OFSAAI Process Modelling Framework Orchestration Guide

Copyright © 2019 Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

For information on third party licenses, click here.
### Document Control

<table>
<thead>
<tr>
<th>Version Number</th>
<th>Revision Date</th>
<th>Change Log</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Created Nov 2018</td>
<td>Captured enhancement done in 8.0.7.0.0 release.</td>
</tr>
<tr>
<td>2.0</td>
<td>Modified April 2019</td>
<td>Updated the following sections for the enhancements done in 8.0.7.1.0 ML:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Configuring Application Object Model (AOM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Executing Run Pipeline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Abort Run Pipeline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Resume Run Pipeline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Re-run Run Pipeline</td>
</tr>
<tr>
<td>3.0</td>
<td>Modified May 2019</td>
<td>Added a new Appendix for <a href="#">Support APIS for Java External APIs</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Added new section <a href="#">Application Registration</a></td>
</tr>
</tbody>
</table>
Table of Contents

1 Introduction .................................................................................................................. 9
   1.1 Key Features of Process Modelling Framework ................................................................. 9
   1.2 Process Pipeline Flow ......................................................................................................... 10
   1.3 Accessing Process Modelling Framework ........................................................................... 10

2 Process Modeller ......................................................................................................... 12

3 Components for Designing Your Process Flow ...................................................... 14
   3.1 Transitions ............................................................................................................................. 14
   3.2 Gateways ................................................................................................................................. 14
   3.3 Connector ................................................................................................................................. 15
   3.4 Human Task .............................................................................................................................. 15
   3.5 Service Task .............................................................................................................................. 15
   3.6 Sub Pipeline ............................................................................................................................... 15
   3.7 Inline Process ............................................................................................................................. 15
   3.8 OFSAA Widgets ......................................................................................................................... 15

4 Artifacts of Process Modelling ................................................................................. 16
   4.1 Application Package ............................................................................................................... 16
   4.2 Application Registration ......................................................................................................... 16
      4.2.1 Object Type Entry ............................................................................................................ 16
      4.2.2 Object Type Process Mapping ......................................................................................... 17
   4.3 Data Fields ............................................................................................................................... 17
      4.3.1 Adding a Data Field ........................................................................................................ 17
      4.3.2 System Data Fields ......................................................................................................... 19
      4.3.3 AOM Data Field ............................................................................................................. 19
   4.4 Application Rules ....................................................................................................................... 19
      4.4.1 SQL Application Rule: .................................................................................................. 20
      4.4.2 Stored Procedure Application Rule: .............................................................................. 21
      4.4.3 For Function Application Rule: ..................................................................................... 23
      4.4.4 For Java Application Rule: ............................................................................................ 24
      4.4.5 For Java External API Application Rule: ...................................................................... 26
      4.4.6 For Outcome Rules Application Rule ............................................................................... 28
4.4.7 Expression Application Rule ................................................................. 28
4.4.8 Rest Service Application Rule ............................................................... 30
4.4.9 For Attribute Expression Application Rule ............................................. 32
4.4.10 For Advanced Attribute Expression Application Rule ......................... 33
4.4.11 For JSON Path Expression Application Rule ......................................... 34
4.4.12 JSON Read From DB Application Rule ............................................... 35
4.4.13 JSON Write To DB Application Rule ..................................................... 37

4.4.14 Configuring Application Object Model (AOM) ....................................... 38
4.4.1 .1 Attribute Types .................................................................................... 39
4.4.1 .2 AAI_AOM_APP_COMP_ATTR_TL Table.............................................. 39
4.4.1 .3 Usage of AOM Attributes in Run Pipeline ........................................... 40

4.5 Configuring Application Object Model (AOM) ....................................... 38
4.5.1 Attribute Types ...................................................................................... 39
4.5.2 AAI_AOM_APP_COMP_ATTR_TL Table .................................................. 39
4.5.3 Usage of AOM Attributes in Run Pipeline ............................................. 40

5 Designing a Pipeline .................................................................................... 41

5.1 Orchestration of a Business Pipeline ........................................................ 41
5.1.1 An Example of a Business Pipeline ....................................................... 41
5.1.2 Creating a Business Pipeline ................................................................. 41

5.2 Run Pipeline ............................................................................................. 43
5.2.1 An Example of Run Pipeline ................................................................. 44
5.2.2 Creating a Run Pipeline ......................................................................... 44
5.2.3 Design your Run Pipeline using OFSAA Widgets ................................. 45
5.2.4 Design your Run Pipeline using Sub Pipeline ...................................... 45
5.2.5 Design your Run Pipeline using Inline Process .................................... 46
5.2.6 Design your Run Pipeline using combinations of OFSAA Widgets, Sub Pipeline and Inline Process 46
5.2.7 Executing Run Pipeline ........................................................................ 46
5.2.8 Abort Run Pipeline ............................................................................... 48
5.2.9 Resume Run Pipeline ............................................................................ 49
5.2.10 Re-run Run Pipeline ........................................................................... 49

6 Human Tasks ............................................................................................... 50

6.1 How to Use Human Task ......................................................................... 50
6.1.1 Activity Tab .......................................................................................... 51
6.1.2 Action Tab for Creating Tasks/Notification ......................................... 51
6.2 Additional Functionalities for Human Tasks ............................................ 54
6.2.1 Setting Task Expiry

6.2.2 Setting Task Escalation

6.2.3 Setting Reminder for Your Task

7 Service Tasks

7.1 How to Use a Service Task

7.1.1 Activity Tab

7.1.2 Implementation Tab

8 Configuring OFSAA Tasks in Your Process Flow

8.1 How to Use an OFSAA Widget

8.1.1 Dynamic Parameters for OFSAA Widgets

9 Orchestrating External Models/Components in Your Process Flow

9.1 How to invoke External Model through Web Service

9.1.1 Data Preparation

9.1.2 Webservice Invocation

9.1.3 Data Extraction

10 Configuring Custom Components

10.1 AAI_WF_COMPONENT_REGISTRATION Table

10.2 AAI_WF_COMPONENT_PARAMETERS Table

11 Executing Parallel Tasks

11.1 How to Use Parallel Gateways

12 Calling another Pipeline from Your Parent Pipeline

12.1 How to Configure Sub Pipeline

13 Inline Process

13.1 An Example of Inline Process

13.2 How to Add an Inline Process

14 Configuring Email for Human Tasks

14.1 AAI_EMAIL_CONFIG Table
17.3 Delegation ...................................................................................................................................... 100
17.3.1 Adding a Delegate .................................................................................................................... 101
17.3.2 Viewing Delegation ................................................................................................................... 103
17.3.3 Modifying Delegate Details ....................................................................................................... 104
17.3.4 Revoking Delegation ................................................................................................................ 104
17.3.5 Deleting Delegation .................................................................................................................. 104

18 Appendix B: Support APIs for Java External APIs .................................................................105

18.1 Connection API .............................................................................................................................. 105
18.1.1 Jar Files Required .................................................................................................................... 105
18.1.2 ConnectionAdapter Methods .................................................................................................... 105
18.1.3 Connection to Config Schema .................................................................................................. 105
18.1.4 Connection to Atomic Schema ................................................................................................. 106
18.2 Logging API .................................................................................................................................... 107
18.2.1 Jar Files Required .................................................................................................................... 107
18.2.2 Debug Message .......................................................................................................................... 107
18.2.3 Error Message .......................................................................................................................... 107
1 Introduction

Process Modeling Framework (PMF) is a design and execution framework that enables Process Pipeline developers to implement various Pipelines modelled by business analysts. Process Pipeline developers use the framework to orchestrate the Business Pipelines and Run Pipelines within OFSAA, and also to design the artifacts that participate in the Pipelines, in order to complete their implementation.

The Process Modeling Framework consists of Process Modeling components for modeling Pipelines and Process Monitor components for monitoring instantiated Pipelines of OFSAA applications. See Process Flow for more information on how these tools fit into the Pipeline design and implementation.

Process Modeller aids in representing the various artifacts required for modeling and provides implementation details of the OFSAA process artifacts.

- OFSAA Process Pipeline or Run Pipeline
- Reusable process components like Sub Pipeline
- Process data (Data Fields)
- Implementation of various types of Human Tasks / Service Tasks
- Business Rules (Application Rules)
- Various External services implementations and other artifacts needed for complex implementations
- Configuring Notifications

1.1 Key Features of Process Modelling Framework

- Support for visual modeling of the pipelines.
- Support for registration of Process /Activity/ Transition Logic implementation, separated from the modeling itself.
- Built-in orchestration engine (included within OFSAAI’s runtime) for task execution (interactive model as opposed to the batch model supported through Rule Run Framework).
- Published interface for abstraction of task implementation.
- Representation of the pipeline-routing rule logic in Java/ PL-SQL / Web-service.
- Stitching of OFSAA Components within the Process Pipeline
- Orchestration and execution of RRF Run
- Reminder, Escalation, and Expiry of tasks
- Registration of Custom Widgets
- Process Monitoring Admin Tool to view the execution Process Instances.
1.2 Process Pipeline Flow

1.3 Accessing Process Modelling Framework

From the OFSAA Landing screen, click Administration.
• Select the **Information Domain** from the drop-down list.

• Click **Process Modelling Framework** tile to display the sub menu.

• Click **Process Modeller** to launch the Process Modeller.

• Click **Process Monitor** to monitor currently running processes.

• Click **Delegation** to launch the Delegation framework.
2 Process Modeller

The Process Modeller window displays the existing Business Process Pipelines and Run Pipelines with the details such as Process ID, Process Name, Process Description, Version, Instance, Application, and Last Modified details.

You can do the following tasks from this window:

- Click + to create a new Pipeline.
- Click the Process Name link to launch and edit the Process Flow.
- Click - to delete a Pipeline.
- Click : to view the following sub menu:
  - Click View to see the process flow.
  - Click Copy to create a new Pipeline with the same process flow.
  - Click Process Flow Monitor to monitor the Pipeline.
  - Click Test Process Flow to test/check whether the Business Pipeline you designed works as expected.
  - Click Execute Run to execute a Run Pipeline.
• Click **Export Process** to export a Process.

• Using the **Search** grid, you can search for a specific Pipeline by providing a keyword from Process ID, Process Name or Process Description and clicking **Q**. Click **C** to reset the Search fields.

• You can sort the Pipelines based on Process ID, Process Name, or Application. Click the **Sort by** drop-down and select the attribute by which you want to sort.

• You can use **Filter Pipeline** field to filter pipelines based on pipeline type. For example, if you want to view only Run Pipelines, remove **Process** from the **Filter Pipeline** field.

• Click **Process Monitor** to go to **Process Monitor** window.
3 Components for Designing Your Process Flow

The Process Flow tab has a floating toolbar and a drawing canvas. Drawing canvas is used to design the Process flow with the Tools, Activities, and OFSAA Widgets available in the floating toolbar.

3.1 Transitions

Transition is used to control the flow between various components in the Process flow. Transition connects two activities and the flow is configured based on conditional expression or decision rule. You can use Transition Line if you want to use straight line to connect the activities or Transition Curve if you want to use curved lines, based on your requirement.

3.2 Gateways

Split refers to a condition where an incoming transition is split to multiple transitions. In Merge, multiple incoming transitions are merged to a single transition. The Splitting and Merging of Activities are modeled through Gateways. Gateway can be Sequential, Parallel, or Multi Choice.
3.3 Connector

Connector helps to connect two activities with different path (other than the default), in case if the default path is overlapping with some existing flow.

3.4 Human Task

Human task requires human intervention to move to next Activity. For more information on stitching human tasks in your Process flow, see Human Tasks section.

3.5 Service Task

Service task typically invokes an application component (for example, activity to invoke a business rule to calculate certain threshold). For more information on stitching service tasks in your Process flow, see Service Tasks section.

3.6 Sub Pipeline

Sub Pipeline provides reusability of Pipelines. Using Sub Pipeline component, you can call another Pipeline from your parent Pipeline. For more information on how to use Sub Pipeline, see Calling another Pipeline from Your Parent Pipeline section.

3.7 Inline Process

Inline Process is used to group a set of processes within a process flow. It is a process flow within another process flow, which is not reusable like Sub Pipeline. For more information, see Inline Process section.

3.8 OFSAA Widgets

OFSAA widgets are used to execute OFSAA components like T2T definitions, PLC definitions (DT), Rules (Classification Rule and Computation Rule), Models in EMF, RRF Runs, Data Quality Groups, and RRF Processes through Process Modeller. For more information, see Configuring OFSAA Tasks in Your Process Flow section.

You can register a new component by entering details in the AAI_WF_COMPONENT_REGISTRATION table. For more information, see Configuring Custom Components section.
4 Artifacts of Process Modelling

Before you start designing your Pipeline, it is required to get introduced to some artifacts of Process Modelling Framework.

4.1 Application Package

Application package is a concept used to group Pipelines, Application Rules, and Data Fields which are required for an Application. When you create a Pipeline, you should select the Application Package in which the Pipeline needs to be available. Similarly, when you define a Data Field or an Application Rule, you can set it to be available across Pipelines created in that Application Package.

You can add a new package by adding a new entry in the AAI_WF_APP_PACKAGE_B table.

<table>
<thead>
<tr>
<th>V_APP_PACKAGE_ID</th>
<th>V_APP_PACKAGE_DESC</th>
<th>V_DEFINITION_PAGE_URL</th>
<th>V_IS_EMAIL_REQUIRED</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Global</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>10</td>
<td>Business Restructure</td>
<td>Restructure/managed_grid.jsp?userId=[ASSIGNEEUSERS]&amp;block= -</td>
<td>N</td>
</tr>
<tr>
<td>100</td>
<td>Platform</td>
<td>-</td>
<td>Y</td>
</tr>
<tr>
<td>11</td>
<td>Questionnaire</td>
<td>solution/abc_qtns/ConvertRedirectPMFlogin.jsp?appCode=aqs</td>
<td>Y</td>
</tr>
</tbody>
</table>

V_APP_PACKAGE_ID – Enter a unique application package ID.
V_APP_PACKAGE_DESC – Enter a description for the application package.
V_DEFINITION_PAGE_URL – Enter the URL of the definition page of the Application.
V_IS_EMAIL_REQUIRED – Set this as Y for configuring email at Application Level.

4.2 Application Registration

This section is applicable only for Business Pipeline.

Application registration is required to define the Entity and the Attributes which needs to updated for a Business Pipeline. These entries need to be seeded in the AAI_WF_APP_REGISTRATION and AAI_WF_APP_DEFINITION_MAP tables.

4.2.1 Object Type Entry

The AAI_WF_APP_REGISTRATION table stores information like the Object Type, Fact Table, Primary Key Column Name, Object Name, Object Type Name and so on for an Application Package.

<table>
<thead>
<tr>
<th>V_APP_PACKAGE_ID</th>
<th>V_OBJECT_TYPE</th>
<th>V_FACT_TABLENAME</th>
<th>V_PRIMARY_KEY_COL_NAME</th>
<th>V_OBJECT_NAME_COL</th>
<th>V_OBJECT_TYPE_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>FR_EE</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>2</td>
<td>FR_SRT</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>3</td>
<td>AMI_SRV</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>4</td>
<td>AMI_ERC</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>5</td>
<td>AMI_SD</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>6</td>
<td>FR_ON</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>7</td>
<td>KIC_FRM</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>8</td>
<td>KIC_RD</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>9</td>
<td>KIC_CRP</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>10</td>
<td>CS_SAN</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>11</td>
<td>CS_SRV</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>12</td>
<td>CS_EO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>13</td>
<td>CS_EO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>14</td>
<td>CS_EO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>15</td>
<td>CS_EO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>16</td>
<td>CS_EO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>17</td>
<td>CS_EO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>18</td>
<td>CS_EO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>19</td>
<td>CS_EO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
<tr>
<td>20</td>
<td>CS_EO</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>Case Management</td>
</tr>
</tbody>
</table>
4.2.2 Object Type Process Mapping

The AAI_WF_APP_DEFINITION_MAP table stores Process IDs against the required Object Type. Note that multiple object types can be mapped to a single Process.

<table>
<thead>
<tr>
<th>Process ID</th>
<th>Object Type</th>
<th>Value</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>2</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>3</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>4</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>5</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>6</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>7</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>8</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>9</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>10</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>11</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>12</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>13</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>14</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>15</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
<tr>
<td>16</td>
<td>AAI_DAO</td>
<td>Y</td>
<td>02/17</td>
</tr>
</tbody>
</table>

4.3 Data Fields

Data Field, which is also known as Process Variable, helps Process Pipelines to access and store information from outside application. Often the process flow is based on the value of this information. In other cases, this information is the result of running the tasks in the Pipeline.

4.3.1 Adding a Data Field

1. From the Process Modeller window, click corresponding to the Pipeline for which you want to add a Data Field. The Process Flow tab is displayed.
2. Select the DataFields tab.
3. Click Add. The Data Field Details window is displayed.

4. Enter the details as given in the table:
### Field Name | Description
--- | ---
**Data field Code** | Enter the Variable Name/Code, which needs to be used by application to read or write in to this variable. This field is non translatable.
**Data field Description** | Enter a brief description of the Data field.
**Data field Type** | Enter the Data Field type. The supported types are String, Integer, Int, Float, AOM (Application Object Model), and Boolean. For more information on where we use AOM Data Field type, see [AOM Data Field section](#).
**Initial Value** | Enter the default value for the Data.
**Is Mandatory** | This field is applicable for Data Field Type of AOM. By default, for AOM Data Field, this is selected as **Yes**. Select **No** if you do not want this parameter to be displayed as Execution Parameter for the Run Pipeline.
**Scope** | Select the scope of the Data Field from the drop-down list. The options are:
- **Process**: Select Process if you want to use the Data Field only in the current process.
- **Package**: Select Package if you want to use the Data Field across all the processes in the package.

5. **Click OK.**

#### 4.3.2 System Data Fields
Some data are tracked internally by the System using a predefined set of Data Fields such as Status of Process. You can access these activity instance attributes in the same way you access regular data objects, but you cannot assign them new values.

#### 4.3.3 AOM Data Field
The AOM Data Fields are automatically created from the entries in the AAI_AOM_APP_COMP_ATTR_MAPPING table. These Data Fields, which are marked as mandatory, will be displayed in the **Select Run Parameters** window while executing Run Pipeline. For configuring AOM Data Fields, see [Configuring Application Object Model (AOM)](#).

#### 4.4 Application Rules
The Application or API Rule is the interface between the process engine and the application, including any parameters to be passed.

Based on their usage these are categorized into three types.

- **Execution Rule**: These are Business Logic executed as Task by an Activity.
- **Decision Rule**: This rule returns Boolean value "True/False", used in decision making during split/branching of transition.
• **Selection Rule**: This rule fetches some value, useful to get value dynamically from a table or other source.

  For example, `select v_created_by from fct_expenses where id=101`

### 4.4.1 SQL Application Rule:

This Application Rule is used to execute any SQL queries in the Process Flow.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the SQL Application Rule.</td>
</tr>
<tr>
<td>Rule Type</td>
<td>Select the <strong>Rule Type</strong> from the drop-down list. The SQL Application Rule can be used as Decision Rule, Execution Rule or Selection Rule based on your requirement.</td>
</tr>
<tr>
<td>Execution Type</td>
<td>Displays the Application Execution Type as SQL.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Implementation Detail| • **Decision Rule**- For Decision Rule the SQL Statement should return 'PASS' for success condition.  
  For example, select 'PASS' from dual where (EXPENSES) <= (THRESHOLD)  
  **Note:** {EXPENSES}, {THRESHOLD} are user defined Data Fields.  
  • **Execution Rule**- For Execution Rule the SQL Statement can be any DML statement.  
  For example, update fct_expenses set expenses={EXPENSES} where id = {WF_ENTITYID}  
  • **Selection Rule**- For Selection Rule the SQL Statement should be a Select statement which returns a list of values.  
  For example, select v_created_by from fct_expenses id = {WF_ENTITYID}                                                                                                                                                                                                                                           |
| Return Parameter     | Select the Data Field which will receive the return parameter of the SQL Rule, from the drop-down list.  
  • For ExecutionRule type, the business logic is implemented in the method and the parameter value returned from the SQL Rule is saved in the mapped Data Field.  
  • In case of SelectionRule type, the Java method should be a String value.  
  • In case of DecisionRule type, the Java method should return Boolean values “True/False”.                                                                                                                                                                                                                         |
| Scope                | Select the **Scope as Process** to use the Application Rule only in the current process or **Package** to use the Application Rule across all the processes in the package.                                                                                                                                                                                                                           |

### 4.4.2 Stored Procedure Application Rule:

This Application Rule is used to call a Stored Procedure in your Process Flow.
Field Name | Description
--- | ---
Name | Enter a unique name for the SQL Application Rule.
Rule Type | Select the **Rule Type** from the drop-down list. The Stored Procedure Application Rule can be used as Decision Rule, Execution Rule or Selection Rule based on your requirement.
Execution Type | Displays the Application Execution Type as Stored Procedure.
Procedure Name | Enter the Stored Procedure Name.
Input Parameters | Select the list of Data Fields which will be passed as input parameters, from the drop-down list.
Parameter Mode | Enter the Parameter Mode in JSON format. For example, suppose you have given 3 parameters as input parameters, enter parameter mode as ['IN','IN','OUT']. During Execution of Stored Procedure,
  - In case of Decision Rule type, the first return parameter should return value 'PASS' for success evaluation.
  - In case of Selection Rule type, the first return parameter value is taken as Selection data.
  - In case of Execution Rule, the procedure return OUT parameter value overwrites the current value of respective mapped Data Field.
### Field Name | Description
--- | ---
Scope | Select the **Scope** as **Process** to use the Application Rule only in the current process or **Package** to use the Application Rule across all the processes in the package.

#### 4.4.3 For Function Application Rule:

This Application Rule is used to call Database functions in your Process Flow.

![Rule Details](image)

### Field Name | Description
--- | ---
Name | Enter a unique name for the Application Rule.
Rule Type | Select the rule type from the drop-down list. The available rule types are Decision Rule, Execution Rule and Selection Rule.
Execution Type | Displays the Application Execution Type as Function.
Function Name | Enter the Function Name.
### Field Name | Description
--- | ---
**Input Parameters** | Select the list of Data Fields which will be passed as input parameters, from the drop-down list.

**Parameter Mode** | Enter the Parameter Mode in JSON format. For example, suppose you have given 3 parameters as input parameters, enter parameter mode as ["IN","IN","OUT"].
During Execution of Function,
- In case of Decision Rule type, the first return parameter should return value ‘PASS’ for success evaluation.
- In case of Selection Rule type, the first return parameter value is taken as Selection data.
- In case of Execution Rule, the procedure return OUT parameter value overwrites the current value of respective mapped Data Field.

**Return Parameter** | Select the Data Field which will receive the return parameter of the Java function, from the drop-down list.
- For ExecutionRule type, the business logic is implemented in the method and the parameter value returned from the Java method is saved in the mapped Data Field.
- In case of SelectionRule type, the Java method should be a String value.
- In case of DecisionRule type, the Java method should return Boolean values “True/False”.

**Scope** | Select the **Scope** as **Process** to use the Application Rule only in the current process or **Package** to use the Application Rule across all the processes in the package.

### 4.4.4 For Java Application Rule:

This Application Rule is used to call Java functions in your Process flow.
### Application Rules

#### Field Name

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the Application Rule.</td>
</tr>
<tr>
<td>Rule Type</td>
<td>This Rule Execution type supports only ExecutionRule type.</td>
</tr>
<tr>
<td>Execution Type</td>
<td>Displays the Application Execution Type as JAVA.</td>
</tr>
</tbody>
</table>
| Implementation Detail | Enter the complete java class name which implements the Interface:  
                        com.ofs.aai.service.wf.external.base.Activity.  
                        The implementation class has to override the method with the Business Logic.  
                        executeTask(List<Data Field> Data Fields)  
                        The Data Fields are passed by reference, so changes can be made in Data Fields  
                        value directly by the implementation class, which will be recognized by the WorkflowEngine. |
| Return Parameter | Select the Data Field which will receive the return parameter of the Java function,  
                        from the drop-down list.  
                        • For ExecutionRule type, the business logic is implemented in the method and the parameter value returned from the Java method is saved in the mapped Data Field.  
                        • In case of SelectionRule type, the Java method should be a String value.  
                        • In case of DecisionRule type, the Java method should return Boolean values “True/False”. |
The class and its dependent file (or jar) needs to be available in webcontainer class path. For example, `<TOMCAT_HOME>/webapps/<context>/WEB-INF/lib/<forecast.jar>`

### 4.4.5 For Java External API Application Rule:

This Application Rule is used to call Java External API in your process flow. You need to specify the Class Name and the method of the API.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>Select the <strong>Scope</strong> as <strong>Process</strong> to use the Application Rule only in the current process or <strong>Package</strong> to use the Application Rule across all the processes in the package.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the Application Rule.</td>
</tr>
<tr>
<td>Rule Type</td>
<td>Select the <strong>Rule Type</strong> from the drop-down list. The Java External API Application Rule can be used as Decision Rule, Execution Rule or Selection Rule based on your requirement.</td>
</tr>
<tr>
<td>Execution Type</td>
<td>Java External API</td>
</tr>
<tr>
<td>Class Name</td>
<td>comForecast</td>
</tr>
<tr>
<td>Method</td>
<td>forecastPD</td>
</tr>
<tr>
<td>InputParameters</td>
<td>INFODOM_CODE</td>
</tr>
<tr>
<td>Return Parameter</td>
<td>STATUS</td>
</tr>
<tr>
<td>Scope</td>
<td>Process</td>
</tr>
</tbody>
</table>

![Rule Details](image_url)
### Application Rules

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution Type</td>
<td>Displays the Application Execution Type as JAVA External API.</td>
</tr>
<tr>
<td>Class Name</td>
<td>Enter the complete java class name which implements the Business Logic.</td>
</tr>
<tr>
<td>Method</td>
<td>Enter the method which you want to execute.</td>
</tr>
<tr>
<td>Input Parameters</td>
<td>You can pass Input Parameters for the method using Data Fields. Select the</td>
</tr>
<tr>
<td></td>
<td>required Data Fields from the drop-down list.</td>
</tr>
<tr>
<td>Return Parameters</td>
<td>Select the Data Field which will store the Return Parameter of the method,</td>
</tr>
<tr>
<td></td>
<td>from the drop-down list.</td>
</tr>
<tr>
<td>Scope</td>
<td>Select the <strong>Scope</strong> as <strong>Process</strong> to use the Application Rule only in the</td>
</tr>
<tr>
<td></td>
<td>current process or <strong>Package</strong> to use the Application Rule across all the</td>
</tr>
<tr>
<td></td>
<td>processes in the package.</td>
</tr>
</tbody>
</table>

### NOTE

1. The class and its dependent file (or jar) needs to be available in webcontainer class path. For example, `<TOMCAT_HOME>/webapps/<context>/WEB-INF/lib/<forecast.jar>`

2. For details on the supported API's for using as Java External APIs, see Appendix B.
4.4.6 For Outcome Rules Application Rule

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the Application Rule.</td>
</tr>
<tr>
<td>Rule Type</td>
<td>Displays the rule type as DecisionRule. This Rule Execution type supports only DecisionRule type.</td>
</tr>
<tr>
<td>Execution Type</td>
<td>Displays the Application Execution Type as Outcome.</td>
</tr>
<tr>
<td>Outcomes</td>
<td>Select the outcome for which you want to add the rule.</td>
</tr>
<tr>
<td>Scope</td>
<td>Select the <strong>Scope</strong> as <strong>Process</strong> to use the Application Rule only in the current process or <strong>Package</strong> to use the Application Rule across all the processes in the package.</td>
</tr>
</tbody>
</table>

4.4.7 Expression Application Rule

This is same as SQL execution type. You need to specify only the where clause in the Expression field. It can be any SQL expressions including ‘AND’/ ‘OR’.
### Field Name | Description
--- | ---
Name | Enter a unique name for the Application Rule.
Rule Type | Select the **Rule Type** from the drop-down list. The Expression Application Rule can be used as Decision Rule, Execution Rule or Selection Rule based on your requirement.
Execution Type | Displays the Application Execution Type as Expression.
Expression Type | Select Expression Type as SQL to use SQL expressions or JSON to use JSON expressions.
Expression | Enter the expression in SQL format or JSON format.
Return Parameter | Select the Data Field which will receive the return parameter of the Expression, from the drop-down list.
- For ExecutionRule type, the business logic is implemented in the method and the parameter value returned from the Application Rule is saved in the mapped Data Field.
- In case of SelectionRule type, the Application Rule should be a String value.
- In case of DecisionRule type, the Application Rule should return Boolean values “True/False”.
Scope | Process

![Rule Details](image-url)
4.4.8 **Rest Service Application Rule**

This Application Rule is used to call external components in your Process flow.

### Edit API Details

- **Name**: RestServiceCell
- **RuleType**: ExecutionRule
- **ExecutionType**: REST Service
- **Method Type**: POST
- **Url**: http://host:1350/fsdf806/rest-api/v1/PMFService/startWorkflowProcess
- **Query Param**: 
- **Headers**: 

```json
{"Content-Type":"*/*","Authorization":"*/*"}
```
- **Data**: 

```json
{"SummaryPayload":{"objectid":"Rest12","objecttype":"MOCK","infodom":"OFSSBRCINFO","segment":"","userid":"OFSAID","locale":"en_US","securitymap":{},"applicationparams":{}}}
```
- **Return Parameter**: REST_RETURN
- **Scope**: Process
- **Is Proxy Required**: Yes

### Field Name | Description
--- | ---
Name | Enter a unique name for the Application Rule.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule Type</td>
<td>Select the <strong>Rule Type</strong> from the drop-down list. This Application Rule can be used as Decision Rule, Execution Rule or Selection Rule based on your requirement.</td>
</tr>
<tr>
<td>Execution Type</td>
<td>Displays the Application Execution Type as Rest service.</td>
</tr>
<tr>
<td>Method Type</td>
<td>Select the method type from the drop-down list. The options are <strong>GET</strong> and <strong>POST</strong>.</td>
</tr>
<tr>
<td>URL</td>
<td>Enter the REST URL that needs to be called. For example, <code>&lt;IP Address/hostname of the Web Server :&lt;servlet port&gt;/&lt;context name&gt;/restPMF/PMFService/startWorkflowProcess</code> (A rest URL to start the workflow).</td>
</tr>
<tr>
<td>Query Param</td>
<td>Enter the <strong>Query Parameters</strong> that needs to be passed to the rest API. For example, <code>http://example.com/foo?bar</code></td>
</tr>
<tr>
<td>Headers</td>
<td>Enter any headers that needs to be passed to the rest API. For example, &quot;Content-Type&quot;: &quot;application/json&quot;</td>
</tr>
<tr>
<td></td>
<td>To pass the header values dynamically, use the following format: <code>Content-Type: TYPE, Authorization: CREDENTIAL</code> Where TYPE and CREDENTIAL are data fields.</td>
</tr>
<tr>
<td>Data</td>
<td>Enter if any actual data that needs to be passed to the rest API. Data can be of type RAW, JSON, Form Data and so on. For example, <code>{&quot;objectid&quot;:&quot;912&quot;,&quot;objecttype&quot;:&quot;1000&quot;,&quot;infodom&quot;:&quot;OFSCAPADQINFO&quot;,&quot;segment&quot;:&quot;OFSCAPADQINFO&quot;,&quot;userid&quot;:&quot;ORMUSER&quot;,&quot;locale&quot;:&quot;en_US&quot;,&quot;securityMap&quot;:{},&quot;applicationParams&quot;:{&quot;testparam&quot;:&quot;value1&quot;,&quot;testparam2&quot;:&quot;value2&quot;}}</code></td>
</tr>
</tbody>
</table>
| Return Parameter | Select the Data Field which will receive the return parameter of the Expression, from the drop-down list.  
|                | • For ExecutionRule type, the business logic is implemented in the method and the parameter value returned from the Application Rule is saved in the mapped Data Field.  
|                | • In case of SelectionRule type, the Application Rule should be a String value.  
|                | • In case of DecisionRule type, the Application Rule should return Boolean values ‘True/False’.                                             |
| Scope         | Select the **Scope** as **Process** to use the Application Rule only in the current process or **Package** to use the Application Rule across all the processes in the package. |
| Is Proxy required | Select **Yes** if proxy is required for the Rest Service. That is, if the Rest API is outside OFSAA. For information on how to enable proxy, see Enabling Proxy for REST Service Application Rule section. |
4.4.8.1 Enabling Proxy for REST Service Application Rule

This section explains how to configure the Proxy details if it is required for the Rest Service Application Rule.

Add the following entries in the AAI_WF_GLOBAL_SETTINGS table:

<table>
<thead>
<tr>
<th>V_PARAM_NAME</th>
<th>V_PARAM_VALUE</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROXY_SERVER_IP</td>
<td>For example, <a href="http://www.proxy.myserver.com">www.proxy.myserver.com</a></td>
<td>Provide the IP address of the Proxy server.</td>
</tr>
<tr>
<td>PROXY_SERVER_PORT</td>
<td>For example, 80</td>
<td>Provide the port number of the Proxy server.</td>
</tr>
</tbody>
</table>

4.4.9 For Attribute Expression Application Rule

![Attribute Builder](image)

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the Application Rule.</td>
</tr>
<tr>
<td>Rule Type</td>
<td>Displays the rule type as DecisionRule. This Rule Execution type supports only DecisionRule type.</td>
</tr>
<tr>
<td>Field Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Execution Type</td>
<td>Displays the Application Execution Type as Attribute Expression.</td>
</tr>
<tr>
<td>Attribute</td>
<td>Select the attribute for which you want to define the application rule, from the drop-down list. The list displays the attributes configured for the selected application and component. For more information, see Configuring Application Object Model (AOM) section. Click Add to add values to the selected attributes. A row is added in the Attribute Values pane. Click the Value column to select the values for the attribute from the drop-down. You can select one or more values. You can delete a row by clicking button. You can select multiple attributes and click Add to assign values to those attributes.</td>
</tr>
</tbody>
</table>

### 4.4.10 For Advanced Attribute Expression Application Rule

This is an advanced version of Attribute Expression Application Rule with additional logical conditions and assignment operators. The expression can be dynamically built and will return True or False value after evaluation. This is used as a DecisionRule in transitions.

You can define this application rule with multiple conditions and nested groups.
<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rule Name</td>
<td>Enter a unique name for the Application Rule.</td>
</tr>
<tr>
<td>AND/ OR</td>
<td>Select the logical operator to be used for the conditions in a group.</td>
</tr>
<tr>
<td>Add Condition</td>
<td>When you click Add Condition, a new row gets added. To define a condition, select the attribute, the operator and the value from the drop-down lists. Multiple values can be selected for each attribute.</td>
</tr>
<tr>
<td></td>
<td>- Attribute- The drop-down list displays the attributes configured for the selected application and component. For more information, see Configuring Application Object Model (AOM) section.</td>
</tr>
<tr>
<td></td>
<td>- Operator- Available options are in,=,&lt;&gt;.,&lt;,&lt;=,&gt;,&gt;=.</td>
</tr>
<tr>
<td></td>
<td>- Value- Displays the values configured for the selected attributes. Select the required value.</td>
</tr>
<tr>
<td></td>
<td>Click Remove Condition to delete already added condition.</td>
</tr>
<tr>
<td>Add Group</td>
<td>Click Add Group if you want to have nested conditions. For each group, select the required logical operator as AND or OR.</td>
</tr>
<tr>
<td></td>
<td>Click Remove Group to delete a group of conditions.</td>
</tr>
</tbody>
</table>

### 4.4.11 For JSON Path Expression Application Rule

This Application rule is used to extract data from the JSON Path Expression, which gets returned from a Rest API or Web Service call, and you can store it into a Data Field for further processing.

![Rule Details](image)
### Field Name | Description
--- | ---
Name | Enter a unique name for the Application Rule.
Rule Type | Select the rule type from the drop-down list. The available rule types are Decision Rule and Execution Rule. For Decision Rule, output of JSON Path Expression will be compared with RHS expression and rule will return either true or false accordingly. For execution rule, JSON Path Expression is evaluated and the output is returned to the DataField selected as Output DataField.
Execution Type | Displays the Application Execution Type as JSON Path Expression.
JSON Input | Select the Data Field in which the output of Web Service is stored from the drop-down list. You should select a DataField which has JSON as its value.
JSON Path Expression | Enter the JSON path expression. For more information, see JsonPath Expressions section.
Operator | This field is displayed only if Rule Type is selected as Decision Rule. Select the required operator for comparison from the drop-down list. The options are =,<,>,>= and <=.
RHS Expression | This field is displayed only if Rule Type is selected as Decision Rule. Enter the expression to which you want to compare the JSON path expression.
Output DataField | This field is displayed only if Rule Type is selected as Execution Rule. Select the DataField to which you want to return the value of JSON Path Expression, from the drop-down list.
Scope | Select the scope of the Application Rule from the drop-down list. The options are:
  - Process- Select Process if you want to use the Application Rule only in the current process.
  - Package- Select Package if you want to use the Application Rule across all the processes in the Application package.
Convert To Type | Select JSON ARRAY to store the output in Array format or select String to store as a string, from the drop-down list.

### 4.4.12 JSON Read From DB Application Rule

This Application Rule is used to read data from the database in JSON format.
### Field Name

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Enter a unique name for the Application Rule.</td>
</tr>
<tr>
<td>Rule Type</td>
<td>Only Execution Rule type is supported.</td>
</tr>
<tr>
<td>Execution Type</td>
<td>Displays the Execution Type as JSON Read From DB.</td>
</tr>
<tr>
<td>Table Name</td>
<td>Enter the table name from which you want to read the data.</td>
</tr>
<tr>
<td>Column List</td>
<td>Enter the column names of the selected table.</td>
</tr>
<tr>
<td>Where Condition</td>
<td>Enter the filter condition (where clause) of the SQL query.</td>
</tr>
<tr>
<td>Return JSON Type</td>
<td>Select the JSON type of the returned value as JSON Object or JSON Array based on your requirement.</td>
</tr>
<tr>
<td></td>
<td>• JSON Object- Select this option if the returned value is a single row.</td>
</tr>
<tr>
<td></td>
<td>• JSON Array- Select this option if the returned data has multiple rows.</td>
</tr>
</tbody>
</table>
### Field Name | Description
--- | ---
Output DataField | Select the DataField to which you want to return the value of the SQL query, from the drop-down list.
Scope | Select the scope of the Application Rule from the drop-down list. The options are:
- Process- Select Process if you want to use the Application Rule only in the current process.
- Package- Select Package if you want to use the Application Rule across all the processes in the Application package.

### 4.4.13 JSON Write To DB Application Rule
This Application Rule is used to write the data in JSON format into the database.

**Field Name | Description**
--- | ---
Name | Enter a unique name for the Application Rule.
Rule Type | Only Execution Rule type is supported.
Execution Type | Displays the Execution Type as JSON Write To DB.
Table Name | Enter the table name to which you want to write the data in JSON format.
4.5 Configuring Application Object Model (AOM)

This module helps in creating set of attributes for a given application in an abstract way, so that frameworks like PMF and other modules can leverage to retrieve application attributes and its values.

Each application is identified using an application package ID. For configuring package IDs, see Application Package section.

Against each package id, the set of attributes needs to be seeded in the “AAI_AOM_APP_COMP_ATTR_MAPPING” table.

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source JSON</td>
<td>Enter the data in the JSON format that you want to write to the database.</td>
</tr>
<tr>
<td>Output DataField</td>
<td>This field is not applicable.</td>
</tr>
<tr>
<td>Scope</td>
<td>Select the scope of the Application Rule from the drop-down list. The options are:</td>
</tr>
<tr>
<td></td>
<td>• Process- Select Process if you want to use the Application Rule only in the current process.</td>
</tr>
<tr>
<td></td>
<td>• Package- Select Package if you want to use the Application Rule across all the processes in the Application package.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>APP_COMP_ATTR_MAP_ID</td>
<td>Enter a unique ID for the attribute. You need to enter Attribute name and description for each attribute ID entered here in the AAI_AOM_APP_COMP_ATTR_TL table. See AAI_AOM_APP_COMP_ATTR_TL Table section.</td>
</tr>
<tr>
<td>V_ATTR_CODE</td>
<td>Name of the attribute.</td>
</tr>
<tr>
<td>N_ATTR_TYPE_ID</td>
<td>ID of the attribute type. The values of the attributes are fetched based on attribute type.</td>
</tr>
<tr>
<td></td>
<td>1001- Static</td>
</tr>
<tr>
<td></td>
<td>1002- SQL Query</td>
</tr>
<tr>
<td></td>
<td>1003- JavaAPI</td>
</tr>
<tr>
<td></td>
<td>1004- Hierarchy</td>
</tr>
<tr>
<td></td>
<td>1005- Multi Select Hierarchy</td>
</tr>
<tr>
<td></td>
<td>For more information, see Attribute Types.</td>
</tr>
<tr>
<td>V_ATTRIBUTE_VALUE1</td>
<td>Values to be fetched for the attribute. Based on the attribute type, you need to pass the values.</td>
</tr>
<tr>
<td>V_ATTRIBUTE_VALUE2</td>
<td></td>
</tr>
<tr>
<td>N_APP_ID</td>
<td>Application code for which the current attribute is configured. For example, if you are configuring Run execution parameters for IFRS application, enter the application ID of IFRS here.</td>
</tr>
<tr>
<td>N_COMP_ID</td>
<td>Component code for which the attribute is configured.</td>
</tr>
</tbody>
</table>
### 4.5.1 Attribute Types

The values of attributes are fetched based on the attribute types. Following are the attribute types with their IDs:

- **1001 (Static)** - Store attribute values in the `AAI_AOM_STATIC` table as `V_STATIC_ID` and `V_STATIC_VAL`.
- **1002 (Query)** - Enter the SQL query in `V_ATTRIBUTE_VALUE1` in the `AAI_AOM_APP_COMP_ATTR_MAPPING` table, which has to be fired to fetch the attribute values.
- **1003 (JavaAPI)** – Enter the method that is configured for `V_ATTRIBUTE_VALUE1` for the required attribute. The configured method in the class path is invoked to get the attribute values in this case.
- **1004 (Hierarchy)** – Specify the Hierarchy code to be fetched in `V_ATTRIBUTE_VALUE1` in the `AAI_AOM_APP_COMP_ATTR_MAPPING` table.
- **1005 (Multi Select Hierarchy)** – Specify the Multi Select Hierarchy entries in the `AAI_AOM_APP_COMP_ATTR_TL` table.

### 4.5.2 AAI_AOM_APP_COMP_ATTR_TL Table

In this table, for each `APP_COMP_ATTR_MAP_ID` table, enter the locale specific Attribute Name, Description as shown:

<table>
<thead>
<tr>
<th>APP_COMP_ATTR_MAP_ID</th>
<th>V_ATTR_NAME</th>
<th>V_ATTR_DESC</th>
<th>V_LOCALE_CODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Legal Entity</td>
<td>Legal Entity</td>
<td>en_US</td>
</tr>
<tr>
<td>2</td>
<td>Consolidation Type</td>
<td>Consolidation Type</td>
<td>en_US</td>
</tr>
<tr>
<td>3</td>
<td>Reporting Currency</td>
<td>Reporting Currency</td>
<td>en_US</td>
</tr>
<tr>
<td>4</td>
<td>Bucket Conversion</td>
<td>Bucket Conversion</td>
<td>en_US</td>
</tr>
<tr>
<td>5</td>
<td>Interpolation Method</td>
<td>Interpolation Method</td>
<td>en_US</td>
</tr>
<tr>
<td>6</td>
<td>PD Interpolation Method</td>
<td>PD Interpolation Method</td>
<td>en_US</td>
</tr>
<tr>
<td>7</td>
<td>Source Run Id</td>
<td>Source Run Id</td>
<td>en_US</td>
</tr>
<tr>
<td>8</td>
<td>FIC MISO</td>
<td>FIC MISO</td>
<td>en_US</td>
</tr>
<tr>
<td>9</td>
<td>Run Execution Description</td>
<td>Run Execution Description</td>
<td>en_US</td>
</tr>
</tbody>
</table>
4.5.3 **Usage of AOM Attributes in Run Pipeline**

While executing Run pipeline, the Select Run Params window displays the AOM fields which are marked as mandatory.

For example, for the attributes stored in `APP_COMP_ATTR_MAP_ID` table as shown in the previous figure, the Select Run Params window will be displayed as shown:

[Select Run Params window image]
5 Designing a Pipeline

Business pipelines are defined in OFSAA to design and execute sequence of tasks which are either OFSAA tasks or external tasks, to derive a well-defined outcome. This flow is defined by using various OFSAA artifacts from the component toolbar.

Using Process Modeler, we can:

a) Orchestrate a Business pipeline
b) Orchestrate a Run pipeline using PMF modeling

5.1 Orchestration of a Business Pipeline

Business Pipeline is used to design a Business Process which consists of a sequence of tasks either internal or external tasks through well-defined interfaces. Using the designer, you can design the entire business flows consisting of various types of tasks or another business pipeline.

5.1.1 An Example of a Business Pipeline

In the example shown, we use various OFSAA widgets like Run, LoadT2T, RuleType3, TransformDT, MFModel and activities like Human Tasks and Service tasks, which are related to each other through transitions. For executing tasks in parallel, we used Parallel Gateways.

5.1.2 Creating a Business Pipeline

Click on in the Process Modeller Summary window.
1. Enter a unique Process ID, Process Name and a description.

2. Select the appropriate app package in which you want to create the process. For more information, see Application Package section.

3. Select Process Modelling from the Type drop-down list.

4. Select the information domain in which you want to create the Business Pipeline, from the Infodom drop-down list. The list displays all the infodoms mapped to the applications configured in your OFSAA instance.

5. Click Save & Launch. The Process Flow canvas is displayed.
6. Click \[\text{Start}\] from the \textit{Tools} pane in the floating toolbar and click on the drawing canvas where you want to place it. This \textit{Start} activity indicates the beginning of the \textit{Process}.

7. Design your \textit{Process} with various components available in the \textit{Process Flow} tab. For more information on each component, see \textit{Components for Designing Your Process Flow} section.

**NOTE**

### 5.2 Run Pipeline

Run Process is used to create a Run definition in Rule Run Framework (RRF) using PMF Process. Visual representation of the Run is enabled through PMF by construction of a Run Pipeline. Various OFSAA widgets that enables construction of Run Pipeline are available in the Component toolbar.
5.2.1 An Example of Run Pipeline

5.2.2 Creating a Run Pipeline

1. Seed AOM Data Fields. For more information, see Configuring Application Object Model (AOM) section.

2. Click in the Process Modeller Summary window.
3. Enter a unique Process ID, Process Name and a description.

4. Select the appropriate app package in which you want to create the process. For more information, see Application Package section.

5. Select Run from the Type drop-down list.

6. Select the information domain in which you want to create the Run Process, from the Infodom drop-down list. The list displays all the infodoms mapped to the applications configured in your OFSAA instance.

7. Click Save & Launch.

8. Click Start from the Tools pane in the floating toolbar and click on the drawing canvas where you want to place it. This Start activity indicates the beginning of the Process.

5.2.3 Design your Run Pipeline using OFSAA Widgets

You can construct Run pipeline using only OFSAA Widgets. For details, see Configuring OFSAA Tasks in Your Process Flow section.

5.2.4 Design your Run Pipeline using Sub Pipeline

You can construct Run Pipeline using already constructed pipelines. The same pipeline can be used across different Run pipelines. For more information, see Calling another Pipeline from Your Parent Pipeline section.
5.2.5 Design your Run Pipeline using Inline Process

You can construct Run Pipeline by grouping a set of tasks as an Inline Process. For more information, see Inline Process section.

5.2.6 Design your Run Pipeline using combinations of OFSAA Widgets, Sub Pipeline and Inline Process

You can construct Run pipeline by combining OFSAA widgets, Sub pipeline and Inline Process.

5.2.7 Executing Run Pipeline

You can execute Run Pipeline from UI or using a command line utility called wfExecExternal.sh.

5.2.7.1 From UI

To execute Run Pipeline from UI:

1. From the Process Modeller Summary window, click corresponding to the Run Pipeline you want to execute:

2. Click Execute Run to execute the Run Pipeline. The Select Run Params window is displayed.
3. Select values for the Run Parameters and click **OK**.

The execution of the Run Pipeline is triggered using the selected FIC MIS DATE. The RUNSKEY is generated and inserted into the "DIM_RUN" table. For the RUNSKEY generated, the corresponding user selected parameters are inserted in to the "RUN_EXE_PARAMETERS" table.

### 5.2.7.2 Using Command Line Utility

A command line utility ./wfExecExternal.sh is available in $FIC_DB_HOME/bin folder.

To execute Run Pipeline using command line utility

1. Navigate to $FIC_DB_HOME/bin folder.

2. Execute the script file using the following command:

   ```bash
   ./wfExecExternal.sh processInstanceId processId objectId objectType infodom segment locale 'applicationparams' 'securityparams'
   ```

   - **processInstanceId** - Instance Id of the Process or Run Pipeline
   - **processId** - Process ID of the Run Pipeline. This is a mandatory parameter.
   - **objectId** - Specify a unique Object ID. This is a mandatory parameter.
   - **objectType** - Specify the Object Type if it is defined in the aai_wf_app_definition_map table.
   - **infodom** - Information Domain Name
- **segment** - Segment Name
- **locale** - Locale selected. For example, en_US
- **'applicationparams'** - Specify values for the Run execution parameters stored in APP_COMP_ATTR_MAP_ID within single quotes separated by comma.
- **'securityparams'** - Specify any security parameters within single quotes separated by comma

For example:

```
./wfExecExternal.sh null 1542221676429 CECLShellLatest null OFSIFRSINFO OFSIFRSINFO AAAIUSER en_US
```

**TIP**
Null should be passed if you do not want to pass value for a parameter.

### 5.2.8 Abort Run Pipeline

The Abort feature facilitates you to abort a Run Pipeline which is in the process of execution.

To abort a Run Pipeline

1. From the Process Monitor window, click corresponding to the Run Pipeline you want to abort and click *Abort*.

2. Enter comments if any, for aborting the Run Pipeline execution.
5.2.9 Resume Run Pipeline

You can resume a Run Pipeline which has not been executed successfully or which has been explicitly interrupted, or cancelled, or put on hold during the execution process. By resuming a Run Pipeline, you can continue its execution directly from the point of interruption or failure and complete executing the remaining tasks.

To resume a Run Pipeline

1. From the Process Monitor window, click corresponding to the Run Pipeline you want to resume and click Resume.

2. Enter comments if any, for restarting the Run Pipeline execution.

5.2.10 Re-run Run Pipeline

You can re-run a Run Pipeline which has previously been executed, irrespective of the previous execution state.

To re-run a Run Pipeline

1. From the Process Monitor window, click corresponding to the Run Pipeline you want to re-run and click Re-Run.

2. Enter comments if any, for re-running the Run Pipeline execution.
6 Human Tasks

Human Task is used if an activity requires a human intervention to go to the next activity.

6.1 How to Use Human Task

1. From the Process Flow tab, click Human Task from the Activities pane in the floating toolbar and click on the drawing canvas where you want to place it.

2. Double-click icon. The Activity tab is displayed.
6.1.1 Activity Tab

1. **Activity ID** is auto populated.
2. Enter Activity Name and Description.
3. Select the **Status** of the activity from the drop-down list. The list displays the seeded values in the AAI_WF_STATUS_B table.

6.1.2 Action Tab for Creating Tasks/Notification

Action/Task is used to inform the assigned user about an action to be completed in the current stage of Workflow. You can add multiple tasks for an activity. Task can be assigned to a user, user group, users with a particular role, or users selected by an application rule. Task is sent to the Application Inbox of the assigned users.

1. Click 💾 to define Actions.
2. Click Add from the Action tab to add a new Task.

6.1.2.1 Defining Task Details
1. Task ID is automatically generated.
2. Enter a Task Name. By default, task name is prefixed with “TaskName_”.
3. Select the decision rule so that when the condition is satisfied the Task will be sent to the users associated to this task.
4. Select the required option for Generate:
   - On Entry- Task is sent before executing the activity business logic.
   - On Exit- Task is sent before leaving the activity and moving to next activity.
   - On Stage- When there is a transition where the target and source is the same activity, and if that transition happens, then OnStage tasks are triggered.
5. Select the Yes checkbox to enable the Task.

### 6.1.2.2 Setting Email Notification

You should have configured Email settings before you set email notification for your tasks. For more details on how to configure email settings, see Configuring Email for Human Tasks section.

1. Select the Yes checkbox if an email needs to be sent for the task.
2. Select the email template to be used from the drop-down list. Templates are populated from AAI_WF_Email_Template table.
6.2 Additional Functionalities for Human Tasks

6.2.1 Setting Task Expiry

1. Select the required option from the **Expiry** drop-down list:
   - Global Expiry Setting- Select this option to set the task expiry based on global setting. Global setting can be set from backend.
   - Never Expire- Select this option if the task should not expire.
   - Expire After- Select this option if you want to set the task expiry after some days and/or hours.
   - Dynamic Value – Select this option if you want the user assigned to the task to set the task expiry date and time dynamically. This value needs to be entered in the code.

2. If **Expiry** is selected as **Expire After**, enter the number of days and/or hours after which the task should be expired.
6.2.2 Setting Task Escalation

1. Select the Escalation criteria:
   - Global Escalation Setting - Select this option to set the task escalation based on global setting. Global setting can be set from backend.
   - Never Escalate - Select this option if escalation is not required for the task.
   - Escalate After - Select this option if you want to escalate if the task is not addressed after some days and/or hours.

[NOTE]
Escalation should be set after expiry of the task. If you have selected Never Expire option for Expiry, you cannot set escalation.

2. If Escalate After is selected for Escalation, enter the number of days and/or hours after which the escalation should be triggered.

3. Enter the maximum number of escalation level. 1 indicates escalation to the immediate manager. 2 indicates escalation to manager’s manager and so on.

4. Enter the custom escalation Java Class which you want to call.

5. Select the escalation path from the drop-down list. The options are Default, People Hierarchy and Custom Rule.

6. Select the type of notification message from the drop-down list. Notification messages are populated from AAI_WF_Templates table.
6.2.3 Setting Reminder for Your Task

Reminders will be sent to the assigned user as an open task in their inbox.

1. Enter the number of times you want to set the reminder from the Recurrence field.

2. Select Task Start Date if you want to send reminder after the defined number of days and/or hours, from the start date of the task. Select Task Expiration Date if you want to send reminder before the defined number of days and/or hours from the end date/expiry date of the task.

3. Select the number of days and/or hours from the Relative Date after/before which you want to set the reminder.

4. Select the Notification Message you want to send, from the drop-down list. Notification messages are populated from AAI_WF_Templates table.

5. Click OK.
7 Service Tasks

Service Task is an automatic task which gets triggered in the Process flow. It is used to execute the Business Logic which is defined through Application Rule of Execution Rule type. For more information on configuring Application Rule, see Application Rule section.

Service Tasks are typically used to invoke External Model Service through Rest API, External Java APIs, Stored Procedures, and Functions.

More detailed explanation of invoking external model service is available in How to invoke External Model through Web Service section.

---

7.1 How to Use a Service Task

1. From the Process Flow tab, click Service Task from the Activities pane in the floating toolbar and click on the drawing canvas where you want to place it.

2. Double-click icon. The Activity tab is displayed.

7.1.1 Activity Tab

1. Activity ID is auto populated.

2. Enter Activity Name and Activity Description.
7.1.2 Implementation Tab

1. Click 🔄. The Implementation tab is displayed.

2. Select the information domain where the Execution Rule you want to execute is available, from the Infodom drop-down list. The list displays all the Infodoms mapped to the applications configured in your OFSAA instance.

3. Select the Execution Rule which needs to be executed for this activity. For more information on how to define an Application Rule, see Application Rules section.
   - Click 🔄. The Participant Details window is displayed with all Application Rules of Execution Rule types available in your Process.
   - Click the Name link of the Application Rule to view the details.
   - Select the required Rule and click Ok.

4. Add Parameters you want to pass to the Execution Rule using Data Fields. You can pass Static values or Dynamic Values. In case of Dynamic, the value needs to be entered during execution of the workflow.
   - Click 🔄 Add under Parameters. The Binding Details window is displayed.
5. Select an application rule which you want to execute before executing the Execution Rule.
   - Click corresponding to Pre Rule and select the required application rule.

6. Select an application rule which you want to execute after executing the Execution Rule.
   - Click corresponding to Post Rule and select the required application rule.
8 Configuring OFSAA Tasks in Your Process Flow

OFSAA widgets are used to orchestrate OFSAA components like T2T definitions, PLC definitions (DT), Rules (Classification Rule and Computation Rule), Models in EMF, Run, and Data Quality Groups into your Process Flow using Process Modeller.

8.1 How to Use an OFSAA Widget

1. From the Process Flow tab, click the required widget under OFSAA Widget toolbar in the left pane and click the canvas. The available OFSAA Widgets are LoadT2T, TransformDT, RuleType2, RuleType3, MFModel, Run, and DataQualityGroups.

2. Double-click the widget. On the Right Pane, the Activity tab is displayed.

3. Enter the Activity Name and Activity Description.

4. Based on the OFSAA widget, the Dynamic Parameters are displayed. For more information, see Dynamic Parameters for OFSAA Widgets section.

5. To view the definition of the OFSAA component you are using, mouse over its icon and click View. The Definition window of the OFSAA component is displayed.
8.1.1 Dynamic Parameters for OFSAA Widgets

8.1.1.1 LoadT2T

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datastore Name</td>
<td>Select the Information Domain in which the Data Mapping you want to execute is present, from the drop-down list.</td>
</tr>
<tr>
<td>Load Mode</td>
<td>Table to Table should be selected for Data Mapping definitions such as T2T, T2H, H2T, H2H and L2H definitions.</td>
</tr>
<tr>
<td>Source Name</td>
<td>Select the required source on which the Data Mapping definition you want to execute is defined, from the drop-down list.</td>
</tr>
<tr>
<td>File Name</td>
<td>Select the Data Mapping definition you want to execute, from the drop-down list. Based on the selected Load Mode and Source Name, the list displays the corresponding definitions.</td>
</tr>
<tr>
<td>Data File Address</td>
<td>This field is not applicable to Load Mode selected as Table to Table.</td>
</tr>
<tr>
<td>Default Value</td>
<td>Select the Data Fields you want to pass as parameters for the selected Data Mapping definition. For information on creating Data Fields, see Data Fields section. For additional information, see Task Component Parameters section in OFS Analytical Applications Infrastructure User Guide.</td>
</tr>
</tbody>
</table>
### 8.1.1.2 TransformDT

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datastore Name</td>
<td>Select the Information Domain in which the PLC you want to execute is present, from the drop-down list.</td>
</tr>
<tr>
<td>Rule Name</td>
<td>Select the Post Load Changes (DT) definition you want to execute from the drop-down list. The list displays the Post Load Changes definitions in the selected Information Domain.</td>
</tr>
</tbody>
</table>
| Parameter List   | Select the Data Fields you want to pass as parameters for the selected Data Mapping definition.                                              
|                  | For information on creating Data Fields, see Data Fields section.                                                                            |

### 8.1.1.3 RuleType2 or RuleType3

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datastore Name</td>
<td>Select the Information Domain in which the RRF Rule you want to execute is present, from the drop-down list.</td>
</tr>
</tbody>
</table>
8.1.1.4 MFModel

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datastore Name</td>
<td>Select the Information Domain in which the RRF Rule you want to execute is present, from the drop-down list.</td>
</tr>
<tr>
<td>Model Code</td>
<td>Display the codes of the EMF Models defined under the selected Infodom.</td>
</tr>
</tbody>
</table>

**Rule Code**
- Display the codes of the RRF Rules defined under the selected Infodom.
- Select the required Rule from the drop-down list.

**Build Flag**
- Select the required option from the drop-down list as “Yes” or “No”.
- Build Flag refers to the pre-compiled rules, which are executed with the query stored in database. While defining a Rule, you can make use of Build Flag to fasten the Rule execution process by making use of existing technical metadata details wherein the rule query is not rebuilt again during Rule execution.
- Built Flag status set to “No” indicates that the query statement is formed dynamically retrieving the technical metadata details. If the Build Flag status is set to “Yes” then the relevant metadata details required to form the rule query is stored in database on “Save” of a Rule definition. When this rule is executed, database is accessed to form the rule query based on stored metadata details, thus ensuring performance enhancement during Rule execution. For more information, see Significance of Pre-Built Flag section in OFS Analytical Applications Infrastructure User Guide.

**Optional Parameters**
- Select the Data Fields you want to pass as parameters for the selected Data Mapping definition.
- For information on creating Data Fields, see Data Fields section.
### Field Name | Description
--- | ---
**Operation** | The All definition for the Operation field conveys the process of extracting the data from the flat files and applying the run regression on the data extracted. For Batches that are being built for the first time the data will be extracted from the flat files and the run regression will be applied on it.

**Optional Parameters** | Select the Data Fields you want to pass as parameters for the selected Data Mapping definition. For information on creating Data Fields, see [Data Fields](#) section.

### Run

#### Dynamic Parameters for RUN

**Infodom** | OFSIFRSINFO
--- | ---
**Run Type** | Base Run
**Execution Rule** | IFRS9 - EIR Adjustment Run
**Run Parameters** | INFCDOM_CODE

#### Field Name | Description
--- | ---
**Infodom** | Select the information domain from the drop-down list.
**Run Type** | Select Base Run or Simulation Run based on the type of the Run you want to execute, from the drop-down list.
**Execution Rule** | Select the Run you want to execute from the drop-down list.
**Run Parameters** | Select the Data Fields you want to pass as parameters for the selected Data Mapping definition. For information on creating Data Fields, see [Data Fields](#) section.
### 8.1.1.6 DataQualityGroups

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DQ Group Name</td>
<td>Refers to the Data Quality Groups consisting of associated Data Quality Rule definition(s). Select the required DQ Group from the drop-down list.</td>
</tr>
<tr>
<td>Rejection Threshold</td>
<td>Specify the percentage of Rejection Threshold (%) limit in numeric value. This refers to the maximum percentage of records that can be rejected in a job. If the percentage of failed records exceeds the Rejection Threshold, the job will fail. If the field is left blank, the default value is set to 100%.</td>
</tr>
</tbody>
</table>
| Additional Parameters     | Specify the Additional Parameters as filtering criteria for execution in the pattern Key#Data type#Value; Key#Data type#Value;...etc. Here the Data type of the value should be “V” for Varchar/Char, or “D” for Date with “MM/DD/YYYY” format, or “N” for numeric data. For example, if you want to filter some specific region codes, you can specify the Additional Parameters value as $REGION_CODE#V#US;$CREATION_DATE#D#07/06/1983;$ACCOUNT_BAL#N#10000.50;  
**Note:** In case the Additional Parameters are not specified, the default value is fetched from the corresponding table in configuration schema for execution. |
### Property | Description
---|---
Fail if Threshold Breaches | Select Yes or No from the drop-down list. If Yes is selected, execution of the task fails if the threshold value is breached. If No is selected, execution of the task continues. Note: For Custom Check type DQ Rules in Hive Infodoms, the execution of the task will not fail even if the threshold is breached. This is a limitation.
Parameters | Select the Data Fields you want to pass as parameters for the selected Data Mapping definition. For information on creating Data Fields, see [Data Fields](#) section.
9 Orchestrating External Models/Components in Your Process Flow

You can use external models or external components in your process flow by using Rest Service Application Rule or External Java API Application Rule.

9.1 How to invoke External Model through Web Service

A typical External Web Service invocation will have three steps:

1. Data Preparation
2. Web Service Invocation
3. Data Extraction

9.1.1 Data Preparation

Before invoking Web Service, we need to prepare data to be passed across to the Web Service. This can be done by configuring Application Rule.

In the above example, data preparation is done through JSON Read From DB Application Rule as shown below, which reads data from a table, converts into JSON and stores output to JSON_DB_DATA Data Field.
9.1.2 Webservice Invocation

Consider you want to use an external model called ForecastModelPost, which is available in http://<IP Address/hostname of the Web Server>:<servlet port>/<context name>/rest-api/v1/ModelWebService/ForecastModelPost.

1. Launch the Process for which you want to add an external model.

2. Go to Application Rule tab and click Add > Rest Service.
3. Enter the details as shown. For more information on configuring a Rest Service, see Rest Service Application Rule section.

4. In this example, the Data sent to the webservice is a combination of both static and dynamic value (~~JSON_DB_DATA~~ is data field holding a JSON string which is prepared as previous step in the pipeline.)

5. Click Save. The ModelWebService Application Rule gets created in your logged in Information Domain.


7. Click Service Task from the Activities pane and click on the drawing canvas where you want to place it.

8. Double-click the Service Task icon.
9. Enter the Activity details such as Activity Name and Activity Description.

10. Click 📸. The Implementation tab is displayed.

11. Select the information domain where the ModelWebService Application Rule is defined, from the Infodom drop-down list. The list displays all the Infodoms mapped to the applications configured in your OFSAA instance.

12. Click 📧 in the Execution Rule field. The Participant Details window is displayed with all Application Rules of Execution Rule types available in your Process.

13. Select ModelWebService and click Ok.
9.1.3 Data Extraction

The response from the Web Service needs to be processed depending on the application requirement. PMF has capabilities to process the JSON and store in the output table.

You can use JSON PATH expressions to extract the relevant information from the Web Service response. For more information on JSON PATH expressions, see For JSON Path Expression Application Rule section.

Similarly, the response can be stored back to the database using JSON Write To DB Application Rule as shown:

![Edit API Details](image)

For more information, see JSON Write To DB Application Rule section.
10 Configuring Custom Components

For configuring a new custom component as an OFSAA Widget, you need to have entries in AAI_WF_COMPONENT_REGISTRATION table and AAI_WF_COMPONENT_PARAMETERS table. Additionally, executeComponent() method needs to be implemented in the Implementation class and the jar should be present in web-inf/lib path.

10.1 AAI_WF_COMPONENT_REGISTRATION Table

An entry into AAI_WF_COMPONENT_REGISTRATION table will create a new custom component in the PMF OFSAA widgets.

- **V_COMPONENT_ID** – Specify a unique ID for the component.
- **V_COMPONENT_EXE_IMPL** - This is the interface which needs to be implemented by the component owner, so that during the execution this component, this API is invoked.
- **V_COMPONENT_UI_LAUNCH** - This is the URL of the page which needs to be opened when the component is double clicked in the canvas.
- **V_COMPONENT_NAME** – Specify a name for the component, which will be displayed for the custom component in the OFSAA Widgets.
- **V_COMPONENT_ICONS** - All the icons and menu are configured in this field.

For example,
# 10.2 AAI_WF_COMPONENT_PARAMETERS Table

The AAI_WF_COMPONENT_PARAMETERS table needs to be populated with the relevant parameters:

<table>
<thead>
<tr>
<th>V_COMPONENT_ID</th>
<th>V_PARAMETER_NAME</th>
<th>I_PARAMETER_ORDER</th>
<th>I_DISPLAY_ORDER</th>
<th>V_PARAMETER_CODE</th>
<th>TYPE_OF_DISPLAY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IP Address</td>
<td>1</td>
<td>3</td>
<td>SMEDE_ADDR</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Database Type</td>
<td>2</td>
<td>1</td>
<td>IFC_PP_DATA_STORE_TYPE</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>Database Name</td>
<td>3</td>
<td>2</td>
<td>IFC_PP_DATA_STORE</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>Load Node</td>
<td>4</td>
<td>4</td>
<td>IFC_PP_LOADMODE</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Source Name</td>
<td>5</td>
<td>5</td>
<td>SMSC_SRC_NAME</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>File Name</td>
<td>6</td>
<td>6</td>
<td>GLOBAL_NAME</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>Data File Name</td>
<td>7</td>
<td>7</td>
<td>T2TROP_DEF_VALUE_DATA_FILE_NAME</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>Default Value</td>
<td>8</td>
<td>8</td>
<td>DTDD_DEFDATAVAL</td>
<td>1</td>
</tr>
</tbody>
</table>

- **V_COMPONENT_ID**: Specify the ID for the component. It should be the same as that given in the AAI_WF_COMPONENT_REGISTRATION table.
- **V_PARAMETER_NAME**: Specify the Parameter name which will be required by the component.
- **I_PARAMETER_ORDER**: Specify the Parameter order.
- **I_DISPLAY_ORDER**: Specify the order in which the Parameters need to be displayed in the UI.
- **V_PARAMETER_CODE**: Specify Parameter code.
- **TYPE_OF_DISPLAY**: Specify the type of display for the parameter. For example: if text input required the value should be 1 or selection drop down value should be 2.
- **V_DEFAULT_VALUE**: Specify if any default value needs to be provided for the parameter.
- **V_CLASS_NAME**: Specify the class name for listing the drop down values.
- **V_METHOD_NAME**: Specify the name of the method for listing values.
11 Executing Parallel Tasks

Parallel Gateway is used to execute multiple tasks in parallel. In the usual flow, tasks are executed sequentially.

In the shown example, after a task is executed, Parallel Gateway gets invoked. All the OFSAA components which are placed between Parallel Gateways are executed simultaneously. It waits till all components are executed, and then moves to the next activity in the Process Flow.
11.1 How to Use Parallel Gateways

This section explains how to design 3 tasks such as RuleType 2, MFModel and Run widget to be executed in parallel.

2. Launch the Process. It displays the Process Flow tab.
3. Click [Start] from the Tools pane in the floating toolbar and click on the drawing canvas where you want to place it. This Start activity indicates the beginning of the Process.
4. Click Parallel Gateway from the Tools pane in the floating toolbar and click on the drawing canvas where you want to place it.
5. Click Transition Curve and click Start (from where the transition starts) and click (the activity to which you want to have the transition).
6. Click RuleType2 from the OFSAA Widgets pane, and then click the drawing canvas where you want to place it.
7. Click Transition Curve from the Tools pane and click Parallel Gateway and then click.

8. Configure the RuleType2 widget. For more information, see Configuring OFSAA Tasks in Your Process Flow section.

9. Click MFModel from the OFSAA Widgets pane, and then click the drawing canvas where you want to place it.

10. Click Transition Curve from the Tools pane and click Parallel Gateway and then click.

11. Configure the MFModel widget. For more information, see Configuring OFSAA Tasks in Your Process Flow section.

12. Click Run from the OFSAA Widgets pane, and then click the drawing canvas where you want to place it.

13. Click Transition Curve from the Tools pane and click Parallel Gateway and then click.
14. Configure the Run widget. For more information, see Configuring OFSAA Tasks in Your Process Flow section.

15. Click Parallel Gateway from the Tools pane in the floating toolbar and click on the drawing canvas where you want to place it.

16. Draw transitions from RuleType2, MFModel and Run widgets to the merging Parallel Gateway as shown:

You can drag and position the widgets to avoid overlapping of widgets or transition lines.
12 Calling another Pipeline from Your Parent Pipeline

Reusability is important while designing your pipeline. Sub Pipeline is the mechanism in the PMF to call another pipeline from your parent pipeline.

12.1 How to Configure Sub Pipeline

1. From the Process Flow tab, click Sub Pipeline under Activities toolbar in the left pane and click the drawing canvas where you want to place it.

2. Double-click . The Sub Process Details window is displayed.
3. Enter the details as given in the table:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>App Package ID</td>
<td>Select the Application Package from which you want to call a process, from the drop-down list. The package IDs that are seeded from backend are displayed in the list.</td>
</tr>
<tr>
<td>Process ID</td>
<td>Select the Process which you want to call within your workflow, from the drop-down list. The list displays all processes defined for the selected Application Package.</td>
</tr>
<tr>
<td>Object ID</td>
<td>Select the Object ID from the drop-down list. Object ID is the entity ID used to identify if a workflow needs to be started from the beginning or from the current stage. Object ID drop-down list is populated from the Data Fields.</td>
</tr>
<tr>
<td>Object Type</td>
<td>Select the object type from the drop-down list. Object Type is used to identify a workflow that is passed by the application.</td>
</tr>
<tr>
<td>Data field pass</td>
<td>Select Yes to pass the parameters passed to the parent workflow to be passed to the selected sub pipeline.</td>
</tr>
<tr>
<td>Calling Mode</td>
<td>• Synchronous – Select this option if you want the sub pipeline to complete before the flow moves to the next activity.</td>
</tr>
<tr>
<td></td>
<td>• Asynchronous – Select this option if you do not want to wait for the sub pipeline to complete. And the parent workflow proceeds to the next activity.</td>
</tr>
</tbody>
</table>
13 Inline Process

Inline Process is mostly used for readability and usability. It is used to group a set of tasks as an Inline Process. This helps to represent a Process Flow with so many tasks /OFSA widgets in a systematic and organized way. You can have Inline Process inside another Inline Process.

For example, if an RRF Run definition has multiple sub processes (RRF), each sub process can be represented as an InLine Process.

13.1 An Example of Inline Process

Click icon. It opens up the process flow diagram of the IFRS_Rating_Classification_Proc as shown:
In this example, IFRS_Rating_Classification_Proc Inline Process has an Inline Process named Rating Reclassification. Double-clicking opens up its process flow as shown:
Multiple levels of Inline Process is supported. You can go to any level by clicking the Inline Process name displayed in the header. Click to go to the parent process flow.

13.2 How to Add an Inline Process

1. Launch the Process to which you want to add an Inline Process. It displays the Process Flow tab.

2. Click Start from the Tools pane in the floating toolbar and click on the drawing canvas where you want to place it. This Start activity indicates the beginning of the Process.

3. Expand Activities floating toolbar and click InLine Process.

4. Click the drawing canvas where you want to place it.

5. Click Transition Curve and click Start (from where the transition starts) and click (the activity to which you want to connect).

6. Mouse over the icon. The sub menu is displayed.

7. Click to modify the Inline Process name.

8. Click Ok.

10. Design the process flow using various components available in the Process Flow tab. For more information on each component, see Components for Designing Your Process Flow section.
14 Configuring Email for Human Tasks

Following table entries should be made for setting Email notification:

14.1 AAI_EMAIL_CONFIG Table

This table holds the SMTP server configurations.

- **V_PROTOCOL** - SMTP
- **V_HOST** - SMTP/ Mail Server ID
- **V_PORT** - SMTP Server Port
- **V_AUTHENTICATION** - Either False or True
- **V_USER_NAME** - Login name to SMTP/ Mail Server ID from which mail will be triggered. This is required if **V_AUTHENTICATION** is set as True.
- **V_PASSWORD** - Password to login into SMTP/ Mail Server. This is required if **V_AUTHENTICATION** is set as True.

<table>
<thead>
<tr>
<th>V_PROTOCOL</th>
<th>V_HOST</th>
<th>V_PORT</th>
<th>V_AUTHENTICATION</th>
<th>V_USER_NAME</th>
<th>V_PASSWORD</th>
<th>V_SECURITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>smtp</td>
<td>internal_email_router.oracle.com</td>
<td>25</td>
<td>false</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

14.2 AAI_WF_APP_PACKAGE_B Table

The AAI_WF_APP_PACKAGE_B table entry is for configuring email at Application Level.

- Set the **V_EMAIL_REQUIRED** parameter value to Y in AAI_WF_APP_PACKAGE_B table.
- Set **V_EMAIL_TYPE** as:
  - 0 – To receive no notification mails
  - 1 – To get mails instantly
  - 2 – To get bulk mail (Additionally, you need to set **V_BULK_MAIL_TRIGGER** value to Y in the AAI_WF_BULK_MAIL_TRIGGER table). A single mail will be sent with all the pending notifications from last trigger, as a PDF attachment. Once the bulk mail is sent, **V_BULK_MAIL_TRIGGER** value is automatically set to N.
  - 3 – To get mail with attachment

**V_BULK_TEMP** – This is used to set the template for the bulk emails. You can keep this as blank if bulk email is not set.
14.3 **AAI_WF_APP_REGISTRATION Table**

This table holds email configuration at module or entity type level.

Set the `V_IS_EMAIL_REQUIRED` parameter value to `Y` in **AAI_WF_APP_REGISTRATION** table.

![Table Example](image)

14.4 **AAI_WF_ACTIVITY_TASK_BTable**

The **AAI_WF_ACTIVITY_TASK_B** table holds email configuration at each activity (Task & notification) level.

Set the `V_EMAIL_REQUIRED` parameter value to `Y` against the task.

Additionally, you can set the `V_EMAIL_TEMPLATE` value based on **AAI_WF_EMAIL_TEMPLATE** table.

![Table Example](image)

14.5 **AAI_USER_PREFERENCE Table**

In this table, you can set the user preference of how to receive the notification mails.

<table>
<thead>
<tr>
<th>V_USER_ID</th>
<th>N_EMAIL_NOTIF_REQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER1</td>
<td>1</td>
</tr>
<tr>
<td>USER2</td>
<td>2</td>
</tr>
</tbody>
</table>

- 0 – To receive no notification mails
- 1 – To get mails instantly
- 2 – To get bulk mail (Additionally, you need to set `V_BULK_MAIL_TRIGGER` value to `Y` in the **AAI_WF_BULK_MAIL_TRIGGER** table). A single mail will be sent with all the pending notifications from last trigger, as a PDF attachment. Once the bulk mail is sent, `V_BULK_MAIL_TRIGGER` value is automatically set to `N`.
- 3 – To get mail with attachment
14.6 AAI_WF_EMAIL_TEMPLATE Table

This table is used to provide the template for the email which needs to be sent.

<table>
<thead>
<tr>
<th>V_MAIL_ID</th>
<th>V_MAIL_SUBJECT</th>
<th>V_APP_PACKAGE_ID</th>
<th>V_MAIL_TYPE</th>
<th>V_MAIL_TEMPLATE_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>This is to inform you that a TASK 'T' has been created</td>
<td>0</td>
<td>Notification</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>This is to inform you that a TASK 'T' has been created</td>
<td>0</td>
<td>Task</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>This is to inform you that a TASK 'T' has been created</td>
<td>0</td>
<td>Task</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>This is to inform you that a TASK 'T' has been created</td>
<td>0</td>
<td>Task</td>
<td>4</td>
</tr>
</tbody>
</table>

14.7 AAI_WF_BULK_MAIL_TRIGGER Table

If you have set N_EMAIL_NOTIF_REQ parameter to 2 in AAI_USER_PREFERENCE table, additionally you need to set V_BULK_MAIL_TRIGGER value to Y in the AAI_WF_BULK_MAIL_TRIGGER table.

14.8 CSSMS_USR_PROFILE Table

This table is used to store the email id of the users, to which the notification mails need to be sent.

<table>
<thead>
<tr>
<th>V_USR_ID</th>
<th>V_EMAIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>USER1</td>
<td><a href="mailto:user1@oracle.com">user1@oracle.com</a></td>
</tr>
<tr>
<td>USER2</td>
<td><a href="mailto:user2@oracle.com">user2@oracle.com</a></td>
</tr>
</tbody>
</table>

14.9 AAI_MAIL_AUDIT_TRAIL Table

This is where all mail trails are stored. The status changes once mail is sent. Can be used for debugging purposes.
15 Process Monitor

The Process Monitor is used to monitor current stage of the Process for different instances. After integration with an Application the workflow can be invoked. After invoking it goes through all the stages defined. Process Monitor shows all the stages finished, current stage and stages to come if any. Your user group should be mapped to the function role WFMACC (Workflow Monitor Access) to access Process Monitor window.

This window displays all the Workflows which are invoked from the Application with details such as Entity Name, Entity ID, Process Name, Process Description, Execution Start Time, Last Execution Time, Last Updated By and Status.

Click to display the following sub menu for Run Pipeline:

- **Abort** - to abort an ongoing Run Pipeline.
- **Resume** - to resume a Run Pipeline
- **Re-run** - to execute a Run Pipeline again irrespective of the previous execution status.
Using the Search grid, you can search for a specific Pipeline by providing a keyword from Process ID, Process Name or Process Description and clicking \( \text{search} \). Click \( \text{reset} \) to reset the Search fields.

You can sort the Processes displayed in the Process Monitor window based on Entity Id, Entity Name, Process ID or Process Name. Click \( \text{go to Process Modeller} \) to go to Process Modeller window.

15.1 Monitoring a Business Process

1. From the Process Monitor window, click the Entity ID link corresponding to the process you want to monitor.

In the Process Monitor window, status of activity is represented as given below:

- This indicates that the execution of activity is successful.
- This indicates that the activity is currently running or waiting for user’s input to proceed.
- This indicates that the execution of activity is failed.
- This indicates that the activity is yet to be executed.

Double click Inline Process icon \( \text{inline process} \) or Sub Pipeline icon \( \text{sub pipeline} \) to monitor the tasks inside them.
2. Click \( \text{Refresh} \) to refresh the pane. Click \( \text{View Logs} \) to view the Process Logs pane.

The Process Logs pane shows all the execution stages of the process. Click the Process Monitor pane to close the Process Logs pane.

You can use \( + \) and \( - \) to zoom in and out the Process Monitor window.

### 15.2 Viewing Activity Logs

This feature allows you to view logs of each activity from the Process Monitor window.

To view Activity logs

1. Double-click on the activity icon whose logs you want to view. The Activity Definition details are displayed.
2. Click to view the Activity Logs.

The log shows all the execution stages of the selected Activity, if it is already executed. Otherwise, it will be blank.
15.3 Viewing Execution Log for OFSAA Tasks

This option allows you to view the execution logs for OFSAA widgets. This option is not available for Service tasks or human tasks.

To view Execution logs

1. Double-click on the activity icon whose logs you want to view. The Activity Definition details are displayed.

2. Click Execution Log.

3. Select the required Log File from the drop-down list and click View Log. The log information will be displayed in the Log File Contents pane.

4. Click Download to download the execution log details.
16 Invoking PMF Pipeline

You can invoke a PMF pipeline using the following ways:

- Application UI
- Within PMF Summary Screen UI
- Operations Module
- Command Line Execution

16.1 Application UI

Application can invoke PMF in the following ways:

16.1.1 Java API

```java
WorkflowEngineAPI.startWorkflowProcess(String objectId, String objectType, String infodom, String segment, String userID, String locale, Map<String, String> applicationParams, Map<String, String> securityMap)
```

16.1.2 Stored Procedure

```sql
create or replace procedure startWorkflowProcessAsynch(objectId IN VARCHAR2,
objectType IN VARCHAR2,
infodom IN VARCHAR2,
segments IN VARCHAR2,
userID IN VARCHAR2,
locale IN VARCHAR2,
appParams IN array_varchar,
secMap IN array_varchar) is
```

16.1.3 Rest Service

```json
URL: <contextPath>/PMFService/startWorkflowProcess
Method: POST
Consumes("application/json")
Produces("text/plain")
Sample Input Params:

``{"SummaryPayload":{"objectid":"123","objecttype":"QTNR","infodom":"OFSAAAIINFO","segment":"OFSAAAIINFO","userid":"AAAIUSER","locale":"en_US","securitymap":{},"applicationparams":
```
16.2 Within PMF Summary Screen UI

16.2.1 Using Test Workflow

Testing Process Flow option is used to check whether the Process flow you designed works as expected.

1. From the Process Modeller window, click corresponding to the Pipeline you want to test. The Sub menu is displayed.

2. Click Test Process Flow. The Execute Process Flow window is displayed.

3. Enter an Object ID to identify the Process flow. This will be displayed as Entity ID in the Process Monitor window.

4. Enter the Application Parameters which are required in the Process flow that you are testing. Specify parameters in JSON format. This is an optional field.
   
   For example, {"WF_RUNSK":"15","WF_MISDATE":"12/31/1999"}

5. Click OK.

16.2.2 Using Execute Run

See Executing Run Pipeline section on how to invoke a Run pipeline from the PMF screen.

16.3 Operations Module

Execution of a PMF Pipeline can be triggered from the Operations module as a batch.
1. From the *Batch Maintenance* window under Operations module, create a new Batch. For more information, see *Adding Batch Definition* section in the *OFS Analytical Applications Infrastructure User Guide*.

2. Create a new Task with task component as Workflow Execution. For more information, see *Adding Task Details* section in the *OFS Analytical Applications Infrastructure User Guide*.

   a. Select the PMF Pipeline you want to execute from the *Workflow* drop-down list.
   
   b. Enter any parameters you want to pass during the execution of the Pipeline in the *Optional Parameters* field.
   
   c. Click *Save*.

3. Execute the Batch from the *Batch Execution* window. For more information, see *Batch Execution* section in the *OFS Analytical Applications Infrastructure User Guide*.

### 16.4 Command Line Execution

A shell script file `wfExec.sh` is available in `ficdb/bin` folder.

To execute the utility, navigate to `$FIC_DB_HOME/bin` and execute `wfExec.sh` with parameters such as `objectId`, `objectType`, `Infodom`, `segments`, `userId`, `locale`, `appParams`, and `secMap`. 
17 Appendix A

17.1 Configuring Group Approval for Human Tasks

Group approval/Group Consensus can be used to decide whether the flow has to move to a particular activity based on the response of a single member, majority of members of the group, or all members of the group. If more than one group is present, then you can design to move the flow to a particular activity based on the response of a single group, all groups or the majority of groups.

17.1.1 Configuring Parallel Group Approval

Parallel group approval is used when you want to send the task to all users in the task group simultaneously.

To configure parallel group approval:

1. From the Process Modeller window, select the required Process and click Edit. The Process Flow tab is displayed.
2. Double-click the Activity for which you want to configure user approval.
3. On the right pane, click 🗼. The Actions tab is displayed.
4. Click Group Approval to configure group approval. The Task Stage Details window is displayed.

![Task Stage Details Window]

5. Select Parallel to configure parallel group approval.
6. Click Add. A row is added to define the voting formula for target activity.
NOTE

It is recommended that you define voting formula for all activities. If voting formula is not defined for an activity and if someone in a task group selects that activity, the workflow moves to that activity.

7. Select the required Target Activity from the drop-down list.
8. Select the required option from the Voting Formula drop-down list. The options are:
   - Approved by anyone- If any one of the users from any task group chooses the selected Target Activity, the flow moves to the selected activity. If no one chose it, it checks for the voting formula defined for the next Target Activity.
   - Approved by anyone from every Group- If at least one user from every task group chooses the selected Target Activity, the flow moves to that activity.
   - Approved by overall majority- If majority of the users from all task groups choose the selected Target Activity, the flow moves to that activity. For example, if there are 2 task groups and 15 users in each group, then at least 16 users (majority of 30 users) should choose the selected Target Activity, for the flow to proceed to that activity.
   - Approved by majority from each Group- If majority of the users from each task group choose the selected Target Activity, the flow moves to that activity. For example, consider there are 3 task groups and each group has 15 users, then from each group at least 8 users should favour the Target Activity to move the flow to that activity.
   - Approved by everyone- All the users in all the task group should choose the selected Target Activity for the flow to move to that activity.
   - Approved by overall percentage- If the specified percentage of users in the task group chooses the selected Target Activity, the flow moves to that activity. Enter the percentage in the Value field.
9. Select the Default Target Activity from the drop-down list. This is the activity which will be executed if none of the condition is satisfied.
10. Click Ok.

17.1.2 Configuring Sequential Group Approval

Sequential Group approval is used when you have multiple tasks for an activity.

To configure sequential group approval:

1. From the Process Modeller window, select the required Process and click Edit. The Process Flow tab is displayed.
2. Double-click the Activity for which you want to configure user approval.
3. On the right pane, click 🔄. The Actions tab is displayed.
4. Click Group Approval to configure group approval. The Task Stage Details window is displayed.
5. Select **Sequential** to configure sequential group approval.

6. Select the **Condition to trigger intragroup** from the drop-down list. Intragroup is the task group of each task in a particular activity. The sequence in which each intragroup will be considered for voting is based on the sequence in which the tasks are added in the *Task Stage* pane in the *Actions* tab. The options are:
   - **Responded by a Member from the Group** - If anyone from the task group of the first task responds, it goes to the next task group and wait till someone from that task group respond and so on.
   - **Responded by Overall Majority** - If majority of users in the task group of the first task respond, it goes to the next task group and wait till majority of users in the task group of the second task respond and so on.
   - **Responded by Everyone in each Group** - Once all users in the task group of the first task have responded, it moves to the next task group. Then it waits till everyone in the second task group respond and so on.

7. Click **Add**. A row is added to define the voting formula for target activity.

**NOTE**

It is recommended that you define voting formula for all activities. If voting formula is not defined for an activity and if someone in a task group selects that activity, the workflow moves to that activity.

8. Select the required **Default Target Activity** from the drop-down list. This is the activity which will be executed if none of the condition is satisfied.

9. Select the required option from the **Voting Formula** drop-down list. The options are:
   - **Approved by anyone from all the groups** - If anyone selects the Target Activity, the flow moves to that activity.
• **Approved by Overall Majority** - If majority of the users select the Target Activity, the flow moves to that activity.

• **Approved by Everyone in each group** - If everyone in the group selects the Target Activity, the flow moves to that activity.

• **Approved by Overall Percentage** - Provide the percentage in the Value field. If the specified percentage of the users select the Target Activity, the flow moves to that activity.

10. Select the activity which needs to be executed if the condition fails, from the Default Target Activity drop-down list.

### 17.2 JsonPath Expressions

JsonPath expressions always refer to a JSON structure in the same way as XPath expression are used in combination with an XML document. The "root member object" in JsonPath is always referred to as `$` regardless if it is an object or array.

JsonPath expressions can use the dot–notation

```
$.store.book[0].title
```

or the bracket–notation

```
$['store']['book'][0]['title']
```

#### 17.2.1 Operators

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>$</code></td>
<td>The root element to query. This starts all path expressions.</td>
</tr>
<tr>
<td><code>@</code></td>
<td>The current node being processed by a filter predicate.</td>
</tr>
<tr>
<td><code>*</code></td>
<td>Wildcard. Available anywhere a name or numeric are required.</td>
</tr>
<tr>
<td><code>..</code></td>
<td>Deep scan. Available anywhere a name is required.</td>
</tr>
<tr>
<td><code>.&lt;name&gt;</code></td>
<td>Dot-notated child</td>
</tr>
<tr>
<td><code>['&lt;name&gt;' (, '&lt;name&gt;')]</code></td>
<td>Bracket-notated child or children</td>
</tr>
<tr>
<td><code>[[&lt;number&gt; (, &lt;number&gt;)]</code></td>
<td>Array index or indexes</td>
</tr>
<tr>
<td><code>[start:end]</code></td>
<td>Array slice operator</td>
</tr>
<tr>
<td><code>[?(&lt;expression&gt;)]</code></td>
<td>Filter expression. Expression must evaluate to a boolean value.</td>
</tr>
</tbody>
</table>

#### 17.2.2 Functions

Functions can be invoked at the tail end of a path - the input to a function is the output of the path expression. The function output is dictated by the function itself.
### Function

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>min()</td>
<td>Provides the min value of an array of numbers</td>
<td>Double</td>
</tr>
<tr>
<td>max()</td>
<td>Provides the max value of an array of numbers</td>
<td>Double</td>
</tr>
<tr>
<td>avg()</td>
<td>Provides the average value of an array of numbers</td>
<td>Double</td>
</tr>
<tr>
<td>stddev()</td>
<td>Provides the standard deviation value of an array of numbers</td>
<td>Double</td>
</tr>
<tr>
<td>length()</td>
<td>Provides the length of an array</td>
<td>Integer</td>
</tr>
</tbody>
</table>

#### 17.2.3 Filter Operators

Filters are logical expressions used to filter arrays. A typical filter would be `?[(@.age > 18)]` where `@` represents the current item being processed. More complex filters can be created with logical operators `&&` and `||`. String literals must be enclosed by single or double quotes `([?(@.color == 'blue')]` or `[?(@.color == "blue")])`

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>==</code></td>
<td>left is equal to right (note that 1 is not equal to '1')</td>
</tr>
<tr>
<td><code>!=</code></td>
<td>left is not equal to right</td>
</tr>
<tr>
<td><code>&lt;</code></td>
<td>left is less than right</td>
</tr>
<tr>
<td><code>&lt;=</code></td>
<td>left is less or equal to right</td>
</tr>
<tr>
<td><code>&gt;</code></td>
<td>left is greater than right</td>
</tr>
<tr>
<td><code>&gt;=</code></td>
<td>left is greater than or equal to right</td>
</tr>
<tr>
<td><code>=~</code></td>
<td>left matches regular expression <code>?[(@.name =~ /foo.*?/i)]</code></td>
</tr>
<tr>
<td><code>in</code></td>
<td>left exists in right <code>?[(@.size in ['S', 'M'])]</code></td>
</tr>
<tr>
<td><code>nin</code></td>
<td>left does not exists in right</td>
</tr>
<tr>
<td><code>subsetof</code></td>
<td>left is a subset of right <code>?[(@.sizes subsetof ['S', 'M', 'L'])]</code></td>
</tr>
<tr>
<td><code>size</code></td>
<td>size of left (array or string) should match right</td>
</tr>
<tr>
<td><code>empty</code></td>
<td>left (array or string) should be empty</td>
</tr>
</tbody>
</table>

#### 17.2.4 Path Examples

```json
{
    "store": {
        "book": [
            {
                "category": "reference",
            }
        ]
    }
}
```
17.3 Delegation

This feature facilitates you to delegate the tasks/notifications assigned to you to another user. The delegate can be your peer, someone from your immediate subordinate or someone from your all
subordinates. Additionally, you can revoke active delegations whenever required. Your user group should be mapped to the function role WFDELACC (Process Delegation User) if you want to define delegation.

This window displays all the delegations which are defined by the logged in user with details such as Delegation ID, Delegator, Delegate To, Start Date, End Date, and Status. You can add a new Delegation, view, modify, delete and revoke a delegation.

17.3.1 Adding a Delegate

To add a delegate

1. From the Delegation window, click Add. The Delegation Details window is displayed.
2. Enter the details as tabulated:

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delegation ID</td>
<td>Displays the auto generated Delegation ID.</td>
</tr>
<tr>
<td>Delegator</td>
<td>Displays the User ID of the logged-in user.</td>
</tr>
<tr>
<td></td>
<td>If your user group is mapped to the function role WFDELGADM, you can select the delegator from the drop-down list.</td>
</tr>
<tr>
<td>Identify Delegate</td>
<td>Select the required option from the drop-down list. The options are:</td>
</tr>
<tr>
<td></td>
<td>• Peers – Select this option if you want to delegate your tasks to your peer, who reports to your own manager.</td>
</tr>
<tr>
<td></td>
<td>• Subordinates- Select this option if you want to delegate your tasks to your immediate subordinates.</td>
</tr>
<tr>
<td></td>
<td>• All subordinates – Select this option if you want to delegate your tasks to someone who comes under you in your organization.</td>
</tr>
</tbody>
</table>
### Field Name | Description
--- | ---
Delegate To | Select the user to whom you want to delegate your tasks from the drop-down list. Based on the selected option from the Identify Delegate drop-down list, the users are displayed in this drop-down list. For example, if Peers is selected as Identify Delegate, this drop-down list displays all the peers in your organization. The data is fetched from AAI_EMPLOYEE_MASTER table.
Start Date and End Date | Specify the duration for which you want to delegate your tasks by selecting the Start Date and End Date from the calendar.
Notification Required | Turn ON the toggle button if you want to send a notification to the delegate or delegator.
Filter | This field is enabled only if the Notification Required toggle button is turned ON. Select to whom you want to send the notification. You can set to send notification to Delegator and Delegate. Select when you want to send the notification. The options are On Defining and On Execution.
Notification Message | This field is enabled only if the Notification Required toggle button is turned ON. Select the notification message you want to send to the delegate or delegator.
Scope | Select the scope of the delegation from the drop-down lists. The options are:
- All- Select this option to delegate all your tasks.
- Application- Select this option if you want to delegate all your tasks for a particular Application only.
- Process- Select this option if you want to delegate all your tasks for a particular Process only.
Application | This field is enabled only if Application or Process is selected as Scope. Select the required Application from the drop-down list. All your tasks related to the selected Application will be delegated to the selected user.
Process | This field is enabled only if Process is selected as Scope. Select the required Process from the drop-down list. The list displays all processes related to the selected Application. All your tasks related to the selected process will be delegated to the selected user.
Comments | Enter if you want to add any comments for the delegation.

3. Click **Save**.

### 17.3.2 Viewing Delegation

This option allows you to view the details of existing delegations. From the **Delegation** window, select the required delegation and click **View**. You can view the Delegation details.
17.3.3 Modifying Delegate Details

To modify delegate details

1. From the Delegation window, select the delegation you want to modify and click Edit. The Delegation Details window is displayed.
2. Modify the required details. For more information, see Adding a Delegate section.

17.3.4 Revoking Delegation

You can revoke only active delegations.

To revoke delegation

1. From the Delegation window, select the delegation you want to revoke and click Revoke.
2. Click OK in the confirmation message box.

17.3.5 Deleting Delegation

You cannot delete active delegations.

To delete a delegation

1. From the Delegation window, select the delegation you want to delete and click Delete.
2. Click OK in the confirmation message box.
18 Appendix B: Support APIs for Java External APIs

18.1 Connection API

For establishing connection with the Database, ConnectionAdapter class provided by PMF can be used.

18.1.1 Jar Files Required

The following jar available at $FIC_HOME/ficweb/webroot/WEB-INF/lib folder contains the ConnectionAdapter class that contains connection related API’s.

aai-pmf-common.jar

Referenced Files in Jar for Connection:

The aai-pmf-common.jar provides the following classes that can be used for query execution.

• ConnectionAdapter
• PreparedStatementDecorator

18.1.2 ConnectionAdapter Methods

The ConnectionAdapter class has the following API’s:

public static Connection getDBConnections() – For Config Connection
public static Connection getDBConnections(String infodom, Boolean isMetaConnection) – For Atomic Connection
public static void closeResultSet(ResultSet rs)
public static void closePreparedStatement(PreparedStatement ps)
public static void closeConnection(Connection conn)
public static void commitTransaction(Connection conn)
public static void rollBackTransaction(Connection conn)

18.1.3 Connection to Config Schema

To open a Config Schema connection, the getDBConnections method of ConnectionAdapter has to be invoked.

Connection configConn = ConnectionAdapter.getDBConnections();

For example:

public boolean testMethod(String attr1) {
    Connection configConn = null;
    PreparedStatementDecorator prepStatement = null;
    ResultSet rs = null;
    try {
        configConn = ConnectionAdapter.getDBConnections();
        "} catch (...) {
            configConn.close();
        "}
prepStatement = new PreparedStatementDecorator(configConn, query);
    prepStatement.setString(1, attr1);
    rs = prepStatement.executeQuery();
    while (rs.next()) {
        return true;
    }
}
}
catch (Exception e) {
    WorkflowUtil.logDebug("Error while updating process execution status..." + e);
}
}
finally {
    ConnectionAdapter.closeResultSet(rs);
    ConnectionAdapter.closePreparedStatement(prepStatement);
    ConnectionAdapter.closeConnection(configConn);
}
return false;
}

18.1.4 Connection to Atomic Schema

Opening Connection: To open an atomic connection the getDBConnections method of ConnectionAdapter has to be invoked with infodom and isMetaConnection as parameters.

Connection atomicConn = ConnectionAdapter.getDBConnections(infodom, false);

For example:

public boolean testMethod(String attr1) {
    Connection atomicConn = null;
    PreparedStatementDecorator prepStatement = null;
    ResultSet rs = null;
    try {
        atomicConn = ConnectionAdapter.getDBConnections(infodom, false);
        prepStatement = new PreparedStatementDecorator(configConn, query);
        prepStatement.setString(1, attr1);
        rs = prepStatement.executeQuery();
        while (rs.next()) {
            return true;
        }
    }
}
APPENDIX B: SUPPORT APIs FOR JAVA EXTERNAL APIs

LOGGING API

18.2 Logging API

For logging into an application, the WorkflowUtil class provided by PMF can be used.

18.2.1 Jar Files Required

The following jar available at $FIC_HOME/ficweb/webroot/WEB-INF/lib folder contains the WorkflowUtil class that contains Logging related API's.

- aai-pmf-common.jar

Referenced File in Jar:

The aai-pmf-common.jar provides the following class that can be used to implement Logging.

- WorkflowUtil

18.2.2 Debug Message

Debug messages can be used to log information that is required for debugging.

Signature

public static void logDebug(String logMessage)

Examples:

WorkflowUtil.logDebug( "Your Message" );
WorkflowUtil.logDebug( "Message" + variableName );

18.2.3 Error Message

Error message can be used to log an exception.

Signature:
public static void logError(Exception e)

For example:
WorkflowUtil.logError(e);