




    // iText allows to add metadata to the PDF which can be viewed in your Adobe

    // Reader

    // under File -> Properties


    private static void addMetaData(Document document) {

        document.addTitle("My first PDF");

        document.addSubject("Using iText");

        document.addKeywords("Java, PDF, iText");

        document.addAuthor("Lars Vogel");

        document.addCreator("Lars Vogel");

    }

    private static void addLogo(Document document) throws DocumentException,
                                                          FileNotFoundException,
                                                          MalformedURLException,
                                                          IOException,
                                                          URISyntaxException {

        //ExternalContext ext = FacesContext.getCurrentInstance().getExternalContext();
        //String path = ext.getRequestContextPath();
        //path += path.endsWith("/") ? "logo.jpg" : "/logo.jpg";
        //System.out.println("PATH TO IMAGE" + path);

        //String url = ext.encodeResourceURL(path);
        URL                                    url                                    =
FacesContext.getCurrentInstance().getExternalContext().getResource("/logo.jpg");
        System.out.println("XXXXXXX URL IMAGE: " + url.toString());
        Image image1 = Image.getInstance(url);
        document.add(image1);

        Paragraph preface = new Paragraph();
        addEmptyLine(preface, 2);
        document.add(preface);
    }

    private static void addTitlePage(Document document) throws DocumentException {

        Date month = new Date();

        SimpleDateFormat currentMonth = new SimpleDateFormat("MMMM");

        Paragraph preface = new Paragraph();

        // We add one empty line

        addEmptyLine(preface, 1);
```

```
        // Lets write a big header

        preface.add(new Paragraph("Repayment Agreement", catFont));

        addEmptyLine(preface, 5);

        document.add(preface);



        // Start a new page

        // document.newPage();

    }

    private static void addContent(Document document)   throws DocumentException {

        BindingContext bctx = BindingContext.getCurrent();
        BindingContainer bindings = bctx.getCurrentBindingsEntry();
        AttributeBinding monthlyRepaymentAmountBinding =
                    (AttributeBinding) bindings.get("monthlyRepaymentAmount");
        String                              monthlyRepaymentAmount                              =
monthlyRepaymentAmountBinding.getInputValue().toString();
        AttributeBinding numberOfRepaymentBinding =
                    (AttributeBinding) bindings.get("numberOfRepayment");
        String numberOfRepayment = numberOfRepaymentBinding.getInputValue().toString();
        int intMonth = Integer.valueOf(numberOfRepayment).intValue();

        document.add(new Paragraph("The following repayment schedule has been agreed between
the Bank and the Borrower.", redFont));
        document.add(new Paragraph("Installment              Amount", subFont));
        for (int i=1; i<=intMonth; i++) {
            document.add(new Paragraph("Month #" + i + "             " + monthlyRepaymentAmount,
smallBold));
        }

        System.out.println("monthlyRepaymentAmount is "+monthlyRepaymentAmount);

    }



    private static void addEmptyLine(Paragraph paragraph, int number) {

        for (int i = 0; i < number; i++) {

            paragraph.add(new Paragraph(" "));

        }

    }


    public static oracle.binding.BindingContainer getBindings() {

        return BindingContext.getCurrent().getCurrentBindingsEntry();

    }

}
```

11.3.2 PDFGeneratorBean.java

```
package support;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.OutputStream;
import javax.faces.application.FacesMessage;
import javax.faces.context.ExternalContext;
import javax.faces.context.FacesContext;
```

```
import javax.servlet.ServletContext;
import javax.servlet.http.HttpServletResponse;
import oracle.adf.view.rich.component.rich.RichDocument;
import oracle.adf.view.rich.component.rich.RichForm;
import oracle.adf.view.rich.component.rich.layout.RichPanelBox;
import oracle.adf.view.rich.component.rich.layout.RichPanelGroupLayout;
import oracle.adf.view.rich.component.rich.layout.RichPanelStretchLayout;
import oracle.adf.view.rich.component.rich.nav.RichCommandButton;


public class PDFGeneratorBean {

    private RichPanelStretchLayout psl1;

    private RichForm f1;

    private RichDocument d1;

    private RichCommandButton cb1;

    private RichPanelGroupLayout pgl1;

    private RichPanelBox pb1;


    public void setPsl1(RichPanelStretchLayout psl1) {

        this.psl1 = psl1;

    }


    public RichPanelStretchLayout getPsl1() {

        return psl1;

    }


    public void setF1(RichForm f1) {

        this.f1 = f1;

    }


    public RichForm getF1() {

        return f1;

    }


    public void setD1(RichDocument d1) {

        this.d1 = d1;

    }


    public RichDocument getD1() {

        return d1;

    }


    public void setCb1(RichCommandButton cb1) {
```

```
        this.cb1 = cb1;
    }

    public RichCommandButton getCb1() {
        return cb1;
    }

    public void setPgl1(RichPanelGroupLayout pgl1) {
        this.pgl1 = pgl1;
    }

    public RichPanelGroupLayout getPgl1() {
        return pgl1;
    }

    public void setPb1(RichPanelBox pb1) {
        this.pb1 = pb1;
    }

    public RichPanelBox getPb1() {
        return pb1;
    }

    public void addMessage(FacesMessage.Severity type, String message) {
        FacesContext fctx = FacesContext.getCurrentInstance();
        FacesMessage fm = new FacesMessage(type, message, null);
        fctx.addMessage(null, fm);
    }

}
```

11.4 Completeness Checking Code

```
org.w3c.dom.Element element;

try {
    element =
(org.w3c.dom.Element)getVariableData("inputVariable","payload","/ns1:LoanApplication/ns1:appli
cation/ns2:identification/ns3:firstName");
    String firstName = element.getFirstChild().getNodeValue();

    addAuditTrailEntry("element First name: " + element.getTextContent());
    addAuditTrailEntry("First name = " + firstName);

    element =
(org.w3c.dom.Element)getVariableData("inputVariable","payload","/ns1:LoanApplication/ns1:appli
cation/ns2:identification/ns3:lastName");
```

```
    String lastName = element.getFirstChild().getNodeValue();

    addAuditTrailEntry("element Last name: " + element.getTextContent());
    addAuditTrailEntry("Last name = " + lastName);

    String EMAIL_PATTERN = "^[_A-Za-z0-9-\\+]+(\\.[_A-Za-z0-9-]+)*@" + "[A-Za-z0-9-]+(\\.[A-
Za-z0-9]+)*(\\.[A-Za-z]{2,})$";
    Pattern pattern = Pattern.compile(EMAIL_PATTERN);
    Matcher matcher;
    element =
(org.w3c.dom.Element)getVariableData("inputVariable","payload","/ns1:LoanApplication/ns1:appli
cation/ns2:contact/ns4:email");
    String emailAddr = element.getFirstChild().getNodeValue();

    addAuditTrailEntry("element Email: " + element.getTextContent());
    addAuditTrailEntry("Email = " + emailAddr);

    matcher = pattern.matcher(emailAddr);
    boolean emailValid = matcher.matches();

    String errorMsg = "";

    if (firstName.equals("")) {
        errorMsg += "First Name is empty; ";
    }

    if (lastName.equals("")) {
        errorMsg += "Last Name is empty; ";
    }

    if (!emailValid) {
        errorMsg += "Email address is invalid; ";
    }

    setVariableData("outputVariable","payload", "/client:processResponse/client:errorMessage",
errorMsg);

    if (!errorMsg.equals("")) {
        setVariableData("outputVariable","payload",
"/client:processResponse/client:result","incomplete");
    } else {
        setVariableData("outputVariable","payload",
"/client:processResponse/client:result","complete");
    }
} catch (Exception ex) {
    addAuditTrailEntry(ex.getMessage());
}
```

11.5 Common Errors

Below is a collection of common errors the author has encountered during this tutorial lab project. Referring to this list can save you massive time for process implementation if you are new to Oracle BPM technologies.

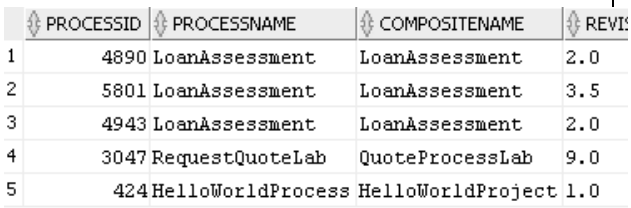
	Error	Resolution
1.	User form appears with no fields for data entry although data objects have been assigned to human task and ADF form.	Cause: if process data object is compound, meaning they contain another data object as elements, then Oracle BPM cannot initialize compound object, even though “auto initialize” is selected in its properties. Solution: either define simple data object only, i.e. only contains primitive fields (string, integer, etc), or initialize compound data object by using Script task and XML literal.
2.	A process/project data object has been defined but not appear in human tasks or other places for reference.	Make sure that “Save” button is pressed straight after data object is added. Then, check again the data object has been added in the list of process

	Error	Resolution
		data objects. Then, check if data object is shown in the reference place.
3.	An error message relating to data transfer between tasks: e.g. subLanguageEvent or Exception	It is likely caused by invalid data schema. Whenever change any business object, all dependent artifacts need to be updated accordingly: process variables, task variables including variables used in rules, service tasks. For example, a LoanInfo business object is deleted and replaced with a new object called CreditInfo, or even it is replaced with new object with the same name because Oracle BPM has a mechanism to keep track of object version and strict data validation will raise an exception if the object has been replaced.
4.	In BPM Process Workspace, Homepage > Applications > Error message: “Runtime Operation Error”. This happens after deployment of a new process version.	There is a conflict with a previous process version. Undeploy previous version would resolve this error.
5.	Compilation error - XML-20108: (Fatal Error) Start of root element expected. in rdf files.	Open rdf file. Add: <?xml version="1.0" encoding="UTF-8" ?> <test> </test> Save
6.	Error in running Ant tasks to create seed demo users. Error message: The name is undefined…..	"Cause: The name is undefined" means the task is not installed in your ant environment. <foreach> is not a task of vanilla ant but needs the ant addon antcontrib available for ant. After installing antcontrib you should use <taskdef resource="net/sf/antcontrib/antlib.xml"/> to activate all antcontrib tasks. Don't use <taskdef resource="net/sf/antcontrib/antcontrib.properties"/>as mentioned on http://ant-contrib.sourceforge.net/ as net/sf/antcontrib/antcontrib.properties contains only tasks for ant versions before Ant 1.6.x Download ant_contrib library (.jar file) from http://ant-contrib.sourceforge.net/ and save to ant/lib directory.
7.	[12:04:19 PM] Weblogic Server Exception: weblogic.application.ModuleException: [12:04:19 PM] Caused by: java.lang.ClassNotFoundException: oracle.bpel.services.datacontrol.types.Number [12:04:19 PM] See server logs or server console for more details. [12:04:19 PM] weblogic.application.ModuleException:	Login to BPM Workspace > Administration Assign sufficient members to the roles without any members. Add Approval Group, assign member to the group. Redeploy the project.

	Error	Resolution
	[12:04:19 PM] Taskflow deployment failed to deploy to server. Remote deployment failed [12:04:19 PM] #### Deployment incomplete. #### [12:04:19 PM] Remote deployment failed (oracle.jdevimpl.deploy.common.Jsr88RemoteDeployer) com.sun.faces.config.ConfigurationException: CONFIGURATION FAILED! oracle.bpel.services.datacontrol.types.Number It happened to UI projects, not the process project.	
8.	When logging into BPM Workspace with a user to start a process, there is no corresponding process under "Application" section.	Log in BPM Workspace > Administration with administrator role. Go to Roles. Check the Role to start the first activity of the process is there. One possible error is it is not created during deployment process. Create the role manually, add users. Log in with the starting user again.
9.	No credential mapper entry found for password indirection user=sys for data source quote.	Go to your Application Properties (in JDev) -> Deployment and uncheck the option "Auto Generate and Synchronize weblogic-jdbc.xml Descriptors During Deployment", rebuild and deploy the application. Explain: Its a WLS feature (Password Indirection). While generating ear file for an application from JDev, it will generate a **-jdbc.xml file for each DB connection in the application resources, set the indirect password attribute, update weblogic-application.xml to add each** -jdbc.xml file as a module and update web.xml (if it exists) to add a resource reference to each jdbc jndi name. However, since there is no server to deploy to, Jdev will not place the passwords in the ear file. The EAR file will not deploy as is. The passwords for the data sources must be setup on the server before the application will run correctly. Check out this link to get some more info and steps to achieve this (http://download.oracle.com/docs/cd/E12839_01/webcenter.1111/e10148/jpsdg_deploy.htm#BGBEHJAJ)
10.	Error when running application	One cause maybe the ADF Framework is not used properly.
11.	Performance wio lib	Update to WIN64 Lib
12.	Error when re-deploying UI components for process forms	We must undeploy the componen on the server before deploying it again. Cannot override in this case

	Error	Resolution
13.	When deploying: warning: Error assignee not specified	
14.	Error file XSD while compiling the project	Open the Business Object property, select option to view all XML structure. There will be more detailed messages to locate the error.
15.	Weblogic Server Exception: weblogic.application.ModuleException: Context path '/workflow/EnterApplicationForm' is already in use by the module: /workflow/EnterApplicationForm application: LoanAssessmentUI	Must undeploy on server before deploying again with override option
16.	Server Error: XPath processing error	Check to make sure the initialized data through XML Literal or XPath Expression is well-formed and qualified. Should check with an XML Validation Tool like XML Spy rather than manual check (very error-prone).
17.	oracle.bpm.bpmn.engine.model.runtime.microinstructions.TrappableException: faultName: {{http://schemas.oracle.com/bpel/extension}subLanguageExecutionFault} messageType: {{http://schemas.oracle.com/bpel/extension}RuntimeFaultMessage} cause: {faultName: {{http://schemas.oracle.com/bpel/extension}subLanguageExecutionFault} messageType: {{http://schemas.oracle.com/bpel/extension}RuntimeFaultMessage} cause: {XPath expression failed to execute. An error occurs while processing the XPath expression; the expression is bpmn:getDataOutput('loanApplication'). The XPath expression failed to execute; the reason was: ORABPEL-77005 Uninitialized data element. DataOutput loanApplication is not initialized in flow element Enter Application Form. Make sure to initialize DataOutput loanApplication before using it in flow element Enter Application Form. Contact oracle support to resolve the issue. . Check the detailed root cause described in the exception message text and verify that the XPath query is correct. } }	No data input is created for Loan Application (Input flow failed)
18.	Displayed form has blank data. The form's labels and non-editable elements are shown but there is no editable elements with data.	It shows that data initialization failed due to some reason during the human task or form creation. One fix is: remove the current human task of the form, remove the associated ADF form. Re-create a new human task and then a new ADF form.
19.	Displayed form is blank: no editable and non-editable elements are displayed, no data. Just a completely white blank page.	There is an error with the server: shut down and restart the SOA server.

	Error	Resolution
20.	BPEL Exception: Mismatch Assign. cannot set a nonmessage value to a message-based variable. An attempt to assign a nonmessage value to a message-based variable failed. Verify the BPEL source for invalid assign activities. ORABPEL-09225. Log file might reports: Model.xml file not found Model.xml file not found for component XX Model.xml file was not generated for the component If component is BPEL 2.0, Model.xml file is not needed and thus does not get generated and this error is expected	This is due to the wrong setVariable method. For example: setVariableData("outputVariable", "payload", "/client:processResponse/client:result", "complete"); The outputVariable is a message type variable, but if we call setVariableData("outputVariable", "complete"), this means set it to a non-message value. Another possible error is the Initialize dialog of the variable property. Can accidentally click on the wrong node. May have to modify directly in the BPEL XML file to remove the initialization. Another note is setVariableData would raise exception if the assigned data was not initialized. Initialization can be done via assigning any value from any value subnode of inputVariable to the concerned subnode of outputVariable.
21.	BEA-000000> <<.> Error while setting task display, this can happen with app loading issue, trying to load for 5>	Redeploy the task form project associated with a human task to only Admin and SOA server, not to other servers (BAM, OSB). This error appears when these servers are not running.
22.	WARNING: Structure is not serialized for: CheckCreditHistoryForm_CheckCreditHistoryHumanTask	Review the data variable in the payload, check if they exist in the Business Objects under Business Catalog or have any errors? Check the XSD files in the businessCatalog/BusinessObjects and XSD folder. In some cases, to fix this error we have to remove and re-create the business object.
23.	Approve, Reject buttons are not available though they are the selected actions in the human task	Check the Human Task to make sure APPROVE and REJECT are among task actions. Refresh the Data Control to make sure the actions from Human Task consistently transferred to Data Control (right click, select Edit Definition, Re-create….., and then select Refresh in the Data Control palette. We have to re-deploy the UI form project to server after this change.
24.	Getting error while deployment of the above saying The XML-Schema file for the fact com.oracle.xmlns.bpm.bpmobject.hello module.reviewobject.ObjectFactory could not be found in the composite. The XML-Schema file xsd/ReviewObject.xsd could not be found in the composite. The XML fact com.oracle.xmlns.bpm.bpmobject.hello	Need to delete the Business Object and re-create it, repeat the business rule creation, this may fix the error.

	Error	Resolution
	module.reviewobject.ObjectFactory of target namespace assumes the existence of the schema file in the composite. Check the composite for the schema xsd/ReviewObject.xsd and make sure it has target namespace. The schema is expected in the project xsd or businesscatalog folder. If the error persists, contact Oracle Support Services.	
25.	Login BPM Workspace and error message: "Runtime Operation Error" ORABPEL-30017 Invalid task definition. The task definition at default/PrjTaskReminderNotification!1.0*soa_5cb0ae06-9485-488f-91b1-1b2872276fb9/Initiator could not be read. The task definition is associated with workflow default/PrjTaskReminderNotification!1.0*soa_5cb0ae06-9485-488f-91b1-1b2872276fb9/Initiator. Make sure that the task definition is available at the specified URL and that it is a valid XML document. If it does not work, the error message in server log is: Could not locate composite for workflow component XXX Ensure composite has been successfully deployed, and that the SOA server has completed loading composites	This happens when there are more than one active record in BPM_CUBE_PROCESS table. You just have to make sure that there is only one active record per process. Execute the following query by connecting to the SOA_INFRA schema of the database select processId, processName, compositeName, revision, status, from BPM_CUBE_PROCESS; If it returns multiple records for the same ProcessName, make the status of all those entries to -1,commit and redeploy the process. Now, you can see the process in the workspace without any problem  If the above still does not work, we might need to re-deploy the same process with the version number as indicated in the error message. Redeploy and run the process again. The status of the process now should be 1 and it should run as normal.
26.	Form display does not have the expected data object elements	Check Data Control, under the corresponding data control, Task > Payload, the data object has been there. If it is not been there (likely), check if it has been added to data configuration of the Human Task associated with the activity. Add if it is not there. If it has been there, refresh data control to make sure it is added to the data control.
27.	The approval and reject action of the form: form does not disappear after pressing these buttons. When reopening	

	PROCESSID	PROCESSNAME	COMPOSITENAME	REVI
1	4890	LoanAssessment	LoanAssessment	2.0
2	5801	LoanAssessment	LoanAssessment	3.5
3	4943	LoanAssessment	LoanAssessment	2.0
4	3047	RequestQuoteLab	QuoteProcessLab	9.0
5	424	HelloWorldProcess	HelloWorldProject	1.0

	Error	Resolution
	the form, the approva/reject button have already been inactive.	
28.	Cannot connect BPM MDS to the Server Caused by: oracle.bpel.services.workflow.client.WorkflowServiceClientException: javax.naming.CommunicationException [Root exception is java.net.ConnectException: t3://2002:83b5:b81e:0:0:0:83b5:b81e:8001: Destination unreachable; nested exception is: java.net.SocketException: Permission denied: connect; No available router to destination] at oracle.bpel.services.workflow.client.WorkflowServiceClientContext.createInitialContext(WorkflowServiceClientContext.java:686) at oracle.bpel.services.workflow.client.WorkflowServiceClientContext.getJNDIInitialContext(WorkflowServiceClientContext.java:341) at oracle.bpm.client.BPMServiceClientContext.getJNDIInitialContext(BPMServiceClientContext.java:228) at oracle.bpm.client.impl.BPMServiceRemoteClient.getHistoryService(BPMServiceRemoteClient.java:339) ... 14 more Caused by: javax.naming.CommunicationException [Root exception is java.net.ConnectException: t3://2002:83b5:b81e:0:0:0:83b5:b81e:8001: Destination unreachable; nested exception is: java.net.SocketException: Permission denied: connect; No available router to destination] at weblogic.jndi.internal.ExceptionTranslator.toNamingException(ExceptionTranslator.java:40) at weblogic.jndi.WLInitialContextFactoryDelegate.toNamingException(WLInitialContextFactoryDelegate.java:788) at weblogic.jndi.WLInitialContextFactoryDelegate.getInitialContext(WLInitialContextFactoryDelegate.java:366)	When trying to connect to the WebLogic Server Administration Server from WLST using localhost as the host name, the following message may be displayed if the listen-address attribute of the Administration Server has been restricted to certain IP addresses: javax.naming.CommunicationException [Root exception is java.net.ConnectException : <t3://HOST:PORT> : Destination unreachable; nested exception is: java.net.ConnectException: Connection refused; No available router to destination **Workaround** Use either of the following workarounds: • Check that the listen-address attribute of the Administration Server has been set correctly in the domain configuration file. You can either remove the listen-address line or simply comment it out. Oracle recommends that you comment it out in case you need to know the value at a later time. For example, in the domain configuration file: • <server> • <name>AdminServer</name> • <ssl> • . • . • . • </ssl> • <machine>*machine_name*</machine> • <!-- listen-address>*machine_ip_address*</listen-address --> • </server> • Use the host name of the Administration Server, instead of localhost, in the WLST connect command.

	Error	Resolution
	at weblogic.jndi.Environment.getContext(Environment.java:315) at weblogic.jndi.Environment.getContext(Environment.java:285) at weblogic.jndi.WLInitialContextFactory.getInitialContext(WLInitialContextFactory.java:117) at javax.naming.spi.NamingManager.getInitialContext(NamingManager.java:667) at javax.naming.InitialContext.getDefaultInitCtx(InitialContext.java:288) at javax.naming.InitialContext.init(InitialContext.java:223) at javax.naming.InitialContext.<init>(InitialContext.java:197) at oracle.bpel.services.workflow.client.WorkflowServiceClientContext.createInitialContext(WorkflowServiceClientContext.java:682) ... 17 more javax.mail.MessagingException: connect failed; nested exception is: java.net.SocketException: Permission denied: connect at com.sun.mail.pop3.POP3Store.protocolConnect(POP3Store.java:161) at javax.mail.Service.connect(Service.java:288) at javax.mail.Service.connect(Service.java:169) at com.myapp.MailboxConnection.connect(MailboxConnection.java:66) caused by: java.net.SocketException: Permission denied: connect	Ok, it doesn't look like a security manager issue. It looks like something in the operating system on that machine is preventing your application from connecting to that host. Try the tips in the JavaMail FAQ for debugging connection problems Problem solved by uninstalling Norton AntiVirus.
29.	java.net.MalformedURLException: Unsupported protocol: t3	Add wlclient.jar and wljmxclient.jar to the server java classpath to process t3 protocol.

	Error	Resolution
	or BPM MDS cannot establish connection with the server	
30.	Look at the diagnostic log: oracle.adf.controller.activity.ActivityLogicException: ADFC-06014: An exception occured when invoking a task flow finalizer ADF_FACES-30200:Fatal exception during PhaseId: RESTORE_VIEW 1. The UIViewRoot is null, this is usually caused by previous exceptions, for more complete debugging information turn the logging level to fine.	"The UIViewRoot is null, this is usually caused by previous exceptions"
31.	ADFC-12000: State ID in request is invalid for the current session ADF_FACES-60096:Server Exception during PPR, #3	Review this link: http://one-size-doesnt-fit-all.blogspot.in/2011/10/pageflowscope-with-unbounded-task-flows.html **How does ADF technically solve identifying the separate tabs** **"…**Side note: Behind the scenes the server is smart enough to check the session parameters against the previous known connection/session to stop intruders impersonating another user's session ... you can test this by intercepting the next request before it goes out and changing the _adf.ctrl-state parameter before it hits the server. ADF will complain displaying the following error message "ADFC-12000: State ID in request is invalid for the current session." **For this error: open BPM/Workspace in another browser window, maybe clear the cache of current window, and conduct the transaction in the new window.**
32.	There was an error deploying the composite on soa_server1: Deployment Failed: Error in getting XML input stream: oramds:/deployed-composites/default/LoanAssessment_rev5.0/SCA-INF/classes/xsd/InitializationData - FULL.xml: Illegal character in path at index 96: oramds:/deployed-composites/default/LoanAssessment_rev5.0/SCA-INF/classes/xsd/InitializationData - FULL.xml.	Reason: InitializationData - FULL.xml filename contains spaces. This is invalid for MDS upload. This error will not occur if this project is not yet linked to MDS. Fix: rename file without spaces. In addition, Oracle BPM might automatically use previous settings/files if the new deployment reuses a previous revision number, which makes this error still occur. This happens even there is no such XML file in deployed .jar file (*deploy subfolder of project folder*) or server cache (*user_projects\domains\domain1\servers\soa_server1\dc*). In order to avoid this error, new deployment must use an unused revision number or a number without having this error before.

	Error	Resolution
33.	Process fault occurs without any user error message. If viewing process instance in Enterprise Manager console, the instance shows a system fault at Java embedding activity (RuntimeFault).	This may be caused by java codes in Java Embedding activity. One debugging method is insert addAuditTrailEntry() in codes to investigate sources of error.
34.	Cannot create MDS connection to server. Test connection always return "Test Failed".	One reason is firewall on local computer or remote server.
35.	When refreshing Data Control in BPM Studio, there is warning: "Structure is not serialized for <data control>"	There is an error with the payload associated with the human task in terms of schema validation. There is a tricky error: Oracle BPM only allows one array element business object. If more than one array are defined in the business object, it would not be able to be parsed successfully.
36.	In the Project schema files list pop up from selecting data for a human task, Oracle BPM displays both obsolete and current schema files, i.e. there are duplicate items in the list, only one of them is current to be valid for selection, other may have parsing error.	Notice to scroll down the list to select the right one.
37.	the exception reported is: java.lang.RuntimeException: failed to compile execlets of	This is a compilation error of Java Embedding code. One issue may be the missing "import" statement for one of required classes used in the code. Open BPEL process, switch to Source view, add "<import location="<java path to package and class>" importType="http://schemas.oracle.com/bpel/extension/java" />" to the XML source. This applies to even native Java classes, e.g. java.util.regex.Pattern.