ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
COLLEGE OF NATURAL SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

ONLINE AMHARIC WORD PROCESSOR FOR MOBILE PHONES

BY

ABUBEKER YIMAM

A THESIS SUBMITTED TO
THE SCHOOL OF GRADUATE STUDIES OF ADDIS ABABA UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE
OF MASTER OF SCIENCE IN COMPUTER SCIENCE

March, 2013

Addis Ababa

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

COLLEGE OF NATURAL SCIENCES

DEPARTMENT OF COMPUTER SCIENCE

ONLINE AMHARIC WORD PROCESSOR FOR MOBILE PHONES

By: Abubeker Yimam Hussen

Approved by Board of Examiners:

Name	Signature
1. Dr. Dida Midekso, Advisor	_____
2. Dr. Solomon Atnafu, Examiner	_____
3. _____	_____

ACKNOWLEDGMENT

First of all, I would like to thank the Almighty God (Allah) His loving and kindness in bestowing me health, strength, patience and protection throughout my study. Second, I would like to sincerely thank my advisor, Dr. Dida Midekso, for his invaluable and stimulating interest and unreserved support, most generously given during this thesis work. I pay respect and express indebtedness to him because of his guidance, advice, consistent supervision as well as moral support throughout this work.

Also, I would like to sincerely thank all of questionnaire participants, friends and colleagues for their assistance, encouragement and inspiration during this thesis work.

Last, but not least, I thank the Computer Science Department of Addis Ababa University for hosting me to pursue my M.Sc. studies and Ministry of Education for providing me the financial assistance.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

API –Application Interface

AVKB – Amharic Virtual KeyBoard

CSS – Cascading Style Sheet

DDR - Device Description Repository

HTML – Hyper Text Markup Language

HTTP – Hyper Text Transfer Protocol

iOS – iPhone Operating System

IP – Internet Protocol

ISO-International Organization for Standardization

IT- Information Technology

LCD – Lowest Common Denominator

MIME - Multipurpose Internet Mail Extensions

mm - millimeter

MS - Micro-Soft

OBML - Opera Binary Markup Language

OLAWP – Online Amharic Word Processor for Mobile Phones

OMA – Open Mobile Alliance

OS – Operating System

PC – Personal Computer

PDA – Personal Digital Assistant

PHP - Hypertext Preprocessor

SMS – Short Message Service

S60 – Symbian 60

TCP – Transmission Control Protocol

WAN – Wide Area Network

WAP – Wireless Application Protocol

WCSS – Wireless Cascading Style Sheet

WML – Wireless Markup Language

WURFL – Wireless Universal Resource File

WWW - World Wide Web

W3C - World Wide Web Consortium

XHTML-MP – Extensible Hyper Text Markup Language Mobile Profile

XML – Extensible Markup Language

Abstract

Nowadays, accessing of web information and services at anytime from anywhere is becoming realistic using WAP enabled mobile devices. Mobile device users become rapidly growing from time to time. In this thesis, an online Amharic word processor for mobile phone is designed and developed by considering the physical screen size, scripting language, text entry method of mobile phones and nature of Amharic script. In order to design and implement the OLAWP, it is necessary to obtain the visiting mobile phone capability attributes information first. The attributes that are used in this study are mobile phone physical screen height and status about whether the mobile phone supports client-side scripting language or not. These capability attributes are obtained from the WURFL DDR in the cloud which is used to detect and return the mobile phone capabilities to our site. The returned values are stored and used as an input in the design of the word processor interface, to select the type of the virtual keyboard and style sheet used, and to decide the scripting language used. The OLAWP site is designed and implemented to enable users to prepare simple Amharic documents wherever they go as far as the WAP enabled mobile phones exist at their hand and the Internet infrastructure exist. In the design of OLAWP, the major activities are device detection and redirection to the compatible page, user and file manager, designing of the word processor interface by loading appropriate virtual keyboard. In this work, the mobile phones are categorized into three groups: small, medium and large, and the Amharic virtual keyboard is grouped into small and large based on the mobile phone screen height. Small mobile phones used small type of Amharic virtual keyboard and other groups used the large Amharic virtual keyboard. Finally, the OLAWP prototype evaluation is conducted through questionnaire by participating 15 different users in various types (in screen size, text entry method, browser and brand) of mobile phones. The evaluation result showed that better satisfaction and performance on some mobile phones which are touch based, and UC and Android browsers mobile phones compare to keypad based and Opera mini browser mobile phones.

Keywords: Amharic keyboard, Amharic word processor, mobile cloud service, mobile Amharic word processor, mobile virtual keyboard, online word processor, Wireless Application Protocol.

CHAPTER ONE
INTRODUCTION

1.1 Overview

Nowadays, Information Technology (IT) is growing rapidly and the users become interested to know and use it in order to make easy their day to day activities, and cope up with the time. In this fast growing of IT, the Internet and wireless technologies are playing a great role even though most of the technologies developed for the Internet have been designed for desktop and larger computers that support medium to high bandwidth connectivity on reliable data networks [1].

It is well known that wireless devices give opportunities to remain connected as we move from place to place. A huge amount of information is available on the Internet that mobile phone users will also need access to it. These desires of users are met by mobile devices and Internet infrastructure. Obtaining of information and services through Internet from anywhere at any time is an interest of everybody. Because getting accurate information and services are important elements for day to day activities for better decision making [2].

Recently, the mobile devices become widely available and rapidly growing in connecting people and transferring information as well. However, these devices have more constrained computing environment as compared to desktop computers. As we know wireless devices give opportunities to remain connected as we move from place to place.

Accessing web content and getting online services from anywhere at any time using wireless devices is possible due to the advent of WAP (Wireless Application Protocol) technology. It is thus; possible to efficiently communicate data over wireless WANs. WAP is a global standard that brings Internet content and services to wireless devices [3].

Mobile phones are the most widely used wireless devices and their growth is also very fast compared to personal computers. For instance, in Ethiopia, mobile users are highly increasing from time to time since the introduction of mobile technology. The Ethio-telecom reported that the number of mobile subscribers has exceeded 15 million [4]. According to Wireless

Intelligence, more than a billion of mobile phone connections were only added within 18 months and mobile phones become the most popular consumer devices in the world which is three times of the number of personal computers [5]. Also recent study shows that mobile devices will become the primary tool for Internet connectivity by 2020 [6].

The other concept that comes together with the development of Internet and network is cloud computing. Cloud computing is a model which is convenient, on-demand network access to a shared pool of configurable computing resources like storage, application, services, servers and networks which are provided through the Internet [7].

Cloud computing is the fundamental evolution of computing in which processing and data storage resources move away from user's desktop computers into large data centers to offer cloud services. Cloud services are applications, products and services that take advantage of cloud computing by hosting the processing and data storage resources in the cloud and making them available on demand. Now, high access bandwidth in both wired and wireless domains is available [8].

In our day to day activities, we commonly use software as a product, unlike software as a service. In cloud computing, only a thin client interface such as a web browser is needed on the client-side, all the remaining services (resources) exist on the service provider side. This is very helpful for devices that have low storage space and low processing capacity like mobile phones.

The combination of cloud based application and smart phones have provided extraordinary access of information for users from wherever they are. The coming of cloud based service is transforms the users' expectation to access information and increases the desire of obtaining of information and services at their fingertips. Now, most social network service providers have given this type of service which is accessible anywhere from various computing terminals. Cloud users can buy the service that they need rather than installing the whole IT infrastructure and application on their terminal. Instead of paying for the whole product, cloud customers pay only for what they use [9].

For the real implementation of cloud computing in the mobile device environment, mobile Internet technologies (i.e. WAP) play a great role. Since the mobile storage capacity is limited, so extending the storage facility through mobile online service and by which it will save mobile

devices battery consumption, in case of high computation, and save storage space of the device as long as the service and the data is provided and stored on remote server.

Cloud or online based services will overcome the problem of platform compatibility of software, like Amharic SMS software. The Amharic SMS messaging software is only available for particular mobile models specifically for Java program supporting mobile devices like Nokia and Android. So far, there is no mobile based Amharic word processor which enables users to prepare simple Amharic text even if most people have mobile phones on their hand.

Thus, the purpose of this concrete work is to design and implement Amharic word processor for mobile phones by taking into consideration the display size, script languages support, text entry techniques of mobile phones and the nature of Amharic script.

1.2 Statement of the Problem

There are large numbers of mobile users in Ethiopia and almost all of them use English language based mobile phones even if recently some mobile company has developed some Amharic script based mobile phones (for example, Nokia developed Nokia 1200, 1208, 2630 and 2760) [10]. There are also some Amharic SMS software that are developed for some limited type of mobile models (like Android, Nokia), but such applications are not applicable in other mobile phone platforms i.e. the application lacks compatibility among the phones.

A number of researches were conducted on wireless based systems due to the wide availability of wireless devices [11, 12, 13]. In [12] and [13], the researchers designed a virtual Amharic keyboard for PDAs that accept inputs with stylus. These devices are not used widely as the standard 12-key mobile phones. The 12-key mobile phones use their keypads, and virtual keyboard (which was developed together with the application) using the arrow keys for text entry. The Amharic text entry for wireless devices study in [11] was designed and developed for standard 12-key mobile phones but only incorporates 210 Amharic characters even if Unicode 6.1 incorporates around 358 Ethiopic characters and it is only applicable for Java program supported mobile phones. About 148 characters are reduced due to the mobile phones memory capacity limitation.

In Ethiopia, more than 15 million people use mobile phones [4] and they are either forced to go to computer centers or use their own computers when they need to prepare a short letter even if they have the mobile phones.

Also, most mobile Internet users didn't have an opportunity to write and post Amharic script text on different web forums, social networks, mails, blogs and other sites from their mobile phones. Currently, if a user wants to write Amharic text on the web from his/her mobile phone, s/he is forced to close the browser and go to the phone based Amharic SMS software (if it has) to write a text. After writing the text, copy it, open the site using the WAP browser and finally paste into the target site area.

Due to the above listed problems the researchers are motivated to design and develop an Amharic word processor for mobile phones that works on most mobile phone platforms.

1.3 Objectives

1.3.1 General Objective

The general objective of this research is to design and implement an Online Amharic Word Processor for Mobile Phones.

1.3.2 Specific Objectives

In order to achieve the above mentioned general objective, the following specific objectives are accomplished in this research work.

- Customize and develop online Amharic virtual keyboard.
- Develop an online Amharic word processor prototype.
- Test the prototype with various mobile phone platforms and screen sizes.

1.4 Scope and Limitation

The mobile phone based online Amharic word processor focuses on limited word processing functionalities like creating, formatting and saving text documents, and other functionalities like exchanging text documents/messages through email and SMS.

1.5 Methods of the Study

1.5.1 Literature Review

Before start the OLAWP, deep study is made in the literatures that are written related to the mobile phones, WAP technology, and text entry methods to have a clear picture about the work. Research works on these topic areas are reviewed to understand the nature of mobile phones, mobile web, nature of Amharic language and text entry techniques to consider while the OLAWP designed. Some cloud based word processing services of mobile devices are also reviewed under the related work section of this work.

1.5.2 Selection of Development Language and Tool

In this study, XHTML-MP web development language is applied, which is specifically designed to develop WAP applications for wireless devices by considering the capabilities of the mobile phones. XHTML-MP is a static web development language which is used to present the contents that are carried out by the dynamic web scripting languages. The dynamic part of the system is employed using client-side script (if the mobile phones support client-side scripting language, JavaScript) and PHP, server-side scripting languages. The XHTML-MP web mark-up language is used to implement the front side of the system and the back end (or the database) is implemented using MySQL database. Macromedia Dreamweaver is used as a development editor tool for the development of OLAWP prototype.

1.5.3 Testing of OLAWP

The OLAWP prototype is frequently tested using Nokia and Android emulators while the prototype is being developed. After the prototype is completed, it was tested on various mobile phone platforms, display sizes and display types (i.e. touch and non-touch screens) on the actual devices. To test the OLAWP prototype on the actual device, the OLAWP prototype was hosted on *http://www.3owl.com*, which is a free web hosting site. The OLAWP database also imported to this site.

Finally, the usability of the OLAWP has been conducted by considering different users and mobile phones. The responses about the usability were collected through questionnaires and analyzed.

1.6 Document Organization

This thesis report consists of Eight Chapters including the current Chapter. Chapter Two presents literature review for the understanding of basic concepts related to the nature of mobile devices, accessing of mobile web using WAP technology and its architecture, and text entry technique of mobile devices are reviewed. In Chapter Three, some cloud based word processing applications which are developed for Latin scripts are reviewed. The next Two Chapters, Chapter Four and Five, focus on the analysis and design of OLAWP by identifying the functional and non-functional requirements, and by taking into consideration deign goals and constraints of mobile phones respectively. Chapter Six presents system development tools like development editor, Apache web server, WURFL package, development markup and scripting languages, and emulators used in the development of the prototype. OLAWP prototype is also presented by considering sample screen shoots of the prototype. In Chapter Seven, the OLAWP prototype evaluation that conducted on the actual devices and the usability of the prototype evaluated based on the data collected from the users via questionnaire. Finally, Chapter Eight gives conclusion of this thesis work and points out future works related to this work.

CHAPTER TWO
LITERATURE REVIEW

2.1 Overview

Accessing web site through mobile devices becomes very common due to the advent of WAP technology. The capabilities of mobile devices have grown phenomenally from time to time. Currently, the number of mobile phones is more than thrice that of computers in the world and the mobile web is used as a means to bridge the digital divide, to bring the power of computer and Internet to the rest of the world [14]. Today, ubiquitous email and web access is a reality that is experienced by millions of users worldwide through their PDAs, smart phones and other mobile devices [15].

Under this chapter various literature studies related to mobile phone devices, WAP technology and text entry methods discussed to understand our work.

2.2 Literature Review

2.2.1 Mobile Devices

Mobile devices are commonly used for voice communication, but now the usage of mobile devices for data communications purpose is growing rapidly. Such data usage includes SMS, mobile web, video, location-sensitive applications, m-commerce, m-education, m-governance, m-health, m-bank and etc. These applications indicate that the mobile devices are being transformed from communication devices to content and transaction devices.

The mobile subscribers become increasing highly in our country, Ethiopia; also in the world. At the end of 2011, the numbers of mobile phone subscribers around the world were reached 6 billion. Among these, there are many mobile-only web users (around 1.1 billion) which do not (or very rarely) use a desktop, laptop or tablet to access the web [16]. The growth of mobile web penetration is strongest in Asia and Africa, where as the PC penetration is low [16]. In Ethiopia, the number has reached more than 15 million subscribers [4].

Wireless devices are produced in various designs for different purposes. Among them, mobile phones are the largest segment of wireless devices and have a specially designed processor and

operating systems like Symbian, Windows Phone, Android, Bada, iOS and so on. In the last few years, a variety of mobile phone is available in the market from simple feature phones to smart phones with sophisticated applications. Such devices are small in size and enabled users to carry with them everywhere they go. There are a number of different screen sizes of mobile devices; typically, they have screen sizes of 120x120 pixels up to 320x240 pixels. This means you can show about 6 lines of 25 characters displayed in a screen. They may have a QWERTY keyboard or stylus, or have just the numeric keypad for input. The standard input mechanism for mobile phones is a 12-keypad with additional function keys. Some phones have a joystick that can be used for navigation [14].

2.2.2 WAP Technology and Architecture

The WAP technology is the combination of the wireless technology and the Internet. The aim of WAP technology is providing a new service for the customers everywhere they go via their mobile devices. Such technology developments initiate some of the largest wireless vendors (such as Ericsson, Motorola, Nokia, and Openwave) to unite and create the WAP Forum, the standardizing organization of the WAP in June, 1997 [2]. Now, WAP Forum has consolidated into the Open Mobile Alliance (OMA). Open Mobile Alliance is a standard body which develops open standards for the mobile phone industry to facilitate global user adoption of mobile data services by specifying market driven mobile service enablers that ensure service interoperability across devices, geographies, service providers, operators, and networks while allowing businesses to compete through innovation and differentiation [17].

WAP is an open and global standard in order to bringing Internet content and services to wireless devices [3]. It provides an XML based language - WML and XHTML-MP which are used to develop WAP pages. WAP is designed to solve some of the problems caused when small, low powered devices on different platforms try to use low bandwidth wireless network technology to access services or data-intensive content via the Internet.

Most WAP enabled mobile phones can now access the Internet via their nearest Internet service providers (e.g. in Ethiopia, Ethio-Telecom). Mobile accessible websites make the users to browse the Internet anytime, anywhere and free them from the boundaries of the desktop and allow accomplishing tasks from anywhere.

The web has evolved greatly over the years. From the early days of simple HTML markup linking a few pages to the web 2.0 collaborative environment. The development of web 2.0 is increasing in availability of tools and services that are accessed directly through the browser rather than on the user desktop. Similarly, mobile web browsing technology is evolving from WAP 1.x to WAP 2.0, by introducing different enhancements for mobile content development. Especially, WAP 2.0 provides support for protocols such as IP, TCP and HTTP, which are designed for PC. This permits wireless devices to utilize existing Internet technologies. WAP 2.0 also provides different application environment, which enables delivery of information and interactive services to wireless devices [2].

Trying to browse the web on a phone, handheld computer, or personal digital assistant, you spend most of the sessions scrolling through the screen to find the content that you want. The current approaches to solve this problem either introduce new formats and mechanisms to make new web pages adaptive to different display sizes or attempt to automatically transform existing web pages into formats that can be browsed on mobile devices. There are many suggestions about mobile web application developments today. The two most common techniques to present web content in a wide variety of mobile phones are *adaptation* and *Lowest Common Denominator (LCD)* [18].

Adaptation (also known as multi-serving) is a means of delivering content as per the visiting device capabilities. Different techniques are used for adaptation; including *detection*, *redirection, setting correct MIME types, changing links, and removing or scaling graphics*. Detection is getting of information about the accessing device based on the user agent string while trying to access the website. Redirection is a technique of designing different version on web pages for each or group of mobile devices that fit with them. When the user requests a service, the system gets the user's device information and redirects to the page that is designed for that specific type (or group) of mobile device(s). Scaling is also another technique which applies different ratio on graphics and other contents of the web page based on the visiting device capability. The MIME type is the application type that the web server defines to be recognized by the web browsers of the client. Therefore, setting correct MIME type means choosing the MIME type dynamically for that specifically visiting device [14].

The LCD method establishes a minimum set of features expected from the device and develops content adhering to those limited features [14].

Now, the most common method of developing mobile web is using XHTML-MP (Extensible Hyper Text Markup Language Mobile Profile). XHTML-MP is the standard language for mobile web (or WAP application) development. It is built on top of XHTML Basic. W3C developed XHTML Basic originally for mobile devices but Open Mobile Alliance (OMA) added support for WAP CSS (WCSS) and other usability enhancements over XHTML Basic and defined it as XHTML-MP. Hence, XHTML-MP has been adopted as a standard by device manufacturers and supported by most mobile phones [14].

The WAP architecture has a gateway between the server and the client (WAP browser). The WAP is embedded in each WAP enabled wireless device in both sides; the mobile phone and WAP gateway. The server-side implements the other end of the protocol which is capable of communicating with any WAP client. The server-side is known as a WAP gateway and routes requests from the client to web server. The WAP gateway can be located either in the Internet service provider premises as shown in Figure 2.1 or in the WAP application provider premises with the web server as shown in Figure 2.2 [2].

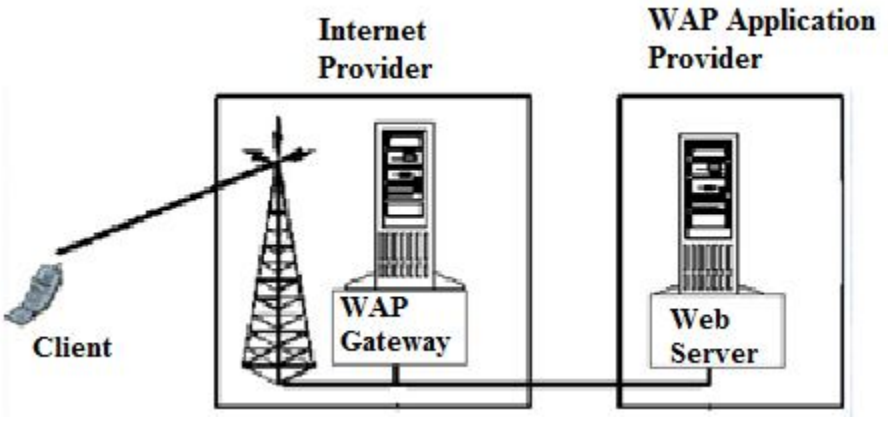


Figure 2.1: *WAP gateway in the Internet provider premises*

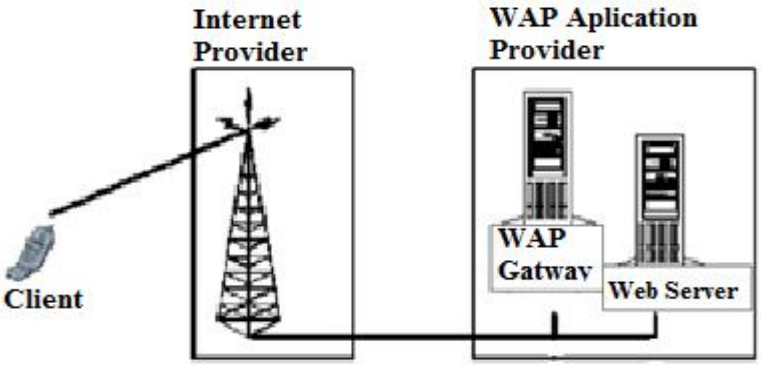


Figure 2.2: *WAP gateway in the WAP application provider premises*

As shown in Figure 2.3 [19], the WAP gateway acts as a proxy server for a mobile client and translates WAP requests to WWW requests, so the WAP client is able to submit requests to the web server. The encoders translate the content coming from the server into compact formats to reduce the size of data over the wireless network and understandable by the WAP client [19]. The decoders also translate the content coming from the client micro-browser into the format of the web server understands. The wireless application environment provides WAP micro-browser for interaction between WAP and wireless devices. This browser relies on WAP markup languages such as WML and XHTML-MP.

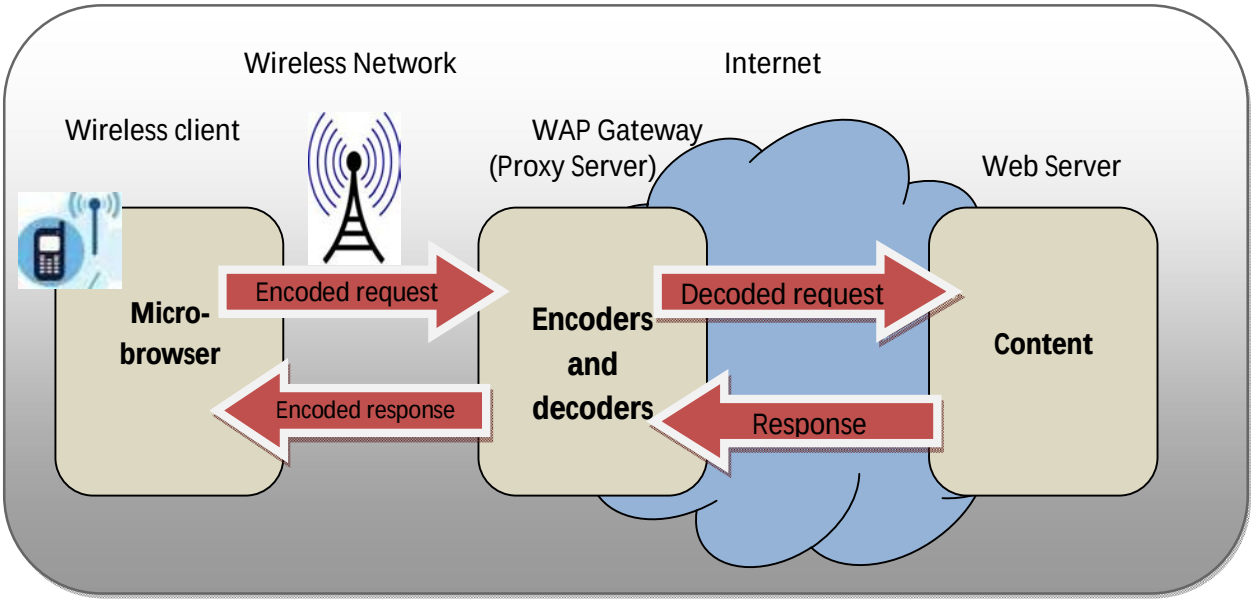


Figure 2.3: *WAP basic architecture*

This WAP infrastructure ensures that mobile users can access a wide variety of contents and applications, and allows application developers to build web services and applications that can run on a large base of mobile terminals [19].

2.2.3 Mobile Phones Text Entry

Commonly, mobile phones used either their keypad or virtual keyboards for text entry purpose. Keypad based text entry device, like the standard 12-key mobile phones, each key represents three to four characters in Latin scripts. In Amharic script, each key represents three to four basic characters even if each base character has a minimum of seven additional different non-basic characters. For instance, Amharic base character "ሀ" (pronounced as "**ha**") has seven non-basic character forms (ሁ, ሂ, ሃ, ሄ, ህ, ሆ, ሇ).

In Unicode standard 6.1 [20], Ethiopic script incorporates 358 number of characters with the range of 1200-137F. Appendix A shows the Unicode standard 6.1 of Ethiopic characters. Incorporating and mapping all these characters on the 12-standard keypad mobile phone is difficult. The study in [11] designed and developed keypad representation for Amharic script. But, the study didn't incorporate all Amharic characters. Instead, it eliminated superfluous (redundant) characters, replaceable characters, Ethiopic numerals, and labelized characters due to the memory capacity limitation of the mobile phones [11].

Text entry can be performed either using physical keyboard or virtual keyboard. These two types of text entry techniques are discussed in the following sub sections.

A. Physical Keyboard

The physical keyboard size can be reduced in two ways. One way is to shrink the size of each key. Typing on these keyboards is slow and difficult due to their reduced size. The other way is to share each physical button with multiple letters as the standard 12-key mobile phones. There are a number of text entry techniques developed by different researchers in order to simplify text entry. Some of these are less-tap [21], multi-tap [11], two-key (for Latin scripts) [11] and three-key (for Amharic scripts) [11] input methods.

Less-Tap: The less-tap text entry study [21], for English language script, was designed based on the frequency of the characters. In this method, the most frequent characters can only be typed using one key press. In English keypad, only 3 or 4 keys are found in one button. As shown in Figure 2.4, the characters are arranged based on the frequency of the characters on each key. For instance, key number '**3**' displays "**edf**" character arrangement instead of the common "**def**" arrangement. In this case, character "e" is more frequent character than characters "d" and "f", and character "d" is more frequent than character "f".

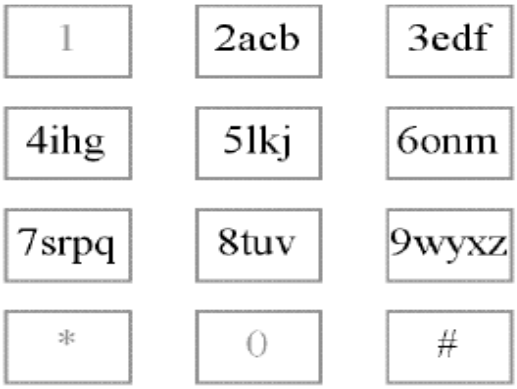


Figure 2.4: *Less-tap 12-key keypad*

Multi-tap: This requires the user to press buttons repeatedly to get the required character. This is commonly found in most 12-key mobile phones and the arrangement follows the natural English alphabetical letter as shown in Figure 2.5. In Amharic script, a multi-press requires the user to press two or more keys to compose a single Amharic character [11]. Researchers implemented this method in two approaches. In the first approach, when one of the base characters (e.g. "ሀ") is pressed from the keypad a menu displays the seven non-basic characters (e.g. ሀ, ሁ, ሂ, ሃ, ሄ, ህ, ሆ) that enable the user to select one of the non-basic characters. Keypad arrow keys can be used to navigate and select a key among the seven forms of the basic characters. In the second approach of multi-press input method, the keypad alone is used without selecting the appropriate order from the top right most corner of the screen. For instance, to write the name **"ከበደ (kebede)"**, the user presses the key labeled '**5**' twice and the key labeled '**3**' twice to write letter **"ከ (ke)"**. Similarly, the user again taps the key labeled '**2**' twice and the key labeled '3' twice to enter letter **"በ (be)"**. Finally, the user again taps the key labeled "**3**" once and the key labeled "**3**"

twice to write the character **“ደ (de)”**. This results in a total of 11 key taps to compose the name “ከበደ” (**Kebede**) [11].

Figure 2.5: *Standard 12-key keypad*

Two-key Input Method: this method is used for Latin scripts and requires two successive key presses to specify a character. The first key press, like the multi-press method, selects the group of characters (e.g. key for ‘**WXYZ**’). The second press is used for disambiguation. One of the number keys 1, 2, 3, or 4, is pressed to identify the position of the character within the group. For example to enter the character ‘**Z**’, the user presses ‘**9**’ and ‘**4**’ (‘Z’ is fourth character in ‘WXYZ’). Each character A-Z is entered with exactly two key presses [11].

Three-Key Input Method [11]: this method is used in Amharic scripts similar to the two-key input method of Latin scripts. The first key press is used to select the group of characters (e.g. “**ሰረሸ**”). For instance, pressing the key labeled ‘**3**’ selects the character group “**ሰ, ረ, ሸ**”. Then pressing the key ‘**3**’ corresponds to selecting the third character from the selected group. In this case, “**ሸ**” (pronounced as ‘**she’)** is the third letter in the group. Then, the user presses one of the keys labeled ‘1’, ‘2’, ‘3’, ‘4’, ‘5’, ‘6’ or ‘7’, to select the order of the character selected in the previous key press. Figure 2.6 shows that the alphabetical arrangement of keypad mapping for Ethiopic scripts by considering a total of 210 characters [11].

Figure 2.6: *Ethiopic keypad based on alphabetical arrangement*

B. Virtual Keyboard

A virtual keyboard is a software component that allows a user to enter characters into an input box area. A virtual keyboard can usually be operated with multiple input devices, which may include a touch screen, an actual keyboard and a computer mouse or mobile phone arrow keys [22]. However, physical keyboard, unlike virtual keyboard, requires connectivity mechanism with the PDA using either wireless or a cable. This indicates that virtual keyboard is the best alternative for text entry on PDA environment [13]. Though virtual keyboards are available for many languages, there has been little effort to design and develop virtual keyboard for Amharic script [12, 13]. In a virtual keyboard, the character keys are displayed on the screen of the mobile devices and tapped with a stylus or a finger or through a pointing device. The most common Latin script based virtual keyboard is the QWERTY layout which is used in PC and smart phones [24]. For Amharic scripts, there is a "**ሀለሐ**" virtual keyboard layout designed for Android phones [12] (*Figure 2.7*) and AVKB (Amharic Virtual KeyBoard) [13] (*Figure 2.8*). The AVKB virtual keyboard (also called "**የመነ**" virtual keyboard) is designed by considering the character frequency occurrence of the Amharic scripts [25], which is analogue to QWERTY English keyboard layout.

Figure 2.7: *The "ሀለሐ" virtual keyboard*

Figure 2.8: *The Amharic Virtual KeyBoard (AVKB)*

CHAPTER THREE
RELATED WORK

In this chapter, some previous works which are related to our work will be presented. The related works focused on cloud based mobile phone Amharic word processor and existing virtual based Amharic text entry methods.

The Internet and wireless technologies are growing fast and they provide information and service almost everywhere you go. The common desktop application that we know like word processor and spreadsheet documents have become available at your computing device (like mobile phone) from the cloud to access, edit and share them the way to go.

Nowadays, cloud-based service becomes common and widely available for users. Google, Microsoft, Amazon and other companies provide many cloud services. For instance, Google Docs and Zoho writer provide document editing service for specific types of smart mobile devices.

Google Docs is one of the known cloud based services for Internet users to prepare documents without any application software. Google Docs mobile is developed for some specific mobile phones in order to edit and share a document. Depending on the mobile devices and the operating system used, you may access the Google Docs through your mobile browser to view, edit and share documents. Android 2.2+ and iPhone which used iOS 3.0+ versions phones can edit a document through the Google Docs cloud service. Other mobile phones which support HTML like BlackBerry, Window Phones and Nokia S60 can only view Google documents. The Google Docs support different foreign languages like Arabic, Bulgarian, Chinese, English, French and so on, based on the languages setting of Google Docs that you set on the PC [26].

Zoho writer is a free to use online word processing application with offline and synchronizing functionalities. It is one of the cloud applications among other Zoho mobile application. Zoho writer for mobile is developed and used for Internet explorer browser based mobile phones which run on the window mobile OS [27]. It supports iPhone, Android and BlackBerry, Windows Phone and Symbian 60 (S60) platforms [28].

The other common application for document editor in mobile phones is Document To Go [29]. It is a suite of applications (Word To Go, Sheet To Go and Slideshow To Go) with one application for each support MS Office format: Word To Go support MS-Word, Sheet To Go support MS-Excel and Slideshow To Go support MS-PowerPoint. In this application the document creation and editing process are performed on the phone and can be stored on the phone memory or send the file as email attachment. The document to be edited may be sent as an email attachment and the Document To Go accessed the email message that contains attachments in order to edit. Finally, the document can be sent either from within a Document To Go application as a new message or attached to an existing message [29].

The above mobile based document editors developed for specific type of mobile phone models (commonly, smart phones). Smart phones used virtual keyboard as a text entry method and the documents editors designed and developed for such type of devices. So far there is no mobile based Amharic text editor reported, even if some works are done on text entry method for Android smart phones [12, 13]. The researchers designed and developed a virtual based text entry method only which can be applicable for Android smart phones. The "ሀለሐ" and "የመነ" virtual keyboard for Amharic scripts were designed and developed to help localized mobile phone applications developments. On Android mobile phones, they can be used to prepare an Amharic text by switching the input method from mobile phone keypad to Amharic keyboard as shown in Figure 3.1. Figure 3.2 shows the "ሀለሐ" Amharic virtual keyboard on the actual mobile phone which is used as a text entry method, but only compatible with Android OS based mobile phones.

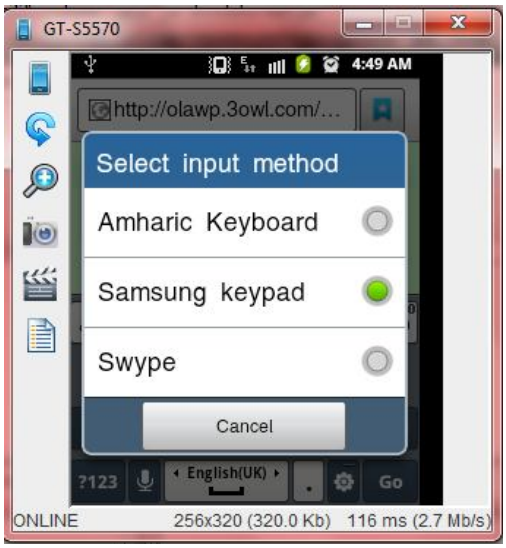


Figure 3.1: *Keyboard selection*

Figure 3.2: *"ሀለሐ" Amharic virtual keyboard for text entry*

The above described mobile phone based word document editors are designed and developed for specific type of mobile models (that is smart phones) and browser type even if they provide many functionalities. Also, they commonly support Latin based scripting language only. Moreover, they commonly used the built-in virtual keyboard of the mobile phone as a text entry even if many mobile phones used their keypad as a text entry. Finally, a phone based "ሀለሐ" and "የመነ" Amharic virtual keyboards were designed and developed for Amharic text entry purpose which only run on Android based smart phones.

But, in our work, the researchers tried to design and develop an online word processor for mobile phones which can be accessible from any WAP enable mobile phones either touch based or keypad text entry mobile phones. The OLAWP is designed and developed to support Amharic language as a primary goal even it also supports English language. It designed and developed a virtual keyboard, which is integrated with the OLAWP, based on the screen size of the mobile phones.

In this work, the virtual keyboards is designed by modifying the previous adopted virtual keyboards by adding some keys, like space and enter keys, and rearranging the keyboard layout to be applicable for both touch and keypad based mobile phones. The virtual keyboard modification discussed in Section 5.3.4 in detail.

CHAPTER FOUR

SYSTEM ANALYSIS

In this chapter, the proposed system will be described, the functional and non-functional requirement of the OLAWP system will be identified and its functionalities will be described by use case diagram, and the static and dynamic part of the system will be analyzed using class diagram and sequence diagrams respectively.

4.1 Proposed System

4.1.1 Overview of the System

OLAWP system is aimed to enable users to write simple text using their own mobile phone in Amharic script based on the online service. OLAWP is a web based online service which expected to be accessible for most types of mobile phones in different format based on the size of the screen and type of the script language that the device supports. The system will be designed to be functional for most WAP enabled mobile phones.

The user can enter Amharic characters using the keypad arrows keys and enter (ok) key for non-touch display mobile phones, pointing button, or stylus (finger) for touch display mobile phones through pressing the displayed button on the screen. After preparing the document the user can save it in the cloud, also can send to his/her own email or other friends either as email or SMS. Figure 4.1 shows the general work flow of the proposed OLAWP system.

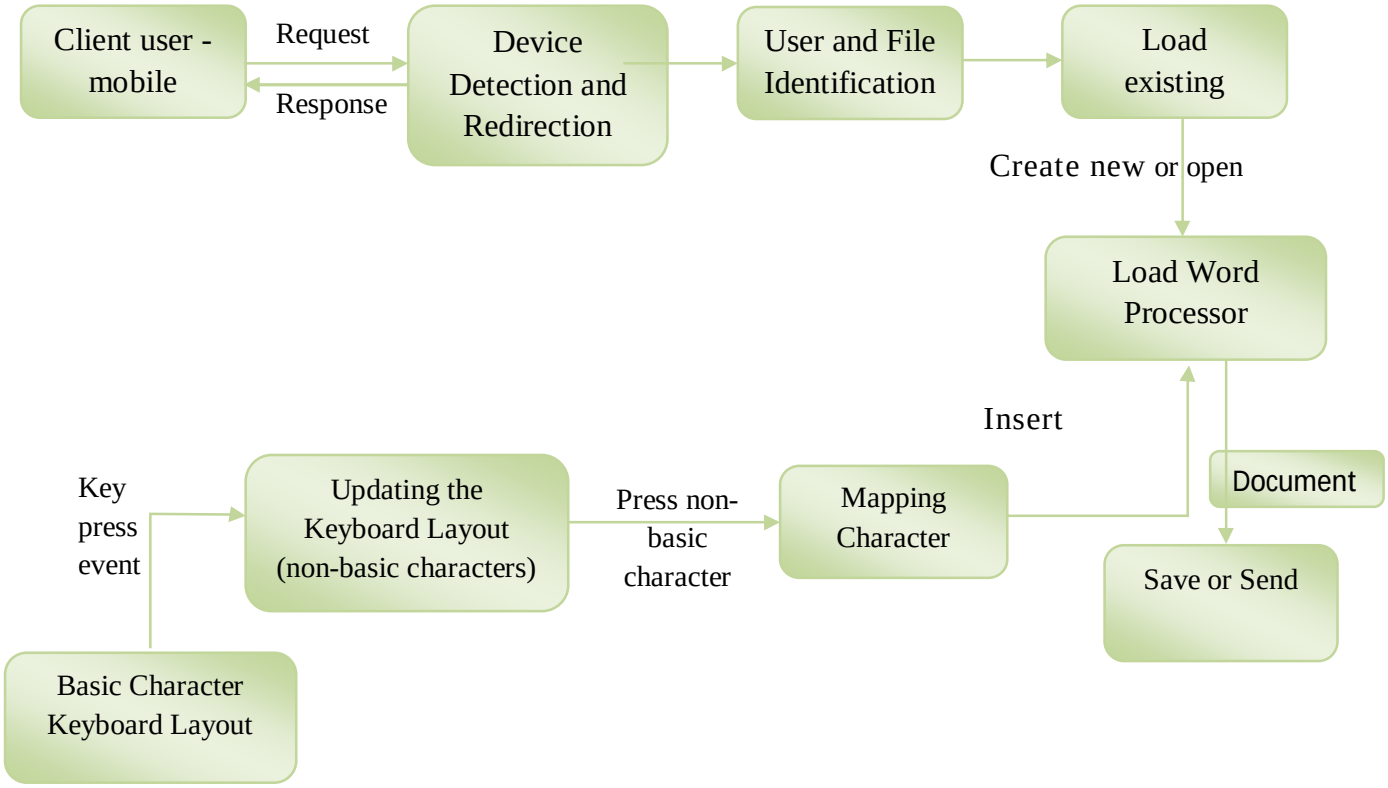


Figure 4.1: *Work flow of OLAWP system*

As can be shown in Figure 4.1, the OLAWP system detects the accessing device information using the device detection and redirection component which information helps to load limited number of files on the screen, to specify the size of the text box area, to decide the type of virtual keyboard used and to select appropriate style sheet. Then, user authentication and retrieval of documents process using user and file manager component. If files exist on the logged in user, the files displayed as a list on the screen. Later on, the users can do what they want either open existing one for viewing, or editing or creating new documents. When a document opened, the Amharic virtual keyboard is loaded (displayed on the screen) in order to enable the user to write Amharic characters. The text box area enables the user to apply different formatting styles like font style, color and size, and text alignment. Finally, the user either can save for future use or can send as a message or email. Actually the text entry part of the work flow diagram adopted from previous work [13].

4.1.2 Functional Requirements

The OLAWP system provides the following functionalities:

- The system should allow the user to type the Amharic characters using online Amharic virtual keyboard on the OLAWP site.
- The system should also allow the user to enter English text when s/he needs.
- The system should allow the user to store documents on the cloud (remote server).
- The system should allow the user to format documents, like font style, text alignment, font size and color, and delete and insert characters from/to the text.
- The system should allow the user to send the message either as email or as SMS message.

Table 4.1 shows the OLAWP system use case by categorizing into two:

Table 4.1: *List of OLAWP use cases*

Class of use cases	Use cases
Use cases related to user	Registration
	Login
	Create File
	Format Document
	View File
	Delete File
	Save Document
	Send Document
	Insert Character
	Delete Character
	Load Virtual keyboard
	Switch Keyboard Layout
Use case related to Device detector system (WURFL Cloud)	Device Detection

4.1.3 Non-Functional Requirements

The non-functional requirements that are expected from the system are:

- The system must be compatible on various mobile phones.
- Provide different arrangement of keyboard layout for novice and experienced users.
- The OLAWP site should be available throughout the year.
- The system must be easy to learn and use.
- The authorization mechanism of the system will block the unwanted attempts to access users' document or the server.

4.1.4 Constraints

- OLAWP requires an Internet connection to give service.
- The server application shall be available 365 days and 24 hours.

4.1.5 Assumptions and Dependencies

- OLAWP is dependent on the availability of Internet connection.
- OLAWP needs a web server to host.
- The user must have WAP enabled mobile phone to get the service.
- The user must have the ability to use mobile Internet.
- The accuracy of the information is the responsibility of the users.

4.2 System Models

In this section, the OLAWP system is described using use case diagram in order to show its functionality provided by the system as shown in Figure 4.2. Use case focuses on the external behavior of the system. Moreover, the dynamic and static behaviors of the OLAWP system is described using sequence diagrams and class diagram respectively.

4.2.1 Use Case Diagram

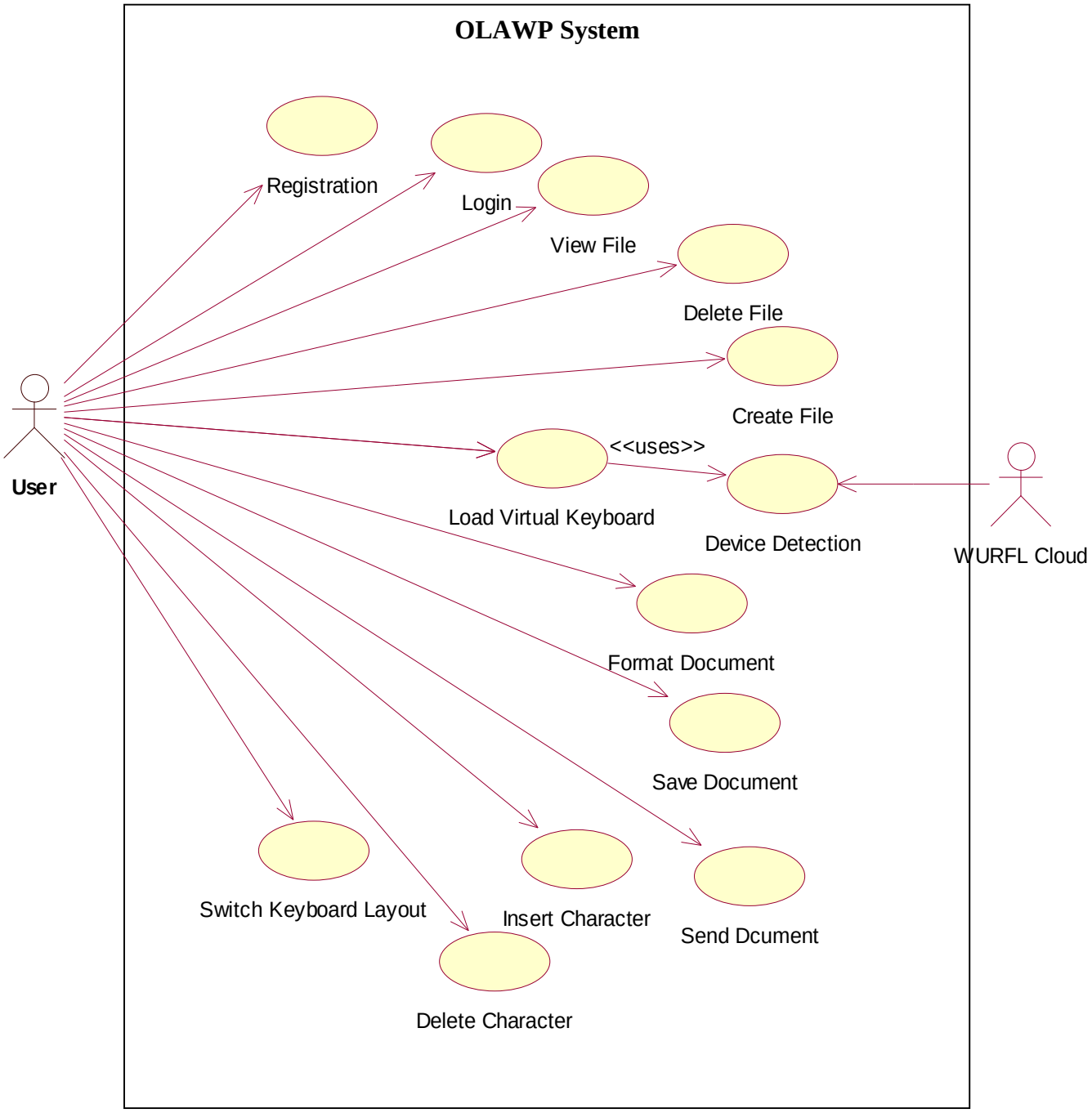


Figure 4.2: *Use case diagram of the OLAWP System*

4.2.2 Use Case Description

The following subsequent tables describe each use case in the OLAWP system.

Table 4.2: *Use case description for Registration*

Use Case 1*:Registration*
Primary actor*: User (New)*
Pre Condition: *None*
Main scenario: 1. *User browses OLAWP system site.* 2. *User clicks the Register button* 3. *System loads the registration form* 4. *User fills the registration form and submit the form* 5. *System validates filled data* 6. *System registers the user's data on the database* 7. *System sends confirmation message for the user* ***Alternate Scenario:*** 5. *When user does not complete all the necessary fields or invalid data entered* a. *System redisplays the form with an error message* b. *User refills the invalid input fields and submit the form*
Post condition: *User becomes member and can use the system*

Table 4.3: *Use case description for Login*

Use Case 2*: Login*
Primary actor*: User*
Pre Condition: *Registered user*
Main scenario: 1. *User browses OLAWP system site.* 2. *Once the page loaded, user should enter user code and password, and submit* 3. *System validates and forward to user's personal page* ***Alternate Scenario:*** 3. *When user enters invalid user code or password* a. *System displays error message and invite the user to try again* b. *user presses retry again link* c. *System redisplays the login form* d. *User fills the input fields and submit*
Post condition: *User's page opened*

Table 4.4: *Use case description for Create File*

Use Case 3*: Create Document (File)*
Primary actor*: User*
Pre Condition: *logged in*
Main scenario: 1. *Click on "New" button* 2. *Text area box will be ready for writing* 3. *User can write a text using virtual keyboard* 4. *User can save or/and send the document*
Post condition: *File (document) will be created*

Table 4.5: *Use case description for Format Document*

Use Case 4*: Format Document*
Primary actor*: User*
Pre Condition: *logged in*
Main scenario: 1. *List of documents displayed* 2. *Click on the document that you want to format (or a new document)* 3. *System opens the document (or a new document)* 4. *Format the document as you want (font color, size and alignment)* 5. *Save/update the document*
Post condition: *The file(document) will be updated*

Table 4.6: *Use case description for View Document*

Use Case 5*: View Document(File)*
Primary actor*: User*
Pre Condition: *logged in*
Main scenario: 1. *System checks the screen size of the device* 2. *Display/Load limited number of existing files based on the screen size* 3. *If the number of files are large, a group of files only displayed and can use the "Next" button to view other files* 4. *Click on the file(document) that you want to open*
Post condition: *The document will be viewed on a word processor*

Table 4.7: *Use case description for Delete Document*

Use Case 6*: Delete Document*
Primary actor*: User*
Pre Condition: *logged in*
Main scenario: 1. *List of documents displayed* 2. *Select the document(s) that you want to delete* 3. *Click on delete button* 4. *Confirmation message displayed (Yes or No)* 5. *Click on Yes* ***Alternate Scenario:*** 5. *When user clicked No* a. *The system cancel document deletion*
Post Condition: *List of existing document will be updated*

Table 4.8: *Use case description for Save Document*

Use Case 7*: Save Document*
Primary actor*: User*
Pre Condition: *logged in*
Main scenario: 1. *Open new text box area* 2. *Write a text that you want* 3. *Click on "Save" to save the document on server* 4. *File name box displayed* 5. *Type file(document) name* 6. *Click on "Save" button* 7. *Confirmation message displayed* ***Alternate Scenario:*** 6. *If file name existed before* a. *"file already exist" message displayed* b. *System requests the user override or not (Yes or No)* c. *If " Yes" the existing file replaced* ***Alternate Scenario:*** *6.c) if user click "No"* *i) the system redisplay file name box*
Post condition: *Document stored at the server for future use*

Table 4.9: *Use case description for Send Message*

Use Case 8*: Send Document*
Primary actor*: User*
Pre Condition: *logged in*
Main scenario: 1. *List of documents displayed* 2. *Open(create new) document that you want to send* 3. *Click on "Send" button* 4. *Enter recipient address* 5. *Click "Send" button* 6. *Confirmation message displayed* ***Alternative scenario:*** 5. *Incorrect recipient address* a. *Display error message* b. *User again should enter the address*
Post condition: *Document should be delivered to recipient(s)*

Table 4.10: *Use case description for Insert Character*

Use Case 9*: Insert Character*
Primary actor*: User*
Pre Condition: *Text area box and keyboard layout should be loaded*
Main scenario: 1. *User clicks on the base character* 2. *System displays all the non-basic families of the pressed character key* 3. *User presses the required character key to be inserted* 4. *System appends the mapped character to text area box*
Post Condition: *The inserted character will be displayed in the text area box.*

Table 4.11: *Use case description for Delete Character*

Use Case 10*: Delete Character*
Primary actor*: User*
Pre Condition: *At least one character in the text box area exist*
Main scenario: 1. *User presses the backspace button (<-) from base character layout panel.* 2. *System removes the last character from the content of the text box area*
Post Condition: *Text box area will update and show its content with removed character*

Table 4.12: *Use case description for Load Virtual Keyboard*

Use Case 11*: Load Virtual Keyboard*
Primary actor*: User*
Pre Condition: *Open text area box either for existing or new document*
Main scenario: 1. *If the screen size of the device is medium or large* 2. *System will load the "ሀሸሐ" Amharic virtual keyboard layout by default* ***Alternate Scenario:*** 1. *If the device screen size is small* a. *System will load first part of small Amharic virtual keyboard layout by default* b. *When user click next(>) button the second half of the keyboard layout will be loaded*
Post condition: *The Amharic virtual keyboard will be displayed on the Screen*

Table 4.13: *Use case description for Switch Keyboard Layout*

Use Case 12*: Switch Keyboard Layout*
Primary actor*: User*
Pre Condition: *Open text area box either for existing or new document*
Main scenario: 1. *If the mobile phone is medium or large screen size* 2. *If the system is currently displaying the "**ሀለሐ**"Amharic layout then pressing the button **"የ**", it will change the layout to "**የመነ**" layout.* 3. *If the system is currently displaying the **"የመነ**" layout then pressing the button "**ሀ**", it will change the layout to Amharic layout.* 4. *When the user wants to type English text press "**E**" button to switch to English layout* 5. *System loads upper case based English virtual keyboard* 6. *To enter lower case English letter, press "**e**" button* 7. *System loads lower case based English virtual keyboard*
Post condition: *The layout is changed either from "**ሀለሐ**" to "**የመነ**" or "from "**የመነ**" to "**ሀለሐ**", or English upper or small keyboard layout.*

Table 4.14: *Use case description for Device Detection*

Use Case 13*: Device Detection*
Primary actor*: WURFL Cloud*
Pre Condition: *logged in*
Main scenario: 1. *When user browse OLAWP site, the user agent information sent to WURFL cloud* 2. *WURFL cloud retrieves mobile phone capability information (screen height and weather the device support JavaScript or not)* 3. *The WURFL cloud returns detection values* 4. *OLAWP stores screen height and JavaScript status values in session* 5. *If the mobile phone support JavaScript* 6. *System forwards to appropriate page which designed using JavaScript* 7. *The text box area designed based on the screen height and appropriate virtual keyboard will be chosen* ***Alternate Scenario:*** 5. *If the device is not support JavaScript* a. *System forwards to another page which designed using PHP*
Post condition: *Compatible or appropriate word processor page loaded*

4.2.3 Sequence Diagram

It is a type of interaction diagram, which is used to illustrate the dynamic behavior of the system using a sequence of diagrams which emphasizes the time ordering of the messages. A sequence diagram shows a set of objects and the messages sent and received by these objects in a given use case. The above identified use cases are described subsequently using sequence diagrams in Appendix B from Figure 4.3 to Figure 4.15.

4.2.4 Class Diagram

Class diagram is used to identify domain objects or conceptual classes, relationship between conceptual classes, attributes and operations. It is used to model static design view of the system using a collection of vertices and arcs. Figure 4.16 depicts the class diagram for the set of classes that are identified in our system.

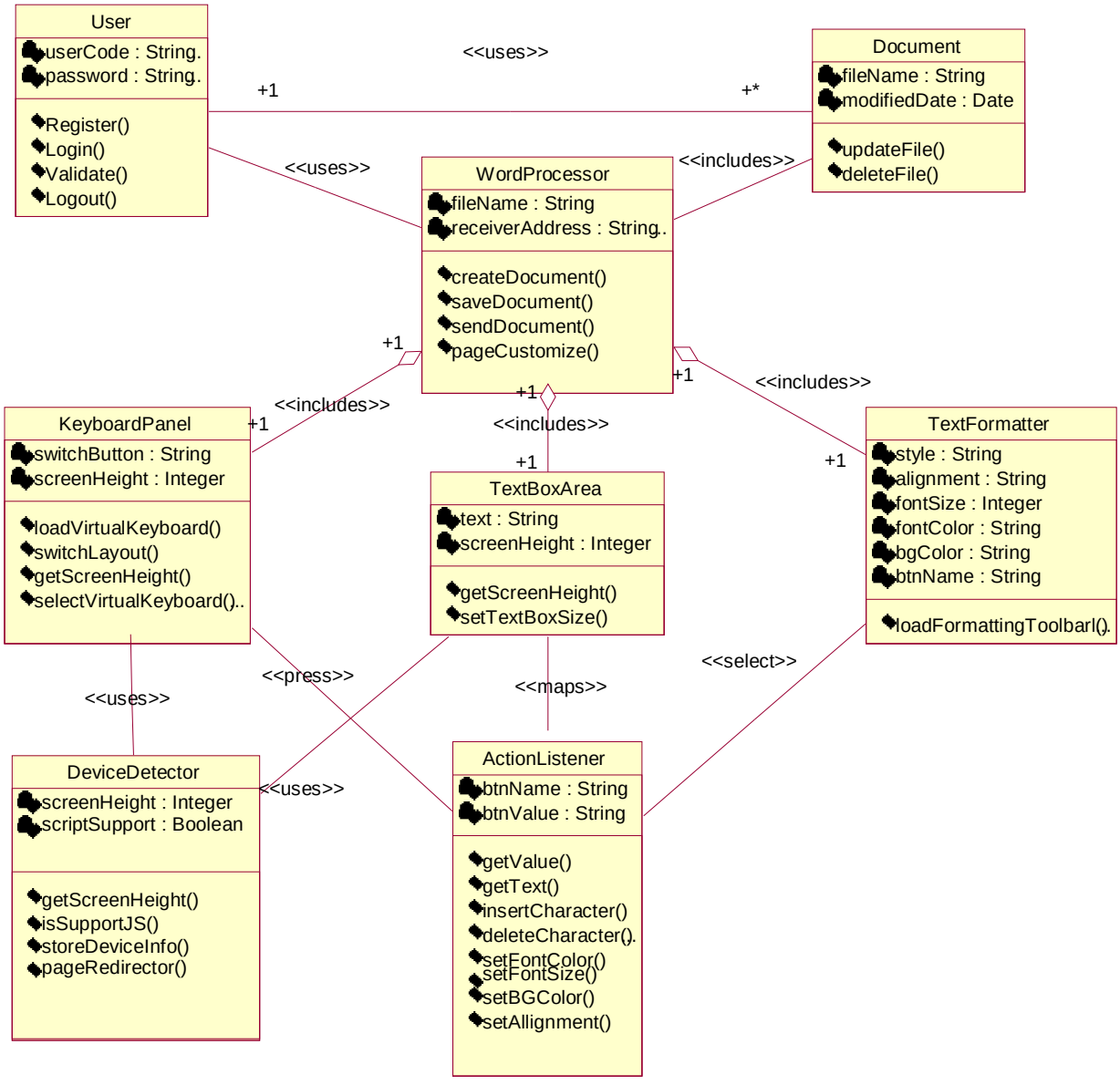


Figure 4.3: *Class diagram*

CHAPTER FIVE

SYSTEM DESIGN

Under this chapter the architecture of the OLAWP system is discussed by setting the design goals, considering mobile phone constraints and nature of Amharic script with the selected approach to solve these constraints, and it discussed in detail by decomposing the OLAWP system into manageable components.

5.1 Design Goals

The OLAWP web based system is designed to allow anybody to prepare simple Amharic text using their mobile phones. The design goals represent the desired qualities that the OLAWP should considered when making design decisions. In the design of OLAWP the following quality attributes area considered.

- **Compatibility:** the OLAWP system is an online based to make the service compatible for most WAP enabled mobile phones. As far as the mobile phones used a web browser, it is accessible from any mobile phones by considering the size of the screen to be compatible or fit with the visiting mobile phones.
- **Security:** is an authorization mechanism of the system which will block the unwanted attempts to the server and also let the system decided on which privilege may the user has on the document. Security is an importance issue for cloud based application as far the service accessible by any user without place and time boundary. Therefore, OLAWP only accessible to authorized users. Moreover, the data stored in the server should be kept securely.
- **Efficiency:** users are not interested to wait till the OLAWP page loads from the server. So, it is necessary to avoid graphics while designing user interfaces and minimizing the size of the files in order to minimize the response time. Furthermore, to avoid the interaction delay between the client terminal and the server for very common frequent actions (like formatting and typing), a client-side script language should be necessarily applied for devices which support client-side scripting language.

- **Availability:** the availability of the OLAWP system is up to the Internet connection of the client. Since, it is a cloud based service; the OLAWP web site shall be attainable all the times. The OLAWP server should be available all over the year.
- **Usability:** The OLAWP should be developed to be easy for users understanding. In order to be usable for most users; it is necessarily design the interface on their own language, Amharic. Furthermore, for Internet based service, the user should obtain a service with a minimum number of interactions because the interaction between the client and the server takes some time to process for each user action.

Moreover, mobile phones constraints, such as display size, script language and text entry method should be necessarily considered in the design. Also, the nature of the Amharic script should be taken into consideration when an Amharic based mobile service designed. In this work, the following mobile phones and nature of Amharic script challenges should be taken into consideration to achieve the design of OLAWP.

- **Display Size:** Different mobile phones come in various display size which makes the development of mobile phone based web application more challenging. For instance, the Amharic virtual keyboard is not displayed the same way for device that has large display and small display size. For large display size mobile phones, all the basic characters can be displayed at once, where as in small display size mobile phones, some of the basic characters are displayed at a time.
- **Nature of Amharic Script:** According to Unicode standard version 6.1 [20], Ethiopic script consists of 358 characters including all basic and non-basic characters, Amharic digits, punctuation marks and labelized characters [like ቐ, ሏ, ሟ, ሯ, ሷ, ኳ, ዅ, ሯ,ሻ, ሿ, …]. Due to the abundance of the characters, mapping all these characters on the mobile phone keypad buttons and/or on the virtual keyboard at a time is difficult and requires more memory space.
- **Text Entry Technique:** according to screen type, the text entry may be carried out either keypad or stylus (or finger) based. Text entry technique is difficult for non-Latin script (like Amharic) web based systems because when the users click on the input box area, the built-in phone's text composer is automatically opened or loaded. This composer is directly integrated to the built-in languages of the mobile phones.

- **Scripting Language:** Mobile devices are used their WAP browser to access information and services. Based on the client-side scripting language of the mobile phones browser that support, the mobile phones categorized into two; devices which do support client-side scripting language and which do not.

In order to overcome the above listed constraints, different techniques are used in the design of OLAWP architecture. In this work adaptation method is chosen, which includes different techniques such as detection, redirection and scaling as discussed before in Section 2.2.2. Detection is applied to obtain the visiting mobile phone capability information from the WURFL cloud DDR. Based on this result, the redirection and scaling technique applied on the OLAWP system to overcome the display size and script language variations, and applied the style sheet which is appropriate to the visiting device. Two different types of OLAWP pages are designed to treat mobile phones which support client-side scripting language and which do not support it (which will be discussed more in Section 5.3).

Virtual keyboard is chosen as a text entry method which is designed and developed by integrates with the OLAWP page in order to handle all the Amharic scripts in different phases. Rather than loading all the Amharic character mapping operations into the mobile phones, it only loads part of characters as needed from the remote server.

Moreover, a cloud based service selected on the infrastructure of WAP to minimize the compatibility problem among mobile phones even though there is a variation on the browsers. The XHTML-MP markup language is used to implement the WAP based prototype, because it is the current standard markup language for wireless devices and most devices support it.

5.2 Architecture of the System

The OLAWP system is an online web based system where the data exist at the server-side and the client access it through the mobile browser. For such system a layer design pattern is well suited to implement. The OLAWP web based service comprised of three layers as shown in Figure 5.1 and discussed them as follows:

Presentation Layer: it is a top layer which provides a browser based user interface which supports for the interaction between the user and the system. The user can obtain OLAWP service from the web through the mobile browser (also called micro-browser), which is accessible for all registered users. The page will be loaded based on the scripting language that the mobile phone supported, and the screen size in order to fit the screen and to make easily accessible by the users.

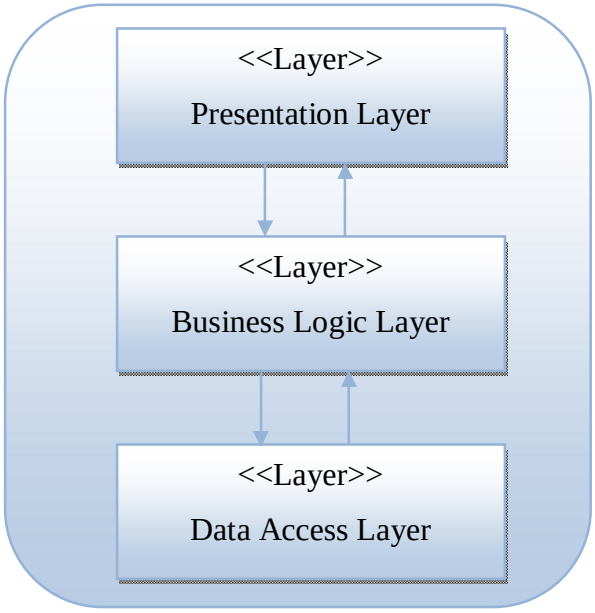


Figure 5.1: *Presentation, business logic, and data access layers*

Business Logic Layer: is responsible to pass the data from data access layer to presentation layer and implements the functionality of the OLAWP. It is responsible for setting of text box area size and selecting of virtual keyboard according to the visiting mobile phones, and performing key mapping into the text box area and text formatting operations.

Data Access Layer: This is the bottom layer that provides persistent storage and access of data. In this work, the OLAWP database tables and documents stored in this layer.

By taking into consideration the design goals and constraints discussed in Section 5.1, the overall architecture of OLAWP is developed as shown in Figure 5.2. As shown in Figure 5.2, the online Amharic word processor for mobile phones architecture contains four major components: Word Processor, Keyboard Layout Manager, Device Detector & Redirector and User and File Manager Components, and two supportive components: Page Customizer and User's Page. Each component discussed briefly in Section 5.3 by decomposing the OLAWP system. The database used to store the files permanently and accessible from your mobile phones wherever you are. OLAWP is a cloud-based service which is accessible through WAP gateway.

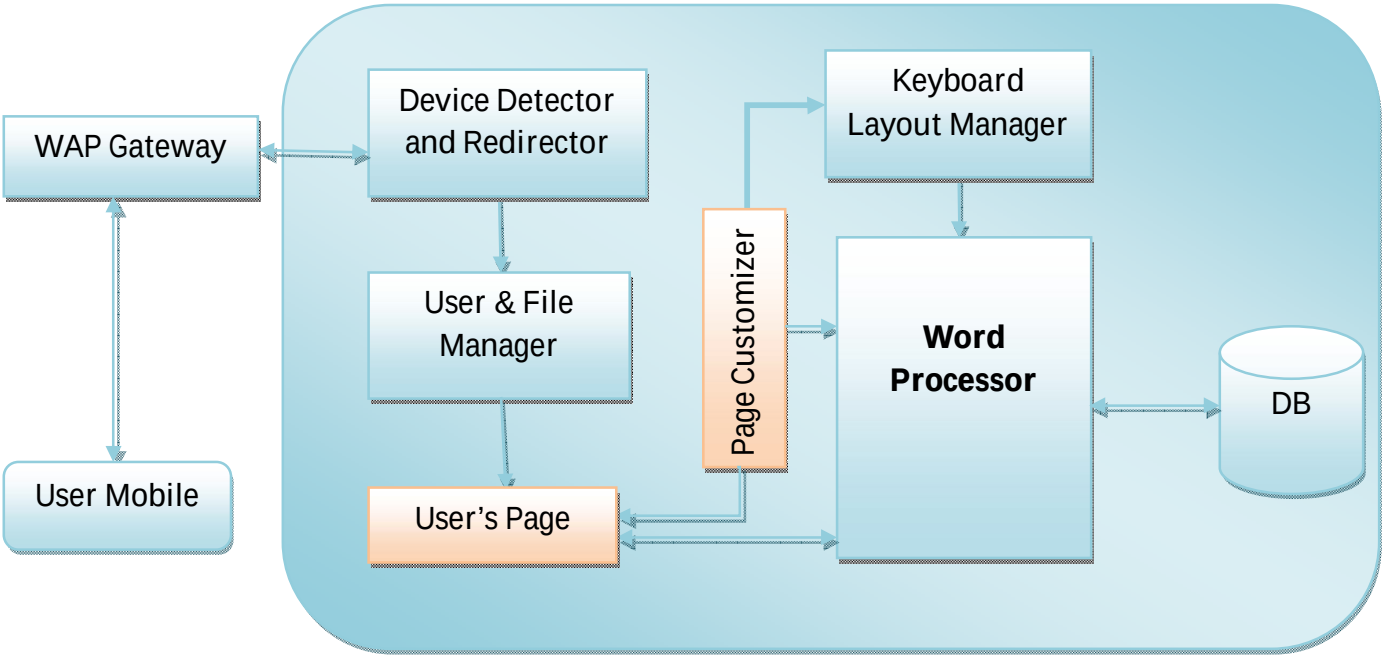


Figure 5.2: *Overall architecture of OLAWP system*

5.3 System Decomposition

Decomposing system into manageable components or subsystems which are loosely coupled to each other and strongly cohesive internally helps to reduce the complexity of the system. In this work, the OLAWP system is decomposed into four components. The component diagram is used to show the organization and dependencies among a set of components as shown in Figure 5.3.

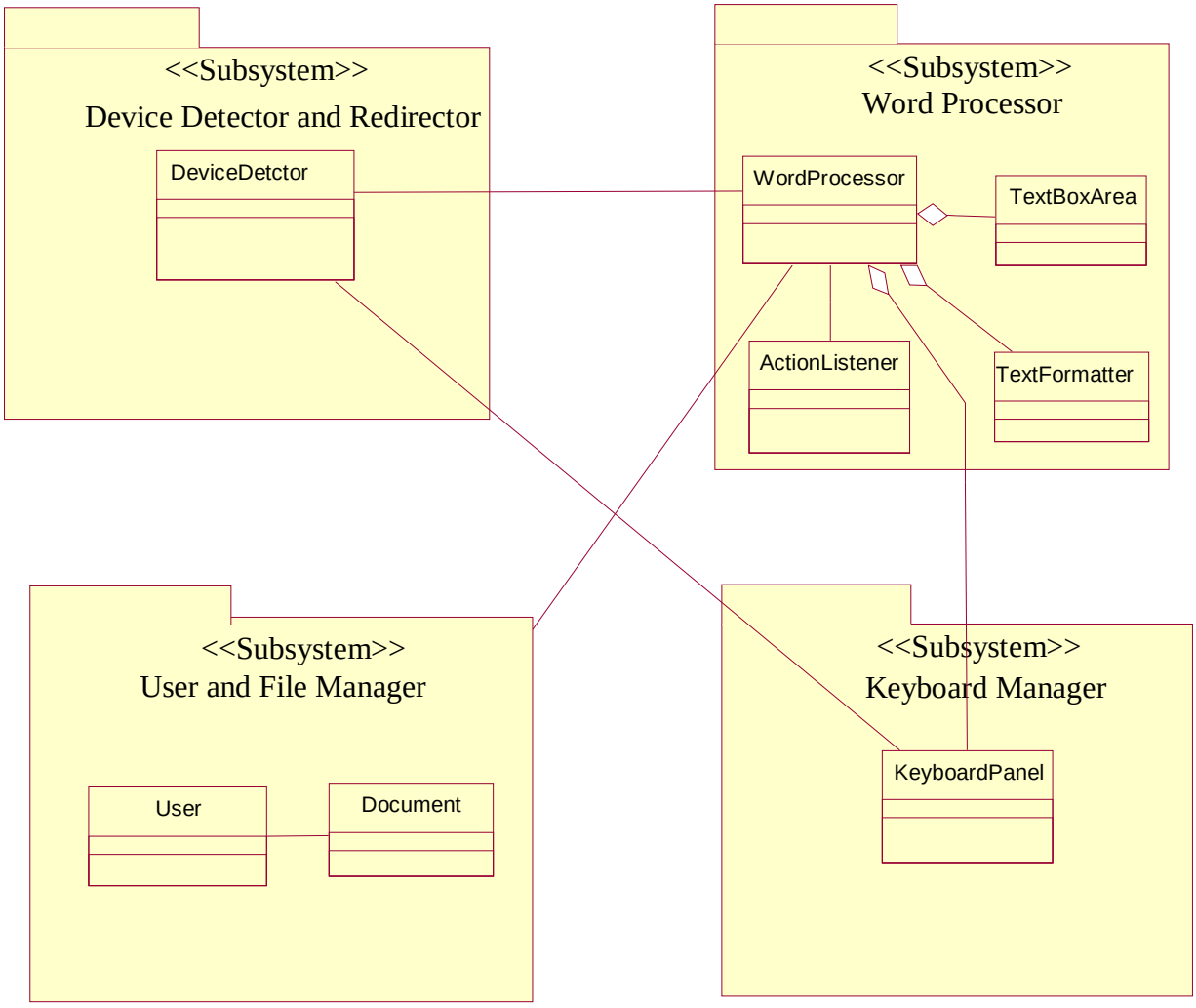


Figure 5.3: *Component diagram of OLAWP*

5.3.1 Device Detector and Redirector

The device detector and redirector component has two responsibilities as the name indicates; detecting mobile phone capability information and redirecting to the appropriate page. Device detector is responsible for detecting of mobile phone capability using the user agent string while the user is trying to access the system site. User agent string is a unique value for each device and sent by the browser. Based on the user agent string, the device detector and redirector component matches the string value with the mobile WURFL DDR from the WURFL cloud to retrieve the necessary capability information about the visiting mobile phones. Such capability information is used as an input for other OLAWP architecture components. Even though many different capability attributes of mobile phones can be obtained from the WURFL cloud (like device model, text entry method, browser type, etc) using the device detector component, only the physical screen height and supporting scripting language attributes are considered in the design of OLAWP.

These two capability attributes play a great role for designing appropriate text box area, selection of virtual keyboard type for each specific or group of mobile devices, to apply client scripting language and proper style sheet.

The redirector used the returned result (which is a boolean type) of the scripting language attribute. If the mobile phone supports a client-side scripting language (return value is true or 1), the redirector will redirect to the OLAWP page that is designed with the help of client-side scripting language (i.e. JavaScript). Otherwise, it redirects to the other page as shown in Figure 5.4.

According to the detector information, the redirector will check whether the device supports JavaScript or not, which is described diagrammatically in Figure 5.4. As shown in Figure 5.4, if the device supports JavaScript, the redirector will redirect to the OLAWP page version (Page1) that supports JavaScript language. Otherwise, it will redirect to the other page version (Page2) that is designed only using server-side scripting language, PHP script, for dynamic part of the system. Page customization is performed on both user's file page and the word processor pages (Page1 and Page2).

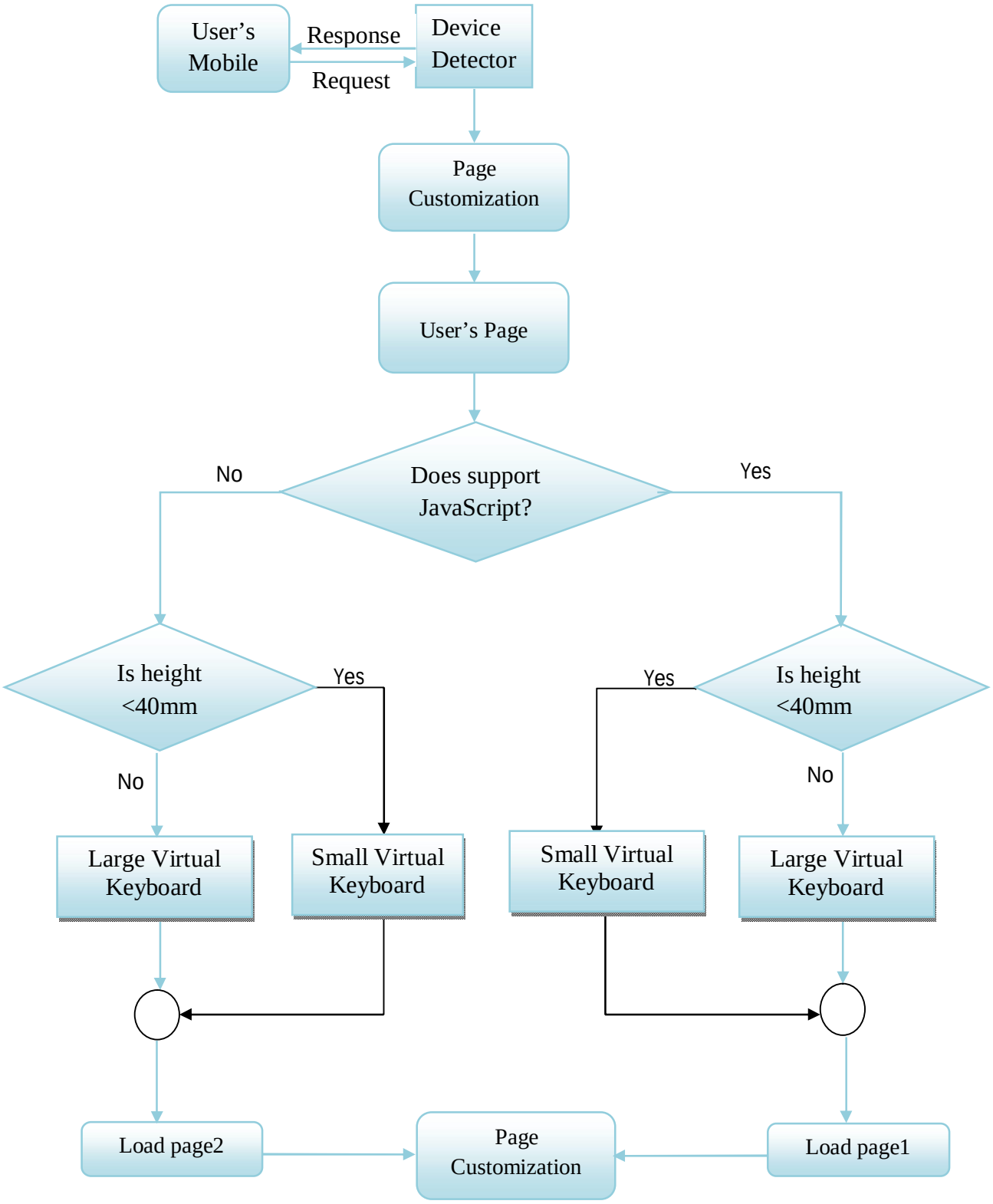


Figure 5.4: ***Device detector and redirector work flow***

Figure 5.4 shows how the device detector and redirector component works on the actual device when a user tries to access the OLAWP site.

According to the physical screen height, the mobile phones are categorized into small (less than 40mm), medium (less than 50mm) and large (greater than or equal to 50mm) mobile phones as described in Table 5.1. For each category, different style sheets are applied on the page that is loaded in order to use proper font size and fit the content with the visiting device.

Table 5.1: *Types of device, virtual keyboard and style sheet used based on screen height*

No.	Device category	Screen height (in millimeter)	Virtual keyboard used	Style sheet used
1	Small	<40	Small size	Style sheet 3
2	Medium	<50 & >=40	Large size	Style sheet 2
3	Large	>=50	Large size	Style sheet 1

As described in Table 5.1, the physical screen height attribute is used to categorize the mobile phones, virtual keyboards and style sheet used. Both medium and large mobile phone categories make use of large size virtual keyboard for text entry purpose, but different style sheets (style sheet 2 and 1 respectively) are applied, and small mobile phone category uses small size virtual keyboard and style sheet 3. The page customization carried out these three style sheets category to customize the font size of the OLAWP contents in order to fit the contents with the screen.

If the mobile phone capability information doesn't exist in the WURFL DDR, the device detector will return a default screen height which is 27mm and returns false (-1) for scripting language status. Therefore, it is considered as a small mobile phone and loads a page which is developed using PHP script only (i.e. Page2).

5.3.2. User and File Manager

It is a component which is responsible for authentication of users and controlling of files that are related to the authorized user. But, the user should be registered in the system first. Once registered, the user can create and send a simple text file to his/her email address or others. This component is responsible for the controlling of text files that are linked to each user. When the user is logged in, the system loads all the files that are related to that specific user which include user saved files (draft), sent files and income files (or messages). Here, the user can open a word processor to create a document, delete existing files and view existing ones. The user's page only displays a few numbers of stored files, which depends on the physical display height of the mobile phones. The number of files displayed at a time on the three different categories of mobile phones is listed in Table 5.2. The result of this component is displayed on the user's page.

Table 5.2: *Number of files displayed on each mobile phone categories*

Mobile Category (physical screen height in mm)	**Number of Files displayed at a time**
Large (>=50)	7
Medium (>=40 & <50)	5
Small (<40)	3

The number of files is specified in order to avoid scrolling of pages. But, the user can view more files using the next navigation button. The files are sorted and displayed based on the time of creation or modified date.

5.3.3 Word Processor

As shown in Figure 5.2 and 5.3, word processor is the main component of the OLAWP system which supports Amharic language users to prepare simple Amharic text letter with some text formatting facilities (such as font style, font size, font color and alignment) using their own mobile phones.

The word processor component is divided into three panels: virtual keyboard layout panel (bottom panel), the text box area (middle panel) and the menu and formatting toolbars (top panel) as shown in Figure 5.5. The virtual keyboard is used as a means of text entry in all types of mobile phones (either touch screen or non-touch screen) by pressing or selecting the keys. The middle panel refers to the text box area which is an area where the characters are inserted and displayed when the user presses button from the virtual keyboard. The top panel contains the menu and formatting tool bars which are used to open a new or existing document, to format a text, to save for future use and exchange with others.

Text formatting tool bar is responsible to take action on the text when the user presses the formatting buttons. The keyboard mode selection keys are also included in the top panel which are used to switch from "ሀለሐ" (ሀ) to "የመነ" (የ) for large size Amharic virtual keyboards, from capital letters ("E") to small letter ("e") for English and vice versa.

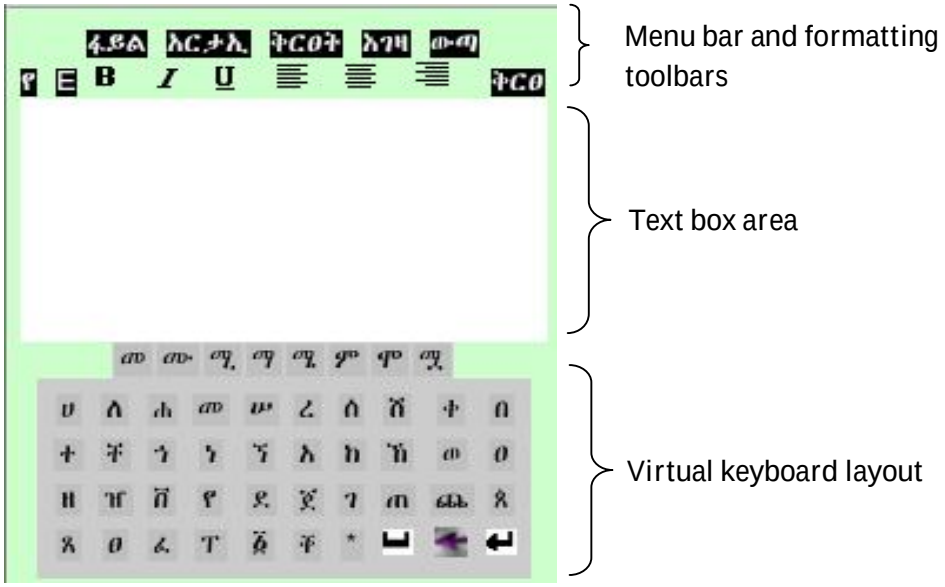


Figure 5.5: *The proposed word processor interface*

In order to design the word processor an adaptation method (redirection and scaling techniques) are chosen and applied. Because, they are appropriate methods to provide services based on the devices capabilities. The redirection and scaling types of adaptation techniques are used based on the script language and the physical screen size of the mobile phones respectively.

The word processor page is designed in two different versions according to the client-side scripting language capability of the mobile phones. The first page version (Page 1) is designed using JavaScript for mobile phones which support client-side scripting language as shown in Figure 5.4. The JavaScript helps to make the interaction faster, and is used to implement the key mapping operation to the text box area and to format a text. On the contrary, devices which do not support client side script language use server-side scripting language (PHP selected in this work) which performs every action on the server-side. Both versions of word processor are loaded from a remote server and the difference is where the text entry and formatting operations are performed (i.e. either at the server side or client side according to the device capability). So, a redirector will redirect to either of the two word processor page versions based on the accessing device capability which is obtained from the device detector and redirector component.

In addition to redirection approach, a scaling technique is also applied on the word processor interface to design the text box area that fits the screen of the mobile phone. The scaling technique used the physical height of the visiting device which is stored in the device detector and redirector component. Different display size mobile phones would have different size of text box area to fit the screen. So, designing of flexible text box area according to the mobile phone is necessary in order to be accessible on most mobile phones. The text box area is set to hundred percent (100%) of the mobile phone screen width and half (1/2) of the screen height of the mobile phone. The remaining half (1/2) is used for the virtual keyboard panel and the top panel (i.e. menu and formatting toolbars).

Moreover, the page customization is applied on the word processor page using different style sheets based on the mobile phone category as described before in Table 5.1.

5.3.4 Keyboard Layout Manager

The keyboard layout manager component provides an Amharic virtual keyboard text entry method to prepare simple Amharic document. Also, it is responsible for mapping the user action into the appropriate Amharic character font and sending the character to the text box area.

As a text entry method, virtual keyboard is chosen in this study. Even though virtual keyboard is used in the system, displaying all the Amharic characters on a mobile device instantly is very difficult. So, it is necessary to group, divide and load the virtual keyboard in different phases from the cloud when needed.

Two different sizes of Amharic virtual keyboards are designed for text entry: large size and small size virtual keyboards based on the height of display size as described in Table 5.3. The large size type of virtual keyboard character arrangement or layout is adopted from the previous works [12, 13] with small modifications. Large virtual keyboard has been designed in two different layouts: the modified "ሀለሐ" [12] and "የመነ" [13] keyboard layouts as indicated in Table 5.3. Table 5.3 shows the virtual keyboard types used in the three different mobile categories.

Table 5.3: *Virtual keyboard types*

Virtual Keyboard Type	Physical Screen Height in mm	Mobile Category	Virtual Keyboard Layout
Small	<40	Small	"ሀለሐ"
Large	>=40	Medium and Large	"ሀለሐ" or "የመነ"

The OLAWP modified both the previous "የመነ" and "ሀለሐ" virtual keyboards layout in order to be compatible and used for all types of mobile phones. The "የመነ" virtual keyboard [13] divided the virtual keyboard layout into three panels. The middle panel contains the basic character and punctuation selection keys within four rows. When a key is pressed from the first two rows, the non-basic characters are displayed orderly on the top panel based on the characters frequency. When a key is pressed from the last two rows, the non-basic, punctuation and numeric characters are displayed on the bottom panel of the virtual keyboard layout. But, in OLAWP the "የመነ" virtual keyboard layout is represented in two panels and the top panels (shown in Figure 5.6)

only used to display the non-basic, Arabic and Amharic numerals and punctuation mark characters, and the non-basic characters are ordered in the ordinary Amharic character sequences.

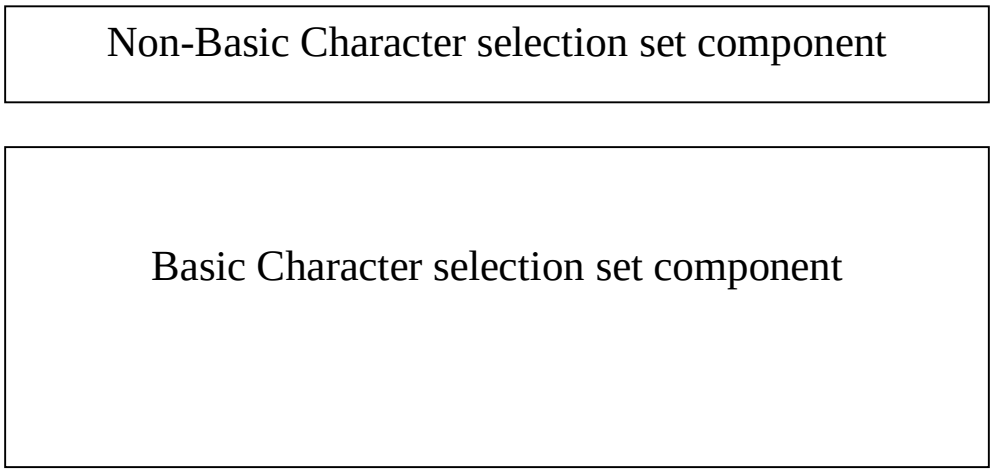


Figure 5.6: *Amharic virtual keyboard layout*

Also, the fourth row of the "የመነ" virtual keyboard characters, which contains the second most 10 frequent basic characters (ከ , ደ , አ , ቸ , ቀ , ጠ, ፈ , ሀ , ዘ , and ሐ) in Amharic scripts, is moved into the second row in the modified "የመነ" virtual keyboard. Actually, the "የመነ" was designed for PDA devices, but the OLAWP is designed for all types of mobile phones which have touch and non-touch based screens. Non-touched based screen (keypad based) mobile phones used their arrow keys to select the character, so that the frequent characters should be moved to the top to minimize the key movement. In addition, labelized characters like (ቕ, ኗ, ሟ and so on), space and enter keys are added in the modified "የመነ" virtual keyboard via (ቕ) selection key, space (␣) and enter (↵) keys respectively as shown in Table 5.4.

Table 5.4: *Modified large size "የመነ" Amharic keyboard layout*

የ	መ	ነ	ተ	ለ	በ	ረ	ወ	ሰ	ገ
ከ	ደ	አ	ቸ	ቀ	ጠ	ፈ	ሀ	ዘ	ሐ
ሠ	ኘ	ኟ	ዐ	ጸ	ፀ	ኸ	ጨ	ፐ	ጎ
ኽ	ጾ	ዥ	ቨ	ፏ	ቕ	*	␣	⇦	↵

Table 5.5: *Modified large size "ሀለሐ" Amharic keyboard layout*

ሀ	ለ	ሐ	መ	ሠ	ረ	ሰ	ሸ	ቀ	በ
ተ	ቸ	ኀ	ነ	ኘ	አ	ከ	ኸ	ወ	ዐ
ዘ	ዠ	ዸ	የ	ደ	ጀ	ገ	ጠ	ጨ	ጰ
ጸ	ፀ	ፈ	ፐ	⁖	ቐ	*	␣	⇦	↵

On the other hand, the "ሀለሐ" Amharic virtual keyboard was designed by categorizing the basic characters, and numeric and punctuation selection keys with five rows [12]. But, the modified virtual keyboard has been designed in four rows and include labelized characters using representative labelized selection key (ቐ) and enter key (↵) as described in Table 5.5.

As shown in Figure 5.6, the keyboard layout has basic character selection set component (bottom panel) and non-basic character selection set component panel (top panel). The bottom panel displays the basic 34 Amharic characters ("ሀ" to "ፐ"), Amharic and Arabic digits, Amharic punctuation mark and labelized characters selection keys, and space, enter and backspace keys. These basic characters keys are displayed in the first phase when the word processor page is loaded. All the basic characters are arranged in four rows and ten columns as shown in Table 5.4 and 5.5 for both "የመነ" and "ሀለሐ" virtual keyboard layouts. On the other hand, the small virtual keyboard groups the basic characters into two categories and loads the virtual keyboard one after the other. All Ethiopic scripts which are defined by Unicode standard 6.1[20] are incorporated except character "ደ" (Ethiopic syllable dda), which is not used in Amharic language rather it is used to represent the Oromiffa character "dh".

The top panel displays the seven (or eight) non-basic characters when the user presses basic characters in the bottom panel. This panel also displays Amharic and Arabic digits, punctuation marks and labelized characters when the user clicks on the numeric keys (⁖ or 123), symbol (*) and labelized key (ቐ) button respectively. In Unicode standard 6.1, all the basic characters have eight forms except the "ዐ " and "ኸ" characters, which have seven forms including the basic character itself. The keyboard layout is changed based on the display size information that is obtained from the device detector and redirector component. When the user presses one of the basic characters, the keyboard layout will be adjusted so that all the families of the pressed

character will be shown in the top panel. Then, the user can press the key representing the required character in the top panel; then this character will be displayed within the text box area as the user inputs.

Small physical screen size (i.e. their screen height less than 40mm) mobile phones used small size "ሀለሐ" virtual keyboard that is displayed in two phases; each phase has three rows with seven columns as shown in Table 5.6.

Table 5.6: *Small size Amharic virtual keyboard layout*

(a) First Part

ሀ	ለ	ሐ	መ	ሠ	ረ	ሰ
ሸ	ቀ	በ	ተ	ቸ	ኀ	ነ
ኘ	አ	ከ	ኸ	ወ	ዐ	ዘ

(b) Second part

ዠ	ዸ	የ	ደ	ጀ	ገ	ጠ
ጨ	ጰ	ጸ	ፀ	ፈ	ፐ	*
	፩	ቐ	␣	<-	↵	

The user also can select keyboard layout mode, for medium and large mobile phone types, either "የመነ"(**ዩ**) for experienced users, or "ሀለሐ" (**ሀ**), commonly for novice user.

For English text entry, the QWERTY keyboard [24] layout is adopted for this system. Only the Symbol (*), Number (123), Backspace (<-) and Enter (↵) keys are added to the existing QWERTY keyboard layout [24] as shown in Figure 5.7. In addition, the user can switch from capital letter mode to small letter mode and vice versa using the switch button **"E"** or **"e".** The switch mode buttons exist together with the formatting toolbar at the top panel of the word processor.

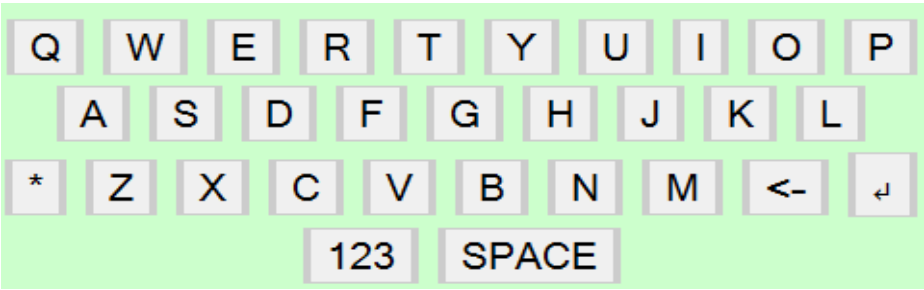


Figure 5.7: *Modified QWERTY virtual keyboard*

CHAPTER SIX

IMPLEMENTATION

In this chapter, the development tools used in the implementation of OLAWP prototype is presented and the prototype itself is discussed by taking its sample snapshots from the real mobile phones.

6.1 The System Development Tools

6.1.1 Development Editor

The implementation of the OLAWP site used a web development editor to write the code in XHTML-MP, PHP, JavaScript and CSS. Particularly, Macromedia Dreamweaver 8 web editor used in the development process of OLAWP prototype. Macromedia Dreamweaver 8 is a web development tool that can efficiently design, develop and maintain standards-based websites and applications [30].

6.1.2 Markup and Scripting Languages

The main types of content that mobile browsers support are XHTML-MP. XHTML-MP is a standard used for all browser-based services using mobile phones introduced with WAP 2.0 [14]. The XHTML-MP is a mobile adaptation of XHTML by excluding those features not appropriate for devices with small screens. Early mobile phones, low-powered types like Nokia 1100, 1200, and so on did not support such markup language rather they only support WML to develop WAP applications.

So, XHTML-MP markup language is chosen to present the OLAWP contents, but the dynamic part of the implementation has been done using client-side (JavaScript) and server-side (PHP) scripting languages. The XHTML-MP also supports CSS which describes how documents are presented on screen. The CSS enables application developers to create mobile specific versions of the same content easily, simply by creating different style sheets. There are three different types of style sheets that are prepared for small, medium and large categories of mobile phones as mentioned in Table 5.1.

6.1.3 Apache Web Server

Apache is a free software which is aimed at serving a great deal of widely popular modern web platforms. While the OLAWP prototype developing a VertrigoServ is used to execute the PHP script. VertrigoServ has been developed as a highly professional, easy to install package consisting of Apache web server, PHP programming language, MySQL and PhpMyAdmin (tool written in PHP intended to handle the administration of MySQL) for Window platforms [31]. It helps to test web pages locally on both computer and emulator browsers.

After the OLAWP prototype development is completed, it is deployed on a free web host provider site (http://www.3owl.com), which used an Apache server and tested it on the actual mobile phones.

6.1.4 WURFL Package

WURFL (Wireless Universal Resource File) is a device description depository (DDR) package, which is huge open-source XML based database information regarding mobile phones and their capabilities to support web design. It contains descriptions of thousands of mobile devices that can be used in any application. This description refers to the capabilities information of the mobile devices. Such information is very helpful while designing the web applications for mobile devices.

When a web browser visits OLAWP site, it sends a user agent value (which is a unique value for each device), along with the request of OLAWP page. The user agent contains information about the type of device and browser that is being used. The device detector and redirector component can easily obtain the capability of the visiting device from the WURFL Cloud by integrating the WURFL Cloud Client API which is provided freely into the OLAWP. The user agent value send to the WURFL Cloud and it returned the required capability information and store on the OLAWP site to redirect to the appropriate OLAWP page that is compatible with the visiting device and helps to specify the OLAWP page according to the mobile phone screen height.

The WURFL Cloud Client is an interface to the WURFL Device Description Repository (DDR) which contains detailed definitions for thousands of mobile devices. Using the WURFL Cloud Client is simple and straightforward. All that is needed is a few lines of code to query the DDR

which will return the latest device definition. By using the device definition you can then generate the appropriate markup to create an optimized mobile experience for your visitors [23]. In this thesis work the free WURFL cloud subscription used, which provides 2 mobile phone capabilities detection, such as screen height and the client-side scripting language (JavaScript) supporting capability attributes.

6.1.5 Emulators

Mobile emulator is a desktop application that emulates mobile device hardware and operating systems. It allows us to test and debug our applications and see how they are working. It is the most useful tool to test WAP applications while developing. In the development process of the prototype, the PC browsers, Android and Nokia emulator browsers are used to test the OLAWP prototype.

6.2 Prototype

A prototype implementation has been developed to demonstrate the architecture which was proposed in the previous chapter. The OLAWP prototype has been deployed on the web server and the service is accessed through the WAP gateways which exist at the Internet provider side. In order to test on the actual mobile phones, OLAWP prototype should be hosted first on the web server. The OLAWP prototype is hosted as a sub-domain name of *www.olawp.3owl.com* under the free web host service provider site *www.3owl.com.*

6.2.1 Interfaces

The OLAWP prototype interfaces are developed in both Amharic and English languages. The user can select either of the two depending on his/her own preferences and experience. However, in the following subsequent discussion the Amharic based interface snapshots are taken into consideration.

In order to use the OLAWP service (i.e. word processor), a user should first be registered on the OLAWP site (*www.olawp.3owl.com*) to be a member and can use the service provided. When the user browses the OLAWP site, the login page with Amharic interface is displayed as shown in Figure 6.1. All the snapshots that are shown are taken from the actual Samsung GT-S5570 model

mobile phone which uses an Android OS. The user can press the "**En**" button to switch to English language based interface and can switch back to Amharic using "**አማ**" button.

New user can register by pressing the button **"ምዝገባ (Register)"** that shown on the login page, which loads first.

Figure 6.1: *Login page*

The registration form is simple and it requires only *user code* and *password* as shown in Figure 6.2. Mobile phones text entry is not fast like PCs. Due to this, only little user information is required for registration process which avoids scrolling of pages or long interaction for registration.

Figure 6.2: *Registration form*

Once a user is registered on the system, the user can use his/her *user code* and *password* to log in to OLAWP site, and can access existing files and create simple Amharic text files. When a user logged in, the User and File manager component of OLAWP load the user's personal files on user's page, which displays three, five or seven list of Inbox files for small, medium and large screen mobile phones respectively as mentioned in Table 5.2. This page categorizes files into Inbox, Sent and Draft files as shown in Figure 6.3.

Figure 6.3: *User's page*

Users can open existing files which are stored, received and sent files to edit or view from the user's file page or can create a new document file by pressing on "New (አዲስ)" link. When the action is taken to open an existing or a new document, the redirector component redirects to either of the two OLAWP version pages.

Later on, the existing document or new word processor will be opened. Figure 6.4 shows a new word processor which is ready for writing.

Figure 6.4: *Word processor*

In order to prepare Amharic document, the user should use the Amharic virtual keyboard either **"ሀለሐ"** or **"የመነ"** keyboard layout by switching from one to another for large or medium mobile phone categories. Press **"የ "** to switch from **"ሀለሐ"** to **"የመነ"** keyboard layout and press "ሀ " to return to the **"ሀለሐ"** keyboard layout. Numeric and labeled characters can be accessed by pressing the **"፩ "** and **"቞ "** keys from the virtual keyboard. It is also possible to incorporate English text on the document by pressing **"E"** from the top panel which switches the virtual keyboard to QWERTY English virtual keyboard layout.

When one of the basic Amharic character key is pressed from the virtual keyboard, the non-basic characters will be displayed at the top of the virtual keyboard as shown in Figure 6.5.

Figure 6.5: *Appearance of word processor after basic "ሙ" key is pressed*

The formatting tools enable the user to format the text font style, size and color, text background color and text alignment. To view the font size, font color and background color, the user should press **"Font (ቅርፀ)"** button which is shown in Figure 6.6.

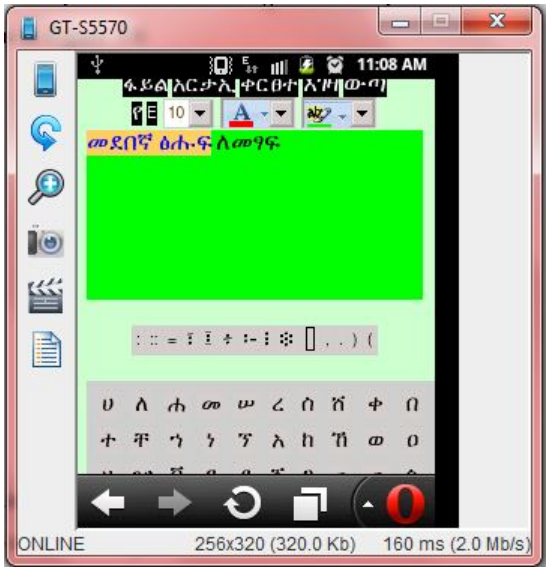


Figure 6.6: *Appearance of word processor after "ቅርፀ" button is pressed*

After the document is prepared, the user can save the document for future use, send to other phone via SMS or send to email and access from a computer to print the document.

For SMS application, ViaNett HTTP API is used to send SMS message in order to show the functionality. The HTTP API is a simple and powerful API to send SMS messages. ViaNett allowed five free SMS messages to test the web applications with one registered email account [32].

If the written text is sent as SMS, the formatting applied on the text will be discarded and only the text is sent as a message if the message length is less than 160/70 characters for Latin/non-Latin scripts, otherwise the message is stored on the server (or cloud) and send the link where the message found. Using the link, the user can view or get the message sent to him/her.

CHAPTER SEVEN
EVALUATION OF OLAWP

7.1 Overview

The OLAWP prototype is developed to enable users to prepare simple Amharic text using their phones. In order to evaluate the prototype, different volunteers participated using their own mobile phones. The prototype usability evaluation was performed based on the ISO 9241-11 usability testing attributes, such as efficiency, effectiveness and user satisfaction.

Actually, the prototype test was conducted on both mobile phone emulator and real mobile phone while developing the prototype. After the development was completed, the prototype was deployed on the free web hosting site *(http://www.olawp.3owl.com*) to conduct the testing on the actual mobile phones by the participating users.

The aim of the testing is to evaluate the compatibility of the OLAWP prototype in various mobile phones, satisfaction of the user and performance (efficiency and effectiveness) of the prototype while using the OLAWP prototype.

7.2 Usability Testing

Usability testing is an essential testing method in the development of application before releasing to the market. ISO defines usability of a product as - the extent to which the product can be used by specified users to achieve specified goals with ***effectiveness*** (refers to completeness and accuracy with which users achieve certain goals**),** ***efficiency*** (refers to how fast users can accomplish a task while using an application), and ***satisfaction*** in a specified context of use. Effectiveness, efficiency and user satisfaction are the three major usability attributes which are identified by ISO 9241-11 standard [33].

Usability testing is accomplished by identifying representative users, representative mobile phones, and representative tasks which are related to the OLAWP prototype. Totally 15 volunteer respondents are chosen to conduct prototype testing survey in various types of mobile phones with different screen size, browser types and different text entry. The questionnaire is

prepared by considering the above indicated three usability attributes. The questionnaire contains 18 questions in two parts as indicated in Appendix C.

7.3 Test Result and Discussion

In the testing process, 15 representative users are chosen based on the mobile phone type that they used. Among 15 respondents, 9 of them were male and 6 of them were female. According to the mobile phones category used, 2 of the respondents used small mobile phone types, 6 of them used medium sized mobile phones and the remaining used large mobile phone category types. All respondents had experience in Internet browsing, and seven of them had experience in Amharic SMS software.

Table 7.1 shows detail information about the mobile phones used in the testing process.

Table 7.1: *Description of mobile phone used in the testing process*

No	Device model	OS used	Browser used	Mobile category	Screen type
1	Samsung GT-S5360	Android	Android	Large	Touch
2	Apple iPhone	iOS	Opera mini	Large	Touch
3	Apple iPhone	iOS	Opera mini	Large	Touch
4	Samsung GT-S6102	Android	Opera mini	Large	Touch
5	Samsung GT-S5570	Android	Android	Large	Touch
6	Samsung GT-S5570	Android	UC Browser	Large	Touch
7	Nokia C2	Symbian	Opera mini	Large	Touch
8	Samsung GT-S3350	Bada	UC Browser	Medium	Non-touch
9	Nokia 6303	Symbian	Opera mini	Medium	Non-touch
10	Nokia E5	Symbian	Opera mini	Medium	Non-touch
11	Nokia 2700	Symbian	Opera mini	Medium	Non-touch
12	Nokia N72	Symbian	Opera mini	Medium	Non- touch
13	Nokia 2700	Symbian	Opera mini	Medium	Non-touch
14	Nokia 3110	Symbian	Opera mini	Small	Non-touch
15	Nokia200	Symbian	Opera mini	Small	Non-Touch

As shown in Table 7.1, three different mobile phone types (Samsung, Apple and Nokia) and three different mobile browsers (Android, Opera mini and UC) are used in the testing process. As a text entry method, seven mobile phones were touch based and the remaining were keypad based mobile phones.

The prototype test was conducted on different mobile phones and browsers even though most of them are tested on Opera mini mobile browser as shown on Table 7.1. On UC and Android browsers the typing speed and text formatting is faster than the Opera mini browser. This is because of the Opera mini only supports and executes very limited number of JavaScript elements. The Opera mini browser works in a different manner compare to other browsers. Rather than accessing the web resource directly, Opera mini browser will access through its servers. Basically, when you request a web page from Opera mini, a request will be sent to the Opera mini servers. They retrieve the page, convert it into OBML (Opera Binary Markup Language) which is a very compact binary markup format that reduces the page size by up to 90%, and then send it to user's phone [34].

With other browsers (UC and Android), the client-side script is executed on the client-side and the interaction is very fast especially for the touch-screen based mobile phones. The key press movement on touch screen is very fast compared to keypad based text entry mechanisms. In case of keypad text entry method, key navigation is required which is achieved using the arrow keys to select the required key, and needs to press the arrow keys many times till you get the required key. The testing result also showed that most touch based mobile phone users who used UC and Android browsers can type Amharic text better than keypad based mobile phone users.

Table 7.2 summarizes the respondents' result of the OLAWP questionnaire of each question in number.

Table 7.2: *OLAWP questionnaire respondents result summary in number*

Question No.	Strongly Agree	Agree	Neutral	Disagree	Strongly Disagree
1	9	4	2	0	0
2	2	6	4	3	0
3	2	9	4	0	0
4	2	5	6	2	0
5	3	7	5	0	0
6	2	4	6	3	0
7	2	2	2	1	0
8	1	3	8	3	0
9	2	10	2	1	0
10	12	3	0	0	0
11	12	3	0	0	0
12	3	12	0	0	0
13	5	10	0	0	0
14	All mobile phones (15 of them) can run OLAWP prototype.				

For discussion purpose, we categorized the user responses into three categories: strongly agree and agree in one group as a good result, neutral, and disagree and strongly disagree in one group as a poor result. According to the respondent, the OLAWP prototype could run in all types of mobile phones which are listed in Table 7.1. As the result shows, the OLAWP prototype 100% fits visitor mobile phone screen and looks good. But, the interaction speed showed different results depend on the text entry method and the browser type used as discussed above. As shown in Table 7.2, row 6 and 8 are results of question 6 and 8 of the questionnaire which are related to the interaction with the virtual keyboard and the general interaction with the OLAWP prototype. The result showed that 33.33% of respondents can interact easily with the system; most of the respondents (46.67%) response is neutral and the remaining didn't feel comfort. Most of the unsatisfied users were used non-touched based mobile phones, Opera mini browsers and server-side scripting languages in which the interaction was poor.

All the respondents had a good response for question 11, 12 and 13 which are related to text clarity, steps to accomplish a given task and menu items consistency respectively and the results shown in row 11, 12 and 13 in Table 7.2 and 7.3. Easiness of the prototype also evaluated based

on the respondents' personal opinion, among them 73.33% of them can used easily and the remaining 26.67% of respondents' response was neutral.

As shown in Table 7.3, 53.33% of the respondents were satisfied while preparing Amharic text using OLAWP; 26.67 % of respondents' response were neutral and the remaining are disagreed on it. Most of satisfied users were use large mobile phone category, touch based screen and UC and Android browsers. Table 7.3 describes in percent which was described in Table 7.2 in number.

Table 7.3: *OLAWP questionnaire respondents' result summary in percent (%)*

Question No.	Strongly Agree	Agree	Neutral	Disagree
1	60	26.67	13.33	0
2	13.33	40	26.67	20
3	13.33	60	26.67	0
4	13.33	33.33	40	13.33
5	20	46.67	33.33	0
6	13.33	26.67	40	20
7	28.57	28.57	28.57	14.29
8	6.67	20	53.33	20
9	13.33	66.67	13.33	6.67
10	26.67	73.33	0	0
11	80	20	0	0
12	20	73.33	6.67	0
13	33.33	66.67	0	0
14	100% yes			

Generally, the prototype evaluation showed that better satisfaction and work in a good manner on touch based Android and UC browsers mobile phones, and client-side script language supporting devices. Also, it was compatible (fit the screen) and run in all tested mobile phones. Basically, this is a beginning work to enable users to prepare simple Amharic text from any type of their mobile phones. So that, the researchers believed that the result is good as a beginning work.

CHAPTER EIGHT

CONCLUSION AND FUTURE WORK

8.1 Conclusion

Nowadays, accessing of web information and services at anytime from anywhere is becoming realistic using WAP enabled mobile devices. Mobile device users become rapidly growing from time to time.

This thesis work is aimed to design and implement an online Amharic word processor for mobile phones which enabled most Amharic language users to prepare simple Amharic text document using their own mobile phones wherever they go. In order to design the architecture of the OLAWP, the functional and non-functional requirements are identified and analyzed using the use case diagram, sequence diagram and class diagram, which show the static and dynamic behaviors of the system developed, OLAWP.

There are a number of various mobile phones which have various screen size, use different script and web markup languages, and various text entry methods. These are some of the challenges that should be considered in the design of the mobile based WAP application. In this work, the researchers chose the adaptation technique in order to provide the online Amharic text document preparation for most WAP enabled phones that are compatible with most mobile phones by considering the device capability information. In this study, the mobile phone capability attributes, such as screen size and script language support are taken into consideration on the design and development of OLAWP, and a virtual keyboard method is selected as a means of text entry which is developed by integrating with the OLAWP.

The OLAWP system contains four major components: device detector and redirector, user and file manager, word processor and virtual keyboard manager. When the user first trying to access the OLAWP site the device detector send the user agent string of the visiting device to the WURFL cloud and which is responsible for device capability detection and returns the requested capability information about the accessing device to the device detector. Based on the returned values, the user and file manager will load limited number of user's inbox documents on user's

page. In addition, the mobile phone capability values are used to design the text box area, to select the virtual keyboard type and the style sheet used, to specify the word processor size.

The OLAWP text box area size sets half of the screen height and total width of the screen, and the remaining portion is used for the menu & formatting toolbars and virtual keyboard. The virtual keyboard loaded can be either small (for screen height less than 40 mm) or large (for screen height greater than or equal to 40mm). The large and medium mobile phone used either "ሀለሐ" or "የመነ" large Amharic virtual keyboard layout where as small mobile phones used the "ሀለሐ" Amharic virtual keyboard layout which is loaded in two phases. For English text entry, the QWERTY virtual keyboard also adopted and implemented in this work.

For implementation purpose, the current WAP standard markup language, XHTML-MP, is used to present the contents on the client terminal, mobile phone. The dynamic part of this work implemented using PHP and JavaScript scripting languages.

The OLAWP prototype is evaluated through a questionnaire about the OLAWP prototype which is prepared by considering the ISO 9241-11 usability test standard attributes. In the prototype evaluation, 15 different users participated using various mobile phones which was vary in their input method, screen size, brand and browser used. The respondents used their real mobile phones to evaluate the OLAWP prototype which is hosted on a free web host service provider server.

According to the respondents' response, the result showed that better satisfaction and performance on some mobile phones which are touch based, UC and Android browsers, and client side scripting language supporting mobile phones compared to non-touched, Opera mini browser based, and only server-side scripting language supporting mobile phones.

8.2 Future Work

In this study only physical display size, script language and text entry techniques of mobile phones, and the nature of Amharic language are considered. But, there are other many issues that should be necessarily considered when an online mobile phone web services are developed, such as browser type, screen resolution, screen type and so on. The evaluation of prototype testing

result showed that the browser type used has a great impact on the interaction with the OLAWP and response time for a given action. Thus, as a future work browser type can be considered to enhance the user satisfaction. Moreover, as shown in the evaluation result, the text entry in touch based and client-side script language supporting browser mobile phones are faster than the keypad based and server-side script language browser mobile phones. Hence, designing and implementing for touch screen mobile phones and client-side scripting language supporting browsers are more appropriate.

References

[1] William Kehr, "*Wireless Appllication Protocol,*" http://web.mst.edu/~mobildat/WAP/index.html, Last accessed on Jun. 25, 2012.

[2] Teferi Assefa and Solomon Atnafu, "*WAP for Ethiopic Content an Application for Accessing Marketing Information,"* M.Sc. thesis, Addis Ababa University, Department of Computer Science, 2002.

[3] Chris Bennett, *"Wireless Application Protocol 2.0,"* Nov. 2001, http://www.informit.com/articles/article.aspx?p=23999, Last accessed on December 15, 2012

[4] "*Ethio-Telecom Press Release April 20, 2012, Ethio telecom holds the 4th session with Distributors,"* http://www.ethiotelecom.et/press/news.php?id=69, Last accessed on Jun. 16, 2012.

[5] "*Personal Computers Outnumbered by Mobile Phones,"* http://www.techsling.com/2010/10/personal-computers-outnumbered-by-mobile-phones/, Last accessed on Oct. 03, 2011.

[6] J.Q. Anderson and L. Rainie, "*The Future of the Internet III*," Pew Internet and American Life Project, 2008; www.pewInternet.org/Reports/2008/The-Future-ofthe- Internet-III.aspx, Last accessed on Sept. 28, 2011.

[7] "*NIST Cloud Computing Program*," http://www.nist.gov/itl/cloud/, Last accessed Sept. 23, 2011.

[8] http://www.aepona.com/white-papers/network-as-a-service-and-mobile-cloud-computing/, Last accessed on Dec. 12, 2012.

[9] "*Cloud computing issues and impacts,"* Global Technology Industry Discussion Series, http://www.ey.com/Publication/vwLUAssets/Cloud_computing_issues,_impacts_and_insights/$File/Cloud%20computing%20issues%20and%20impacts_14Apr11.pdf, Last accessed on Dec. 29, 2012

[10] Elena Balan, "*Nokia Brings Amharic Keypad to Mobile Phones*," September, 2007, http://news.softpedia.com/news/Nokia-Brings-Amharic-Keypad-to-Mobile-Phones-66590.shtml**,** Last accessed on Dec. 23, 2012

[11] S.Abebe, S.Atnafu and S.Kinde, "*Ethiopic Keyboard Mapping and Predictive Text Inputting Algorithm in a Wireless Environment*," ICTES-2004, Addis Ababa, Ethiopia, 2004.

[12] A.Workneh, Y.Getachew, G.Tamene and S.Atnafu, "*The 'ሀሌሐ' Virtual Ethiopic Keyboard for Smart Phones*," M4D2010 conference, Kampala, Uganda, 2010.

[13] G. Tamene and S. Atnafu, "*Design and Implementation of Virtual Keyboard for Amharic Text Entry on PDA System*," M.Sc. project, Addis Ababa University, Faculty of Informatics, Department of Computer Science June, 2008.

[14] Nirav Mehta, "*Mobile Web Development Building mobile websites, SMS and MMS messaging, mobile payments, and automated voice call systems with XHTML-MP, WCSS, and mobile AJAX*," Packt Publishing Ltd., Birmingham, Feb., 2008.

[15] M.Satyanarayanan, P.Bahl, R.Caceres and N.Davies, "*The Case for VM-based Cloudlets in Mobile Computing*," http://research.microsoft.com/pubs/102364/cloudlets09.pdf, Last accessed on Jul. 15, 2012.

[16] "*Global mobile statistics 2012 Home: all the latest stats on mobile Web, apps, marketing, advertising, subscribers, and trends...,*" http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats, Last accessed Sep. 30, 2012.

[17] "*Open Mobile Alliance*," http://www.openmobilealliance.org/, Last accessed on Dec. 10, 2012.

[18] Y.Chen, X.Xie, W.Y.Ma, and H.Z.Zhang, "*Adapting Web Pages for Small-Screen Device,*" Microsoft Research, Asia, IEEE Computer Society, 2005.

[19] Neal Leavitt, "*Will WAP Deliver the Wireless Internet?*," IEEE Computer, Vol. 33, pp. 16-20, May 2000.

[20] "*Ethiopic Script Range 1200:137F*," http://www.unicode.org/charts/PDF/U1380.pdf, Last accessed on May 24, 2012.

[21] Pavlovych A. and Stuerzlinger, W., "*Less-Tap: A Fast and Easy-to-learn Text Input Technique for Phones*," Proceedings of Graphics Interface '03. Waterloo, Ontario: Canadian Human-Computer Communication Society, 97-104., 2003.

[22] "*Virtual Keyboard, "* http://en.wikipedia.org/wiki/Virtual_keyboard, Last accessed on Sep. 17, 2012.

[23] *"WURFL Cloud - Getting Started,"* http://www.scientiamobile.com/wurflCloud/gettingStarted/, Last accessed on Jan. 15, 2013.

[24] Andrew Alexander Patterson, "*Dynamically Resizing Keys for PDA Text Entry*," An M.Sc. project at University of Strathclyde Department of Computer and Information Sciences, Sept., 2005.

[25] Worku Alemu, "*The application of OCR Techniques to the Amharic Script*," M.Sc. thesis, Addis Ababa University, Faculty of Informatics, Department of Information Science, 1997.

[26] "*Google Docs for mobile,*" http://www.google.com/mobile/docs/index.html, Last accessed Dec. 28, 2012.

[27] Julio Franco (2008, Mar), "*Zoho Extends Mobile Support/Offline Capability with Google Gears*," http://www.techspot.com/community/topics/zoho-extends-mobile-support-offline-capability-with-google-gears.100392/**,** Last accessed on Feb 3, 2013.

[28] Raju Vegesna, "*Introducing Zoho Mobile*," http://www.zoho.com/announcements/blog/introducing-zoho-mobile.html , Apr. 28, 2009.

[29] "*Documents To Go User Manual for Android,*" http://m.kasernet.com/manual/DTG_Android_Manual_100902.pdf, Last accessed on Jan. 4, 2013.

[30] "*Dreamweaver Release Notes*," http://www.adobe.com/support/documentation/en/dreamweaver/dw8/releasenotes.html, Last accessed on Dec. 15, 2012.

[31] "*VertrigoServ Freeware Web Environment, "* http://vertrigo.sourceforge.net/, Last accessed on Oct. 19, 2012.

[32] "*HTTP GET/POST API*," http://www.vianett.com/en/developers/api-overview/http-get-post-api, Last accessed on Sept. 28, 2012.

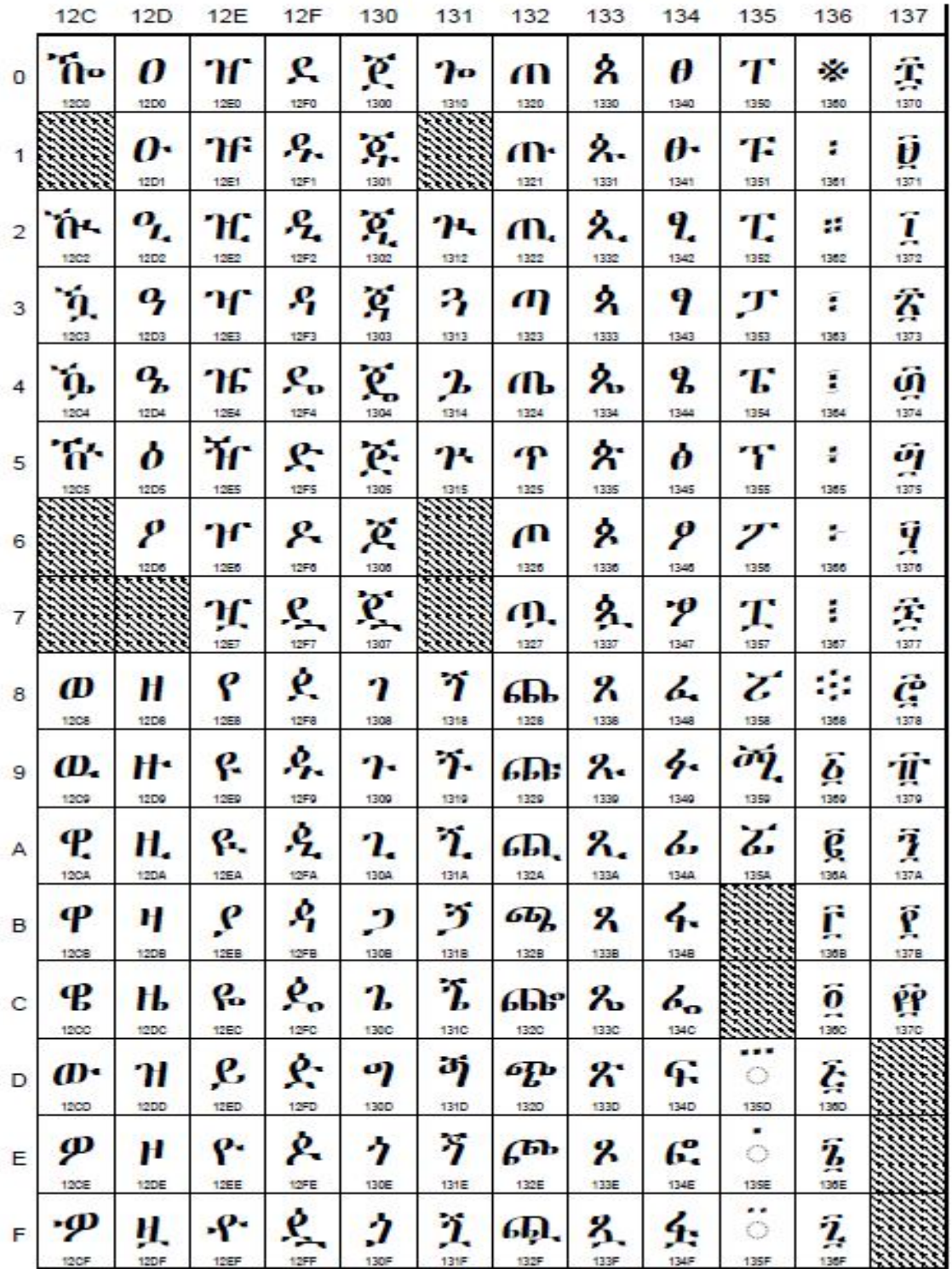
[33] "*Ergonomic Requirements for office work with visual display terminals – Part 11: Guidance on usability*," ISO 9241-11, International Organization of Standardization, 1995

[34] Chris Mills, *"JavaScript support in Opera Mini 4,"* http://dev.opera.com/articles/view/javascript-support-in-opera-mini-4/, Last accessed on Jan. 10, 2013.

Appendix A - Unicode standard Version 6.1 Amharic (or Ethiopic) scripts [20]

1200 Ethiopic 12BF

	120	121	122	123	124	125	126	127	128	129	12A	12B
0	ሀ 1200	ሐ 1210	ሠ 1220	ሰ 1230	ቀ 1240	ቐ 1250	በ 1260	ተ 1270	ኀ 1280	ነ 1290	አ 12A0	ኰ 12B0
1	ሁ 1201	ሑ 1211	ሡ 1221	ሱ 1231	ቁ 1241	ቑ 1251	ቡ 1261	ቱ 1271	ኁ 1281	ኑ 1291	ኡ 12A1	
2	ሂ 1202	ሒ 1212	ሢ 1222	ሲ 1232	ቂ 1242	ቒ 1252	ቢ 1262	ቲ 1272	ኂ 1282	ኒ 1292	ኢ 12A2	ኲ 12B2
3	ሃ 1203	ሓ 1213	ሣ 1223	ሳ 1233	ቃ 1243	ቓ 1253	ባ 1263	ታ 1273	ኃ 1283	ና 1293	ኣ 12A3	ኳ 12B3
4	ሄ 1204	ሔ 1214	ሤ 1224	ሴ 1234	ቄ 1244	ቔ 1254	ቤ 1264	ቴ 1274	ኄ 1284	ኔ 1294	ኤ 12A4	ኴ 12B4
5	ህ 1205	ሕ 1215	ሥ 1225	ስ 1235	ቅ 1245	ቕ 1255	ብ 1265	ት 1275	ኅ 1285	ን 1295	እ 12A5	ኵ 12B5
6	ሆ 1206	ሖ 1216	ሦ 1226	ሶ 1236	ቆ 1246	ቖ 1256	ቦ 1266	ቶ 1276	ኆ 1286	ኖ 1296	ኦ 12A6	
7	ሇ 1207	ሗ 1217	ሧ 1227	ሷ 1237	ቇ 1247		ቧ 1267	ቷ 1277	ኇ 1287	ኗ 1297	ኧ 12A7	
8	ለ 1208	መ 1218	ረ 1228	ሸ 1238	ቈ 1248	ቘ 1258	ቨ 1268	ቸ 1278	ኈ 1288	ኘ 1298	ከ 12A8	ኸ 12B8
9	ሉ 1209	ሙ 1219	ሩ 1229	ሹ 1239			ቩ 1269	ቹ 1279		ኙ 1299	ኩ 12A9	ኹ 12B9
A	ሊ 120A	ሚ 121A	ሪ 122A	ሺ 123A	ቊ 124A	ቚ 125A	ቪ 126A	ቺ 127A	ኊ 128A	ኚ 129A	ኪ 12AA	ኺ 12BA
B	ላ 120B	ማ 121B	ራ 122B	ሻ 123B	ቋ 124B	ቛ 125B	ቫ 126B	ቻ 127B	ኋ 128B	ኛ 129B	ካ 12AB	ኻ 12BB
C	ሌ 120C	ሜ 121C	ሬ 122C	ሼ 123C	ቌ 124C	ቜ 125C	ቬ 126C	ቼ 127C	ኌ 128C	ኜ 129C	ኬ 12AC	ኼ 12BC
D	ል 120D	ም 121D	ር 122D	ሽ 123D	ቍ 124D	ቝ 125D	ቭ 126D	ች 127D	ኍ 128D	ኝ 129D	ክ 12AD	ኽ 12BD
E	ሎ 120E	ሞ 121E	ሮ 122E	ሾ 123E			ቮ 126E	ቾ 127E		ኞ 129E	ኮ 12AE	ኾ 12BE
F	ሏ 120F	ሟ 121F	ሯ 122F	ሿ 123F			ቯ 126F	ቿ 127F		ኟ 129F	ኯ 12AF	


12C 12D 12E 12F 130 131 132 133 134 135 136 137
0 1 2 3 4 5 6 7 8 9 A B C D E F

Appendix B – OLAWP Sequence Diagrams

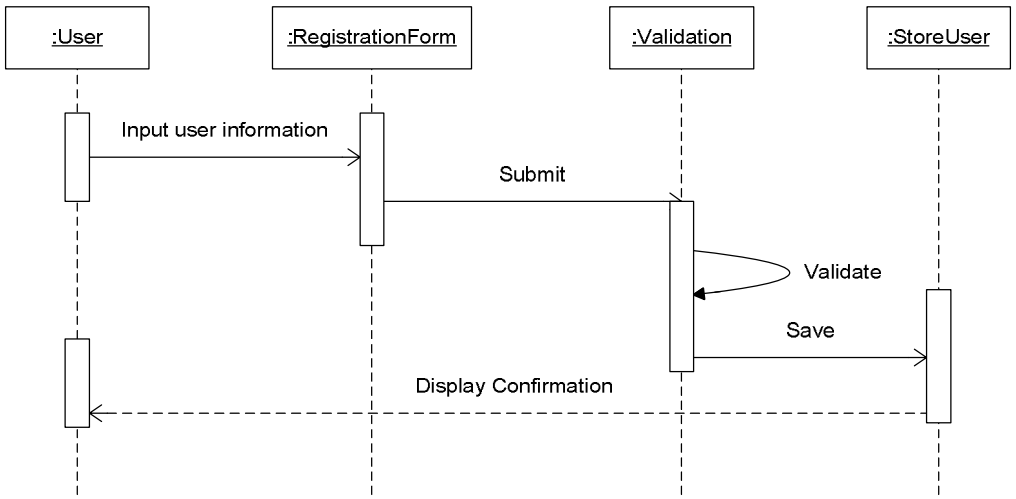


Figure 4.4: *Sequence diagram for Registration use case*

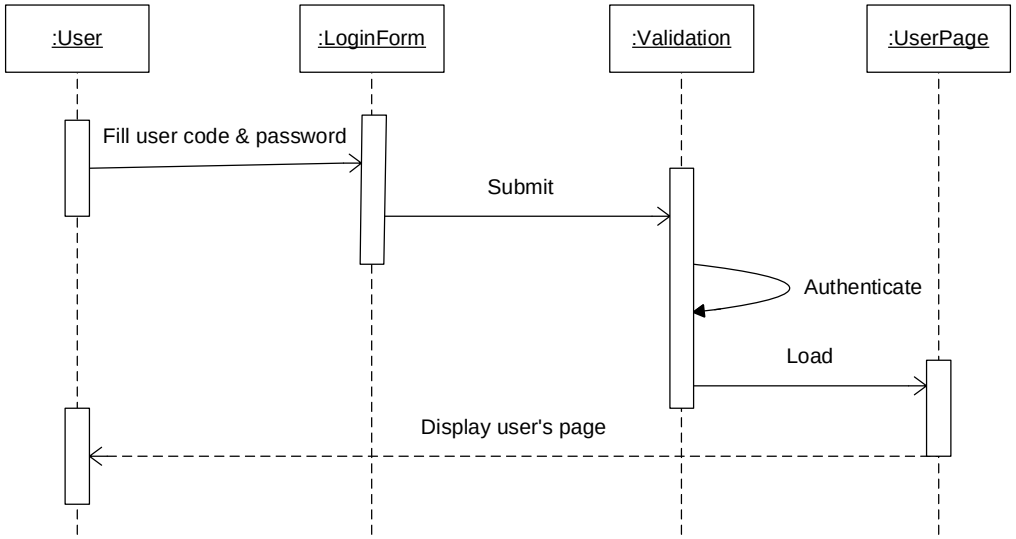


Figure 4.5: *Sequence diagram for Login use case*

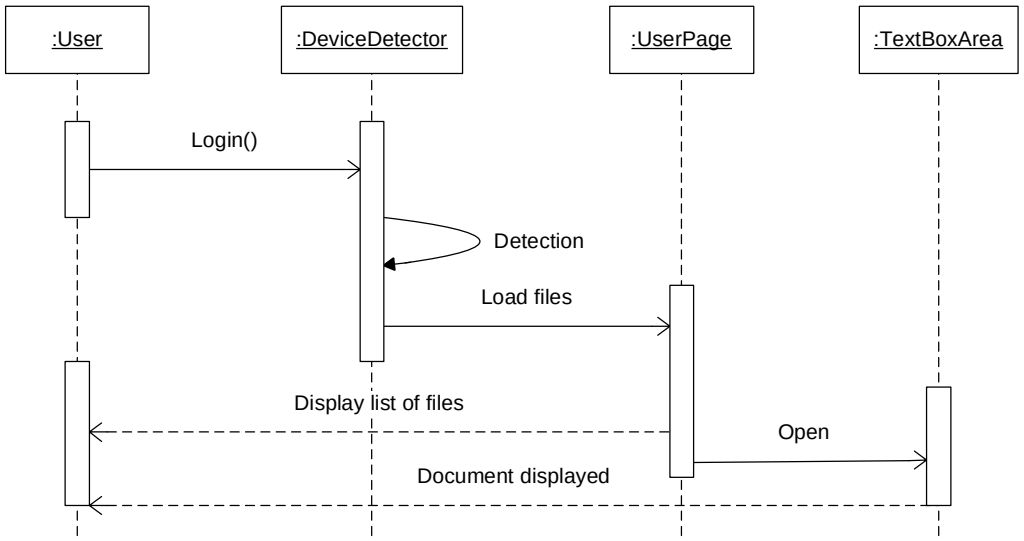


Figure 4.6: *Sequence diagram for View Document use case*

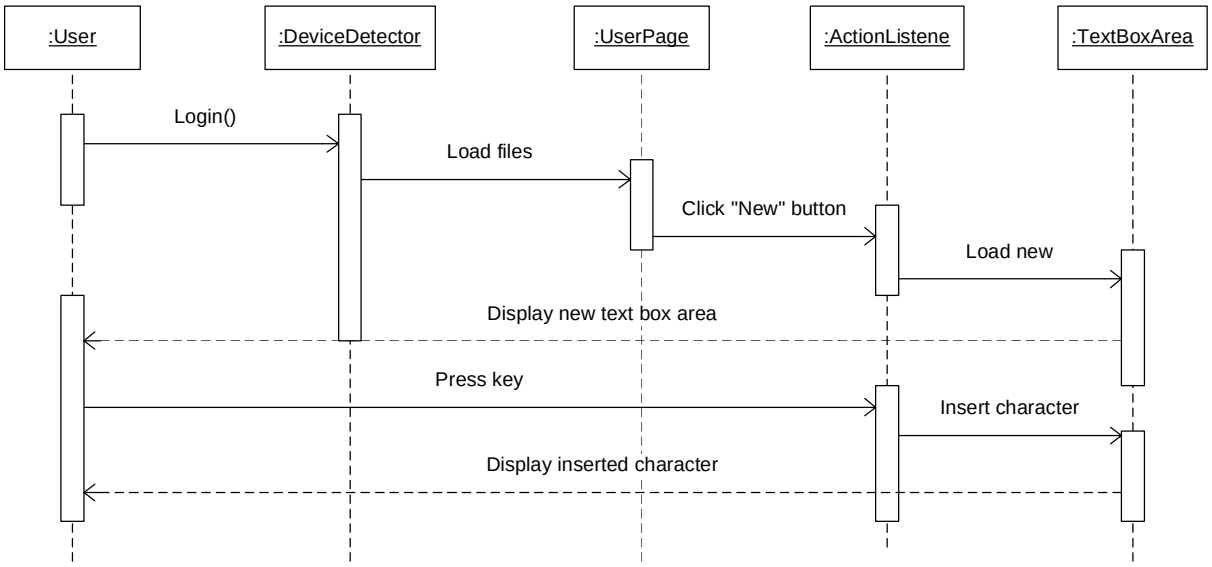


Figure 4.7: *Sequence diagram for Create Document use case*

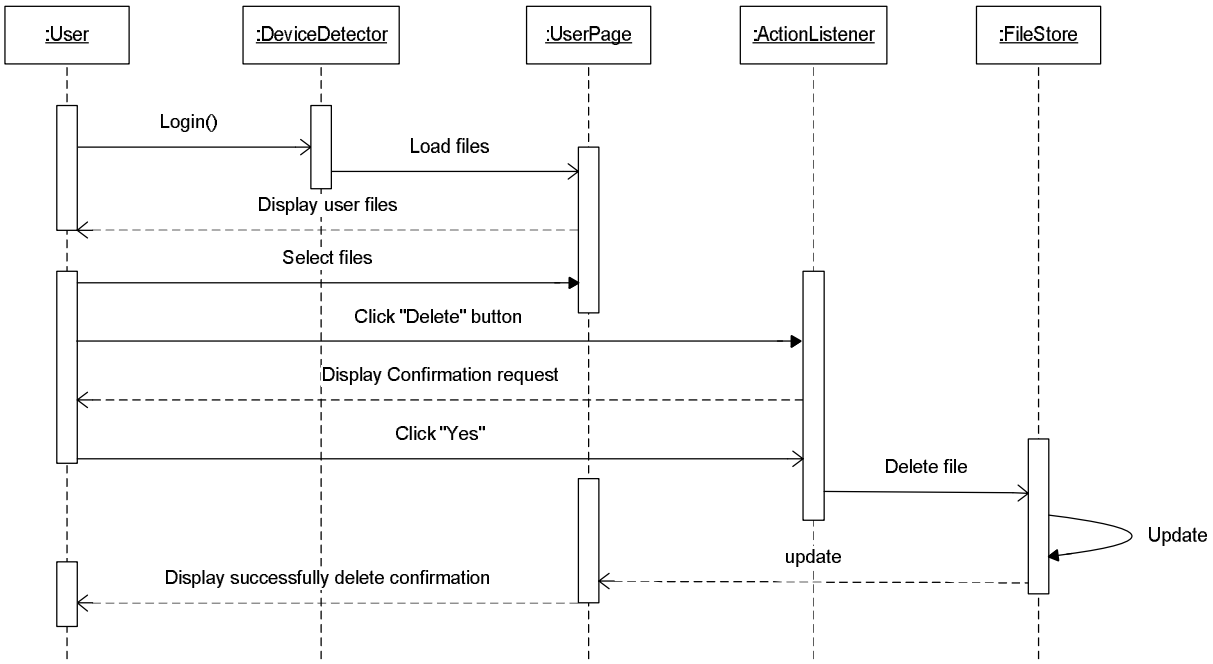


Figure 4.8: *Sequence diagram for Delete Document use case*

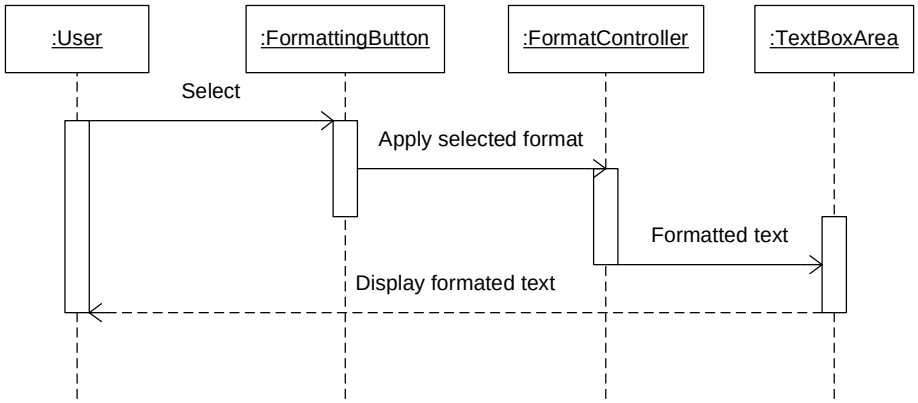


Figure 4.9: *Sequence diagram for Format Document use case*

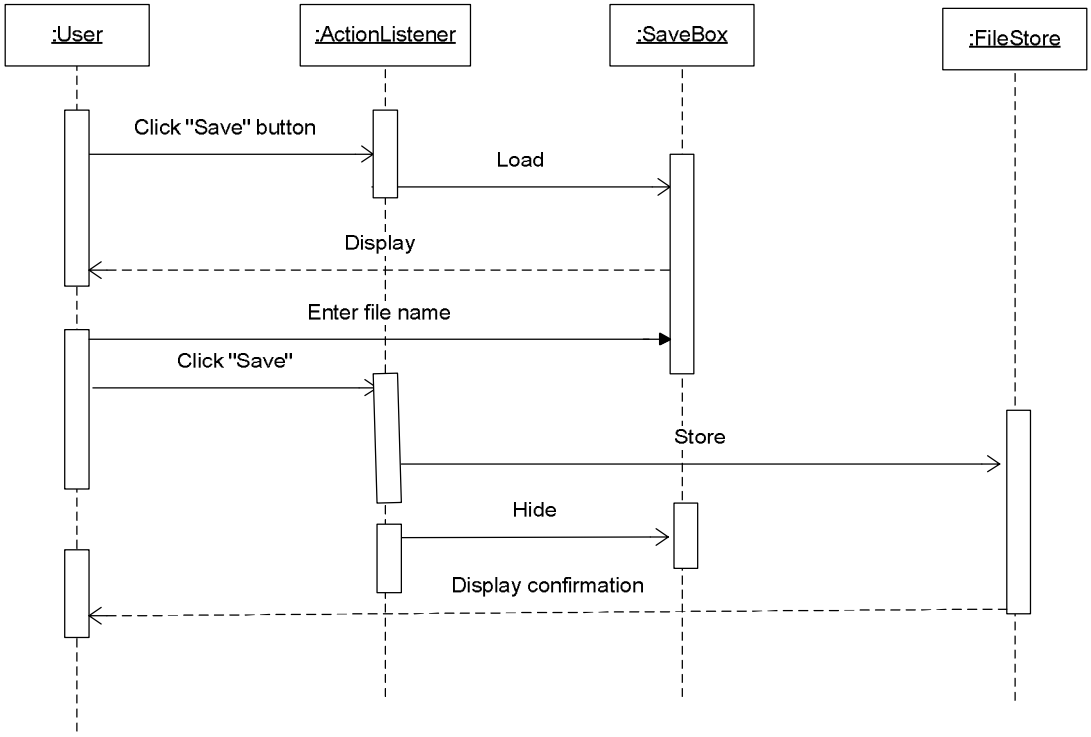


Figure 4.10: *Sequence diagram for Save Document use case*

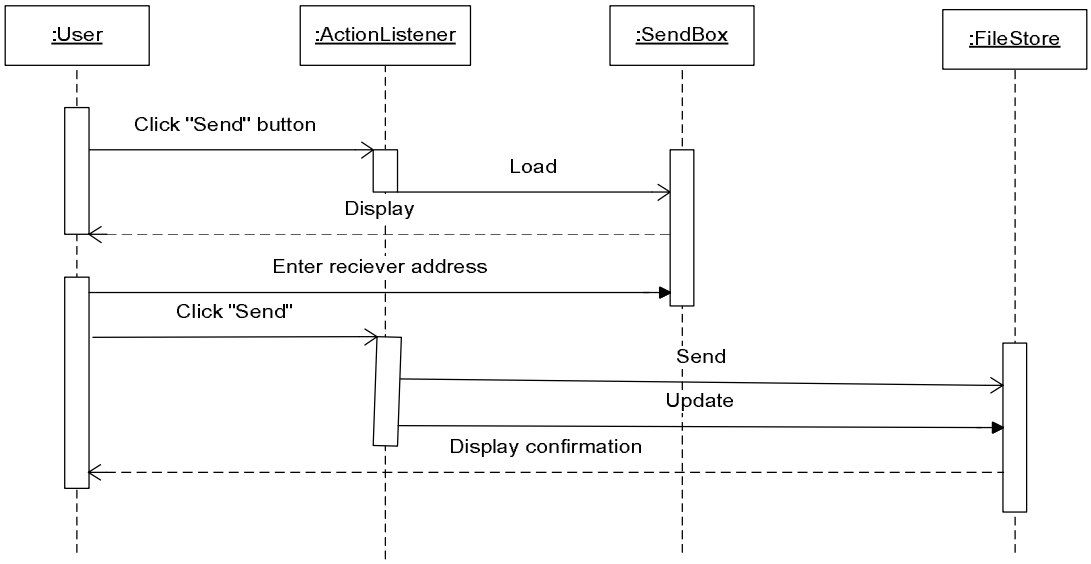


Figure 4.11: *Sequence diagram for Send Document/message use case*

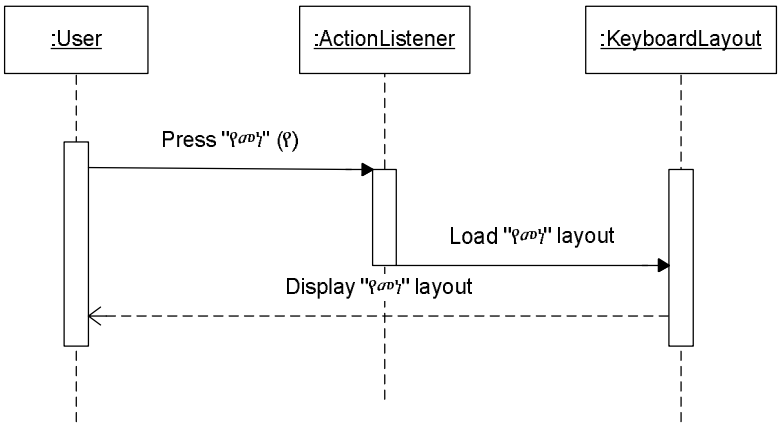


Figure 4.12: *Sequence diagram for Switch Keyboard Layout use case*

By default, the system loads the "ሀለሐ" keyboard layout. If the user wants to use the "የማነ" keyboard layout, the user should press the "የማነ" (የ) switch button.

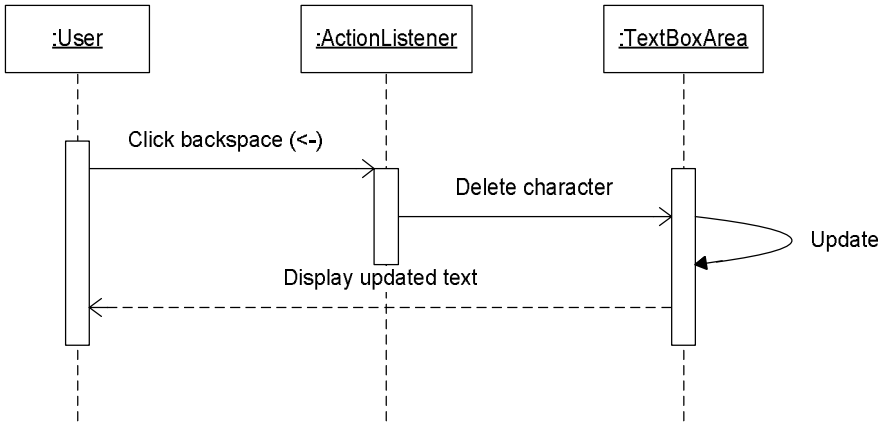


Figure 4.13: *Sequence diagram for Delete Character use case*

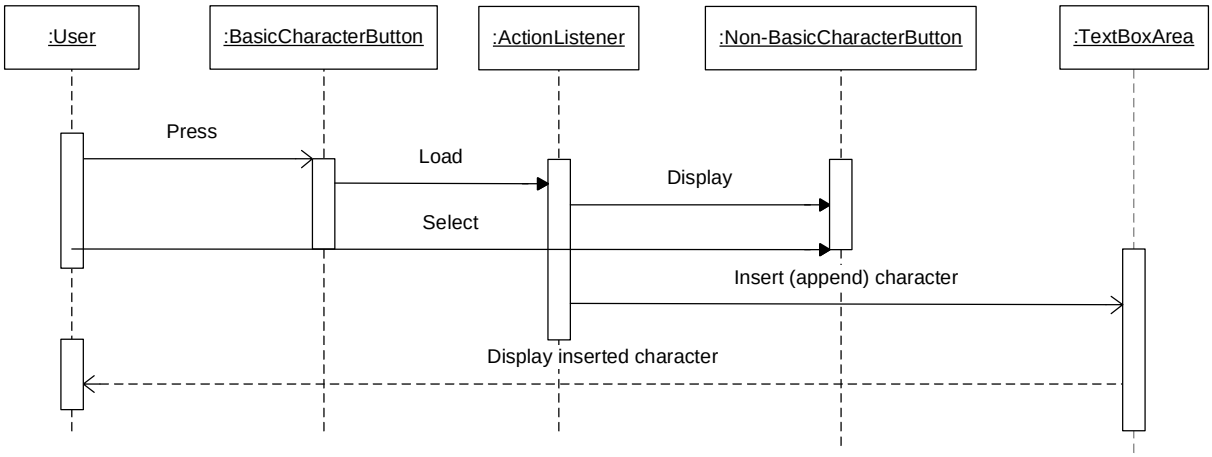


Figure 4.14: *Sequence diagram for Insert Character use case*

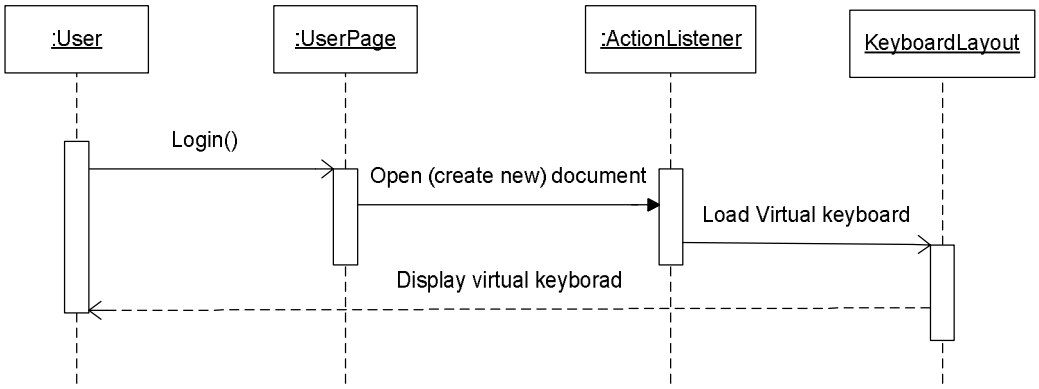


Figure 4.15: *Sequence diagram for Load Virtual Keyboard use case*

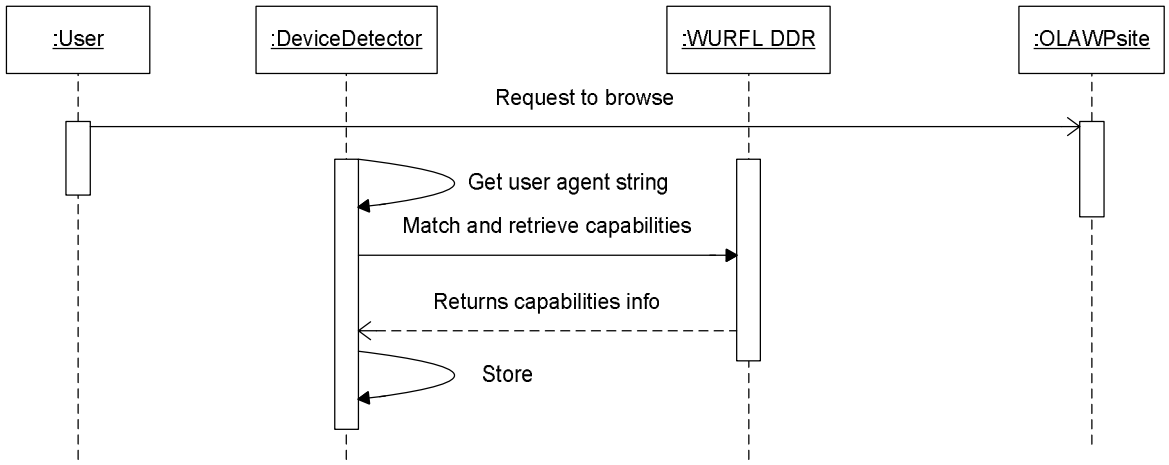


Figure 4.16: *Sequence diagram for Device Detection use case*

Appendix C – Questionnaire

OLAWP usability testing questionnaire which helps to know user satisfaction about the service provided.

I. Background Information about the device and the user

1. Your mobile model name_____
2. Your Mobile phone browser name _____
3. Do you have Internet browsing experience? A) Yes B) No
4. Are you familiar with Amharic SMS software? A) Yes B) No

Prototype

The following items relate the different features of OLAWP. Please indicate your agreement by making "✓" in the boxes.

No.	Questions	Strongly agree	Agree	Neutral	Disagree	Strongly Disagree
1	Do you think the terminology used in the prototype is consistent?					
2	Are you satisfied when you prepare Amharic text?					
3	Do you think the OLAWP prototype can be used easily?					
4	Do you think the system is good enough to prepare Amharic document?					
5	Are you able to format text easily?					
6	Do you think the typing speed using virtual keyboard is satisfactory?					
7	Do you think the typing speed is fast using the virtual keyboard when you compare to the keypad based Amharic text entry method?					
8	Do you think response time for most operations is fast enough?					

9	Does the system give enough description when error happens?					
10	Does the OLAWP prototype interface fits your mobile phone screen?					
11	Do you think the text which appears on the screen is clearly readable throughout the system?					
12	Do you think the interaction to accomplish tasks is simple and complete with a few commands?					
13	Do you think the menu items (or navigation buttons and links) consistently be located and work without failure?					
14	Is your phone capable to run the OLAWP prototype?	Yes B) No				

Please write any other comment about the OLAWP system: _____

_____

_____

_____

Declaration

This thesis work is my original work and has not been presented for a degree in any other university, and that all sources of material used for the thesis have been duly acknowledged.

Name: **Abubeker Yimam**

Signature: _____

Date: _____

Advisor Confirmation:

Name: **Dida Midekso (PhD)**

Signature: _____

Date: _____

April, 2013

Addis Ababa, Ethiopia