DEVELOPMENT OF A MOBILE APPLICATION SECURITY

By

ARE, Moshood Ajadi

PGD/MCS/2007/1232

Submitted to Department of Mathematics/Computer Science
Federal University of Technology,
Minna

In partial fulfillment of requirements leading to the award of

Postgraduate Diploma (PGD) in Computer Science

Federal University of Technology, Minna

May, 2010

DECLARATION

I hereby declare that the research work embodied in this project is an original work carried out by me in the department of Mathematics / Computer Science, Federal University of Technology, Minna under the supervision of Mallam A. M. Saliu. All materials consulted have been duly acknowledged under the references.

....

Are, Moshood Ajadi Date

CERTIFICATION

This project titled, "DEVELOPMENT OF A MOBILE APPLICATION SECURITY" by Are, Moshood Ajadi, meets the regulations governing the award of Postgraduate Diploma in Computer Science of Federal University of Technology, Minna.

.....
Mal. A. M. Saliu
Supervisor

04/07/2010
.....
Date

.....
Prof. N. I. Akinwande
Head of Department

.....
Date

DEDICATION

This project is dedicated to Almighty Allah, who preserved my soul till this time. May His name be praised forever-more and to my caring parents whose love and care manifested in my work.

ACKNOWLEDGEMENT

All praises are due to Almighty Allah, the Most Beneficent and the Most Merciful, who has been taking care of me and ever remains my source of inspiration in the accomplishment of this project work. May the Peace and Blessings of Allah (SWT) be on our leader, Prophet Muhammad (SAW), his households and all that follow his footpaths till the day of resurrection.

I deeply appreciate the guidance of my supervisor, Mallam A. M. Saliu who patiently corrected my work at every step; I will like to remember all you have done.

I wish to sincerely thank the Head of Department – Dr. N. I. Akinwande and other lecturers in the department too numerous to mention for their support and guidance throughout my study.

My due appreciation goes to my lovely parent: Mr. & Mrs. Are for, their prayer, their moral, parental and financial support given to me since I was born and also to my lovely wife for her prayer and priceless support.

My heartfelt gratitude goes to my loving Brothers, to my loving Sisters, my loving friends, my course mates and others

Finally, my profound gratitude goes to my wife and children; Zeenat, Waliyyullahi and Ishaq for their encouragement and support.

ABSTRACT

Mobile computing systems are computing systems that may be easily moved physically and whose computing capabilities may be used while they are being moved. Common examples include Laptops, Personal Digital Assistants (PDAs) and Mobile phones. The researcher observes the fervent need for individuals, groups or companies to avail themselves the opportunity to do such common mobile task like making or receiving calls, sending or receiving a message using the computer in a secure platform such that access levels are defined for a group of users. The study considers the software design and implementation of a mobile computing application used to compose, send and receive or save text messages; making call and other forms of activities that could be carried out on phones.

The researcher was inspired by the need to provide a mobile application whose prominent feature is its capability to provide stepwise security level for all anticipated users through the use of login interface as different levels of security were assign to a particular group of user to ensure optimum security and integrity of the application. Status or Error reports are also generated at request of the users for message and call history. The researcher deems it fit to implore the use Microsoft Visual Basic to write and run or implement the mobile application security on mobile computers.

TABLE OF CONTENTS

CHAPTER ONE

GENERAL INTRODUCTION

1.1 Background of the Study

Mobile computing systems are computing systems that may be easily moved physically and whose computing capabilities may be used while they are being moved. Examples are laptops, personal digital assistants (PDAs), and mobile phones. By distinguishing mobile computing systems from other computing systems we can identify the distinctions in the tasks that they are designed to perform, the way that they are designed, and the way in which they are operated. There are many things that a mobile computing system can do that a stationary computing system cannot do; these added functionalities are the reason for separately characterizing mobile computing systems.

Among the distinguishing aspects of mobile computing systems are their prevalent wireless network connectivity, their small size, the mobile nature of their use, their power sources, and their functionalities that are particularly suited to the mobile user. Because of these features, mobile computing applications are inherently different from applications written for use on stationary computing systems. (BBC, 2009)

The application development and software engineering disciplines are very young engineering disciplines compared to those such as structural, mechanical, and electrical engineering. Software design and implementation, for the most part, remain part art and part science. However, there are definite signs of maturation with the development of architectures, metrics, proven tools, and other methodologies that give an engineering discipline its structure. Whereas there are a variety of methodologies, techniques, frameworks, and tools that are used in developing software for stationary systems, there are very

few for mobile systems. Although mobile computing systems have existed as long as their stationary counterparts, most of the mature tools, methodologies, and architectures in software engineering today address the needs of stationary systems.

In wireless connectivity, mobile computing devices found a great way to connect with other devices on the network. In fact, this has been a great source of confusion between *wireless communications* and *mobile computing*. Mobile computing devices need not be wireless. Laptop computers, calculators, electronic watches, and many other devices are all mobile computing devices. None of them use any sort of wireless communication means to connect to a network. Even some hand-held personal assistants can only be synchronized with personal computers through a docking port and do not have any means of wireless connectivity. There are a variety of physical waveguide channels such as fiber optics or metallic wires. Wireless communication systems do not use a waveguide to guide along the electromagnetic signal from the sender to the receiver. They rely on the mere fact that electromagnetic waves can travel through space if there are no obstacles that block them. Wireless communication systems are often used in mobile computing systems to facilitate network connectivity, but they are not mobile computing systems. (Afuan, 2002)

Recently, computer networks have evolved by leaps and bounds. These networks have begun to fundamentally change the way we live. Today, it is difficult to imagine computing without network connectivity. Networking and distributed computing are two of the largest segments that are the focus of current efforts in computing. Networks and computing devices are becoming increasingly blended together. Most mobile computing systems today, through wired or wireless connections, can connect to the network. Owing to the nature of mobile computing systems, network connectivity of mobile systems is

increasing through wireless communication systems rather than wired ones and this is quickly becoming somewhat of a non mandatory distinguishing element between mobile and stationary systems. Though it is not a requirement for a mobile system to be wireless, most mobile systems are wireless. Nevertheless, let us emphasize that wireless connectivity and mobility are orthogonal in nature though they may be complementary.

For example, we can have a PDA that has no wireless network connectivity; however, most PDAs are evolving into having some sort of wireless connectivity to the network. Also, though it is important to understand that stationary and mobile computing systems are inherently different, this does not mean that they do not have any commonalities. We will build on existing software technologies and techniques used for stationary systems where these commonalities exist or where there is a logical extension of a stationary technique or technology that will mobilize it.

Due to the constant comparison between mobile systems and other types of systems, we will have to have a way to refer to the "other" types of systems. We will use the terms *non mobile* and *stationary* interchangeably. Although mobile is an industry-wide accepted terminology to distinguish a group of systems with the characteristics that we have just mentioned, there is no consensus on a system that is not a mobile system. For this reason, we will simply use the terms stationary or non mobile when speaking of such systems. It is also important to note that there is probably no system that is truly not mobile because just about any system may be moved. We will assume that cranes, trucks, or other large vehicles are not required for moving our mobile systems! A mobile system should be movable very easily by just one person.

There are four pieces to the mobile problem: the mobile user, the mobile device, the mobile application, and the mobile network. We will distinguish the mobile user from the stationary user by what we will call the *mobile condition:*

the set of properties that distinguishes the mobile user from the user of a typical, stationary computing system. We will wrap the differences between typical devices, applications, and networks with mobile devices, applications, and networks into a set of properties that we will call the *dimensions of mobility*: *the set of properties that distinguishes the mobile computing system from the stationary computing system.* (Alatalo, 2001). Once we have some understanding of the mobile problem, we will look at some established nonproprietary methodologies and tools of the software industry trade such as Unified Modeling Language (UML). (Audu, 2000)

Although small systems are easy for a single person or a small group to comprehend and develop, large systems are more difficult to design successfully, because there are often many people and entities controlling different aspects of the system and defining how they should work from their own professional specialty or prerogative. For example, a large company requesting a new piece of software might assign the job to a project manager who has a thorough understanding of the overall system requirements, whereas a software developer assigned to work on the system is likely to care more about the ways that individual portions of a system work on a detailed level and less about the practical requirements of users and management. Similarly, an end user of the system is likely to care about how the user interface is organized and that the software is built to facilitate ease of use for everyday users, rather than that a particular software component was designed explicitly or that the project fulfills the stated requirements that its originator decided on. The process of building software can be very complex, and, moreover, there are few cases where a single person has full comprehension of how a system should be designed to fulfill all of its requirements. (Bauer, 2000)

Designing a system that takes into consideration all of the different requirements for the system, from the viewpoint of its stakeholders, developers,

users, domain experts, and others, and which still can adapt to change readily and without causing unforeseen problems is next to impossible without defining the system in a manner that illustrates the various facets of a system, but still recognizing a common set of entities between those facets. Because the UML uses common elements in the different diagrams, it becomes much easier to see the ramifications of a change throughout a system.

During the development process, team members often let their individual goals take priority over the project at the expense of the business goals for which it was designed. Their misguided targets are often caused by an adherence to outdated requirements and their inaccurate interpretation of them. So keeping a model synchronized with the requirements that are being defined while maintaining accuracy becomes of the utmost importance. (Fowler and Scott, 1999).

The solution to the problem leads us to modeling. Because modeling helps the design phase so significantly, a by-product often reduced costs of the system. Furthermore, it is an effective way to ensure at the outset that a system can be built, that the costs of doing so are not unreasonable, and that the system will fulfill the business requirements and meet the needs of its users. Modeling is not unique to software. It is used in a variety of disciplines to think through a system or product, describe it, and discover design flaws before it is built.

Modeling is used in architecture, in mathematics, in the sciences (seismology is a good example), in civil engineering, in auto manufacturing, and in an almost infinite number of things that are conceptually complex and benefit from a modeling illustration of the problem at hand, its ability to encourage understanding of its parts, and the facilitation of communication among the people involved. You can think of the model as a blueprint of the software system to be built. A model should include different perspectives of the system from the viewpoints of the various team members, such as developers, end

users, and the entity that determined the need for the system and instigated its development. The UML provides different views based on who is interpreting the model and in what way. (Bauer, 2001)

What the UML provides in a nutshell, then, is a manner of modeling software that provides a full range of views, from very general overviews of how the system works as a whole, to detailed interactions and descriptions of how each object functions and communicates. Modeling a software system has many benefits and goals. First, modeling helps people to visualize a system as they want it to be. It provides a template for constructing the system, which specifies details of system implementation in a specific enough way that software developers can implement it rapidly and with fewer work stoppages to clarify requirements. By separating system development by architecture, UML allows architects to focus on building systems, whereas developers are able to implement them more rapidly because the system's various components are already defined for them. Architects can specify how something should be built without implementing it themselves, yet this can be done in a specific manner that essentially gives the developer a blueprint for the software. (Baragiy, 1998).

Another benefit of UML is that it helps us to document the decisions that are made throughout the design process. In complex systems, it is difficult for a single person to understand both a specific component of a system and its context in the system as a whole. One way of dealing with this limitation is to narrow our focus to one aspect at a time. UML facilitates this by allowing us to model individual parts of a system while also providing a broad overview.

More importantly, however, UML, because it is a standardized language that is well documented by a detailed specification, can be understood by anyone with sufficient training. By providing a standard, UML encourages application

designers to use a consistent vocabulary and methodology. The more complex the project, the higher the likelihood that you would fail to complete it successfully or that it will not be completed as intended. Worse, many systems start out simply and then become more complex; thus the initial simple design fails to encompass the complexity that the system grows into. When this happens without an adequate modeling system, things rapidly spiral out of control and result in heaping piles of "spaghetti code" (code so unwieldy that it looks like spaghetti in a visual sense). (Dewan, 2002)

Security and privacy are of utmost importance to mobile-based services. Without providing proper security and privacy, few users are willing to use a system that can reveal their current location or history of locations to third parties. Examples of problems that may arise if proper security is not implemented for location services are unwanted marketing, invasion of privacy by governmental or commercial entities, and identity theft or other criminal activities. There are several aspects to security and privacy of location information, the most important are the following:

1. *Access Security*: There must be a proper authentication and authorization mechanism in place for those systems that access the location of a given device. Any system that can obtain location information must in turn provide secure access to any related data through proper authentication and authorization.

2. *Data Security*: Any system used to cross-reference any information that identifies the user associated with a device through profiles, billing, etc. must be completely secured. The content that specifies the location of the device must be transmitted through a secure mechanism (e.g., encryption).

3. *User Control*: The user must have control in specifying whether the location of his or her device is shared with any secondary systems within or outside of the primary wireless network. (Claessens, 2003).

1.2 Problem Description

Mobile security is the concept of complementing the generic user interfaces and render graphical user interfaces for a variety of visual text-driven devices such as personal digital assistants to provide and ensure optimum security for the mobile application users. The security grouping levels determine access levels granted in the application for each user.

1.3 Aims and Objectives

This project is aimed at:

- Keeping out malicious parties who are trying to gain access to things that they are not allowed to access and ensuring that information and system access in a mobile applications are not inadvertently given to parties actively seeking a system breach.

To achieve the above aforementioned aims, we need to:

(1) Introduce a taxonomy of mobile applications security problems.
(2) Look at a few approaches in solving these problems.
(3) Review those problems that remain unsolved.
(4) Look at the picture of mobile application design and see where security concerns may be.

1.4 Scope and limitation of the Study

The research borders on the development of a mobile portable software or application to be used with mobile devices. It is meant to simulate the basic functionalities of the mobile device in a safe platform. The application is expected to work on all mobile devices.

1.5 Significance of the Study

Security is always one of the biggest concerns when designing any application, particularly distributed applications. Unfortunately, there remain many unresolved problems with security concerns of mobile applications. This project provides some solutions to mobile applications security problems. The research work is particularly relevant to individuals or organisations who desire a simple, secure and relatively cheap application used for communication. Its security features makes it especially very useful, dynamic and professional.

1.6 Definitions of Basic Terms

Some basic terminologies needed to understand this project are:

1. *System Requirements View*: This view is defined by the end users' interaction with the system and other systems and is manifest in the form of a use case diagram.

2. *Design View*: This view is used where the system vocabulary is defined. Diagrams included in the design view include class, object, interaction, state, and activity diagrams.

3. *Process View*: This view models the processes and procedures of a system. Diagrams related to the process view are the same as for the design view, but with an emphasis on the active classes.

4. *Implementation View*: This view includes diagrams that are useful to software developers as they create the system and includes the sequence and collaboration diagrams.

5. *Deployment View*: This view describes the system from the viewpoint of a system engineer.

6. *System*: A collection of subsystems organized for a purpose, described by a set of models, possibly from different viewpoints.

7. *Subsystem*: A grouping of elements that constitute a specific behavior offered by the containing elements.

8. *Model*: An abstraction representing a complete and self-consistent simplification of reality created to better understand the system.

9. *View*: A projection into the organization and structure of a system, focused on one aspect of that system.

10. *Diagram*: Semantically, in the UML specification, a graphical presentation of a set of elements.

11. *Class*: UML classes are perhaps the single most frequently used artifact in UML. UML classes encapsulate the attributes and behaviors shared by a certain group of entities. UML classes closely follow the definition of classes in object oriented programming.

12. *Interface*: A named set of operations that characterize the behavior of an element.

13. *Data Type*: A type with values that have no individual identity. These can include primitive data types, built-in data types, and enumerated types.

14. *Signal*: An asynchronous message sent from one instance to another to communicate things such as state, status, and events.

15. *Component*: A physical element of a system that provides the realization of a set of interfaces. Components can include source code, executable code, libraries, and data files.

16. *Node*: A physical element of a system that is able to do computations. Nodes exist at run time and typically have memory and processing capabilities.

17. *Use Case*: A set of action sequences whose result is of value to a particular actor, as well as variant cases of those sequences. In the proposed UML version 2.0 draft, the relationship between use cases and state diagrams are explicit.

18. *Subsystem*: A group of elements that specify the behavior of its containing elements.

19. *Class diagrams* show classes, interfaces, and collaborations and the relationships among them. Class diagrams are used to represent the static design of a system.

20. *Object diagrams* show a group of objects and their relationships. Object diagrams show static views of objects, which are snapshots of a system at a given point in time. Object diagrams, like class diagrams, show the static view of a system, but from the perspective of a specific scenario, rather than a general case.

21. *Collaboration diagrams* are a type of interaction diagram and are semantically equivalent to sequence diagrams. They emphasize the organization of and relationships among objects that send and receive messages. Collaboration diagrams show a set of objects involved in an interaction, the relationships among them, and the messages they send and receive. Collaboration diagrams are used to illustrate the dynamic view of the system.

22. *Sequence diagrams*, like collaboration diagrams, are interaction diagrams. They are semantically equivalent to collaboration diagrams. When designing a system, in fact, you often start with a sequence diagram and then turn it into a collaboration diagram to determine the structure. Sequence diagrams emphasize the order of messages at a moment in time. They show a group of objects and the messages that are sent and received arranged sequentially according to their temporal progression. Sequence diagrams are used to illustrate the dynamic view of the system.

23. *Activity diagrams* show the dynamic view of the system by capturing the flow from one activity to the next within a system and are semantically equivalent to state diagrams. Activity diagrams model a group of activities and the flow of activity, sequential or branching, from one to the next, as well as the objects that participate in that flow, either as users of the system or recipients of the action. Activity diagrams typically emphasize the flow of control among objects but can be used for more generic purposes as well.

24. *State chart diagrams* show a State Machine, which includes states, transitions, activities, and events, and are semantically equivalent to activity diagrams. Like activity diagrams, they show a dynamic view of the system. State chart diagrams, or state diagrams, are particularly important in modeling how a particular class, interface, or collaboration behaves and are used to illustrate behaviors that are ordered by events.

25. *Component diagrams* model physical software components, such as source code, libraries, and executables, and the relationships among them, particularly as they relate to realized interfaces. They are used to model a static view of the system's implementation and typically map to classes, interfaces, or collaborations.

26. *Deployment diagrams* model a set of nodes—that is, computational resources—and the relationships among them. As such, deployment diagrams model a static view of a system's deployment. Deployment diagrams are related to component diagrams, because a node typically contains one or more components.

27. *Use case diagrams* show a set of scenarios depicting interactions with the system and the resulting behavior. They show the relationships between the system and its users and together represent snapshots of the system in action or its static views. They are analogous to the film industry's storyboards used for a movie production.

(Fowler and Scott, 1998; Bauer, 2001; Graham 2002)

CHAPTER TWO

LITERATURE REVIEW

2.0 Advances in Computing Technology

One of the very first computing machines, the abacus, which was used as far back as 500 B.C., was, in effect, a mobile computing system because of its small size and portability. As technology progressed, the abacus evolved into the modern calculator. Most calculators today are made with an entire veer of mathematical functions while retaining their small size and portability. The abacus and calculators became important parts of technology not only because of their ability to compute but also because of their ease of use and portability. You can calculate the proceeds of a financial transaction anywhere as long as you had an abacus in 500 B.C. or have a calculator today. But, calculating numbers is only one part of computing. (Afaur, 2002)

Other aspects of computing, namely storage and interchange of information, do not date as far back as the abacus. Though writing has always been a way of storing information, we can hardly call a notebook a computing storage mechanism. The first mobile storage systems can be traced back only as far as the advent of the age of electronics.

A mobile computing system, as with any other type of computing system, can be connected to a network. Connectivity to the network, however, is not a prerequisite for being a mobile computing system. Dating from the late 1960s, networking allowed computers to talk to each other. Networking two or more computers together requires some medium that allows the signals to be exchanged among them. This was typically achieved through wired networks. Although wired networks remain the predominant method of connecting computers together, they are somewhat cumbersome for connecting mobile

computing devices. Not only would network ports with always-available network connectivity have to be pervasive in a variety of physical locations, it would also not be possible to be connected to the network in real time if the device were moving. Therefore, providing connectivity through a wired system is virtually cost prohibitive. This is where wireless communication systems come to the rescue.

By the 1960s, the military had been using various forms of wireless communications for years. Not only were wireless technologies used in a variety of voice communication systems, but the aviation and the space program had created great advances in wireless communication as well. First, the military developed wireless communication through line of sight: If there were no obstacles between any two parts, you could send and receive electromagnetic waves. Then came techniques that allowed for wireless communication to encompass larger areas, such as using the atmosphere as a reflective mechanism. But, there were limitations on how far a signal could reach and there were many problems with reliability and quality of transmission and reception.

By the 1970s, communication satellites began to be commercialized. With the new communication satellites, the quality of service and reliability improved enormously. Still, satellites are expensive to build, launch, and maintain. So the available bandwidth provided by a series of satellites was limited. In the 1980s cellular telephony technologies became commercially viable and the 1990s were witness to advances in cellular technologies that made wireless data communication financially feasible in a pervasive way. (Cheng, 2005)

Today, there are a plethora of wireless technologies that allow reliable communication at relatively high bandwidths. Of course, bandwidth, reliability, and all other qualitative and quantitative aspects of measuring wireless technologies are relative to time and people's expectations (as seems

These are some of the questions that need to be answered before using LBS as monitoring a person can have psychological effect on the person being monitored. In case of monitoring criminals or suspects by police or security agencies the question of individual freedom came, as enforcing someone's freedom is not at all ethical when the person is only suspected of committing the crime.

Control (Legal) – Commonly GPS and other LBS devices are used to control and offer various types of services to the user. Personally it controls one's own direction of moving in guiding the right way. In case of child tracking, parents have exclusive right to look after their children, as it is not possible for the young ones to make their own decision. So it is their legal right to monitor their children thereby reflecting a sense of caring. In case of law enforcement, special laws provide legal rights to police or security departments to keep an eye on criminals or suspects.

Trust (Social) – In social life trust is the most essential part in human relationship. However, the use of LBS is being practiced in low trust conditions. Monitoring someone with the help of tracking system really affects personal relationship but as far as tracking criminals by cops or tracking children by parents are concerned, it is for the welfare of the individual & society.

Privacy (Ethical) – As a human being, everyone has the right to privacy or being free from intrusion or disturbances in one's personal life. But in case of LBS or any other telecommunication technologies dealing with transformation of various kinds of information, it becomes essential to provide adequate security to these kinds of data for not being misused by any unauthorized person. Tracking and monitoring someone without his/her consent is purely unethical so needs high level of security. But again as in case of law and order where tracking devices are used to monitor criminals becomes essential for the

to be with everything else in life!). Though most wireless networks today can transmit data at orders of magnitude faster speeds than just ten years ago, they are sure to seem archaically slow soon. It should, however, be noted that wired communication systems will almost certainly always offer us better reliability and higher data transmission bandwidths as long as electromagnetic communications is the primary means of data communications.

The higher frequency sections of the electromagnetic spectrum are difficult to use for wireless communications because of natural noise, difficulty of directing the signal (and therefore high losses), and many other physical limitations. Since, by Nyquist's principle, the bandwidth made available by any communication system is bound by the frequencies used in carrying the signal, we can see that lack of availability of higher frequency ranges places a limitation on wireless communication systems that wired communication systems (such as fiber optic–based systems) do not have to contend with. (Lathi, 1989)

Because the greatest advances in mobile communications originated in the military, it is no surprise that one of the first applications of wireless communication for mobile computing systems was in displaying terrain maps of the battlefield. From this, the global positioning system (GPS) evolved so that soldiers could know their locations at any given time. Portable military computers were provided to provide calculations, graphics, and other data in the field of battle. In recent years, wireless telephony has become the major provider of a revenue stream that is being invested into improving the infrastructure to support higher bandwidth data communications.

When object-oriented languages began to appear in the mid-1970s, they were conceptually new for software developers and architects accustomed to using procedural languages. Because procedural languages have a well-defined flow, they are easy to model with simple flow charts. That model was only able to

express a small part of how object-oriented systems interoperate. Something else was needed to illustrate the way that object-oriented systems interacted.

There was no shortage of ideas about how object-oriented software should be modeled, as a variety of different methodologies quickly emerged. The late 1980s saw a plethora of competing methodologies and plenty of experimenting within those. By 1994 there were close to fifty of them competing to become the de facto standard. As we mentioned earlier, the result of not standardizing on a single methodology is a fragmented vocabulary and the inability to communicate. By providing a standard, UML encourages a consistent methodology. (Bauer, 2009)

2.2 Review of Related Works on Mobile Applications Security

Jesse Burns in his book, Mobile Application Security on Android state that Android has a unique security model, which focuses on putting the user in control of the device. Android devices however, don't all come from one place, the open nature of the platform allows for proprietary extensions and changes. These extensions can help or could interfere with security, being able to analyze a distribution of Android is therefore an important step in protecting information on that system.

Android *permissions* are rights given to applications to allow them to do things like take pictures, use the GPS or make phone calls. When installed, applications are given a unique UID, and the application will always run as that UID on that particular device. The UID of an application is used to protect its data and developers need to be explicit about sharing data with other applications. Applications can entertain users with graphics, play music, and launch other programs without special permissions.

Malicious software is an unfortunate reality on popular platforms, and through its features Android tries to minimize the impact of malware. However, even

unprivileged malware that gets installed on an Android device (perhaps by pretending to be a useful application) can still temporarily wreck the user's experience. Users in this unfortunate state will have to identify and remove the hostile application. Android helps users do this, and minimizes the extent of abuse possible, by requiring user permission for programs that do dangerous things like:

– directly dialing calls (which may incur tolls),
– disclosing the user's private data, or
– destroying address books, email, etc.

Generally a user's response to annoying, buggy or malicious software is simply to uninstall it. If the software is disrupting the phone or mobile device enough that the user can't uninstall it, they can reboot the phone (optionally in safe mode, which stops non-system code from running) and then remove the software before it has a chance to run again.

Security Responsibilities of Developers

Developers writing for Android need to consider how their code will keep users safe as well as how to deal with constrained memory, processing and battery power. Developers must protect any data users input into the device with their application, and not allow malware to access the application's special permissions or privileges. How to achieve this is partly related to which features of the platform an application uses, as well as any extensions to the platform an Android distribution has made.

One of the trickiest big-picture things to understand about Android is that every application runs with a different UID. Typically on a desktop every user has a single UID and running any application launches runs that program as the users UID. On Android the system gives every application, rather than every person, its own UID. For example, when launching a new program (say by starting an *Activity*), the new process isn't going to run as the launcher but with its own identity. It's important that if a program10 is launched with bad parameters the

developer of that application has ensured it won't harm the system or do something the phone's user didn't intend. Any program can ask Activity Manager to launch almost any other application, which runs with the application's UID.

Android Permissions Review

Applications need approval to do things their owner might object to, like sending SMS messages, using the camera or accessing the owner's contact database. Android uses *manifest permissions* to track what the user allows applications to do. An application's permission needs are expressed in its AndroidManifest.xml and the user agrees to them upon install. When installing new software, users have a chance to think about what they are doing and to decide to trust software based on reviews, the developer's reputation, and the permissions required. Deciding up front allows them to focus on their goals rather than on security while using applications. Permissions are sometimes called —manifest permissions‖ or —Android permissions‖ to distinguish them from file permissions.

To be useful, permissions must be associated with some goal that the user understands. For example, an application needs the READ_CONTACTS permission to read the user's address book. A contact manager app needs the READ_CONTACTS permission, but a block stacking game shouldn't. Keeping the model simple, it's possible to secure the use of all the different Android inter-process communication (IPC) mechanisms with just a single kind of permission. Starting *Activities*, starting or connecting to *Services*, accessing *ContentProviders*, sending and receiving broadcast *Intents*, and invoking *Binder* interfaces can all require the same permission. Therefore users don't need to understand more than —My new contact manager needs to read contacts‖.

Once installed, an application's permissions can't be changed. By minimizing the permissions an application uses it minimizes the consequences of potential security flaws in the application and makes users feel better about installing it. When installing an application, users see requested permissions in a dialog similar to the one shown. Installing software is always a risk and users will shy away from software they don't know, especially if it requires a lot of permissions.

Ken(2009) views, **Application security** as an all encompassing measures taken throughout the application's life-cycle to prevent exceptions in the security policy of an application or the underlying system (vulnerabilities) through flaws in the design, development, deployment, upgrade, or maintenance of the application, .

Applications only control the use of resources granted to them, and not *which* resources are granted to them. They, in turn, determine the use of these resources by users of the application through application security. (Online Wikipedia, 2010)

Open Web Application Security Project (OWASP) and Web Application Security Consortium (WASC) updates on the latest threats which impair web based applications. This aids developers, security testers and architects to focus on better design and mitigation strategy. OWASP Top 10 has become an industrial norm is assessing Web Applications.

The Windows Mobile documentation uses a set of security terms that have specific meanings, and understanding these terms will help you understand security on Windows Mobile devices.

Trusted and Normal

A *trusted* process can call any API and write to any registry key. There are essentially no limits on what it is allowed to do. A *normal* process is forbidden from calling certain APIs and writing to certain registry keys. For the list of APIs and registry keys.

Normal execution mode is designed to reduce the amount of code that needs Trusted execution access to the device. By using Normal mode to run an application, you can reduce the risk that an error-prone application can cause accidental damage on the device. Using Normal mode can also reduce the likelihood that malicious or error-prone code misuses an application, and it can help minimize the damage that can result from security vulnerabilities in application code.

Although Normal execution mode is designed to reduce access to system and device resources, it should not be considered a primary method to contain the damage caused by malicious code that is written with the intention to do harm. Running code from untrustworthy sources comes with risk. The primary defense against malicious code is to not run it at all on the device. Windows Mobile devices implement code signing that can be used for this purpose.

Privileged, Unprivileged, and Unsigned

An *privileged* application is signed with a certificate that is in the privileged certificate store on the device. An *unprivileged* application is signed with a certificate that is in the unprivileged certificate store on the device. An application that is unsigned has no certificate.

Privileged certificate means a certificate that is in the privileged certificate store on a specific device. Note that there is nothing intrinsic to the certificate

itself that is privileged. It is only the presence of the certificate in the privileged certificate store on a particular device that makes the certificate privileged.

Note that the terms trusted and normal refer to how an application *runs*, whereas privileged, unprivileged, and unsigned refer to how an application is *signed*. This is an important distinction; for example, it is entirely possible to have an unsigned application run trusted.

One-Tier and Two-Tier

A device that is *one-tier* is one where any process that runs, runs trusted. A device that is *two-tier* is one where a process runs either trusted or normal. On a two-tier device, only privileged applications run trusted.

Currently, Pocket PC only supports one-tier. Smartphone supports either one-tier or two-tier, but the vast majority of Smartphones are two-tier.

Mobile2Market

Mobile2Market is a program operated by Microsoft for independent software vendors that provides a unified market for applications for mobile devices. The code signing program within Mobile2Market is open to all application vendors for Windows Mobile devices, and it allows you to have your application signed with one of the Mobile2Market certificates. All Windows Mobile 2003 devices that currently ship contain the Mobile2Market unprivileged certificate, and that is likely to remain true of Windows Mobile 5.0 devices.

2.3 Security/Privacy of Mobile Application devices and Location Based Serivc

The mobile devices has become the most widely deployed computing platform in the world. Analysts predict some 2.6 billion mobile phones in regular use by 2009, with the sales of smartphones representing the fastest growing market [BBC, 2005]. For many people, the mobile phone is the first computer they

encounter, and certainly they only computer they carry with them most of their waking moments.

Mobile computers in general, and mobile phones in particular, present unique challenges not only in terms of user interface, battery life, and form factor, but also in terms of ensuring their users' privacy and security. Emerging mobile payment and ticketing solutions require the secure transmission and storage of financial information, while electronic health records or access certificates/tokens might imply the use of highly sensitive personal information on such devices. Wireless connectivity such as WiFi, Bluetooth, or NFC facilitates decentralized tracking, while content sharing applications and collaborative games enable unobtrusive social data mining. Providing security for such wireless communication is a general problem, further complicated by the fact that there is hardly any a priori information about potential communication partners.

Advances in mobile hardware architectures and tremendous growth of high-speed wide-area cellular systems deliver anytime and anyplace availability of network-based information services to the owners of mobile computing devices. Ever-growing popularity of mobile devices, such as smart phones and netbooks, drives the demand for the development of innovative software and hardware architectures, applications, and network services aimed at these devices. As a result, we are witnessing a paradigm shift in the way people use computing technology in their everyday lives.

Behind this dramatic increase in the use of mobile communications and mobile devices, there is an implicit assumption of information security, privacy and trust (SPT). However, for this assumption to be correct, wireless communication and backend infrastructure, mobile device architectures, and mobile applications must be designed from the ground up with security and privacy concerns in mind. Security, privacy and trust cannot be an afterthought

sake of society as a whole. Here, social security is counted higher than Individual safety and security.

Security (Technological) – Again for maintaining privacy, security system should be strong. Every technology has both positive and negative impact on human life and LBS also has shortcomings by locating accurate information data or even easily given access to unauthorized person. On one hand LBS enhances both national and personal security but create another problem for the privacy of individual by not providing a foolproof security system to that highly sensitive information stored in its database. For obtaining security, one needs to do a little compromise on his/her privacy but to what extent is a question.

However, in the whole privacy and security issues of LBS, there are chiefly four points came as control, trust, privacy and security as legal, social, ethical and technological aspects. But all four are mutually exclusive as control decreases trust, trust enhances privacy, which needs security, and security again increases control.

Managing security and privacy policies is known to be a difficult problem. Studies have shown that **lay users often do not know their own policies or are unable to express them**. Even in a desktop computing environment, end users have great difficulty using the Windows XP file permission system to create security policies for file access. In **mobile and pervasive computing settings**, this situation is often exacerbated by the limitations of access devices and the numerous tasks users concurrently engage in. To make matters worse, desired security and privacy settings are not just difficult to articulate, but they also tend to change over time. In short, emerging demands for empowering end

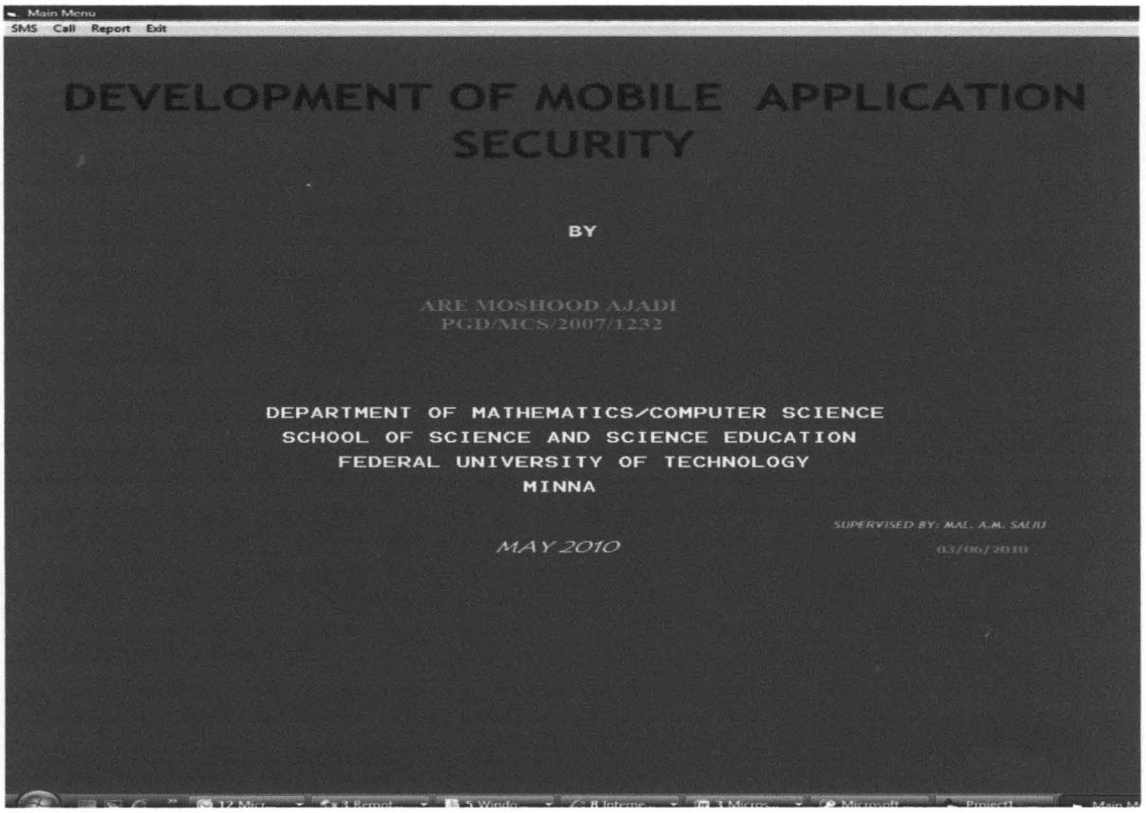
users to set up policies are often unrealistic. This in turn may result in new sources of vulnerability and high levels of user frustration.

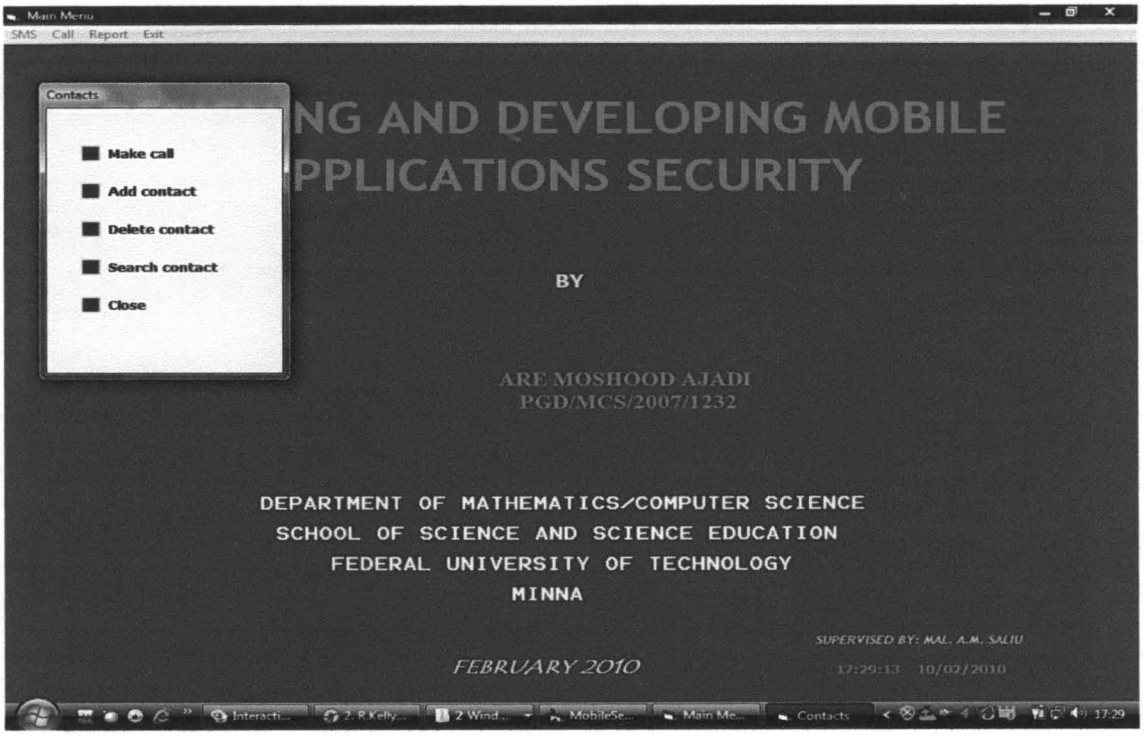
We believe it is important that new user interfaces be developed to effectively and efficiently support lay users in understanding and managing security and privacy policies – their own as well as those implemented by systems and individuals with whom they interact. Solutions in this area have traditionally taken a relatively narrow view of the problem by limiting the expressiveness of policy languages or the number of options available in templates, restricting some decisions to specific roles within the enterprise, etc. As systems grow more pervasive and more complex, and as demands for increasing flexibility and delegation continue to grow, it is imperative to take a more fundamental view that weaves together issues of security, privacy and usability to:

- Systematically evaluate key **tradeoffs between expressiveness, tolerance for errors, burden on users and overall user acceptance**, and
- Develop novel mechanisms and technologies that help **mitigate these tradeoffs**, maximizing accuracy and trustworthiness while minimizing the time and effort required by end users.

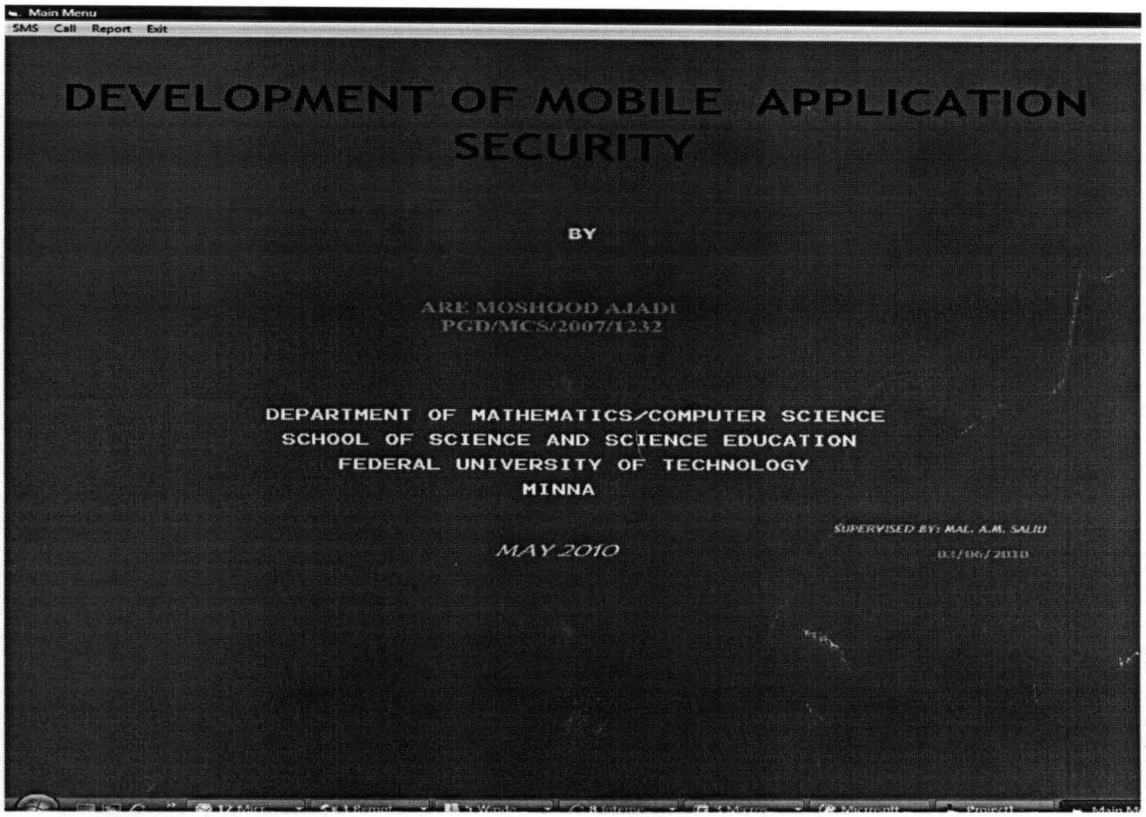
The objective of this project is to develop new interfaces that combine **user-centered design** principles with **dialog, explanation and learning technologies** to assist users in specifying and refining policies. This involves developing **policy authoring tools for a growing collection of pervasive computing applications and evaluating the effectiveness of these tools with users in longitudinal studies.** Evaluation metrics look at both accuracy and overall user acceptance, including user burden. **Users should feel that they have adequate control over the behavior of the applications they interact with.**

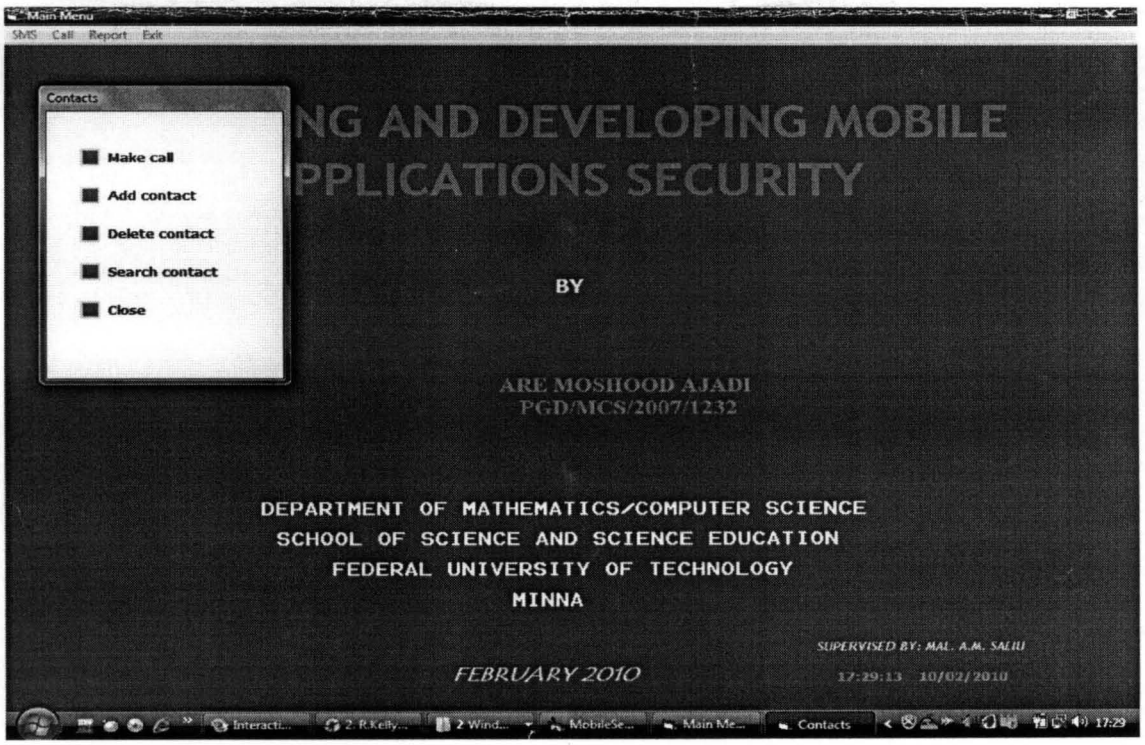
2.4 PROGRAM OUTPUT



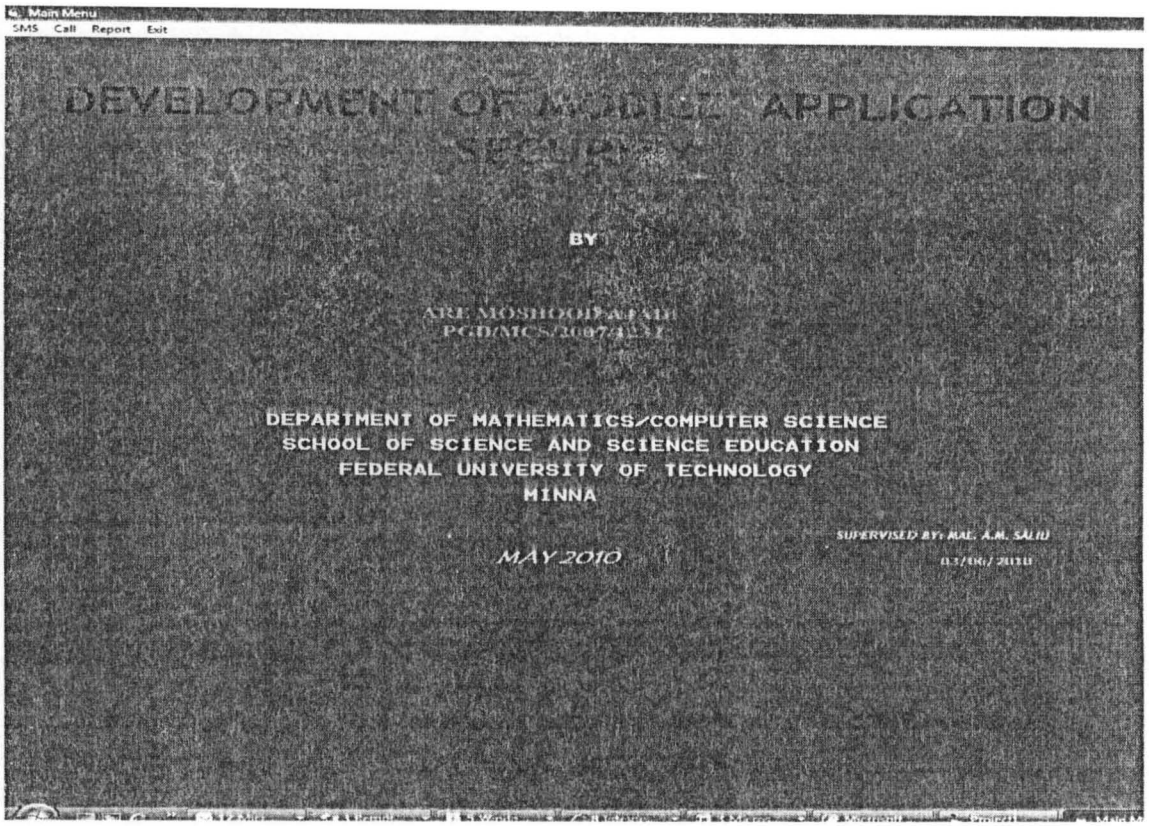


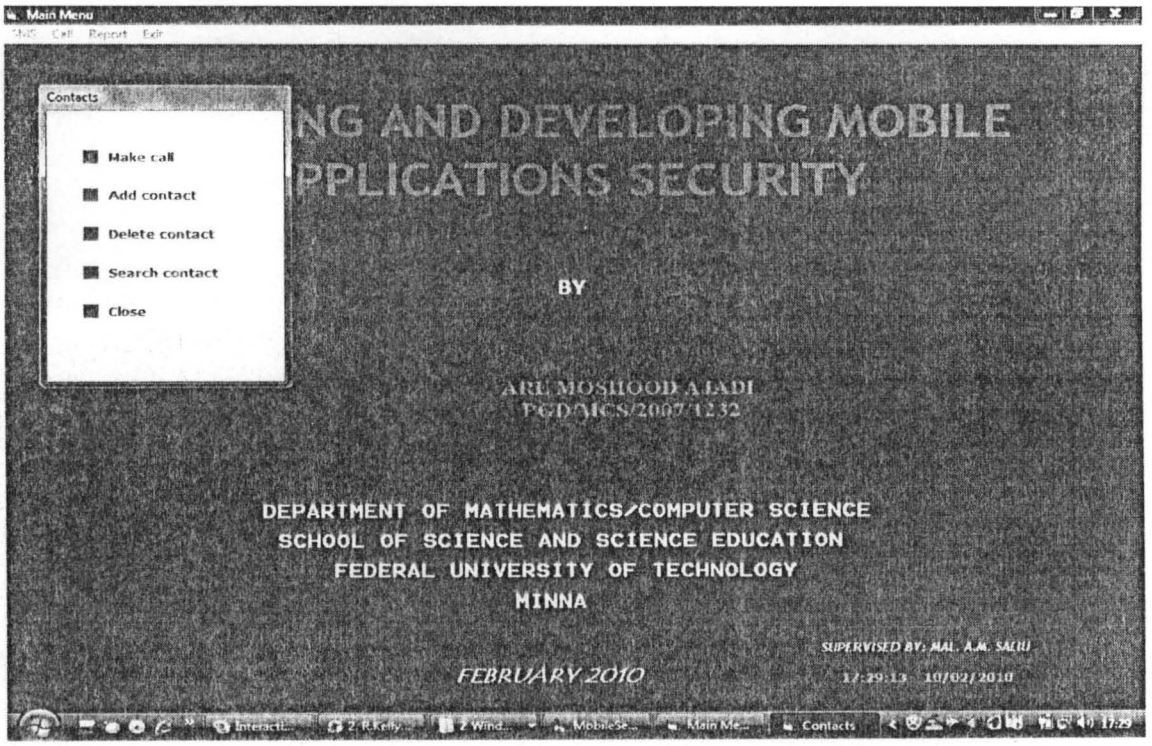
2.4 PROGRAM OUTPUT

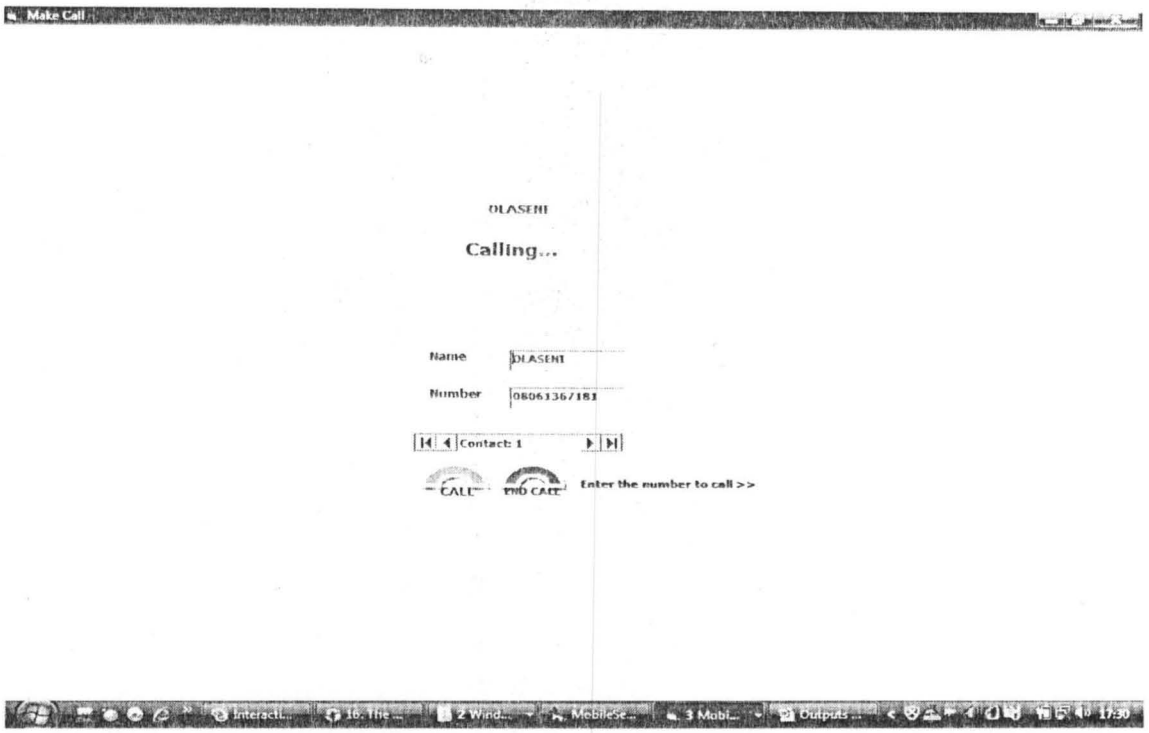


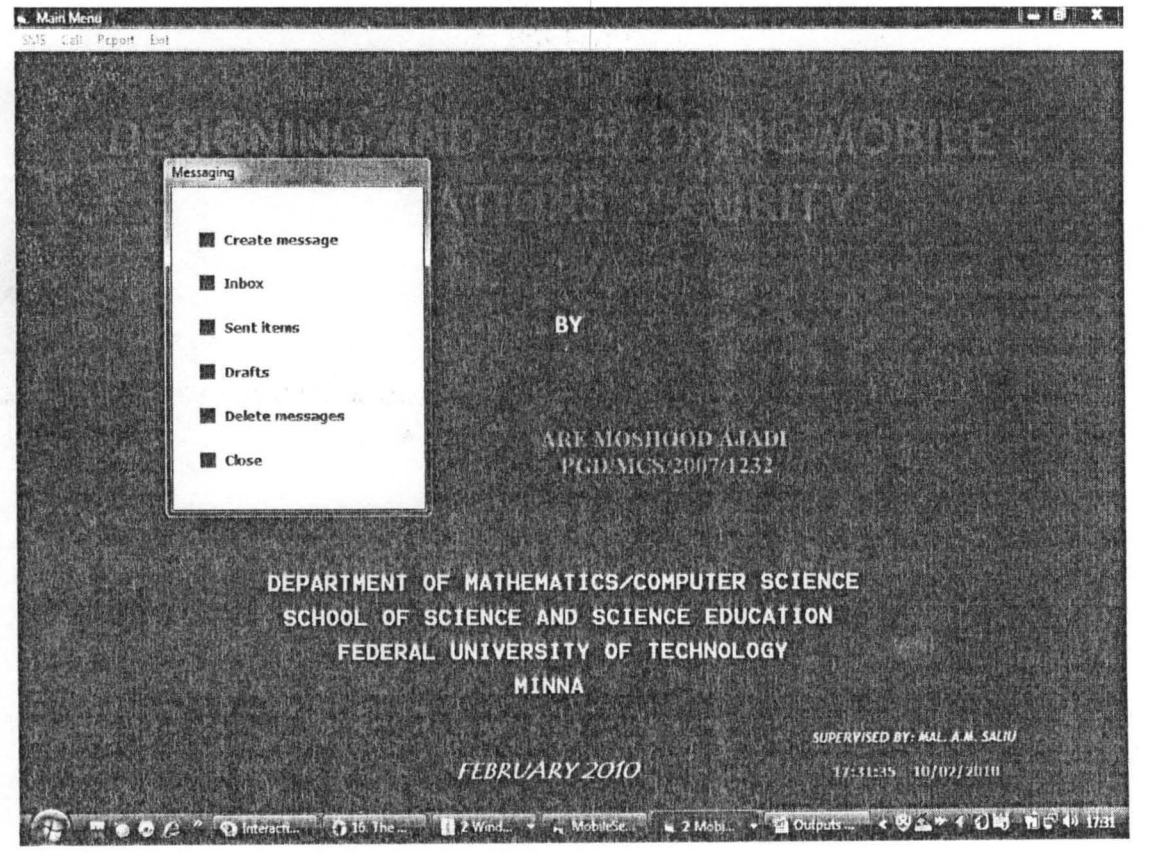


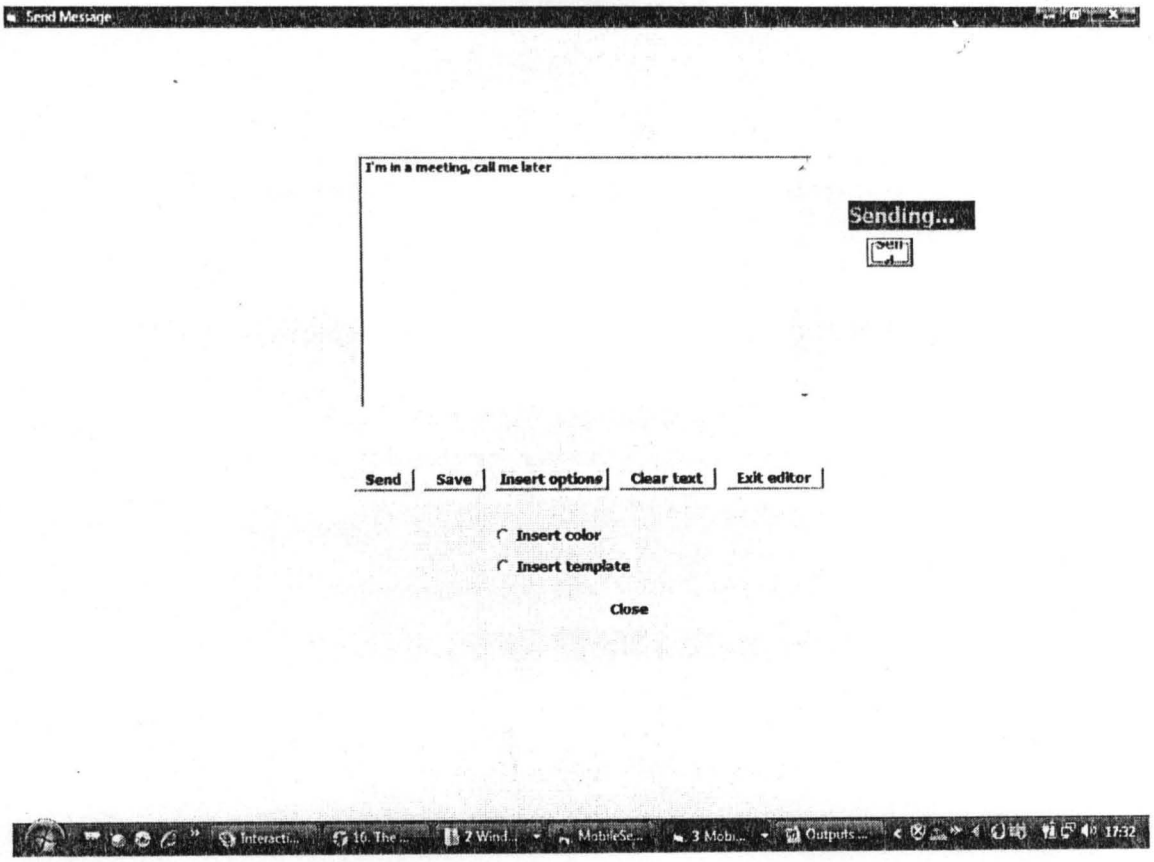
2.4 PROGRAM OUTPUT






Make Call
OLASENI
Calling...
Name
OLASENI
Number
08061367181
Contact: 1
CALL
END CALL
Enter the number to call >>


Main Menu
SMS Call Report Exit
Messaging
Create message
Inbox
Sent items
Drafts
Delete messages
Close
BY
ARE MOSHOOD AJADI
PGD/MCS/2007/1232
DEPARTMENT OF MATHEMATICS/COMPUTER SCIENCE
SCHOOL OF SCIENCE AND SCIENCE EDUCATION
FEDERAL UNIVERSITY OF TECHNOLOGY
MINNA
SUPERVISED BY: MAL. A.M. SALIU
FEBRUARY 2010
17:31:35 10/02/2010


Send Message
I'm in a meeting, call me later
Sending...
Send
Save
Insert options
Clear text
Exit editor
Insert color
Insert template
Close
Interacti...
10. The ...
2 Wind...
MobileSe...
3 Mobi...
Outputs ...
17:32

CHAPTER THREE

3.0 ANALYSIS AND DESIGN OF MOBILE SECURITY

3.1 Controlled Access

Security and privacy are of utmost importance to mobile-based services. Without providing proper security and privacy, few users are willing to use a system that can reveal their current location or history of locations to third parties. Examples of problems that may arise if proper security is not implemented for location services are unwanted marketing, invasion of privacy by governmental or commercial entities, and identity theft or other criminal activities. There are several aspects to security and privacy of location information, The most important are the following:

1. *Access Security*: There must be a proper authentication and authorization mechanism in place for those systems that access the location of a given device. Any system that can obtain location information must in turn provide secure access to any related data through proper authentication and authorization.

2. *Data Security*: Any system used to cross-reference any information that identifies the user associated with a device through profiles, billing, etc. must be completely secured. The content that specifies the location of the device must be transmitted through a secure mechanism (e.g., encryption).

3. *User Control*: The user must have control in specifying whether the location of his or her device is shared with any secondary systems within or outside of the primary wireless network.

Some of the key features of a system that offers location-based service and the clients to such a system must be the following:

1. The system must allow the users to configure policies regarding where and when their location information may be obtained and/or shared.

2. The system must allow the users to specify with whom their location information may be shared.

3. The system must automatically remove all historical data about a user's location unless otherwise allowed by the user.

4. The mobile-based service must not expose specific information to its client systems on why the location of a particular user may not be available. For example, the client system must not be able to request whether the user has specified to be unavailable to that particular client or during a particular time window.

5. The error margin in the exact location of the user must not be provided unless specified by the user.

6. The client system must specify a reason for which the location is obtained. Only trusted systems should be able to obtain location information.

All other features typical to a secure computing interface such as authentication, authorization, and encryption must be made available by the location-based service to its clients. Of course, what we have outlined here is only a small, but important, subset of features needed for truly secure acquisition and exchange of location information. In Europe and the United States there are existing and evolving legislation to assure the privacy rights of the users.

Then there is the matter of privacy. There are many instances when a user will allow his or her location to be known, but only within a certain range of accuracy and with anonymity. In this context, we describe two metrics that

Beresford and Stajano have developed for measuring location privacy, one based on anonymity sets and the other based on entropy (Beresford and Stajano, 2003). The duo hypothesize that although location information may be exposed, the number of nodes that can link a given location to an actual identity associated with the thing whose location was measured make up the anonymity set. It should seem obvious that the bigger the anonymity set size is, the more private the interactions of the thing with the outside world become. Likewise, the more things inside the set move around and the more the locations that we are measuring move around (entropy), the more privacy the thing whose location is being measured at some fixed time(s) has. This work is significant as a start in allowing us to quantify just how private the location information associated with something or someone may be. This quantification of privacy may eventually be used, directly or indirectly, as a user-adjustable threshold.

The security and privacy of the location of devices that are open to the network is one of the ongoing areas of research, development, and legislation brought about by consumer concerns. Whereas location data about a particular user or aggregates of users can be extremely helpful in many applications, the user must always have the control over whether or not such information is obtained and with whom and which subsystems it may be shared. (Intel, 2003)

3.2 Security in Wireless Networks

We need security for two reasons:

1. To keep out those malicious parties who are trying to get access to things that they are not allowed to access and

2. To ensure that information and system access are not inadvertently given to parties not actively seeking a system breach.

So, the goal is to keep data and system access from being exposed to parties who should not access them, whether or not those parties are actively seeking a breach. Such a breach can happen at different points: hardware, software, and communication channels. Because of this, we can use the OSI model and its taxonomy to group the various types of security concerns for mobile applications. We will start from the top and work our way down:

1. *Application Layer Security*: This the most important layer for securing our mobile application. As software application developers, practically speaking, we have the most control at this layer. Assuming that all other layers below this one are not secure, we can still build a secure application if we exercise due caution at this layer. This does not mean that this is the best course of action to take, but as this is the layer over which we as software application developers have the most control, we need to pay the most attention to it. In the case of standalone applications, the OSI model does not have much meaning as it is primarily used to represent networked applications. Nevertheless, we can think of a standalone application to exist entirely at this level: We have complete control over whatever security features we need to implement. The operating system may provide features that make things easier. For example, for a simple Palm application that uses no networking, the security concerns may entail encrypting all of the data and using sufficient usernames and passwords for authentication and authorization in the application. At this same layer, the application layer, a networked application that uses HTTP for communication on the same platform (Palm OS) may additionally include usage of encrypted communication using techniques such as DS3 or a similar technology as well as authentication for whatever other computing system is communicating with our application.

2. *Presentation Layer and Session Layer Security*: SSL (Secured Socket Layer) is probably the most important technology of interest here. Though there are

other possible techniques that use public and private keys for secure transfer of data, SSL is by far the most popular and the one for which nearly all platforms provide support. If you remember, when we looked atWAP, the security mechanism of WTLS provides the SSL implementation for the WAP protocol. Implementing SSL by which we mean actually writing the SSL specification as a library for your operating system or trying to implement it on an end-to-end system such as WAP) is not something that we will worry about in this text as it is outside of the scope of the typical work that an application developer has to do; it is nothing trivial! What is interesting to note is that the size of the public and private keys may be smaller than desired because of the resource limitations on mobile devices. Keep in mind that the effort to break a security key by brute force is, in the case of the best encryption algorithms, exponentially relative to the size of the public and private keys. Here we encounter a problem that is not solved in a simple manner: If the device does not have enough resources, it is tough to justify spending whatever it has on encryption and decryption. There has been discussion among mobile device vendors in providing hardware-based solutions for SSL to provide a more efficient method for secure communications.

3. *Transport Layer and Network Layer Security (IPSec)*: These layers are, respectively, the home of TCP and IP (as well as other equivalent protocols). Whereas SSL assures that all communications are secure, IPSec assures that the nodes that are communicating are not malicious and masquerading as nodes that they are not. IPSec also provides more low-level encryption and allows us to do "IP Tunneling." IPSec is particularly important in the infrastructure that supports the mobile application. As Mobile IP technologies mature, IPSec will become more and more relevant to the actual mobile device. Because of the lack of deployment prevalence, there are no solid security solutions introduced specific to the needs of Mobile IP (but, as discussed earlier, there are home agents and foreign agents and this architecture brings additional considerations

not properly addressed by IPSec). There are some suggested solutions. For instance, Fasbender, Kesdogan, and Kubits [Fasbender et al.] propose a nondisclosure method that first recognizes the differences between a system based on Mobile IP deployment and a regular IP-based system and then tries to address these new requirements (see the reference for further details). However, these various techniques have yet to be employed in real deployments and stand the test of time and the patience of hackers.

4. *Data Link Layer Security*: This is where things like MAC (Medium Access Control) addresses belong. It is tough to cause a security breach through the data link layer because it is typically hardware implemented. Hardware is also, of course, susceptible to security problems, but hardware vendors typically test much more rigorously than software vendors (as the costs of mistakes are much higher) and it is much more difficult to get significant malicious programs such as viruses onto the hardware to begin with.

5. *Physical Layer Security*: Perhaps the biggest differences between security implementations of mobile systems and stationary systems are a byproduct of the fact that mobile systems are typically connected to the network through a wireless connection. Wired systems, whether fiber-optic cables, coaxial cable, or twisted- pair wires, limit access to the bits and bytes traveling across the communication

channels that they provide inside their physical medium. However, bits and bytes are all over the space between two wireless nodes waiting to be read as there is no limiting "conduit" in the case of wireless communication.

Not only that, but intrusion detection is enormously more difficult in wireless systems where signal attenuations, phase shifts, and other phenomena are part of the physical condition of the network and cannot be used reliably to indicate security breaches.(Yim, 2001)

3.3 Security and Ad Hoc Networking Technologies

Considering this taxonomy of security issues based on the OSI model, the dimensions of mobility, and the mobile condition of the mobile user leads us to the following security issues that are unique from any of those concerns experienced by stationary applications:

1. Secure authentication and authorization of nodes.

2. Secure communications between the authenticated and authorized nodes of the network over a wireless connection (at various OSI layers using the correspondingly appropriate techniques such as SSL at the presentation/session layers).

3. Secure deployment of an application on the target device.

4. Secure storage and retrieval of information on the mobile device.

5. Securing information collected or provided by the mobile application infrastructure (e.g., location information).

6. Securing any conversion of content required for supporting multimodal applications.

7. Securing synchronization and exchange of information among different channels in a multichannel communication environment.

8. Defending against the fraudulent usage of the wireless service.

9. Defending against various Denial of Service attacks that may interrupt service to the network users (mobile application users in our case) or make other security breaches possible. (Intel, 2003; Hansmann, 2002)

In addition to these concerns, once again, we bring up the dimensions of mobility. As we have repeatedly mentioned in this text, the dimensions of mobility are the fundamental bases for those difference we see between mobile applications and stationary applications. Security is a part of most stationary and mobile applications. So, we can go back to the dimensions of mobility to see the differences in requirements, design, and implementations of security between mobile and stationary applications. In other words, we need to consider the following:

1. How do security concerns change when the location of the device and application are changing, when the application is using location information in its internal logic, when there exists some LBS infrastructure, and when the location information must be provided not only securely but also privately?

2. Is security compromised by the QOS? For example, some systems do not appropriately secure dropped packets. Although this level of security may not be important for a given system, we still need to be aware of it. Also, we noted that QOS is a dimension of mobility largely because of the intermittent connectivity of the mobile user but also because the connectivity may be provided through a wireless network. As most wireless networks for consumer mobile devices are cell based, we need to be worried about an entire arena of security problems that occur because of the cell-based architecture such as security at handover points. It is also important to keep in mind that we have to provide offline-security. The user may need to use the application while it is not connected to the network, so we may be required to implement different security mechanisms including implementing authentication and authorization on the device and on the network.

3. Security is almost always dependent on device capabilities as it takes device resources to encrypt/decrypt data. For example, the size of the encryption key

may need to be smaller for some devices than others as they may not have the processing power to encrypt and decrypt the data in a timely manner.

4. The power supply is only important in security if the device has different modes of operation depending on the available amount of power. Obviously, not every application or transaction within the same application requires the same amount of security. The key is to make sure that the security of those transactions that must be secure is never compromised regardless of the device mode of operation.

5. Various user interfaces require different types of security. For those developers who have developed GUI-based applications, the biggest difference is in understanding the very different security techniques used in VUIs. For example, it may be fine to display some secure and private information to the user on the GUI application as he or she may be able to hide it from the surroundings, but the same is not true in the case of a VUI; we do not want to play a text-to-speech clip of the user's bank account balance or, at the very least, we want to give the user the choice to hear or view his or her balance. Voice itself can also be used to test liveliness as well as authentication. These features are very critical. Although there is no way to tell who is typing information on a mobile device (unless the device has a fingerprint reader or some other biometric interface, which is very unlikely), there are proven technologies that not only allow us to recognize a user but also allow us to make sure that the user is live at the time of the recognition and the audio being received by the system is not a recording. Examples of such technologies are included in products offered by Nuance, IBM, and Speech Works.

Another critical issue is to provide proper security for intermodal and inter channel communications. The modes and channels involved in a multimodal user interface are not independent. There are interactions that may be linked to one of many different aspects such as temporal synchronization between

different channels or multimodal sessions that involve some user interface logic to render various components through the appropriate channel and mode. Many security issues are compounded by the multichannel and multimodal nature of interactions in mobile applications.

6. As we have noted, because mobile devices are smaller and cheaper than PCs, they have proliferated greatly. There are more different types of devices in the mobile market; their life cycle is shorter than the life cycle of PCs for many reasons, among which is the much lower cost to manufacture them. The problem of device proliferation, coupled with the distributed nature of mobile applications, magnifies the scalability problem of implementing security. Each user may have many different mobile devices. For example, a typical user may have a mobile device in his or her automobile (telematic device), a cell phone, a PDA, a laptop, and possibly a tablet PC, each of which may connect to the network thorough a distinct channel or set of channels and have its own set of security needs and requirements.

7. Actively interacting with the user presents us with more privacy problems then security problems. Whereas we can use the user's response to the initial transaction for authorization, we must be able to authenticate the user prior to sending out that initial message. Sending the initial interaction or pushed message to the wrong user in itself is a security flaw.

8. In addition to the dimensions of mobility outlined (whose effect on security has been discussed in points 1 through 7), the mobile condition of the user introduces the following new concerns:

a. Mobile devices are more susceptible to theft and loss. It is much more difficult to misplace your PC than it is your phone. The physical size of the device has much to do with this. The smaller things become, the easier it is to lose them. Though location-based technologies can help us in finding the

device, device security and important information may be compromised by the time the device is found or recovered.

b. With mobile users, there is a range of environments to consider in a security policy; with or without a VPN (Virtual Private Network), users may connect to the network directly, through a corporate Internet service provider, or through their own Internet service provider, thereby using a variety of different security guidelines established by different organizations (Clarkin, 2003). These differences may cause security breaches because something that is secure in one network may not be secure in another.

c. As we noted, the life cycle of mobile devices tends to be shorter than their stationary counterparts, the average being somewhere between eighteen and twenty-four months. Rapid development of mobile technology to meet higher user expectations has led to security being seen as too much work in a compressed timeline (Claessens et al., 2003).

d. Many mobile devices use SIM cards. Securing the configuration and reconfiguration of these SIM cards in itself is a security issue. Although this is largely out of the hands of a mobile developer who is developing third-party software for mobile devices, it is crucial to have a system for detecting when the configurations on the SIM card change, thereby leading to a change of behavior or intention on the part of the device.

Having discussed all of these various concerns and how we can categorize them for mobile applications, we must take a further step, before all others, in designing a secure system: *We must determine the threat levels*. This is perhaps the single most neglected step in most systems. One of the anti patterns that we have mentioned has been that, once a solution works, we have a tendency to use that solution for all sorts of problems, whether or not the solution is a good fit. This is often the case for system-wide or application-layer security

implementations. Without the proper threat recognition in which the various levels of threat, sources of threat, and the cost of security are addressed, trying to solve the security problems of a system, whether mobile or not, becomes a haphazard series of jumps between isolated symptoms instead of a systematic solution to the roots of security problems. In the case of mobile applications, what you need to keep in mind while determining the threat levels are the following:

1. Mobile applications are a superset of their stationary counterpart. Therefore, you must take into account all of those concerns of stationary applications.

2. Consider the new security concerns introduced by the various dimensions of mobility and the very distributed nature of mobile applications.

3. Consider the appropriateness of the level of security concern for each part of the mobile application. It is easy to overestimate or underestimate the level of security required for a particular transaction. The requirements-gathering process is critical here. Also, different parts of the application may require different security levels.

Finally, determining the threat level alone will not answer the questions that management will ask. Management is always looking for a return on investment in any project and implementing security is no different. In fact, security tends to be an area that is often not properly assessed. Some numbers that can help here have been published by Stanford and MIT's Sloan School of Management. They have defined a Risk on Security Investment (ROSI) that, based on the empirical evidence they have gathered, is at 21% at the design stage, 15% at the implementation stage, and 12% at the testing stage. This further verifies our approach that security is largely an architectural and a system-wide problem that must be solved at design time (Intel, 2003). We will

not go into the details of the justifications of these numbers; refer to the papers referenced here for those justifications.

Although in this chapter we will concentrate on the security provided by wireless technologies that give the mobile application connectivity to the network, somewhere along the line we will also look at security issues of things that are very unique to mobile applications such as security within ad hoc networking technologies and mobile agent security.

Currently (and likely to be the case for the near future), security is implemented in a very proprietary manner by many wireless networks that provide network connectivity for mobile applications. This is particularly true in the long-range wireless technologies, where it makes sense for the large telecommunication companies to implement proprietary technologies closely tied to their infrastructure assets (such as CDMA equipment). However, there are also standards that give security a full and open treatment such as WAP, Bluetooth.

We have already looked at various wireless technologies that are most relevant to mobile computing. Particularly of interest are Bluetooth and WIFI in the short range family and CDMA-based and GSM/3GPP/TDMA technologies in the long range family. We will look at the various security aspects of these technologies individually in this section.

Generally speaking, we are not only concerned with securely establishing communication channels and securing the transmission over the channel, but there is a concern typically foreign to stationary devices and networks: fraudulent usage of bandwidth. Because of the physical barrier that exists with the wired infrastructure in wired communications, stealing network bandwidth typically costs more than it is worth. This is not so in the case of wireless communications because there are no physical barriers.

There are numerous problems at the bottom layers of the OSI model that hardware manufacturers must deal with. For example, maintaining the ability to decrypt data packets in the presence of packet loss is very difficult (Aziz and Diffie, 1993). This is especially true when we also do not want the dropped packets themselves to become a tool for a security breach. Fortunately, most if not all of these problems are the concerns of hardware manufacturers and typically do not even concern those writing operating systems for mobile devices.

One of the short-range wireless networking technologies that we discussed before. Every Bluetooth device in a Piconet generates a secret key when the user enters a personal identification number (PIN). Devices authenticate each other in multiple steps as follows:

1. The claimant (the device trying to be authenticated) sends a message based on a 48-bit address to the verifier (the device challenging the authentication).

2. The verifier sends back a 128-bit random number as a challenge.

3. The claimant then creates a signed response based on using a Secure Hash Function (specifically SRES). This function is, in turn, based on the secret key of the device, the random number sent during the challenge, and the 48-bit address and sends the message back to the verifier.

4. The verifier generates its own SRES and compares the SRES received from the claimant to the one that it generated.

5. The claimant also creates its own 96-bit cipher to encrypt the messages once authenticated.

A Bluetooth deployment can operate in three different security modes. First, we can have no security, which means any Bluetooth-enabled device can join the

network without requiring authentication. Second, we can enable service-level security, which basically means turning on security at the data link layer and monitoring access to various services. In this method, we have authenticated and authorized nodes accessing services in each other and the communication between them is confidential. Finally, in the third method, link-layer security is enforced by having each node authenticate the other node that it connects to (two-way authentication) and then encrypt all of the messages back and forth based on a key that only the two nodes involved in the communication share.

The general problem with ad hoc PANs such as Bluetooth is that once the security of one node is compromised, it spreads throughout the system; it is tough to track down where the breach started. The basic assumption of Bluetooth and similar PANs is that the user is in control of the network and participates in the process of distributing the secrets that are to be shared among the nodes (Candolin, 2000). This should be somewhat obvious based on the definition of PANs, but it tends to get lost in the fact that Bluetooth is being used to form larger networks that are more like LANs. Bluetooth provided adequate security for what it is meant to do: replace wires!

The other set of short-range wireless technologies that we discussed were 802.11-based technologies. The first version of the security mechanism in 802.11 is called Wired Equivalent Protocol (WEP). 802.11 requires the maintenance of an ACL of MAC addresses. The access point is always considered to be secure and maintains this list of MAC addresses. So, the end points have to authenticate with the access point. This is done by a 40-bit shared-key RC4 (Rivest Cipher four, designed by Ron Rivest of RSA) for exchanging the data and an encryption challenge issued by the access point followed by response encrypted by the end station. There is also the temporal key integrity protocol (TKIP), which is a patch for 802.11 implementations

designed to correct vulnerabilities in the wired equivalent privacy protocol, particularly the reuse of encryption keys (Varshney, 2003).

This security scheme has been highly scrutinized because the birth of 802.11. In fact, security varies in the different flavors of 802.11. The criticisms of 802.11, particularly 802.11b, have been that eavesdropping is possible (though not easy) and that the dropped packets are not properly encrypted. Typical security threats to 802.11 come at the physical layer. Active attacks include simple transmission at the 2.4-GHz frequency range that can cause denial of service as well as a host of other problems. Passive attacks can include masquerading a malignant client as a valid participant in the network and causing problems by flooding the access point with bad transmission or using other techniques. Another type of attack may include sniffing packets in the air, modifying them, and retransmitting them either to reveal information that will open other security holes or to send bad data so that the messages are altered even though parties remain authenticated and authorized. Finally, RC4 key generation implementations in the first versions of 802.11 devices were "weak" and left a security hole by producing keys that could be discovered in a matter of hours.

When an 802.11 network is deployed, we can set it to authenticate the users that try to join with a one-stage challenge–response called Open System Authentication or with a two-stage challenge–response called Shared-Key Authentication. The Open System Authentication model is the default setting on most 802.11 equipment (routers, etc.) and basically allows anyone to join the network. This is actually one of the most significant ways in which an 802.11 network is exposed!

Many home users and even some commercial users of 802.11 equipment buy the equipment and install it without understanding the security implementations. We can also set the 802.11 equipment to operate in a Shared-

Key Authentication mode, which basically means enabling encryption of transmissions between the joining node and the network. Joining the network is only possible if the node trying to join has knowledge of the "secret key" (a password) based on which cryptographic keys are generated. Some of the common complaints about this have been the following:

1. RC4 key scheduling is weak. It is not impossible to break RC4 keys.

2. The client does not authenticate the access point. This is a big problem! This means that a malicious party masquerading as the access point has a good chance of hijacking unsuspecting nodes.

3. The user does not participate in the authentication process. Many have suggested that every session should have its own "secret key" to enable user participation.

4. Denial of Service attacks are fairly easy to stage as you can keep the access point busy by continually trying to authenticate with it.

5. There is nothing currently built into the 802.11 standard that addresses anomalies or intrusions. For instance, data can be extracted out of the air and analyzed so that the encryption keys may be discovered. It is hard to tell if a full-proof technology for intrusion and anomaly detection will exist for wireless transmissions but being aware of this vulnerability is important. A suggested solution here is to have an external system monitor the over-the-air transmissions external from the 802.11 network itself to assure that there are no malicious parties.

6. Because 802.11 deploys one-way authentication, it is vulnerable to man-in-the middle attacks.

One of the common mistakes in deploying 802.11x-based networks, or any other type of short-range wireless network in the same frequency range, is relying on the coverage range as a security mechanism. The "Pringles can" trick (where the aluminum-coated cylindrical can of Pringles potato chips is used to form an antenna) or other types of antennas can catch signals considerably beyond the maximum range specified.

Another big way that network security may become compromised in 802.11x networks is with the theft or loss of the device. Because each device that connects to a WAP-enabled 802.11 network has a key, security is compromised by loss or theft of the device. As with any other point of breach, this can cause further security breaches in the system.

Of course, one suggested solution to all of these problems has been to strengthen the application-layer security (firewalls, antivirus software, etc.). Nonetheless, 802.11 technologies have some significant security holes. Perhaps the single best solution to making 802.11 networks secure is to use all of the measures that are available to us at the same time: LEAP, TKIP, TLS/SSL, and whatever else may be offered by the vendors. Most of the time, even in wired systems, either we cannot make a system 100% full-proof secure or the cost of doing so is much higher than the cost of data loss and security breach. In the end, success derives from the value delivered to the customers. So, the best solution is to make it as difficult as possible for malicious parties to breach the security of the system. Of course, an inefficient alternative to all of this is to use a VPN that encrypts all transmissions.

This is a bit redundant and inefficient because of multiple encryptions performed by different layers of hardware and software. We discussed long-range wireless network technologies and standards such as CDMA, TDMA, GSM, and 3GPP in considerable detail previously in this text. As previously discussed, we are most interested in various cellular-based long-range

technologies. Each one of the technologies and standards that we discussed to addresses security concerns in its own way. Some standards build on other standards while adding their own security mechanisms on top of the security mechanism offered by the underlying technologies. Of particular interest in the real world are two categories of cellular technologies: those tied to the CDMA family of technologies and those tied to the GSM family of technologies (remember that most GSM deployments are currently on TDMA-based networks).

One of the first and most prevalent security problem with 1G and 2G networks has been the fraudulent usage of the network, also known as "bandwidth theft." These thefts include simple techniques such as stealing identification codes of vulnerable devices and masquerading as those devices and more sophisticated attacks such as redirection attacks, in which a device is redirected to contact a false base station, thereby giving it the authentications signals it needs to contact a valid base station, authenticate, and authorize. Subsequently, the false base station uses the authenticated session for fraudulent network usage or may cause other security breaches such as stealing and modifying the communications emanating from or going to the wireless node. To prevent fraudulent use of wireless service, the GSM network authenticates the identity of a user through a challenge–response mechanism, in which the user proves its identity by providing a response to a time-variant challenge raised by the network (Zhang, 2002).

Unlike stationary computing systems where a device is bound to be connected to the same network for an elongated period of time, mobile devices come in and out of networks frequently. This has been part of the rise in ad hoc networking technologies implemented at different layers of the OSI model that we looked at previously. MANETs (mobile ad hoc networks) introduce an

entirely new set of concerns in security. Let us see how some of the previously discussed concerns change when dealing with MANETs:

1. Because most MANETs allow more information exchange with an unknown network (e.g., unknown IP address) and because the communicating nodes of a network are constantly changing, *masquerading* is a larger threat in MANETs than other comparable environments. Extra caution must be taken in authentication of a participating node before any data are exchanged.

2. *Eavesdropping* is always a threat and even more so in wireless environments, but this probably increases even more with ad hoc networks because every participating node is always revealing just a little more information than it would if it were not a participant in an ad hoc network.

3. Depending on the type of MANET, *DOS attacks* may become very easy. The most rudimentary attack may be simply disguising as a different node every single time and going through the discovery process. Though without the proper credentials no access is granted, the system may be flooded with network traffic and one or more nodes may be stressed while participating in the discovery process an endless number of times.

4. The possibility of attacks from previously authenticated nodes is substantially increased in MANETs. Although prior, during, and shortly after authentication, a node may not be acting malicious or become infected with a malicious program such as a worm or virus, it may become malicious or become infected after it has been authenticated because it may be participating in other MANETs.

Ad hoc and peer-to-peer systems have an entire slew of security concerns of their own that lie outside the scope of this text. Keep in mind that all of these problems are typically at the application layer unless the peering or ad hoc connectivity features are being provided by the wireless networking

infrastructure. If so, you need to make sure that you understand the features of the underlying infrastructure and the settings that will allow your particular application to operate securely within the boundary of the threat-level assessment and return on security investment.

One of the biggest hurdles in making location information available to mobile applications has been security and privacy concerns. In a world where users find more and more information collected about them every day, it is very important to make sure that they retain the right to block the mobile application from discovering their location. Furthermore, we must ensure that whatever location information is exchanged remains completely safe within the system. From this perspective, the availability of location information opens a whole new can of worms when it comes to security. At the same time, location information can be used to strengthen whatever security mechanism is in place.

Specifically, we can use location information to keep malicious parties from spoofing and making DOS attacks. If a given device is introducing itself with one identity and the location information about the device indicates some conflict with this identity, we know that there is a significant possibility that this device poses a security threat to our system. Also, if one node tries to initiate DOS attacks on a MANET or similar type of a network where the discovery mechanism of the network can be used to attack it, location information may be used to "block out" nodes that may be exhibiting such behavior because identifying them any other way may not be possible (for if they are masquerading, then we cannot use their identity to block them out).

We strongly recommend that all information regarding location information, whether relative or absolute, be encrypted when transmitting at the application layer. In addition, it is also recommended that all networked mobile applications use SSL or a similar technology for transmitting the location information at the presentation and session layers. Although this may seem

overkill, we simply recommend it to keep the data secure if the system is compromised either at the application layer or at the presentation and session layers.

We know by now that most of our concerns lie within the application layer where we have the most amount of control So, how is security different here for location-based services and applications that use those services? Leonhardt and Magee [Leonhardt and Magee, 1997] recognize two different methods of solving the problem within distributed applications: label-based and matrix-based protection.

The label-based model, also referred to as the Mandatory Access Control, specifies a read and write level for a resource that a user is trying to access. The matrix-based access control list basically extends the traditional one-dimensional ACL security model where each security right is mapped to a specific location, be it relative or absolute, depending on the implementation of the application. Matrix-based access control offers a flexibility and expressiveness far superior to label-based access control [Leonhardt and Magee, 1997].

The various security problems with mobile agents stems from the fact that they are autonomous; this autonomy makes it difficult if not impossible to identify mobile agents with complete certainty (an authentication problem) and keep them from doing something they should not be doing (an authorization problem). The security threats of mobile agents can be classified into four broad categories: agent to agent, agent to platform, platform to agent [Jansen et al., 1999].

At the same time, because of their autonomy, mobile agents can come in handy in building security software such as an antivirus and intrusion detection

software. Jansen and colleagues, for example, show how to build an intrusion detection system.

In fact, in the mobile environment where there is a proliferation of devices, mobile agents can present an ideal way of detecting intrusion as they provide a scalable solution to a distributed security problem (both because they can run on multiple platforms where the mobile agent host is available and because the workload is distributed among many nodes at run time). In fact, Jensen and colleagues specifically recognized several technologies of interest to us as areas where either we need to consider the security aspects of mobile agents or mobile agents may be a suitable solution for our system:

(1) Wearable computing, in which the mobile computing device may be worn.
(2) Pico-cellular wireless systems, in which the network is made of many very small cells.
(3) Ad hoc networks, in which autonomous systems are used for networks in an ad hoc manner (as previously discussed).
(4) Other mobile devices in general where mobile agents may make sense.

Also, active networks are another novel approach to network architecture. In such networks the switches of the network perform customized computations on the messages flowing through them; active networks make use of intelligent packets that are no longer just data bits but contain mobile code that allows for the active participation in routing, fault-tolerance, and QOS decisions [Jansen et al. 1999].

Hence, mobile agents can be a suitable solution for addressing the various security concerns within active networks.

3.4 Location Information, Security and Privacy

Internationalization and localization standards for mobile applications are typically much more stringent than their stationary counterparts. Namely, because the users of mobile applications are moving, they are passing through different locales. An application whose business rules may change (e.g., the tax rate may be different between adjacent locale boundaries such as states or provinces) can benefit greatly from knowing where the user is or where the user is heading—information that may be provided by a location-based service. Likewise, because the spoken language, the written language, units of measurement, business rules, and a variety of other things may change when the mobile user crosses the boundaries of one country into another, location information can be of great help in determining the location.

The key here is that internationalization and localization can benefit from location-based information in that determining the location of the user and the subsequently reliant application rules can be automated. However, location-based systems do not really rely on internationalization and localization techniques; the location model as well as the implementation of the location-based system typically does not depend on the location being measured. The association between location and locale information may need to be personalized. For example, a bilingual user who speaks both English and German may travel from the United States to Germany with a GSM phone and want to see the appropriate interface depending on the location. In contrast, an American user who does not speak German may go to Germany and may want to continue viewing the system in English though there may be some incorrect data (e.g., prices may have a dollar sign instead of a Euro sign).

Therefore, location information facilitates easier, better, and more automated localization and internationalization of other aspects of the mobile application such as the user interface. Also, based on the network infrastructure, the

implementation of the location-based systems may vary. Nonetheless, inherently, neither the implementation of location-based system, nor the interface to it, requires localization and internationalization.

At the time of authoring this text, location-based services are one of the fastest growing areas of computing. The most recent efforts focus on making location information pervasive throughout all computing systems while allowing mobile computing devices and systems to take advantage of this information. A prime example is the SRI International's Digital Earth project. The goal of the Digital Earth project was to develop infrastructure for an open, distributed, multiresolution, three-dimensional representation of the earth, into which massive quantities of georeferenced information can be embedded (Brecht et al., 2002).

Digital Earth has been designed to be able to provide geographic representations of locations in three modes: *text based, map based*, and *TerraVision based.* The textbased format is for the simplest of devices, within which the resource-starved low-end mobile devices such as some mobile phones may fall. The map-based representation allows viewing of locations with PCs, high-end mobile devices capable of rendering graphics well, and other devices with sophisticated twodimensional GUIs. TerraVision is a distributed, interactive terrain visualization system developed by SRI International [Brecht et al.]. It allows three-dimensional viewing of geographical locations, using GeoVRML, a set of extensions for VRML.

It is important to note that this project takes into account human-made structures as well as the natural terrain of earth. Consequently, buildings and other types of human-made structures are modeled and updated periodically. Along with this, the Digital Earth project leverages DNS (Domain Name Server) and HTTP, two of the base technologies for the Internet, to build a fault-tolerant, scalable, and secure system for providing location information.

SRI has recently proposed a new DNS extension of .geo, to ICANN, to be used for retrieval of location information using the Digital Earth project.

There is one more crucial thing about the Digital Earth project. The GeoWeb is fully described in RDF. This is a very significant achievement because it allows the integration of geographical information with other semantic information available on the Web. This in turn gets us one step closer to producing a better environmental context for applications by integrating the semantic meaning of the geographical information with semantic meanings of all of the other information and behavior involved in the application.

There are also remaining challenges. As Jensen recognizes (Jensen, 2002), current GIS systems and supporting applications do not possess the necessary robustness and scalability criteria to hold detailed data about movement of mobile things, whether these are users or things that make up the topography of the geography, as rapidly as needed. We still need further standardization in the modeling of spatial information, and we need to create standards accepted by governmental, academic, and commercial entities that implement solutions for accessing location and mobility information.

Location sensitivity is one of the dimensions of mobility that offers the most promise for new ideas in automating tasks to make mobile computing valuable. Usage of location information is certain to become more prevalent in mobile computing.

Security is always one of the biggest concerns when designing any application, but particularly distributed applications. Distributed applications operate over networks, involve multiple users, and have many other properties that make them more vulnerable to security breaches. Though there are stand-alone mobile applications, as we have discussed earlier in this text, most mobile applications, at their core, are distributed applications. Unfortunately, to date,

there remain many unsolved problems with security concerns of mobile applications.

Our goal in this project will be to first introduce a taxonomy of mobile application security problems, look at a few general approaches in solving these problems, and finally review those problems that remain unsolved. Security also tends to be a system-wide problem, not just an application problem, whether dealing with mobile or stationary applications. So, we are not out to show you sample code, standards, or specific techniques; such discussions are completely beyond the scope of our discussion. Our main purpose is to take a step back and look at the big picture of mobile application design and see where security concerns may be. Security is intimately bound to the design of the platform for which the mobile application is being built. Dealing with such concerns can be trivial as all we need to do is to understand the security infrastructure of the mobile platform and implement the appropriate APIs in our applications (e.g., what WAP may let us do with WTLS or how we can author secure applications on the Palm platform).

However, most mobile application or mobile platforms do not exist in isolation. Therefore, the bigger picture is that most mobile applications are really distributed applications being used by mobile users on mobile devices. And this is where security gets tricky. In this chapter, we will concentrate on the big picture that lets us see as many holes as possible. The solutions may be dependent on the mobile infrastructure on which you must deploy your application or the mobile application itself.

Just like all of the other topics in this text, our interest is not in reviewing solutions and problems specific to some specific technology or review APIs. We consistently believe, in approaching the various problems associated with mobile application development, that focusing on specific implementations is not the right approach because mobile technologies are changing far too

quickly. So, let us start with a general taxonomy of the type of security problems that we may approach during the life cycle of our application.

3.5 Distinguishing between Privacy and Security

The mobile application developer faces two distinct challenges: delivering a secure application and delivering a private application. Privacy means very different things in different countries and is subject to country-specific sets of laws. For our purposes, we define security within the confines of laws and regulations in United States, Australia, and Europe, which tend to be somewhat similar compared to the rest of the world. Candolin defines privacy within the realm of wirelessly connected systems to be of four components [Candolin, 2002]:

1. *Data Privacy*: The contents of the transaction should be protected from disclosure to an unauthorized party.

2. *Source and Destination Privacy*: The parties involved in the transaction should not be revealed to an unauthorized party.

3. *Location Privacy*: The location of the parties making the transaction, whether physical (geographical) or logical (with respect to the network), should not be disclosed to an unauthorized party.

4. *Time Privacy*: The exact time when a transaction occurs should not be disclosed to an unauthorized party.

The definition by Candolin is meant to address wirelessly connected mobile applications, but it is indeed sufficient for all mobile applications as the wirelessly connected mobile applications face a superset of challenges of wired mobile applications to provide proper security and privacy.

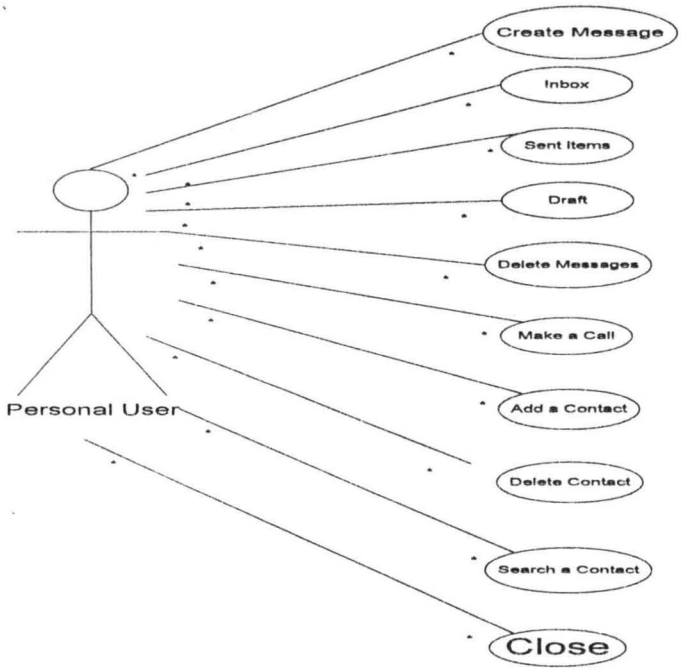
Obviously, the first step in providing privacy is achieving security. So, in a way, although security and privacy are two completely different things, security is a precondition to privacy. As we mentioned previously, security is largely a design-time problem. So, in the manner that we have selected in this text, let us try to use UML to document this design-time concern and its effect on the implementation of the system.

We have already seen that UML component diagrams allow us to show high-level system diagrams, interaction diagrams provide a way for showing how various systems may interact securely, and class diagrams are ideal for the internal implementation of our application. We can still apply all of these diagrams to show the high-level design and behavior of secure applications. SecureUML, as proposed by Lodderstedt and colleagues, is a methodology, with UML extensions, to integrate information relevant to access control into application models defined with UML [Lodderstedt et al., 2002].

The focus of Secure UML is to define a metamodel that can be used to define a framework within UML with which to model various security processes such as authentication and authorization. As conceived by its authors, SecureUML is mainly used for stationary serverside applications. Nonetheless, there is nothing in the metamodel that would keep us from applying it to mobile applications. Along with the other extensions that we have introduced, particularly those that address location mobility, SecureUML allows us to create diagrams that represent the three basic principles of security: determining *who* has what *roles* and what roles and users in those roles should be able to access *which* resources.

3.6 UML Illustration for Program Access

Using USE CASE UML Diagram to Show the Personal Level of Access of a Mobile Application User .



CHAPTER FOUR

4.0 IMPLEMENTATION AND DOCUMENTATION

4.1 Program Design Methodology

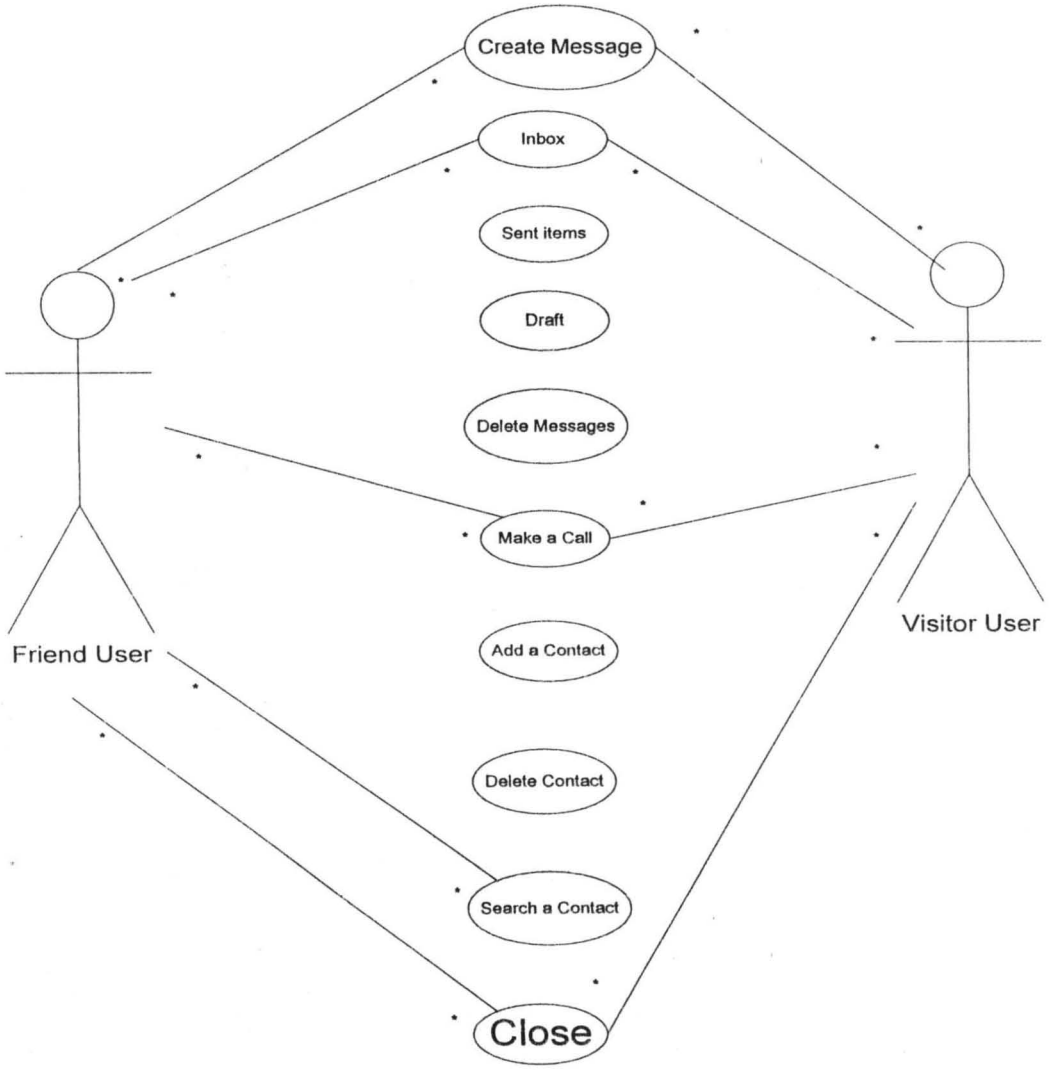
This section comprises the program design methodology adopted in developing the program for the mobile application security for mobile devices. A program by definition is an algorithm specially expressed in a particular high level language capable of execution by a computer system. However, a well defined approach used in setting the program sets of instruction to execute a particular procedural task is referred to as program methodology. Therefore, a program design methodology is the approach defined for designing program sets of instruction for execution of procedural task. There are two main types of program design approaches for any given programming languages. The aim of this program is to computerize some of the manually computed models considered in the previous chapter, the spread of cholera the objective of this automated system is as follow:

1. Speed: instead of having to solve the problem manually, the written program gives instruction to the computer which executes for as many intervals as it's instructed to do. It is convenient and saves time
2. Accuracy: mistakes can easily be made when manual process is used. With the computer more accurate results are gotten if the right instruction is supplied.

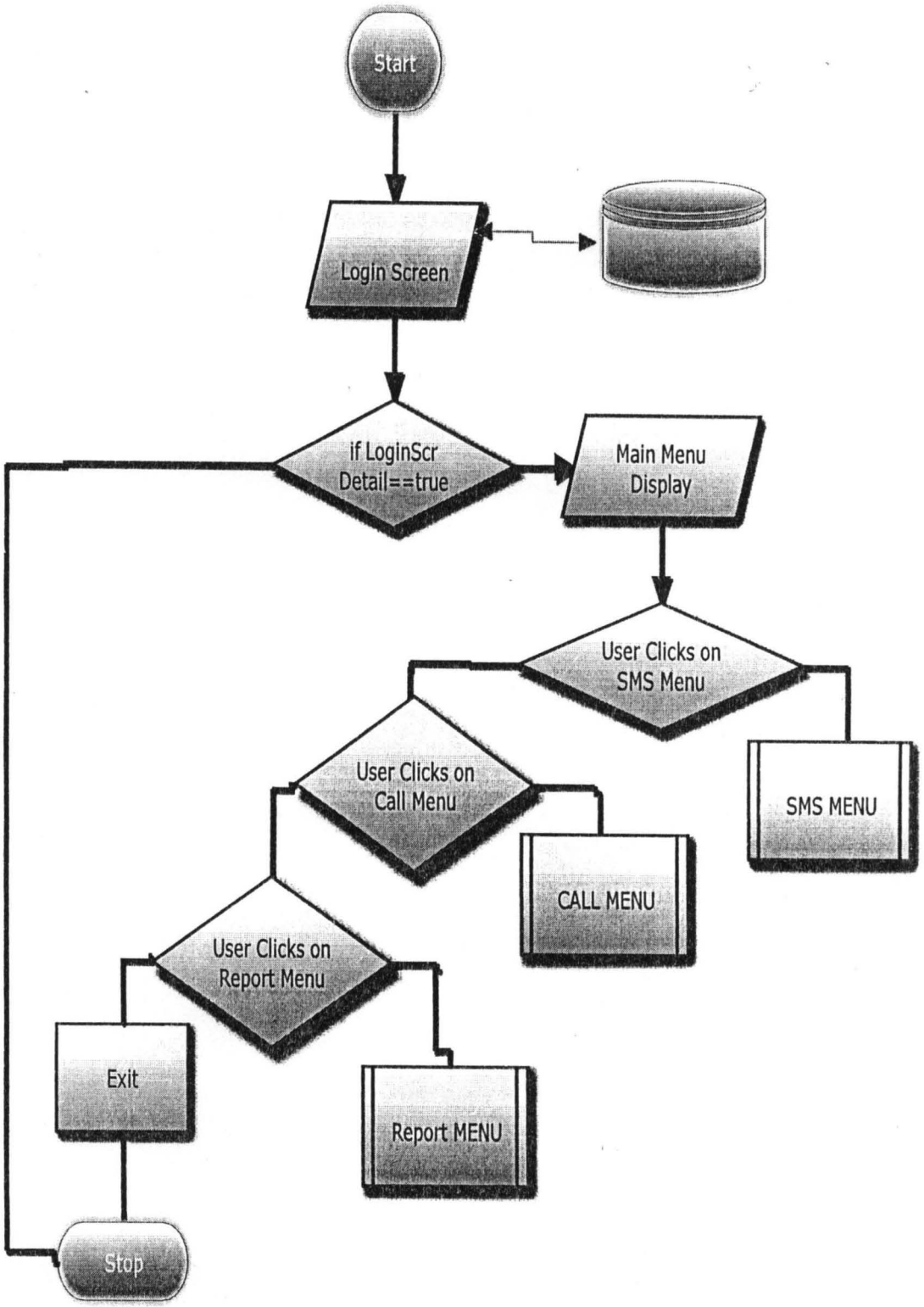
Programming language are sets of instructions in a specific syntax that can be interpreted and executed by a computer to accomplish a procedural task here are some basic characteristic of a programming language

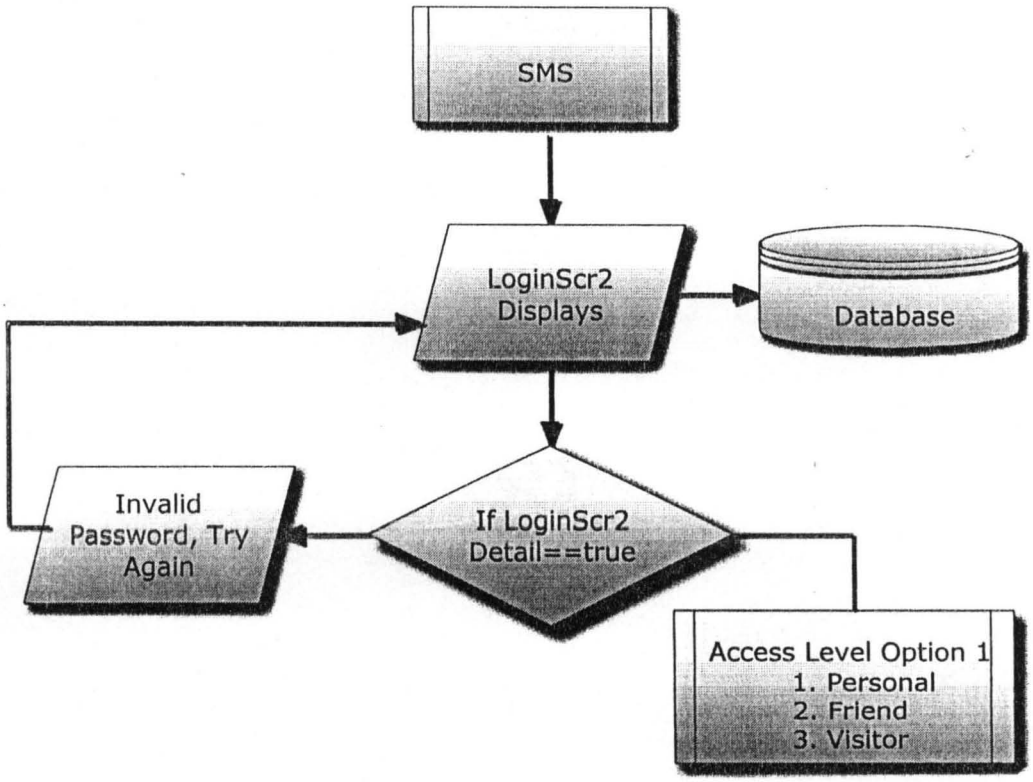
1. Finiteness: the number of stages of an instruction must be finite.
2. Precision: programming steps must be void of assumption

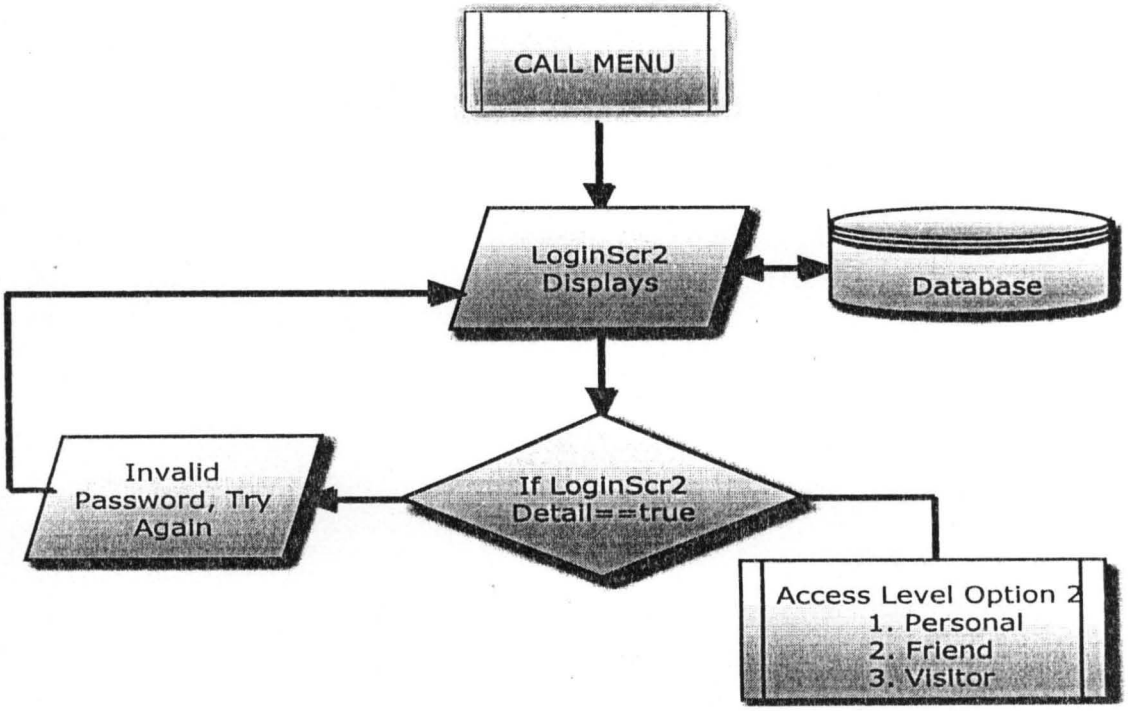
Using USE CASE UML Diagram to Depict the Personal and Visitor Level of Access of a Mobile Application User .

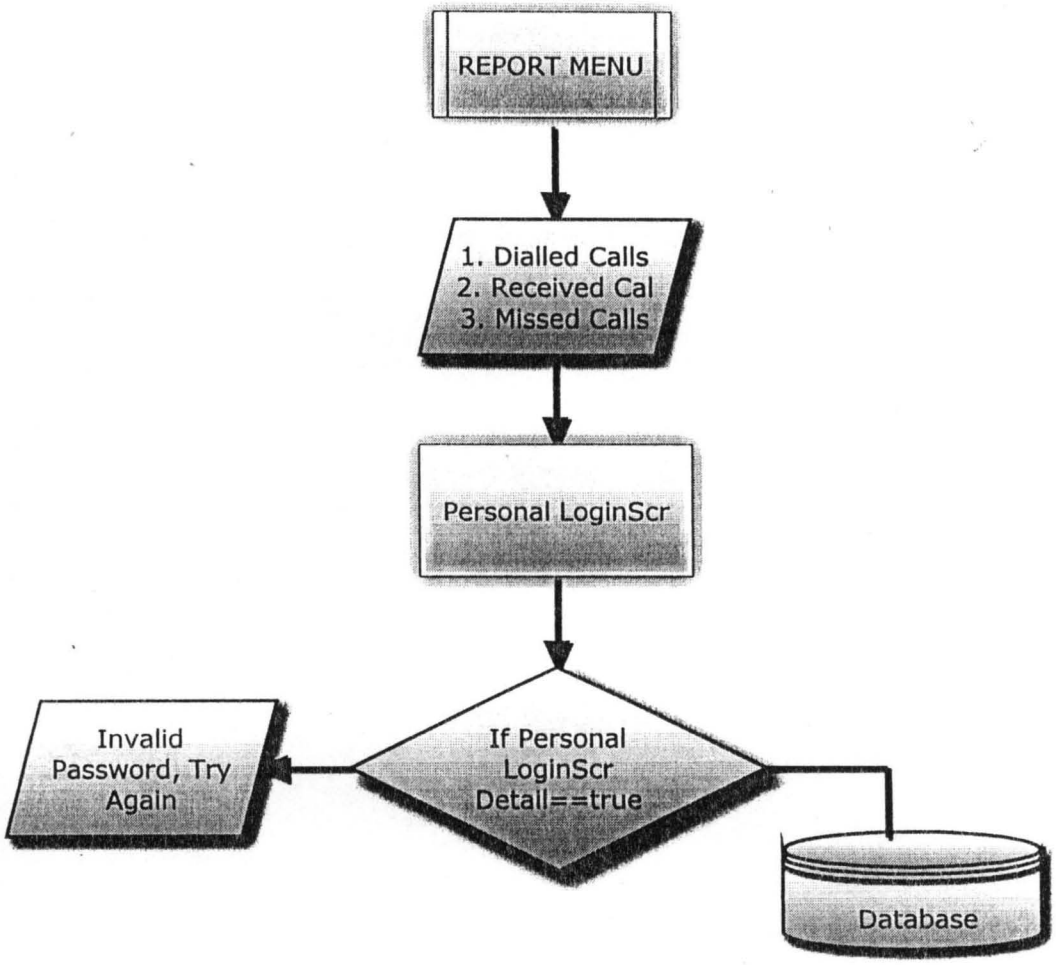


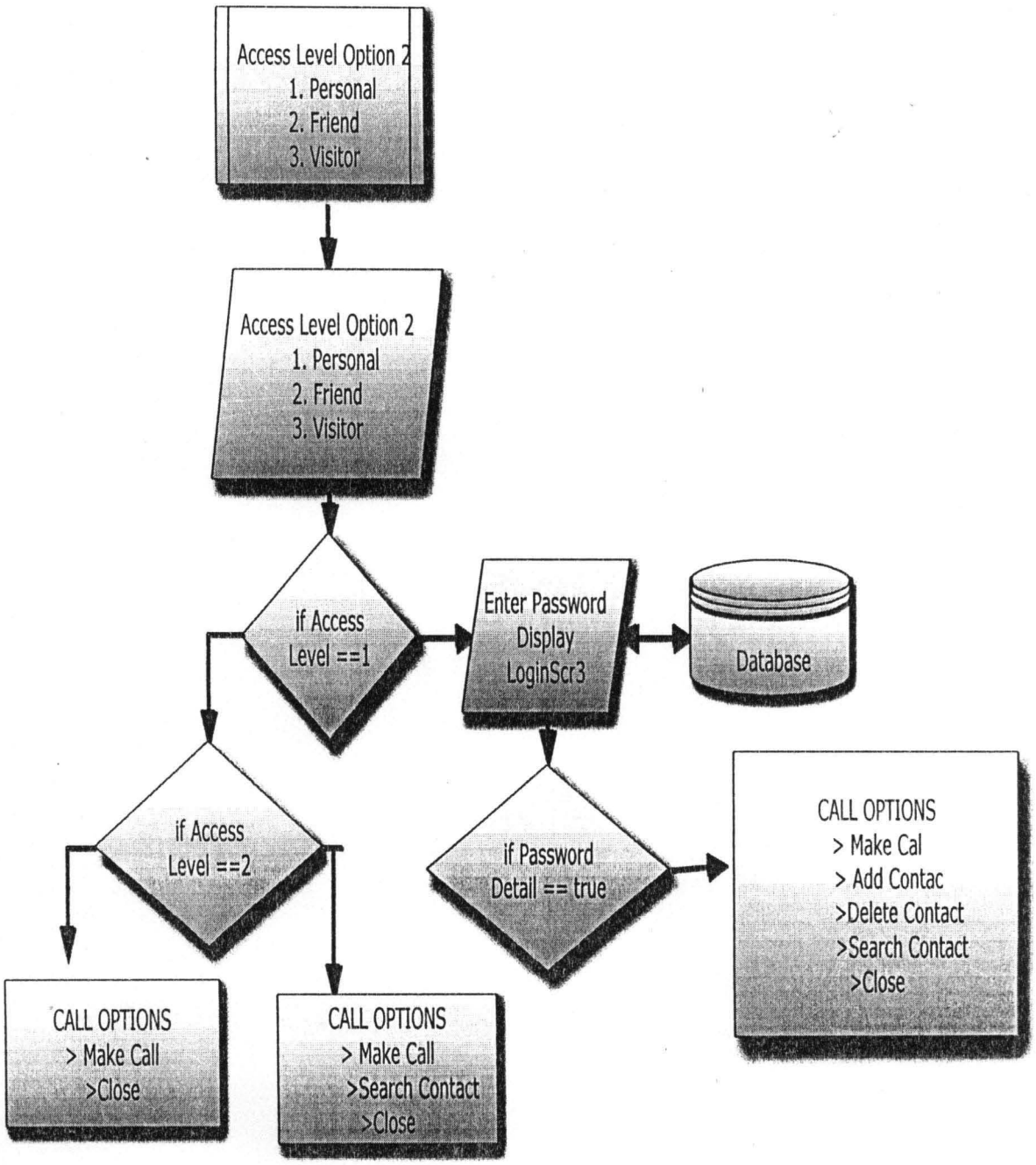
FLOWCHARTS OF THE MOBILE APPLICATION

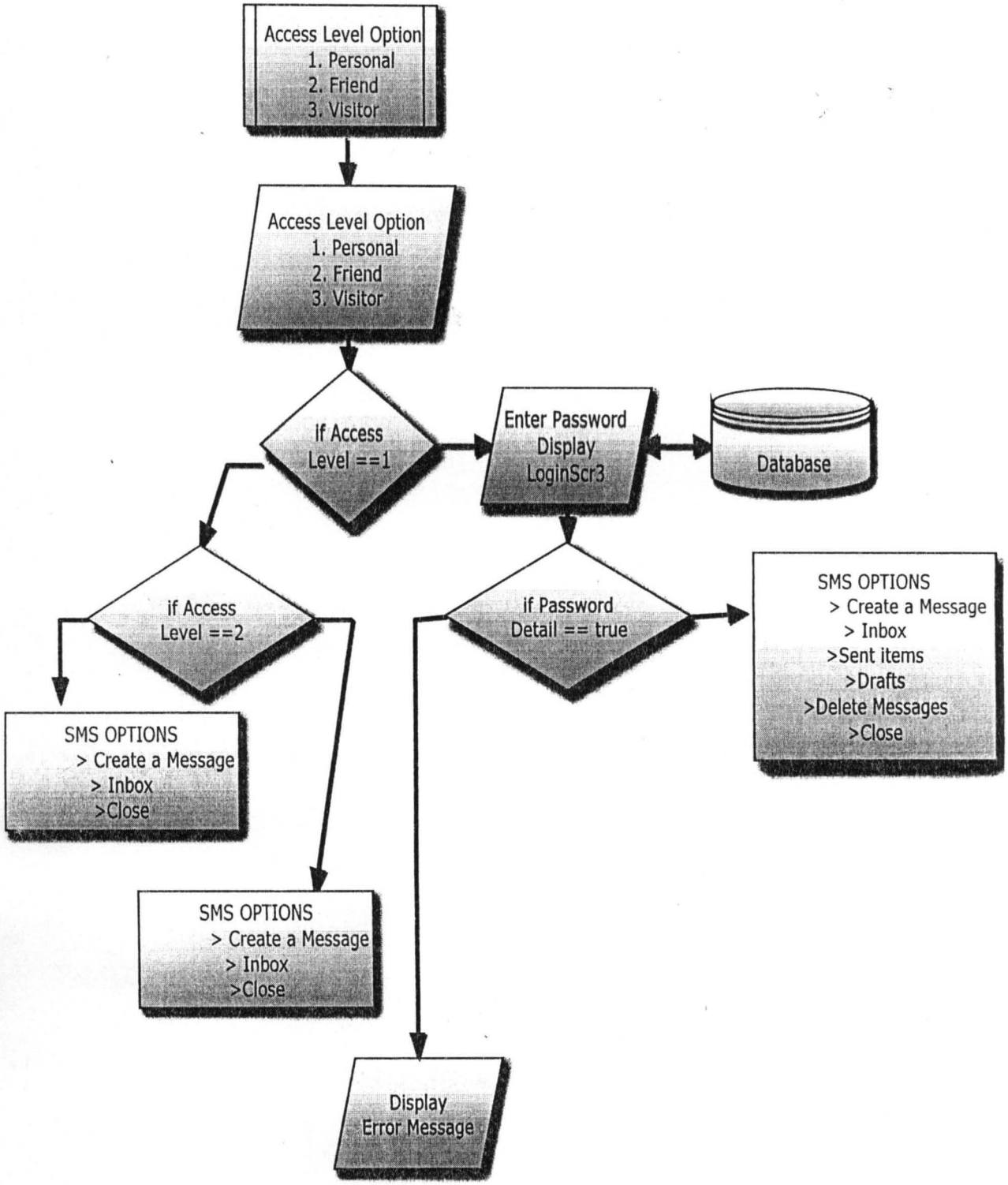



SMS
LoginScr2
Displays
Database
Invalid
Password, Try
Again
If LoginScr2
Detail==true
Access Level Option 1
1. Personal
2. Friend
3. Visitor


CALL MENU
LoginScr2
Displays
Database
Invalid
Password, Try
Again
If LoginScr2
Detail==true
Access Level Option 2
1. Personal
2. Friend
3. Visitor


REPORT MENU
1. Dialled Calls
2. Received Cal
3. Missed Calls
Personal LoginScr
Invalid Password, Try Again
If Personal LoginScr Detail==true
Database


Access Level Option 2
1. Personal
2. Friend
3. Visitor
Access Level Option 2
1. Personal
2. Friend
3. Visitor
if Access
Level ==1
Enter Password
Display
LoginScr3
Database
if Access
Level ==2
if Password
Detail == true
CALL OPTIONS
> Make Cal
> Add Contac
>Delete Contact
>Search Contact
>Close
CALL OPTIONS
> Make Call
>Close
CALL OPTIONS
> Make Call
>Search Contact
>Close


Access Level Option
1. Personal
2. Friend
3. Visitor
Access Level Option
1. Personal
2. Friend
3. Visitor
If Access
Level ==1
Enter Password
Display
LoginScr3
Database
if Access
Level ==2
if Password
Detail == true
SMS OPTIONS
> Create a Message
> Inbox
>Sent items
>Drafts
>Delete Messages
>Close
SMS OPTIONS
> Create a Message
> Inbox
>Close
SMS OPTIONS
> Create a Message
> Inbox
>Close
Display
Error Message

3. Termination: there must be termination point for an instruction that has a repeated execution.
4. Output: it must be able to generate result after
5. Flexib ility: it must be able to accommodate changes when required

(Baragiy, 1998).

[#'

Visual Basic (VB) which facilitates a highly structured program design has been used in this piece of work; the following are the significant feature of Visual Basic, namely:

a. Visual Basic compiler do not produce native code for a particular platform but rather byte code instruction for the Visual Basic virtual machine making, Visual Basic code work on a particular platform is then simply a matter of writing a byte code interpreter to simulate a Visual Basic what this means is that the same compiled byte code will this means is that the same compiled byte code will run unmodified on any platform that support Visual Basic. Hence, it is CROSS PLATFORM.
b. Visual Basic is an object oriented program, that is, Visual Basic deals with object and classes
c. Built on C^{++}: One of the major factors that affects the rapid adoption of Java is the fact that is was built to follow the syntax of C^{++}.

4.2 Programming Development Guidelines

A Visual Basic (VB) program is an application written specifically for Windows by using the programming language. To write a Visual Basic programming language, we follow these steps:

A. *Planning the program:* This determines exactly what the program is to accomplish.

B. *The designing the user interface Source*: The complete set of screens and images used in a program is called the program's user's interface. Through a good user interface, the user is able to make optimum use of the of the program

C. *Building the program*: Building a windows based application by using these three programming concepts

   i. Create the user – interface by using Visual Basic Controls
   ii. Set the character or properties of element in the user interface as needed.
   iii. Write the program code for one or more user interface element as needed

To build the interface element, we need to click an interface control in the Visual Basic toolbox and then draw the user interface elements by clicking and dragging, clicking to position one corner of the element and then dragging to create a rectangle that is the size needed.

After creating the element in a textbox, for example, retiming by setting properties to fit the element is the next procedure. For a textbox, properties to make text in the textbox can be set to bold, italic underline and so on. To complete the program by typing the code for one or more user interface. Elements in a special window called the code window. Writing program codes gives more control over how program using the design control.

The next is to test carefully program, compile it into an executable program and distribute it, testing a program involves clicking it against a variety of real life operating conditions to determine whether it is working correctly a problem that stops the programs from running or from producing the expected results is called a software defect or bug. This process can be repeated beginning with

planning and an analysis of any Nero goods. The steps complete the software development life cycle which is illustrated in the figure 4.1 below:

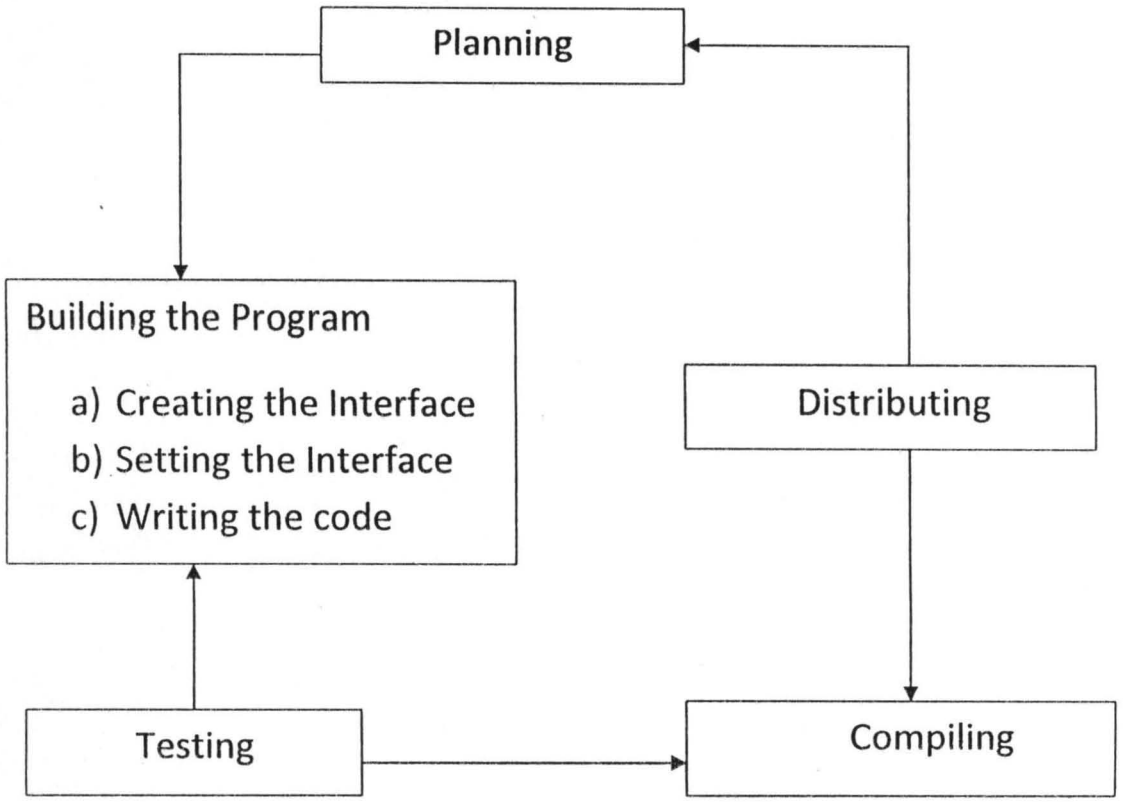


Figure 4.1: A Complete Programming Process.

4.3 Software Requirement

The computer System to be used must be amongst other relevant application software have good working system operational software. These systems software include Microsoft Disk Operating System, Windows Operating System and Visual Basic Compiler Software packages fully installed, each file must be compiled as Visual Basic type before execution take place. Although Visual Basic is fully a graphically developing environment and programming

language area for cry from the early Basic interpreters, the elegance and simplicity of the original Basis has to a great extent remained in the language.

4.4 Hardware Requirement

The hardware requirement for the properties system includes the following:

a) At least a Pentium IV system
b) At least a 20 GB – Hard Disk Drive
c) CD – Rom Drive
d) A Flash Drive or Compact Disk
e) A minimum of 14" Monitor (VDU)
f) A Standard or enhanced keyboard
g) A mouse with pad
h) A printer

4.5 Choice of Programming language used

Visual Basic's power and case of use are the primary reasons why it was chosen as the programming language for the project. It is a programming language constructed primarily with the aim of humanizing applications of data structure and access strategy. It has a lot of merit, which includes:

i. Large number/amount of data can be stored
ii. Data Integrity
iii. Easy to write and understand
iv. Debugging is very easy
v. Fast Retrieval of information

CHAPTER FIVE

5.0 CONCLUSION AND RECOMMENDATIONS

5.1 Discussion of Result

In fact, SMS is the delivery of alphanumeric messages to mobile phones over wireless networks. SMS is not inherently a wireless communication technology. It is a value-added service designed to run on long-range wireless networks. SMS messages can be sent from a mobile device or from an SMSC (Short Messaging Service Centre), routed by an SMSC and arrive at a suitable destination as an SMS message, an email or some other forms of electronic messages .

Two things make SMS fundamentally different from the other data access technologies. It can be delivered whether or not there is an ongoing voice call and it is an asynchronous messaging system that allows for flexibility in the temporal behaviour of the network and related delivery attributes.

SMS was first developed in Europe in the early 1990s and became prevalent in the mid 1990s. In the United States, it was not until the very late 1990s that SMS was available and usage grew slowly but steadily from thereon. To date, SMS is by far the most successful data application used on wireless networks

SMS does not require usage of one type of wireless network over the other; it can be implemented over whatever network is available. However, to date, it is primarily implemented on TDMA and CDMA networks and will be supported by the 3G variant of those networks or more modern networks.

5.2 Constraints / Limitations

The limitations encountered in this project include:

- **Finance:** This slowed down the speed of completion of the project

- **Time:** I had to share my time between classroom lectures and researching for the project
- **Limited Internet Access:** There are no browsing facilities in the school for student research purpose.

5.3 Summary and Conclusion

The basic architecture of a telecommunication infrastructure that can deliver SMS messages is provided in this project. One important thing to note is that SMSCs all implement signaling system connectivity. This is crucial for delivery of messages among disparate networks and is a big part of what the implementation of an SMSC gateway includes.

Most carriers, for security reasons, do not offer third-party connectivity to their SMSC or any connected part of the infrastructure of an SMS system. Unfortunately, this means that the only way for a programmer to write an SMS application is to interface with the carrier's SMTP servers that are then connected to the SMSC.

In other words, creating an SMS message, as far as we are concerned, is the same as creating an email. Whereas the length of the message is supposed to be 160 characters, depending on the network but with a maximum of anywhere from 100 to 280 characters for an SMS message.

Of course, we only have to do this if the device we are using is not SMS enabled. With SMS-enabled devices (typically mobile phones and PDAs), you simply compose your short message and address it to the phone number of the recipient and off it goes. There is one other way to send SMS messages:

You can use a mobile phone, or another SMS-enabled device, as a proxy into the carrier's network. Here are the steps to do this:

i. Connect your mobile phone, PDA or whatever SMS enabled device you have to your PC with an RS232 cable, USB Cable, or whatever connection provided
ii. Install that phone as a MODEM to the PC.
iii. You can now send an SMS message using the SMS-enabled device as a MODEM by sending an "AT" Command.

SMS will eventually be replaced with the more advance EMS and MMS. One interesting thing about SMS is that, because of its pervasiveness, it is occasionally used as a text based application –layer transport protocol. In other words, we can build a mobile application that resides on the device, in one of the environments we have looked at such as J2ME and Windows CE, and use SMS to send and receive messages from some other mode on the network. The SMS messages could, for example, hold SOAP envelopes. (Waldo, 2001)

5.4 RECOMMENDATIONS

We have certainly not covered all of the various areas of wireless networking that relate to mobile application development. Our attempt has been to giving an introduction to some of the most pervasive technologies to give us a good understanding of the limitations and capabilities of the infrastructure that our mobile applications will be using for communication. The core service of the more advanced data networks such as various 802.11 networks and 3GPP-based networks may evolve along a number of different paths.

We recommend that the governing rules for determining this evolution can be largely driven to an equilibrium point where the wireless carriers can make money and expand their market, the device manufacturers can be continue to introduce more and more advanced devices, the third – party application developers will find a way to introduce applications quickly to a mobile software market place suffocated by the carriers, and the consumers will

continue discovering new value in mobile applications that they can use through their wireless connection to the network. Wireless networks will be changing fast. The key for the mobile application developer is to keep up with these changes and to design applications that resist becoming obsolete by these changes.

REFERENCES

Afuan, O.M (2002) Mobile Devices: Migration for Improved Application Performance . Carleton University, Ottawa, Canada

Agre, J.A, Akinyemi L.J, Masuka M.B and Thakkar, F.H (2001) A Layer Architecture for Location-based Services in Wireless Ad Hoc Network. IEEE Press.

Alatalo, M.U. and Peraaho, B.T (2001) Designing Mobile Aware Adaptive Hypermedia.

Baragiy, G.H and Reed, Y.U. (1998) - " Why is it so Hard to Define Software Architecture?"

Bauer V. S (2001) "UML Class Diagrams: Revisited in the Context of Agent-Based Systems"

BBC (2009) Computing device Security. United Kingdom

Claessens et al (2003). "Pioneering Advanced Mobile Privacy and Security."J. Claessens.

Dewan (2002). Replication for Mobile Computing. P. Dewan, University of North Carolina

Dimitri A.G. (2002). Mobile Platforms for Mobile Agents V. Dimitri, University of Brussell at Vrije, 2002.

Foley and Van Dam (1983) Fundamentals of interactive computer Graphics. J.D foley and A. Van Dam, Addison – Wesley, 1983

Fowler and Scott (1999) UML Distilled, 2nd ed. M. Fowler and K Scott, Addison Wesley, 1999.

Graham (2002) "Using UML to Drive Java can Alleviate Chaos". B Grahamin EE Times, Rose Real-Time Technical Marketing, Rational Software Ltd.

Kanata, Ontario, April 1, 2002, available at http://www.33times.com/story/OEG2002032950025.

Hager (2002). Mobile Adhoc Network Security. C.T. Hager, Virginia Polytechnic Institute, 2002.

Hansmann (2002). SynCML: Synchronizing Your Mobile Data. U Hansmann, Prentice - Hall, 2002.

Mandel, Koch and Maier (1999) ''Extending UML to model Hypermedia and Distributed System'' L. Mandel, N.Koch and C. Maier, in Bayerische forschungsstifung, February 1999, available at http://projekte.fast.de/projekte/forsoft/intoohdm/index.html.

Scott (2001) Service Discovery Protocol (SDP). M. Scott, 2001, Available at http://www.dsc.ed.ac.uk/home/slipc/protocol/sdp.html.

German Research Center for Artificial Intelligence 1998.

Waldo (2001) ''Mobile Code Distributed Computing and agents." J Waldo in IEEE Intelligent System, March/April 2001.

WAP (2000) Wireless Application Protocol WAP 2.0 Technical white paper, WAP forum, available at http://www.wapforum.org/what/WAPwhitepaper1.pdf.

Yim et al. (2001) Architecture-Centric Object -- Oriented Design Method for Multi-Agent System. H. Yim, K Cho, J. kim and S. Park, KAIST(Korea Advanced Institute of Science and Technology)

Y.Lim and S. Li, Microsoft Research Asia,(2003) Scalable Portrait Video for Mobile Video Conferencing. J. Keman Yu, T.He

Zhang, N.I (2001) Mobile computing and wireless Networks. Y Zhang, 2001, available at http://www.cs.utexas.edu/users/ygz/395T-OIS.

Instructions on the Use of the Program

Download the attached files ***Mobile Security.exe*** and ***Mobile.mdb***. You can create a folder and name the folder "Mobile Security" put the two files into the folder don't separate them.

Right click on ***Mobile Security.exe*** and click on ***Pin to start menu***. You can now be launching it from there.

But make sure you download the ***Mobile Security.exe*** and ***Mobile.mdb*** in the same folder

Your login password is **MOSHOOD** or **moshood**

After login, it will display the welcome page where you can then make call or send SMS

When you click on **SMS** or **Call** it will display where you login, the password there is **AJADI** or **ajadi**

After you login, you can now select a category i.e. **Personal user, Friend** and **Visitor** and their password is **ARE123** or **are123** for personal user, **FRD123** or **frd123** for friend and **VST123** or **vst123** for visitor for **both SMS** and **Call**.

Appendix II

Program Codes

```
Private Sub Form_Load()
Label3.Caption = Date
End Sub

Private Sub mnuCall_Click()
frmCallS.Show
End Sub

Private Sub mnuDail_Click()
drpDial.Show
End Sub

Private Sub mnuExit_Click()
End
End Sub

Private Sub mnuSMS_Click()
frmCode.Show
End Sub

Private Sub Timer1_Timer()
Dim ab As Long
ab = QBColor(Rnd * 15)
Label1.ForeColor = ab
End Sub

Private Sub Timer2_Timer()
```

```
a = a + 40
Label7.Left = Label7.Left + a
If Label7.Left = 9000 Then
Label7.Left = Label7.Left + a
End If
End Sub

Private Sub Timer3_Timer()
Label2.Caption = Time
End Sub

Private Sub Timer4_Timer()
a = a + 40
Label7.Left = Label7.Left - a
If Label7.Left = 9000 Then
Label7.Left = Label7.Left - a
End If
End Sub

Private Sub Timer5_Timer()
Timer4.Enabled = True
Timer2.Enabled = False
End Sub

Private Sub Timer6_Timer()
Timer4.Enabled = False
Timer2.Enabled = True
End Sub

Private Sub cmd0_Click()
```

```
txtOutput.Text = txtOutput.Text & "6"

End Sub

Private Sub cmd7_Click()
txtOutput.Text = txtOutput.Text & "7"

End Sub

Private Sub cmd8_Click()
txtOutput.Text = txtOutput.Text & "8"

End Sub

Private Sub cmd9_Click()
txtOutput.Text = txtOutput.Text & "9"

End Sub

Private Sub cmdPlus_Click()
txtOutput.Text = txtOutput.Text & "+"

End Sub

Private Sub cmdSave_Click()
Dim Stmp As String
If txtOutput.Text = "" Then
MsgBox "Enter the number", vbCritical + vbOKOnly, "Add contact"
Else
Stmp = InputBox("Enter the name")
```

```
 If Len(Stmp) = 0 Then Exit Sub
With Data2.Recordset
.AddNew
!Name = Stmp
!Number = txtOutput.Text
.Update
MsgBox "Contact added", vbInformation + vbOKOnly, "Contact"
End With
End If

End Sub

Private Sub Form_Load()

Label10.Caption = Date

With Data1
.DatabaseName = App.Path & "\mobile.mdb"
.RecordSource = "Contact"
.Refresh
End With

With Data2
.DatabaseName = App.Path & "\mobile.mdb"
.RecordSource = "Contact"
.Refresh
End With

With Data3
.DatabaseName = App.Path & "\mobile.mdb"
```

```
.RecordSource = "CALL"
.Refresh
End With

End Sub

Private Sub Data1_Error(DataErr As Integer, Response As Integer)
 'This is where you would put error handling code
 'If you want to ignore errors, comment out the next line
 'If you want to trap them, add code here to handle them
 MsgBox "Data error event hit err:" & Error$(DataErr)
 Response = 0  'throw away the error
End Sub

Private Sub Data1_Reposition()
 Screen.MousePointer = vbDefault
 On Error Resume Next
 'This will display the current record position
 'for dynasets and snapshots
 Data1.Caption = "Contact: " & (Data1.Recordset.AbsolutePosition + 1)
 'for the table object you must set the index property when
 'the recordset gets created and use the following line
 'Data1.Caption = "Record: " & (Data1.Recordset.RecordCount *
(Data1.Recordset.PercentPosition * 0.01)) + 1
End Sub

Private Sub Data1_Validate(Action As Integer, Save As Integer)
 'This is where you put validation code
 'This event gets called when the following actions occur
 Select Case Action
```

```
    Case vbDataActionMoveFirst
    Case vbDataActionMovePrevious
    Case vbDataActionMoveNext
    Case vbDataActionMoveLast
    Case vbDataActionAddNew
    Case vbDataActionUpdate
    Case vbDataActionDelete
    Case vbDataActionFind
    Case vbDataActionBookmark
    Case vbDataActionClose
  End Select
End Sub

Private Sub Frame1_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As
Single)
Label7.FontUnderline = False
Label7.ForeColor = vbBlack
End Sub

Private Sub Image1_Click()
Text1.Visible = False
lblCalling.Visible = False
Timer2.Enabled = False
MsgBox "Call ended", vbInformation + vbOKOnly, "Calling..."
End Sub

Private Sub Image2_Click()
Text1.Visible = True
Text1.Text = txtName.Text
lblCalling.Visible = True
```

```
Timer1.Enabled = True
Timer2.Enabled = True
Image3.Visible = True
Label3.Visible = True
End Sub

Private Sub Image3_Click()
Text1.Visible = False
lblCalling.Visible = False
Timer2.Enabled = False
MsgBox "Call ended", vbInformation + vbOKOnly, "Calling..."
End Sub

Private Sub Image4_Click()
Text1.Visible = True
Text1.Text = txtOutput.Text
lblCalling.Visible = True
Timer1.Enabled = True
Timer2.Enabled = True
Image1.Visible = True
Label5.Visible = True
End Sub

Private Sub Label7_Click()
Frame2.Visible = True
txtOutput.Visible = True
Frame1.Visible = False
End Sub
```

```
Private Sub Label7_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As
Single)
Label7.FontUnderline = True
Label7.ForeColor = vbRed
End Sub

Private Sub Timer1_Timer()
Dim ab As Long
ab = QBColor(Rnd * 15)
lblCalling.ForeColor = ab
End Sub

Private Sub Timer2_Timer()
lblCalling.Visible = True
MsgBox "The number is not available", vbInformation + vbOKOnly, "Calling..."
lblCalling.Visible = False
Text1.Visible = False

With Data3.Recordset
.AddNew
!Name = txtName.Text
!Number = txtNumber.Text
!Date = Label10.Caption
!Time = Label9.Caption
.Update
End With

End Sub

Private Sub Timer3_Timer()
```

```
Label9.Caption = Time
End Sub

Private Sub cmdExit_Click()
Unload Me
End Sub

Private Sub cmdSave_Click()
With Data1.Recordset
.AddNew
!message = txtOutput.Text
!Time = Label1.Caption
!Date = Label2.Caption
.Update
End With

MsgBox "Message saved", vbInformation + vbOKOnly, "Save message"

End Sub

Private Sub cmdSen_Click()
Timer1.Enabled = True
lblSending.Visible = True
txtNum.Visible = False
End Sub

Private Sub cmdSend_Click()
Dim Stmp As String
  Stmp = InputBox("Enter receiver's number:")
  If Len(Stmp) = 0 Then Exit Sub
```

```
 txtNum.Text = Stmp
 Frame2.Visible = True
End Sub

Private Sub Command3_Click()
Frame1.Visible = True
End Sub

Private Sub Command4_Click()
txtOutput.Text = ""
End Sub

Private Sub Form_Load()

Label2.Caption = Date

With Data1
.DatabaseName = App.Path & "\mobile.mdb"
.RecordSource = "save"
.Refresh
End With

With Data2
.DatabaseName = App.Path & "\mobile.mdb"
.RecordSource = "send"
.Refresh
End With

List1.AddItem "Please call"
List1.AddItem "I'm at home.Please call"
```

```
List1.AddItem "I'm at work. Please call"
List1.AddItem "I'm in a meeting, call me later"
List1.AddItem "Meeting is cancelled"
List1.AddItem "See you at"
List1.AddItem "See you in"
End Sub

Private Sub Label3_Click()
Frame1.Visible = False
End Sub

Private Sub List1_Click()
txtOutput = txtOutput & " " & List1.Text
List1.Visible = False
End Sub

Private Sub Option1_Click()
CommonDialog1.Flags = &H1&
   CommonDialog1.ShowColor
   txtOutput.ForeColor = CommonDialog1.Color
   Option1.Value = False
End Sub

Private Sub Option2_Click()
List1.Visible = True
Option2.Value = False
End Sub

Private Sub Timer1_Timer()
lblSending.Visible = False
```

```
With Data2.Recordset
.AddNew
!message = txtOutput.Text
!receiver = txtNum.Text
!Date = Label2.Caption
!Time = Label1.Caption
.Update
End With
MsgBox "Message sent", vbInformation + vbOKOnly, "Sending..."

Frame2.Visible = False
End Sub

Private Sub Timer2_Timer()
Label1.Caption = Time
End Sub
```