Addis Ababa Institute of Technology

Center of Information Technology and Scientific Computing

Department Software Engineering

Ad Platform - Software Design Specification

Team Members

Enkusellasie Wendwosen

Haftamu Meresa

Hiwot Bishaw

Melkamu Mitiku

Advisor

Abraham Getachew

January 21, 2017

Table of Contents

List of Tables

List of figures

Definitions, Acronyms, Abbreviations

Definition

No		Definition
1	Webmaster	Website owner, Website Operator, that generates income from the website he/she operates

Acronyms

No		Definition
1	SDS	Software Design Specification

Abbreviations

No		Definition
1	Ad	Advertisement

1. Introduction

1.1 Purpose

The Advertisement industry is a very complicated industry with multiple stakeholders. Each stakeholder has a unique contribution to the survival of the industry. When we approached the design of this system, it was done so taking this crucial fact into consideration. The purpose of this document is to translate the business requirements we identified in the SRS document and also business processes into technical design that will be used to develop the application.

This document is intended for anyone interested in understanding the design and architecture of our system, who has experience reading UML diagrams which include but not limited to sequence diagrams, class diagrams, component diagrams. This document includes but is not limited to the following information for the Ad Platform; system overview, system architecture, object model and detailed system design.

1.2 General Overview

Ad Platform is, as its name suggests, a platform where businesses will be able to advertise their products and services. A platform that will enable web publishers monetize their websites by letting us advertise on their platform. The 'Ad Platform' name is merely a developmental project name and in no ways reflects the final name for the product. We might change the name of the project if necessary and we will identify the change that were made and why.

The system has to be able to do main three tasks efficiently; in order to accomplish its business requirements. The three tasks are, delivering ads; collecting analytics, analyze and classify webpages. The system architecture is designed with these tasks in mind. The system has four main subsystems; Ad Delivery Network, WebPage Classifier, Cross System Analytics Repository and Website Plugin. Each subsystem is providing functionality for one of the tasks listed above.

1.3 Goals

Some of our goals when designing the system are listed in this section. The major goal of the Ad Platform is to cater to the end user with relevant ads that will encourage the user to interact with the ad without being intrusive. In order to satisfy this we have devised the WebPage Classifier subsystem of which it's sole purpose is to classify webpages from a wide

variety of categories. After the webpage is classified, ads in a similar category will be displayed on that webpage.

The second major goal of the ad platform is to collect as many relevant stats as possible and provide publishers and advertisers with all the information they need to make informed decisions. In order to do this we have designed the Cross System Analytics Repository subsystem; whose sole purpose is to collect stats from the plugin, store them securely and do some relevant analytics.

The third goal is to make the user interface as intuitive as possible to all actors on the system. This goal specifically refers to the web panel and plugin. A web panel that website publishers and advertisers will use to view stats, ads and more. Since the web panel is the primary way for all actors to interact, it has to very user friendly. To accomplish this goal we are using rich client side javascript libraries to enhance the user experience. We also hope to support multilingual functionality.

This design attempted to separate the main functionalities into separate subsystems. This is done so in part to increase performance of the system. We believe performance will increase as each subsystem will be independent of the other. For example, The system can deliver ads and at the same time collect stats about the webpages those ads are being displayed in. This is also a critical feature when scalability is considered.

1.4 Development Methods & Contingencies

In order to coup with the complexity and changing nature of our system; we have stated in our proposal document that our approach for design and development of our system is RAD and specifically SCRUM.

Choosing such an approach enables us to look at the requirements from a different perspective, which is that they are ever changing. This means that during development, we are able to entertain requirement changes that are essential to the system. We have planned, as we stated in our proposal document, to do at least three iterations. With each successive iteration we will incorporate change to requirements .

2. System Architecture

2.1 Architecture Design Considerations

This section will provide an outline of the various components and subsystems of the Ad Platform system. As stated in the previous section, the system has four main subsystems. This section will present each subsystem in a multitude of diagrams. We have also stated the rationale behind each subsystem. In this section we will go over some design considerations taken into consideration when designing the system architecture.

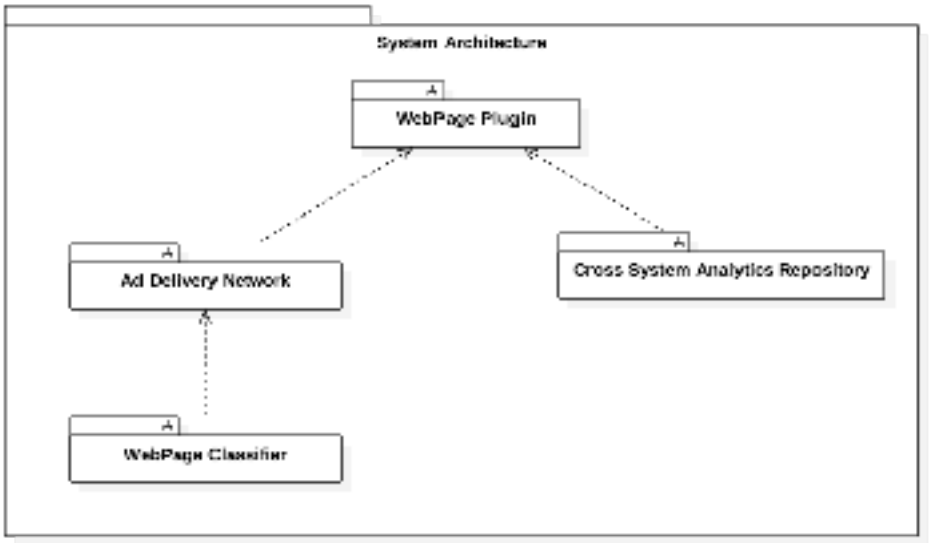
The main design consideration that went into the Website Plugin is minimization of it's size. The size needs to be as small as possible, in order to be loaded on browsers even in low internet connections. When size is a constraint, every feature that is considered to be added needs careful consideration. If a feature is not absolutely essential it cannot be added.

The ad delivery network's main design consideration is speed of delivery. Every request to this server needs to be analyzed and classified by another subsystem that this subsystem works closely with. That subsystem is the WebPage classifier. But subsequent requests made for a certain webpage need to be cached, because it is highly unlikely that webpages get updated very often. Hence this subsystem has a caching mechanism that makes the system more efficient and fast.

The WebPage analyzer subsystem works closely with the ad delivery network to classify webpages. Its sole task is analysis of webpages, isn't concerned with caching. The webpage classifier implements Natural Language Processing Algorithms to effectively extract topic from a webpage. To do so it relies on a set of keywords that are collected for each topic. Hence the main design consideration for this subsystem is getting smarter upon each classification. This subsystem will learn new keywords and will be better at classification.

The main design consideration for the Cross System Analytics Repository is data independence. What we mean by that is the stats that are collected on each Ad or User should be represented in a general manner. Hence when designing the persistence for this subsystem, we devised a type specification mechanism where all the stats are represented in just two tables. One for the type of the stat and another for the stat itself.

Figure 1 System architecture



2.1Subsystem decomposition

In this section we will dive deeper into each subsystem and show their internal components using UML 2.0 Component Diagrams. We have shown diagrams for all subsystems except the WebPage Classifier subsystem, of which its details haven't been finalized yet.

Figure 2 Website Plugin Network Component Diagram

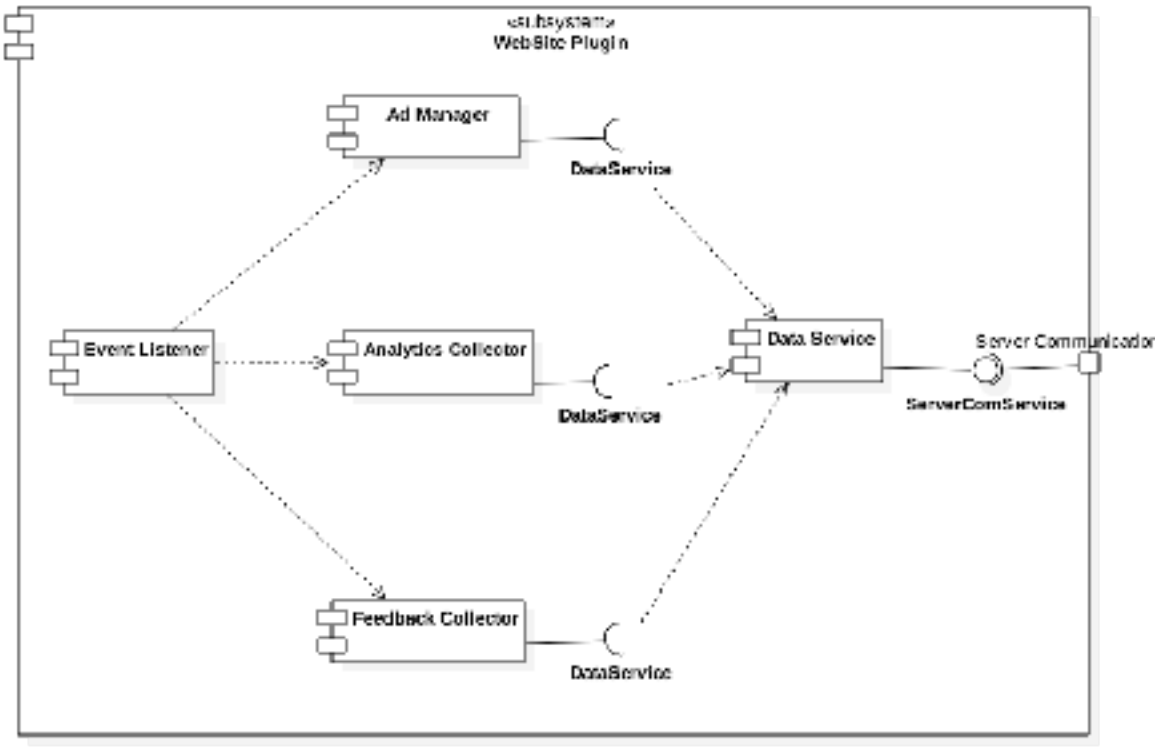


Figure 3 Ad Delivery Network Component Diagram

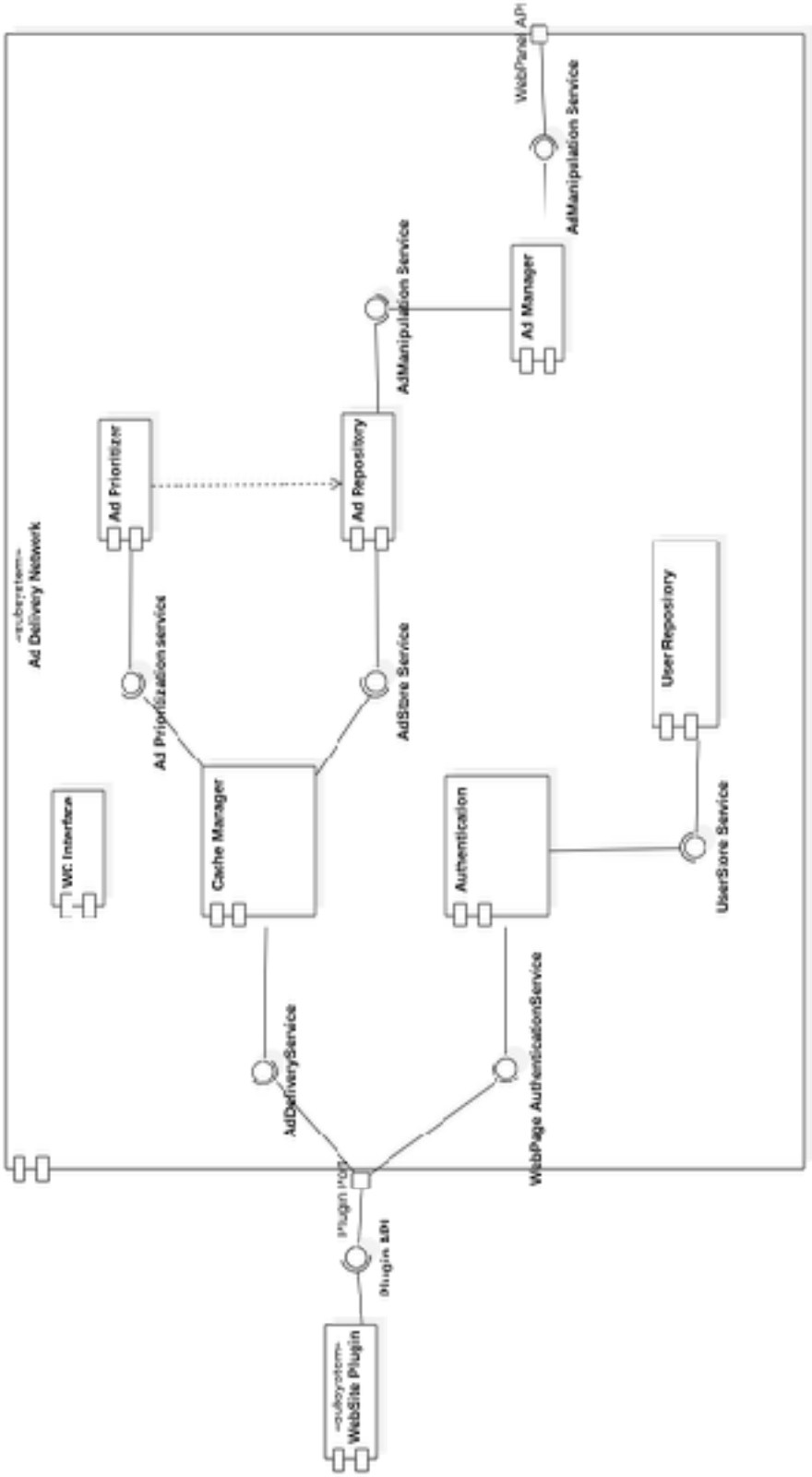
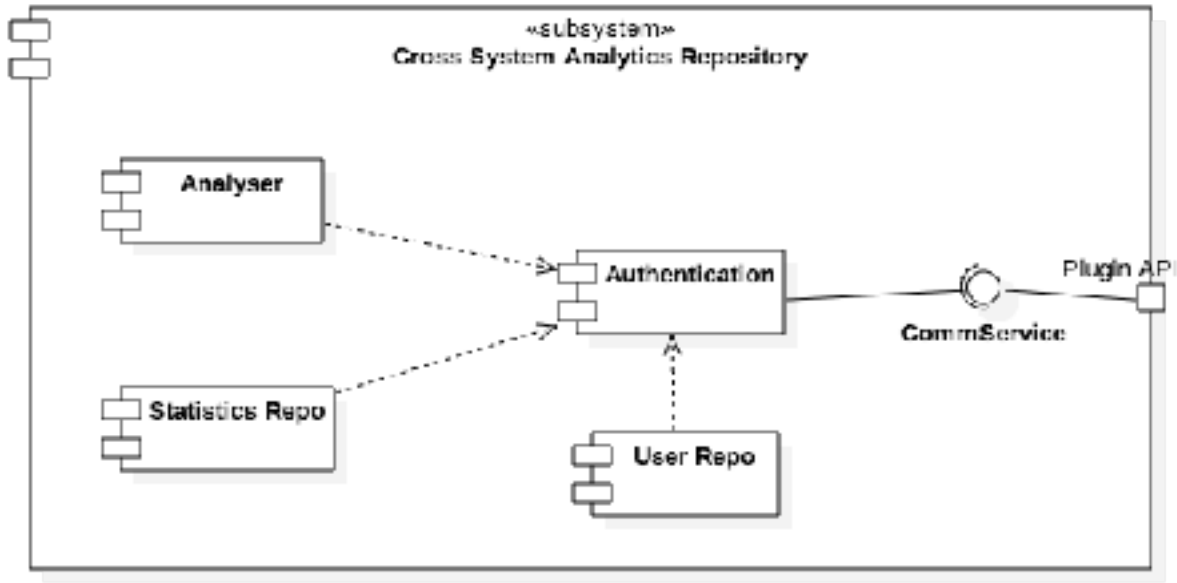


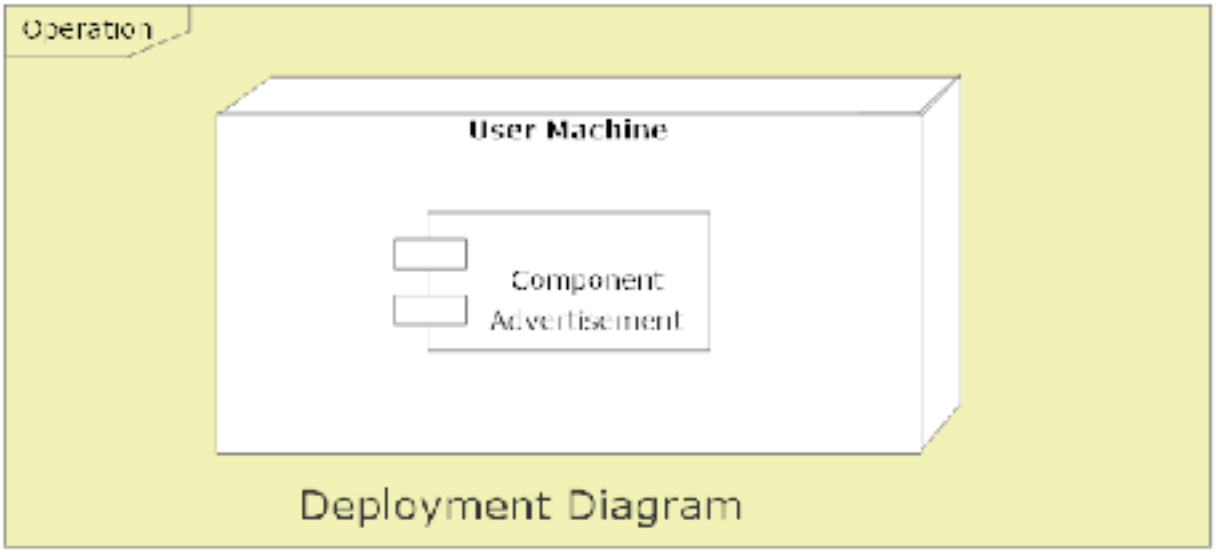
Figure 4 Cross System Analytics Repository Component Diagram



2.2 Hardware/software mapping

UML Deployment diagram.

Figure 5 Deployment Diagram



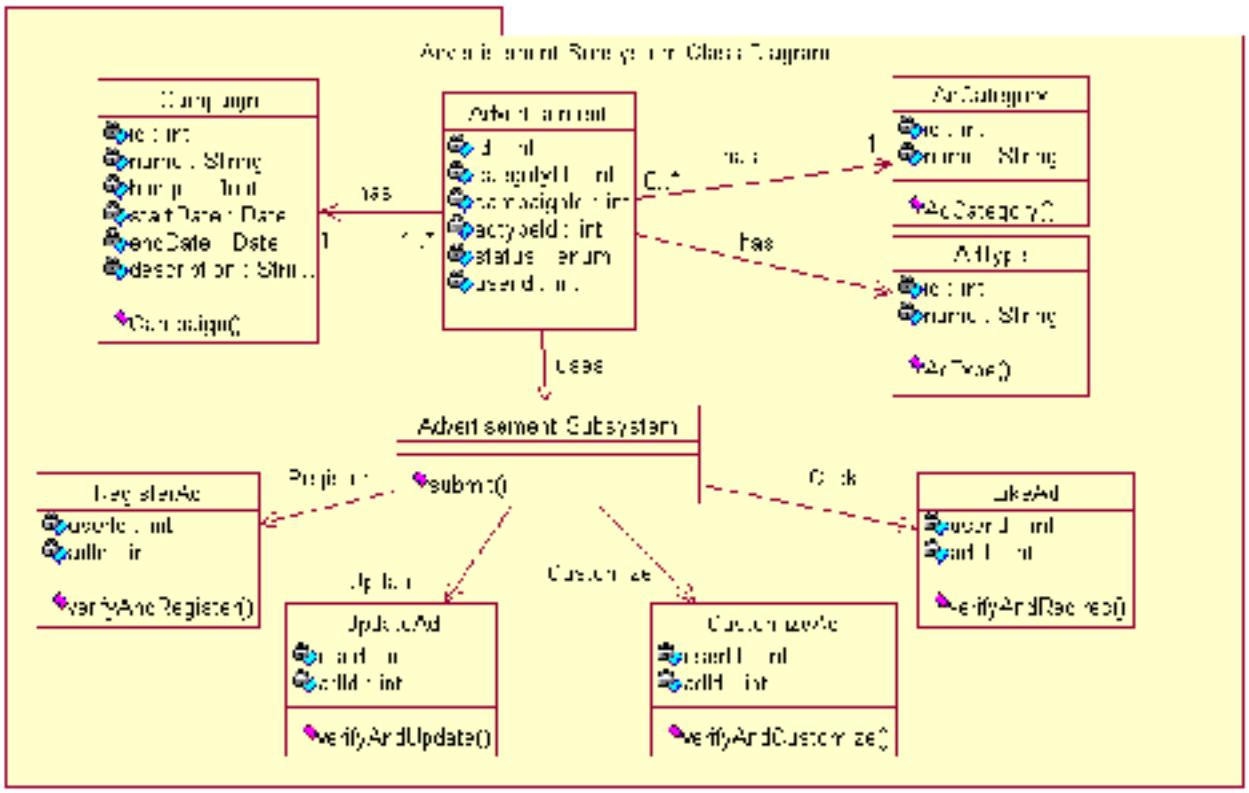
3. Object Model

3.1 Class Diagram

We have provided a Unified Modeling Language (UML) based type of static structure diagram that describes the structure of our system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. Note that for each class minimum number of properties are specified. Because unnecessary properties will make the diagram complicated.

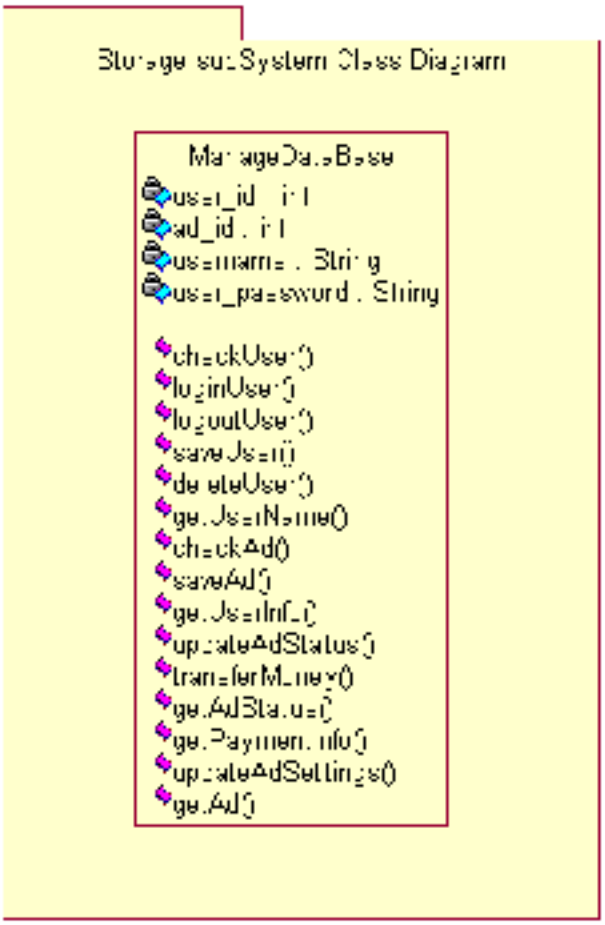
3.1.1 Advertisement Class Diagram

Figure 5 Advertisement class Diagram



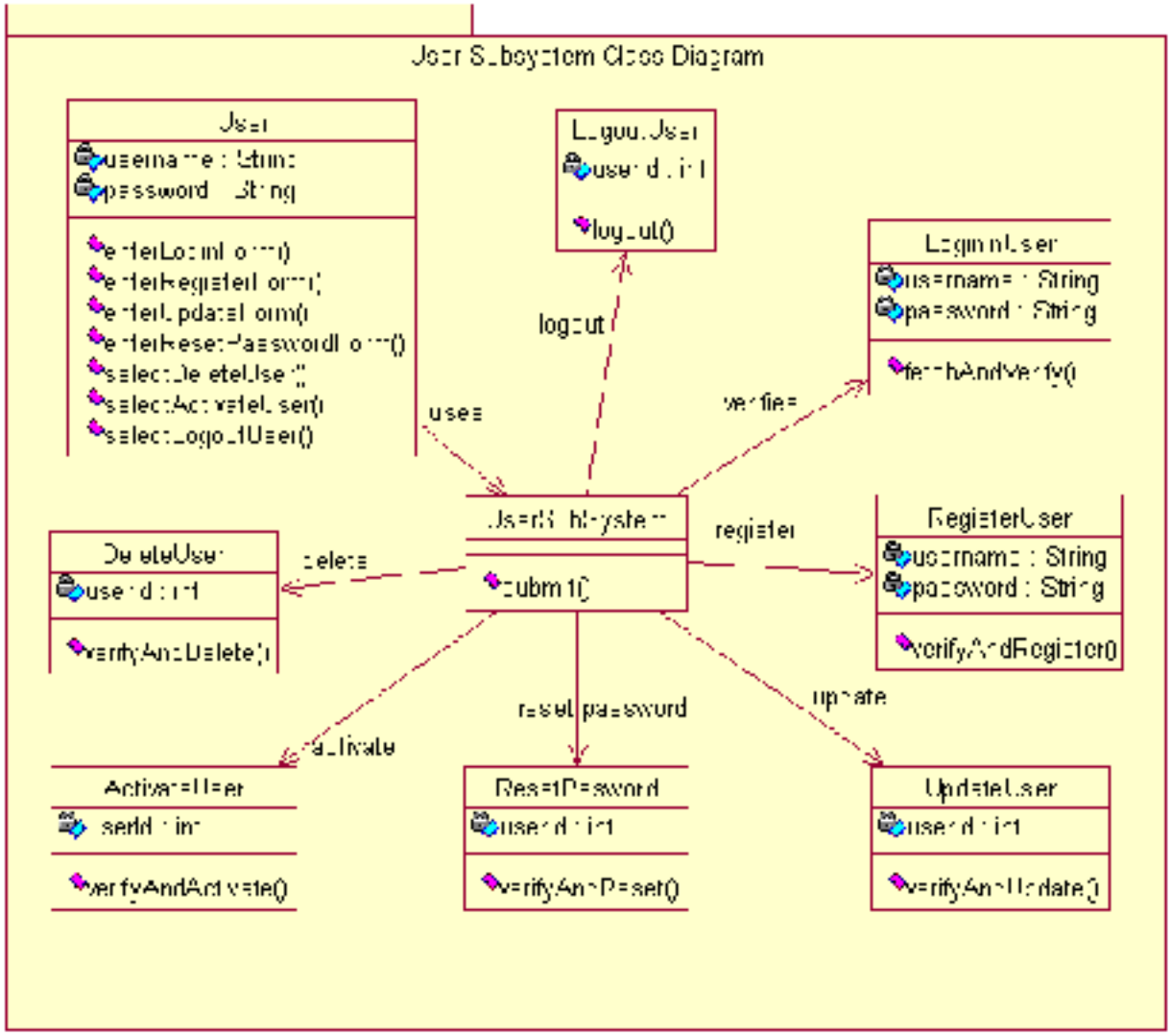
3.1.2 Storage class Diagram

Figure 5 Storage class Diagram



3.1.4 User subsystem class Diagram

Figure 7 User class Diagram



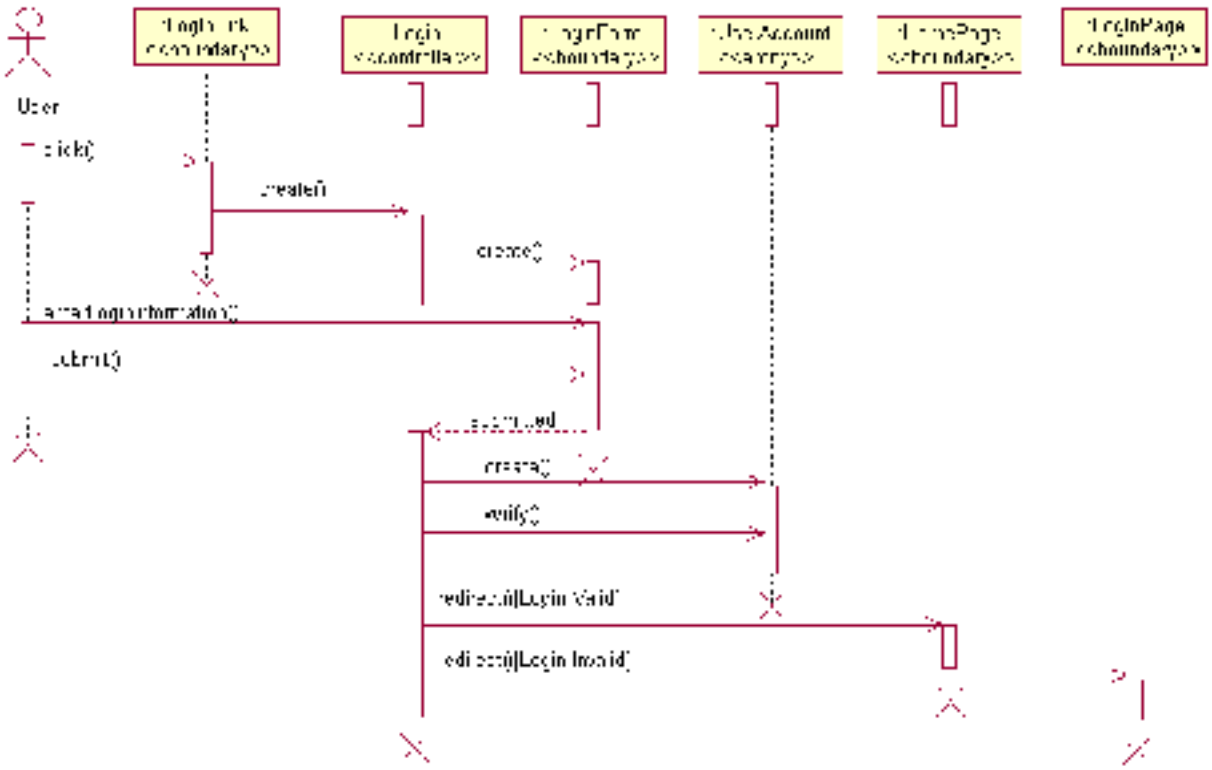
3.2 Sequence Diagram

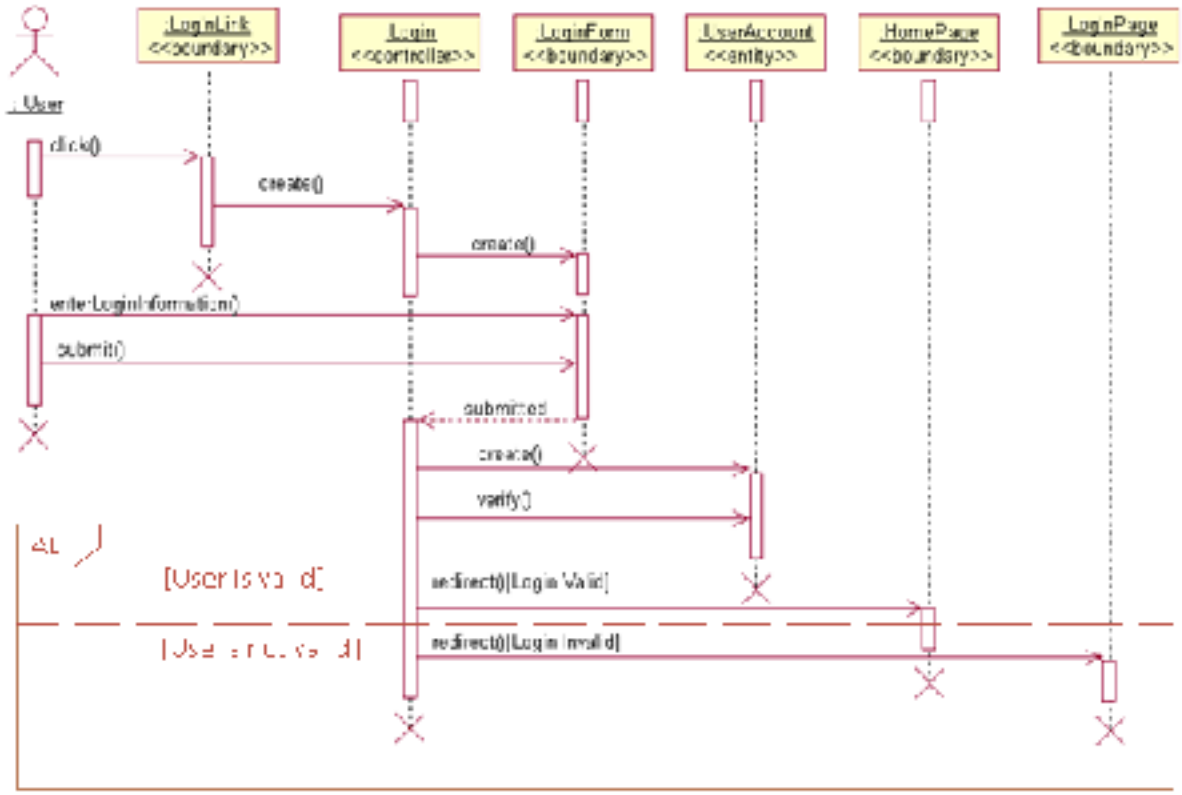
This Section shows how processes operate with one another and in what order. We depict the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

3.2.1 Accounts System

3.2.1.1 The system should allow users to login

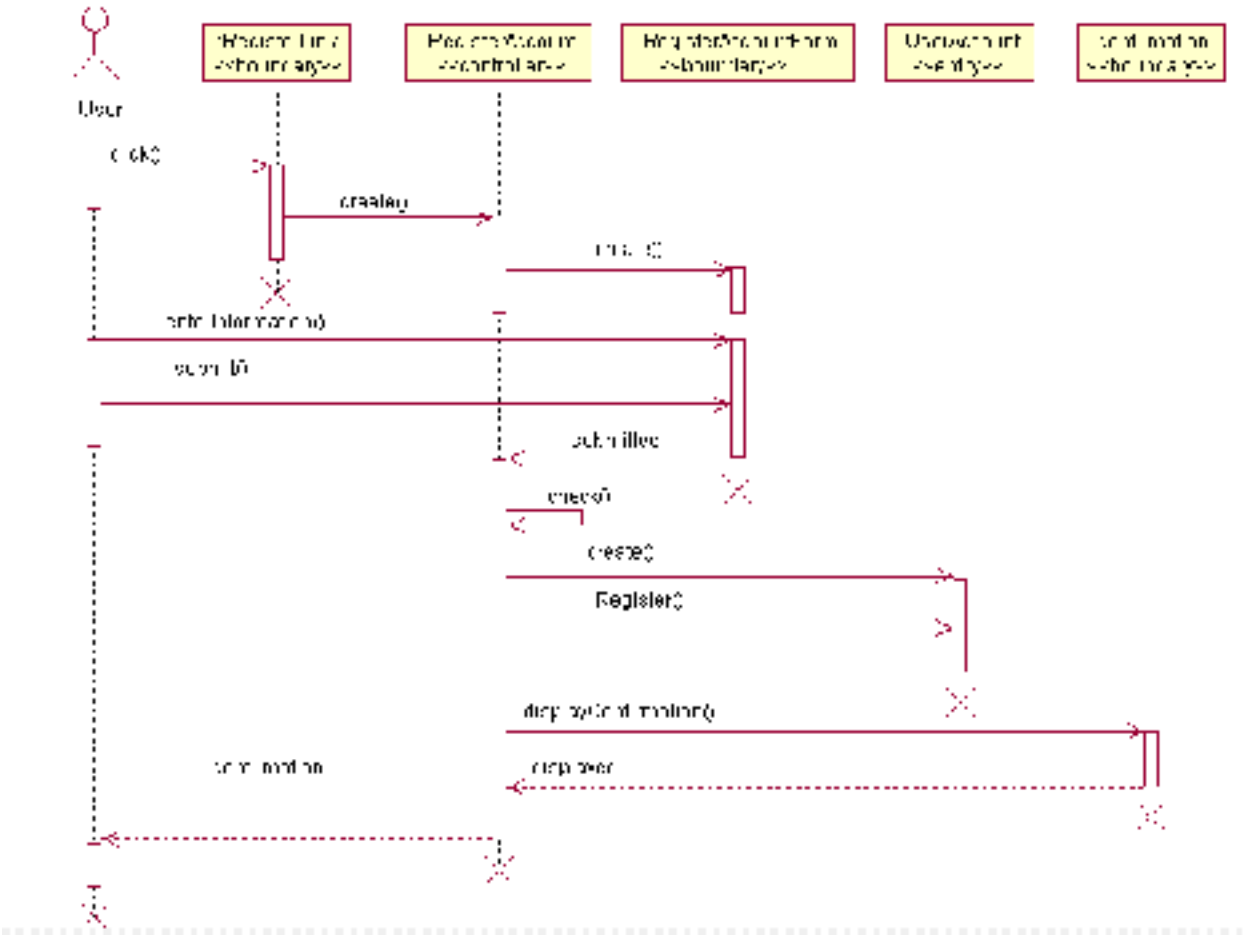
Figure 8 sequence diagram - login




:User
LoginLink
<<boundary>>
Login
<<controller>>
LoginForm
<<boundary>>
UserAccount
<<entity>>
HomePage
<<boundary>>
LoginPage
<<boundary>>
click()
create()
create()
enterLoginInformation()
submit()
submitted
create()
verify()
[User is valid]
redirect()[Login Valid]
redirect()[Login invalid]

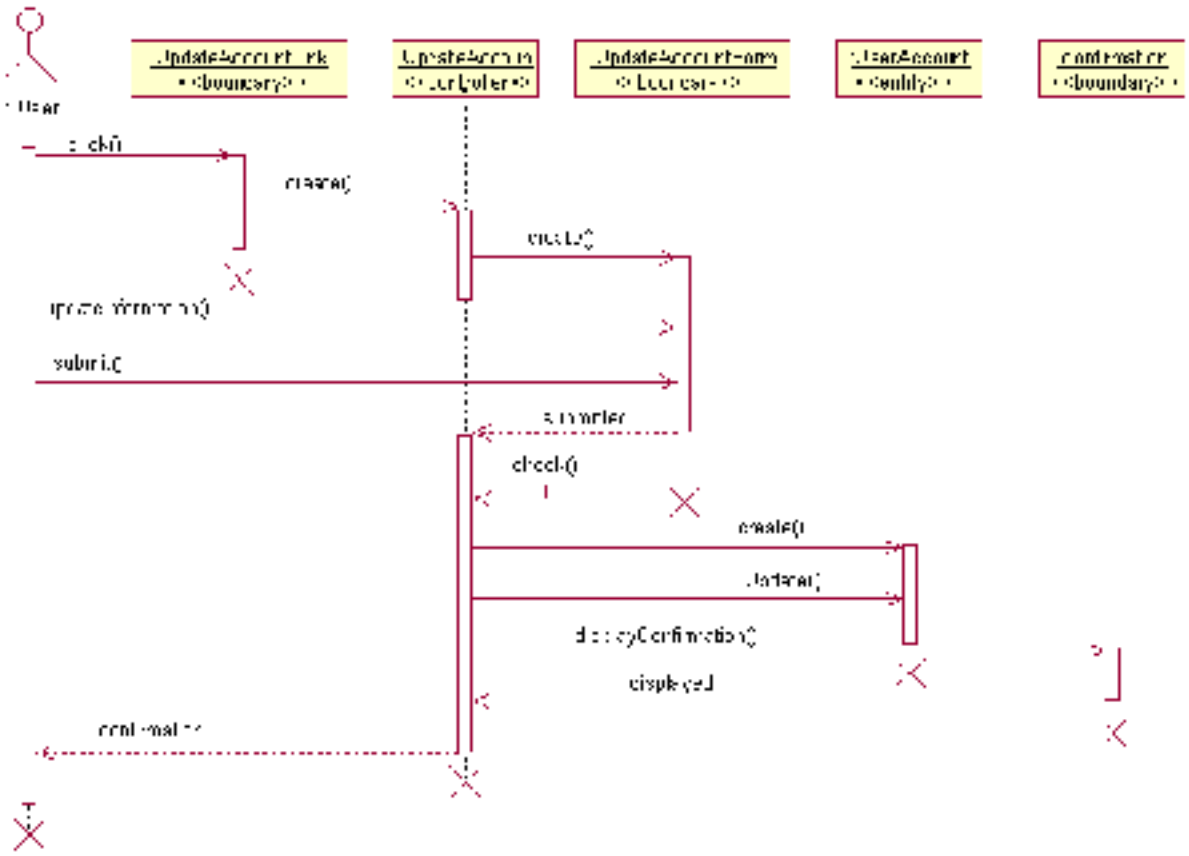
3.2.1.3 The system should allow users to register an account

Figure 10 Sequence Diagram for register an account



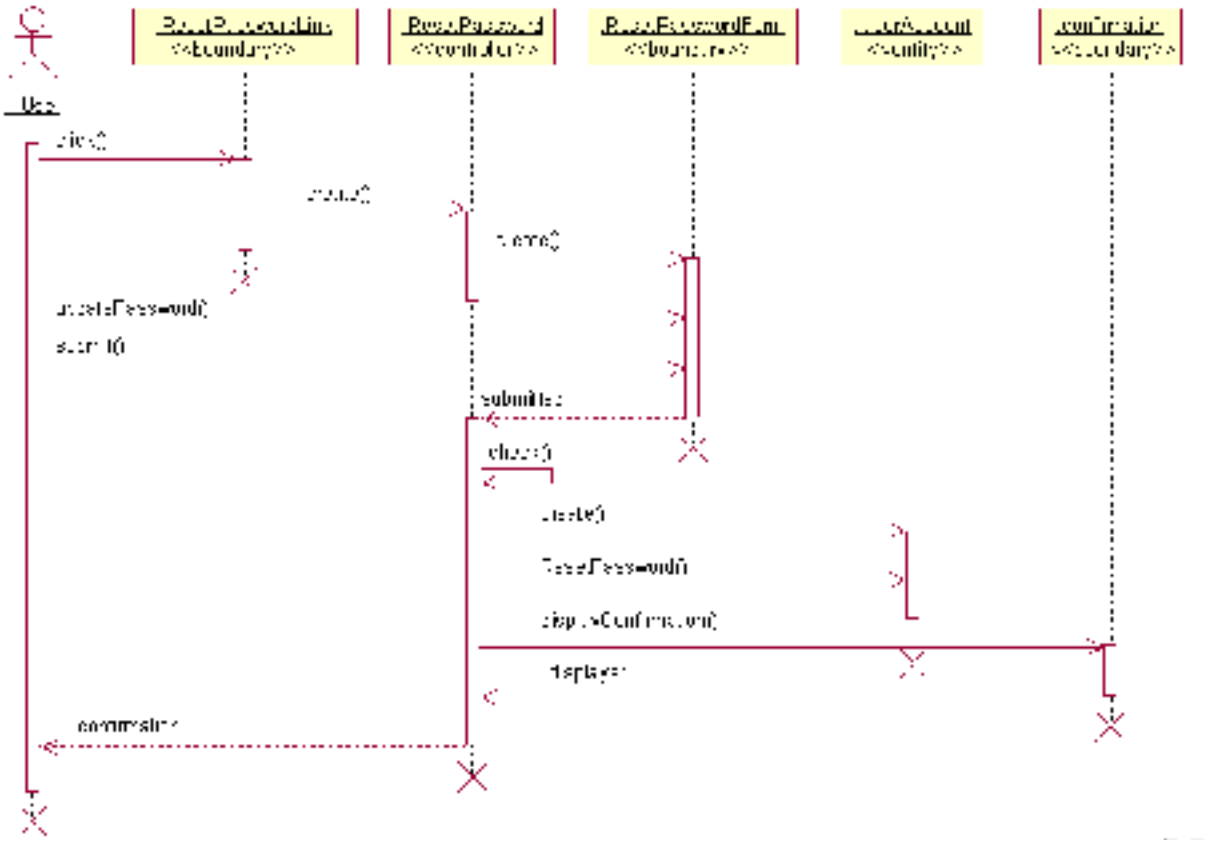
3.2.1.4 The system should allow users to update their account

Figure 11 Sequence Diagram for update an account



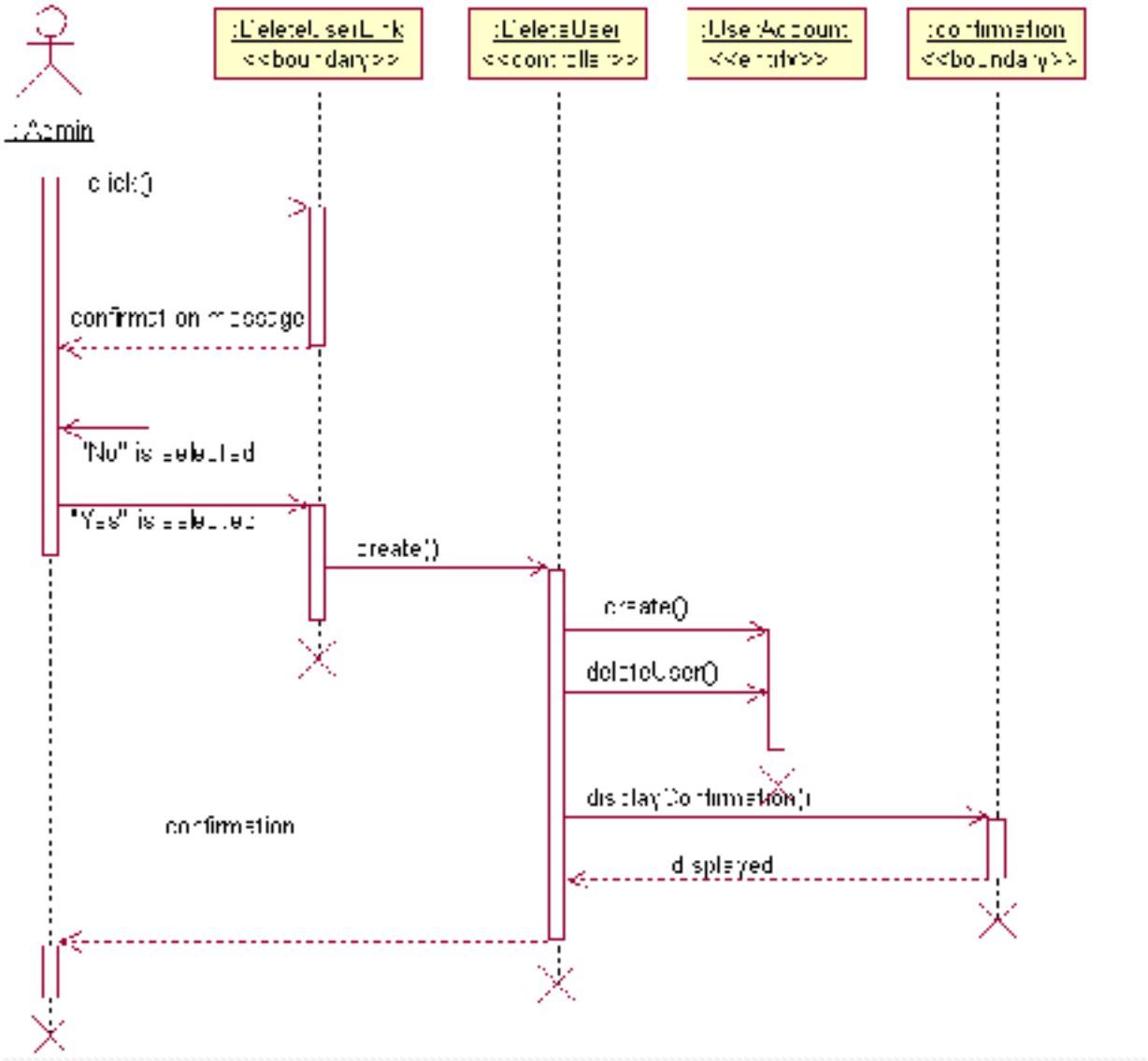
3.2.1.5 The system should allow users to reset their password

Figure 12 Sequence Diagram for reset password



3.2.1.7 The system should allow the administrator to delete user account

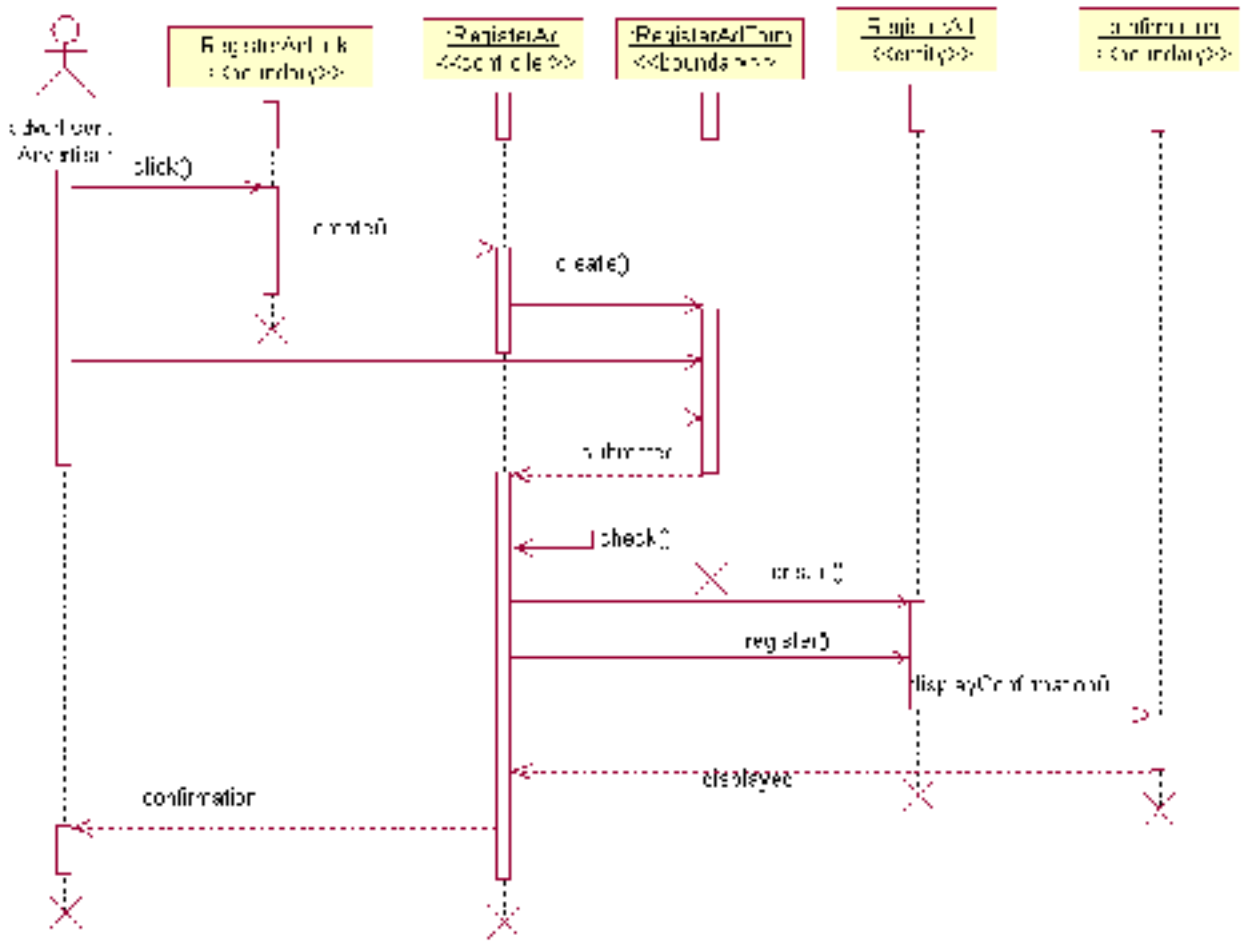
Figure 14 Sequence Diagram for delete user



3.2.2 Advertisement System

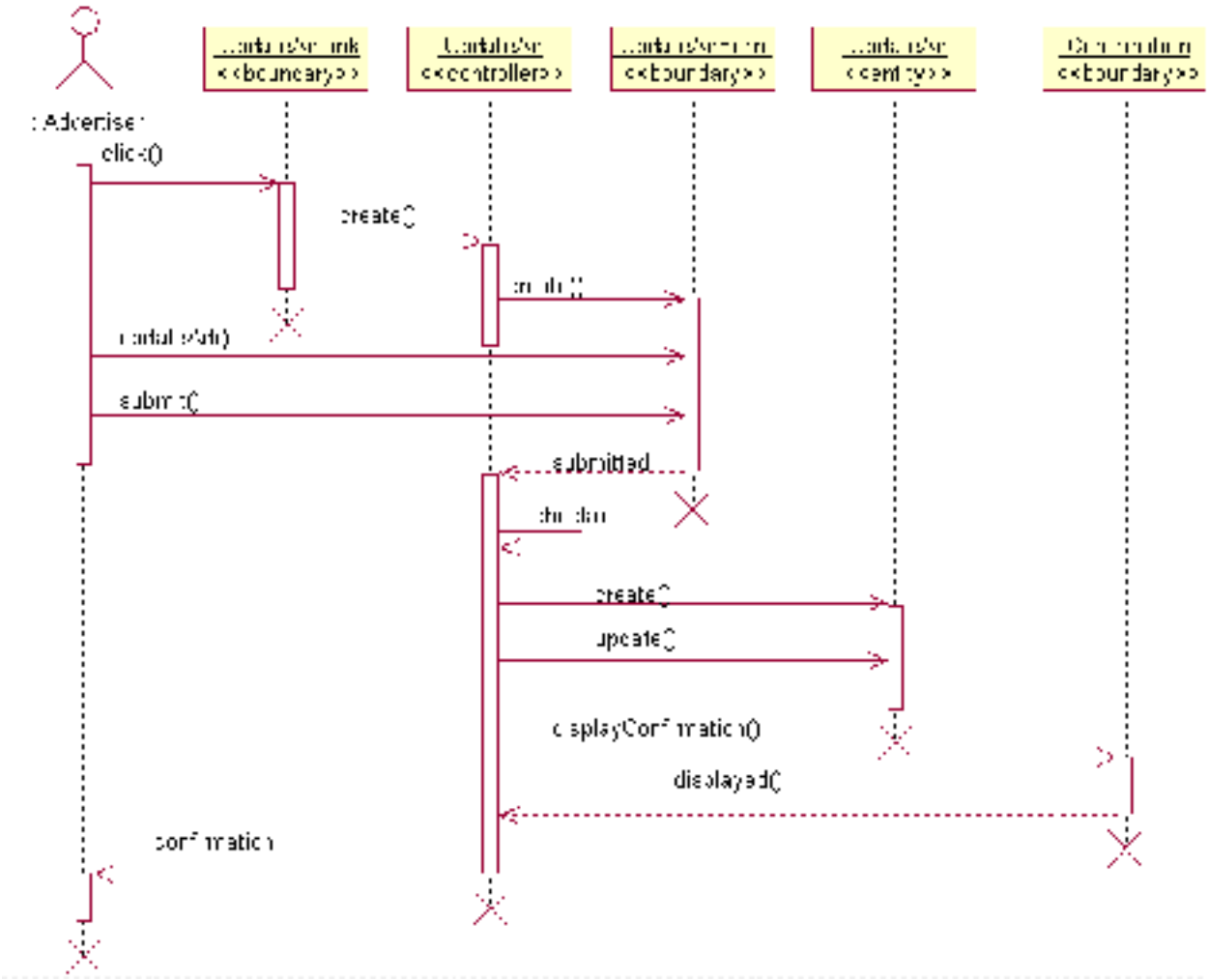
3.2.2.1 The system shall allow the advertiser to add new advertisement

Figure 15 sequence diagram – add new Ad



3.2.2.2 The system shall allow the advertiser to update advertisement

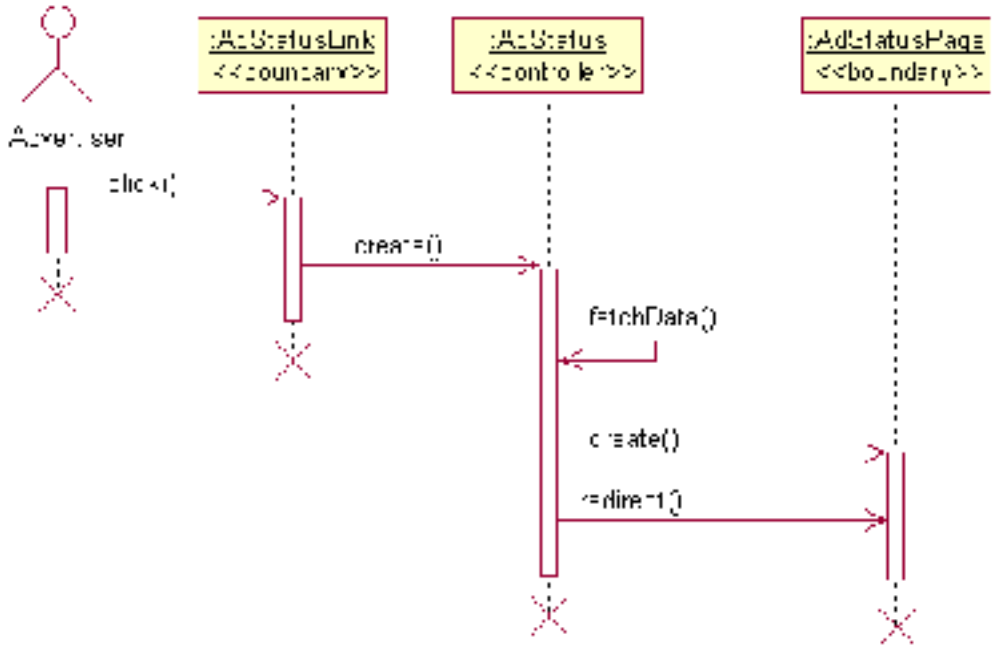
Figure 16 Sequence Diagram for update ad



3.2.3 User data System

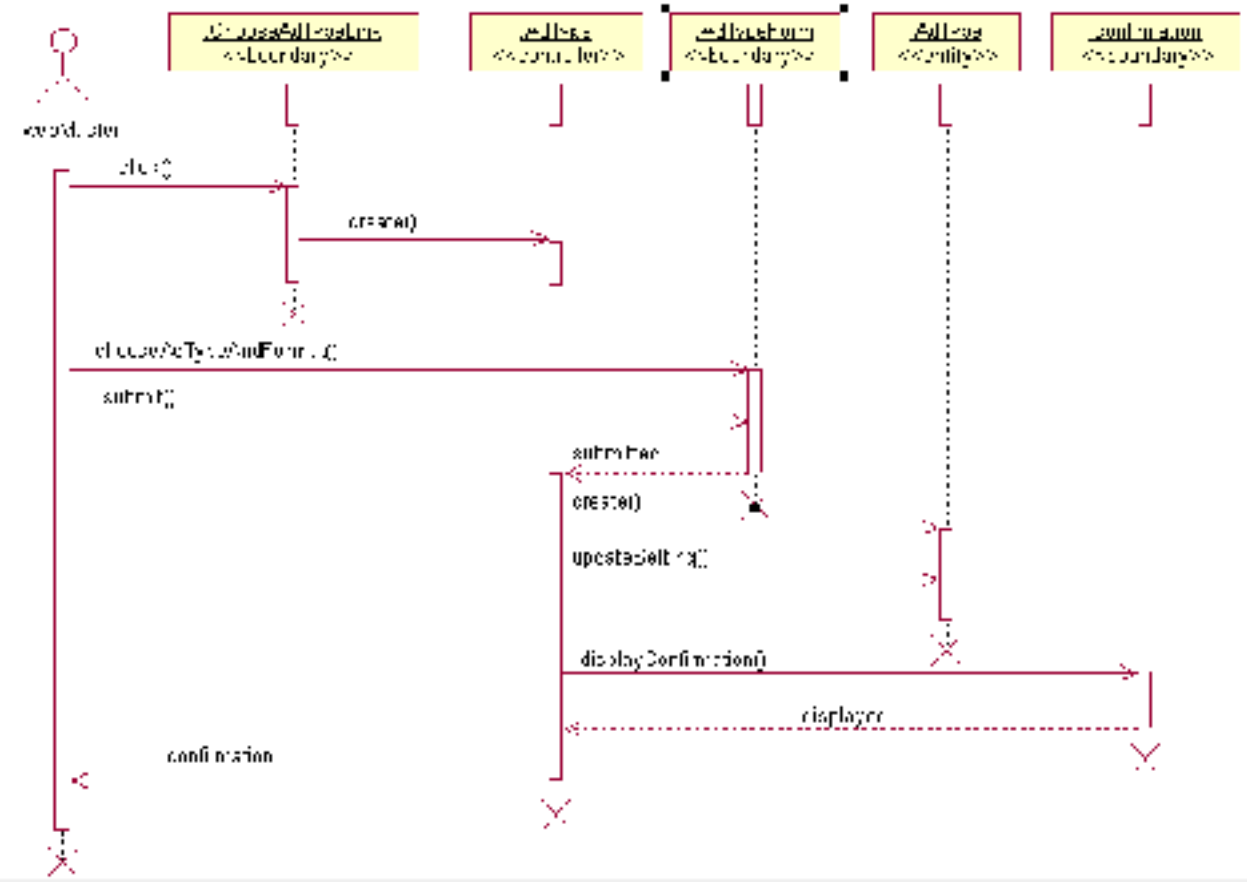
3.2.2.1 The system shall allow the advertiser to view his ad status

Figure 18 Sequence Diagram for view ad status



3.2.3.3 The system shall allow the webmaster to choose ad types and formats

Figure 20 Sequence Diagram for choose ad type and formats



3.3 State chart Diagram (optional element)

We have no an object that can be in many states.

4. Detailed Design

The classes represented here are the ones identified on our class diagram. And we have also add the methods and classes identified in sequence and state chart diagram.

4.1 Detail Design for User Class

Table: 4 User class

User
+name: String - password: String +address: String -email: String -id: String -status : ENUM
+createAccount() +activateUser() +updateProfile() +resetPassword() +logout() +deleteUser()

Table: 5 Attributes description for User class

Attribute	Type	Visibility	Invariant
name:	String	Public	Name <> NULL and must contain first, middle and last name and shouldn't contain special characters and integers.
password	String	Private	password <> NULL must be at least 6 characters that combines digits, numbers
address	String	Public	address <> NULL and it must be between 12 to 20 characters
Email	String	Private	Email <> NULL ✓ Must contain @ ✓ Must contain. (dot) ✓ Position of @ >1 ✓ Position of (dot)>position of @ + 2 ✓ Position of (dot)+3<= total length of email address and the total character of the Email is at least 5 characters
status	ENUM	Public	Status <> NULL and it must be of on the state of "Active" or "Passive" with default value of "Passive"

Table: 6 Operation description for User class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition

createAccount()	Public	Void	.	The clients personal information shouldn't exist	The clients personal information should exist
activateUser()	Public	Void	.user: id	User that matches with the id shouldn't be activated yet	User account should be active
updateProfile()	Public	Void	User: id	User wants to update his account	User account should be updated
deleteUser()	Public	Void	User : id	Administrator wants to delete a user and user with the specified id should exist	User with the specified id shouldn't exist

4.2 Detail Design for Advertisement Class

Table: 7 Advertisement class

Advertisement
+name: String + CategoryId: integer +CampaignId: integer -path: String -id: integer -status : ENUM
+ regiaterAd() + activateAd() +updateAd() +customizeAd() +showPaymentStatus() +viewAdStatus() +viewAllAds() +clickAd()

Table: 8 Attributes description for Advertisement class

Attribute	Type	Visibility	Invariant
name:	String	Public	Name <> NULL and must contain first, middle and last name and shouldn't contain special characters and integers.
category_id	Integer	Public	Categoty_id <> NULL and it is foreign key mapping to category of advertisements
campaign_id	Integer	Public	Campaign_id <> NULL and it is foreign key mapping to campaign advertisements created by a specific advertiser
path	String	Public	Path <> NULL and must be working URL address that redirects to the origin of the advertisement
status	ENUM	Public	Status <> NULL and it must be of on the state of "Active" or "Passive" with default value of "Passive"

Table: 9 Operation description for Advertisement class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
registerAd()	Public	Void	.	Advertiser wants to add new advertisement	New advertisement should exist
activateAd()	Public	Void	.Ad: id	Ad that matches with the id shouldn't be activated yet	Ad should be active
updateAd()	Public	Void	Ad: id	Advertiser or owner of the Ad with the id number "id" wants to update the Ad	Ad should be updated
customizeAd()	Public	Void	User : id	Webmaster wants to customize Advertisements which displays in his page	Ads should be displayed on the webmaster's site based on the customization.

4.3 Detail Design for Storage Class

Table: 10 Storage class

Storage
-userId: String - adId: String -webmasterId:String -transactionId:String -transactionDate:DateTime +adClicks:integer +adImpression:integer +transactedMoney:Double +adlikedGeography:List<Double>

Table: 11 Attributes description for Storage class

Attribute	Type	Visibility	Invariant
userId	String	Private	userId <> NULL, is a foreign key must be referentially integrated with the user table ID
adId	String	Private	adId<> NULL, is a foreign key must be referentially integrated with the user table ID
transactionId	String	Private	transactionId<> NULL, is a foreign key must be referentially integrated with the user table ID
transactionDate	DateTime	Public	transactionDate <> NULL • Must be the exact time of when that transaction made • with now() insert property
adClicks:	Integer	Public	adClicks<> NULL
adImpression	Integer	Public	adImpression<> NULL
adLikedGeography	List<Double>	Public	adLikedGeography <> NULL • should be in longitude, latitude format
transactedMoney:	Double	Private	transactedMoney<> NULL • depends on adClicks and adImpression, or calculated using adClicks and adImpression

Table: 12 Operation description for Storage class

Operation	Visibility	Return type	Argument	Pre-Condition	Post Condition
getUser()	Public	User object	user : Id	user must login User should exist	-
getAd()	Public	Ad object	.Ad: id	user must login The Ad should exist	-
updateAdStatus()	Public	bool	• Ad: id • Integer:adImpression • Integer:adClicks • List<double> : adLikedGeography • Webmaster:id	The Ad should exist The Advertisement should be displayed on a publisher website The Advertisement should be Clicked to update the like status	Ad status should be updated Users(Advertiser and publisher) could see notifications The transaction money will be calculated
getAdStatus()	Public	AdStatus object	User : id	user must login The Ad should exist The Ad status should exist	-
updateAdTranaction()	public	bool	• Ad: id Ad:adClicks Ad:adImpression DateTime:transactionDate Webmaster:id	The Ad should exist Webmaster should be registered	Ad transaction should be updated Users(Advertiser and publisher) could see notifications

getAdTransaction()	public	AdTransaction Object	Webmaster:id Ad: id	user must login	-
updateAdTopic()	public	bool	Ad: id String:topic	The topic should be new	New Ad topic will be available. The Ads should categorized using the updated topic lists
getAdTopic()	public	string	Ad:id	user must login The Ad should exist	-

References

[1] A survey on "***AN ARCHITECTURE OF INTERACTIVE WEB ADVERTISING SYSTEM***" by Hsiangchu Lai and Tzyy-Ching Yang from Department of Information Management, National Sun Yat-sen University, Taiwan, R.O.C

[2] SDS document of Autistic Conversational Skills Software by Wei Lin

Web resource

[3] http://www.tutorialspoint.com/uml/index.htm at December 31, 2016