NOTEPHONES SYSTEM

by

Allen Mwangonde

A project submitted in partial fulfilment of the requirements for
the degree of
BSc Computer Science (Hons)

University of the Western Cape

2012

Date: Saturday, 24 November 2012

University of the Western Cape

Abstract

NOTEPHONES SYSTEM

by

Allen Mwangonde

Supervisor: **Professor Isabel Venter**
Department of Computer Science

As a variety of phone devices becomes pervasive, easy access to minutes of meetings can help work groups better communicate ideas and information. To explore this idea further, interviews were carried out to find out how members of a work and research group share these meeting notes. It became apparent that introducing a mobile-based system in which members will have access to minutes and other ancillary data with ease and less need for a computer would be valuable. Computers, cell phones and instant messaging are integral parts of our lives today, hence the idea to design NotePhones, a minutes-sharing system with a lightweight process, an interface, and hardware that distinguish it from previous systems. This minute sharing application has been designed to run on even inexpensive windows cell phones and other personal digital devices. It is hoped that the experience with using NotePhones will facilitate and promote the sharing of minutes for various groups like meetings, conferences and classes.

.

DEDICATION

To those from whom I learned, and

To those whom I love.

TABLE OF CONTENTS

LIST OF FIGURES

Number *Page*

LIST OF TABLES

ACKNOWLEDGMENTS

First and above all, I praise God, the Almighty, for providing me with this opportunity and granting me the capability to proceed successfully. Apart from my personal effort, the success of any project depends largely on the encouragement and guidelines of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

I would like to show my greatest appreciation to Professor Isabella M. Venter. I cannot thank you enough for her tremendous support and help. I feel motivated and encouraged every time I attend her meeting. Without her encouragement and guidance, this project would not have materialized. I still vividly remember the first time I met Isabel, as she warmly greeted me into her office. At that time, I had little idea of research, and wasn't even sure about which area of Computer Science to do my Honours Degree in. Ever since I joined Isabel's group, I have learnt a lot from her in all aspects of research, from identifying and formulating problems to writing, presenting, and disseminating research results. Isabel has been instrumental in shaping my career until now, and I will continue to benefit from all her guidance far into the future.

GLOSSARY

Client – is any person, or organisation for whom the member works, or undertakes to provide mobile-based aid, in any way.

GUI – Graphic User Interface allows people interaction with programs in more ways than typing; it uses graphical icons, and visual indicators, rather than text-based interfaces.

High Tech Computer Corporation (HTC) is a Taiwanese manufacturer of smartphones and tablets.

ICT4D –Information and Communication Technology for Development is a general term referring to the application of Information and Communication Technologies (ICTs) within the fields of socioeconomic development, international development and human rights such as Digital Economy, e-Learning, e-Government or Open Access or Free Software.

Member – includes all categories of corporate membership defined in the Work Society's Articles of Association.

Microsoft .NET Compact Framework is a collection of runtime libraries which allow you to run .NET Compact 3.5 applications and services on your Windows Mobile device.

Microsoft Visual Studio – is an integrated development environment (IDE) from Microsoft used to develop console and graphical user interface applications along with Windows Forms applications, web sites, web applications, and web services in both native code together with managed code for all platforms supported by Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework and Microsoft Silverlight

Minutes – the record of issues discussed at a meeting

PDAs – personal digital assistants

Smart phone – -is a mobile phone built on a mobile computing platform, with more advanced computing ability and connectivity than a feature phone such as ;high-resolution touchscreens; web browsers that can access and properly display standard web pages rather than just mobile-optimized sites; and high-speed data access via Wi-Fi and mobile broadband.

SQL – Structured Query Language is a programming language designed for managing data in relational database management systems (RDBMS).

System–means all applications involving the use of computer and information technology. The term does not imply any particular mode of processing, e.g. local batch or remote real time, etc. "System" may be interpreted as encompassing non-computer procedures and disciplines, e.g. clerical, manual, etc.

User – is any person, department or organisation served by the mobile-based systems.

UML

Chapter 1

THE USER REQUIREMENTS DOCUMENT

Introduction

The management of meeting minutes is a generally cumbersome task. While a web-based system is more efficient than the traditional paper-based minute management system, this project aims to go a step further by providing a mobile-based system for managing minutes. This document will guide the development of a mobile management system. It will give an overview of the project and discuss the rationale of the project and what can be expected from the system once completed.

Today's generation has not just changed incrementally from those of the past, nor simply changed their slang, clothes, body adornments, or styles, as has happened between generations previously - but a really big discontinuity has taken place. One might even call the change a "singularity" (Marc, Digital Natives Digital Immigrants, 2001) – an event which changes things so fundamentally that there is absolutely no going back. This so-called "singularity" is the arrival and rapid dissemination of digital technology in the last decades of the 20th century. This generation represents the first generation that has grown up with digital technology. People have spent their entire lives surrounded by and using computers, videogames, digital music players, video cams, cell phones, and all the other toys and tools of the digital age. Today's average college graduates have spent less than 5,000 hours of their lives reading, but over 10,000 hours playing video games (not to mention 20,000 hours watching TV) (Marc, Digital Natives Digital Immigrants, 2001). Computer games, email, the internet, cell phones and instant messaging are integral parts of their lives.

It is thus obvious that the digital generation, as a result of this ubiquitous environment, would expect of a minute management system to allow users to easily navigate and retrieve minutes from recorded meetings via their cell phones, ipad,WAP phones, PC/Laptop, Pocket PC and PDAs, etc.

This chapter discusses the user's views of the problems related with current minute management systems and what is expected of the software that can be developed to alleviate them.

User's View of the Problem

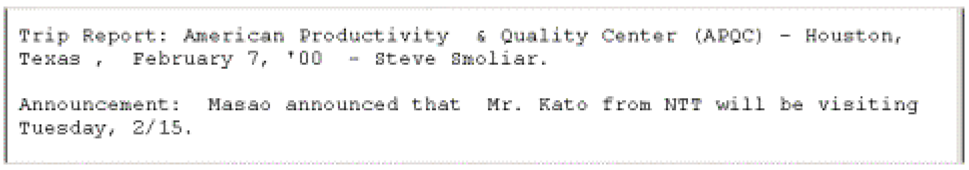
Traditional paper-based minute management is time consuming and involves the use of a considerable amount of paper. Participants also require reminders on meeting dates and times and their duties. Problems also arise when participants wish to consult previous minutes which may not readily be available. Paper-based filing systems allow paper documents to reside in only one place at a time. It is difficult to share minutes, and members generally make their own copies. According to Ms Fatima Jacobs and René Abbott, 'the average document gets copied 19 times, and of course, many of these copies also get filed'. Finding and retrieving a document using a paper-based system is slow. In situations where minutes contained in a document are required immediately, the delay may cause meeting postponements. In addition, re-filing paper documents wastes time and may result in misplacement of the files.

A web-based system such as the prototype that was developed by Mahangu, (2010) overcomes many of these problems. However, this requires connection to the internet, and most probably, a computer. This means that users cannot access the system when they do not have access to these (in some cases very expensive) resources. People on the go may not have the time to find a computer in order to check emails and read the minutes. By introducing a mobile-based system, users will have access to minutes and other ancillary data without needing access through a computer.

A Brief Description of the Problem Domain

The Computer Science Department at the University of the Western Cape has been using emails to share minutes of meetings with participants for over one and a half years. Notes are taken on paper by a single person, transcribed and sent out in an email to all staff members concerned. Figure 1 shows a sample of these notes. The transcription is typed using an email application and not a word processor. Transcription is a tedious process, and the note taker often has to track down people in the meeting to clarify what was said or to obtain information that was shown on a slide. Part of this is due to the difficulty of capturing everything in a meeting. Grammatical errors such as the spelling of names and technical terminologies usually need checking.

Two people took notes over this period of a year and a half. For a span of several months, one person served as the primary note taker with the other occasionally substituting. Then another person rotated in as the primary note taker. These two were part of the staff at the department but not researchers, and were inexperienced users of computer technology.



```
Trip Report: American Productivity  & Quality Center (APQC) - Houston,
Texas ,  February 7, '00  - Steve Smoliar.

Announcement:  Masao announced that  Mr. Kato from NTT will be visiting
Tuesday, 2/15.
```

Figure 1 Sample of email meeting minutes

Complete Description of the Problem

NotePhones is a mobile-based, collaborative minute sharing application that runs on mobile devices. This will allow users to access minutes quickly using their own handsets without relying on the computer. The drive to create this system came from the realization that people often leave meetings without a shared understanding or a record of the important points that occurred. Assigning a scribe to record the minutes is one solution, but it is onerous and sometimes produces a biased record. Using web-based meeting support tools is another solution, but this requires an expensive, fixed infrastructure that limits the locations where minutes can be accessed. In additional, people on the go may not have the time to find a computer in order to check emails and read the minutes. Since many people are now carrying small, inexpensive cell phones, NotePhones will provide a better platform for meeting support tools.

A survey was conducted to determine the preferences for accessing meeting minutes. A key question was whether people like to have meeting minutes delivered as email or to have the minutes on the phone for browsing; i.e. the push/pull question. This survey involved interviewing 4 people in their offices by asking them four questions (see Appendix B).They were also given an opportunity to make comments after answering these questions. For question (1), 3 of the 4 subjects answered that they use and read the emails. One subject commented that he 'looked more carefully if he missed the meeting'. Another read them for the 'spin'. For question (2), 3 preferred phone, 2 preferred the email, 1 preferred both, and 1 said 'it doesn't matter'. One subject commented: 'won't go to the internet café to look, only email'. Another commented that it was 'easier to find things on the phone than through email'. Yet a third person saved all the email minutes and felt that he could find things by searching through them.

This survey indicates that in order to support the habits of phone users, it is desirable to have phone access to the meeting minutes. NotePhones is a collaborative minute sharing tool that lets people walk away from any meeting, presentation, or class with a low-overhead record of what transpired. Previous research in this area through a survey of minutes management system users (Mahangu, 2010), and interviews with potential NotePhones users indicates that most users require a simple interface for minute sharing and management. This can be provided by NotePhones. The participants can sort and filter the minutes by time, project, date, and type, thus simplifying the organisation and cross-reference of minutes.

Software Expectations

According to the survey conducted, the following is expected of the software product:

- A simple interface that will be easy to use and understandable on a basic mobile devices. Users require a system with such an interface of uncomplicated buttons, plain English language and easy to follow description of some procedures (if necessary).
- A user authentication to ensure that only authorise users have access to the minutes. Authentication is the process of discovering and verifying the identity of a principal by

examining the user's credentials and validating those credentials against some authority. The information obtained during authentication is directly usable by the source code. That is, once the identity of the principal is discovered, you can use security tools such as .NET Framework role-based security, to determine whether to allow that principal to access your code. A variety of authentication mechanisms are used today, many of which can be used with .NET Framework role-based security. Some of the most commonly used mechanisms are application-defined mechanisms, use of usernames, passwords, and unique numbers. To some extent each work group or research group can agree on how to set these authentication parameters such as defining a username by a certain unique number and a password by date of birth.

- Sending out notifications to users on meeting schedules and other tasks

- Compatibility with all phones so that every member with any phone type should be able use this system and have access to meeting minutes via this NotePhones system

- Complete minutes management. Despite the fact that mobile devices have limited memory capabilities, this system should still be able to provide all functionalities necessary for minutes management such as capturing of minutes and agendas, audio and video recording, etc.

- Fully configurable and fully secured. This involves changing the alert time or schedules and adding minutes or adding apology for failing to attend the meeting, and making sure that only registered users should be able to make changes.

What is not expected from the Software Solution

The system is not expected to capture minutes automatically .They will still have to captured and loaded onto the system manually. Hence the task of minutes capturing and recording still rests on the skills of the minutes taker (usually the secretary).

A user will need a phone that supports .NET Compact Framework (.NET CF) 3.0 and SQL Server CE (Compact Edition) and Windows Mobile 6.0 Professional SDK.

In addition, this software will not facilitate the dissemination of audio recordings of minutes. The issue of strong security measures to limit access of minutes to the members only may not really be addressed as expected because the main aim of the project is to develop a system that will just promote easy access to minutes through mobile devices other than computers.

Lastly, this applications will use flat files to store information; requiring that each user have access to a shared network directory in order to use the system.

Conclusion

This project sets out to create a mobile-based minute management system which will be simple to use and provide easy and quick access to users. Such a system will alleviate some of the problems associated with paper-based and web-based systems. As previously stated, it is hoped that the experience with using NotePhones will facilitate and promote the sharing of minutes for various groups like meetings, conferences and classes.

In the next chapter all these user requirements discussed above shall be analysed from the designer's point of view. Currently, further research is being carried out on the current system and possible solutions to address the user requirements shall be suggested.

C h a p t e r 2

REQUIREMENTS ANALYSIS DOCUMENT

Introduction

In this chapter the requirements identified in the previous chapter are analysed and the problem is looked at from the designer's point of view. The focus is on the system and software requirements and possible solutions to the problem.

Designer's Interpretation of the Problem

Mobiles have become an integral part of our lives. They have become best aide acting as schedulers, reminders, organizers, file sharers, personalized music systems and much more than one can expect. Additionally, project managers or designers, and users in the developing world, especially users in rural areas, live in vastly different contexts (UCT, 2009). Fortunately, the use of mobiles is beginning to bridge the digital divide, and that leads us to the issue of developing mobile applications.

The rise of the mobile phone has challenged the predictions that many information and communication technology for development (ICT4D) specialists offered about minute management in the developing world. Instead of embracing community paper-based minutes management system that offered shared access to minutes, many people, especially those on the go, do not have the time to find a computer in order to check and read the minutes. This project is rightly building a mobile system that leverages the platform to promote economical and easy ways of accessing meeting minutes.

Breakdown of the Problem

NotePhones system is a mobile application that will provide easy access to minutes. It will also track and remind members of meeting events before they occur. It will display detailed meeting schedules, minutes of previous meetings or any regularly scheduled event in a list. The system will also alert members of events at any specified time.

Complete Analysis of the Problem

- **Self-explanatory with simple interface language**
  NotePhones provides a very simple to use yet very powerful graphical user interface (GUI) to manage all minutes. Sleek and stylish interface with description for every button and simple language. No complicated terminology, easy to follow instructions, i.e. user friendly

- **Complete Minutes Management**
  It can be configured to set reminders for minutes, meeting schedules and other such events.
- **Authentication**
  It will have a user authentication process to ensure that only authorised users have access to the minutes. In cases where the device is stolen or the system hacked into, the unauthorised user will face the hurdle of acquiring an ID/password to access the application.
- **Alerts**
  NotePhones will remind users of all current and upcoming events and alert staff members on meeting dates and times and their duties.The alerts will be displayed in a small pop-up window in the phone tray. Users will be able to conveniently snooze these alerts.
- **Fully configurable**
  NotePhones shall be fully configurable. A user will be able to configure the system in order to change the alert time, snooze interval, days for which the upcoming meeting will be held. In addition, configurations to enable/disable the pop-ups will be possible without the need to restart the system after changing the settings.
- **Simplifying the organization and cross-referencing of minutes**
  This will be done according to dates, months and subject.
- **Detailed Help**
  NotePhones is comprehensive.
- **Lightweight process**
  NotePhones is small package to run on mobile devices with small memory.
- **NotePhones Limitations**
  - No capturing of minutes automatically and audio recordings of minutes.
  - Handset operability: There are a large number of different mobile phone devices and it is a big challenge for the designer to offer mobile minute sharing solution on any type of device. Some of these devices support Java ME and others support SIM Application Toolkit, a WAP browser, or only SMS.Support This application will only support the .NET Compact Framework (.NET CF) 3.0 and SQL Server CE (Compact Edition) environment e.g. Smart phone and PDAs.

Current Solution

At present there are two official systems that are solutions to the problem. The first solution makes use of hard papers, rather than emphasizing computer version techniques. The second one is the web-based system.

- **Paper-based**

  The managing of tasks, agenda formation and minutes is performed by hand and paper. During every meeting the secretary of the meeting records minutes by hand, types it up and sends it to the participants for correction and to see which tasks were allocated to them. The documentation of this system's processes depend mainly on the secretary. Whether the participants have carried out their task will only be known at the next meeting. (Mahangu, 2010)

- **Web-based**

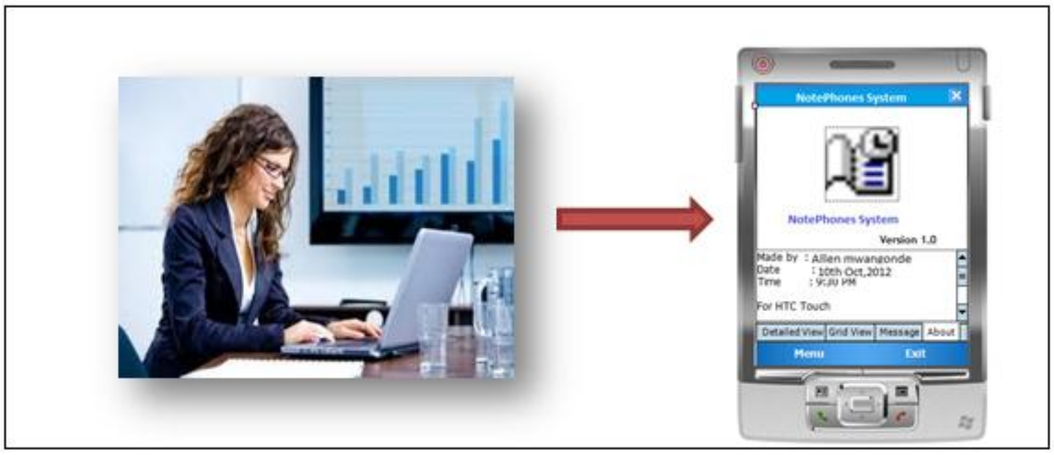
  The second current solution to the user's problem is a web-based system (see Figure 2) which assists with the compilation of an agenda for a meeting, store the minutes that will be manually entered in a database after every meeting. The system then sends an email to participants that have been assigned specific tasks during a meeting and reminds them of their duties at regular intervals until the task has been completed. If a task has been completed the participants indicate via email and they are never reminded again. The system was found to be 50% accurate using a sample size of 4 potential users.

Possible Solution

The possible solution to the user requirements is to develop a mobile-based system (NotePhones) which will be simple to use and provide easy and quick access to users. Such a system will alleviate the problems associated with paper-based and web-based systems.

The code in the NotePhone system will be able to scan the application database filled with people's information for their meetings to match with the current date. If it matches, the code will automatically send an SMS to the member's mobile number (as a reminder). Hence, there will be no need to remember any meeting schedules once they are fed into the application. The code will take care of the rest.

Modeling the Solution



Made by : Allen mwangonde
Date : 10th Oct,2012
Time : 9:30 PM

For HTC Touch

Figure 2: From paper-based to mobile-based

System Requirements

The system is expected to work under the following conditions (see the glossary for definitions of the terms used in the table):

Table 1: Tools used

	Computer	Mobile device
Supported operating systems	Windows® Vista® Windows® 7 Windows XP	Windows mobile
Hardware	2 MB of available hard disk space. (An additional 2 MB of space must be also available for temporary setup files.)	
Software	SQL Server 2005 CE SDK Windows mobile device centre Windows Mobile 6 Standard SDK Refresh Windows Mobile 6.0 Professional SDK Refresh	NET Compact Framework (.NET CF) 3.0 SQL Server 2005 CE (Compact Edition)
Programming language	Visual Studio 2008 Service Pack 2 (SP2	

Conclusion

In this chapter the user requirements were analysed and a detailed discussion of the system requirements was provided. The designer's view of the problem and the possible solutions were explored.

The complete breakdown of the functionality in which the system will be implemented were discussed. In the next chapter, User Interface Specification (UIS) will be discussed as well as modelling techniques for the development of the prototype.

C h a p t e r   3

USER INTERFACE SPECIFICATION

Introduction

In the previous chapter, the user's requirements were analysed and a possible solution was identified. The designer also modelled the system that will be designed. The purpose of this chapter is to provide a detailed specification of the NotePhones user interface. These requirements will detail the outwardly observable behaviour of the program. The user interface provides the means for the user to interact with the program. The user interface specification (UIS) is intended to convey the general idea for the user interface design and the operational concept for the software. This document will be updated with additional details as the analysis and design activities progress.

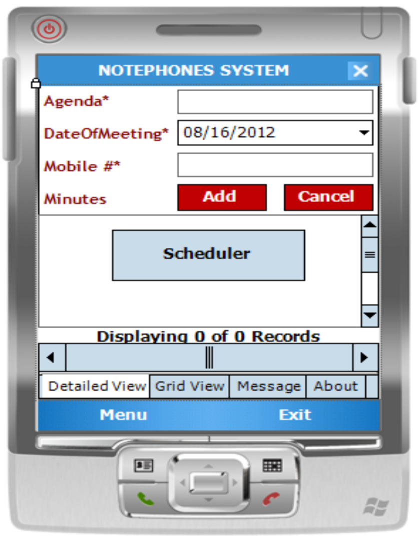
Description of the complete user interfaces

The interface will consist of six main functions, four of which will be carried out by the common user (the member), and the other two functions will only be available to the administrator (the secretary responsible for entering/updating agendas and the minutes) of the system (see Figure 8). The administrator can also act as a common user.

The interface consists of four simple and easy-to-use screens:

Screen # 1 & 2: Detailed View

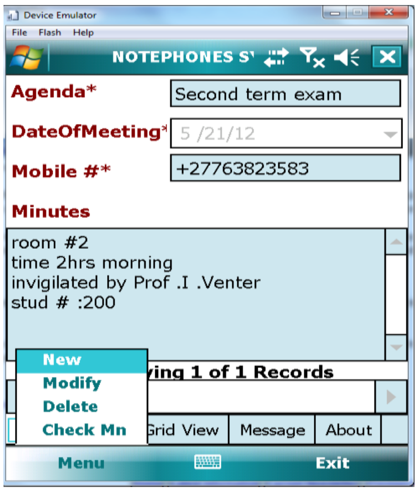
The first screen is the detailed screen for viewing the agenda (see Figure 3). This is a minutes archive page where one can view the minutes of a previous meeting and a page where a new agenda will be added or deleted.


NOTEPHONES SYSTEM

Agenda*	
DateOfMeeting*	08/16/2012
Mobile #*	
Minutes	Add / Cancel

Figure 3: Detailed View

The detailed view will also provide per record detail for each field. In addition, it will enable the addition and deletion of minutes. The user will be able modify or update the minutes using the menu tab (see figure 4).



Agenda* Second term exam

DateOfMeeting* 5 /21/12

Mobile #* +27763823583

Minutes

room #2
time 2hrs morning
invigilated by Prof .I .Venter
stud # :200

ing 1 of 1 Records

Figure 4: Menu tab

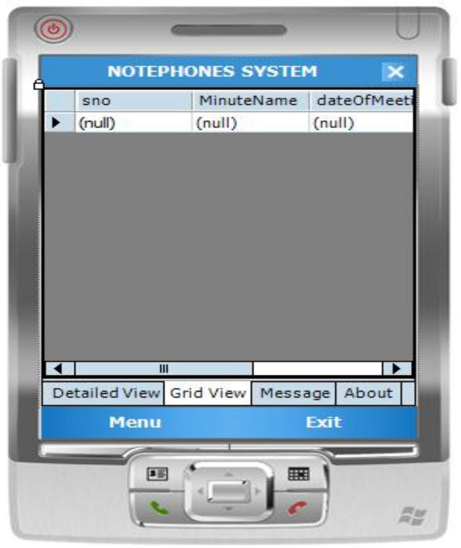
This interface has the following three functions for the administrator:

- Creation of a new agenda and editing of agenda;
- Creation of a new user and deleting an existing user; and
- Sending of reminders for the upcoming meetings and tasks assigned to each member.

It is necessary to point out that since the phone screen is relatively smaller than a computer or laptop, tabbed control (with great creativity) is used to segregate the functionality in multiple tabs so that the screen breathes with space and does not look jammed up.

Screen #3: Grid View

The second interface is the grid view which shows all the records on one screen (see figure5). This screen shows all the records of the minutes that were inserted into the table when the database was created. A user will can scroll the screen right and left and resize the columns with the column dividers on the header row to see all of the fields in the table.



	sno	MinuteName	dateOfMeeti
▶	(null)	(null)	(null)

Figure 5 : Grid View

Screen # 4: Message window

The next interface is the message window which will display a text box containing the messages that will be sent automatically to the user (see figure). This screen shows a text box which contains the minutes or apologies that will be sent automatically to the user. This is where the user will be able to write and send requests for new agendas and send apologies in case of failure to attend the meeting. The user will use the menu tab to make necessary corrections on the sent minutes within a certain period of time beyond which the system will not allow any changes to be made to the minutes.

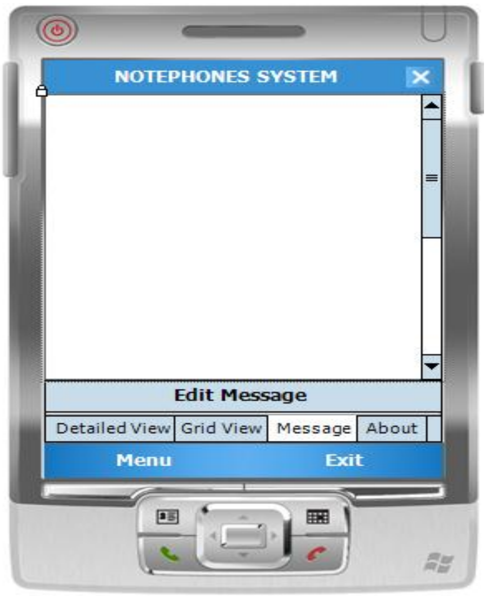
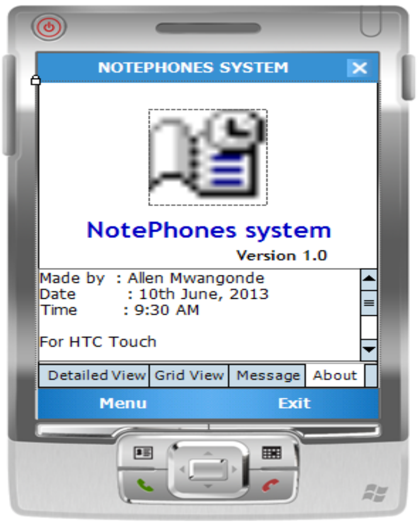


Figure 6 : Message window

Screen # 5: About window

Figure 7 shows the 'about screen' which contains information about the designer of the NotePhones System and the date of its development.


NOTEPHONES SYSTEM

NotePhones system

Version 1.0

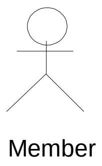
```
Made by  : Allen Mwangonde
Date       : 10th June, 2013
Time       : 9:30 AM

For HTC Touch
```

Figure 7 About Window

How the user interacts with the interface

Use Case Diagram

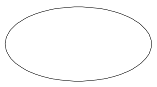
Staff Member



The staff member is drawn as a stick man with a name written below it. The member carries out the use cases they are associated with. An administrator (secretary) could also perform a use case in the role of a member.

Use cases

Each use case is something that the member would be able to do. It represents a use of the system that can be performed by a user. Each use case in a diagram is drawn as a bubble containing the name of the use case.



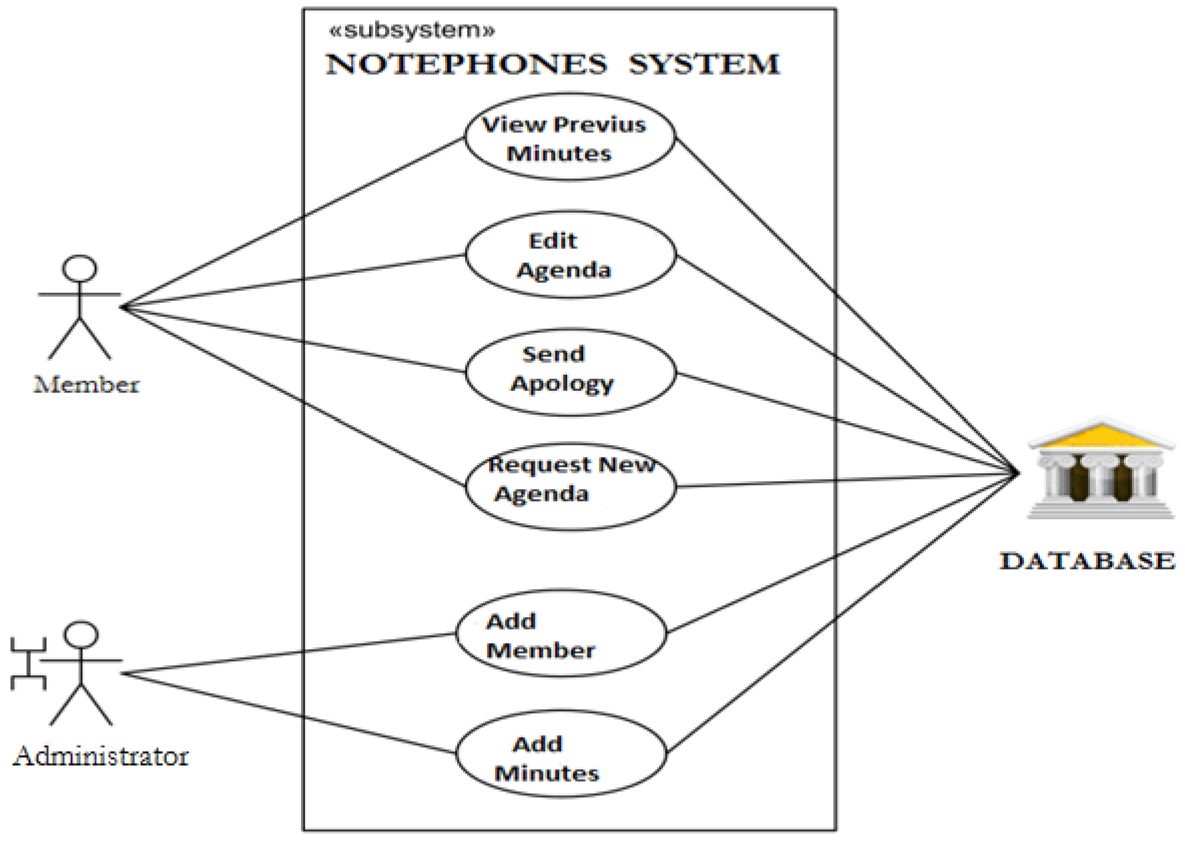


Figure 8 :Use Case Diagram

A staff member can request new agendas from the database, edit and view minutes and send an apology for not attending the meeting (see figure 8). The administrator can execute the same instructions as a staff member but is allowed two more functions. The benefits of generalization eliminate duplicate behaviour and attributes; this will ultimately make the system more understandable and flexible.

Conclusion

This chapter has specified the user interface, how it will look like and also how it is expected to behave. This chapter has also explained the interaction of the user with the system. In the next chapter, the object oriented analysis (OOA) design will be specified.

C h a p t e r 4

HIGH LEVEL DESIGN

Introduction

In the previous chapter the user interface specification was covered and also how the user will interact with the system. In this chapter, the objects will be analysed in terms of an object-oriented view of the problem. The relationship between the objects will be shown while the class diagrams will display the attributes and methods of each class.

Class Diagram

Structure diagrams are useful throughout the software lifecycle. The class diagrams will be used to document the system's soon-to-be-coded classes. The purpose of the class diagrams is to show the objects being modeled within the system.

One good approach to application design is to look at the different objects that will be involved in the application. In this case, an object is simply something to store information about. Designing an application based on its component entities (objects) is a basic principle of object oriented programming. In this application, the main entity is a member (regular user). There are two different kinds of users: regular user and an administrator. Two approaches can be taken to represent these different types of user. We could describe two different entities, one for each type of user, or we could just have one user entity with an attribute that defines its type. The latter approach will be used in this application.

When different entities have a common base but still have major differences, it can be beneficial to devise of a base entity that is inherited by the other entities. For instance, while all vehicles have a common foundation in that they all have wheels and move forwards and backwards, they can still be very unique. Each vehicle has properties common to all vehicles, such as tyres, steering wheels, and radios. At the same time each vehicle is also quite unique. Cars are smaller and usually carry more passengers than trucks. Trucks carry fewer passengers but have more cargo space. Taxis are similar to cars but carry fare-paying passengers and have CB radios and fare calculating equipment. We could even have shown a taxi as being a sub-entity of a car, inheriting all the traits of a car and then adding special traits of its own.

The only unique property for the user entity of this application is whether they are an administrator or a regular user. With such a small difference between the different users, it is better to create only one entity with a property that designates the user's administrative responsibilities, which is the reason why the second approach is used to represent these different types of user. Other entities in our application include minutes and events (Login). It is apposite to look at each entity in detail:

Member Table

A member is any user in this system. Users can login to the application, set their status, and view the status of others. A user has the following properties: Name, Phone Number, Username, Password, Administrator (Yes/No), Start/end Date (see table 2 below).

A primary key is a column in the member table that will uniquely identify every row in the table. In the member table, column, called ID_no, is the primary key. This column will be an integer and will be incremented automatically by SQLCe through the use of the AUTO_INCREMENT keyword.

However an attempt was made to use Name or perhaps Phone Number as a primary key, but a company may have more than one Allen Mwangonde, or even at times employees may have to share phones. While it is possible to have more than one column together as a composite primary key (such as making the primary key the combination of Name and Phone Number), there is need to be certain that every entry will be unique. For instance, such certainty will not be achieved when the two users with the same name happen to share the same phone.

Table 2 : Member

S.no.	**Column Name**	**Data Type**	**Length**	**Description**
1	*ID*_no	integer	5	Unique identification of a member
2	Name	Text	20	Name of a staff member
3	Username	text	20	Login username
4	Password	text	15	Login password
5	Start Date	Date/Time	8	Date when member is registered
6	End Date	Date/time	8	Date when member is expelled
7	Mobile number	Text	20	Mobile number of the staff member
8	Status	Text	20	Permanent/temporary

Minutes Table

The minutes table contains incoming and outgoing minutes that are shared among staff members In this system, minutes will be sent to every staff member's mobile phone. The system will match all the dates of the meeting and months stored in this table with the current date and month. If matched, it will send minutes. The minutes table has the following properties: agenda, date of meeting, mobile number, and name of minutes (see table 3). In this table, column, called Sno is the primary key. This column will be an integer and will be incremented automatically by SQLCe through the use of the AUTO_INCREMENT keyword.

Table 3: Minutes

S.no.	**Column Name**	**Data Type**	**Length**	**Description**
1	Sno	Integer	5	Unique identification of the minutes
2	Agenda	Text	100	Names of agenda of the minutes
3	DateOfMeeting	Date/Time	8	Date of conducting the meeting
4	Mobile number	Text	20	Mobile number of the staff member
5	Minutes	Text	500	List of minutes discussed in a meeting

Login Tables

An event is simply a change in a user's status. A user can login or fail to login (an event of whether or not this user is a registered staff member in the system). In addition, this application will only allow a user to login as an administrator or a regular user.

There are multiple options available when working on a login tables, but typically it comes down to one of two options (**Reed**, **1999**):

- Store usernames and passwords in a Structured Query Language (SQLCe/SQL) table, and use a single user account for all instances of the application that connect to SQLCe. Each instance of the application will connect to one server account, and then validate the user against the tables that have been created; or

- Create SQLCe/SQL users for each account, and let SQLCe/SQL handle authentication and privilege management.

These approaches both have their respective benefits and drawbacks. When you build your own login system, you potentially have greater control over users, groups, and privileges. You have simpler permission management because only one native SQLCe user account is needed. On the other hand, there is more potential for security problems because something you create yourself does not have the review process seen in the SQLCe server. In addition, you have a security weakness in the fact that every client installation has a copy of the central user account for connecting to SQLCe, and that account has to have the maximum permissions needed for an administrative user. These concerns are not so great when you are developing a web application, because it is easier to manage the one web instance than a collection of desktop applications.

When you use the security built-into SQLCe, you inherit a robust privilege management system that has been thoroughly reviewed and tested. You can limit privileges on a fine-grained level, and even assign different permissions based on which host the user connects from. The tradeoff is one of complexity: making the most of the built-in SQLCe authentication is more difficult than building one of your own, and you must add a user to the SQLCe server every time you want to add a new user to your application.

Because this is not a web-based application, it will be built using the built-in SQLCe privilege system. It will prompt users for a server address, username, and password (see table 4). The server will set a variable to the address of the SMS Messaging Server that a user registered his/her account on such as South African server (www.Bulksms.co.za), United Kingdom server (www.Bulksms.co.uk), United Kingdom server (usa.Bulksms.com), International server(www.Bulksms.net), and the German server (www.Bulksms.de). SMS Messaging Server is an SMS messaging framework that enables an application to send, receive and process SMS- and e-mail messages. The SMS Messaging Server can be well integrated into VBScript environments. This document will not describe how the SMS Messaging Server can be integrated into your own projects (for further information use any of the page links in this paragraph).

Table 4: Login

S.no.	**Column Name**	**Data Type**	**Description**
1	UserName	Text	Member's identification name
2	Password	Text	Password for a registered user
3	Server	Text	Database server address

Since the entities are defined, they can be added to a diagram (see figure 9).

Data Flow Diagram (DFD)

The main entities which will influence this application are the member and minutes (see figure 9). The login entity will be used as an event for database tables (this is not included in figure 9). In addition, the login entity will also help form classes, a building block of our final application (the use of classes in application programming is referred to as object oriented programming).

When developing an application, it is often enough to quickly sketch out the entities and their properties and relationships using a system like the Unified Modeling Language (UML).

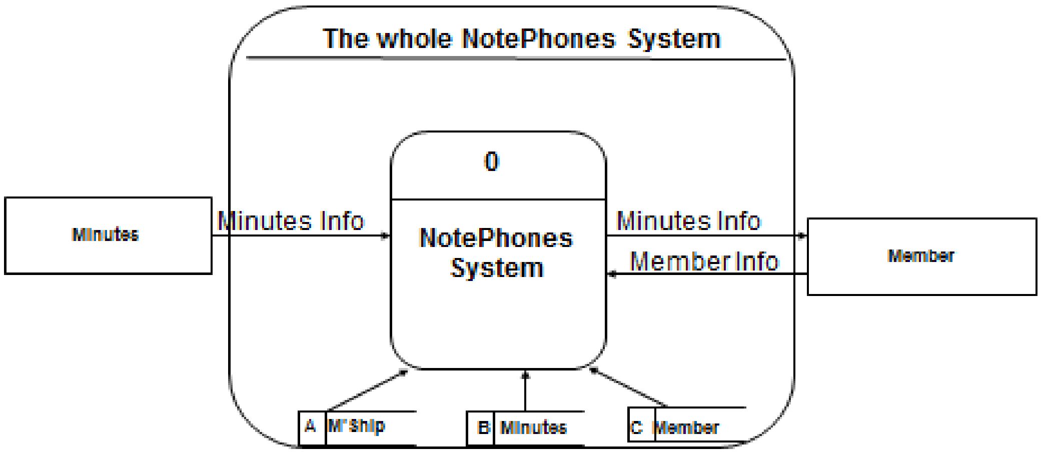


Figure 9 :Context Diagram

Figure 10 below describes all six use cases identified using the use case diagram in Figure 8 above. It provides a clear picture on how the two main entities (from figure 9) will perform these use cases. As noted earlier, a use case is something that the member would be able to do; it represents a use of the system that can be performed by a user.

It will be seen from figure 10 that a member can request minutes by name from the system and the minutes will be sent to him/her as a report. The user will also be able to modify the database by deleting or updating it. In addition, the user can edit the database options for the member database, minutes database and login database, and can also migrate one or more of the above databases to MS SQL Server or SQLCe.

The arrows in figure 10 indicate the direction of the data's flow. Double arrows indicate information flowing in either direction. The rounded rectangles numbered 1 to 5 represent the use of the system that will be performed by a member whereas the two rectangles represent the two objects discussed in figure 9. The database is represented by D1 where all incoming and outgoing information is stored.

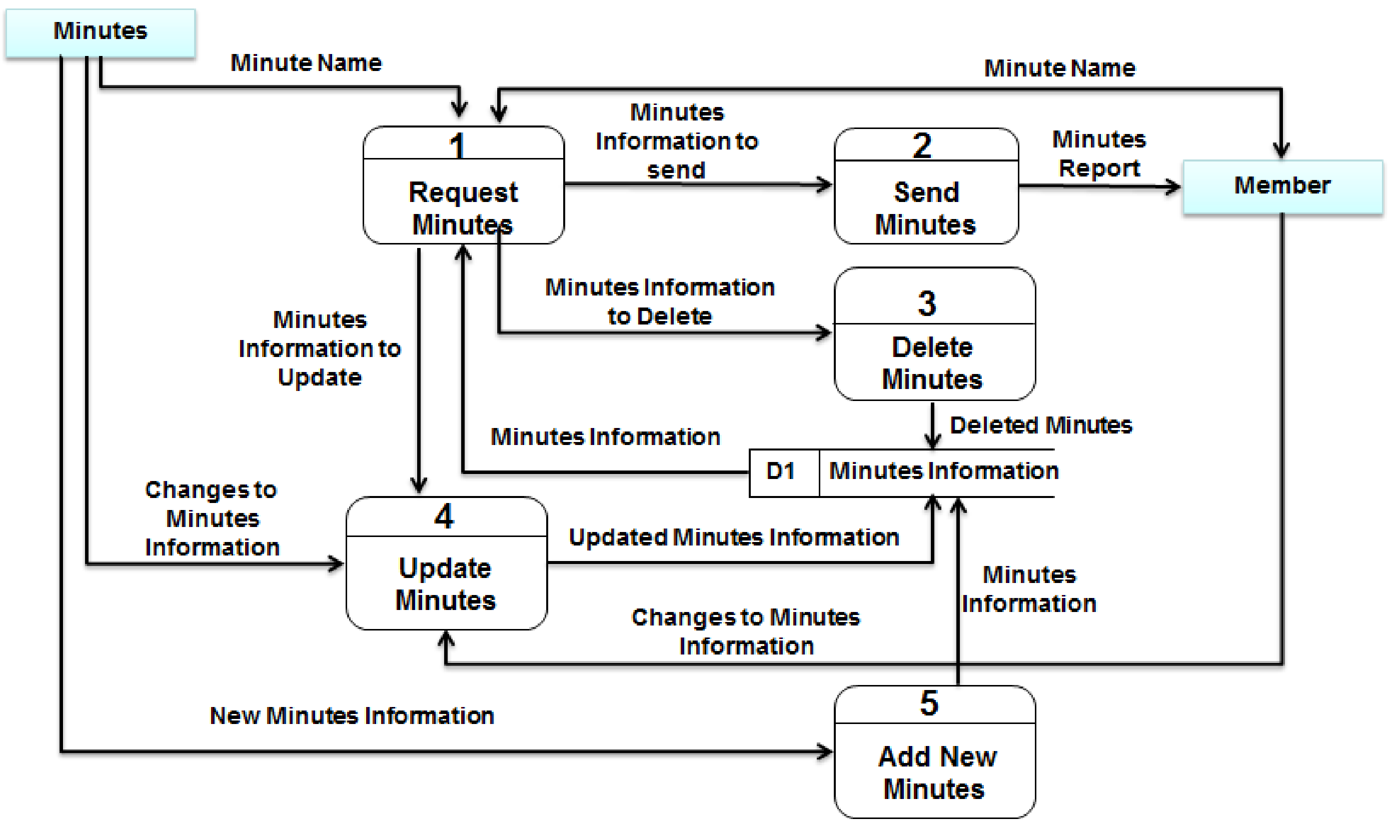


Figure 10 :Level 0 DFD

Flows to & from data stores

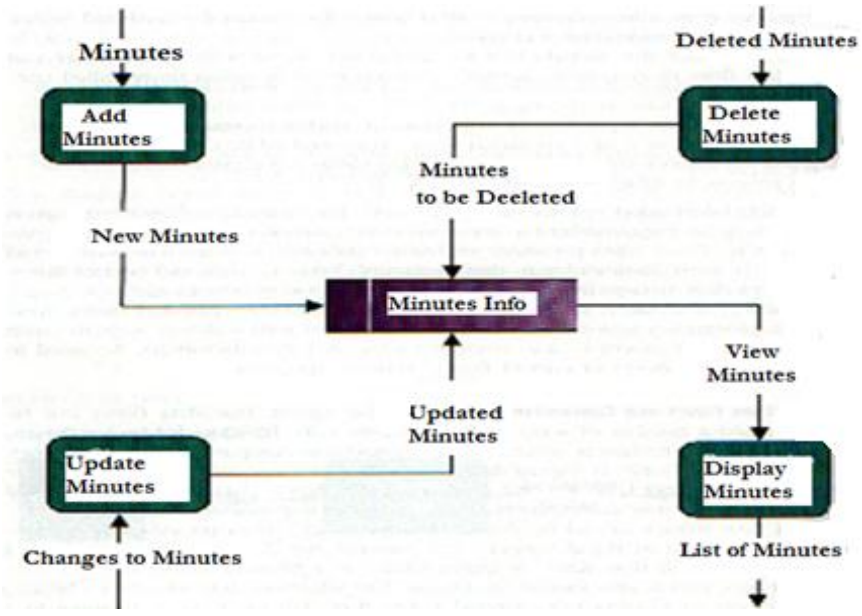


Figure 11 : Flow Diagram

The flow diagram in figure 11 shows the flow of minutes between objects in the system (see Figure 9). After a meeting, the minutes are entered into the NotePhones system which sends them to the staff members for viewing through mobile devices such as cell phones. The staff member views the minutes and can request for minutes. The secretary or administrator can edit or update the minutes at any time. The system generates a database ('minutes info') where the details of the minutes and members are stored.

Conclusion

In this chapter the high level design was discussed. The various classes involved and how they are expected to interact with each other was shown by means of diagrams. In the next chapter, the system design will be explained in terms of pseudocode for each of the objects.

C h a p t e r 5

LOW LEVEL DESIGN

Introduction

The previous chapter looked at object oriented analysis which described the object-oriented view of the problem, where every object was described. In this chapter, the object-oriented design of the system is given as an outline of the class functions and the pseudocode for each object of the system as analysed in the previous chapter.

Details of Class Functions and Pseudocodes

The NotePhones Application is a collection of classes. Each class has several properties and functions ('methods') defined. A class is a container for data and code. The data within the class can be accessed with properties. The code is referred to as methods. A method in a class can be a procedure that performs some sort of operation on the data within the class. Or a method could be a function that performs some operation on the data, and returns that data back from the class. In Visual Basic.NET, to be able to call a method from an instance of this class, the method must be declared **Public**. If a method is declared **Private**, only other methods within the class can call that method.

The following three classes will be used in this application:

Class: Minutes

This class represents the Minutes Database, i.e. the collection of all Minutes Records in the database. Please read Minutes Class for a complete overview of all properties and methods of this class.

Class: Member

This class represents a single Member record in the Member Database. The class provides properties, where each property represents a single record field. Please read Member Class for a complete overview of all properties and methods of this class.

Class: Login

This class defines all registered users who will be able to login and use the NotePhones Application. There are properties for username, password, etc. Please read Login Class for a complete overview of all properties and methods of this class.

Minutes Class

Table 5: Minutes class

Function	**Description**
Open	Opens the Minutes database
Load	Loads a message
MenuAdd	Enables textboxes for data entry Gives focus to First text box (txtname)
MenuExit	Exits the Application
ToggleTextBoxReadOnly	Gives focus to Detail View screen but does not activate it Enables Add Record button Toggles scroll bar to avoid data loss Fires the initialization of Detail View
addNewData	Refills the table with new row
MenuDelete	Deletes minutes
CheckMinutes	Checks the current date ,if there any messages to send ,send messages
Close	Closes the minute database

Open function

Open the minutes database. You must open the minutes database before you can perform any operation on the minutes database, like counting records, creating new records, deleting records, etc. When you have finished accessing the database, you must call close in order to close the database.

Parameters:

ReadWrite (Optional, Boolean). Default is true; false opens the database for read only.

Return value:

Always 0. Check LastError property to see if the function was completed successfully.

Pseudocode:

```
If NOT MINUTESDB.STATE = open THEN Open
ELSE IF MINUTESDB.STATE = open THEN Fill Table AND close
End If
```

Close function

Close the minutes database. You must call this function to close the minutes database that was open by the open call. You can even call this function if a preceding open was not completed successfully (the function will then simply be ignored).

Parameters:

None.

Return value:

Always 0. Check LastError property to see if the function was completed successfully

Pseudocode:

```
If(MINUTESDB.LastError <> 0 )THEN Quit
MINUTESDB.Close
End If
```

Load function

Load a message from the minutes database.

Parameters:

Minutes ID (sno) - Record ID of the message in the minutes database

Return value:

A new message object. Check LastError property to see if the function was completed successfully

Pseudocode:

```
IF (MINUTESDB.LastError <> 0 ) THEN
     Quit
ELSE
     SET objMessage AS objMessageDB THEN
     IF formLoadFlag = False THEN
          FILL the data from adapter to dataset
          SET DateTimePicker.MaxDate = Today
          formLoadFlag
    ELSEIF formLoadFlag = True
        SELECT Case bdTabControl.SelectedIndex
        Case 0
               SET MenuModify.Enabled = True
```

```
                        SET MenuDelete.Enabled = True
                        SET the scroll bar value
                        IF MINUTESDataSet.MINUTES.Rows.Count > 0 THEN
                            ShowDataInForm
                        ELSEIF MINUTESDataSet.MINUTES.Rows.Count < 0 THEN
                            showMessage("Database is empty. Please fill records.")
                        End If
                        Case 1
                            SET MenuModify.Enabled = False
                            SET MenuDelete.Enabled = False
                            IF MINUTESDataSet.MINUTES.Rows.Count < 1 THEN
                                showMessage("Database is empty. Please fill records.")
                            End If
                       END SELECT
                  END IF
             END IF
```

MenuAdd Function

Create a new message in the message database.

Parameters:

None.

Return value:

A new minutes object.

Pseudocode

```
    IF ToggleTextBoxReadOnly =False THEN
    Enable textboxes for data entry AND
    SET cmdAdd.Text.visible = "Update" OR “Modify”
    End IF
    If cmdAdd.Text = "Update" Then
    CALL addNewData() THEN SET
    cmdAdd.Visible = False
    cmdCancel.Visible = False
     Make the textboxes read only
    ELSEIF cmdAdd.Text = "Modify" THEN
```

```
updateRecord in MINUTESDB table
 SET cmdAdd.Visible = False
SET cmdCancel.Visible = False
Make the textboxes read only

END IF
```

AddNewData Function

Insert records into the minutes database.

Parameters:

Record ID of the minutes in the minutes database

Return value:

Message object.

Pseudocode:

```
INSERT Records into MINUTESTable
Fill MINUTESDataSet.MINUTES
If record count > 0 THEN
 ShowMessage ("Entry added successfully.")
 END IF
```

MenuDelete function

Delete records from the minutes database. A filter is applied. When an empty string is passed as filter, all messages are deleted.

Parameters:

Filter (String) A minutes filter. Pass an empty string to filter all minutes.

Return value:

Message object.

Pseudocode

```
SET  ans As Integer = MsgBox("Are you sure you wish to delete the current record?",
MsgBoxStyle.YesNo, Me.Text)

IF ans = yes THEN
Delete records from MINUTESTable .MINUTESDataSet
            ShowMsgbox ("Entry deleted successfully.")
```

```
End If
```

CheckMinutes function

Checks records from the minutes database. A flag is applied. When a date string is passed as flag, it matches all the minutes and months stored in the server with the current date and month. If matched, it will send an automated SMS to the intended users.

This function addresses the need to share minutes on mobile devices provided by the NotePhones application. This application module will make collaboration among staff members possible by sending automatic reminders to all participants to make them make necessary preparations for the meeting.

Parameters:

Flag (String). A minutes flag. Pass an empty string to filter all minutes.

Return value:

Message object

Pseudocode

This code matches all the meeting dates and months stored in MINUTES table with the current date and month. If matched, send SMS, else skip. SmsMessage is an inbuilt library for sending messages.

```
For i = 0 To MINUTESDataSet.MINUTES.Rows.Count – 1
If MINUTESDataSet.MINUTES.Rows.Count > 0
    CheckMinutes
    If (Date.Month.Year = Date.Now.Month.Year ) THEN
        Match SmsMessage(MINUTESDataSet.MINUTES(i).mobileNumber, AND
        MINUTESDataSet.MINUTES.Agenda & vbCrLf & txtMessage.Text
        Send SmsMessage
   Else
        Skip
  End If
End IF
```

Member Class

Table 6 : Member class

Function	**Description**
Open	Opens the member database
Close	Closes the member database
Insert	Adds new members to the member database
Update	Modifies records in the member database
Delete	Enables the user to delete some records from the member database

Open function

Open the member database. When you are finished accessing the database, you must call close in order to close the database.

Parameters:

ReadWrite (Optional, Boolean). Default is true; false opens the database for read only.

Return value:

Always 0. Check LastError property to see if the function was completed successfully.

Pseudocode:

```
If NOT MENBERDB.STATE = open THEN Open
ELSE IF MEMBERDB.STATE = open THEN Fill Table AND close
End If
```

Close function

Close the member database. You must call this function to close the member database that was open by the open call.You can even call this function if a preceding open was not completed successfully (the function will then simply be ignored).

Parameters:

None.

Return value:

Always 0. Check LastError property to see if the function was completed successfully

Pseudocode:

```
If(MEMBERDB.LastError <> 0 )THEN Quit
MEMBERDB.Close
End If
```

Insert function

Insert a new member in the member database.

Parameters:

New table records

Return value:

A new message object. .

Pseudocode

```
SET SQLCeCommand as ("INSERT into MEMBERDB")
If NOT MENBERDB.STATE = open THEN Open
ELSE IF MEMBERDB.STATE = open THEN Fill Table AND close
End If
SET rowsEffected As Integer
CALL SQLCeCommand
IF (rowsEffected > 0) THEN
        Close MEMBERDB
        Show message "Records inserted successfully"
ELSE
        Show message "Records failed"
End If
```

Update Function

Uptade records from the Member Database.

Parameters:

Record ID of the minutes in the Member database

Return value:

Message object.

Pseudocode:

```
UPDATE Records MEMBERTable
Fill MINUTESDataSet.MEMBER
If  record count > 0 THEN
 ShowMessage ("Entry updated successfully.")
 END IF
```

Delete function

Delete records from the member database. A filter is applied. When an empty string is passed as filter, all selected records are deleted.

Parameters:

Filter (String).

Return value:

Message object

Pseudocode

```
SET SQLCeCommand as ("DELETE from MEMBERDB”)
If NOT MENBERDB.STATE = open THEN Open
ELSE IF MEMBERDB.STATE = open THEN Fill Table AND close
End If
SET rowsEffected As Integer
CALL SQLCeCommand
IF (rowsEffected > 0) THEN
        Close MEMBERDB
        Show message "Records deleted successfully"
 ELSE
        Show message "Records failed"
End If
```

Login Class

Table 7: Login class

Function	**Description**
loginform	Ensures a user is valid.

Loginform function

Accept or deny user to have access the application by login or failing to login. The loginforn function verifies the username and password with a database which contains all usernames and passwords of all members. If it finds that the username and the password given correspond. with those in the login table,then the login form connect the user to the application.

Parameters:

String (username or passowrd). Default is true; false opens the database for read only.

Return value:

Form object. Check LastError property to see if the function was completed successfully.

Pseudocode:

```
If NOT MENBERDB.STATE = open THEN Open
ELSE IF MEMBERDB.STATE = open THEN Fill Table AND close
End If
IF nothing has been submitted THEN
    the login form is displayed
ELSE
     IF something has been submitted THEN
         check if the user that was entered is valid
             IF the user is valid THEN
             set the cookies and session variables to show that the user has logged in
             redirect the user to the home page
             ELSE IF the user isn't valid, THEN
                 display any necessary errors
                 display the login form again
             END IF
        END IF
  END IF
```

The prototype application

The following steps would allow access to the system:

- Open mobile device emulator in visual studio

- Load the index file

- Select option from 1 to 4 from the main menu and follow the instructions on the screen.

Conclusion

This chapter has looked at the object-oriented design , giving details of the class methods and the pseudocodes of each class method of the system. It has also analysed the functions which show how the user will interact with the system. The chapter has further outlined the architectural framework and a prototype software system that will improve minutes sharing processes using multi-channel mobile-based technology to connect students, faculty and the research community, irrespective of the communication device (WAP phones, PC/Laptop, Pocket PC and PDAs) owned by any group member. The next chapter will present a detailed documentation of testing.

C h a p t e r 6

IMPLEMENTATION

Introduction

In the previous chapter, the object-oriented design of the system was given as a detailed outline of the class functions/methods and the pseudocode for each class method of the system. It also analysed the functions which show how the user will interact with the system. The previous chapter holds the introductory keys to this chapter, which are now detailed.In this chapter, a method of implementation is discussed which is the cell phone implementation of the application. The source code of the programs is fully documented.

Software Deployment

The deployment of this VB.net application works with the SQL database, SMS functions added to it, and SMS gateway. It consists of two applications:

i. Mobile application – allows sending SMS through a gateway to the server(indicated by blue arrows in the figure 12 below)
ii. Application on server receives SMS from the gateway and updates the database, or application on server sends SMS through the gateway to mobile phone.(indicated by green arrows in the figure12 below)

In this configuration, to allow the mobile application to send automatic SMS reminders, the minutes records will have to be manually inserted into a database table by the administrator, and to receive automatic SMS reminders, the application is programmed with intervals of time to check the database table to see if new records are inserted into it (see figure 12).On this figure, the SMS gateway (OZEking) will shares a database with the application. The SMS gateway will be responsible for taking the SMS messages from the database to the mobile network. It will connect to the mobile network through a GSM phone attached to a computer or through the Internet. Incoming messages will also be handled by this SMS gateway. If an SMS message comes in, it will insert it into the database.

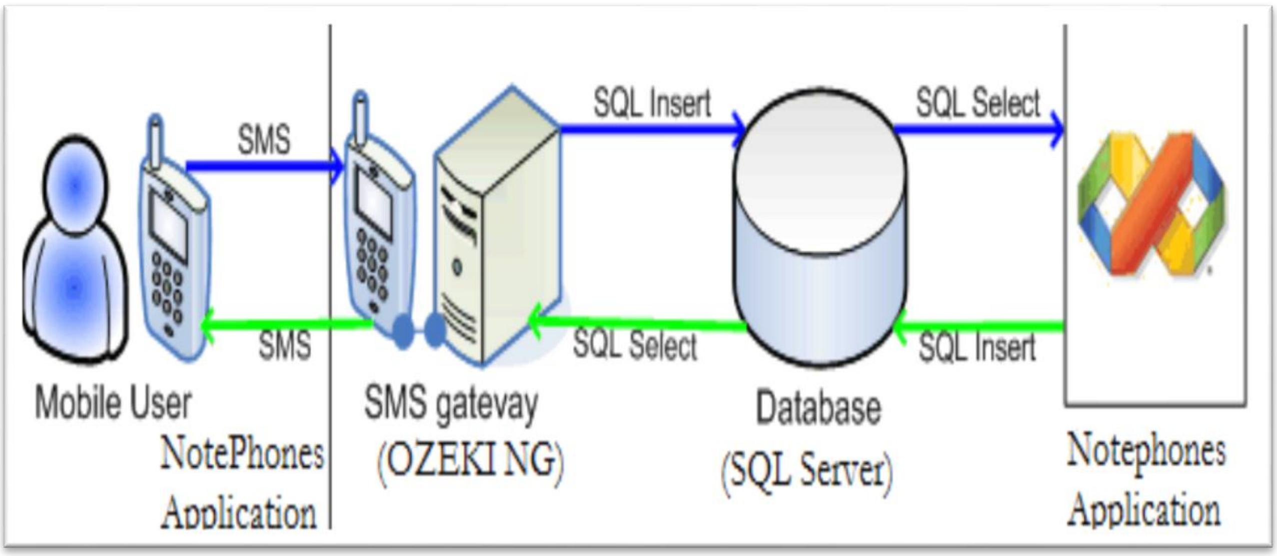


Figure 12 :How to send / receive SMS from NotePhones system through SMS Gateway

Progress

Interface changes

There are not much changes made to interface of this application as compared to the interface given in chapter three.

Changes to the Tools and Languages used

The number of tools and languages used has increased.

i. OZEking - SMS Gateway described above. There are a number of SMS gateway which would comply with this application's requirements in this system and OZEking gateway was selected
ii. Windows Mobile SDK-This is a software development kit for windows applications. The SDK enables developers to build applications using the VB.net programming language. It includes a cellular emulator which is used for testing of applications before being imported to the real mobile phone device.

    Cellular Emulator is a software-based emulator to help developers and testers develop and test Windows Mobile platform applications. Cellular Emulator replaces the radio module, FakeRIL, in both development and test environments. The advantage of Cellular Emulator is that it provides not only voice but also data connectivity. Moreover, the Cellular Emulator is a powerful tool to test various applications under different wireless network conditions in GSM/GPRS and/or UMTS networks. (Reed, 1999)

Cellular Emulator, or CellEmu, offers the following features:

- CellEmu is a UI-driven application desktop application, making it easy to operate.
- CellEmu device images incorporate an actual production radio driver.
- CellEmu supports PPP data connection.

iii. .NET Compact Framework- The .NET Compact Framework is a hardware-independent environment for running programs on resource-constrained computing devices - it is a subset of the .NET Framework class library and also contains classes exclusively designed for it. It inherits the full .NET Framework architecture of the common language runtime and managed code execution.

iv. SQL –The SQL Server Compact Edition is a lightweight relational database for use as a compact local database on a PC, Tablet PC, Windows Mobile powered, or Windows CellEmu-based device. Developers will be able to use the same database to develop occasionally-connected applications consistently across all Microsoft client platforms. The database can be used for local storage as well as for data synchronization with other editions of SQL Server.

v. Device emulator- Device Emulator provides greatly improved performance at executing ARM instruction. It reduces cold-boot time and improves application execution speed. It provides high resolution input, the ability to change emulator options on-the-fly and has added driver support in the BSP for peripherals such as battery and headset/speakerphone with Windows Mobile 6 emulator.

vi. VB.Net- Visual Basic .NET(VB.NET), is an object oriented programming language that can be viewed as an evolution of the classic visual basic(VB), which is implemented on the .NET.. The use of integrated development environment (IDE) which is based on the Eclipse IDE. It enables developers to quickly and efficiently create code, test and deploy software for the Windows operating system. (Reed, 1999)

vii. A PC and Nokia E 710 mobile phone

Challenges

- Every programming and software engineering project has hardships and this one was no exception. There were a lot of technical problems encounter throughout the development of the project. They were technical challenges when it came to version control, when identifying the correct windows mobile phone which was most suitable for this project.

- During the development and implementation of this application ,the device emulator was used and since sending and receiving SMS messages is a real time activity and emulators do not support real time message sending, this presented a challenge. It was not easy to debug on the actual mobile phone, but the device emulator tool makes it easy to test applications which interact with cellular phone based functionality. One advantage of using this tool to test SMS sending applications is that it avoids the charges typically associated with sending SMS messages via real devices.

- . The most testing challenge was identifying and configuring the SMS gateway suitable for the application to be able to send /receive SMS text messages to alert staff members of updated minutes information, such as the potential for the staff member to chair the next meeting.

Code Documentation

The source code given in this section shows the basic functionalities of the project on windows. It gives an outline of the three classes and methods discussed in the previous chapter

Login script documentation

The following code shows how the login form works.At design time, the form's AcceptButton and CancelButton properties were set to btnLogin and btnCancel. That lets those buttons fire when the user presses Return or Escape.

When the user clicks the Cancel button, the code sets the form's DialogResult property to DialogResult.Cancel. That automatically closes the form and returns DialogResult.Cancel to the calling routine (if the form is displayed using the ShowDialog method).

When the user clicks the OK button, the code does some simple validation of the user name and password. If the user name and password combination is valid, the code sets the form's DialogResult property to DialogResult.OK. That automatically closes the form and returns DialogResult.OK to the calling routine.

```
Private Sub btnCancel_Click(ByVal sender As Object, ByVal e _
    As System.EventArgs) Handles btnCancel.Click
    DialogResult = DialogResult.Cancel
End Sub


Private Sub btnOK_Click(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles btnOK.Click
```

The following code makes sure that the user entered username and passowrd.

```
    If txtUserName.Text.Length = 0 Then
        MsgBox("You must enter a user name", _
            MsgBoxStyle.Exclamation)
        txtUserName.Focus()
    ElseIf txtPassword.Text.Length = 0 Then
        MsgBox("You must enter a password", _
            MsgBoxStyle.Exclamation)
        txtPassword.Focus()
    ElseIf PasswordInvalid(txtUserName.Text, _
        txtPassword.Text) Then
        ' The user name or password is invalid.
        MsgBox("User name/password invalid", _
            MsgBoxStyle.Exclamation)
        txtUserName.Focus()
    Else
        ' The user name and password is valid.
        DialogResult = DialogResult.OK
    End If
End Sub
```

The following code is for validating the user login if the selection mode is set to Stored Procedures. It will execute the specified stored procedures with the username and password parameters and return true if user exists and false when not exists

```
Private Function PasswordInvalid(ByVal user_name As String, ByVal password As String) As Boolean Try
    Dim dsUser As New DataSet
    Dim Conn As New SqlConnection(Me.ConnectionString)
    Conn.Open()
    Dim myCmd As New SqlCommand(Me.StoredProcedureName, Conn)
    Dim PrmUserId As SqlParameter = myCmd.Parameters.Add(Me.UserIdParameter, _
        Me.TxtUsername.Text.ToString().Trim())
    Dim PrmPassword As SqlParameter = _
        myCmd.Parameters.Add(Me.UserPasswordParameter, _
    Me.TxtPassword.Text.ToString.Trim())
    myCmd.CommandType = CommandType.StoredProcedure
    Dim MyDa As New SqlDataAdapter(myCmd)
    MyDa.Fill(dsUser)
    Conn.Close()
    If dsUser.Tables(0).Rows.Count > 0 Then
      RaiseEvent OnUserExists(dsUser.Tables(0).Rows(0))
      Return True
    Else
      Return False
    End If
  Catch ex As Exception
    MessageBox.Show("Error While Executing the Stored Procedures", _
        "Executing Stored Procedure Error", _
        MessageBoxButtons.OK, MessageBoxIcon.Exclamation)
    Return False
  End Try
End Function
```

The following code is the Function that will be invoked to check the database

```
Private Function CheckLogin() As Boolean
  Try
    If Me.SelectionMode = QueryType.SelectQuery Then
      If Me.ExecuteSelectQuery = False Then
        RaiseEvent OnUserInvalid()
      End If
    Else
      If Me.ExecuteStoredProcedure = False Then
        RaiseEvent OnUserInvalid()
      End If
    End If
  Catch ex As Exception
  End Try
End Function
```

The following code shows the Stored Procedure which were used to validate the login procedure

```
Create procedure dbo.LOGINSP_ValidateLogin
    @Userid varchar(50),
    @UserPassword varchar(50)
    as
    BEGIN
        select  iUser_id as userid,
            vUser_LoginName LoginName,
            bUser_Main as Type
        from TBLUSERTABLE
        where
            vUser_LoginName =@Userid
            and vUser_Password=@UserPassword
    END
```

Agenda creation script documentation

The following code is the Function that will be invoked to check the database of the minutes records

```
Private Function getResult4mQuery(ByVal strQuery As String) As Integer
Dim cn As New Data.SqlServerCe.SqlCeConnection()
Dim cmd As New Data.SqlServerCe.SqlCeCommand(strQuery, cn)
im dr As Data.SqlServerCe.SqlCeDataReader
        cn.Open()
        cmd.CommandType = Data.CommandType.Text
        dr = cmd.ExecuteReader()
        dr.Read()
        getResult4mQuery = dr(0)
        cn.Close()
End function
```

The following code checks records from the minutes database. A flag is applied. When a date string is passed as flag, it matches all the minutes and months stored in the server with the current date and month. If matched, it will send an automated SMS to the intended users.

```
Private Sub checkminutes()
        Dim i As Integer = 0

        Try
            For i = 0 To MINUTESDataSet.MINUTES.Rows.Count - 1
                If MINUTESDataSet.MINUTES(i).dateOfMeeting.Day = Date.Now.Day And
MINUTESDataSet.MINUTES(i).dateOfMeeting.Date.Month = Date.Now.Month Then
                    Dim sms As New SmsMessage(MINUTESDataSet.MINUTES(i).mobileNumber, _
                    "Hi " & MINUTESDataSet.MINUTES(i).Agenda & vbCrLf & txtMessage.Text)

                    sms.To.Add(New Recipient(txtMobileNumber.Text))
                    sms.Body = txtMessage.Text
                    'sms = New SmsMessage(txtMobileNumber.Text,txtText.text)
                    sms.Send()
                    Dim sms2 As New SmsMessage("+999999999999", _
                     "Hi " & MINUTESDataSet.MINUTES(i).Agenda & vbCrLf & txtMessage.Text)
            sms2.Send()
          End If
       Next
    Catch ex As Exception
       showMessage("The database is empty. Fill some entries here.")
    End Try
 End Sub
```

Staff member script documentation

The following code insert new records in he member table database

```
Public Function InsertRecord(ByVal Table As String, ByVal Fields As String, ByVal Values As String)
          ' INSERT INTO Country (Code, Name) VALUES ('AAA','Test Name')"
          Dim sqlCmd As SQLCeCommand = New SQLCeCommand("INSERT INTO " & Table & "(" &
  Fields & ") VALUES (" & Values & ")", RemoteSQLCeConnection)
          Try
             If Not RemoteSQLCeConnection.State = ConnectionState.Open Then
                RemoteSQLCeConnection.Open()
             End If
             Dim rowsEffected As Integer = 0
             If RemoteSqlCeConnection.State = ConnectionState.Open Then
                rowsEffected = sqlCmd.ExecuteNonQuery()
             End If
             If (rowsEffected > 0) Then
                Remote SqlCeConnection.Close()
                Return "Inserted"
             End If
             RemoteSQLCeConnection.Close()
          Catch ex As Exception
             'MsgBox(ex.Message)

             RemoteSQLCeConnection.Close()
             Return "Failed"
          End Try
          If RemoteSQLCeConnection.State = ConnectionState.Open Then
             RemoteSQLCeConnection.Close()
          End If
          Return "Failed"
       End Function
```

The following code updates the staff member records

```
Public Function UpdateRecord(ByVal Table As String, ByVal fields As String, ByVal where As String)
          ' Dim sqlCmd As SQLCeCommand = New SQLCeCommand("UPDATE" & Table & "SET
Name='Test2' WHERE Code='AAA'", RemoteSQLCeConnection)
          ' Dim sqlCmd As SQLCeCommand = New SQLCeCommand("UPDATE" & Table & "SET
Name='Test2',Code='me' WHERE Code='AAA'", RemoteSQLCeConnection) for two or more
          Dim sqlCmd As SQLCeCommand = New SQLCeCommand("UPDATE " & Table & " SET "
& fields + " WHERE " + where, RemoteSQLCeConnection)
          Try
             If Not RemoteSQLCeConnection.State = ConnectionState.Open Then
                RemoteSQLCeConnection.Open()
             End If
             Dim rowsEffected As Integer = 0
```

```
            If RemoteSQLCeConnection.State = ConnectionState.Open Then

                rowsEffected = sqlCmd.ExecuteNonQuery()
            End If
            If (rowsEffected > 0) Then
                Return "Updated"
            End If
            RemoteSQLCeConnection.Close()
        Catch ex As Exception
            'MsgBox(ex.Message)

            RemoteSQLCeConnection.Close()
        End Try
        Return "Failed"
    End Function
```

The following code deletes some records from the database

```
Public Function DeleteRecord(ByVal Table As String, ByVal where As String)
        ' DELETE FROM Country WHERE Code ='AAA'
        Dim sqlCmd As MySqlCommand = New MySqlCommand("DELETE FROM " & Table & "
WHERE " & where, RemoteMySqlConnection)
        Try
            If Not RemoteMySqlConnection.State = ConnectionState.Open Then
                RemoteMySqlConnection.Open()
            End If
            Dim rowsEffected As Integer = 0
            If RemoteMySqlConnection.State = ConnectionState.Open Then
                rowsEffected = sqlCmd.ExecuteNonQuery()
            End If
            If (rowsEffected > 0) Then
                Return "Deleted"
            End If
            RemoteMySqlConnection.Close()
        Catch ex As Exception
            RemoteMySqlConnection.Close()
        End Try
        Return "Failed"
    End Function
```

Conclusion

In this chapter an approach for implementing the project was discussed in detailed together with the source code documentation. The code documentation for the processes performed to create an agenda and send minutes. The VB.NET scripts consisted of the methods,SQL queries and comments. This chapter has also discussed how the project was implemented on the mobile phone emeulator.The next chapter describes how the system will be tested to check whether the functionality is correct.

Chapter 7

SYSTEM TESTING

Introduction

The previous chapter focused on the implementation of the application. It gave a detailed documentation of the code used and explained how each part works to make the application usable, functional and how each component contributes to the project. In this chapter, system testing will be carried out to verify and ensure that the system meets its design specifications and requirements. This chapter will discuss the usability, functionality and the performance of the system and then evaluate and document the results.

Testing plan and criteria

There are several possible testing methodologies when an application is fully developed and now ready for deployment, but, they can be divided into two main types. These two types are the functionality and usability testing methodologies of the system (Mahangu, 2010).

Functionality testing

System testing involves different strategies for evaluating the functionality of a piece of software:

- Unit testing

  This strategy is used to test whether the individual units of the system are fit for use. For example, testing can be done in respect of the command buttons, checkbox etc.
- Integrated testing

  This testing strategy combines individual software modules to be tested as a group. For example, creating an agenda, recording minutes and adding members to a committee and modules are tested together to ensure that the system is fully functional.
- Black box testing

  This technique uses valid and invalid input to test the output and system behaviour.

Usability testing

Usability testing measures the usability, or ease of use, of a specific object or set of objects. To perform this test, the users were asked to evaluate the usability of the system by giving a rating between 1 and 5, with 1 being inefficient and 5 being very efficient. While measuring functionality and usability, the system was tested by the same four users previously interviewed during requirements specification. The tests helped to discover errors and areas that required improvement. The participants' details are recorded in the table below (see Table 8). The first two participants were able to log onto the system and view minutes. They were also able to request minutes for previous meetings. The period for testing per participant was picked by guesswork through several tests to enhance efficiency. The last two participants were able to create new minutes and send to other users. They were also able to update (delete, modify etc) the minutes.

Table 8: Participants' details

Name of participant	**Department**	**Modules Tested**	**Time Taken(Mins)**
Fatima Jacobs	Computer Science	Login, request & view minutes	5
Lorein Wesso	Statistics	Login, request & view minutes	5
Rene Abbott	Computer Science	Insert, send and update minutes	10
Leslie Selbourne	Statistics	Insert, send and update minutes	10

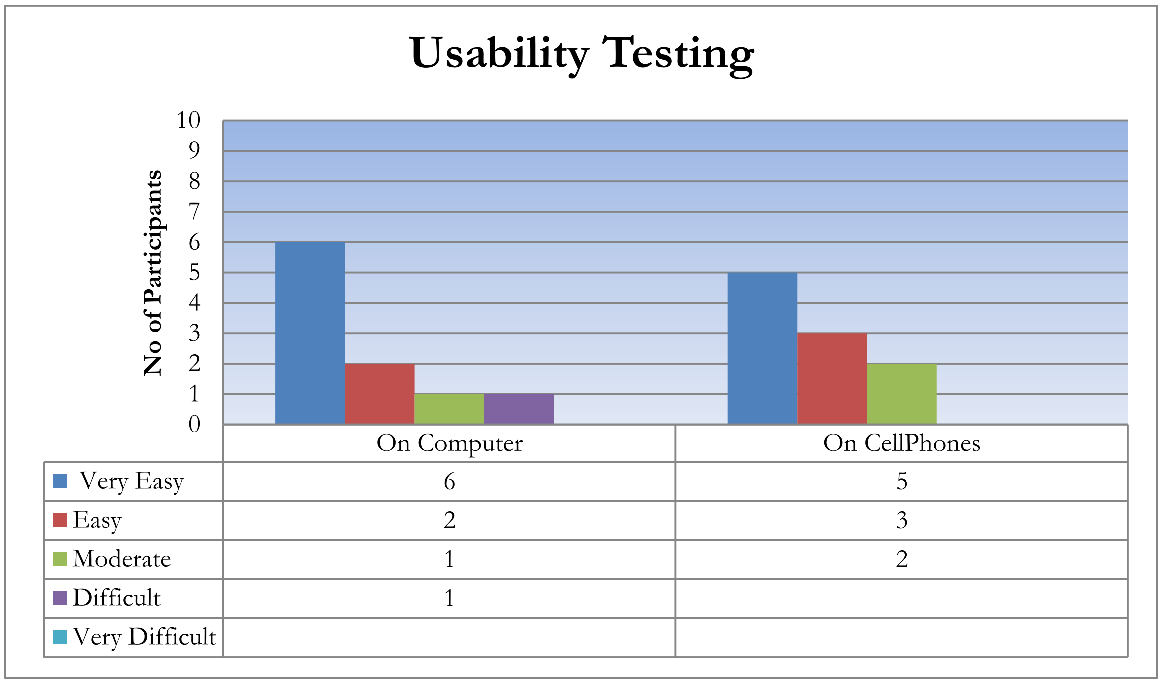
Testing results and discussion

In order to obtain feedback from users, a participation consent form had to be issued to the user (see Appendix G). A scenario was drawn to facilitate and guide the performance of some tasks on the system and some evaluation questions were prepared to evaluate the functionality and usability of the system (see Appendix H). This document will show an analysis of the user (referring to all users) test feedback which was carried out in two weeks.

Usability testing **(Week 1)**

Usability testing focused mainly on the general appearance and the user friendliness of the system. According to the results obtained, users had the same judgment of the interfaces (administrator and general user). Hence, the information obtained represents the general appearance and user friendliness of the whole system interface (see Figure 13).

The four users and some computer science students tested the usability of the system. It was found that the application is relatively straight forward because it does send and receive minutes. Participants said it was easy to use and after seeing the demo they did not need to spend much time learning it. In the demo, there were three people involved at one time where one took the role of the administrator, another one took the role of Head of Department (HOD) and the last one took the role of a common user. The administrator looked at how easy it is to use the interface of the application on the computer by typing some minutes and trying to save and send them to the HOD and other staff members. The HOD used a mobile device to test the interface of the application; how user friendly the tabbed control buttons are and tried to see if there is enough space for requesting and updating minutes on the mobile device. Note that the HOD was chosen just for formality that the administrator would preferably send minutes first to the HOD for checking before they are sent to the other staff members. The last user tested almost the same thing as the HOD, except that he was unable to modify the minutes by checking if the modify, delete and update control tabs were not enabled if he login as a common user. A clear picture of how the users who tested the system evaluated and analyzed the interface is shown in Figure 13.



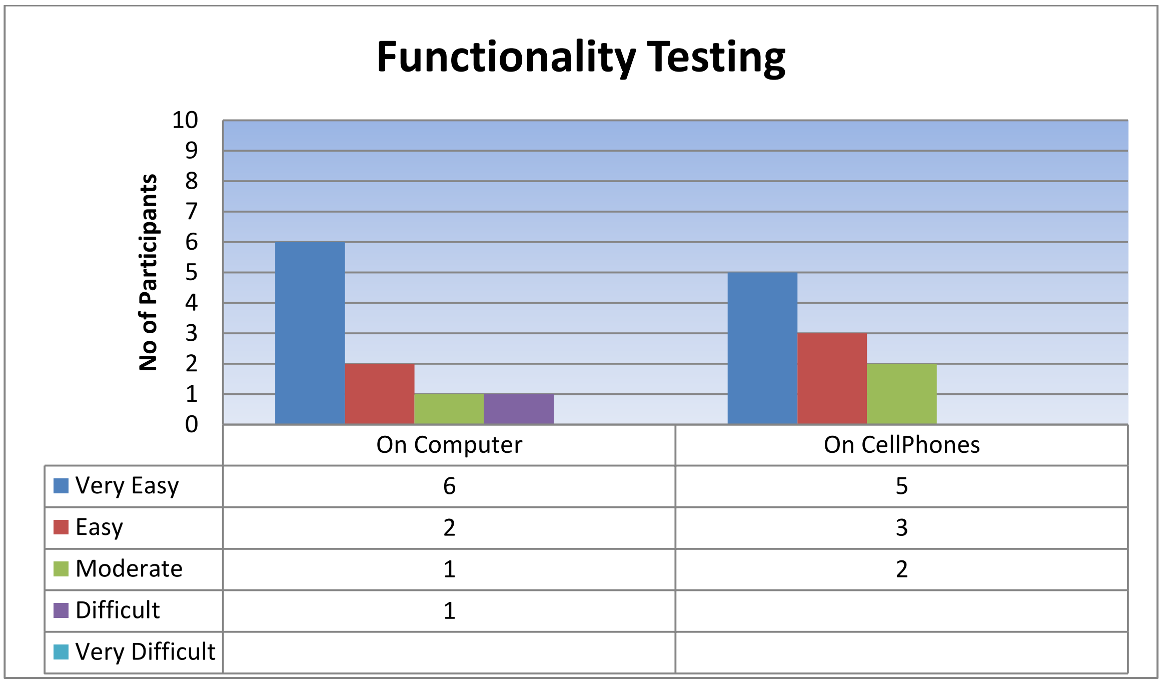
	On Computer	On CellPhones
🟦 Very Easy	6	5
🟥 Easy	2	3
🟩 Moderate	1	2
🟪 Difficult	1	
🟦 Very Difficult		

Figure 13: Usability testing

Users displayed a positive attitude to the general appearance and design of the interface on the both the computer and mobile phones. The application is very easy to understand and easy to learn as shown in Figure 13, this increased user involvement and encouraged users to use the application more. For instance, on the first test, some of the test participants suggested that some aspects of the system be changed, like the size, visibility and background colour of the control buttons. After these changes were made, they were satisfied. In addition, most users reported that the system was easy to understand and use. It was much-admired for its simplicity and the fact that few mouse clicks were required to complete an entire process. Links worked accurately making it easy to navigate among pages.

***Functionality testing* (Week 2)**

Functionality testing involved testing interface pages, control buttons and the database. During this test, users tested the core functions of the system by sending and receiving the minutes. The same scenario used in usability testing section was deployed in this section. Two-thirds of the users said that the system functions are easy to perform and 2 in 3 found the system functionality very easy to follow (see Figure 14). All the users agreed that they would love to use a system like this one.



	On Computer	On CellPhones
🟦 Very Easy	6	5
🟥 Easy	2	3
🟩 Moderate	1	2
🟪 Difficult	1	
🟦 Very Difficult		

Figure 14: Functionality testing

In addition, the performance of the application was also tested. This was done to check how quickly the application responds to user input, the sending and receiving time of minutes, testing for full, partial, or upgrade install/uninstall processes on different versions of windows mobile devices and how the application works on different windows mobile devices. The results showed that the application's performance is reasonable. Any response time that is less than 3 seconds on a mobile application is considered reasonable (Chao, 2010). It was also discovered that the application's quick load time depends on the mobile phone being used. The application installed correctly without errors on windows mobile devices and also the uninstall process was as painless and successful as it should be. Devices running windows phones 7.0 and lower (such as Nokia Lumia 7.1) do not support this application, so the application could not be installed on them. Lastly, the application uses SMS which users are already familiar with. SMS are therefore accessible to all of the target users.

Recommendations

During system testing, users made significant comments and recommendations about possible ways of improving this system. Some corrections were made to rectify errors. Some functionality was not successfully implemented on the system due to time constraints and therefore will be recommended for future work. For instance, converting the meeting minutes to a pdf file, automatic recording of minutes by the system during the meeting and extending the platform limitations the current system has to all windows phone devices.

Conclusion

This chapter has discussed the techniques used to test the system and the test results of various components of the application. It has also given graphical representations of the test results. The testing was successful as all the testing strategies conducted were completed successfully. The testing involved testing the system's functionality and usability. During functionality testing, users tested the core functions of the system by sending and receiving the minutes and most users found the system's functionality very easy to follow. Similarly, during usability testing users found that the application is very easy to understand and easy to learn. This increased user involvement and encouraged users to use the application more. The users also tested the performance of the application and found that the application's performance is reasonable. Thus, the conclusion can be made that the user's requirements gathered in chapter 1 were met. The following chapter provides a detailed user guide documentation that will help users to use the system.

C h a p t e r   8

USER MANUAL

Introduction

The previous chapter discussed the system testing process, highlighting the strategies used, the test results acquired and the system performance, and also listed some recommendation for improvement from the testers. This chapter gives a detailed system user guide that is aimed at giving out easy-to-follow instructions and guidelines on how to use the application. The user guide includes administrator user guide and the regular user (staff member) guide in which a regular user and administrator are provided with the information relevant to their respective tasks. It also discusses how the user will first install the application on their mobile devices and computers and how they should continue to use it as the need arises.

System Requirements

NotePhones System® requires a computer with Windows mobile device centre, Windows Vista, or Windows 7. It also needs a phone that supports .NET Compact Framework (.NET CF) 3.0 and SQL Server CE (Compact Edition). Windows mobile devices 6.0 above support the system by default.

Application installation

In order for the user to have the application on their mobile device or computer, the user has to download the application from http://www.cs.uwc.ac.za/~amwangonde/#downloads. The file which the user must download is a NotePhones system file. Once the user has the file on their device, he or she can navigate to wherever they have saved it and tap on it to run it. When the user has tapped the file, the install interface will pop-up. When the install screen shows up, the user will be shown a number of permissions that the application needs in order to perform all its tasks. At this point, the user should select install and wait for a few seconds depending upon the performance of the device being used. After the user selects the install button, the application will install and save itself on the physical memory of the device and the application icon will be found on the phones menu screen. A user can confirm that the application is appearing in the list of programs by going to *Control Panel* → *Add or Remove Programs.*

Getting started

The NotePhones application window includes standard Windows components. Of particular interest are the following:

The **Title Bar** displays not just the NotePhone product title but also the name of the archive you are currently working with. The **Toolbar** provides quick ways of performing the most common tasks. The **Main Window Area** displays information about the files in the archives. Finally, the **Status Line** displays the number and sizes of currently selected files, along with other archive information.

The user can start the application by going to the menu screen where the application icon/shortcut is and tapping the icon to start the application. Interaction with the internals of the application can then begin.

Administrator user guide

This user guide is designed for administrators/secretaries to provide them with information so that they can carry out the agenda formulation and minutes recording process effectively. The user guide provides interfaces which the administrator will use on the computer and comprises of the following components: the login screen and control buttons which allow users to navigate around the system and to perform tasks.

Login screen

The first step for the administrator user to be able to use the system is to get authentication through the login page. The login page requires the administrator to enter a username and password (see Figure 15). The following steps are to be carried out during this login process

- Enter username and password
- Click on the 'Login' button. Note that if login information is incorrect, access will be denied

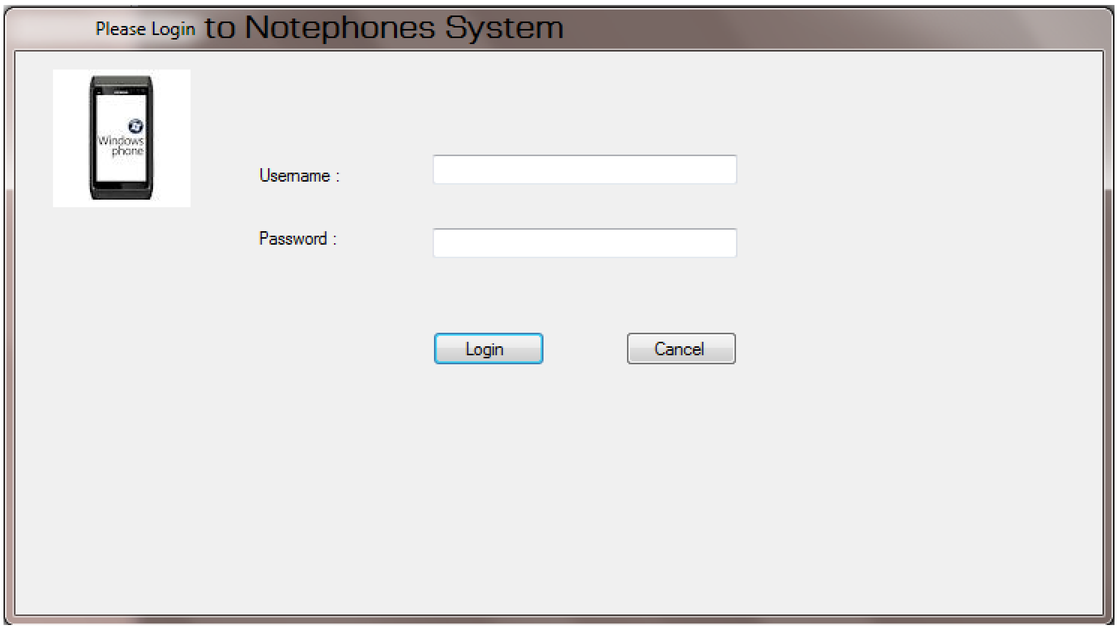
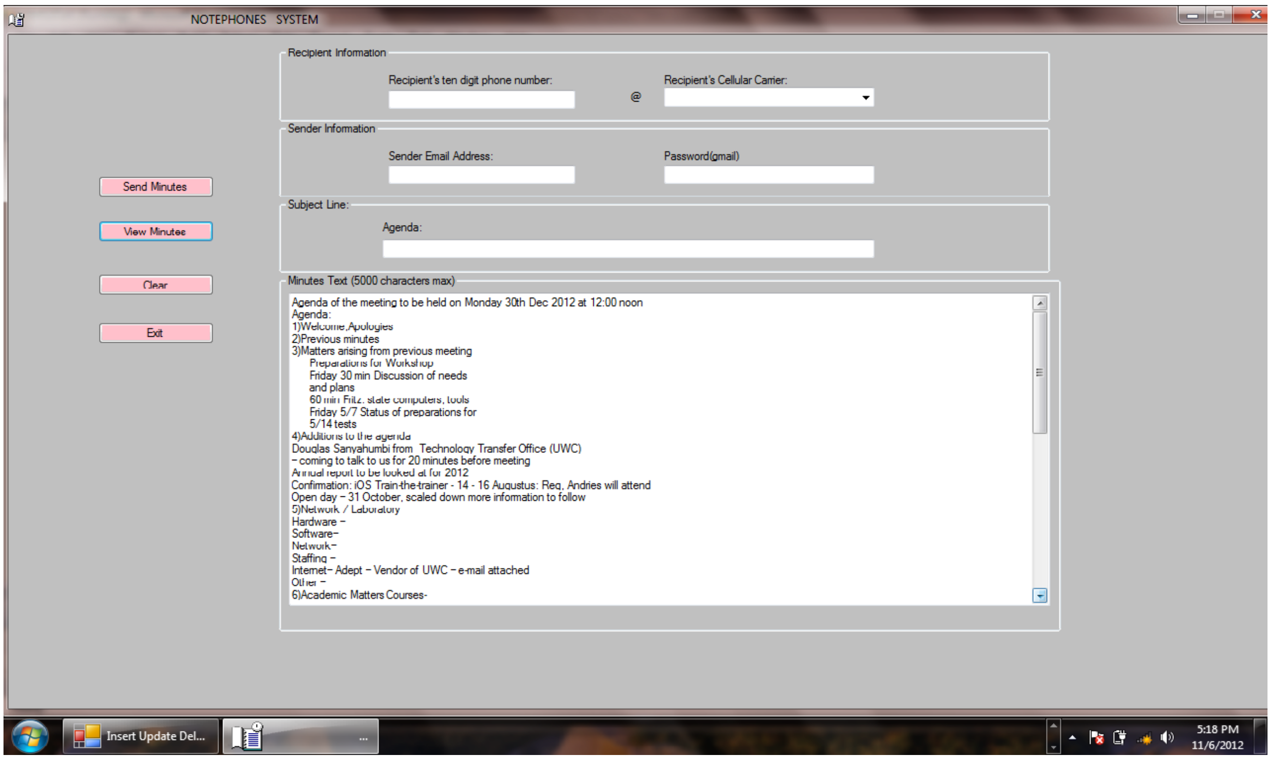


Figure 15: Login screen

Creating a new agenda

Once the administrator is logged into the system, he or she must fill in relevant information in the text boxes provided. The agenda and minutes created must then be sent to all staff members by clicking a 'send minutes' control button as shown in Figure 16 below:.

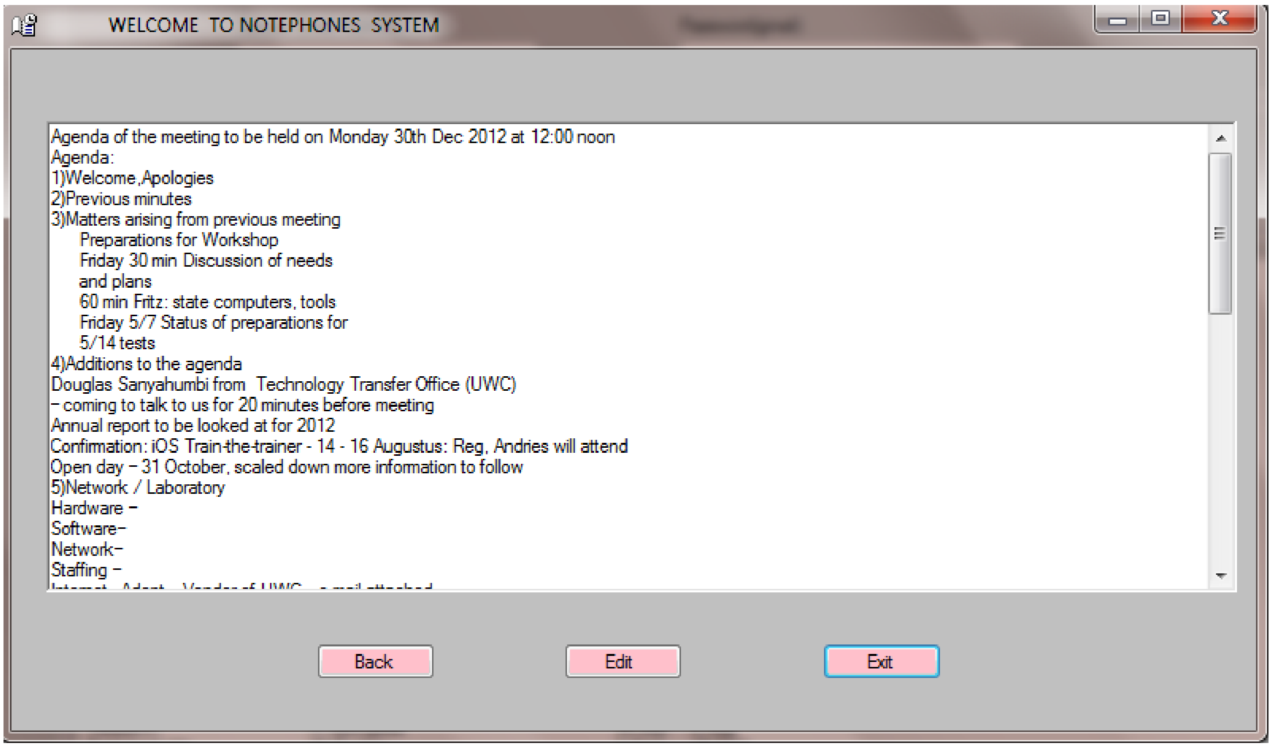


```
Agenda of the meeting to be held on Monday 30th Dec 2012 at 12:00 noon
Agenda:
1)Welcome,Apologies
2)Previous minutes
3)Matters arising from previous meeting
    Preparations for Workshop
    Friday 30 min Discussion of needs
    and plans
    60 min Fritz. state computers, tools
    Friday 5/7 Status of preparations for
    5/14 tests
4)Additions to the agenda
Douglas Sanyahumbi from  Technology Transfer Office (UWC)
– coming to talk to us for 20 minutes before meeting
Annual report to be looked at for 2012
Confirmation: iOS Train-the-trainer - 14 - 16 Augustus: Req, Andries will attend
Open day – 31 October, scaled down more information to follow
5)Network / Laboratory
Hardware –
Software–
Network–
Staffing –
Internet– Adept – Vendor of UWC – e-mail attached
Other –
6)Academic Matters Courses-
```

Figure 16: Creating an agenda

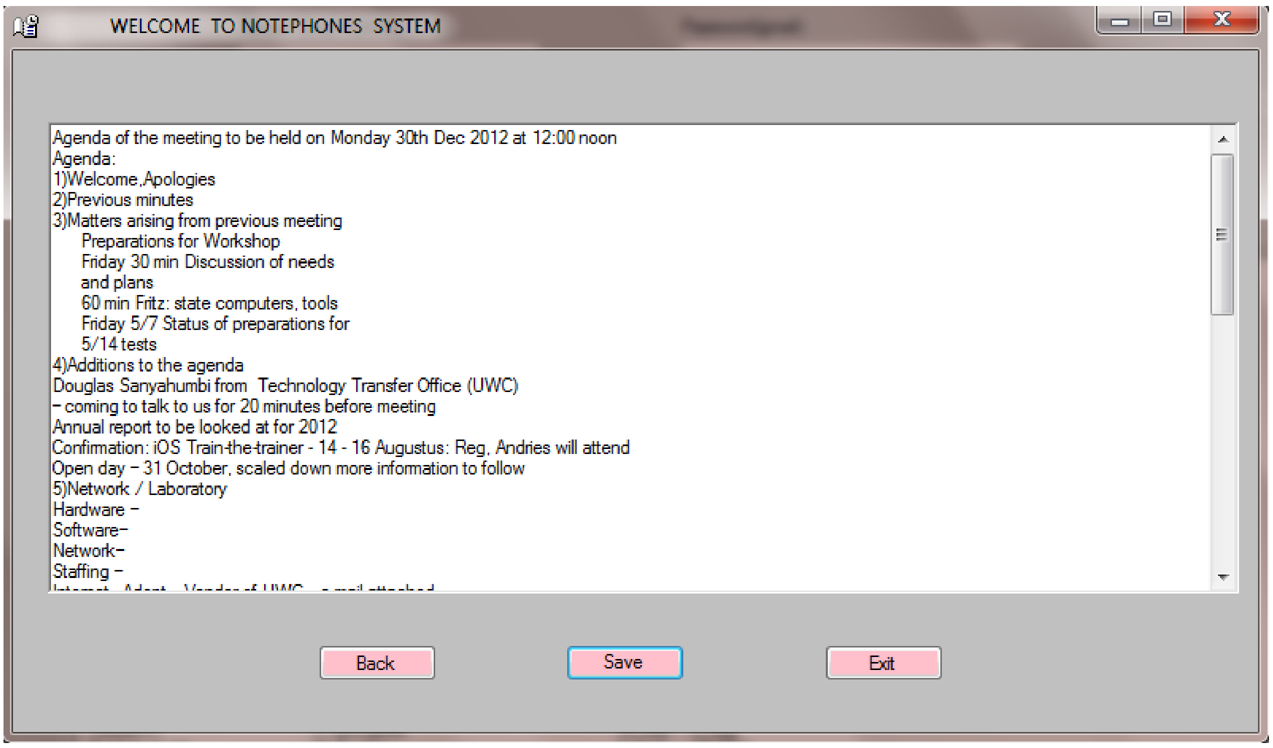
Saving created agendas and minutes

When the agenda has been created, it can be saved by clicking 'view minutes' control button (see Figure 17). In this screen, the agenda can be edited and saved to the database (see Figure 18)



Agenda of the meeting to be held on Monday 30th Dec 2012 at 12:00 noon
Agenda:
1)Welcome,Apologies
2)Previous minutes
3)Matters arising from previous meeting
Preparations for Workshop
Friday 30 min Discussion of needs
and plans
60 min Fritz: state computers, tools
Friday 5/7 Status of preparations for
5/14 tests
4)Additions to the agenda
Douglas Sanyahumbi from Technology Transfer Office (UWC)
– coming to talk to us for 20 minutes before meeting
Annual report to be looked at for 2012
Confirmation: iOS Train-the-trainer - 14 - 16 Augustus: Reg, Andries will attend
Open day – 31 October, scaled down more information to follow
5)Network / Laboratory
Hardware –
Software–
Network–
Staffing –

Figure 17: Edit agendas

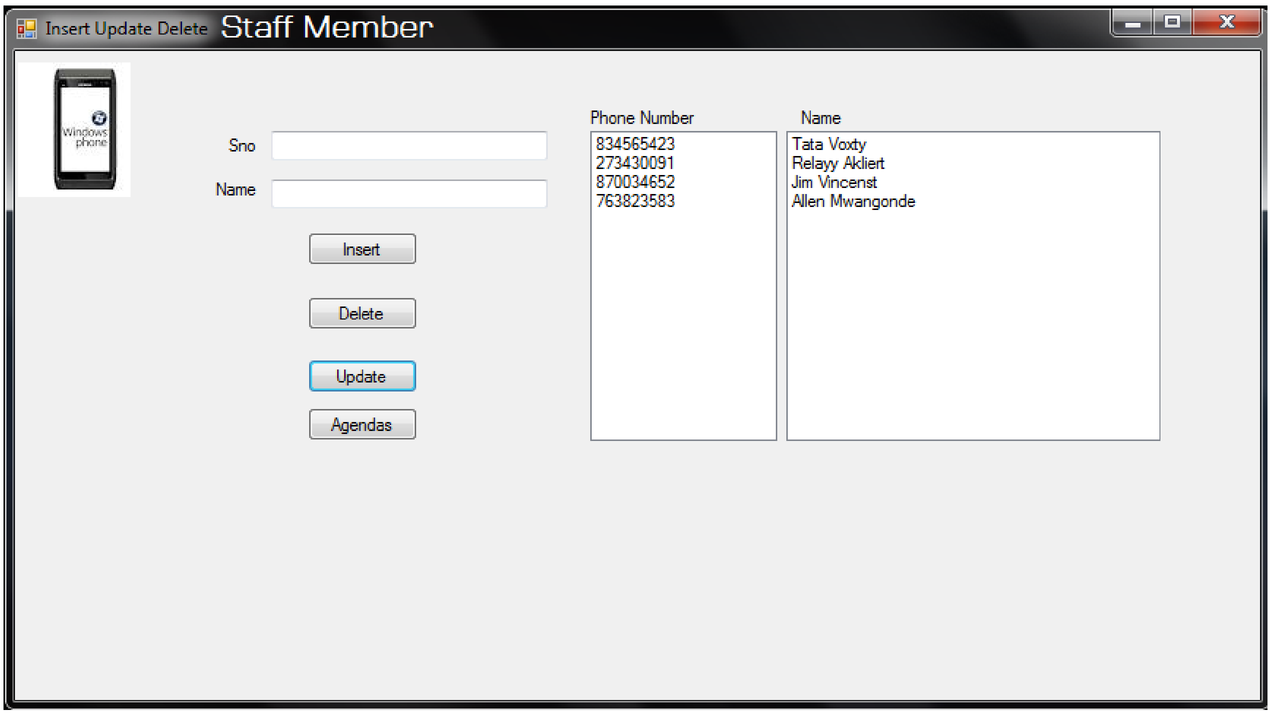


Agenda of the meeting to be held on Monday 30th Dec 2012 at 12:00 noon
Agenda:
1)Welcome,Apologies
2)Previous minutes
3)Matters arising from previous meeting
    Preparations for Workshop
    Friday 30 min Discussion of needs
    and plans
    60 min Fritz: state computers, tools
    Friday 5/7 Status of preparations for
    5/14 tests
4)Additions to the agenda
Douglas Sanyahumbi from Technology Transfer Office (UWC)
- coming to talk to us for 20 minutes before meeting
Annual report to be looked at for 2012
Confirmation: iOS Train-the-trainer - 14 - 16 Augustus: Reg, Andries will attend
Open day – 31 October, scaled down more information to follow
5)Network / Laboratory
Hardware –
Software–
Network–
Staffing –

Figure 18: Save agendas

Creating new staff member

The administrator will also be required to insert information of all new staff members, delete or update them if necessary as shown in Figure 19. By clicking on the ‘agendas’ control button, the administrator will be able to navigate to the creation of agendas and minutes as shown above in Figures 16, 17 and 18.



Phone Number	Name
834565423	Tata Voxty
273430091	Relayy Akliert
870034652	Jim Vincenst
763823583	Allen Mwangonde

Figure 19: Insert/delete/update Members

Regular member user guide

This user guide is designed for regular/common users to provide them with information so that they can use the system effectively. The interfaces described in this user guide are based on the premise that the regular user will be using the application on a mobile devices while the previously described administrator user guide on the computer.

Login screen

The first step for the user to be able to use the system on a mobile device is to get authentication through the login screen. The login screen requires the user to enter a username and password (see Figure 20). The following steps are to be carried out during this login process:

- Enter username and password
- Click on the 'login' button. Note that if login information is incorrect, access will be denied

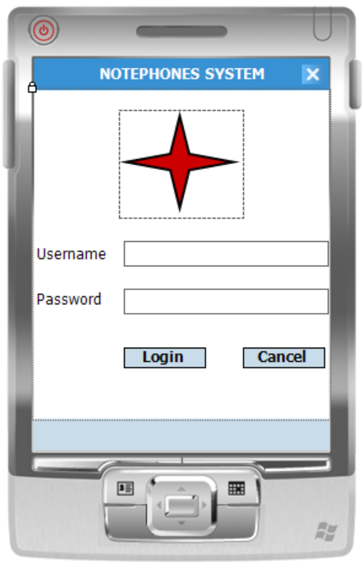


Figure 20: Login screen

Tabbed control buttons

It is necessary to point out that since the phone screen is relatively smaller than a computer or laptop, tabbed control (with great creativity) is used to segregate the functionality in multiple tabs so that the screen breathes with space and does not look jammed up (see Figure 21). Once the regular user is logged into the system, he or she must then use the simple and easy to use tabbed controls to fill in relevant information the text boxes provided. The various tabbed controls provided such as viewing minutes from 'message' button and adding minutes from 'add' button can be used to view the messages and edit the minutes respectively. Editing can be done in cases where the regular user assumes that the administrator omitted certain details during minutes creation process. The user may also launch requests for new agendas and minutes and send an apology from the 'menu' button as shown in Figure 22.

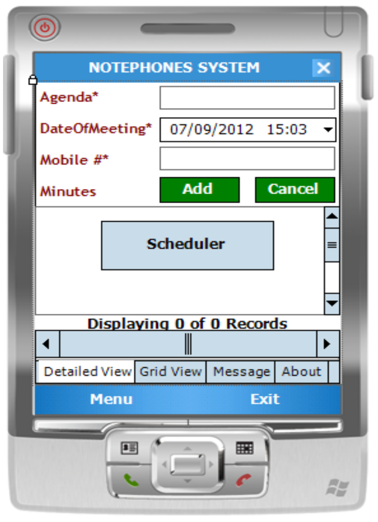


Figure 21: Tabbed control buttons

Figure 22 also shows the 'scheduler' button which will automatically close the application after 10 seconds of inactivity. This is done to prevent, when the code fires, the window from remaining open and consume the phone's valuable memory and hence drain the battery. Clicking the 'scheduler' button before 10 seconds expire will ensure continuity of the system for the current session. It is important to note that the 'scheduler' time can be modified, that is, a user can make the system delay for longer than 10 seconds before closing automatically. Lastly, the 'exit' button is the button that the user will select when in order to escape the application to the phone home screen. Thus, this button frees all resources which are used by the application. It is also important for the purpose of starting over should the user wish to do so.

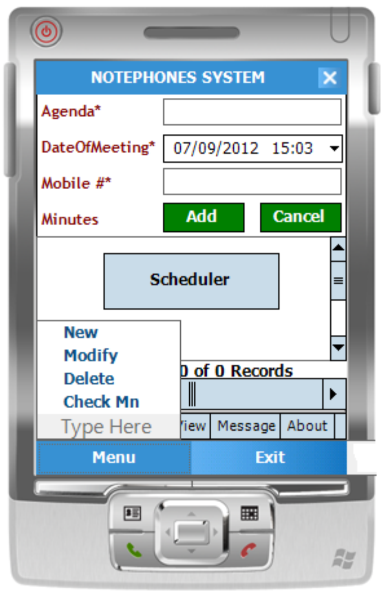


Figure 22: Menu control button

Conclusion

This chapter has discussed the user guide documentation. The user guide included administrator user guide and the regular user (staff member) guide in which a regular user and administrator are provided with the information relevant to their respective tasks. The chapter has also discussed how the user will first install the application on their mobile devices and computers and how they should continue to use it as the need arises. The following chapter provides a summary about the NotePhones System and the research carried out through this project.

Chapter 9

SUMMARY

This thesis studied several aspects of managing minutes of a meeting in mobile devices including data modeling and design theory, query processing, modifications and versioning, and the application of minutes in mobile devices. The next paragraphs summarize the main contributions of this thesis and discuss general directions for future research arising from therefrom.

The NotePhones System project was proposed and carried out with a view to address problems experienced when managing meeting minutes with a web-based system and the traditional paper-based minute management system. The main aim of the project was to go a step further by providing a mobile-based system for managing minutes. This thesis has documented the development of a mobile management system. Generally, the chapters in this thesis discuss the basic processes followed to achieve the completion of this project. These are requirements gathering, analysis, design, coding, implementation, testing and maintenance. The Interactive Model was the software development process used in this project. The requirements were gathered in chapter one and were analysed in chapter two and three. Designing, coding, implementation and testing processes followed in chapters four through seven respectively. The steps were iterated a number of times. Maintenance was however not part of the scope of this project.

All the objectives that have not been met while implementing the system for the purpose of this project are considered for future work. A very general direction for future research is to explore other minutes management applications that could benefit from the incorporation of techniques from the mobile-based system. One such candidate is Information Extraction (IE), whose goal is to extract structured data from semi-structured data or from unstructured information such as plain text (Marc, 2001). Information extraction is an inherently uncertain process. Thus, capturing the of minutes data produced by IE, and sending the minutes to the user as a 'first-class' part of the result, may improve IE's overall utility and usability.

APPENDICES

APPENDIX A

PROBES

QUESTIONNAIRE FOR MOBILE MINUTE MANAGEMENT SYSTEM

The management of minutes is a generally cumbersome task. While a web-based system is more efficient than the traditional paper-based minute management system, this project aims to go a step further by providing a mobile-based system for managing minutes.

The following is expected of the software:

- It will provide a simple interface that will be easy to use and understand on a basic mobile phone.
- It will require user authentication to ensure that only authorise users have access to the minutes.
- It will send out notifications to users on meeting schedules and other tasks

Your answers to the following questions will assist in the development of this software:

1. What system are you using to manage minutes, paper-based or web-based system? What problems do you have with the existing system?
2. Do you think a simplified system that can be accessed by phone can be useful?
3. If yes, how? And if not, why not?
4. What functionalities would you regard as important/ would you like the system to have?

APPENDIX B

INTERVIEWS

Users interviewed

- Fatima Jacobs (Computer Science)
- Lorein Wesso (Statistics Department)
- René Abbott (Secretary Computer Science Department)
- Leslie Selbourne (Secretary Stats Department)

Ms Fatima Jacobs

1) What system are you using to manage minutes, paper-based or web-based system? What problems do you have with the existing system?
    - We are using a paper-based system to manage minutes
    - Problems involved are as follows
        - Time consuming
        - Writing speed of minute-taking
        - Academic jargons used are usually unfamiliar
        - Forgetting when rewriting the meaning of speedy shortcuts used
2) Do you think a simplified system that can be accessed by phone can be useful? If yes, how? And if not, why not?
    - Yes
        - Saves money since you do not need internet to access minutes
        - Keep members up with the discussions
3) What functionalities would you regard as important/ would you like the system to have?
    - User-friendly interface
    - Permission on what to/not delete
    - Storage facility for important facets

Lorén Wesso

1) What system are you using to manage minutes, paper-based or web-based system? What problems do you have with the existing system?
    - We are using a paper-based system to manage minutes
    - Problems involved are as follows
        - People may not understand text minutes
        - Not easily accessible. Staff members usually do not check their email
        - Spelling errors
        - Printing of minutes is a hassle
2) Do you think a simplified system that can be accessed by phone can be useful? If yes, how? And if not, why not?
    - Yes
        - Saves money since you do not need internet to access minutes
        - Easy access. People are always on the go and may not have the time to check emails to read minutes
3) What functionalities would you regard as important/ would you like the system to have?
    - Some sort of security measures to be able to access minutes of meetings
    - Clear audio features to help people listen to the minutes discussed
    - Playback features should be incorporated
    - Backups if audio fails

René Abbott

1) What system are you using to manage minutes, paper-based or web-based system? What problems do you have with the existing system?
    - We are using a paper-based system to manage minutes
    - Problems involved are as follows
        - Not easily accessible. Staff members usually do not check their email
        - People do not like to be quoted for security reasons

2) Do you think a simplified system that can be accessed by phone can be useful? If yes, how? And if not, why not?

    - Yes

        - Easy access. People are always on the go and may not have the time to check emails to read minutes.

3) What functionalities would you regard as important/ would you like the system to have?

    - Security measures
    - User-friendly interface

Leslie Selbourne

1) What system are you using to manage minutes, paper-based or web-based system? What problems do you have with the existing system?

    - We are using a paper-based system to manage minutes
    - Problems involved are as follows

        - Not easily accessible when one does not have internet access
        - Time consuming
        - Wastage of paper
        - Adding matters to agenda is not easy

2) Do you think a simplified system that can be accessed by phone can be useful? If yes, how? And if not, why not?

    - Yes

        - Easier access as it is a good alternative
        - Quicker

3) What functionalities would you regard as important/ would you like the system to have?

    - User-friendly interface
    - Authentication of users
    - Simplified way of adding items to agenda
    - Notifications of meeting dates/reminders

APPENDIX C

TERM 1 PLANNING

Meeting dates & times /Tasks	~~**Wednesday 15th Feb**~~ ~~**10h00**~~	~~**Wed 22nd Feb**~~ ~~**12h00**~~	~~**Wednesday 29th Feb**~~ ~~**10h00**~~	~~**Wednesday 7th March**~~ ~~**10h00**~~	**Wednesday 14th March** **10h00**	**Holiday**	**2nd April** **14h00**	**Wednesday 4th April** **10h00**
Comments	~~**Identify users to interview**~~	~~**Identify users to interview or give a questionnaire to.**~~ ~~*Note questionnaire or probes and add as appendix.*~~	~~**Identify users to interview or give a questionnaire to.**~~ ~~*Note questionnaire or probes and add as appendix.*~~	~~*Identify someone (Writing Centre) to read your doc. & organise someone for checking the presentation.*~~	***Mock Presentation on Tuesday 2nd April (have a look at what was done before at 14h00)***		**Presentation – Tuesday another mock presentation & the real McKo on Wednesday! (09h00 – 11h00**	
Thesis Document	~~Create document using Honours Project Guidelines from the website as well as Thesis doc from Word~~	~~See previous week! Complete the project outline (using Honours Project Guidelines)~~	~~See that you understand how to use the Styles and how to compile an Index, the Table of Contents, List of Figures etc.~~	~~and start with RAD.~~	○ Check Index and add indexes ○ Bibliography – at least 5 entries.	Finalise write-up. Let someone **proof** read your document!		
URD	~~*Fill in headings*~~ ~~*Look at the questionnaire.*~~	~~*Fill in headings*~~ ~~Start write-up of URD.~~	~~Finnish with URD write-up. Interview stakeholders.~~	~~Edit with the suggestions I made. Complete URD~~				
RAD	~~n/a~~	~~Fill in headings~~		~~Start write-up of RAD.~~	Write-up RAD	Complete RAD		
Literature Survey	~~Familiarising yourself with your topic, and how it's implemented.~~ ~~Read and explore. literature on your topic.~~	~~Read and explore. some more – use Google Scholar..~~	~~Read and explore. some more – use Google Scholar..~~ ~~Add all literature found to your bibliography – use the Harvard Notation~~	~~Keep on reading~~				
Presentation/deliverable	~~Write one paragraph that describes what you want to do, and why you want to do it This will be used as the abstract~~	~~Write one paragraph that describes what you want to do, and why you want to do it This will be used as the abstract~~				Use thesis to Prepare slides for mock presentation		Presentation
Website	~~Ask Frieslaar~~ ~~intelligent.networking@gmail.com~~	Ask Frieslaar intelligent.networking@gmail.com See	~~Put plan onto website~~ ~~Ask Frieslaar~~ ~~intelligent.networking~~	~~Ask Frieslaar about the~~	Ask Frieslaar about the server.	Add URD & RAD to website		

		Create website	~~@gmail.com See~~ ~~Create website~~	~~server.~~ ~~Add URD to website~~	Add URD to website			

APPENDIX D

TERM 2 PLANNING

Tasks	~~11th April~~	~~18th April~~	~~25th April~~	~~2th May~~	~~9th May~~	~~16th May~~	23rd May	30th May
Comments		~~*Combined meeting*~~		~~*Combined meeting*~~				
Thesis Document	~~Create 3 new chapters with subheadings for UIS, OOA and OOD – see p3~~	~~Complete editing. Start with the write up of the UIS~~	~~Start write up of the OOA & complete the UIS write-up~~	~~Complete write-up of OOA.~~		~~Complete write-up of OOA.~~	Complete write up of OOD Hand in completed document to supervisor on Monday the 28th May	
OOA	~~Read through the documentation of this carefully and see that you understand it!~~		~~Start with analysing the RAD to create OOA~~	~~Complete OOA~~				
OOD				~~Start write up of OOD~~	~~Complete OOD~~	~~Complete write up OOD~~	Complete OOD	
UIS		~~Start with User Interface Specification~~	~~Complete UIS~~			~~Update changes to UIS~~		
GUI & prototype			~~Start with the planning of the prototype~~	~~Program GUI/ prototype~~	~~Program GUI/ prototype~~	~~Program~~	Complete GUI/ prototype	
Other	~~Look at previous projects~~	~~Look at previous projects~~						
Presentation						~~Look at what must be in the presentation. Come with some suggestions.~~		*Prepare slides for mock presentation* ***(28th May) & actual presentation on the 30th***
Web-site	~~Update web site~~					~~Check and update~~		**Put new plan, thesis & presen-tation on web site**

APPENDIX E

TERM 3 PLANNING

Planning for the 4th Term

Tasks for the week of the	**~~1st Oct~~**	**~~8th October~~**	**~~15th October~~**	**22nd Oct**	**29th Oct**	**05th Nov**	**12th Nov**	**19th Oct**	**23rd Nov** **Presentation**
Thesis Document	~~Revise and edit document~~			Update the document as well as all figures etc. with the new interface Revise chapter 2-4 Add the programs	Complete all editing of 3rd term's documentation	**Start** by identifying all the tasks that the program must be able to do - write the User's guide		Finalise user's guide & Thesis documentation. Ask colleague or someone to edit /proofread it!!	Hand in documentation on Monday 8th November
Programming tasks not completed		Revise program and add all outstanding functionality			Revise programme & add what is still outstanding	And make changes to thesis doc to reflect this!		Revise programme & create installation disc.	Revise programme Finalise Installation disc.
Design Test suites					Read up about evaluation of programs – get ideas of how you want to do evaluation. Add to documentation. Edit the chapters where you referred to it initially.	Design test suites Choose your "users" Questionnaires etc. Ethical aspects			

Execute tests, revise code or even project design							Execute tests & keep record of it by writing it up in Thesis doc. Create graphs that can be included in Thesis doc.		
Presentation									*Mock Presentation on the 20th Nov.* *Prepare for presentation and install on relevant computers. Test if it works!!*
Website			**Update NB**	**Update NB**				**Final Update NB**	

APPENDIX F

TERM 4 PLANNING

Tasks for the week of the	**~~1st Oct~~**	**~~8th October~~**	**~~15th October~~**	**22nd Oct**	**29th Oct**	**05th Nov**	**12th Nov**	**19th Oct**	**23rd Nov** **Presentation**
Thesis Document	~~Revise and edit document~~			Update the document as well as all figures etc. with the new interface Revise chapter 2-4 Add the programs	Complete all editing of 3rd term's documentation	**Start** by identifying all the tasks that the program must be able to do - write the User's guide		Finalise user's guide & Thesis documentation. Ask colleague or someone to edit /proofread it!!	Hand in documentation on Monday 8th November
Programming tasks not completed		Revise program and add all outstanding functionality			Revise programme & add what is still outstanding	And make changes to thesis doc to reflect this!		Revise programme & create installation disc.	Revise programme Finalise Installation disc.
Design Test suites					Read up about evaluation of programs – get ideas of how you want to do evaluation. Add to documentation. Edit the chapters where you referred to it initially.	Design test suites Choose your "users" Questionnaires etc. Ethical aspects			

Execute tests, revise code or even project design							Execute tests & keep record of it by writing it up in Thesis doc. Create graphs that can be included in Thesis doc.		
Presentation									*Mock Presentation on the 20th Nov.* *Prepare for presentation and install on relevant computers. Test if it works!!*
Website			**Update NB**	**Update NB**				**Final Update NB**	

APPENDIX G

PARTICIPANTS CONSENT FORM

The Project topic: NotePhones System

Name of student: Allen Mwangonde

Name of participant: ____________________

Department: ________________________

I (Participant) agree to participate in this research on the basis that:

1. This agreement is of my own free will
2. I have had the opportunity to ask any questions about the study
3. I realize that I may withdraw from the study at any time, without giving a reason and without any effect on my education
4. I have been given full information regarding the aims of the research and have been given information with the Researcher's names on and a contact number and address if I require further information.
5. All personal information provided by myself will remain confidential and no information that identifies me will be made publically available.

   Signed: .... Date: .... (Participant)

   Signed: .... Date: .... (Student)

APPENDIX H

TEST RESULTS TABLES

Table 9: Usability Testing

PARTICIPANTS	EASY	V.EASY	MODERATE	DIFFICULT	V.DIFFICULT
1					
2					
3					

4					
5					
6					
7					
8					
9					
10					

Table 10: Functionality Testing

PARTICIPANTS	EASY	V.EASY	MODERATE	DIFFICULT	V.DIFFICULT
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

BIBLIOGRAPHY

Glossary of Terms, ITU-infoDev ICT Regulation Toolkit. (2010). *http://www.ictregulationtoolkit.org/Glossary.*

Chao, L. (2010). HTC to Sell Branded Smartphones in Mainland Push. *The Wall Street Journal.*

Landay, J. A. (1999). Making sharing pervasive: Ubiquitous computing for shared note taking. *IBM SYSTEMS JOURNAL, Vol.38*(No.4).

Luz, M.-M. B. (2006, 10 24). Meeting browsing.

Mahangu, A. (2010). *Minutes Management System.* University of the Western Cape, Computer Science Department. Unpublished Honours Project.

Marc, P. (2001). Digital Natives Digital Immigrants. *Vol. 9*(No. 5).

Reed, P. R. (1999, November 14). *Developing Applications with Visual Basic and UML* . Retrieved from http://www.amazon.com/exec/obidos/ASIN/0201615797/vbSQLCecom-20

UCT. (2009, july 29). *UCT centre in ICT4D.* Retrieved from UCT centre in ICT4D: http://www.ict4d.cs.uct.ac.za/

Zuckerman, E. (2010). *Decentralizing the Mobile Phone:A Second ICT4D Revolution?* Retrieved from http://itidjournal.org/itid/article/viewFile/631/271