**FACULTY OF ENGINEERING AND COMPUTING**

**School of Computing**

**MSc. DEGREE**

**IN**

**Software Engineering**

***Project Proposal***

**Name: Chathura Samarasinghe**

ID Number: K2166813

**Project Title: Business Support System Framework for Provisioning of Telecom Value-Added Services**

**Supervisor: Prof. Ruvan Abeysekara**

Date: December 4, 2021

**Abstract**

Value-added services (VAS) stand for non-core telecommunication services offered by telecom companies to their subscribers. These services usually do not align with the main services provided by telecom such as voice calls. VASs are a collection of advanced optional services that are designed for promoting telecom providers’ primary business. VASs can be intelligently used to increase telecom providers’ revenue by providing the best services to the subscribers and keeping them busy on their mobile devices.

For the telecom operators, it is critical to maintaining high availability of the VAS software system. Also, it is crucial to serving every VAS request of subscribers without getting blocked and at a minimum response time. Currently, in-use software system of the client telecom company does not have the capacity to serve every request and keep up with high loads. It often goes offline due to technical issues of this legacy software system. Also, it takes a considerable amount of time to create and deploy new VASs to the subscribers.

This product focus on a new approach to replace the current monolithic software system. The proposed solution introduces a micro-services based distributed software platform deployed on Kubernetes network. A platform with all features complemented towards telecom provider’s effort to stay ahead of competitors with rapid value-added service creation and deployment.

Table of Contents

[1. Introduction & Background 1](#_Toc89528771)

[1.1 Introduction 1](#_Toc89528772)

[1.2 Motivation 1](#_Toc89528773)

[1.3 Problem in brief 2](#_Toc89528774)

[1.4 Proposed Solution 2](#_Toc89528775)

[1.5 Key Features 3](#_Toc89528776)

[2. Aim & Objectives 3](#_Toc89528777)

[2.1 Aim 3](#_Toc89528778)

[2.2 Objectives 3](#_Toc89528779)

[4. Technology Stack 3](#_Toc89528780)

[4.1 Frontend application 4](#_Toc89528781)

[4.2 Backend system 4](#_Toc89528782)

[5. Design 5](#_Toc89528783)

[6. Resource Requirement 6](#_Toc89528784)

[7. Development Methodology 7](#_Toc89528785)

[8. Work Plan 8](#_Toc89528786)

[8.1 Deliverables and Milestones 8](#_Toc89528787)

[9. Discussion 8](#_Toc89528788)

[9.1 Security concerns 8](#_Toc89528789)

[References 9](#_Toc89528790)

**Glossary of Terms**

|  |  |
| --- | --- |
| Micro-service architecture | A sub variant of service oriented structural architecture type |
| Message Queue | A message queue is a queue of messages passes between services |
| SMS Gateway | This enables software to send and receive SMS messages |
| WAP | Wireless Application Protocol is a technology that allow your mobile phone to browse the Web. It's a protocol for the transmission of data over low bandwidth wireless networks. [1] |
| OCS | Online Charging System |
|  |  |
| PCRF | Policy and Charging Rules Function |
|  |  |
| API | Application Programming Interface |
| VAS | Value Added Service |
| Server node | A physical server computer acting as a single unit of a server cluster |
| Subscriber | Customer of the business |
| MSISDN | Identifier of subscribers’ mobile number |

# Introduction & Background

## 1.1 Introduction

In today’s competitive world, it is critical for telecom providers to get into the market as soon as possible and capitalize their subscribers by providing valuable services in addition to their core telecoms services such as voice calls, internet access, and text/multimedia messaging. One strategy to provide such services to the subscribers is the use of Value-Added Services (VAS).

These services are provided either for free or for a limited subscription and they fall under a wide variety of service categories, namely, e-commerce, gaming, news, music, TV, and more…

VAS is often provided as short-term and long-term sustainable services. Creating of VASs depends on multiple factors such as market needs and competitor activities.

VAS is used by telco service providers to;

* Increase profit by selling add-on services
* Attract more customers
* Create comfortable service packages and provide more control to the customer over the services [2]

Currently, the client of this proposed system uses a traditional monolithic software platform to manage these VASs.

## 1.2 Motivation

The process of VAS request processing is; customer sends an SMS or a USSD query containing command keywords requesting a specific service, then an SMSC gateway will receive this message and forward it to the VAS system. VAS system will then do all the processing by going through related services and sending an SMS as a response back to SMSC and then to the subscriber number.

In order to serve customer requests of VAS, the VAS management platform interacts with a number of in-house built software systems (APIs, web services), payment gateways, and vendor-specific platforms such as Huawei OCS for charging, Huawei PCRF for rule-based charging, etc… The process of serving VAS requests is typically complicated due to the nature of internal system infrastructure, monolithic architectures, and the practices are being followed by the provider when it comes to software development. This is common to every telco provider in the market.

There are key challenges that exist in the current business environment.

* High time to market.

Creation and deployment take time to wrap up.

* By the time changes are deployed, requirements are expired in the market.
* Difficulty to keep up with the new technology stack due to customizations of core services.
* No standardization leads to an integration mess

Modern software solution is needed to mitigate those key challenges of the business.

## 1.3 Problem in brief

Due to the complexities that were pointed out in section 1.2, VAS management platforms suffer from a number of issues, namely, maintenance and operations, monitoring, downtimes, request blockings, etc…

When the business requires to make a change existing service it would take a few days to weeks to build a patch, test, and deploy it to production.

In a competitive market, those issues often lead to negative impacts on business and decreased customer satisfaction. It requires a modern platform to overcome those concerns.

## 1.4 Proposed Solution

The proposed solution takes a new approach based on micro-service architecture to ensure the availability and integrity of the system. This will completely replace the software system which is currently in use and provide a much more simplified user interface to develop VASs and deploy them to the subscribers.

The implementation will result in the following essentials.

* Performance Enhancement.
* Technology Upgrade.
* Ability to configure services through a user interface.
* Minimized complexity.

By using this platform, the business can effectively deploy new services, change them when required, monitor the system without a hassle.

## 1.5 Key Features

* **Low-code service aggregator user interface**

Proposed solution introduces a low code API designer and builder, there will not be any need to implement codes and write test cases by using any programming languages. This low code API designer can be used to design service flows, aggregate external APIs, and create unified responses. This generates an XML as the service definition and it will be stored in the database, later when the service is requested backend processing engine will read the corresponding XML file and process it with the given data.

* **Micro-service stack**
* **Reusable telco specific adaptors**

Solution includes a set of adaptors that can be used to interact with external services such as API gateways, payment gateways, etc…

* **Server-less support for service aggregation APIs**

Each API build using API designer can be deployed to the system and works as HTTP endpoints. There is no need of separate HTTP servers to run.

# Aim & Objectives

### 2.1 Aim

### Develop a BSS (Business Support System) framework to manage and control Value Added Services.

### 2.2 Objectives

1. To provide a formal framework for VAS request handling.
2. To provide a web application to VAS provisioning.
3. To carry out research on technologies available under JavaEE/Spring ecosystem and low-code technologies.
4. To build a graphical low-code workflow builder.
5. To build a processing engine to execute workflows.
6. To provide a dynamic REST API deployment platform.

# 4. Technology Stack

The proposed solution will be built using carefully chosen modern software liberties, frameworks, and technologies. The backend APIs and the codebase will be developed in Java EE and the frontend web user interface will be implemented in JavaScript. The entire system will be deployed in an in-house Kubernetes cluster to ensure high availability.

Below tech stack will be used to implement the overall system.

### 4.1 Frontend application

Since this application is developed to run on a modern web browser the frontend application will be written in JavaScript with React JS library. React JS library is chosen to implement the frontend application to get built-in facilities of MVC architecture. According to the surveys [3], React JS is one of the most popular web frontend frameworks currently in use by many companies and developers. Also, React JS supports a wide range of 3rd party JavaScript libraries that are available for user interface development.

Below technologies will be used to develop frontend application.

* JavaScript
* React JS
* Webpack – to bundle runtime packages [4] application
* Yarn – to manage libraries
* Nginx – deployment server

### 4.2 Backend system

The backend system has four main parts.

* Micro-service stack
* Message Queues
* SMS gateway
* Database

Services of the micro-service stack will be written in Java using Java Enterprise Edition. This will be developed on top of Spring boot, Spring MVC framework. Spring framework is a widely used Java-based web framework with a lot of built-in modules to support almost every commonly used technology. Spring framework is selected since it provides support to database access, message queue access, and MVC architecture.

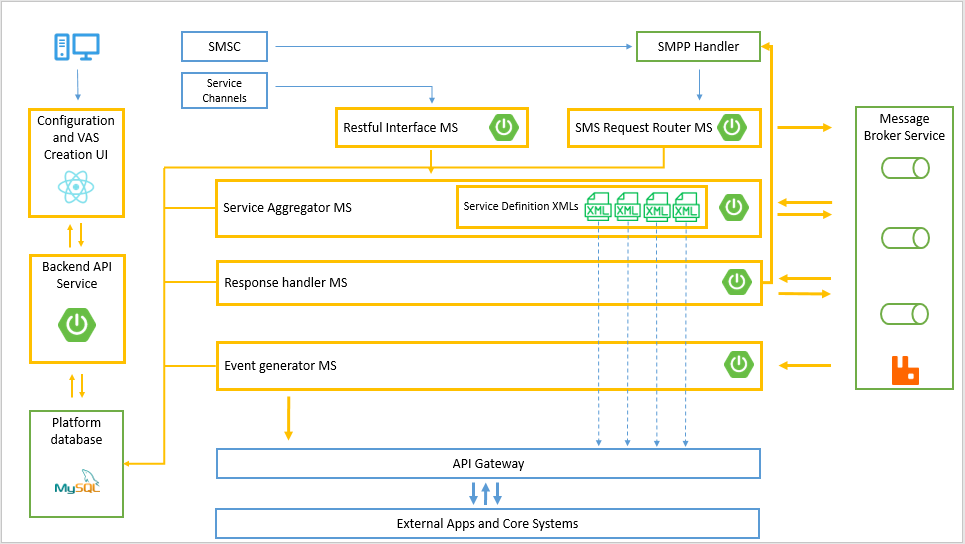
As for the messages queues, RabbitMQ is selected since it is one of the most stable and efficient message brokers available. A message queue is needed to ensure high availability and keep all VAS requests intact if any of the core systems went offline.

Kannel is an open source WAP and SMS gateway system that lets software to communicate with an SMSC

MySQL will be used as the underlying database to store request details, SMS messages, responses, statistics details, and telemetry data.

# 5. Design

* The request via SMS is sent by the mobile will reach SMSC.
* SMSC will then act as a communicator and make a bridge to transfer the request to SMPP Client (Kannel Server) as an SMTP request.
* SMPP Client then passes the request as an HTTP API Call to the SMS Router.
* SMS Router will then accept and send the HTTP call to a message broker (RabitMQ), which acts as a Queue Service. RabitMQ adds the call to one of the queues in the message broker for processing. A response will be then sent back to the SMPP Client, confirming the request sent has been accepted by the queue service
* The SMS Router can read and extract responses from the queue server by itself as well.





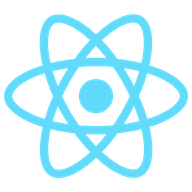


Figure 4.1 – System design

# 6. Resource Requirement

|  |  |
| --- | --- |
| Client side | Server side |
| Software Resources   * Modern web browser with JavaScript ECMA 5 enabled   Hardware Resources   * Computer device   Network Resources   * Intranet connectivity | Software Resources   * Nginx server latest version * Kubernetes * Docker, Docker private registry * MySQL 8.0 * Java 1.8 or higher * RabbitMQ * Kannel   Hardware Resources   * Server cluster with 6 nodes for Kubernetes deployment   (8GB RAM, 4 core processor, 100GB HDD, a dedicated IP for each node)  Network Resources   * SMS gateway access * Internal API gateway access |

Table 5.1 – Resource requirements

# 7. Development Methodology

This software platform will be developed according to the principles defined in Kanban agile methodology.

Kanban is a visual system for managing work as it moves through a process. Kanban visualizes both the process (the workflow) and the actual work passing through that process [5].

Kanban methodology is selected since it promotes an incremental approach to develop systems.

The nature of this proposed system is mainly feature based. Each feature can be easily divided in to individual tasks. These tasks then can be implemented in a sequence of iterations.

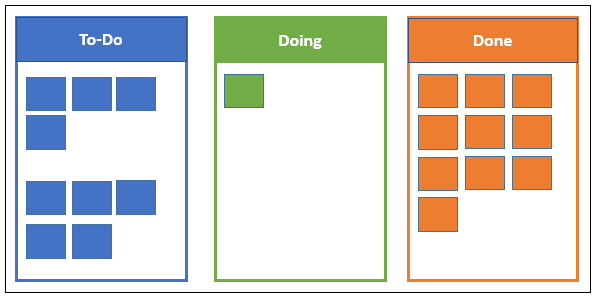


Figure 7.1 – Kanban development methodology

Initially all the identified tasks will be added to the “To-Do” section of the Kanban board, then a task will be picked up to work on. That then going to be added in to the “Doing” section. “Doing” section contains all the tasks that are being developed. Once, development of a task is completed, it will then be added to the “Done” section. “Done” section holds all the tasks that had been completed.

# 8. Work Plan

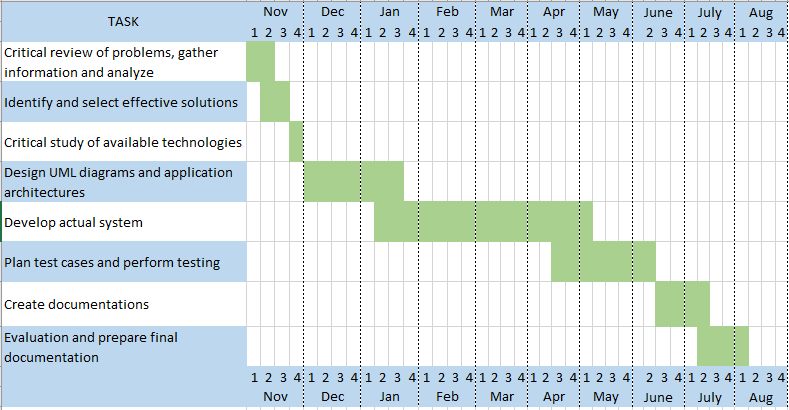


Figure 8.1 – Work plan

### 8.1 Deliverables and Milestones

* Design UML diagrams
* Develop strategy for keyword routing
* Implement micro-services
* Implement process engine
* Develop testing strategy and test cases
* Implement in-bound and out-bound SMS channels
* Prepare demo Docker/Kubernetes cluster
* Create product documentation

# 9. Discussion

### 9.1 Security concerns

This system is deployed in an internal server environment of the clients’ server infrastructure. Therefore, at the deployment and user acceptance test (UAT) phase, certain security best practices must be followed.

Access control and privilege enforcement mechanisms must strictly be followed according to the guidelines provided by the internal security team in order to secure internal services from malicious, unwanted activities and unprecedented mistakes.

The system must provide a secure credential store to store service tokens. Service aggregator micro-service uses a credential store to keep access tokens, API keys, and passwords of certain external services such as API gateway, SMSC gateway, and payment gateway. These data must be stored in cryptographically secure storage. Also, in an event of a security breach or related concern, stored data must be invalidated and re-imported.

Personally identifiable data of customers/subscribers except the MSISDN should not append into logs files. Once the processing is completed, customers’ data must be cleared from the session.

Since the proposed solution is a core framework for VAS management, it does not require authorizations and approvals from gateway systems at the development phase.

# References

[1] devel@kannel.org, “Kannel: Open Source WAP and SMS Gateway,” *Kannel.org*, 2020. https://www.kannel.org/glossary.shtml#1.11.

[2] S. Sutherl, “What is VAS? - Adapt IT Telecoms,” *telecoms.adaptit.tech*, 2021. https://telecoms.adaptit.tech/blog/what-is-vas/.

[3] “Stack Overflow Developer Survey 2020,” *Stack Overflow*. https://insights.stackoverflow.com/survey/2020#most-popular-technologies.

[4] “webpack,” *webpack*, 2019. https://webpack.js.org/.

[5] Digite, “What is Kanban?,” *Digite*, Oct. 04, 2017. https://www.digite.com/kanban/what-is-kanban/.