Supporting Mobile Application Developer through Java IDE Using Contextual Inquiry

Olalekan Samuel Ogunleye
Meraka Institute, Council for Scientific and Industrial Research
Pretoria, South Africa

Gary Marsden
Department of Computer Science, University of Cape Town
Cape Town, South Africa

1 Introduction

Mobile Devices are becoming popular more and more with millions of users acquiring them every day. For instance, in china, there are more than 400 million mobile users and this number is increasing (Young, 2005). Also, the mobile device market in the United States is increasing at an annual rate of 22% (Chen *et al*, 2003). Devices such as mobile cellular phones, Personal Digital Assistants (PDAs), music players etc run Java software such as games and business enterprise applications (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) where mobile devices are used often for personal use and as commercial tool. This means that application 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).

However, Integrated Development Environments (IDEs) – such as Eclipse, Visual Studio, NetBeans, Borland JBuilder Enterprise with MobiSet 3, Sun Microsystems' Sun ONE (Open Network Environment) Studio 4 Mobile Edition, Metrowerks CodeWarrior Wireless Studio 7, S5 Systems' jVise (based on IBM Eclipse technology), etc. are tools of choice for developing mobile applications and they are also instrumental in developing individual components for mobile applications (Soroka *et al*, 2006). With the exception of Visual Studio, these IDEs are all based on Java. Developing mobile applications using any of these development environments is a complex task (Soroka *et al*, 2006). However, one vision of Java mobile applications developers is to deliver robust and comprehensive applications for various mobile devices that one can easily carry, through using one of the IDEs mentioned above.

The popularity of mobile applications and services are now such that this research feels it is time to look at how well mobile applications developers are supported through these existing development tools. This research uses Contextual Inquiry (CI) (Holtzblatt and Beyer, 1997) to investigate how mobile applications developers can be supported through Java IDEs in order to identify problems that are encountered when using Java IDEs to develop mobile applications. We also assess the utility of CI for extracting the design requirements for the IDEs.

NetBeans IDE was considered as the ideal IDE to use for this research. This is because it is an open source IDE and it is considered as the most widely used Java IDE for developing mobile applications for mobile devices (Benson *et al*, 2004). Therefore, this research was interested in finding ways to improve the usability of Java

IDEs for mobile applications development and to provide more support for Java mobile applications developers through the IDEs. The techniques of CI recommend observing activities as they occur in their natural context in order to be able to portray the process of the work as well as the discovery of the places where technology could be applied to defeat the observed difficulties (Cross and Warmack, 2000). This method was chosen because it would provide data about the detailed problems faced by Java mobile applications developers when using a Java IDE to develop mobile applications and it will also provide guidance on the design of the support framework (Jones and Marsden, 2005; Preece *et al*, 2007).

1.1 Contextual Inquiry (Observing Java Mobile Developers)

CI, as described by Beyer and Holtzblatt is a structured approach to the collection and understanding of data from fieldwork with the purpose of building a tool that supports the user of a system. It is a method that provides the researcher and/or designer with a grounded and detailed knowledge of users' work as a basis for their design (Wixon and Raven, 1994). This is usually achieved by fostering a strong relationship with the users. This will determine how well the researcher/designer understands the users in order to be able to support them. And users are always assumed as the expert in their work (Holtzblatt and Beyer, 1997; Jones and Marsden, 2005).

CI is always achieved through a face – to – face interaction using an apprenticeship model which provides an attitude of inquiry and learning while the users are being studied (Holtzblatt and Beyer, 1997; Jones and Marsden, 2005) and it defines a clear set of concerns rather than a list of specific questions which enables the researcher/designer to focus on a few key issues and gather concrete data during the session that they may have with the users. The importance of CI is that you can ask questions and prompt for explanations immediately (Holtzblatt and Beyer, 1997; Preece *et al*, 2007).

Goal	**Run code for various devices**
Context	Code was working fine but could not be adapted to different mobile devices without changing the code
Outcome	While trying to make sure that the code works fine on all the devices, there are various versions of codes of emanated from only one written codes.

Table 1: Table Showing the action that users wish to achieve using Java IDE to develop Mobile Application

2 Background and Literature Review

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 in growth rate from the year 2000, along with an explosion of mobile service adoption in Africa, America and Asia (Jones and Marsden, 2005).

However, most of the development of Java mobile applications takes place, not on the particular device itself but on a personal computer (PC). Therefore, 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).

2.1 Literature Review

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 Enterprise with MobiSet, Sun ONE (Open Network Environment) Studio 4 Mobile Edition, Metrowerks Code Warrior Wireless Studio 7, 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. To this end, 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.

Munson and 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 (Soroker *et al.*, 2006).

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.

3 Methodology

This research focuses on the study of how java 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. This means that, we wanted to observe and examine Java mobile applications 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-Centred 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.

3.1 A User Centre Design Approach

The importance of user-centred design is based on involving the user throughout the whole life of product design (Nivala, 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-Centred 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-centred design research methodology, what is of great consequence is that users are involved. Some types of user-centred 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-centred 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-centred design (UCD) is the encouragement of the entire system development procedure with user-centred 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-centred 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-centred 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). Fig. 1 below that was adapted from (Rowan, 2006), shows the flow of design process for user-centred design approach.

In order to conduct this research study, we therefore used User-Centered Design as a guiding principle and specific techniques such as Contextual Inquiry (Holtzblatt, 2005) were 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 Java mobile programmers encounter, Contextual Inquiry helps us to focus on observation and in-work interviews in order to extract users' requirements and be able to suggest a suitable solution to support Java mobile application developers.

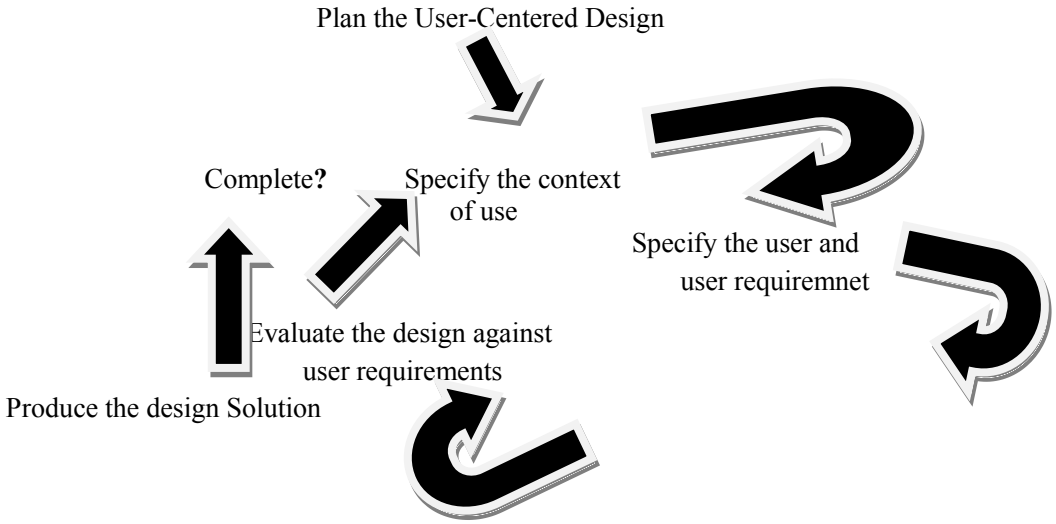

Figure 1: User-centered Interactive Systems.

3.2 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) explained 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 the controlled environments and
- Indirect observations 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 researcher/observers and the researcher/observer records what is going on.

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 but a device is powered 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 (1993) 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). Therefore, they 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 determines 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).

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).

Therefore, using CI, Sixty four mobile applications developers were observed in their place of work, (that is, the Computer Science department laboratory of the University of Cape Town), most of who were postgraduates' students. This was done during the first and second semester of the session. The researcher met with each mobile applications developer and explained the motivation behind CI – to identify programming difficulties that mobile applications developers experience while developing mobile applications using one IDE or the other in order to be able to provide support for these difficulties. This would be achieved by observing the developers as they develop their mobile applications using various Java IDEs. However, as developers develop their applications, the researcher recorded the observations on both the paper and video.

Hypothesis about the programmer actions were formed. This was later shown to the developers. For example, "You want to be able to port your applications to various mobile devices by using the functionalities provided by the IDE." and the developers would reply, "Actually, I want to be able to write one set of code and by using this IDE, I want my application to work on different mobile devices without changing the codes." This is represented in Table 1 above. Participants were however, compensated for their time and participation.

4 Supporting Mobile Developers (The Design)

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 Enhanced IDE by adding an extra module to it. Therefore, we wish to reconfigure the IDE by adding a mobile application pre-processor to it, allowing Java mobile developers to pre-process mobile applications for various mobile devices.

4.1 Design and Design Decision

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. 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. 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).

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 was done through a development environment (IDE) since almost all Java mobile developers are now developing mobile applications through one IDE or the other (Soroker *et al.*, 2006).

Therefore, we designed a plugin to be incorporated into the NetBeans IDE. The plugin we built is called Mobile Tools for NetBeans (MTN) which can be used to aid the development of mobile applications that can be easily ported into different mobile devices through using NetBeans IDE without the need to adapt the application for each mobile device profile. MTN's major function is to help Java mobile developers pre-process source code to adapt mobile applications to various mobile devices. The goal is to keep only one form of source code which, when pre-processed, 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.

5 Evaluation (A Task-Based Approach)

The results from the contextual inquiry have been applied in implementing a system to support Java mobile developers as presented in the design section. However, Jones and Marsden (2005) argued that designers or researcher may not know how useful their system is until an evaluation has been carried out. Hence, the design was evaluated. 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, 2004). 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.

However, 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 were not so interested in how efficient the users are in using the system, 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 were also not interested in knowing whether 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).

5.1 Development of the Evaluation Tasks and the Hypotheses

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

Task 1: To develop a simple mobile application and pre-process it 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 pre-process 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, (2003) suggestion on tasks to be used during evaluation and therefore were considered most important. The efficacy of the tasks was also reviewed by colleagues as well as the consulting HCI expert during the design of the 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 (Preece *et al*, 2007). This also helped us to decide the criteria for what would constitute successful completion of the task.

However, the following hypotheses were formulated in order to test the viability of the null hypotheses:

- ***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.***

The null hypotheses therefore is as follows:

- **With the current System, user will not be able to format their Java mobile application to various mobile devices to suite their needs**

5.2 The Pilot Study

In order to successfully evaluate this system, it is important that a pilot study be conducted (Preece, *et al.*, 2007). "*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. They were however, not allowed to participate in the final usability evaluation because of biasness 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 deeper problems in the full evaluation.

5.3 Subjects in the Pilot Study

The subjects in the pilot study consisted of sixteen people (5 undergraduates students and 10 postgraduate students of computer science (by Postgraduate we mean Honours, Masters and PhD degree), and 1 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.

5.4 Result and Discussion from the Pilot Study

Eleven of the subjects that participated in the pilot study did not find any limitations with the prototype. Four 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 device 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.

5.5 Subjects in the Evaluation

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, 60 subjects, all of who were students from the Computer Science department (31 PhD, 15 Masters, 9 Honours and 5 undergraduates). 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 and their time in the evaluation study. Fourty-Nine of the subjects were males while eleven were females. 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.

5.5.1 The Evaluation Environment

In order to guarantee comfort and provide a familiar environment, the evaluation was conducted in a usability laboratory 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).

5.5.2 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 more than 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

5.6 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 study and to be able to present the result that were obtained from the evaluation study that was conducted during this research, the data gathered were analyzed 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. It also forms 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).

5.7 Results

This section discusses the result of the evaluation that was conducted.

5.7.1 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.

5.7.2 Learning to use the system

The result captured and analyzed from the questionnaires that were filled out by the subject in response to the questions that were asked under category question "learning to operate the system" showed 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.

The result of our observation coupled with the users' response from the questionnaire show 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. These results show that the time to learn and operate the system was very quick.

5.7.3 System Capability

The result of the analysis shows that 78% of the subjects stated that the system was very fast; it took less than 10 seconds to pre-process an application for 15 different mobile devices.

The result also shows that 73% of the subjects agreed that the speed of operation of the systems was very fast and 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, 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.

User response to usability showed that users found the system satisfying. And 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.

5.7.4 Overall Result and Discussion

Johnson (2008) argues that responsiveness is the most important factor in determining user satisfaction with a system. 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 pre-process 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. This is so because the original system does not support these activities and hence our hypothesis is considered to have been proven.

5.7.5 Revisiting the Hypotheses

Refer to the Hypotheses earlier stated in section 5.2 which 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 pre-process 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 research study.: Based on this discussion, our hypotheses are considered to have been proven.

5.7.6 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. This had been achieved by this research.

The nature of NetBeans is such that it is modular, 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 which helped to provide support for Java mobile developers .

6 Conclusion

The major goal of this research was to establish how we can support Java mobile application developers through a Java IDE. NetBeans was the IDE of choice. As stated earlier, we chose NetBeans because through our initial study we discovered that NetBeans is considered as the most widely used IDE for Java mobile application development (Benson, *et al.*, 2004). A mobile tool call Mobile Tools for NetBeans (MTN) has been designed and developed to support our study. 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, and satisfying solution than those currently available. In addition, we presented the analysis and result of the usability evaluation that was conducted.

Through this study, we have been able to establish that contextual inquiry, which forms part of the new generation observation methodology, is the best to improve the usability of a system. This is because it allows a researcher to learn more about the users' activity in order to be able to provide support for them.

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 study 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 study focussed on better supporting java mobile applications developers in creating mobile applications using Java platform for a variety of J2ME-enabled mobile platform. We were able to achieve

this through the NetBeans IDE.It is therefore our belief that more researchers and designers should be able to use the ideas presented in this chapter to support their intended users.

References

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.

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.

Benson, C., Muller-Prove, M., and Mzourek, J. (2004). Professional usability in open source projects: GNOME, OpenOffice.org, Netbeans. *Proc CHI 2004* ACM Press (2004), 1083 – 1084.

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.

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.

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

Cross, K., and Warmack, A. (2000). Contextual Inquiry: Quantification and Use in Videotaped Analysis. *CHI (2000)*, ACM Press (2000), pp. 317-318.

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

Giraudo,U., Bordegoni, M. (2005). An empirical study on a user interface for a haptic based clay modeling tool from best practice user's observation. *Proceedings of HAAMAHA 2005.*

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

Holtzblatt, K. and Beyer, H. (1997). Contextual Design: Using Customer Work Models to Drive Systems Design.

Ext. Abstracts CHI '97. ACM Press (1997), 184-185. Atlanta, USA: ACM.

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

Jones, M., and Marsden, G., (2005). *Mobile Interaction Design. 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.

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.

Mace, M. (2006). *Mobile Opportunity*. http://mobileopportunity.blogspot.com/2006/05/flashversus-windows-can-adobe-break.html.

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

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*. http://www.useit.com/alertbox/20030120.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.

Raven, M.E and Flanders, A. (1996). Using Contextual Inquiry to Learn About your Audiences. *SIGDOC Asterisk Journal of Computer Documentation*. Vol 20, No. 1 pp 1 – 13. ACM, 1996.

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

Scholtz, J. (2004). *Usability Evaluation*. *Publication No. 545.* National Institute of National Institute of Standards and Technology. http://www.itl.nist.gov/iad/IADpapers/2004/Usability%20E valuation_rev1.pdf

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

Spool, J. and Schroeder, W. (2001). Testing web sites: Five users is nowhere near enough. *Ext Abstracts CHI 2001.* ACM press (2001), 285-286.

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

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

Thomas, K. (1999). Designing a Task-Based Evaluation Methodology for a Spoken Machine Translation System. *Proc. 37th ACLCL.* ACLCL Press (1999), pp 569- 572.

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

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

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

Wilson, A. (2006). *Write Once, Deply Anywhere*. Retrieved December 27, 2006, from 101 Writing Applications for mobile devices and reducing device fragmentation with NetBeans Mobility Pack: http://www.netbeans.org/community/magazine/html/03/mobpack/

Wixon, D., and Raven, M., E. (1994). Contextual Inquiry: Grounding your design in user's work. *Proc. CHI '94.* ACM Press (1994), pp 409-410.

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.

Czajkowski, K., Fitzgerald, S., Foster, I., & Kesselman, C. (2001). Grid information services for distributed resource sharing. In *10th IEEE International Symposium on High Performance Distributed Computing* (pp. 181–184).

Foster, I., Kesselman, C., Nick, J., & Tuecke, S. (2002). *The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration*. Technical report, Global Grid Forum.

Gusfield, D. (1997). *Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology*. Cambridge: Cambridge University Press.

Hern´andez, M. A. & Stolfo, S. J. (1995). The merge/purge problem for large databases. *SIGMOD Record*, 24(2), 127–138.

Smith, T. F. & Waterman, M. S. (1981). Identification of common molecular subsequences. *Journal of Molecular Biology*, 147, 195–197.

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