Context Aware Instant Messenger with Integrated Scheduling Planner

by

MUHAMAD AKIRI BIN MD NAWI

11230

ICT

Dissertation submitted in partial fulfilment of
the requirements for the
Bachelor of Technology (Hons)
(Information & Communication Technology)

SEPTEMBER 2011

Universiti Teknologi PETRONAS
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

Context Aware Instant Messenger with Integrated Scheduling Planner

by

Muhamad Akiri bin Md Nawi

A project dissertation submitted to the

Information and Communication Technology Programme

Universiti Teknologi PETRONAS

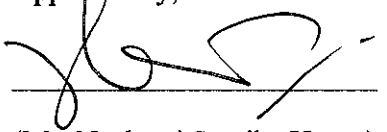
in partial fulfilment of the requirement for the

BACHELOR OF TECHNOLOGY (Hons)

(INFORMATION AND COMMUNICATION TECHNOLOGY)

September 2011

Approved by,



(Ms. Nazleeni Samiha Haron)

UNIVERSITI TEKNOLOGI PETRONAS

TRONOH, PERAK

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

MUHAMAD AKIRI BIN MD NAWI

Abstract

The purpose of this paper is to do a research on context awareness concept in instant messaging applications. The widely used instant messaging systems usually incorporate the same context awareness function that displays the current status of other users. There are many prototypes of context aware instant messenger program that already being developed and being tested in order to ensure the applications are able to display the users' status in a more reliable and accurate manner. However, the availability indicators provided by instant messenger only useful for users who are planning to have communication with other users. This kind of context aware does not help users to use the instant messaging systems for another purpose such as to schedule an appointment with one of their contact list.

By realizing this issue, the design of a Context-Aware Instant Messenger with Integrated Scheduling Planner system is proposed to cater the problems. The solution application is designed for all computer users especially for the staffs in organizations and universities students which will enable them to do any appointment through instant messaging system. Throughout this document, the existing prototype of instant messaging systems is discussed regarding their context awareness concept and at the same time is compared to identify their strong and weaknesses.

From the finding of the research, new project plan is discussed in order to develop a new brand prototype of instant messaging application starting from planning, analysis, design, prototype development until prototype system testing. Prototyping methodology is chosen for the development of this application because of the time constraint. There are three prototypes with different level of functionality has been produced by this project to exhibit the scheduling planner function.

Based on the research done, the Context-Aware Instant Messenger with Integrated Scheduling Planner system was found to transcend current instant messaging products by having new feature which enable the users to schedule any event with their contact list.

Acknowledgement

First and foremost, praise to ALLAH s.w.t for His blessings and mercy for giving me an excellent health in completing this project. I would like to express my gratitude to many parties.

I would like to express the deepest appreciation to my supervisor, Ms. Nazleeni Samiha binti Haron for her invaluable advice and ideas, patience guidance and constant supervision throughout every stage of my final year project research, for being understanding regarding on the problems face upon completion of this project. I gratefully acknowledge for the opportunity to be supervised by her.

I am also highly thankful to my external examiner, Dr Hon Hock Woon, Senior Researcher at MIMOS Berhad and my second examiner which is Dr. Rohiza Ahmad for giving guidance regarding the content of this report and suggestions to improve this project, I gratefully acknowledge the detailed work on assessment, evaluation and format for this project.

I would like to thank to the 53 students of Universiti Teknologi PETRONAS who helped in answering the questionnaire survey and 32 more students who took part in the usability testing for the developed system.

An utmost appreciation is extended to my family and my friends for their encouragement, concern and support to me in completing this thesis. My warmest thanks to all individuals that have contributed ideas, comments, criticized and for having made my time in completing this project so enjoyable, thank you so much.

Last but not least, I would like to thank Universiti Teknologi PETRONAS, especially Department of Computer & Information System for providing the facilities and wonderful education.

Table of Contents

List of Figures

Pages

List of Tables **Pages**

Abbreviations

IM – Instant Messenger / Instant Messaging
UI – User Interface

CHAPTER 1
INTRODUCTION

1.1 Background

Nowadays, internet-based communication tool are widely used by people around the world as it eases them to communicate with other people without any cost at any places as long as there is internet connection. Internet-based communication tools refer to forums, chats, voice over Internet Protocols (VoIPs), emails and websites. E-mail, instant messenger (IM) and VoIP normally used by staffs within an organization to communicate with each other. Usually, these people use e-mail application for example MS Outlook, to deliver important information as well as to schedule a meeting or discussion with other co-workers or subordinates. Meanwhile, instant messenger programs designed for organizational usage (i.e. MS Office Communicator) is preferably being used for a simple chat with other co-workers as an alternative to VoIP. On the other hand, free instant messenger programs such as Yahoo Messenger, Google Talk, Skype etc are used to communicate with friend or buddies. It can be said that most of computer users having this application installed in their computers, because of the high level of usefulness of this program.

Instant messaging system provides availability indicators (online/offline and active/idle) for every buddy in the contact list, which is probably the main feature that makes it becomes the premier medium for people to communicate with their buddies, besides it is free of charge. This availability indicators feature lists IM as one of the context awareness applications. Context aware applications mean that the applications use context information (location, temperature, date and time, lighting, noise level, etc.) and react to it, by adapting their behavior according to changes in context. Context aware IM refers to presence awareness context that it provides, by mentioning whether the other

users are available or not for communication. This kind of context really helps the users in determining whether they can communicate with the intended users or not at that time.

Nevertheless, the availability indicators only useful for users who are planning to have communication with other users. How about if the user wants to use the instant messaging systems for another purpose, for example, to schedule an appointment with one of his/her contact list, without interrupting the person if he/she is not available for chatting or currently is offline. Maybe that user can send the other user an offline message to inform about the appointment, if the person is currently offline, but it is not an effective way to ensure that the person are fully alert with the appointment in case he/she deletes the message after reading it. The presence awareness context still could not handle this issue.

1.2 Problem Statement

Computer users usually prefer to contact other users through IM compared to email or SMS because of its simplicity. The current popular IM applications only implement basic context-awareness function by displaying the network status of the other users, i.e. whether they are online or offline. If a user has something urgent to deal with another user, for example to schedule a meeting or to have an online discussion later on, the person cannot reach and inform the other person on the spot if he or she is offline. Some people might think, this is not a big deal since the person that needs to be reached can be contacted through email, phone call or SMS. But the main concern here is to make use instant messaging program as the only communication channel for that purpose instead of using e-mail to schedule the meeting.

Let say, a user want to schedule an appointment with one of his/her contact list, without interrupting the person if he/she is not available for chatting or currently is offline. How the users can ensure that the intended person will be alerted with the scheduled appointment? Given the scenario where the information about the scheduled

appointment should not being informed through chat message as it could be annoying to the intended buddy as he/she is displaying "busy" status at the time. If the buddy is currently offline, how to ensure that he/she notices about the scheduled appointment, without need to read offline message or do sticky note to ensure he/she does not forget about that when he/she read it from offline message? This kind of situation could not be settled only by the presence awareness context implemented by contemporary IM programs.

1.3 Project Objectives

The implementation of this project is to overcome the stated situation in the problem statement. The project is about to study the concept of context-awareness in one of the internet-based communication tools; the instant messaging application followed by the development of a context-aware instant messenger with scheduling planner. The main objective is to add a new context-aware feature to the existing instant messenger programs; the scheduling planner which can schedule meeting, discussion and so on between the users based on user inputs.

Significance of the project

i. To reduce unsuccessful communication attempts by updating the availability of the users automatically.
ii. To enhance the efficiency of communication via instant messenger.

1.4 Scope of Study

This project aims to research about context-aware concept in instant messaging application and then come out with one prototype to implement the concept. The context-aware concept that will be implemented in the instant messaging application is the scheduling planner function, which will be able to schedule any meeting, discussion, appointment or whatever event planned by the assignor to the assignee (an IM user to

his/her buddies/contact list). This application is targeted to all computer users especially the staffs in organizations and universities students who are among the dominance in IM usage around the globe. This application will be developed in mobile platform using Android OS.

CHAPTER 2
LITERATURE REVIEW

2.1 Importance of Instant Messenger

Instant messaging (IM) has proved its usefulness: its popularity is growing fast, and it has been introduced into corporate use as well [1]. Several researches have been done regarding the use of IM in business world and the results are the followings:

- 90% of businesses will use IM by 2004. (*Gartner IM Trends*)
- Corporate IM is expected to replace 65% of e-mail usage by 2004. (*Information Week*)
- 65 million workers are already using instant messaging, and that number is expected to grow to 350 million by 2005. (*IDC Research*)
- Corporate IM usage is expected to account for nearly 60% of all online traffic by 2005.(*Ferris Research*) [14]

According to the research firm, Gartner, as recently as the first financial quarter of 2002, few businesses understood the importance of instant messaging as it relates to the enterprise. IM was thought of as a vehicle for social interaction and many businesses frowned upon its use, simply because it was seen as a threat to worker productivity [14]. However, its usage in the corporate organization will increase dynamically based on the research done, most probably because of its special functionality.

End users, however, saw the benefit of being able to communicate in real time with their co-workers and began what industry experts call a "grass roots" effort to bring instant messaging into the corporate fold. They found the presence awareness in IM regarding their contact lists status is very useful [14]. Instant messaging provides a better communications channel than what is provided by the telephone and e-mail. Because

employees could see who is online and who is not, they could ask a quick question of someone who is available and not waste time waiting for a telephone call to be returned or an e-mail to be answered. They pointed out that instant messaging is a cost-effective alternative to travel, Web conferencing could met their communication needs just as nicely as a business trip and chatting with a co-worker in another country using IM is much less expensive than telephoning them.

Among the most important factors behind IM's quick conquest of the business world is geography. Within many companies today, workers are spread across the globe and are much more mobile. Finding someone often requires more than just making a phone call or sending an e-mail message. With instant messaging, a correspondent knows who is available, and who is not, at any moment [15]. As more IT departments become convinced of the value of IM as a business communications tool, many companies nowadays tend to make use the technology as an added value to the available communication channels in their companies.

2.2 Context Awareness in Instant Messaging Systems

IM has been described as "near-synchronous computer-based one-on-one communication" [2]. It is said to consist of two components: synchronicity and presence awareness. Synchronicity refers to real-time information transfer [3]. Presence is defined as the ability and willingness of a user to communicate with other users. This meaning has lately been designated as availability [4, 5]. Presence-awareness means that presence information is used to decide whether communication with a user can be initiated. This information can be used by the user about to initiate communication, or the IM application can deliver instant messages only to users with appropriate presence status [1].

Presence awareness is part of context-awareness. Term context-awareness is widely used in computer science field to summarize the awareness functionality of systems or applications. According to the Encarta World English Dictionary, context is defined as

"the circumstances or events that form the environment within which something exists or takes place" [6]. Chen and Kotz (2000) redefines context as "the set of environmental states and settings that either determines an application's behavior or in which an application event occurs and is interesting to the user" [7]. From the stated definitions, a context-aware application can be defined as an application which knows something about the current context it is in, and act upon it [8].

There are a lot of context-aware instant messaging applications available, ranging from the most-popular being used, for organizational usage as well as the one designed as a prototype for research papers. Basically, all of these IMs only incorporate a must have context-aware concept on it, which is presence awareness. IMs built specifically for research papers usually have more context-aware functionality since they were designated to test certain researches on context awareness concept.

2.3 Context-aware Instant Messengers

BuddySpaceLive is an extension of the BuddySpace open source application. BuddySpace is an instant messaging (IM) client that enables location-based interactions by associating location information to each of the interacting users. Besides the usual interface of an IM application, BuddySpace allows the user to inform his current position within a map that had been previously published in a map repository. The user's location is shared among all users that are in his buddy list (i.e. is shown in their maps). BuddySpace interacts with other IM applications through a Jabber server using the XML-based Jabber open protocol, which provides portability with most common IM platforms, such as MSN Messenger, Yahoo Messenger and ICQ [9].

In BuddySpaceLive (BSL), on the other hand, the user's location is obtained from MoCA infrastructure (Mobile Collaboration Architecture, a middleware for developing collaborative applications for mobile users), which publishes this location as context information through its LIS service [9]. A LiveMapServer interacts between LIS service and each BuddySpaceLive client to send notifications when a client location changes

[10]. Using SRM, BSL gets symbolic location of a user, instead of just the physical location, as provided by BuddySpace [9].

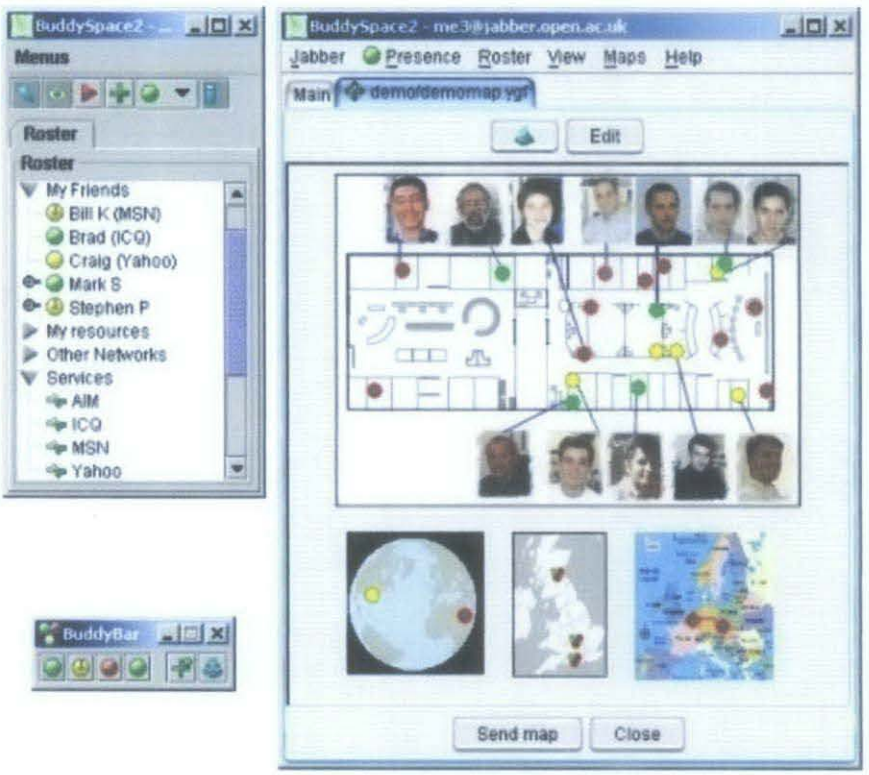


Figure 2.1 BuddySpace application's interface

BuddySpace is built for the aim to provide enhanced capabilities for users to manage and visualise the presence of colleagues and friends in collaborative working, gaming, messaging, and other contexts. Meanwhile BuddySpaceLive tends to enhance the way of how BuddySpace shows the presence of buddies, in a better manner using symbolic location. Both of these applications concern more on the presence awareness context, rather than any context-aware concept. Most probably because this is the feature that differentiates it from other communication means such as email or telephone [13].

Other IM systems which are developed by Sun Microsystems also give concern on presence-awareness concept. The prototypes named ConNexus and Awarenex, respectively designed for one main purpose, but for usage in two different platforms. Both of these applications are created to facilitate remote communication by integrating

communication tools and awareness capabilities into the workplace context. ConNexus prototype integrates awareness information, instant messaging, and other communication channels in an interface that runs on a desktop computer. The Awarenex prototype extends that functionality to wireless handheld devices, such as a Palm [11]. Three design goals are identified for creation of ConNexus which are:

- Provide awareness cues to help people find opportune times to initiate contact
- Integrate multiple communication tools to facilitate transitions from one to another
- More naturally support the process of starting, maintaining, and ending communication

The Awarenex prototype was developed by taking into account the mobility in collaborative technology. This prototype focused on handhelds devices (wireless Palms, RIM Blackberry) and also integrated telephony into the system. Both ConNexus and Awarenex aim to enable mobile users who were away from their desktop computer to maintain awareness of their colleagues, and to be able to maintain awareness of mobile users when they left their desktop [11].

The images below shows the Awarenex program displayed on a desktop computer, Palm Pilot, and RIM Blackberry device. Activity icons show if the user is currently engaged in any communication activities [12].

Figure 2.2 Awarenex's GUI on Desktop PC

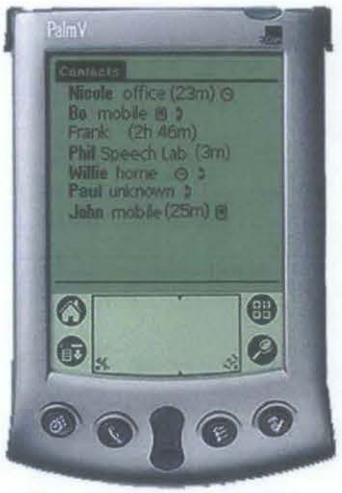


Figure 2.3 Awarenex's GUI on Palm Pilot

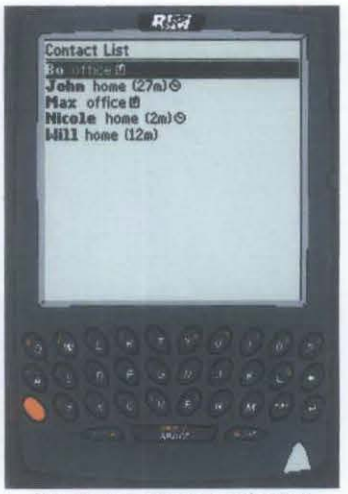


Figure 2.4 Awarenex's GUI on RIM Blackberry

For example, the icon with the lightning bolt indicates a user is actively engaged in an instant message conversation. A telephone icon indicates that the person is currently talking on the telephone. A clock icon denotes a current calendar appointment. If a user is not actively using any device, an indication of how long since the most recent activity is given in parentheses (e.g. computer idle time or when a cell phone was last used) [12].

The main features of ConNexus and Awarenex is the clear statement on the application's interface about the current activity performed by the buddies. This detailed context awareness concept helps other users a lot in determining whether or not to initiate interaction to their contact list. It is being noticed that these instant messaging programs do not implementing the usual presence awareness design, which is presented by "available", "busy" or "away" status as done by other existing software of their kind. Both of these IM applications manage to escape from the standard design and come out with more useful signage rather than just green, red and orange status, which is proven more efficient and reliable. It would be better if these programs also include the usual colour-based status but with advanced and little modification for example if a user is currently talking on the telephone or having a meeting based on the calendar information, the colour status will automatically change to red to indicate he/she is not available for the moment.

Xiaolei Zhang et al. pointed out that there are three limitations exist in current presence services: limited vocabulary, manual update, and unitary distribution. They proposed Smart instant Messenger (SIM) in their IM research project. At the general level, they enrich the vocabulary of presence using context information, including the user's activity, location, device capability etc. The other level encapsulates the user's preferences and willingness towards various communications, which is represented by a set of user-defined rules [13]. Overall, SIM attempts to expand the potential of instant messaging by suggesting new means of communication for the pervasive computing environment. SIM highlighted three main issue in its design; context-aware presence management, activity-based grouping and vicinity-based grouping, and user-centric resource configuration.

Context-aware presence management deals with the ability of the system to provide availability of IM users. A user's willingness to be contacted is either manually set by the user, or inferred by the system according to some predefined rules. The reasoning of the latter is based on the actual situation of the user and the user's relationship with the buddy. Therefore, the sharing of presence is adaptive and different buddies may see different statuses at the same instant. The process considers the user's preferences and exercises fine control over how a user's availability is distributed [13]. Following is the examples of pre-specified rules:

(1) Rule 1: If I'm in a meeting, set my presence as ''DND'' (Do Not Disturb);

(2) Rule 2: If I'm in a meeting, set my presence to my group members as ''Available'';

(3) Rule 3: If I'm in a meeting, set my presence to my supervisor as ''Available''.

When people collaborate in a task, they need a means for quick questions and timely responses. They also would like related electronic resources to be ready for sharing. Consider the situation of a clinical discussion about a certain patient. If a doctor can see in his contact list the other doctors (remote or co-located), nurses, electronic patient records, projector, the real-time monitoring sensors of the patient, and so on as a group, the efficiency of communication and accuracy of diagnosis will both be improved [13]. This scenario has brought Xiaolei Zhang et al. to implement activity-based grouping and vicinity-based grouping feature in the SIM.

User-centric resource configuration feature told that the system design should be directed to support the users and their tasks as the top priority. The system needs to discover and select a suitable resource needed by users for certain tasks and establish the communication with the resource. For example, when a user looks for a photo editing application, the icon of a suitable application appears on the contact list. The result is so because it has been the user's habit to use only these two operations for photo-editing. When the user invokes the service, the requested operations are wrapped in messages and directed to the resource [13]. SIM basically focuses on the presence awareness functionality based on the feature of context-aware presence management that it

highlighted. Meanwhile activity-based grouping and vicinity-based grouping and user-centric resource configuration are additional functions to differentiate itself from the available research projects which concerning on the presence awareness concept.

Lilsys, a prototype system developed by Begole et al, uses passive detection of unavailability using ambient sensors, while preserving privacy by abstracting the person's context details into an unavailability inference [16]. Begole et al. state that there is a need for mechanisms to help understand not only whether an intended recipient is reachable but also how receptive she is to being called [16]. Lilsys includes a motion sensor which is combined with speech and device activity to detect a person's presence [13]. Lilsys infers user's availability from sensor data, such as sound, phone usage, and computer activity [18]. Using inferred availability, the system gives cues to other users as 'neutral', 'possibly unavailable', or 'probably unavailable' [1].

ConChat is a context-aware chat program that enriches electronic communication by providing contextual information and resolving potential semantic conflicts between users. Using contextual cues, users can infer during a conversation what the other person is doing and what is happening in his or her immediate surroundings. For example, if a user learns that the other person is talking with somebody else or is involved in some urgent activity, he or she knows to expect a slower response. Conversely, if the user learns that the other person is sitting in a meeting directly related to the conversation, he or she then knows to respond more quickly [17].

One problem that plagues all conversations, both face to face and electronic is ensuring that all parties mean the same thing when they say something. Semantic conflicts typically arise because the two parties are in different contexts. For example, when mentioning a time (say, 8:00 p.m.), the user would typically refer to his or her own time zone. However, if the other party were in a different time zone, that person might interpret the time to be in his or her own time zone. Another example is if an American and a Canadian are chatting, and the Canadian says "$10." The Canadian might have meant 10 Canadian dollars, whereas the American might take it as 10 US dollars [17]. In

order to overcome this kind of issue, ConChat has come out with the function of avoiding semantics conflicts. For example, it automatically tags the currency of the sender's country when discussing money [13].

ConChat sends the followings contextual information between users:

1. Party location
2. Number of other people in the room
3. The identities of the other people in the room
4. Room temperature, light, and sound
5. Other applications and devices running in the room
6. User's mood (such as happy, sad, or excited)
7. User's status (such as "on the phone" or "out to lunch")
8. Activity in the room (such as a meeting, lecture, or presentation)

ConChat main functionalities are sharing contextual information and avoiding semantics conflicts. Based on the eight contextual cues from the feature of sharing contextual information, this IM still weigh on the presence and availability function to enhance the effectiveness of the communication. What make it different from the other IM systems is that it added a new function to avoid semantics conflicts, which look simple but really useful for certain users if they are in different countries with their buddies.

The discussed instant messaging systems above is all about the presence awareness, whether to indicates the availability of the users more accurately, to give the information of current locations the users are in or the actual activities that the users are doing. Contrary with the IM programs, this project tries to implement another new concept in context-aware instant messaging program, an added function that can alert IM users about any scheduled event, meeting or discussion proposed by their buddies or persons in their contact list. The upcoming additional feature would be stated as "scheduling awareness". Based on the available research papers and journal articles in the internet, so far, there is no such feature being implemented yet in instant messaging applications.

2.4 Scheduling Planner

Usually, when a staff wants to have a discussion with one of his or her co-worker or a manager plan to have a meeting with his or her subordinates, they will send an invitation for the planned discussion or meeting through email normally using MS Outlook. After the co-worker or the subordinates receive the meeting notification through the email, they are given choices whether to accept or decline the invitation for the scheduled meeting. If they are available during the time of the scheduled meeting and choose to accept it, the meeting details will be integrated into their Outlook calendar. The users then will be prompted to choose a certain time to be notified by the alarm before the meeting start (i.e. 15 minutes before the meeting). When the time for the meeting notification is reached, a popup window comes out to alert the user regarding the meeting which will be 15 minutes from that time. In this case, people depend on the email application to schedule the meeting and also to be notified before the time of the meeting.

There is a context-aware meeting system which is presented as a case study, supports members of a university department in arranging meetings. The system provides a personal calendar that contains the appointments details of each user, despite of having current location and current activity of the system's user. On the contact list of the system, there are information such as the individual's title, the person's department and the office of the employee. In order to arrange the meetings, the personal calendars of several users will be matched to find a free time slot. Because the personal calendar of a user can also contains private appointments, a public calendar of the user should be provided by the system that only discloses the occupied time slots of the person. If a meeting should be arranged, only the public calendars of the system's users need to be matched such that further details about the private appointments are not revealed. Information about a meeting should encompass the meeting's participants, the meeting date, the place of the meeting and a topic description [19].

The meeting system does not only support the arrangement of meetings, but it also encompass the presence awareness function, whereby the current activity of the user will be automatically set to busy if the user is attending a meeting or currently has an appointment and vice versa. The public calendars of the users, which are used to find suitable meeting dates, are derived from the personal calendars of the users and, therefore, do not need to be provided by the users themselves.

As discussed in the introduction chapter, this research is about to provide a scheduling planner which is integrated into IM in order to facilitate the users especially those in the office environment to arrange meeting with their co-workers, without the need to use the email application anymore for that purpose. Since not many works related to this kind of application, this project would be very useful to the stated users. The context-aware meeting system above would be a good guideline to create an IM with scheduling planner for this project. The criteria explained by the context-aware meeting system will be implemented in this IM project. The next chapter will be discussing on the suitable platform for the development of the context-aware instant messaging with integrated scheduling planner.

CHAPTER 3
METHODOLOGY

3.1 Research Methodology

This research was conducted in order to study and understand more about the context awareness concept in instant messaging application. The context aware definition, type of context aware implemented in instant messaging system and its contribution towards the use of the application are clearly observed in order to come out with a new project within the scope. The objective of the research is also to ensure that the chosen topic is appropriate for the development of application later on, which is a new feature named scheduling planner to be incorporated with the IM.

In order to gather all the information that is related to the project, the method of qualitative research is used. There are two main types of research methodology; quantitative methodology and qualitative methodology. Quantitative methodology is the type by which researchers test the significance of the hypothesis, in other words to answer the questions: How much? Is there a relationship? Quantitative methods tend to be systematic and use numbers. However, qualitative methodology is the type by which researchers are depending on the observations and descriptions. It is subjectively and descriptive, no numbering facts. This kind of method is used to assess knowledge, attitudes, behaviours, and opinions of people depending on the research topic. Researcher, in this type of method uses his or her opinion and experience which are not allowed to be used in quantitative method at all. Since the quantitative method does not involve numbers or statistics, it is suitable for this project research.

Qualitative research often categorizes data into patterns as the primary basis for organizing and reporting results. Qualitative researchers typically rely on the following methods for gathering information: participant observation, non-participant observation, field notes, reflexive journals, structured interview, semi-structured interview, unstructured interview, and analysis of documents and materials. For this project, most of the facts and figures are obtained from field notes, journals and analysis of documents and materials. All of them are part of the public domain information belonging to those who did projects and researches about context aware instant messaging application.

Some facts also found outside the research done which is from the author's experience in real-life involving the use of multiple instant messenger programs. Based on this self experience, one or two facts have been learnt regarding the IM and could be used and applied in this project. Apart from that, discussion with project supervisor also being done regarding the requirement of the project and the guidelines that need to be followed and how to find the reliable source for the project. A set of questionnaire about the IM usage also provided and was given to the university students as a survey.

By using qualitative research method, the author should be able to determine and decide which platform is most suitable for this project. There are two choices of platform to be used, which is desktop computing or mobile computing. At the end of the research, the concern is justified whether to build the application for desktop or for mobile computing environment by taking into account the feasibility aspect and the relevancy of the project.

3.2 System Development Methodology

3.2.1 Prototyping Methodology

System development methodology is a form of standardisation or framework that is used to structure, plan, and control the process of software development. The system methodology used in the development of this project is the prototyping methodology. As the time given to finish this project is less than 12 months, prototyping methodology is the most suitable method as an approach to handling selected parts of a larger, more traditional development methodology (i.e. incremental, spiral, or rapid application development (RAD)). At the end of this project, a working prototype of instant messaging application with scheduling planner is produced. The prototype contains several basic functions such as user authentication, adding and approving a new friend, messaging with a friend in list, showing online and offline users and most importantly the schedule planner function.

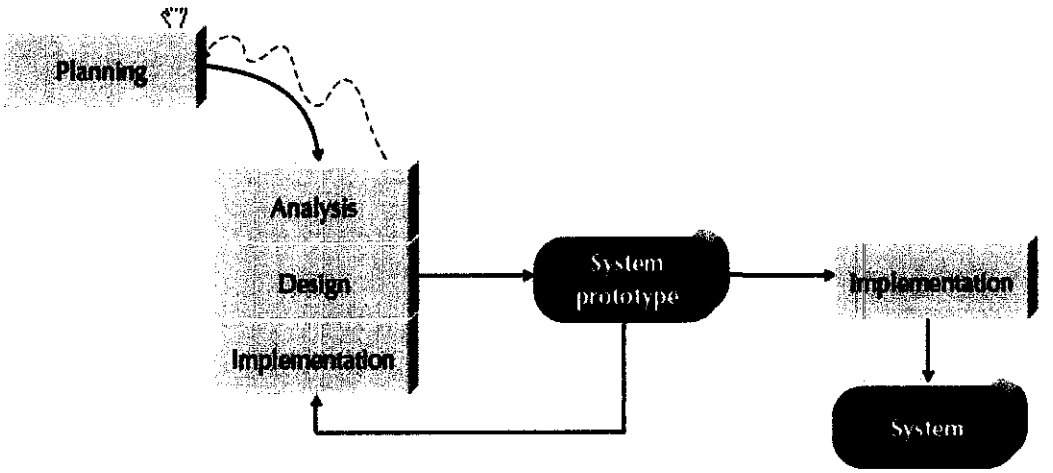


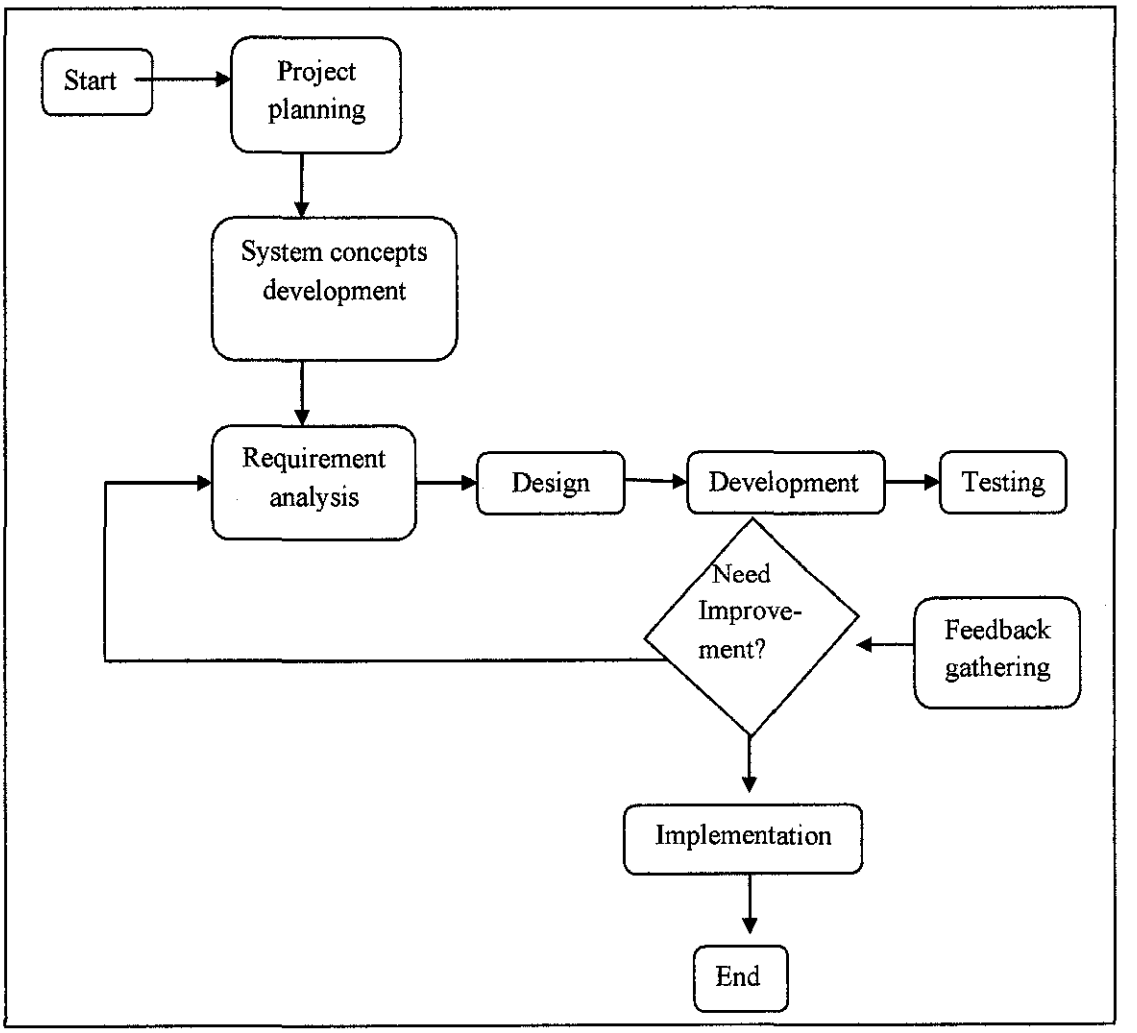
Figure 3.1 Prototyping Methodology

3.3 Project Phase Details

3.3.1 Planning

In this phase, a clear definition of problem statement is conducted. The reason is to know the suitable solution and how to provide answer for the particular solution. In order to provide the solution, system development plan is conducted like the timeline and milestone, analysis phase plan, architecture, user interface design plan, Gantt chart and etc. To come out with a good plan, research is done to identify which method and platform most suitable for this project by analyzing related context aware instant messaging system development project.

Figure 3.2 Project Flowchart



Key Milestone

- Data Gathering
- Analyze Requirement
- Design Architecture & System
- Develop Prototype
- Prototype Testing

Gantt chart

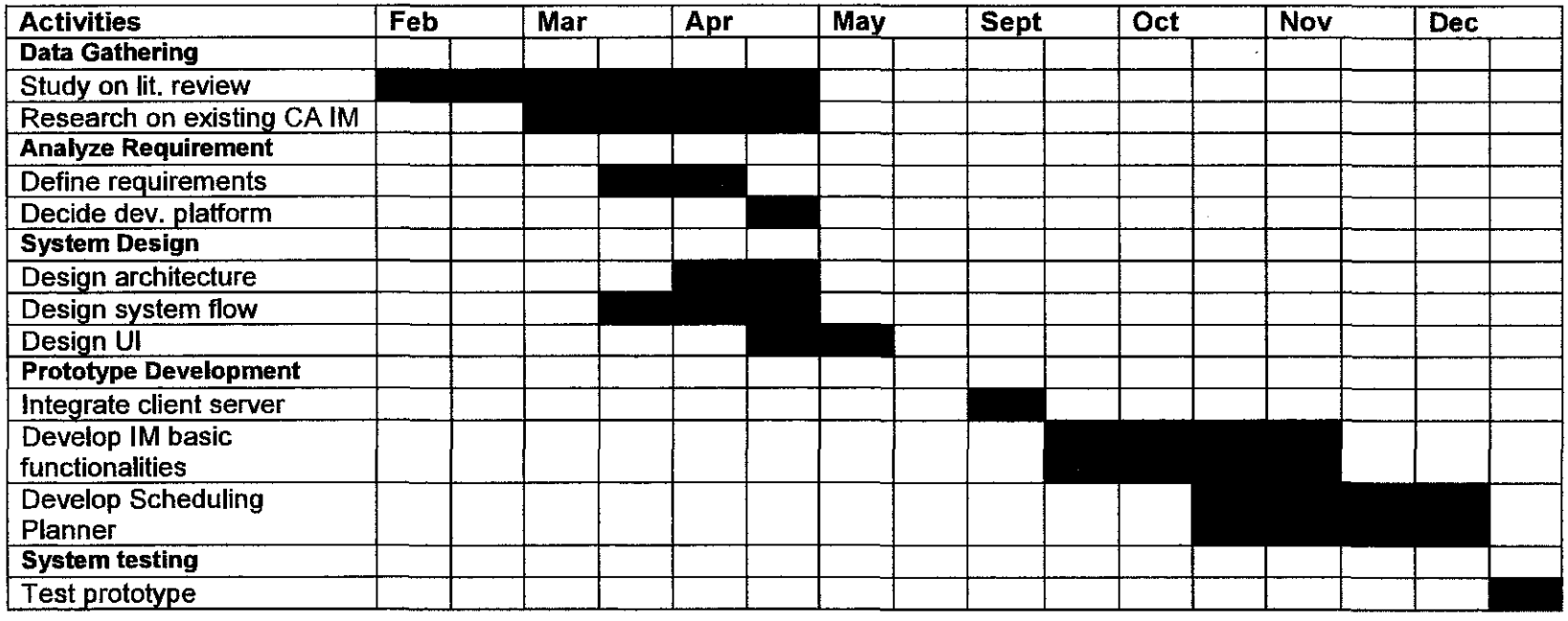


Figure 3.3 Project Gantt chart

Brief Description on the Project

The Context –Aware Instant Messaging with Integrated Scheduling Planner will be using presence awareness context as other IM system to provide the presence status or the availability of the user to be contacted. Meanwhile, for the Scheduling Planner function, the system requires access to user's personal calendar, in order to get appointment details of the user such as date and time. These details then will be matched with any appointment issued by the user's friend or contact list, so that the time and date of the new appointment does not clash with other appointment of the user. Thus, the context required for this function would be the date, time and also the calendar. To make it more simple to refer later on, let just naming these three contexts as schedule awareness. These contexts can be accessed from any application that is used by the user to store his/her personal calendar which contains all details of his/her appointments. The presence and schedule context then will be interpreted into meaningful data, such as the context of schedule awareness will be used to match and check whether there are clash or not with the new appointment issued by the user's buddies. The presence context will be the indicator whether the new appointment will be informed or not to the user. It means, if the user is online, then the system will pop up a window informing about the new appointment issued by his/her friend, and if the user is currently offline, the notification will notify the user when he/she go online. The pop up notification of the appointment will contain the details such as date, time, subject, agenda, place and number of participants.

3.3.2 Analysis

In the analysis phase, the suitable platform for development of the application is discussed in details. The available source codes of any suitable open source instant messaging programs is also evaluated and reviewed in order to get the best source code to be used to implement this project. Researcher has decided to use the source code written by Ahmet Oguz Mermerkaya to develop this project. Installation of IDE is required in order to code the program. This phase completes when all tools and system

configurations is built to support the development of the instant messenger in the next phase.

Platform for The Application Development

There are many instant messaging systems available and most of them are freeware. Few of this freeware IM are also open source IM whereby the source code of this IM is available for any users to study, change or make improvement on it. Nevertheless, almost all of these IM is developed for desktop computing environment. Nowadays, computing trend is more concern on the development of mobile applications resulting from the drastic changes in mobile computing technologies in the last few years. In the last few decades, developers are racing in producing software that can be used in multi-cross platform of desktop computing such as the system which is running in Windows should also be available for Linux, Unix and Mac OS. As the trend changing, current developers now is more focusing on the mobile-based system in order to cope with the users need and requirement who are increasingly keep moving from desktop computing to mobile computing for their daily tasks and communication.

Based on this reason, this project seems more suitable for mobile platform besides that there are only several IMs have been developed for mobile environment and none of them have advance features like their versions for desktop usage, especially the feature which will be implemented by this project. The mobile platforms must be compared first so that a suitable operating system for mobile could be chosen in order to develop a prototype for this application. By referring to the thesis produced by Bo Stendal Sorensen for his Context Aware Phonebook project [8], the mobile platforms are compared on the basis of the criteria below.

- Ability to access/read (e.g. in an API) main systems of the OS such as NotificationManager, Calendar.
- Internet connectivity, the ability to use regular sockets and the HTTP protocol.
- Ability to run background services.
- Programming language.

In order to create the instant messaging application, the program needs to be able to read and get access to the calendar and NotificationManager.

The comparison criteria of the platforms and the results have been summarized in the table below:

Table 3.1 Comparison of Mobile Platforms

	iPhone	Android	Symbian	Windows Mobile
Accessing calendar	Yes	Yes	Yes	Yes
Accessing NotificationManager	No	Yes	Yes	Yes
Internet connectivity	Yes	Yes	Yes	Yes
Background services	No	Yes	Yes	Yes
Programming Language	Objective-C	Java	C++ & Python	C & C++

After having comparison on the mobile platforms, Android OS is chosen for the development of the instant messaging system with integrated scheduling planner. The decision is also based on previous experiences with the Java language and the Android platform. Smartphones are ideal for context-aware applications since they are relatively powerful and contain various sensors. The two most promising platforms were iPhone and Android because of the popularity, high usability, powerful CPUs and available sensors. The Android platform was preferred to iPhone because it uses Java as the main programming language and there were several context-aware frameworks available for Java, it provides access to more core OS functionality, it does not require any certification or developer registration to deploy the software to hardware, and the Android SDK is available on multiple platforms [20].

3.3.3 Design

During design phase, system architecture is clearly defined. It includes suitable UML modeling diagram that will help in the development phase. Output from this phase is the graphical user interface layout, database architecture, modeling diagram like activity diagram of how the system works when user initiate a scheduled appointment to his/her buddy and etc.

Basic System Architecture

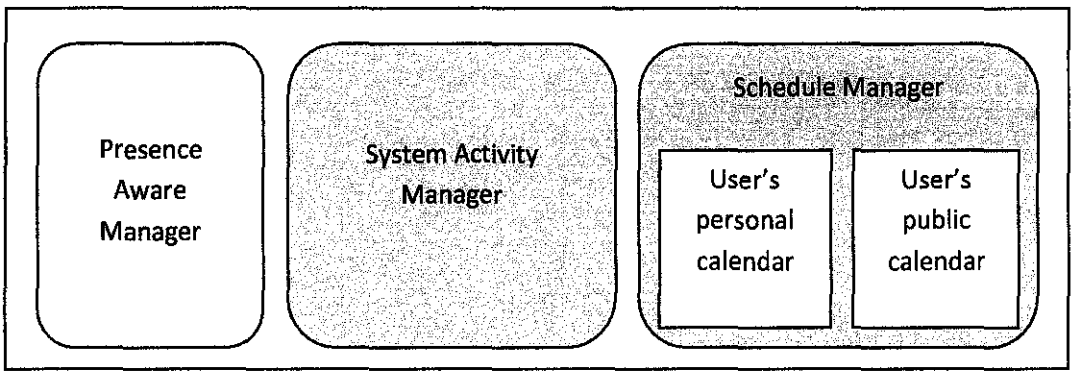


Figure 3.4 Context-Aware IM with Scheduling Planner Basic Architecture

Presence Aware Manager maintains the dynamical context (availability status) of the user. The System Activity Manager is responsible for providing asynchronous communication (handle chat activity between user and buddies). The Schedule Manager will be in charge for arranging the scheduled event issued by user's buddy, by matching the event's date and time with the user's public calendar, if there is free time slot at the time of the event.

Detailed System Architecture

Below is the detailed system architecture for the Context-Aware Instant Messenger with Scheduling Planner. It has three main components, IM client, web service and IM server.

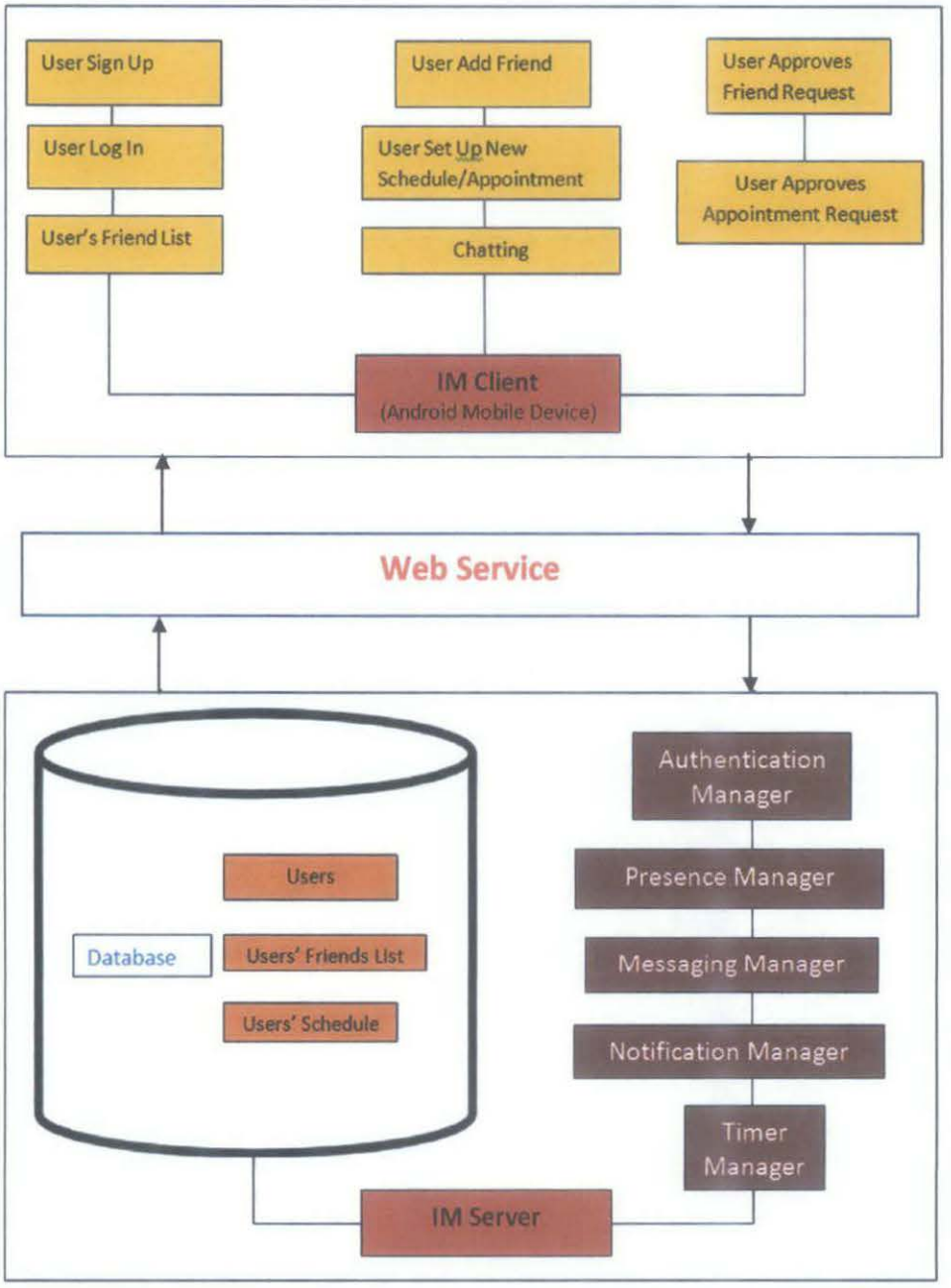


Figure 3.5 Context-Aware IM with Scheduling Planner Detailed Architecture

Scheduling Planner System Flowchart

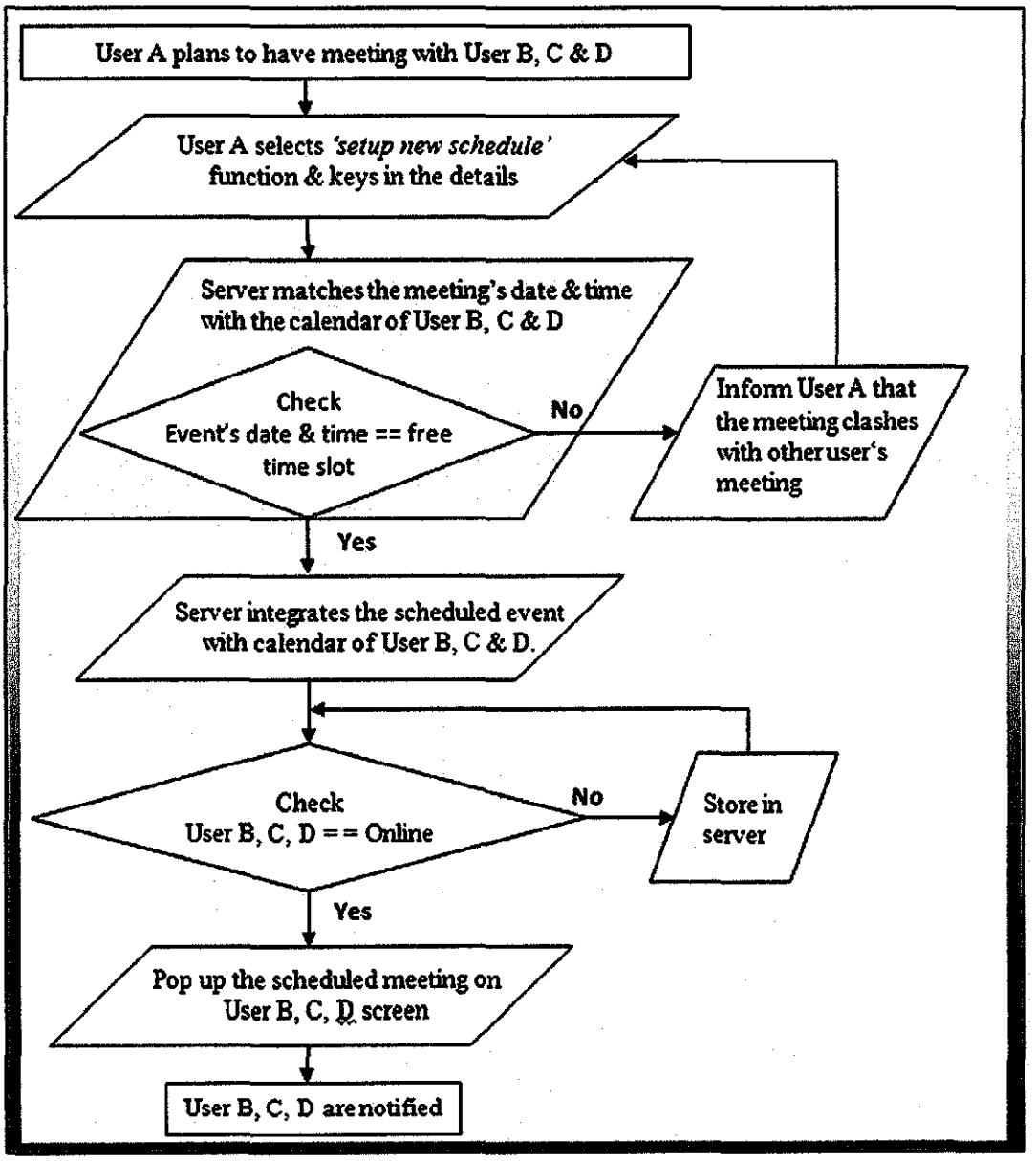


Figure 3.6 System Flowchart

3.3.4 Development

In development process, first of all is to identify which are the most suitable tools or technology need to be used in order to develop the instant messaging system. Then, with the chosen tools and platform, the product is developed based on the output produce during the design phase. For this project, below are details of tools and technology being used.

Tools and Technology

- Programming Language – Java
- Platform – Android operating system.
- Other tools
  - Server and client PC / mobile phone
  - Eclipse
  - Android Software Development Kit (SDK)

The development output for this project is the system prototype. For this project there are three prototypes:-

- Prototype 1 – Focus on developing the most important functionality
- Prototype 2 – Incorporate the main research element, the scheduling planner function
- Prototype 3 – Add on with other less important functionality

3.3.5 Testing Methodology

Usability testing has been conducted regarding to this project in order to determine either the developed project had met the five important element of usability. The five important elements are:-

- *Learnability* – either it is easy for users to accomplish the tasks the first time they encounter the design or the architecture of the system
- *Efficiency* – either it is easy for users to apply back the knowledge once they have learn regarding the design or the architecture of the system

- *Accuracy* – either there is any mistake done by the users when they use the application (And were they fatal or recoverable with the right information?)
- *Emotional response* – either the users confident, stressed or happy during and after using the application (How does the person feel about the tasks completed? Would the user recommend this system to a friend?)
- *Satisfaction* – either the design of the interface is pleasant to the users' eye and either users love the system's functioning being tested

The usability testing has been conducted to a group of 32 people, which are university students. Each of the participants had been given around 5 minutes of time in order for them to explore the system by themselves. No orders or directions were given to the participant in order to obtain the absolute results of the usability of the system. After the participants felt satisfied with the exploration of the system, each participant was given a set of questionnaire related to the five elements of usability. Results obtained from the questionnaire were gathered and analyzed in order to determine the usability of the system.

CHAPTER 4
RESULTS & DISCUSSION

4.1 Data Gathering

4.1.1 Observation

Table 4.1: Medium for Important Communication

Item no	Technology / Medium being used for important communication	Professional workers	Students
1.	E-mail	Yes, very frequently	Yes, but not frequently
2.	IM	Yes, seldom	Yes, very frequently
3.	Mobile Phone	Yes	Yes
4.	SMS	Yes	Yes, frequently
5.	Word of mouth	Yes	Yes, very frequently

*based on researcher's observation as a university student for almost 5 years and as a practical student during industrial training for 8 months.

Table 4.2: Medium Used to Schedule Appointment

Item no	Technology / Medium being used to schedule appointment/ meeting/discussion	Professional workers	Students
1.	E-mail	Yes, very frequently (formal method)	No
2.	IM	No	Yes, very frequently
3.	Mobile Phone	No	Yes
4.	SMS	No	Yes, frequently
5.	Word of mouth	Yes	Yes, very frequently

*based on researcher's observation as a university student for almost 5 years and as a practical student during industrial training for 8 months.

Discussion on Observation

Many professional workers nowadays uses internet in their smart phone as a way to facilitate communication related to their job matters such as to check email, access websites and obtain business-related information on the go. Based on researcher internship experience as a trainee in an IT company, staffs at the company usually schedule the meeting with their co-workers and subordinates through email (MS Outlook). Having two internship supervisors who mainly involve with clients everyday making them not always be in their workstations. Sometimes there were out of office to meet clients and sometimes they went to field sites for a couple of days or even weeks. Hence, the application proposed here would be a good application to be implemented.

Based on researcher's observation as stated in the table, there would be a need to develop a system or application which can be used by this kind of people in their tasks whereby it make their activity easier in a context that they can schedule meeting or appointment on the go in a structured and organized manner with the advanced and sophisticated technology that available today in a single step only. The single step refers to the situation where the users use the IM with built-in Scheduling Planner to schedule appointment with any individual in their contact lists by filling up the appointment details and touch/push "*send*" button. The receivers then will only view the appointment request and do one thing, to accept it or not. If they accept it by touching/ pushing "*accept*" button, the scheduled appointment will be placed in their activity calendar and the system will alert them when the appointment's date and time reached, the same concept if we use the function in commercial email application such as Outlook.

4.1.2 Questionnaire Result

Below are the questionnaire results on 53 respondents who are students of Universiti Teknologi PETRONAS (UTP).

Table 4.3: Questionnaire Result Table

Question no.	Question category	**Yes**	Yes Percentage	**No**	No Percentage
1	Experience of using IM apps	**53**	100%	**0**	0%
2	Experience of using mobile IM	**15**	28.3%	**38**	71.7%
3	Experience of schedule meeting/ discussion	**53**	100%	**0**	100%
4	Frequently doing appointment for meeting/ discussion	**29**	54.7%	**24**	45.3%
5	Experience of using IM to set the appointment	**43**	81.1%	**10**	18.9%
6	Choose to use IM with special scheduling function to schedule meeting/discussion	**39**	73.6%	**14**	26.4%

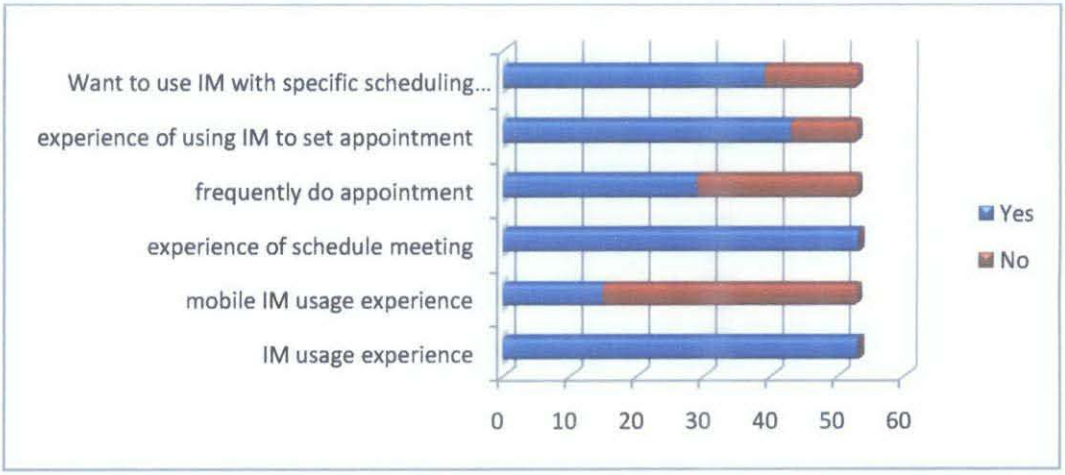


Figure 4.1: Questionnaire Result Graph

4.2 Data Analysis

4.2.1 Data analysis based on question categories

Based on the survey, 100% of the participants have been using instant messaging (IM) application like Yahoo messenger (YM), Google Talk (GTalk) and Skype. IM is widely used by students because it makes the communication easier between them whereas it works as an alternative for them to contact their friends without any cost unlike making phone call or sending SMS.

Meanwhile only one third of them have use instant messaging system for mobile such as GTalk for Android and Skype for Android even though most of the students use smart phone. This is due to the new emergence of android technology in the market and most of the students do not familiar yet with mobile instant messaging besides having money constraint to subscribe for internet line for their phones.

100 % of the participants have experience in schedule meeting and group discussion with their friends. Students always have to do meeting and discussion related to their subjects matters such as in doing assignment, study group, as well as non-academic matters such as meeting for club's event and so on.

The graph states that more than 50% of respondents frequently do appointment for meeting and discussion. That would be because the students are actively involved with events, clubs and association's activities that require them to frequently have meeting.

81.1% of the respondents have use IM as a medium to set meeting or discussion with their friends. They usually inform their friends by chatting them about the meeting's date and time. Even IM is not the preferable way to do that because they still can inform about the meeting through SMS and phone call but it shows that IM is widely used one of the medium for that purpose.

73.6% of the respondents would like to use IM as a medium for them to schedule meeting if the application have a special feature to schedule appointment in which it automatically notifies the person when it reaches the appointment time.

4.3 Project Deliverables

The instant messaging system with scheduling planner functional had been developed and delivered. Followings are the main interfaces of the application:-

4.3.1 Login Interface (Start Screen)

Figure 4.2: Login Interface

Figure 4.3: Login UI with Sign Up Menu

Figure 4.2 is the application's login screen. It is the first screen the user will encounter after opening the application. If the user is a new user, he or she has to register an account first in order to use the application. To register, push/touch (depend on mobile phone keypad design) on existing *"menu"* button on the phone and user will get the menu function as in Figure 4.3.

4.3.2 Sign up Interface

Figure 4.4: Sign Up Interface

Figure 4.5: Filled Up Sign Up UI

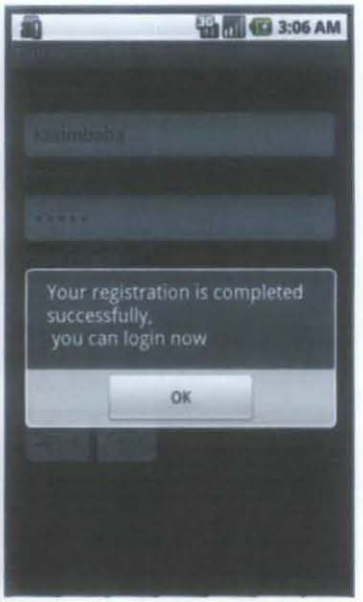


Figure 4.6: Sign Up Successful

When user choose the "*Sign Up*" menu, a Sign Up screen in Figure 4.4 will appear. Figure 4.5 shows the example of filled up registration form. After the user touches the "*Sign Up*" button, a pop up screen notifying the user that the registration is successful will appears as in Figure 4.6.

After the pop up appears, user will be redirected to the Login screen. Now, user can log in the application by filling in the username and password that he or she has registered previously.

4.3.3 Friend List Interface

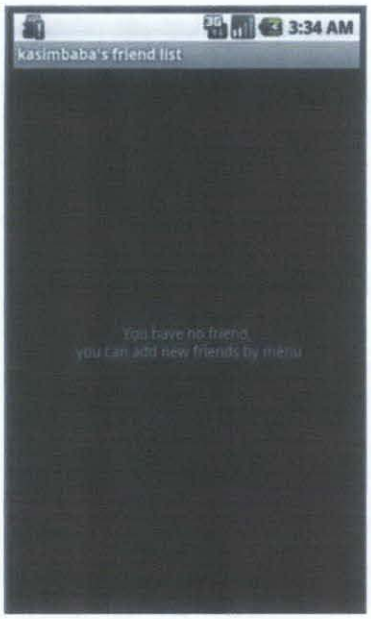


Figure 4.7: Friend List Interface

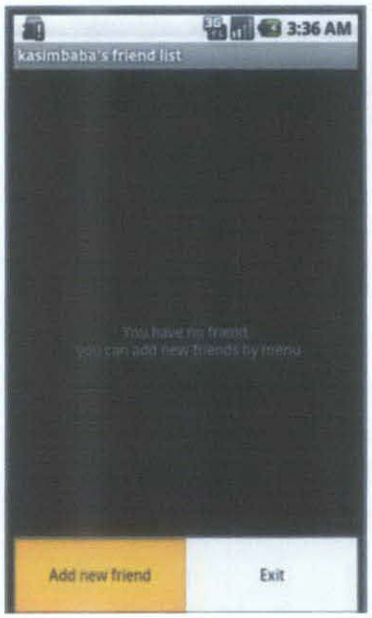


Figure 4.8: Friend List UI with Add Friend Menu

After successfully log in into the system, user will be displayed with friend list interface as in the Figure 4.7. If the user is a new user, the user interface (UI) will display an empty friend list whereas the user needs to add friend in order to communicate or messaging with his or her friend. To add friend, push on/touch the existing *"menu"* button on the phone and user will get the menu function as in Figure 4.8.

4.3.4 Add Friend Interface

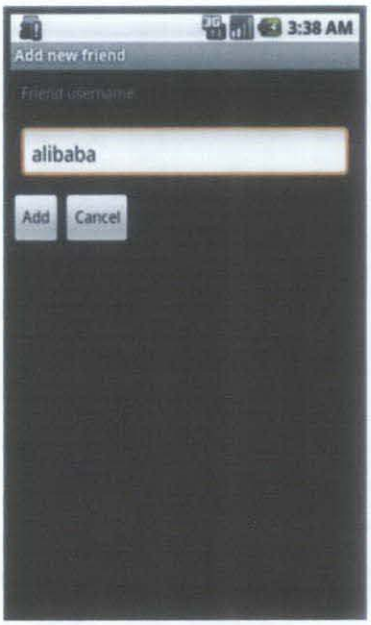
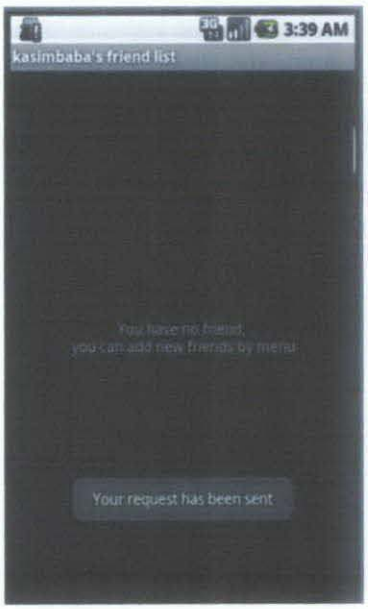


Figure 4.9: Add Friend Interface

Figure 4.9 above will come out when user touches the "*Add Friend*" menu in the previous section. After filling up friend's username in the provided field and tapping the "Add" button, a small notification appears informing user that the add friend request is sent as in the below image.



4.3.5 Friend Request Notification

Figure 4.10: Friend Request Notification

Figure 4.11: Unapproved Friend UI

Figure 4.12: Approve Friend Menu

User's friend will receive a notification stating that he or she got a new friend request as in the Figure 4.10. The user's friend then need to approve his/her friend request so that they would be able to communicate via this application. To approve the friend request, tick the friend name and push on/touch the existing *"menu"* button on the phone and user will get the menu function as in Figure 4.11. Select the option as in Figure 4.12 and both of them will get the friend list interface as following;-

4.3.6 Set up Schedule Interface (Project's Main Function)

Figure 4.13: Set Up Schedule Menu

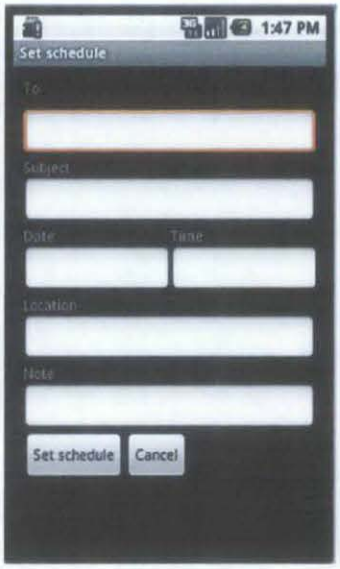


Figure 4.14: Set Up Schedule UI

Figure 4.15: Filled Up Set Up Schedule UI

Now come to the part of setting up schedule function. To make appointment for meeting, discussion and so on, push on/touch the existing *"menu"* button on the phone and user will get the menu function as in Figure 4.13. Select *"Set up new schedule"* option in the menu list and Set Schedule interface will come out like the one in Figure 4.14. Fill up the Set Schedule form as shown in the example in Figure 4.15 above and tap "Set Schedule" button. If the appointment's date and time clash with his friend's other appointment, a pop up screen will appear to notify the user about that.

4.3.7 Appointment Request Notification

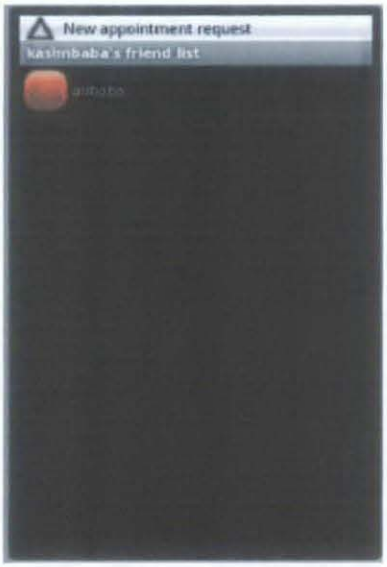


Figure 4.16: Appointment Request Notification

Figure 4.17: View Appointment Request

Figure 4.16 shows the other user gets notification stating that he gets new appointment request. The user needs to approve the scheduled appointment and it will be stored in that user's schedule or calendar. The final product would be a system that also provides notification (alarm sound) to alert the user when the appointment reaches its time.

4.4 Usability Testing

Referring to section 3.3.5, the results of the usability testing being conducted are as Figure 4.18 below:-

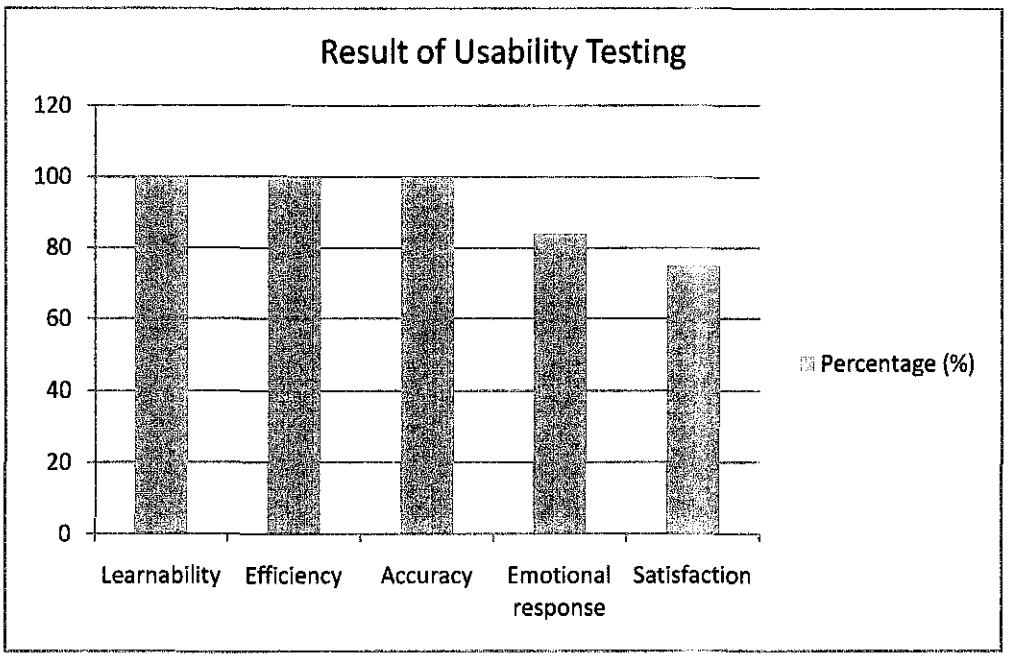


Figure 4.18: Level of System Usability

Figure 4.18 shows the percentage of achievement towards the five elements of Usability which are Learnability (100%), Efficiency (100%), Accuracy (100%), Emotional response (84.4%) and Satisfaction (75%). From these results, the researcher had discovered that all respondents like to use the application. The main issue found here is that few of the respondents do not like the interface. That would be because the interface is too simple and plain whereby it only contains text fields and buttons with dark background. As the system prototype used in this usability testing process is not the real complete product, it can be said that this aspect of usability testing would not affecting the overall result. Overall, the respondents seem satisfied with the scheduling planner function and its interface as it is not confusing and easy to use.

CHAPTER 5
CONCLUSION & RECOMMENDATION

This project highlights a new context awareness concept in instant messaging program. The design and development of an instant messenger with the stated new context awareness concept is performed mainly through the methodology being described in this paper. The main contribution of this project is a prototype of instant messenger program with integrated scheduling planner.

The development of an instant messenger program with scheduling planner is really relevant to the objective of this project because the application is designed to solve the problem being stated earlier. Before this, computer users especially those who are in the office sector usually make appointment with their co-workers or subordinates through email. By developing this application, they can use the instant messaging application to schedule the appointment instead of using email, which probably would be easier for them.

As a conclusion, throughout this project, researcher has successfully covered the study of context aware concept in IM system as well as developed IM with Scheduling Planner function. Based on the developed system, there is a functionality aspect that can be improved which is by informing free time slot of the intended users whenever the event's stated time clash with the intended users' current schedule.

In the future, this kind of application would be interesting if it can also support online conference which would be useful for the organization in the case if the employees would like to have informal conference or meeting with other staffs without need to go to meeting room or in case all meeting room are occupied at that time. This kind of future work would be good to be done.

References

1 Perttunen M, Riekki J., Inferring Presence in a Context-Aware Instant Messaging System.

2. Nardi, B., Whittaker, S., & Bradner, E. (2000) Interaction and Outeraction: Instant messaging in action. Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW 2000), New York, USA, 2000, 79-88.

3. Bradbury, D. (2001) Pigeon post for the 21st century. *Computer Weekly*, 2001 October 18, 1–6.

4. Fogarty, J., Lai, J., and Christensen, J. (in press) Presence versus Availability: The Design and Evaluation of a Context-Aware Communication Client. To appear in *International Journal of Human-Computer Studies (IJHCS).*

5. Greene, D., O'Mahony, D. (2004) Instant Messaging & Presence Management in Mobile Ad-Hoc Networks. In Proc. of the IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW'04). Orlando, USA, March 14-17, 2004.

6. Tianruo Y, Design of a Middleware and HIML (Human Interaction Markup Language) for Context Aware Services in a Ubiquitous Computing Environment, Embedded and Ubiquitous Computing: International Conference EUC 2004.

7. Guanling Chen and David Kotz (2000), *A survey of context-aware mobile computing research.* Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College.

8. Sørensen B.S. (2010), *Context Aware Phonebook.* B.Sc. Degree Thesis, Department of Informatics and Mathematical Modelling, Technical University of Denmark.

9. Antunes da Rocha R.C., Endler M. (2006), *Supporting Context-Aware Applications: Scenarios, Models and Architecture*, Department of Information, Pontificia University of Catholic, Rio De Janeiro.

10. LAC – Laboratory for Advanced Collaboration, BuddySpaceLive <http://www.lac.inf.puc-rio.br/moca/applications/buddyspacelive/>

11. Tang J.C., Yankelovich N., Begole J.B., Kleek M.V., Li F., Bhalodia J. (2001), *ConNexus to Awarenex: Extending Awareness to Mobile Users.* Sun Microsystems Laboratories.

12. Awareness and Communication for Mobile Users <http://labs.oracle.com/netcomm/awarenex.html>

13. Zhang ., Law C.F., Wang C.L. and Lau F.C.M (2009), Towards Pervasive Instant Messaging and Presence Awareness, *International Journal of Pervasive Computing and Communications* Vol. 5 No. 1, pp. 42-60.

14. Instant Messaging in the Workplace (2003) <http://searchunifiedcommunications.techtarget.com/definition/Learn-IT-Instant-Messaging-in-the-Workplace>

15. I.M. Generation Is Changing the Way Business Talks (2006) <http://www.nytimes.com/2006/04/05/technology/techspecial4/05message.html>

16. Begole, J.B., Matsakis N.E, Tang J.C. (2004), Lylsis: Sensing Unavailability.

17. ConChat <dsonline@computer.org>

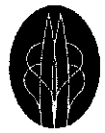
18. Tang, J.C., Begole, J. (2003), Beyond Instant Messaging. ACM Queue 2003 November, Volume 1, Issue 8.

19. Simons C., Wirtz G. (2007), *Modeling context in mobile distributed systems with the UML*, Distributed and Mobile Systems Group, Otto-Friedrich-Universitat Bamberg, 96045 Bamberg, Germany.

20. Wang A.I, Ahmad Q.K. (2010), *Context-Aware Model Framework*, Dept. of Computer and Information Science, Norwegian University of Science and Technology.

21. Planning an Oracle Communications Instant Messaging Server Sizing Strategy <http://wikis.sun.com/display/CommSuite/Planning+an+Instant+Messaging+Sizing+Strategy>

Appendices



UNIVERSITI TEKNOLOGI PETRONAS

FINAL YEAR PROJECT

Survey on IM Usage and How Student Schedule Meeting or Discussion with Their Friends

Please tick (/) for the correct answer.

1. **Have you ever use instant messaging (IM) system such as Yahoo Messenger, Gtalk, Skype etc?**
   Yes ( ) No ( )
   If no, state why?
   _____
   _____

   If yes, proceed to next question.

2. **Have you ever use IM systems in mobile phone?**
   Yes ( ) No ( )

3. **Have you ever schedule meeting or discussion with your friends?**
   Yes ( ) No ( )

4. **How frequently do you schedule meeting/discussion with your friends?**
   Never ( ) seldom ( )
   Often ( ) always ( )

5. **Have you use instant messaging system to set the meeting/discussion?**
   Yes ( ) No ( )
   If no, state why?
   _____
   _____

6. **Would you like to use IM application to set the meeting/discussion with your friends if the IM systems have special feature which enable you to schedule meeting/discussion with your contact list and it automatically notifies the person when it reaches the time?**
   Yes ( ) No ( ) Do not care ( ) Love it ( )

   If no, state why?
   _____
   _____

Questionnaire used during Usability Testing.

Usability Testing on Context-Aware Instant Messenger with Scheduling Planner

This questionnaire is intended in providing the researcher valuable information with regards to the topic mentioned. This questionnaire will be evaluating respondent opinion with regards to:-

1. The usability of the application developed which will be divided into three main categories, which are learnability, efficiency and satisfaction.

Learnability

1. Is it easy for you to be able to navigate through the system at the first view?

1	2	3	4	5
Very Easy				Very Difficult

2. Is it easy for you to get the concept of the system when you first try it?

1	2	3	4	5
Very Easy				Very Difficult

3. Rate the overall easiness to use the application?

1	2	3	4	5
Very Easy				Very Difficult

Efficiency

1. How fast can you use the function in the system?

1	2	3	4	5
Very slow				Very fast

2. Is it easy for you to perform back the tasks/function in the system?

1	2	3	4	5
Very difficult				Very easy

Accuracy

1. Is there any mistakes did you make when you use the system?

1	2	3	4	5
Once				Several times

Emotional response

1. How do you feel about the tasks completed?

1	2	3	4	5
Very unpleasant				Very pleasant

Satisfaction

1. Rate the pleasantness of the system's interface, such as the location of the button?

1	2	3	4	5
Very unpleasant				Very pleasant

2. Rate the pleasantness of the interface's color to the eye?

1	2	3	4	5
Very unpleasant				Very pleasant

Opinion on research's implementation

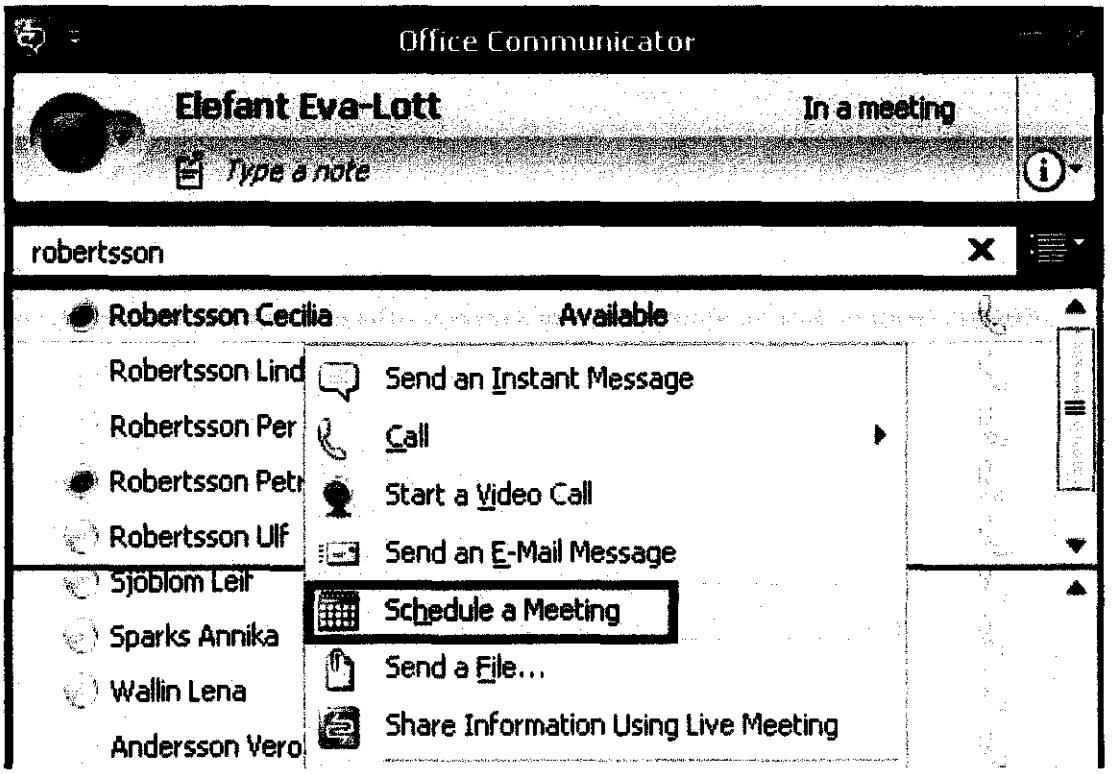
1. It is easy to grab the concept of schedule planner function in instant messaging system?

1	2	3	4	5
Very difficult				Very easy

2. Rate the research implementation, either it is suitable to be implemented or not?

1	2	3	4	5
Very unpleasant				Very pleasant

Schedule a meeting via Office Communicator



Sample codes for generating Scheduling Planner, written in Eclipse with Android SDK.

SetSchedule.java

```
package com.androidIM;

import com.mekya.interfaces.IAppManager;

import com.mekya.services.IMService;


import android.app.Activity;

import android.app.AlertDialog;

import android.app.Dialog;

import android.content.ComponentName;

import android.content.Context;

import android.content.DialogInterface;

import android.content.Intent;

import android.content.ServiceConnection;

import android.os.Bundle;

import android.os.IBinder;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;


//import com.mekya.interfaces.IAppManager;


public class SetSchedule extends Activity {

        protected static final int FILL_ALL_FIELDS = 0;

        public static final String SUBJECT = "subjectText";  //addition code for UnApprovedSchedule.java

        private EditText toText;

        private EditText subjectText;

        private EditText dateText;

        private EditText timeText;

        private EditText locationText;

        private EditText noteText;
```

```
    private IAppManager imService;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.set_schedule);
        setTitle("Set schedule");

        Button setScheduleButton = (Button) findViewById(R.id.setSchedule);
Button cancelButton = (Button) findViewById(R.id.cancel_setSchedule);
toText = (EditText) findViewById(R.id.to);
subjectText = (EditText) findViewById(R.id.subject);
dateText = (EditText) findViewById(R.id.date);
timeText = (EditText) findViewById(R.id.time);
locationText = (EditText) findViewById(R.id.location);
noteText = (EditText) findViewById(R.id.note);

setScheduleButton.setOnClickListener(new OnClickListener(){
                public void onClick(View arg0)
                {
                        if (toText.length() > 0 &&
                                subjectText.length() > 0 &&
                                dateText.length() > 0 &&
                                timeText.length() > 0 &&
                                locationText.length() > 0 &&
                                noteText.length() > 0
                                )
                        {
                                Thread thread = new Thread(){

                                        @Override
                                        public void run() {
        imService.setScheduleWithFriend(toText.getText().toString(),
```

```
    subjectText.getText().toString(),
                                                                    dateText.getText().toString(),
                                                                    timeText.getText().toString(),
    locationText.getText().toString(),
                                                                    noteText.getText().toString());
                                                }
                                        };
                                thread.start();
                                Toast.makeText(SetSchedule.this, R.string.request_sent, Toast.LENGTH_SHORT)
                                    .show();
                                finish();
                        }
                        else{
                                showDialog(FILL_ALL_FIELDS);
                        }
                }

    });

    cancelButton.setOnClickListener(new OnClickListener(){

                public void onClick(View arg0) {
                        finish();
                }

    });

}
@Override
protected void onResume() {
    super.onResume();
    bindService(new Intent(SetSchedule.this, IMService.class), mConnection , Context.BIND_AUTO_CREATE);
}
```

```
    @Override
    protected void onPause() {
       super.onPause();
       unbindService(mConnection);
    }
    private ServiceConnection mConnection = new ServiceConnection() {
       public void onServiceConnected(ComponentName className, IBinder service) {
                imService = ((IMService.IMBinder)service).getService();
       }
       public void onServiceDisconnected(ComponentName className) {
                imService = null;
                Toast.makeText(SetSchedule.this, R.string.local_service_stopped,
                                Toast.LENGTH_SHORT).show();
       }
    };
    protected Dialog onCreateDialog(int id) {
  switch (id)
  {
    case FILL_ALL_FIELDS:
    {
       return new AlertDialog.Builder(SetSchedule.this)
                .setTitle(R.string.set_new_schedule)
                .setMessage(R.string.signup_fill_all_fields)
                .setPositiveButton(R.string.OK, new DialogInterface.OnClickListener() {
                        public void onClick(DialogInterface dialog, int whichButton) {
                        }
                })
                .create();
    }
    default:
                return null;
  }
 }
}
```

UnApprovedSchedule.java

```
package com.mekya;

import com.mekya.interfaces.IAppManager;

import com.mekya.services.IMService;

import com.mekya.types.ScheduleInfo;

import android.app.ListActivity;

import android.app.NotificationManager;

import android.content.ComponentName;

import android.content.Context;

import android.content.Intent;

import android.content.ServiceConnection;

import android.os.Bundle;

import android.os.IBinder;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.ArrayAdapter;

import android.widget.ListView;

import android.widget.Toast;


public class UnApprovedSchedule extends ListActivity {


        private static final int APPROVE_SELECTED_SCHEDULE_ID = 0;

        private String[] subjects;

        String approvedScheduleReqs = new String();

        String discardedScheduleReqs = new String();

        private IAppManager imService;


        @Override

        public void onCreate(Bundle savedInstanceState){

           super.onCreate(savedInstanceState);


           Bundle extras = getIntent().getExtras();
```

```
    String title = extras.getString(ScheduleInfo.SUBJECT);

    subjects = title.split(",");

    setListAdapter(new ArrayAdapter<String>(this, android.R.layout.simple_list_item_multiple_choice, subjects));

    getListView().setChoiceMode(ListView.CHOICE_MODE_MULTIPLE);

    // canceling schedule request notification
    NotificationManager NM = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
    NM.cancel(R.string.new_schedule_request_exist);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    boolean result = super.onCreateOptionsMenu(menu);

    menu.add(0, APPROVE_SELECTED_SCHEDULE_ID, 0, R.string.approve_selected_schedules);

    return result;
}

@Override
public boolean onMenuItemSelected(int featureId, MenuItem item)
{

    switch(item.getItemId())
    {
            case APPROVE_SELECTED_SCHEDULE_ID:
            {
                    int reqlength = getListAdapter().getCount();

                    for (int i = 0; i < reqlength ; i++)
```

```
                {
                        if (getListView().isItemChecked(i)) {
                                approvedScheduleReqs =
approvedScheduleReqs.concat(subjects[i]).concat(",");
                        }
                        else {
                                discardedScheduleReqs =
discardedScheduleReqs.concat(subjects[i]).concat(",");
                        }
                }
                Thread thread = new Thread(){
                        @Override
                        public void run() {
                                if ( approvedScheduleReqs.length() > 0 ||
                                        discardedScheduleReqs.length() > 0
                                )
                                {
                                        imService.sendScheduleReqsResponse(approvedScheduleReqs,
discardedScheduleReqs);

                                }
                        }
                };
                thread.start();

                Toast.makeText(UnApprovedSchedule.this, R.string.request_sent,
Toast.LENGTH_SHORT).show();

                finish();
                return true;
        }

  }

  return super.onMenuItemSelected(featureId, item);
```

```
    }

    @Override
    protected void onPause()
    {
       unbindService(mConnection);
       super.onPause();
    }

    @Override
    protected void onResume()
    {
       super.onResume();
       bindService(new Intent(UnApprovedSchedule.this, IMService.class), mConnection , Context.BIND_AUTO_CREATE);
    }

    private ServiceConnection mConnection = new ServiceConnection() {

       public void onServiceConnected(ComponentName className, IBinder service) {
                imService = ((IMService.IMBinder)service).getService();


       }
       public void onServiceDisconnected(ComponentName className) {
                imService = null;
                Toast.makeText(UnApprovedSchedule.this, R.string.local_service_stopped,
                                Toast.LENGTH_SHORT).show();
       }
    };

}
```