Oracle Financial Services Analytical Applications Infrastructure

Process Modelling Framework Orchestration Guide

Release 8.1.2.0.0

June 2024

ORACLE
Financial Services

ORACLE

OFS AAI Process Modelling Framework Orchestration Guide

Document Control

The following table provides details for the creation of the document and the updates done in the document.

Table : The Details of Revision to the Document

Version Number	Revision Date	Change Log
8.0	June 2024	• Added information related the icc batch calls and associated pmf execution identifier (35687894)
7.0	April 2024	• Included F2T Load model option in LoadT2T (36447944) • Updated sample for multiple legal entities (36403126) • Updated Parameters to include data Fields for the selected Data Mapping definition (36491684)
6.0	October 2023	• Execute Pipeline Endpoint Details updated for ERROR WHILE TRYING TO INVOKE PMF WORKFLOW FUNCTIONS USING API (35921440) • Added note to add service account user in headers while executing the following rest APIs (35888560) ▪ Execute a Pipeline ▪ Resume a pipeline ▪ Abort a pipeline ▪ Re-run a pipeline • Added Execution order and the screenshot in the Timer .(35045830) • Added a note to execute Updatepipeline.sh when improting PMF process of lower version (35525571)
5.0	February 2023	Added ESIC Widget details to Dynamic Parameters for Widgets.
4.0	December 2022	Updated the Execute, Abort, Resume, and Re-run Pipeline sections. Added the Status API section.
3.0	August 2022	Inline process is not supported from versions 8.0.8.5.0 onwards. All references to the Inline process are removed.
2.0	June 2022	Added the information on how to invoke PMF workflow functions: Resume, Rerun and Abort using rest service. Added the section: Configure Timer using a Command-Line Utility.

Version Number	Revision Date	Change Log
1.0	November 2021	Created the user guide for the features added and enhancements done for the Process Modelling Framework in the OFSAA Release v8.1.2.0.0.

Table of Contents

1 Introduction

Process Modelling Framework (PMF) is a design and execution framework that enables Process Pipeline developers to implement various Pipelines modeled 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, 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

Topics:

1.1 What is New in this Release of OFSAAAI Application Pack

This section lists new features and changes in the OFSAAAI Application Pack for Process Modeller Framework.

1.1.1 New Features in Release 8.1.2.0.0

This section lists the new features described in this User Guide.

Table : New features in the OFSAAAI Application Pack Release 8.1.2.0.0

Feature	Description
Performance Optimization	The performance in PMF is optimized to: • Reduce database requests to improve resource utilization. • Optimize Query Execution time using Indexes.
Updates to User Role Mapping and Access Rights	The Users in PMF can be configured to have only PMF Process Execution Privilege (WFACCNEXE Role Code). See the User Role Mapping and Access Rights Section for more details about this feature.

For more details, see the Oracle Financial Services Advanced Analytical Applications Infrastructure Release 8.1.2.0.0 Readme.

1.1.2 Deprecated Features

There are no Deprecated Features in this release.

1.1.3 Desupported Features

There are no Desupported Features in this release.

1.2 Categorization of Pipelines

The Categorization of Pipelines creates clear segregation of the nature of tasks being orchestrated.

For details, see the Oracle Financial Services Analytical Applications Infrastructure Extension Pack Online Documentation.

Applying the Extension Pack categorizes and updates the pipeline types appropriately depending on the nature of tasks that are being orchestrated. Further explanation on the nature of tasks and categorization is as follows:

1. Workflow Pipeline
2. Run Pipeline
3. Sub-Run Pipeline
4. Business Process Pipeline

You can define and orchestrate Business Process Pipelines only if the Oracle Financial Services Analytical Applications Infrastructure Extension Pack (OFSAAIE) is enabled in the OFS AAI setup. The Business Process Pipeline allows orchestration of various tasks and pipelines across Workflow, Run, and Sub-run Process Pipelines of any pipeline type whether it is a Business, Run, or Workflow Pipeline(s).

The various widgets in the PMF Canvas display depending on the type of pipeline selected, which are described as follows:

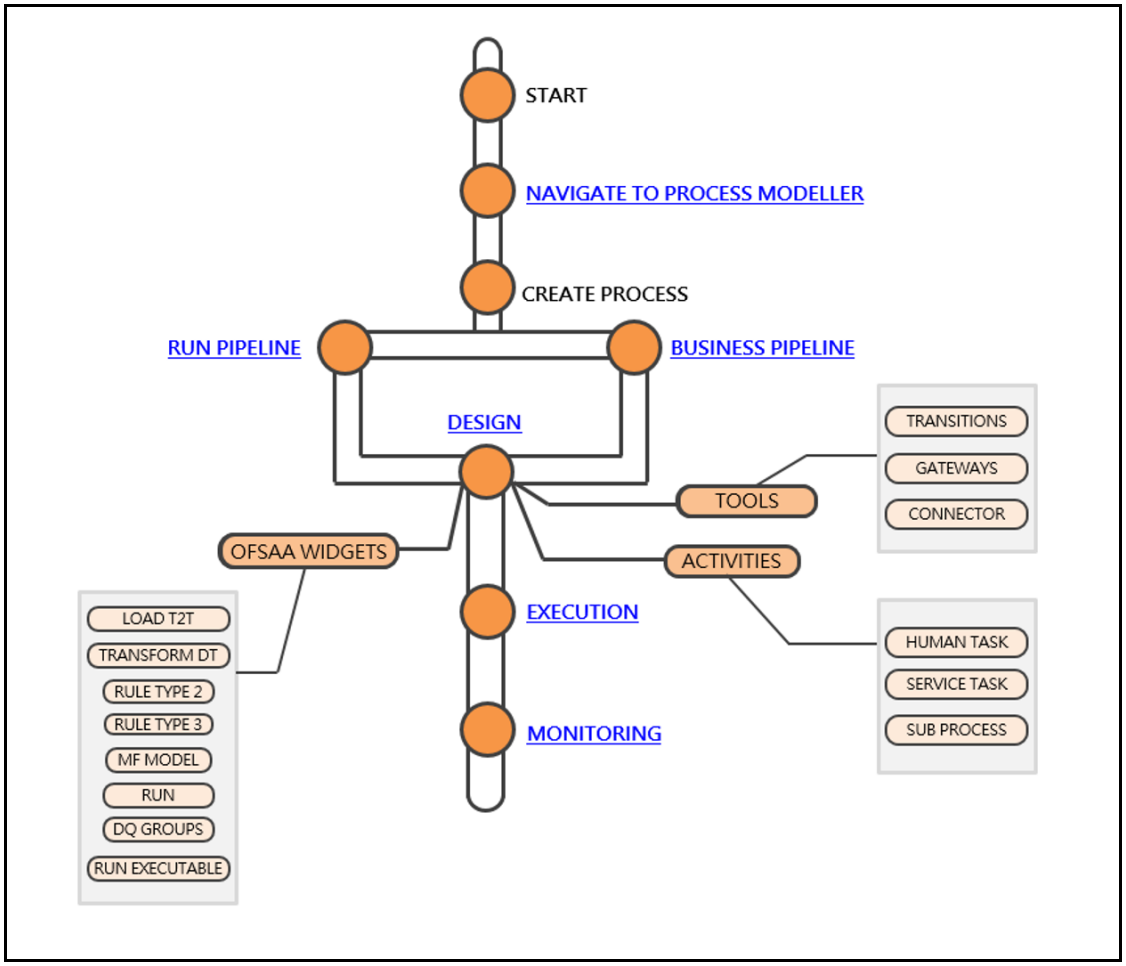
1. **Run Pipeline:** If any orchestrated pipeline consists of OFSAA tasks and service calls that run within the OFSAA context, it is categorized as Run Pipeline and it allows to stitch or orchestrate other processes of the type "subrunprocess". The Sub-run Process is further explained, or the Run can be mapped within a Run, which allows you to configure an array of tasks and orchestrate them based on the decisions attached.
2. **Workflow:** This represents a state-machine pipeline that allows you to orchestrate manual and automated system tasks, which helps you to stitch a state-change machine by performing manual or system tasks. If there is a Sub-pipeline, then you can stitch a Workflow pipeline. This process also allows you to have service tasks that make REST calls to systems outside of OFSAA.
3. **Subrunprocess:** This is a process similar to the Run process where a Sub-run Process Pipeline allows you to orchestrate the tasks in OFSAA as well as the services and API that run within the OFSAA context. You can also map other processes of the type "subrunprocess" within a Sub-run Process.
4. **Business Process:** This process allows you to access a full range of orchestration where the Business Process Pipeline can interleave a Run Pipeline to a Workflow Pipeline or can orchestrate a Sub-Run process or Workflow process. It enables the widget to have a full range of tasks in OFSAA as well as workflow tasks. This process also allows you to have service tasks that make REST calls to systems outside of OFSAA.

1.3 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 the 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.

1.4 Process Pipeline Flow

Figure : Process Pipeline Flow



> **TIP** After you click the links in the Process Pipeline Flow, press ALT+ Left Arrow to come back to this page.

1.5 Prerequisites

The following is the prerequisite to access and perform functions in the PMF user interface:

- User Role Mapping and Access Rights

1.5.1 User Role Mapping and Access Rights

User access to the PMF UI and the ability to perform functions in it is dependent on the mapping of the user profile to the roles in the OFS AAI application and the access rights assigned.

The following user role mapping is mandatory for PMF:

Table : User Role Mapping for PMF

Role Code	Role Name	Functionality
WFACC	Workflow Access	Assign this role to the user to access the Process Modeller menu from the Navigation Tree. **NOTE:** The mapping of this role does not allow view, edit and add actions.
WFKAFKA	Kafka Producer Consumer	Assign this role to the users who are required to perform the Event-based Orchestration of Process Flow. **NOTE:** The mapping of this Role displays the following icons in the PMF Canvas: • Event Producer • Event Consumer The messages sent by Banks to Topics to trigger the PMF Workflow are consumed by the Event Consumer. The messages from the PMF Workflow to the Banks (Topics) is sent by the Event Producer.
WFMACC	Workflow Monitor Access	Assign this role to the user to access the Process Monitor window. **NOTE:** The mapping of this role does not allow view, edit, and add actions.
WFREAD	Workflow Read	Assign this role to the user to view the PMF workflow.
WFWRITE	Workflow Write	Assign this role to the user to perform view, edit, and add actions in PMF.
WFACCNEXE	Workflow Execute On Read	Assign this Role to Users who have to Execute, Re-run, or Abort a Process. **PREREQUISITE:** Ensure that the Users are assigned the Workflow Access (WFACC) Role before you assign this Role.

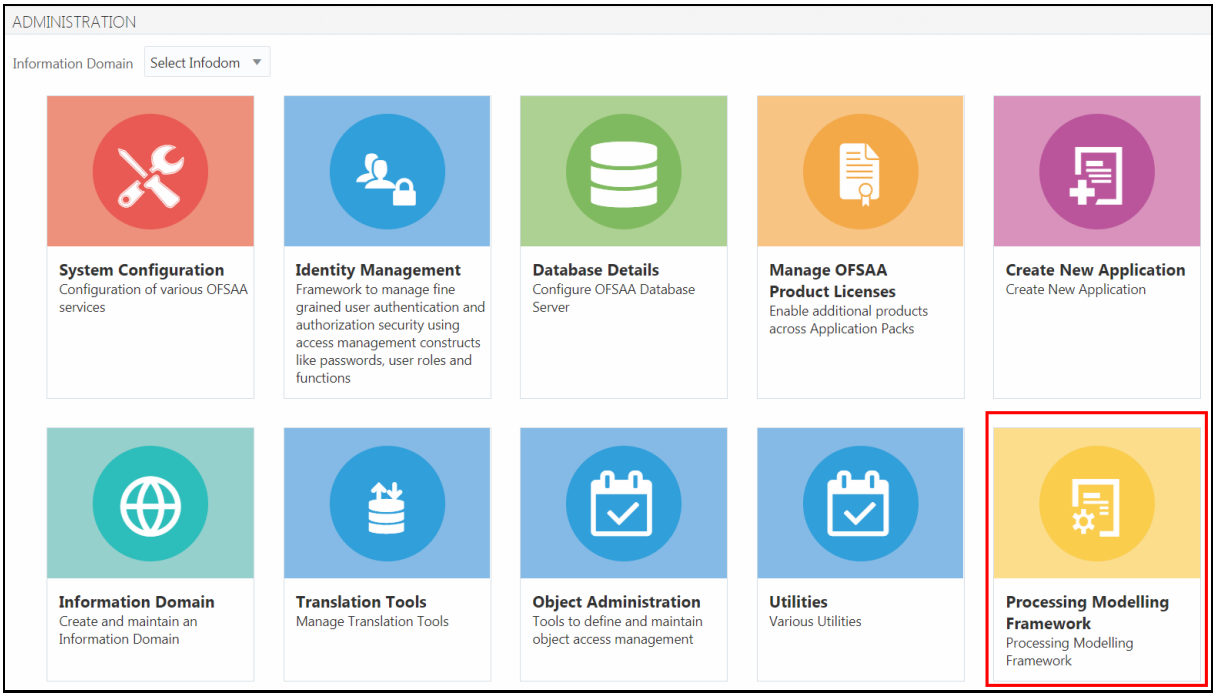
NOTE For administrators, ensure that they are mapped to all the roles described in the preceding table to allow them to perform all types of operations in PMF.

1.6 Access Process Modelling Framework

The following are the steps to access Process Modelling Framework:

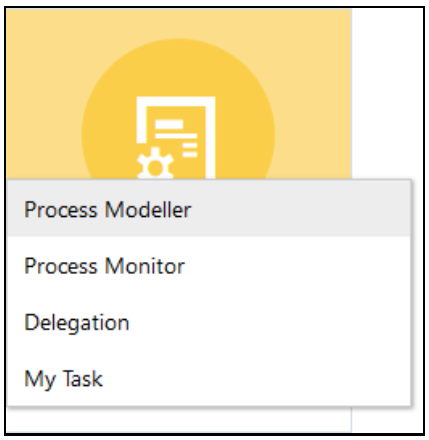
1. From the OFSAA Landing screen, click **Administration**.

Figure : Administration Home Window



2. Select the **Information Domain** from the drop-down list.
3. Click the **Process Modelling Framework** tile to display the sub-menu.

Figure : Process Modelling Framework Tile



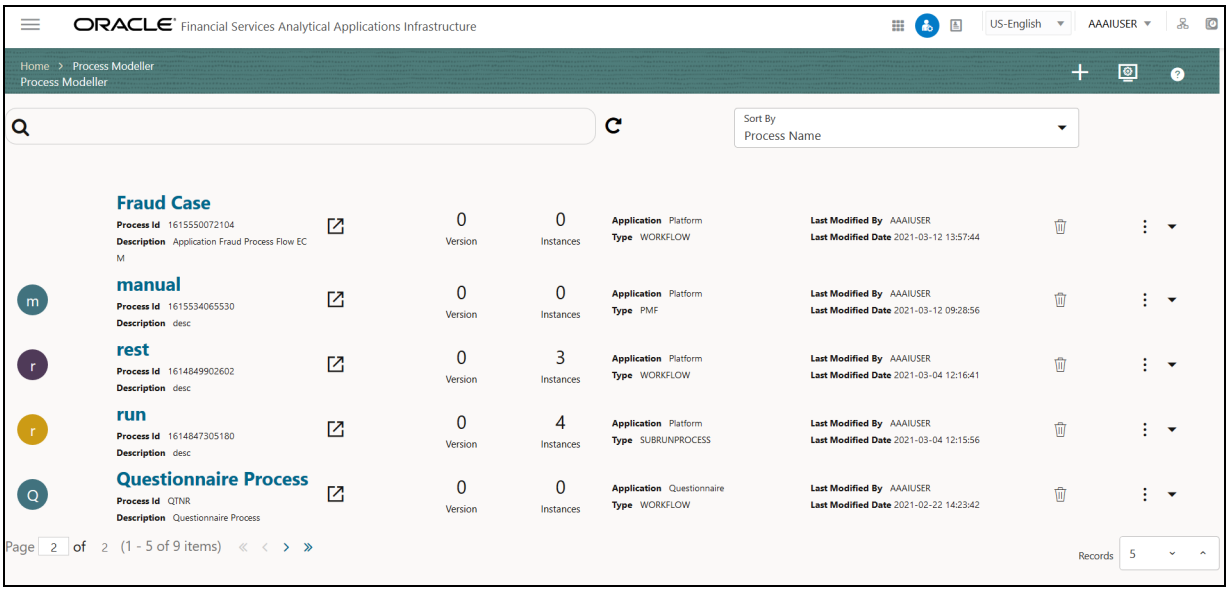
4. Click **Process Modeller** to launch the Process Modeller.
5. Click **Process Monitor** to monitor currently running processes.
6. Click **Delegation** to launch the Delegation framework.

2 Process Modeller

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

> **NOTE** Always execute `$FIC_DB_HOME/bin/UPDATEPIPELINETYPE.SH` when you import a PMF process from a lower version.

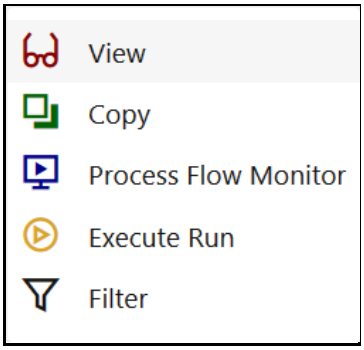
Figure : Process Modeller window



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 Pipeline.
- Click  to delete a Pipeline.
- Click  to view the following sub-menu:

Figure : Process Modeller sub-menu



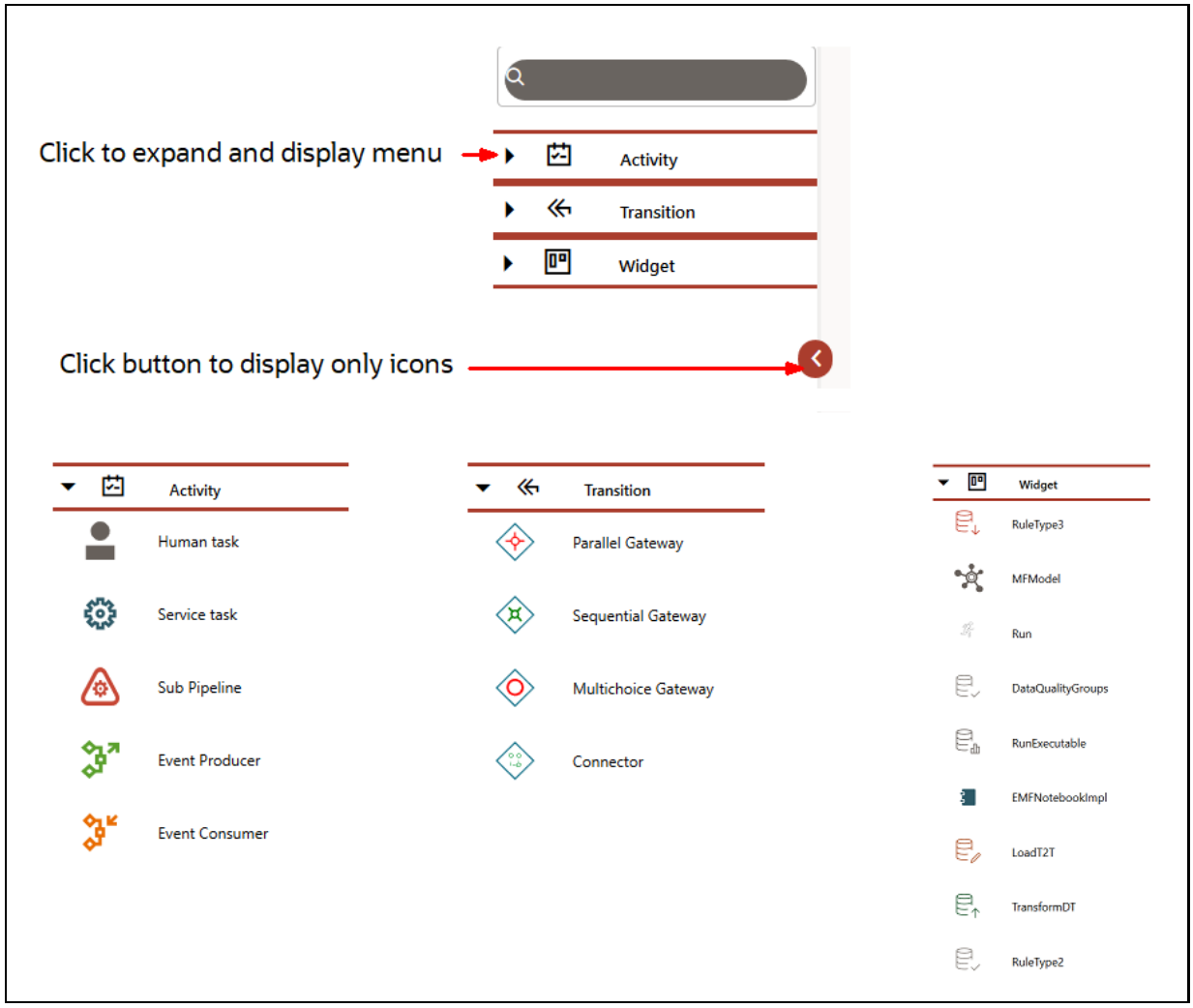
  - Click **View** to see the process flow.
  - Click **Copy** to copy and create a new Pipeline with the same process flow.
  - Click **Process Flow Monitor** to monitor the Pipeline.
  - Click **Execute Run** to execute a Run Pipeline.
  - Click **Filter** to apply a filter condition to a Run Pipeline.
- 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  . Click the Reset search icon  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 the **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  to go to the Process Monitor window.

3 Components for Designing Your Process Flow

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

Figure : Components for Designing your Process Flow



Topics:

- Widgets

3.1 Transition

A 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.

3.1.1 Gateways

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

- **Parallel Gateway**: A Parallel gateway is used when you want to have multiple transitions/flows that should be executed in parallel.
- **Sequential Gateway**: A Sequential gateway is used when you want to have multiple transitions/flows that should be run in sequence.
- **Multi Choice Gateway**: A Multi Choice gateway is used when you want to execute multiple transitions/flows based on the decision rule.

ATTENTION If you use a Parallel or Multi Choice gateway in your pipeline, ensure that after all the activities added to these gateways, it is merged or closed again with a Parallel or Multi Choice gateway. respectively. If there is no more activity to be performed after the Parallel or Multi Choice gateway, it is mandatory to add empty service task activity. Otherwise, the status is not updated correctly, and the next activity execution does not happen.

3.1.2 Connector

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

3.2 Human Task

A Human Task requires human intervention to move to the next activity.

For more information on stitching human tasks in your Process flow, see the Human Tasks section.

3.3 Service Task

Service task typically invokes an application component (for example, activity to invoke a business rule to calculate a certain threshold).

For more information on stitching service tasks in your Process flow, see the Service Tasks section.

3.4 Sub Pipeline

Sub Pipeline provides the reusability of Pipelines. Using the 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.

3.5 Event Producer

Event Producer creates messages as per the JSON definition and invokes the Producer API by passing the message and the Topic ID.

For more information on how to use Event Producer, see the Producer Activity section.

3.6 Event Consumer

Event Consumer makes asynchronous requests to execute activities in a process flow based on the conditions of event occurrences.

For more information on how to use Event Consumer, see the Producer Activity section.

3.7 Widgets

Widgets are used to execute OFSAA components such as T2T definitions, PLC definitions (DT), Rules (Classification Rule and Computation Rule), ESIC, Models in EMF, RRF Runs, Run Executable, 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 the Configuring Custom Components section.

4 Artifacts of Process Modelling

Before you start designing your Pipeline, you must be introduced to some artifacts of the Process Modelling Framework.

- Application Package
- Application Registration
- Data Fields
- Application Rules
- Application Object Model (AOM)

4.1 Application Package

Application Package is a concept used to group Pipelines, Application Rules, and Data Fields that 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.

Figure : AAI_WF_APP_PACKAGE_B table

V_APP_PACKAGE_ID	V_APP_PACKAGE_DESC	V_LANGIN	V_DEFINITION_PAGE_URL	V_IS_EMAIL_REQUIREI
0	Global			N
10	Business Restructure		Restructure/manage_grid.jsp?userId={ASSIGNEEUSERS}&loca	Y
100	Platform			Y
11	Questionnaire		solution/abc_qtnr/QtnrRedirectFrmPMFInbox.jsp?appCode={app	Y

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 the Application Level.

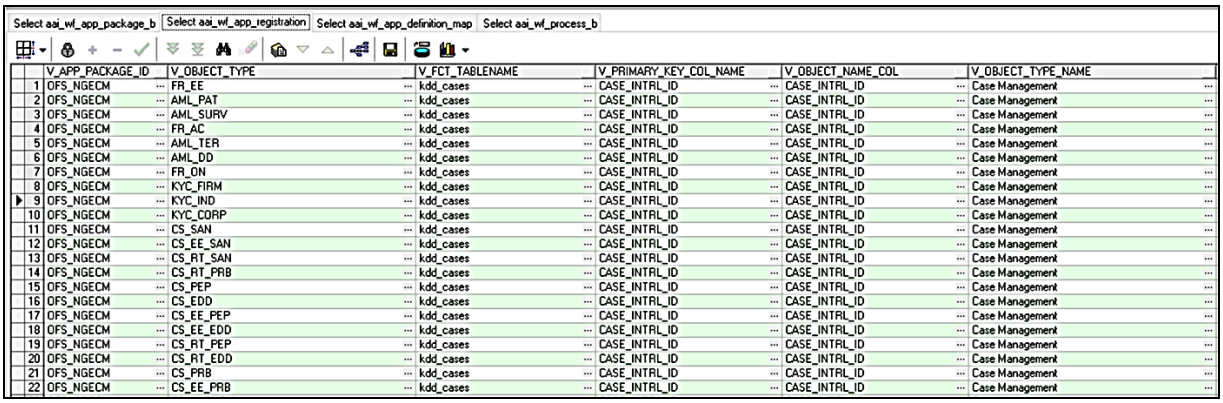
4.2 Application Registration

Application registration is required to define the Entity and the Attributes that need to be updated for a 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, and Object Type Name, and so on for an Application Package.

Figure : AAI_WF_APP_REGISTRATION table



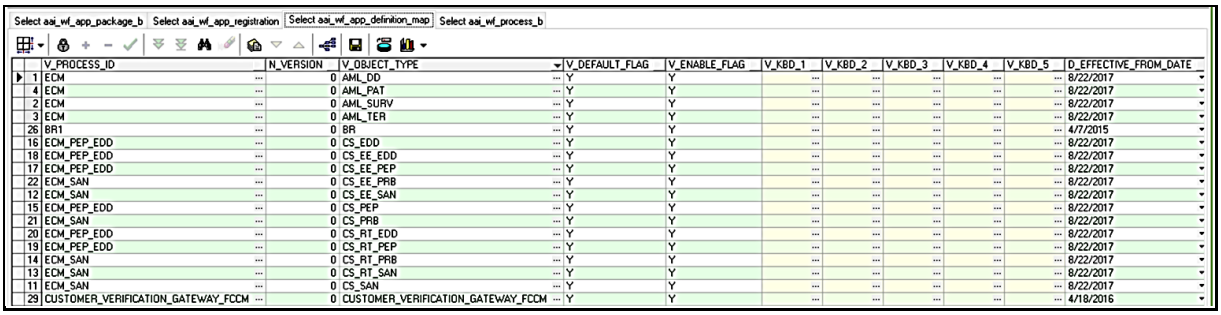
Select aai_wf_app_package_b | Select aai_wf_app_registration | Select aai_wf_app_definition_map | Select aai_wf_process_b

	V_APP_PACKAGE_ID	V_OBJECT_TYPE	V_FCT_TABLENAME	V_PRIMARY_KEY_COL_NAME	V_OBJECT_NAME_COL	V_OBJECT_TYPE_NAME
1	OFS_NGECM	FR_EE	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
2	OFS_NGECM	AML_PAT	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
3	OFS_NGECM	AML_SURV	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
4	OFS_NGECM	FR_AC	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
5	OFS_NGECM	AML_TER	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
6	OFS_NGECM	AML_DD	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
7	OFS_NGECM	FR_ON	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
8	OFS_NGECM	KYC_FIRM	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
9	OFS_NGECM	KYC_IND	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
10	OFS_NGECM	KYC_CORP	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
11	OFS_NGECM	CS_SAN	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
12	OFS_NGECM	CS_EE_SAN	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
13	OFS_NGECM	CS_RT_SAN	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
14	OFS_NGECM	CS_RT_PRB	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
15	OFS_NGECM	CS_PEP	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
16	OFS_NGECM	CS_EDD	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
17	OFS_NGECM	CS_EE_PEP	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
18	OFS_NGECM	CS_EE_EDD	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
19	OFS_NGECM	CS_RT_PEP	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
20	OFS_NGECM	CS_RT_EDD	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
21	OFS_NGECM	CS_PRB	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management
22	OFS_NGECM	CS_EE_PRB	kdd_cases	CASE_INTRL_ID	CASE_INTRL_ID	Case Management

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.

Figure : AAI_WF_APP_DEFINITION_MAP table



Select aai_wf_app_package_b | Select aai_wf_app_registration | Select aai_wf_app_definition_map | Select aai_wf_process_b

	V_PROCESS_ID	N_VERSION	V_OBJECT_TYPE	V_DEFAULT_FLAG	V_ENABLE_FLAG	V_KBD_1	V_KBD_2	V_KBD_3	V_KBD_4	V_KBD_5	D_EFFECTIVE_FROM_DATE
1	ECM	0	AML_DD	Y	Y						8/22/2017
4	ECM	0	AML_PAT	Y	Y						8/22/2017
2	ECM	0	AML_SURV	Y	Y						8/22/2017
3	ECM	0	AML_TER	Y	Y						8/22/2017
26	BR1	0	BR	Y	Y						4/7/2015
16	ECM_PEP_EDD	0	CS_EDD	Y	Y						8/22/2017
18	ECM_PEP_EDD	0	CS_EE_EDD	Y	Y						8/22/2017
17	ECM_PEP_EDD	0	CS_EE_PEP	Y	Y						8/22/2017
22	ECM_SAN	0	CS_EE_PRB	Y	Y						8/22/2017
12	ECM_SAN	0	CS_EE_SAN	Y	Y						8/22/2017
15	ECM_PEP_EDD	0	CS_PEP	Y	Y						8/22/2017
21	ECM_SAN	0	CS_PRB	Y	Y						8/22/2017
20	ECM_PEP_EDD	0	CS_RT_EDD	Y	Y						8/22/2017
19	ECM_PEP_EDD	0	CS_RT_PEP	Y	Y						8/22/2017
14	ECM_SAN	0	CS_RT_PRB	Y	Y						8/22/2017
13	ECM_SAN	0	CS_RT_SAN	Y	Y						8/22/2017
11	ECM_SAN	0	CS_SAN	Y	Y						8/22/2017
29	CUSTOMER_VERIFICATION_GATEWAY_FCCM	0	CUSTOMER_VERIFICATION_GATEWAY_FCCM	Y	Y						4/18/2016

4.3 Data Fields

Data Field, which is also known as Process Variable, helps Process Pipelines to access and store information from external sources. 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.

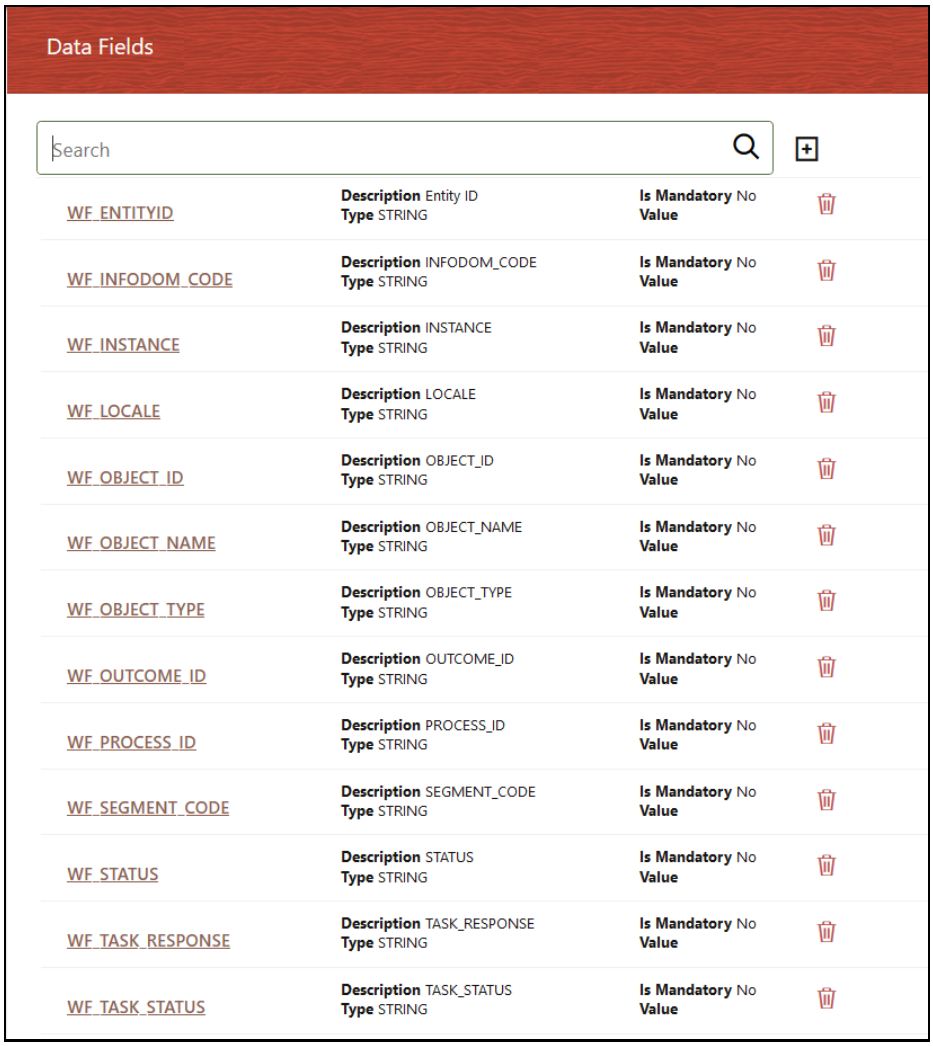
This section has the following subsections:

- Adding a Data Field
- System Data Fields
- AOM Data Field

4.3.1 Add a Data Field

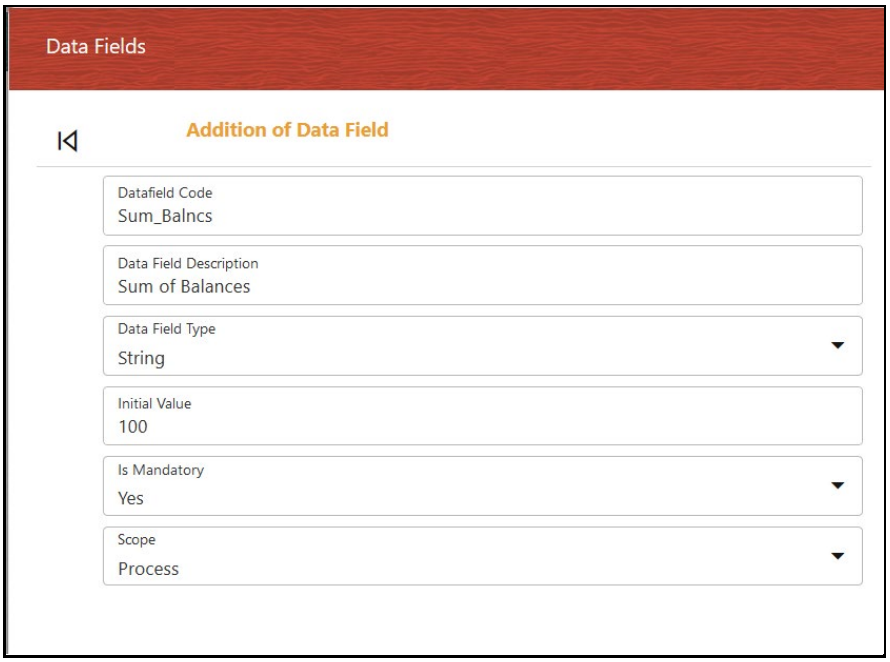
1. From the **Process Modeller** window, click the  submenu icon corresponding to the Pipeline for which you want to add a Data Field. The **Process Flow** tab is displayed.
2. Select  **Data Fields** from the header to display the **Data Fields** window.

Figure : Data Fields Window



3. Click **Add** to display the **Addition of Data Field** window.

Figure : Addition of Data Field Window



4. Enter the details as given in the table:

Table : Data Field Details Description

Field Name	Description
Data Field Code	Enter the Variable Name/Code, which needs to be used by the application to read or write into 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 the AOM Data Field type, see the AOM Data Field section.
Initial Value	Enter the default value for the Data.
Is Mandatory	This field applies to the 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 the **Accept** icon to save it.

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, are displayed in the **Select Run Parameters** window while executing Run Pipeline. For configuring AOM Data Fields, see Configuring Application Object Model (AOM) section.

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.

To access Application Rules, do as follows:

1. From the **Process Modeller** window, click the  submenu icon corresponding to the Pipeline for which you want to add an application rule. The **Process Flow** tab is displayed.
2. Select  **Application Rules** icon from the header to display the **Application Rules** window.
3. Click **Add**  to display the Add Application Rules window. You can delete a rule by clicking the  button.

Based on their usage the Application Rules are categorized as follows:

- **Execution Rule**: These are Business Logic executed as Tasks by an Activity.
- **Decision Rule**: This rule returns the 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`

Following are the available Application Rules:

- SQL Application Rule
- Stored Procedure Application Rule
- Function Application Rule
- Java Application Rule
- Java External API Application Rule
- Outcome Rules Application Rule
- Expression Application Rule

- Rest Service Application Rule
- Attribute Expression Application Rule
- Advanced Attribute Expression Application Rule
- JSON Path Expression Application Rule
- JSON Read From DB Application Rule
- JSON Write To DB Application Rule

4.4.1 SQL Application Rule

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

Figure : Add SQL Application Rule Details Window

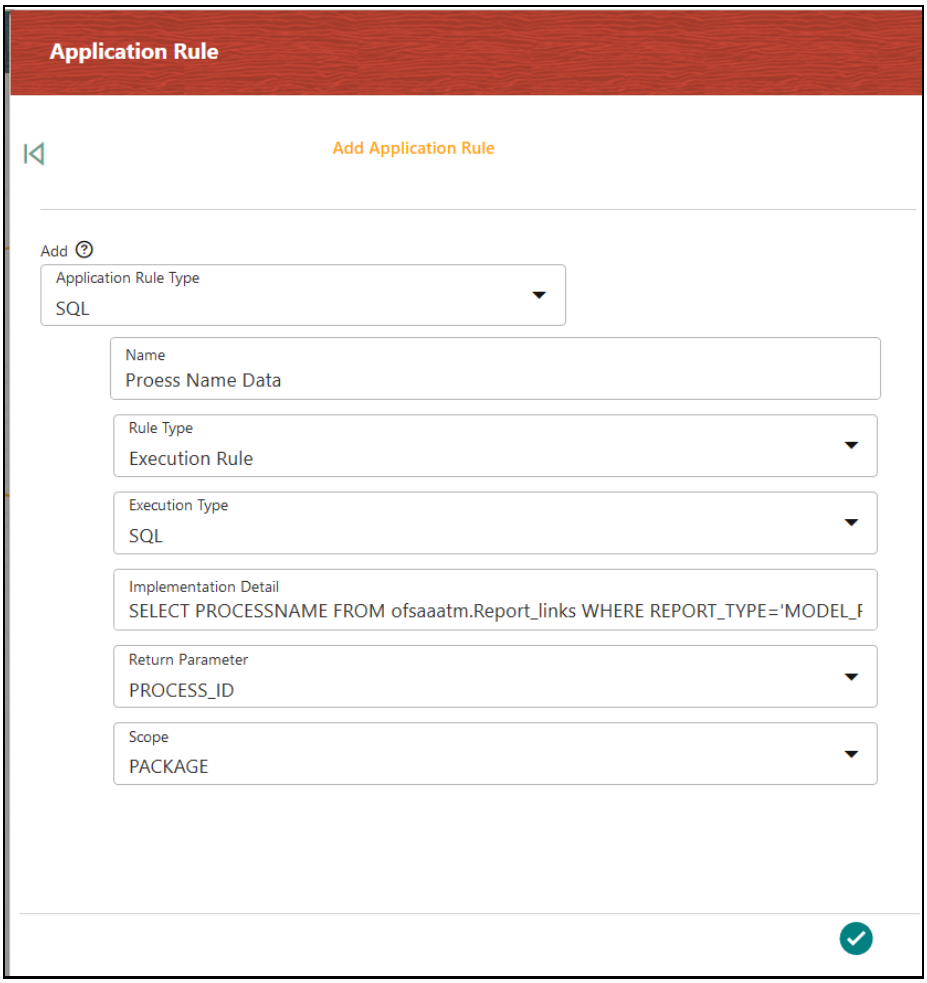


Table : SQL Application Rule Details Description

Field Name	Description
Application Rule Type	Select SQL as the Application Rule Type from the drop-down list.
Name	Enter a unique name for the SQL Application Rule.
Rule Type	Select the **Rule Type** from the drop-down list. The SQL Application Rule can be used as a Decision Rule, Execution Rule, or Selection Rule based on your requirement.
Execution Type	Displays the Application Execution Type as SQL.
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 that returns a list of values. For example, select v_created_by from fct_expenses id = {WF_ENTITYID}
Return Parameter	Select the Data Field that receives the return parameter of the SQL Rule, from the drop-down list. • For Execution Rule 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 the case of Selection Rule type, the Java method should be a String value. • In the case of the Decision Rule 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.

Figure : Stored Procedure Application Rule Details Window

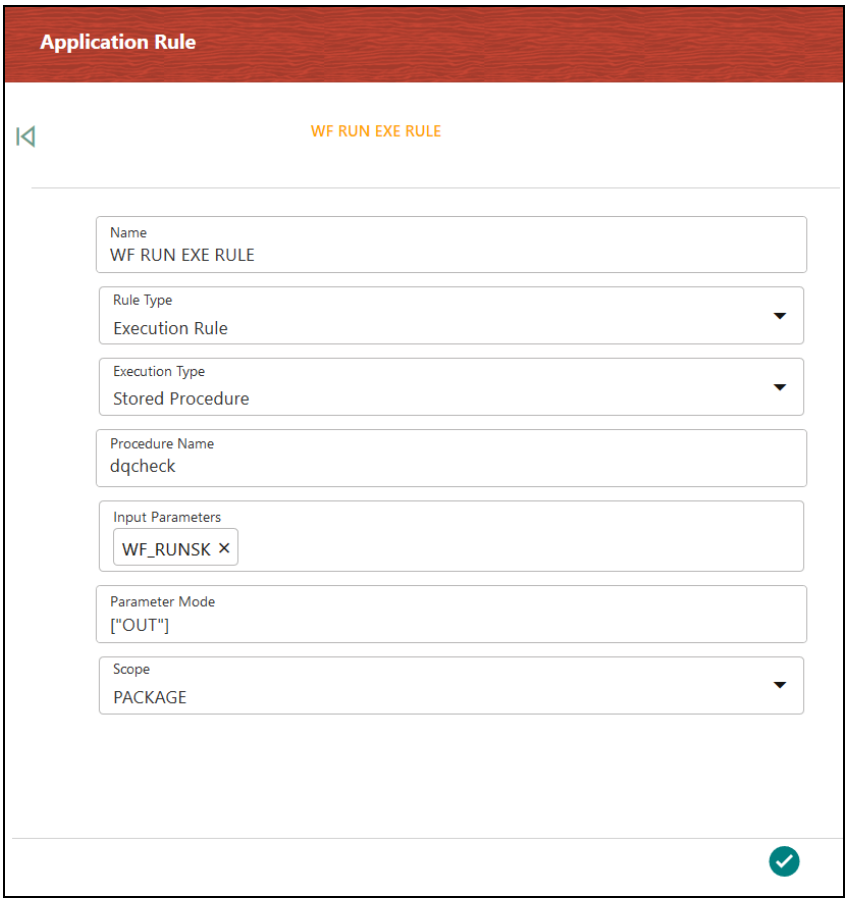


Table : Stored Procedure Application Rule Details Description

Field Name	Description
Name	Enter a unique name for the Stored Procedure Application Rule.
Rule Type	Select the **Rule Type** from the drop-down list. The Stored Procedure Application Rule can be used as a 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 that are passed as input parameters, from the drop-down list.

Field Name	Description
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 the case of the Decision Rule type, the first return parameter should return the value 'PASS' for success evaluation. • In the case of Selection Rule type, the first return parameter value is taken as Selection data. • In the case of the Execution Rule, the procedure return OUT parameter value overwrites the current value of the respective mapped Data Field.
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 Function Application Rule

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

Figure : Function Application Rule Details Window

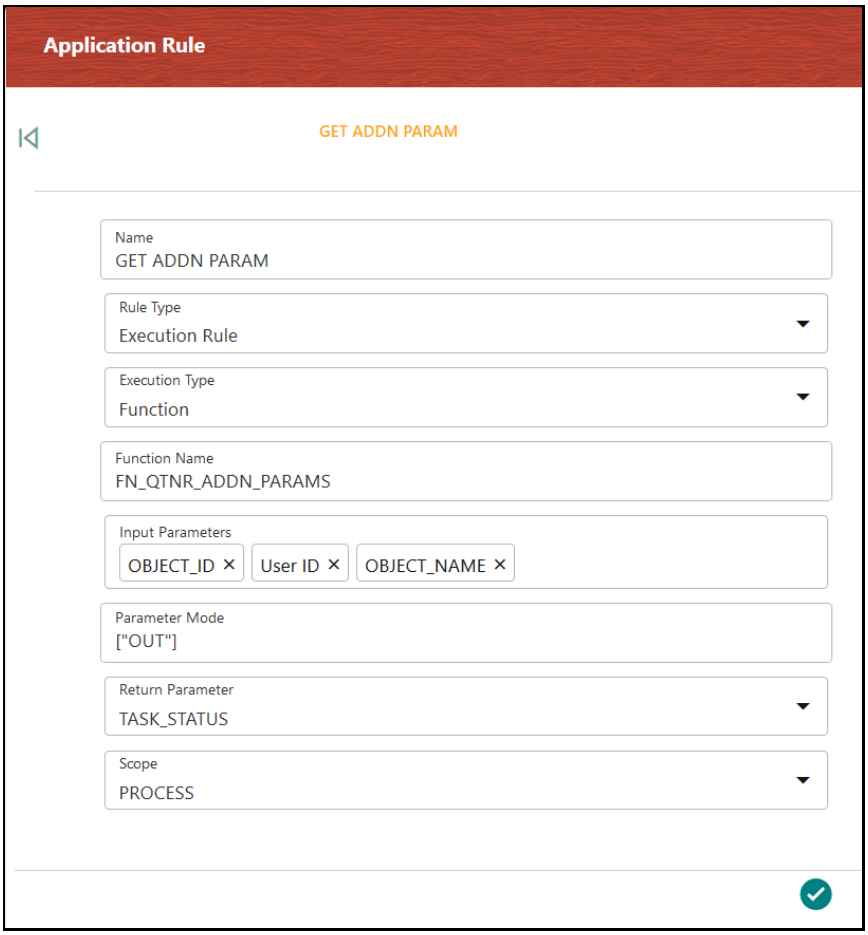


Table : Function Application Rule Details Description

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.
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 the case of the Decision Rule type, the first return parameter should return the value 'PASS' for success evaluation. • In the case of Selection Rule type, the first return parameter value is taken as Selection data. • In the case of the Execution Rule, the procedure return OUT parameter value overwrites the current value of the respective mapped Data Field.
Return Parameter	Select the Data Field that receives the return parameter of the Java function, from the drop-down list. • For Execution Rule 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 the case of Selection Rule type, the Java method should be a String value. • In the case of the Decision Rule 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 Java Application Rule

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

Figure : Java Application Rule Details Window

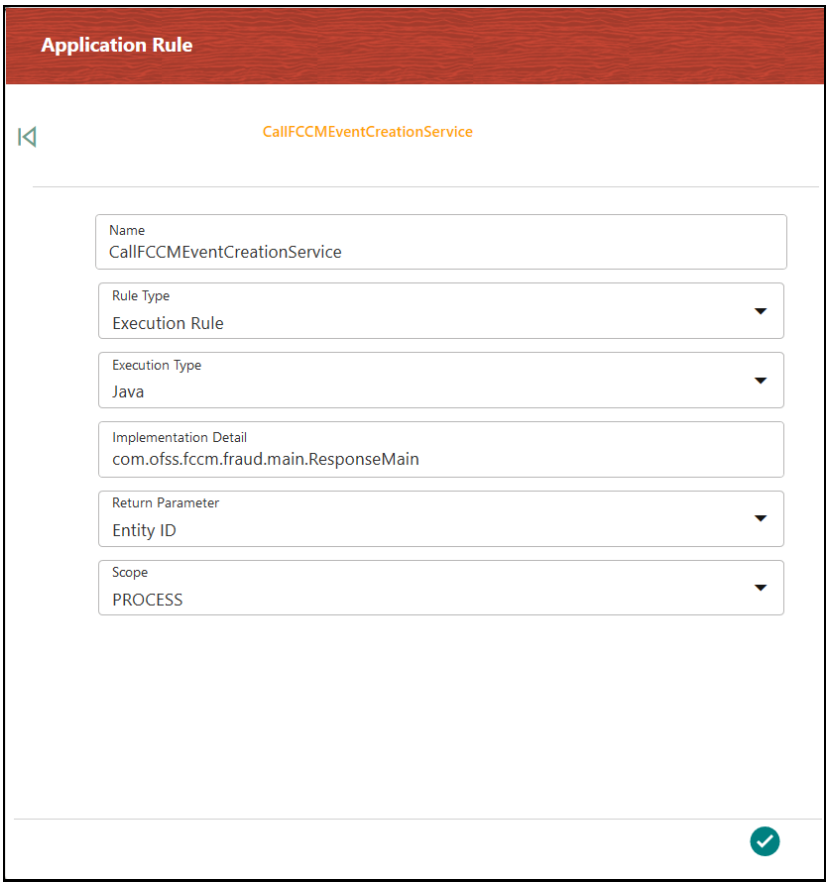


Table : Java Application Rule Details Description

Field Name	Description
Name	Enter a unique name for the Application Rule.
Rule Type	This Rule Execution type supports only the Execution Rule type.
Execution Type	Displays the Application Execution Type as JAVA.
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 Workflow Engine.

Field Name	Description
Return Parameter	Select the Data Field that receives the return parameter of the Java function, from the drop-down list. • For Execution Rule 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 the case of Selection Rule type, the Java method should be a String value. • In the case of the Decision Rule 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.

NOTE The class and its dependent file (or jar) need to be available in the web container classpath.
For example, `<TOMCAT_HOME>/webapps/<context>/WEB-INF/lib/<`**forecast.jar**`>`

4.4.5 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.

Figure : Java External API Application Rule Details Window

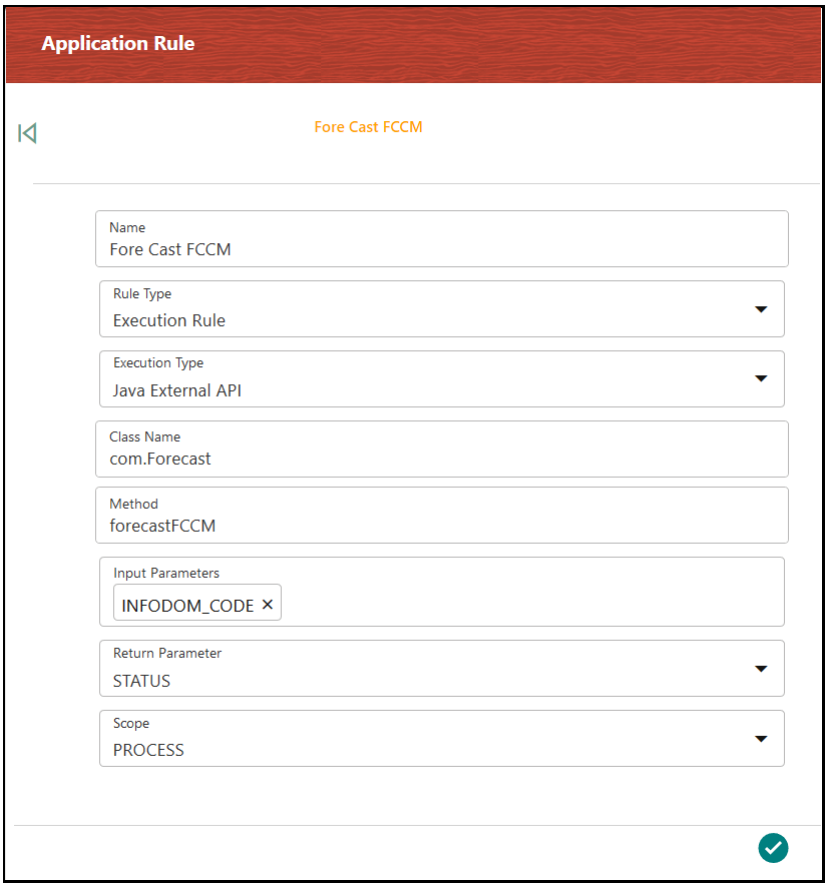


Table : Java External API Application Rule Details Description

Field Name	Description
Name	Enter a unique name for the Application Rule.
Rule Type	Select the **Rule Type** from the drop-down list. The Java External API Application Rule can be used as a Decision Rule, Execution Rule, or Selection Rule based on your requirement.
Execution Type	Displays the Application Execution Type as JAVA External API.
Class Name	Enter the complete java class name that implements the Business Logic.
Method	Enter the method that you want to execute.
Input Parameters	You can pass Input Parameters for the method using Data Fields. Select the required Data Fields from the drop-down list.

Field Name	Description
Return Parameters	Select the Data Field that stores the Return Parameter of the method, from the drop-down list.
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.

NOTE

1. The class and its dependent file (or jar) need to be available in the web container classpath.
   For example,
   `<TOMCAT_HOME>/webapps/<context>/WEB-INF/lib/<`**forecast.jar**`>`
2. For details on the supported APIs for use as Java External APIs, see Appendix B.

4.4.6 Outcome Rules Application Rule

Figure : Outcome Rule Details Window

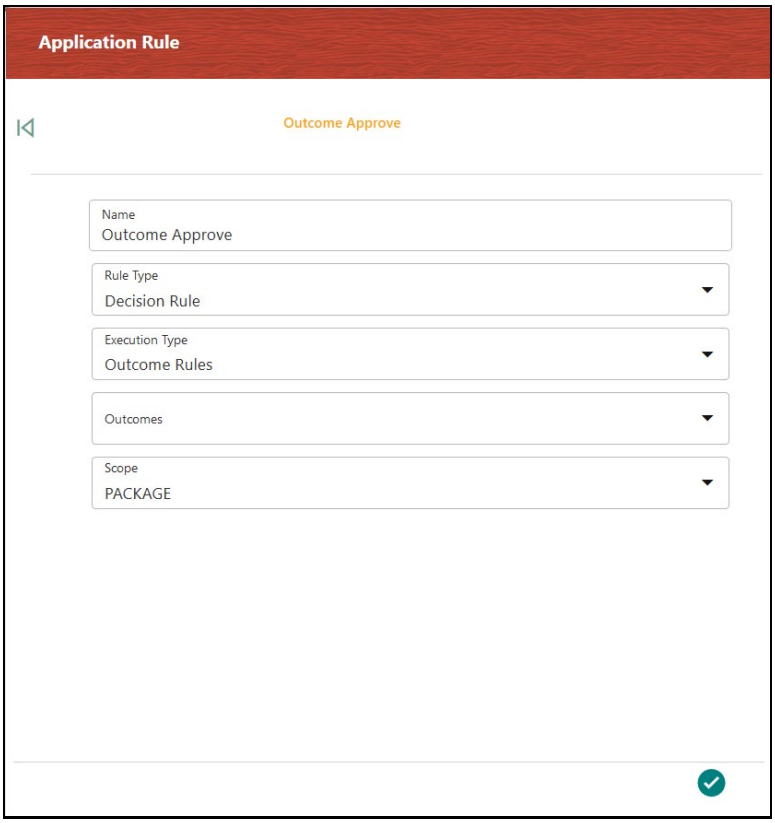


Table : Outcome Rule Details Description

Field Name	Description
Name	Enter a unique name for the Application Rule.
Rule Type	Displays the rule type as Decision Rule. This Rule Execution type supports only the Decision Rule type.
Execution Type	Displays the Application Execution Type as Outcome.
Outcomes	Select the outcome for which you want to add the rule.
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.

4.4.7 Expression Application Rule

This is the same as the SQL execution type. You need to specify only the where clause in the Expression field. It can be any SQL expressions including 'AND'/ 'OR'.

Figure : Expression Rule Details Window

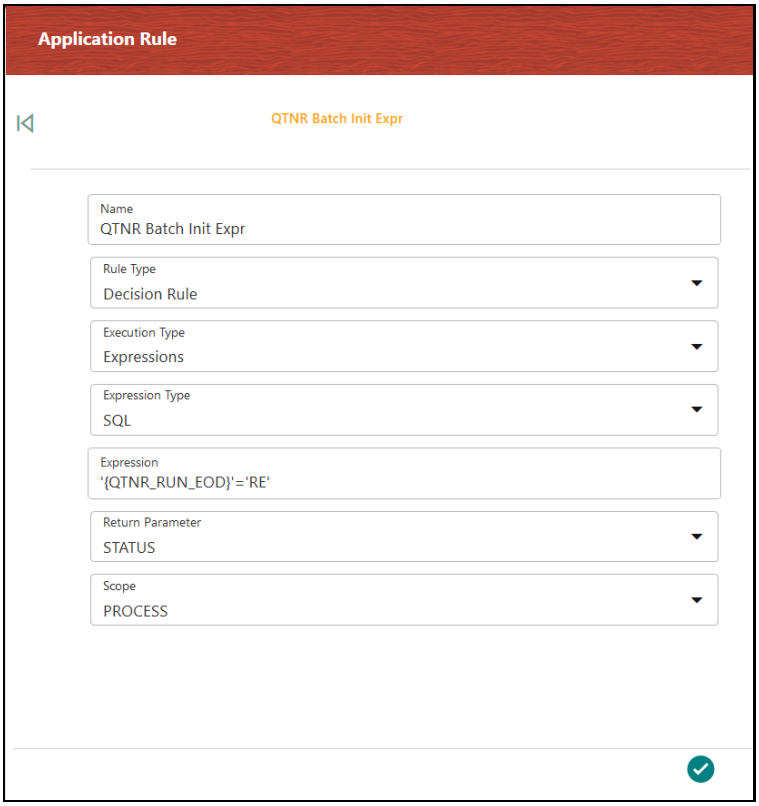


Table : Expression Rule Details Description

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 a 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 that receives the return parameter of the Expression, from the drop-down list. • For Execution Rule 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 the case of Selection Rule type, the Application Rule should be a String value. • In the case of the Decision Rule 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.

4.4.8 Rest Service Application Rule

This Application Rule is used to call any Rest services (internal or external) in your Process flow.

Figure : Rest Services Rule Details Window

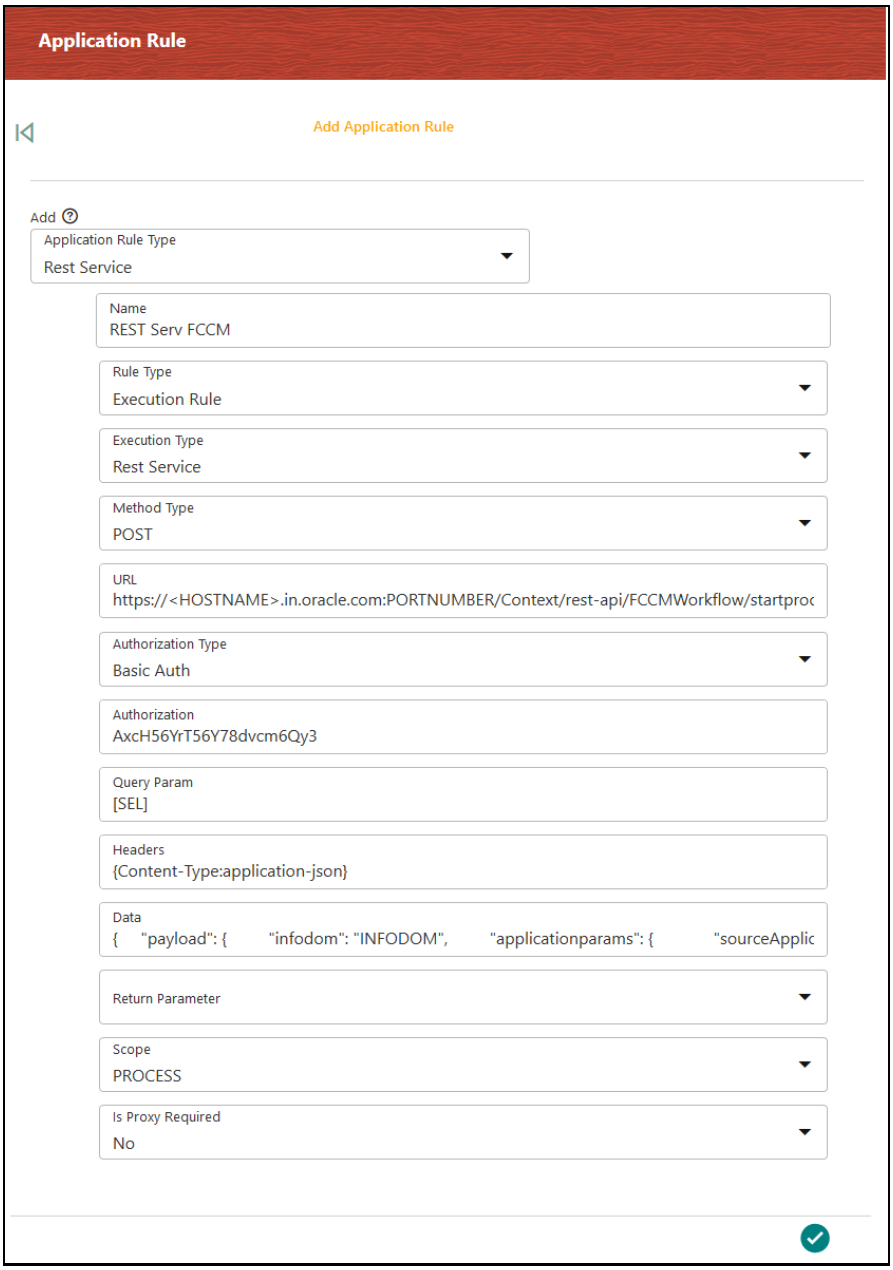


Table : Rest services Rule Details Description

Field Name	Description
Name	Enter a unique name for the Application Rule.

Field Name	Description
Rule Type	Select the **Rule Type** from the drop-down list. This Application Rule can be used as a Decision Rule, Execution Rule, or Selection Rule based on your requirement.
Execution Type	Displays the Application Execution Type as Rest service.
Method Type	Select the method type from the drop-down list. The options are **GET** and **POST**.
URL	Enter the REST URL that needs to be called. For example, `<IP Address/hostname of the Web Server >:<servlet port>/<context name>/restPMF/PMFService/startWorkflowProcess` (A rest URL to start the workflow).
Authorization Type	Select the authorization type from the drop-down list. The options are: • **No Auth**- Select this option for the rest services that do not need an authorization header. • **Basic Auth**- Select this option if you want to authenticate the invocation of the Rest service.
Authorization	This field is displayed only if you have selected **Basic Auth** as **Authorization Type**. • In the case of OFSAA local user, enter the User ID only. • In the case of an external user, enter the base 64 encoded string. For more information, see Authentication of Rest Service. • In case this field is left blank, logged-in user credentials will be taken as authorization header. **Note**: If Authorization is given in the Header explicitly, then it will take preference over the value given in the **Authorization** field.
Query Param	Enter the **Query Parameters** that need to be passed to the Rest API. For example, http://example.com/foo?bar
Headers	Enter any headers that need to be passed to the Rest API. For example, `"content-type": "application/json"` To pass the header values dynamically, use the following format: {Content-Type:~~TYPE~~,Authorization:~~CREDENTIAL~~} Where TYPE and CREDENTIAL are data fields.
Data	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, `"{\n \"objectid\":\"912\",\n \"objecttype\":\"1000\",\n \"infodom\":\"OFSCAPADQINFO\",\n \"segment\":\"OFSCAPADQINFO\",\n \"userid\":\"ORMUSER\",\n \"locale\":\"en_US\",\n \"securityMap\" :{},\n \"applicationParams\" :{\n \"testparam\":\"value1\",\n \"testparam2\":\"value2\"\n }\n \n}`

Field Name	Description
Return Parameter	Select the Data Field which will receive the return parameter of the Expression, from the drop-down list. • For Execution Rule 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 the case of Selection Rule type, the Application Rule should be a String value. • In the case of the Decision Rule 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 a 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 the REST Service Application Rule section.

4.4.8.1 Enabling Proxy for the 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 : AAI_WF_GLOBAL_SETTINGS Entries Table

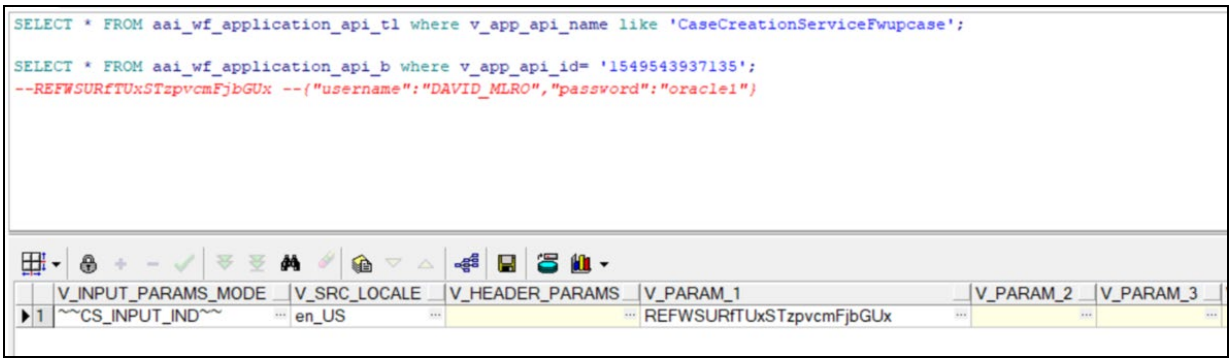
V_PARAM_NAME	V_PARAM_VALUE	Description
PROXY_SERVER_IP	For example, www.proxy.myserver.com	Provide the IP address of the Proxy server.
PROXY_SERVER_PORT	For example, 80	Provide the port number of the Proxy server.

4.4.8.2 Authentication of Rest Service

Basic auth is supported for authentication of the rest service. You have to encode your username and password using the Online encoder (https://www.base64encode.org/) and add the encoded value in the **Authorization** field in the **Rule Details** window. The format of the user name and password to be entered in the online encoder should be username:password. For example, if we give DAVID_MLRO:oracle1, it is converted to "REFWSURfTUxSTzpvcmFjbGUx".

You can also add this as a parameter V_PARAM_1 in the AAI_WF_APPLICATION_API_B table. This needs to be entered as given in the following figure.

Figure : AAI_WF_APPLICATION_API_B table



4.4.9 Attribute Expression Application Rule

Figure : Attribute Builder Window

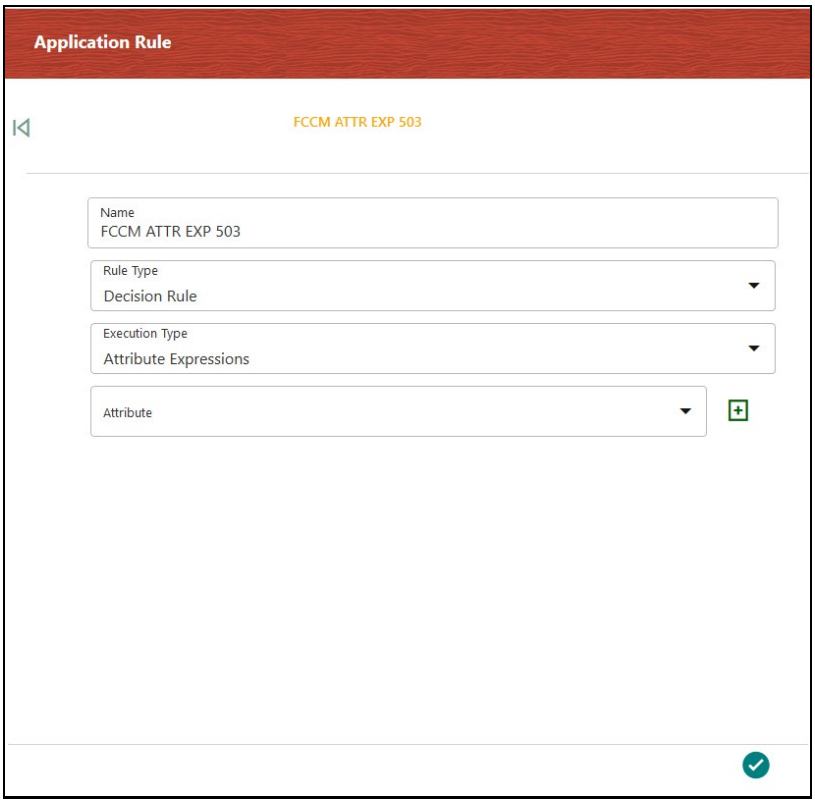


Table : Attribute Builder Description

Field Name	Description
Name	Enter a unique name for the Application Rule.

Field Name	Description
Rule Type	Displays the rule type as Decision Rule. This Rule Execution type supports only the Decision Rule type.
Execution Type	Displays the Application Execution Type as Attribute Expression.
Attribute	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** window. 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 the button. You can select multiple attributes and click **Add** to assign values to those attributes.

4.4.10 Advanced Attribute Expression Application Rule

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

You can define this application rule with multiple conditions and nested groups.

Figure : Advanced Attribute Expression Window

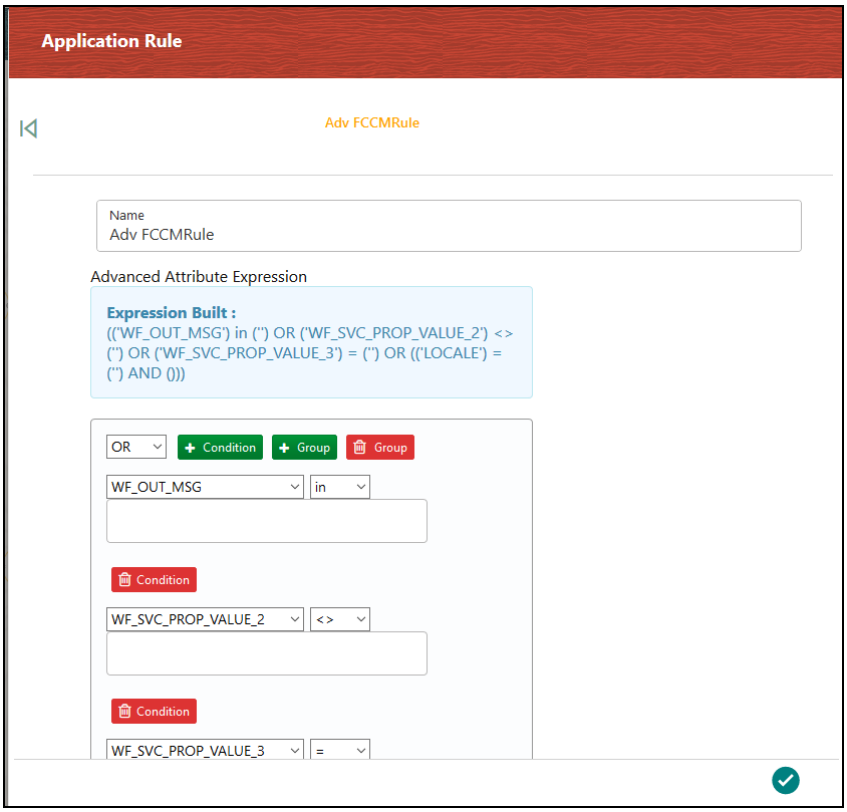


Table: Advanced Attribute Expression Description

Field Name	Description
Rule Name	Enter a unique name for the Application Rule.
AND/ OR	Select the logical operator to be used for the conditions in a group.
Add Condition	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. • 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. • Operator- Available options are in,=,<>,<,<=,>,>=. • Value- Displays the values configured for the selected attributes. Select the required value. Click **Remove Condition** to delete an already added condition.

Field Name	Description
Add Group	Click **Add Group** if you want to have nested conditions. For each group, select the required logical operator as AND or OR. Click **Remove Group** to delete a group of conditions.

4.4.11 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.

Figure: JSON Path Expression Rule Details Window

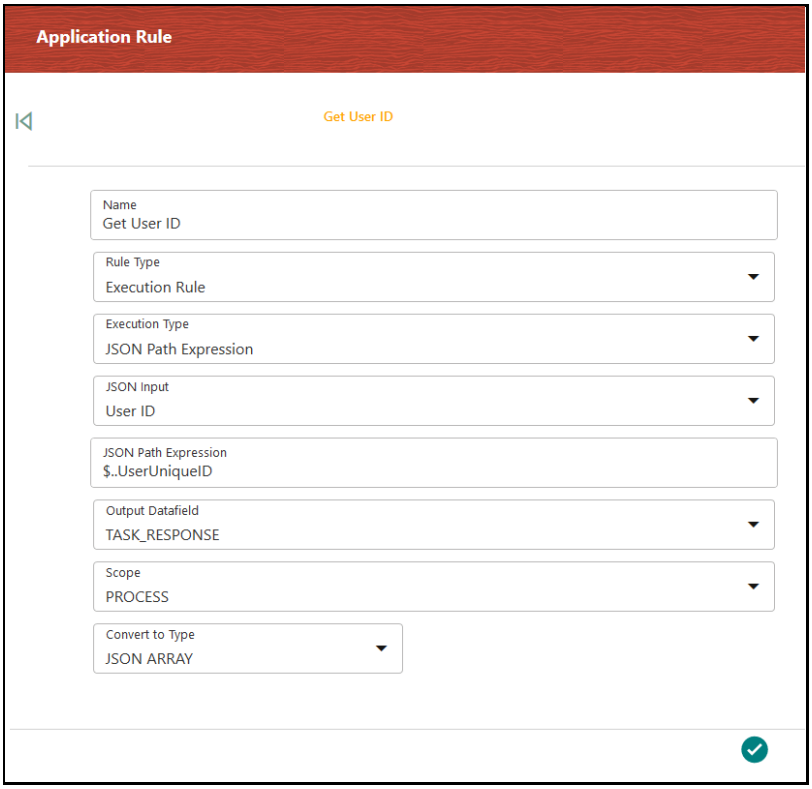


Table : JSON Path Expression Rule Details Description

Field Name	Description
Name	Enter a unique name for the Application Rule.

Field Name	Description
Rule Type	Select the rule type from the drop-down list. The available rule types are Decision Rule and Execution Rule. For Decision Rule, the output of JSON Path Expression is compared with RHS expression, and the rule returns as 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 the 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.

Figure : JSON Read from DB Rule Details Window

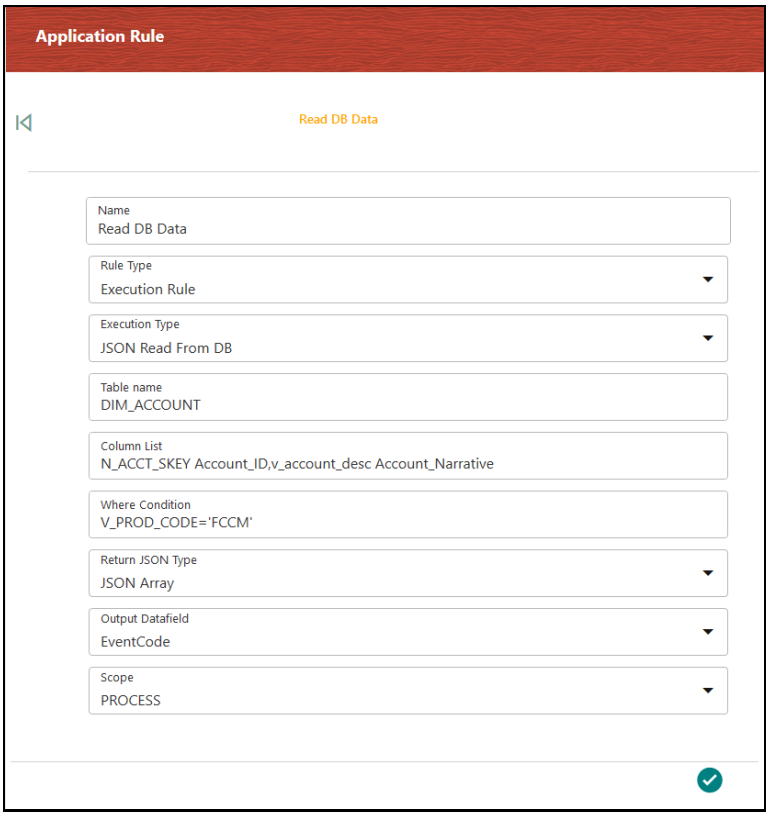


Table : JSON Read from DB Rule Details Description

Field Name	Description
Name	Enter a unique name for the Application Rule.
Rule Type	Only the Execution Rule type is supported.
Execution Type	Displays the Execution Type as JSON Read From DB.
Table Name	Enter the table name from which you want to read the data.
Column List	Enter the column names of the selected table.
Where Condition	Enter the filter condition (where clause) of the SQL query.
Return JSON Type	Select the JSON type of the returned value as JSON Object or JSON Array based on your requirement. • JSON Object- Select this option if the returned value is a single row. • JSON Array- Select this option if the returned data has multiple rows.

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.

Figure : JSON Write to DB Rule Details Window

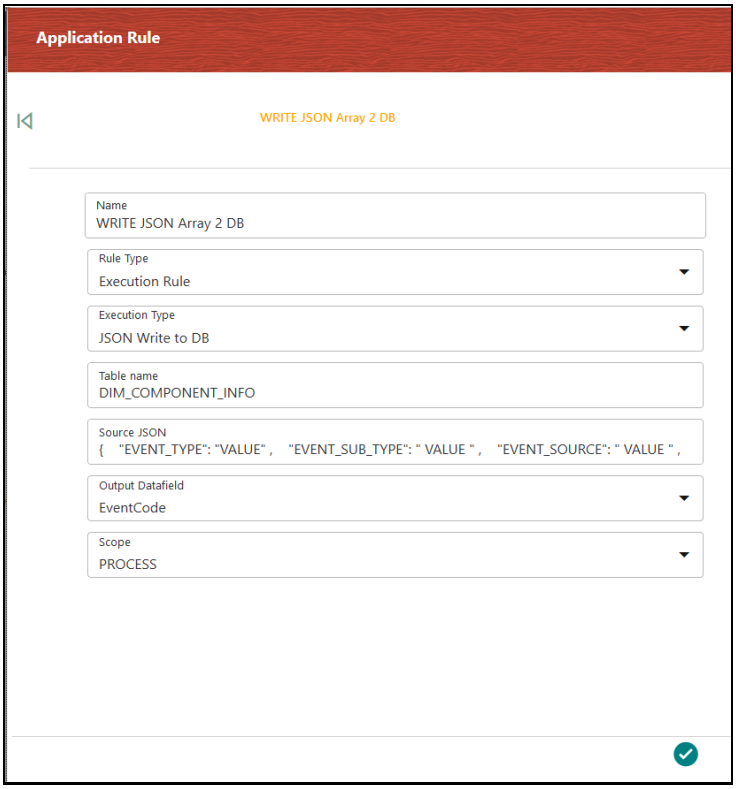


Table : JSON Write to DB Rule Details Description

Field Name	Description
Name	Enter a unique name for the Application Rule.

Field Name	Description
Rule Type	Only the 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.
Source JSON	Enter the data in the JSON format that you want to write to the database.
Output DataField	This field is not applicable.
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.

4.5 Configuring Application Object Model (AOM)

This module helps in creating a set of attributes for a given application abstractly so that frameworks like PMF and other modules can leverage to retrieve application attributes and their values.

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

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

4.5.1 AAI_AOM_APP_COMP_ATTR_MAPPING Table

In this table, make entries for each attribute as given in the following table:

Table : AAI_AOM_APP_COMP_ATTR_MAPPING Table Description

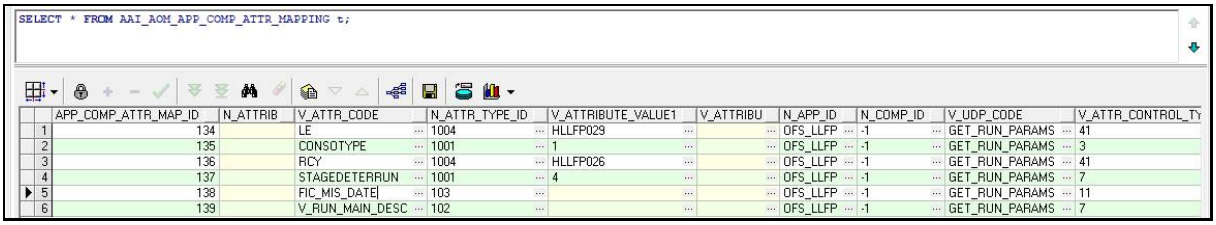
Column Name	Description
APP_COMP_ATTR_MAP_ID	Enter a unique ID for the attribute. You need to enter the Attribute name and description for each attribute ID entered here in the AAI_AOM_APP_COMP_ATTR_TL table. See the AAI_AOM_APP_COMP_ATTR_TL Table section.
V_ATTR_CODE	Name of the attribute.
N_ATTR_TYPE_ID	The ID of the attribute type. The values of the attributes are fetched based on the attribute type. 1001- Static

Column Name	Description
	1002- SQL Query 1003- JavaAPI 1004- Hierarchy 1005- Multi-Select Hierarchy 103- Date field 102- Text box field For more information, see Attribute Types.
V_ATTRIBUTE_VALUE1 V_ATTRIBUTE_VALUE2	Values to be fetched for the attribute. Based on the attribute type, you need to pass the values.
N_APP_ID	Application code for which the current attribute is configured. For example, if you are configuring Run execution parameters for the IFRS application, enter the application ID of IFRS here.
N_COMP_ID	Component code for which the attribute is configured.
V_UDP_CODE	Special property used by applications (user-defined). For example, 'GET_STATUS' –to get the status for the workflow.
V_ATTR_CONTROL_TYPE	Enter the Control type ID to be used for the attribute. For example, 3 is used for drop-down list, 7 for textbox, 11 for date control, 41 is for hierarchy, and 42 for Multi-Select hierarchy

4.5.2 Example for Run Pipeline

The following figure shows an example of entries in the AAI_AOM_APP_COMP_ATTR_MAPPING table for Run Pipeline:

Figure : AAI_AOM_APP_COMP_ATTR_MAPPING table



	APP_COMP_ATTR_MAP_ID	N_ATTRIB	V_ATTR_CODE	N_ATTR_TYPE_ID	V_ATTRIBUTE_VALUE1	V_ATTRIBU	N_APP_ID	N_COMP_ID	V_UDP_CODE	V_ATTR_CONTROL_TY
1	134		LE	1004	HLLFP029		OFS_LLFP	-1	GET_RUN_PARAMS	41
2	135		CONSOTYPE	1001	1		OFS_LLFP	-1	GET_RUN_PARAMS	3
3	136		RCY	1004	HLLFP026		OFS_LLFP	-1	GET_RUN_PARAMS	41
4	137		STAGEDETERRUN	1001	4		OFS_LLFP	-1	GET_RUN_PARAMS	7
5	138		FIC_MIS_DATE	103			OFS_LLFP	-1	GET_RUN_PARAMS	11
6	139		V_RUN_MAIN_DESC	102			OFS_LLFP	-1	GET_RUN_PARAMS	7

4.5.3 Attribute Types

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

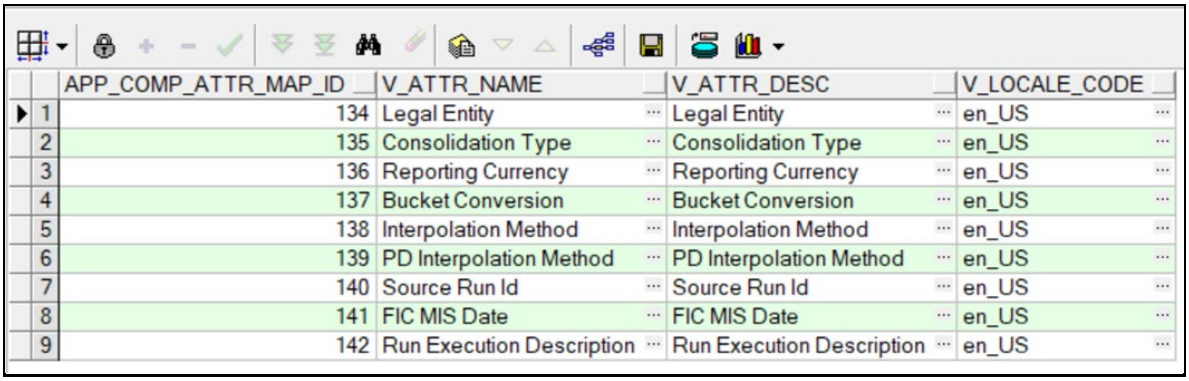
- **1001 (Static)** - Store attribute values in the AAI_AOM_STATIC and AAI_AOM_STATIC_TL tables.
- **1002 (Query)** - Enter the SQL query in V_ATTRIBUTE_VALUE1 in the AAI_AOM_APP_COMP_ATTR_MAPPING table, which has to be fired to fetch the attribute values.

- **1003 (JavaAPI)** – Enter the method that is configured for `V_ATTRIBUTE_VALUE1` for the required attribute. The configured method in the classpath 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.
- **103 (DATE)** - This is used for configuring FIC_MIS_DATE. The V_ATTR_CONTROL_TYPE value should be 11.
- **102 (Text Box)** - This is used for the Description field. The V_ATTR_CONTROL_TYPE value should be 7.

4.5.4 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:

Figure : APP_COMP_ATTR_MAP_ID table



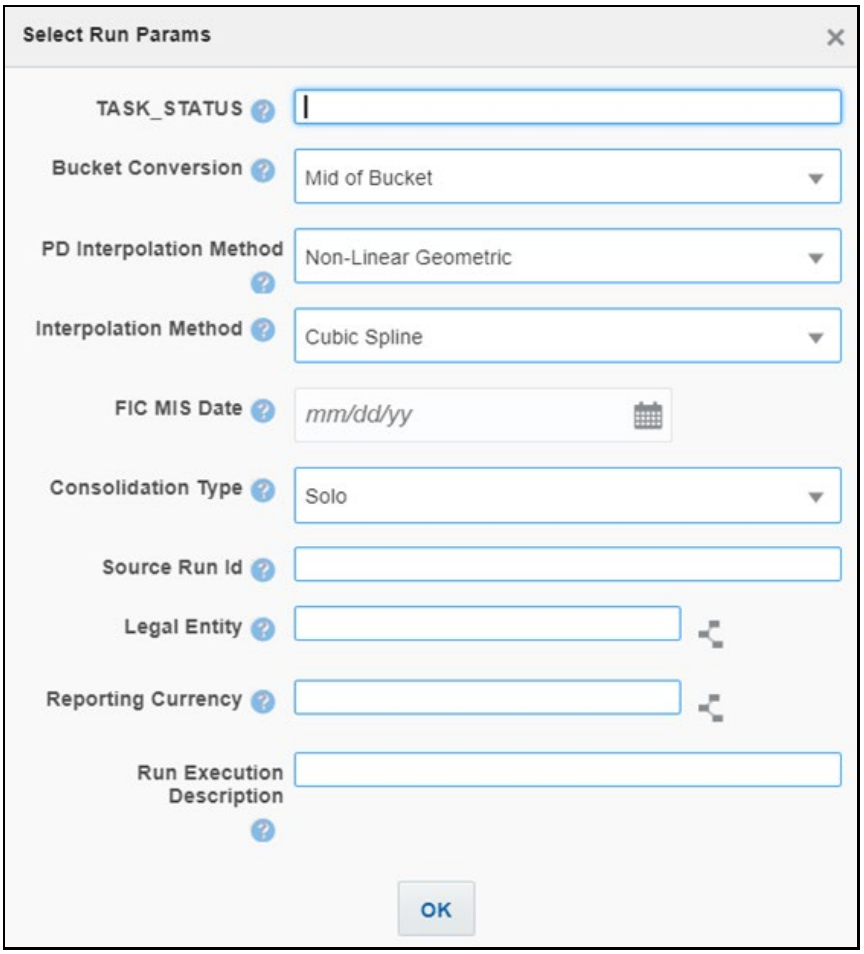
	APP_COMP_ATTR_MAP_ID	V_ATTR_NAME	V_ATTR_DESC	V_LOCALE_CODE
1	134	Legal Entity	Legal Entity	en_US
2	135	Consolidation Type	Consolidation Type	en_US
3	136	Reporting Currency	Reporting Currency	en_US
4	137	Bucket Conversion	Bucket Conversion	en_US
5	138	Interpolation Method	Interpolation Method	en_US
6	139	PD Interpolation Method	PD Interpolation Method	en_US
7	140	Source Run Id	Source Run Id	en_US
8	141	FIC MIS Date	FIC MIS Date	en_US
9	142	Run Execution Description	Run Execution Description	en_US

4.5.5 Usage of AOM Attributes in Run Pipeline

While executing the Run pipeline, the Select Run Params window displays the AOM fields that 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 is displayed as shown:

Figure : Select Run Params window



5 Design a Pipeline

Business pipelines are defined in OFSAA to design and execute the sequence of tasks that 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 Modeller, we can perform as follows:

1. Orchestrate a Business pipeline.
2. Orchestrate a Run pipeline using PMF modeling.
3. Perform Additional Functionalities

Topics:

5.1 Orchestrate a Business Pipeline

A business Pipeline is used to design a Business Process that 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

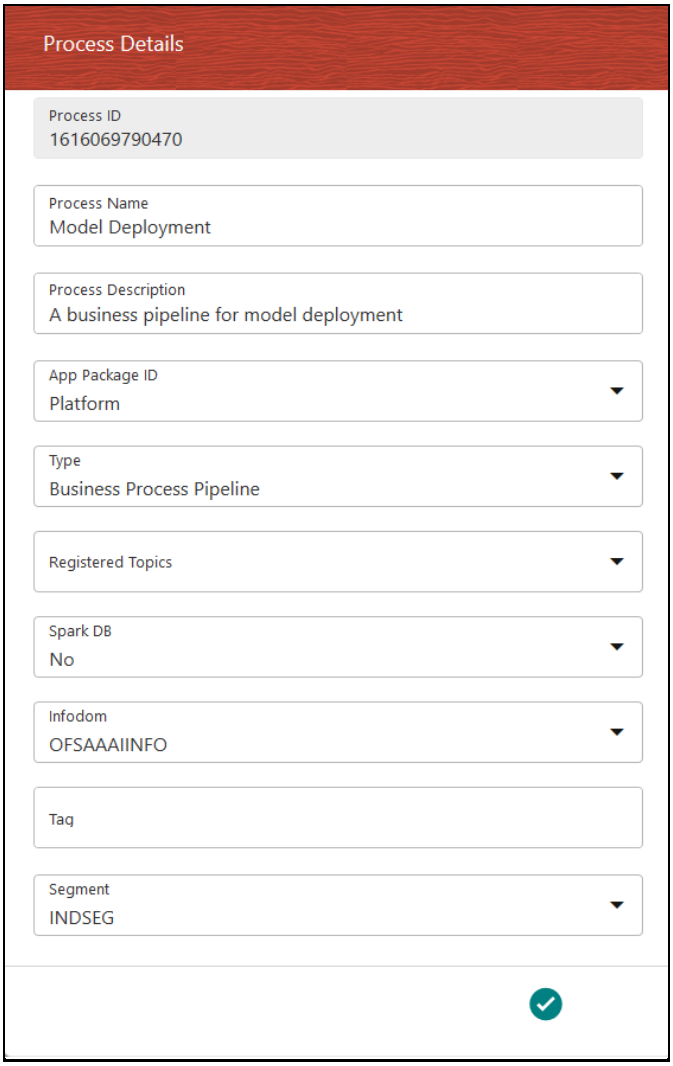
Figure : Business Pipeline Example

In the example shown, we use various activities such as Human Tasks and Service tasks, which are related to each other through transitions. For executing tasks in parallel, we use Parallel Gateways.

5.1.2 Creating a Business Pipeline

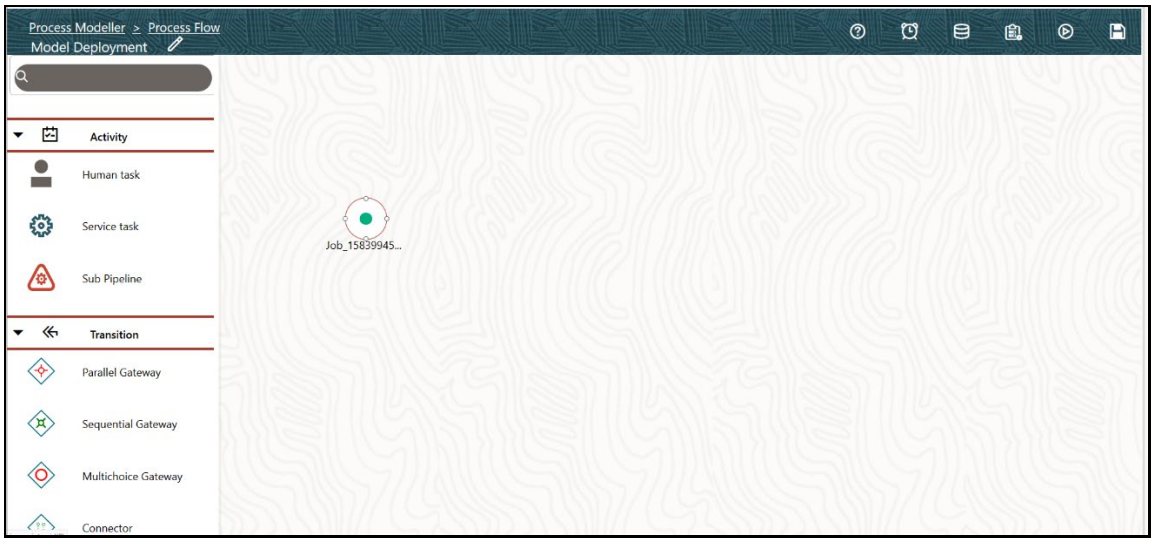
Click  in the **Process Modeller** Summary window.

Figure : Process Details window to create a Business Pipeline



1. Enter a unique Process Name and a description. The system creates the unique Process ID.
2. Select the appropriate app package in which you want to create the process. For more information, see Application Package.
3. Select **Business Process Pipeline** 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 that are mapped to the applications configured in your OFSAA instance.
5. Click the **Accept** icon to save it. The Process Flow canvas is displayed.

Figure : Process Flow Canvas



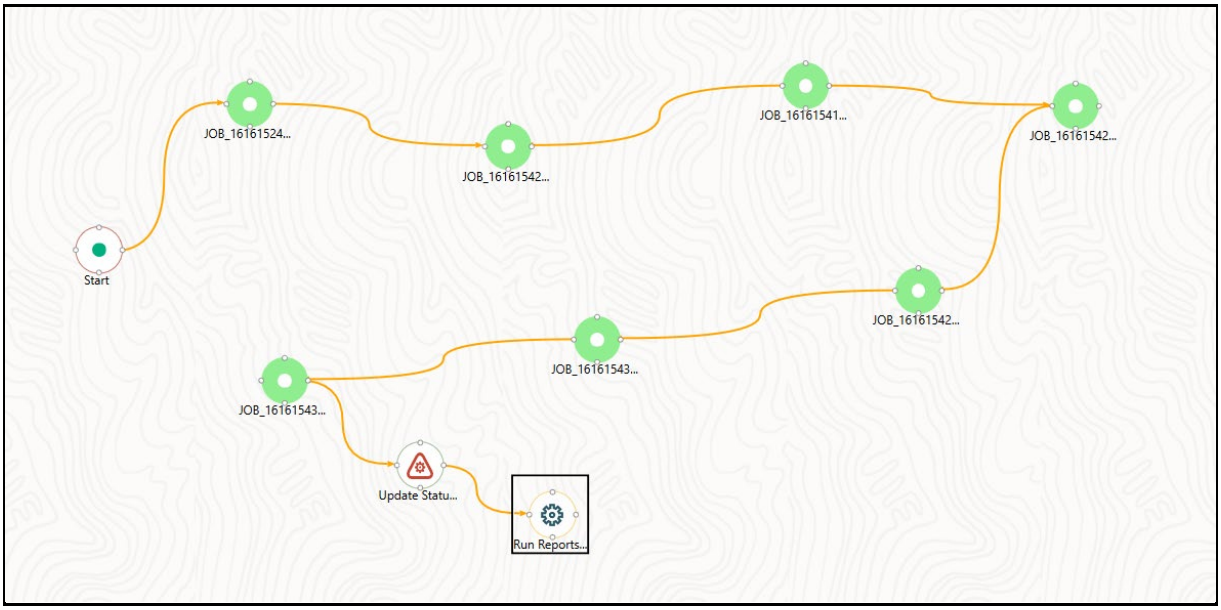
6. By default,  **START** from the toolbar appears. This Start activity indicates the beginning of the Process.
7. Design your Process with various components available in the **Process Flow** tab. For more information on each component, see the Components for Designing Your Process Flow section.

5.2 Run Pipeline

A 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 the construction of a Run Pipeline. Various Widgets that enable the construction of a Run Pipeline are available in the Component toolbar.

5.2.1 An Example of Run Pipeline

Figure : Run Pipeline Example



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 Process Name and a description. The Process ID is a system-generated unique value.
4. Select the appropriate app package in which you want to create the process. For more information, see the Application Package section.
5. Select **Run Pipeline** from the **Type** drop-down list.
6. If you have selected Run Pipeline, set the **Process Execution On Failure**:
   a. **Yes** – Select **Yes** to execute the process, even when the process fails.
   b. **No** - (Default). Select **No** to stop the process execution, during a process failure.

   This step is applicable while you create sub-run pipelines also.
7. 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 that are mapped to the applications configured in your OFSAA instance.
8. Click the **Accept**  icon to save it. The Process Flow canvas is displayed.
9. By default,  **START** from the toolbar appears. This Start activity indicates the beginning of the Process.

5.2.3 Design your Run Pipeline using Widgets

You can construct a Run pipeline using only Widgets.

For details, see Configuring OFSAA Tasks in Your Process Flow.

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 the Calling another Pipeline from Your Parent Pipeline section.

5.2.5 Design your Run Pipeline using combinations of Widgets and Sub Pipeline

You can construct Run pipeline by combining Widgets and Sub pipeline.

5.2.6 Applying Filter Condition on Run Pipeline

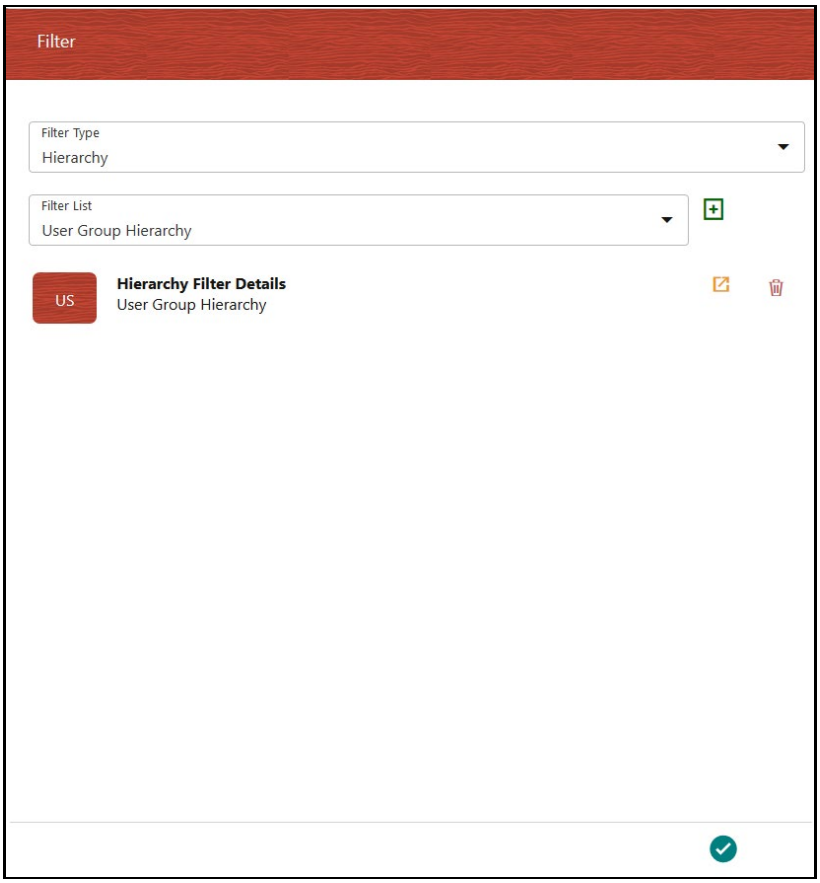
This section details how to apply filter conditions at the Run Pipeline level. You can apply filter conditions on the Sub Pipeline level also.

For more information, see Calling another Pipeline from Your Parent Pipeline.

To apply a filter on a Run Pipeline

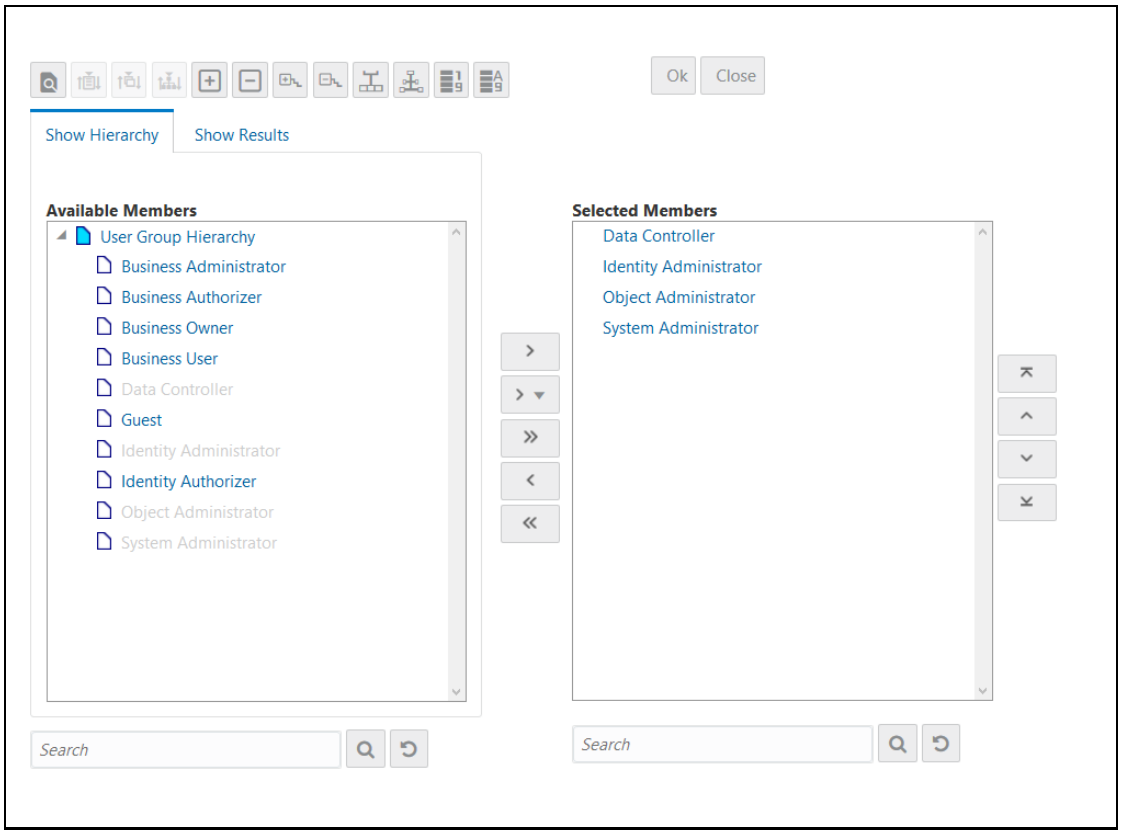
1. From the **Process Modeller Summary** window, click the  submenu icon corresponding to the Run Pipeline for which you want to apply filter condition and click **Filter**. The **Filter Details** window is displayed.

Figure : Filter Details Window



2. Select the **Filter Type** as **Hierarchy** from the drop-down list. Currently, only Hierarchy Filter is supported.
3. Select the Filter from the **Filter List** drop-down list. This list displays all Business Hierarchies defined in the information domain.
4. Click **Add Filter**. The Filter is displayed under the **Hierarchy Filter Details** window.
5. Click to view the **Hierarchy Browser** window and select the hierarchy members.

Figure : Hierarchy Browser Window



6. Select a member/node and click > to select the same. Click > ▾ to select the member as Self, Self Children, Parent, Siblings, and Children.

   For more information, see Hierarchical Member Selection Modes.

   In the **Hierarchy Browser** window you can also:

   - Click to sort members based on the path.
   - Click to sort hierarchy (top to bottom).
   - Click to sort based on level.
   - Click or to expand or collapse the members under a node.
   - Click or to expand a branch or collapse the selected branch.
   - Click to focus only on the selected branch. The **Available Values** pane shows the members of the selected branch only. Click to go back to normal view.
   - Click to display member's numeric codes in the right. The icon changes to .
   - Click to display member's numeric codes on the left. The icon changes to .

- Click  to show only member names. This is the default view. The icon changes to  .
- Click  to display member's alphanumeric codes in the right. The icon changes to  .
- Click  to display member's alphanumeric codes on the left. The icon changes to  .
- Click  to display only member names. This is the default view. The icon changes to  .
- Select a member and click or to re-arrange the members in the **Selected Values** pane.
- Select a member and click to move it to the top or click to move it to the bottom.
- Click  to launch the Search panel. Here you can search based on **Dimension Member Numeric Code**, **Dimension Member Name,** or **Dimension Member Alphanumeric Code**. You can also search in the grid based on member name using the **Search** field.

7. Click **OK** to save.
8. Click  to remove any selected filters.

5.2.6.1 Hierarchical Member Selection Modes

To aid the selection process, certain standard modes are offered through a drop-down. The available modes are **Self**, **Self Children**, **Parent**, **Siblings**, and **Children**.

Based on the hierarchy member security applied, the nodes/members of the hierarchy are displayed in enabled or disabled mode. The members that are in enabled mode only can be selected. That is, the members that are mapped to your user group only can be selected. For example, if you choose **Self Children**, the immediate children of the selected hierarchy that are mapped to your user group only will be moved to the selected pane.

- The **Self** mode is the default mode displayed. In this mode, only the specific member selected in the available pane is selected on the selected pane.
- Choose the **Self Children** mode when you want a specific member and only its immediate children to be selected onto the selected pane.
- Choose the **Parent** mode when you want to select only the parent member of a selected member onto the selected pane.
- Choose the **Siblings** mode when you want to select all the sibling members of the selected member (those members under the same parent) onto the selected pane.
- Choose the **Children** mode when you want only the immediate children of a specific member to be selected onto the selected pane mode.

You can also click to select all the members to the **Selected Values** pane. Click to deselect a selected member from the **Selected Values** pane or click to deselect all the members.

5.2.7 Execute a Pipeline

You can execute a Pipeline using the following options:

- From UI
- Using Command Line Utility
- Using ICC Batch

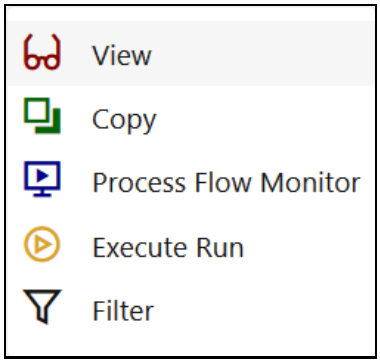
If the option **Process Execution On Failure** is set to **No** while creating/modifying the Run pipeline or a sub-run pipeline, the execution will be stopped if there is a failure.

5.2.7.1 From UI

To execute a Pipeline from UI:

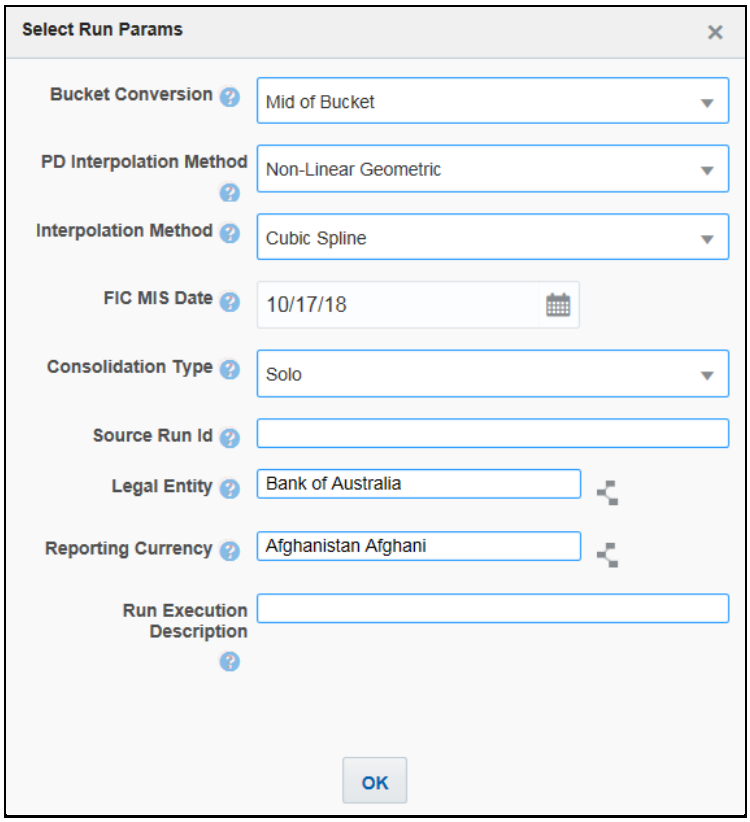
1. From the **Process Modeller Summary** window, click the submenu icon corresponding to the Pipeline you want to execute:

Figure : Sub-menu corresponding to the Run Pipeline



2. Click **Execute Run** to execute the Pipeline. The **Select Run Params** window is displayed.

Figure : Select Run Params window



3. Select values for the Run Parameters and click **OK**.

The execution of the 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 into 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 Pipeline using the command line utility

1. Navigate to `$FIC_DB_HOME/bin` folder.
2. Execute the script file using the following command:

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

   - `processInstanceId` - Instance Id of the Process or Pipeline
   - `processId` - Process ID of the Pipeline. This is a mandatory parameter.
   - `objectId` - This is an auto-generated unique Object ID. Enter `'$objectId'` as mandatory parameter value.
   - `objectType` - Specify the Object Type if it is defined in the aai_wf_app_definition_map table.

- infodom - Information Domain Name
- userID - Specify the user ID
- 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 a comma.
- 'securityparams'- Specify any security parameters within single quotes separated by comma

Example for multiple legal entities:

```
./wfExecExternal.sh null 1492150193532 testRunStageDet null ERMINFO
ERMUSER IFRSEG en_US '{"hierDetailsLE":[
{"hierValueCode":"MBB","value":"MBB",
"hierNodeCode":"[HLLFP029].[MBBG].[MBB]","
leafCondition":"CASE WHEN DIM_ORG_STRUCTURE.f_latest_record_indicator =
'Y' THEN DIM_ORG_STRUCTURE.v_entity_code END='AU'",
"hierNodeDesc":"MBB"},
{"hierValueCode":"MSL","value":"MSL","hierNodeCode":"[HLLFP029].[MBBG].
[MSL]","leafCondition":"CASE WHEN
DIM_ORG_STRUCTURE.f_latest_record_indicator = 'Y' THEN
DIM_ORG_STRUCTURE.v_entity_code END='AU'","hierNodeDesc":"MBL"}],
"CONSOTYPE":"S","FIC_MIS_DATE":"2019-04-
19","V_RUN_MAIN_DESC":"TestStageDetRun"}' null
```

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

5.2.7.3 Using ICC Batch

To execute a Pipeline from the **Operations** menu as an ICC batch, the ./pipelineExecutor.sh script file is made available in the $FIC_DB_HOME/bin directory. You must enter the specified parameters in the ./pipelineExecutor.sh file required for **Run Pipeline** execution and then create an ICC Batch to call this script file through the Run Executable component.

To update the parameters in the ./pipelineExecutor.sh script file for Run Pipeline execution, follow these steps:

1. Navigate to the $FIC_DB_HOME/bin directory.
2. Update the script file by modifying the values that are enclosed within the ## symbols. The following is an example for the format:

   ```
   pmfscriptexe="./wfExecExternal.sh null '##PROCESS_NAME##' $objectId
   null '##INFODOM##' '##SEGMENT##' '##USER##' en_US '{##EXECUTION PARAM
   JSON##, \"FIC_MIS_DATE\":\"$datevar\"}' null"
   ```

   The following list provides descriptions of the script file parameters:

   - Process_Name - Enter the process name of the Pipeline. This is a required parameter.

- `$objectId` - This is an auto-generated unique Object ID that you must not modify. Retain `$objectId` in the script as this mandatory parameter is created by the PMF application during the execution of the Run Pipeline.
- `INFODOM` - Enter the Information Domain name here.
- `SEGMENT`- Enter the Segment name here.
- `USER` - Enter the user ID here.
- `locale` - Modify the locale as required. For example, for English-United States, enter en_US.
- `EXECUTION PARAM JSON` - Enter the Run Execution parameters in JSON format.

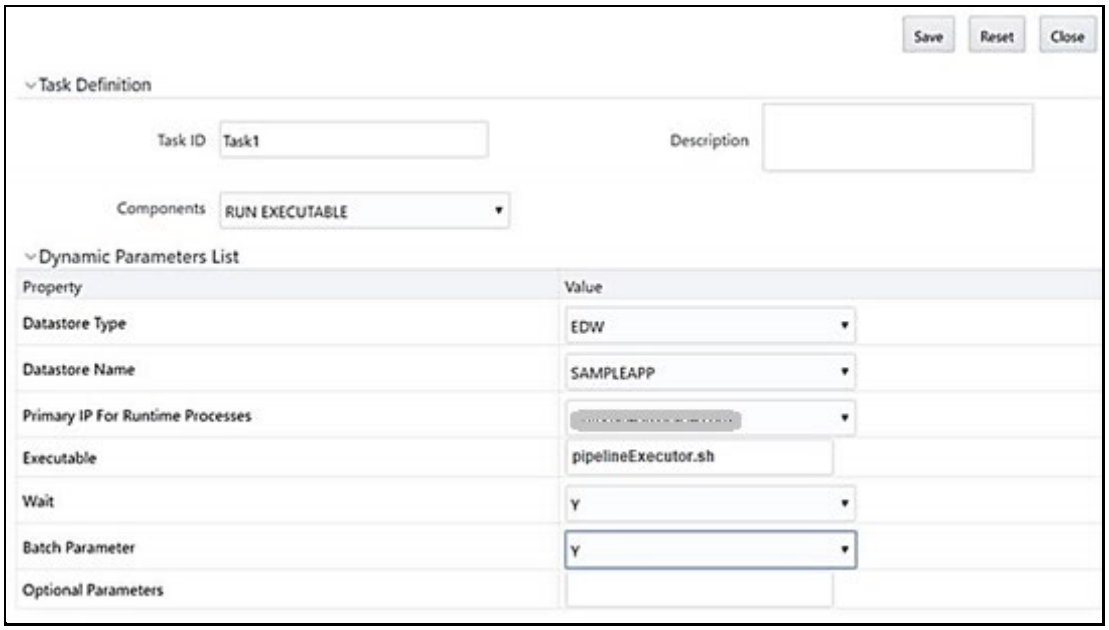
  The following code snippet is an example:

```
{\"hierDetailsLE\":[{\"hierValueCode\":\"ORG_NAME\",\"value\":\"ORG_
NAME\",\"hierNodeCode\":\"ORG_NAME\",\"leafCondition\":\"CASE WHEN
DIM_ORG_STRUCTURE.f_latest_record_indicator = 'Y' THEN
DIM_ORG_STRUCTURE.v_entity_code
END='ORG_NAME'\",\"hierNodeDesc\":\"ORG_NAME\"}],\"CONSOTYPE\":\"SOL
O\",\"hierDetailsRCY\":[{\"hierValueCode\":\"INR\",\"value\":\"India
n
Rupee\",\"hierNodeCode\":\"INR\",\"leafCondition\":\"DIM_CURRENCY.v_
iso_currency_cd = 'INR'\",\"hierNodeDesc\":\"Indian
Rupee\"}],hierDetailsUDRS: [{\"hierValueCode\": \"BISNONSECSTD-
SETUP1\", \"value\": \"BIS Non Sec STD - Setup
1\",\"hierNodeCode\":\"BISNONSECSTD-SETUP1\",\"leafCondition\":
\"USR_DEFINED_RUN_PARAMETERS.V_RUN_PARAMETER_SETUP_CODE =
'BISNONSECSTD-SETUP1'\", hierNodeDesc: \"BIS Non Sec STD - Setup
1\"}],\"FIC_MIS_DATE\":\"$datevar\",\"V_RUN_MAIN_DESC\":\"RUN MARCH
09\"
```

- `$datevar` - Retain `$datevar` variable in the execution param JSON specified in the preceding list without any modification as this gets replaced by the FIC_MIS_DATE selected during batch execution as the date value for the Run execution.

3. Create a new batch in the **Batch Maintenance** window in the **Operations** menu.
   For more information, see the **Adding Batch Definition** section in the OFS Analytical Applications Infrastructure User Guide.
4. Create a new task with the **Run Executable** task component.
   For more information, see the **Adding Task Details** section in the OFS Analytical Applications Infrastructure User Guide.

Figure : Call pipelineExecutor.sh in the Executable Field



- Call the `pipelineExecutor.sh` script file in the **Executable** field.
- Select **Y** in the **Batch Parameter** drop-down list for all cases.
- Select **Y** in the **Wait** drop-down list if you want the Run executable to wait to finish the task execution and then update the task status.

5. Click **Save** to execute the Batch.
6. Schedule a batch through the **Batch Scheduler** window in the **Operations** menu or schedule and execute the batch through the External Scheduler (ESIC).
   For more information, see the **Batch Scheduler** and **External Scheduler Interface Component** sections in the OFS Analytical Applications Infrastructure User Guide.

5.2.7.4 ICC BATCH CALLS AND Associated PMF EXECUTION IDENTIFIER LINKs

Execute the following query to know the PMF process instances executions and the associated ICC batch execution IDs.

```
SELECT DISTINCT PROCESSDET.V_PROCESS_NAME PROCESSNAME,
QUE.V_PROCESS_INSTANCE_ID PROCESSINSTANCEID,
COALESCE(DATA.V_ACTIVITY_ID,QUE.V_ACTIVITY_ID) JOBID,
COALESCE(ACTIVITYDET.V_ACTIVITY_NAME,ACTIVITYDET.V_ACTIVITY_DESC) JOBNAME,
COALESCE(DATA.STARTDATE,QUE.D_EXECUTION_START_TIME) STARTTIME,
COALESCE(DATA.ENDDATE,QUE.D_EXECUTION_END_TIME) ENDTIME,
NVL(QUE.V_EXECUTION_STATUS,QUE.V_STATUS) JOBSTATUS, DATA.V_BATCH_RUN_ID,
DATA.V_TASK_DESCRIPTION,DATA.V_BATCH_DESCRIPTION,
DATA.V_INFO_DATE, DATA.V_BATCH_STATUS, DATA.V_BATCH_ID,
DATA.V_GROUP_CODE, DATA.APP_BATCH_RUN_ID, MAP.V_OBJECT_ID,
AWPE.V_INSTANCE_PROPERTY

FROM AAI_WF_PROCESS_ENGINE_ACT_QUE QUE
```

```
INNER JOIN (SELECT PROCESSB.V_PROCESS_ID, PROCESSTL.V_PROCESS_NAME
FROM AAI_WF_PROCESS_B PROCESSB, AAI_WF_PROCESS_TL PROCESSTL
WHERE PROCESSB.V_PROCESS_ID = PROCESSTL.V_PROCESS_ID
AND PROCESSTL.V_LOCALE_CODE = 'en_US'
) PROCESSDET
ON (PROCESSDET.V_PROCESS_ID = SUBSTR (QUE.V_PROCESS_INSTANCE_ID,0,INSTR
QUE.V_PROCESS_INSTANCE_ID,'_',-1) -1))
INNER JOIN AAI_WF_APP_INSTANCE_MAP MAP
ON (MAP.V_PROCESS_INSTANCE_ID = QUE.V_PROCESS_INSTANCE_ID AND
AP.V_PROCESS_INSTANCE_ID = '1702359232479_01991722-baf2-4f08-ba61-
8cd2d86e12a' )
INNER JOIN AAI_WF_PROCESS_EXECUTION AWPE
ON (AWPE.V_PROCESS_ID = PROCESSDET.V_PROCESS_ID AND
WPE.V_PROCESS_INSTANCE_ID = '1702359232479_01991722-baf2-4f08-ba61-
8cd2d86e12a' )

INNER JOIN (
SELECT ACTB.V_PROCESS_ID, ACTTL.V_ACTIVITY_NAME, ACTB.V_ACTIVITY_ID,
ACTTL.V_ACTIVITY_DESC, ACTTL.V_LOCALE_CODE
FROM AAI_WF_ACTIVITY_B ACTB, AAI_WF_ACTIVITY_TL ACTTL
WHERE ACTB.V_PROCESS_ID = ACTTL.V_PROCESS_ID
AND ACTB.V_ACTIVITY_ID = ACTTL.V_ACTIVITY_ID
AND ACTB.V_EXECUTION_MODE != 'START'
AND ACTTL.V_LOCALE_CODE = 'en_US'
AND ACTB.V_PROCESS_ID = ACTTL.V_PROCESS_ID
) ACTIVITYDET
ON ( ACTIVITYDET.V_PROCESS_ID = SUBSTR (QUE.V_PROCESS_INSTANCE_ID,0,INSTR
QUE.V_PROCESS_INSTANCE_ID,'_',-1) -1)
AND ACTIVITYDET.V_ACTIVITY_ID = QUE.V_ACTIVITY_ID )

INNER JOIN (
SELECT DISTINCT WFTASK.V_PROCESS_INSTANCE_ID, WFTASK.V_ACTIVITY_ID,
WFTASK.V_TASK_REF_ID,
WFTASK.V_IS_PROCESSED, WFTASK.D_DATE_PROCESSED, BATCHINFO.V_BATCH_RUN_ID,
BATCHINFO.V_TASK_DESCRIPTION,
BATCHINFO.V_BATCH_DESCRIPTION, BATCHINFO.V_INFO_DATE,
BATCHINFO.V_BATCH_STATUS, BATCHINFO.V_BATCH_ID,
BATCHINFO.V_GROUP_CODE, BATCHINFO.STARTDATE, BATCHINFO.ENDDATE,
BATCHINFO.APP_BATCH_RUN_ID

FROM AAI_WF_ASYNC_TASK_POLL_STATUS WFTASK
INNER JOIN (
SELECT DISTINCT BR.V_BATCH_RUN_ID, BT.V_TASK_DESCRIPTION,
V_BATCH_DESCRIPTION,
V_INFO_DATE, V_BATCH_STATUS, V_BATCH_ID, V_GROUP_CODE, IM.STARTDATE,
IM.ENDDATE,
REGEXP_REPLACE(BT.V_BATCH_RUN_ID,'','' || BT.V_TASK_ID || '_',1,1)
APP_BATCH_RUN_ID
FROM BATCH_RUN BR, BATCH_TASK BT,
( SELECT MSGLOG.V_BATCH_RUN_ID, MAX(MSGLOG.TIMESTAMP) ENDDATE,
IN(MSGLOG.TIMESTAMP) STARTDATE
FROM ICC_MESSAGELOG MSGLOG GROUP BY V_BATCH_RUN_ID) IM
WHERE BR.V_BATCH_RUN_ID = BT.V_BATCH_RUN_ID AND BR.V_BATCH_RUN_ID =
M.V_BATCH_RUN_ID
) BATCHINFO
```

```
ON (BATCHINFO.V_BATCH_ID = WFTASK.V_TASK_REF_ID) WHERE
FTASK.V_PROCESS_INSTANCE_ID = '1702359232479_01991722-baf2-4f08-ba61-
8cd2d86e12a'

UNION
SELECT DISTINCT WFTASKHIST.V_PROCESS_INSTANCE_ID, WFTASKHIST.V_ACTIVITY_ID,
WFTASKHIST.V_TASK_REF_ID,
WFTASKHIST.V_IS_PROCESSED, WFTASKHIST.D_DATE_PROCESSED,
BATCHINFO.V_BATCH_RUN_ID, BATCHINFO.V_TASK_DESCRIPTION,
BATCHINFO.V_BATCH_DESCRIPTION, BATCHINFO.V_INFO_DATE,
BATCHINFO.V_BATCH_STATUS, BATCHINFO.V_BATCH_ID,
BATCHINFO.V_GROUP_CODE, BATCHINFO.STARTDATE, BATCHINFO.ENDDATE,
BATCHINFO.APP_BATCH_RUN_ID
FROM AAI_WF_TASK_POLL_STATUS_HIST WFTASKHIST
INNER JOIN (
SELECT DISTINCT BR.V_BATCH_RUN_ID, BT.V_TASK_DESCRIPTION,
V_BATCH_DESCRIPTION, V_INFO_DATE, V_BATCH_STATUS, V_BATCH_ID,
V_GROUP_CODE, IM.STARTDATE, IM.ENDDATE,
REGEXP_REPLACE(BT.V_BATCH_RUN_ID,'','' || BT.V_TASK_ID || '_',1,1)
APP_BATCH_RUN_ID

FROM BATCH_RUN BR, BATCH_TASK BT,
( SELECT MSGLOG.V_BATCH_RUN_ID, MAX(MSGLOG.TIMESTAMP) ENDDATE,
MIN(MSGLOG.TIMESTAMP) STARTDATE

FROM ICC_MESSAGELOG MSGLOG GROUP BY V_BATCH_RUN_ID ) IM
WHERE BR.V_BATCH_RUN_ID = BT.V_BATCH_RUN_ID AND BR.V_BATCH_RUN_ID =
IM.V_BATCH_RUN_ID) BATCHINFO

ON (BATCHINFO.V_BATCH_ID = WFTASKHIST.V_TASK_REF_ID) WHERE
WFTASKHIST.V_PROCESS_INSTANCE_ID = '1702359232479_01991722-baf2-4f08-ba61-
88cd2d86e12a'
) DATA
ON (DATA.V_PROCESS_INSTANCE_ID = QUE.V_PROCESS_INSTANCE_ID AND
DATA.V_TASK_DESCRIPTION = QUE.V_ACTIVITY_ID)
WHERE QUE.V_PROCESS_INSTANCE_ID = '1702359232479_01991722-baf2-4f08-ba61-
88cd2d86e12a'

UNION

SELECT DISTINCT PROCESSDET.V_PROCESS_NAME PROCESSNAME,
MAP.V_PROCESS_INSTANCE_ID PROCESSINSTANCEID,

QUE.V_ACTIVITY_ID JOBID,
NVL(ACTIVITYDET.V_ACTIVITY_NAME,ACTIVITYDET.V_ACTIVITY_DESC) JOBNAME,

QUE.D_EXECUTION_START_TIME STARTTIME, QUE.D_EXECUTION_END_TIME ENDTIME,
NVL(QUE.V_EXECUTION_STATUS,QUE.V_STATUS) JOBSTATUS,

'' V_BATCH_RUN_ID, '' V_TASK_DESCRIPTION, '' V_BATCH_DESCRIPTION, ''
V_INFO_DATE, '' V_BATCH_STATUS,

'' V_BATCH_ID, '' V_GROUP_CODE, '' APP_BATCH_RUN_ID, MAP.V_OBJECT_ID,
AWPE.V_INSTANCE_PROPERTY
```

```
FROM AAI_WF_APP_INSTANCE_MAP MAP
LEFT OUTER JOIN AAI_WF_APP_DEFINITION_MAP DEFN ON (DEFN.V_OBJECT_TYPE =
MAP.V_OBJECT_TYPE)

INNER JOIN AAI_WF_PROCESS_ENGINE_ACT_QUE QUE ON (QUE.V_PROCESS_INSTANCE_ID =
MAP.V_PROCESS_INSTANCE_ID)

INNER JOIN (
SELECT PROCESSB.V_PROCESS_ID, PROCESSTL.V_PROCESS_NAME
FROM AAI_WF_PROCESS_B PROCESSB, AAI_WF_PROCESS_TL PROCESSTL
WHERE PROCESSB.V_PROCESS_ID = PROCESSTL.V_PROCESS_ID
AND PROCESSTL.V_LOCALE_CODE = 'en_US'
)PROCESSDET
ON (PROCESSDET.V_PROCESS_ID = SUBSTR (QUE.V_PROCESS_INSTANCE_ID,0,INSTR
(QUE.V_PROCESS_INSTANCE_ID,'_',-1) -1) )

INNER JOIN (
SELECT ACTB.V_PROCESS_ID, ACTTL.V_ACTIVITY_NAME, ACTB.V_ACTIVITY_ID,
ACTTL.V_ACTIVITY_DESC, ACTTL.V_LOCALE_CODE

FROM AAI_WF_ACTIVITY_B ACTB, AAI_WF_ACTIVITY_TL ACTTL
WHERE ACTB.V_PROCESS_ID = ACTTL.V_PROCESS_ID
AND ACTB.V_ACTIVITY_ID = ACTTL.V_ACTIVITY_ID
AND ACTB.V_EXECUTION_MODE != 'START'
AND ACTTL.V_LOCALE_CODE = 'en_US'
AND ACTB.V_PROCESS_ID = ACTTL.V_PROCESS_ID
) ACTIVITYDET
ON (ACTIVITYDET.V_PROCESS_ID = SUBSTR (QUE.V_PROCESS_INSTANCE_ID,0,INSTR
(QUE.V_PROCESS_INSTANCE_ID,'_',-1) -1) AND ACTIVITYDET.V_ACTIVITY_ID =
QUE.V_ACTIVITY_ID )

INNER JOIN AAI_WF_PROCESS_EXECUTION AWPE
ON (AWPE.V_PROCESS_ID = SUBSTR (QUE.V_PROCESS_INSTANCE_ID,0,INSTR
(QUE.V_PROCESS_INSTANCE_ID,'_',-1) -1) AND AWPE.V_PROCESS_ID =
PROCESSDET.V_PROCESS_ID AND AWPE.V_PROCESS_INSTANCE_ID =
'1702359232479_01991722-baf2-4f08-ba61-88cd2d86e12a' )
WHERE MAP.V_PROCESS_INSTANCE_ID = '1702359232479_01991722-baf2-4f08-ba61-
88cd2d86e12a'
ORDER BY JOBID, JOBNAME, STARTTIME, ENDTIME;
```

5.2.8 Abort a Pipeline

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

5.2.8.1 To abort a Pipeline using User Interface

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

5.2.9 Resume a Pipeline

You can resume a Pipeline which has not been executed successfully or which has been explicitly interrupted, or canceled, or put on hold during the execution process. By resuming a Pipeline, you can

continue its execution directly from the point of interruption or failure and complete executing the remaining tasks.

5.2.9.1 To Resume a Pipeline using User Interface

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

5.2.10 Re-run a Pipeline

You can re-run a Pipeline that was previously executed, irrespective of the previous execution state.

5.2.10.1 To Re-Run a Pipeline using User Interface

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

5.2.11 Invoke PMF Workflow Functions using API

This section details you to invoke PMF RESTful APIs (from external systems) that are packaged in the OFSAA Applications

Prerequisite:

You must get the bearer token before invoking any PMF function using an API.

For more details on how to get the bearer token, see **Instance Access Token** section of OFS Analytical Applications Infrastructure User Guide.

Authorization:

The following table lists the Authorization details.

Type	Required	Value
`Bearer Token`	Yes	Bearer token generated value as a prerequisite

5.2.11.1 Execute Pipeline

End Point Details:

- Method – `POST`
- REST Endpoint - `rest-api/v1/PMFService/execute`

The following table lists the Request Headers.

Name	Type	Required	Value
`Content-Type`	String	Yes	`application/json`

Name	Type	Required	Value
ofs_remote_user	String	Yes	Login ID of the user Note: The User ID or Service accounts are "SMS Auth Only" in case of SSO and LDAP configured setups.

Request JSON Parameters:

The following table lists the Request Body.

Name	Type	Description
SummaryPayload	Object	This is an object with the properties mentioned below.
objectID	String	Specify an unique string identifier for the execution. The Object ID can have a maximum of 20 characters. Allowed characters are alphabets , numbers, Hyphen(-), and Underscore(_). No other special characters are allowed. Example: ProcessExecObj1
processId	String	Process ID of the pipeline For Example: RNBISPMF002
objectType	String	Specify Object type if exists or null. For Example: null
infodom	String	Information Domain Name. For Example: OFSA829INF3
locale	String	Locale selected. For Example: 'en_US'
Securitymap	Object	Specify security map if exists or null
applicationparams	Object	Specify values for the run time execution parameters Specify values for the Run execution parameters in JSON format..

Request JSON Sample:

```
/rest-api/v1/PMFService/execute
{
"SummaryPayload": {
"objectId": "ProcessExecObj1",
"processId": "RNBISPMF002",
```

```
"objectType": "null",

"locale": "en_US",

"infodom":"OFSA829INF3",

"securitymap" :{},

"applicationparams":{"FIC_MIS_DATE":"2020-04-
01","BASEL_CONF_ID":"22","CONSOTYPE":"SOLO","IS_BACKDATED_EXEC":"Y","SIGCUR"
:"YES","EXP_DATA":"INCR"}

}

}
```


Response JSON Parameters:

This section provides the list of parameters in the JSON Response.

The following table provides details for the query parameters returned in the response body.

Name	Type	Description
processInstanceId	String	Instance Id of the process being executed
stopExecutionFlag	String	Flag indicates if the execution stopped due to internal failure
processStatus	String	It indicates status of the process initiation
processID	String	Process ID of the pipeline
workflowStatus	String	Status of the process initiation
lastActivity	String	It is the next activity ID in the process to be executed
responseJSON	String	Response of a particular task, if any
responseStatus	String	Rest API standard status

Response JSON Sample


```
{
    "payload": {
        "processInstanceID": "1634536080515_3f80680b-3276-42ac-a41c-
a573d0697c23",
        "stopExecutionFlag": false,
        "processStatus": "SUCCESS",
```

```
    "processID": "1634536080515",
    "workflowStatus": "COMPLETED",
    "lastActivity": "Job_1583994521890",
    "responseJSON": "",
    "responseStatus": "SUCCESS"
}
```

5.2.11.2 Resume a Pipeline

End Point Details:

- Method – POST
- REST Endpoint - `/rest-api/v1/PMFService/resumeprocess`

The following table lists the Request Headers.

Name	Type	Required	Value
`Content-Type`	String	Yes	application/json
`ofs_remote_user`	String	Yes	Login ID of the user. Note: The User ID or Service accounts are "SMS Auth Only" in case of SSO and LDAP configured setups.

Request JSON Parameters:

The following table lists the Request Body.

Name	Type	Description
`processInstanceId`	STRING	Instance Id of the Process being resumed. For Example: "`PMFINV024_2743749b-32c6-4cc5-b58b-dda49c3d1f4d`"
`Locale`	STRING	Locale selected. For Example: 'en_US
`infodom`	STRING	Information Domain Name. For Example: "`OFSA829INF3`"

Request JSON Sample:

```
/rest-api/v1/PMFService/resumeprocess
{
"processInstanceId": "PMFINV024_2743749b-32c6-4cc5-b58b-dda49c3d1f4d",
"locale": "en_US",
"infodom": "OFSA829INF3"
```

```
}
```

Response JSON Parameters:

This section provides the list of parameters in the JSON Response**.**

The following table provides details for the query parameters returned in the response body.

Name	Type	Description
`processInstanceId`	STRING	Instance Id of the Process being resumed.
`message`	STRING	It indicates the process message description.
`status`	STRING	Status of the process execution.

Response JSON Sample

```
{
    "message": "Process has been resumed",
    "status": true
}
```

5.2.11.3 Abort a Pipeline

End Point Details:

- Method – POST
- REST Endpoint - `/rest-api/v1/PMFService/abortprocess`

The following table lists the Request Headers.

Name	Type	Required	Value
`Content-Type`	String	Yes	application/json
`ofs_remote_user`	String	Yes	Login ID of the user Note: The User ID or Service accounts are "SMS Auth Only" in case of SSO and LDAP configured setups.

Request JSON Parameters:

The following table lists the Request Body.

Name	Type	Description
processInstanceId	STRING	Instance Id of the Process being aborted. For Example: "PMFINV024_2743749b-32c6-4cc5-b58b-dda49c3d1f4d"
Locale	STRING	Locale selected. For Example: 'en_US
infodom	STRING	Information Domain Name. For Example: "OFSA829INF3"

Request JSON Sample:

```
/rest-api/v1/PMFService/abortprocess

{

"processInstanceId": "PMFINV024_2743749b-32c6-4cc5-b58b-dda49c3d1f4d",

"locale": "en_US",

"infodom": "OFSA829INF3"

}
```

Response JSON Parameters:

This section provides the list of parameters in the JSON Response.

The following table provides details for the query parameters returned in the response body.

Name	Type	Description
processInstanceId	STRING	Instance Id of the Process being aborted.
message	STRING	It indicates the process message description.
status	STRING	Status of the process execution.

Response JSON Sample

```
{

"message": "Abort workflow success..",

"status": true

}
```

5.2.11.4 Re-run a Pipeline

End Point Details:

- Method – POST
- REST Endpoint - rest-api/v1/PMFService/rerunprocess

The following table lists the Request Headers.

Name	Type	Required	Value
Content-Type	String	Yes	application/json
ofs_remote_user	String	Yes	Login ID of the user Note: The User ID or Service accounts are "SMS Auth Only" in case of SSO and LDAP configured setups.

Request JSON Parameters:

The following table lists the Request Body.

Name	Type	Description
processInstanceId	STRING	Instance Id of the Process being Rerun For Example: "PMFINV024_2743749b-32c6-4cc5-b58b-dda49c3d1f4d"
Locale	STRING	Locale selected. For Example: 'en_US
infodom	STRING	Information Domain Name. For Example: "OFSA829INF3"

Request JSON Sample:

```
/rest-api/v1/PMFService/rerunprocess
{
"processInstanceId": "PMFINV024_2743749b-32c6-4cc5-b58b-dda49c3d1f4d",
"locale": "en_US",
"infodom": "OFSA829INF3"
}
```

Response JSON Parameters:

This section provides the list of parameters in the JSON Response**.**

The following table provides details for the query parameters returned in the response body.

Name	Type	Description
processInstanceId	STRING	Instance Id of the Process being Rerun.
message	STRING	It indicates the process message description.
status	STRING	Status of the process execution.

Response JSON Sample

```
{
PROCESS_INSTANCE_ID": "1661750871424_7ab3f4a7-4eef-409e-aa9c-6345bb9e4a4e",
"message": "Process has been Rerun",
 status": true
}
```

5.2.11.5 Status API

The Status feature provides execution status of the given process instance IDs.

End Point Details:

- Method – POST
- REST Endpoint - /rest-api/v1/PMFService/execute/status

The following table lists the Request Headers.

Name	Type	Required	Value
Content-Type	String	Yes	application/json

Request JSON Parameters:

The following table lists the Request Body.

Name	Type	Description
processInstanceId	STRING	Process Instance ID of the PMF Execution

Request JSON Sample:

```
/rest-api/v1/PMFService/execute/status

{
"processInstanceIdList":[

  DMI_workflow_517e7b92-175d-4190-8718-9a759298a49d",
  "1660304393933_8e550687-d2df-4481-85ee-5a0737122a1e"
```

```
  ]
}
```

Response JSON Parameters:

This section provides the list of parameters in the JSON Response**.**

The following table provides details for the query parameters returned in the response body.

Name	Type	Description
processInstanceId	STRING	Instance Id of the Process being Rerun.
status	STRING	Status of the process execution.

Response JSON Sample

```
{"processExecutionStatus":[{"process_instance_id":"PMFINV024_7fb3e9fc-fb88-4caf-9200-4691095f165b","status":"RUNNING"}]}
```

5.3 Additional Functionalities

This section describes the additional functionalities that can be performed in the **Process Modeler** window.

Topics:

- Modifying a Pipeline
- Viewing a Pipeline
- Copying a Pipeline
- Deleting a Pipeline

5.3.1 Modifying a Pipeline

This option allows you to modify a Pipeline.

To modify a pipeline perform the following:

1. From the **Process Modeler** window, search for the Pipeline and click the Pipeline name. The **Process Flow** tab is displayed.
2. Modify the Process Flow, Definition, Application Rules, and Data Fields as required.
3. If you are modifying the Run Pipeline, set the **Process Execution On Failure**:
    a. **Yes** – Select **Yes** to execute the process, even when the process fails.
    b. **No** - (Default). Select **No** to stop the process execution, during a process failure.

NOTE

When you try to delete a component in the Business Pipeline that is used as a Sub Pipeline, a confirmation message is displayed.

After you click **OK** to confirm the deletion, an alert is displayed with the list of the Business Pipelines where the Sub Pipeline is used.

You should click **OK** again to delete the component.

5.3.2 Viewing a Pipeline

This option allows you to view the workflow of an already existing Business Process.

From the **Process Modeler** window, click  to view the sub-menu and select **View**. You can view the Process Flow of the Pipeline.

5.3.3 Copying a Pipeline

This option facilitates you to quickly create a new Business Process based on an existing Process by updating the Process flow or other required details.

To copy a Business Process

1. From the **Process Modeller** window, click the  submenu icon to view the sub-menu and select **Copy**. The **Process Details** window is displayed.
2. Enter a unique **Process ID**, **Process Name,** and **Process Description**. If you select the same App Package ID, then Data Fields and Application Rules are also copied.
3. Select the newly created Process and click the Process Name. Modify the Process flow and other details if required.

5.3.4 Deleting a Pipeline

This option allows you to delete a Business Pipeline or Run Pipeline.

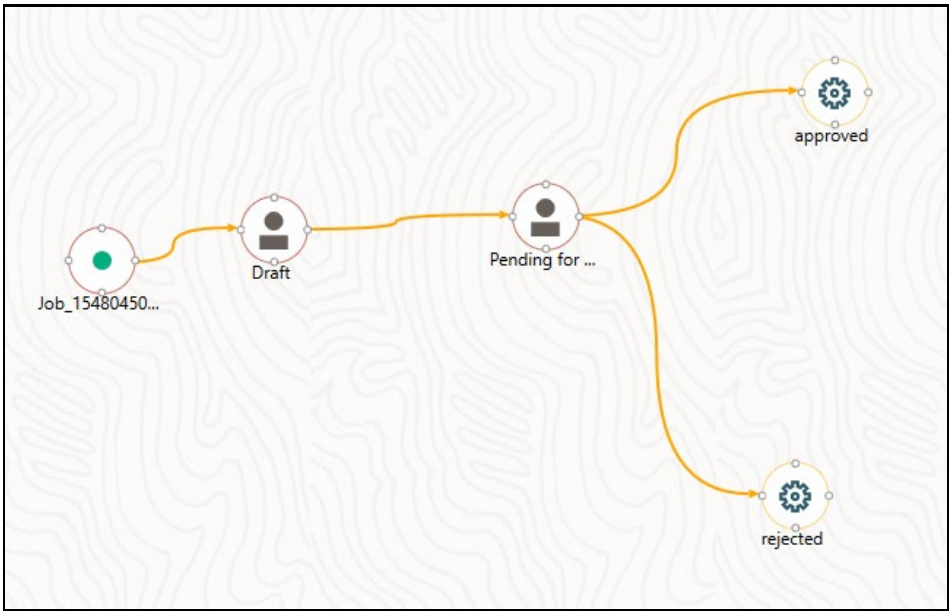
To delete a Pipeline perform the following:

1. From the **Process Modeller** window, click the  icon corresponding to the Pipeline you want to delete.
2. A confirmation message is displayed.
3. Click  **Accept** to confirm the deletion. If this Pipeline is used as a Sub Pipeline in other Business Pipelines, an alert is displayed with the list of Business Pipelines where this is used.
4. Click **Delete Anyway** to delete the Pipeline or click **Cancel** to cancel the delete operation.

6 Human Tasks

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

Figure : Human Tasks Flow



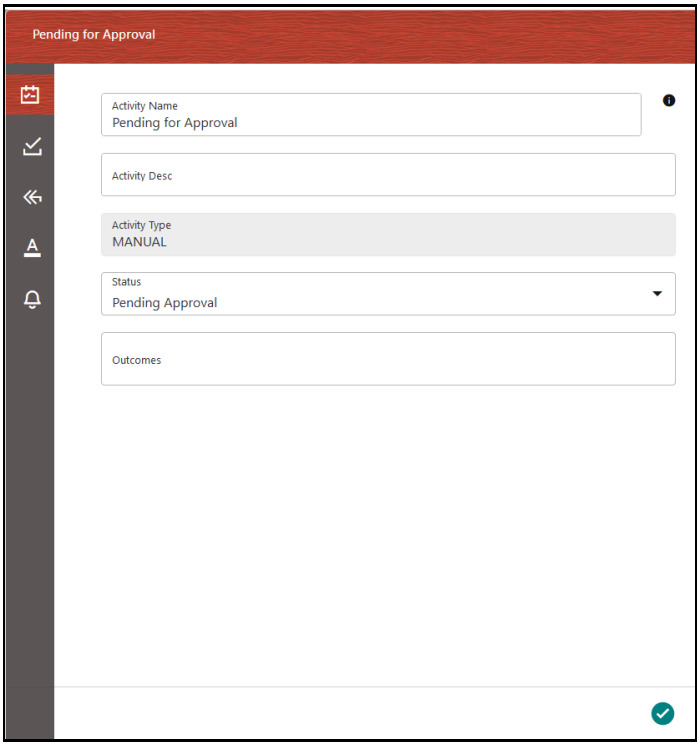
Topics:

6.1 How to Use Human Task

1. In the **Process Flow** window, select the  Human Task icon from **Activity** in the toolbar and Drag and drop it on the drawing canvas where you want to place it.
2. Double-click the  icon on the canvas to display the **Activity** window is displayed.

6.1.1 Activity Window

Figure : Activity Tab



1. Enter Activity Name and Description.

   Mouse over the **?** icon to view the **Activity ID**.

2. 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 Window to Create Tasks and Notifications

Action or 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. A task can be assigned to a user, user group, users with a particular role, or users selected by an application rule. The task is sent to the Application Inbox of the assigned users.

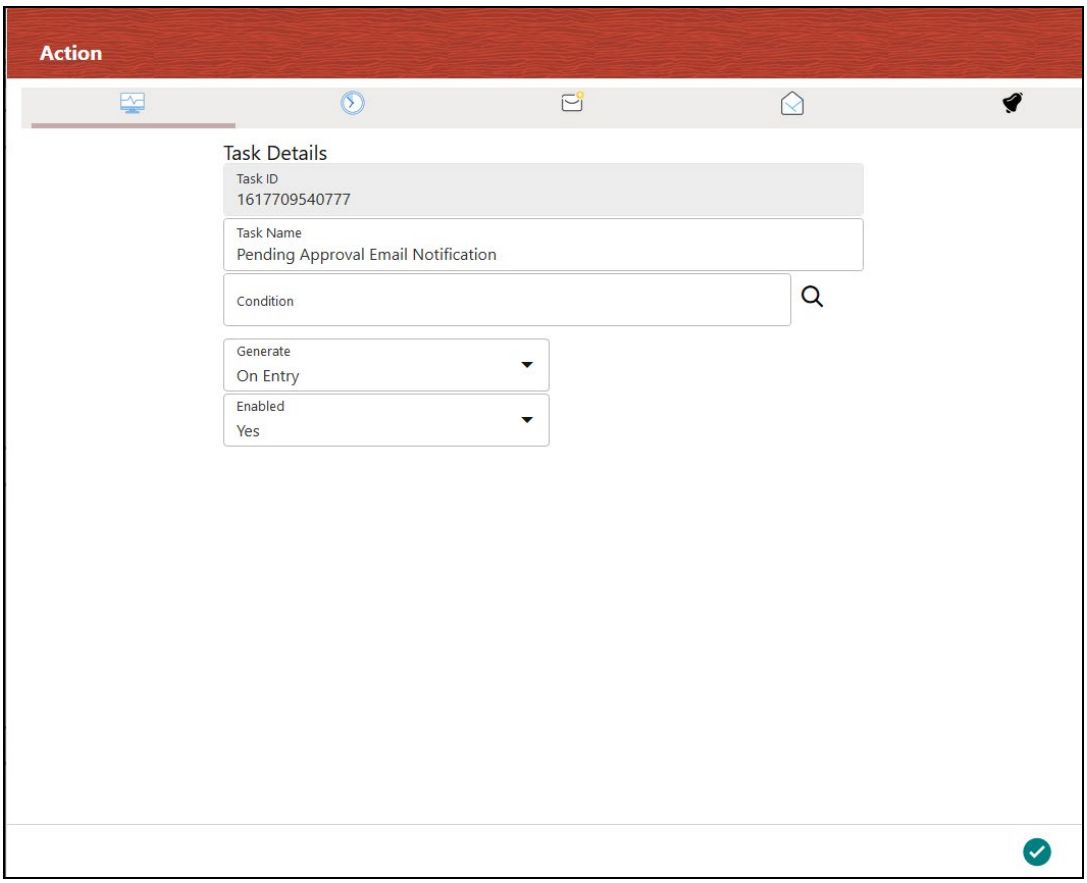
1. Click the Action icon to define Actions in the Action window.

Figure : Action Window to Create Tasks and Notifications

2. Click **Add** from the **Action** window to add a new Task.

6.1.2.1 Defining Task Details

Figure : Action Details Window - Defining Task Details

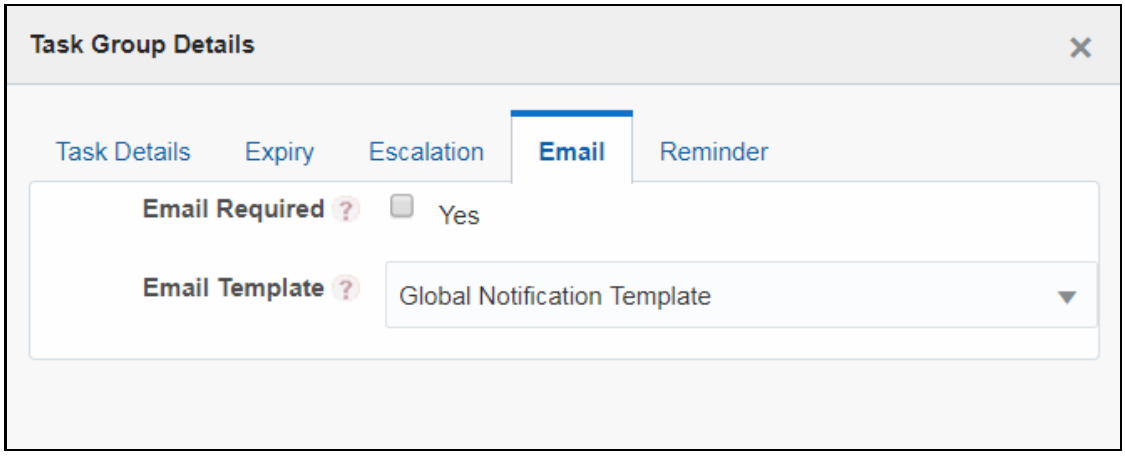


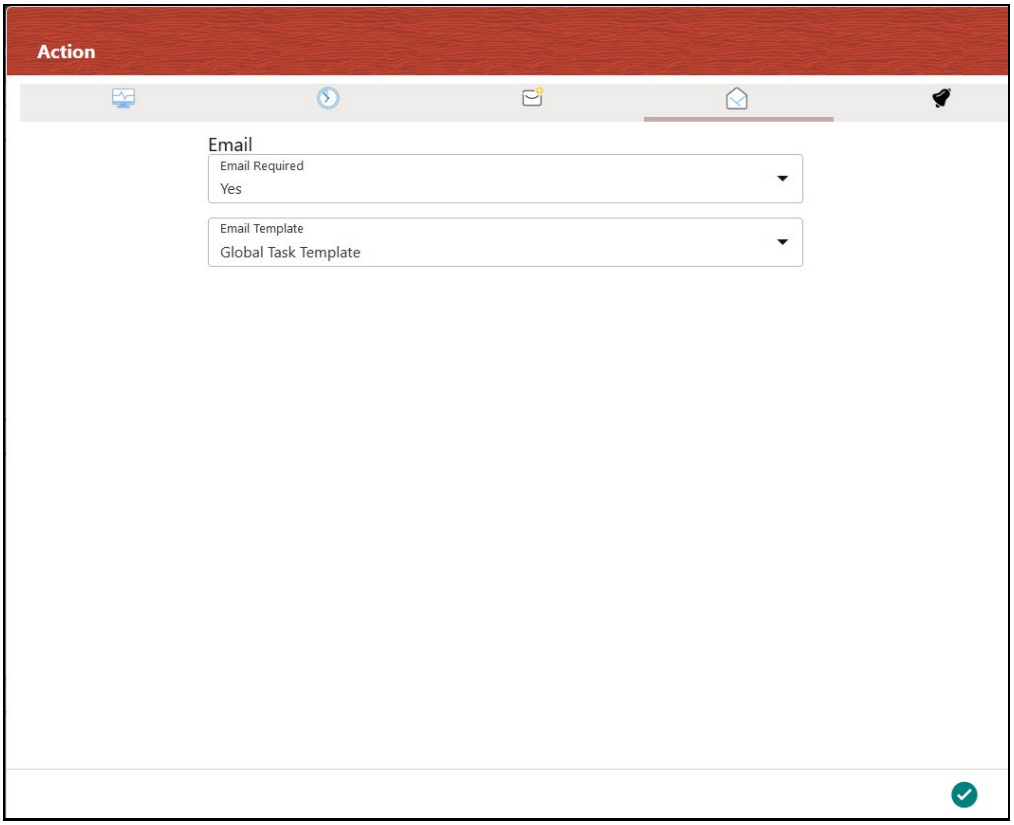
Task ID is automatically generated.

1. Enter a Task Name.
2. Select the decision rule so that when the **Condition** is satisfied, the Task is sent to the users associated with this task.
3. 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 the next activity.
   - On Stage- When there is a transition where the target and source are the same activity, and if that transition happens, then OnStage tasks are triggered.
4. Select **Yes** to enable the Task.

6.1.2.2 Setting Email Notification

Figure : Action Details Window - Setting Email Notification





You must configure Email settings before you set email notifications for your tasks.

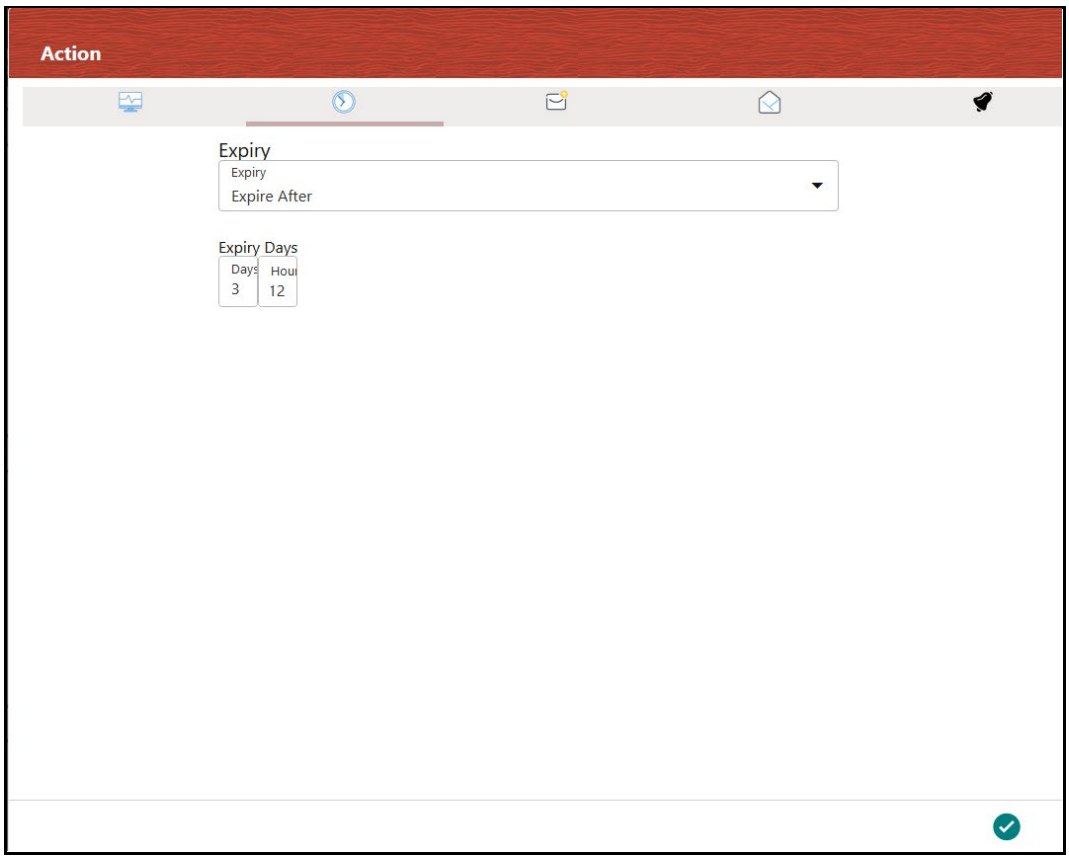
For more details on how to configure email settings, see Configuring Email for Human Tasks section.

1. Select **Yes** if it is required to send an email for the task.
2. Select the email template to be used from the drop-down list. Templates are populated from the `AAI_WF_Email_Template` table.

6.2 Additional Functionalities for Human Tasks

6.2.1 Setting Task Expiry

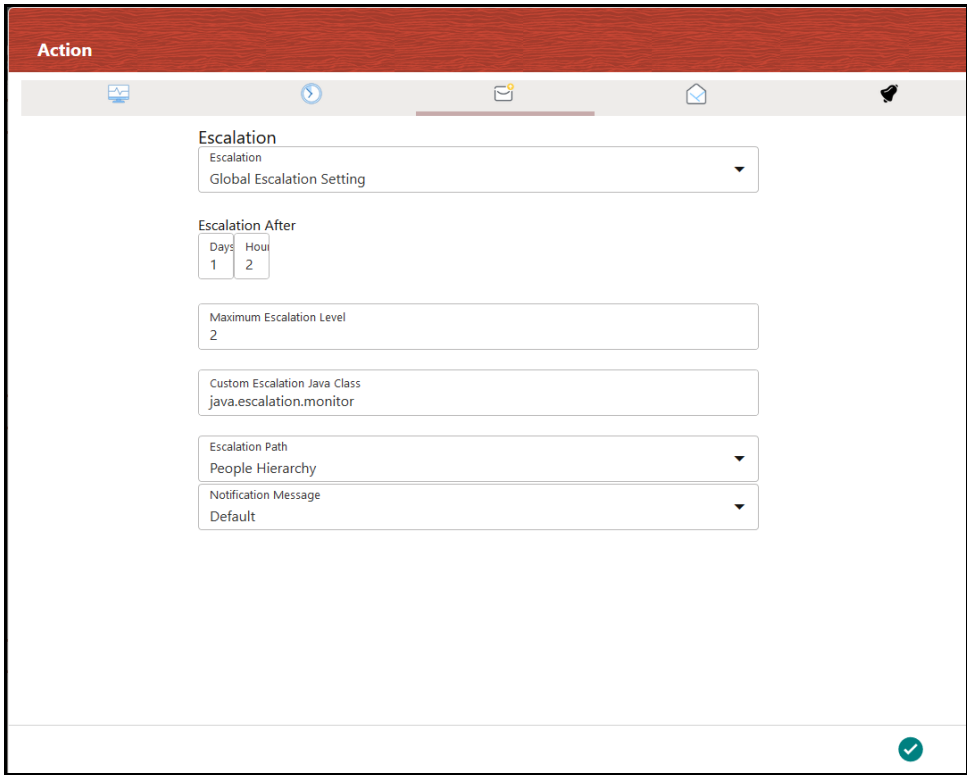
Figure : Action Details Window - Setting Task Expiry



1. Select the required option from the **Expiry** drop-down list:
   - Global Expiry Setting- Select this option to set the task expiry based on a global setting. Global setting can be set from the 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 into the code.
2. If **Expiry** is selected as **Expire After,** enter the number of days and/or hours after which the task must expire.

6.2.2 Setting Task Escalation

Figure : Action Details Window - Setting Task Escalation



1. Select the **Escalation** criteria as follows:
   - Global Escalation Setting- Select this option to set the task escalation based on a global setting. Global setting can be set from the 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.

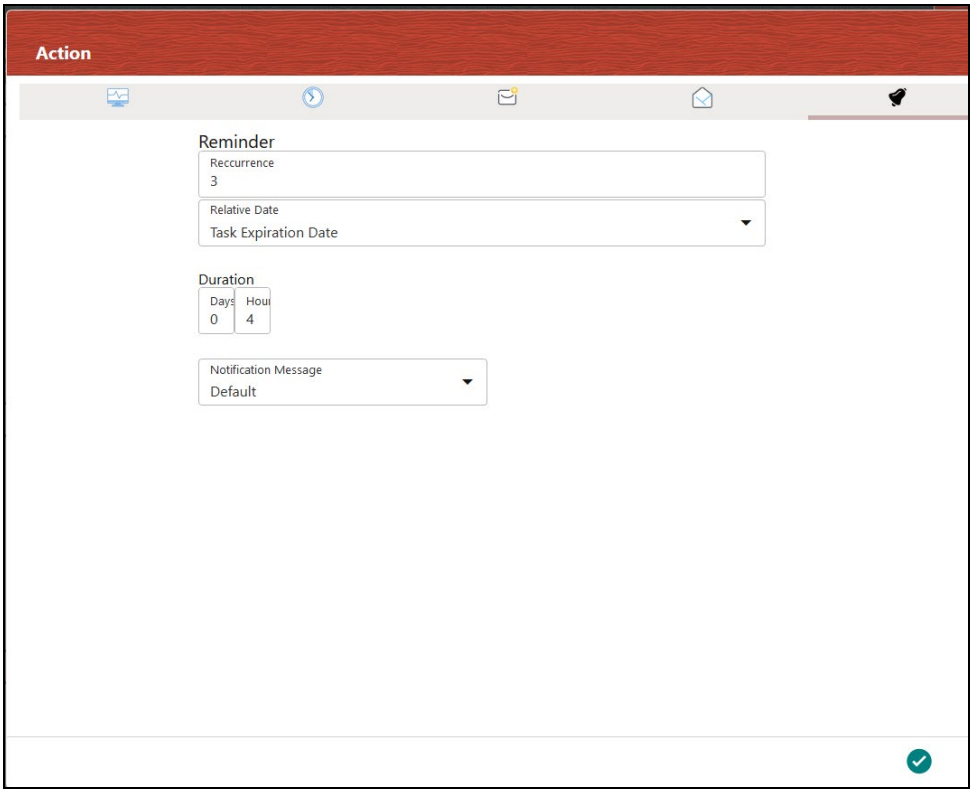
> **NOTE** Escalation should be set after the expiry of the task. If you have selected the Never Expire option for Expiry, you cannot set escalation.

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 levels. 1 indicates escalation to the immediate manager, 2 indicates escalation to the manager's manager, and so on.
4. Enter the custom escalation Java Class that 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 the `AAI_WF_Templates` table.

6.2.3 Setting Task Reminder

Figure : Action Details Window - Setting Reminder for Your Task



Reminders are 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 a 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 a reminder before the defined number of days and/or hours from the end date/expiry date of the task.
3. Select the Duration in 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 the `AAI_WF_Templates` table.
5. Click **Accept** to save.

7 Service Tasks

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

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

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

Figure : An Example: Service Tasks Flow

Topics:

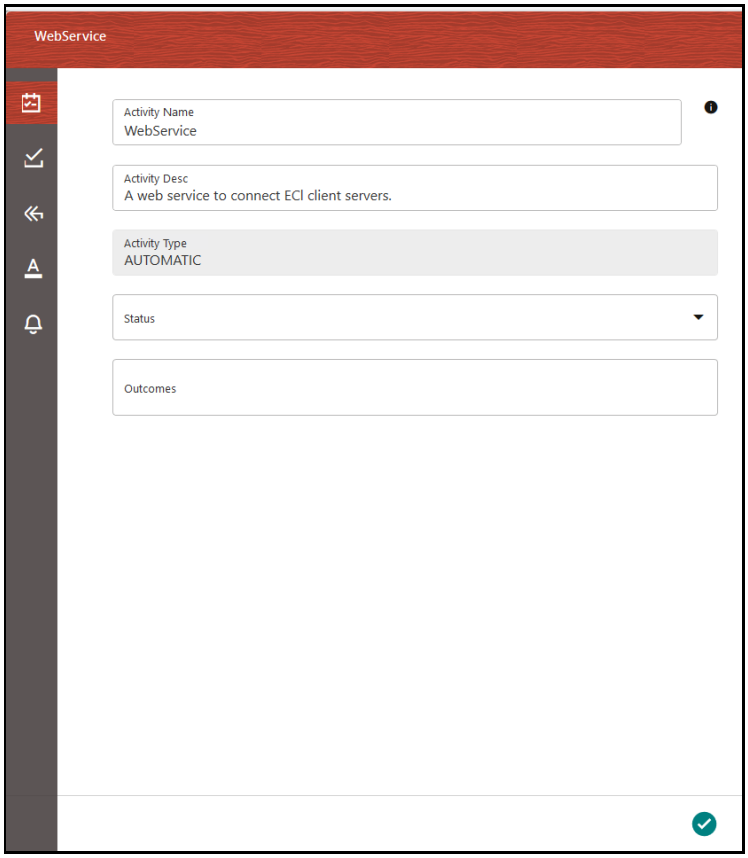
- How to Use a Service Task

7.1 How to Use a Service Task

1. In the **Process Flow** window, Select the **Service Task** icon from **Activity** in the toolbar and Drag and drop on the drawing canvas where you want to place it.
2. Double-click the icon on the canvas to display the **Activity** window.

7.1.1 Activity Tab

Figure : Activity Window

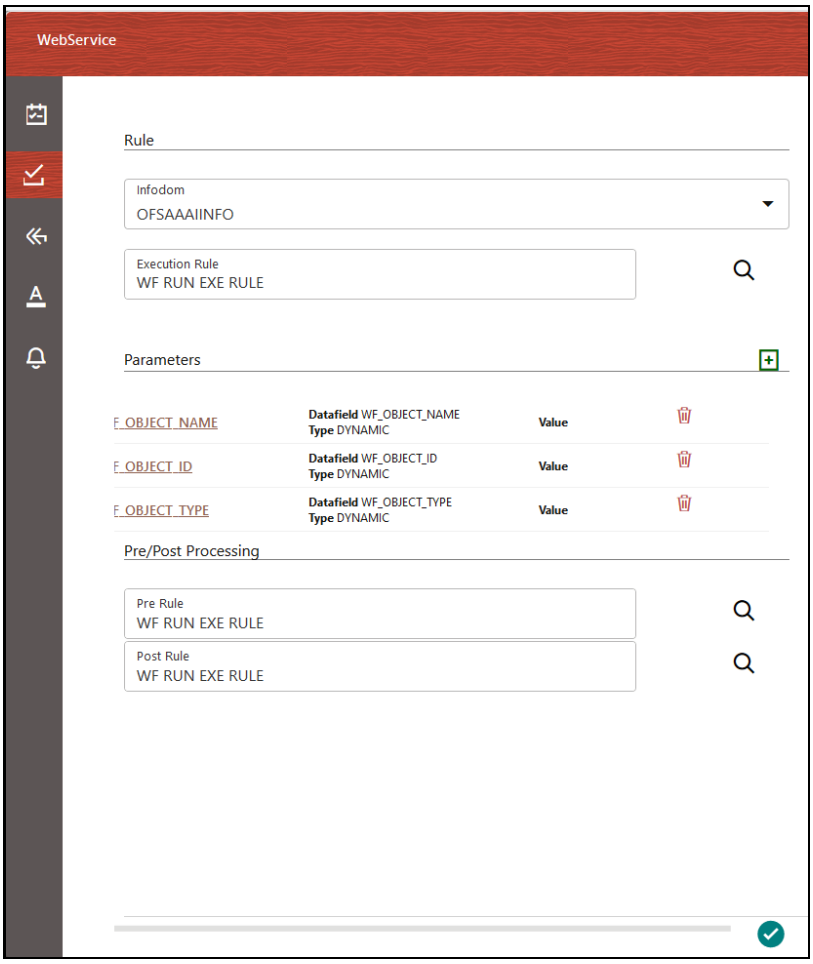


3. **Activity ID** is auto-populated and you can view it by mouse over the **i** icon.
4. Enter Activity Name and Activity Description.
5. Select Status and Outcomes as required.
6. Click **Accept** to save.

7.1.2 Implementation Tab

1. Click . The **Implementation** window is displayed.

Figure : Implementation Window



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** that needs to be executed for this activity.

   For more information on how to define an Application Rule, see the 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 the case of Dynamic, the value needs to be entered during the execution of the workflow.

   - Click **Add** under **Parameters.** The **Participant Details** window is displayed.

Figure : Participant Details Window

- Select the **Data Field** to which you want to pass the value. The list displays all Data Fields for the current Process or Package.
- Select the **Parameter Type** as Static to pass a static value to the selected Data Field in the **Value** field or as Dynamic to pass the value during execution of the workflow.

NOTE The added parameters are displayed under the **Parameters** section in the Implementation Window.

Select the Parameter and click it to edit it.

Click to delete it.

5. Select an application rule that you want to execute before executing the Execution Rule.
   - Click the icon corresponding to **Pre Rule** and select the required application rule.
6. Select an application rule that you want to execute after executing the Execution Rule.
   - Click the icon corresponding to **Post Rule** and select the required application rule.

8 Configuring OFSAA Tasks in Your Process Flow

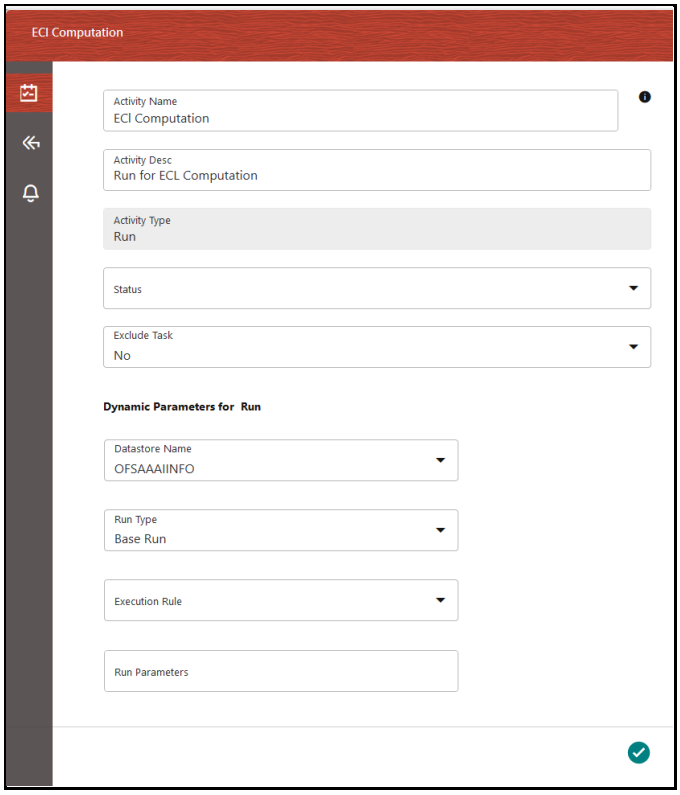
Widgets are used to orchestrate OFSAA components such as T2T definitions, PLC definitions (DT), Rules (Classification Rule and Computation Rule), Models in EMF, Run, Run Executable, and Data Quality Groups into your Process Flow using the Process Modeller.

Topics:

8.1 How to Use a Widget

1. In the **Process Flow** window, click the required widget under **Widget** in the toolbar and Drag and drop it onto the canvas. The available Widgets are RuleType3, MFModel, Run, DataQualityGroups, RunExecutable, EMFNotebookImpl, LoadT2T, TransformDT, and RuleType2.
2. Double-click the widget. The **Activity** window is displayed.

Figure : OFSAA Widget Activity Window



3. Enter the **Activity Name** and **Activity Description**.
4. Based on the OFSAA widget, the **Dynamic Parameters** are displayed.

   For more information, see the Dynamic Parameters for Widgets section.

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.

8.2 Dynamic Parameters for Widgets

The Dynamic Parameters in the Activity window are different for each Widget. This section provides information for the following Dynamic Parameters specific to the selected Widget.

- RuleType3
- MFModel
- Run
- DataQualityGroups
- RunExecutable
- ESIC
- EMFNotebookImpl
- LoadT2T
- TransformDT
- RuleType2

8.2.1 RuleType3

In the **Process Flow** window, click the **RuleType3** icon in **Widget** from the toolbar and Drag and drop it onto the canvas. Double-click the widget to display the **Activity** window.

Figure : Dynamic Parameters for RuleType3

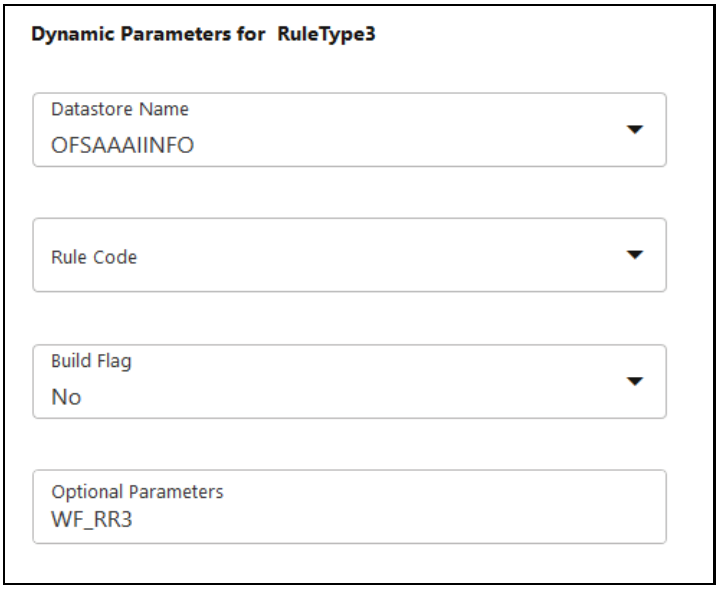


Table : Dynamic Parameters for RULE_EXECUTION Description

Field Name	Description
Datastore Name	Select the Information Domain in which the RRF Rule you want to execute is present, from the drop-down list.
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 the 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 are stored in the database on “Save” of a Rule definition. When this rule is executed, the database is accessed to form the rule query based on stored metadata details, thus ensuring performance enhancement during Rule execution. For more information, see the Significance of Pre-Built Flag section in the 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 the Data Fields section.

8.2.2 MFModel

In the **Process Flow** window, click the **MFModel** icon in **Widget** from the toolbar and Drag and drop it onto the canvas. Double-click the widget to display the **Activity** window.

Figure : Dynamic Parameters for MODEL

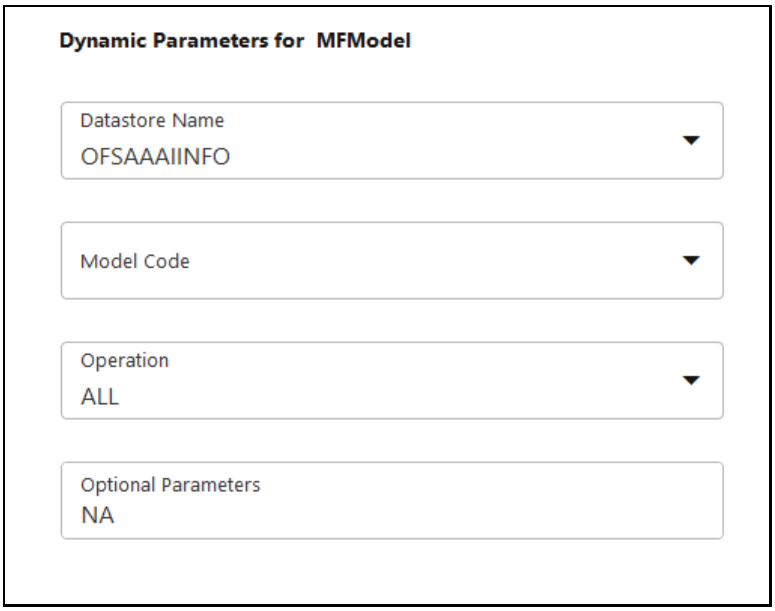


Table : Dynamic Parameters for MODEL Description

Field Name	Description
Datastore Name	Select the Information Domain in which the RRF Rule you want to execute is present, from the drop-down list.
Model Code	Display the codes of the EMF Models defined under the selected Infodom.
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 is extracted from the flat files and the run regression is 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 the Data Fields section.

8.2.3 Run

In the **Process Flow** window, click the **Run** icon in **Widget** from the toolbar and Drag and drop it onto the canvas. Double-click the widget to display the **Activity** window.

Figure : Dynamic Parameters for RUN

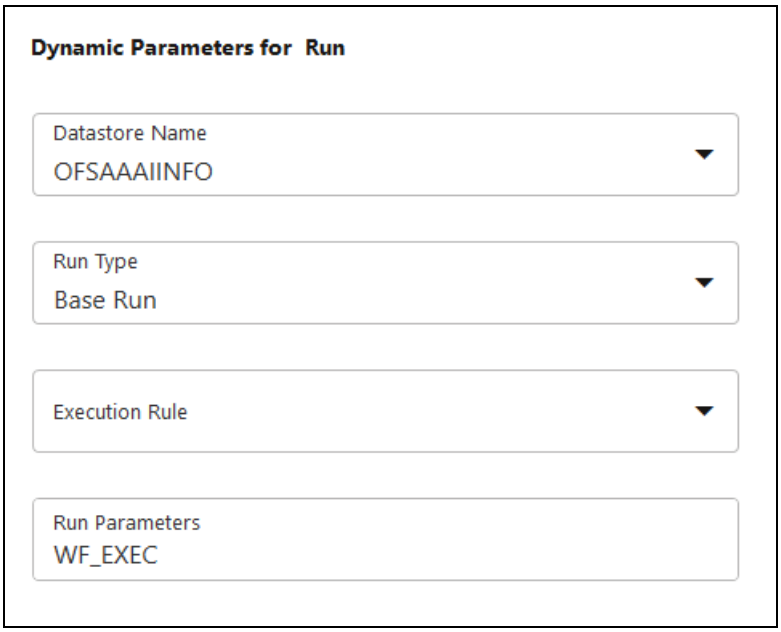


Table : Dynamic Parameters for RUN Description

Field Name	Description
Datastore Name	Select the required datastore 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	Enter the Data Fields you want to pass as parameters for the selected Data Mapping definition. Use comma-separated values if you want to enter more than one Data Field. For information on creating Data Fields, see the Data Fields section.

8.2.4 Data Quality Groups

In the **Process Flow** window, click the **DataQualityGroups** icon in **Widget** from the toolbar and Drag and drop it onto the canvas. Double-click the widget to display the **Activity** window.

Table : Dynamic Parameters for RUN DQ RULE Description

Property	Description
DQ Group Name	Refers to the Data Quality Groups consisting of associated Data Quality Rule definition(s). Select the required DQ Group from the drop-down list.
Rejection Threshold	Specify the percentage of the 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 fails. If the field is left blank, the default value is set to 100%.
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 the configuration schema for execution.
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, the execution of the task continues. **Note**: For Custom Check type DQ Rules in Hive Infodoms, the execution of the task does 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 the Data Fields section. **Note**: Runsk by default is sent to DQ, even if you are not passing any parameters.

8.2.5 RunExecutable

In the **Process Flow** window, click the  **RunExecutable** icon in **Widget** from the toolbar and Drag and drop it onto the canvas. Double-click the widget to display the **Activity** window.

Figure : Dynamic Parameters for RUN EXECUTABLE

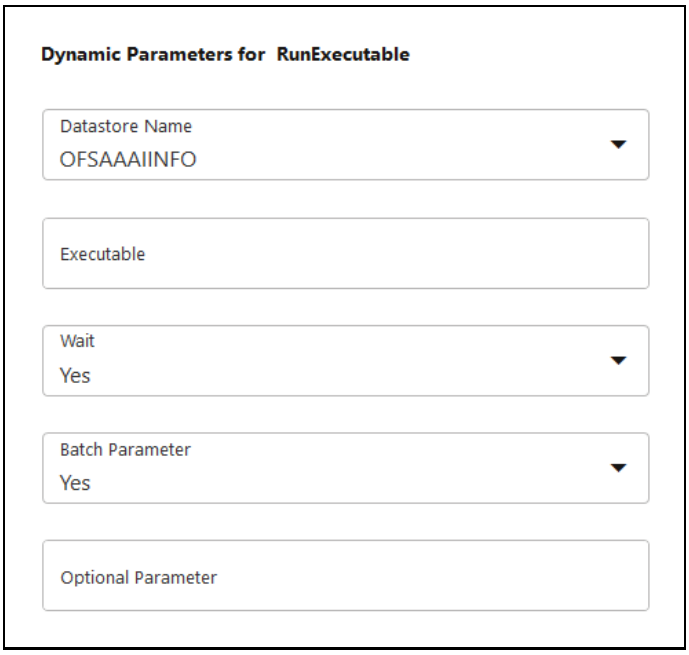


Table : Dynamic Parameters for RUN EXECUTABLE Description

Field Name	Description
Datastore Name	Refers to the name of the Information Domain. By default, the Information Domain to which you are connected is selected.
Executable	Refers to the executable path on the DB Server. The Executable parameter contains the executable name as well as the parameters to the executable. These executable parameters have to be specified as they are specified at a command line. In other words, the Executable parameter is the exact command line required to execute the executable file. The path to the executable has been entered in quotes. Quotes have to be used if the EXE name has a space included in it. In other words, the details entered here should look exactly as you would enter it in the command window while calling your executable. The parameter value is case-sensitive. So, ensure that you take care of the spaces, quotes, and case. Additionally, commas are not allowed while defining the parameter value for executable. To pass parameters like $RUNID, $PHID, $EXEID, $RUNSK to the RUN EXECUTABLE component, specify RRFOPT=Y or rrfopt=y along with other executable details.

Field Name	Description
Wait	When the file is being executed you have the choice to either wait till the execution is completed or proceed with the next task. Select **Y** (Yes) or **N** (No) from the drop-down list. • **Y-** Select this if you want to wait for the execution to be completed • **N**- Select this if you wish to proceed. If the task is using FICGEN/RUN EXECUTABLE component and there is no precedence set for this task, then the **WAIT** should always be set to **'N'**.

Batch Parameter	• **Y**- Select **Yes** if you want to pass the Batch parameters to the shell script file being executed.
	▪ If **Wait** is selected as **Y** and **Batch Parameter** is selected as **Y**, following parameters are passed to the executable:
	`NIL <BatchExeRunID> <ComponentId> <Task> <Infodate> <Infodom> <DatstoreType> <IPAddress>`
	▪ If **Wait** is selected as **N** and **Batch Parameter** is selected as **Y**, following parameters are passed to the executable:
	`<BatchExeRunID> <ComponentId> <Task> <Infodate> <Infodom> <DatstoreType> <IPAddress>`
	• **N**- Select **No** if the Batch parameters should not be passed to the shell script.

Optional Parameter	The following runtime parameters can be passed during run execution: • $RUNID • $PHID • $EXEID • $RUNSK • $MISDATE • $BATCHRUNID Values for the runtime parameters are implicitly passed while executing the Run definition. **Note**: The length of PHID is 255 characters and TASKID is 100 characters.

8.2.6 ESIC

In the **Process Flow** window, click the  **ESIC** icon in **Widget** from the toolbar and drag and drop it onto the canvas. Double-click the widget to display the **Activity** window.

Figure : Dynamic Parameters for ESIC

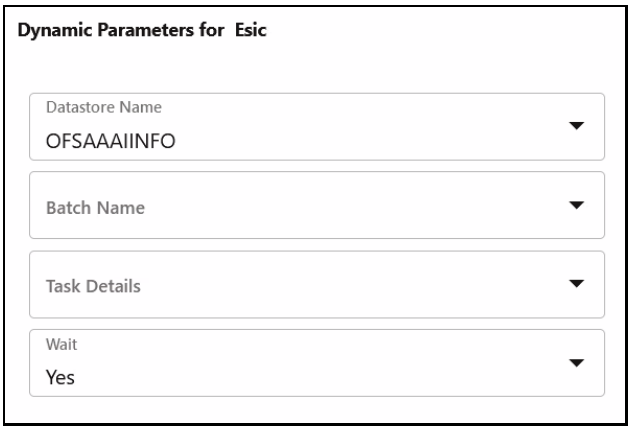


Table : Dynamic Parameters for ESIC Description

Field Name	Description
Datastore Name	Refers to the name of the Information Domain. By default, the Information Domain to which you are connected is selected.
Batch Name	Lists all the batches associated with the selected Datastore. Select the required Batch from the drop-down list, for execution.
Task Details	Lists the tasks associated with the selected Batch. You can either execute one task or all the tasks associated with the selected batch • To execute one specific task, select the task from the drop-down list • To execute all the tasks associated with the selected batch, do not select any tasks in the Task Details field.
Wait	If enabled (set to Yes), execution waits for the ESIC component to finish task execution and then update the task status.

8.2.7 EMFNotebookImpl

In the **Process Flow** window, click the **EMFNotebookImpl** icon in **Widget** from the toolbar and Drag and drop it onto the canvas. Double-click the widget to display the **Activity** window.

Figure : Dynamic Parameters for EMFNootbookImpl

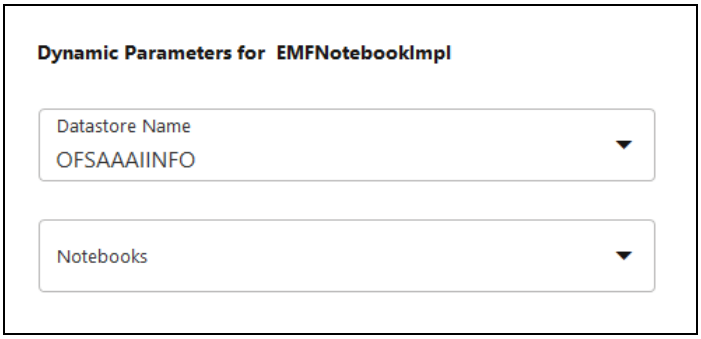


Table : Dynamic Parameters for EMFNootbookImpl Description

Field Name	Description
Datastore Name	Select the Information Domain from the drop-down list where the EMFNotebook exists.
Notebooks	Select from the drop-down the EMF Notebook that you want to implement in a task.

8.2.8 LoadT2T

In the **Process Flow** window, click the **LoadT2T** icon in **Widget** from the toolbar and drag and drop it onto the canvas. Double-click the widget to display the **Activity** window.

Table : Dynamic Parameters for LOAD DATA Description

Field Name	Description
Datastore Name	Select the Information Domain in which the Data Mapping you want to execute is present, from the drop-down list.
Load Mode	Select Table to Table or File to Table for Data Mapping definitions such as T2T and F2T definitions.
Source Name	Select the required source on which the Data Mapping definition you want to execute is defined, from the drop-down list.
File Name	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.
Default Value	Select the Data Fields you want to pass as parameters for the selected Data Mapping definition. For information on creating Data Fields, see the Data Fields section. For additional information, see the Task Component Parameters section in the OFS Analytical Applications Infrastructure User Guide.

8.2.9 TransformDT

In the **Process Flow** window, click the **TransformDT** icon in **Widget** from the toolbar and Drag and drop on onto the canvas. Double-click the widget to display the **Activity** window.

Figure : Dynamic Parameters for TRANSFORM DATA

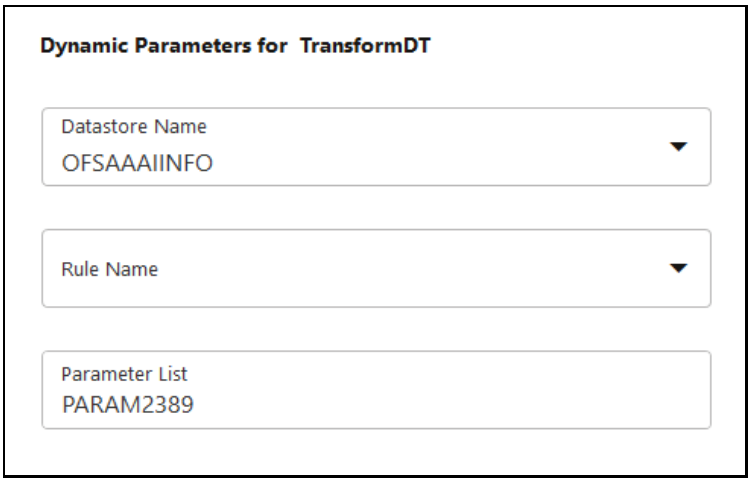


Table : Dynamic Parameters for TRANSFORM DATA Description

Field Name	Description
Datastore Name	Select the datastore name in which the PLC you want to execute is present from the drop-down list.
Rule Name	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.
Parameter List	Enter the Data Fields you want to pass as parameters for the selected Data Mapping definition. Use comma-separated values if you want to enter more than one Data Field. For information on creating Data Fields, see the Data Fields section.

8.2.10 RuleType2

In the **Process Flow** window, click the **RuleType2** icon in **Widget** from the toolbar and Drag and drop it onto the canvas. Double-click the widget to display the **Activity** window.

Figure : Dynamic Parameters for RULE_EXECUTION

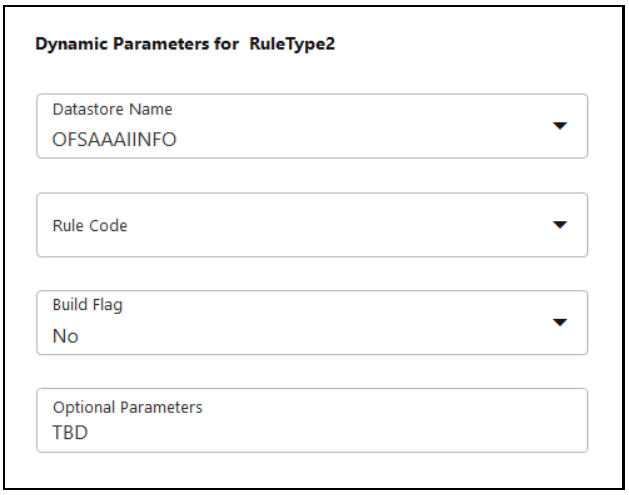


Table : Dynamic Parameters for RULE_EXECUTION Description

Field Name	Description
Datastore Name	Select the Information Domain in which the RRF Rule you want to execute is present, from the drop-down list.
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 the 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 are stored in the database on “Save” of a Rule definition. When this rule is executed, the database is accessed to form the rule query based on stored metadata details, thus ensuring performance enhancement during Rule execution. For more information, see the Significance of Pre-Built Flag section in the 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 the 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.

Topics:

9.1 How to invoke External Model through Web Service

Figure : Invoking External Web Service flow

A typical External Web Service invocation has the following steps:

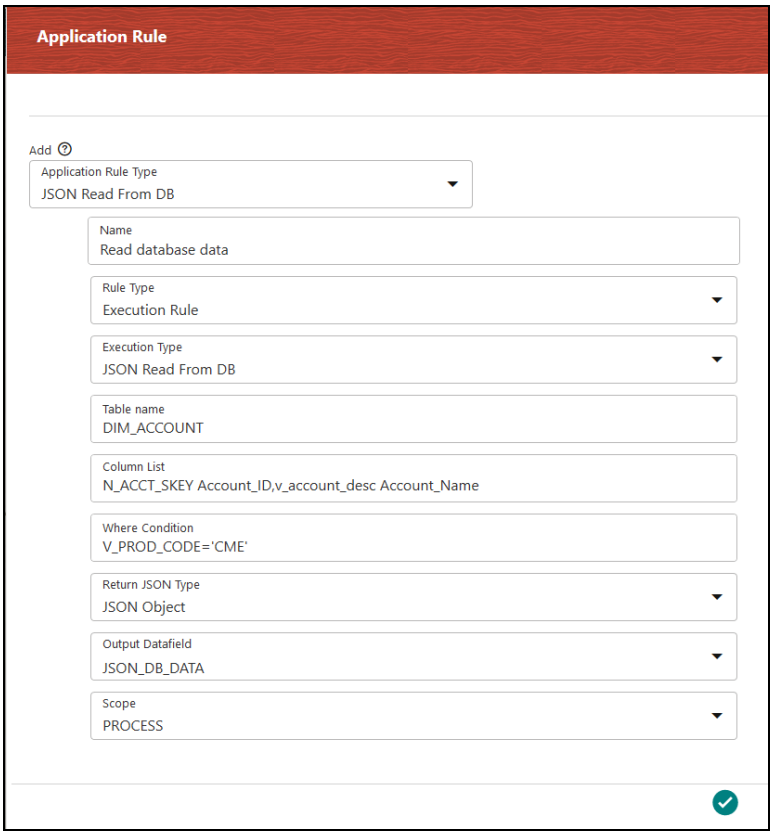
1. Data Preparation
2. Web Service Invocation
3. Data Extraction

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 the 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 it into JSON, and stores output to JSON_DB_DATA Data Field.

Figure : Add API Details



For more information, see JSON Read From DB Application Rule section.

9.1.2 Webservice Invocation

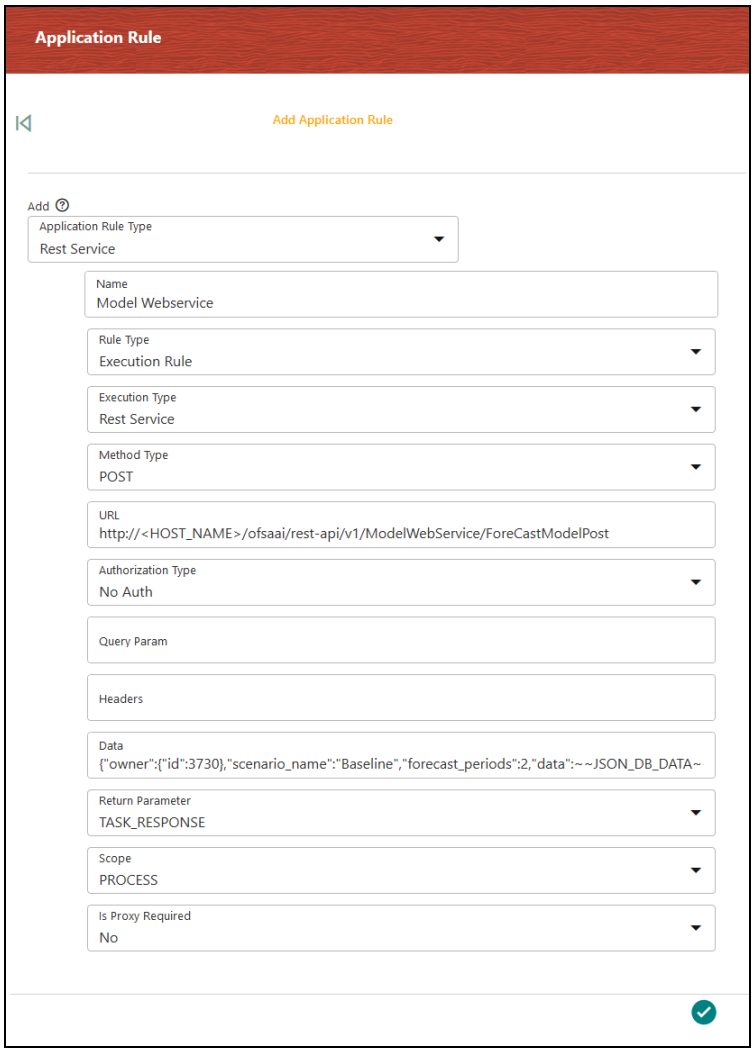
Consider you want to use an external model called ForecastModelPost, which is available in `http://<IP Address/hostname of the Web Server>:<servlet port>/<context name>/rest-api/v1/ModelWebService/ForecastModelPost`.

To access Application Rules, do as follows:

1. From the **Process Modeller** window, click the icon corresponding to the Pipeline for which you want to add an application rule. The **Process Flow** tab is displayed.
1. Select **Application Rules** icon from the header to display the **Application Rules** window.
2. Click **Add** to display the Add Application Rules window.

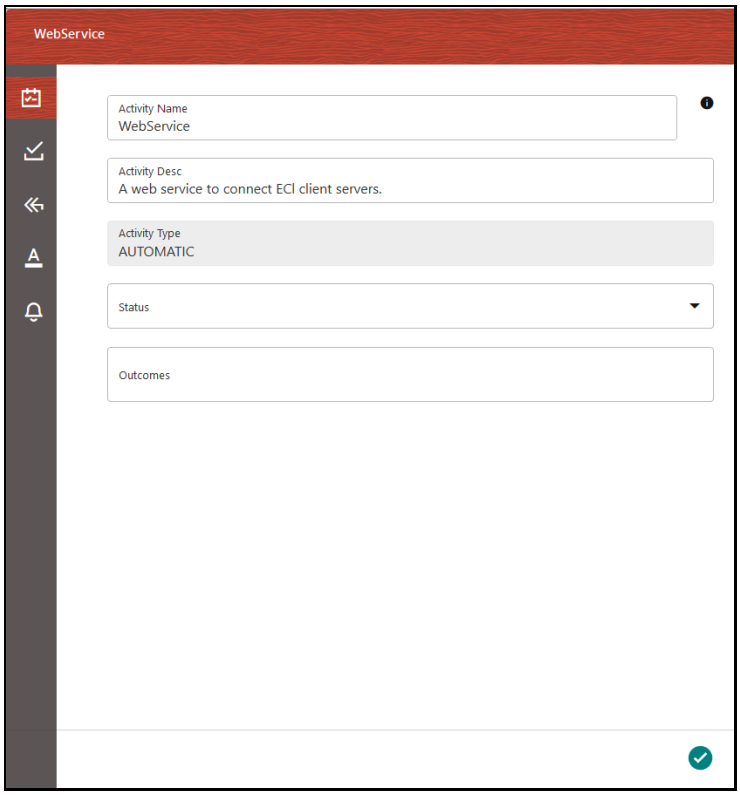
   You can delete a rule by clicking the button.
3. Select **Rest Service** from the **Application Rule Type** drop-down.

Figure : Webservice Invocation Rule Details



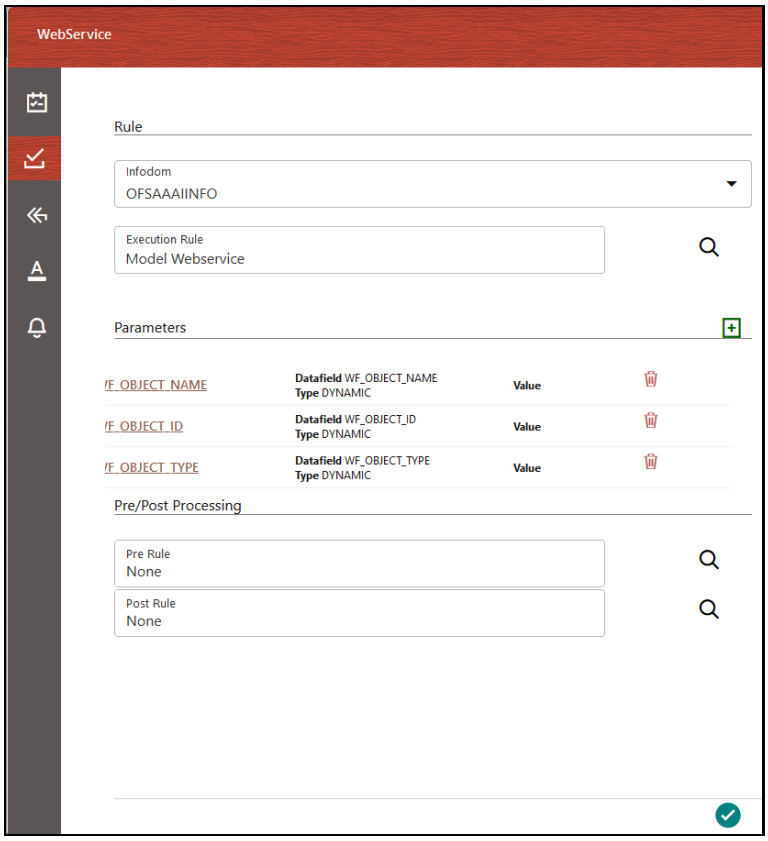
4. Enter the details as shown. For more information on configuring a Rest Service, see the Rest Service Application Rule section.
5. In this example, the Data sent to the web service is a combination of both static and dynamic value (~~JSON_DB_DATA~~ is a data field holding a JSON string which is prepared as the previous step in the pipeline.)
6. Click **Save**. The ModelWebService Application Rule gets created in your logged-in Information Domain.
7. In the **Process Flow** window, Select the **Service Task** icon from **Activity** in the toolbar and Drag and drop on the drawing canvas where you want to place it.
8. Double-click the icon on the canvas to display the **Activity** window.
9. Double-click the Service Task icon.

Figure : Webservice Invocation Activity details



10. Enter the Activity details such as Activity Name and Activity Description.
11. Click  . The **Implementation** window is displayed.
12. 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.
13. Click  The **Participant Details** window is displayed with all Application Rules of Execution Rule types available in your Process.
14. Select Model Webservice.
15. Click  **Accept** to save.

Figure : Webservice Invocation Participant Details



9.1.3 Data Extraction

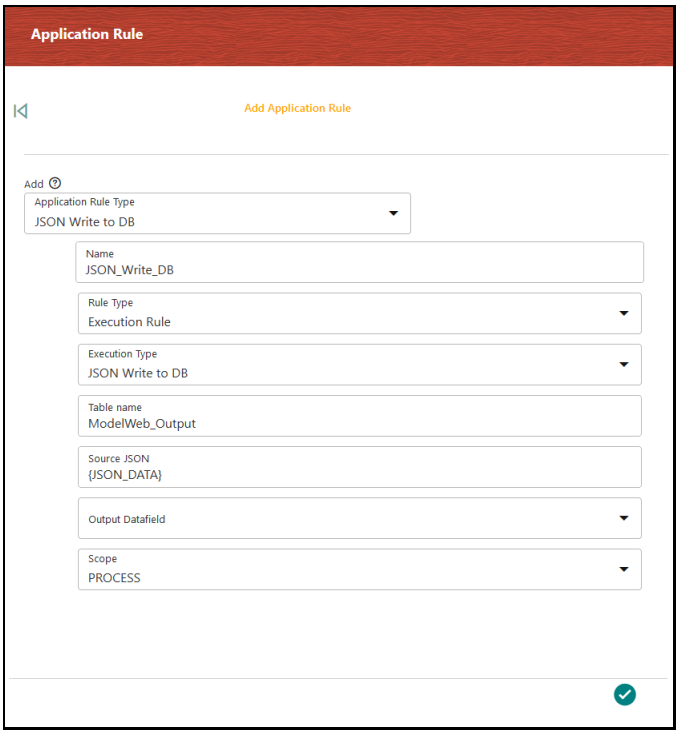
The response from the Web Service needs to be processed depending on the application requirement. PMF has the capabilities to process the JSON and store it in the output table.

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.

Figure : JSON Write To DB Application Rule- Add API Details window



For more information, see JSON Write To DB Application Rule section.

10 Configuring Custom Components

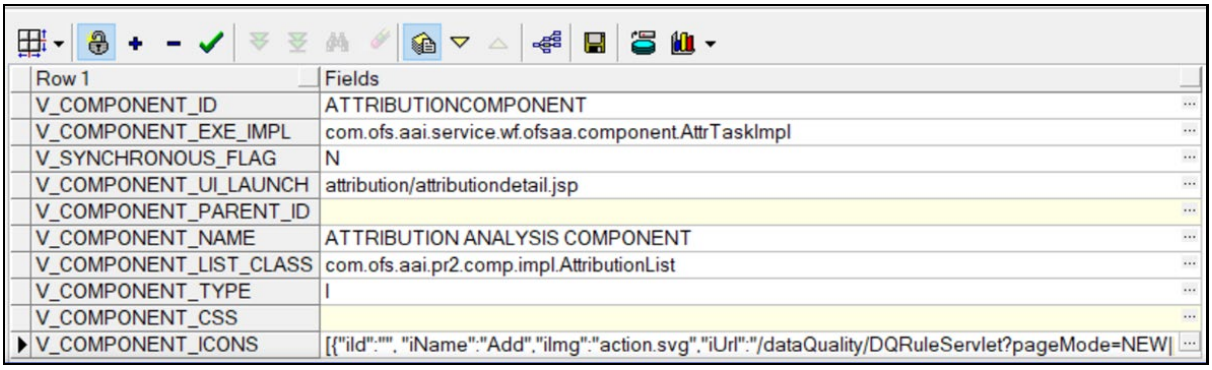
For configuring a new custom component as an OFSAA Widget, you need to have entries in the AAI_WF_COMPONENT_REGISTRATION table and AAI_WF_COMPONENT_PARAMETERS table. Additionally, the executeComponent() method needs to be implemented in the Implementation class and the jar should be present in `web-inf/lib`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

Topics:

10.1 AAI_WF_COMPONENT_REGISTRATION Table

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

Figure : AAI_WF_COMPONENT_REGISTRATION table

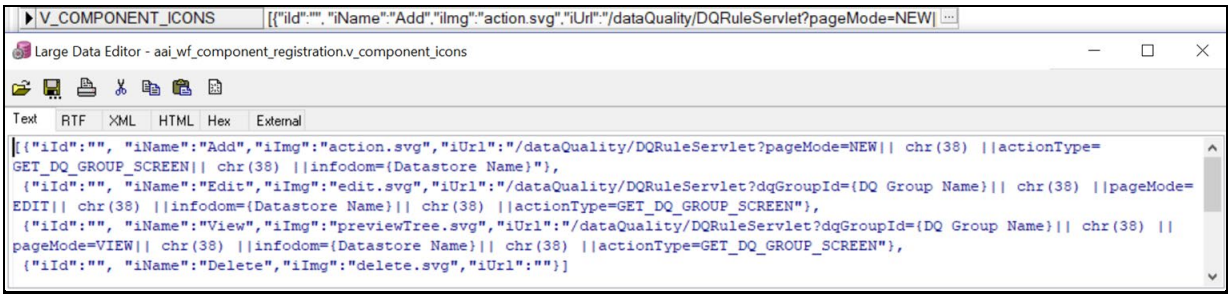


Row 1	Fields
V_COMPONENT_ID	ATTRIBUTIONCOMPONENT
V_COMPONENT_EXE_IMPL	com.ofs.aai.service.wf.ofsaa.component.AttrTaskImpl
V_SYNCHRONOUS_FLAG	N
V_COMPONENT_UI_LAUNCH	attribution/attributiondetail.jsp
V_COMPONENT_PARENT_ID	
V_COMPONENT_NAME	ATTRIBUTION ANALYSIS COMPONENT
V_COMPONENT_LIST_CLASS	com.ofs.aai.pr2.comp.impl.AttributionList
V_COMPONENT_TYPE	I
V_COMPONENT_CSS	
V_COMPONENT_ICONS	[{"ild":"", "iName":"Add","ilmg":"action.svg","iUrl":"/dataQuality/DQRuleServlet?pageMode=NEW|

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

For example:

Figure : AAI_WF_COMPONENT_REGISTRATION Example table



10.2 AAI_WF_COMPONENT_PARAMETERS Table

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

Figure : AAI_WF_COMPONENT_PARAMETERS table

	V_COMPONENT_ID	V_PARAMETER_NAME	I_PARAMETER_ORDER	I_DISPLAY_ORDER	V_PARAMETER_CODE	TYPE_OF_DISPLAY
1	LOAD DATA	IP Address	1	3	SMSLB.IP_ADDR	2
2	LOAD DATA	Datastore Type	2	1	ICC.FE.LBL_DATA_STORE_TYPE	2
3	LOAD DATA	Datastore Name	3	2	ICC.FE.LBL_DATA_STORE	2
4	LOAD DATA	Load Mode	4	4	ICC.FE.LBL_LOADMODE	2
5	LOAD DATA	Source Name	5	5	SMSLB.SRC_NAME	2
6	LOAD DATA	File Name	6	6	GLOBAL.FIL_NAM	2
7	LOAD DATA	Data File Name	7	7	ETL.PROP_DEF.LABEL_OR_TEXT_DATA_FILE_NAME	1
8	LOAD DATA	Default Value	8	8	DTDQ.DFLT_VAL	1

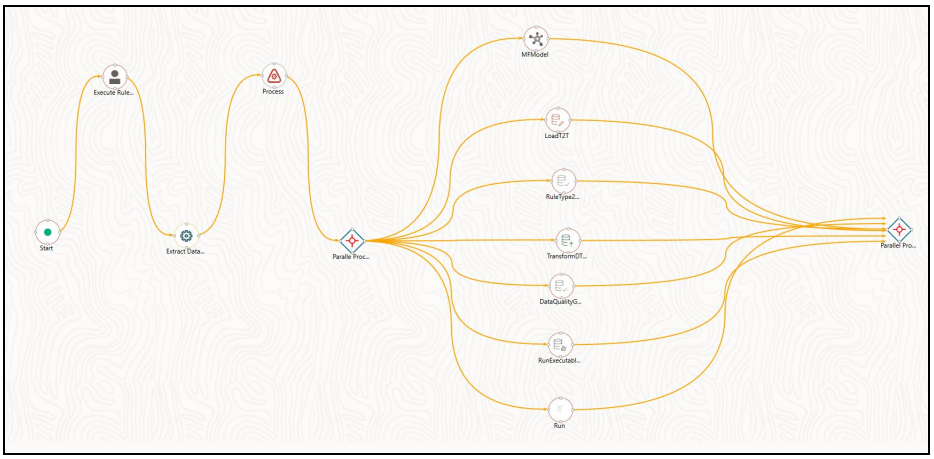
V_DEFAULT_VALUE	V_CLASS_NAME	V_METHOD_NAME
	com.ofs.aai.service.wf.ofsaa.component.T2TTaskImpl	
	com.ofs.aai.service.wf.ofsaa.component.T2TTaskImpl	

- V_COMPONENT_ID- Specify the ID for the component. It should be the same as that is 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 the 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.

Figure : Executing Parallel Tasks flow



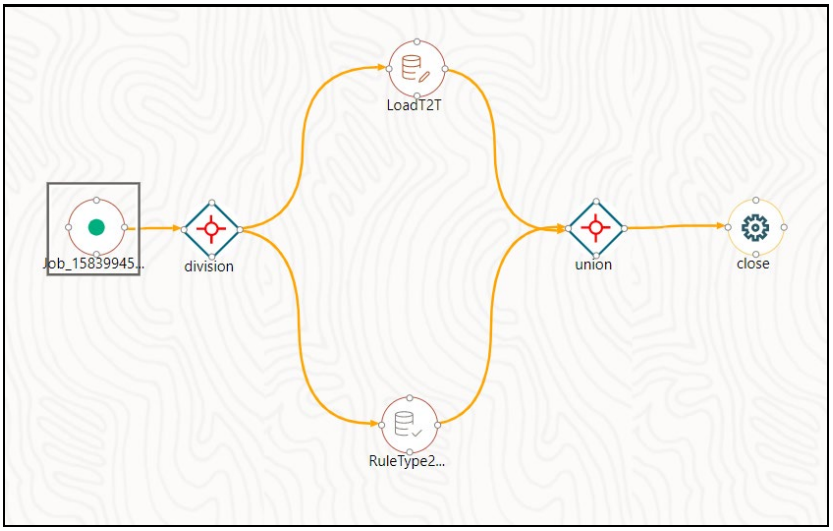
In the shown example, after  is executed, Parallel Gateway  gets invoked. All the OFSAA components, which are placed between Parallel Gateways,  are executed simultaneously. It waits until all components are executed, and then moves to the next activity in the Process Flow.

Topics:

- How to Use Parallel Gateways

11.1 How to Use Parallel Gateways

Figure : RuleType 2 and Load T2T Widget Parallel Execution Flow



This section explains how to design tasks such as RuleType 2 and Load T2T Widget to be executed in parallel.

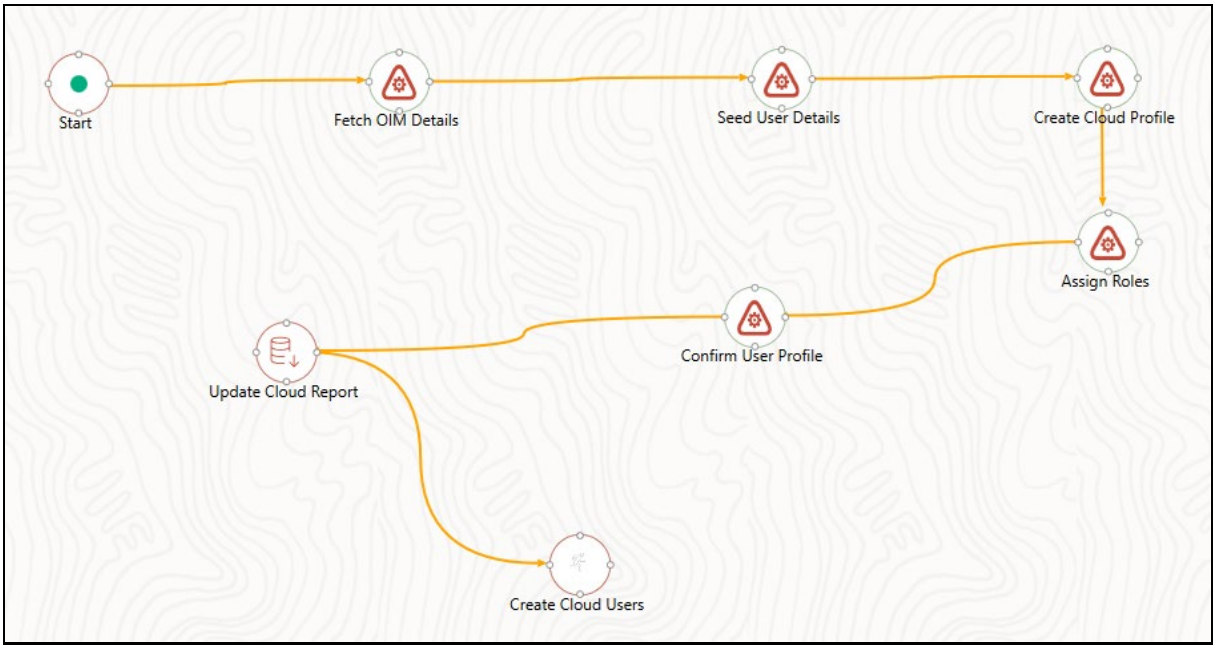
1. Create a Process Modelling Process.
2. Launch the Process. It displays the **Process Flow** tab.
3. By default, the  **Start** appears on the canvas. Drag and drop to a desired area on the canvas. This Start activity indicates the beginning of the Process.
4. Click  Parallel Gateway from the toolbar and click on the drawing canvas where you want to place it.
5. Drag and drop the Parallel Gateway widget onto the Activity from where the transition starts.
6. Drag and drop the Activity on to the Parallel Gateway widget where the transition progresses.
7. Click  **RuleType2** from the **Widgets** window, and then click the drawing canvas where you want to place it.
8. Drag and drop the Parallel Gateway widget onto the Activity to make the transition connection.
9. Configure the RuleType2 widget. For more information, see Configuring OFSAA Tasks in Your Process Flow section.
10. Click  **LoadT2T** from the **Widgets** window, and then click the drawing canvas where you want to place it.
11. Drag and drop the **LoadT2T** widget onto the Parallel Gateway widget to make the transition connection.
12. Configure the **LoadT2T** widget. For more information, see Configuring OFSAA Tasks in Your Process Flow section.
13. Drag and drop the Parallel Gateway widgets onto the Run widget to make the transition connection.
14. Click  Parallel Gateway from the toolbar and click on the drawing canvas where you want to place it.
15. Connect the RuleType2 and LoadT2T widgets onto the Parallel Gateway widget to make the transition connections.

    You can drag and position the widgets to avoid overlapping 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. You can add filters to a Sub Pipeline. Then all Run Pipelines that are using the Sub Pipeline inherit those filters used in the Sub Pipeline.

Figure : Calling another Pipeline from Your Parent Pipeline

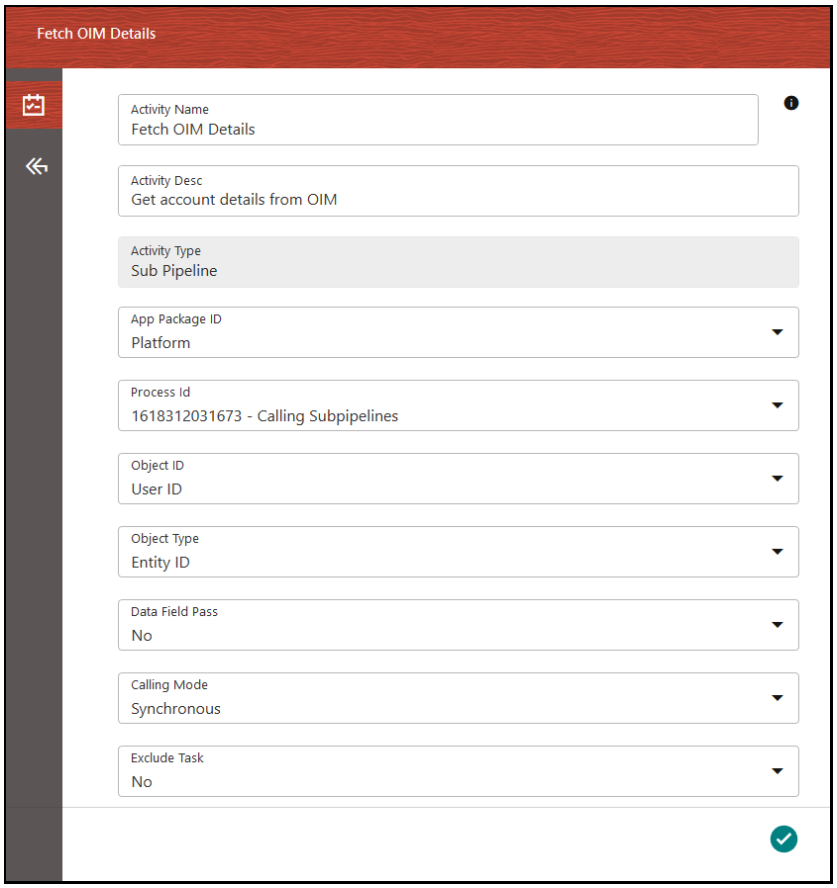


Topics:

12.1 How to Configure Sub Pipeline

1. From the **Process Flow** tab, click **Sub Pipeline** from the **Activity** toolbar and click the drawing canvas where you want to place it.
2. Double-click the Sub Pipeline icon on the canvas. The **Sub Process Details** window is displayed.

Figure : Sub Process Details Window



3. Enter the details as given in the table:

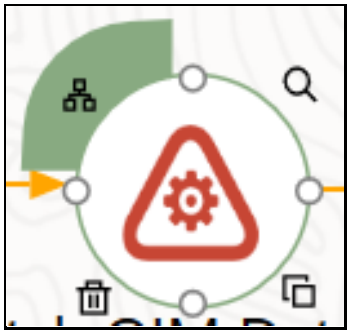
Table : Sub Process Details Description

Field Name	Description
App Package ID	Select the Application Package from which you want to call a process, from the drop-down list. The package IDs that are seeded from the backend are displayed in the list.
Process ID	Select the Process that you want to call within your workflow, from the drop-down list. The list displays all processes defined for the selected Application Package.
Object ID	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 of the current stage. Object ID drop-down list is populated from the Data Fields.
Object Type	Select the object type from the drop-down list. Object Type is used to identify a workflow that is passed by the application.
Data field pass	Select Yes to pass the parameters passed to the parent workflow to be passed to the selected sub pipeline.

Field Name	Description
Calling Mode	• Synchronous – Select this option if you want the sub pipeline to complete before the flow moves to the next activity. • Asynchronous – Select this option if you do not want to wait for the sub pipeline to complete. Besides, the parent workflow proceeds to the next activity.

4. Mouse-over the Sub Pipeline icon to display the submenu.

Figure : Mouse-over Sub Pipeline Icon to display the Submenu



5. Click  to display the Filter details.

   For more information on selecting Filters, see Filter Details.

13 Configuring Email for Human Tasks

Following table entries should be made for setting Email notification:

13.1 AAI_EMAIL_CONFIG Table

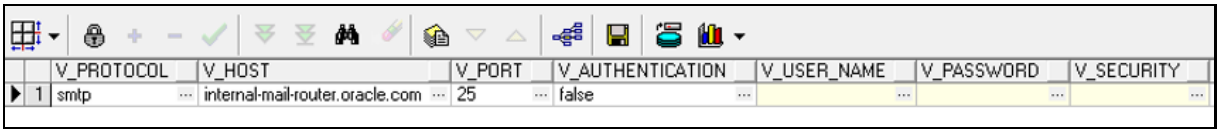
This table holds the SMTP server configurations.

- `V_PROTOCOL` - SMTP
- `V_HOST` –SMTP/ Mail Server ID

> **NOTE** If the SMTP HostName does not function or displays as invalid, use the IP address of the SMTP server in the AAI_EMAIL_CONFIG table.

- `V_PORT` - SMTP Server Port
- `V_AUTHENTICATION` - Either False or True
- `V_USER_NAME` - Login name to SMTP/ Mail Server ID from which mail is 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.

Figure : AAI_EMAIL_CONFIG Table



	V_PROTOCOL	V_HOST	V_PORT	V_AUTHENTICATION	V_USER_NAME	V_PASSWORD	V_SECURITY
1	smtp	internal-mail-router.oracle.com	25	false			

13.2 AAI_WF_APP_PACKAGE_B Table

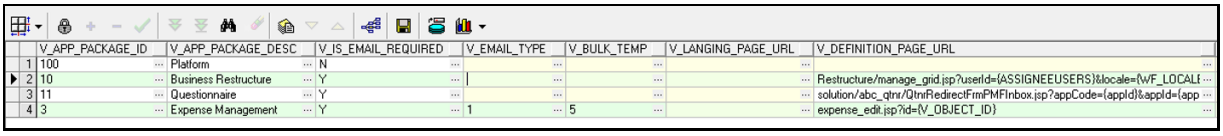
The `AAI_WF_APP_PACKAGE_B` table entry is for configuring email at the Application Level.

- Set the `V_EMAIL_REQUIRED` parameter value to Y in the `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 is sent with all the pending notifications from the last trigger, as a PDF attachment. After the bulk mail is sent, the `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 blank if the bulk email is not set.

Figure : AAI_WF_APP_PACKAGE_B Table



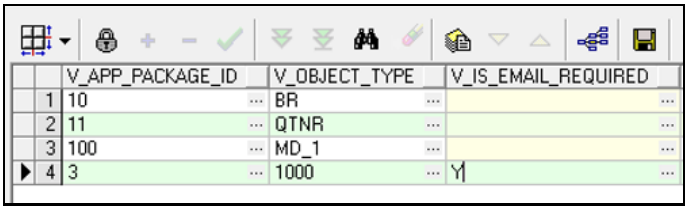
	V_APP_PACKAGE_ID	V_APP_PACKAGE_DESC	V_IS_EMAIL_REQUIRED	V_EMAIL_TYPE	V_BULK_TEMP	V_LANGING_PAGE_URL	V_DEFINITION_PAGE_URL
1	100	Platform	N				
2	10	Business Restructure	Y				Restructure/manage_grid.jsp?userId={ASSIGNEEUSERS}&locale={WF_LOCALE
3	11	Questionnaire	Y				solution/abc_qtnr/QtnrRedirectFrmPMFInbox.jsp?appCode={appId}&appId={app
4	3	Expense Management	Y	1	5		expense_edit.jsp?id={V_OBJECT_ID}

13.3 AAI_WF_APP_REGISTRATION Table

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

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

Figure : AAI_WF_APP_REGISTRATION Table



	V_APP_PACKAGE_ID	V_OBJECT_TYPE	V_IS_EMAIL_REQUIRED
1	10	BR	
2	11	QTNR	
3	100	MD_1	
4	3	1000	Y

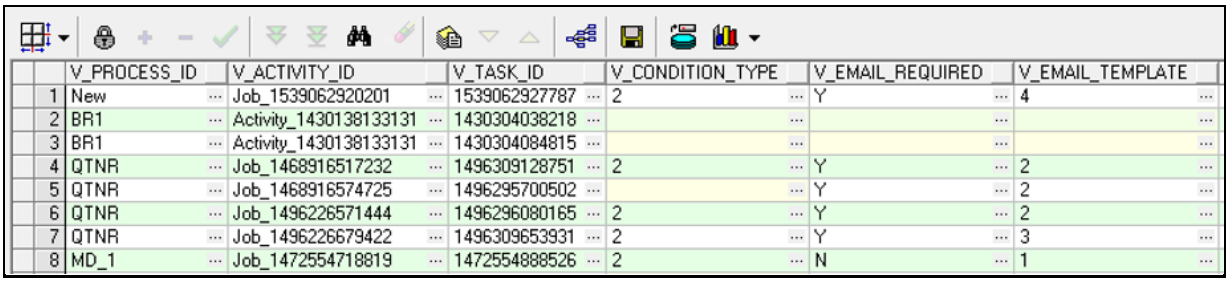
13.4 AAI_WF_ACTIVITY_TASK_B Table

The `AAI_WF_ACTIVITY_TASK_B` table holds the 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 the AAI_WF_EMAIL_TEMPLATE table.

Figure : AAI_WF_ACTIVITY_TASK_B Table



	V_PROCESS_ID	V_ACTIVITY_ID	V_TASK_ID	V_CONDITION_TYPE	V_EMAIL_REQUIRED	V_EMAIL_TEMPLATE
1	New	Job_1539062920201	1539062927787	2	Y	4
2	BR1	Activity_1430138133131	1430304038218			
3	BR1	Activity_1430138133131	1430304084815			
4	QTNR	Job_1468916517232	1496309128751	2	Y	2
5	QTNR	Job_1468916574725	1496295700502		Y	2
6	QTNR	Job_1496226571444	1496296080165	2	Y	2
7	QTNR	Job_1496226679422	1496309653931	2	Y	3
8	MD_1	Job_1472554718819	1472554888526	2	N	1

13.5 AAI_USER_PREFERENCE Table

In this table, you can set the user preference for receiving the notification emails.

Table : Email Notification User Preference

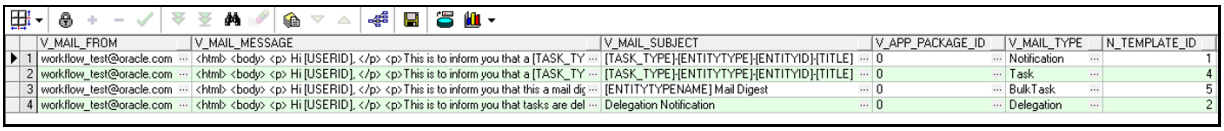
V_USER_ID	N_EMAIL_NOTIF_REQ
USER1	1
USER2	2

- 0 – To receive no notification emails
- 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 is sent with all the pending notifications from the last trigger, as a PDF attachment. After the bulk mail is sent, the `V_BULK_MAIL_TRIGGER` value is automatically set to N.
- 3 – To get mail with attachment

13.6 AAI_WF_EMAIL_TEMPLATE Table

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

Figure : AAI_ WF_EMAIL_TEMPLATE Table



	V_MAIL_FROM	V_MAIL_MESSAGE	V_MAIL_SUBJECT	V_APP_PACKAGE_ID	V_MAIL_TYPE	N_TEMPLATE_ID
1	workflow_test@oracle.com	<html> <body> Hi [USERID], This is to inform you that a [TASK_TY	[TASK_TYPE]-[ENTITYTYPE]-[ENTITYID]-[TITLE]	0	Notification	1
2	workflow_test@oracle.com	<html> <body> Hi [USERID], This is to inform you that a [TASK_TY	[TASK_TYPE]-[ENTITYTYPE]-[ENTITYID]-[TITLE]	0	Task	4
3	workflow_test@oracle.com	<html> <body> Hi [USERID], This is to inform you that this a mail dig	[ENTITYTYPENAME] Mail Digest	0	BulkTask	5
4	workflow_test@oracle.com	<html> <body> Hi [USERID], This is to inform you that tasks are del	Delegation Notification	0	Delegation	2

13.7 AAI_WF_BULK_MAIL_TRIGGER Table

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

13.8 CSSMS_USR_PROFILE Table

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

Table : Email ID and User ID for Email Notification

V_USR_ID	V_EMAIL
USER1	user1@oracle.com
USER2	user2@oracle.com

13.9 AAI_MAIL_AUDIT_TRAIL Table

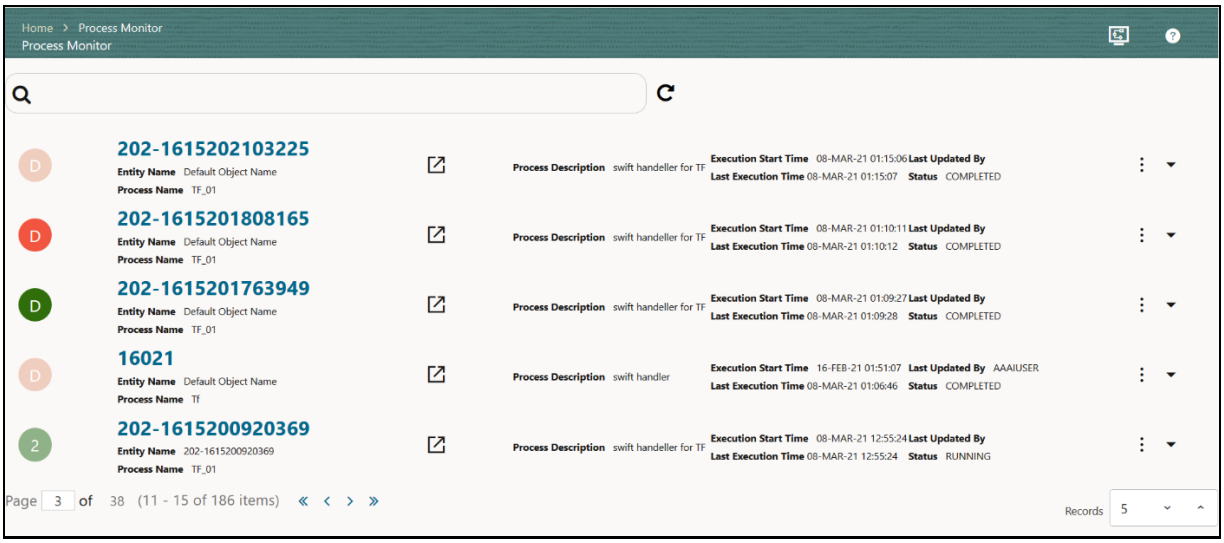
This is where all email trails are stored. The status changes after the emails are sent. This can be used for debugging purposes.

14 Process Monitor

The Process Monitor is used to monitor the 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 needs to be mapped to the function role WFMACC (Workflow Monitor Access) to access the Process Monitor window.

Topics:

Figure : Process Monitor Window

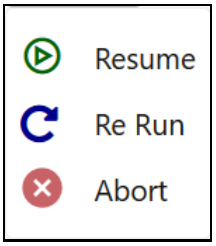


This window displays all the Workflows that are invoked from the Application with details such as Entity Name, Entity ID, Process Name, Process Description, Execution Start Time, Last Execution Time, Last Updated By, and Status.

Click to open the Process Monitor in a new window.

Click to display the following submenu for Run Pipeline:

Figure : Process Monitor Submenu



- Resume: Select to resume a Run Pipeline.

- Re-run: Select to execute a Run Pipeline again irrespective of the previous execution status.
- Abort: Select to abort an ongoing Run Pipeline.

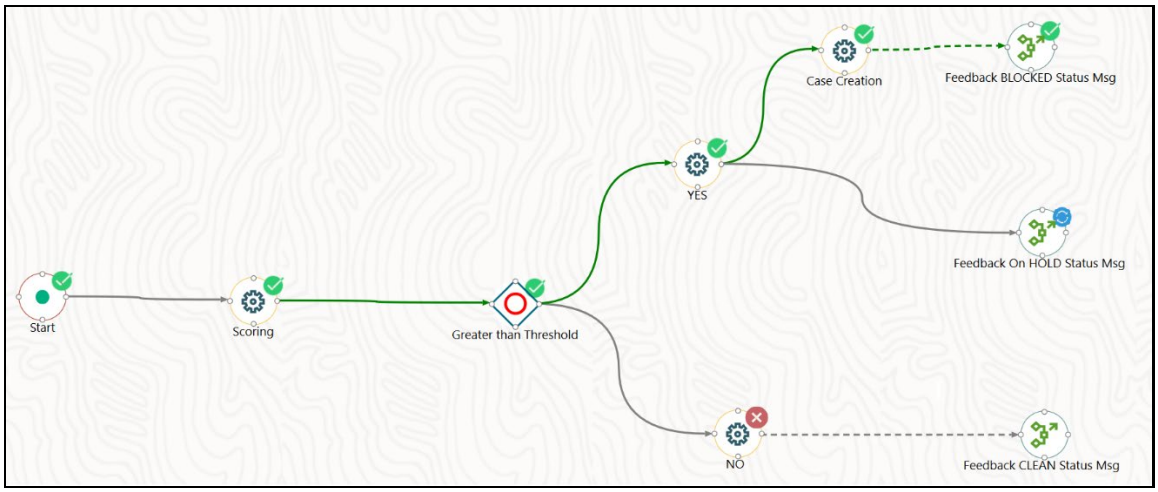
Using the **Search** grid, you can search for a specific Pipeline by providing a keyword from Process ID, Process Name, or Process Description and clicking . Click to reset the Search fields.

You can sort the Processes displayed in the **Process Monitor** window based on Entity Id, Entity Name, Process ID, or Process Name. Click to go to the **Process Modeller** window.

14.1 Monitoring a Business Process

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

Figure : Monitoring a Business Process



In the **Process Monitor** window, the status of the activity is represented as follows:

- : This indicates that the execution of the activity is successful.
- : This indicates that the activity is currently running or waiting for the user's input to proceed.
- : This indicates that the execution of activity is failed.
- No icon overlapping the activity indicates that it is yet to be executed.

Double click the **Sub Pipeline** icon to monitor the tasks inside them.

Click to refresh the window.

You can also set auto-refresh in the header. Select **Enabled** in **Auto Refresh**, enter a value from 1 to 10 in **Refresh Interval (In Min)** and then click **Apply**.

Figure : Configure Auto Refresh

The **Activity Execution** window shows all the execution stages of the process. Click anywhere outside the window to close the **Activity Execution** window.

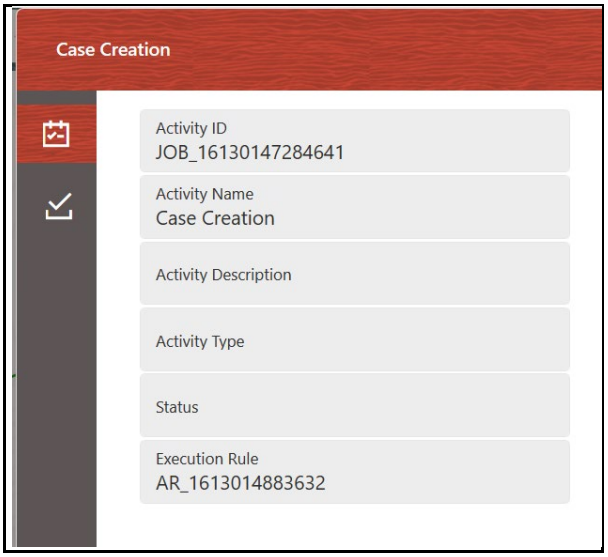
14.2 Viewing Activity Execution Logs

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

To view Activity execution logs:

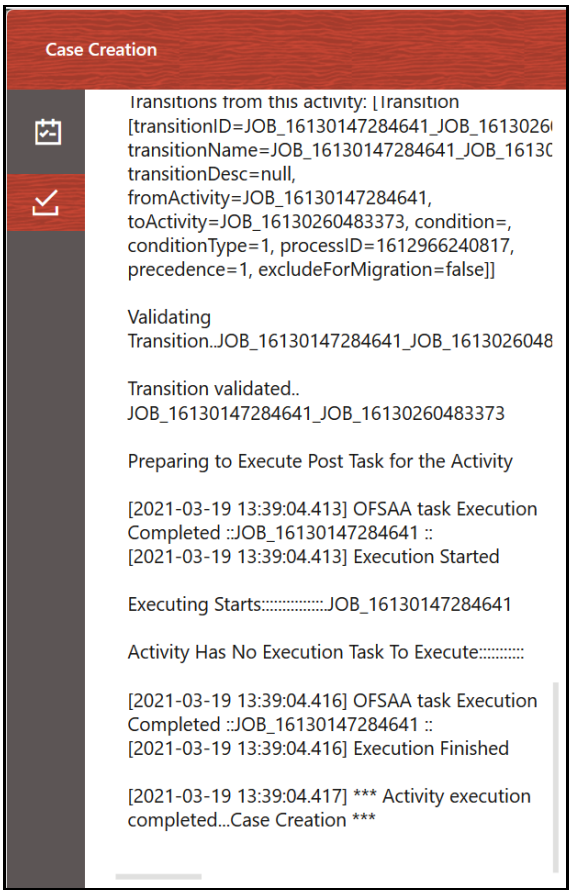
1. Double-click an Activity to view the **Activity Execution** window. The Activity Definition details are displayed.

Figure : Activity Execution Details Window to View the Activity Execution Logs



2. Click the **Execution Log** icon to view the activity execution log.

Figure : Activity Execution Logs Window



The log shows all the execution stages of the selected Activity if it is already executed. Otherwise, it is blank.

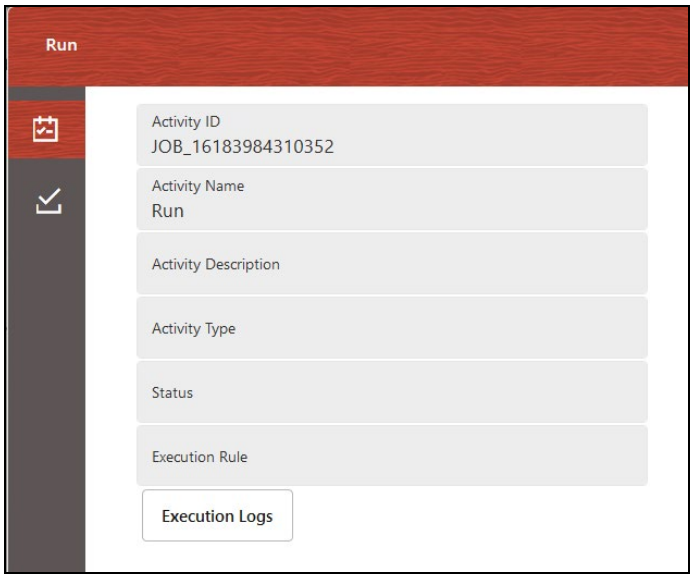
14.3 Viewing Execution Log for Widgets

This option allows you to view the execution logs for Widgets. This option is not available for Service Tasks or Human Tasks.

To view Execution logs, do as follows:

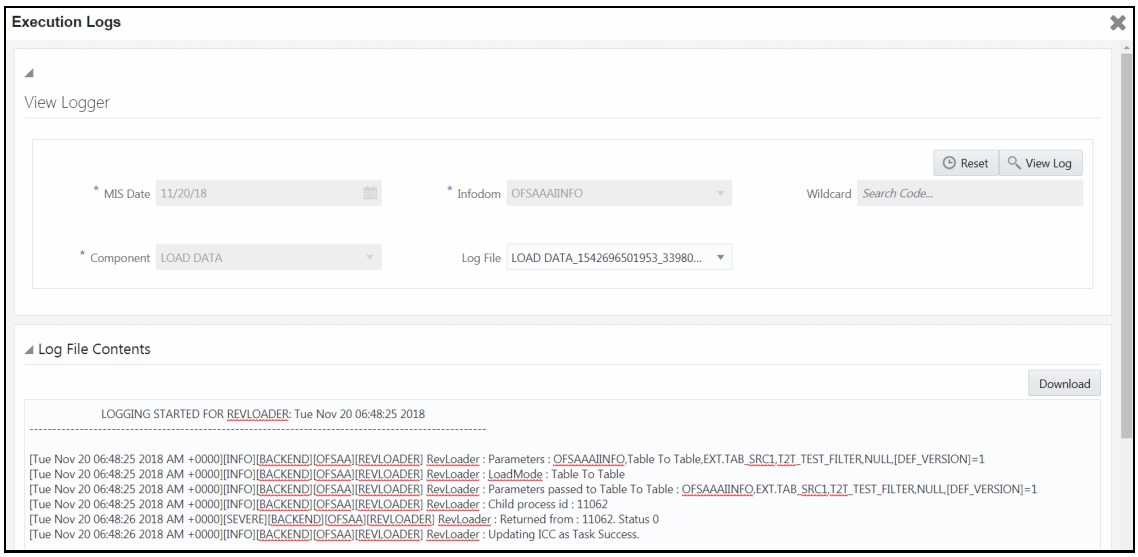
1. Double-click on the activity icon whose logs you want to view. The Activity Execution details are displayed.

Figure : Activity Execution Details Window to View the Execution Log for Widgets



2. Click **Execution Logs**.
3. Select the required **Log File** from the drop-down list and click **View Log**. The log information is displayed in the **Log File Contents** window

Figure : Log File Contents window



4. Click **Download** to download the execution log details.

15 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

15.1 Application UI

An application can invoke PMF workflow in the following ways:

15.1.1 Java API

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

15.1.2 Stored Procedure

```
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
```

15.1.3 Rest Service

```
URL: <contextPath>/PMFService/startWorkflowProcess

Method: POST

Consumes("application/json")

Produces("text/plain")

Sample Input Params:

"{\"SummaryPayload\":{\"objectid\":\"123\",\"objecttype\":\"QTNR\",\"in
fodom\":\"OFSAAAIINFO\",\"segment\":\"OFSAAAIINFO\",\"userid\":\"AAAIUS
ER\",\"locale\":\"en_US\",\"securitymap\"

:{},\"applicationparams\"

:{\"testparam\":\"value1\",\"testparam2\":\"value2\"}}}";
```

15.2 Within PMF Summary Screen UI

15.2.1 Using Execute Run

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

15.3 Operations Module

Execution of a PMF Pipeline can be triggered from the Operations module as a batch.

1. From the **Batch Maintenance** window under the Operations module, create a new Batch.

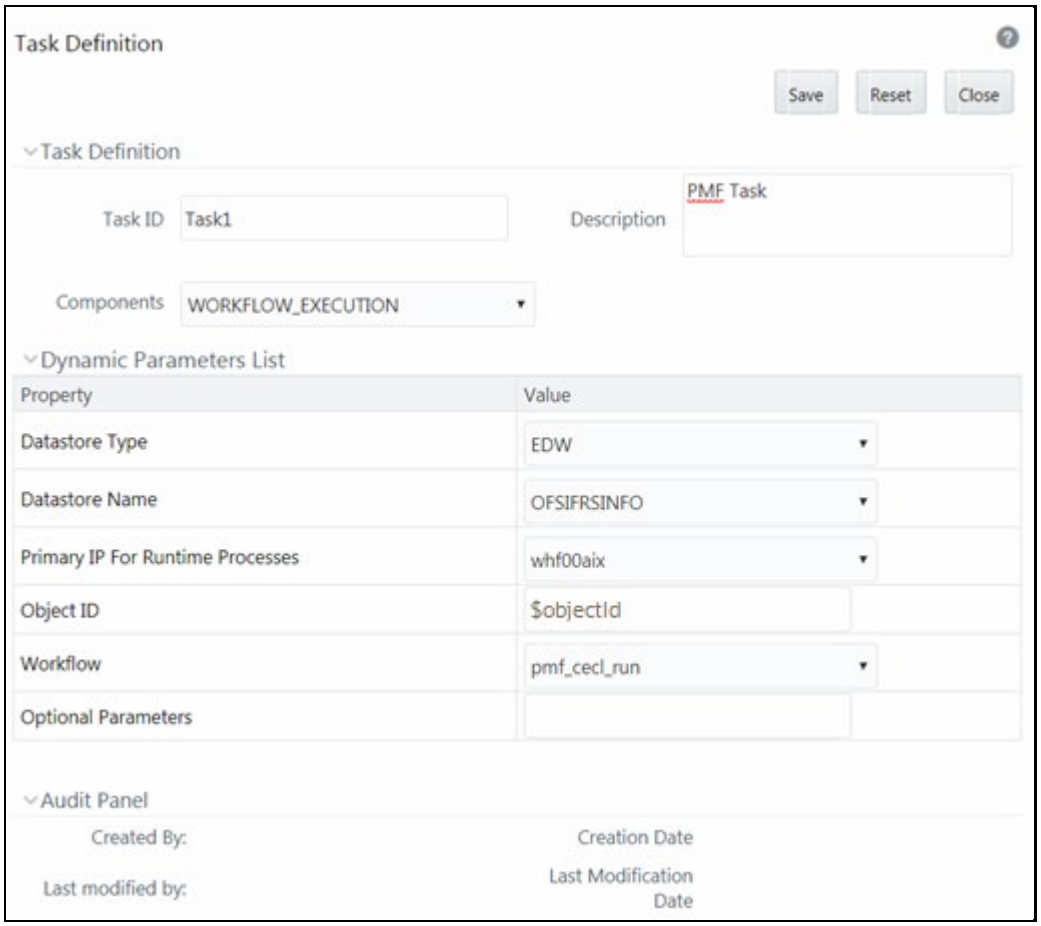
   For more information, see **Adding Batch Definition** section in the OFS Analytical Applications Infrastructure User Guide.

2. Create a new Task with a task component as Workflow Execution.

   For more information, see **Adding Task Details** section in the OFS Analytical Applications Infrastructure User Guide.

> **NOTE** To avoid entering the Object ID each time you run a batch, enter the value `$objectId` in the **Object ID** field. This automatically generates the ID in the OFSAA system.

Figure : Task Definition Window to Create a New Task



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 the **Batch Execution** section in the OFS Analytical Applications Infrastructure User Guide.

15.4 Command Line Execution

A shell script file `wfExec.sh` is available in the `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.

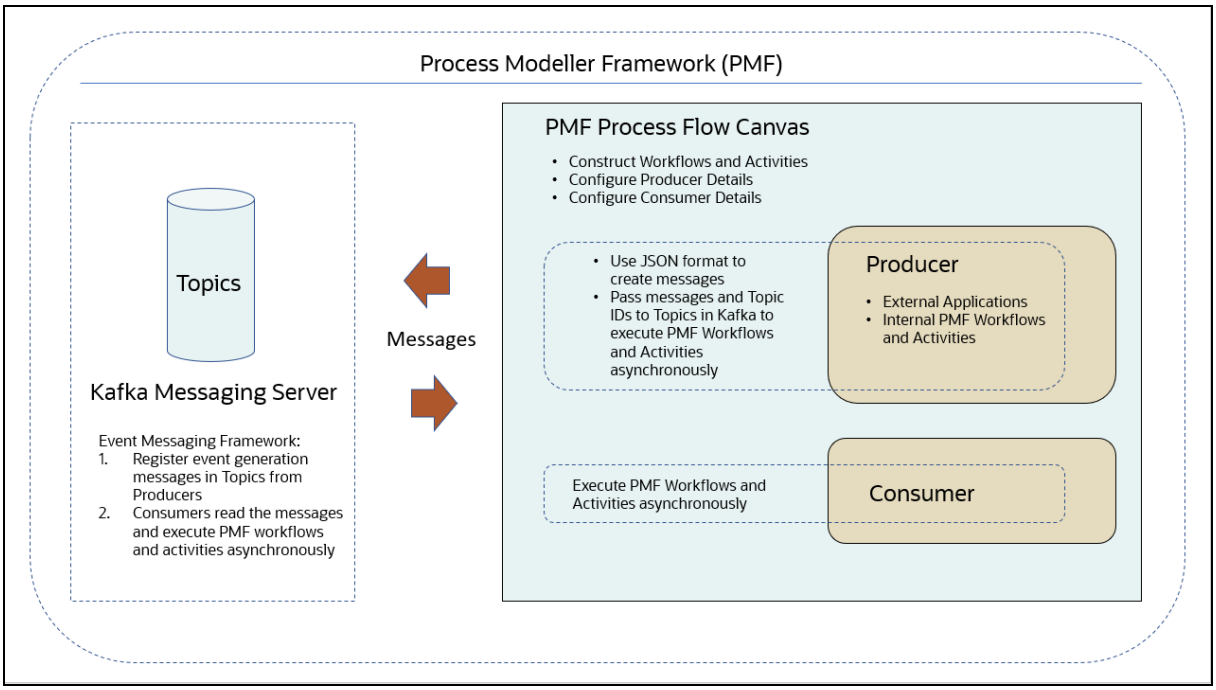
16 Event-Based Orchestration of Process Flow

The Event Framework integrated into PMF is an event-driven messaging framework that uses Apache Kafka (messaging server) to process information asynchronously and in near real-time. Producers that are either external applications or PMF internal processes, send messages (notify about an event occurrence) to Kafka Topic(s). Consumers, which are internal processes within PMF, read the messages from Kafka Topic(s) and execute other events or activities configured in the PMF Canvas. The framework uses the REST transfer protocol to send messages and the JSON data format to construct the messages.

> **NOTE** The Event-Based Orchestration of Process Flow (based on Apache Kafka) feature requires that you apply the OFSAAIE Pack License in the OFSAA setup.
>
> For more information, see the OFSAAI Extension Pack Documentation.

The notification of the generation of an event to a Topic can execute a workflow or any individual task asynchronously. Previously, the orchestration of workflows in PMF followed a synchronous pattern and executed the activities in the order of sequence configured. In other words, the execution or completion of the activity was required before another activity in the workflow could be executed. However, with the integration of the Event Framework, in addition to the existing sequential workflow, you can now also execute asynchronously not only the workflows constructed in the PMF canvas but also individual activities within the workflows and in any order of the desired configuration.

Figure : The Event-Based Process Flow Workflow



Topics:

16.1 Producer Activity

The Event Framework enables the Producers to create messages as per the JSON definition and invokes the Producer API by passing the message and the Topic ID. This message is recorded in the Event Framework Audit Table for reference purposes.

The Producer and Consumer of messages are mapped to the same Topic (a unique identifier for a data stream in Kafka) and the exchange of data takes place (In PMF, execution of Process Flows and Activities).

See the JSON Definition for Events section for information about JSON definition formats and examples.

16.2 Consumer Activity

The Event Framework enables the Consumer (internal processes in PMF) to make asynchronous requests to execute activities in a process flow based on the conditions of event occurrences. The messaging service can also be used by external applications to post messages to Topics in Kafka to execute events in individual components or execute the whole process asynchronously. In other words, activities do not have to wait for other activities to be completed (as in a synchronous request) and can be messaged to execute events scheduled for the specific activities.

The Producer and the Consumer of the messages routed through the Kafka messaging server are mapped to the same topic (mapped by Producer ID, Topic ID, and Consumer ID) for the following metadata types:

- Event Type
- Event Sub Type
- Event Source

16.3 Seed Data in the Event and Event Subtype Tables

The Event and Event Subtype Tables require the initial population of data (seeding) as a configuration step.

> **NOTE** Before you can use the Event-based Orchestration of Process Flow, seed the data as discussed in this section. The seeding of data is a prerequisite configuration.

To Seed Data in the Event and Event Subtype tables, configure the arguments and Run the following script to add data to the Event Tables in the AAI Config Schema:

```
./SaveEventInfo.sh <EVENT_TYPE> <EVENT_SUB_TYPE> <EVENT_SOURCE>
<EVENT_DESCRIPTION> <LOCALE_CODE>
```

Table : Argument Description Table for Event Subtype Tables

Argument	Description
EVENT_TYPE	The type of classification at the Parent level for the various tasks. For example: The type of product such as Insurance, loans, and Deposits.
EVENT_SUB_TYPE	The further classification of the Parent level Event Type. For example: For the Loan Event Type, the subtypes such as long-term loans, short-term loans, and loan transfers.
EVENT_SOURCE	The Source of the OFSAA Application for the Event to be added.
EVENT_DESCRIPTION	The description of the Event that is to be added.
LOCALE_CODE	The locale for the UI.

For example:

```
./SaveEventInfo.sh ECMEVENT AML_DD ECM 'ECM_Type_EVENT' en_US
```

> **NOTE** For more information about additional configurations related to Event Framework Metadata, see Appendix C: Set Up Event Framework Metadata.

16.4 Create an Event Process Flow in the Process Modeller

Use the Activities in the PMF Canvas to create an Event Workflow and execute it so that the process of communication of data between the Producers (external applications or PMF internal processes) and

consumers (PMF internal processes) is established and the events are triggered from the PMF Canvas. Configure to execute each task independently as events in the process flow.

To create an Event Process Flow, follow these steps:

1. Log in to the OFSAA Application.
2. Click **Applications** from the Header to display the applications in the Tiles menu.
3. Select an Application.

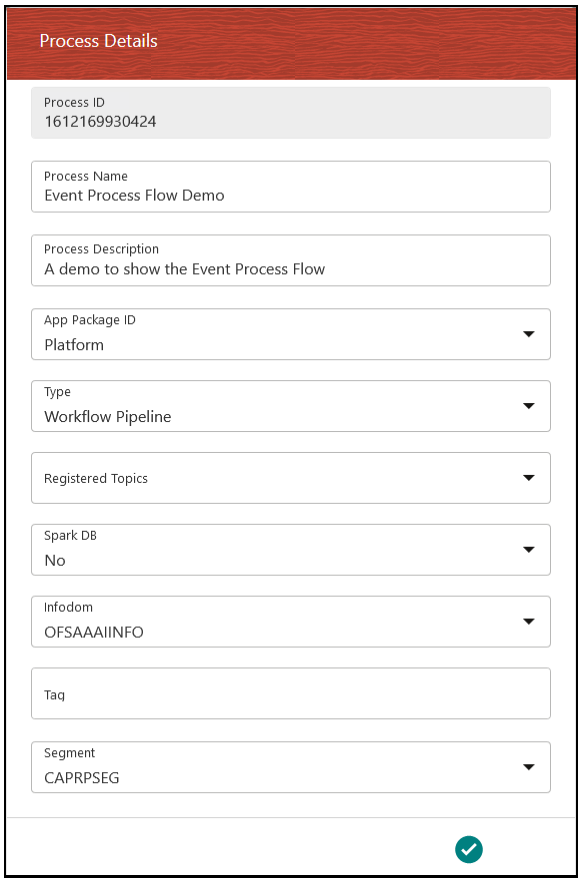
   For example, Financial Services Enterprise Modeling.

   The Navigation Tree displays a menu.
4. Click **Common Tasks**,

> **NOTE** The navigation path to access the **Process Modelling Framework** window can vary in applications based on the Menu Tree configuration.

5. Select **Process Modelling Framework** to display a submenu.
6. Select **Process Modeller** from the submenu to display the **Process Modeller** window.
7. Click the + icon in the window to display the **Process Details** drawer window.

Figure : The Process Details Drawer Window

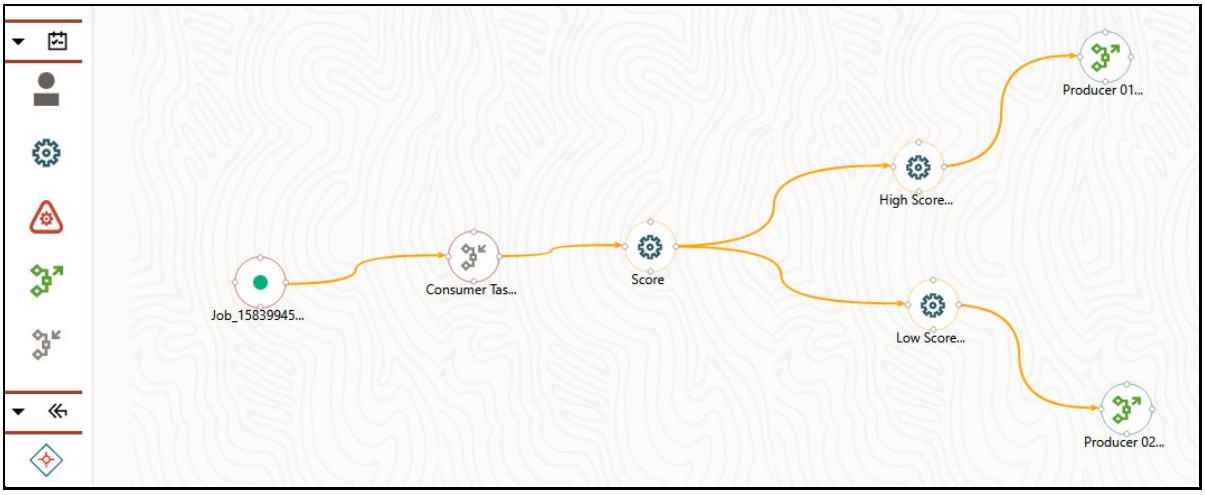


8. In the **Process Details** drawer window, enter the required details as follows:
   a. Enter a name in the **Process Name** field. The **Process ID** field is created by the application and is read-only.
   b. Enter a description for the process in the **Process Description** field.
   c. Select **Platform** from the **App Package ID** drop-down.
   d. Select **Workflow Pipeline** from the **Type** drop-down.
   e. Do not select any value from **Registered Topics** as it does not apply to an event process.
   f. Do not select any value from **Spark DB** as it does not apply to an event process.
   g. Select the required Infodom from the **Infodom** drop-down.
   h. Enter any appropriate tags in the **Tag** field. This field is optional.
   i. Select the required application segment from the **Segment** drop-down.
9. After entering the details, click the **Accept** icon to save it. The Process Flow is created and is displayed in the PMF Canvas. By default, a job of the type **START** is created on the Canvas.
10. Drag and drop the required activities from the **Activity** menu into the Canvas to construct the Process Flow.

11. Drag and drop the **Event Producer** and **Event Consumer** activities into the Canvas.
12. Click the icon to save the process definition.

The following illustration provides a simple example of the Event Process Flow with Producer and Consumer activities displayed on the Canvas. For more information about configuring Consumers and Producers, see the Configure an Event Consumer and Configure an Event Producer sections.

Figure : A Simple Representation of Event Process Flow on the PMF Canvas



16.5 Configure an Event Producer

After adding an Event Producer activity on the Canvas, configure the activity. You can add details for Activity, Transition, and Notifications.

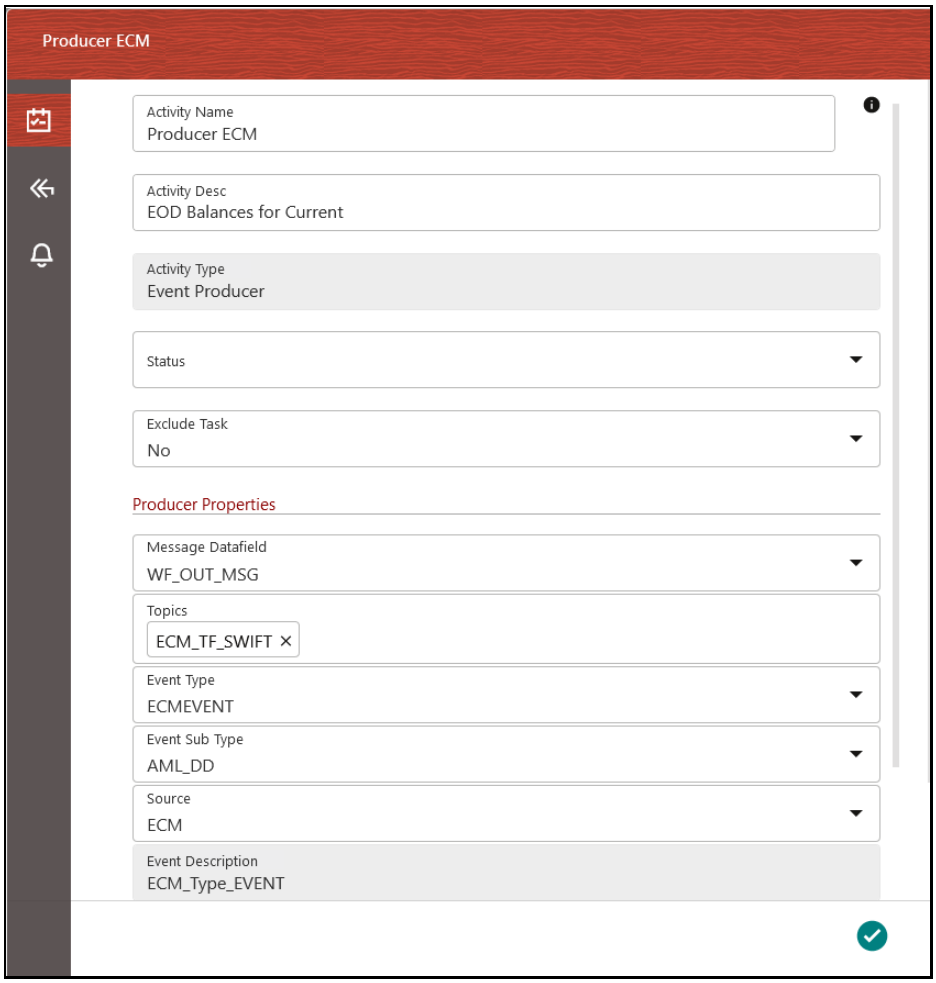
To configure an Event Producer, follow these steps:

1. Click an Event Process Flow in the **Process Modeller** window to display it in the PMF Canvas.
2. Double-click the **Event Producer** activity icon, or click the **Edit Activity** icon that displays when you mouse over the activity, to display the activity configuration details in the **Producer Activity** drawer window.

   The window displays three tabs: **Activity**, **Transition**, and **Notifications** The Activity tab requires mandatory details, whereas the Transition and Notifications tabs are optional.

To configure details in the Activity tab, follow these steps:

Figure : The Producer Activity Tab



a. Enter a name for the Producer Event in the **Activity Name** field.

b. Enter a description for the Producer Event in the **Activity Description** field. By default, the **Activity Type** field displays as Event Producer and is read-only.

c. Select the appropriate status from the **Status** drop-down.

d. Select **Yes** or **No** from the **Exclude Task** drop-down.

e. In the **Producer Properties** section, select the required type of message for the metadata from the **Message Datafield** drop-down.

f. Select the required topics from the **Topics** drop-down.

Topics is a virtual group(s) that stores or publishes data. It can have multiple consumers subscribing for the information.

g. Select the required event type from the **Event Type** drop-down.

The type of classification at the Parent level for the various tasks.

For example:

The type of the product such as Insurance, Loan, and Deposits.

h. Select the required event subtype from the **Event Sub Type** drop-down.

The further classification of the Parent-level Event Type.

For example:

The subtype for the Loan Event Type such as long-term loans, short-term loans, and loan transfers.

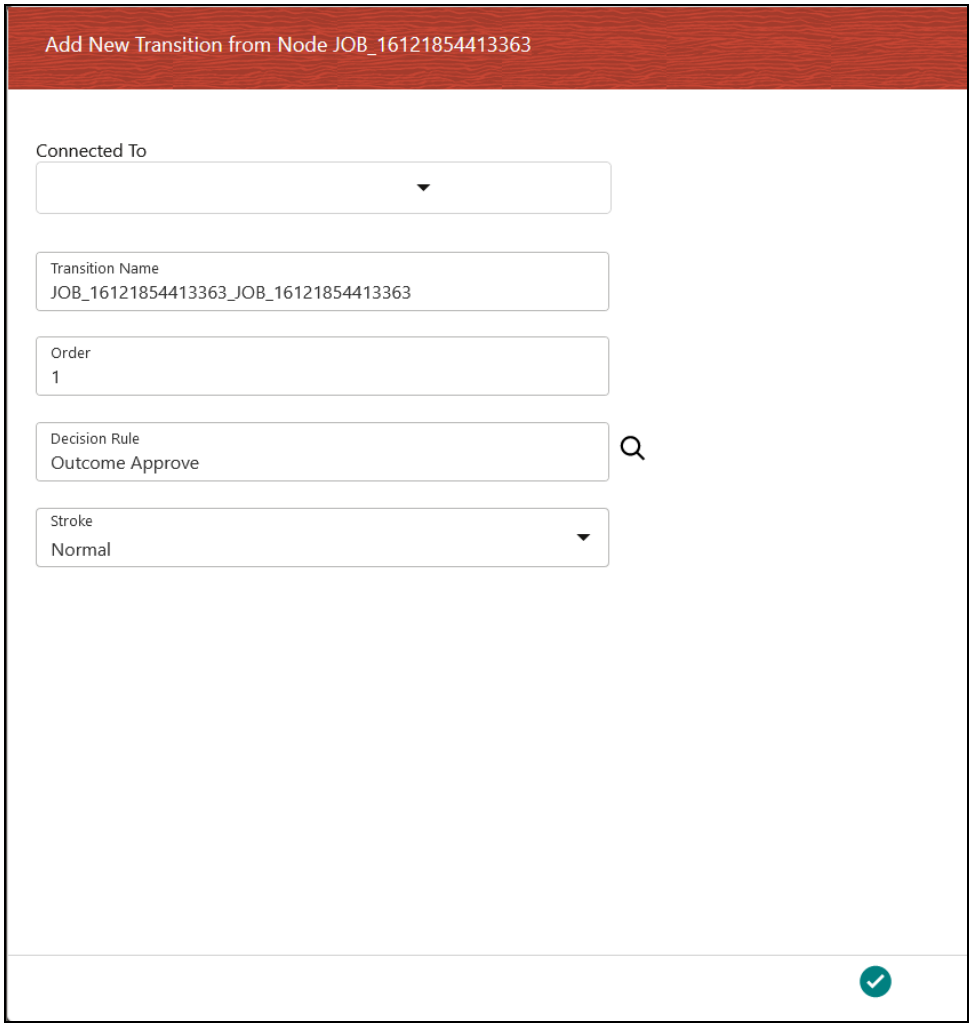
i. Select the required source OFSAA application from the **Source** drop-down.

The **Event Description** field is populated based on the selection in the **Event Sub Type** drop-down.

j. After entering the details, click the **Accept** icon to save the details.

To configure details in the Transition tab, follow these steps:

Figure : The Transition Tab



a. Click the **Transition** tab.

b. Click the icon to display the fields required for transition details.

c. Select the required job to be connected to from the **Connected To** drop-down.

d. Enter a name for the transition in the **Transition Name** field.

e. Enter the order of execution in the **Order** field. For example, 1 marks the order as to be executed first.

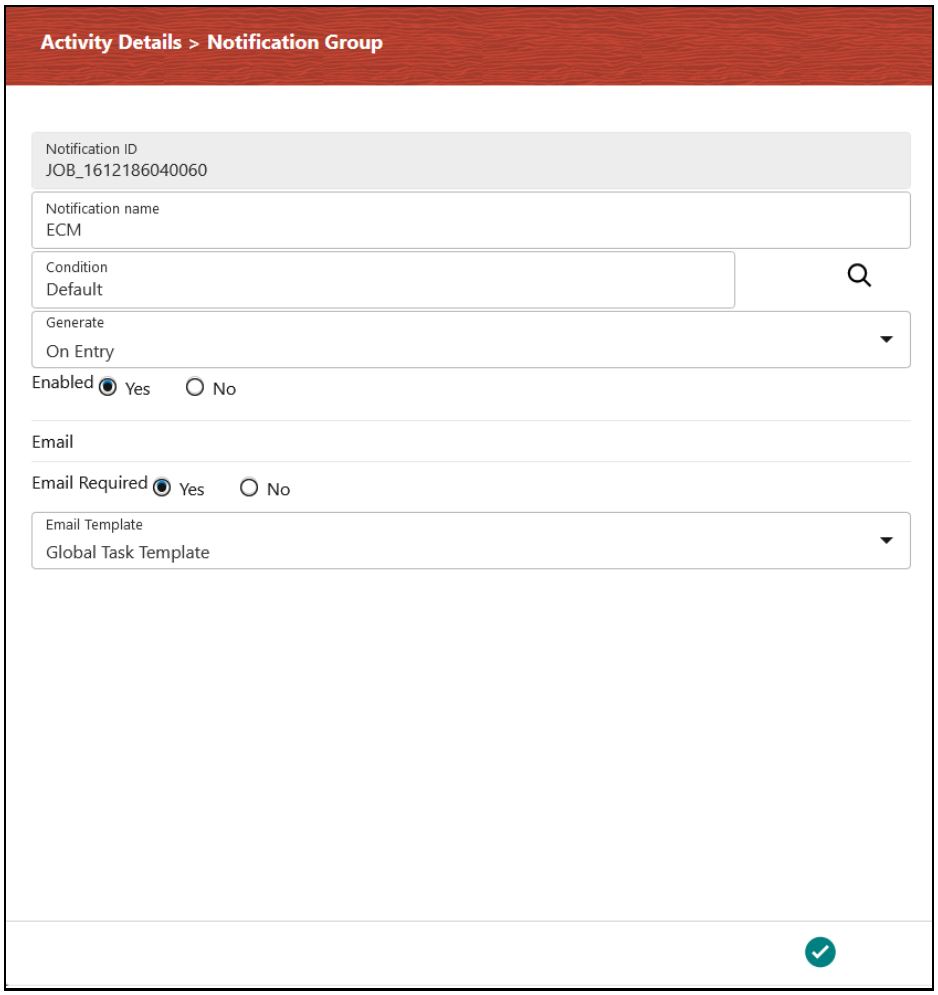
f. Click the **Search** icon from the **Decision Rule** drop-down and select the required application rule. Click the **Accept** icon to save the details.

g. Select the stroke format from the **Stroke** drop-down.

h. After entering the details, click the **Accept** icon to save the details.

To configure details in the Notifications tab, follow these steps:

Figure : The Notifications Tab



a. Click the **Notifications** tab.

b. Click the  icon to display the fields required for the Notifications.

c. Enter a name for the notification in the **Notification Name** field. The **Notification ID** field is auto-populated based on the ID from the application.

d. Click the **Search** icon from the **Condition** drop-down and select the required application rule. Click the **Accept**  icon to save the details.

e. Select the condition to generate the notification from the **Generate** drop-down.

f. Select **Yes** in the **Enabled** field to activate the notification.

g. In the **Email** section, select **Yes** in the **Email Required** field to select to receive the notification through email. Select **No** to disable this option.

h. Select the required email template from the **Email Template** field.

i. Click the **Accept**  icon to save the details.

16.6 Configure an Event Consumer

After adding an Event Consumer activity on the Canvas, configure the activity. You can add details for Activity, Consumer, Implementation, and Notifications.

To configure an Event Consumer, follow these steps:

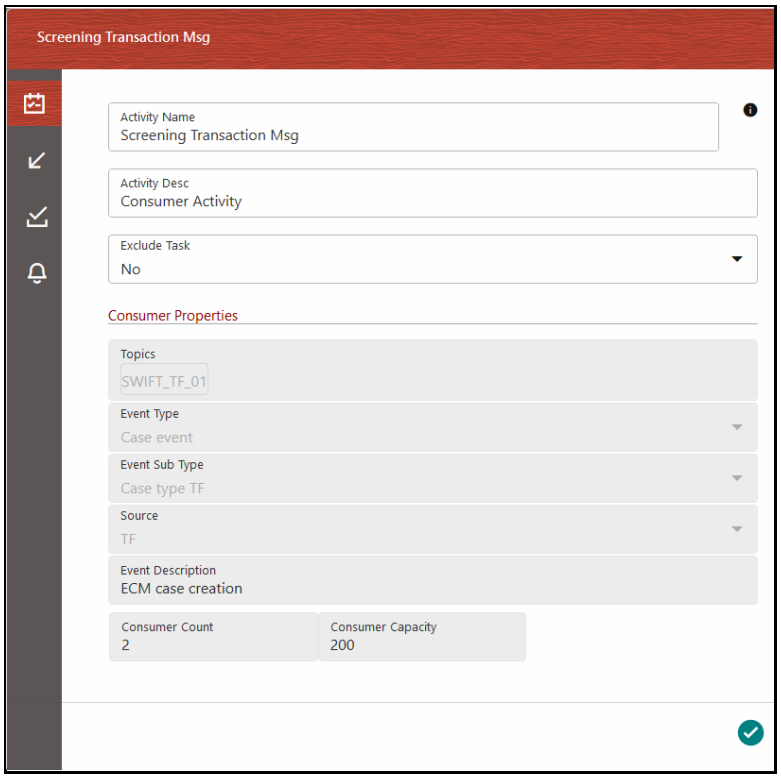
1. Click an Event Process Flow in the **Process Modeller** window to display it in the PMF Canvas.

2. Double-click the **Event Consumer**  activity icon, or click the **Edit Activity**  icon that displays when you mouse over the activity, to display the activity configuration details in the **Producer Activity** drawer window.

   The window displays four tabs: **Activity**  , **Consumer**  , **Implementation**  , and **Notifications**  The details in the Notification tab are optional.

   To configure details in the Activity tab, follow these steps:

Figure : The Consumer Activity Tab



a. Enter a name for the Consumer Event in the **Activity Name** field.

b. Enter a description for the Consumer Event in the **Activity Description** field. By default, the Activity Type field displays as Event Consumer and is read-only.

c. Select the appropriate status from the **Status** drop-down.

d. Select **Yes** or **No** from the **Exclude Task** drop-down.

e. In the **Consumer Properties** section, the following fields are displayed based on the selection in the Configure an Event Producer window:

f. The selected topic from the **Topics** drop-down.

g. The selected event type from the **Event Type** drop-down.

h. The selected event subtype from the **Event Sub Type** drop-down.

i. The selected source OFSAA application from the **Source** drop-down.

The **Event Description** field is populated based on the selection in the Event Sub Type drop-down.

j. Consumer Count – The number of threads to be created for a given consumer.

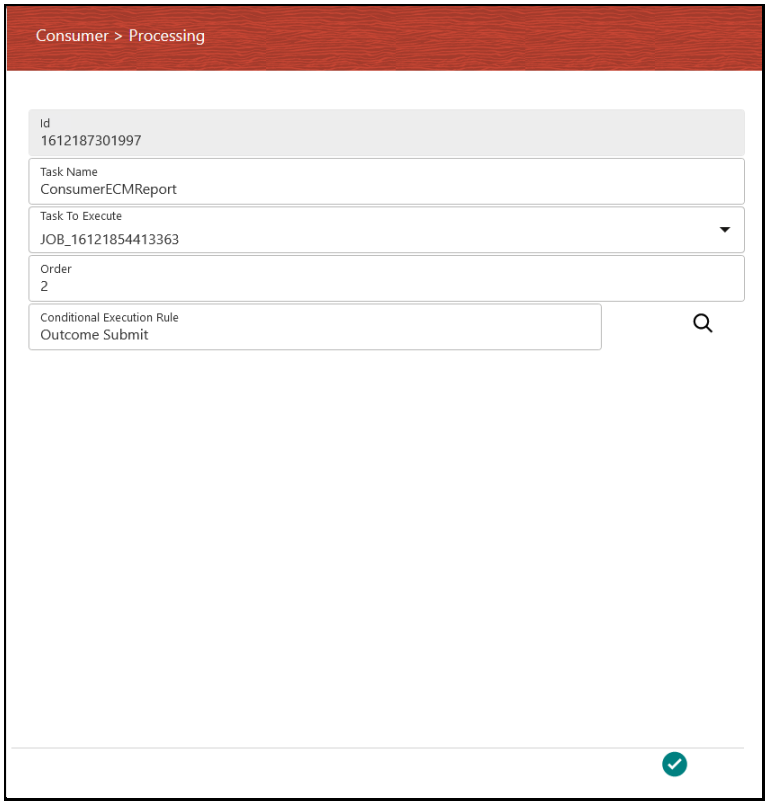
k. Consumer Capacity – indicates the number of messages that each of the threads can process.

NOTE By default, the Consumer Count is 2 and Consumer Capacity is 200.

l. After entering the details, click the **Accept** icon to save the details.

To configure details in the Consumer tab, follow these steps:

Figure : The Consumer Tab



a. Enter a name for the consumer task in the **Task Name** field. By default, the **ID** field displays the unique identifier for the consumer task.

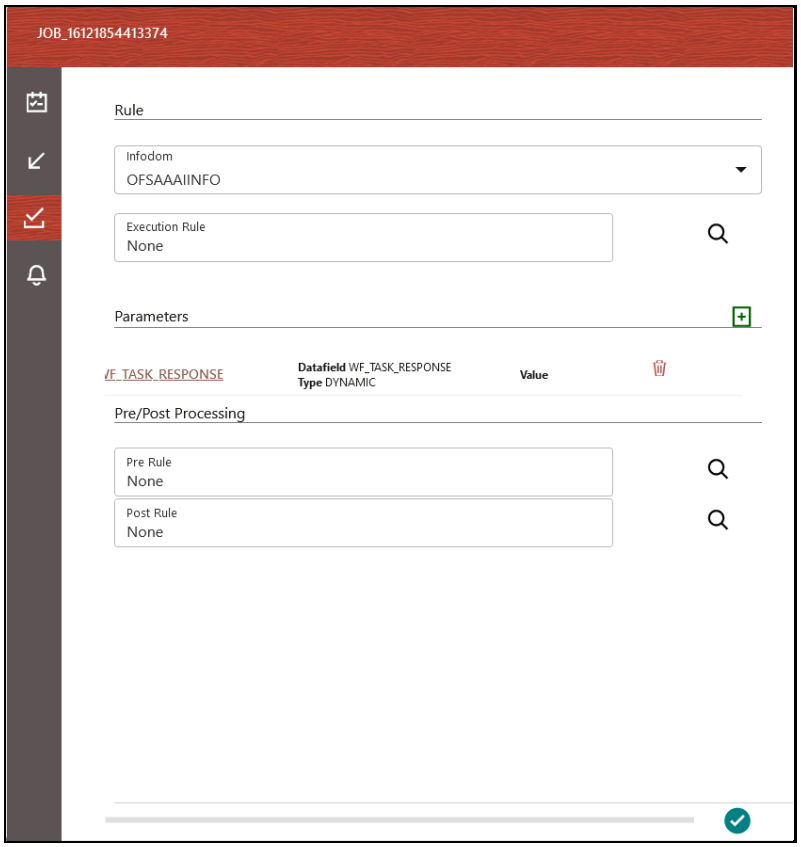
b. Select the appropriate jobs from the **Tasks To Execute** drop-down.

c. Enter the order of execution in the **Order** field. For example, 1 marks the order as to be executed first.

d. Click the **Search** icon from the **Conditional Execution Rule** drop-down and select the required application rule. Click the **Accept** icon to save the details.

To configure details in the Implementation tab, follow these steps:

Figure : The Implementation Tab



a. Click the **Implementation** tab to select the rules required for the execution.

b. Select the required infodom from the **Infodom** drop-down.

c. Click the icon to display the popup with fields required for the **Parameters** section.

d. Select the required job to be connected to from the **Connected To** drop-down.

e. Select the required data field from the **Data Fields** drop-down.

f. Select the required parameter type from the **Parameter Type** drop-down. The options are **Dynamic** and **Static**. If you select Static, then enter the value in the **Value** field. Click the **Accept** icon to save the details.

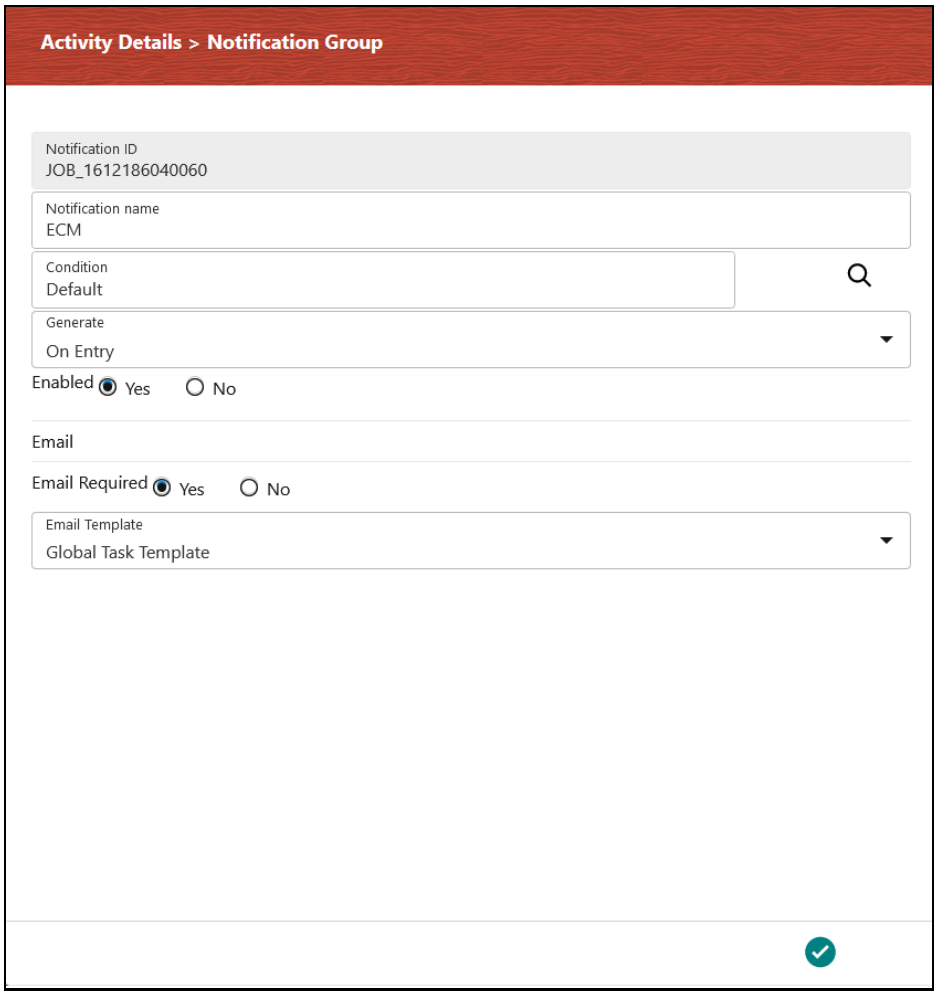
g. In the Pre/Post Processing section, click the **Search** icon from the **Pre Rule** drop-down and select the required application rule to be executed before the processing of the event. Click the **Accept** icon to save the details.

h. Click the **Search** icon from the **Post Rule** drop-down and select the required application rule to be executed after the processing of the event. Click the **Accept** icon to save the details.

i. After entering the details, click the **Accept** icon to save the details.

To configure details in the Notifications tab, follow these steps:

Figure : The Notifications Tab



a. Click the **Notification** tab.

b. Click the **Add** icon to display the fields required for the Notifications.

c. Enter a name for the notification in the **Notification Name** field. The **Notification ID** field is auto-populated based on the ID from the application.

d. Click the **Search** icon from the **Condition** drop-down and select the required application rule. Click the **Accept** icon to save the details.

e. Select the condition to generate the notification from the **Generate** drop-down.

f. Select **Yes** in the **Enabled** field to activate the notification.

g. In the **Email** section, select **Yes** in the **Email Required** field to select to receive the notification through email. Select **No** to disable this option.

h. Select the required email template from the **Email Template** field.

i. Click the **Accept** icon to save the details.

16.7 Use Case: Event-Based Framework Execution in Real-time Transaction Monitoring

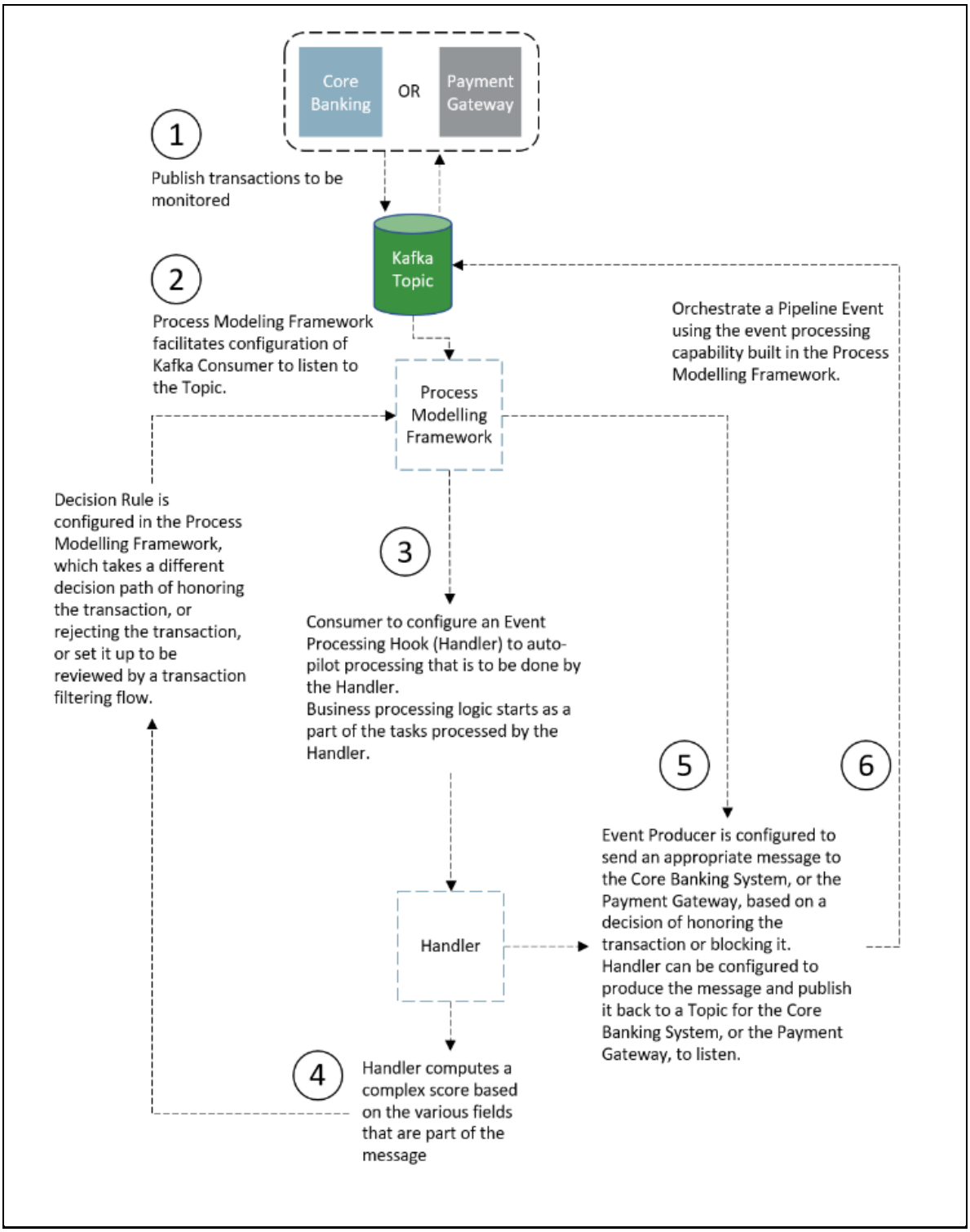
The operational mechanism of the Event Processing Pipeline is better understood with a Use Case that describes the execution of real-time Transaction Monitoring.

The following steps describe the Use Case Configuration:

1. The transactions that are to be monitored and honored are published by the Core Banking System, or the Payment Gateway, to a predefined Kafka Topic.
2. The Process Modeling Framework facilitates the configuration of a Kafka Consumer which listens to the Topic.
3. The Consumer allows configuring an Event Processing Hook (Handler) which auto-pilots any further processing that is to be done by the Handler. The Business Processing Logic starts as a part of the tasks processed by the Handler.
4. The Handler computes a complex score based on the various fields that are part of the message.

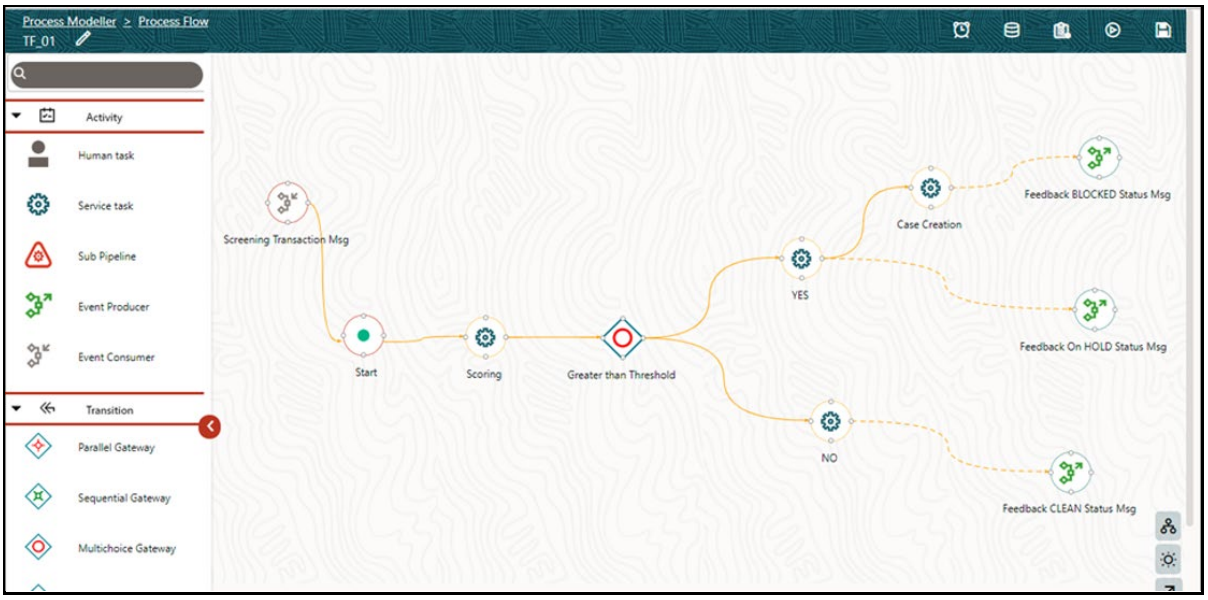
   For example, the transaction amount added to the transaction withdrawal profile set for the customer for a specific look-back period exceeds a threshold and so cannot be honored.
5. A Decision Rule is configured in the Process Modelling Framework, which takes a different decision path of honoring the transaction, or rejecting the transaction, or set it up to be reviewed by a transaction filtering flow.
6. The Event Producer, which shows up further in the flow, is configured to send an appropriate message to the Core Banking System, or the Payment Gateway, based on a decision of honoring the transaction or blocking it. The Handler can be configured to produce the message and publish it back to a Topic for the Core Banking System, or Payment Gateway, to listen.
7. A loosely coupled Pipeline Event can be orchestrated using the event processing capability built in the Process Modelling Framework.

Figure : Use Case Flow Chart for Transaction Monitoring



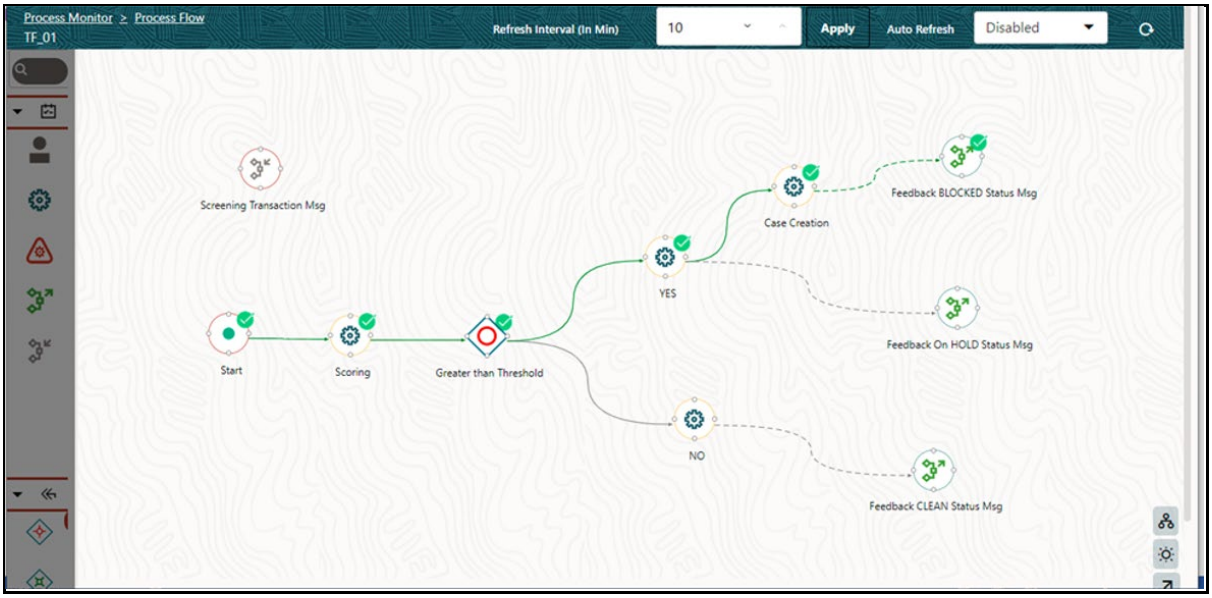
The following illustrations of the Process Monitoring Window provide a visual view of the execution path the message has traveled:

Figure : Process Monitoring Window Showing Execution Path



The execution path of the transactions and the various data fields that determined the execution path can be monitored using the monitoring capability within PMF. The information shows the lineage of the data flow which has resulted in the transaction being honored or blocked or put on hold.

Figure : Process Monitoring Window Showing Lineage of Data Flow



17 Timer

You can use the Timer feature in Process Modeller to execute activities within a Process Flow, at a set time and frequency in the future. If you have set Timers to auto-trigger the execution of the PMF process, then after a certain duration it will re-trigger, if activities pause.

Topics:

17.1 Prerequisites

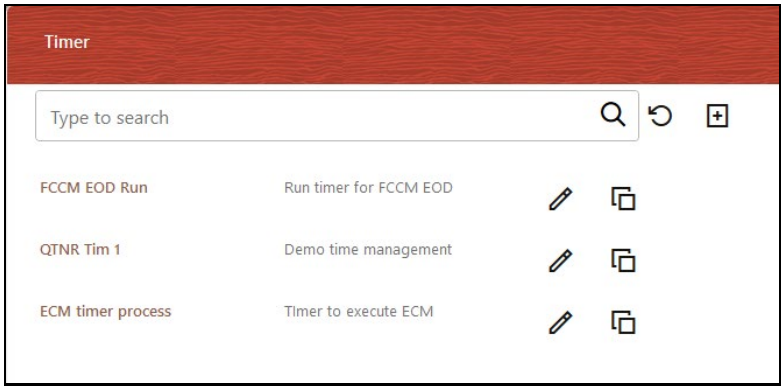
Ensure to meet the following prerequisites

1. Click a Process Flow from the list in the **Process Modeller** window to display it in the PMF Canvas.
2. Click the **Timer** icon from the header of the PMF Canvas to display the **Timer List** drawer window.

Figure : The Timer in the PMF Canvas Header

3. Click the following icons in the **Timer List** window for specific features:
   - **Add** : To add a Timer.
   - **Edit** : To edit a Timer.
   - **Copy** : To copy a Timer and create another with similar details.

Figure : The Timer List Window

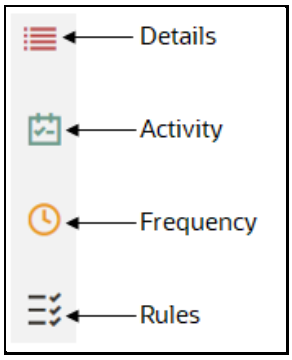


Click the icons in the **Timer List** window to display configuration details in the **Timer Details** drawer window. The window displays the following tabs:

- **Details:** The window to configure timer identification details.

- **Activity:** The window to map activities that are in the process to the timer.
- **Frequency:** The frequency of execution of the timer.
- **Rules:** The list of all application rules mapped to the process flow.

Figure : Tabs in the Timer Details Window



17.2 Configure a Timer

You can configure a timer using one of two methods:

1. Using the command line utility
2. Using the Timer window on the GUI

17.2.1 Configure Timer using a Command Line Utility

A shell script file **wfTimerInit.sh** is available in the ficdb/bin folder. To execute the utility, navigate to $FIC_DB_HOME/bin and execute **wfTimerInit.sh** with parameters as highlighted in the following image.

The format is:

```
./<File_name> <action_type> <Application_ID>
```

Action Type – Allowed values are 1 to 6.
For example, if the APPID is ECM, the possible action types are:

- 1 = ECM sets Active Flg to 'Y' for the input APPID
- 2 = ECM sets Active Flg to 'N' for the input APPID
- 3 = ECM sets V_EXN_FREQ_TYPE to 'MINUTE' for the APPID
- 4 = ECM sets V_EXN_FREQ_TYPE to 'HOUR' for the APPID
- 5 = ECM 10 increases or decreases the N_EXN_QUEUE_SIZE for the Input APPID
- 6 = ECM 10 increases or decreases the frequency value for the Input APPID

- Application Id - ECM, KYC, FSDF etc., which is defined in PMF by applications

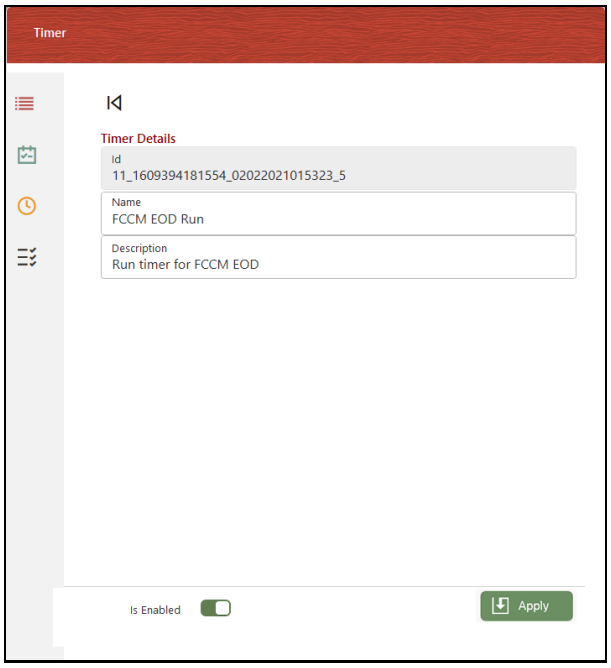
17.2.2 Configuring a Timer using the GUI

To configure a Timer, follow these steps:

1. Click a Process Flow in the **Process Modeller** window to display it in the PMF Canvas.
2. Click the **Timer** icon from the toolbar to display the **Timer** drawer window.
3. Click the **Add** icon to configure in the **Timer Details** window.

   To configure Timer details in the Details tab, follow these steps:

Figure : The Timer Details Tab



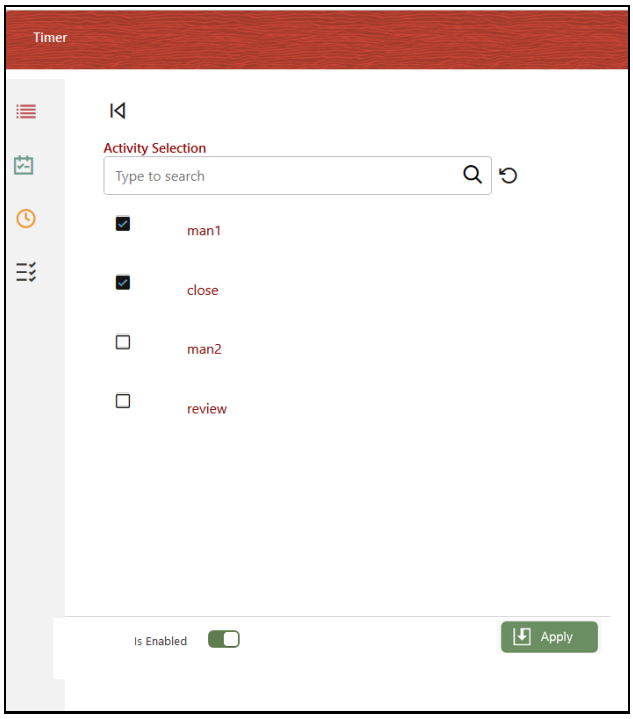
   a. Enter a name for the Timer in the **Name** field. The application auto-populates the **Id** field with a unique identifier and is read-only.

   b. Enter a description for the Timer in the **Description** field.

      You have to map the Timer to an activity on the canvas from the Activity tab.

      To configure details in the Activity tab, follow these steps:

Figure : The Activity Tab



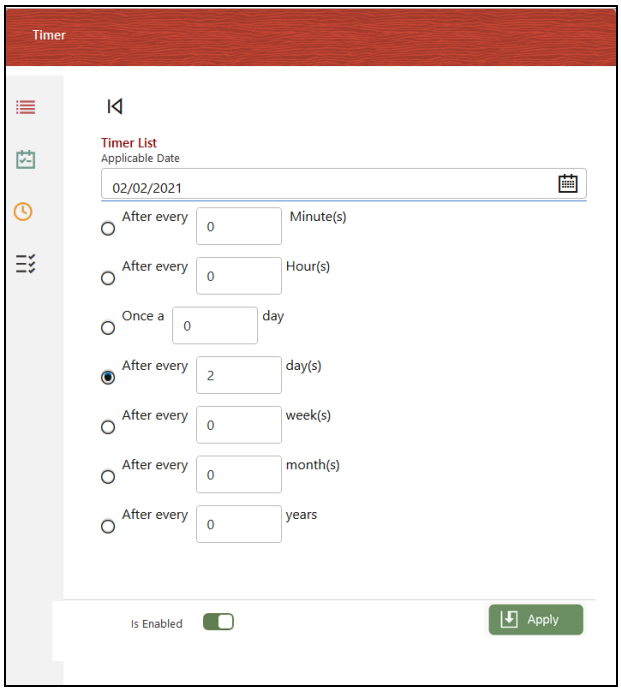
a. Click the **Activity** tab.

b. Select the required activities from the list. These activities are the activities that exist in the Process Flow Canvas.

After mapping an Activity to the Timer, you must set the frequency of execution of the rule on the activity from the Frequency tab.

To configure details in the Frequency tab, follow these steps:

Figure : The Frequency Tab



a. Click the **Frequency** tab to set the frequency of execution of the timer.

b. Select the date from which the timer is required to start executing in the **Applicable Date** date-picker.

c. Select the required frequency from the following and enter a value:

- After every Minute(s)

  Enter the number of minutes after which to execute the Timer. The timer is executed periodically after the entered number of minute(s) elapses.

- After every Hour(s)

  Enter the number of hours after which to execute the timer. The timer is executed periodically after the entered number of hour(s) elapses.

- Once a day

  Select the Once a day option if you want the timer to run only once.

- After every days(s)

  Enter the number of days after which to execute the timer. The timer is executed periodically after the entered number of day(s) elapses.

- After every weeks(s)

  Enter the number of weeks after which to execute the timer. The timer is executed periodically after the entered number of week(s) elapses.

- After every month(s)

Enter the number of months after which to execute the timer. The timer is executed periodically after the entered number of month(s) elapses.

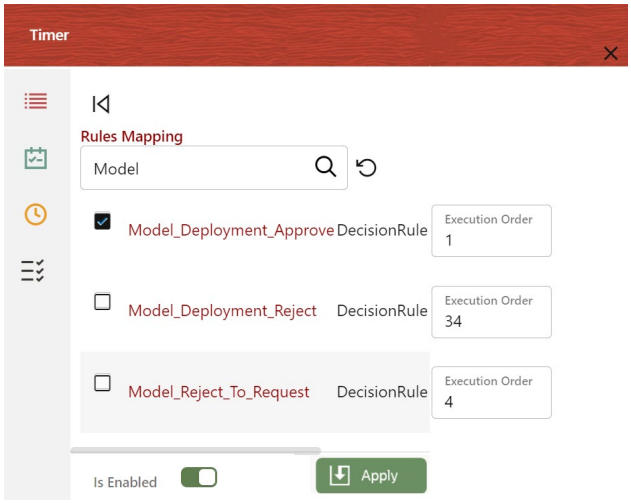
— After every year(s)

Enter the number of years after which to execute the timer. The timer is executed periodically after the entered number of year(s) elapses.

After you set the frequency, you must map the activity to execute the selected rule(s). The Timer executes the selected rules on the activity for the set frequency.

To map rules to the Timer from the Rules tab, follow these steps:

Figure : The Rules Tab



a. Click the **Rules** tab.

b. Select the rules required from the list.

> **NOTE** You can select only:
>
> 1. Decision and Execution Rules from the Rules Mapping list.
> 2. One Decision Rule from the Rules Mapping list to avoid conflict in the Decision Rule execution.
> 3. After the execution of a decision rule, the Timer stops retriggering or executing.

4. Set the **Execution Order** to execute the rules in sequence.

   The execution order by default is set to 1. You can set it to any number from 1 – 999. The rules are sorted in ascending order and executed sequentially.

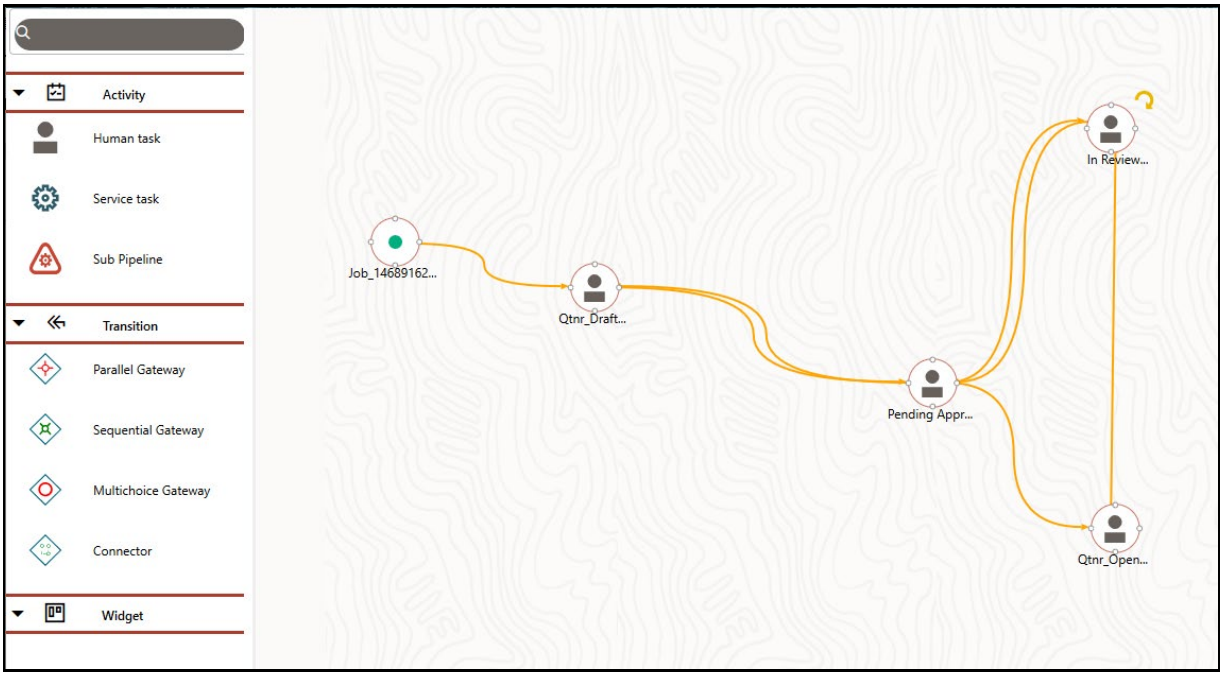
5. Click the **Apply** 

icon to save the details. Select the **Is Enabled**  toggle button to enable the Timer. If you keep it in the Deselect state, the Timer will not be enabled. However, you can enable it later when required.

17.3 Use Case: Timer Execution on a Questionnaire Workflow

The application and functioning of the Timer feature can be better understood by showing it to you in a use case. In this scenario, suppose that the Questionnaire in PMF is orchestrated with a workflow as shown in the following illustration:

Figure : Example of a Questionnaire Workflow Orchestrated on the Canvas

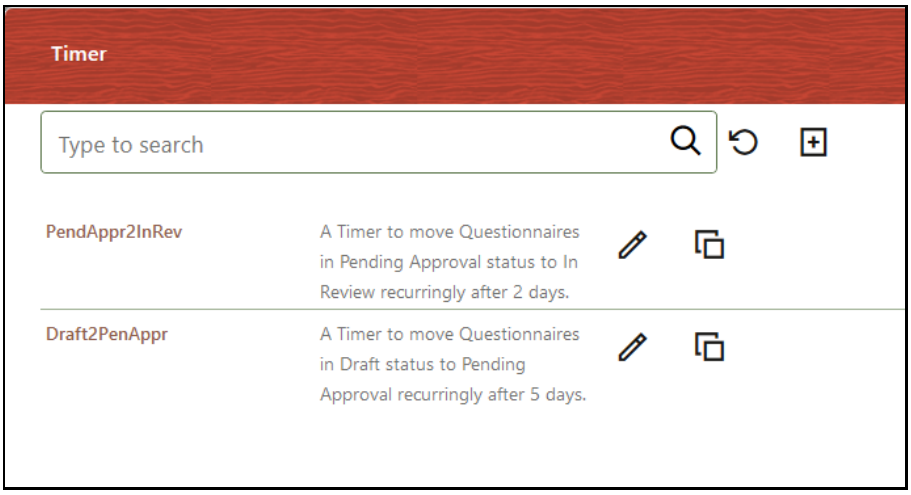


On the canvas, you see a process flow of activities, which are manual. In other words, they depend on human intervention for the execution of actions to move to the next activity. Use Timers to execute certain application rules recurringly on the activities on the canvas after a while lapse if the manual activity has not been executed.

Let us configure a Timer system on the preceding Questionnaire Process Flow.

> **NOTE** For instructions on how to configure the Timer details in the PMF UI, see the Configure a Timer section.

Figure : List of Timers Created for the Questionnaire



As shown in the previous illustration, there are two timers created: one for the Draft activity on the canvas and the other for the Pending Approval activity. We will go into the details in the following on how to configure these Timers to execute these activities.

Timer to execute the movement of questionnaires in the Draft status to the Pending Approval status every five days.

a. Details: Entered name and description for the Timer.

b. Activity: Selected the Questionnaire Draft activity from the list of activities. So, the timer is mapped to this activity.

c. Frequency: Selected After Every _ Days and entered 5 in the field after selecting the applicable date. The frequency of the execution of the timer is set to execute application rules on the activity every five days.

d. Rules: Selected the Draft to Pending Approval Decision Rule. The activity is mapped to execute the rule.

The result is that the Timer will execute the application rule and might move a questionnaire that has been in the Draft status for five days to the Pending Approval status if users do not move it manually.

Similarly, the Timer to execute the movement of a questionnaire in the Pending Approval status to the In Review status every two days is configured.

The result is that the Timer will execute the application rule and might move a questionnaire that has been in the Pending Approval status for two days to the In Review status if users do not move it manually.

18 Java Message Service (JMS)

The Java Message Service (JMS) is a messaging standard that allows application components based on the Java Platform Enterprise Edition (Java EE) to receive and send messages.

Topics:

18.1 Prerequisites

Ensure to meet the following prerequisites

1. Set the **ENABLE_JMS** parameter to **'Y'**, in the CONFIGURATION table in the CONFIG Schema, to enable JMS Capability.
2. Map the **'JMSWRITE'** Role to OFSAA Logged in User.
3. Restart OFSAAI Servers.

18.2 Configuring JMS for Message Queues

After configuring JMS messaging standard, Messages type application rule is enabled in PMF.

To create a Messaging Application rule,

1. Select the Application Rule Type as **Messaging**.
2. Select any of the following queues for Messages type application rule.
   - Active MQ
   - IBM MQ – File based, and channel based MQ
   - WebLogic MQ

18.2.1 Parameters for Active MQ

The configuration parameters for Active MQ are

- **URL**: URL to access active MQ console
- **Connection Factory**: The connection factory associated with the queue. By default, this is set to '**ConnectionFactory**'.
- **Connection User (optional)** : The user name to access the queue. This is required only if the queue requires authentication.
- **Connection Key (optional)**: The password to access the queue. This is required only if the queue requires authentication.
- **Queue or Topic Name**: The name of the queue.
- **Action**: Set the Action to Send or Receive.

- **Output Datafield** : Select the required Output Datafield.
- **Scope** : Set the Scope to Process or Package.

18.2.2 Parameters for IBM MQ – File Based

The configuration parameters for IBM MQ file-based messages queue are

- **Connection Type**: Select the connection type from the following:
  - File
  - Channel
  - LDAP
  - TCP
- **Connection Factory**: The connection factory associated with the queue. By default, this is set to '**ConnectionFactory**'.
- **Connection User (optional)** : The user name to access the queue. This is required only if the queue requires authentication.
- **Connection Key (optional)**: The password to access the queue. This is required only if the queue requires authentication.
- **Queue or Topic Name**: The name of the queue.
- **Action**: Set the Action to Send or Receive.
- **Output Datafield** : Select the required Output Datafield.
- **Scope** : Set the Scope to Process or Package.

18.2.3 Parameters for IBM MQ – Channel Based

The configuration parameters for IBM MQ file-based messages queue are,

- **Channel Name**: The name of the channel linked to the IBM Channel based MQ. Console.
- **Host Name: Port**: The host name and port details of the Queue console.
- **Queue Name**: The name of the queue.
- **Username (optional)**: The username to access the queue. This is required only if the queue requires authentication.
- **Password**: The password to access the queue. This is required only of the queue requires authentication.

18.2.4 Parameters for Weblogic MQ

The configuration parameters for WebLogic MQ are

- **URL**: URL to access WebLogic MQ console.
- **Connection Factory**: The connection factory associated with the queue. By default, this is set to '**jms/ConnectionFactory**'.

- **Queue Name**: The name of the queue
- **Username (optional)**: The username to access the queue. This is required only if the queue requires authentication.
- **Password**: The password to access the queue. This is required only of the queue requires authentication.

19 Appendix A

Topics:

19.1 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.

19.1.1 Configuring Parallel Group Approval

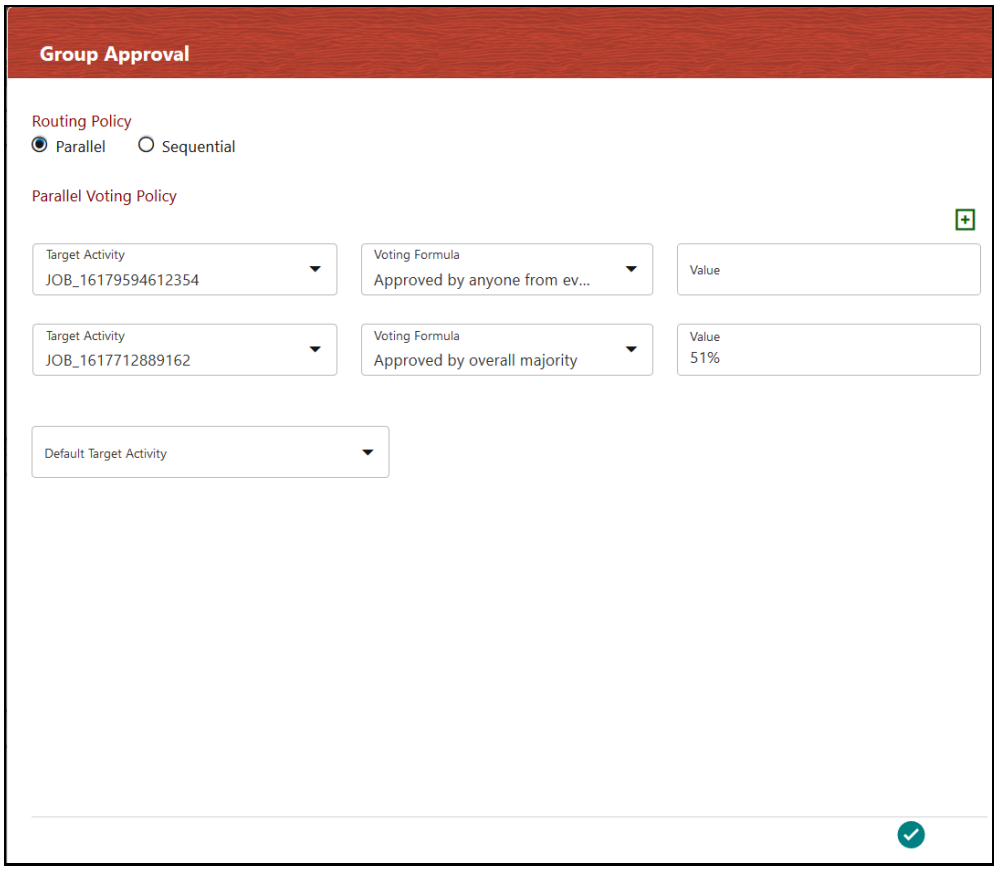
Parallel group approval is used when you want to send the task to all users in the task group simultaneously.

To configure parallel group approval:

1. Log in to the OFSAA Application.
2. Click **Administration**  from the **Header** to display the administration tasks in the **Tiles** menu.
3. Select the required **Information Domain** from the drop-down list.
4. Select **Process Modelling Framework** to display a submenu.
1. Select **Process Modeller** from the submenu to display the **Process Modeller** window.
2. From the **Process Modeller** window, select the required Process and click to open.

   The **Process Flow** tab is displayed.
3. Double-click the Activity for which you want to configure user approval.
4. On the window, click the  icon to display the **Actions** window.
5. Click **Group Approval** to configure group approval to display the **Group Approval Details** window.

Figure : Group Approval Details Window to Configure Parallel Group Approval



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

> **NOTE** It is recommended that you define the voting formula for all activities. If a 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 **Target Activity** from the drop-down list.
9. 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 the 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 the 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 favor the Target Activity to move the flow to that activity.
- **Approved by everyone**- All the users in all the task groups 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.

10. Select the **Default Target Activity** from the drop-down list. This is the activity that is executed if none of the conditions is satisfied.
11. Click the **Accept** icon to save it

19.1.2 Configuring Sequential Group Approval

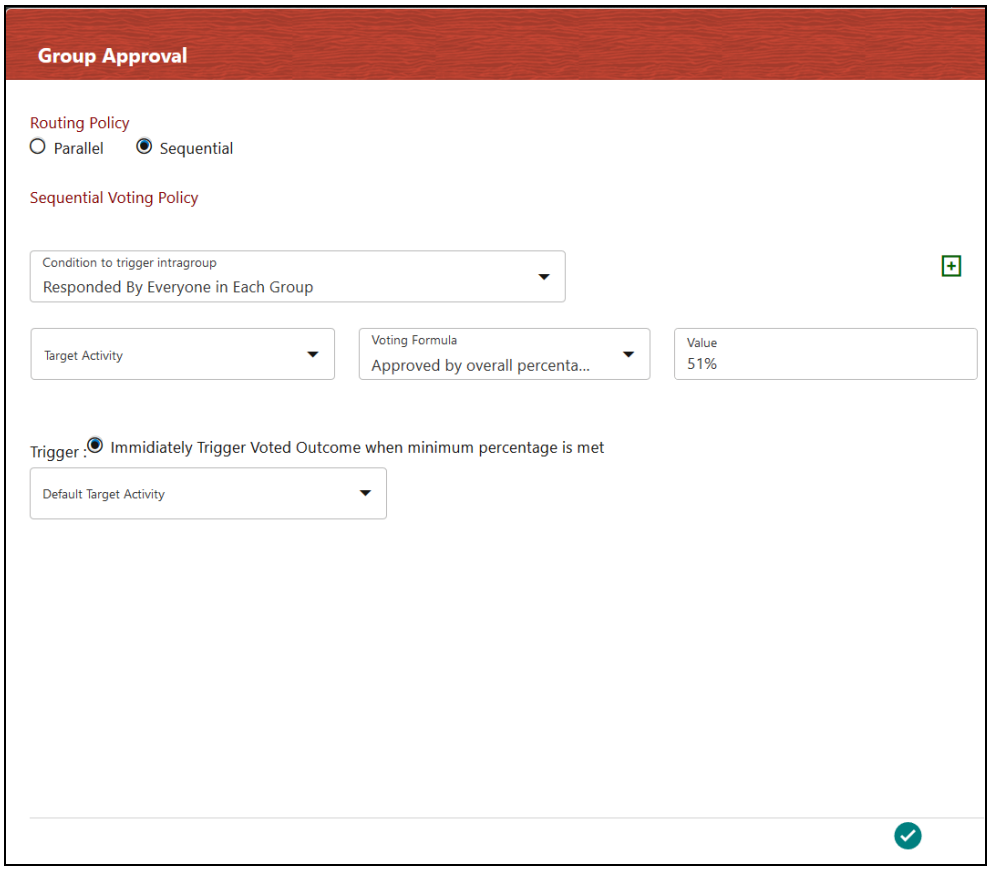
Sequential Group approval is used when you have multiple tasks for an activity.

To configure sequential group approval:

1. Log in to the OFSAA Application.
2. Click **Administration** from the **Header** to display the administration tasks in the **Tiles** menu.
3. Select the required **Information Domain** from the drop-down list.
4. Select **Process Modelling Framework** to display a submenu.
5. Select **Process Modeller** from the submenu to display the **Process Modeller** window.
6. From the **Process Modeller** window, select the required Process and click to open.

   The **Process Flow** tab is displayed.
7. Double-click the Activity for which you want to configure user approval.
8. On the window, click the icon to display the **Actions** window.
9. Click **Group Approval** to configure group approval to display the **Group Approval Details** window.

Figure : Group Approval Details Window to Configure Sequential Group Approval



10. Select **Sequential** to configure sequential group approval.
11. 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** window 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 waits till someone from that task group responds and so on.
    - **Responded by Overall Majority-** If the majority of users in the task group of the first task respond, it goes to the next task group and waits till the 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 responds and so on.
12. Click **Add**. A row is added to define the voting formula for target activity.

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

13. 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.
14. 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 the 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.
15. Select the activity which needs to be executed if the condition fails, from the **Default Target Activity** drop-down list.

19.2 JsonPath Expressions

JsonPath expressions always refer to a JSON structure in the same way as XPath expression is 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']
```

19.2.1 Operators

Table : Operator Description table

Operator	Description
`$`	The root element to query. This starts all path expressions.
`@`	The current node being processed by a filter predicate.
`*`	Wildcard. Available anywhere a name or numeric are required.
`..`	Deep scan. Available anywhere a name is required.
`.<name>`	Dot-notated child

Operator	Description
['<name>' (, '<name>')]	Bracket-notated child or children
[<number> (, <number>)]	Array index or indexes
[start:end]	Array slice operator
[?(<expression>)]	Filter expression. The expression must evaluate to a boolean value.

19.2.2 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.

Table : Function Description table

Function	Description	Output
min()	Provides the min value of an array of numbers	Double
max()	Provides the max value of an array of numbers	Double
avg()	Provides the average value of an array of numbers	Double
stddev()	Provides the standard deviation value of an array of numbers	Double
length()	Provides the length of an array Integer	Integer

19.2.3 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 : Filter Operators table

Operator	Description
==	Left is equal to the right (note that 1 is not equal to '1')
!=	Left is not equal to the right
<	Left is less than right
<=	Left is less or equal to the right

Operator	Description
>	Left is greater than right
>=	Left is greater than or equal to the right
=~	Left matches regular expression [?(@.name =~ /foo.*?/i)]
in	Left exists in right [?(@.size in ['S', 'M'])]
nin	Left does not exist in right
subsetof	Left is a subset of right [?(@.sizes subsetof ['S', 'M', 'L'])]
size	Size of the left (array or string) should match right
empty	Left (array or string) should be empty

19.2.4 Path Examples

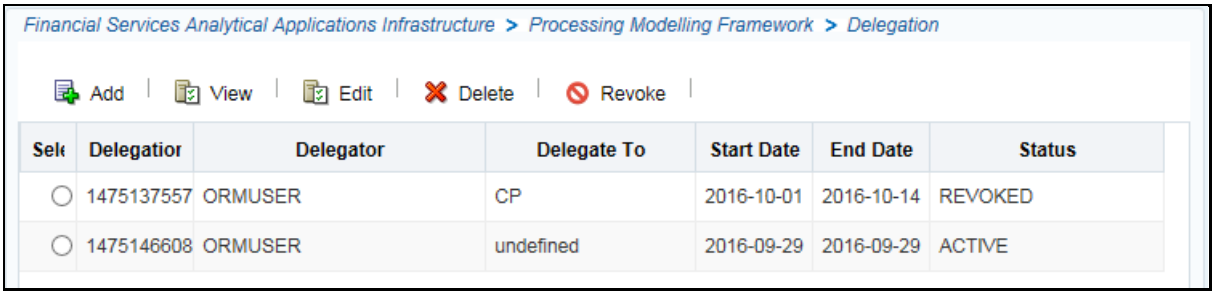
```
{
    "store": {
        "book": [
            {
                "category": "reference",
                "author": "Nigel Rees",
                "title": "Sayings of the Century",
                "price": 8.95
            },
            {
                "category": "fiction",
                "author": "Evelyn Waugh",
                "title": "Sword of Honour",
                "price": 12.99
            },
            {
                "category": "fiction",
                "author": "Herman Melville",
                "title": "Moby Dick",
                "isbn": "0-553-21311-3",
                "price": 8.99
            },
            {
```

```
                "category": "fiction",
                "author": "J. R. R. Tolkien",
                "title": "The Lord of the Rings",
                "isbn": "0-395-19395-8",
                "price": 22.99
            }
        ],
        "bicycle": {
            "color": "red",
            "price": 19.95
        }
    },
    "expensive": 10
}
```


19.3 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.

Figure : Delegations defined by the Logged-in user



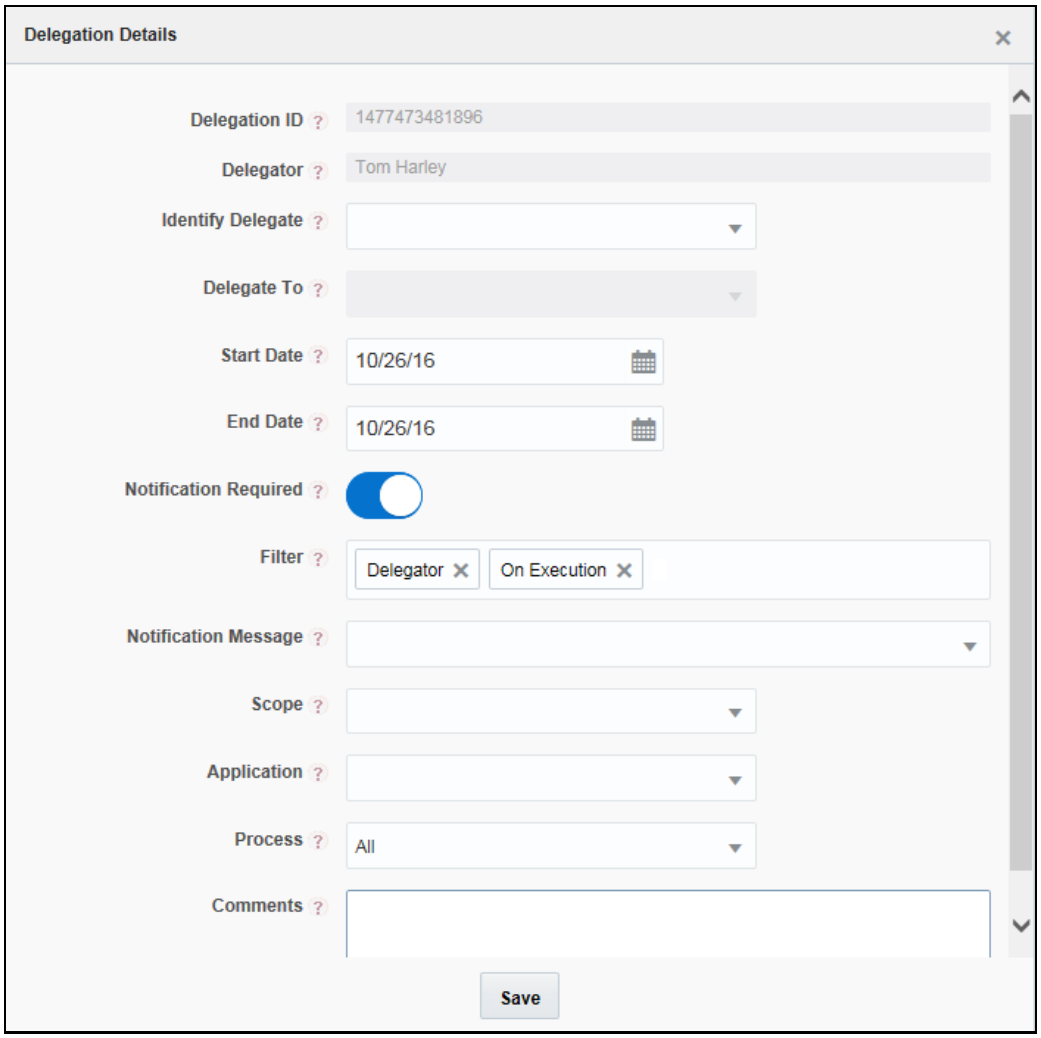
This window displays all the delegations that are defined by the logged-in user with details such as Delegation ID, Delegator, Delegate To, Start Date, End Date, and Status. You can add a new Delegation, view, modify, delete and revoke a delegation.

19.3.1 Adding a Delegate

To add a delegate

1. From the **Delegation** window, click **Add**. The **Delegation Details** window is displayed.

Figure : Delegation Details Window



2. Enter the details as tabulated:

Table : Delegation Details Description

Field Name	Description
Delegation ID	Displays the auto-generated Delegation ID.
Delegator	Displays the User ID of the logged-in user. If your user group is mapped to the function role WFDELGADM, you can select the delegator from the drop-down list.

Field Name	Description
Identify Delegate	Select the required option from the drop-down list. The options are: • Peers – Select this option if you want to delegate your tasks to your peer, who reports to your manager. • Subordinates- Select this option if you want to delegate your tasks to your immediate subordinates. • All subordinates – Select this option if you want to delegate your tasks to someone who comes under you in your organization.
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 the 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 a 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 are 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 are delegated to the selected user.
Comments	Enter if you want to add any comments for the delegation.

3. Click **Save**.

19.3.2 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.

19.3.3 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.

19.3.4 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.

19.3.5 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.

19.4 JSON Definition for Events

The JSON definition created by Producers consists of metadata fields and payload. The metadata field contains data to identify the event type used by the Event Framework and the payload consists of data that is used by the Consumer.

For more information about Producers, see the Producer Activity section.

The following is an example for the format of the JSON definition:

```
{
   "EVENT_TYPE": "VALUE" ,
```

```
   "EVENT_SUB_TYPE": " VALUE " ,
   "EVENT_SOURCE": " VALUE " ,
   "EVENT_KEY":"VALUE",
   "EVENT_REF_ID":" VALUE"
   "CORRELATION_ID": " VALUE ",
   "EVENT_TIMESTAMP":" VALUE " ,
   "PAYLOAD": {
   //business specific JSON definition  of the data being passed
               }
}
```

The description for the JSON parameters in the preceding example are as follows:

- **EVENT_TYPE:** The type of event at the parent level. For example, CASE.
- **EVENT SUB TYPE:** The subtype of the event. For example, CLOSURE, CREATE, UPDATE.
- **EVENT SOURCE:** The source application where the event occurs. For example, ECM

  The Type, Sub Type, and Source determine the action to be executed by the consuming service.
- **CORRELATIONID:** The identifier to use in case of a request or response model. It is a unique value and in alphanumeric format. For example, A45211B11. Service A sends an event to B and in return expects event B to send a response that maps the response to the original request.
- **EVENT_KEY:** The key used by the Producer for the event. For example, 100.
- **EVENT_REF_ID:** The unique ID created by the sender for the event that will be used to handle duplicate messages. For example, 100.
- **EVENT_TIMESTAMP:** The date and time of the creation of the event. Check the latency or determine the date on which the event is applicable. For example, 02/029/2021 12:49.
- **PAYLOAD:** The business data required by the consumer service in the JSON format. The event framework not only parses the payload but also expects the consuming service to interpret and process the message.

  An example for the payload is shown in the following:

```
{
    "payload": {
        "infodom": "INFODOM",
        "applicationparams": {
            "sourceApplicationId": "APP-123456",
            "onboardSystemCaseId": "0000",
            "onboardSystemCaseStatus": "",
            "gkycId": "xxx123yy123-abc123de456",
            "gkycCaseId": "CASE12345",
            "event": "Response Returned",
```

```
            "ACTION": "CASE90000",
            "comments": [
                "comment:RFI_response sent in document",
                "comment:RFI_document attached"
            ],
            "createdDate": "mm-dd-yyyy hh-mm-ss UTC"
        },
        "userid": "EXAMPLE_USER",
        "objecttype": "DA_DCA",
        "locale": "en_US",
        "objectid": "OBJID0000",
        "securitymap": {}
    }
}
```

20 Appendix B: Support APIs for Java External APIs

Topics:

20.1 Connection API

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

20.1.1 Jar Files Required

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

`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

20.1.2 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)
```

20.1.3 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();
        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;
    }
```

20.1.4 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;
            }
    }
catch (Exception e)
{
WorkflowUtil.logDebug("Error while updating process execution status...+ e);
}
finally {
ConnectionAdapter.closeResultSet(rs);
   ConnectionAdapter.closePreparedStatement(prepStatement);
   ConnectionAdapter.closeConnection(configConn);
    }
  return false;
    }
```

20.2 Logging API

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

20.2.1 Jar Files Required

The following jar available at `$FIC_HOME/ficweb/webroot/WEB-INF/lib` folder contains the WorkflowUtil class that contains logging-related APIs.

`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

20.2.2 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);
```

20.2.3 Error Message

An error message can be used to log an exception.

Signature:

```
public static void logError(Exception e)
```

For example:

```
WorkflowUtil.logError(e);
```

21 Appendix C: Set Up Event Framework Metadata

To configure additional details, you must set up the Event Framework Metadata. The sections in this topic provide detailed information for the configuration.

Topics:

21.1 Prerequisites

The Kafka Server must be installed and running.

21.2 Set Up Topics

To store and publish data from a Producer writing to a Kafka Topic for Consumers to read the data in the Topic in the OFSAA System certain configurations are required for Topic Metadata details and creation of the Topic in the Kafka Server, which is available in the following sections.

21.2.1 Create a Topic in the Kafka Server

The instruction in this section is an example for reference on how to create Topics in the Kafka Server.

To create a Topic in the Kafka Server, follow these steps:

a. Open the `kafka-topics.sh` file in the `$FIC_HOME/ficdb/bin` directory.

b. Execute the `kafka-topics.sh` file in the format shown in the following:

```
./kafka-topics.sh --create --zookeeper <host of zookeeper:<port in zookeeper.properties> --replication-factor 1 --partitions <more than given while creating topic in PMF> --topic <topic_name>
```

For example,

```
./kafka-topics.sh --create --zookeeper 192.0.2.1:7780 --replication-factor 1 --partitions 4 --topic ECM_TF_SWIFT
```

21.2.2 Add Topic Metadata Details

To add the details for Topic Metadata, configure the arguments and Run the script in the following format which adds data to the relevant table in the AAI Config Schema:

```
./saveTopic.sh <TOPIC_NAME> <BOOTSTRAP_SERVERS> <TOTAL_PARTITIONS>
```

Table : Argument Description Table for Topic Metadata

Argument	Description
TOPIC_NAME	The name of the Topic that you want to add. Topics is a virtual group(s) that stores or publishes data. It can have multiple consumers subscribing for the information.
BOOTSTRAP_SERVERS	The details of the server on which Kafka is installed. The Topic listens to this Server.
TOTAL_PARTITIONS	The total number of partitions for the Topic.

For example:

```
./saveTopic.sh ECM_TF_SWIFT 192.0.2.1:7780 4
```

21.2.3 Activate Consumer Group

The script to activate a Consumer Group is used to activate deactivated Consumer Groups. The Consumers are initialized and instantiated at Server startup. Only Consumer Groups that are active are instantiated at Server startup. The activation sets the Active Flag to 'Y' for a given Consumer.

The Consumer ID and Topic ID have to be passed to the Shell Script which requires activation. Configure the arguments and run the following script in the Event tables in the AAI Config Schema:

```
./activateconsumergroup.sh <N_CONSUMER_ID> <N_TOPIC_ID>
```

Table : Argument Description Table for Activate Consumer Group

Argument	Description
N_CONSUMER_ID	The Unique Identifier of the Consumer.
N_TOPIC_ID	The Unique Identifier of the Topic.

For example:

```
./activateconsumergroup.sh 1612 SWIFT_TF_01
```

21.2.4 Deactivate Consumer Group

The script to deactivate a Consumer Group is used to deactivate a Consumer Group that is active. The Consumers are initialized and instantiated at Server startup. Only Consumer Groups that are active are instantiated at Server startup. The activation sets the Active Flag to 'N' for a given Consumer. The Consumer Groups that are deactivated are not be instantiated.

The Consumer ID and Topic ID have to be passed to the Shell Script which requires deactivation.

Configure the arguments and Run the following script in the Event tables in the AAI Config Schema:

```
./deactivateconsumergroup.sh <N_CONSUMER_ID> <N_TOPIC_ID>
```

Table : Argument Description Table for Deactivate Consumer Group

Argument	Description
N_CONSUMER_ID	The Unique Identifier of the Consumer.
N_TOPIC_ID	The Unique Identifier of the Topic.

For example:

```
./deactivateconsumergroup.sh 1612 SWIFT_TF_01
```

21.2.5 Decrease Consumers

This script is used to decrease the number of running Consumers for a given Consumer Group Name, Topic, and Application Node. It applies only till the current application instance is running.

Configure the arguments and run the following script in the Event tables in the AAI Config Schema:

```
./consumerdecrease.sh <TOPIC_NAME> <CONSUMERGRPNAME> <DECREASEBY>
<HOSTNAME:PORT>
```

Table : Argument Description Table for Decrease Consumers

Argument	Description
TOPIC_NAME	The name of the Topic.
CONSUMERGRPNAME	The name of the Consumer's Group.
DECREASEBY	The number of Consumers that you want to decrease.
HOSTNAME:PORT	The details of the server on which Kafka is installed.

For example:

```
./consumerdecrease.sh ECM_TF_SWIFT ECM Users 15 192.0.2.1:7780
```

21.2.6 Increase Consumers

The script is used to increase the number of running Consumers for a given Consumer Group Name, Topic, and Application Node. It applies only till the current application instance is running.

Configure the arguments and run the following script in the Event tables in the AAI Config Schema:

```
./consumerincrease.sh <TOPIC_NAME> <CONSUMERGRPNAME> <INCREASEBY>
<HOSTNAME:PORT>
```

Table : Argument Description Table for Increase Consumers

Argument	Description
TOPIC_NAME	The name of the Topic.
CONSUMERGRPNAME	The name of the Consumer's Group.
INCREASEBY	The number of Consumers that you want to increase.
HOSTNAME:PORT	The details of the server on which Kafka is installed.

For example:

```
./consumerincrease.sh ECM_TF_SWIFT ECM Users 15 192.0.2.1:7780
```

21.2.7 Replay Messages

This script replays messages persisted to the Message Audit Tables to reprocess messages for failure cases or other reasons. The message is picked from the audit tables and processed instead of the Kafka Topic. The Event ID, which is available as a key in the Message Audit Tables, is used to replay the message. Any message that is read from the Kafka Topic is persisted to Message Audit Table with a Unique Event ID and processed later.

Configure the arguments and run the following script in the Event tables in the AAI Config Schema:

```
./replaymessage.sh <EVENT_ID>
```

Table : Argument Description Table for Replay Messages

Argument	Description
EVENT_ID	The unique identifier of the Event.

For example:

```
./replaymessage.sh JOB1234
```

21.2.8 Save Consumer Properties

This script adds or updates Consumer Properties for a given Consumer ID (Consumer Group). The Properties are used to fine-tune the functioning of the Kafka Consumers.

The Shell Script accepts Consumer ID, Kafka Consumer Property Name, Value, and Datatype.

The properties have to be as given by the Kafka Consumer API. See the following link for a list of properties:

https://docs.confluent.io/platform/current/installation/configuration/consumer-configs.html

Configure the arguments and Run the following script in the Event tables in the AAI Config Schema:

```
./saveConsumerProperties.sh <CONSUMER_ID> <CONSUMER_PROP_NAME>
<CONSUMER_PROP_VALUE> <CONSUMER_PROP_DATATYPE>
```

Table : Argument Description Table for Save Consumer Properties

Argument	Description
CONSUMER_ID	The Unique Identifier of the Consumer.
CONSUMER_PROP_NAME	The name of the Consumer Property that is related to the selected Consumer ID.
CONSUMER_PROP_VALUE	The value of the Consumer Property.
CONSUMER_PROP_DATATYPE	The data type of the Consumer Property.

For example:

```
./saveConsumerProperties.sh 1612 ECM_WR WRITE CHAR
```

21.2.9 Save Producer Properties

This script adds or updates Consumer Properties for a given Producer ID. The properties are used to fine-tune the functioning of the Kafka Producer.

The Shell Script accepts Producer_ID, Kafka Producer Property Name, Value, and Datatype.

The properties have to as given by the Kafka Producer API. See the following link for a list of properties:

https://docs.confluent.io/platform/current/installation/configuration/producer-configs.html

Configure the arguments and run the following script in the Event tables in the AAI Config Schema:

```
./saveProducerProperties.sh <PRODUCER_ID> <PRODUCER_PROP_NAME>
<PRODUCER_PROP_VALUE> <PRODUCER_PROP_DATATYPE>
```

Table : Argument Description Table for Save Producer Properties

Argument	Description
PRODUCER_ID	The Unique Identifier of the Producer.
PRODUCER_PROP_NAME	The name of the Producer Property that is related to the selected Producer ID.
PRODUCER_PROP_VALUE	The value of the Producer Property.
PRODUCER_PROP_DATATYPE	The data type of the Producer Property.

For example:

```
./saveProducerProperties.sh PROD1234 ECM_SWIFT_WR  WRITE CHAR
```

21.2.10 Start Consumers

This Shell Script is used to start Consumers that are stopped for a given Application Node, Consumer Group Name, and Topic.

If you pass ALL in place of the consumergroupname argument, it results in the start of all Consumers in the given Application Node and Topic.

If the number of running Consumers is equal to the configured count in the Consumer Config Tables, no action is applied.

Configure the arguments and Run the following script in the Event tables in the AAI Config Schema:

```
./startconsumers.sh <TOPIC_NAME> <CONSUMERGRPNAME>/<ALL>
<HOSTNAME:PORT>
```

Table : Argument Description Table for Start Consumers

Argument	Description
TOPIC_NAME	The name of the Topic.
CONSUMERGRPNAME	The name of the Consumer's Group.
ALL	This value is passed to select all Consumers.
HOSTNAME:PORT	The details of the server on which Kafka is installed.

For example:

```
./startconsumers.sh ECM_TF_SWIFT ECM Users/ALL 192.0.2.1:7780
```

21.2.11 Stop Consumers

This shell script is used to stop consumers for a given Application Node, Consumer ID (Consumer Group Name), and Topic. After the servers are restarted, the consumers are initialized and instantiated.

If you pass ALL in place of the consumergroupname argument, it results in the stop of all Consumers in the given Application Node and Topic.

Configure the arguments and Run the following script in the Event tables in the AAI Config Schema:

```
./stopconsumers.sh <TOPIC_NAME> <CONSUMERGRPNAME>/<ALL> <HOSTNAME:PORT>
```

Table : Argument Description Table for Stop Consumers

Argument	Description
TOPIC_NAME	The name of the Topic.
CONSUMERGRPNAME	The name of the Consumer Group.
ALL	This value is passed to select all Consumers.

Argument	Description
HOSTNAME:PORT	The details of the server on which Kafka is installed.

For example:

```
./stopconsumers.sh ECM_TF_SWIFT ECM Users 15 192.0.2.1:7780
```

22 Appendix D: Archival Utility for PMF Tables

To archive PMF tables that contain large amount of data accumulated over a period of time, use the following archival utlities.

Topics:

22.1 Archive By Executing Shell Script

Complete the following steps to archive data from AAI_WF_PROCESS_ENGINE_ACT_QUE & AAI_WF_PROCESS_ENGINE_ACT_HIST table, by executing a Shell script.

1. Login to the Server where OFSAAI installed.
2. Go to Bin directory

   ```
   cd $FIC_DB_HOME/bin
   ```

3. Execute `wfEngineActDataArchive.sh` as shown below.

   Syntax

   ```
   nohup ./wfEngineActDataArchive.sh INPUT1 INPUT2 INPUT3 INPUT4 <Date –
   YYYYMMDD> "<No. of records to be left from the latest N-order>"<table
   to be archived,QUE/HIS>" &
   ```

   Example:

   ```
   nohup ./wfEngineActDataArchive.sh INPUT1 INPUT2 INPUT3 INPUT4 20200115
   "2,QUE" &
   ```

Note:

- The first 4 input entries (Input 1 to Input 4) are dummy inputs hence you can provide any value.
- 5th parameter (Date -YYYYMMDD) is Date in the format of YYYYMMDD
- 6th parameter (<table to be archived, QUE/HIS>) is to indicate which table has to be archived (QUE or HIST) & the number of records to be excluded from latest N_ORDER

22.2 Archive Using Batch UI

Complete the following steps to archive data from AAI_WF_PROCESS_ENGINE_ACT_QUE & AAI_WF_PROCESS_ENGINE_ACT_HIST table, using Batch UI.

1. Login to ECM application UI with admin user
2. Create a Data Transformation (DT) of type External Library. Select the `wfEngineActDataArchive.sh` file as part of DT definition.
3. Add 6 parameters

4. Create ICC batch using the above DT.

   In the Parameter List select the number of records to be excluded & table information as -- 2,QUE or 3,HIST

5. Execute the batch by selecting the date beyond which data needs to be archived.

Note:

- The first 4 input entries (Input 1 to Input 4) are dummy inputs hence you can provide any value.
- 5th parameter (Date -YYYYMMDD) is Date in the format of YYYYMMDD
- 6th parameter (<table to be archived, QUE/HIS>) is to indicate which table has to be archived (QUE or HIST) & the number of records to be excluded from latest N_ORDER