IBAANA: THE CROSS-PLATFORM MOBILE INSTANT MESSENGER

BY

Name: Mohammed Balarabe Olateju
Reg No: H00139396
Programme: Msc. I.T (Software Systems)
Year: 2012/2013
Supervisor: Mr. Talal Sheikh

Date: 15/08/2013
I, MOHAMMED BALARABE OLATEJU, confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: ..................................................................................

Date: 15/08/2013
Acknowledgements

All praises and thanks to Allah for His blessings and guidance. First of all, wish to thank my parents for their undying support in powering the achievement my dreams. I also like thank to my supervisor Mr. Talal Sheikh, for his selfless encouragement and motivation. Finally, I will like to thank my uncle, Kabir Folorunsho for his advice, assistance and encouragements on this project.
Abstract

The Ibaana Instant messenger is the idea that humans can interact better using images than with text. It aspires to provide users with the ability to express themselves with real-time image customization, giving them advantage to not only communicate with others but express themselves to each other. The Ibaana Instant messenger also aims to bridge the gap between SMS messaging and Internet messaging to allow users seamlessly communicate regardless of their connection type.

This report explains the motivation, aims and objectives of the Ibaana instant messenger. It goes further to explain how the techniques and technologies used compare to existing products. This report also explains in details, the software engineering decisions, techniques, technologies and processes undergone in the development of the cross platform mobile instant messenger.

Finally a conclusion reviewing the projects with its original aims, objectives and plan, and how the project could be improved in the future is discussed.
# Table of Contents

Acknowledgements ................................................................................................................................. 2  
Abstract ................................................................................................................................................... 3  
Table of Figures ...................................................................................................................................... 6  
1. Chapter 1 [Introduction] ................................................................................................................. 7  
   1.1. Introduction ............................................................................................................................. 7  
       1.1. Aims and Objectives ............................................................................................................... 8  
2. Chapter 2 [Literature Review] ........................................................................................................ 9  
   2.1. Social Network ........................................................................................................................ 9  
   2.2. Expression with Images ........................................................................................................ 12  
   2.3. Instant Messaging ................................................................................................................. 13  
       2.3.1. Architecture ................................................................................................................... 13  
       2.3.2. Protocols ....................................................................................................................... 15  
   2.4. Mobile Applications .............................................................................................................. 16  
       2.4.1. Cross-Platform Development ........................................................................................ 16  
       2.4.2. Usability ........................................................................................................................ 18  
   2.5. Image Processing and Distributed Application Architecture ................................................ 22  
   2.6. Evaluation ............................................................................................................................. 27  
   2.7. Professional, Legal, Ethical and Social Issues ...................................................................... 30  
       2.7.1. Professional Issues ........................................................................................................ 30  
       2.7.2. Legal Issues ................................................................................................................... 30  
       2.7.3. Ethical Issues ................................................................................................................ 31  
       2.7.4. Social Issues .................................................................................................................. 31  
3. Chapter 3 [Software Development Process] ................................................................................. 32  
   3.1. Methodology and Project Plan .............................................................................................. 32  
   3.2. Requirement Engineering Process ........................................................................................ 34  
       3.2.1. Stakeholders .................................................................................................................. 34  
       3.2.2. Requirements ................................................................................................................ 35  
       3.2.3. Backlog ......................................................................................................................... 38  
   3.3. Proposed Overall System Model ........................................................................................... 41  
   3.4. 1st Iteration ............................................................................................................................ 42  
       3.4.1. Sprint Goals .................................................................................................................. 42  
       3.4.2. Design ........................................................................................................................... 42  
       3.4.3. Implementation ............................................................................................................. 43  
       3.4.4. Testing ........................................................................................................................... 51
3.5. 2nd Iteration ................................................................. 52
  3.5.1. Sprint Goals .............................................................. 52
  3.5.2. Design ................................................................. 53
  3.5.3. Implementation ..................................................... 53
  3.5.4. Testing and Deployment ......................................... 59
  3.5.5. Evaluation ............................................................. 60

3.6. 3rd Iteration ................................................................. 62
  3.6.1. Sprint Goal .............................................................. 62
  3.6.2. Design ................................................................. 62
  3.6.3. Implementation ..................................................... 63
  3.6.4. Testing ................................................................. 65
  3.6.5. Evaluation ............................................................. 67

Conclusion ........................................................................... 70
Further Work ......................................................................... 71
References ............................................................................. 72
Appendices ............................................................................. 77
  1. Project Events .............................................................. 77
  2. Screen Shots .............................................................. 80
    2.1. SMS Authentication Message Received to Default SMS Application ......................... 80
    2.2. Database Server Network Output .......................................................... 80
    2.3. Image Processing Server’s CPU Usage ...................................................... 81
    2.4. Sharing Images on Facebook .............................................................. 81
    2.5. The Servers Deployed as Amazon Server Instances ........................................ 82
  3. Evaluation Data Gathering Documents ............................................. 83
    3.1 Agreement Form ........................................................ 83
    3.2 User Evaluation Tasks ...................................................... 84
    3.3 User Questionnaire ........................................................ 84
    3.4 User Interview ........................................................... 85
    3.5 User Observation Form ...................................................... 86
    3.6 Expert User Questionnaire .................................................... 88
    3.7 Expert User Interview ........................................................ 89
  4. Project Log ..................................................................... 91
  5. Evaluation Result Materials (Only Available in Hard Copies) ..................... 125
### Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AIM System Architecture used by the AOL Messenger</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>A Comparison of Supported APIs of Cross-Platform Tools</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>A Comparison of the four design principles</td>
<td>19</td>
</tr>
<tr>
<td>4</td>
<td>Three Layer Design Guideline for Mobile Application</td>
<td>21</td>
</tr>
<tr>
<td>5</td>
<td>A separation of Image information and Image in Server and Database Respectively</td>
<td>23</td>
</tr>
<tr>
<td>6</td>
<td>The Five GIMP Layers</td>
<td>24</td>
</tr>
<tr>
<td>7</td>
<td>Suggested Service Oriented Architecture of An Image Processing web service</td>
<td>25</td>
</tr>
<tr>
<td>8</td>
<td>The Apache ZooKeeper Application Service</td>
<td>27</td>
</tr>
<tr>
<td>9</td>
<td>Distributed Evaluation Framework Workflow</td>
<td>28</td>
</tr>
<tr>
<td>10</td>
<td>Project Plan Showing The Aspects Of Each Iteration</td>
<td>33</td>
</tr>
<tr>
<td>11</td>
<td>Power-Interest Matrix for Stakeholders</td>
<td>34</td>
</tr>
<tr>
<td>12</td>
<td>Stakeholder Power Matrix For the Ibaana Messenger</td>
<td>35</td>
</tr>
<tr>
<td>13</td>
<td>Proposed Overall System Architecture Before The Start of The Project</td>
<td>41</td>
</tr>
<tr>
<td>14</td>
<td>Assumed Architecture to be created for the 1st Sprint</td>
<td>43</td>
</tr>
<tr>
<td>15</td>
<td>Mobile Client Code for connection to IM Server</td>
<td>44</td>
</tr>
<tr>
<td>16</td>
<td>Storing Session Info Into Mobile Device database using Phonegap API</td>
<td>45</td>
</tr>
<tr>
<td>17</td>
<td>Web Application Using the File Upload library &amp; Sockets to transfer data</td>
<td>47</td>
</tr>
<tr>
<td>18</td>
<td>Overall Architecture of the Ibaana Instant Messenger After 1st Sprint</td>
<td>48</td>
</tr>
<tr>
<td>19</td>
<td>Login Screen</td>
<td>49</td>
</tr>
<tr>
<td>20</td>
<td>Chat Screen</td>
<td>49</td>
</tr>
<tr>
<td>21</td>
<td>Image Processing Dialog Form</td>
<td>50</td>
</tr>
<tr>
<td>22</td>
<td>Confirmation On Sending Processed Image</td>
<td>50</td>
</tr>
<tr>
<td>23</td>
<td>Processed Image Displayed on Receiving User’s Screen</td>
<td>51</td>
</tr>
<tr>
<td>24</td>
<td>Project Plan After The 1st Iteration</td>
<td>52</td>
</tr>
<tr>
<td>25</td>
<td>Routing Server Forwarding Data After Receiving-{ Figure 17 Full Code}</td>
<td>53</td>
</tr>
<tr>
<td>26</td>
<td>IP Server Sending &amp; Receiving Data Structure In A Predefined Form</td>
<td>54</td>
</tr>
<tr>
<td>27</td>
<td>Code that adds and Styles the Buttons for the Home.html file</td>
<td>56</td>
</tr>
<tr>
<td>28</td>
<td>Screen Shot Showing the Sample Text Applied to An Image</td>
<td>58</td>
</tr>
<tr>
<td>29</td>
<td>Overall Architecture after the 2nd Sprint</td>
<td>58</td>
</tr>
<tr>
<td>30</td>
<td>Comparison Between The Whatsapp IM(left) and The Ibaana IM(right) Interfaces</td>
<td>61</td>
</tr>
<tr>
<td>31</td>
<td>Proposed Architecture For The 3rd Sprint</td>
<td>62</td>
</tr>
<tr>
<td>32</td>
<td>Initializing Embedded Server and Adding A Custom Servlet</td>
<td>64</td>
</tr>
<tr>
<td>33</td>
<td>Overall Architecture of the Ibaana IM after the 3rd Sprint</td>
<td>64</td>
</tr>
<tr>
<td>34</td>
<td>Questionnaire Results for Experts</td>
<td>68</td>
</tr>
<tr>
<td>35</td>
<td>Questionnaire Results for Users</td>
<td>69</td>
</tr>
</tbody>
</table>
1. Chapter 1 [Introduction]

1.1. Introduction

The 21st Century has ushered in a wave of social network platforms and networks for users to engage and interact with each other like Facebook, WhatsApp and Instagram. Social networks such as Facebook and Instagram are mostly activity-based social networks where users post information based on their daily lives and experiences. Instant Messaging platforms like WhatsApp, Skype and Blackberry messenger are also a means where users can interact with each other mostly by using text, pictures or videos. These social networks allow users to communicate, interact and keep in touch with their friends, family over long distances in different ways. The missing thing with these forms of social networks is the fact that none of them provides a better way for users to express themselves other than the basic form of communication with the use of text messages, externally edited pictures and videos. Although technologies like Instagram provide users with the ability to manipulate images using filters, while Instant messengers like Skype allow users to communicate with videos. None of these technologies provide the right blend of support for users to express their thoughts, feelings and expressions by manipulating images in real-time.

“Ibaana” which means expression is a Mobile Instant Messenger that aims to provide users with the platform to express themselves better during their interaction with other users through image manipulation. The motivation behind this idea can be expressed in the saying “A picture is worth a thousand words”. Users should not be constricted to having to only communicate with finalized text, pictures or videos but they should be able to easily express what they feel and thought. There has been an attempt to do so with the use of emoticons where users can tell other parties how they feel with emoticons like smiley and angry emoticons. Despite this, users cannot express their thoughts all the time as it might be more complex than a simple emoticon. The Ibaana messenger can provide this, giving users the means to create, manipulate and customize images in real-time and sent them to other users without the need for advance image editing skills.

User interaction today is a deeply divided one when it comes to the means of communications as users are segmented between SMS or internet-based interactions during communication. When users want to communicate with each other they have to choose between either SMS or internet-based communication which has left out the sections of people that prefer SMS communications out of the social network. Ibaana Messenger plans to bridge the gap between these two means of communication by providing a way for SMS users to communicate directly with users with internet subscriptions and vice versa. This will not only bring back together these types of users but also reduce the over reliance and dependency of social networking on internet-based communication.
1.1. Aims and Objectives

The main aim of this project is to improve instant messaging interaction between users by providing users with ways to create, manipulate and customize images and send to other users in real-time. It also aims to users share the images that they were able to create on social networks like Facebook and Twitter.

An additional aim of this project is to bring together SMS users and internet users thereby reducing the dependency on internet based communication.

Although mobile image processing applications already exist, this project’s orientation is to improve on these applications by providing image manipulations in real time as users communicate with each other. That means users do not have to go back and forth between applications to manipulate, save, select then send the images to each other. The Ibaana mobile instant messenger application also tries as much as possible to reduce the load on the mobile device thereby allowing faster image manipulation regardless of the client device specification. The approach towards achieving this is to provide manipulation of the image on a remote server and send it back to the mobile device seamlessly. The ultimate result of this would be high user experience and convenience.
2. Chapter 2 [Literature Review]

This chapter uses extant literature to delve in and compare the impacts existing products have had on the modern world and draws out the gaps the Ibaana Instant Messenger aims to fill. Existing techniques, methods and related products are also explained, and how this project aims to utilize them by either building on them or choosing and alternative technology.

2.1. Social Network

Social networks have had a great impact on the modern world. To comprehend just how much impact it as had, first the evolution of the social network platforms needs to be examined to understand the needs of users that caused changes in social networking. The histories, growth, monetary worth, negative and positive impacts of social networks also need to be explored to foster a proper understanding of today’s social networking environment.

Social networks began from personal computers with the use of instant messaging applications. The concept of instant messaging started in the 1960’s from institutions like the Massachusetts Institute of Technology (MIT) transferring text-based information between computers. The first instant messenger to be introduced to the public was the ICQ instant messenger developed by Mirabilis in 1996. AOL introduced the AIM instant messenger to public users in 1997 which gathered 53 million users 8 years later. Yahoo messenger was introduced in 1998 and AIM started introducing new features whereby user could not only join chat room service but also transfer files between each other (Pentrozio, 2012).

Instant messengers were limited because the only way to discover new contacts was to join chat rooms. From 2000 to 2005 a number of web based social networks such as Friendster, LinkedIn, MySpace and Facebook where created with the idea of bringing users with similarities together. These technologies allow users with similar interests and backgrounds to find each other out without the need for the installation of any software apart from the web browsers used to access them. For example, LinkedIn allows users to find other individuals with similar business interests or universities to interact and share information with each other. Social networks like Facebook even allow other developers to create applications on their platforms that can be used by the users on the network (Goble, 2012). In 2008, Facebook introduced a web chatting application known as Facebook Chat allowing users to send web based instant messages to each other (Pentrozio, 2012).

Facebook was created by Mark Zuckerberg in 2004 while he was studying at Harvard University, it was developed after he created other social networks like face mash and course match. The idea
behind Facebook was to allow individuals share their personal profile and information (Phillips, 2007). Facebook users grew from a number of users in educational institutions when it started to 30 million users in July 2007 (Phillips, 2007), today there are over 1 billion users on Facebook (Tam, 2013). In November 2010, Facebook’s estimated worth was 41 billion dollars (Womack, 2010) and in January 2012 Facebook was estimated to be worth 83 billion dollars (Protalinski, 2012). Facebook has also influenced the way people communicate on the internet where a piece of information posted on the social network can be spread as news feeds to other viewers in a few minutes which could make it viral on the internet. This has made Facebook a very important means of communication for people around the world today. Facebook has also brought about negative effects on its users like the constant comparison of users’ lives which sometimes leads to low self-esteem, it is also reported to have increasingly made its users narcissistic (Gill, 2013).

Before 2005, users mostly interacted and shared information using text and images. PayPal Co-workers, Chad Hurly, Jewed Karim  and Steve Chen created YouTube in 2005 as a website where users could upload and share personal videos on the internet for free. The uploaded videos could be viewed by any user without having to be downloaded or requiring any special software to be installed on the users’ pc (Fitzpatrick, 2010). The ability of people to be share upload personal videos instantly was seen as a new way of communicating. This way of communicating has also brought about some social effects like posting of violent videos by individuals to give an impression on their peers. YouTube has also been used in divorce conflicts where spouse upload demeaning videos about each other and even political events are sometimes affected by the videos shared on YouTube. For example, John McCain, the American presidential candidate for the 2008 election, had a video uploaded on YouTube showing him giving a statement that he was comfortable with the American military staying in Iraq for 100 years (Margolis, n.d.). This video no doubt had an effect on the result of the election. Google acquired YouTube for a reported sum of 1.6 billion dollars in 2006. In 2010, YouTube was rated as the 3rd most viewed website in the world, only Google and Facebook had more page visits than it (Fitzpatrick, 2010).

Twitter is a micro-blogging internet service created in 2006, it involves users sharing small text-based pieces of information similar to SMS messages. It was created and co-founded by Jack Dorsey and Noah Jack, the unique selling point of twitter is that is used to inform other users about current events and activities in small pieces of text that represented useful trending information called twits (Malik, 2009). In 2007, almost 5 thousand tweets and 300 thousand messages where sent on twitter, in 2010 the statistics grew to 50 million tweets per day (Beaumont, 2010) and in 2012, 200 million active users where registered on twitter (O’Carroll, 2012). According to Crunchbase(n.d), twitter has received an approximate funding of 1.16bn dollars. Twitter can be used to find out about trending news and events as well as finding individuals with the same way of thinking. It also helps individuals
in businesses to portray themselves or their business in profitable ways (Jeanty, 2012a). The negative impact of Twitter is that it increases users’ quest for fame and popularity since it works mainly on followers. Twitter according to Jeanty (2012b), more than any other social network, drives users to self-obsession (Jeanty, 2012b).

Instagram is a mobile application launched in 2006 developed by Systrom and Mike Krieger. It allows its users take pictures, apply image filters and share them on social networks like Facebook and Twitter including its own network (Tolentino, 2012). Instagram is a great way not only for users to know what their contacts are doing but to see it with the help of the shared photos. In 2012, Instagram announced that 150 million photos had been shared (Instagram, n.d(a)) and it had about 30 million users on its social network (Instagram, n.d(b)). The criticism against Instagram is that now users do not need to be professional photographers to take world-class images which could affect the photography industry (O’Donohue, 2012). In 2012, Facebook announced the acquisition of Instagram for a reported sum of 1 billion dollars just weeks before Facebook’s IPO (Primack, 2012).

Mobile platforms serve as an extension for already existing social networks on web and desktop platforms which means social networks providers can grant their services on instantly mobile devices without the necessity of having desktop or personal computers. Social networks like Twitter and Facebook have created applications for mobile access, making the network easily available from almost anywhere and they extended features like instant messaging to the mobile platforms (Pentrozio, 2012). This advantage has made mobile access to social networks more popular than regular web access through computers with Facebook reporting more user access to its network through phone than computers in January 2013 (Kelly, 2013). Mobile platforms also boast of an array of mobile instant messengers like Blackberry Messenger, WhatsApp and KiK that have gained a lot of popularity in recent years (Vermeulen, 2012).

Social networks have a lot of appeal from a large number of end users and have also attracted investment from businesses that see social networks as a platform for future opportunities. Social networks have also not only changed the way information, media and ideas are shared and received, but have change the speed at which they can be viewed by other social network users. Today social networks can be accessed through mobile, web and pc platforms with every platform having its own distinctive feature and advantages in serving its users. The shift of social networks went from desktop instant messengers applications to web sites. Now with more mobile devices in the market, it is possible that mobile devices are the new platforms for social networks.
2.2. Expression with Images

The Ibaana messenger aims to enhance image based communication to further facilitate interaction between users. An in-depth research into the differences between image and text based communication is discussed to determine which is better for expression.

Text messaging is a fundamental mode of communication where short messages can be sent from one individual to another using a variety of devices and technologies. In the last 10 years text messaging has grown from 12 million to 135 billion messages sent every month. The increasing text messaging is also as a result of new technologies like instant messaging, Facebook and Twitter (Battestini et al., 2010). The large scale analysis on text messages done by Battestini et al. (2010) illustrates that individuals use text message as the basic way of interacting with each other over audio and video communication especially for short messages that do not require long durations of communication.

Photo Sharing has become one of the pre-dominant modes of interaction in social networks with almost every social network from instant messengers like the Blackberry Messenger to social networks like Facebook and Twitter adopting photo sharing technologies to allow better interaction with friends and followers. House (2005) suggests that photo sharing not only allows individuals express their own unique view of the world by sharing important life events, ideas and philosophy, but it is also a tool for social presentations, to give friends and contacts an idea of how individual see themselves. Photo sharing can also be a way sharing complex information in a simple way where text messages would require far more interaction. For example, a group leader sending an image of a clock to group members to emphasis and illustrate a fast approaching deadline (House N, 2005).

Makela et al. (2000) conducted a field study on image base mobile interaction by providing a platform for user to interact with images. Among the results of the field study is the discovery that individuals could express messages, stories, illustration and jokes far better with images than text which makes it a more viable tool for socializing. Another discovery from the study was that some of the individuals especially children prefer editing images rather than just taking the pictures for events and sharing them with their friends and family. Individuals at the age of fifty and above also prefer expressing themselves but in more artistic forms, and they also enjoyed using images with their family to play games and keep up with their family’s daily lives. Makela. et al. (2000) also discovered that the individuals engaged in the field study, needed more advance skills to illustrate what they wanted to express in some cases. A number of the working individuals preferred more straight forward and functional interaction like setting a time for meetings using text messages since image interaction required more time. Some messages illustrated in images could not be understood except when the
receiving parties understood the individual or the context of the message. This factor could be used to send secret messages that can only be understood by specific people. From the interaction of the subjects, it was gathered that in some situation text annotation within images is unavoidable to fully express and convey information to the other parties (Makela et al., 2000).

Although expression of feeling and ideas in text-messages have improved in recent years due to ideas emoticons, image based interaction is a more precise way expression but sometimes even image interaction need text annotation to fully give meaning to the message. These features can be combined in the Ibaana Instant Messenger to make sure all areas of interaction are adequately covered.

2.3. Instant Messaging

Investigating the methods that are used today to create instant messengers is imperative to the development of this project. To give a solid foundation on which to build and to provide option for making decision on the project, the different architecture and protocols used in developing popular instant messengers explained below along with the advantages and disadvantages of each aspect.

2.3.1. Architecture

The overall architecture of an instant messaging application is one area that always needs to be planned before its implementation to ensure easy scalability and optimal performance. To do this, various architectural approaches of already existing instant messengers need to be investigated and examined, doing this eliminates future obstacles that might be encountered during the development and deployment processes.

In the creation of instant messengers two standard approaches have been used by three of the most widely used instant messengers, these approaches are the asymmetric and symmetric client-server architectural approach (Jennings et al., 2006). The asymmetric approach used by AOL and MSN Messengers involves assigning various functions to different servers. For example, the AIM System Architecture used by AOL Messenger consists of servers providing individual functions from User Authentication to chat room services as shown in 0. The symmetric approach used by Yahoo Messenger, where a single server node from a cluster performs all the operations, therefore users only need to connect to one server to access all the services.

The advantage of using the asymmetric approach is that scaling the instant messaging applications would be easy since all the parts of the application are modularized. The symmetric approach makes
application deployment easy as the requirement will remain the same for the systems. In addition to having client server models Instant Messenger providers also make use of the peer to peer model where users communicate with each other directly without a server as an intermediary.

The advantage of using client server model is that the instant messenger providers have the power of communications between users which makes them overcome technical issues. On the other side, the peer to peer model causes less traffic as users don’t have to go through the central servers to communicate, this advantage is why instant messaging providers provide the peer to peer model for users during voice-chat communications (Jennings et al., 2006). These factors will have considered for each component during the software development processes of the Ibaana Instant Messenger.

Figure 1. **AIM System Architecture used by the AOL Messenger.**

*(Jennings et al., 2006, p.18)*
2.3.2. Protocols

The protocols used to create Instant Messengers also determine the performance of the applications therefore to successfully create an industry standard mobile instant messenger (IM). An exploration into the cutting-edge technologies being used today is mandatory for this project.

Instant Messenger providers today use their own protocols to develop their IM like the Yahoo Messenger application that uses the YMSG protocol and the Windows Live Messenger which also uses its own protocol called MSNP protocol (Khoshbakhtian et al., 2008). There are sets of open Instant messaging protocols that have been developed for public use, the two competing standards according to Jennings, et al (2006) are SIMPLE and XMPP.

The SIMPLE is an extension that was developed by the IETF Group which aims to add instant messaging functionality to the SIP protocol. The IETF specified that the SIMPLE protocol uses three methods to execute functions namely subscribe, notify and message. The subscribe function is used to find out the availability of another user on the network while the notify function is used to transfer the availability information of a user on the network and the message function is used to send instant messages to other users on the network (Debbabi and Rahman, 2004). The SIMPLE protocol consists of two models of messaging and chatting namely the session model and pager model. The session model is used when long piece of information is sent from one user to another while the pager model which is less complex, is used to send short messages (Jennings et al., 2006).

XMPP is an XML based instant messaging technology with three basic functionalities called stanzas. The three stanzas are presence, message and IQ. The presence stanza is used to broadcast a user’s availability while the message stanza is used to send messages between users and the IQ stanza is used to send requests and responses between users. The XMPP protocol has undergone various improvements due to its community support after it was first introduced in 1999. A collection of extensions called XEP (XMPP Extension Protocols) have been introduced making XMPP a versatile technology to be used from interactive social networks applications to cloud computing (Hornsby and Walsh, 2010).

Lu, Lei, and Zhang (2012) designed and implemented an SMS gateway using the XMPP protocol that was able to send SMS from a GSM modem to a serial line. With enough effort, this implementation can be used on cross-platform mobile applications providing the vision of cross-network communications of the Ibaana messenger a boost.
In Terms of the Protocol, the XMPP protocol’s large community, programming languages support and scalability make it a better choice for developing the Ibaana instant messenger when compared to the SIMPLE protocol.

### 2.4. Mobile Applications

In developing Mobile applications, not only is the right blend of development tools needed to achieve a quality product but also mobile applications must provide usability and good users experience. A comparison of the tools needed to develop cross-platform applications and the design principles for creating user friendly mobile applications are discussed.

#### 2.4.1. Cross-Platform Development

In creating mobile applications, users’ expectations are the most crucial of factors as they always expect the highest quality even for free applications. In recent years, there have been growing demands for mobile applications on all the popular mobile platforms, which in turn has caused rapid development in the field of mobile software development. Due to these events, careful considerations are needed when selecting the tools and approaches to develop a mobile application.

Application development for today’s smartphone is a heavily fragmented one, where the mobile manufactures have completely different Application Programming Interfaces (APIs) from each other. This requires variety skill sets and familiarity with different programming languages in various runtime environments (Smutný, 2012). For these reasons, creating a cross-platform mobile application can be time consuming and also expensive but with specific tools and environments, these conditions can be improved. With newly introduced tools, developers can now use specific more widely used API like HTML, CSS and JavaScript on different platforms. This not only reduces the cost and time taken to develop an application, but also the ease of development and deployment (Yancy, 2013). Although these new tools have been adopted by organisations some disadvantages still exist like the lack of support for some operating system features, tendency of some applications to run slowly and the limited support for 3D and graphics (Yancy, 2013).

Four of the most popular mobile cross-platform development tools that support the major mobile operation systems platforms are PhoneGap, RhoMobile, DragonRad and MoSync, each of these tools have their own unique distinction depending on the requirement of the application to be developed.
The factors to be compared when choosing any of the above mentioned tools are the development features, supported APIs, development environment and architecture.

<table>
<thead>
<tr>
<th>API Name</th>
<th>Rhodes</th>
<th>PhoneGap</th>
<th>MoSync</th>
<th>MoSync</th>
<th>DragonRad</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accelerometer</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Barcode</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bluetooth</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Calendar</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Camera</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Capture</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Compass</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Connection</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Contacts</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Device</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>File</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Geolocation</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Menu</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Nickel</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Notification</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Screen Rotation</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Storage</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

**TABLE IV**

**COMPARISON ON MAIN SUPPORTED API’S.**

*Figure 2. A Comparison of Supported APIs of Cross-Platform Tools.*

*(Palmieri et al., 2012, p.186)*

Development features are the programming languages that must be used to develop applications using a specific tool. For example, MoSync, RhoMobile and PhoneGap use JavaScript, HTML and HTML5 and CSS but the MoSync tool has a version that supports C and C++ development (Palmieri et al., 2012). The DragonRad tool has to be used with its native language called D&D which could reduce its chances of adoption in contrast to other tools due to the required learning curve.

The supported APIs are the features each tool can make use from the mobile operating system, for example, mobile operating system features like Bluetooth and storage features can be utilized using the Rhodes tool. PhoneGap has the highest number of supported APIs compared to other tools having all the supported APIs apart from the menu feature which makes it a good choice for the development of multi-featured applications. The Rhodes tool supports the menu features although it supports three less features to the PhoneGap tool. DragonRed supports seven less features to the PhoneGap tool and the MoSync tool supports five to four less features to the PhoneGap tool with its Javascript and C/C++ versions respectively (Palmieri et al., 2012).
RhoMobile and DragonRad tools come with their own development environments known as Integrated Development Environment (IDE), the RhoMobile and DragonRad IDE’s are called RhoStudio and DragonRad respectively. The PhoneGap tool is not accompanied by an IDE and needs to integrated with other IDE’s to be deployed on a specific platform, for example, it has to be used with Visual Studio IDE to be deployed on a windows mobile operating systems platform. The MoSync tool comes with an IDE’s built on the Eclipse IDE (Palmieri et al., 2012). The PhoneGap can be time consuming for software developers that may want to deploy application on multiple platforms compared to other tools.

All the cross-platform tools that have been mentioned have architectures that support the transformation of the specific code used to develop application to the native code of the mobile platform but none supports as much software engineering principles as the Rhodes tool. The Rhodes tool makes use of Model-View-Controller (MVC) design pattern and also an ORM Framework called Rhom. This feature gives the Rhodes tool an upper hand in the development of enterprise applications for mobile platforms when compared with other tools (Palmieri et al., 2012).

The Phonegap tool in relation to the Ibaana Instant messenger has the highest advantage as it supports the most API’s for mobile features.

2.4.2. Usability

Designing a usable cross-platform mobile application that provides rich user experience is a tough job due to the large number of various mobile devices. These mobile devices are different in almost everything, from the features they provide, to the design of the user interfaces and hardware. For example, hardware designs for smartphones mainly come in three categories namely touchscreens, on-board keyboard and touchscreens with on-board keyboards. This diversity makes it difficult for principles to be derived on how to make cross-platform mobile applications to deliver a high level of user experience without tutorials provided to the users. The common mistakes made in designing mobile applications also have to be investigated so that these mistakes can be easily avoided in the development process.

Norman and Nielsen (2010) identified that due to the lack of enforcement of usability standards by mobile device manufacturers such as Apple and Blackberry, mobile software developers are free to whatever they want, which has been misused. Norman and Nielsen (2010) also discovered that the emergence of gestural or touch screens have brought about new problem in human computer interaction (HCI) not only from third party developers but also mobile manufactures. An example of a
usability problem is the invisibility of the delete gestural control in the email application of the Apple iPad’s interface. Another prominent problem is the lack of non-destructive and consistent functions. For example, the gestural control to delete an email is nowhere to be found when reading the email but is available when viewing all emails. This feature is an inconsistent design as it also should be applicable in other areas, in some cases users do not know it even exists. There is also the fact that there is little or no “undo” functions present in mobile devices compared to traditional GUIs. To solve these problems, HCI principles like feedback, discoverability and reliability need to be applied to mobile device interfaces like they have been to traditional GUIs (Norman and Nielsen, 2010).

<table>
<thead>
<tr>
<th>Existing Guideline</th>
<th>HCD</th>
<th>SGR</th>
<th>7UG</th>
<th>W3CBP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specify context of use</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specify user &amp; organizational requirement</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Produce design solution</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Evaluate design against user requirements</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Enable frequent users to use shortcuts</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Offer informative feedback</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Consistency</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Reversal of actions</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Error prevention and simple error handling</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Reduce short-term memory load</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Design for multiple and dynamic contexts</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Design for small devices</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Design for speed and recovery</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Design for “top-down” interaction</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Allow for personalization</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Design for enjoyment</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Don’t repeat the navigation on every page</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Clearly distinguish selected items</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

HCD – Human-Centred Design
SGR – Shneiderman’s Golden Rules
7UG – Seven Usability Guideline
W3CBP – W3C Mobile Web Best Practices

Figure 3. A Comparison of the four design principles

(Ayob et al., 2009, p.428)

HCI is a field that aims to provide users with a system that provides high usability and user experience by defining principles for designing user interfaces. Ayob, et al (2009) developed a three step guideline on how to develop mobile applications using the Shneiderman’s Golden Rules of Interface Design, Seven Usability Guideline for Mobile Device, Human-Centred Design and Mobile
Web Best Practices design principles. The Shneiderman’s golden rule of interfaces design was developed for traditional user interfaces but could be adapted for mobile interfaces. It involves design rules like reducing short-memory load, design dialogs to yield closure and design for enjoyment. The seven usability guidelines for mobile devices by Abid Warsi concentrates on mobile specific issues. It has guidelines like the prevention of repeated navigation on every page, displaying of essential items only and clear distinguishing of selected items. These seven guidelines are very important as they are geared towards solving basic problems with mobile interfaces. The Human Centered design principle is used to design user interfaces according to perceived tasks which the user will be performing, this approach makes sure the user requirements are met not only in functionality, but with the user interface. The Mobile Web Best Practice by the world wide web consortium is designed to make sure that mobile interfaces are simple and provide navigation from each section to the other. The use of these principles provides a guide toward making more usable user interfaces by going through each aspect of user interface design as shown in the figure below.

The guidelines are divided into Analysis, design and testing, each part has specific design principles that need to be considered. For example, in the analysis stage, a context of use must be considered which means the developers need to consider tasks, the environment and the use of system by the user before going into the design of the user interface. The design stage involves considering the device where the application would be used and creating appropriate designs to suite the devices. It also involves applying HCI principle, with a mobile interface scope. The testing stage is where the user interface is evaluated, this would most likely involve triangulation to get most information about the usability of the interface, field studies are also an option in this stage (Ayob et al., 2009).

<table>
<thead>
<tr>
<th>STAGE NO</th>
<th>STAGE NAME</th>
<th>STAGE DETAILS</th>
</tr>
</thead>
</table>
| 1        | Analysis   | **Context of Use (Specific User and Organisational Requirement)**  
1. Identify and document user’s tasks  
2. Identify and document organizational environment  
3. Define the use of the system |
| 2        | Design     | **Context of Medium (Produce Design Solution)**  
1. Enable frequent users to use shortcuts  
2. Offer informative feedback  
3. Consistency  
4. Reversal of actions |
5. Error prevention and simple error handling
6. Reduce short-term memory load
7. Design for multiple and dynamic contexts
8. Design for small devices
9. Design for speed and recovery
10. Design for “top-down” interaction
11. Allow for personalization
12. Don't repeat the navigation on every page
13. Clearly distinguish selected items

3 | Testing | Context of Evaluation (Evaluate design against User Requirements)
   |         | 1. Quick approach
   |         | 2. Usability testing
   |         | 3. Field studies
   |         | 4. Predictive evaluation

**Figure 4. Three Layer Design Guideline for Mobile Application**

*(Ayob et al., 2009, p.429).*

Rauch (2011) also provides useful techniques for designing mobile interfaces but the most important of it is the consideration of unique requirements and issues of mobile interface. Mobile devices are not as old as desktop or personal computers and are therefore, not as developed. A proper study of how mobile devices are used, the best design for each platform and the best interface for each size of screen has to be considered before venturing into designing an application. Other important things to consider are, the development of mobile prototypes to test on actual devices after they have been tested on emulators. A consideration of platform interface differences, a plan for implementation of mobile translation, Implement mobile accessibility and providing a mobile web site as an alternative for the actual application are also important thing to be considered.

Part of creating good usability and good user experience is providing appropriate documentation for users, Rauch (2011) defined an eight step process for user interface documentation. First of all, the documentation starts as an initiative that must be managed and treated like a project. Then a proof of concept needs to be developed as a plan on how the documentation process will proceed. A project leader should be assigned as the head of the project, and a proper training on what needs to be done
and how it needs to be done has to be provided to the team members. Next, a three step process of converting files, testing files and delivering files to the decided format is required, which reduces the risk of errors in the future. The final stage is gathering feedback from users to measure customer satisfaction for the product which is the provided documentation.

Although not all of these principles need to be applied for the Ibaana messenger a selective combination of the principles will likely result in good usability and high level of user experience.

2.5. Image Processing and Distributed Application Architecture

The main objective of the Ibaana Messenger is to enable users express themselves with images which would require real time image editing features. To be able to develop a mobile application that has this feature, a study of image processing and editing concepts, and its technologies are crucial to having a high quality final product.

Any application that processes images is expected to perform the most common image editing functions like grey scale processing, cropping, colour adjustment and even the addition of text to images. The Ibaana messenger should be a distinctive application by providing a better way of handling image because images should be the main form of communication between users.

Indexing and searching of applications will be one of the most important functions of the Ibaana messenger where images will need to be retrieved frequently. Xia et al (2010) explains that manual annotation of image information whereby stored images are retrieved by searching the filenames can be a performance intensive task, this task will most likely cause unnecessarily high server load in an industry environment. A better approach would be to store annotations or image information in a different database that can be queried when needed and used to retrieve the image from a file server as illustrated in Figure 5.
Better searching and indexing algorithms can be derived when information is separated as illustrated above. The information of images can be used to search and sort images without causing the high traffic of searching each image on the file server. Context-Aware indexing of image annotations can also be a more informative way of storing image annotation. Xia et al. (2010) suggests that image context such as the date of creation, GPS location, personal information of author and social information of images can serve as a means of providing an addition of detailed information that will be used in retrieving stored images. These information can be stored dynamically by gathering the various information automatically from the user’s device or information provided manually by user’s input. Traffic caused by image retrieval can be reduced by first saving thumbnail of each image stored in the file server. The Image thumbnails can be viewed before the actual high quality image is requested by the user and sent over the network. Traffic can also be reduced by caching images on the local systems to stop constant re-uploading of images from the file server on every user visit (Xia et al., 2010).

In an effort to reduce the workload and required skill for editing images, Brooks and Dodgson (2010) proposed a concept of replicated image editing. Replicated image editing involves the change of neighbouring pixels brought about from the editing of a single pixel. Usually the user will have to specify the portions that need to be changed manually like “marqueeing” in Photoshop which is a skill demanding procedure. Replicated Image editing is divided into three operations namely, replicated painting which involves altering colours of specific pixels, replicated cloning where one image is
clone onto the other, and replicated warping where certain regions of the image can be expanded or contracted.

Image processing techniques will only be best utilized if the right blends of technologies are used to implement them. Due to the complexity of creating an image processing application, implementation of an open source image editor could be combined with network technologies for the best results.

GIMP stands for GNU Image Manipulation Program, it is an open-source pixel based image processing application under the GNU license and is highly customizable and extensible. GIMP is widely seen as a substitute to Adobe Photoshop according to Ryan (2008) due to it being freely available. The best aspect of GIMP is the amount of image manipulation features which makes it one of the most widely used image editors programs (Oliver et al., 2006).

![The Five GIMP Layers](image)

*Figure 6. The Five GIMP Layers*

( Oliver et al., 2006, p.5)

The GIMP architecture is divided into five layers name the User Interface Layer (UI), Extensions Layer, GIMP Core Layer, Plug-ins Layer and the GIMP Libraries Layer as shown in Figure 6. The UI layer is the layer of the application where users can perform image manipulations by triggering events from the UI layer’s components. The GIMP core layer is the part of the application that performs the system level tasks such as configuring the GIMP system and loading plugins. The extensions layer is the layer that performs read-only operations like saving and printing images while the plug-ins layer is
responsible for altering the image by providing functions like blurring and transformation. The extensions and plug-ins layer can be extended with an array of programming languages like Java, Perl and Python. The GIMP libraries layer is the part of the application that provides the services other parts of the application use like the mathematical functions and colour manipulations functions (Oliver et al., 2006). This is a great option for the Ibaana messenger architecture as it can be used as the remote image processing server.

The processing of the user’s images by the Ibaana Instant Messenger will be done on a remote server which means there will be an active connection between the user and the server during the image manipulation. Liang, Xiao and Huang (2009) proposed a three tier service oriented architecture for image processing consisting of the user interface layer, services management layer and services implementation layer as illustrated in Figure 7. The user interface layer is how the user is going to communicate with the application which could be a native device application or a web interface. The service management layer consists of the service publication, service discovery, service invocation, service registry components and image processing services’ publication. The function of the service management layer is to serve as an intermediary between the image processing service and user interface, various components like security integration can also be added in the layer. The service implementation layer is where the image processing takes place and the image is returned to the services management layer on completion (Liang et al., 2009).

![Suggested Service Oriented Architecture of An Image Processing web service](image)

*(Liang et al., 2009, p.3)*
The server side applications that will be accessed by multiple devices on different networks need to be distributed for seamless provision of services to users and to balance the high amount of user request. An adequate analysis on the best methods and technologies is required, for not only creating a scalable cluster of application servers but also optimizing the transfer and streaming of images between the client devices and servers.

First of all, there has to be a consideration on how the user will edit the image from the client. When the user tries to edit the image which is on the server, a request is sent to the server to process the image. After the image has been processed by the server, the client needs to see the changes that have been performed on the image. A reasonable approach to fulfil this requirement would be Image Streaming. Deshpande and Zeng (2001) proposed the image streaming method whereby images can be sent in an upscale resolution manner. For example, an image can be sent progressively in such a way that the first image displayed would have 50 per cent of its original resolution until the complete image is sent to the client. This technique greatly reduces the server load and could give feedback to the user on how the image has been modified. If delivery of the images ever needs to be tested while streaming, authentication can be done both at stream level and content level (Zhang et al., 2007).

One of the biggest challenges of industry standard applications is coping with the high volume of user requests, in fault tolerant and scalable ways. This brings in the deployment of distributed server architecture where multiple servers are deployed to work hand in hand for processing requests, taking over another server’s duty in case of failure, and providing easy of scaling up the number of server when needed (IBM, n.d.). Apache Zookeeper is an application that enable cluster of servers to share the processing of user requests and keep all application servers synchronized at all times. The Zookeeper application works on different server by first assigning a specific server from a list of server as the leader. The leader makes sure the data on all servers are synchronized with each other but does not process user requests itself. This allows application to be continuously backed up on multiple server nodes at all times, it also provides efficient routing and shared processing of user requests (Schlichting and Gavin., 2012).
2.6. Evaluation

Software must be evaluated to test every aspect from its functionality to its usability before it is introduced to the end users. After the data from the evaluation is gotten, it must be compared and analysed to provide enough details for future decisions to be made on the improvement of the software. To achieve these objectives, relevant evaluation methods and analysis techniques are explored.

Quality software undergoes unit and integrated tests to validate the functionality of the software as a whole. Regardless of this, the Ibaana messenger is a distributed application that will likely consist of different components in various locations. This circumstance means the software needs to be evaluated to test its interactivity with each component in terms of their interaction with each other on how well they work to serve the users.

Kovács et al (2008) proposes an evaluation framework for distributed mobile applications where by the software is evaluated on three levels namely the user level, the application level and the communication architecture level.

The user level of the framework evaluates how the user interacts with the application, in terms of finding the components of the application requested for by the user to better understand what the user wants most from the application. This can be done by studying user activities and turning them to
phases to understand the user’s workflow, this can be used to not only understand what the users do but why they do it. The user level framework is illustrated with the UML activity diagram (Kovács et al., 2008).

The application level evaluation aims to evaluate how each component of the application works with the other and study the dependency and performance of the components when they communicate with each other. The dependency evaluation can be done by using mobile trace generators like VanetMobiSim to measure the communication between the components and find out the inter-process performance (Kovács et al., 2008).

The communication architecture level evaluation involves the evaluation of the supporting hardware and software on which the application is deployed and executed on. The performance of the application is measured on these services (hardware and software resource) to determine how it can be improved (Kovács et al., 2008).

![Distributed Evaluation Framework Workflow](image)

Figure 9. Distributed Evaluation Framework Workflow
(Kovács, 2008, p.32)

The User Interface is what users use to interact with the application therefore the user interface evaluation is a crucial step in developing an application. User Interface evaluation can be done by heuristic evaluation which is the process whereby usability experts evaluate a software interface using a set of usability principles. An example of a heuristic is the Nielsen heuristics which consists of ten principles, examples of these heuristics are recognition rather than recall, match between system and
real world, and documentation. The advantage of this type of evaluation is that it possible to get quality and reliable feedback since experts are the subject of the evaluation. The disadvantages of the method are that it requires knowledge of HCI principles and experts could be hard to find as a result (U.S. Department of Health & Human Services, n.d.).

Usability evaluation involves users using software to provide developers with information on how well the application can be used. The main aim of usability evaluation is to get feedback about the usability of an application directly from the people that might be using it in the future. Nayebi, Desharnais and Abran (2012) considers usability to be divided into three aspects namely efficiency, learnability and user satisfaction. They also propose that mobile usability can be evaluated with laboratory experiments, field studies and hands-on measurements.

Laboratory experiments take place in a controlled environment where users are given tasks to perform and are observed. The advantage of this method is that only the users that are performing the experiment are involved in the evaluation and all the functions of the application can be tested. It is also easy to gather data to be analysed and used for improvements. The disadvantage of the laboratory experiments is that it isolates the users from the real world and this could alter their interaction with the application which will not properly represent how the application will be used. (Nayebi et al., 2012).

Field studies involve users performing tasks where they can be observed and interviewed about their experiences in a normal environment. This method provide more real life interaction for the user as it is performed in the real world but certain entities in the environment could also alter the evaluation process (Nayebi et al., 2012).

The hand-on measurement involves users measuring the effectiveness of certain aspects of the application for example, the display load, clarity of operation possibilities and completeness of function. The is good in getting information on what features users like the most but measurement can be inaccurate as function of application could be had to measure (Nayebi et al., 2012).

These evaluations are useful for both formative and summative evaluations of the Ibaana Instant Messenger to properly measure its usability, user experiences. This would identify how well it achieves its main objective of improving interactions between users. The user data from the evaluation can be gathered through interviews, questionnaires and observation. The results can then be analysed and represented using charts and diagrams to provide comparative analysis on its performance. Even if some evaluations such as the distributed evaluation framework might not be easy to perform, similar evaluations can be conducted with other applications.
2.7. Professional, Legal, Ethical and Social Issues

In creating the Ibaana messenger, it is important to investigate and review the professional, legal and ethical issues that may be encountered and decide on how to guard against them. Only issues that are within the scope of the Ibaana messenger as an academic project will be identified.

2.7.1. Professional Issues

The Ibaana messenger project is an academic project under the university and should be treat as one. All the codes, conducts and practices of a master’s project according to Heriot Watt University regulations must be adhered to. Adhering to the rules and regulations of the master projects means that the ibaana messenger project adheres to the codes and conducts of any organisational body that the master degree programme is accredited to (Farrow, 2006).

One of the Professional issues that could occur related to this project is not referencing sources of information that are used in researching or developing the project. This issue could lead to plagiarism, causing heavy penalties on the student involved. Another professional issue that can occur is use of a current student’s academic work in developing the project, this will lead to plagiarism that will also causing heavy penalties to be imposed on the parties involved.

2.7.2. Legal Issues

Due to the fact that this master’s project will be used in United Arab Emirates (U.A.E) and the main campus the university is in the United Kingdom (U.K), it must conform to the laws of the two countries. Legal issues that are within the scope of this project are laws that deal with the users’ privacy law of the U.K and the prohibited content law of the U.A.E.

The U.K Data Protection Act of 1998 is a law that prevents user information from being used for other reason the user might not wanted the information to be used for, and prevents user information to be stored for longer than needed. The law also prevents user information that is not needed for the context of use to be collected and any user information collected must be stored securely. To prevent any unforeseen events or situations, this law has to be adhered to during evaluation processes (U.K Parliament, n.d.).
The Prohibited content categories of the U.A.E’s internet access management policy is a list of content and websites that are prohibited to be accessed or used in the U.A.E. Examples of the prohibited content are dating sites, websites with content for learning criminal skills like fraud, and gambling sites. Exceptions to this law are chatting services and social networks (Etisalat, 2013). Although this law clearly authorizes applications like the Ibaana messenger, consideration of the prohibited content must always be remembered when developing new software features.

2.7.3. Ethical Issues

According to Farrow (2006) the stakeholder analysis helps to identify what is ethical to any party involved in the project. Another way of finding out and resolving probable ethical issues is performing the six useful tests which are the golden rule, legal test, smell test, parent test, media test and the market test while working with end users (Farrow, 2006).

During evaluation, users can be informed of how their information will be used and discarded to prove and assure them of the ethics of how their information will be treated. Users can also be asked about how they expect their information to be managed.

2.7.4. Social Issues

The Ibaana messenger project is a social network application which means social issues have to be considered before they become a problem. Privacy of a user’s information which includes photos, personal data, contacts etc., must only be available to the parties specified by the users. Communication information like chats and shares should also only be available to the individuals involved in the communication. A social issue that can arise from the use of the Ibaana Messenger is users sharing what could be considered as inappropriate content with other users. This issue can be investigated and actions like account deactivation can be used as penalties. Users must also be verified to make sure that they are who they claim they are to prevent malicious behaviours. Security measures like the encryption of data and implementation of access controls could also be applied to prevent unauthorised access into the user’s accounts and social networks information.
3. Chapter 3 [Software Development Process]

This chapter explains the entire software development process for the Ibaana Instant Messenger. It also includes the decisions, techniques, technologies used as well as its development. The chapter starts by explaining how and why certain software engineering methodologies were used to create the project plan for the development processes. Then it goes further by discussing the requirements engineering process and its results. Each stage of the software development process is enumerated in iterations to explain in details how the development process went and why certain decisions were made.

3.1. Methodology and Project Plan

This project uses multiple aspects of different software engineering methodologies such as the scrum and extreme programming to manage the project stages and ensure maximum productivity. First of all, the basic software engineering processes as suggested by Sommerville (2011, p.29) which consist of requirement, design, implementation, testing, operation and maintenance are the foundation of the development process. Other fundamental principles used are, the agile principle of working prototypes being the measure of success and creating software in iterative increments.

One of the principles used is the sprint from the scrum methodology where the design, implementation and testing processes make up a sprint. Normally, the events of a sprint are the sprint planning meeting, daily scrum, sprint review and sprint retrospective (Schwaber and Sutherland, 2011). But this project only applies the daily scrum meetings and sprints retrospective into the sprints as it is an individual project, the project log can be found in the appendices section. After every sprint there was an evaluation and documentation stage which lasted for a week. Wherever there was an evaluation, both the sprint events and evaluation were documented. These sprint stage and evaluation and documentation stage major are what make up the iteration for this project as shown by the original overall project plan in the Figure 10.

Some other practices used in this project are from the extreme programming software methodology. Although extreme programming is meant for team based software development, the simple design, courage, design improvement practices have been adopted for this project. The Simple design practice encourages software developers to write codes in the simplest way possible, this means that the code is written for an exact requirement which prevents a waste of time and resources (Lindstrom and Jeffries, 2003). Design improvement which is also known as refactoring, is a practice of removing poorly designed code to increase cohesion and decoupling. Instilling courage into every party
involved in the project to strive for a better software quality (Lindstrom and Jeffries, 2003). These practices are specifically tailored for the implementation stage of the project to improve the quality of the software.

<table>
<thead>
<tr>
<th></th>
<th>1st Iteration</th>
<th></th>
<th>2nd Iteration</th>
<th></th>
<th>3rd Iteration</th>
<th></th>
<th>4th Iteration</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Start of Project</td>
<td>20 days</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
</tr>
<tr>
<td>Split Requirement</td>
<td>0 days</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
</tr>
<tr>
<td>into iterations</td>
<td>1 day</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
<td>Mon 06/05/13</td>
</tr>
<tr>
<td>1st Sprint</td>
<td>15 days</td>
<td>Tue 07/05/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
</tr>
<tr>
<td>Evaluation and</td>
<td>5 days</td>
<td>Tue 28/05/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
<td>Sun 02/06/13</td>
</tr>
<tr>
<td>Documentation</td>
<td>End of 1st Iteration</td>
<td>Wed 02/06/13</td>
<td>Wed 02/06/13</td>
<td>Wed 02/06/13</td>
<td>Wed 02/06/13</td>
<td>Wed 02/06/13</td>
<td>Wed 02/06/13</td>
<td>Wed 02/06/13</td>
</tr>
<tr>
<td>2nd Sprint</td>
<td>20 days</td>
<td>Mon 03/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
</tr>
<tr>
<td>Evaluation and</td>
<td>16 days</td>
<td>Mon 03/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
</tr>
<tr>
<td>Documentation</td>
<td>End of 2nd Iteration</td>
<td>Mon 24/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
<td>Sun 30/06/13</td>
</tr>
<tr>
<td>3rd Sprint</td>
<td>20 days</td>
<td>Mon 01/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
</tr>
<tr>
<td>Evaluation and</td>
<td>16 days</td>
<td>Mon 01/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
</tr>
<tr>
<td>Documentation</td>
<td>End of 3rd Iteration</td>
<td>Mon 22/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
<td>Sun 28/07/13</td>
</tr>
<tr>
<td>4th Sprint</td>
<td>13 days</td>
<td>Mon 29/07/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
</tr>
<tr>
<td>Evaluation and</td>
<td>8 days</td>
<td>Mon 29/07/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
</tr>
<tr>
<td>Documentation</td>
<td>End of Project</td>
<td>Thu 06/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
<td>Wed 14/08/13</td>
</tr>
</tbody>
</table>

*Figure 10. Project Plan Showing The Aspects Of Each Iteration*

The Gantt chart above illustrates the project plan for the Ibaana messenger, it consist of methods used from the scrum and iterative development. The two methodologies have been combined to give the project enough flexibility with techniques from different agile methodologies. The project was divided into four iterations, with each of the iterations having a sprint, an evaluation and a documentation stage except for the first iterations which had an additional stage for splitting the requirements for the iterations. The project starts with a one day phase to split the requirements into incremental iteration with the most basic requirements being in the early iterations. After the requirement splitting phase, the sprint containing the basic software engineering process begins which would implement the practices of the scrum methodology. After the sprint is done, there would be a formative evaluation of the software, followed by the documentation of the whole process including the new requirements gathered from the evaluation. These phases would be repeated in each iteration. All the iterations duration of a month except the final duration that would have only taken two weeks as presumably only minor changes will need to be done after the three iterations. The deliverables set for each of the iteration are a working prototype and a section of the overall documentation.

Each evaluation should be conducted in two days and will involve normal users and expert evaluators. Information would be gathered from them with interviews, questionnaires and observation data.
3.2. Requirement Engineering Process

In order to properly generate requirements for this project, the stakeholders were first identified. Then the requirements were generated to fulfil the goal of the project with considerations to all the stakeholders of the project. The next step was to convert the generated requirements to a backlog, so that each requirement would be listed with the most important requirements listed first. The creation of the backlog made it easy for sprint goals to be set in accordance with the requirements.

3.2.1. Stakeholders

Stakeholders are individuals that are directly or indirectly affected by a project Sommerville (2011, p.94). In identifying the stakeholders, a power-interest matrix can be applied where stakeholders are identified by the power they have over a project and also the amount of interest they have in the project (Thompson, n.d.). The figure below shows the power-interest matrix.

![Power-Interest Matrix](image)

*Figure 11. Power-Interest Matrix for Stakeholders (Thompson, n.d.)*

The Ibaana Messenger’s present stakeholders can be identified as Heriot-Watt University, the present and future software developers, mobile operating system vendors, development tools vendors, software hosting vendors, the public and end users. The power-interest matrix for the stakeholders is illustrated by the figure below.
The power matrix proposes that stakeholders with high power and high interest are people that must be fully engaged and satisfied at all times. The stakeholders with high power and low interest must be kept satisfied but should not be bothered with excessive communication. Low power and high interest stakeholders must be adequately informed to ensure that there are no major issues with the project as they also contribute to the final product of the project. Stakeholders with low power and low interest only need to be monitored (Thompson, n.d.).

3.2.2. Requirements

Requirement discovery involves getting information from stakeholders on how the final product should be, which is done through interviews, questionnaires and observation (Sommerville, 2011, p.114). Due to unavailability of the stakeholders the best option would be to perform the requirement discovery during the evaluation processed as an incremental process.

According to Sommerville (2011, Ch. 4) there are three types of requirements specification namely User, Functional and Non-Functional requirements. The user requirements are what the software should provide to user, the functional requirements are what the software should perform as a whole and non-functional requirement are the constraints of the software.

I. User Requirements

1. Users must be able to perform instant messaging functions with other users.
2. Users must be able to share and edit pictures on popular social networks like Facebook and Twitter.

3. Users must be provided with enough tools to be able to create images from scratch.

4. Users must be able to edit images and pictures without the need of technical skills.

5. Users must be able to express themselves adequately creating images like memes and comic strips.

6. Users must be able to send messages via SMS to other users connected through the internet.

7. Users must be able to edit images from a remote server with ease and efficiency.

8. Users must be able to share images with contacts in a social network dedicate for the software.

II. Functional Requirements

1. The system should be able to provide transfer of messages, images and files from one client to another via the internet and SMS.

2. The system should be able store the communication history of users.

3. The system must allow users post their edited images on social networks like Facebook and Twitter.

4. The system must provide users with advance image processing features remotely allowing them manipulate images on the server from mobile devices.

5. The system must provide enough creative tools with a user friendly interface for complex image manipulation.

6. The system must provide batch of image processing tasks to minimize the need for multiple operations to be performed by the user when editing images.

7. The system must provide templates and functions for popular types of images e.g memes and comic strips.

8. The system must provide SMS to internet communication between users.

9. The system must provide fast image processing for the user.
10. The system must ensure seamless communication between all components between the server components and between the clients and the server.

11. The system must provide users with a dedicated social network for sharing images between contacts.

12. The system must be developed in such a way that it can be scalable and does not cause unnecessary system overhead.

III. Non-Functional Requirements

a. Product Requirements:

1. User messages and files should be stored on the server in a compressed manner to minimize disk space.

2. The recommended hardware, network and software requirements needed for all components to run smoothly, like the recommended requirement for the image processing component must be met.

3. The minimum downtime of the application should be one hour in a day.

b. Organization Requirements

1. All users must register an account and have a unique customer number which will be used as the main form of identification and authentication.

2. Due to the cost of implementing a remote server, an outsource cloud infrastructure will be used for maintainability and scalability.

3. Open-source free software development technology will be used in developing the application for cost-effectiveness and flexibility.

4. User information will not be accessed by any party except with permission from the user or investigation based on regulation violation reports from other users.

c. External Requirements

1. The software must conform to all national and international software regulations.
2. An end-user license agreement explaining the terms of use of the software must be agreed between the end user and the software provider.

3. A privacy statement assuring users about the handling their personal data must be presented before use of the software.

4. A disclaimer about the content of other users or third parties in the social network must be presented to the end user before the use of the software.

3.2.3. Backlog

The Backlog is the overall requirement of the project which is divided into lists and sub-lists of prioritized objectives. This makes setting and identifying sprint goals and objective easier, it also makes sure that the sprint goals are always tied to the requirements. The backlog it prioritized in a way that each sub-list means that a certain percentage of the list is achieved, this makes measuring the progress easy. Although the list is prioritized, the features to be implemented are selected in another of the most basic functionalities and added to the sprint goal to make sure development is done in increments. Below is the backlog of the requirements, the mandatory requirements have been marked with the “**” sign.

1. System must be able to provide network where users with the ability to share images with contacts in a social network dedicate for the software.
   
   i. Users must be able to perform basic instant messaging functions with other users. **
   
   ii. The system should be able store the communication history of users. **
   
   iii. All users must register an account and have a unique customer number which will be used as the main form of identification and authentication.
   
   iv. User information will not be accessed by any party except with permission from the user or investigation based on regulation violation reports from other users.

2. Users must be able to express themselves adequately creating images like memes and comic strips.
   
   i. Users must be able to edit images and pictures without the need of technical skills. **
   
   ii. Users must be provided with enough tools to be able to create images from scratch.
iii. Users must be able to edit images from a remote server with ease and efficiency. **
iv. The system must provide fast image processing for the user. **
v. The system must provide enough creative tools with a user friendly interface for complex image manipulation.
vi. The system must provide users with advance image processing features remotely allowing them manipulate images on the server from mobile devices. **
vii. The system must provide batch of image processing tasks to minimize the need for multiple operations to be performed by the user when editing images.
viii. The system must provide templates and functions for popular types of images e.g memes and comic strips. **

3. Users must be able to share and edit pictures on popular social networks like Facebook and Twitter.
   i. The system must allow users post their edited images on social networks like Facebook and Twitter. **
   ii. The system should be able to provide transfer of messages, images and files from one client to another via the internet and SMS.
   iii. Users must be able to send messages via SMS to other users connected, through the internet. **
   iv. The system must provide SMS to internet communication between users. **

4. The system must ensure seamless communication between all components between the server components and between the clients and the server.
   i. The system must be developed in such a way that it can be scalable and does not cause unnecessary system overhead. **
   ii. User messages and files should be stored on the server in a compressed manner to minimize disk space.
   iii. The recommended hardware, network and software requirements needed for all components to run smoothly. For example, the recommended requirement for the image processing component must be met. **
   iv. The minimum downtime of the application should be one hour in a day. **

5. The software must conform to all national and international software regulations.
i. An end-user license agreement explaining the terms of use of the software must be agreed between the end user and the software provider.

ii. A privacy statement assuring users about the handling their personal data must be presented before use of the software.

iii. A disclaimer about the content of other users or third parties in the social network must be presented to the end user before the use of the software.
3.3. Proposed Overall System Model

Figure 13. Proposed Overall System Architecture Before The Start of The Project
3.4. 1st Iteration

The project plan was divided into different iterations with the foremost iterations to implement the most basic functionality. The most important aim of the first iteration was to provide the basic functions on which to build the other functionalities, each stage is explained in the coming subtopics. The Original planned iterations were for one month. Below are the selected goals for the 1st iteration.

3.4.1. Sprint Goals

Main Goals
1. Users must be able to perform basic instant messaging functions with other users.
2. The system must be developed in such a way that it can be scalable and does not cause unnecessary system overhead.
3. The recommended hardware, network and software requirements needed for all components to run smoothly, for example the recommended requirement for the image processing component must be met.

Other Goals
1. Users must be able to edit images from a remote server with ease and efficiency.
2. The system must provide fast image processing for the user.
3. The system must provide users with advance image processing features remotely allowing them manipulate images on the server from mobile devices.

3.4.2. Design

The overall basic goal for the first sprint was to provide the basic functionality where users would be able to login to the chat service, send messages, perform simple image effects and send images to other users. An assumed architecture of the application to do this was created before the iterations (Figure 14).
Figure 14 illustrates how the mobile client application would communicate directly with all components at the first sprint to make sure that users can interact with each other by sending messages and images.

3.4.3. Implementation

The first step toward implementing the basic feature of the ibaana messenger was to make sure users were able to chat with each other that meant implementing an instant messenger server. Before choosing the server application to use, the software platforms which the instant messenger server application was to be deployed on were researched. The sales force platform was seen as the best options as the Google’s app engine does not allow the use of sockets by applications and another option, which is the Microsoft’s Azure platform is more expensive. When choosing and instant messenger server, the Ignite Realtime’s Openfire and the Apache’s Vysper xmpp servers where compared. Although the open fire applications server had more features, the Apache Vysper application server was chosen. This was because the Openfire server had so many un-needed features like the web setup interfaces and Yahoo, Skype and Facebook login capabilities that could unnecessarily affect the speed, security and performance of the IM server at runtime. Even though the Vypser application server was a simple stripped functioning instant messenger applications server it had all the needed features of an Instant Messaging (IM) application server.
Although it took some time because of the inadequate documentation on dependencies, the instant messenger server was up and running on the local machine.

The GIT version control system was used as measure of versioning and backup and the “SugarSync” application that provides auto synchronized backup was also used for all parts of the application. A log on the daily scrum meetings and sprint retrospective was also written in accordance with scrum methodology and is available in the appendices section. The Blackberry Bold 9780 mobile version was chosen as the model for simulating the running of the code as the physical device was also readily available for real life testing.

Firstly, to be able to connect to the IM server, the BOSH (xep-0124 standard) module that was available with the Vysper application was added to the IM server. The BOSH component which stands for Bidirectional-Stream Over-Synchronous HTTP is a component that makes it possible for users to connect to an IM server with the HTTP protocol. Secondly, the Strophe.js JavaScript library was also added as part of the client application to be able to send XMPP stanzas to the IM server. The Strophe.js JavaScript library is used to connect to IM server with the HTTP protocol. It has features that allow the client applications to be continuously connected to the server to mimic the behaviour of normal chat clients. After these operations were done, creating the code to log in users to the IM Server was successful. It had to be implemented as a normal web application and tested on the browser then migrated to the simulator. At this point the migrated application on the simulator was able to send messages and log in users to the server. This was tested by login in to the browser and simulator and sending PING to the server and messages from the browser to simulator and vice versa with the IM server being the intermediary.

```javascript
var conn=new Strophe.Connection("http://192.168.3.199:8080/bosh/");
$("#status").bind('click',function(){
    //On click, login the user to the IM server
    conn.connect("user1@vysper.org","password1",function(status){
        if(status===Strophe.Status.CONNECTED)
        {
            $("#status").html("Connected");
            //navigate to chat page if connection is successful
            newPage("screens/chat.html");
        }
        else if(status===Strophe.Status.DISCOUNTED)
        {
            $("#status").html("Disconnected");
        }
        else if(status===Strophe.Status.AUTHFAIL)
        {
            $("#status").html("Authentication Failed");
        }
        else if(status===Strophe.Status.CONNFAIL)
        {
            $("#status").html("Connection Failed");
        }
        else if(status===Strophe.Status.AUTHENTICATING)
        {
            $("#status").html("Authenticating");
        }
        else if(status===Strophe.Status.CONNECTING)
        {
            $("#status").html("Connecting");
        }
    });
});
```

Figure 15. Mobile Client Code for connection to IM Server
To make the mobile client more user friendly, the Jquery mobile framework was used to add a customized user friendlier interface to the mobile application. The Jquery mobile framework encouraged the use of all the screens in one document which is a wrong software development practice, therefore the decision was made to separate the login and chat screen. This quickly caused a problem because after loading the chat screen the login session was lost. New code was created to save the username, RID and SID temporarily to the mobile device using the Phonegap API and retrieved on transition to the new screen. The simulator could not handle this kind of task for unknown reasons and the ripple emulator plug-in was recommended by the developer’s community. This was a very good switch as all the code could be run on the browser instead of the time consuming testing and migrating of the code to the simulator. At this point a good user interface was created, and the client and IM server were functioning well. The only thing that remained for the sprint was the Image processing server.

```javascript
function storeSessionInfo(){
    // Method to create a database and perform an operation
    db = window.openDatabase("Session","1.0","SessionInfo",500000);
    db.transaction(populateDB,errorCB,successCB);
}

function populateDB(tx){
    // If the database exists drop it and add new session data
    tx.executeSql('DROP TABLE IF EXISTS SESSION');
    tx.executeSql('CREATE TABLE IF NOT EXISTS SESSION(username unique, rid,sid)');
    tx.executeSql('INSERT INTO SESSION(username, rid, sid) VALUES (?,?,?)',[username,rid,sid]);
}

function errorCB(tx,err){
    // If any error occurs display an alert notification
    alert("Error processing SQL: "+err);
}

function successCB(){
    // If operation is successful display a success alert notification
    alert("success!");
}
```

Figure 16. Storing Session Info Into Mobile Device database using Phonegap API

The GIMP as mentioned in Section 2 was seen as the ideal applications for processing images, and so the task of making it part of an image processing server for the Ibaana messenger project began. The setup of the GIMP started with instructions provided in the GIMP community for developing GIMP...
on windows as GIMP is meant for Linux OS. At first setting up GIMP, went smoothly but quickly became a tiring and frustrating exercise as a lot of instructions and required files were not included in the instructions, just setting it up required compiling C and C++ code using GNU MAKE and MSYS. It took far more time than anticipated as compiling the code took hours and errors where encountered frequently which required visits to various forums. After the setup was completed, GIMP was unstable and could not run properly enough for development. It was clear that GIMP could not fulfil the needs of this project and was best suited for Linux users and GUI interactions rather than server side processing.

After visiting forums and developer community about resolving the problems of GIMP and trying to use the GIMP-console component to process images, the IMAGE MAGICK software was suggested. This proved to be the right solution for creating an image processing server. IMAGE MAGICK is an application specifically geared towards the server side processing of images. Although it was created using C and C++, there are variety of libraries that provide interface for languages like PHP, Perl and Java. It needs to be installed into any platform it will run on which is a double edge sword because that makes it harder to be deployed remotely. On the other side, it also means it is called and terminated only when needed by the standalone running program after image processing is done which means it is likely to take fewer system resources therefore having a good performance.

Starting the creation of the image processing server meant verifying that the software was indeed able to process images from a running desktop application. A standalone java application was created and the library (im4java) for communicating with IMAGE MAGICK through the command line was used to test IMAGE MAGICK’s capabilities. Not only was it a success, but it had a fast performance as stipulated from the study of its architecture. The standalone java application was called the Image processing server interface as it did not contain all the components for a complete application server.

Now that a basic working image processing server had been created, the next step was to connect it with the mobile application client. A design problem loomed, the only way to connect the mobile client with the image processing server was to use socket connection to communicate with the image processing server. This was against the intended architectural goal of the Ibaana Instant Messenger 1st iteration because even though it would be possible using Node.js, it would add high workload to the mobile client applications thereby reducing performance. The best option was to add a web application in-between the mobile application and the image processing server, so that the mobile application would be able to submit an image as a form to the web application. When the form is submitted to the web application, the web application forwards the image to the image processing server for processing. The implementation was done in stages first the client/server behaviour
between the web application and image processing server was simulated using two J2SE standalone classes, but only text was used to test the idea. After that was achieved, the web application implementation replaced the client application. This was done by submitting images via a web form to the application and forwarded to the image processor by the means of sockets. The web applications was created using J2EE servlet technology for optimal speed, and the file upload library was used for easy file upload using on the servlet platform. Furthermore, a web server was added to the overall architecture to display the processed image. The reasoning behind this was that sending back the image in full to the web application would have taken too much network and hardware resources. An improvement needed to be made, as users will need to choose an effect to be applied before sending the image to the web application. The added feature was applied by opening two different sockets for sending the image as a file and the effect to be applied to the image as text to the image processing server.

```java
ServletContext servletContext = this.getServletConfig().getServletContext();
    File repository = (File) servletContext.getAttribute("javax.servlet.context.tempdir");
DiskFileItemFactory tempFileFactory; // code for temporarily getting the image
public class ProcessImage extends HttpServlet {
    Socket echoSocket = new Socket("localhost", 5050); // Create Socket Connection
    try {
        FileItemIterator itemIterator = upload.getItemIterator(request);
        while (itemIterator.hasNext()) { // while some form data still exists.....
            FileItemStream it = itemIterator.next();
            if (it.isFormField()) { // Send other field data to Image processing(IP) server
                int temp;
                char[] value = new char[15];
                BufferedReader in = new BufferedReader(new InputStreamReader(it.openStream()))
                while ((temp = in.read(value)) != -1) {
                    System.out.println(value);
                }
            } else { // If not field data then it's an Image data send it to IP server
                System.out.println(it.getFieldName());
                while ((userInput = it.openStream().read()) != -1) {
                    echoSocket.getOutputStream().write(userInput);
                }
            }
        }
    }
```

*Figure 17. Web Application Using the File Upload library & Sockets to transfer data*
The final step of the iteration was to be able to send an image selected by the user on the mobile device to the web application and for the web application to forward the image to the image processing server. When the image processing is done, a link to the file on the web server would be sent as a reply to the mobile application and the user would be able to send that link to another user. It was possible to achieve the final step by using Ajax from Jquery to implement the publishing of the image form asynchronously from the mobile client application and receive the image as a dialog box. After the image is processed, a link to the image can be sent to another user which will trigger a popup on the receiving user’s interface. Below is the architecture and workflow of the overall application after the 1st sprint.

![Overall Architecture of the Ibaana Instant Messenger After 1st Sprint](image)

In order to conform to different aspects requirements like server side processing and having dedicated servers for each function, new components were added to the overall system. The User Request and Service Routing Server (routing server) and the web server where introduced as shown in the above figure. The routing server is responsible for forwarding images between the mobile client and image processing server. The web server stores files to be views by the mobile devices, and the mobile IM server is directly connected to the mobile device. This is because going through the routing server as planned would have caused congested network traffic especially as there are needs to be frequent connections between the mobile device and IM server for the user to remain logged in. The screen shots of the mobile client application are also available below.
Figure 19.  Login Screen

Figure 20.  Chat Screen
Figure 21. Image Processing Dialog Form

Figure 22. Confirmation On Sending Processed Image
3.4.4. Testing

The application was tested against the goals of the sprint which it achieved as users could send messages as well as process and send image to each other with an optimized performance. Although there was a bug with the Phonegap API in providing the picture selection interface for the mobile device so the normal input file element hard to be used. It will be hopefully fixed at the start of the next iteration.

The application could not be deployed and evaluated for this iteration. The main reason the application could not be deployed and evaluated was because the 1st iteration took far more time than anticipated extending to two weeks more than expected. This was mostly down to the issues with GIMP. A working prototype is the measure of progress according to the agile methodologies used in this project so the sprint could not be stopped until a prototype had been created. Another reason the application could not be evaluated properly was because the picture Phonegap API that allows users select pictures from a picture library or use the phone’s camera could not be implemented. Thankfully the iteration was extended by combining the time from the fourth iteration into the first iteration thereby leaving the other two iterations intact. The prototype was also shown to the supervisor for feedback and approval. The modified project plan after the 1st Iteration is illustrated by the Gantt chart below.
3.5. 2nd Iteration

3.5.1. Sprint Goals

The goals of the first sprint were orientated towards making sure the basic functionalities of the Ibaana instant messenger could be implemented and built upon. This meant other advanced features that would contribute to make the application a full instant messenger where not available. The goals created in the sprint focused on not only making the Ibaana instant messenger a rich instant messenger but also to move it away from the local system to a mobile application. This would require making all the components of the applications work together and to make it robust and stable enough for it to work both on the local system and on the internet.

To achieve these objectives, specific sprint goals were set to further the all-round development of the project. The selected goals mentioned below aim to improve Ibaana instant messenger by:

1. Introduce all components of the architecture to the overall system.
2. Provide three more complex Image manipulation options.
3. Provide text annotation on Images.
4. Allow users share their expressions on Facebook and Twitter.
5. Provide better instant messaging network features i.e. contact list and chat history.
The sprint goals were set by selecting requirements from the sprint backlog with high priorities at the current stage of the development. An appropriate test to make sure the application could be hosting online was also seen as an added sprint goal.

### 3.5.2. Design

The design aim of the 2nd iteration is to create an overall architecture to achieve the proposed overall system architecture illustrated in the Figure 18 to provide a robust, scalable and an easily deployable system.

### 3.5.3. Implementation

In accordance with the methodology of always refractoring and improving application code, the 2nd sprint started with the refractoring of all application components. The most important refactor was done on the routing server, this was because the image processing was heavily dependent on the structure of the user interface. For example, the image effects like grayscale and emboss effects were structured in the form before the image field itself and this was the order the Image processing server processed information. It was dangerous because any change to the form on user interface would result in a change in all the other components. To resolve this dependency, the code refactoring was done in such a way that only the structure of the routing server and the image processing server were the same. This effectively decoupled the user interface from the other components which in turn maximizes scalability and flexibility of the mobile client application. The steps for processing the image between the routing server and the image processing server could not be decoupled because sockets were used as the mode of communication between the two of components. The figures below show the section of code before and after the refactoror.

```java
if(it.isFormField()){
    OutputStream outputStream=echoSocket.getOutputStream();
    while((temp=in.read(value))!=-1){
        outputStream.write(String.valueOf(value).getBytes());
        echoSocket.close();
    }
} else{
    echoSocket= new Socket("localhost", 5050);
    while ((userInput = it.openStream().read()) != -1) {
        echoSocket.getOutputStream().write(userInput);
    }
}
```

*Figure 25. Routing Server Forwarding Data After Receiving:*{ Figure 17 Full Code}
try {
    LinkedList<Integer> fileData=new LinkedList<Integer>(); //To Hold Image Data Bits Format
    char [] imgEffect = new char[50]; //To Image Effect(String)
    //Get Items From Submitted Form
    FileItemIterator formItems = upload.getItemIterator(request);
    while(formItems.hasNext()) {
        FileItemStream currentItem=formItems.next();
        FileItemStream currentImage=formItems.next();
    }
    //If It is a Form Field then it’s the Image Effect, Assign to the Image effect variable
    if(currentItem.isFormField()) {
        BufferedReader in = new BufferedReader(
                new InputStreamReader(currentItem.openStream()));
        while(in.read(imgEffect)!=-1) {
            System.out.println("Effect: "+String.valueOf(imgEffect)); //Printout the type of effect
        }
    } //If it is not a form field then it’s the uploaded file, get all the data
    else {
        int imageIntData;
        System.out.println("ImgField: "+currentItem.getFieldName()); //Print the fieldname
        while ((imageIntData = currentItem.openStream().read()) != -1) {
            fileData.addLast(imageIntData);
        }
    }
    //Open Socket and Image Effect To Image Processing Server
    Socket ipSocket = new Socket(IPSERVER, 5050);
    OutputStream outputstream= ipSocket.getOutputStream();
    outputstream.write(String.valueOf(imgEffect).getBytes());
    ipSocket.close();
    //Open Socket and Send Image to Image Processing Server
    ipSocket= new Socket(IPSERVER, 5050);
    while(!fileData.isEmpty()) {
        ipSocket.getOutputStream().write(fileData.pollFirst());
    }
    ipSocket.shutdownOutput();
    //Recieve Url of Image of Processed Image from Image Processing Server Print it out
    BufferedReader in = new BufferedReader(new InputStreamReader
            (ipSocket.getInputStream()));
    String temp;
    while((temp=in.readLine())!=null) {
        out.println(temp.trim()); //Print Image Url
    }
} catch (FileUploadException ex) {
    Logger.getLogger(ProcessImage.class.getName()).log(Level.SEVERE, null, ex);
} finally {
    out.close();
}

Figure 26. IP Server Sending & Receiving Data Structure In A Predefined Form
The first iteration involved creating a prototype of the Ibaana IM that could only be simulated on a local system. As illustrated in Figure 18 to process an image, a request containing the image and image effect to be applied is sent to the Routing and Request Server which is forwarded to the image processing server. Since all components work on the same physical machine, the image processing server stored the image in the local directory of the web server then applied an effect to the image. The local directory where the image was stored could be accessed by the web server using a specific URL. That URL is sent to the routing server which is forwarded to the mobile client which uses the URL to access the image. This workflow is flawed because for the application to be deployed remotely, the web server that hosts the images and image processing server had to be on the same machine. This approach defeats the aim of having servers that are dedicated to specific functions and scalability. A way to solve this problem would be to have a server that acts both as a file server and a web server where images can be transferred using file transfer protocols and can be stored and displayed using specific URL links. Then the file server can be used even if all the components are the same or different physical machines.

A database server had to be introduced to the overall architecture as all information being utilized were pieces of static text. The database server was also integral to achieving the chat history feature. The database needed to be added in a way that other servers would be able to connect over networks to get required information.

After conducting a number of researches, the XAMPP server was selected to be able to provide the same functionality as online ftp services of both FTP transfer and HTTP access to files. Upon implementing the XAMPP server, the apache ftp library was used by the Image processing application to transfer the image via ftp to the File server. For the database server, the MongoDB database was chosen for its flexibility, performance and machine scalability (horizontal scaling). The image processing server was also improved to make sure the files transferred from the routing server were stored and transferred with unique names rather than on image being replaced over and over again.

After introducing the components, the next thing was to make sure that each component had its own specific function for the whole system to be a fully operational. The best way to this was to implement the user authentication, chat history and contact list along with its already existing features. To implement these features, information must be gotten from the database server. Testing all these features altogether would mean there is communication between all the components ensuring that all components have been introduced, and there is communication and operational efficiency.

After carefully studying the MongoDB database technology, the database was modelled to store and retrieve the login details, chat history and contact list. First the data was stored in the database and the login authentication was implemented in the IM Server. Then the contacts list and history features
were implemented in the routing server. To complete the implementation, the three features were extended to extend to the mobile client with use of AJAX’s asynchronous form submission. The overall architecture was further verified by testing all the implemented features, which was successful as all components communicated well with each other to produce the desired result. At this point, the “overall architecture” and “extended instant messaging feature” sprint goal had been achieved.

The next sprint objective that was to be implemented was the Social network sharing feature where users could share images on social networks like Twitter and Facebook. But before the implementation was finished a User Interface problem was encountered. The structure of the code for the mobile client was not scalable because all the codes were inside only one JavaScript file. The code was too long and different sections of the codes that had almost no relation to each other existed together making it look like a blob anti-pattern. The blob anti pattern is bad software design where all functions and data are stored in a single class. It was a real dilemma as object oriented features present in Java were not available in JavaScript therefore it would have been the wrong decision to try to create classes for the different sections of code. To solve this problem, each code section were moved to different JavaScript files that represented functions of every the page of the mobile client. For example, an “_index.js” file was created that contains all the functions for the login in users to the instant messenger service. In each file, functions where separated by event listeners to further separate the sections of code in a logical way. For instance, Figure 27 shows how a code segment is surrounded by the “pagebeforechange” event which executes before “home.html” page is displayed. This approach also makes the code for the mobile client more robust and scalable as developers only need to add new code to a specific section to improve or create new features.

```javascript
$(document).bind("pagebeforechange",function(event, data){
    if(data.options.target=="screens/home.html")//if the destination of the user is home.html
        for(contact in contacts){ //for each contact create a button for the contact
            var button=$("#contacts").append("<button id="contact"+contact+"" class='contact' data-role='button' for='message' id='send'">"+contacts[contact]+"</button>");
            $("#contact"+contact+"").button(); //apply jquery mobile style to the button
            $("#contact").buttonMarkup();
        }
});
```

Figure 27. Code that adds and Styles the Buttons for the Home.html file

After the code improvement was completed, it was discovered that the sections of code that adds stores the chat session of the user was not needed as the global variables stores the data across the files. This was good news as removing the section of code would also reduce the overhead need to
store and receive data on the user’s device. The new code structure also worked well with the practice of having different html pages for screens.

In implementing the social sharing features, the original plan was to be able to share pictures on Facebook and the Twitter. For the Ibaana Messenger to be able share images on either Facebook or Twitter, an application on the social platform needed to be created to server as a proxy for posting content on the social network. The implementation of sharing images on Facebook was successful but to be able to view an image in full, users need to click on the link to the online server where the image is store. This problem occurred due to the inadequate information and documentation on Facebook application on being able to upload images on the time line of users. After tedious digging of the poorly organized Facebook documentation it was discovered that the graph API would have to be used to upload images to the user’s timeline. It was decided that the best thing would be to implement both the feature to upload images to users on Facebook and Twitter in the next sprint. As other tasks like the implementation of complex image effects, testing, deploying and evaluation of the applications were still yet to be done, there was little time for the end of the sprint. The good thing to take away was that users could still share images if the FTP server is online and the information needed to improve on the feature was gotten.

With time running out, and only two days remaining for the end of the sprint, the next objective was to implement the image annotation feature and the three complex animations. The complex animations chosen for implementation were 2 Frame Animation, 2 Animation with text annotation and a morphing effect. These complex animations where chosen to make interaction between users more interesting and fun thereby improving user experience. The text annotation on the other hand was chosen to improve expression between users as explained in Chapter 2.2 that text annotation on images improves user interaction. Within the two days remaining for the sprint, the text annotation and 2 animation features were introduced to the system. To be able to do this the mobile client application, the routing server and the image processing server interface application were modified. The two features can be used by the user from the “2 frame Animation” and “Add Text” in the drop down list of the image processing form. When either of the two options is selected, a new button for inserting another image for the “2 frame Animation” is shown while a text field is added for the “Text Annotation” option. The two frame animation effect takes 2 to 3 seconds for one frame to change to the other and the text in the text annotation effect appears at the bottom of the image. After the respective fields are filled and the form is submitted, the confirm dialog shows the processed image as illustrated in Figure 28.
Due to the fact that the end of the sprint had been reached according to the modified project plan, the best choice was to postpone all unimplemented features to the next sprint. Figure 29 shows the overall architecture of the application after the second sprint.

When the user enters the login details in the mobile client application and click the “Login button” the client application sends the details to the Instant messaging server directly. The Instant messaging
server gets the login details and authenticates the user by connecting to the database management server with the user login details. If the login attempt succeeds the mobile client sends a request to the routing server to get the user’s contacts from the database server. The routing server gets the contacts from the database server and forwards it back to the mobile client application. To process an image, the mobile form sends the image along with the type of effect to be applied to the routing server which is forwarded to the image processing server. The image processing server processes the image and sends it to the file server via FTP and sends back the HTTP URL of the processed image to the routing server. For the mobile client to be able to access and display the image, the routing server sends the URL of the image to the mobile client. If the client confirms the sending of the image to another user, the link is forwarded to the receiving mobile client to be displayed as a new message.

3.5.4. Testing and Deployment

All the objectives of the sprint were fully met except for the “three complex Image manipulation options” sprint object, only the 2 frame animation effect was complete. The social sharing feature was not totally completed as Image could be shared on Twitter. Tests were carried out to verify that the application worked on the local system. To really be sure the application could be deployed for use outside the system, the application components needed to be on remote platforms and also tested.

At the start of the first sprint the best choice for deploying the application components online remotely was force.com. The original plan was to deploy the routing server on the heroku platform since they claim to support J2EE (Java web applications) applications and migrate the database to be an add-on service for the application. If the routing server had a database server as an add-on service on Heroku, other instances would still be able to connect to it. Part of the plan was also to deploy the J2SE (Java desktop applications) applications to the Force.com platform. The exact details on how the applications would be deployed were not provided on platforms like Microsoft’s Azure and Salesforce’s Force.com for non-registered users. Registration requires account details.

On registering on Force.com and trying to deploy the application components as planned, the full details were provided which showed that the applications needed to be created using the salesforce apex programming language. Furthermore, to deploy java web app on the heroku platform the web application would need to be embedded into a J2SE which would have altered the whole workflow the application component. The Alternative platform, Microsoft Azure became next choice to host the application components as they also claimed to support java applications. The deployment of the components required a migration from the Netbeans IDE (Integrated Development Environment) to
the Eclipse IDE. After the migration was complete, numerous attempts to deploy the application were 
futile as the documentation for deploying the application was old and incorrect and therefore did not 
work. It was later discovered that the Microsoft Azure platform was specifically created to host 
applications developed with Microsoft’s .Net technologies.

After several attempts to find another suitable hosting vendor, the Amazon hosting platform was 
suggested by Mr Talal (Supervisor). It was exactly the right platform for deploying all the server side 
components as the Amazon EC2 service provides the total freedom to deal with the software 
environment. Meanwhile, the Amazon’s Elastic Beanstalk service provides the same environment 
used in deploying the routing server J2EE web application on the local machin. Another very 
beneficial aspect of the Amazon platform service is that it provides each instance (Software 
Environment) with an internal IP address and an external IP address. Which means components can 
communicate within the internal amazon network making the entire system run faster than if the 
components communicated over the internet. They even provide 750 hours free instance uptime for 
new users, the only down side to service is the internal IP address of instances change when they are 
restarted.

The Deployment of each component on the Amazon service went more smoothly than previously, 
after some time was take to study and read the amazon documentation. Each software environment 
was configured to suite the application it would be host. For example, Windows Server 2012 was 
chosen as the Operating system for the Image processing server and the IMAGEMAGICK and Java 
Development Kit were installed to support the running of the Image Processing server Interface 
application. It took some time to get up and running with the technologies and documentation. It also 
took some time to set up and configure the software environment but in the end the deployments were 
successful. The remote deployments were tested by using the mobile client from the emulator which 
was successful as all implemented feature worked properly.

3.5.5. Evaluation

It took two weeks for the server side components to be successfully deployed remotely which left 
little time for the desired evaluation for the 2nd Sprint to be carried out. To get some feedback about 
the progress of the application, a meeting with the supervisor was setup in order for a brief 
demonstration on the software to be performed. The demonstration received a good feedback with a 
suggestion on an improved User Interface (UI) being the only indicated improvement to be done. The 
illustration and discussion below properly explains what was wrong with the User Interface when 
compared to other industry standard instant messengers.
The Figure 30 shows the comparison between the better designed Whatsapp IM owned by Apple and the Ibaana IM. The Whatsapp IM has different types of text fields for the previous messages sent between the users. This gives the user a detailed history of the interactions not only for the received messages, but also for the sent messages. The Ibaana IM’s Interface does not offer that which could affect the user experience. Another clear distinction between the two interfaces is the input section. The Whatsapp UI puts all the elements used for interacting with the other users such as the send button and the text input field at the bottom of the screen. This makes the UI more user-friendly as the user does not need to go around the screen to get access to the different elements. On the other hand the Ibaana IM’s Interface has the interaction elements in two different locations which makes it less user friendly than the Whatsapp IM. For better user interaction with the mobile client, the input section of the Ibaana IM should be more like that of the Whatsapp IM as shown in the Figure above.

Due to the amount of problems encountered while trying to deploy the server side components, the 2nd Sprint ended on the 29th of July instead of on the 11th as shown in the Figure 24 of the project plan after the 1st sprint. This meant that 2 weeks from the 3rd Sprint had been used, therefore best thing to do was to set the goals for the next sprint that would improve on the already existing features of the application. The weekends were used as an overtime measure to make up for the lost time and achieve the goals of the next sprint. The Gantt chart of how the project went is available in the appendices.
3.6. 3rd Iteration

3.6.1. Sprint Goal

The events of the 2nd Iteration left just about two weeks for the 3rd sprint and the 2nd Sprint’s documentation also yet to be complete. Due to this circumstance, it was decided that there would be only three sprint goals. The weekends would also be used as a measure of overtime to complete the documentation of the 2nd Iteration before the end of the 3rd sprint. The sprint goals for the 3rd sprint were to implement the SMS to Internet messaging function, to create a better UI Design and to add more predefined expressive images to the application. The sprint goals were set in a way that would achieve specific objectives of the overall backlog goals thereby gradually completing the overall project.

3.6.2. Design

It was stipulated that to be able to implement the SMS to internet function, a new SMS server should be added. The new SMS server would serve as a bridge between the users logged in to the IM Server with internet connections, and the users logged in with by SMS. By establishing server to server connections between the two servers, users would be able to send messages to each other. The Figure below gives a good representation of the design.

![Proposed Architecture For The 3rd Sprint](image-url)
3.6.3. Implementation

At the start of the implementation, various SMS services were investigated to find out how they could be integrated with this project. The first idea in accordance with the design of the project was to have the SMS IM server function as the internet to SMS gateway. After further research on applications like the Kannel WAP and SMS gateway application, it was discovered that trying to do so might take too much time for the existing duration of the project. The second option was to use an existing SMS gateway service, the Nexmo service was suggested by the supervisor. The Nexmo SMS service is an online service that can be used to receive and send SMS. It serves as a gateway server that allows the sending of SMS messages either through its website, or a custom web application. It also allows users to send SMS messages to a custom application that implement the Nexmo web service. Although it was a great solution for the project, understanding how to use it took some time as there was little documentation of using it as a web service.

The Nexmo web service provides the usage of what is called a “CALLBACK URL”. The call back url is what Nexmo uses to forward or send an messages to a web application. A phone number known as a long virtual number is also provided as the receiving number of the web application that makes use of the Nexmo web service.

In order for the IM server to be able to receive http request, a new http endpoint was needed for the already existing server. To do that, a Jetty server was embedded into the application and servlets were created using J2SE class. The Jetty server is a J2EE server that can be embedded into J2SE applications such as the IM Server application. The servlet acted as the call back url to which SMS from the Nexmo service could be forward to. To test whether the service worked, SMS messages were sent to the long virtual number which were forwarded and printed out to the by IM Server. The class for starting the embedded server and the adding the custom servlet are shown in the Figure 32.

After confirming that the SMS to internet messaging function was operational, the next step was for the users to be able to login to the IM Server. It was also important to make sure messages received by the IM server were forwarded to other users. To able to simulate the behaviour of receiving information from the Nexmo service, an html page with a web form was created containing the fields that will be sent with the http request. The smack library was successfully used to login in the users to the server and send message to other users. The smack library is a java library used to create instant messenger clients, it contains functions like buddy list and chat history that industry standard applications have. To confirm the feature, a simulated test was run using the technique mentioned above. Any errors later found would require little change in the server side code.
With the implementation of the SMS functionality complete, the Nexmo SMS service became a part of the architecture of the application as shown in Figure 33.

```java
public class SmsWebServerListener {
    public static void start() {
        Server server = new Server(6060); //Start the server from org.eclipse.jetty.server.Server
        ServletContextHandler context = new ServletContextHandler(ServletContextHandler.SESSIONS);
        context.setContextPath("/"); //Set the Context Path
        server.setHandler(context);
        context.addServlet(new ServletHolder(new GetSmsServlet()), "/getSms"); //Add Servlet
        try {
            server.start();
            server.join();
        } catch (Exception ex) {
            Logger.getLogger(SmsWebServerListener.class.getName()).log(Level.SEVERE, null, ex);
        }
    }
}
```

**Figure 32. Initializing Embedded Server and Adding A Custom Servlet**

The workflow of the SMS chat starts by the user clicking the SMS MODE button of the mobile client application and login in to sending a username and password. The credentials are sent to the virtual long number of the Nexmo service which forwarded to the IM server. The IM server validates the credentials and logs the user to the IM service then the IM server replies with the “true” SMS message to the mobile device. When the mobile client receives the true text message it navigates the user to the
chat screen were the participant of a chat can be manually specified. If the any of the users sends a message to each other, the message is forwarded to the IM server and relayed to the user. The history and contact list features could not be supported because of the limitation of the number of characters allowed in an SMS message. The implementation of the SMS sending feature is a platform specific feature. This means the code for sending and receiving messages had to be implemented depending on what platform it would be deployed on. This is because the PHONEGAP API does not support any SMS feature. The two other sprint goals could not be implemented due to time constraints.

### 3.6.4. Testing

Before the evaluation could be carried out, all the implemented functions of the application were tested on both emulator and an actual device to verify the functionality and differences on both environments. The table below shows the tests carried out and the results.

<table>
<thead>
<tr>
<th>TESTCASE</th>
<th>EXPECTED RESULT</th>
<th>RESULT ON SIMULATOR</th>
<th>RESULT ON MOBILE DEVICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Attempt Login with correct credentials. Username: <a href="mailto:user1@ibaana.com">user1@ibaana.com</a> Password: password1 Type: Login function Test</td>
<td>The status at the top of the screen should display “connected” before navigation to the contact list page.</td>
<td>Status shows “Connected” message before navigation to contact list page.</td>
</tr>
<tr>
<td>2</td>
<td>Attempt to Login with False credentials Username: <a href="mailto:user1@ibaana.com">user1@ibaana.com</a> Password: password2 Type: Authentication Security Test</td>
<td>The “Authentication Failed” message should be displayed with no navigation to a new page.</td>
<td>The mobile client doesn’t navigate, and the “Authentication Failed” message is displayed</td>
</tr>
<tr>
<td>3</td>
<td>Select a contact and send the message “Hi” to the first user on list of contact. (user will be logged in on another client) Type: IM Message Function</td>
<td>The receiving user should see a “Message received: Hi” on screen.</td>
<td>The receiving user should see a “Message received: Hi” on screen. And the Client Application can to send a message to other contact</td>
</tr>
<tr>
<td>Test (including Contact list)</td>
<td></td>
<td>contact list function works</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------------------------</td>
<td>---------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>4 The receiving user sends back a “Hi There message”. <strong>Type: IM Message Function Test</strong></td>
<td>The message should be seen by the first user</td>
<td>The message should be seen by the first user.</td>
<td>The message should be seen by the first user.</td>
</tr>
<tr>
<td>5 Log back out try to log back to find out if the chat history was saved. <strong>Type: Chat History Tests</strong></td>
<td>The previous message received should be visible in the message test field.</td>
<td>The previous messages received are visible.</td>
<td>The previous messages received are visible.</td>
</tr>
<tr>
<td>6 Apply the “grayscale” effect to an image and sending it to user. <strong>Type: Image Processing Test</strong></td>
<td>It should apply the effect to image and send to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
</tr>
<tr>
<td>7 Apply the “blur” effect to an image and sending it to user. <strong>Type: Image Processing Test</strong></td>
<td>It should apply the effect to image and send to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
</tr>
<tr>
<td>8 Apply the “emboss” effect to an image and sending it to user. <strong>Type: Image Processing Test</strong></td>
<td>It should apply the effect to image and send to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
</tr>
<tr>
<td>9 Apply the “addText” effect to an image and sending it to user. <strong>Type: Image Processing Test</strong></td>
<td>It should apply the effect to image and send to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
</tr>
<tr>
<td>10 Apply the “2 frame animation” effect to an image and sending it to user. <strong>Type: Image Processing Test</strong></td>
<td>It should apply the effect to image and send to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
<td>It applies the effect to image and sends to the other user.</td>
</tr>
<tr>
<td>11 Share a processed image on Facebook. <strong>Type: Social Sharing Feature</strong></td>
<td>An image should be on the user’s timeline on facebook.</td>
<td>The image displays on the user’s timeline on facebook.</td>
<td>The image displays on the user’s timeline on facebook.</td>
</tr>
<tr>
<td>12 Log in using SMS Mode <strong>Type: SMS Function Test</strong></td>
<td>The chat screen should be show after</td>
<td>The mobile client navigates to the</td>
<td>Unable to receive login</td>
</tr>
</tbody>
</table>
Send a message to another user whom has logged in using the internet

Type: SMS Function Test

The other user should receive the message sent by the user making use of the SMS mode.

A simulated message is received by the other user.

Unable to chat due to default text message application problem.

Test the application using the android simulator.

Test: Cross-platform Test

The user should be able to log in and send messages to an android user.

All functionalities apart from the SMS function worked.

No available physical device.

All the tests were successful on the simulator but the SMS feature was not properly functioning on the mobile device. This was because the authentication message that the client application needed to be able to navigate the use to the chat page was received by the default SMS application of the mobile device as shown in Appendix 2.1.

3.6.5. Evaluation

To know how well the application meets the requirements of the user as well as understand the challenges end user could face, the mobile application was evaluated. There were three type of evaluations carried out for the Ibaana Instant Messenger. The first was the expert evaluation, which focused on allowing individual with a good knowledge of Human Computer Interaction principle to use the application and give their feedbacks on its usability. The second type of evaluation was the user evaluation with users of less technical skills evaluating the application and providing feedback on their user experience. The third and final type of evaluation involved using monitoring details from
the platform providers to understand the performance of the components of the application. Individuals that participated signed an agreement to ensure that any data provided during the evaluation would be treated with care and deleted after the purpose of the evaluation was fulfilled.

The expert evaluation was performed with three individuals. Series of tasks were given to the individuals to perform. It contained steps for interacting with another user using the mobile client. After the tasks had been completed, the individuals were asked to fill a questionnaire and answer interview questions on the usability of the application and what could be improved in the application. The questionnaire was used to get the quality of certain usability aspects based on a rating from “very bad”, “bad”, “ok”, “good” and “very good”. The interview questions were given to the individuals in other for them to give more precise feedbacks in their own words.

![Figure 34. Questionnaire Results for Experts](image)

**KEYS:** - 1 ➔ Very Bad, 2 ➔ Bad, 3 ➔ OK, 4 ➔ Good, 5 ➔ Very Good

The Figure 34 shows the result of the questionnaire taken by the experts. The client application performed badly in aspects like visibility of system status, page navigation, error prevention and “recognition, recovery and diagnosing errors”. While in principles like page layout and content, consistency and standards, recognition rather than recall and the match between system and real world
it receive moderate ratings. The application however, received high ratings in ease of data entry and user control and freedom.

In the interview questions given to the experts, they were asked about the usability of the application. The application received negative reviews as they felt the user interface could be much better. When asked about the user experience, two out of three of the individuals gave it a positive feedback as they felt it was a unique way of interacting with other users. Finally when they asked about what they would have like to see improved, various suggestions from a more standard IM interface to sound feedbacks and emoticons were given.

The user evaluation also offered the users with questionnaire and interviews but the information gathered was more focused on the user experience and less technical. In addition, the users were observed while using the client application to interact with other users. An observation form was used to details their events. Two individuals conducted the user evaluation. The figure below illustrates the result of the user evaluations.

![Figure 35. Questionnaire Results for Users](image)

**KEYS:** - 1→ Very Bad, 2→ Bad, 3→ OK, 4→ Good, 5→ Very Good

Figure 35 shows that users enjoyed almost every aspect of the user experience only the page navigation had a “bad” rating from one of the users. This was also supported by the observation results. Users took some time to understand the application but had a good interaction and user experience as they expressed funny amusement with message that they received and sent. When asked about their experience while using the application both users expressed delight on being able to express themselves better with images. They suggested a better user interface for a better user
experience. When asked about how they would want the application to be improved, options like better notifications and better UI were suggested.

As discussed in Chapter 2.6, to properly evaluate the performance of overall system, the performance between each component has to be analysed. During the other evaluations, the amazon network monitor was used to record the speed and performance of each component and how they communicated and sent data to each other.

For the IM and Database Servers the most amount of resources used during the evaluation were the network resources. For the IM Server, the average network influx was between 200 to 300 bytes with a maximum of 500 bytes. It had an average outgoing network bandwidth was 0 to 250 kilobytes (KB) and a peak of 100 KB. The database server’s incoming network traffic only went to just above 20 Megabytes (MB) when requests were sent to it. The outgoing network resource had an average of below 50 KB but frequently rose to around 100 kilobytes. It also had a maximum traffic of above 150KB during the evaluations. For the routing server, image processing server and file server the most notable use of resources were the CPU usage of the amazon instance. The routing server had an average CPU usage of 20% at the time of the evaluation which it exceeded in certain cases. The image processing server had an average of 80% CPU usage and a peak of 100% CPU usage. Perhaps the most surprise thing with the image processing server is the few number of hard disk reads considering the amount of times it had to save and process images. Finally, the file server had variations in its CPU Usage as it had peaks of 100% CPU usage during the first evaluations but had peak usage between 40% and 60% occurring in the last two evaluations.

**Conclusion**

Considering the fact that the project had a number of challenges and unfamiliar technologies had to be learnt and used, the end result was a good one. I had to learn a lot of new technologies to implement all the components like the mobile client, the IM server and the image processing server. I learnt new technologies like Phonegap, JQuery Mobile, ImageMagick, Strophe.js, the embedded jetty server and other technologies used in this project. The use of project management methodologies like scrum and extreme programming made adjusting to change and documentation easier. It also determined the quality of the final prototype. The expectation of having to create prototypes that meet both the requirement and user needs have ensured continuous broadening of creative thinking and problem solving. Although not all requirements could be achieved, the testing and evaluation of the final prototype provided an insight into the quality of the product and idea. All in All, it was a very worthwhile and productive experience that created a product that can be extensively improved since the entire components have been implemented.
Further Work

There is no limit to the improvements that can be made to Ibaana Instant Messenger and what it can provide to its users. For the application to be usable by all end users, firstly, the problems encountered during the development process and those discovered in the evaluation process need to be fixed. Secondly, Since all components of the application have been implemented new features can be added like, an introduction of picture libraries, allowing users to create comic strips and creating expressive images from scratch. The ability for users to express themselves can be taken further by providing them with the ability to create custom videos and even combining it with images. Introducing new platforms for users like adding a web platform where users can get access to the features of the Ibaana instant messenger will expand the reach of the application. This will not only bridge the gap of communication between mobile and web users but also integrate users with different means communication. SMS users would also be able to send messages and expressions to web and mobile users and vice versa if the web platform is introduced.
References


Appendices

1. Project Events
2. Screen Shots

2.1. SMS Authentication Message Received to Default SMS Application

![Screen Shot of SMS Authentication Message]

2.2. Database Server Network Output

![Network Output Graph]

*Times are displayed in UTC.*
2.3. Image Processing Server’s CPU Usage

![CPU Utilization Graph]

Monitored Instances: i-144000d71

Times are displayed in UTC.

2.4. Sharing Images on Facebook

Mohammed Olateju
July 4 via IbaanaMobileClient

Ibaana Image
Ibaana Image
Like · Comment

Ibaana Image
Ibaana Image
Like · Comment
2.5. The Servers Deployed as Amazon Server Instances

<table>
<thead>
<tr>
<th>Name</th>
<th>Instance</th>
<th>AMI ID</th>
<th>Root Device</th>
<th>Type</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>ibaanaiMServer</td>
<td>i-04091865</td>
<td>ami-c30360aa</td>
<td>ebs</td>
<td>t1.micro</td>
<td>running</td>
</tr>
<tr>
<td>ibaanaiDBServer</td>
<td>i-8a96f94</td>
<td>ami-957edcf1</td>
<td>ebs</td>
<td>t1.micro</td>
<td>running</td>
</tr>
<tr>
<td>ibaanaiPServer</td>
<td>i-14400d71</td>
<td>ami-800bc06</td>
<td>ebs</td>
<td>t1.micro</td>
<td>running</td>
</tr>
<tr>
<td>ibaanaiFileServer</td>
<td>i-5353ae36</td>
<td>ami-600bc09</td>
<td>ebs</td>
<td>t1.micro</td>
<td>running</td>
</tr>
<tr>
<td>ibaanaiRoutingServer</td>
<td>i-a457d8c1</td>
<td>ami-1624987f</td>
<td>ebs</td>
<td>t1.micro</td>
<td>running</td>
</tr>
</tbody>
</table>

**EC2 Instance selected:**

**EC2 Instance:** ibaanaiMServer (i-04091865)

Last updated: 23.23.77.214

**Description**

- **AMI:** ubuntu/images/abs/ubuntu-raring-13.04-amd64-server-20130423 (ami-c30360aa)
- **Zone:** us-east-1a
- **Type:** t1.micro
- **Scheduled Events:** No scheduled events
- **VPC ID:** -
- **Source/Dest. Check:** -
3. Evaluation Data Gathering Documents

3.1 Agreement Form

AGREEMENT

This Evaluation Agreement (Ibaana Instant Messenger) effective as of 12/08/2013, is between Mohammed.B.Olateju, students of Heriot-Watt University (Dubai campus), and ___________________________ __________________________.

CONFIDENTIALITY: Mohammed.B.Olateju shall not (i) transfer or otherwise make the Samples available to any third party; (ii) disclose the Related Information to any third party; or (iii) use the Samples or Related Information for any use that is inconsistent with the terms of this Agreement. Mohammed will limit access to the Samples and Related Information to those of its members and consultants.

All personal data collected will only be used temporarily during the evaluation exercise and will be deleted after the purpose of the evaluation is complete.

DISCLAIMER: This evaluation is only for testing. It has no harm to your health, and the evaluation is only for academic purpose, and it shall not be used in any profit making activity.

By signing this form, you have acknowledge reading this agreement and understanding all the terms and conditions.

I agree

………………………….                                                                ……………………….

Signature                                    Date

15/08/2013
3.2 User Evaluation Tasks

Ibaana Instant Messenger User Evaluation Tasks

Tasks

1. Log in username and password.

2. Select the first contact on the list to chat with.

3. Send a “Hello” message to the contact, if message is sent to you send a "Hay Hello There" reply.

4. Send an image to the contact asking about him/her, if an image is sent to you send an adequate reply.

5. Send an image to the contact to express a happy mood, if an image is sent to you send an adequate reply.

6. Send an image to the contact to express a sad mood, if an image is sent to you send an adequate reply.

7. Send an image to the contact to express confusion, if an image is sent to you send an adequate reply.

8. Send an image to the contact to express anger, if an image is sent to you send an adequate reply.

9. Log Out by navigating back to the login screen.

10. Log In using the SMS Mode by clicking the SMS Mode Button.

11. Send a “Hello from Sms Mode” message to a contact (If you are logged in as user1@ibaana.com send a message to user2@ibaana.com or vice versa).

3.3 User Questionnaire

User Evaluation Questionnaire

Please tick the options to answer the questions about the Application.

<table>
<thead>
<tr>
<th>Questions</th>
<th>Very Bad</th>
<th>Bad</th>
<th>OK</th>
<th>Good</th>
<th>Very Good</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. How Would you rate your experience when using the application?</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>
3.4 User Interview

**Interview Questions for Users**

1. How well were you able to express yourself and how would you rate the User Experience?

2. What do you think needs to be improved in the application?
### 3.5 User Observation Form

<table>
<thead>
<tr>
<th>TEST ID</th>
<th>TEST CASE</th>
<th>TEST VALUE</th>
<th>TEST STEPS</th>
<th>EXPECTED RESULTS</th>
<th>PASS/FAIL</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Log in username and password.</td>
<td>To Login and Authenticate the user to the Ibaana IM Server.</td>
<td>Type Username and password and click the login button to login.</td>
<td>The User is Authenticate d and logs in.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Select a contact to chat with.</td>
<td>To show that users can view added contacts and select one to chat with.</td>
<td>Select the first contact on the list to chat with.</td>
<td>The user should be able to select a contact from the contact list</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Send a &quot;Hello&quot; message to the contact</td>
<td>To test that the users can sent text message to each other.</td>
<td>Send a “Hello” message to the contact, if message is sent to you first send a &quot;Hay Hello There&quot; reply.</td>
<td>The users should Receive and Send Messages to each other.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Send an image to the contact asking about him/her.</td>
<td>To Test the applications ability to let the user inquire about the other user</td>
<td>Send an image to the contact to express a happy mood, if an image is sent to you send an adequate reply.</td>
<td>The users should be able to pick an image and apply an effect for the expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Send an image to the contact to express a happy mood.</td>
<td>To Test the applications ability to let the user themselves through with the image processing feature.</td>
<td>Send an image to the contact to express a happy mood, if an image is sent to you send an adequate reply</td>
<td>The users should be able to pick an image and apply an effect for the expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Send an image to the contact to express a sad mood.</td>
<td>To Test the applications ability to let the user themselves through with the image processing feature.</td>
<td>Send an image to the contact to express a sad mood, if an image is sent to you send an adequate reply.</td>
<td>The users should be able to pick an image and apply an effect for the expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>--------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Send an image to the contact to express confusion.</td>
<td>To Test the applications ability to let the user themselves through with the image processing feature.</td>
<td>Send an image to the contact to express confusion, if an image is sent to you send an adequate reply.</td>
<td>The users should be able to pick an image and apply an effect for the expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Send an image to the contact to express anger.</td>
<td>To Test the applications ability to let the user themselves through with the image processing feature.</td>
<td>Send an image to the contact to express anger, if an image is sent to you send an adequate reply.</td>
<td>The users should be able to pick an image and apply an effect for the expression.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Log Out</td>
<td>To Log Out the User and test the SMS Mode.</td>
<td>Log Out by navigating back to the login screen</td>
<td>The user should be able to log out of the current session.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Log In using the SMS Mode</td>
<td>To the the login and Authenticatio n of user via sms.</td>
<td>Log In using the SMS Mode by clicking the SMS Mode Button.</td>
<td>The users should be able to login using with Sms Technology</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Send a hello message to a contact.</td>
<td>To test the ability for the application to send a message to other users on the internet using sms.</td>
<td>Send a hello message to a contact (If you are logged in as <a href="mailto:user1@ibaana.co">user1@ibaana.co</a> m send a message to <a href="mailto:user2@ibaana.co">user2@ibaana.co</a> m or vice versa).</td>
<td>The user should be able to send sms message to another user</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 3.6 Expert User Questionnaire

#### Heuristic Evaluation Questionnaire

Please tick the options to answer the questions about the Application.

<table>
<thead>
<tr>
<th>Questions</th>
<th>Very Bad</th>
<th>Bad</th>
<th>OK</th>
<th>Good</th>
<th>Very Good</th>
</tr>
</thead>
<tbody>
<tr>
<td>1   How well does the application keep the user informed about what is going on, through appropriate feedback within reasonable time <em>(Visibility of system status)</em>?</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>2   How would you rate the each page’s layout and content?</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>3   How Would you rate the Page navigation?</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>4   How well would you rate the real-world conventions used in the application <em>(Match between system and the real world)</em>?</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>5   How well does the application support undo and redo without having to go through an extended operations <em>(User control and freedom)</em>?</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>6   Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions <em>(Consistency and standards)</em></td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>7   How well does the application eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action <em>(Error Prevention)</em>.</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>8   How well does the application Minimize the user's memory load by making objects, actions, and options visible and have to remember information from one part of the dialogue to another <em>(Recognition rather than recall)</em>.</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>9   How well does the application Help</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
</tbody>
</table>
Name: .....................  Sign: ..............................  Mobile: ..................

3.7  Expert User Interview

**Interview Questions for Experts**

1. How Well Was Usable Was the Application?

2. How well were you able to express yourself and How would you rate the User Experience?
3. What do you think needs to be improved in the application?

Name: ..........................  Sign: ..............................  Mobile: ..............................
4. Project Log

These are the log of the project used in accordance with the sprint methodology. They were recorded after each day of project activity. Exported from the Evernote Application.

**After the 1st Day**

<table>
<thead>
<tr>
<th>Created: 07/05/2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author: Mohammed</td>
</tr>
<tr>
<td>Tags: 1st Iteration, Requirement Spec</td>
</tr>
</tbody>
</table>

Downloaded and installed persivable softwares needed for the application: Like ADT, Visual Studio, Netbeans, Vmware & IOs.

I still have an abstract understanding of what i want to do and i know some of the technologies i need to use to implement them.

**[Sprint Goal]** I have decided that i will try to implement the IM Server and its Client and also implement image processing server in the first Iteration.

Today i will be finding out how how to implement the IM Server.

---

**Finding out about the IM Server[Start of 1st Sprint]**

<table>
<thead>
<tr>
<th>Created: 08/05/2013</th>
</tr>
</thead>
<tbody>
<tr>
<td>Author: Mohammed</td>
</tr>
<tr>
<td>Tags: 1st Iteration, 1st Sprint</td>
</tr>
</tbody>
</table>

I tried to find out how to setup the IM server to use, i stumbled upon openfire XMPP Server. I like it because was implemented using Java and i could also extend it using Java.

I wanted to implement the server using the Openfire but i decided to find out how i would deploy it online. I immediately found out that deploying an IM server online wouldn't be as easy as deploying a web application. And i didn't want to make the IM server itself a web application because web applications are demand-driven and IM applications should be message driven.

I found out that Google App Services doesn't permit applications that created socket connections using XMPP so i couldn't deploy it. The only options are Salesforce.com and Microsoft's Azure both of which are paid services. Although i might prefer salesforce.com to Microsoft as it is cheaper.
Finally I think I will use the Apache project called Vysper, it seems to be a simple xmpp server without the over complicated bells and whistle features in openfire like its web interface log.

**Starting the Development of the IM Server**

Created: 09/05/2013  
Author: Mohammed  
Tags: 1st Iteration, 1st Sprint  
Source: [https://mina.apache.org/vysper-project/user_guide.html](https://mina.apache.org/vysper-project/user_guide.html)

**Accomplish:** Able to run the xmpp server by compiling and running the source code from the Netbeans IDE.  

**Obstacles:** Getting the dependencies where quite difficult as the specific jar files are needed to run the application where not specified and I had to research each one. Fortunately at the end of the day I found a very easy way of getting them which makes me wish I had known how to do it early :(.

I have decided to read more in detail before trying to implement things as I wouldn't run it to as much problems as I did.

**Objective Before The Next Meeting:** Finding out how to create a client to communicate with the server by researching on how to logging and manage users. Also how to make users communicate with each other.

**From IM Server to IM Client**

Created: 12/05/2013  
Author: Mohammed  
Tags: 1st Iteration, 1st Sprint

**Accomplished:** Configuring the development of IM Server. I thought I had gotten all the dependencies I needed to develop the IM server but I didn't. Now I have fully configured the development of the IM server and I am partly through with the configuration of the the IM client's development environment.

I want to develop the client of the application using phonegap on the blackberry platform as I use a blackberry phone.
Obstacles: I had problems configuring the development environment for the blackberry development particularly with the ant tool.

Objective Before the Next Meeting: Connect the mobile client with the XMPP server by allowing users log in to the XMPP Server.

Still Trying to build on Blackberry

Created: 13/05/2013
Author: Mohammed
Tags: 1st Iteration, 1st Sprint

Accomplished: I succeeded in downloading the the phonegap tool and integrating it with the development environment, however i am having problems building the code for the blackberry simulator.

I have decided to simulate the development of the application using the current model of the phone i use so that i can use it for practical testing during development.

Obstacles: I have faced serious problems in building the application but finally solved the problem which is down to the fact that blackberry are not supporting older sdks. I had to revert to the old jdk 6_21 for the build to work.

Objective Before Next Meeting: To Connect the mobile client to the IM Server.

Making the Mobile App an IM Client

Created: 14/05/2013
Author: Mohammed
Tags: 1st Iteration, 1st Sprint

Accomplishment: I was able to add the bosh component as an extension to the xmpp server to allow web access to the server from a client. And used the code from the book "XMPP Programming with Javascript and Jquery" to get up to speed with connecting the a client with the server using http.

Obstacles: I have been unable to connect the mobile client with the xmpp server, i still feel i am missing something but will solve it today Insha-Allah.

Objective before next meeting: To establish a connection between the mobile application and the XMPP Server. And to integrate version control into the development process.
Web Communication with XMPP Server Established

**Created:** 15/05/2013  
**Author:** Mohammed  
**Tags:** 1st Iteration, 1st Sprint

**Accomplished:** I was able to establish communication to the XMPP using the Strophe Javascript Framework and allow users log in to the server from a web browser by adding a slash to the end of the url "localhost/bosh/" which was the solution to the previous problem. I also discovered that rather than test every single code on the simulator i can test the code on a web browser before porting it to a simulator.

I was made each project a git repository and decided not to host my codes online simply because hosting my code online would be for collaboration.

**Obstacles:** I was able to establish communication with the XMPP server but several log in attempts failed which was really frustrating. I kept adding dependencies as i had a hunch that it was down to the dependencies that it wasn't validating. Eventually i added the admin console module which which seemed to be the missing dependency.

**Objective before Next Meeting:** I want to make it possible to send and receive message between users and test it on the simulator.

Login In To The XMPP Server From The Simulator

**Created:** 16/05/2013  
**Author:** Mohammed  
**Tags:** 1st Iteration, 1st Sprint

**Accomplished:** I was able to load a working client from the a webpage to the blackberry simulator which means the IM Client can connect from the smartphone to the XMPP server and login.

**Obstacles:** At first i was unable to connect the im client to the xmpp server even though i was working when hosted on the wamp server i finally solved the problem by loading the simulator the right way, adjusting the permission of the app on the simulator to access the internet and using the public IP address of my computer instead of 127.0.0.1.

**Objective for the Next Meeting:** Send and Receiving information from the IM Client.

*Optional if there is enough time* Designing the UI with Jquery Mobile
All round Communication successful

**Accomplished:** I was able to make sure that a client could send a ping to the server and receive a pong and clients can now communicate with each other even using a simulator. I noticed that the Strophe.js uses the AJAX long-lived call technique in maintaining a connection with the server. I can use google chrome to check the interaction between the client and the server.

**Obstacles:** I didn't face much obstacles in making sure two client could send messages to each other except the learning process. I also didn't have enough time to design the User Interface because I hard to learn how to used JQuery Mobile first.

**Objective Before Next Meeting:** Designing the Mobile Interface with JQuery Mobile to fit the Instant Messaging objectives properly.

IM Client Design Not Fully Complete

**Accomplished:** I was able to learn ways to design the mobile user interface using JQuery Mobile. I have also started the designing it but I have not been able to finished the design since nothing is actually working as they say should.

**Obstacles:** The Layout and the look and feel of the interface is not looking as good I want it to be. I must be missing something. :)

**Objective for Next Meeting:** To Finish the UI Design

Mobile Interface Design Done

**Created:** 21/05/2013
Accomplished: I was able to finish the UI design I wanted, the problem was a missing file which didn't allow the structure to be how it was supposed to be. I now have 2 pages the index.html and the chat page. The index page contains the user login screen and the chat.html is where users can send messages to contacts.

Obstacles: I have found out that there need to be a way of renewing the connection when the user navigates to the chat.html. It isn't ideal to use cookies therefore I would have to used phonegaps API but first I would have to test if the functionality works using cookies.

Objective for Next Meeting: Renewing Connection with the server when navigating from index to chat page after login.

Blackberry Simulator And Phonegap Problems

Accomplished: Found out that connections can be renewed after navigation from one page to another but the username, Session ID and RID have to be stored temporarily. I stored them in a cookie for the web browser use now I have to use the phonegap api to store the session information temporarilily and retrieve it to restore the session with the database.

Obstacles: I have been facing several issues trying to use the storage api for phonegap as any code writing after the database code doesn't work its either an error from me or the simulator & phonegap api don't work well.

Objective for Next Meeting: Store session information temporarily and renew session with XMPP server after login and page navigation.

Problems Solved, Now Using New Emulator
Accomplished: I was able to solve the problem of saving temporary data into a database and retrieving it after login using Ripple Emulator which i would be using from now on as it is what is recommended by Blackberry. I can now start the development of image manipulation.

Obstacles: I had problems with the SQL syntax of phonegap and call back structure of javascript at first but i was able to overcome it.

I was previously switching between the web browser and blackberry simulator to run my code which took a lot of time. But now ripple emulator solves the problem ad i can test my code and understand what is going on by looking at the interaction of the application using google chrome.

Objective Before Next Meeting: Setting up the environment and Learning about GIMP Development.

---

Setting Up the GIMP Environment

Accomplished: I was able to start setting up the development for GIMP, I am now finished with the 1st stage out of 3 on setting up the environment for gimp. I am also half way through the 2nd stage.

Obstacles: I have had problems with how long it take because setting up the GIMP environment involves compiling c and c++ codes using msys and mingw.

Objective Before Next Meeting: Finish the Set up of the GIMP environment and Learning the plug-in Development.

---

GEGL and BABL Problem Solved

Accomplished:
Accomplished: I went 1 step further in the setting up of the gimp environment although there is still a lot to do and the development of plug-in to learn.

Obstacles: I have really hard problems setting up the gimp environment as very little tutorial has been provided and compiling trips up so many dependencies not correctly specified.

Objectives For The Next Meeting: Finished the Setup of the GIMP environment and Learning the plug-in Development.

Gimp Configured....Only [Make] Install Left

Accomplished: I was able to solve a few more dependency problems that where not specified in the tutorial, I even had to email the author of a tutorial article on "how to build gimp" tell him that i have a problem. Thankfully i was able to solve the problem by specifying the path to the TIFF and JPEG Library needed for gimp to support jpeg and tiff files and creating a deprecated python script. Now that the application is configured all that is remaining is building it from the source code so that it can be executed, i have to say that excites me as it will provide me with enough flexiblity when i get up to speed on how to develop it.

Learned about JGIMP framework, which allows the development of GIMP as a server side application with. Hopefully it will be good enough for achieving my objective

Obstacles: The Lack of developer support and documentation is proving to be a big problem as finding solutions to problems are quite hard and not all instructions are available online in one place some instructions have to be searched for or figured out which takes a lot of time.

Objectives For The Next Meeting: To [make] install gimp from the source code.

Almost done with installing installation as more dependencies are
Accomplished: Almost done with installing installation as more dependencies are being found.

Obstacles: Building Gimp is still taking a lot of time as missing files and bugs are frequent.

Objective For Next Meeting: Finish Gimp Building

---

**Light At The End Of Tunnel**

Created: 30/05/2013

Author: Mohammed

Tags: 1st Iteration, 1st Sprint


Accomplished: Yesterday i was able to solve the a couple of problems the compiling the binary files for gimp. But compilation is almost done. I solved the binary compilation problems by downloading available binaries from the stuff.mit.edu website.

Obstacles: Having to search randomly for solutions to problems that are undefined. I also found out that i didn't really understand some of what was written on the building gimp by pantha.com, seems i had to run into some problems before i could understand some of what was written. This mostly due to my unfamiliarity with c/c++ programming and GNU Tools, well now i know about it :).

Objective For Next Meeting: complete compile and install process

---

**gimp-2.7.exe Problems Again**

Created: 02/06/2013

Author: Mohammed

Tags: 1st Iteration, 1st Sprint

Accomplished: NONE

Obstacles: Still trying to solve problem as gimp-2.7.exe and I though of the fact that having to download and copy gimp binary was not sensible as i would be building and compiling continously a permanent solution to the problem needs to be found.
I sent and email to the writer of the article which i am currently using to learn how to compile gimp and he replied and told me that its a problem with my MinGW.

**Objective For Next Meeting:** Find A permanent solution to gimp build and compiling errors.

*Note: I have decided to extend this sprint to 4 or 5 weeks instead of the 3 weeks as originally planned. Due to the Obstacles i have been facing as each sprint must have a working prototype as a measure of success.*

---

**Gimp Compiled**

**Created:** 03/06/2013  
**Author:** Mohammed  
**Tags:** 1st Iteration, 1st Sprint

**Accomplished:** I was able to successfully compile GIMP finally, I got a big help from the writer of an article by sending him an email about what i my problem he quickly told me where the problem was coming from. I traced back my steps and found out i was using 64bit instructions for a 32bit compiler.

I also found out that i cant create a plug-in to communicate over a network so it would be better to integrate gimp with the existing Xmpp server. I was further encourage by this idea as i also found out that i could use the gimp-console to process images which means i could just process an image without the need for the gui. I could create a module to automatically load the gimp-console application, process the image and close it on user's demand. But that means there is a lot of research and work to be done.

**Obstacles:** I tried to run the gimp application but it didn't work due to the fact that it couldn't find some libraries i would have to copy over some libraries as i want gimp to work independent of the environment.

**Objective For Next Meeting:** Make gimp work independently from a directory, so that it can be added to the xmpp server

---

**New Image Manipulator Program Chosen**

**Created:** 04/06/2013  
**Author:** Mohammed
Accomplished: I have decided to change the image manipulations program I am to use for this project from Gimp to Image Magick. The reason being, first of all that the fact that Gimp is very unstable for development and compiling and using takes far too long with little result as my previous experience has shown. The second reason is that Image Magick works very well with command line instructions when compared with gimp which is the way I tend to parse instructions from the xmpp server. Furthermore, I can easily integrate ImageMagick with the overall system as it has a java programming interface to which means the xmpp server can communicate directly with the image manipulation program, and Image Magick seems to be far more stable as it has had 6 releases while gimp has had only 2 releases.

I am very happy that I was able to find a better option than GIMP to be able to have a proper image processing application for this project.

Obstacles: I have had problems with 32bit and 64bit c/c++ programs and will try to evade such problems in the future hopefully I will be able to do that with this new software.

Not finding image magick earlier is definitely a stumbling block, but its mostly down to the fact that there are very few written articles on image magick and the research report was based on articles and journal. Regardless of this, I am happy to have stumbled on it on a forum.

Objective For Next Meeting: Learning more about Image magick and how to integrate it with the xmpp server and client.

Start the creation of Image Processing Server

Created: 05/06/2013

Accomplished: I was able to learn a lot about how to create the image processing server. Right now I think I would need to also add a web server that would get the image from the mobile client and send it to the image processing server for processing. I believe this is the right decision as I want the mobile client to be sort of a thin client. I also found out that there is a java interface(im4java) that I could use to create the server Interface for the image processing server, this interface would receive a file from the web server process the image in an on-demand form and send it back to the web server. I am really pleased that I have been able to find proper architecture and solution for the project.

Obstacles: I found out I would have to use socket and servlets to implement the project, although it isn't complicated it might take some time to get used to. I would also have to make the standalone java interface for the server able to run for a long periods of time but I am not sure exactly how I would do that although im4java have claimed to have proper support for it. I would have to see.
**Objective For Next Meeting:** Implement Socket programming between the web server and image processing server interface.

---

**Image Processing Server Creation Process Started**

**Created:** 06/06/2013  
**Tags:** 1st Iteration, 1st Sprint

**Accomplished:** I was able to create the server interface successfully, it can now process images from a Java application using the command line. I was also able to simulate a client/server socket interaction. I intend to migrate the client to a web server where the web server would send the image to the image processing server interface and the image processing server will process the image and send it back to the web server to be displayed.

**Obstacles:** I predicted that I would have to get used to socket programming and that was the case. I hope for a faster progress with servlet programming.

**Objective of Next Meeting:** Create a client/server model between the web server and image processing server interface.

---

**Image Transfer Between IP Server & Routing Server**

**Created:** 09/06/2013  
**Author:** Mohammed  
**Tags:** 1st Iteration, 1st Sprint

**Accomplished:** I was able to start the creation of the routing web server that now sends an image to the Image processing server interface and gets a reply. I originally wanted to transfer images both ways but found out it would take too much internet and CPU resource. But that means I might have to add an FTP server. Now the web server can accept images, send it to the Image Processing server via sockets and receive a reply after the Image Processing server processes the image.

**Obstacles:** It took a while for me to get used to socket programming and I had to reconsider sending the image because I could see that it was taking some time. I prefer-ed to host it on a server of course that could mean I would be going a bit further outside the scope of my iteration again.

**Objective of Next Meeting:** Add the processing of image feature and making the web server display the processed image after it is sent back by the image processing server.
**Processing and Sending Displaying Images on Reply**

**Created:** 10/06/2013  
**Author:** Mohammed  
**Tags:** 1st Iteration, 1st Sprint

**Accomplished:** I was able to send the images for processing to the image server for processing and allow the images to be displayed by the web server. Although this is a client/server model, the user should be able to choose which effect they would like to apply.

**Obstacles:** I am having problems sending user selection of the type of effect that is to be applied to the image because although I send both the image and the textual selection of the user separately it all ends up together when it is receive by the image processing server interface.

**Objective For Next Meeting:** Allow the user to select the type of effect to be applied along with the image to be processed.

---

**Web/Client Model For Web/Image Server Achieved**

**Created:** 11/06/2013  
**Author:** Mohammed  
**Tags:** 1st Iteration, 1st Sprint

**Accomplished:** It took me all day but I was able to send both the user selection and image to server and create an appropriate response although I might have to make it more robust and scalable later on.

**Obstacles:** I had a lot of problems with sending the information using sockets. I had to open/close 2 sockets to send 2 pieces of information. I am not totally convinced the method will work in a multi-threaded demanding environment with multiple simultaneous requests.

**Objective For Next Meeting:** Allowing the mobile client to send images to the server for processing as done with the web server.

---

**Migrating Functionality With To Mobile Client**

**Created:** 12/06/2013
Accomplished: I was able to design the UI for to allow mobile users send process image, now the UI has options for users to send either images or text although the send image option is not fully functional.

Obstacles: I have had issues with submitting the form as a button that is supposed to load the image for submission isn't working its certainly an issue with jquery mobile which i would have to work around.

Objective For Next Meeting: Allow users process Images via mobile client.

---

Progress on Mobile Client Migration

Created: 13/06/2013
Author: Mohammed
Tags: 1st Iteration, 1st Sprint

Accomplished: I was able to migrate the form functionality of the web form to the mobile client now the user has photo effects options and can process a selected image. Although the UI needs to be be improved i am happy as I am just 1 step closer to finishing this sprint.

Obstacles: I had problem with the phonegap API as i was not able to access phonegap specific picture API for selecting a picture on the user device. Fortunately the file input future also works on devices so i can use it for now and try to figure out the problem with the phonegap API in the next Iteration.

I also have a problem with the response sent by the server as the server sends html encoded message with the image embedded in it to the mobile client as a response. I want the presentation code to always be on the client side, i might make the response of the server to be a link for displaying the processed image from the ftp server.

Objective For Next Meeting: Allow users send processed images to each other.

---

Mobile Client Migration Complete [Marking The End of 1st Sprint]

Created: 16/06/2013
Author: Mohammed
**IBAANA: The Cross-Platform Mobile Instant Messenger**

**2012/2013**

**Tags:** 1st Iteration, 1st Sprint

**Accomplished:** Alhamdulilah, I was able to completely migrate the chosen functionalities of the web server to the mobile client. Now the Mobile client can send images to be processed and send the processed images to other users. When it is sent to a user, a popup with containing the image is shown on the screen of the receiving user.

Now i will try to start the documentation of the project before starting the with the next Iteration

**Obstacles:** I had a number of problems with sending asynchronous multipart message then having to byParse cross platform restrictions. I also had problems with jquery mobile as the popup feature seemed have bugs.

---

**1st Sprint Retrospective**

**Created:** 16/06/2013

**Tags:** 1st Iteration, 1st Sprint

**How the Sprint Went:** Was able to achieve objective of the sprint so yes the sprint went well.

**Improvements:** Definitely need to be more efficient, each day 1 goal was set. To improve performance, 2 goals can be set in a day.

---

**Beginning the 2nd Sprint**

**Created:** 24/06/2013

**Tags:** 2nd Iteration, 2nd Sprint

**Accomplished:** I decided on the goals of the current sprint and was able to refactor and comment on the codes to make them a bit more robust and scalable.

**Sprint Goals:** To improve Ibaana instant messenger by:

1. Introduce all components of the architecture to the overall system.

2. Provide 3 more complex Image manipulation options
3. Provide text annotation on Images

4. Allow users share their expressions on Facebook and Twitter.

5. Provide a better instant messaging network features i.e contact list and chat history.

**Obstacles:** I had a problem in refractoring the Routing server code as i wanted to make the Image processing server less connected to interface because data was forwarded by the routing server was sent to the Image processing server in the same structure as the form on the mobile client. This meant if the structure of the mobile app was changed then there would be a problem in the Image processing server.

**Objective For Next Meeting:** Introduce all components of the overall architecture to the system and establish communications between them.

---

**The Creation Of A Dynamic Architecture**

**Created:** 25/06/2013

**Author:** Mohammed

**Tags:** 2nd Iteration, 2nd Sprint

**Accomplished:** I was able to find out how to implement an overall architecture that can not only work on my local system but also may be able to host the server side part and install the mobile client.

I wanted to implement the architecture in a way that after the Image Processing server interface is done with processing the image, it sends it to a File server that can also display the image via http. I researched and found out there where multiple online vendors that could do it and i could also use xampp server to imitate that behaviour.

I also looked for a way of integrating the database server into the architecture which turns out i could use heroku to host the Routing server and have a database for it. I could also connect to the database from the xmpp server which i would most likely have to do when extending the xmpp server to hold usernames and chat history.

**Obstacles:** I had a lot of problems trying to get a suitable software for imitating an industry standard file server and finding out how to connect to the ftp server using java. But i found out xampp server would do that and i could use apache's ftp java library for sending to an ftp server.

**Objective For Next Meeting:** Implement ftp server into the overall architecture and a database to the overall architecture.
**1 Step to Full Architecture**

**Accomplished:** I was able to merge the new FTP server with the overall architecture of the system. This means that I can now host the whole system online, and make them work with the mobile client. And now the Image processing server can now create temporary files which also means the application is more dynamic and platform independent as a specific location doesn't need to be provided before for getting the image from the routing server.

The next thing to do is add a database server to the architecture. I have decided to use MongoDB as the database technology for its flexibility and performance in a project that will be very susceptible to change.

**Obstacles:** I had some problem with the ftp server as the image wasn't being transferred in the right way. I had to do some research and change the encoding scheme to a binary format.

**Objective next Meeting:** Adding the Database server to the architecture. This can be like killing two stones with one bird as to properly integrate the database server I would have to delegate some functionality to it. Therefore I could implement the goal of having a chat history and contacts which would mean the completion of two goals.

---

**Dynamic Overall Architecture Achieved**

**Accomplished:** I was able to add the database server to the overall architecture and access the database via HTTP from the XMPP Server. Now it will be easy to upload the entire architecture online with only some changes to the access points of the each component.

I was also able to start the customization of the xmpp server to be able to build on the existing server without having to change the original code.
Obstacles: I had to learn how MongoDB works in order to fit it well with the architecture by reading some documentation and watching web casts. I also had to learn how to build on the xmpp server as i was running as it was from the box.

Objective for Next Meeting: To Implement the saving of chat history, contact list and User Authentication using the database server.

Killing Two Birds With One Stone

Created: 30/06/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: After adding the database i was able to store the history of user chat and get contact list from the database using ajax and the web server as the intermediary. The mobile client sends the chat history to the Routing web server and it is stored in the database server. The same thing also happens with the contact list as the mobile client is now able to retrieve user contacts from the database server asynchronous with the help of the Routing server.

Obstacles: I had problems understanding the MongoDB technology and how data should be modelled, although i have been able to create a database that information can be retrieved from i will need to make sure that i have the right Data Model.

Objective Next Meeting: Implementing User Authentication with the use of the database server, 2 objectives of this sprint will be met. I will also have to make sure that the database has the right structure.

User Auth, Chat History & Contact List with database achieved

Created: 01/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: I was able to make sure a user can chat with a contact on the selected user from the list of contacts. I was also able to implement the loading of the previous chat history when entering the chat screen. I was also able to make sure that the mobile application can login a user using the records from the database.
Now the database server is fully functional with the overall system which means I can now focus on the improvement of the behaviour and functionality of application. It also means 2 Objectives of the Overall sprint goal has been met.

**Obstacles:** I tried to improve the structure of the database but it meant I would have to change the structure of the code that gets the information that connects to the database directly there it'll take a lot of time I have decided to add it to what I have to do in the next sprint. Thankfully if I change the Routing and IM server to conform to the database, it necessarily won’t affect the code of the mobile client. :)  

**Objective For Next Meeting:** To implement the sharing of images on Facebook and Twitter

---

**UI Problem Realization**

**Created:** 02/07/2013  
**Tags:** 2nd Iteration, 2nd Sprint

**Accomplished:** I was able provided a better database structure for the application and modify the other components to conform to the new database design. Although it is not the best at the moment, it should be suitable for the goals of this sprint.

**Obstacles:** Although improved the database structure when trying to modify the mobile client to suit the other components, I found out that the arrangement of the code of the was not good enough. Because even though I achieved what I wanted the presentation and flow of functions wasn’t good enough.

**Objective for Next Meeting:** Improve the structure of code of the mobile client and learn about how to post images on Facebook and Twitter.

---

**Mobile Restructuring Started**

**Created:** 03/07/2013  
**Tags:** 2nd Iteration, 2nd Sprint

**Accomplished:** I was able to properly restructure the existing code to a more readable scalable manner. I even found ways of making the mobile client code run faster than usual which is a big psychological boost considering the amount of time spent on it. I also do not need to create databases for storing user session information as the new code make retains the user session in-between page transitions.
Obstacles: Although I was able to modify the code, I had a lot of problems with JQuery mobile’s presentation of the page which was later fixed by research it after countless hours of frustrations. But the mobile client is now up and running with a better and faster code.

Objective Before Next Meeting: I hope to check that all the functionality of the mobile client is working then add the twitter and facebook sharing functionality

Code Restructuring Done & Start Social Posting Implementation

Created: 04/07/2013
Tags: 2nd Iteration, 2nd Sprint

Accomplished: I was able to completely restructure the code in a way that each loaded page has its own dedicated listeners and functions. It makes the code far more organized and scalable.

I also studied how to integrate Facebook & Twitter sharing functionality to the Mobile client which seems to be perfectly possible and quick to do.

I just have to see how easy it is to implement

Obstacles: I had some problems displaying a received image after it was sent from a user because the existing code did not suite pop ups. I had to create separate page instead of the original nested pages for it to work.

Objective Before Next Meeting: Implement The Facebook and Twitter Image sharing feature to the mobile client.

Social Sharing Implementation

Created: 07/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: I was able to implement social sharing on facebook in a way that already conforms with the existing code. Now users can share content to their facebook timelines using ibaana but the image that will be shared has to be online which doesn't conform with the overall architecture as the
interaction between users must remain private and so does the image they process until they want to share it with their friends.

**Obstacles:** I had a lot of problems first with the Facebook documentation as there documentation was all over the place and not as detailed at it was supposed to be due to their marketing of their SDK platform. I API didn't work

For now will just have to implement only the Facebook sharing and postpone the Twitter sharing feature to the next sprint as I have to also deploy the application online for evaluation.

**Objective For Next Meeting:** Implement Facebook photo uploading feature to the existing Facebook sharing feature and also start with the implementation of text annotations and complex image manipulation.

---

**Social Feature Postponed till the next sprint**

**Created:** 08/07/2013

**Author:** Mohammed

**Tags:** 2nd Iteration, 2nd Sprint

**Accomplished:** I was able to read more about how to implement the social features on the mobile client and gather materials I will use in the future. But due to the time constraint I have decided to postpone its implementation as having more image manipulation features for the evaluation is better.

I also decided on what complex image manipulation features to add and I read on how to add the features. The features are:

1. 2 Image Animation
2. 2 Frame Animation then Text
3. Morphing

**Obstacles:** As before I had a lot of problems with the documentation of Twitter and Facebook but I have done enough learning and research and gotten books to hopefully implement the feature in the next sprint.

**Objective For Next Meeting:** Implement text annotation features and adding the PhoneGap picture API to the application.
Accomplished: I was able to implement the text annotation feature where users can add text to an image it applies the text to the top left corner of the image. I definitely will allow users to choose to choose which location they want to apply a text on image in the future.

I was able to do that by introducing another field to the mobile client, routing server and Image processing server specifying the text to be applied. Specifying where it should be applied will require more computation.

Obstacles: I had problems implementing the phonegap feature that allows people allows users select an image from photo library. I had to skip it in order to implement annotation feature i will implement it in this sprint if time permits it.

Objective For Next Meeting: Implement 2 Frame Animation Feature.

[Optional] specify which position to place text on an image

2 Frame Animation Started

Accomplished: I have started the feature for the 2 frame animation of the images. I first tested using a java class to see if image magick would be indeed able to animate two frames of images, which it did. Then i attempted to apply the feature to the whole architecture.

I want to be on schedule with my project plan especially since there is only 35 days to the end of the project therefore i might postpone the other complex image manipulation to the next sprint in other to host the project online and perform evaluations.

Obstacles: I had a lot of problems with extending the web server to send another image to the IP server interface.

Objective Before Next Meeting: Complete the 2 frame Animation feature and start the online hosting process.
2 Frame Animation Finalized

Created: 11/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: I was able to finish the 2 frame animation of images and it also animated on the mobile application i also tried to make sure that the application.

Obstacles: I had a lot of problem implementing the final 2 frame animation to the mobile client with the socket connections but i was able to finally

Objective For Next Meeting: Research on Cloud Hosting.

Cloud Hosting Research

Created: 13/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: I was able to perform major research at a few options for deploying the application. I found that most of the information is shrouded in marketing but i will be focusing on either Force.com and Microsofts Azure.

Obstacles: Hardly any information on how to host application

Objective Next Meeting: Try to Start the hosting of the vysper server application.

Cloud Hosting's Is A Problem(I Knew it was too good to be true)

Created: 14/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint
Accomplished: I was able to register force.com and heroku.com and but haven’t been able to deploy the application on the cloud.

Obstacles: I just found out Sales force.com use the apex programming language which they create themselves. And although i followed the instructions on Windows Azure it doesn’t work which i assume is the problem microsoft as they mostly only really want ASP.NET Developers

Objective Next Meeting: Find A better Hosting service to deploy the Instant Messaging server

Re-Attempting Clouding Hosting

Created: 17/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: None

Obstacles: I retried Hosting the IM server on the cloud but had tons of problems with the azure plug-in as i was getting errors that are not supported by Microsoft. I also had problems with the Maven build's compatibility but i was able to solve the problem with.

Objective Next Meeting: Keep trying to host the applications on Microsoft's azure.

No Option but to keep trying

Created: 18/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: None

Obstacles: Still been unable to host the j2se application on heroku as for the whole day i looked for solutions to hosting the desktop applications on azure with no success.

Objective of Next Meeting: Either host the application

Switching to Amazon Hosting Service
Accomplished: I have decided to switch to the amazon AWS hosting service as I got a recommendation from my supervisor Mr. Talal and I have done further research on it. It makes perfect sense to try it.

Obstacles: None, they had really good documentation. I am just hoping it actually works.

Objective of Next Meeting: To Deploy the IM Server on an amazon server instance

---

**Successfully hosted IM Server**

Created: 21/07/2013

Author: Mohammed

Tags: 2nd Iteration, 2nd Sprint

Accomplished: I was able to successfully deploy and run the IM app server on the amazon instance and test it by connecting the local application client to it.

Obstacles: I had to take some time to get the right maven configuration for all dependencies and files to be packaged with the jar file. I also had to set the security rules for things for it to be accessed by the other components.

Objective of Next Meeting: Deploy other components on the cloud.

---

**Failed to Host Routing Server Web App on Heroku**

Created: 22/07/2013

Author: Mohammed

Tags: 2nd Iteration, 2nd Sprint
Accomplished: I had the impression that deploying the routing server on the Heroku platform would be the right thing to do since Heroku support J2EE web applications and i could get a Mongo db add on which other components could connect to. It didn't work out.

I will now try to have each component on a server instance on amazon as this makes more sense and is more oriented with the original architecture of the system

Obstacles: It didn't work out because to deploy a j2ee application on heroku the project first need to be a maven project which meant i had to change the type of project i was working on. I also had to embed the tomcat applications server in the project and run the application from a java class in the end it didn't work because i got a lot of errors when trying to run it as a normal web application. it didn’t also work when uploaded the files as instructed

Objective of Next Meeting: Deploy Routing Server on Amazon's Beans Stalk.

---

Able To Deploy The IM Server But Not Run IT

Created: 20/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: I was able to deploy the IM server to the amazon instance by building a jar file and uploading it the instance, now all i have to do is run the jar file.

Obstacles: I had problems running the jar file as the dependencies where not loaded with the jar file and missing files error

r kept getting tripped up. I have read the maven descriptor instructions and will try to make sure that all the files need to run the server as a standalone application are included

Objective For Next Meeting: Deploy, Run and Verify the IM Server on the remote machine.

---

Routing Server Successfully Hosted & Project Plan Modification

Created: 23/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint
Accomplished: I was able to successfully upload and run the amazons elastic bean and also access over remotely.

I have decided to modify the project plan a bit for me to be able to complete this project with a quality product. Due to how long it has taken to host the components of the application i will not be able to perform an evaluations of the 2nd Sprint but i chose show my supervisor, Mr Talal, the prototype and he gave me some suggestions on how to improve it. Hopefully i will be able to perform an in-depth evaluation after the 3rd Sprint.

I have also decided to work on the documentation in the weekends as i have start the 3rd Sprint as soon as possible as also make up for the 2nd Sprint's missing documentations.

Obstacles: I had some problems hosting the routing server's web applications with the eclipse's amazon plugin but i was able to host the application by using the console to upload the war file archive.

Objective For Next Meeting: Host and export the database to a mongodb database server.

---

Database Server Successfully Setup and Transferred

Created: 24/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: I was able to set up the database server and transfer both the entire database to the remote server. I also begun the process of hosting of the Image Processing Server by creating a new server instance on amazon but this time it is a windows server.

I am very happy with how well it works as now the mongodb instance start on the instance setup making it almost dedicate to the database functions.

Obstacles: The biggest problem i had was transferring the database to the remote system, some methods required the communication between the two database system while some required . i tried different methods until i got the actual one that worked.
Objective For Next Meeting: Configure the Image Processing Server and ensure test the overall system.

Image Processing Server Setup Successfully

Created: 25/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint

Accomplished: I was able to successfully create and configure the instance for the Image Processing server and also register an account for an ftp service.

The image server's operating system is windows server operating system this was done to make sure that everything would be identical with the local system. All required programs like JDK and Image Magick were also installed.

I also registered for a free ftp service, i am confident that it would serve the purpose of the application. If it doesn't, there are other ftp service that offer both ftp and http protocols along with the service.

The amazon service provide a great way for instances to communicate between each other by providing a private ip for instance to identify and send data to each other. Without the need to communicate over the Internet. That will make sure that the application performs faster as whole.

Obstacles: All components couldn't communicate well because the IP addresses of each instance change from when it is stopped or started. I will have to find a way to statically assign each instance for consistency.

Objective For Next Meeting: Round out this sprint by making sure all components communicate well with each other and testing the mobile client on the blackberry simulator

IP Problems

Created: 26/07/2013
Author: Mohammed
Tags: 2nd Iteration, 2nd Sprint
Accomplished: Learned a bit more about how amazon networks work after reading how IP addresses and DNS names are administered by amazon. And i have found out a way to at least keep the external ip constant which is good news, as the mobile client source code doesn’t have to be changed every time the servers are restarted.

Obstacles: I had IP problems that made yesterday more or less non-productive because after i restarted an instance both the internal and the external IP address changed. I previously thought that only the external IP addresses changed after the instances where restarted.

Objective For Next Meeting: Verify that the history and contact list features are working.

---

**Login successful**

Created: 25/07/2013

Author: Mohammed

Tags: 2nd Iteration, 2nd Sprint

Accomplished: After most of the servers where successfully setup, it was time to at least find out if users could log in to the remote IM server. Login was successful as the IM server was able to connect & authenticate with the database server.

Obstacles: I had to assign the internal IP address of the database server to the IM Server for authentication and upload it to the server which took a lot of time because of the size of the IM Server which was around 40MB.

Objective For Next Meeting: Include the the routing server in remote architecture and verify that the history and contact list features are working.

---

**Remote Routing Server Problem**

Created: 27/07/2013

Author: Mohammed

Tags: 2nd Iteration, 2nd Sprint

Accomplished: Able to make the mobile client communicate with the remote routing server.

Obstacles: The remote routing server has not been able to connect to the database server which hasn’t allowed it to display contact or history information.
**Objective For Next Meeting:** Solve the problem between the remote routing server and the remote database server.

---

**Remote Routing problem fixed & Documentation Continued**

**Created:** 28/07/2013 02:29 PM  
**Author:** Mohammed  
**Tags:** 2nd Iteration, 2nd Sprint

**Accomplished:** I was able to solve the problem with the routing server it turned out that the previous version where the internal IP address of the database server was not assign is what was running on the routing server instance. I have now started working on the documentation, although i won’t start with writing about the 2nd iteration yet. I will only be improve what is already present in documentation until next weekend when i will write about the 2nd Iteration.

**Obstacles:** took some time before i could pin point the exact cause of the problem, but i was able to do that by getting the snapshot log of the apache tomcat application from amazons beanstalk.

**Objective For Next Meeting:** Improve the previously written documentation and ensure the remote image processing server and remote File server are work with the rest of the remote servers.

---

**Project Finally Completely Hosted[End of 2nd Sprint]**

**Created:** 29/07/2013  
**Author:** Mohammed  
**Tags:** 2nd Iteration, 2nd Sprint

**Accomplished:** I was able to fully integrate the remote image processing server and the File server to the existing remote architecture and now all the server components can work remotely.

**Obstacles:** I had a string of problems before i could make the remote servers work properly. First i had a problem with the File Server the FTP service i wanted to use as the file server was good but couldn't pro-grammatically provide the url for uploaded image and every url only lasts for 20 min. I had to overcome that by adding another amazon instance to server as the file server and installing xamp server and configuring it properly.

After i uploaded the image processing server interface and configure it properly, there were problems in communication between the File server and Image processing server due to firewall rules
and how FTP transfer work with the firewall rules. I was able to set the right firewall rule to allow smooth communication between the two components.

**Objective For Next Meeting:** Research and decide what to do for the next sprint and how it should be done.

---

**Start of 3rd and Final Sprint**

**Created:** 29/07/2013  
**Author:** Mohammed  
**Tags:** 3rd Iteration, 3rd Sprint

**Accomplished:** I have been to decide what to do for next 2 weeks of the sprint that will finalize the project. Due to time constraints i only chose 3 goals that i feel will make the project a presentable project. I added Sms to Internet & Popular Images and also added a better ui design goal based on the feedback of my supervisor.

**Sprint Goals**

1. Sms to Internet  
2. Better Design  
3. Popular Image

**Obstacles:** I had problems selecting a good sms gateway service thankfully i called my supervisor who gave me an excellent option.

**Objective For Next Meeting:** Start the Implementation of the sms function

---

**Trying to Figure Out How it will works**

**Created:** 31/07/2013  
**Author:** Mohammed  
**Tags:** 3rd Iteration, 3rd Sprint
Accomplished: I was able to study and understand and how the intermediary sms service (nexmo) is going to fit into the overall architecture of the application and how to implement it on the server side. I also have a rough idea on how to implement it on the client side apparently there hast to be a form that sends sms in what looks like a GET URL for the server component.

Obstacles: Although i understand how the application will work, i still dont completely know it will work on the client side and how i will be able to use it to login users on the server side.

Objective For Next Meeting: Find out how to implement SMS Server and try to start its implementation

Start of Implementation

Created: 01/08/2013
Author: Mohammed
Tags: 3rd Iteration, 3rd Sprint

Accomplished: I have started the implementation of the sms feature by adding the required library and studying more of the communications. Earlier i had a dilemma on how the mobile client would login to the IM server and chat with other users without internet connections. I figured out that i could create another IM server with an sms endpoint to log-in users and also the 2 IM server would be connected together and the users can be connected with the use of the server-to-serve connections. I also found out how to log-in users manually.

Obstacles: I am not still sure of how to send sms from the mobile client to the IM server to communicate with the very little documentation provided by nexmo.

Objective For Next Meeting: Implement sms to internet feature.

1 Step towards SMS Communication

Created: 02/08/2013
Author: Mohammed
Tags: 3rd Iteration, 3rd Sprint

Accomplished: I was able to change the architecture of the application to suite the sms function. I found out that i do not need to add a new independent server, it’s possible to add the sms feature into
the existing server. I also purchased a number for the sms server i am to implement. I also found out
that i could login in users manually after listening for any received sms message.

One conundrum i had was finding out how to send and receive text messages exclusive with the
mobile it turns out that that has to be implemented using the mobile platform specific code.

**Obstacles:** Documentation for implementing the sms feature is quite inadequate and
incomprehensive leaving me having to do search for it all over the internet.

**Objective For Next Meeting:** Implement the Login Feature Through Sms

---

**Slowly But Surely**

**Created:** 05/08/2013

**Author:** Mohammed

**Tags:** 3rd Iteration, 3rd Sprint

**Accomplished:** I have been able to integrate a jetty j2ee web server into the Instant Messaging server
for http requests to be sent to the IM Server. After studying the nexmo technology and going further
research about the call back url i was able to understand how to the workflow of the nexmo service is.

The service hosts a paid number that sms messages can be sent to, when an sms is sent to that number,
the messages if forwarded to your registered web applications using http to send a response to the
callback url.

I was able to embed a web server in the IM Messenger and created a servlet to handle with the http
requests that will be sent by the nexmo service.

**Obstacles:** I had to perform a detailed research to fully understand nexmo service and how it will fit
in to the overral architecture of the application. I also had some problems embedding the server as the
bosh component was already making use of the bosh component. I also found out that due to the
nexmo service i might have to host my application online to be able to test it.

**Objective For Next Meeting:** Implement the Login feature in the IM Server.

---

**Server Side Processing Started**

**Created:** 06/08/2013

**Author:** Mohammed
Accomplished: I was able to figure out a way of testing feature that will be accepting messages from the nexmo service. I found out the way in-which the nexmo service will be sending messages to the web application and decided to use it to send request to the embedded server. After making sure the servlet was able to receive the username and password, I tried to implement the Login Feature

Obstacles: After numerous attempts i couldn't login the user apparently because of tls authentication issue.

Objective For Next Meeting: Implement the Login and sending of message feature to the IM server

---

**SMS Connection Login and Sending Feature Successful**

**Created:** 07/08/2013

**Author:** Mohammed

**Tags:** 3rd Iteration, 3rd Sprint

Accomplished: I was able to successfully implement the login and and sending of the server, of course the sms feature hasn't fully been implemented. Right now i am still working on how the IM Server Component and how it will interact with the SMS Service. I was able to successfully login a user after an sms of the authentication details will be sent to the nexmo service then to the IM Server. I was also able to simulate the sending of a message from the mobile client to another user by sms.

Obstacles: I had some problems with using the smack library as the library i am using is a bit out of date and if i try to update it, it might affect the other components. I had to search online documentation and examples on how to use the smack library as an Xmpp client.

Objective For Next Meeting: Implement and Test the Mobile Client and IM Server SMS Features to ensure internet and sms communication.

---

**Sms Server Side Implementation Done**

**Created:** 08/08/2013

**Author:** Mohammed

**Tags:** 3rd Iteration, 3rd Sprint
Accomplished: I was able to fully implement the server side of sms feature now the Instant Messenger server can relay messages sent to a user via sms. The user can also receive messages if they have been sent via nexmo server (Pseudo Implementation). I was able to mimic how the nexmo service would relay the message sent by sms to a certain user and how a reply would be sent. I sent replies to mobile phone the message was receive from "SMSINFO" sms server.

Obstacles: I had some problems implementing the Message Listener that would get a message and relay it to another user. I was later able to solve it by carefully studying the smack documentation. I also had some versioning problems with the smack library.

Objective For Next Meeting: Implement the Sms feature on the Mobile Client.

---

**Sms Client Pseudo Implementation Complete [End of 3rd Sprint]**

**Created:** 09/08/2013  
**Author:** Mohammed  
**Tags:** 3rd Iteration, 3rd Sprint

Accomplished: I was able to complete the code for sending and listening to sms on the mobile client application. Although i have not been able to test it, I will Hopefully test It on the 12th of August when i will try to test the feature when testing it on a physical device for evaluation.

Obstacles: I couldn't find a way to simulate the sms messaging with neither the simulator nor emulator.

Whats Next=> Today and tomorrow(9th & 10th of August) I'll try to finish the documentation on the Iteration. On Sunday(11th), I'll try to plan for the evaluation and test the mobile client on a physical device. On Monday(12th), I'll try to perform the planned evaluation with users and on the 13th & 14th I will try to finalize the dissertation report for submission on the 15th.

5. **Evaluation Result Materials (Only Available in Hard Copies)**

See next page. Please erase proceeding pages before making it publicly available to protect the information of the individuals.