# Requirement Specification Document

|  |  |
| --- | --- |
| **Target release** | Assurance\_1.0 |
| **Epic** |  |
| **Document status** | IN PROGRESS |
| **Document owner** | [Naveen Kurakula](file:///C:\wiki\display\~nkurakula.bcj) |
| **Developers** | [Chandra Mouli](file:///C:\wiki\display\~carumalla.bcj), Srinivas K, Nikshitha N |

## Goals

The main objective of this project is to provide integration of common services like DMV, Quote, Policy and payment verification which are used by different applications in the same company. This allows code reusability. WSO2-ESB provides abstraction for actual service implementation.

## Background and strategic fit

Effective management of applications in companies has been a tough task. Through this project we propose to develop a solution for management of various applications through implementation of SOA architecture by WSO2-ESB.

## Assumptions

## Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Title** | **User Story** | **Importance** | **Notes** |
| 1 | **DMVAccessApp** | This application provides two services.   * As a consumer, when the request is made for the vehicle list, based on address, the list of vehicles is returned. * As a consumer, when the request is made for the driver history, based on the license number the driver details is returned. | High | * This application interacts with DMV application to get the vehicles list and driver history. |
| 2 | **PolicyGenerationApp** | This application provides various features which are exposed as web services. These services are listed below:   * As a consumer, once the request is made for quote, the quote will be calculated and is returned to the user. * As a consumer, once the update quote request is made, the updated quote will be returned. * As a consumer, once the request for saving the quote is made the quote is saved in the user account * As a consumer, once the request is made for payment, the card details will be processed. If the payment is successful, the policy is generated and returned to the user. * As a consumer, once the update policy request is made, the new policy will be returned. | High | * This application internally calls DMV, Claims and Payment Handler applications. |
| 3 | **VehicleClaimsInfoApp** | This application provides web service to retrieve the claim history of the vehicle based on the VIN number.   * As a consumer, when a request for the claim history of a vehicle is made, this service returns all the claims details of the vehicle based on the VIN number. | High | * This application interacts with Claims application to get the claims history. |
| 4 | **PaymentVerificationApp** | As a consumer, when the user makes the payment request, the card details are sent for verification and the transaction is processed. | High | * This application interacts with BOFA application to validate and process the payment. |

## Task List

**DMVAccessApp:**

* Create a web service “getVehiclelist” which takes mailing address object as parameter.
  + This service should provide a client to consume the web service ”” provided by the DMV application to get the list of vehicles.
  + The list received is again produced as the output of this web service.
* Create a web service “getDriverHistory” which takes driver license as input.
  + This service should provide a client to consume the web service “” provided by the DMV application to get the driver details.
  + The driver details obtained in the above step are produced as the output of this web service.

**PolicyGenerationApp:**

* Create a web service “getQuote” which takes the options entered by the user as input.
  + This service should provide a client to consume the web service “getDriverHistory” provided by the DMVAccessApp to get the driver history based on the driver license.
  + It should provide a client to consume the web service “” provided by the Claims application to get the claims history of the vehicle based on the VIN number.
  + It should have a method “calculateQuote” which uses the details fetched in the above steps and the drools rules engine to calculate the quote estimate.
  + The quote is produced as the output of this web service.
* Create a web service “updateQuote” which takes the options entered by the user as input at updateQuote request.
  + This service calls the “getQuote” service is called with the updated user details.
  + The quote received in the above step is returned to the user.
* Create a web service “saveQuote”.
  + This service should provide methods to save the quote in the user account.
  + Store the Quote details in the database.
* Create a web service “paymentService” which takes the card details entered by the user as input.
  + This service should provide a client to consume the web service “verifyPayment” provided by the PaymentVerifcationApp to verify the card details.
  + This service should have a method “processPayment” to process the transaction. This method returns the status of the payment.
  + This service should have the method “savePolicy” which stores the quote details and the payment details as a policy in the database.
  + The policy created in the above step is produced as the output of this web service.
* Create a web service “updatePolicy”
  + This service should have the method “updatePolicy” which stores the updated policy in the database and the user account.
  + This service returns the status of the update.

**VehicleClaimsInfoApp:**

* Create a web service “getClaimHistory”.
  + This service should provide a client to consume the web service “ ” provided by the Claims application to get the claim details of the vehicle based on VIN number.
  + The claim details obtained in the above step are produced as the output of this web service.

**PaymentVerificationApp:**

* Create a web service “verifyPayment”.
  + This service should provide a client to consume the web service “ ” provided by the BOFA application to verify the payment details entered by the users.
  + This web service returns a Boolean.