**TXTaGO**

**Mobile Messaging As a Service**

|  |
| --- |
| A Project Report Presented to  The Faculty of College of Engineering |
| San Jose State University In Partial Fulfillment Of the Requirements for the Degree Master of Science in Software Engineering |

|  |
| --- |
| By |
| Stephen Kwan |

May - 2013

**Project Source**

<https://github.com/kwanstephen/txtago.git>

**Deployment**

<http://txtago.skwan.cloudbees.net/>

|  |
| --- |
| Copyright © 2013 |
| Stephen Kwan |
| ALL RIGHTS RESERVED |

|  |
| --- |
| APPROVED FOR COLLEGE OF ENGINEERNG |
|  |
| Ronald L. Coens, Industry Advisor |
|  |
| Dr. Lee C. Chang, Academic Advisor |

This page was left blank intentionally

**ABSTRACT**

TXTaGO – Mobile Messaging As a Service

By Stephen Kwan

The mobile messaging ecosystem is divided into four groups. On top, you have the wireless carriers. The top four carriers (Verizon, AT&T, Sprint, T-Mobile) account for 90% of the US’s 300 million subscribers. Then you have the mobile service providers like Sybase 365, Syniverse, and OpenMarket. The mobile service providers are the clearinghouse for mobile messages. They provide carriers with inter-carrier connectivity and content provider aggregation. Next you have the content providers, who are the marketing companies that work with customers to create campaigns to engage mobile subscribers. Lastly, you have the mobile subscribers, who are the consumer or producer of the content.

Sybase 365, the largest mobile message provider, has saturated its market in the mobile messaging business to business (B2B) space. It has reach to over 900 network operators and hundreds of content providers. In order to grow the business further, Sybase 365 needs to provide more than just plumbing for mobile messaging. Sybase 365 needs to provide value-added service, such as the ones offered by the content providers today. This business to consumer (B2C) space will allow Sybase 365 to capture more revenue in the mobile messaging ecosystem.

This project, TXTaGO, will provide mobile messaging as a service. The cloud service, TXTaGO, will act as a content provider to manage the enterprise marketing campaigns and provide plumbing for mobile messages to all end-users. It will offer shortcode and keyword provisioning, subscriber opt-in management, campaign management, and traffic analytics. TXTaGO will utilize the latest in cloud technology, scalable to millions of transactions, priced as a pay-as-you-go model, with all features accessible through API.

**Acknowledgments**

I like to thank Mr. Ronald Coens and Dr. Lee Chang for their guidance on the project. I would also like to thank my family, especially my wife, Susana, for her putting up with my hectic schedule of school and work.

Table of Contents

[Chapter 1 Introduction 11](#_Toc350455746)

[1.1 Project Goals and Objectives 11](#_Toc350455747)

[1.2 Background, Problems and Motivation 11](#_Toc350455748)

[1.3 Project Applications and Impact 12](#_Toc350455749)

[1.4 Project Results and Deliverables 12](#_Toc350455750)

[1.5 Market Research 13](#_Toc350455751)

[1.6 Technology Trends 15](#_Toc350455752)

[1.7 Literature Survey 17](#_Toc350455753)

[Chapter 2 Requirements and Analysis 19](#_Toc350455754)

[2.1 Business modeling 19](#_Toc350455755)

[2.1.1 Activity Diagram and Business Processes 19](#_Toc350455756)

[2.1.2 Domain Class Model 29](#_Toc350455757)

[2.2 Functional Requirements 40](#_Toc350455758)

[2.3 Nonfunctional Requirements 42](#_Toc350455759)

[2.3.1 Quality of Service 42](#_Toc350455760)

[2.3.2 Constraints 43](#_Toc350455761)

[2.4 Use Case Modeling 44](#_Toc350455762)

[2.4.1 Use Case Diagram 44](#_Toc350455763)

[2.4.2 Use Case Description 45](#_Toc350455764)

[2.4.3 Sequence Diagram 55](#_Toc350455765)

[2.4.3.4 Process Payment 58](#_Toc350455766)

[2.4.4 Use Case Interaction Overview Diagram 65](#_Toc350455767)

[2.4.5 Requirement Traceability Matrix 66](#_Toc350455768)

[2.5 Context and Interface Requirements 68](#_Toc350455769)

[2.5.1 Context Requirements 68](#_Toc350455770)

[2.5.2 Interface Requirements 68](#_Toc350455771)

[2.6 Resource Requirements 69](#_Toc350455772)

[2.6.1 Hardware Requirements 69](#_Toc350455773)

[2.6.2 Software Requirements 69](#_Toc350455774)

[2.6.3 Technical Skills Requirements 70](#_Toc350455775)

[Chapter 3 System Design 71](#_Toc350455776)

[3.1 Architecture Design 71](#_Toc350455777)

[3.2 System Logic and Partition Design 72](#_Toc350455778)

[3.3 System Interface and Connectivity Design 90](#_Toc350455779)

[3.3 System Data and Database Design 91](#_Toc350455780)

[3.4 User Interface Design 92](#_Toc350455781)

[3.4.1 REST API Interface 97](#_Toc350455782)

[3.5 Design Trade-offs, Problems and Solutions 101](#_Toc350455783)

[Chapter 4 System Implementation 102](#_Toc350455784)

[4.1 Implementation Summary 102](#_Toc350455785)

[4.2 Develop or Adopt Decision 103](#_Toc350455786)

[4.3 Implementation Process 104](#_Toc350455787)

[4.4 Tools and Environments 106](#_Toc350455788)

[4.5 Standards 107](#_Toc350455789)

[4.6 Implementation Issues and Resolutions 108](#_Toc350455790)

[Chapter 5 Testing Plan 109](#_Toc350455791)

[5.1 Testing Scope 109](#_Toc350455792)

[5.2 Testing Approaches 109](#_Toc350455793)

[5.3 Testing Results 110](#_Toc350455794)

[Chapter 6 Conclusion and Future Work 113](#_Toc350455795)

[6.1 Project Summary 113](#_Toc350455796)

[6.2 Future Work 115](#_Toc350455797)

[Appendix 116](#_Toc350455798)

[References 117](#_Toc350455799)

**Table of Figures**

[Figure 1 Mobile Services Providers 13](#_Toc350455800)

[Figure 2 Content Providers [2] 14](#_Toc350455801)

[Figure 3 Mobile Messaging Value Chain[3] 15](#_Toc350455802)

[Figure 4 Mobile Messaging Growth [3] 16](#_Toc350455803)

[Figure 5 Skype Traffic [5] 17](#_Toc350455804)

[Figure 6 Process Summary Diagram 19](#_Toc350455805)

[Figure 7 Manage Account Activity Diagram 20](#_Toc350455806)

[Figure 8 Manage Campaign Activity Diagram 21](#_Toc350455807)

[Figure 9Manage Contact Activity Diagram 22](#_Toc350455808)

[Figure 10 Manage Content Activity Diagram 23](#_Toc350455809)

[Figure 11 Manage Message Activity Diagram 24](#_Toc350455810)

[Figure 12 Manage Payment Activity Diagram 25](#_Toc350455811)

[Figure 13 Manage Report Activity Diagram 26](#_Toc350455812)

[Figure 14 Manage Subscription Activity Diagram 27](#_Toc350455813)

[Figure 15 Manage User Activity Diagram 28](#_Toc350455814)

[Figure 16 Domain Class Diagram 29](#_Toc350455815)

[Figure 17 Domain Class Listing 30](#_Toc350455816)

[Figure 18 Account State Diagram 31](#_Toc350455817)

[Figure 19 Campaign State Diagram 32](#_Toc350455818)

[Figure 20 Contact State Diagram 33](#_Toc350455819)

[Figure 21 Content State Diagram 34](#_Toc350455820)

[Figure 22 Message State Diagram 35](#_Toc350455821)

[Figure 23 Payment State Diagram 36](#_Toc350455822)

[Figure 24 Report State Diagram 37](#_Toc350455823)

[Figure 25 Subscription State Diagram 38](#_Toc350455824)

[Figure 26 User State Diagram 39](#_Toc350455825)

[Figure 27 Use Case Diagram 44](#_Toc350455826)

[Figure 28 Maintain User Sequence Diagram 55](#_Toc350455827)

[Figure 29 Maintain Account Sequence Diagram 56](#_Toc350455828)

[Figure 30 Process Subscription Sequence Diagram 57](#_Toc350455829)

[Figure 31 Process Payment 58](#_Toc350455830)

[Figure 32 Process Report Sequence Diagram 59](#_Toc350455831)

[Figure 33 Maintain Campaign Sequence Diagram 60](#_Toc350455832)

[Figure 34 Maintain Contact Sequence Diagram 61](#_Toc350455833)

[Figure 35 Maintain Content Sequence 62](#_Toc350455834)

[Figure 36 Process Campaign Sequence Diagram 63](#_Toc350455835)

[Figure 37 Process Message Sequence Diagram 64](#_Toc350455836)

[Figure 38 Use Case Interaction Diagram 65](#_Toc350455837)

[Figure 39 Requirement Traceability Matrix 67](#_Toc350455838)

[Figure 40 System Architecture Diagram 71](#_Toc350455839)

[Figure 41 Design Package Diagram 72](#_Toc350455840)

[Figure 42 Design Class Diagram 73](#_Toc350455841)

[Figure 43 System Interface Diagram 90](#_Toc350455842)

[Figure 44 Database ER Diagram 91](#_Toc350455843)

[Figure 45 Login Page 93](#_Toc350455844)

[Figure 46 Add User Page 94](#_Toc350455845)

[Figure 47 Report Page 95](#_Toc350455846)

[Figure 48 Add Campaign Page 96](#_Toc350455847)

[Figure 49 Component Diagram 102](#_Toc350455848)

[Figure 50 Test Case Summary 112](#_Toc350455849)

[Figure 51 Project Tasks 114](#_Toc350455850)

# Chapter 1 Introduction

## Project Goals and Objectives

The B2B mobile services provider business is a zero-sum game. The number of carriers in the world is fairly static. A gain of one customer by one service provider means a loss by another. Sybase 365 already have 900 plus mobile carriers in its portfolio. In order to grow the business, Sybase 365 needs to expand into the B2C mobile messaging space. The B2C space will open mobile messaging services to all users. Startups and independent developers will be able to gain access to the fastest growing mobile technology sector.

The goal of this project is to enable mobile messaging to the masses. Ninety-eight percent of the mobile phones out there today support mobile messaging. Currently, application to peer (A2P) traffic for these six billion handsets is controlled by the mobile service providers and content providers. TXTaGO will open up the ecosystem to anyone wanting to use it. This mobile messaging as a service is a cloud service with zero setup fee and price at a low pay as you go pricing.

## Background, Problems and Motivation

There are two types of mobile messaging traffic – peer to peer (P2P) and application to peer (A2P). P2P traffic is from one mobile subscriber to another. A2P traffic is from a system/application to a subscriber. Application to peer traffic is the form that’s used by enterprises to communicate with its customers. In order for a startup to participate in A2P traffic there are a lot of setup costs, not to mention lead time in setup and provisioning. The process is costly and slow.

This project will attempt to streamline the process and make mobile messaging accessible to all developers by providing mobile messaging as a service on a cloud platform. The motivation for Sybase 365 is the opportunity to expand its mobile services to developers and capture more of the content provider market. The benefit to the tech industry is the availability of a world class messaging platform to the general public.

## Project Applications and Impact

By providing an open API to access the mobile messaging ecosystem, startups, open source developers, and hobbyist will be able to create innovative applications and services around mobile messaging. As studies has shown “Global SMS traffic to reach 8.7 trillion by 2015” [1]. Mobile messaging has the highest penetration. It is the common denominator between the legacy handsets and smartphones.

Content providers which currently charge companies a premium to access the mobile messaging ecosystem will have to reduce their pricing and provide more value added services to command the premium.

## Project Results and Deliverables

The deliverables for the project will include:

1. Project Report of System and Market Overview, Design, Implementation, and Testing.
2. TXTaGO.com – Prototype. Features include registration, administration, and reporting.
3. TXTaGo REST API. Functionalities include campaign management and statistics retrieval.
4. Source Code on GitHub, <https://github.com/kwanstephen/txtago>

## Market Research

The three biggest mobile service providers in the market today are Sybase 365, Syniverse, and OpenMarket. These service providers have direct connections to the Tier 1 carriers around the world. Below is a table of their reach.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Company** | **Countries** | **Carriers** | **SMS** | **MMS** | **A2P** | **P2P** |
| Sybase 365 | 160 | 900+ | Yes | Yes | Yes | Yes |
| Syniverse | 160 | 900+ | Yes | Yes | Yes | Yes |
| OpenMarket | 160 | 400+ | Yes | Yes | Yes | Yes |

Figure Mobile Services Providers

All three of these mobile service providers cover all the countries in the world. All of them are capable of sending SMS and MMS on both channels A2P and P2P. The only difference between them is the number of direct carrier connections – OpenMarket with 400+ while Sybase 365 and Syniverse with 900+.

Content providers lease bandwidth from the mobile services providers. They are the customer interface to the marketing departments of the enterprises. The top three content providers as rated by *TopTen Reviews* are SumoText, EzTexting, and Trumpia [2]. Based on Figure 2, we can see that each text message costs about $0.05. Some of the basic features are inbound/outbound messages, short code maintenance, keyword management, and reporting.



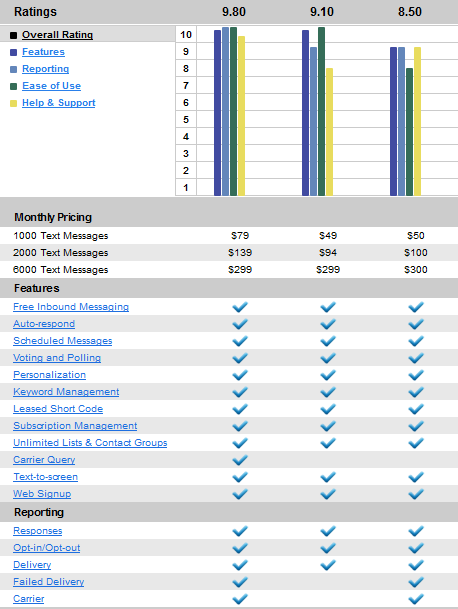


Figure Content Providers [2]

Here is what the mobile messaging value chain look like, see figure below. The direct interaction between the business and content provider are almost non-existent. There are too many content providers. Mobile operators do not want to manage them all. Therefore, the communication to content providers is outsourced to mobile service providers or mobile aggregators.

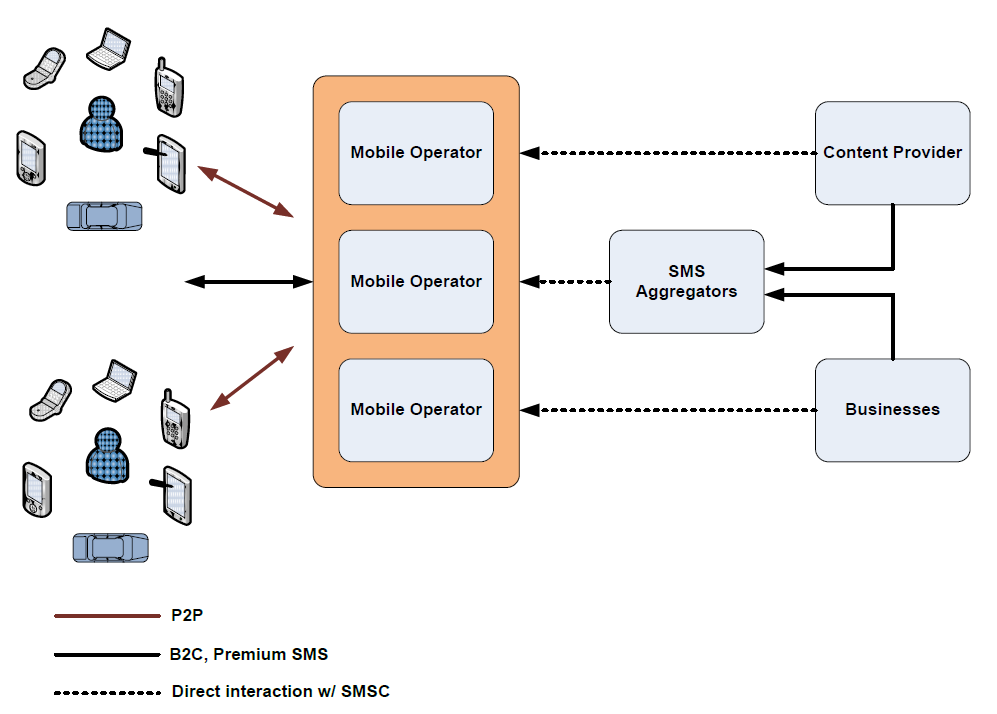


Figure Mobile Messaging Value Chain[3]

## Technology Trends

Mobile message usage is growing. In the United States, 2400 billion messages were send in 2010 [3]. If you look at the figure below, you can see a trend that is rising rapidly.

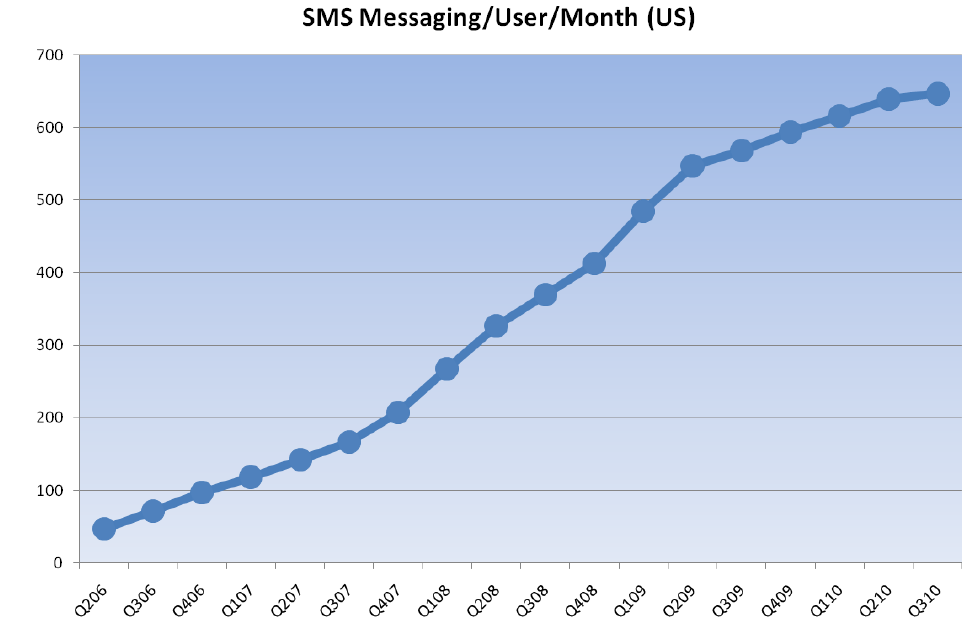


Figure 4 Mobile Messaging Growth [3]

Mobile messaging started out with mainly SMS (short message service). As the need for sending rich content arises, MMS (multi-media messaging service) was invented. MMS can contain many content parts and any mime content type, such as image, video, binary, or text.

As smart phones get more and more popular, data service plans are very common. With the data services and smart phones, a new set of services has popped up in the mobile messaging space. The over-the-top (OTT) services which provide messaging and voice without the need to directly go through the mobile operator’s network. These OTT services are installed as apps on the smart phones and it piggy-back on the data network of the operator. Some of the notable OTT players are Google Voice, Pinger, Gogii, and Skype [4]. If you look at voice component of the OTT service like Skype, though it has been gaining popularity, it has not cannibalize the voice call for the operators, see Figure 5 below. The Skype traffic has definitely grown from 2005 to 2011, but at the same time so has traditional voice traffic. The fact that most consumers will not have a Skype client, mobile operators will be able to continue to grow the voice traffic market. The least common denominator for messaging will always be SMS, which will also allow the operators to preserve the mobile messaging market.

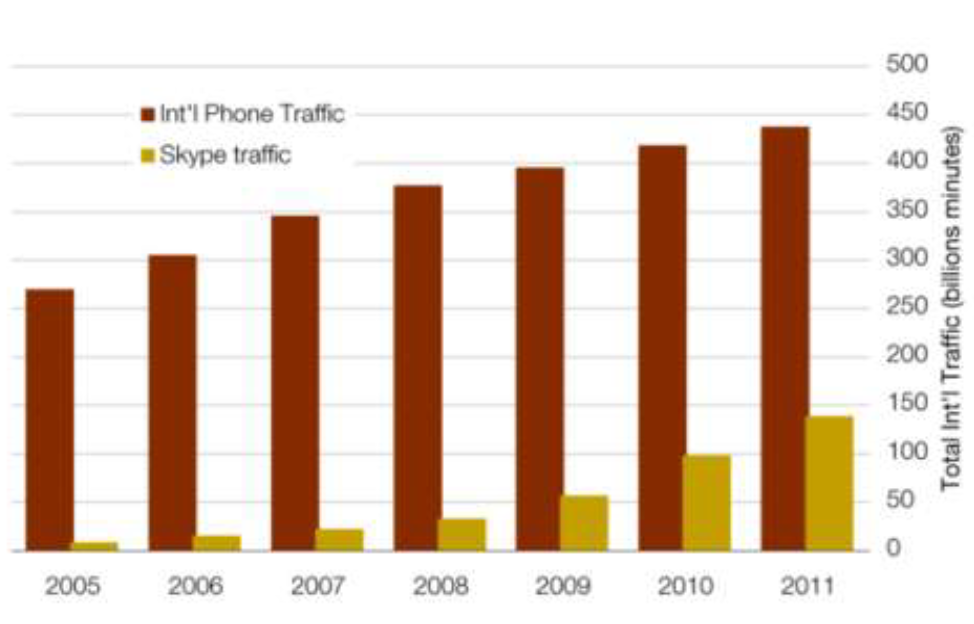


Figure Skype Traffic [5]

## Literature Survey

Mobile Messaging has been around for quite some time. The literature surveys and research projects on mobile messaging are primarily on mobile operators. This project’s focus is on creating a new type of content provider based on cloud technologies. First we looked at the mobile messaging market by analyzing, the article *US Mobile Messaging Market* [3]. Then we survey the competition to see what we are up against by studying, *2012 Best Text Message Marketing Service* [2]. We research what business model would work by looking at examples such as, *The Innovative Business Model behind the Rapid Growth of SMS in China* [6]. Finally, we took a look at how to make the business sustainable by reading*, Customer analysis of monthly-charged mobile content aiming at prolonging subscription period* [7]. These articles have contributed to the analysis, requirements, and product features of our project.

# Chapter 2 Requirements and Analysis

## 2.1 Business modeling

### 2.1.1 Activity Diagram and Business Processes

Figure 6 below summarizes the different processes involved in the TXTaGO system. The inputs to the system consist of the domain object requests and the outputs are the domain object responses.

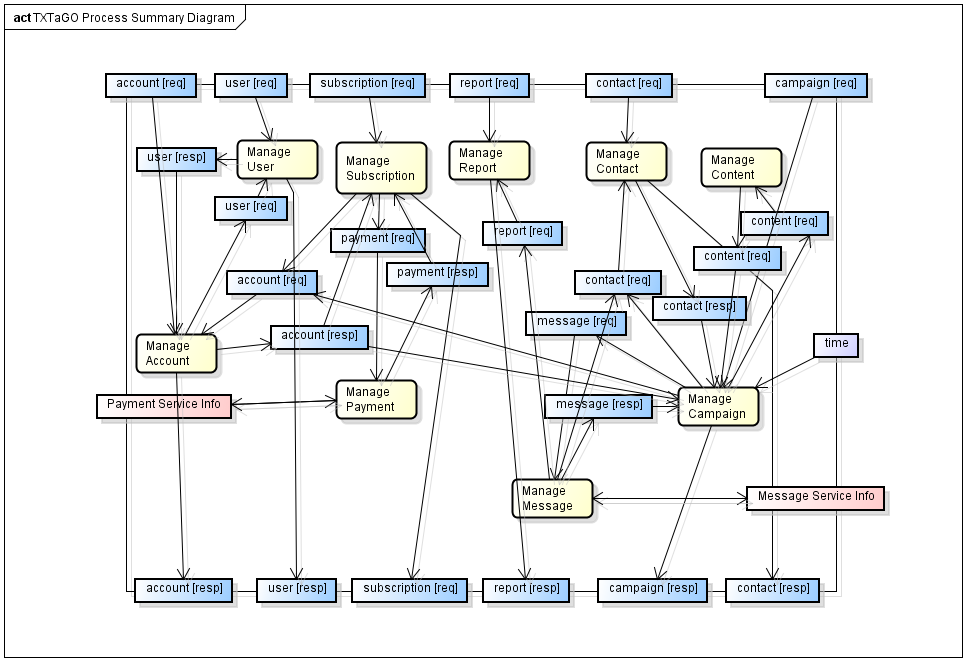


Figure 6 Process Summary Diagram

The decomposed Manage Account Activity Diagram - Figure 7 below, shows the activities involved in the managing an account.

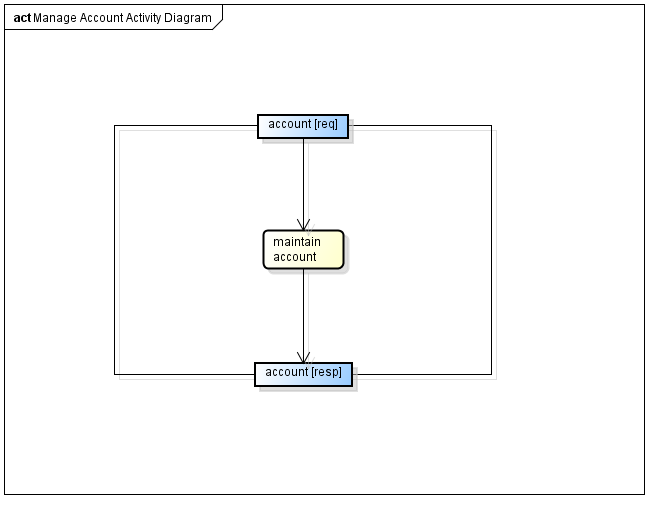


Figure Manage Account Activity Diagram

The decomposed Manage Campaign Activity Diagram - Figure 8 below, shows the activities involved in the managing a campaign.

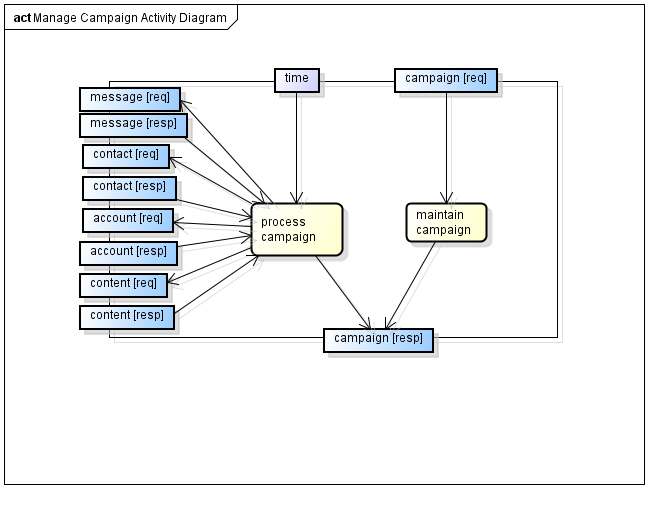


Figure Manage Campaign Activity Diagram

The decomposed Manage Contact Activity Diagram - Figure 9 below, shows the activities involved in the managing a contact.

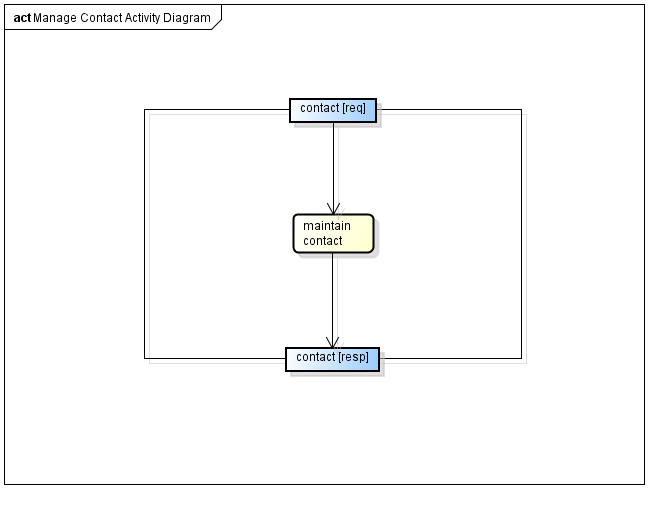


Figure Manage Contact Activity Diagram

The decomposed Manage Content Activity Diagram - Figure 10 below, shows the activities involved in the managing a content.

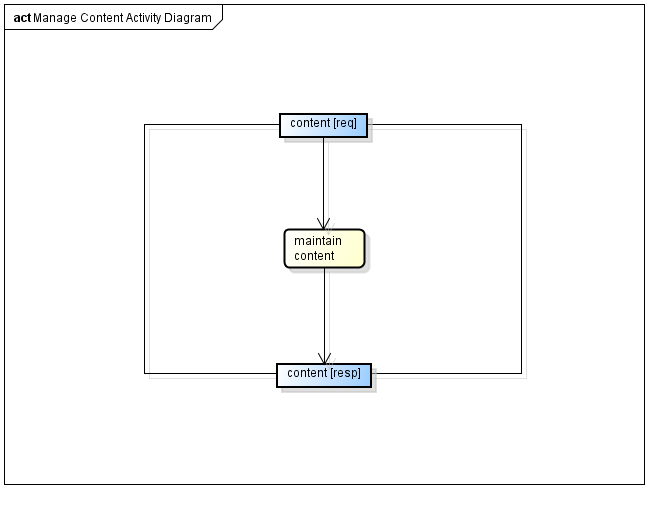


Figure Manage Content Activity Diagram

The decomposed Manage Message Activity Diagram - Figure 10 below, shows the activities involved in the managing a message.

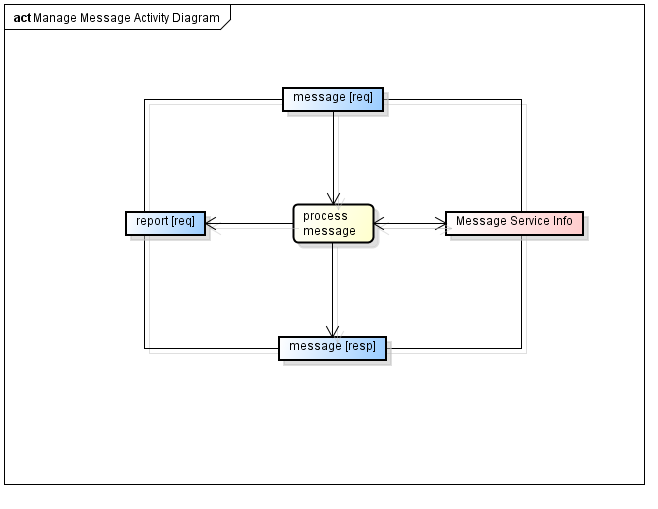


Figure Manage Message Activity Diagram

The decomposed Manage Payment Activity Diagram - Figure 11 below, shows the activities involved in the managing a payment.

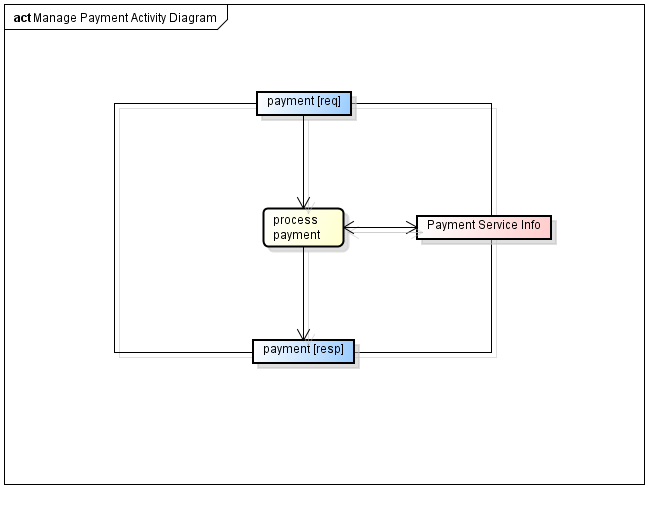


Figure Manage Payment Activity Diagram

The decomposed Manage Report Activity Diagram - Figure 11 below, shows the activities involved in the managing a report.

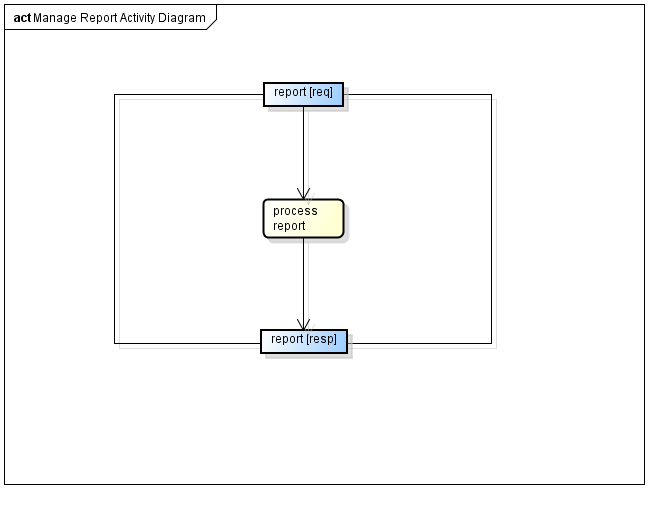


Figure Manage Report Activity Diagram

The decomposed Manage Subscription Activity Diagram - Figure 11 below, shows the activities involved in the managing a subscription.

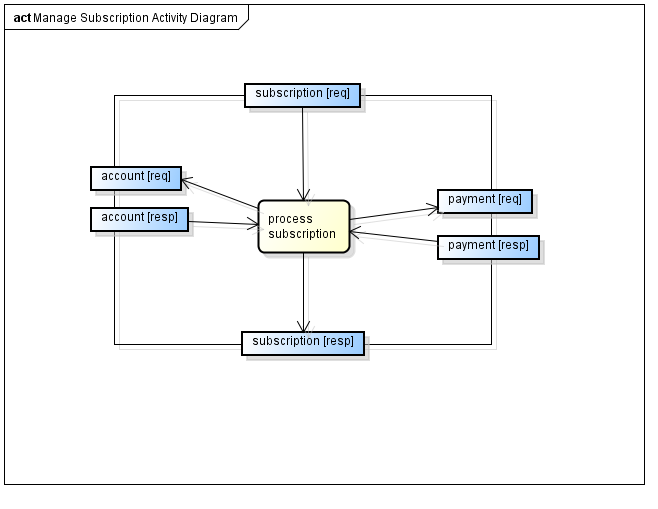


Figure Manage Subscription Activity Diagram

The decomposed Manage Subscription Activity Diagram - Figure 11 below, shows the activities involved in the managing a subscription.



Figure Manage User Activity Diagram

### 2.1.2 Domain Class Model

#### 2.1.2.1 Domain Class Diagram

Figure 16 shows the domain classes for the TXTaGO system.

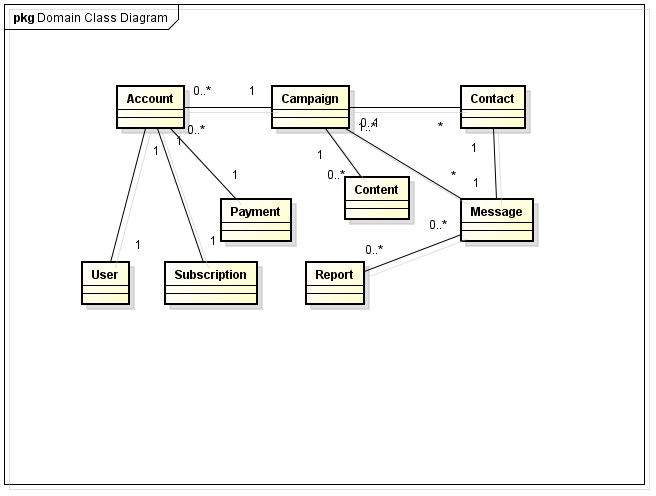


Figure Domain Class Diagram

#### 2.1.2.2 Domain Class Listing

The listing below, Figure 17, shows the main attributes of the domain class for the TXTaGO system.

|  |  |  |
| --- | --- | --- |
| Account  +accounted  +userId  +subscriptionId  +status  +cardNumber  +cvv  +expirationDate  +firstName  +lastName  +address  Campaign  +campaignId  +shortcode  +keyword  +userId  +description  +title  +deliveryDate  +deliveryHour  +status  Contact  +contactId  +phone  +campaignId | Content  +contentId  +campaignId  +contentType  +contentText  +contentUrl  Message  +messageId  +deliveryDate  +shortcode  +destinationMdn  +campaignId  +status  +userId  Payment  +userSubscriptionId  +subscriptionId  +userId  +status  +paidBillingId  +usage  +nextBillingDate | Report  +campaignId  +shortcode  +destinationMdn  +daysBack  +dataPoints  Subscription  +subscriptionId  +price  +messageLimit  +billingCode  +status  User  +userId  +email  +password  +firstName  +lastName  +role  +state |

Figure Domain Class Listing

#### 2.1.2.3 Domain State Diagram

The domain state models show the possible states of the domain objects. The diagrams to follow will illustrate the states of its respective objects.

Figure 18 – Account State Diagram below depicts the possible states of the Account object.

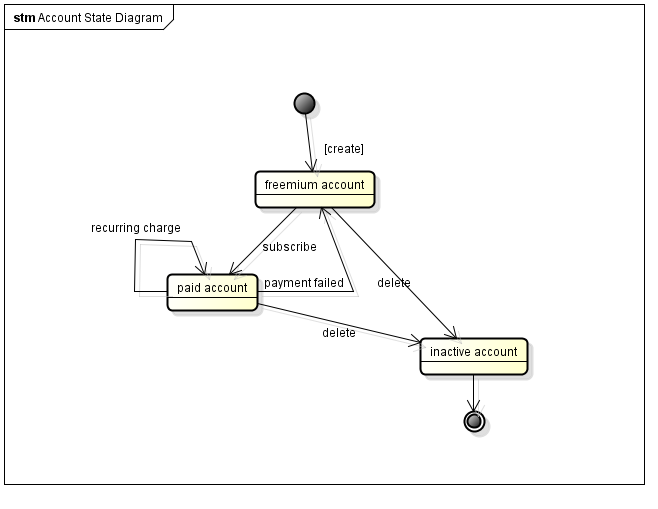


Figure Account State Diagram

Figure 19 – Campaign State Diagram below depicts the possible states of the Campaign object.

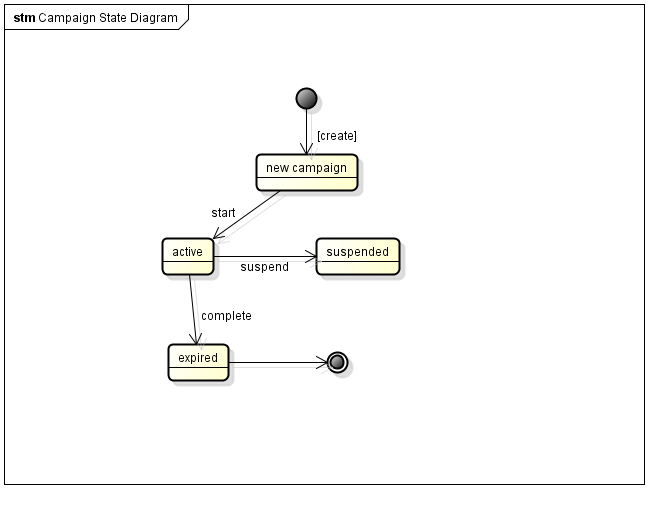


Figure Campaign State Diagram

Figure 20 – Contact State Diagram below depicts the possible states of the Contact object.

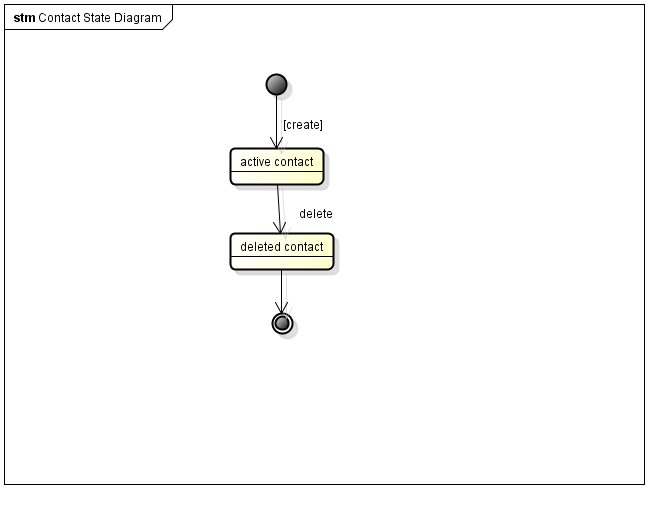


Figure Contact State Diagram

Figure 21 – Content State Diagram below depicts the possible states of the Content object.

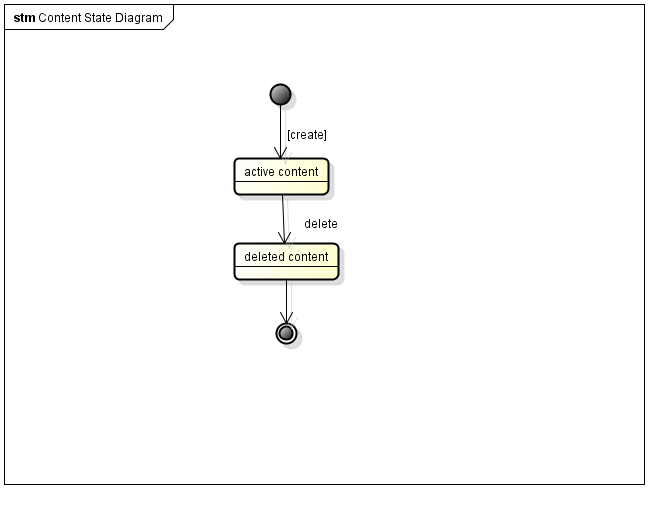


Figure Content State Diagram

Figure 22 – Message State Diagram below depicts the possible states of the Message object.

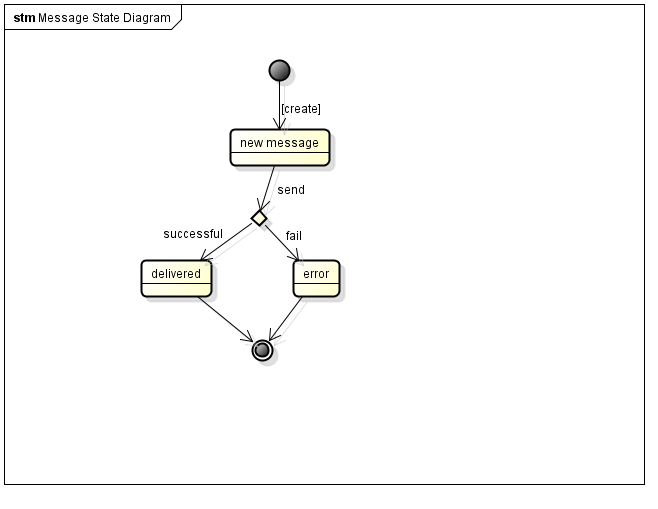


Figure Message State Diagram

Figure 23 – Payment State Diagram below depicts the possible states of the Payment object.

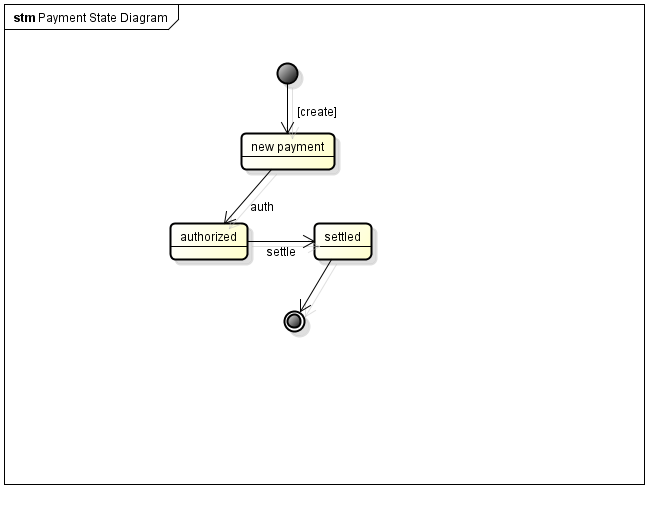


Figure Payment State Diagram

Figure 24 – Report State Diagram below depicts the possible states of the Report object.

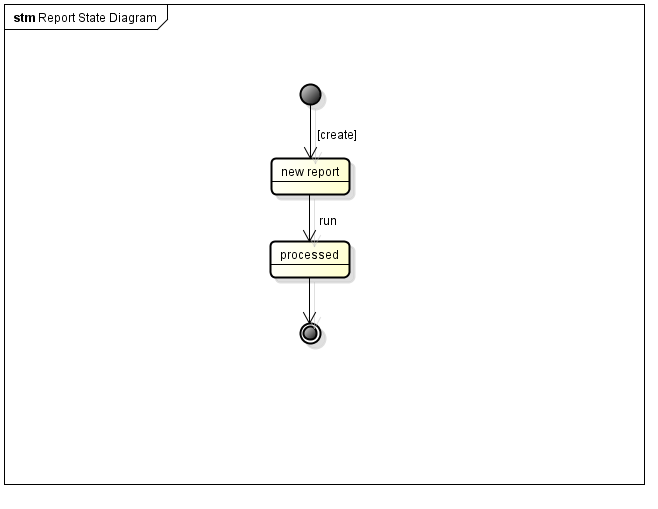


Figure Report State Diagram

Figure 25 – Subscription State Diagram below depicts the possible states of the Subscription object.

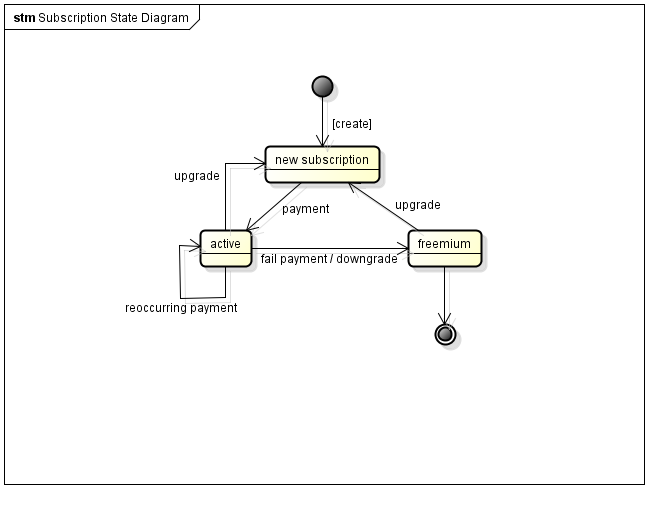


Figure Subscription State Diagram

Figure 26 – User State Diagram below depicts the possible states of the User object.

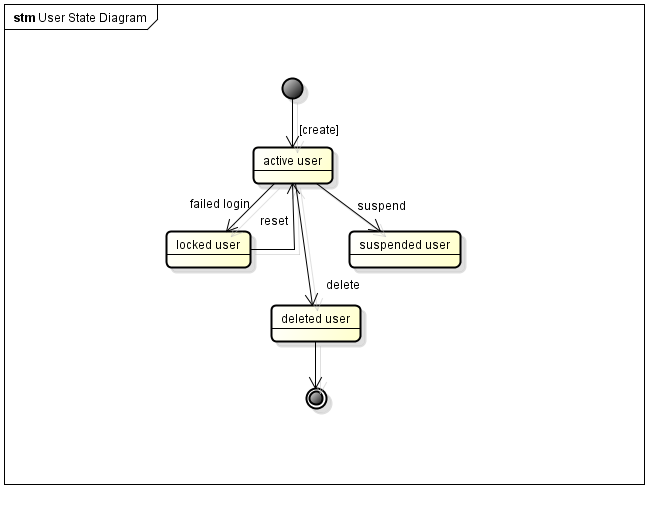


Figure User State Diagram

## 2.2 Functional Requirements

|  |  |
| --- | --- |
| **Requirement ID** | **Description** |
| R1.1 Maintain User |  |
| R1.1.1 | System shall login in user when provided valid username/password |
| R1.1.2 | System shall create valid user session when login |
| R1.1.3 | System shall invalidate session when logged out |
| R1.1.4 | System shall provide CRUD for a user |
| R1.2 Maintain Account |  |
| R1.2.1 | System shall provide CRUD for an account |
| R1.2.2 | System shall reflect the subscription for the account |
| R1.2.3 | System shall maintain message quota in account |
| R1. 3 Process Subscription |  |
| R1.3.1 | System shall provide CRUD for subscription |
| R1.3.2 | System shall provide payment status for subscription |
| R1.3.3 | System shall have knowledge of message quota allotted |
| R1.3.4 | System shall have knowledge of subscription expire and renewal |
| R1.3.5 | System shall support reoccurring subscription |
| R1.3.6 | System shall support subscription upgrade and downgrade |
| R1.3.7 | System shall default to freemium subscription |
| R1.4 Manage Payment |  |
| R1.4.1 | System shall authorize a payment |
| R1.4.2 | System shall settle a payment |
| R1.4.3 | System shall support credits and debits |
| R1.5 Process Report |  |
| R1.5.1 | System shall support daily report |
| R1.5.2 | System shall support a find by sender/recipients report |
| R1.5.3 | System shall support a find days back in report |
| R1.6MaintainCampaign |  |
| R1.6.1 | System shall provide CRUD for a campaign |
| R1.6.2 | System shall accept start and stop campaign date and time |
| R1.6.3 | System shall associate contacts with a campaign |
| R1.6.4 | System shall associate content with a campaign |
| R1.7MaintainContact |  |
| R1.7.1 | System shall provide CRUD for a contact |
| R1.7.2 | System shall not enforce unique contacts |
| R1.7.3 | System shall treat contacts as phone numbers. |
| R1.8Maintain Content |  |
| R1.8.1 | System shall provide CRUD for content |
| R1.8.2 | System shall support multiple content per compaign |
| R1.8.3 | System shall support content of text or multimedia |
| R1.9Process Campaign |  |
| R1.9.1 | System shall support start and stop of campaign on a timer |
| R1.9.2 | System shall support immediate processing of campaign |
| R1.9.3 | System shall trigger mobile messages on start of campagin |
| R2.0 Process Message |  |
| R2.0.1 | System shall send message to mobile service provider |
| R2.0.2 | System shall report on the success of failure of message |
| R2.0.3 | System shall contain the content of the campaign |
|  |  |

## 2.3 Nonfunctional Requirements

### 2.3.1 Quality of Service

|  |  |
| --- | --- |
| **Requirement ID** | **Description** |
| R2.1 Performance |  |
| R2.1.1 | System shall response in a timely manner |
| R2.2 Capacity |  |
| R2.2.1 | System shall have the capacity of 1 million users |
| R2.3 Usability |  |
| R2.3.1 | System shall provide a user friendly UI |
| R2.3.2 | System shall provide meaningful notifications |
| R2.4 Reliability |  |
| R2.4.1 | System shall persist all changes |
| R2.4.2 | System shall recover to a reliable state after shutdown or crash |
| R2.5 Availability |  |
| R2.5.1 | System shall be up 24x7 |
| R2.6 Supportability |  |
| R2.6.1 | System shall provide proper logging |
| R2.6.2 | System shall provide proper monitoring |

### 2.3.2 Constraints

|  |  |
| --- | --- |
| **Requirement ID** | **Description** |
| R3.1 Implementation |  |
| R3.1.1 | System shall be implemented in Java |
| R3.1.2 | System shall run on Linux |
| R3.2 Interface |  |
| R3.2.1 | System shall provide external soap interfaces if interfaces are required |
| R3.3 Operations |  |
| R3.3.1 | System shall follow Java coding guidelines |
| R3.4 Packaging |  |
| R3.4.1 | System shall follow Java coding guidelines |
| R3.5 Compliance to Standards |  |
| R3.5.1 | System shall follow Java coding guidelines |
| R3.6 Security |  |
| R3.6.1 | System shall use SSL over HTTP |
| R3.6.2 | System shall encrypt passwords |
| R3.6.3 | System shall provide encrypted credit card info |
| R3.7 Legal |  |
| R3.7.1 | System shall be PCI compliant |
| R3.7.2 | System shall follow privacy laws |
|  |  |

## 2.4 Use Case Modeling

### 2.4.1 Use Case Diagram

Figure 27 shows the use cases of the system. It consists of four actors and ten use cases.

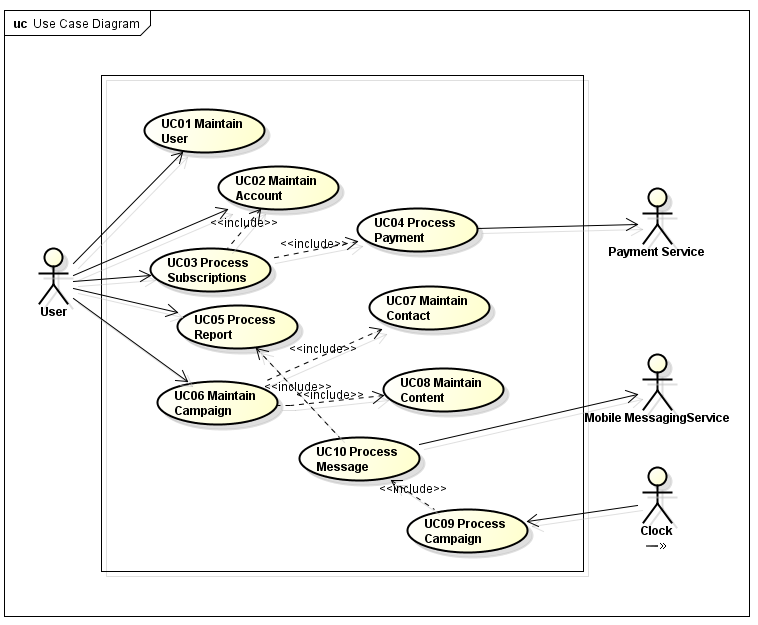


Figure 27 Use Case Diagram

### 2.4.2 Use Case Description

#### 2.4.2.1 UC01 Maintain User

|  |  |  |
| --- | --- | --- |
| ID | UC01 | |
| Name | Maintain User | |
| Type | Base | |
| Actors | Primary: User | |
| Description | User performs User CRUD | |
| Precondition | None | |
| Postcondition | User information is updated and persisted. | |
| Main Flow | User   1. User issues CRUD request | System   1. System validates request 2. System performs create, retrieve, update, or delete User. |
| Alternate Flow | a. User data invalid  1. Error response is return | |
| Special Requirement | None | |

#### 2.4.2.2 UC02 Maintain Account

|  |  |  |
| --- | --- | --- |
| ID | UC03 | |
| Name | Maintain Account | |
| Type | Base | |
| Actors | Primary: User  Secondary: Subscription Subsystem | |
| Description | User performs Account CRUD or System performs CRUD due to subscription processing. | |
| Precondition | None | |
| Postcondition | Account information is updated and persisted. | |
| Main Flow | User   1. User issues CRUD request | System   1. System validates request. 2. System performs create, retrieve, update, or delete Account. |
| Alternate Flow | 1. Subscription System issues CRUD request.   1. System performs create, retrieve, update, or delete Account.  b. User does not have permission  1. Error response is return  c. Account data invalid  1. Error response is return | |
| Special Requirement | None | |

#### 2.4.2.3 UC03 Process Subscription

|  |  |  |
| --- | --- | --- |
| ID | UC03 | |
| Name | Process Subscription | |
| Type | Base | |
| Actors | Primary: User | |
| Description | User changes subscription | |
| Precondition | None | |
| Postcondition | Subscription and Account information is updated and persisted. | |
| Main Flow | User   1. User issues change subscription request | System   1. System validates request 2. System debits payment. UC04 Process Payment |
| Alternate Flow | a. Subscription data invalid  1. Error response is return | |
| Special Requirement | None | |

#### 2.4.2.4 UC04 Process Payment

|  |  |  |
| --- | --- | --- |
| ID | UC04 | |
| Name | Process Payment | |
| Type | Reference | |
| Actors | Primary: Campaign Subsystem | |
| Description | System credit or debit account due to ansubscription change. | |
| Precondition | UC03 Process Subscription | |
| Postcondition | Money is debit or credit from account. | |
| Main Flow | Subscription Subsystem   1. System issues payment request | System   1. System checks request action.    1. If action=debit, authorize is sent to the payment service.       1. Settle is sent to payment service if authorize is successful.    2. If action=credit, credit is sent to payment service. |
| Alternate Flow | a. Payment data invalid  1. Error response is return  b. Auth fails or insufficient fund  1. Subscription is not changed. | |
| Special Requirement | None | |

#### 2.4.2.5 UC05 Process Report

|  |  |  |
| --- | --- | --- |
| ID | UC05 | |
| Name | Process Report | |
| Type | Base | |
| Actors | Primary: User | |
| Description | User runs report | |
| Precondition | None | |
| Postcondition | Report data is return to user | |
| Main Flow | User   1. User issues report request | System   1. System validates request 2. System process report.    1. If type is yearly, aggregate yearly data.    2. If type is monthly, aggregate monthly data.    3. If type is daily, aggregate daily data.    4. If type is hourly, aggregate hourly data.    5. If type is source/destination, get all messages for source/destination. |
| Alternate Flow | a. Report data invalid  1. Error response is return | |
| Special Requirement | None | |

#### 2.4.2.6 UC06 Maintain Campaign

|  |  |  |
| --- | --- | --- |
| ID | UC06 | |
| Name | Maintain Campaign | |
| Type | Base | |
| Actors | Primary: User | |
| Description | User performs Campaign CRUD | |
| Precondition | None | |
| Postcondition | Campaign information is updated and persisted. | |
| Main Flow | User   1. User issues CRUD request | System   1. System validates request 2. System performs create, retrieve, update, or delete User. |
| Alternate Flow | a. Campaign data invalid  1. Error response is return | |
| Special Requirement | None | |

#### 2.4.2.7 UC07 Maintain Contact

|  |  |  |
| --- | --- | --- |
| ID | UC07 | |
| Name | Maintain Contact | |
| Type | Base | |
| Actors | Primary: User | |
| Description | User performs Contact CRUD | |
| Precondition | None | |
| Postcondition | Contact information is updated and persisted. | |
| Main Flow | User   1. User issues CRUD request | System   1. System validates request 2. System performs create, retrieve, update, or delete User. |
| Alternate Flow | a. Contact data invalid  1. Error response is return | |
| Special Requirement | None | |

#### 2.4.2.8 UC08 Maintain Content

|  |  |  |
| --- | --- | --- |
| ID | UC08 | |
| Name | Maintain Content | |
| Type | Base | |
| Actors | Primary: User | |
| Description | User performs Content CRUD | |
| Precondition | None | |
| Postcondition | Content information is updated and persisted. | |
| Main Flow | User   1. User issues CRUD request | System   1. System validates request 2. System performs create, retrieve, update, or delete User. |
| Alternate Flow | a. Content data invalid  1. Error response is return | |
| Special Requirement | None | |

#### 2.4.2.9 UC09 Process Campaign

|  |  |  |
| --- | --- | --- |
| ID | UC09 | |
| Name | Process Campaign | |
| Type | Base | |
| Actors | Primary: User  Secondary: Clock | |
| Description | User triggers start of campaign | |
| Precondition | UC06 Maintain Campaign | |
| Postcondition | Messages are sent to be processed. | |
| Main Flow | User   1. User issues process campaign request | System   1. System validates request 2. System sents messages to contacts. UC10 Process Message. |
| Alternate Flow | 1. Timer detects the start of campaign   1. System sents messages to contacts. UC10 Process Message.  b. Campaign data invalid  1. Error response is return | |
| Special Requirement | None | |

#### 2.4.2.10 UC10 Process Message

|  |  |  |
| --- | --- | --- |
| ID | UC10 | |
| Name | Process Message | |
| Type | Reference | |
| Actors | Primary: Campaign System | |
| Description | Campaign System request Messages to be sent. | |
| Precondition | UC09 Process Campaign | |
| Postcondition | Messages are sent to be mobile service provider | |
| Main Flow | User   1. User issues process message request | System   1. System validates request 2. System sents messages to mobile service provider. 3. Message record is save in the Report System. UC05 Process Report. |
| Alternate Flow | a. message data invalid  1. Error response is return | |
| Special Requirement | None | |

### 2.4.3 Sequence Diagram

#### 2.4.3.1 Maintain User

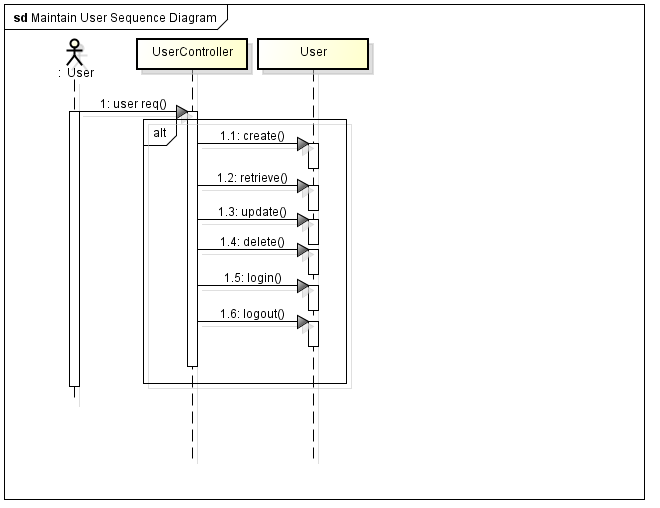


Figure Maintain User Sequence Diagram

#### 2.4.3.2 Maintain Account

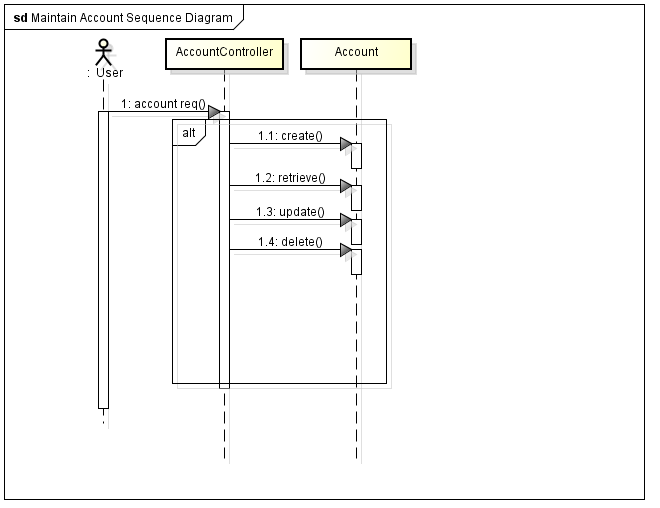


Figure Maintain Account Sequence Diagram

#### 2.4.3.3 Process Subscription

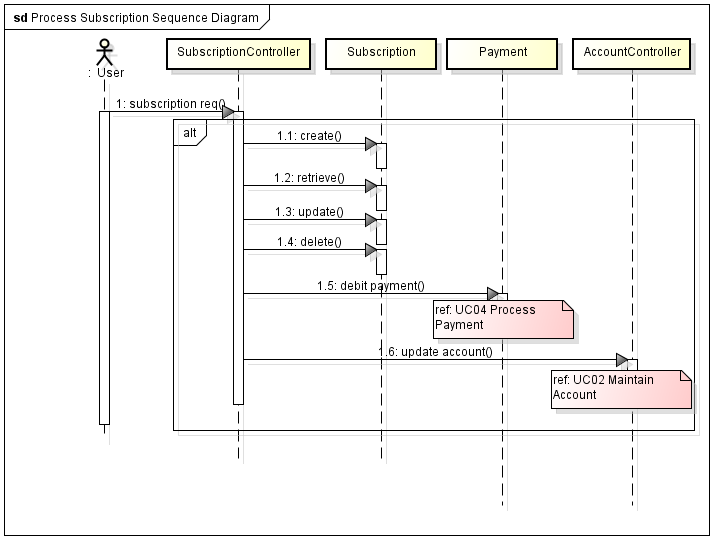


Figure Process Subscription Sequence Diagram

### 2.4.3.4 Process Payment

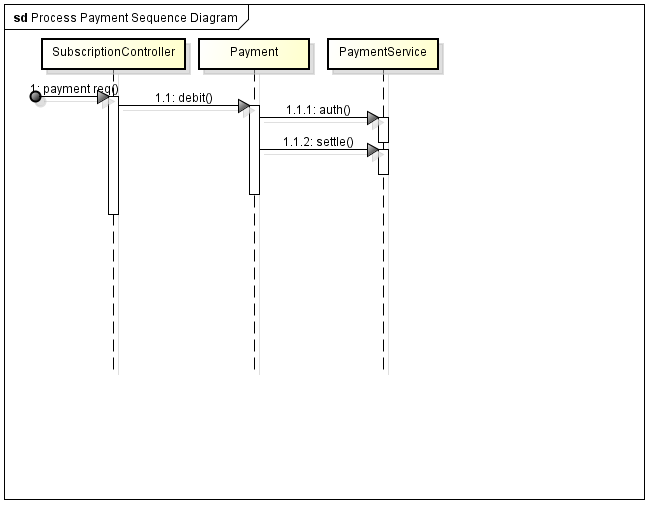


Figure Process Payment

#### 2.4.3.5 Process Report

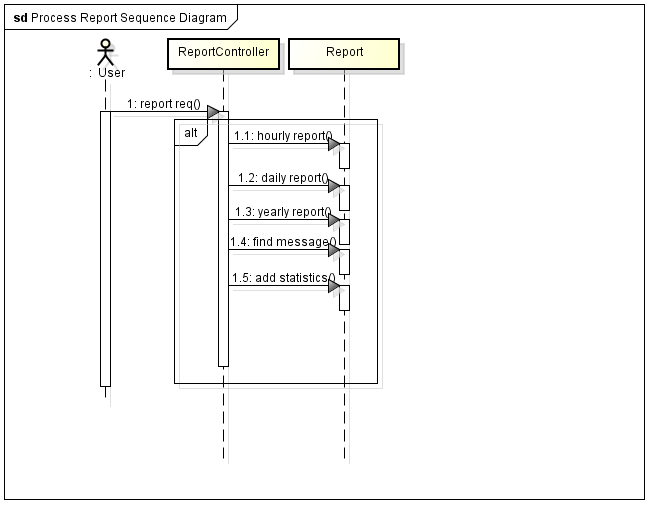


Figure Process Report Sequence Diagram

#### 2.4.3.6 Maintain Campaign

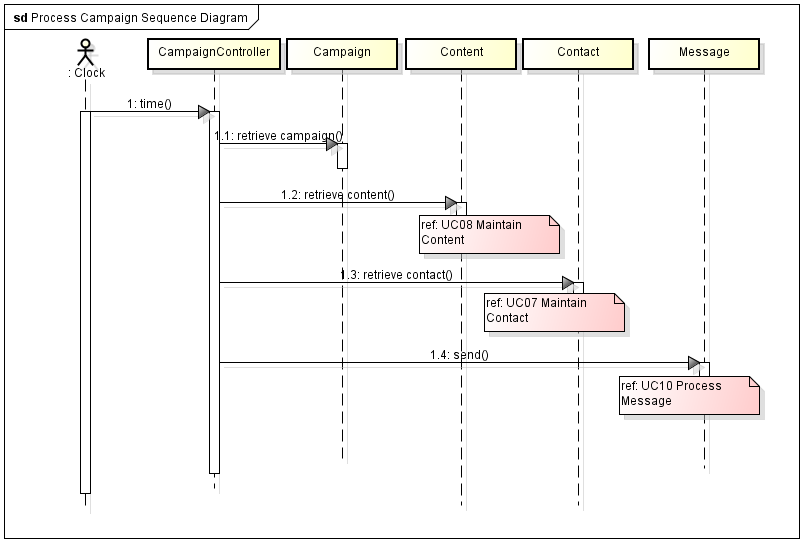


Figure Maintain Campaign Sequence Diagram

#### 2.4.3.7 Maintain Contact

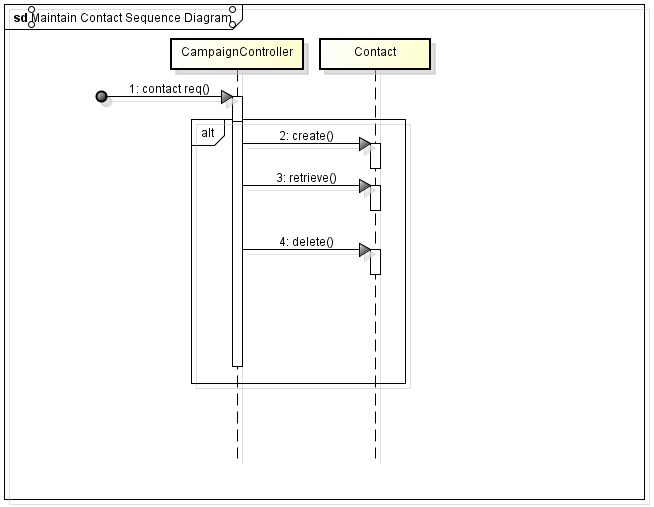


Figure Maintain Contact Sequence Diagram

#### 2.4.3.8 Maintain Content

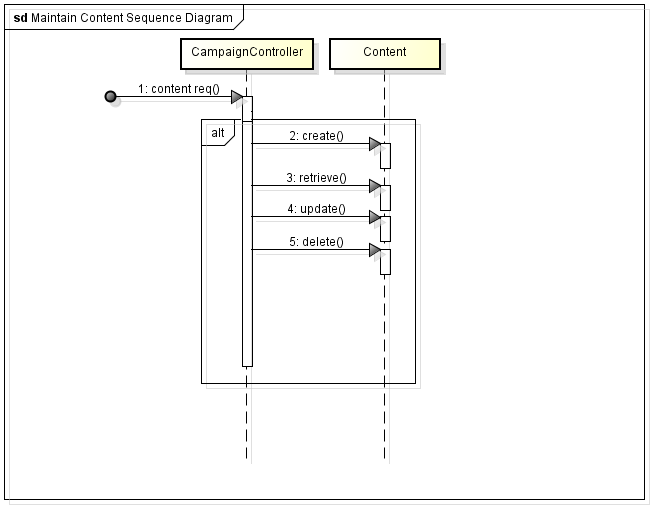


Figure Maintain Content Sequence

#### 2.4.3.9 Process Campaign

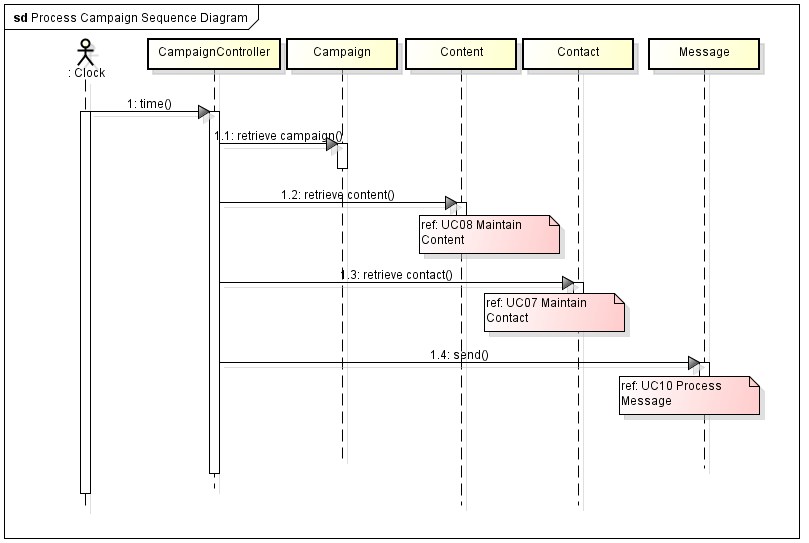


Figure Process Campaign Sequence Diagram

#### 2.4.3.10 Process Message

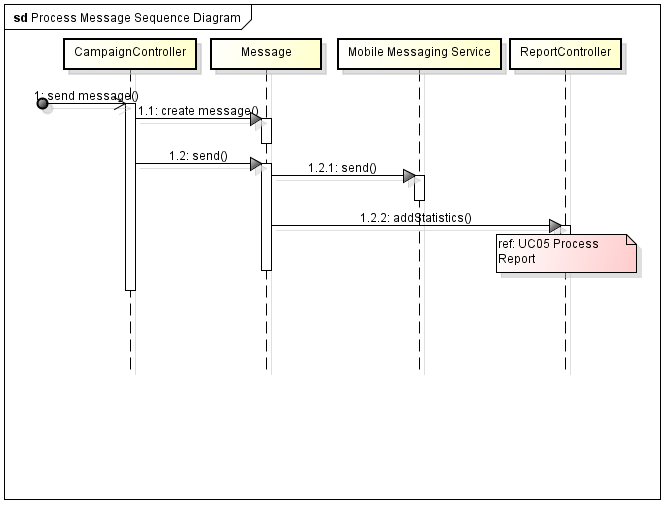


Figure Process Message Sequence Diagram

### 2.4.4 Use Case Interaction Overview Diagram

Figure 38 shows the interaction of the use cases in the system.

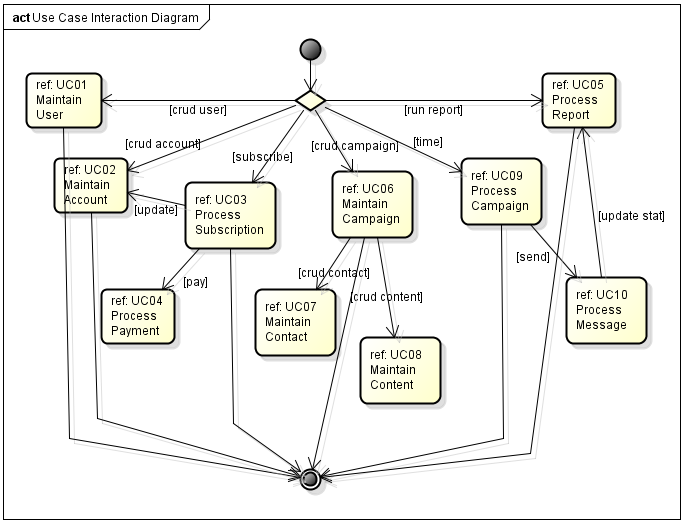


Figure Use Case Interaction Diagram

### 2.4.5 Requirement Traceability Matrix

Figure 39 shows that requirements are being met by the following use cases.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Req ID | UC 01 | UC 02 | UC 03 | UC 04 | UC 05 | UC 06 | UC 07 | UC 08 | UC 09 | UC 10 |
| R1.1.1 | X |  |  |  |  |  |  |  |  |  |
| R1.1.2 | X |  |  |  |  |  |  |  |  |  |
| R1.1.3 | X |  |  |  |  |  |  |  |  |  |
| R1.1.4 | X |  |  |  |  |  |  |  |  |  |
| R1.2.1 |  | X |  |  |  |  |  |  |  |  |
| R1.2.2 |  | X | X |  |  |  |  |  |  |  |
| R1.2.3 |  | X |  |  |  |  |  |  |  | X |
| R1.3.1 |  |  | X |  |  |  |  |  |  |  |
| R1.3.2 |  |  | X | X |  |  |  |  |  |  |
| R1.3.3 |  |  | X |  |  |  |  |  |  |  |
| R1.3.4 |  |  | X |  |  |  |  |  |  |  |
| R1.3.5 |  |  | X | X |  |  |  |  |  |  |
| R1.3.6 |  |  | X |  |  |  |  |  |  |  |
| R1.3.7 |  |  | X |  |  |  |  |  |  |  |
| R1.4.1 |  |  |  | X |  |  |  |  |  |  |
| R1.4.2 |  |  |  | X |  |  |  |  |  |  |
| R1.4.3 |  |  |  | X |  |  |  |  |  |  |
| R1.4.4 |  |  |  | X |  |  |  |  |  |  |
| R1.4.5 |  |  |  | X |  |  |  |  |  |  |
| R1.5.1 |  |  |  |  | X |  |  |  |  |  |
| R1.5.2 |  |  |  |  | X |  |  |  |  |  |
| R1.5.3 |  |  |  |  | X |  |  |  |  |  |
| R1.5.4 |  |  |  |  | X |  |  |  |  |  |
| R1.5.5 |  |  |  |  | X |  |  |  |  |  |
| R1.5.6 |  |  |  |  | X |  |  |  |  |  |
| R1.6.1 |  |  |  |  |  | X |  |  |  |  |
| R1.6.2 |  |  |  |  |  | X |  |  |  |  |
| R1.6.3 |  |  |  |  |  | X |  |  |  |  |
| R1.6.4 |  |  |  |  |  | X |  |  |  |  |
| R1.7.1 |  |  |  |  |  |  | X |  |  |  |
| R1.7.2 |  |  |  |  |  |  | X |  |  |  |
| R.1.7.3 |  |  |  |  |  |  | X |  |  |  |
| R1.8.1 |  |  |  |  |  |  |  | X |  |  |
| R1.8.2 |  |  |  |  |  |  |  | X |  |  |
| R1.8.3 |  |  |  |  |  |  |  | X |  |  |
| R1.9.1 |  |  |  |  |  |  |  |  | X |  |
| R1.9.2 |  |  |  |  |  |  |  |  | X |  |
| R1.9.3 |  |  |  |  |  |  |  |  | X |  |
| R2.0.1 |  |  |  |  |  |  |  |  |  | X |
| R2.0.2 |  |  |  |  |  |  |  |  |  | X |
| R2.0.3 |  |  |  |  |  |  |  |  |  | X |
| Iteration | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| Workflow | RADIT | RADIT | RADIT | RADIT | RADIT | RADIT | RADIT | RADIT | RADIT | RADIT |

Figure 39 Requirement Traceability Matrix

## 2.5 Context and Interface Requirements

### 2.5.1 Context Requirements

|  |  |
| --- | --- |
| **Requirement ID** | **Description** |
|  |  |
| **CR1** | **Development Requirements** |
| CR1.1 | Software shall be written in the Java language |
| CR1.2 | Software shall run on JDK 1.6 or higher |
| CR1.3 | System shall run on Windows, UNIX, Linux |
| CR1.4 | IDE shall be Eclipse |
| CR1.5 | Application shall be Tomcat 6.0 or higher |
| CR1.6 | Data store shall be RDBMS and/or noSQL |
|  |  |
| **CR2** | **Testing Requirements** |
| CR2.1 | Test cases shall cover functional testing |
| CR2.2 | Test cases shall cover integration testing |
| CR2.3 | Test cases shall cover black box testing |
| CR2.4 | Test cases shall cover white box testing |
| CR2.5 | Test cases shall be scripted where possible |
|  |  |

### 2.5.2 Interface Requirements

|  |  |
| --- | --- |
| **Requirement ID** | **Description** |
|  |  |
| **IR1** | **Web Interface Requirements** |
| IR1.1 | Graphical Interface shall be Web based |
| IR1.2 | Web GUI shall be in HTML, Javascript, css |
| IR1.3 | Web style shall be Web 2.0 |
| IR1.4 | Web interface shall be user friendly |
| IR1.5 | Web interface shall provide form validation |
| IR1.6 | Web interface shall be secure |
|  |  |
| **IR2** | **API Interface Requirements** |
| CR2.1 | API shall be HTTP based |
| CR2.2 | API transport shall be REST |
|  |  |

## 2.6 Resource Requirements

### 2.6.1 Hardware Requirements

|  |  |
| --- | --- |
| **Requirement ID** | **Description** |
|  |  |
| **HR1** | **Hardware Requirements** |
| HR1.1 | Minimum 1 primary and 1 failover application server |
| HR1.2 | Application server shall support linux |
| HR1.3 | Minimum 1 RDBMS |
| HR1.4 | Hardware can be physical or virtual |
| HR1.5 | Servers must be network enabled |
|  |  |

### 2.6.2 Software Requirements

|  |  |
| --- | --- |
| **Requirement ID** | **Description** |
|  |  |
| **SR1** | **Software Requirements** |
| SR1.1 | Linux 2.6 or above |
| SR1.2 | JDK 1.6 or above |
| SR1.3 | Tomcat 6.0 or above |
| SR1.4 | Eclipse |
| SR1.5 | RDBMS/noSQL |
|  |  |
|  |  |

### 2.6.3 Technical Skills Requirements

|  |  |
| --- | --- |
| **Requirement ID** | **Description** |
|  |  |
| **TR1** | **Technical Skills Requirements** |
| TR1.1 | Java Programming |
| TR1.2 | Web Programming |
| TR1.3 | Web Design |
| TR1.4 | Linux Administration |
| TR1.5 | Database Development |
|  |  |

# Chapter 3 System Design

## 3.1 Architecture Design

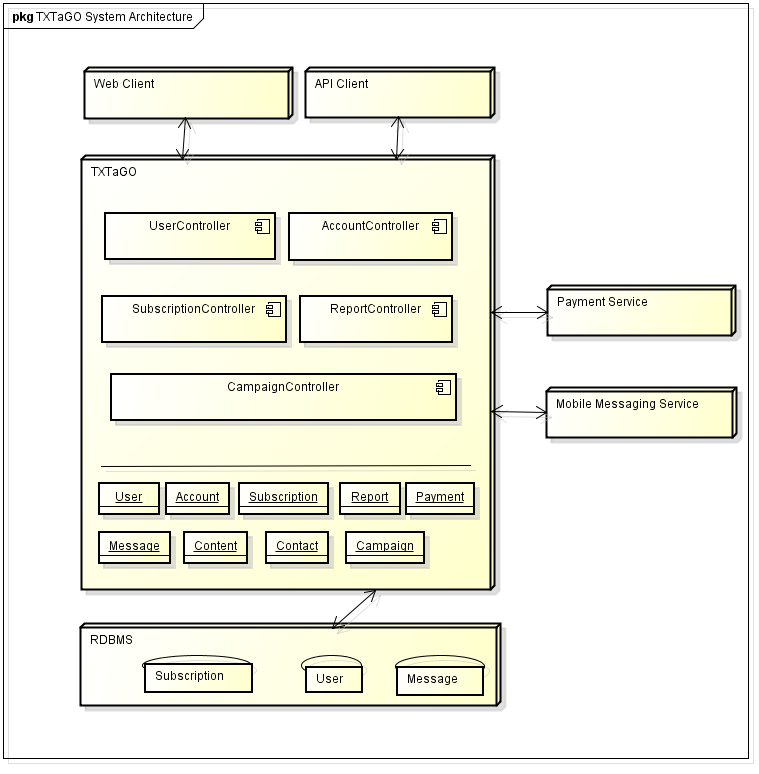


Figure System Architecture Diagram

The above figure (Figure 40) shows the system architecture of TXTaGO. TXTaGO is a 3-tier architecture. The Client, the Application Server and DataStore make up the 3-tiers. TXTaGO also communicates with external services for Payment and Mobile Messaging Service. All the subsystems are exposed through their controllers. The domain objects are used as data transfer objects to communicate between the client, TxtagoService, and the datastore.

## 3.2 System Logic and Partition Design

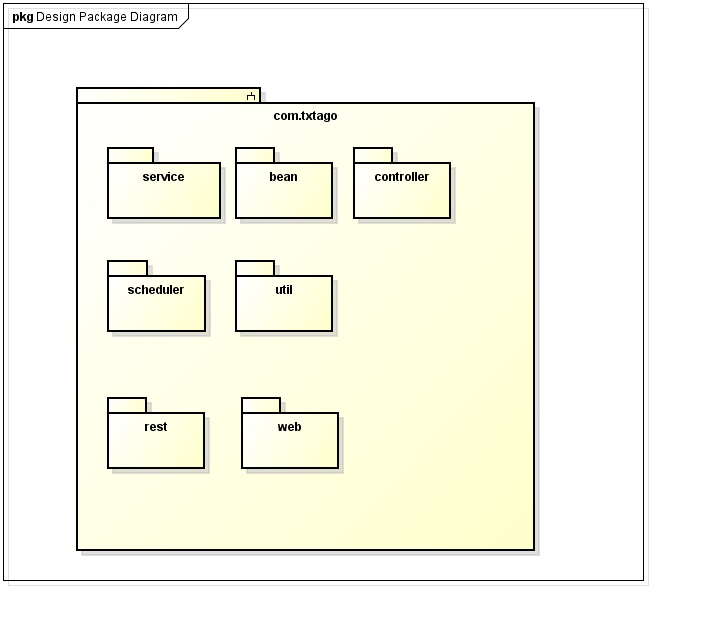


Figure Design Package Diagram

The figure above (Figure 41) shows the seven packages in the TXTaGO system. The core package contains the common classes used in the system, along with the main executing classes. The client package contains two packages – Web and Rest. The Web package contains the MVC classes for the Web graphical user interface. The API package contains the REST interface for programmatic access to the system. The remaining five packages provide the business logic for the system.

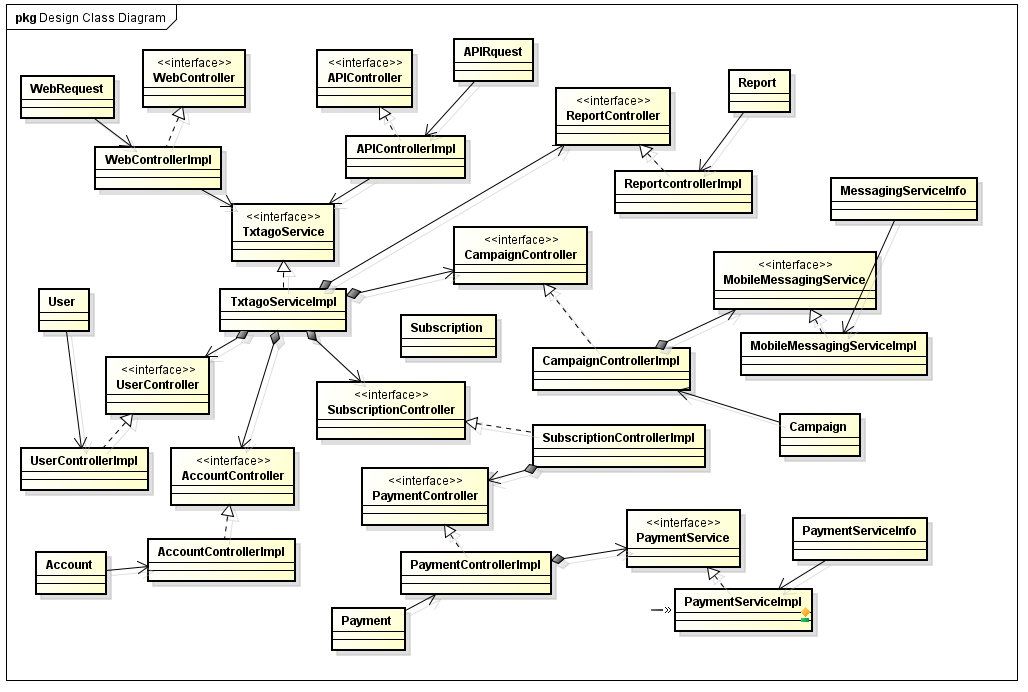
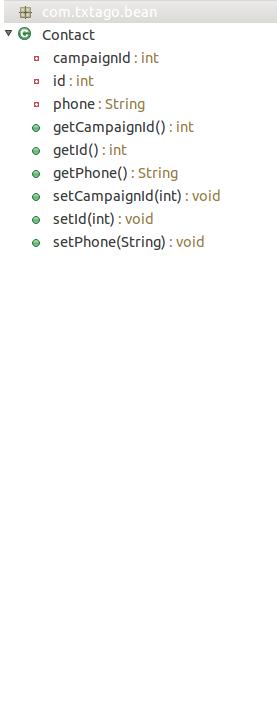
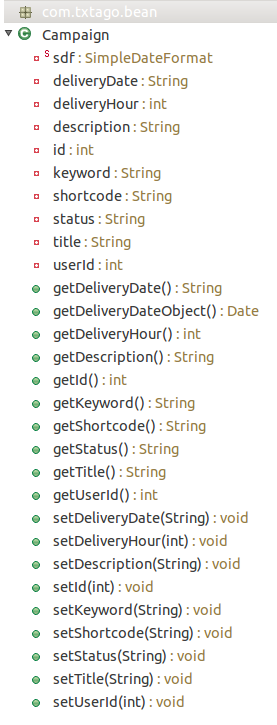
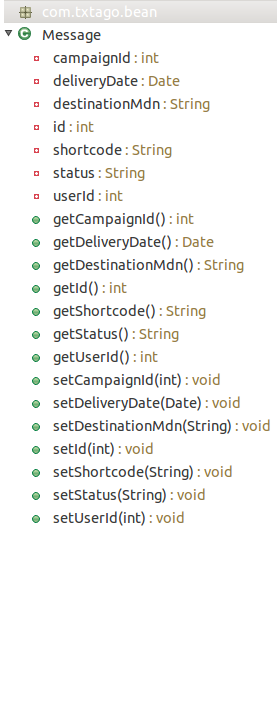
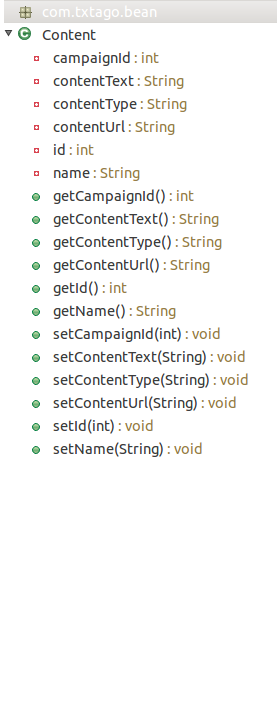
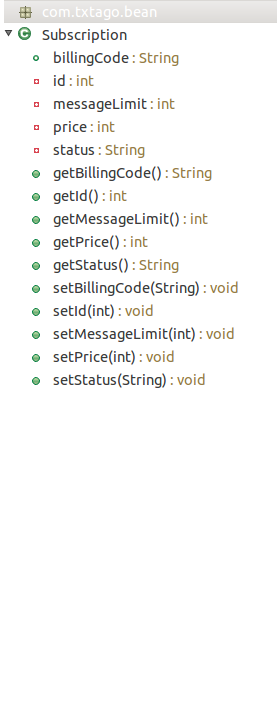
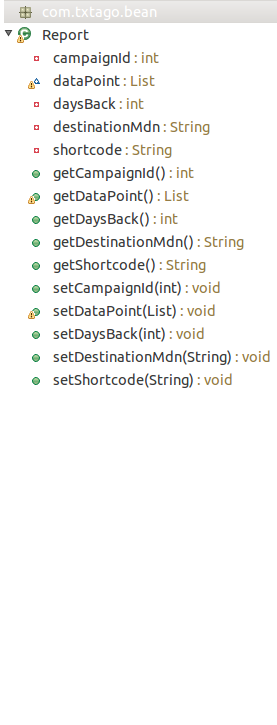


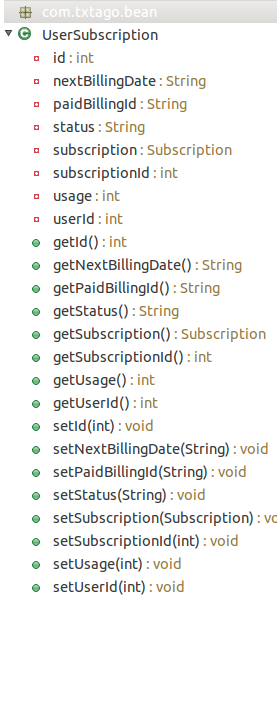
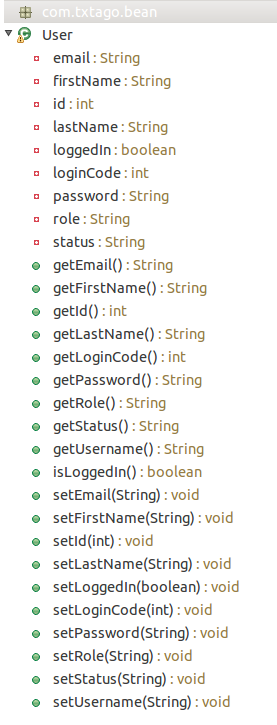
Figure Design Class Diagram

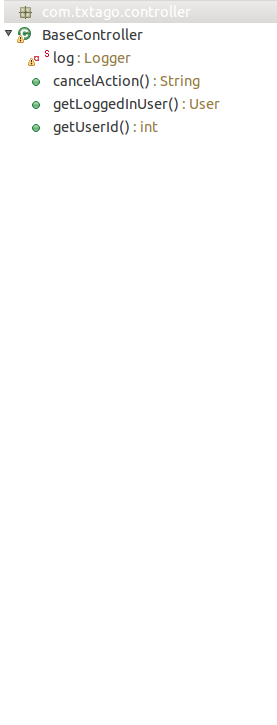
The above figure (Figure 42) shows the preliminary class design of the TXTaGO system and the relationship between the classes. All subsystems are accessed through a controller class. All controller classes are abstracted through an interface.

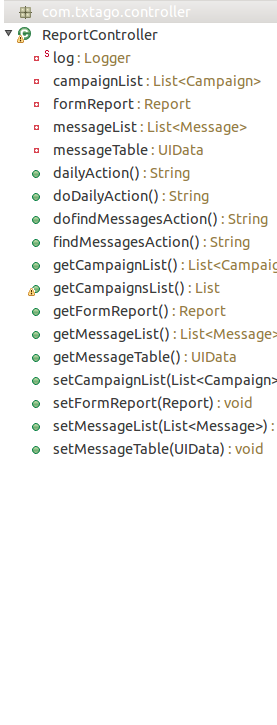
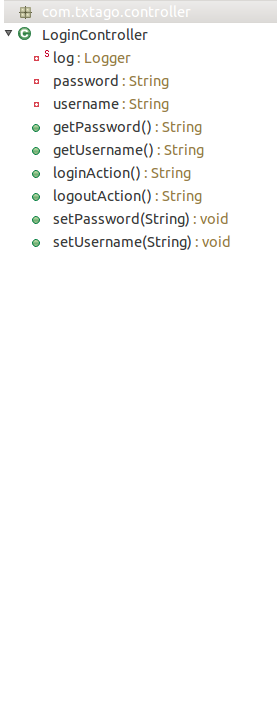


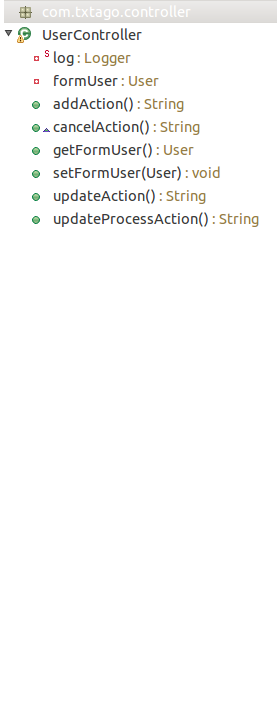


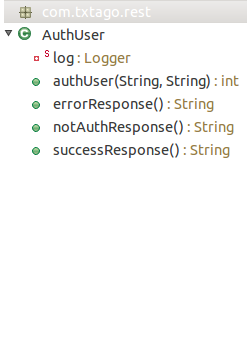


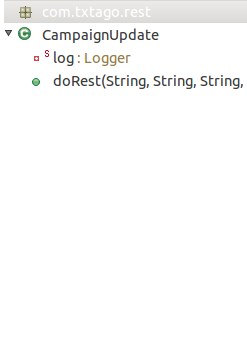
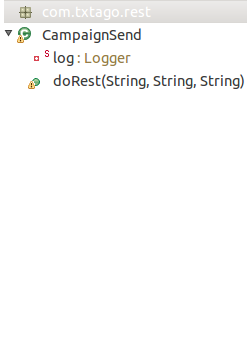


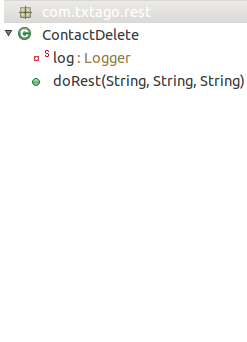
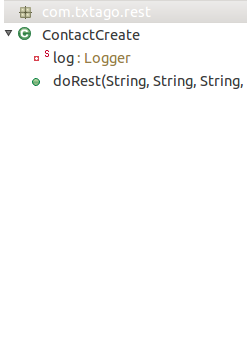


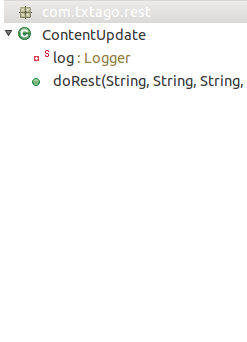
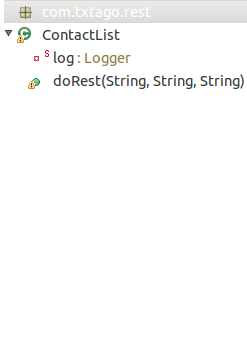


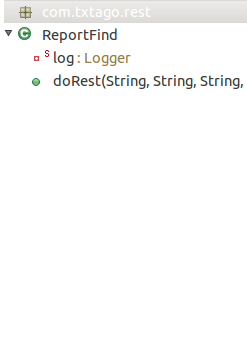
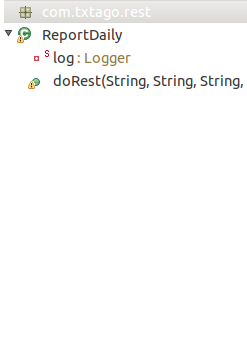


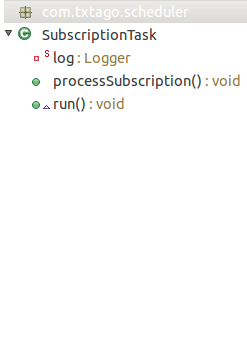
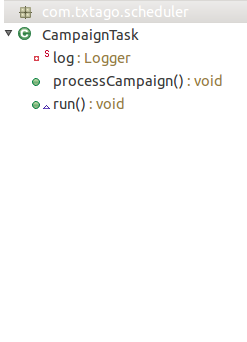


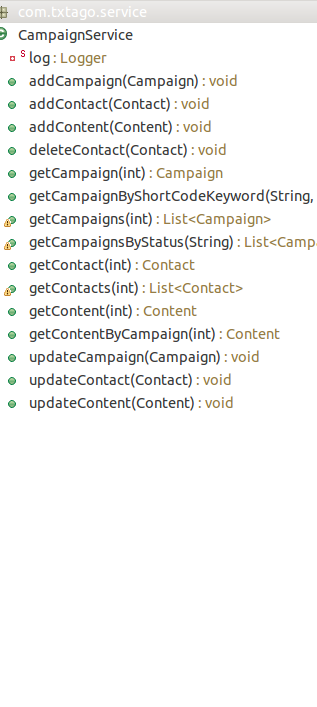
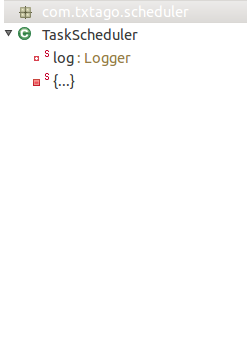


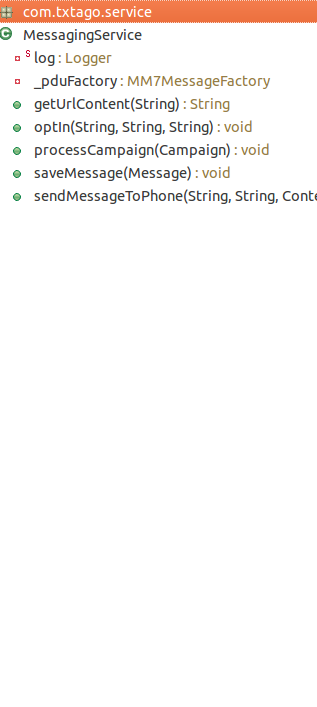




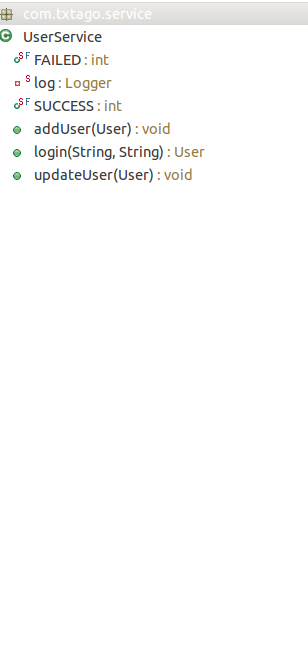


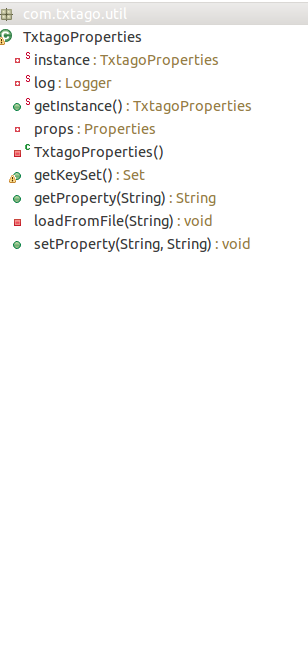
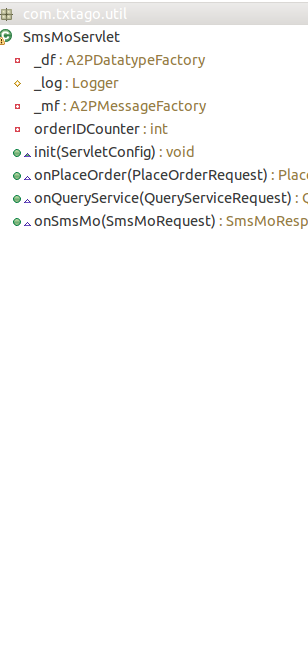


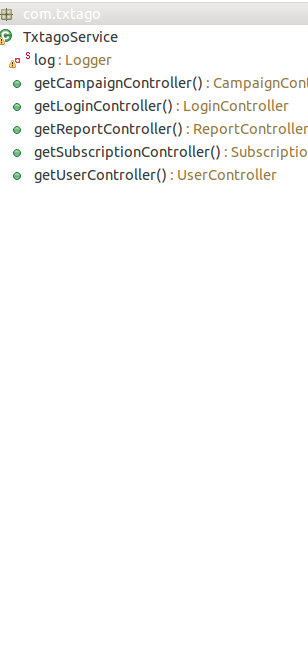












## 3.3 System Interface and Connectivity Design

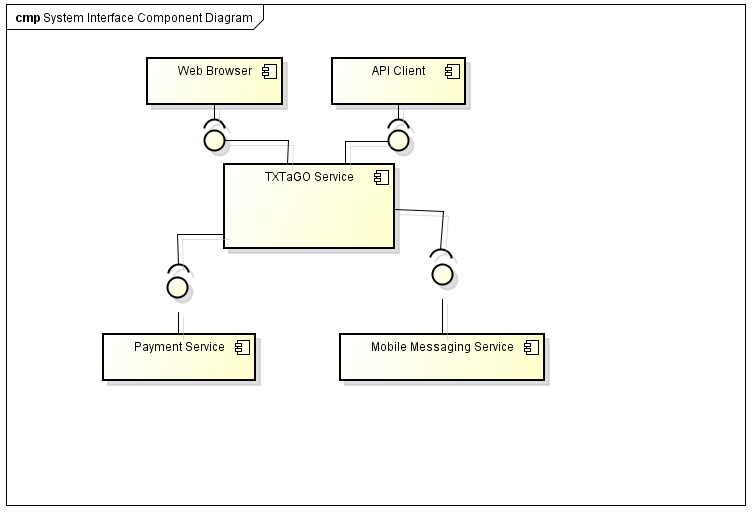


Figure System Interface Diagram

The figure above (Figure 43) shows how the TXTaGO system interfaces with external entities. Communication between the user and TXTaGO can come either from the Web GUI or the REST API. The TXTaGO system also relies on an external service for payment processing and an external mobile messaging service for the processing of mobile messages. The mobile messages are delivered to the physical handset by the mobile service provider.

## 3.3 System Data and Database Design

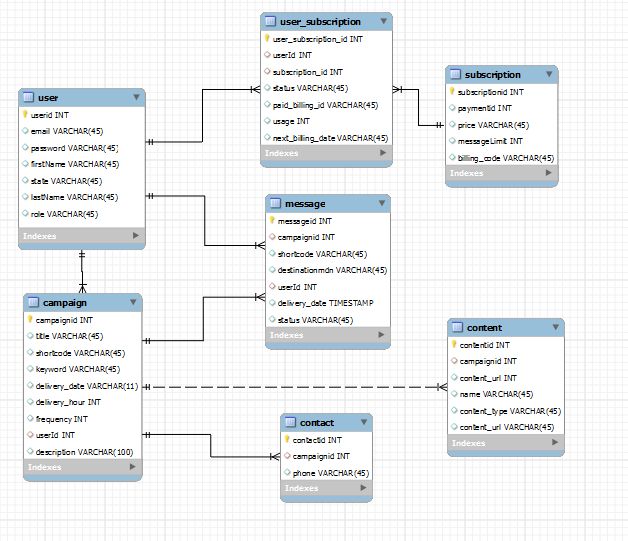
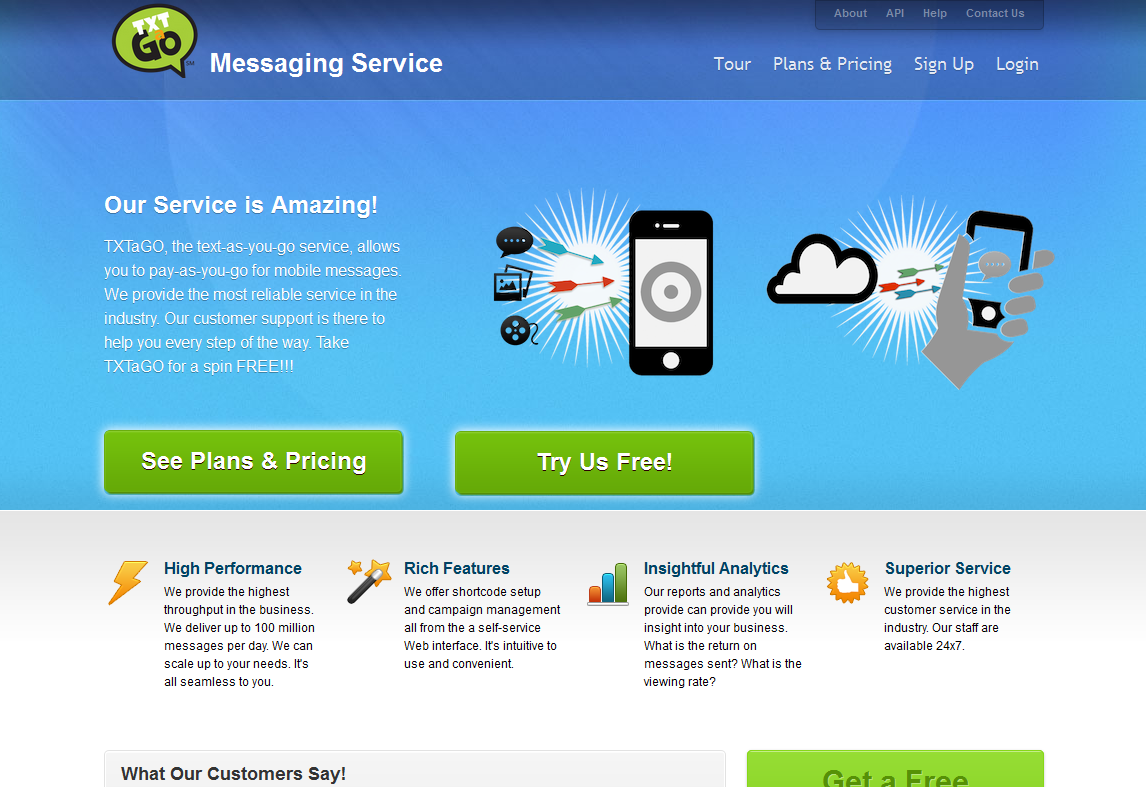


Figure Database ER Diagram

The above figure (Figure 44) shows the preliminary tables and columns for the TXTaGo system.

## 3.4 User Interface Design



The screens below show the preliminary design of the Web graphical user interface. The screens are only a subset. Not all screens are present. Figure 45 shows the *login page*. Figure 46 shows the *add user* page. Figure 46 shows the *report* page. Figure 47 shows the *add campaign* page.

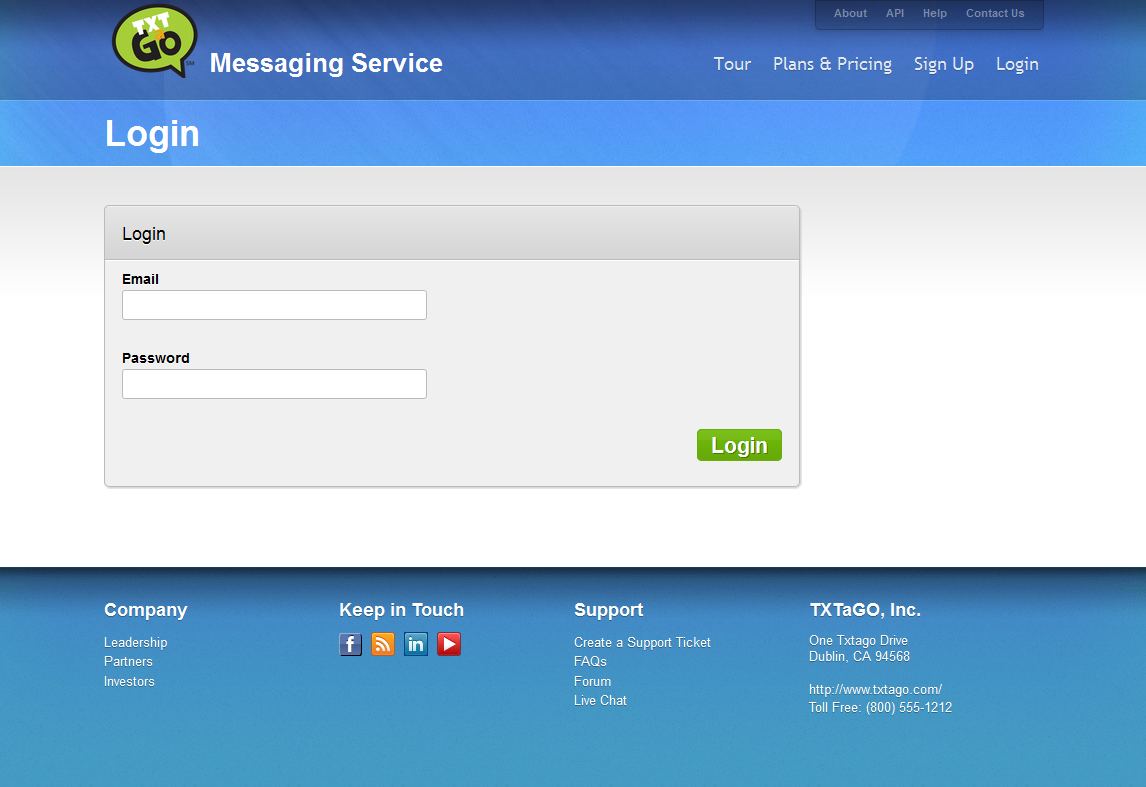


Figure Login Page

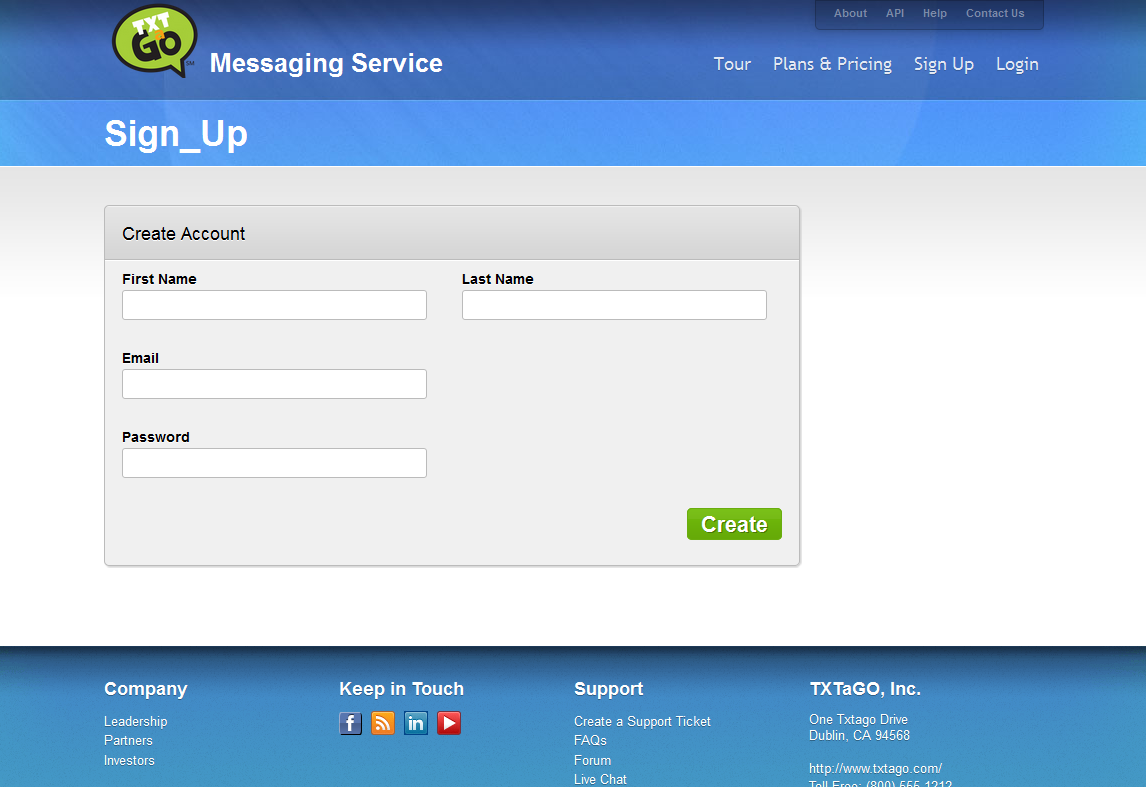


Figure Add User Page

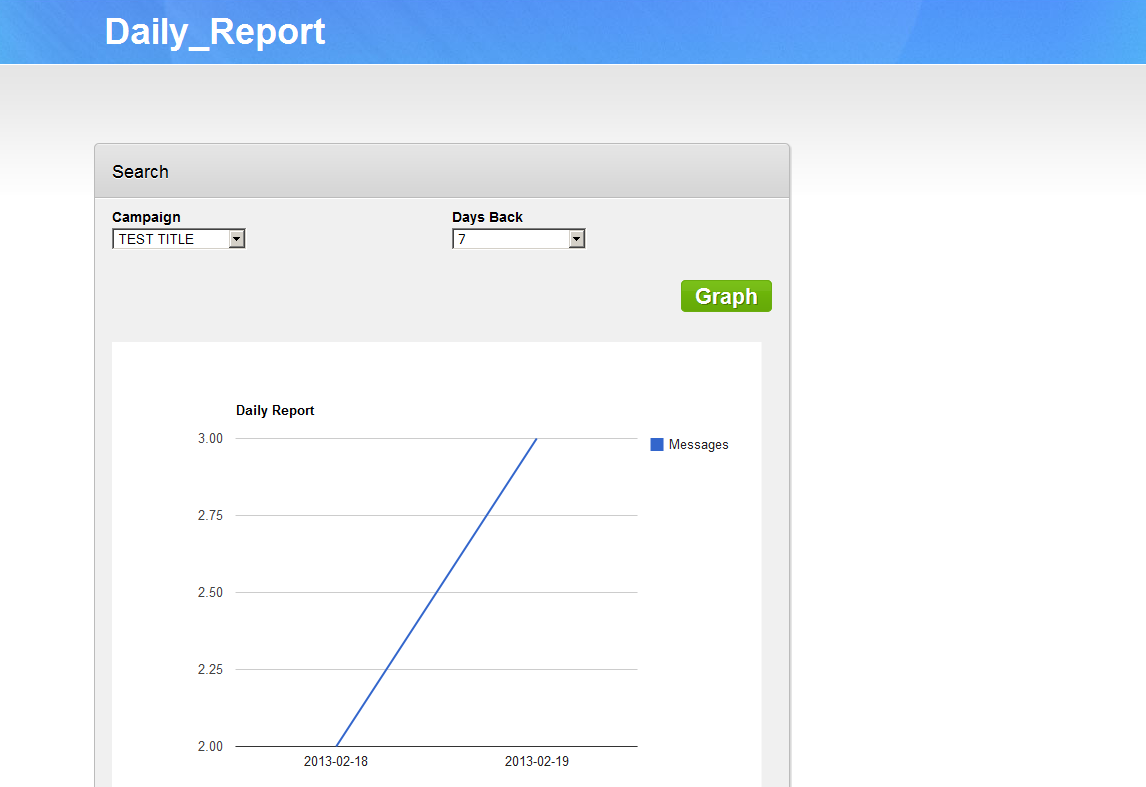


Figure Report Page

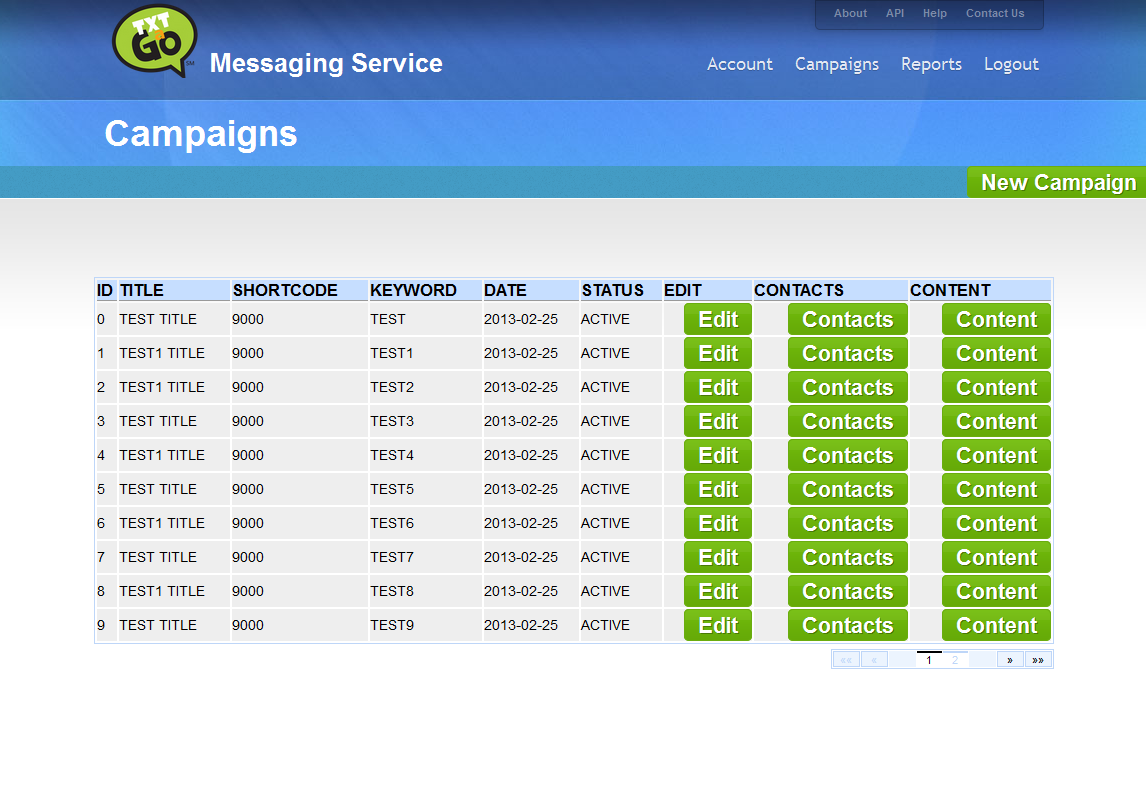
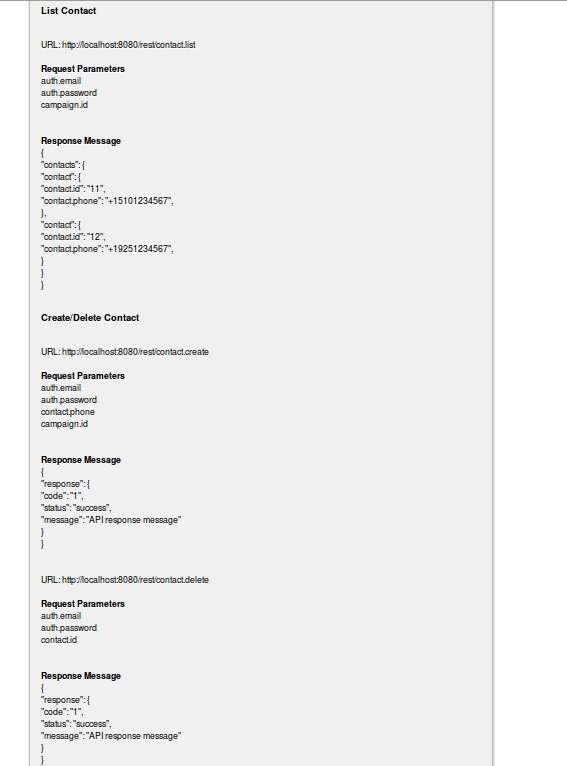


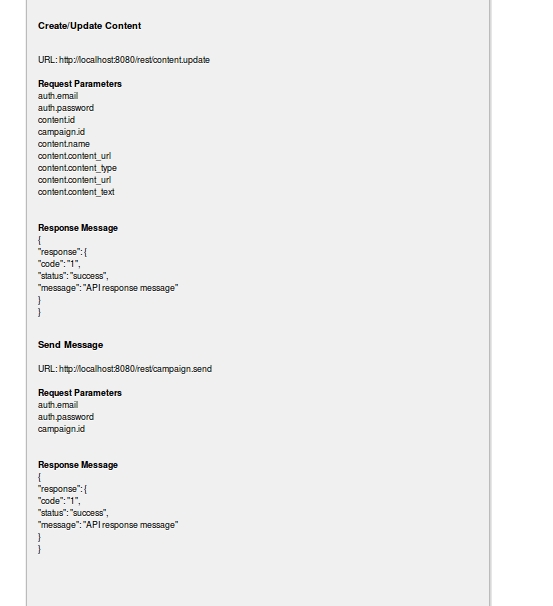
Figure Add Campaign Page

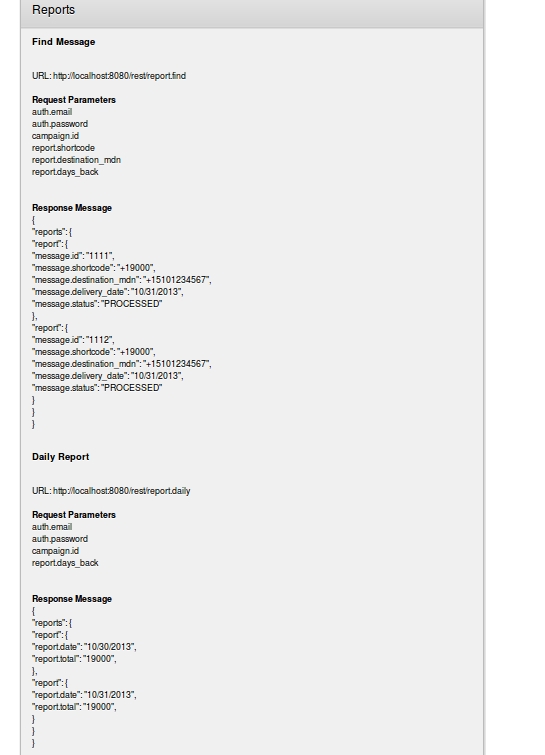
### 3.4.1 REST API Interface

The API descriptions below outline the REST API to TXTaGO. Most of the functionality mirrors the graphical user interface.









## 3.5 Design Trade-offs, Problems and Solutions

We decided to use organized the system packages by layers rather than by subsystems. Most web applications generally are structured by layers – web, business logic, data access. We tried to follow this approach in our design. We also decided to use a payment process service to handle our subscription processing and credit management. Using a payment process service would ensure PCI security compliance. We also use another service for mobile message delivery. This is the only way to can gain access to the mobile operators.

# Chapter 4 System Implementation

## 4.1 Implementation Summary



Figure Component Diagram

Figure 49 shows the components in the TXTaGo System. The implementation of the system will involve developing the TXTaGo services and integrating with the Payment and Mobile Service Providers.

The presentation layer is provided by the Web and Rest packages. Web will present a graphical user interface to the user and the Rest will provide a programmable interface. These two interfaces should be equivalent in features.

The business layer is represented by the Controller and Service packages. All these packages will communicate with and persist to the Data Store. The Subscription package will integrate with an external Payment Service for payment process. The Campaign package will integrate with Mobile Messaging Service for SMS and MMS delivery.

## 4.2 Develop or Adopt Decision

The general principle used to determine if TXTaGo should develop or adopt the software is determined by if the component is core or non-core to the business. If the component is our core to the business, we develop. If it’s infrastructure, server or non-core, we adopt.

Servers like application server, database server, web server, and source control, these make sense to adopt. The functionality already exist, therefore no reason to reinvent the wheel. Another area which makes sense to adopt is services which would be very difficult for us to build and have high barriers to entry. Payment and mobile services are such examples. Payment Process involves banks that are high regulated and mobile services involve carriers which are also highly bureaucratic. To gain access to those entities is not easy. Therefore we are better off by adopting and partnering with a payment processor and mobile aggregator.

The subsystems which we develop are the ones that are core to our business. User management, Reporting, Campaign Management, these are the component where we provide value added to customer. These are the area where TXTaGO can differential ourselves from the competitors.

UI design is also an area that is generally a great candidate to adopt. In our project we purchased our Web template from <http://winithemes.com/templates/wordpress/saas-765>.

## 4.3 Implementation Process

The implementation process we will use for this project is a top-down approach. The implementation will be divided into two phases. The first phase will provide the plumbing for the system. We will create the development environment by completing the following setup tasks:

1. Create source control repository
2. Create a build project
3. Create an eclipse project
4. Install database server
5. Install application server

Next we will implement the framework. The setup tasks and the framework will represent the first phase of implementation. The framework will consist of:

1. Logging utilities
2. Configuration utilities
3. Database access utilities
4. Security utilities

The second phase will be the use case implementation. We will implement the use cases that are defined to build out our system. Full functionality is achieved when all use cases are completed. The following are some of the main system use cases:

1. Maintain User
2. Maintain Account
3. Process Subscription
4. Process Payment
5. Process Report
6. Maintain Campaign
7. Maintain Contact
8. Maintain Content
9. Process Campaign
10. Process Message

On completion of phase 1 and phase 2, the system should be feature complete and ready for testing. Unit testing should be done prior to the end of phase 2. The testing phase will include integration testing, QA testing, performance testing, and user acceptance testing.

## 4.4 Tools and Environments

Listed below are the tools that will be used for the project. Most of the tools used are from open source.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tool** | **Description** | **Justification** | **Area Used** |
|  |  |  |  |
|  | **Platform** |  |  |
| Linux | Operating System | Free and reliable | Platform |
| JAVA | Programming Language | Run anywhere, lots of libraries | Platform |
| Tomcat | Application Container | Stable, most popular | Platform |
| MySQL | RDBMS | Free, very popular | Platform |
| JMS | Messaging Service | Compliments Java | Platform |
| HTML | Web Markup | Web Standard | Platform |
|  |  |  |  |
|  | **Development** |  |  |
| Astah UML | Modeling Tool | Free | Design |
| Eclipse | IDE | Free, popular | Development |
| ANT | Build Management | Free, popular | Development |
| GIT | Source Control | Popular | Development |
|  |  |  |  |
|  | **Testing** |  |  |
| JUnit | Unit Testing | Standard | Testing |
| Selenium | Web Testing | Free | Testing |
| JMeter | Performance Testing | Free | Testing |
|  |  |  |  |

## 4.5 Standards

This project will leverage as much industry standards as possible. The table below will outline the open standards or de facto standards used in completing the project.

|  |  |  |
| --- | --- | --- |
| **Standard** | **Description** | **Area Used** |
| UML | Modeling Language | Design Diagrams |
| SMS | Mobile Text Message | Mobile Message Communication |
| MMS | Multimedia Message | Mobile Message Communication |
| SMPP | Communication Transport for SMS | Mobile Message Communication |
| MM7 | Communication Transport for MMS | Mobile Message Communication |
| SOAP | Communication with Mobile Service Provider | Mobile Message Communication |
| JMS | Java Messaging Service | Mobile Message Communication |
| JDBC | Database Connectivity | Database |
| JAVA | Programming Language | Platform |
| HTML | Web Programming Language | Web |
| CSS | Stylesheet | Web |
| Javascript | Dynamic Web Language | Web |
| ANT/Maven | Build Management | Build/Deployment |
| Eclipse | IDE | Development Tool |
| Servlet | Web Container | Web |
| Linux | Operating System | Operating System |
| Tomcat | Application Container | Application |
|  |  |  |
|  |  |  |

## 4.6 Implementation Issues and Resolutions

Our implementation issues are mostly related to the prototype. If we were to develop a live production system, the issues we encounter would not exist.

Issues:

1. Our mobile service provider (Sybase) requires a direct VPN or whitelisting of source IP for communicating with them. Since we are not officially setup with them for an account, we were not able to obtain a VPN or IP whitelisting.
2. Our payment service provider (Chargify) requires a callback URL to return information from a subscription or account update. A callback URL must be publicly accessible, so we need to deploy our application on a public domain.

Resolutions:

1. Unable to connect to Sybase externally, we decided to connect to them internally. As an employee of Sybase, we were able to deploy our application with the firewall.
2. Deploying our application internally with Sybase means we were not going to be publicly accessible for the Chargify system to callback to us. In order to resolve this, we deployed a public access web page to accept and redirect the Chargify information back to our internal application.
3. Another issue with an internal deployment is the inability to setup a database. We resolve this by using an in-memory standalone database (HSQL).

# Chapter 5 Testing Plan

## 5.1 Testing Scope

The scope of the testing phase includes the unit testing for the individual subsystems. Unit testing should be done from both a white box and black box approach. After all unit testing is complete, integration testing of the system should be done to ensure the communication between the subsystems behave accordingly. Once integration testing is complete, we will proceed to performance testing and user acceptance testing.

## 5.2 Testing Approaches

Below are the three roles required to participate in the testing phase along with their responsibilities.

|  |  |  |
| --- | --- | --- |
| **Role** | Responsibilities | Tools |
| Developer | White box testing, integration testing, performance testing | JUnit, JMeter |
| QA Engineer | White box testing, UI testing | Selenium |
| Analyst | Acceptance testing | Manual |

Developer should ensure the subsystem interfaces execute as according to its pre and post conditions. This is achieved by white box testing. White box testing should be facilitated by JUnit automation where possible. Developer should also ensure the different subsystems communicate as defined by their interfaces and protocols. This should be done with written test cases which covers the sanity of the system. Developer should also conduct performance and scalability testing to ensure the higher availability of the system. This can be done with JMeter.

QA Engineer should create test cases to test primary, secondary, and tertiary scenarios. These test cases may be automated with Selenium. Ad hoc testing should also be conducted to perform arbitrary tasks that may break the system.

Analyst should interact with the system to ensure requirements are met and developed according to specification. UI user friendliness should also be addressed during acceptance testing.

## 5.3 Testing Results

The table below (Figure 50) summarizes the test cases that were run against the TXTaGo system and the result that was observed. The test cases were based on the functional requirements.

|  |  |  |
| --- | --- | --- |
| **TC ID** | **Test Case Description** | **Status** |
| TC1.1 Maintain User |  |  |
| TC1.1.1 | Login user when provided valid username/password | **Pass** |
| TC1.1.2 | Create valid user session when login | **Pass** |
| TC1.1.3 | Invalidate session when logged out | **Pass** |
| TC1.1.4 | Provide CRUD for a user | **Pass** |
| TC1.2 Maintain Account |  |  |
| TC1.2.1 | Create/Update account on Chargify | **Pass** |
| TC1.2.2 | Update user subscription to reflect Chargify | **Pass** |
| TC1. 3 Process Subscription |  |  |
| TC1.3.1 | Provide CRUD for subscription | **Pass** |
| TC1.3.2 | Provide payment status for subscription | **Pass** |
| TC1.3.3 | Update message quota | **Pass** |
| TC1.3.4 | Update subscription expire and renewal | **Pass** |
| TC1.3.5 | Charge reoccurring subscription account | **Pass** |
| TC1.3.6 | Upgrade and downgrade subscription | **Pass** |
| TC1.3.7 | Subscribe to Freemium plan | **Pass** |
| TC1.4 Manage Payment |  |  |
| TC1.4.1 | Chargify authorize a payment | **Pass** |
| TC1.4.2 | Chargify settle a payment | **Pass** |
| TC1.4.3 | Chargify credits and debits | **Pass** |
| TC1.5 Process Report |  |  |
| TC1.5.1 | Run daily report | **Pass** |
| TC1.5.2 | Find by message by sender/recipients report | **Pass** |
| TC1.5.6 | Find by days back in report | **Pass** |
| TC1.6MaintainCampaign |  |  |
| TC1.6.1 | Provide CRUD for a campaign | **Pass** |
| TC1.6.2 | Set start date and time on campaign | **Pass** |
| TC1.6.3 | Create/Delete contacts with a campaign | **Pass** |
| TC1.6.4 | Update/Add content with a campaign | **Pass** |
| TC1.7MaintainContact |  |  |
| TC1.7.1 | Create/Delete contacts with a campaign | **Pass** |
| TC1.7.2 | Create 2 identical contacts | **Pass** |
| TC1.8Maintain Content |  |  |
| TC1.8.1 | Update/Add content with a campaign | **Pass** |
| TC1.8.2 | Add content of text and multimedia | **Pass** |
| TC1.9Process Campaign |  |  |
| TC1.9.1 | Start campaign on a timer | **Pass** |
| TC1.9.2 | Processing campaign now | **Pass** |
| TC1.9.3 | Send mobile messages on start of campaign | **Pass** |
| TC2.0 Process Message |  |  |
| TC2.0.1 | Send message to mobile service provider | **Pass** |
| TC2.0.2 | Report the success of failure of message | **Pass** |
|  |  |  |

Figure 50 Test Case Summary

# Chapter 6 Conclusion and Future Work

## 6.1 Project Summary

In the table below, the detailed tasks of each phase is outlined and its status. Each task is assign to an individual who is responsible for the completion of the task. See Figure 50 below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Task ID** | **Task** | **Owner** | **Status** |
|  |  |  |  |
|  | **Design and Documentation** |  |  |
| 1.1.1 | Project Objective | Stephen Kwan | Complete |
| 1.1.2 | Market Research | Stephen Kwan | Complete |
| 1.2.1 | Business/Use Case Modeling | Stephen Kwan | Complete |
| 1.2.2 | Functional/Non Functional Requirements | Stephen Kwan | Complete |
| 1.2.3 | Context, Interface, and Resource Requirements | Stephen Kwan | Complete |
| 1.3.1 | Architecture Design | Stephen Kwan | Complete |
| 1.3.2 | System Logic Design | Stephen Kwan | Complete |
| 1.3.3 | System Interface Design | Stephen Kwan | Complete |
| 1.3.4 | User Interface Design | Stephen Kwan | Complete |
|  |  |  |  |
|  | **Implementation Phase 1** |  |  |
| 2.1.1 | System Framework Development | Stephen Kwan | Complete |
| 2.1.2 | Subsystem Containers and Communication | Stephen Kwan | Complete |
| 2.1.3 | Development, Integration, and Staging Environments | Stephen Kwan | Complete |
| 2.1.4 | Source Control and Build Environments | Stephen Kwan | Complete |
|  |  |  |  |
|  | **Implementation Phase 2** |  |  |
| 3.1.1 | User System Use Case Implementation | Stephen Kwan | Complete |
| 3.1.2 | Account System Use Case Implementation | Stephen Kwan | Complete |
| 3.1.3 | Report System Use Case Implementation | Stephen Kwan | Complete |
| 3.1.4 | Campaign System Use Case Implementation | Stephen Kwan | Complete |
| 3.1.5 | Subscription System Use Case Implementation | Stephen Kwan | Complete |
|  |  |  |  |
|  | **Integration** |  |  |
| 4.1.1 | Developer Unit Testing | Stephen Kwan | Complete |
| 4.1.2 | Developer Integration Testing | Stephen Kwan | Complete |
|  |  |  |  |
|  | **Testing** |  |  |
| 5.1.1 | QA Test Case Creation | Stephen Kwan | Complete |
| 5.1.2 | QA Testing | Stephen Kwan | Complete |
| 5.1.3 | QA Ad Hoc Testing | Stephen Kwan | Complete |
| 5.1.4 | Analyst Acceptance Testing | Stephen Kwan | Complete |
| 5.1.5 | Developer Bug Fix | Stephen Kwan | Complete |
| 5.1.6 | Developer Performance Testing | Stephen Kwan | Complete |
|  |  |  |  |
|  | **Review** |  |  |
| 6.1.1 | Final Audit of System Production Readiness | Stephen Kwan | Complete |
|  |  |  |  |
|  |  |  |  |

Figure Project Tasks

We learned a lot from this project. We learned how to properly document our work using UML and written design documents. The implementation of the project provided us with valuable experience in working with some of the latest technology. We gain experience with JSF, REST, and Hibernate. We also learn to use cloud platforms and services. We used Chargify, Sybase, and CloudBees. Everything we used and learn from this project can definitely be applied and translated to the real world and professional work life.

## 6.2 Future Work

The goal of this project is to launch TXTaGo. The prototype provided a proof of concept and a concrete realization of the idea to gather support and funding. In order for this product to be production quality we must complete the following:

1. Perform better validation on inputs and output.
2. Tighten security on the application both the Web and the REST API.
3. Deploy to a real RDBMS.
4. Sign up for real service with Sybase and Chargify.
5. Obtain shortcode to use from CTIA
6. Allow multiple contents in MMS message.
7. Host the content for MMS.
8. Provide better analytics.

# Appendix

|  |  |
| --- | --- |
|  | **Description** |
| UML | Modeling Language |
| SMS | Mobile Text Message |
| MMS | Multimedia Message |
| SMPP | Communication Transport for SMS |
| MM7 | Communication Transport for MMS |
| SOAP | Communication with Mobile Service Provider |
| JMS | Java Messaging Service |
| JDBC | Database Connectivity |
| JAVA | Programming Language |
| HTML | Web Programming Language |
| CSS | Stylesheet |
| Javascript | Dynamic Web Language |
| ANT/Maven | Build Management |
| Eclipse | IDE |
| Servlet | Web Container |
| Linux | Operating System |
| Tomcat | Application Container |
|  |  |
|  |  |

# References

[1] Giselle Tsirulnik(Feb 3, 2011)**.***Global SMS traffic to reach 8.7 trillion by 2015: study*. <http://www.mobilecommercedaily.com/global-sms-traffic-to-reach-8-7-trillion-by-2015>

[2] *2012 Best Text Message Marketing Service*.

<http://text-message-marketing-review.toptenreviews.com/>

[3] Chetan Sharma (2011). *US Mobile Messaging Market – Growth and Opportunities*.

<http://mobilebroadbandopportunities.com/chetansharma/Sharma3.pdf>

[4] William Dudley (Sept 22, 2011). *Messaging Gets More Crowded*.

<http://blogs.sybase.com/wdudley/?p=951>

[5] Alan Quayle (Jul 2012). *The Over The Top Messaging Landscape*.

<http://www.openmobilealliance.org/comms/documents/OMA%20OTT%20Messaging%20Issue%201_AQuayle.pdf>

[6] Jiehong Kong (Oct 25, 2006). *The Innovative Business Model behind the Rapid Growth of SMS in China*.ACEM, Shanghai Jiao Tong Univ.Service Systems and Service Management, 2006 International Conference

[7] Y. Takada, N. Komoda, H. Oiso, A. Hiramatsu, Y. Fukaya (2004). *Customer analysis of monthly-charged mobile content aiming at prolonging subscription period*.Computational Cybernetics, 2004.ICCC 2004. Second IEEE International Conference

This page was left blank intentionally