Supporting Mobile Developers through A Java IDE

(Olalekan, Samuel, **OGUNLEYE**)

This thesis is submitted in partial fulfillment of the academic requirements

for the degree of

Master of Science in Computer Science

in the Faculty of Science

University of Cape Town

July, 2008

CERTIFICATION

As the candidate's supervisor, I have approved this dissertation for submission.

A/Professor Gary Marsden

Signature: _____

Date: _____

DECLARATION

I declare that this thesis is my own work. Where collaboration with other people has taken place, or material generated by other researchers is included, the parties and/or materials are indicated in the acknowledgements or are explicitly stated with references as appropriate.

This work is being submitted for the Master of Science in Computer Science at the University of Cape Town. It has not been submitted to any other university for any other degree or examination.

Olalekan Samuel OGUNLEYE Date

DEDICATION

To my dear mother, who did everything possible to educate me.

ABSTRACT

There exist several challenges in supporting mobile applications. For example, creating a separate target application for each device type, leaving developers with a huge maintenance chore. Most desktop applications run on largely homogenous hardware so instead of writing the same code over and over again, developers only need to write modules to implement a particular need. This is because even though there are differences in PC hardware configurations, the same desktop application will work fine on any hardware as the operating system provides an abstract layer. This is the way mobile applications are expected to work. However, this has been divided into dozens of ill-assorted versions. Java mobile applications developers spend more time rewriting code to run on different versions of mobile devices more than they do actually creating application in the first place. This is an intolerable burden for small mobile developers, and it stifles mobile software innovation overall.

Mobile devices differ in a variety of attributes, such as screen size, colour depth and the optional hardware devices they support such as cameras, GPS etc. The differences often require special code or project settings for successful deployment for each device a developer is targeting but this creates a huge logistical overhead. One potential solution that is shipped with NetBeans IDE is to add a new configuration for each device, modify the project properties, add some pre-processing code, then build and deploy the application. In most cases, one configuration for each distribution of the Java Archive (JAR) one plans to build for the project is created. For example, if a developer is planning to support three different screen sizes using two sets of vendor specific APIs, one needs to create six configurations. This reduces the performance of the application drastically

and increases the size at the same time. This is not acceptable for mobile devices where memory size and processor performance are limited.

The goal of this research work is to support mobile application development through a Java IDE (the NetBeans IDE in this case). Therefore, our approach will be to modify the NetBeans IDE to better address the difficulty that was mentioned above – namely targeting applications for different platforms.

Our solution is to integrate another type of a preprocessor into the NetBeans IDE that will help alleviate the problems of the existing tool. Our approach is to directly implement this inside the NetBeans IDE to further support mobile application development with the NetBeans IDE.

ACKNOWLEDGEMENTS

Much adoration, praises and honour to God almighty, the ever living, nourisher of every soul and the bestower of knowledge as the number of His creatures increases to the extent of His pleasure as weighty as His throne and as numerous as His words, who in His infinite mercy brings this program to a successful conclusion.

Esteemed medium like this, always serve as a podium to sing praises of those who in one's process of self realization and continuous effort not only to survive but to live, have in one way or the other helped in navigating through the difficult terrain in one's life. Deep, ineffable feelings, emotion laden heart, yet… How I wish to find the all-conveying words, phrases or sentences to express fully my gratitude to all. How I wish its real, the world of dream where thoughts suffers no emotional attenuation of language, sparing one the agony of translating feelings and emotions into words, words that always fall short of the invested feelings. Yet, for wanting of a better medium, I will turn inside out, becoming all lips, verses upon verses cascading from within and erratically in tune with my mood eulogizing all.

To the greatest supervisor in the whole world, Professor Gary **MARSDEN**, who I remain ever grateful to, for his effort to make the program, especially the research aspect an interesting endeavour. It would have been a hell of time and experience for me, but for your high sense of understanding and cooperation, you are indeed more of a father than a supervisor to me. Your very helpful suggestion, constructive criticism, painstaking and meticulous supervision and constant words of inspiration and fatherly advice throughout the time this graduate study lasted are highly appreciated.

My appreciation goes to Grace Bugembe **Kamulegeya**, for your care, advice and support in the cause of compiling this work. Your psychological and emotional supports as well as numerous words of advice coupled with the proof reading and corrections are never lost on me. I am always indebted to you.

Few people believe in doing well without expecting commensurate, if not greater recompense. Adeniyi **Isafiade,** you belong to the company of these people your overwhelming care and concern for my well being is never lost on me at any point in time. I just want to say a very big THANK YOU because words alone are not enough to express my gratitude. I also remember the likes of Oluwaseun **Oyekola**, Micheal **Adeyeye**, Oluwafemi **Olaofe** and Temitope **Kadiri**; I so much respect the automatic warm attention that I received from you guys throughout the period of this programme, especially the financial assistance I receive from you guys from time to time.

This thesis will not be completed without mentioning the likes of the Abel **Ajibesin** and family. Thank you so much for the advice, moral support and warm reception I always receive from you throughout this programme. Also to the Isaac **Osunmakinde** and family, I will always be full of praise for all the wonderful support that I receive from you throughout this programme.

To all my colleagues in the redeemed Christian Church Fellowship. I say thank you to you all for all your spiritual support through the period of this programme. May God bless you all. You are all wonderful. Also, to all my colleagues in the HCI group, for

helping to enhance the clear focus and the determination to tackle the challenges and make the dreams come true. Andrew **Maunder**, Ming Ki **Chong**, Calvin **Pedzai**, Grace **Kamulegeya**, and especially to Calvin for helping through the graphics design. You are all fantastic guys. Thank you all for being there.

To my mother. Words enough will not give me chance for all your efforts in making sure that I become the person God has planned for me to be. My Prayer is that God will always allow you to eat the fruit of your labour. Thank you dear mama. My appreciation goes to my father for being there for me in times of need and for the words of encouragement and the prayer that works for me in times of trials. To My younger brothers and sisters. Bukola, Adebanji and Suliyat. To you I owe my success and my gratitude will be forever to you, for you are the architect of my life. Worthy of mention is Mrs. Adigun whose support, encouragement and advice I always enjoyed in the time past. Thanks a lot. My gratitude goes to a friend in deed, Olode-Ankirun Akeem Babatunde for your moral support and for being there always when the whole world turns their back on me.

Humane, humble, mentally alert, industrious but always looking relax and above all loving and caring you are. Tozama **QWEBANI**. Your tantalizing love and liberal disposition, highly intelligent mind, woke up the sleeping giant in me. Thank you for all your encouragement, word of advice, moral support, understanding, prayers and most especially the love that I receive from you throughout this programme. What an Ideal perfect partner you make.

Olalekan Samuel

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER ONE

INTRODUCTION

1.1 Introduction

Mobile devices are becoming popular more and more with millions of people acquiring and getting access to them every day. For instance, in China, there are 300 million mobile users and this number is expected to grow to 900 million by the end of 2010 (Young, 2005) and the mobile device market in the United States is increasing at an annual rate of 22% (Chen *et al.*, 2003). Furthermore, in an article published in the Sunday Times of South Africa, it was stated that South African has the fourth fastest growing mobile industry in the world and also the largest digital cellular market outside of Europe (Wesson *et al.*, 2005). There are different types of programmable mobile devices on the market today with each of them having different properties in terms of screen size, networking capability, memory capacity, audio and video support etc. Devices such as cellular phones, Personal Digital Assistants (PDAs), music players etc. can now run Java software such as games and business enterprise application (Young, 2005). New prospects are emerging for applications that are running on these devices especially in this post-PC era (Weyert de Boer *et al.*, 2006). Mobile devices are used often for personal use and as commercial tools in this new era. This means that applications aimed at such devices need to be developed and improved to give way to the construction of new mobile world (Weyert de Boer *et al.*, 2006).

Much of the current focus has been on creating content for mobile devices, since there are already millions of users who now carry these devices in their pockets on a daily basis

(Young, 2005; Jones and Marsden, 2005). Due to these facts, Alexander *et al.* (2006) claims that as mobile devices continue to become more popular, mobile applications and services will form part of the new era of information and communications technology, a prophecy we are already witnessing. Applications that have been developed for PC platforms will now need to be redesigned for the mobile platform (Weyert de Boer *et al.*, 2006).

Mobile applications offer new opportunities to access information, communication and entertainment. Presently, we are witnessing a huge uptake in both business (e.g. mobile banking) and entertainment (e.g. mobile gaming and photo sharing) applications. The popularity of mobile applications and services are now such that we feel it is time to look at how well mobile application developers are supported by the existing development tools. To this end, the focus of this research is to develop a tool to best support mobile applications development. Rather than start from scratch, we wish to extend an existing open-source tool for mobile application development.

Integrated Development Environments (IDEs) such as NetBeans, Eclipse, JBuilder and Visual Studio etc. are tools of choice to develop mobile applications and they are also instrumental in developing individual components for mobile applications (Soroker *et al.*, 2006).

While JBuilder and Visual Studio are not open source IDEs, NetBeans and Eclipse are open source IDEs that are based on the Java platform specifications. NetBeans is the

ideal IDE to alter as it is not only open source but because it is considered as the most widely used IDE for Java application development for mobile devices (Benson *et al.*, 2004). White (2001) and Chen *et al.* (2003) argue that Java has a mature developer community and using Java technology on mobile devices has several important benefits. The major one being the fact that Java has cross-platform compatibility and due to this fact Java code can run smoothly with little or no modification on a wide range of devices (Chen *et al.*, 2003).

1.2 Java in Mobile Devices; Java for Small Things

As a way to help support mobile developers' community, Sun Microsystems introduced the Java 2 Micro Edition (J2ME) in June of 1999 (White, 2001). The aim was to help mobile developers to deliver mobile applications more quickly by reducing code writing and at a low cost in terms of application deployment, and developer learning curves (Young, 2005). J2ME is an environment for running applications on mobile devices such as cell phones, PDAs, set top boxes, and embedded devices which includes a Java Virtual Machines (JVMs) and a subset of the Java 2 Standard Edition (J2SE) APIs (Young, 2005).

In order to address the diverse needs of wide spectrum of mobile devices, J2ME technology is defined by a set of layered specifications (White, 2001). These specifications were designed to allow developers to choose from a combination of configurations, profiles, and Wireless Message APIs (e.g. Bluetooth and RFID). Optional packages allow programmers to construct a complete Java runtime environment that closely fits the requirements of particular range of devices (White, 2001). Configurations

are specifications that address the virtual machine and general Java API running on a large group of devices while profiles are specifications that define the Java API to address specific needs of a particular subset of devices (White, 2001).

An implementation of a J2ME configuration specification (a configuration CLDC[1]) along with an implementation of a J2ME profile specification (a profile, MIDP[2] 2.0 or 1.0) provides a total environment capable of providing a Java application runtime for mobile and embedded devices (White, 2001). This means that a developer could select the MIDP 2.0 Profile, the CLDC 1.1 configuration and the WMA (Wireless Messaging API) optional library as a base for their applications. However, the application would be limited to devices that support the particular combination of MIDP 2.0, CLDC 1.1 and WMA (Young, 2005).

J2ME, aimed at mobile devices, opens up opportunities for the development of portable applications and it permits the development of challenging applications in the wireless world (Lino *et al.*, 2003), allowing applications running on the desktop platform to run on the mobile devices.

Finally, J2ME offers the requirement of being a light-weight mobile development platform which is suitable for running in limited devices.

[1] Connected, Limited Device Configuration
[2] Mobile Information Device Profile

1.3 NetBeans IDE as A Tool for Java Mobile Application

NetBeans Integrated Development Environment (IDE) is an open source, modular and integrated environment (Sun Microsystems, 2001). NetBeans was written in the Java language and because of Java platform independence, it runs on any platform with a Java Virtual Machine that is compliant with the Java 2 platform.

The NetBeans IDE builds its project infrastructure directly on top of Apache Ant (Lino *et al.*, 2003), and stores all the information about a particular mobile application developer's project in an Ant script, a properties file, and a few XML configuration files (Wilson, 2006)

The NetBeans Mobility Pack, which is a standard part of NetBeans, is used to develop Java Mobile Applications. Mobility Pack was quickly recognized as a leading mobile IDE and has been used by Java Mobile Edition developers and leaders (Hasik, 2006). Based on NetBeans IDE, the Mobility Pack includes the key features which contain an end-to-end application development and an Ant based build system (Hasik, 2006). With a highly customizable and open platform, the mobility pack has been seen as a great productivity enhancer for mobile applications developers (Hasik, 2006).

1.4 Between Desktop and Mobile Software

There exist several challenges in supporting mobile applications (Chen *et al.*, 2003). For example, creating a separate target application for each device type, leaving developers with a huge maintenance chore (Robert, 2005). Most desktop applications have common requirements – menus navigation, document management, settings and so forth. Instead

of writing the same code over and over again, developers only need to write modules to implement a particular need (Netbeans, 2006). This is because; even though, there are differences in PC hardware configurations, the same desktop application will work fine on the hardware as long as the hardware has the same operating system running on them. This was how mobile applications were supposed to work, but the environment was allowed to fragment into dozens of incompatible versions. "*Developers of Java mobile applications often complain that they spend more time rewriting code to run on different versions of mobile devices than they do actually creating the applications in the first place"(Mace, 2006).* As Mace (2006) puts it: "*This is an intolerable burden for small mobile developers, and it stifles mobile software innovation overall*."

1.5 Supporting Mobile Applications Development with NetBeans IDE

The goal of this research work is to support mobile application development through the NetBeans IDE. Therefore, our approach will be to modify the NetBeans IDE to better address the difficulty of multiple platforms.

As mentioned earlier, one of the most difficult aspects of developing mobile applications is the device fragmentation. Mobile devices differ in a variety of attributes, such as screen size, colour depth and the properties of the option APIs they support (Wilson, 2006). These differences often require special code or project settings for successful deployment for each device a developer is targeting, but can become a logistics nightmare (Robert, 2005; Wilson, 2006). One potential solution that is shipped with NetBeans IDE is to add a new configuration for each device, modify the project properties, add some pre-processing code, then build and deploy the application. In most cases, one configuration

for each distribution of the Java Archive (JAR) one plans to build for the project is created. For example, if a developer is planning to support three different screen sizes using two sets of vendor specific APIs, one should create six configurations. This reduces the performance of the application drastically and increases the size at the same time (Robert, 2005).

Our solution as presented in this thesis is to integrate another type of a preprocessor into the NetBeans IDE that will help alleviate the configuration problems in NetBeans. Although this has been partially implemented (Robert, 2005), it does not integrate directly with the NetBeans IDE and it was implemented in a complex way such that it will be difficult for mobile developer to figure out how this will be used when developing applications. Our approach was to directly implement this inside the NetBeans IDE to further support mobile application development with the NetBeans IDE.

1.6 Research Question

We want to see if it is possible to integrate a configuration pre-processor into the NetBeans IDE that allows programmers to organize their code more easily for multiple varied devices. Therefore, in this regard, our research question is: How we can support mobile developers through a Java IDE?

1.7 Thesis Outline

This dissertation consists of six chapters. Chapter one discusses the general introduction and background to the research study that was conducted. In Chapter two, we discuss the literature review, background information and related work in the world of mobile

applications. In Chapter three, we present the high – level details of the research design and methodology while in Chapter four, we introduce the design decision, as well as the design and implementation details of the prototype. Chapter five presents the empirical analysis, evaluation and result of the system's performance. Finally, in Chapter six, we present the conclusions and discuss the potential directions for future research.

1.8 Chapter Summary

This chapter has provided a brief introduction of the research study discussed in this dissertation. The discussion offered an indication of the literature upon which this study is based and which provided the background and theoretical underpinning for the study. The following chapter is therefore a comprehensive discussion on some of the literature accessible, which provided the basis and sound motivation for the undertaking of this study.

CHAPTER TWO

RESEARCH BACKGROUND AND LITERATURE REVIEW

2.1 Introduction

Mobile handset evolution began with car-mounted devices and then on through the phases of transportable, hand-portable and pocket phones to the phase of palm phones: a scenario where it is feasible for a person carrying mobile devices in his/her pocket not to even notice its existence (Lee *et al.*, 2005). Over the past twenty years, mobile devices have undergone a conversion from technology-focused professional tool to a mass-market, consumer product which is an important part of daily life of billions of people (Coen *et al.*, 2002), thereby providing concerns for mobile developers on how to keep on improving applications that run on mobile devices in order to satisfy the desire of consumers in regard to living their daily lives.

There are possibilities that the market for mobile devices will grow more than before, due to shrinking hardware, the improving form factors (i.e. entertainment applications and commercial applications), the cost, and the marketing model (Jones and Marsden, 2005). This can be proved by the increase growth rate from the year 2000, along with an explosion of mobile service adoption in Africa, America and Asia (Jones and Marsden, 2005).

Most of the development of mobile applications takes place, not on the particular device itself but on a personal computer (PC). There is every possibility to test the applications

on the computer that is being used for the development of a particular mobile application using emulator(s), but to a limited extent. In the case of J2ME (Java 2 Mobile Edition), testing applications for mobile devices on a desktop computers makes it easy for a developer to forget the expected target platform, yet the mobile phone, PDA or other device may have different behaviours when the application is finally transferred to that hardware. The Integrated Development Environment (IDE) and the emulators offer what can be regarded as a rough estimation of how an application will run when it is finally transferred or ported on a particular device that will run it. Meanwhile, in a worst-case scenario, the application may cease to function well when it is finally transferred to the device even after it has been fully tested on the emulator and it appeared to perform well (this was experienced with a small mobile game which was developed as part of the preliminary research to this work).

In this chapter, we focus on the requirements for developing mobile applications as well as what a good mobile applications environment should contain as part of its requirement to help mobile developers deliver good mobile applications. We start with an overview of mobile technology and that of mobile development environment.

2.2 Overview of mobile Technology

The telecommunication business started more than one hundred years ago when Graham A. Bell made his first call. Inter-personal wireless communication came to the fore when Bell Systems manufactured their first cellular systems in the late 1960s and early 1970s

(Yen. and Chou, 2000). Mobile devices such as mobile telephones came into wider release between the late 1970s and early 1980s (Winter *et al.*, 2004).

Over the past 25 years, the accomplishment, functionalities and capabilities of mobile devices have been enhanced with the sizes decreased by 94% and weight by 93% (Winter *et al.*, 2004).

Although, the mobile technology might have began from car mounted devices, the existing mobile telecommunication devices also include notebook computers, Personal Digital Assistants (PDAs), pen based computers, portable data collection and processing devices as well as hand-held mobile phones (Cooper, 2001; Jones and Marsden, 2005; Van Biljon, 2006).

Technology surrounding mobile communications are often described or classified according to generations and their capacities (Agrawal and Famolari, 1999). These technologies are grouped into First generations (1G), second generations (2G) and third generations (3G) in that order. The existing infrastructure of mobile devices (particularly hand held mobile phones) is based on 2.5G and 3G technology. Many areas in South Africa and in Africa at large do not have the 3G infrastructure (Van Biljon, 2006). However, 2.5G technology (GPRS, CSD, HSCSD and Edge) has been effectively and efficiently implemented, the speed of which allows supplementary applications such as web browsing (including sending and receiving emails with large attachment) (Lukkarit *et al.*, 2004).

2.3 Mobile Devices and their Services

A number of services have been deployed for mobile devices including news, directory services, and payment services. Mobile phone technology has been made popular due to the availability of mobile services at any time in any place (Winter *et al.*, 2004).

However, Hansel *et al.* (2005) argue that three key factors influence the use of mobile devices. These factors are:

- The physical limitation of the mobile devices and the characteristics of the applications running on the devices.
- Usage context of the mobile devices and
- The needs and characteristics of the mobile devices users.

Furthermore, these three factors are fundamental to the understanding of what influence mobile application developers in developing mobile application for any particular device. Starting with some classifications of mobile devices and with the discussion of mobile devices components, the services applicable or available to mobile devices can be addressed.

Young (2003) classified mobile devices as small computing devices that have three major properties, which are:

- They are devices that are bigger than microchip.
- They are devices that are smaller than desktop and
- They are devices that can provide visible software user interface.

For the purposes of this thesis, the key point is that devices have a software interface with which users of the devices can interact.

Mohageg and Bergman (2000) listed three main domain of use of mobile devices, which are entertainment, information access and communication. The mobile review (2003) also noted that manufacturers of mobile devices have numerous classifications, which are:

- Low end devices
- Middle range models
- Outdoors Devices (Devices protected from external influence)
- Business Phones
- Fashion phones
- Communicators and
- 3G Devices

However, the question worthy of note here is how mobile developers can develop applications that work across all makes of mobile devices.

2.4 Difference in Developing for Mobile Devices

Weilling (1999) argued that the proliferation of mobile devices such as smart phones and Personal Digital Assistant (PDA) opens new ways for developing new mobile application systems (such applications as Mobile-commerce or M-commerce system, Mobile-Banking, Mobile Gaming, etc.). In view of this, the line between the desktop computer and the handheld devices is starting to blur. The device capabilities are becoming less important, and the contents are becoming king such that new applications are developed

everyday to advance the production of these devices. The change brought by open platform technologies such as Java technologies and the Symbian operating system has opened up significant opportunities for mobile developers to develop new applications such as m-commerce, mobile games and many others (Abrahamsson *et al.*, 2004). However, these new applications raise some unique challenges. These challenges are associated with an environment of limited resources. Therefore, it becomes imperative to address the effects of such limitations when developing robust mobile applications.

Major differences in mobile devices relate to their physical characteristics such as size, weight, display size, data input mechanism and expandability while their technical characteristics include memory space, processing power and the operating systems (Abrahamsson *et al*., 2004; Van Biljon, 2006).

2.5 Desired Features and Requirements of Mobile Applications

According to the online interviews that were conducted with mobile application developers, two main features desired by mobile applications developers are: portability and facility of service composition to use the legacy systems. Early development of mobile applications was possible using proprietary solutions but portability was practically impossible compounded by the fact that there is diversity of device types that are available (Weyert de Boer *et al.*, 2006).

The process of porting an application from the desktop to a mobile device (that is transferring mobile applications developed on desktop computer) is becoming easier.

However, simply porting the application does not provide the best application for the end-user (Lino *et al.*, 2003). Good mobile applications are more than simply providing the same functionality and quality that the fixed environment provides. It is all about balancing the device limitations, the end-user environment and the needs of the end-user in which the application will become a well-designed application that performs well when it is being used in the real world (Lino *et al.*, 2004).

Users use mobile devices and the applications running on them frequently and for a small amount of time. The ability to quickly start up applications on mobile devices is imperative. Fixed computers have long session times and so a user might be willing to suffer larger startup times; but mobile applications have short session time so proportionately short startup times are required.

Mobile devices are like small tools that fit into one's pocket and this is the way that people expect it to behave. When it is being tapped or touched, the user will expect a response and if they do not get the response they become impatient and try again (Jones and Marsden, 2005). With this in mind, it is of utmost importance that users get some kind of acknowledgement immediately upon performing an action on a particular mobile device (Cheung *et al.*, 2007).

Consistency of experience is important for fixed computer applications but because these applications offer rich experiences, there are many ways to accomplish a given task such as mouse click, toolbars, keystrokes and menus. However, for a mobile application, there is only one way to accomplish a given task and the user gets implicitly trained on how to

do this (Jones and Marsden, 2005). Furthermore, because mobile devices are compact and self-contained, end-users naturally see the whole device as a single and unified experience, but each mobile device has its own configuration or pattern of elements, which is unified as a whole that it cannot be described merely as a sum of its parts. Therefore, a good mobile application does not appear so much as discrete applications (Cheung *et al.*, 2007) but as natural feature extensions of the mobile device experience.

One of the basic ideas of the mobile platform is freedom (Jones and Marsden, 2005; Weyert de Boer *et al.*, 2006), that is the ability to do what you want to do whenever you want to do it. This is because end-users expect to be independent of fixed infrastructures (Jaap van Ekris, 2006). Therefore, every other extra dependency that can be introduced by any mobile application can hamper the feeling of freedom. For example, an application should not be constantly dependent on the internet as it may not always be available.

One of the major challenges that are being faced by mobile applications is the ability to cope with the context and the user (Davies *et al.*, 1998). This is because mobile devices are not only used in an office environment like a desktop computer, but they can be used in an environment such as gym, standing in the rain or running to catch the train. Mobile devices can be used in virtually any environment. This has a great implication on mobile applications and what this means is that a good mobile application should fit the user's life more closely than a desktop application (Jones and Marsden, 2005).

With regard to requirements, mobile devices and their applications operate in much the same way as servers do (Cheung *et al.*, 2007). This is because just like servers, mobile devices and the application running on them are left operating 24hours a day and 7days a week and almost 365 days a year. Mobile devices like cell phones and PDAs together with the application running on them are often left running all time or have standby mode all the time. These modes make sure that at any point in time they come up in a state that closely resembles the state in which they were last operated on (Cheung *et al.*, 2007). Fixed computers are also left operating on all the time but the user still reboot them, log on and log off often, start and shut down applications very frequently. In this way improperly held system resources get flushed at an interval (Schill and Kummel, 1995) but this is not the case with the mobile devices and its applications.

Mobile applications have to deal effectively with the unexpected failures and so this must be taken into consideration when developers are developing applications for mobile devices. The operating system may shut down a background mobile application if it is running on low resources; for example, when battery is getting low and this will not mean that the data stored on it or the application will no longer operate. Mobile devices and its applications like mission-critical servers need to make sure that important data and the state they are managing is kept in longer-term storage which will be able to survive the application unexpectedly vanishing or failing (Jaap, van Ekris , 2006) and this should be done in such a way that data important to users are stored safely in a way that can be recovered in the event of any sudden failure.

2.6 Desired Features and Requirements of Mobile Development

A wide variation of mobile application environments have been created to help in implementing mobile applications. Many of these have created their own new programming toolkit to support mobile application development. For example, NetBeans, Eclipse, Visual Studio.Net Compact Framework, JBuilder etc. These are tools of choice for developing complex mobile application (Soroker *et al.*, 2006) such as mobile games, mobile web services, mobile entertainment applications, mobile commerce and a lot more. All these tools strive to support the full development cycle of mobile applications by combining a rich set of cooperating tools (Soroker *et al.*, 2006) such as user interface builders, compilers, debuggers and a source code editors.

A great deal of research has been conducted to ease the problem of mobile development. A number of systems have been developed to address this requirement; for example, Rover toolkit (Joseph *et al.*, 1997), Lime platform (Picco *et al.*, 2000), CAMAL (Alba and Favela, 2000), etc.

Munsonand Dewan (1997) developed Sync, a Java framework that enables programmers to create arbitrarily complex, synchronized, replicated data objects. Joseph, A.D. *et al.* (1997) developed Rover, which provides a framework for building mobile applications based on a flexible Client-server architecture. Picco *et al.* (2000) developed LIME, a middleware that was written in Java which supports mobile application development. Alba & Favela (2000) developed COMAL a framework for the development of collaborative applications development for handheld computers based on Palm OS. Litiu

and Prakash (2000) developed DACIA, a mobile component framework that supports the development of collaborative applications that allows user mobility. Roth and Unger (2001) developed Quickstep a platform for the development of asynchronous groupware applications running on handheld devices which provide communication and collaboration primitives that allows concentration on application-specific details. Sandoval *et al.* (2004) developed MADEE a development and execution environment for mobile applications which was targeted at handheld devices running Windows CE.

All of these tools were designed to support the implementation of specific features of mobile applications development but they do not consider the generic features of mobile application development.

Furthermore, there are small computing devices (i.e. mobile devices) everywhere, thereby the way people communicate and interact changes every day. However, applications for these devices are developed with more or less the same development tools that are used to develop conventional computer applications. In order to avoid complications in using these tools, developers of handheld computer applications need to find an alternative way for developing mobile applications (Sandoval *et al.*, 2004). This alternative should allow implementation of mobile applications faster and easier with the support from conventional computer applications.

In this case we consider what can be essentially referred to as list of desirable features, but it is worthy of note that a successful mobile application environment cannot be driven

or characterized simply by list of features. Some of these desirable features became clear after experience with some of the environment. The most notable features of all the features particular to mobile application developers is portability to and availability to a wide range of mobile devices.

Portability, while seemingly simple is in fact one of the biggest constraints of application development environments. Besides the issue of coding for multiple diverse devices, target devices can easily be obsolete and as a result, substantial effort put into use will be ignored. Any suitable mobile application development environment must be able to make the final application implemented and maintained on a wide range of mobile devices.

One of the other features is that a good and viable mobile application environment is extensibility. The extensibility of a mobile application environment enables it to be able to interact with external tool components (Soroker *et al.*, 2006).

Another feature is the ability to work with both design and implementation views of the current mobile application in the environment as well as the mapping between them and the ability to keep the mapping up to date. The ability to keep these mappings up to date is the reminiscent of the round trip problem in software engineering development (Salmre, 2004).

In addition, supporting collaboration is one of the key features of any software development platform (Soroker *et al.*, 2006). A mobile application development

environment must be able to support collaboration between programmers. This should be achieved in the sense that it would be possible for different developers to develop the same set of project into different hierarchies of composite projects i.e. one developer will be working on one part of the mobile application and another one on the other part and at the end of it all, they should be able to merge the projects together to form one without generating any problem at the end of the merger. This flexibility should be made possible by any mobile development environment.

"Nothing is as painful as developing an application and discover that it has so many errors when porting it to the target device(s)" (Micheal, 2006). Therefore, the benefit of detecting errors at the earlier stage in the development should be clear.

All these features should be supported together. Proving these via the design of an appropriate system should be one of the tasks involved in supporting mobile application developers through a Java IDE.

Considering these constraints, we propose the construction of an application development tool based on the existing development platform (i.e. NetBeans IDE) that will support and make easier the development of mobile applications that run on mobile devices. This application will include features to devise a more developer-friendly method for cross-platform mobile development as this is the primary aim of this research work.

2.7 Overview of J2ME Platform

According to Topley (2002), a new programming language known as Oak was created in the early 1990. The original focus of this programming language was on mobile devices and entertainment (Codepedia, 2003). "*Oak was developed as a result of the teams' experience with C++ which, despite having many powerful features, proved to be prone to programmers errors that affected software reliability*" (Topley, 2002). However, there was no market for this programming language at this time (Topley, 2002). During this period, there was a momentous beginning of public awareness to the reputation of the internet. As a result of this, a market for internet browsing software began to surface (Topley, 2002). This development compelled Sun to rename Oak to Java. Java was later used to develop a cross-platform browser called HotJava, which was later licensed to Netscape who incorporated it in its own popular browser (Topley, 2002).

Within a couple of years, the cross-platform capabilities of the Java programming language and its potential as a development platform for applications, that could be written once and run on both Windows and Unix operating systems created a lot of interest for commercial users as a way of reducing software costs.

Later in the late 1990s, Sun Microsystems released a second version of Java and this was called Java 2 Platform. It later became necessary to split this development platform into several pieces and the core functionality which was regarded as the minimum requirement to support any Java environment was packaged as Java 2 Standard Edition (J2SE) (Topley, 2002).

Several optional packages were later added to the J2SE core functionality in order to satisfy specific application development domain among which was Java 2 Enterprise Edition (J2EE), which had new technology incorporated in it such as servlets, Enterprise JavaBeans and Java Server Pages. These are all used to develop server side and networking applications (Topley, 2002). Along the line, the package for J2SE was renamed to Java Development Kits (JDK) and from there; versions of JDK starting from 1.1 were released. JDK 1.6 is the latest version in the Java World.

"*While Sun was busy developing and adding some other functionality to Java for internet programming and commercial applications, demand began to grow for Java on mobile devices*" (Topley, 2002). This was due to the fact that mobile devices have smaller amount of memory and J2SE requires far too much memory and processing power to be a viable solution (Sun Microsystems, 2003). This compelled Sun Microsystems to release PersonalJava, known as pJava in 1998 (Codepedia, 2003). However, it was later found that pJava, while it fit bigger mobile devices such setboxes better; it did a poor job on smaller mobile devices such as mobile phones, PDA, etc (Codepedia, 2003).

Therefore, as a way to help the development of applications for smaller mobile devices, Sun Microsystems introduced the Java 2 Micro Edition (J2ME) in June of 1999 (White, 2001). The aim of J2ME was to help mobile developers to deliver mobile applications more quickly on smaller mobile devices by reducing code writing and at a low cost in terms of application deployment, and developer learning curves (Young, 2005)..

J2ME is a platform for small mobile devices that was fashioned with the meaning to replace the variety of JDK based products with additional amalgamated solutions that were based on Java 2 (Topley, 2002; Sun Microsystems, 2003).

Unlike the desktop and server side that are targeted by J2SE and J2EE respectively, J2ME includes a wide range of devices with hugely different capabilities which makes it impossible to create a single platform that will suit all of them (Topley, 2002). It has a specification that defines a set of platforms as opposed to being a single entity. Each of these platforms is a suitable subset of the total collection of various devices that fall within its scope and this subset is defined by one or more profiles, which broaden the basic capabilities of the configuration (Topley, 2002).

Furthermore, J2ME configuration is a requirement that defines the development environment for a variety of mobile devices. Currently, there are two configurations defined in J2ME and these are:

i. Connected Limited Device Configuration (CLDC) which is designed for low end of the devices. Examples of which are mobile cell phones, PDAs etc.

ii. Connected Device Configuration (CDC) which was designed to find solution to the needs of devices that lie between those devices that were addressed by CLDC and the full desktop running J2SE. The devices that fall within this configuration have more memory capacities and more capable processor and can also support much more complete Java Environment.

Each of the J2ME configurations has what is referred to as Java Virtual environment (JVM) which is responsible for its hardware and operating system independence of the devices. It also consists of collection of Java classes that provide programming environment for the application software.

The J2ME profile complements the configuration by adding the additional classes that provide features that are appropriate to a particular device.

2.8 Overview of NetBeans Mobility IDE

NetBeans, an open source application, and originally called Xelfi, started as a student project in the Czech Republic in 1996. The goal was to write a Delphi-like Java IDE in Java. NetBeans was the first Java IDE written in Java.

In 1999, Sun Microsystems adopted NetBeans as its official Java Development tool and NetBeans became the first open source project that was sponsored by Sun.

Along the way people started building applications using NetBeans core runtime and their own plug-ins – applications that were not originally part of the development tool at all. One of the plug-ins that was developed was NetBeans Mobility, an IDE for the development of Java mobile applications.

2.9 Chapter Summary

In this chapter, the background of this thesis has been presented. We also presented the technology surrounding mobile applications as well as the overview of Java mobile technology and that of NetBeans, which serves as the major reason we decided to choose NetBeans as the IDE with which we can support mobile application developers. Also, we explored the differences encountered by mobile application developers in developing applications for mobile devices. We established that even though while the devices capabilities are becoming less important, we still have challenges in mobile devices which are associated with environments of limited resources. We also took an overview of mobile application models and we also looked at some of the environments for programming mobile applications.

We also outlined the requirements and features of mobile applications as well as the environment in which good mobile application can be developed.

Literatures of work done by other researchers were reviewed and discussed. The next chapter focuses on the approaches and methodologies considered appropriate in conducting this research.

CHAPTER THREE

RESEARCH METHODOLOGIES

3.1 Introduction

In the preceding chapters we presented the introduction to this study and the overview of relevant literature as well as the background study. These have provided a motivation as well as evidence concerning the need to investigate how we can support mobile application developers through an enhanced Java IDE for mobile applications development.

The goal of this chapter is to discuss and explain the various approaches used in conducting this research.

3.2 The Research Plan

This research focuses on the study of how mobile application developers can be supported through an enhanced Java IDE. In order to achieve this, we involved existing users of Java IDEs who had experience in developing mobile applications. That is, we wanted to observe and examine programmers as they were using a Java IDE in order to identify the problems they encounter when developing mobile applications through an enhanced Java IDE. The NetBeans IDE has been the focus in this study.

Our broad approach is that of User-Centered Design (Preece *et al.*, 2007), in which we observe the problems experienced by real users and, through an iterative process of design and evaluation, work towards a solution.

3.3 User Center Design Research

The importance of user-center design is based on involving the user throughout the whole life of product design (Nival, 2005). *"The design of a system is not always intuitive and at times leaves user frustrated and unable to complete a simple task"* (Abras *et al.,* 2004). User-Centered Design (UCD) is a broad term used to explain design procedures in which end-users impact how a design takes shape and it is both an extensive philosophy and diversity of methods (Abras *et al., 2004*).

Abras *et al.* (2004) and Holtzblatt *et al.* (2005) argued that although there is a range of ways in which users can be involved in user-centered design research methodology, what is of great consequence is that users are involved. Some types of user-centered design methodology check with users about what their needs are and include them at particular times during the design process; typically requirements gathering and evaluation (task-based evaluation in the context of this research). At the other end of the range there are user-centered design methodologies in which users have a profound influence on the design by being involved as associates and partners with designers all the way through the design process (Abras *et al.*, 2004).

The purpose of user-centered design (UCD) is the encouragement of the entire system development procedure with user-centered activities (Nivala *et al*., , 2005). This is done so as to produce applications that are easy to use and accomplish the needs of the proposed user groups (Jones and Marsden, 2005; Preece, *et al.*, 2007). User-centered design is considered to be imperative particularly when new applications are created (Preece, *et al.*, 2007).

From the above, one can easily infer that user-centered design is an ideal way to tap the knowledge users have about their work practices and carry that over into design (Golub *et al.*, 2001). The Fig. 3.1 below, that was adapted from (Rowan, 2006), shows the flow of design process for user-centered design approach.

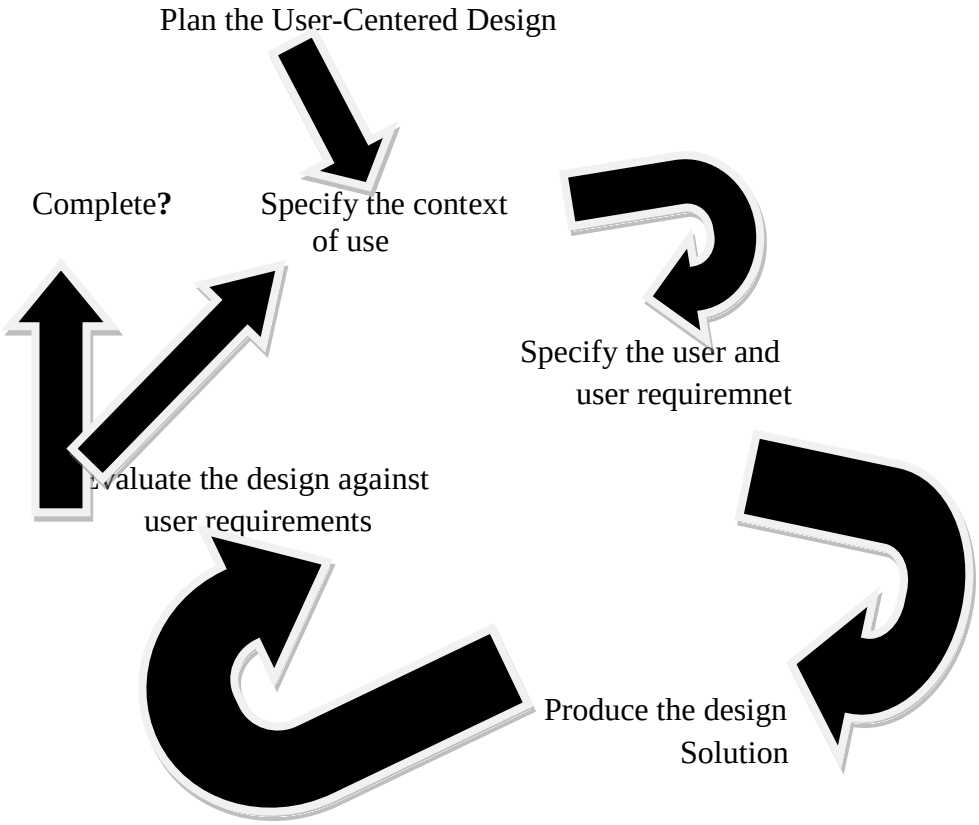

Fig 3.1 User-centered Interactive Systems

In order to conduct this research study, we will therefore use User-Centered Design as a guiding principle and specific techniques such as Contextual Inquiry (Holtzblatt, 2005) will be used to understand the problem this research is trying to solve. An important contribution of these techniques to this research studies is that while User-Centered Design helps us to generate more creative design solutions to the problems that mobile programmers encounter, Contextual Inquiry helps us to focus on observation and in-work

interviews in order to extract user requirements and be able to suggest a suitable solution to support mobile application developers.

3.4 Research Methodology

This section discusses the methodologies used to conduct this research.

3.4.1 Observation Using Contextual Inquiry

Observation is an effective technique for gathering data and forming requirement definitions at any stage of a research or during a system development (Preece, *et al.*, 2007). Dix, *et al.*, (1993) argued that observation, whether formal or informal, is indispensable if a researcher is to get an understanding of the research situation. In addition to this, observations made in the field help to fill in the details and nuances that are not elicited in the initial requirement gathering at the beginning of the research (Preece, *et al.*, 2007).

Giraudo and Bordegoni (2005) argued that in order to design and develop an intuitive and easy-to-use system that will fully support the intended users in carrying out their work, an observation of the users of the system must be carried out by the system designer or the researcher and the results gathered from this observation must be translated to a system that will fully support the user.

However, Preece, *et al.* (2007) explained three different techniques of conducting user observation in the cause of conducting a particular research. These are:

- Direct observation in the field

- Direct observation in controlled environments and
- Indirect observation by tracking users' activities.

Direct observation in the field is useful in a situation where users often find it difficult to accurately explain what they do and when details of the process of activities are assessed. Such tasks are being implemented according to the standards that are required for effectiveness (Preece, *et al.*, 2007). However, during the process, users do what they normally do without being disturbed by the observers and the researcher/observer records what is going on.

Direct observation in a controlled environment is an observation technique that usually occurs in a place other than users' normal environment (Stone, *et al.*, 2005). (For example, in laboratory environments). This type of observation is usually useful during the evaluation of a system or a design (Jones and Marsden, 2005, Preece, *et al.*, 2007) where a specially devised task will be performed with the observer (i.e. the researcher) recording the performance of the users in some way such as timing task or particular sequence of actions (Stone, *et al.*, 2005).

Indirect observation is an observation technique, where some records of past behaviour of users is used to deduce what happens during the event and to track users' activities. According to Preece, *et al.*, (2007), there are two techniques that are commonly adopted in achieving this type of observation. These are using diaries, a situation whereby the users are presented with a diary to write their activities on a regular basis. This means

that the researcher relies on the reported observations of the users (Wilson, 1999). The second technique is by using the interaction log (Preece, *et al.*, 2007). This provides a permanent record of the users' activities while the researchers are not directly available by powering a device to record users' activities in the form of a log that can be subjected to examination at a later stage (Stone, *et al.*, 2005; Preece *et al.*, 2007).

While the first and the last techniques (that is direct observation in the field and indirect observation by tracking users' activities) are usually used during the requirements gathering stage of a project (formative), the second technique (that is direct observation in a controlled environment) is usually used during the evaluation of a system – that is after the requirements gathering phase, when the system has already been designed (summative) (Preece *et al.*, 2007).

Difficulties with these techniques have been documented by different researchers. For example, Jones and Marsden (2005) argued that when using most of these techniques to conduct evaluation, the researcher sees himself and/or herself as an expert in the field by conducting observation to study and understand users. This means that researchers or designers do not see any reason for establishing an intimate relationship with the user. However, Beyer and Holtzblatt (1993) explained that it is difficult to understand users through observations alone and Wilson *et al.* (2002) argued that observation alone without interrupting or dialogue with the users in the context of their work is insufficient and that only users know what they do and why they do it. This can only be uncovered by dialogue with them which can be achieved through intimate relationship with the users.

Furthermore, Preece, *et al.* (2007) explained that most of these observation techniques can be complicated and can result in a lot of data that are not very relevant to the study in question.

Beyer and Holtzblatt (1995) argued that the current concentration on observation techniques is growing out of recognition in such a way that, using the traditional observation techniques (that is the observation techniques explained above) alone are not sufficient enough to support users. They further argued that building systems to support users requires more intimate understanding of the users in the context of users' work. This means that in order to support users, guidance must come from users themselves (Beyer and Holtzblatt, 1993). They therefore argued that there should be an approach that will improve the requirements definition by creating new relationship between researcher/designer and the users in which users will act as the guidance in the research process. Consequently, Beyer and Holtzblatt proposed Contextual Inquiry (Beyer and Holtzblatt, 1993), an observational approach which is tailored to gather data that can be used in designing a system that supports users' need (Preece *et al.*, 2007). It is a research approach for gathering data through observation of users with an intimate interaction and this suggests that the designer should see the relationship as one which involves an apprentice (that is the researcher sees himself/herself as an apprentice rather than an expert) and a master (that is the user under study) (Jones and Marsden, 2005).

Contextual Inquiry is a structured approach to the collection and understanding of data from fieldwork with the purpose of building a system (Preece *et al.*, 2007). It is a method that provides the researcher with a grounded and detailed knowledge of users' work as a

basis for their design (Raven and Wixon, 1997). It is usually achieved by fostering a strong relationship with the user. This relationship between the researcher/designer and the users determine how well the researcher/designer understands the user and this assumes that the users are the experts in their work (Beyer and Holtzblatt, 1995). This is mostly done through a face-to-face interaction using an apprenticeship model, which provides an attitude of inquiry, and learning while the users are being studied (Beyer and Holtzblatt, 1995, Jones and Marsden, 2005).

Kantner *et al.* (2003) argued that Contextual Inquiry defines a clear set of concerns, rather than a list of specific questions and this enables the researcher/designer to focus on few key issues and gather concrete data during the sessions that they may have with the users.

Contextual Inquiry is found on three principles (Raven and Flanders; Preece *et al.*, 2007 1996). These principles are briefly discussed below:

Data gathering usually takes place in the context of users' work. This highlights the significance of going to the users' workplace and observe what is happening (Preece *et al.*, 2007). This means that a researcher cannot really understand what users are doing unless he/she goes to see and experience the inter-related conditions in which users use the system in question (Raven and Flanders, 1996).

The second principle is that there is a level of partnership between the users and the researcher/designer. This is based on the premise that both the researcher and the users are equal (Raven and Flanders, 1996) and that they both collaborate in understanding the

work which the research is being conducted. Furthermore, they both agree that the understanding can only be achieved through the spirit of cooperation (Preece *et al.*, 2007).

The last principle is the principle of focus, which is based on the fact that the researcher is focussing on a particular goal rather than a specific set of questions that may not even help in conducting the research (Preece *et al*., 2007, Raven and Flanders, 1996, Kantner *et al*, 2003).

However, Preece *et al*. (2007) explained that the results from the three principles must be interpreted in order to be able to use it during the design. They explained that the best way to conduct the interpretation of these results is to discuss them with the users.
Therefore, Contextual Inquiry is different from other methods in that it is directed at generating new requirements and new ways to support users (Kantner *et al.*, 2003). This means that it is a discovery method rather than evaluative method and it generates understanding of users more quickly than the traditional observation method (Kantner *et al.*, 2003).

Considering these facts, we adopted Contextual Inquiry as a technique for observing our users (that is the mobile application developers). We observed the users interacting with their Java IDE as they develop mobile applications. We wanted to know the reason why they are using a particular IDE and what frustrations they might experience. This approach opened a direct dialog between the user and the researcher. It helped us in

gathering information and the resulting data from using Contextual Inquiry was more reliable than other potential approaches because it was based on in-the-moment experience (Raven and Flanders, 1996).

3.4.2 The Design Using Rapid Prototyping

The crucial aim of design is to develop a product that helps users achieve their goals (Preece *et al.*, 2007). Design actions only commence once the requirements have been established (Preece, *et al.*, 2007). For users to successfully critique the design, researchers/designers will have to come up with a prototype of their ideas from the requirements that was gathered (Jones and Marsden, 2005). "*A prototype is a limited representation of a design that allows users to interact with it and to explore its suitability*" (Preece *et al.*, 2007). However, Prototyping is the practice of realizing design ideas. (Chee *et al.*, 2007).

Prototyping is an important and vital part of User-Centered Design. It allows researchers/designers to attempt their ideas with users and to gather feedback as fast as possible (Bell College, 2001). The major purpose of prototyping is to involve the users in evaluating the design ideas and acquires their feedback and criticisms in the early stage of development, and to decrease the time and cost in conducting the research (Preece *et al.*, 2007). It also provides a well-organized and valuable way to improve and optimize the design through discussion, exploration, testing and iterative revision. Jones and Marsden, (2005) and Preece *et al.*, (2007) argue that early evaluation can be based on quicker and cheaper prototypes prior to the start of a full-scale implementation. The prototypes can be

altered many times until a better understanding of the system under design has been achieved with the combined efforts of both the designers and the users (Preece, *et al.*, 2007).

Prototyping has been proven to be a valuable technique throughout the disciplines of science and engineering (Stephen *et al.*, 1982). Chee *et al.* (2003) argued that the roles that a prototype plays in system development cannot be overemphasized while it also helps to resolve uncertainty about how well a system designed supports users in their activities.

Rapid prototying has turned out to be a vital means to verifty the performance and feasibilty of systems. This is due to its tremendous times savings ability to use it to gather information on further requirement and on the adequacy of a system (Kochan, 1992). Dix *et al.* (1998) argued that rapid prototyping provides a way for the researcher/designer to be able to see where improvements needed to be made as early as possible and also, rapid prototyping emphasizes the rapid synthesizes and utilisation of design as a way of examining problem and evaluating a solution (Houde and Hill, 1997).

Rapid prototyping is the essential activity that structures innovations, partnership, as well as creativity in design. It embodies a design hypothesis and thus enables the researcher to test this hypothesis as fast as possible (Klemmer, *et al.*, 2005). It is through the creation of prototypes that the researcher/designer learns about the problem he/she is trying to solve which will be made known through a partnership with the intending user by evaluating the design in order to make iteration a core concern as part of following rules

of User-Centered Design (Preece *et al.*, 2007, Jones and Marsden, 2005, Klemmer, *et al.*, 2005).

In this research study, we used rapid prototyping to rapidly prototype a system that can be used to validate our concept of supporting mobile application developers through an enhanced Java IDE.

3.4.3 Experimental Hypothesis

In this research study, we formulated hypothesis which can be proven or disproved by the use of suitable and reliable data (Ranjit, 1999) and in order to obtain the data that can be used to prove or disprove the hypothesis, we conducted an experimental evaluation to obtain suitable and reliable data and then conducted the analysis of this data. As stated earlier, our research question is how we can support mobile application developers through a Java IDE. To this end, we carried out an evaluation of the system that was developed in order to compare the simplicity and efficiency of our system as against the previous system (that is the NetBeans IDE).

3.4.4 Evaluation (A Task-Based Approach)

Users evaluations of systems are achieved by identifying the users, tasks and developing a procedure for capturing the problems that users may have during the evaluation of a system (Scholtz, 2005). The major part of evaluation in this study constituted a comparative evaluation for collection of data. This is generally termed as a task-based evaluation (Thomas, 1999) which was fully employed in this study.

3.4.4.1 Task-based evaluation

Task-based evaluation is a means of observating and/or conducting experiements (Scholtz, 2004) with the intended user of the system/design. Therefore, it is considered to be experimental in nature (Nielsen, 1992; Scholtz, 2004; Zhang *et al.*, 2007; Thomas, 1999).

Task-based evaluation is the process of receiving feedback, such as time to complete a task, error rates (quantitative method) as well as verbal feedback from the users in the form of opinions, problems and the general strength and weakness of the system (qualitative method) after they have completed a particular set of task that are of interest (Faulkner, 2000; Scholtz, 2004). This is frequently carried out during the design process (Preece *et al.*, 1994). Task-based evaluation is compatible with the qualitative data gathering (Nielsen, 1992; Faulkner, 2000) and it has been shown to allow researchers to be able to detect problems associated with the system by getting feedback from the user instead of conducting a full-scale experimental evaluation (Nielsen, 1992; Scholtz, 2004).

Thérése (1996) argued that evaluation forms an important part of any research and development effort, but the goal and focus of evaluation should be narrowed down to a specific focus and particularly the areas in which the study addresses. However, task-based evaluation focuses on evaluating whether the tasks of the users are achieved in using the system rather than evaluating the system performance (Thomas, 1999). In this research, we are not so interested in how efficient the users are in using the application, but rather how well the system conveys the goal of the user.

Therefore, our task-based evaluation methodology of the system to support mobile developers focussed on whether the goal of the users are communicated and whether they are achieved. An advantage of using task-based evaluation in this research is that it allows us to compare our system with the existing system by allowing the user to use both in performing the same operation and give us the feedback (Thomas, 1999).

3.5 Interview

"*Interviewing is a common technique for getting users to reflect on their experience in their own words*" (Jones and Marsden, 2005). Also, it is a type of conversation that is initiated in order to be able to gather information that is relevant to the research being conducted (Sears and Jacko, 2007). Interviewing involves asking questions from the participants in a particular study. Their views, their attitudes towards a particular thing, their perceptions and their behaviours serve as great tool in guiding research studies (Thomas, 1999). The participants that are involved in interviews are asked to reflect over their experiences with the aim of being objective and give the account of how events unfold in their own words.

In their studies, Zimmerman and Muraski (1995) argued that interviews can be seen or regarded as a process that starts with the need for recognition of specific information and then devise appropriate questions to acquire that information. They argued that a person with the appropriate expertise is identified and interviewed and in this process, the information is accessed. In the same way Dix *et al.* (1998) and Jones and Marsden (2005) argued that interviewing users about their experience with a system gives researchers the opportunity of getting direct information. With this in mind we wanted to interview our users with the aim of getting specific and direct information about the system from them.

According to Preece *et al.* (2007) interviews can be divided into three types. These are unstructured interviews, structured interviews and semi-structured interviews. Unstructured interviews are investigative and more like a conversation around a specific topic. The questions that are used in this type of interview require no exact format for the answers (Preece *et al.*, 2007). In structured interviews, predetermined questions that are related to those used in the questionnaire are always used. However, semi-structured interviews, also known as qualitative research interviews, combine the attributes of both structured and unstructured interview. In this type of interview, the interviewer begins with the most general question in order to gain some initial knowledge concerning the person being interviewed (that is the interviewee) and then move on to ask follow-up questions from the interviewee (Nielsen, *et al.*, 2005).

In this research, unstructured interviews were conducted after the evaluation in order to be able to gather more opinions from the user about the system under evaluation. This approached was also used because it provides rich data and it helps given a deep understanding of the research study (Preece *et al.*, 2007).

3.6 The Questionnaire

Questionnaires are a formalized set of questions for eliciting information or a formalized schedule for obtaining and recording specified and relevant information (Johnson, 1999). It is a self-report query based techniques which are produced typically on paper (Sears and Jacko, 2007), but due to the emerging technologies most especially the internet, researchers are now engaging the use of online questionnaire which saves a lot of time and serve to eliminate the problem of participating users' geographical distance (Sears

and Jacko, 2007). A questionnaire is a popular technique due to the fact that it has the potential to reach wide range of audiece and it is cheap to administer and can be analysed rapidly (Jones and Marsden, 2005).

A questionnaire contains a set of questions to be filled out by the users or participant of a particular research topic and the form allows for demographic information, views and opinions of the participants to be obtained. This means that questionnaires are used to elicit users' reaction to and opinion on a system or a design (Kuter and Yilmaz, 2001) and it is a reliable means of obtaining large amount of data (Faulkner, 1998). Questionnaires can be administered via an interviewer and it can also be self-administered, that is the participant reads and answers the questionnaire without any assistance from the researcher (Faulkner, 2000).

The response format of a questionnaire determines whether the questionnaire is unstructured, structured, semi-structure or both (Trochim, 2002). An unstructured questionnaire consists of open questions and allows participants to answer in any way that they want to (Faulkner, 1998; Preece *et al*., 1994). Structured questionnaires are compiled from closed questions which require users to select an answer from a set of predetermined answers (Faulkner, 1998; Preece *et al*., 2000) while a semi-structured question constitutes a mixture of both the structure and unstructured questionnaire.

This research study used questionnaires to establish the user demographics. Also a questionnaire was used to elicit users' opinion, views, interest as well as suggestions,

regarding the system under evaluation during the research process. The format of the questionnaire was semi-structured and it is self-administered. Response formats that were used in the questionnaire included open-ended and scalar (that is the Likert scales) (Olivier, 2004; Jones and Marsden; 2005, Preece *et al.*, 2007).

However, designing a questionnaire was a very difficult task (Jones and Marsden, 2005, Preece *et al.*, 2007). Therefore, apart from the online questionnaires which were used at the initial stage of this research, this research study used those questionnaires that had already been designed, used and tested in previous studies and had their validity tested and reliability verified. By this we mean that, the Questionnaire for User Interaction Satisfaction, (QUIS) (Chin *et al*., 1988) was used in this study. Although, QUIS is designed for evaluating user satisfaction, it has been consistently and frequently applied to evaluations of other aspect of system design (Preece *et al.*, 2007).

Preece *et al*. (2007) argued that the advantage of QUIS is that it has gone through many cycles of refinement and it has also been used for hundreds of evaluation studies which show that it is well tried and tested. This informed the reason that we used and adapt QUIS to match our evaluation.

3.7 Chapter Summary

This study fits most comfortably within the domain of user centered designed research. Therefore, this study will make use of observation (Contextual Inquiry in the context of this research study), prototyping (Rapid Prototyping) and evaluation (task-based approach).

Following well-established procedures of laying out the plans for this research, the research methodologies that need to be followed for a successful completion of this research study were therefore determined. We have been able to establish that Contextual Inquiry, which forms part of the new generation observation methodology, was the best method to learn about our user in this study. We discussed the importance of using a rapid prototyping approach in designing our system to support mobile application developers. We also argued that task-based evaluation is the ideal evaluation that can be used in this research study.

Chapter four gives an in-depth discussion of how User-Centered Design was implemented in the study and discusses the results obtained from applying the research methods. Chapter five presents the details of the evaluation of the design that was implemented.

CHAPTER FOUR

DESIGN DECISION, DESIGN AND IMPLEMENTATION

4.1 Introduction

In the previous chapter, the theoretical background and research methodologies to successfully conduct this research were presented. This chapter outlines the design of the prototype that was developed to support mobile application development. It also presents how the background theory was applied in implementing and designing the prototype.

At this stage, it is imperative to revisit the focus of this research – supporting Java mobile developers with a Java IDE to ease the development of mobile application solutions for mobile devices. This requires extending the NetBeans IDE by adding an extra module to it. Therefore, we wish to reconfigure NetBeans by adding a mobile application preprocessor to it allowing mobile developers to preprocess mobile applications for various mobile devices. This tool is called Mobile Tool for NetBeans (MTN).

4.2 Design Decision (Why Mobile Tools for NetBeans?)

Understanding how and where to improve the environment for Java Mobile developers, working on mobile applications, requires some investigation in order to learn how they do their work, while using a particular Java Integrated Development Environment (IDE) for developing mobile applications (Soroker *et al.*, 2006). In order to slight the extent of our research, we gathered input from developers during the early stage of our research work. As stated earlier, please note that developers in this research are students from computer science class. Therefore, we conducted survey in order to comprehend how

they evaluate their programming experience with Java IDEs, and how well Java IDEs support their work for mobile applications development. The survey included nine questions which were administered by means of interviews and questionnaires. We later followed this up by conducting observation of the users through contextual inquiry (CI) in their various work places. (Work places in this context refers to the various computer laboratories where the developers work). However, in order to complement our research effort, we also conducted an online survey through e-mail to an online user community of Java IDEs. This was in accordance with the suggestion of Zimmerman and Muraski, (1995) and Zimmerman *et al*. (2004).

Deep literature study on how the questionnaire that could be usable for capturing user needs and requirement regarding evnironment for mobile applications development was conducted. However, none of this questionnaire covered the data required in this study. We therefore held series of consultation with an expert in compiling the questionnaire and this led to suggestions being given on how the questionnare would be designed and adminstered. Information about the following aspects was captured:

- Biographic details: Name, gender, academic qualifications, choice of development platform and experience.
- Priorirties in selecting and using a particular development platform, features that are frequently used and the reason for this.
- Final suggestion (that what changes a subject will like to see in the development environment).

In total, 64 people responded to the questionnaire that was distributed. 24 of these were students from a class of computer gaming course while the remaining 40 were those developers from the online user community of Java IDEs

The following data was gathered from the survey: 76% felt that NetBeans supports the way in which they work. However, after follow-up interview, it was clear that mobile developers expect J2ME mobile applications to run correctly on all J2ME-enabled software and hardware platforms (e.g. J2ME-enabled mobile phones). But this is not always the case (Micheal, 2006).

Hence, the result of the survey shows that in a typical development, porting and testing mobile applications takes a longer time than expected in order to accommodate the variety of devices to be supported. To this end, our research focuses on better supporting developers in the creation of mobile applications for a variety of platforms. This can be done through a development environment (IDE) since almost all mobile developers are now developing mobile applications through one IDE or the other (Soroker *et al.*, 2006).

However, in this research study, we have focused on NetBeans IDE. This is because NetBeans IDE is an open source which allows for alteration. Also, our survey showed that NetBeans is considered as a widely used IDE for the development of Java applications and Java Mobile applications (Benson *et al.*, 2004).

The plugin we built is called Mobile Tools for NetBeans (MTN) and it will be used to aid the development of mobile applications that can be easily ported into different mobile devices without the need to adapt the application for each mobile device profile. MTN's major function is to help mobile developers preprocess source code to adapt mobile applications to various mobile devices. The goal is to keep only one form of source code which, when preprocessed, generates code and metadata which can be executed correctly on J2ME-enabled devices. The source code only needs to be written once along with accompanying directives for the tools. A device database, which is an XML file, only needs to be altered to contain all the devices the programmer wishes to target. Tool directives are the simple code snippets that help in the preprocessing stage. All the directives start with the Java comment code (that is the two forward slash //) followed by the pound symbol (#).

4.3 Mobile Tools for NetBeans (MTN) Module

This section presents a detailed discussion of how the MTN module interacts with the NetBeans framework and the techniques that were used in its implementation.

4.3.1 NetBeans Module Development Environment and Mobility Pack Module

NetBeans modules, also known as the Plugins, are an executable Java Archives (JAR) which contains Java classes and provide capabilities for the NetBeans IDE (NetBeans, 2007). Through the development of a plugin, the NetBeans IDE functionality is extended with additional features. Modules are written to interact with the NetBeans open Application Programming Interfaces (APIs). A module is identified by an identifier

called from a manifest file in the NetBeans environment. A manifest file is a special file type that contains information about the files that are packaged in the JAR file.

Modules can be developed independently and are added to the NetBeans platform as a Plugin. This means that modules developed using the NetBeans Environment can be extended using the NetBeans platform.

In essence, the NetBeans environment is an aggregation of modules and Plugins (NetBeans, 2007). Examples include the NetBeans Mobility pack, which integrates support for the development of mobile applications and the NetBeans profiler, which serves as a tool for the optimization of all Java applications in the NetBeans IDE.

All the packs and modules are developed independently and put together to form a single NetBeans platform for the development of various Java applications. As stated earlier, the modules themselves can be extended independently, so in this study, our focus is to extend the NetBeans Mobility pack to support mobile application developers so as to ease the development of application or solutions for mobile devices.

The NetBeans mobility pack is used to write debug and test applications for Java2 Micro Edition Platform (J2ME). It also integrates support for Mobile Information Device Profile (MIDP) 1.0 and 2.0, Connected Limited Device Configuration (CLDC) 1.1 and Connected Device Configuration (CDC). However, the scope of this research is limited to supporting mobile developers writing applications for CLDC-enabled devices.

4.3.2 Defining the XML Implementation

XML is used to define general syntaxes that are used to mark-up data in simple human-understandable tags. It is designed for structured documentation, that is, documents that comprise both content (words, pictures, etc.) and some suggestion of what functions those contents plays in the documentation (Norman, 1998).

It is possible to write applications that retrieve data in XML documents. This means that there is an extensive range of free libraries for Java that can read and write XML in order to focus on the unique needs of our application (Elliotte and Means, 2002). In our case, XML was used to implement device description files and project build files as described in the section "Prototype Implementation"

4.3.3 Implementing the Ant File

ANT, an acronym for Another Neat Tool, is an ever-present Java build tool that was written and developed by the Apache Jakarta Group. It is an Open-Source tool for Java based projects and it was wholly written in Java (Apache Ant, 2006). The NetBeans IDE uses ANT to build its applications. ANT is analogous to the UNIX “make” tool but the configuration is XML-based instead of writing the shell commands that the ‘make’ tool requires (Apache Ant, 2006). ANT takes its instructions from a build.xml file which instructs it on how and what to build.

ANT configuration (i.e. the build.xml file) was used as part of our tool to provide support for mobile developers. Although NetBeans has a built-in build.xml file which can be used to build any project in NetBeans, the fact is that this cannot be easily customized

(that is, it was difficult to edit this file, because an attempt to change the file will destabilize the configuration and also there is a warning message at the beginning of the NetBeans build.xml stating that a user should not make any changes to it), even though NetBeans is an open-source platform. This means that the build.xml file by NetBeans was difficult to setup in order to be used for our MTN application. This, however, prompted the reason for another build tool to be implemented for the purpose of this research study. The build tool that was implemented is an xml which is a standard ANT file for building any Java application generally and this includes Java mobile applications (that is J2ME). It allows mobile developers to control and manage the build process during the compilation of the mobile application source code. The build.xml file is a generally accepted name which indicates the main purpose of the file. It is arranged with the project element, first describing the name of the project, followed by the property and target elements. However, the property and target elements do not need to be sorted in any other but the default target specified by the project must be the first target listed.

4.3.4 Prototype Implementation of Mobile Tools for NetBeans

This section presents the implementation details of the prototype for Mobile Tools for NetBeans.

4.3.4.1 Presenting Mobile Tools for NetBeans (MTN)

In this study, data were gathered from different sources in order to create a potential solution. As stated earlier, Contextual Inquiry (CI) was adopted as a technique for observing our users (that is mobile application developers). The developers were observed interacting with their Java IDEs as they used it to develop mobile applications.

They were examined when using their favourite IDE to develop mobile applications. This is because we were interested in knowing the reasons why they were using a particular IDE and what frustration they might experience while they were using it. This opened a direct dialog between the users and the researcher as well as helping the researcher in gathering information and data that are reliable (Raven and Flanders, 1996).

We observed and chatted with users in various computer laboratories where the users develop mobile application, and this took place over a period of 90 days. The computer laboratories are located in the Computer Science department building. A laboratory (called the senior laboratory) belonging to Computer Science department is located in the Chemistry department building. Users were observed in this laboratory as well.

The findings through observation which is Contextual Inquiry and survey which were followed by questionnaire and interview were integrated into the findings from the literature study. The integration was done in order to answer the research question that was earlier asked at the beginning of this study namely, “How we can support Mobile Applications Developer through a Java IDE?”

The knowledge gathered from these findings were integrated and put together as a solution. We presented the solution as Mobile Tools for NetBeans (MTN) which is based on integrating a mobile preprocessor directly into the NetBeans IDE in order to support mobile developers in mobile applications development.

4.3.4.2 Designing the Mobile Device Collection

The mobile device collection specifies the properties of the J2ME-enabled mobile devices and generic mobile devices. This is a database collection in an XML format. A set of generic devices are defined in XML which can be used to prepare the application for unknown devices properties. The properties of known devices are also defined in this collection and can be used to develop applications for known mobile devices. A mobile device can belong to a category (known as device assembly, such as D series for Samsung, 6600 for Nokia and Razor for Motorola) as well as manufacturer (such as Samsung, Nokia, and Motorola etc) and these are included in the device dataset.

Each device supports different properties such as screen size, screen resolution, audio setup etc. Depending on the type of the device that a developer is targeting, these entire have to be defined in an XML based mobile device collection. An example of the mobile device collection and is given below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--  A SAMPLE code for device database-->
<!DOCTYPE deviceCollections [
<!ELEMENT deviceCollections (mobileCollection,classifier,
deviceassembly,deviceXtics,xtics)>
<!ELEMENT mobileCollections (#PCDATA)>
<!ELEMENT classifier (#CDATA)>
<!ELEMENT deviceassembly (#CDATA)>
<!ELEMENT deviceXtics (#CDATA)>
<!ELEMENT xtics (#PCDATA)>
]>
<deviceCollections>
<mobileCollection>
<classifier>Generic/mmapi</classifier>
```

```
<deviceassembly>Emulator</deviceassembly>
<deviceXtics>Generic</deviceXtics>
<xtics name=“JPlatform” value=“MIDP/2.0” />
<xtics name=“JConfig” Value=“CLDC/1.1” />
<xtics name=“JPackage” Value=“wmapi, mmapi” />
<xtics name=“AudioSetup” Value=“mp3, Hifi, midi” />
</mobileCollection>
<mobileCollection>
<classifier>Samsung/D900</classifier>
<deviceassembly>DSeries</deviceassembly>
<deviceXtics>withCam</deviceXtics>
<xtics name=“JPlatform” Value=“MIDP/2.0” />
<xtics name=“JConfig” Value=“CLDC/1.1” />
<xtics name=“OpSystem” Value=“WindowsMobile 5.0, Symbian OS 7.0s” />
<xtics name=“CamResolution” Value=“VGA” />
<xtics name=“ZoomingSize” Value=“3” />
<xtics name=“LCDSize” Value=“178X210” />
<xtics name=“AudioSetup” Value=“amr, mp3, hifi, midi” />
<xtics name=“memorySize” Value=“60mb” />
<xtics name=“JarSize” Value=“dynamic” />
</mobileCollection>
</deviceCollections>
```

The root element of the mobile device collection XML is the `<mobilecollection>` tag which encapsulates all the inner items that make up the whole device collection that describe the real devices. The `<classifier>` element identifies the device in a unique way simply by stating the name of the manufacturer of the device, separated by a single slash. An important aspect of this element is that, a developer can define various devices; that is a developer can define more than one device at a time simply by separating the classifiers with commas.

The <deviceXtics> element, which acts as Boolean variable, is used to express the Boolean features of the devices. Examples of this include, checking to see if a device has a camera, and if a device has a cursor. Several features of this Boolean potential can also be defined by separating them with comma. Also, the <Xtics> elements always act like variables with a unique name and value and these can contain various names and values separated by commas.

4.3.4.3 Designing the Mobile Device Manufacturer (XML)

This aspect of the XML file implements a definition for the various mobile device manufacturers that are J2ME-enabled. It is important to make sure that every device manufacturer that is defined in the mobile device collection XML is also defined in the device manufacturer XML. This is because users (that is mobile applications programmers) are familiar with a particular manufacturer and the specifications of a particular mobile device and so these must be defined. An example of the device manufacturer xml is given below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--  A SAMPLE code for device manufacturers  -->
<!DOCTYPE deviceManufacturers [
<!ELEMENT deviceManufacturers(manufacturer,name)>
<!ELEMENT manufacturer (#PCDATA)>
<!ELEMENT name (#CDATA)>
]>
<deviceManufacturers>
<manufacturer>
<name>Motorola</name>
</manufacturer>
<manufacturer>
<name>LG</name>
</manufacturer>
```

The <deviceManufacturers> element is the root element which contains all the definitions for the <manufacturer> element. The <manufacturer> element and the <name> elements which are nested together defines the names of the manufacturers.

4.3.4.4 Designing the Mobile Device Assembly

The device collection assembly XML file was implemented to define features and properties for many devices that share them together at a once. However, this was defined as a group in the mobile device collection xml. For example, the Samsung D series share many common properties with each other (e.g., they share the screen properties). Also, the device assembly xml can be used to select the suitable resources (e.g. sound or image) when developing applications for particular group of mobile devices.

An example of the device assemble xml is given below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--A SAMPLE code for mobile device assembly -->
<!DOCTYPE deviceassembly [
<!ELEMENT deviceassembly (assembly,name,classifier,xtics,parent)>
<!ELEMENT assembly (#PCDATA)>
<!ELEMENT name (#CDATA)>
<!ELEMENT classifier (#CDATA)>
<!ELEMENT xtics (#CDATA)>
<!ELEMENT parent (#PCDATA)>
]>
<deviceassembly>
<assembly>
      <name>Samsung-UI</name>
      <classifier>withCommandKeyEffect,
      withSpriteTransformation</classifier>
```

```
      <xtics      name      =      “class.fullscreen”,      Value      =
      “app.samsung.mid.ui.FullCanvas” />
      <xtics name = “JavaPackage”, Value = “samsung-ui” />
      <xtics name = “key.LeftSoftKey”, Value = “-4” />
      <xtics name = “key.RightSoftKey”, Value = “-6” />
      </assembly>
<assembly>
      <name>DSeries</name>
      <parent>Samsung-UI</ability><xtics  name  =  “JPlatform”,  Value  =
      “MIDP/2.0” />
      <xtics name = “JConfig”, Value = “CLDC/1.1” />
      <xtics name = “FullCanvasSize”, Value = 132x132” />
      <xtics name = “ScreenSize”, Value = “132x132” />
      <xtics name = “Emulate.Class”, Value = “SamsungEmulator” />
      <xtics  name  =  “Emulate.Skin”,  Value  =  “Samsung_Emulator_SDK_2.5”
      />
      </assembly>
      </deviceassembly>
```

The <deviceassembly> root element contains all the `<assembly>` elements that are used to define the actual assembly of the devices. The `<name>` elements define the name of the group. The `<classifier>` and the `<xtics>` elements are used just in the same way they were used for the mobile device collection and device manufacturer collection XMLs. The `<parent>` element is used as the name of an extended group and the child group will inherit all the features of the parent group.

4.3.4.5 Designing the XML Build tool

The build XML file informs ANT how to perform the work of building a project (that is building the application developed by mobile developer for mobile devices). This means that the build tool is a standard ANT file that is used to build a Java application. However, in implementing solution for this research study, the build tool was designed to

build Java mobile applications. During the build, four phases are passed through: Preprocessing, compilation, packaging and invoking the emulator.

During the preprocessing phase, the build tool changes the source code in order to adapt the application to different mobile devices. As stated earlier, the build tool is a standard ANT files for building Java application; therefore, the build tool translated the source code into binary bytecode during the compilation phase. This is done by including the APIs that support the target devices that a mobile developer is targeting. However, in the compilation and packaging phase, the application bundles are then created for each target device which will consist of one JAR file with the code for the application and one JAD file which consists of the necessary files needed to install the application on the target device(s). Finally, in the invoking emulator phase, the various emulators for testing the application are invoked. The XML code below shows an example of the build xml tool:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!--  A SAMPLE code for the build tool -->
<!DOCTYPE project [
<!ELEMENTproject(devicePrerequisites,build,emulator)>
<!ELEMENT devicePrerequisites (#PCDATA)>
<!ELEMENT build (#PCDATA>
<!ELEMENT emulator (#PCDATA)>
]>
<project
      name="MTN-Sample"
      default="mtn">
<devicePrerequisites>
<Prerequisites name =”identifier” value =”Generic/midp2,
Samsung/Dseriers” />
</devicePrerequisites>
```

```
<build>
<midlet class = “org.Netbeans.mtn.Sample.sampleMenu” />
</build>
</emulator>
</project>
```

The <project> element is the root element of the build XML tool which has two attributes. These are name attribute and the default attribute. The name attributes specifies the name of the project while the default target specifies the default target (that is the target to be run if no target is specified. The <target> element is used to name the set of task that are to be executed when the build XML is run; the <devicePrerequisite> element is responsible for selecting the devices for which the application is targeted. It has the child element <Prerequisite> with two attributes which are the name attributes to identify the devices and the value attributes to specify the value as well as the unique name of the devices. The <build> element signifies the point where the actual build takes place. It has the child element <midlet> which shows the directory to which the actual mobile application code can be found for build. The <emulator /> element is used to test the application on the development computer just before it is being ported to the real device. The complete build xml file is presented in the appendix.

4.3.5 The NetBeans Mobile Preprocessor

The NetBeans Mobile Preprocessor is a meta-program that is designed to format and manipulate the source code written by Java Mobile developers to provide for targeting multiple mobile devices. A meta-programming tool is a programming tool that is used to manipulate other programs, create and save the format for execution (Peter, 2006).

In this research study, the Mobile Preprocessor was implemented as set of Java classes. Only one class, the preprocessor class, is performing the major task of the MTN while others serve as the supporting classes. However, the UML diagram that shows how the classes fit together with the NetBeans is given below. Figure 4.1 shows the Overall UML Class Diagram while Figure 4.2a, b and c show the particular sections in detail. These classes were developed to interact with the XML files explained in the previous section of this dissertation. This was designed as a module and plugged in to the NetBeans platform.

4.3.5.1 Prototype Overview (The NetBeans Mobile Preprocessor)

The information presented in the implementation section constituted the prototype that was developed as a way of supporting mobile developers through a Java IDE (i.e. NetBeans). The prototype was developed as a module which can be plugged-in to the NetBeans Integrated Development Environment and this is the NetBeans Mobile Preprocessor.

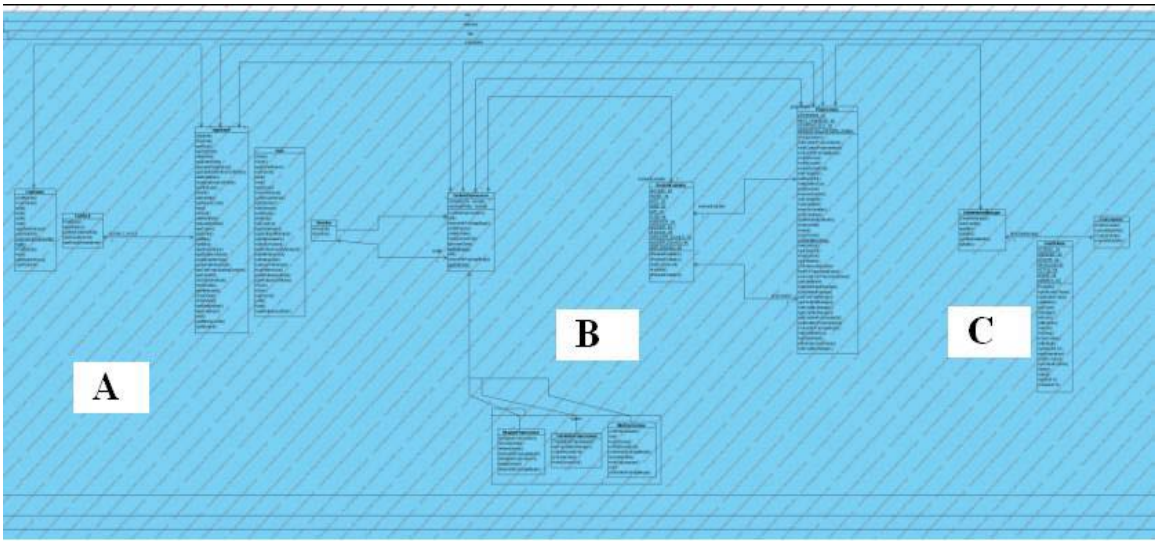

Fig 4.1 Overall UML class Diagram

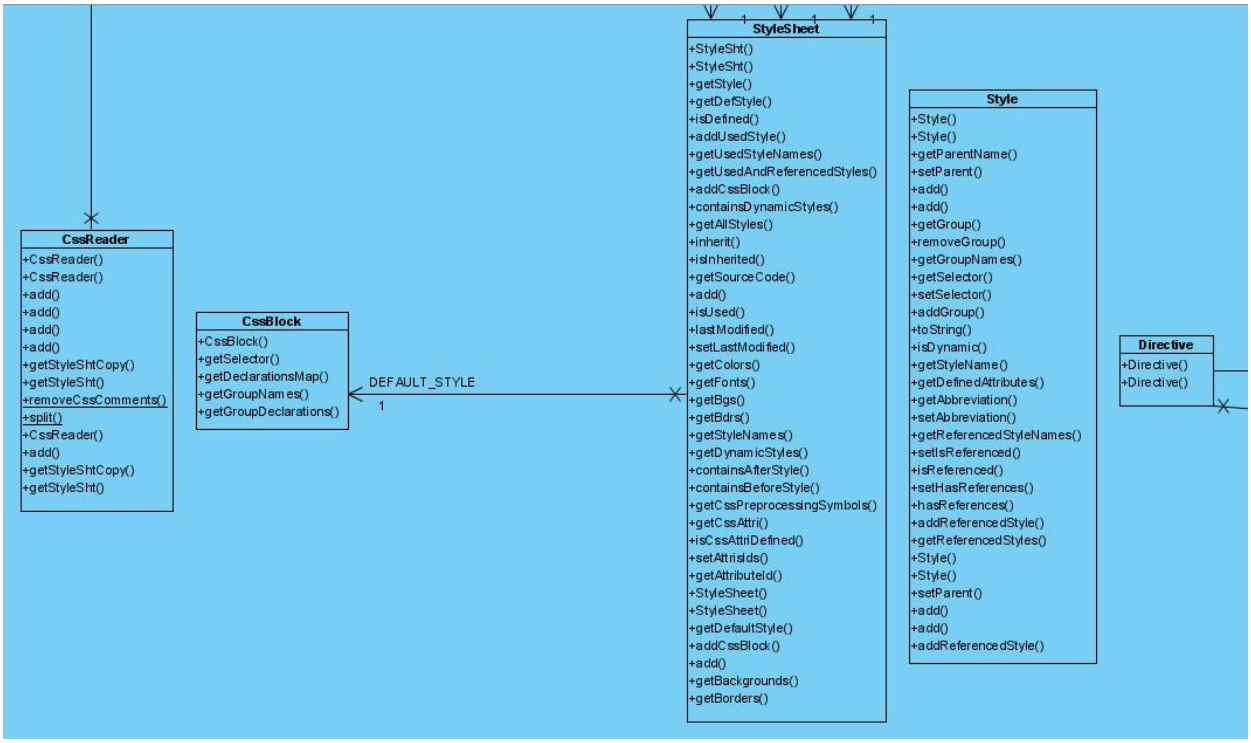

Figure 4.2a shows section A of the overall UML Class Diagram

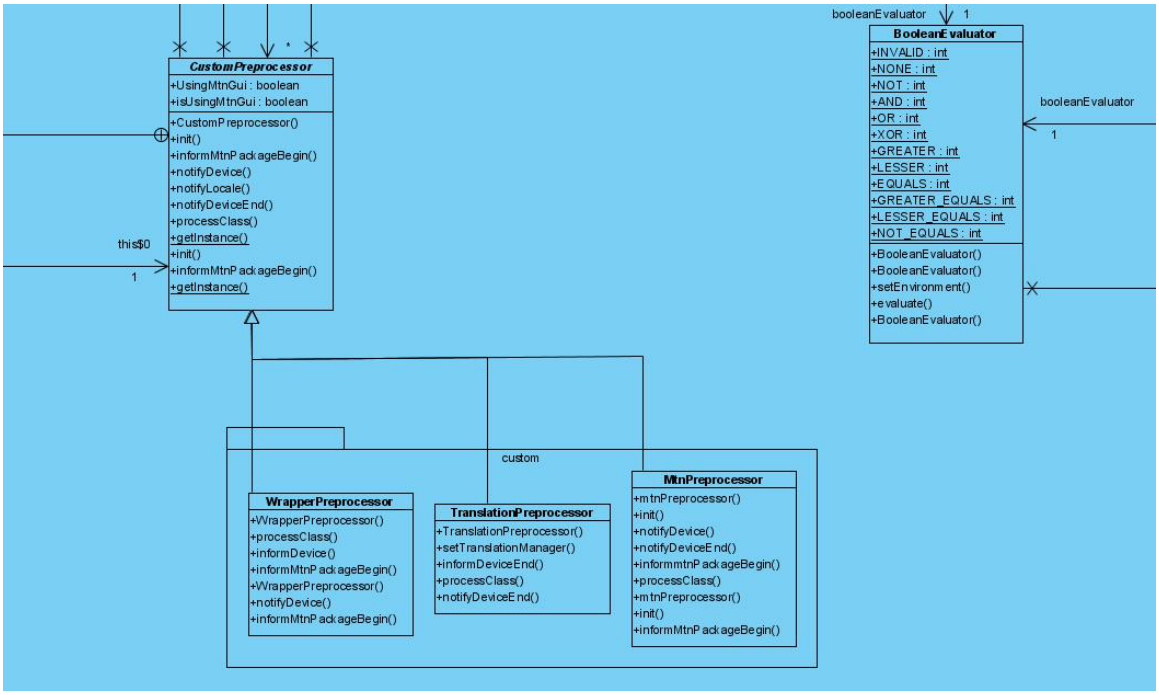

Figure 4.2b shows section B of the Overall UML Class Diagram

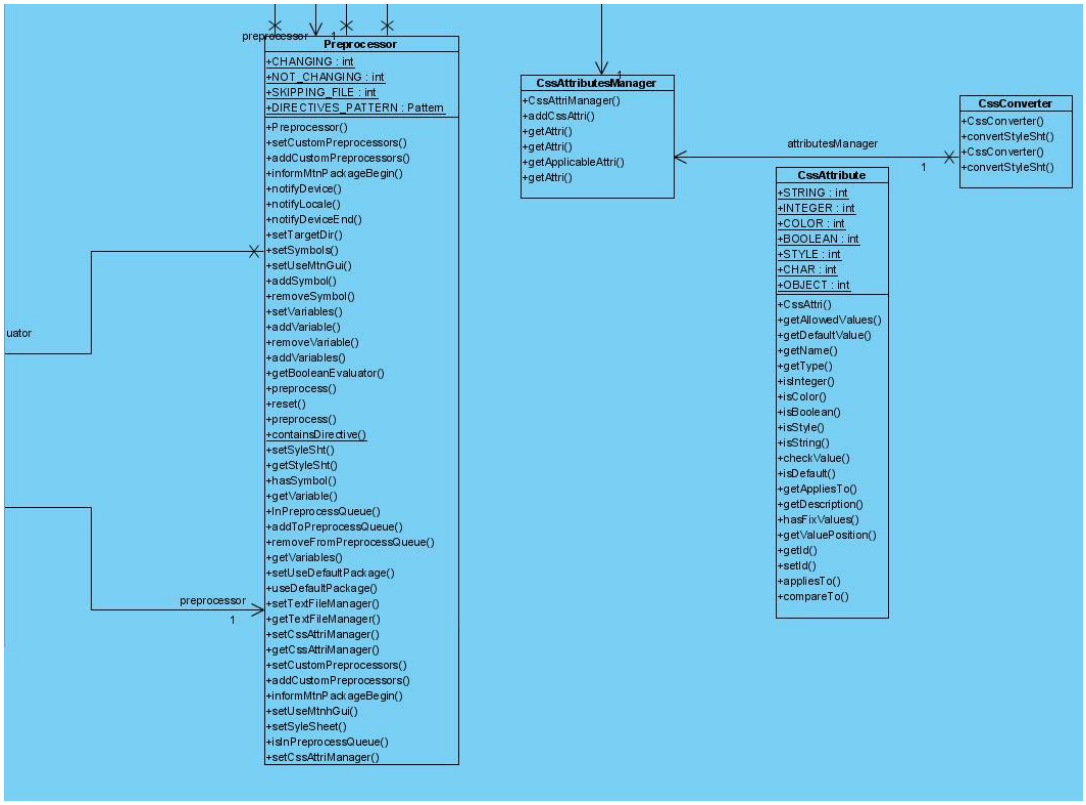

Figure 4.2c shows section c of the overall UML Class Diagram

4.4 How the NetBeans Mobile Preprocessor Works

The Mobile Tools for NetBeans (MTN) changes the application code before it is compiled by the compiler and this is only controlled by using the preprocessing directives. Section 4.3.1 describes the preprocessing directives that are available and implemented in this research. It also allows a developer to adjust their applications to different devices by using the characteristics of the current target device(s) for the preprocessing. Below is a brief description of how the preprocessing directives work. The full list of the preprocessor directives is given in the appendix.

4.4.1 Using the //#ifdefine, //#ifndefine, //#elseifdefine, //#else and //#endifdefine Directives

These directives are used to check whether single and multiple preprocessing symbols are defined or not in the device dataset. The preprocessing symbols are like Boolean variables defined in the <deviceXtics> element of the device dataset, for example `<xtics name = “AudioSetup”, Value = “mp3, hifi, midi” />`.

An example is playing sound only when the MMAPI (Mobile Media API) Samsung API is available:

```
Public void playSound() {
//#ifdefine mtn.api.mmapi || mtn.midp2
Try {
 Audio audio = audiomanager.createAudio( getClass().getResourceAsStream
(“audio.mp3”),“audio/mp3”)
Audio.start();
}catch (Exception e) {
System.out.println(Not able to play the audio” + e )
}
//#elseifdefine mtn.api.samsung-ui
Try {
byte [] dataSound = dataSoundLoad();
Sound sound = new Sound (dataSound, Sound.Tone_Format);
Sound.start(1);
} catch (IllegalArgumentException e) {
System.out.println(“Not able to play the Samsung sound” + e)
}
//#else
System.out.printly(“There is no audio supported for this device”);
//endifdefine
}
```

4.4.2 The //#inex

This preprocessing directive is used to include or exclude a symbol or variable to support the target device(s) that a developer is developing the application for. For example, including a sound only when the target device(s) supports the requirements. This directive accepts all the //#define directives. It can be placed in anywhere in the application code but it is usually placed at the very beginning of the code. For example

```
//#inex mtn.api.mmapi || mtn.api.Samsung-ui
package com.sun.j2me.audio;
public class audioPlayer {
…
}
```

4.4.3 //#def and //#undef

The //#def and //#undef directives are used to define and remove temporary symbols and variables in the source code and their names are usually preceded with *temp*. For example, when complex preprocessing terms have been created, a developer can set a preprocessing symbol that will show the result of these terms when tested. An example is given below:

```
public void animationOrImage
//#ifdefine mtn.midp2 && ( mtn.bitspercolor >= 16)
      //#def temp.message = It is a full Colour!
 //#elseifdefine mtn.useFullScreen
      //#def temp.message = It is a fullscreen!
//#endif
```

4.4.4 Communication between the Preprocessor and the Compiler

Each time a developer compiles the code using the build xml file, the build xml file interacts with the Java compiler as well as the preprocessor class and it makes sure that the preprocessor runs first. The preprocessor then looks for the preprocessing instructions each of which begins with the // and the pound symbol (#). The effective result of this is to change the text of the source code and generate a new source code file which is temporary and the developer does not see these codes. However, the compiler does not read the original source code with the preprocessing directives but reads the output of the preprocessor and compiles the file.

4.4.5 Inheritance of MTN

MTN supported inheritance. This was implemented in the preprocessor class which communicates with the XML files. As stated earlier, a device database which is an XML file, only needs to be altered in order to accommodate the specifications of all the devices that a mobile programmer wishes to target. This means that if for example there is a change in any of the devices that a programmer wishes to target, all that needed to be done by the programmer is to alter the device database to accommodate the changes in the specification. These changes will be reflected automatically when the application(s) is preprocessed.

4.5 Chapter Summary

This Chapter shows how the results gathered from the methodology presented in Chapter three were used to develop the prototype that established the ideas and the research goals presented during this dissertation. A clear indication of the research goal was reviewed

and presented at the beginning of the Chapter. The details of the design decision that led to design and implementation of each constituent of the prototype were also presented.

The prototype was built as a module application that can be plugged-in to the NetBeans Integrated Development Environment and finally the overview of the prototype was presented. As stated earlier, the prototype was a set of Java classes that were put together to serve the purpose of supporting mobile developers.

The next Chapter discusses the Evaluation and result of the prototype that was carried out.

CHAPTER FIVE

EVALUATION AND RESULT

5.1 Introduction

In Chapter two, we presented the theoretical background that formed a basis for this research study while in Chapter three we presented the methodologies that are applicable to carry out this research. The results from these methodologies were applied in implementing a solution to the original problem, hence the design and implementation of a system to support mobile developer as presented in Chapter four. However, Jones and Marsden (2005) argued that designers may not know how useful their system is until an evaluation of that system has been carried out. This chapter presents the details of the evaluation that was conducted during this research study.

5.2 Support Application for User Evaluation (What to Evaluate)

As stated earlier, our evaluation focuses on determining the tasks the users achieved in using the system, rather than evaluating the system performance (Thomas, 1999). In this research we are not so interested in how efficient the users are in using the application, but rather how well the system supports the goal of the user (Preece *et al.*, 2007).To test this, a prototype application has been developed for the purpose of the evaluation. We are not interested in knowing wheather a programmer knows how to write code but rather how well the system can help the programmer achieve the tasks for which it was designed (Dumas and Redish, 1999). To this end, the sample code that we developed for the evaluation purpose was a simple mobile menu. This is a simple application and was developed because we want the tasks that would be carried out by users to be simple

enough so that users will be able to evalaute the system successfully (Dumas and Redish, 1999; Preece *et al.*, 2007).

5.3 Development of the Tasks

The following three tasks were developed in order to evaluate the MTN that was developed to support mobile applications developers.

Task 1: To develop a simple mobile application and preprocess this according to the various devices of their choice based on the experience acquired in the tutorial.

Task 2: To write a build (XML) file based on the experience acquired during the tutorial session.

Task 3: To use the build file to build and preprocess the application to various devices as defined in the device collections.

The chosen topics for the tasks were indentified to be simple to use during the evaluation after Nielsen's (1996) suggestion on tasks to be used during evaluation and therefore were considered most important. The efficacy of the tasks was reviewed by colleagues as well as the consulting HCI expert during the design of questionnaire. A pilot study was also conducted with the potential users who would not be involved in the main evaluation study in order to determine the viability of the experimental procedure. This also helped us decide the criteria for what would constitute successfully completing the task.

5.4 Research Hypothesis

Our hypotheses are:

- ***H1: Users should be able to pre-process their developed mobile application to various mobile devices to suit their needs at once.***
- ***H2: After an initial training session, users should be able to adapt and configure the pre-processor without interference.***

5.5 The Pilot Study

In order to successfully evaluate this system, it is important that a pilot study be conducted (Edwin *et al.*, 2001). "*A pilot study is a small trial run of the main study*" (Preece, *et al.*, 2007) and the major goal of the pilot study is to ensure that the purpose of the final evaluation is evident and feasible before it is conducted and also to identify any potential problem in advance and correct them.Therefore, a pilot study was conducted before the final evaluation was conducted.

However, it is difficult to find subjects who will be involved in this study but Preece *et al.* (2007) suggested that a designer can ask colleagues or peers to participate in the pilot study. Therefore, the pilot study was conducted with subjects who are colleauges and peers and these set of subjects were not allowed to participate in the final usability evaluation because of biasing and the potential to affect the result of the evaluation (Preece, *et al.*, 2007).

In this instance, we actually conducted an extensive pilot study, blending it with an heurisitc evaluation. Not only did we want to test the experimental procedure, but we

wanted to remove as many problems from the environment before testing with 'real' unbiased subjects. Please note, that by heuristic evaluation, we do not mean in the usual sense of employing Nielsen's heuristic (Nielsen J., 2005), but rather expert review by experts in programming. Leveraging their expertiese at this early stage allows us to uncover depper problems in the full evaluation.

5.5.1 Subjects in the Pilot Study

The subjects in the pilot study consisted of six people (two undergraduates students and three postgraduate students of computer science) and one HCI expert whose work was to assess the instructions, and suitability of the questionnaire used to gather data for the evaluation study, while also performing the evaluation study. These subjects are familiar with computer programming and also with the NetBeans Environment. Because of this, it was assummed that subjects will have an experience of the system, even though the functionality of the systems was explained to them. Some of them have also conducted similar evaluations in time past.

5.5.2 Pilot Study Environment

The pilot study took place in the postgraduate laboratory of the Department of Computer Science. The study was administered seperately to each individual. This means that each study consisted of one participant per session which lasted for close to 40 minutes.

The materials used during the pilot study consisted of the development machine running Windows XP service pack 2. Installed on this machine were, the NetBeans development environment and the Java Development Kit 1.5 (that is JDK 1.5).

The pilot study permitted a close observation of the interaction between the subjects and the prototype.These observations exposed the limitations of the system. Therfeore, the pilot study was useful in terms of a straightforward observation of the usage of the system as well as in terms of quick and verbal opinion from the subjects. The data that was gathered during the pilot study, however, was not analysed as part of the final evaluation (Preece, *et al.*, 2007). Instead the pilot study was regarded as a process to allow coarse corrections to be made to the prototype and experiment proceedure.

5.5.3 Result and Discussion from the Pilot Study

Four of the subjects that participated in the pilot study did not find any limitations with the prototype. Two of the subjects, however, discovered that a mobile device manufacturer file had to be implemented. They were therefore disappointed that the trial system did not incorporate data for real devices. This was noted and rectified in the final prototype. The observations made were that users will respond differently when using the prototype and also there were subsequent discussions with the subject in the pilot studies. Further discussion with the subjects (particularly with the HCI expert) led to a suggestion being given on how the instructions and questionnaires to be administered during the evaluation will be handled.

However, the next prototype was designed to address the concerns that were raised during the pilot study. This included implementing a database file that will incorporate data for real mobile devices. This was implemented by declaring the devices' specification in an XML database file. The reason for this change was that participant in

the pilot study argued that users (that is mobile applications programmers) are familiar with a particular manufacturer and the specifications of a particular mobile device.

Finally, the questionnaire meant for the users' evaluation was thoroughly reviewed and deemed to be acceptable.

5.6 The Evaluation

This section describes on the evaluation that was carried out on the prototype.

5.6.1 Selection of Subjects

During evaluation of a system, it is imperative to choose subjects that are people who currently use, or will use, the product (Dumasand Redish, 1999; Nielsen, 2000). However, Preece, *et al.*, (2007) argued that when conducting evaluation, it is important to recruit subjects who represent the sample population for which the system is targeted e.g users with some range of expertise in the context of the study. In this research study, the subjects are those who have had experience in developing mobile applications.

Molich *et al.*, (1999) and Spool and Schroeder, (2001) argued that it will take many more than five users to successfully evaluate a system. Also, Scholtz, 2005 suggested that more than five (5) or seven (7) subjects per cell is the recommendation for the evaluation of a system or design where a cell represents a class of subjects who represent the users. Furthermore, Dumas and Reddish (1999) suggested that the number of subjects in any evaluation should be between 6-12. Therefore, MTN was evaluated with, 10 subjects, all of which were students from the Computer Science department (1 PhD, 4 Masters, 4

Honours and 1 undergraduate). All the subjects have experience of developing mobile applications. Subjects were recruited through e-mail advertisements and through recruitment posters and they were compensated for their participation in the evaluation study. Nine of the subjects were males and one was female. Balancing for gender was considered less important than mobile application experience. This is because we were more concerned about getting experienced mobile application developers to successfully evaluate the system than getting an even gender balance.

5.6.3 The Evaluation Environment

In order to guarantee comfort and provide a familiar environment, the evaluation was conducted in the Department of Computer Science while the users' privacy and confidentiality was maintained throughout the process of the evaluation. This was done in order to consider ethical issues that are related to user evaluation as pointed out by Preece *et al*. (2002).

5.6.4 Evaluation Procedure

After the agreement/consent form was given to subjects to fill, sign and submit, subjects were introduced to the system and evaluation that was to be performed and instruction on how this would be done was given. The purpose of this was to make sure that all subjects were given the same information and instruction.

The subjects were asked to sit alone with a computer system running Windows XP and NetBeans version 5.5 as well as Java Development Kit (JDK) 1.5. Each subject that participated in the evaluation study did so seperately. Before starting the main tasks, the

subjects were given a copy of the sample mobile menu application and a sample of the build.xml file that would be used to run the application and were instructed to explore the sample application for up to 10 to 15 minutes to familiarize themselves with it.

Each subject was then asked to walk through the three tasks and they were asked to tell us what they were thinking as they walk through the samples and as they perform the tasks (think aloud) (Preece *et al.*, 2007; Jones and Marsden, 2005). They were given up to 10 minutes for the first task, 20 minutes for the second task and 10 minutes for the third task. If they did not finish a task within the allotted time they were ask to stop. When all the tasks were completed, the subjects were given a post-test questionnaire which consists of items derived from the QUIS user satisfaction questionnaire to fill and returned before leaving the evaluation room. When the questionnaire was completed, a debriefing session and an unstructured interview were held in which the subjects were asked for their opinion (Preece *et al.*, 2007).

We wanted subjects to complete these tasks to investigate and assess the suitability of the application as realistically as possible based on the following three assessments:

- How well the application was designed.
- How easy the system was to use in terms of time to complete tasks by subjects and error rates during task completion.
- How well the system supports mobile developers in developing applications for specific devices.

In summary, there were four different sections during each evaluation and all these took up to 1 hour on the average. These sessions were:

- Introduction of the system and the experiment to perform
- Tutorial
- Carrying out a task using the system
- Questionnaire administration, debriefing session and the unstructured interview

5.7 Data Analysis

Olivier (2004) argued that the concluding phase in any evaluation study is the analysis of the data gathered. However, experimental analysis requires a statistical analysis of the collected data (Jones and Marsden, 2005, Preece *et al.*, 2007). Therefore, the statistical analysis method(s) that is appropriate to analyze the data collected must be established.

For the purpose of this research study and to be able to present the result that were obtained from the evaluation study that was conducted during the course of this research, the data gathered were analyze using descriptive statistics. Descriptive statistics are used to make a description of the data gathered during a particular study and they provide summaries about the sample and measures which can be done through graphical analysis and they form the basis of the quantitative analysis of the data (Trochim, 2002). However, STATISTICA software was used to perform the descriptive data analysis. The use of descriptive statistics was employed in this study because it simply describes what is or what the data shows by simply reducing a larger amount of data into simpler summary (Trochim, 2002). Furthermore, we use descriptive statistics in order to give us an accurate picture of what is going on in our quantitative data (Straus, 2001).

5.8 Results

This section presents the result from the evaluation study that was conducted. The summary graph for all the data analysis are shown above.

5.8.1 Time to complete Task

The estimated time for the completion of each evaluation session was 1 hour 30 minutes, with the 30 minutes been the time allocated for the introduction and tutorial and 1 hour for the evaluation. However, every subject completed the task in less than 1 hour.

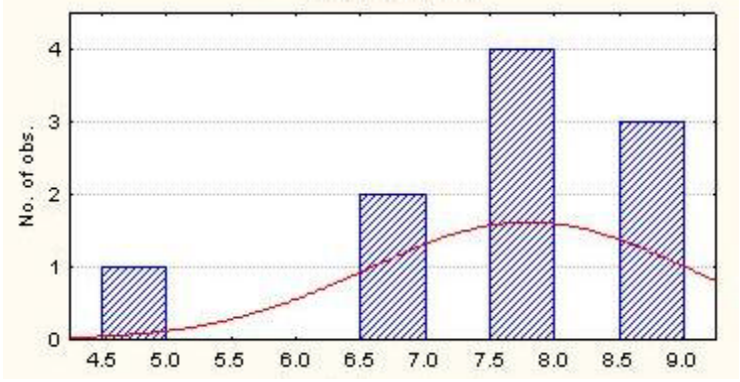

Figure 5.1 Graph result for system operation

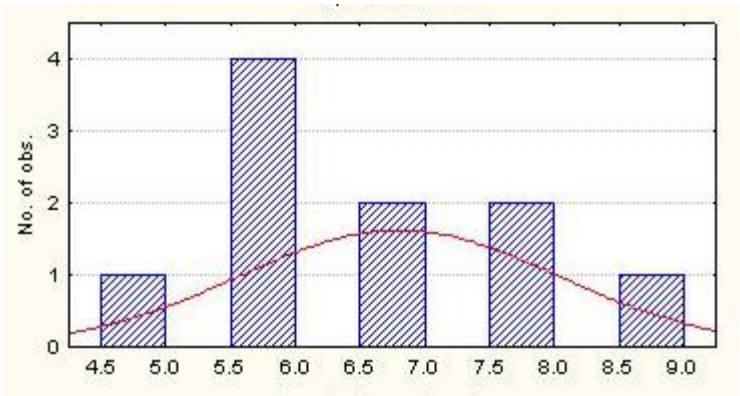

Figure 5.2 Graph result for Getting to use the system

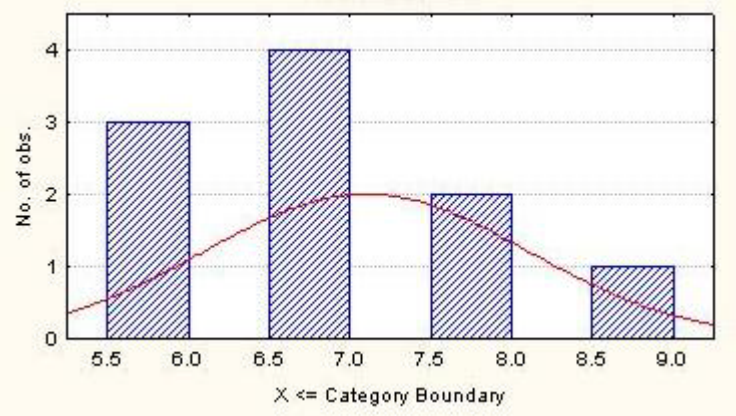

Figure 5.3 Graph result for Straight Forwardness of Task performance

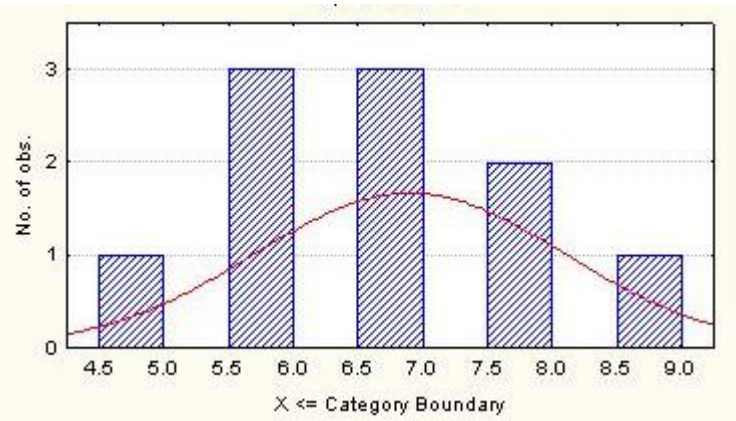

Figure 5.4 Graph result for Task performance on MTN system

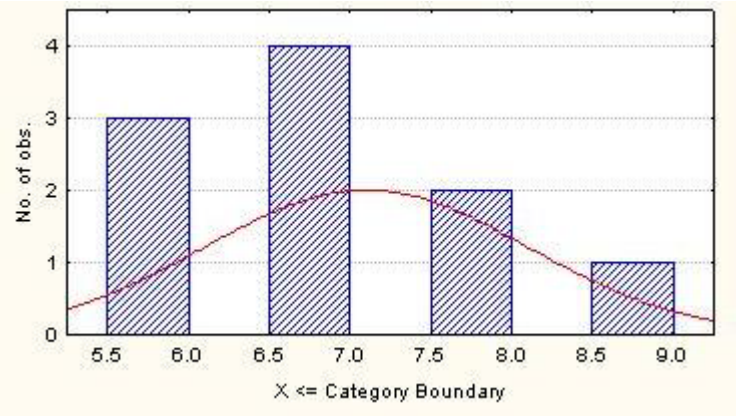

Figure 5.5 Graph Result for Number of steps per Task

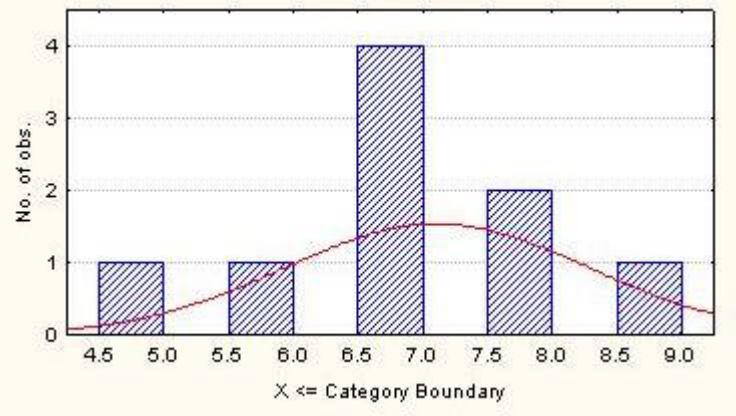

Figure 5.6 Graph Result for logical Sequence of Task

5.8.2 Learning to use the system

Figure 5.1 through Figure 5.6 show the graph of the result captured and analysed from the questionnaires that were filled out by the subject in response to the questions that were asked under the category question learning to operate the system. The responses from the subjects were subjected to descriptive analysis. The result of our observation coupled with the users response from the questionnaire shows that 78% of the subjects find it simple to quickly learn how to operate the system while 68% of the subjects got started with the system quickly. It was observed that only one of the users found it a little difficult to get the scope of the system at the beginning. This is because the user was an undergraduate student and the level of familiarisation and exposure to XML was low. A further interaction with this user showed that the evaluation study was an opportunity to get acquainted with XML. The result from the users' response show that the time to learn and operate the system was veryquick.

5.8.3 System Capability

Figure 5.7 through Figure 5.11 show the results that were captured and analysed from the questionnaires that were filled out by the subjects in response to the questions under the category System Capability. The results from the graph in figure 5.8 shows that 78% of the subjects stated that the system was very fast; it took less than 10 seconds to preprocess an application for 15 different mobile devices.

The result from figure 5.7 also shows that 73% of the subjects agreed that the speed of operation of the systems was very fast. The results from figure 5.10 shows that 76% of the subjects confirmed that the system was reliable because when using the system, no error was encountered. This is because the errors have been pointed during the pilot study and these have been fixed.

However, figure 5.11 shows that 75% of the subjects agreed that the ease of operating the system depends on the level of experience that a subject has in programming Java mobile application.

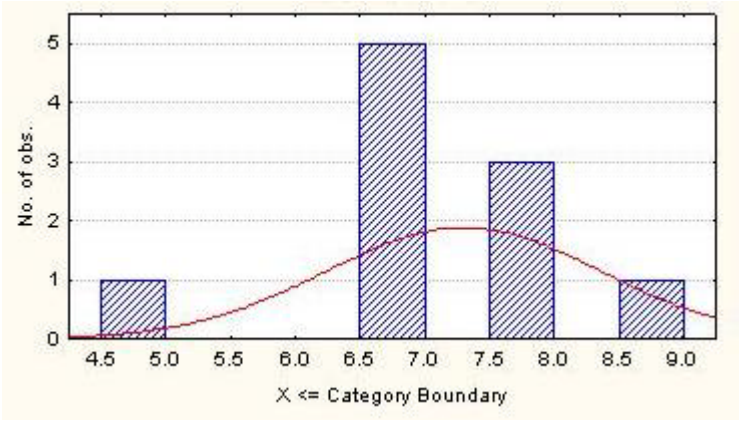

Figure 5.7 Graph result for the system speed

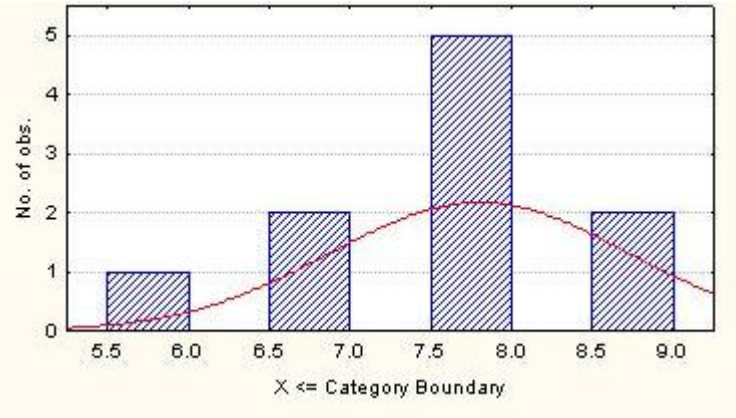

Figure 5.8 Gaph result for Response time to most operations

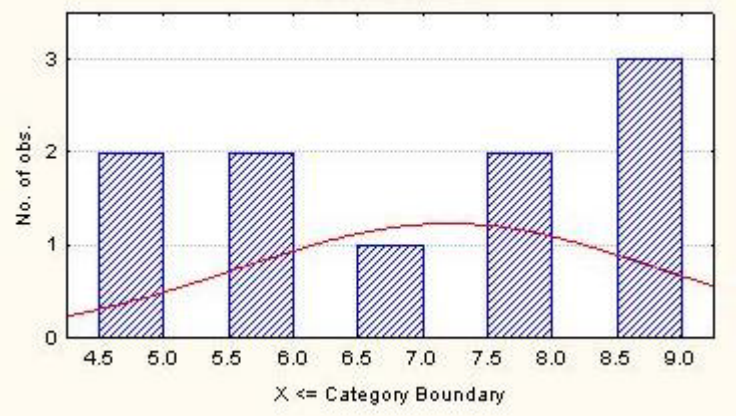

Figure 5.9 Graph result for the reliability of the system

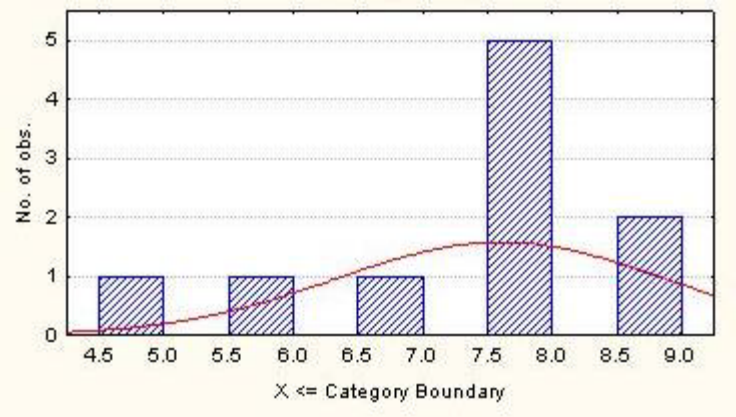

Figure 5.10 Graph Result for the dependability of the system

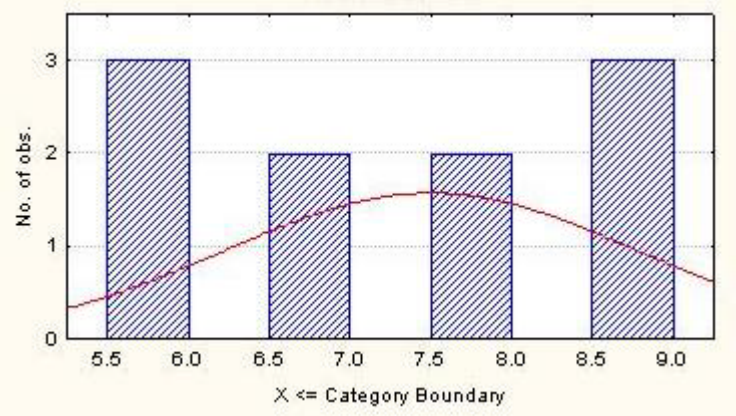

Figure 5.11 Graph result for ease of Operations

Figure 5.12 shows the user response to usability. All the users found the system satisfying. This is shown in Figure 5.12. Users liked the fact that little needed to be done when using the tool as they only need to perform some changes in the configuration file. This was further confirmed in the informal interviews conducted after the evaluation.

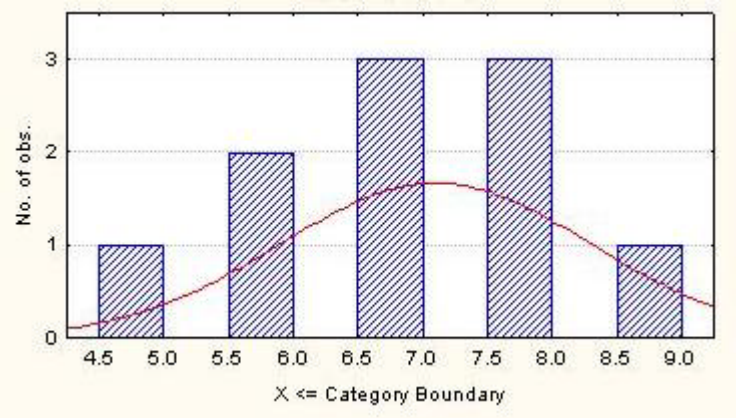

Figure 5.12 Graph result for Overall user reaction to the System

5.8.4 Overall Result and Discussion

Johnson (2008) argues that responsiveness is the is the most important factor in determining user satisfaction with a system.

Figure 5.12 also shows the user response to usability. All the users found the system satisfying.Users liked the fact that little needed to be done when using the tool as they only need to perform some changes in the configuration file. This was further confirmed in the unstructured interviews conducted after the evaluation.

However, our observation of the subjects shows that subjects were not able to perform these tasks within standard NetBeans IDE when the system that was developed was not plugged-in. Subjects are only able to write their mobile application code in the NetBeans IDE but they were not able to preprocess this within the IDE. This is an indication that we have been able to add functionalities that allow users to perform the same tasks in a more convenient way due to the fact that the original system does not support these activities and hence our hypothesis is considered to have been proven.

5.8.5 Revisiting the Hypotheses

The Hypotheses earlier stated in section 5.4 are as follows:

1. ***Users should be able to pre-process their developed mobile application to various mobile devices to suit their needs at once.***
2. ***After an initial training session, users should be able to adapt and configure the pre-processor without interference.***

For Hypothesis 1, the result of the evaluation indicated that user were able to preprocess the source codes for mobile applications to various mobile devices in order to meet up with the requirements of the devices using the MTN that was developed. This was however, not difficult to achieve.

For Hypothesis 2, the result of the evaluation indicated that 78% of the subjects found it simple to quickly learn how to operate the system while 68% of the subjects got started with the system quickly. This is an indication that the time to learn and operate the system was very quick.

Therefore, considering the discussion of these hypotheses, MTN answers the research question "***How we can support mobile developers through a Java IDE?"*** that was earlier asked in this thesis.: Based on this discussion, our hypotheses are considered to have been proven.

5.8.6 Other Consideration

The open nature of the NetBeans IDE enabled us to easily design the system with high quality integration. This is because in order to be able to design and develop the system, we needed access to the Netbeans source code and this was easily available. It was discovered that NetBeans can be seen as a platform itself which can be used to developed a plugin that can be added to it in order to improve its functionality which our research had achieved.

The nature of NetBeans is such that it is self-hosting, that is save for the core components of the NetBenas IDE – it is implemented as a series of plugins and this provided us with a wealth of facilities as well as examples from which to work.

5.9 Chapter Summary

In this Chapter, we have presented the evaluation study that was carried out in this research. We describe the pilot study and the final evaluation. The results gathered were analysed and presented. The results from the evaluation are adequate to show that users were satisfied with the system and this was the goal of the study.

CHAPTER SIX

CONCLUSION AND RECOMMENDATION

6.1 Introduction

The major goal of this research study was to establish how we can support mobile application developers through a Java IDE. NetBeans was the IDE of choice. This was because NetBeans was considered as the most widely used IDE for Java application development for mobile devices (Benson, *et al.*, 2004). A mobile tool call Mobile Tools for NetBeans (MTN) has been designed and developed to support our research. A set of different configuration descriptions for mobile devices was designed, implemented and were put together to form the MTN. We conducted the evaluation of the tools to establish whether the tool presented a more effective, efficient, satisfying solution than those currently available. The previous chapter presented the analysis and result of the usability evaluation that was conducted. This Chapter recaps and concludes the research study.

6.2 Reflection on Research Question

As stated earlier, this research was aimed at supportting mobile application developers through a Java IDE. However, in order to be able to bring this study into conclusion, it is imperative that the research question be revisited and also to briefly discuss the findings to the research question.

The major research question that was asked at the beginning of this study is how we can support mobile applications developers through a Java IDE. However, the approach that was chosen to investigate our research question followed a User-Centered Design

research which followed a well-established procedures in which the laying out of the research plans was properly conducted.

6.3 Reflections on the Research Findings

The data that were gathered from the survey, questionnaire and interview made it clear that in a typical development, porting and testing mobile applications takes a longer time than expected in order to accommodate the wide variety of mobile devices to be supported. This was then found to be against the expectations of mobile applications developers who expect J2ME applications will run correctly on all J2ME-enabled software and hardware platforms (e.g. J2ME-enabled mobile phones). Also a finding from our research made it clear that almost all Java mobile applications developers develop mobile applications through NetBeans, not only because it is free but because it is an open source development environment which has attracted many developers around the globe and thereby having a larger community of mobile applications developers.

Due to the above, our research focussed on better supporting mobile applications developers in creating mobile applications for a variety of mobile platform that are J2ME-enabled and this we were able to achieve through the NetBeans IDE.

6.4 Response to the Research Question

As a way of synthesising the results from the findings together, the Mobile Tool for NetBeans (MTN) has been developed. MTN is the response to the research question that was asked in this study.

Since NetBeans IDE was purely written in Java for the development of Java mobile applications and other Java applications (such as J2SE and J2EE etc.), MTN consist of different Java classes put together to interact and communicate with set of XML files and datasets, all of which have been presented and discussed in chpater four of this thesis. These were done in order to better support mobile applications developers in creating mobile content for variety of mobile platforms.

6.5 Reflection on MTN Evaluation

Jones and Marsden (2005) argued that designers may not know how useful their system is until an evaluation of that system has been carried out. However, Soroker, *et al.*, (2006), argued that determining how and where to improve the environment for application developers working on mobile applications requires conducting evaluation of the system to support them. Due to these facts we conducted an evaluationof MTN as we consider this as an important stage in the research process, since it allowed us the opportunity to iterate on the design of the system with feedback from the users. However, a task-based evaluation of the system was conducted in which users were required to use the MTN to perform so useful tasks in order to be able to gather useful feedback from the users. The results from the evaluation however showed that users seemed to enjoy using the system.

6.6 Research Contribution

The main contribution of this research is in the fields of both HCI and software engineering, specifically providing novel support for mobile developers by integrating a configuration pre-processor into the NetBeans IDE which allows mobile application developers to organise their code more easily for multiple devices. However, rather than focussing on the user interface issues, we proposed a support idea that will allow mobile applications developers to better develop mobile applications that will be portable across various J2ME-enabled platforms. The result of this research study has shown that there are still more ways in which mobile application developers can be supported.

6.7 Recommendation

6.7.1 How Java Should improve the experience of mobile developer

"One of the original motivation for creating Java was to create a programming language where compiled source code could run on any operating system" (Kirk, 2003). However, some issues exist with Java running on mobile devices. These issues include the need to reduce the number of supported Java libraries in mobile applications. For Java mobile applications to work correctly on mobile devices, the source code is first compiled into intermediate byte-codes, which are then interpreted at run time by a platform specific Java Virtual Machine (JVM). This means that a JVM must be available for a mobile device before this code can be run. Considering a situation where a mobile device does not have a JVM installed on it, this aspect of Java for mobile application development needs to be improved.

Also, from the experience of using Java programming languages to develop applications, it is gathered that Java is a strongly typed programming language that prevents programmers from creating new data types not found in Java and which might not be anticipated by the developers of Java. Therefore, Java should be improved in a way that new data types can be created by the application developers.

Lastly, an improvement should be made to Java in such a way that there should be collaborations between developers of mobile application using Java programming language and other programming languages.

6.8 Future Work

The research carried out during this dissertation was used to establish a proof of concept about supporting mobile developers through a particular IDE. Several limitations of this study warrants response. Some of this limitations include:

- Supporting mobile Developers through a Stand-Alone Mobile application development environment. This should have a rich Graphical User Interface (GUI).
- Supporting mobile development through a mobile development environment that will help mobile developers develop mobile applications that will run on all mobile devices at once without any modification of any form to the written code.

6.9 Concluding Remarks

This research that was conducted and presented in this dissertation provides support for mobile application developers through a Java IDE. It is our belief that more researchers should use the ideas presented in this dissertation.

REFERENCES

Abrahamsson, P., Hanhineva, A., Hulkko, H., Ihme, T., Jäälinoja, J., Korkale, M., Koskela, J., Kyllönen, P., and Salo, O. (2004). Mobile – D: An Agile Approach for mobile Application Development. . *Proceedings of the 19th annual ACM SIGPLAN conference on Object – Oriented Programming, Systems, Languages and Applications OOPSLA '04". Volume 39*, pp. 174-175. Vancouver, British Columbia, Canada.: ACM.

Abras, C., Maloney-Krichmar, D., and Preece, J. (2004). User-Centered Design. *In Bainbridge (Ed), W. Encyclopedia of Human-Computer Interaction. Thousand Oaks* , Sage Publications.

Agrawal, P., and Famolari, D. (1999). Mobile Computing in the next generation Wireless Wireless. *Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications* (pp. 32-39). Seattle, Washinton: ACM Press, New York.

Alba M & Favela, J. (2000). Supporting handheld collaboration through COMAL. *Proceedings of 6th International Workshop on Groupware* (pp. 52-59). Isla Madeira, Portugal: IEEE Computer Society.

Al-Qaimari, G. and McRostie, D. (2001). KALDI: A Computer Aided Userbility Engineering tool for supporting testing and analysis of user interaction. In A. Blandford, J. Vanderdonckt and P. Gray (eds), People and Computers XV – Interaction Without Frontiers:. *Joint proceedings of HCI 2001 and IHM 2001*, (pp. 153-169).

Bell, College. (2001). *Human Computer Interaction: 17. Prototyping*. Retrieved June 6, 2007, from http://hamilton.bell.ac.uk/btech/hci/hcinotes17.pdf

Berenson, M.L. and Levine, D.M. (1979). *Basic business statistics: Concepts and.* New Jersey: Prentice Hall.

Beyer, H., and Holtzblatt, V. K. (1993). Making Customer-Centered Design Work For Teams. *Communications of the ACM. Vol 36. No 10*, pp. 93-103. ACM.

Brad, A. (1992). Creating User Interface Software. In Languages for developing user interfaces (eds). USA: Jones and Bartlett.

Benson, C., Muller-Prove, M., and Mzourek, J. (2004). Professional usability in open source projects: GNOME, OpenOffice.org, Netbeans. *Proceedings of the ACM CHI|ACM Overview* (pp. 1083 – 1084). Vienna, Austria: ACM.

Cheung A., Grandison, T., Johnson, C., and Schonauer, S. (2007). Ïnfïnïty: A Generic Platform for Application Development and Information Sharing on Mobile Devices. *Proceedings of MobiDE '07* (pp. 25-31). Beijing, China: ACM.

Chen, G., Kandemir, M., Vijaykrishman, N., Irwin, M.J., Mathiske, B., and Wolczko, M. (2003). Heap Compression Memory - Constrained Java Environment. *Proceedings of the 18th annual ACM SIGPLAN conference on Object – Oriented Programming, Systems, Languages and Applications OOPSLA '03. 38*, pp. 282-301. California: ACM.

Chin J.P., Diehl V.A.., and Norman K. L. (1988). Development of an instrument measuring user satisfaction of the human-computer interface. *CHI '88: Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 213 - 218). Washington, D.C., USA : ACM.

Coen, A., Dai, L., Herzig, S., Gaul, S., and Linn, V. (2002). *The Analysis: Investigation of the cellular phone industry*. Retrieved October 21, 2006, from http://web.syr.edu/~spgaul/Final%20Essay%204.1.doc

Cooper, M. (2001). Internet: A Life-Changing Experience. *IEEE MultiMedia , vol. 08, no. 2*, 11-15.

Cox K, Walker D. What makes a good computer system? In: Cox K, Walker D, (eds.). User-interface design. Singapore: Prentice Hall; 1993. p. 1-32

Davies, N., Friday, A., Wade, S. P., and Blair, G. S. (1998). L2imbo: A distributed systems platform for mobile computing. *Journal of Mobile Networks and Applications , Volume 3* (Issue 2), 143 – 156.

Dillon, A. (2001). *The evaluation of software usability . In W. Karwowski (Ed), Encyclopedia of Human Factors and Ergonomics.* London:: Taylor and Francis.

Dowdy, S. and Weardon, S. (1983). *Statistics for research.* New York: John Wiley.

Dumas, J.S. and Redish, J.C. (1999). *A practical guide to usability testing.* Great Britain:: Cromwell Press.

Eclipse RCP. (2007). Retrieved October 14, 2007, from EClipse Rich Client [24] Platform: http://www.eclipse.org/rcp.

Faulkner, C. (1998). *The Essence of Human Computer Interaction.* Great Britain:: Prentice-Hall.

Faulkner, X. (2000). *Usability engineering.* London: Macmillan Press.

Giacoppo, S. A. (2001). *The role of theory in HCI. CHARM: Choosing human-computer*. Retrieved February 18, 2007, from University of Maryland: http://www.otal.umd.edu/hci-rm/

Golub, E., Bederson, Ben and Greenberg, S. (2001). *User Centered Design and Prototyping*. Retrieved September 7, 2007, from http://www.cs.umd.edu/class/fall2002/cmsc434-0201/notes4.pdf

GPRS Review. (2006). *GPRS (General Packet Radio Service), HSCSD & EDGE*. Retrieved May 18, 2007, from What is GPRS: http://www.mobile-phones-uk.org.uk/gprs.htm

Hansel T. R., Eriksson, E., Adreas, L., (2005). Mixed interaction space – Expanding the interaction space with mobile devices. *Proceedings of the HCI 2005, Edinburg, Scotland* (pp. 365 – 380). London: Springer.

Hasik, L. (2006, Sptember 1). *Sun OpenSourced Netbeans Mobility Pack*. Retrieved March 19, 2007, from (http://www.javalobby.org/java/forums/t78085.html).

Holtblazz, K., Wendell, J. B., and Wood, S. (2005). *Rapid Contextual Design* . San Francisco: Morgan Kaufmann.

Holtzblatt, K, and Jones, S. (1993). *Contextual Inqury:principles and Practice, In Praciticipatory Design: Principles and Practices.* New York: Lawrence Earlbaum.

Holtzblatt, K. and Beyer, H. (1997). Contextual Design: Using Customer Work Models to Drive Systems Design. *CHI '97 extended abstracts on Human factors in computing systems: looking to the future* (pp. 184-185). Atlanta, USA: ACM.

Jaap, van Ekris . (2006, April 24). *What is in A Good Mobile Application Anyway?* Retrieved June 26, 2007, from Modern Nomads: Make Mobile Devices Work For You: http://modernnomads.info/articles/read.php?article_id=5

Johnson U. O. (1999). *Introduction to Project Writing for Business and Financial Studies.* Enugu, Nigeria: Sunny Enterprises.

Jones M. and Marden G. (2005). *Mobile Interaction Design* (1st edition ed.). England: Wiley.

Joseph, A.D., Tauber, J. A., and Kaashoek, M. F., (1997). Mobile computing with Rover toolkit. *IEEE Transactions on computers , Vol. 46* (3), 337-352.

Kuter, U. and Yilmaz, C. (2001). *Survey methods: Questionnaires and interviews.* Retrieved September 13, 2007, from CHARM Survey Methods: http://www.otal.umd.edu/hci-rm/survey.html

Kantner, L., Sova D. H., and Rosenbaum, S. (2003). Alternative Methods for Field Usability research. *SIGDOC* (pp. 68 - 72). San Francisco: ACM.

Lee, Y. S., Ryu, Y. S., Shin, D. J., Nussbaum, M. A., and Tomioka, K. (2005). *Usability Testing with Cultural Groups in Developing a Cell Phone Navigation System.* Retrieved October 16, 2007, from http://uweb.txstate.edu/~yr12/Papers/HCII2005_Submission_Cultural.pdf

Litiu, R., and Prakash, A. (2000). Developing adaptive groupware applications using a mobile component framework. *Proceedings of the 2000 ACM conference on Computer Supported Cooperative Work* (pp. 107-116). Philadelphia, PA, USA: ACM.

Lukkari, J., Korhonen, J., and Ojala, T., (2004). SmartRestaurant – Mobile Payment in Context – Aware Environment. *Proceedings of the Sixth International Conference on Electronic Commerce. ICEC'04* (pp. 575 – 582). New York, USA: ACM.

Mace, M. (2006). *Mobile Opportunity*. Retrieved July 20, 2007, from http://mobileopportunity.blogspot.com/2006/05/flash-versus-windows-can-adobe-break.html

Mobile Review . (2003). *Classification of cellular*. Retrieved December 19, 2006, from http://www.mobile-review.com/articles/2003/classification2-en.shtml

Mohageg, M. and Bergman, E. . (2000). Information Appliance Design at Sun Microsystems. . *Proceedings of the CHI '00, Extended abstract on Human factors in computing systems* (pp. 63 – 64). Hagues, The Netherlands: ACM presss.

Mouton, J. (2001). *How to succeed in your masters and doctoral theses.* Pretoria, South Africa: Van Schaik.

Munson, J.P. and Dewan P. (1997). Sync: a Java framework for mobile collaborative applications. *IEEE Computer Society , 30* (6), 59-66.

Netbeans, W. (n.d.). *Netbeans IDE features*. Retrieved December 27, 2006, from http://www.netbeans.org/features/platform/index.html

Nielsen, C. M., Overgaard, M., Pedersen, J., and Stage, J. (2005). Feedback from Usability Evaluation to User Interface Design. In *Human Computer Interaction - INTERACT 2005* (pp. 391-404). Berlin, Germany: Springer.

Nielsen, J. and Phillips, V.L. (1993). Estimating the relative usability of two interfaces: Heuristic, formal, and empirical methods compared. *Proceedings of the SIGCHI conference on Human factors in computing systems.* (pp. 214- 221.). Amsterdam, The: ACM.

Nielsen, J. (1996). *International usability testing. In E. del Galdo and J. Nielsen (Eds), International user interfaces.* New York: John Wiley and Sons.

Nielsen, J. (2003., January,). *Recruiting test participants for usability studies*. Retrieved June 16, 2007, from Alertbox,: http://www.useit.com/alertbox/20030120.html.

Nielsen, J. (1992). The usability engineering life cycle. *Journal of IEEE Computer Society , 25, Issue 3*.

Nielsen, J. (2003, August). *Usability 101:*. Retrieved February 21, 2007, from Introduction to usability. Alertbox: http://www.useit.com/alertbox/20030825.html.

Nivala, A. M., Sarjakoski, L. T., and Sarjakoski, T., (2005). User-Centred Design and Development of a Mobile Map Service. *Proceedings, ScanGIS'2005* (pp. 109-123). Stockholm, Sweden: Department of Planning and Environment, Sweden.

NTT DoCoMo. (2007). *http://www.nttdocomo.com/*. Retrieved September 18, 2007, from http://www.nttdocomo.com/

Olivier, M. S. (2004). *Information technology research: A practical guide for Computer* (2nd Edition ed.). Pretoria, South Africa: Van Schaik.

Picco, G. P., Murphy, A. L., and Roman, G. (2000). Developing Mobile Computing Applications with Lime. *Proceedings of International Conference on Software Engineering* (pp. 766-769). ACM.

Preece, J., Rogers, Y., and Sharp, H. (2007). *Interaction Design – Beyond Human – Computer Interaction.* West Sussex: John Wiley & Sons Ltd.

Preece, J., Rogers, Y., and Sharp, H. Benyon, D., Holland, S., and Carey, T. (1994). *Human-Computer Interaction.* Wokingham, UK: Addison-Wesley.

Preece, J., Rogers, Y., and Sharp, H. (2002). *Interaction Design – Beyond Human – Computer Interaction* (1st edition ed.). West Sussex, England: John Wiley & Sons Ltd.

Robert, V. (2005). *Pro J2ME Polish: Open Source Wireless Java Tools Suite* (1st Edition ed.). USA: Apress.

Rowan, S. (2006, September). *The UCD Process*. Retrieved November 07, 2007, from http://www.upa.org.nz/wp-content/uploads/2006/09/upanz-govis-sept-2006.pdf

Sandoval, G. L., Chávez, E. E., Caballero, J. C. P., (2004). *A development platform and execution environment for mobile applications*. Retrieved November 15, 2006, from http://www.clei.cl/cleiej/papers/v7i1p4.pdf

Schill A., and Kummel, S. (1995). Design and implementation of a support platform for distributed mobile computing. *Proceedings of the Distributed System Engineering 2* (pp. 128-141). United Kingdom: BCS, IEE and IOP publishing Ltd.

Scholtz, J. (2004). *Usability Evaluation. Publication No. 545.* National Institute of National Institute of Standards and Technology Retrieved October 19, 2007, from http://www.itl.nist.gov/iad/IADpapers/2004/Usability%20Evaluation_rev1.pdf

Sears, A., and Jacko, J., A. (2007). *The Human-computer Interaction Handbook: Fundamentals, Evolving Technologies.* New York, USA: CRC Press.

Shneiderman, B. (1987). Designing The User Interface Strategies For Effective Human-Computer Interaction. *ACM SIGBIO Newsletter , 9* (1), 6.

Shneiderman, B. (1998). *Designing the user interface: Strategies for effective humancomputer* (3rd Edition ed.). Massachusetts, USA: Addison-Wesley.

Soroka, D., Cáceres, R., Dig, D., Schade, A., Spraragen, S., and Tiwari, A. (2006). Pegboard:: A framwork for developing mobile applications. *Proceedings of the MobiSys '06* (pp. 138-150). Uppsala, Sweden: ACM.

Spool, J. and Schroeder, W. (2001). Testing web sites: Five users is nowhere near enough. *Extended abstracts of CHI 2001* (pp. 285-286). Seattle, Washington : ACM.

Stone, D., Jarrett, C., Woodroffe, M., and Minocha, S. (2005). *User Interface Design and Evaluation.* San Francisco: Mourgan Koufhman.

Straus, R. A. (2001). *Using Sociology: An Introduction from the Applied and Clinical.* Rowman & Littlefield.

Sun Microsystems, (2001). *The Netbeans Tools Platform: A Technical Overview.* USA: Sun Micorsystesm Inc.

Sun Microsystems, 2. (2003). *Java 2 Micro Edition (J2ME).* Retrieved June 14, 2006, from http://java.sun.com/j2me/

Thomas, K. (1999). Designing a Task-Based Evaluation Methodology for a Spoken Machine Translation System. *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics* (pp. 569-572). Maryland, USA: Association for Computational Linguistics .

Thomas, S., J. (1999). *Designing Survey that Works: A Step-by-Step Guide* (1st edition ed.). California: Corwin Press Inc.

Topley, K. (2002). *J2ME in a Nutshell: A Desktop Quick Reference* (1st ed.). USA: O' Reilly.

Trochim, W. (2002). *Research Methods Knowledgebase*. Retrieved December 9, 2006, from http://trochim.human.cornell.edu/kb/

Unknown, author. (2004). *Characteristics of Mobile Applications*. Retrieved July 22, 2007, from http://www.informit.com/content/images/0321269314/samplechapter/salmre_ch02.pdf

Van Biljon, J. A. (2006). *A models for representing the motivational factors that influence mobile phone usage variety.* Pretoria: Department of Computer Science, University of South Africa.

Weilling, G. S. (1999). *Designing Adaptive Environment – Aware Applications for Mobile Computing.* PhD. Thesis, The State University of New Jersey, Graduate School in Computer Science, New Brunswick, New Jersey.

Wesson, J. L., and Van der walt, D. F. (2005). Implementing Mobile Devices: Does the platform really make a difference. *Proceedings of the SAICSIT 2005* (pp. 208-216). ACM.

Weyert de Boer, Janousek, S., and Leggett, R., (2006). *Foundation Flash for Mobile Devices* (1st ed.). USA: Apress.

White, J. (2001). An Introduction to Java 2 Micro Edition (J2ME); Java in Small Things. *Proceedings of the 23rd International Conference on Software Engineering (ICSE'01)* (pp. 724-725). Toronto, Canada: IEEE.

Wilson, A. (2006). *Write Once, Deply Anywhere*. Retrieved December 27, 2006, from

Writing Applications for mobile devices and reducing device fragmentation with NetBeans Mobility Pack: http://www.netbeans.org/community/magazine/html/03/mobpack/

Wilson, D. R., Ramey, J., Holtzblatt, K., Beyer, H., Hackos, J., Rosenbaum, S., Page, C., Laakso, S. A., and Laakso, K. (2002). Usability in Practice: Field Methods Evolution and Revolution. *CHI 2002* (pp. 880 - 884). Minnesota USA: ACM.

Young, T., J. (2005). *Using Aspect J to build software product line for mobile devices: Masters Thesis.* University of British Columbia. Vancouver: Department of Computer Science.

Yen, D.C. and Chou, D.C. (2000). Wireless communications: Applications and Managerial Issue. *Industrial management and Data Systems* , 436-443.

Young, T. (2003). *Research Paper: Software Interface Design for Small Devices.* Vancouver, Canada: Software Practices Laboratory, University of British Columbia.

Zhang, P., Plettenberg, L., Klavans, J. L., Oard, D. W., Soregel, D. (2007). Task-based Interaction with an Integrated Multilingual, Multimedia Information System: A Formative Evaluation. *Proceedings of Joint Conference on Digital Libraries JCDL '07* (pp. 117-126). Vancouver, British Columbia, Canada.: ACM.

Zhao, X. (2003). *A Comparative Analysis of Java and.NET Mobile Development Environments for Supporting Mobile Services.* Rhodes University, Computer Science. Grahams town: Rhodes University.

Zimmerman D.E and Muraski. (1995). *The Elements of Information Gathering Phoenix.* Arizona USA.: The Oryx Press.

APPENDIX A

QUESTIONNAIRE FOR USER EVALUATION

Subject Name: _____

Age: _____

Gender: Male / Female **(Please circle the appropriate one)**

1. System Experience

How long did it take it to work on the system?

- Less than 1 hour
- 1 hour to less than 2 hours
- 2 hours and above

2. Overall user reactions

Please circle the numbers which most appropriately reflect your impressions about using this application. Not Applicable = NA.

2.1 Overall reactions to the system:	terrible 1 2 3 4 5 6 7 8 9 wonderful	NA
2.2	frustrating 1 2 3 4 5 6 7 8 9 satisfying	NA
2.3	dull 1 2 3 4 5 6 7 8 9 stimulating	NA
2.4	difficult 1 2 3 4 5 6 7 8 9 easy	NA
2.5	rigid 1 2 3 4 5 6 7 8 9 flexible	NA

Please write other comments you may have about using the application here:

_____

_____

_____

_____

3. Learning

Please circle the numbers which most appropriately reflect your impressions about using this application. Not Applicable = NA.

3.1 Learning to operate the system — difficult / easy
1 2 3 4 5 6 7 8 9 NA

3.1.1 Getting started — difficult / easy
1 2 3 4 5 6 7 8 9 NA

3.1.2 Time to learn to use the system — difficult / easy
1 2 3 4 5 6 7 8 9 NA

3.2 Tasks can be performed in a straight-fo manner — nev / always
1 2 3 4 5 6 7 8 9 NA

3.2.1 Number of steps per task — too many / just right
1 2 3 4 5 6 7 8 9 NA

3.2.2 Steps to complete a task follov logical sequence — never / always
1 2 3 4 5 6 7 8 9 NA

Please write other comments you may have about learning here:

_____

_____

_____

_____

4. System Capabilities

Please circle the numbers which most appropriately reflect your impressions about using this application. Not Applicable = NA.

4.1 System speed — too slow / fast enough
1 2 3 4 5 6 7 8 9 NA

4.1.1 Response time for most operatio — too slow / fast enough
1 2 3 4 5 6 7 8 9 NA

4.2 The system is reliable — never / always
1 2 3 4 5 6 7 8 9 NA

4.2.1 Operations are — undependable / dependable

		1 2 3 4 5 6 7 8 9	NA
4.2.2	Ease of operation depends on your level of experience	never always	
		1 2 3 4 5 6 7 8 9	NA

Please write your comments about system capabilities here:

_____

_____

_____

5. General

If this system were to be redesigned, please tell us what you would like to see added or change in the system.

_____

_____

_____

Consent Form for evaluation

Subject's Name_____

Email _____

Phone (optional)_____

I agree that I have been provided with the necessary information about the new pre-processor application and the procedure for the completion of its evaluation. I therefore give my consent to take part in its evaluation study.

I also understand how my data will be handled as it has been explained to me and no notes or log will contain any personal information without my consent to the agreement below. Any video and/or audio data that may be collected during the evaluation will only be viewed by the evaluator alone.

I understand that I can withdraw from this evaluation study at any point I desire to do so without prejudice or penalty of any kind.

I will/will not (please delete inappropriate one) be happy if the department of Computer Science of the University of Cape Town uses excerpt of video from my session for educational purposes and no excerpt will be shown that could be construed as unflattering or embarrassing for me.

Signature _____ **Date** _____

Evaluator's Name_____

Evaluator's Signature_____ **Date**_____

APPENDIX B

Preprocessing Directives	Description	Examples
//#ifdefine	Used to check to see if a signle preprocessing symbol is defined	//#ifdefine mtn.api.mmapi
//#ifnotdefine	Used to check to see if a single preprocessing symbol is not defined	//#ifnotdefine mtn.api.mmapi
//#elseifdefine	Used for branching to check for any other single preprocessor	//#elseifdefine mtn.midp2
//#elseifnotdefine	Used for branching and check whether another single preprocessor symbol is not defined	//#elseifnotdefine mtn.midp2
//#else	Used for branching in the //#ifdefine directive	//#else
//#endifdefine	Used to end an //#ifdefine block	//#endifdefine
//#inex	Used to include or exclude file	//#inex mtn.gui || mtn.mp3
//#def	Used to define a temporary preprocessing symbol	//#def temp.useAudio
//#undef	Used to remove a defined temporary preprocessing symbol	//#undef temp.useAudio
//#css	Used to define or assign css to item or screen	//#css text

Table 1.0: Full list of the Preprocessing Directives

APPENDIX C

Source Code (Complete XML File for the Build tool)

```
<project
     name="MTN-Sample"
     default="mtn">

<!—Specify the wtk.home property directory -->
<!-- containing the Wireless Toolkit -->
<property name="wtk.home" value="C:\WTK25" />
<!— Specify the home directory of the mtn -->
<property name="mtn.home" value="C:\Program Files\MTN" />

<!-- Specify the home directory for the devices emulators.
Nokia, Siemens, Samsung, sony-ericsson and Motorola
emulators will be used for the purpose of this sample -->

<!—Specify the directory which contains the Siemens-
emulator

     <property name="siemens.home" value="C:\siemens" />
-->
<!—Specify the directory which contains the Nokia-emulators

     <property name="nokia.home" value="C:\Nokia" />
-->
<!—Specify the directory which contains the  Sony-Ericsson
SDK
     <property name="sony-ericsson.home"
value="C:\SonyEricsson\SE_SDK" />
-->

<!-- The directory which contains the Motorola-emulators
     <property name="motorola.home" value="C:\Program
Files\Motorola\SDK v4.4" />
-->


<!—Specify the directory which contains the  Samsung SDK
     <property name="samsung.home"
value="C:\Samsung\J2ME_SDK" />
-->

<!-- Definition of the MTN task:                        -->
<taskdef name="mtn"
     classname="org.netbeans.mtn.ant.mtnTask"
```

```
     classpath="${mtn.home}/import/mtn-
build.jar:${mtn.home}/import/jdom.jar:${mtn.home}/import/pr
oguard.jar:${mtn.home}/yguard-
lib.jar:${wtk.home}/wtklib/kenv.zip"/>
<!-- build targets, each target can be called via "ant
[name]",
      e.g. "ant clean", "ant test mtn" or just "ant" for
calling the default-target -->

<target name="test"
     description="This target is called to skip first
obfuscation step"
     >
     <property name="test" value="true" />
     <property name="dir.work" value="build/testing" />
</target>

<target name="init">
     <property name="test" value="false" />
     <property name="dir.work" value="build/realtest" />
     <property name="deploy-url" value="" />
</target>

<!-- In this target the mtn task from ant is used.
-->
<!-- It has 2 sections:
-->
<!--    2. The deviceRequirements-section chooses the
devices -->
<!-- for which the application is target and optimized -->
<!--    3. The build-section controls the actual build of
the system -->
<!--
                                                 -->
<target name="mtn"
          depends="init"
          description="This is the controller for the mtn
build tool."
          >
     <mtn>
         <!—The general settings, will basically form the
JAD-attributes of the system. -->


<!-- selection of supported devices -->

<devicePrerequisites if="test">
```

```
               <Prerequisite name="Identifier"
value="Generic/midp1" />
          </ devicePrerequisites>

          < devicePrerequisites ifnot ="test">
               <Prerequisite name="Identifier" value="Sony-
Ericsson/P900, Samsung/DSeries, Nokia/Series60Midp2,
Generic/midp2, Generic/midp1" />
               <!-- or could use other devices for real
builds, e.g. :
               <or>
                    <and>
                         <Prerequisite name="JavaPackage"
value="samsung-ui" />
                         <Prerequisite name="BitsPerPixel"
value="16+" />
                    </and>
               </or>
                -->
          </devicePrerequisites>

         <!-- build settings -->
          <!--
           -->
          <build
               symbols="SampleSymbol, additionalExample"
               fullscreen="menu"
               workDir="${dir.work}" >
               <!-- definition of the midlets -->
               <midlet class="
org.netbeans.mtn.sample.sampleMenu" name="sampleMenu" />
               <!-- project-wide variables - used for
preprocessing.  -->
               <!-- You can set localized variables in the
resources/localized.txt files provided.  -->
               <variables includeAntProperties="true" >
                    <!-- example variables:
               the following snipet use this variable:
                    //#= private static final String in the
Java-code -->
          <!-- You can set the dir attribute to "another
sources" in order to allow alternative design. -->
               <sources
                    dir="sources"
                    defaultexcludes="yes"
                    >
```

```
                    <!-- you can add other files depending
the criterias:
                    <filesource
                         dir="sources/multimedia"
                         includes="*.wav"
                         if="mtn.audio.wav"
                    />
                    < filesource
                         dir="sources/multimedia"
                         includes="*.mid"
                         if="mtn.audio.midi and not
mtn.audio.wav"
                    />
                    -->
                    <!-- Set the localization criteria to
create localized versions for your application: -->
                    <localization locales="af_AF, en_US"
ifnot="test" />
                    <localization locales="en_US", "en_UK"
if="test" />
               </sources>


               <!-- Define JAD attributes for the devices:
-->
               <jad>
                    <attribute name="Samsung-Category"
value="GameMenu" if="mtn.samsungGroup.DSeries" />
               </jad>

          </build>


          <!—Invoking the emulator(s) -->
          <emulator
               wait="true"
               trace="class"
               securityDomain="trusted"
               Profilerenable="true"
               MemoryMonitorenable="true"
               NetworkMonitorenable="true"
               if="test"
               >
               <!--
               <parameter name="-Xjam"
value="transient=http://localhost:8080/${mtn.jadName}" />
               -->
```

```
          </emulator>

     </mtn>
</target>

<target name="clean"
          description="This section calls a clean build.
(Ant Clean) should be called whenever changes are made to
mobiledevices collection xml, device manufacturer xml or
the deviceassembly xml">
     <delete dir="build" />
     <delete dir="dist" />
</target>

</project>
```

APPENDIX D

Source Codes (XML Files used to store datasets of devices as well as the requirements For Mobile Device Collection)

```
<mobileCollection>
     <classifier>Generic/mmapi</classifier>
     <deviceassembly>Emulator</deviceassembly>
     <xtics name = “JPlatform”, Worth = “MIDP/2.0” />
     <xtics name = “JConfig”, Worth = “CLDC/1.1” />
     <xtics name = “JPackage”, Worth = “wmapi, mmapi” />
     <xtics name = “AudioSetup”, Worth = “mp3, Hifi, midi”
     />
</mobileCollection>
<mobileCollection>
     <classifier>Samsung/D900</classifier>
     <deviceassembly>DSeries</deviceassembly>
     <deviceXtics>withCam</deviceXtics>
     <xtics name = “JPlatform”, Worth = “MIDP/2.0” />
     <xtics name = “JConfig”, Worth = “CLDC/1.1” />
     <xtics name = “OpSystem”, Worth = “WindowsMobile 5.0,
     Symbian OS 7.0s” />
     <xtics name = “CamResolution”, Worth = “VGA” />
     <xtics name = “ZoomingSize”, Worth = “3” />
     <xtics name = “LCDSize”, Worth = “178X210” />
     <xtics name = “AudioSetup”, Worth = “amr, mp3, hifi,
     midi” />
     <xtics name = “memorySize”, Worth = “60mb” />
     <xtics name = “JarSize”, Worth = “dynamic” />
</mobileCollection>
<mobileCollection>
     <classifier>Motorola/V3</classifier>
     <deviceassembly>Razr</deviceassembly>
```

```
        <deviceXtics>withCam</deviceXtics>
        <xtics name = “JPlatform”, Worth = “MIDP/2.0” />
        <xtics name = “JConfig”, Worth = “CLDC/1.1” />
        <xtics name = “OpSystem”, Worth = “WindowsMobile 5.5,
        Symbian OS 7.0s” />
        <xtics name = “CamResolution”, Worth = “VGA” />
        <xtics name = “ZoomingSize”, Worth = “2” />
        <xtics name = “LCDSize”, Worth = “174X206” />
        <xtics name = “AudioSetup”, Worth = “mp3, hifi, midi”
        />
        <xtics name = “memorySize”, Worth = “30mb” />
        <xtics name = “JarSize”, Worth = “dynamic” />
</mobileCollection>
<mobileCollection>
        <classifier>SonyEricsson/P800,       SonyEricsson/P900,
        SonyEricsson/P900i</classifier>
        <deviceassembly>PSeries</deviceassembly>
        <deviceXtics>withCam, withCursor</deviceXtics>
        <xtics name = “JPlatform”, Worth = “MIDP/2.0” />
        <xtics name = “JConfig”, Worth = “CLDC/1.1” />
        <xtics name = “OpSystem”, Worth = “WindowsMobile 5.5,
        Symbian OS 7.1s” />
        <xtics name = “CamResolution”, Worth = “VGA” />
        <xtics name = “ZoomingSize”, Worth = “5” />
        <xtics name = “LCDSize”, Worth = “178X210” />
        <xtics name = “AudioSetup”, Worth = “mp3, hifi, midi”
        />
        <xtics name = “memorySize”, Worth = “512mb” />
        <xtics name = “JarSize”, Worth = “dynamic” />
</mobileCollection>
```

Source Codes (XML Files used to store datasets of manufacturers)

```
<deviceManufacturers>
         <manufacturer>
         <name>Motorola</name>
</manufacturer>
<manufacturer>
     <name>LG</name>
</manufacturer>
<manufacturer>
     <name>Samsung</name>
</manufacturer>
<manufacturer>
     <name>Sharp</name>
</manufacturer>
<manufacturer>
     <name>SonyEricsson</name>
</manufacturer>
<manufacturer>
     <name>Panasonic</name>
</manufacturer>
<manufacturer>
     <name>Siemens</name>
</manufacturer>
</deviceManufacturers>
```