



Document Version / Details: Ver. 1.0/ 15-May-2019

**Solution Architecture Design**

**Revision History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Version No. | Modified By | Reviewed By | Authorized By | Release Date | Modifications Done |
| 1.0.0 | Tarun Kumar |  |  |  | 17-May-2019 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Table of Contents

[1 Introduction 5](#_Toc10474203)

[1.1 Purpose 5](#_Toc10474204)

[1.2 Audience 5](#_Toc10474205)

[1.3 Document Structure 5](#_Toc10474206)

[1.4 Scope 5](#_Toc10474207)

[2 Overview 6](#_Toc10474208)

[2.1 Organization 6](#_Toc10474210)

[2.2 Key Pain Areas 6](#_Toc10474211)

[2.3 Program Name 6](#_Toc10474212)

[2.4 Program Objectives/Problem Statement 6](#_Toc10474213)

[2.5 Critical Success Factors 7](#_Toc10474214)

[2.6 Solution Architecture Assumptions 7](#_Toc10474215)

[3 Project Scope 8](#_Toc10474216)

[3.1 In Scope 8](#_Toc10474218)

[3.2 Out of Scope 8](#_Toc10474219)

[4 LTI’s Approach 9](#_Toc10474220)

[5 Execution Strategies 10](#_Toc10474221)

[6 Integration Solution Blueprint 11](#_Toc10474222)

[6.1 Enterprise Integration Conceptual Architecture 11](#_Toc10474224)

[6.2 Enterprise Integration Solution Approach 1 14](#_Toc10474225)

[6.3 Enterprise Integration Solution Approach 2 16](#_Toc10474226)

[6.4 Proposed Technical Integration Architecture 16](#_Toc10474227)

[6.4.1. Integration using Request/Replay Model 16](#_Toc10474228)

[7 Pattern to be used 19](#_Toc10474229)

[8 Architecture Principles and Decisions 22](#_Toc10474230)

[8.1 Principles 22](#_Toc10474232)

[8.2 Architectural Decisions 23](#_Toc10474233)

[8.3 Outstanding Issues 23](#_Toc10474234)

[8.4 Architectural Risks 24](#_Toc10474235)

[8.5 Reference Architecture 25](#_Toc10474236)

[9 Proposed Solution Architecture 26](#_Toc10474237)

[9.1 Technical Architecture 26](#_Toc10474239)

[10 Physical (Deployment) Architecture 28](#_Toc10474240)

[10.1 Deployment Architecture 28](#_Toc10474242)

[10.2 Environment Requirement 28](#_Toc10474243)

[10.3 Integration View 28](#_Toc10474244)

[10.4 Information Model 29](#_Toc10474245)

[10.5 Processes and Services View 29](#_Toc10474246)

[10.6 Non-Functional Requirements & Solutions 29](#_Toc10474247)

[10.7 Migration 29](#_Toc10474248)

[10.8 Organizational Policies and Guidelines 30](#_Toc10474249)

[11 Appendix 31](#_Toc10474250)

[11.1 Glossary 31](#_Toc10474252)

# Introduction

## Purpose

This document is the result of the architecture design phase for **one of life Administration function(Generate quote)** of HSBC UK insurance application*.* In this, we have identified the possible problems at the architecture design phase of the project before much of the system is implemented.

Strategic decisions must be cross checked carefully with the experts (reviewers) because they are essential for the system which is being developed, especially if the system needs to be operational for very long-time scales.

## Audience

This Architecture design document is intended for stakeholders, Architects and the Subject Matter Experts who provide valuable inputs to help define and accept these requirements and to ensure that the requirements specified in this document are completely met in **target state of HSBC Insurance.**

This document is further intended for use by the **HSBC Insurance** development team for designing and implementing the specified application as well as estimating the effort required for its development and deployment.

**Pre-Requisites**

<Fill in as required>

## Document Structure

<Provide the details of reference documents and other links that are relevant>

## Scope

In this document the possible architecture for target state is described; this is the most appropriate and final proposed architecture for **target state of HSBC Life Insurance.**

# Overview



## Organization

HSBC is the global leader in banking, financial and insurance service organisations. It works in multiple regions like UK, SG, HK, China etc for insurance lifecycle. HSBC is one of the world’s largest banking and financial services organisations. HSBC offers a comprehensive range of banking and financial services including retail banking and wealth management; commercial, investment and private banking; insurance; forfaiting and trustee services; securities and capital markets services.

## Key Pain Areas

* Lack of flexibility and scalability due to a rigid architecture of policy life administration
* Difficulty in integrating with underlying modular, and flexible IT architectures for payment system and customers data.
* Piled up technology debt and lack of platform support of Mainframe
* Slow development lifecycle and difficulty in code maintainability
* Inability to refactor old code and code is not adaptable to changing new business requirements.
* Challenges to support existing policies as lack of business function like policy enquiry and claim enquiry.
* Complex request/response transformation for mainframe system

## Program Name

Generate quotation journey is initiated by the customer, Staff or third party in distribution layer. In this journey, Customer/Staff/Third-party sends personal information, employment details and insurance details on which the premium gets calculated at the downstream layer and this premium is presented to the customer. Premium gets re-calculated, if customer modifies the premium changing factors such as lifestyle, etc.

## Program Objectives/Problem Statement

* Identifying the current pain points
* Map current (AS IS) and target (To Be) state of the application/IT systems
* Analyzing the current application and code performance
* Paint target state architecture and modernization options
* Assess resource requirements, and impact analysis
* Suggest modernization options, and create a roadmap with priorities

## Solution Architecture Assumptions

* Changes within the applications in BSL and Lifepen is not the responsibility of Integration team
* Assuming Integration layer will have all required credentials to interact with upstream and downstream applications whenever needed.
* Upstream and downstream applications should be in sync with availability of systems
* HSBC will provide a single point of contact (SPOC) for timely resolution of any issues during the execution of this project.
* HSBC would ensure the timely availability of required stakeholders, software, Code repositories, storages and hardware, various environments along with necessary access before the start of the project.
* HSBC team will provide details on flow logic, validation, compliance, rules, orchestration logic, adaptor of the existing platform.
* Technical, functional and business details about existing legacy applications will be provided. HSBC Team should be available for query resolution and acceptance testing.

# Project Scope



## In Scope

* Provide design and appropriate solution for quotation generation based on the API-led connectivity.
* Understanding of Integration and decompose the mainframe contract from BSL to new system api
* Implementation includes requirement gathering, analysis, microservice architecture design and development of one poc which will implement LTI approach.

## Out of Scope

* Procurement or installation of any software, products, tools.
* Changes/ modifications to lifepen/BSL involved in the integrations
* Preparation of business use cases or business requirement documents
* Fixing source data quality problems or changing exiting business logic.
* Infrastructure vulnerability assessment and fixes
* Additional interfaces, services, Integration, testing, defect fixing & support not mentioned in Scope

# Integration Solution Blueprint

Integration Architecture Blueprint specifies the building blocks needed for an effective implementation of integration solutions. It ensures consistent quality in the implementation of integration strategies as a result of a simple, tried-and-tested structure, and the use of familiar integration pattern.

APIs make it possible for the enterprise’s core assets to be reused, shared, and even monetized to produce new revenue streams. The recommended integration approach is building APIs using API led connectivity which will work a bridge between the modern front-end applications (Channels) and the legacy mainframes. It will look beyond the traditional boundaries of Lifepen(Policy Administration) and extend the ways in which their data and logic connect.

Following key aspects in the design and development of the target-state application

* Adopt a modular approach for ease development and maintenance using Microservices
* Domain-driven design represent an abstraction of a business domain, which aims to capture the exemplary aspects of a specific implementation for this domain
* Think in terms of APIs to make integrations and interoperability easier
* Create an API layer to facilitate seamless communication
* Break monolithic blocks into fine grained components
* decoupling with a promise of simpler,
* Easier and cheaper services that are more reusable
* Create scalable, and flexible Data, business, technology, and information architectures with a future, platform usage perspective



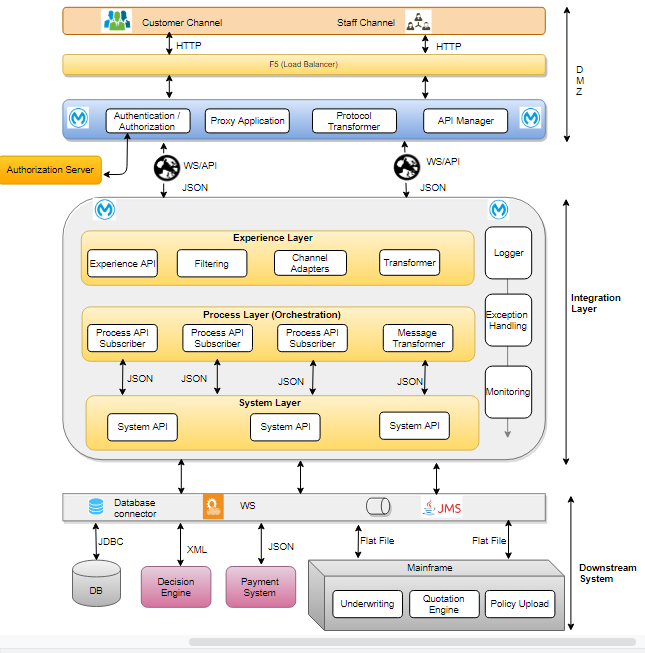
## Enterprise Integration Conceptual Architecture

Integration Architecture uses common standardized techniques, components, and patterns, and is based on the layered architecture principle.

As per requirement, LTI will leverage MuleSoft to implement System APIs (as a bridge) over mainframe. MuleSoft provides support to IBM WebSphere and its capacity to transparently use Spring for advanced configuration makes it an ideal choice for data flow from/to Mainframe via R2DS. MuleSoft integration based on IBM WebSphere helps in defining interfaces for connectivity, message transformations and exception handling.

The proposed integration solution provides loosely coupled and independent services to achieve centralized communication amongst internal systems. The solution is scalable to accommodate future integrations in a simple and cohesive way without impacting the existing integrations.

Based on HSBC’s well-defined scope of work and recommended technology stack, following is a conceptual view of the enterprise integration architecture handling both REST API and messaging model between Channels and Manufacturing applications.



**Fig01: Conceptual Enterprise Architecture for HSBC Life insurance – Downstream Integration (Overall)**

As the diagram says, there are three different layers in HSBC life insurance architecture. The first one is the distribution layer which lies in eDMZ. Second one is the integration layer where lies the Experience, Process and System APIs. The last one or the downstream system is the manufacturing layer where lies Lifepen etc.

## Enterprise Integration Solution Approach

Enterprise Service Bus (ESB) will enable integration of services/applications by providing reliable set of mediation capabilities like application connectivity, data validation, message transformation, data enrichment, routing, service orchestration, and security. MuleSoft will define integration patterns on Message broker so that it can leverage its distinct framework to define interfaces for connectivity, as well as message transformations and exception handling.

The Listen/Reply pattern will enable messages to be sent/receive message from/to liefepen system synchronously. Listen/Reply messaging enables event-driven architectures and decouple applications to increase performance, reliability and scalability.

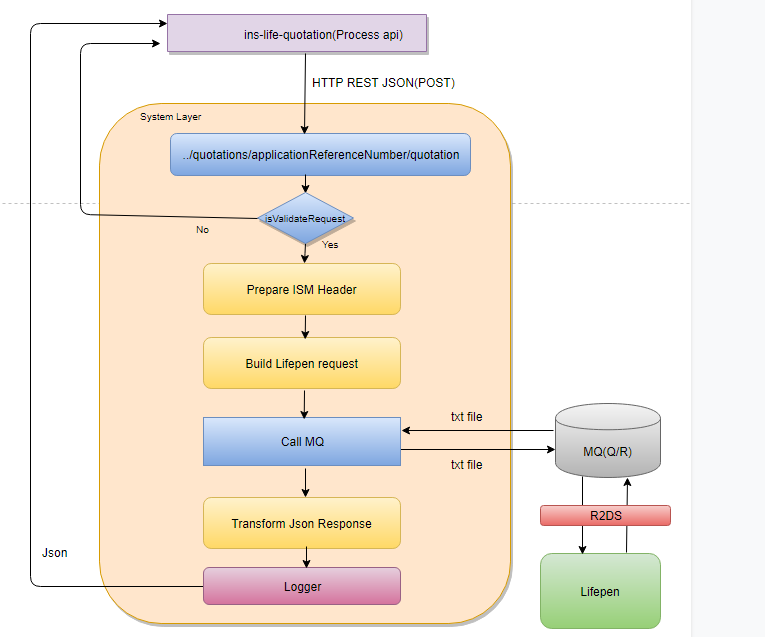
# Proposed Solution Architecture



## Technical Architecture

BSL layer acts as the intermediate layer between distributor layer and various downstream interfaces like lifepen, CDU. It is represented in a 3-layer pattern (Experience layer, Process layer and System Layer) to factor reusability, scalability and agility requirements. But as of now, one of process api prepare lifepen request after applying complex algorithm and call one core lifepen function (generate quote) directly. To decompose this functionality, system layer api will be ready which would enable HSBC to incorporate new changes into this layer without much impact into the system. This approach has the following advantages:

* + System layer enables loose coupling by providing a platform to lifepen system. This also avoids any sort of interference with an already complex IT ecosystem present in the business.
  + Provide a means of accessing these underlying systems of records and exposing the data in canonical formats.
  + Defines the contract RAML to describe how to interact with the domain.
  + For a quotation api can contain resources with methods like GET, POST, PUT, and DELETE, and the related schemas (JSON) and responses (200, 400, 500, etc).
  + System layer can be independent/isolated serving one purpose or will connect with backend system through existing MQ.



.

# Physical (Deployment) Architecture

The Physical/Deployment Architecture provides a detailed model of the way components will be deployed across the system infrastructure



## Deployment Architecture

## Environment Requirement

< During the lifecycle of the Project, various software environments need to be built starting from development till production. These details to be provided in this section>

## Integration View

<The Integration view of the architecture aims at providing the details of high level interactions among the various applications involved in the solution>

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Serial Number** | **Source Application** | **Destination Or Provider Application** | **Interface Type** | **Key Input** | **Key Output** | **Description** |
| 1 | <Name of the App> | <Destination App> | Online<Interface Type> (Interface name) |  |  |  |
| Online (Interface Name) |  |  |  |

## Information Model

### High Level Conceptual Data Model

The conceptual information model provides a framework on how the stated business requirements can be supported in terms of a model and also the base for corresponding logical model.

### Model & Definitions

Level 0 conceptual data or information model shows high level conceptual subject areas and concepts under them

### Draft Logical Data Model

<Provide draft Logical data or information model. Can be Erwin model >

## Processes and Services View

### High Level Service Catalogue

The Services Catalogue defines high level list of services along with the operations provided. It also provides high level view of the contracts. It is intended for single view of the services provided so that they can be reused

### Solution - Processes and Services View

The combined process and services view of the solution gives much needed linkage between process, services and users interfaces

## Non-Functional Requirements & Solutions

<Provide the details of the Non Functional Requirements that are required for the solution. More details on this is covered as part of the NFR requirements collection template as part of the SDLC.>

## Migration

### Run-time Process Migration and Maintenance

<Migration related aspects to be covered and defined in detail in this section.>

## Organizational Policies and Guidelines

<Organizational Policies to be considered and relevant ones to be mentioned in this section>

### Data Protection

### Application Software Governance

### Information Security

### Production Change Control

# Appendix



## Glossary

This section provides the definitions of all the terms, acronyms and abbreviations that are used in this document.

|  |  |
| --- | --- |
| **Term** | **Description** |
|  |  |
|  |  |
|  |  |
|  |  |