****

|  |
| --- |
| **Member Compliance Garage** |
| Solution Design Document |

****

**Table of contents**

[1.1 Constraint 1](#_Toc47619676)

[1.2 Assumption 1](#_Toc47619677)

[1.3 Risk 1](#_Toc47619678)

[1.4 Payment Services 1](#_Toc47619679)

[1.4.1 BillDesk Payment Services 1](#_Toc47619680)

[1.4.2 Testing in UAT 1](#_Toc47619681)

[1.4.3 Payment Services Flow(Future State) 1](#_Toc47619682)

[1.4.4 General Payment Services Process Flow 1](#_Toc47619683)

[1.5 Data Model 1](#_Toc47619684)

[1.6 Micro services 1](#_Toc47619685)

[1.6.1 Payment Service 1](#_Toc47619686)

[1.7 Sequence Diagrams for Key Business Process Flows of the Journeys 1](#_Toc47619687)

[1.7.1 BillDesk Payment Sequence Flow 1](#_Toc47619688)

[1.7.2 Common Payment Flow 1](#_Toc47619689)

## Constraint

## Assumption

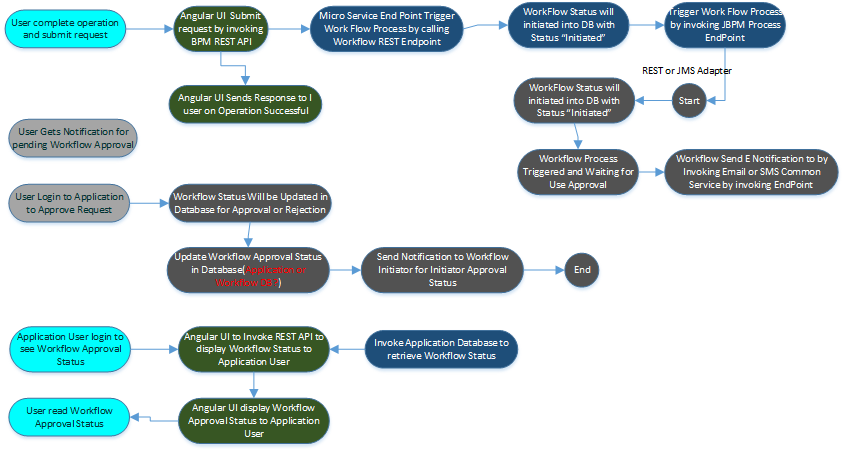


## Risk

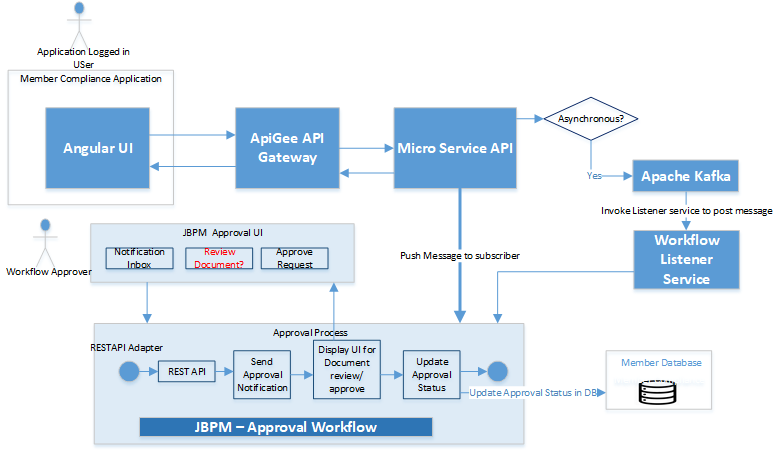


## BPM Services

### BPM Services – Activity



### BPM Service – Process Flow



**Action Items: -**

**Meeting Minutes**

**Questionnaire**



**Review Comments**

**Process Flow Steps:**



#### Invoke BPM Workflow through REST API

**Constraint:**

* There Is no request parameter accepted by this REST API hile triggering JBPM Process. Hence exchange of request data needs to be only through database.

| **Ame** | **Type** | **Description** |
| --- | --- | --- |
| deploymentId | java.lang.String | This is the name (id) of the deployment the RuntimeEngine should interact with. |
| baseUrl | java.net.URL | This is the URL of the deployed jbpm-console, kie-wb or BPMS instance.  For example: http://127.0.0.1:8080/jbpm-console/ |
| Username | java.lang.String | This is the user name needed to access the JMS queues. |
| Password | java.lang.String | This is the password needed to access the JMS queues. |

Invoke BPM Service through Feign Client

|  |
| --- |
|  |

#### Invoke BPM Workflow through JMS

BPM Process Flow to expose REST END Point for other Services to invoke this common Workflow Service through REST Endpoint and hence no JMS Integration required.

#### Installing and configuring Database with JBPM

#### Defining Users and Groups for Approval process

Connect with Application Authentication Database for fetching list of Users and Roles to be configured inworkflow

Run Scheduler process to sync up users, roles and group on frequent interval

Map these users and groups into the process for various approval/operations

#### Human Task in Workflow

Create Human Task and configure input and out parameters in the process flow.

|  |  |  |  |
| --- | --- | --- | --- |
| *S.No* | *Workflow Component* | *Workflow Attributes and value* | *Comments* |
|  | *Functional Description* |  |  |
| *2* | *Task Type* | *User Task* |  |
|  | *Script Code (Java)* |  |  |
|  | *Compensation and Loop Characteristics* |  |  |
|  | *Actors* |  |  |
|  | *On Entry Script* |  |  |
|  | *On Exit Script* |  |  |
|  | *Input Data Mapping* |  |  |
|  | *Output Data Mapping* |  |  |
|  |  |  |  |

#### Any reusable Sub Process – To be created

#### Decision and Business Rules

Use Rule Task to invoke or specify business Rules. . JBPM provides added support for Rules and also accept any dynamic rule parameter as input for the business rules.

#### Invoke REST API from Business Process

Use Service Task to invoke API.

#### Invoke Database through Workflow Task

Create Java Script Task and provide custom implementation to invoke Database.

Check if Data Object can be used for this purpose?

#### Send Email Notification through Process Flow Task

Create Web Servia Task and invoke Notification Service /Email endpoint with URL to send any notification through process flow

#### Security

**Transport Layer Security**

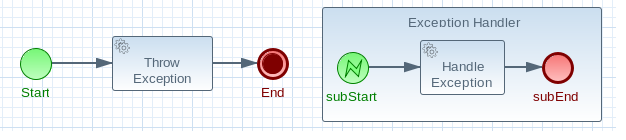
JBPM needs to be exposed to HTTPS with 443 and communication between Micro Services and JPBPM through HTTPS.

**Data Security**

No PCI Details are stored and hence no Encryption/Decryption needed for data.

#### Exception and Error Handling

**Exception handled in Process Flow**

****

**Exception handled in Code**

Exception thrown by Script Task

**Exception Service and Configuration**

**Business Exception**

| **nt types** | **Description** |
| --- | --- |
| Errors | Error Events can be used to signal when a process has encountered an unexpected situation: signalling an error is often called *throwing* an error.  Boundary Error Events in a different part of the process can then be used to *catch* the error and initiate a sequence of activities to handle the exception.  Errors themselves can be extended with extra information that is passed from the throwing to catching event. This is done with the use of an Item Definition. |
| Compensation | Exception handling activities *associated* with the normal activies in a Business Transaction are triggered by *Compensation Events*.  There are 3 types of compensation events: Intermediate (a.k.a. Boundary) (catch) events, Start (catch) events, and Intermediate or End (throw) events.  Compensation Boundary (catch) events may only be attached to activites (e.g. tasks) that could cause an exception. These Boundary events are then *associated* (not linked!) with a Task that will be executed if the Boundary event catches a (thrown) Compensation signal.  Start (catch) events are used when defining an *Compensation Event SubProcess*, which requires them in order to be able to catch a (thrown) Compensation signal.  Compensation Intermediate and End events are used in order to throw Compensation Events. These events often follow decision nodes that determine whether the workflow executed up to that point has succeeded. If not, the path including the Intermediate or End Event is chosen in order to trigger Compensatoin for the activities that did not succeed. |

#### Logging

How logging is handled in JBPM

#### Reusability

Identify common process for reusability – Error and Exception handling or any other sub business process pro

Identify common Task for reusability like – Error and Exception handling, Logging

## **Data Model**

**Data model of Audit table**

JBPM maintain Workflow Metadata table structure and part of this workflow status also maintained.

**Do we need separate table ?**

Table Name: WORKFLOW\_STATUS

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Column Type** | **Description** |
| Id | number  default id\_WORKFlow.nextval | Table primary key  ID\_SEQ |
| **Trace\_Id** | varchar2(20) |  |
| **Span\_Id** | varchar2(20) |  |
| **Process\_Deploy\_URL** | varchar2(100) | URL where BPM specific process is deployed |
| **Process\_Deploy\_ID** | varchar2(20) | Get this ID when BPM Process is deployed |
| **Process\_Deploy\_NAME** | varchar2(200) | Name of the deployment for a specific process |
| **Status** | varchar2(10) | Values can be “Success”, “Failed”, pending |
| **Error\_Code** | varchar2(10) | Error Code to be logged in case of failure |
| **Error\_Message** | varchar2(150) | Error like “Unable to reach InfoBip Service” |
| **Comments** | varchar2(150) | Any other comments in Json format (String) |

## **Micro services**

### BPM Service

**Do we need separate service and end point for this?**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *S.No* | *Common User Story/Function* | *Service Name* | *Service Description* | *Resource Endpoint* | *API Spec* |
| *2* | *Get Workflow Status* | *BPM Service* | *This end point is to get Workflow Status based upon process/deployment ID by querying Workflow Database* | *//Status/{Process/DeployID}* | *Method: GET*  *Request PARAM:*  *{*  *}*  *Response PARAM:*  *{*  *To be Determined*  *}|* |
|  | *Update WorkFlow Status* | *BPM Service* | *This end point is to post workflow status into Workflow Datbase or Application Database* | /Status | Method: POST  *Request PARAM:*  *{*  *}*  *Response PARAM:*  *{*  *To be Determined*  *}|* |
| *4* |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

## **Sequence Diagrams for Key Business Process Flows of the Journeys**

### Common Workflow Service Sequence

