Computer Technology and Application 3 (2012) 533-543

Mobile Interaction Capturing System in Real Environments

Iván Pretel García and Ana B. Lago Vilariño

Deusto Institute of Technology (DeustoTech), University of Deusto, Bilbao 48007, Spain

Received: July 27, 2012 / Accepted: August 24, 2012 / Published: August 25, 2012.

Abstract: Mobile applications are being used in a great range of fields and application areas. As a result, many research fields have focused on the study and improvement of such devices. The current Smartphones are the best example of the research and the evolution of these technologies. Moreover, the software design and development is progressively more focused on the user; finding and developing new mobile interaction models. In order to do so, knowing what kind of problems the users could have is vital to enhance a bad interaction design. Unfortunately, a good software quality evaluation takes more time than the companies can invest. The contribution revealed in this work is a new approach to quality testing methodology focused on mobile interactions and their context in use where external capturing tools, such as cameras, are suppressed and the evaluation environments are the same as the user will use the application. By this approach, the interactions can be captured without changing the context and consequently, the data will be more accurate, enabling the evaluation of the quality-in-use in real environments.

Key words: Human computer interaction, quality in use, mobile services, context-awareness.

1. Introduction

Computers are being used in a wide range of environments and application areas. Consequently, several research fields have been developed in the study and improvement of such devices. Furthermore, the continuous decreasing of the price of these devices and the connection fees provided by the mobile operators has led to the massive use of the Smartphones. Nowadays, these devices have evolved from being exclusively used in business environments to being available for every member of the Information and Knowledge Society. In fact, according to the last Cisco Visual Networking Index [1], the average Smartphone usage has nearly tripled in 2011 and the average amount of traffic per Smartphone in 2011 has been 150 MB per month, up from 55 MB per month in 2010. As a result, Smartphones have been introduced not only into a lot of business models but also into all fields of human activity linked by the key capability: full connectivity. By this key, everybody can be aware of their surrounding environment such as economics and entertainment among others.

These new social tendencies create the need for better awareness and readiness to face these demands quickly. Software quality is becoming increasingly important due to the exigencies of the new and aggressive mobile software market. In addition, the software design and development is progressively more focused on the user. Due to this tendency, a great amount of resources are invested in the long-term ambition of finding and developing mobile device interaction models. Knowing how users feel using the mobile devices and what kind of problems could they have is vital to enhance a bad interaction design. Unfortunately, a good software quality evaluation takes more time and money than the companies can invest.

Ana B. Lago Vilariño, Ph.D., research fellow, research fields: human computer interaction, mobile services, context-awareness.

Corresponding author: Iván Pretel García, Ph.D. student, research assistant, research fields: human computer interaction, mobile services, context-awareness. E-mail: ivan.pretel@deusto.es.

The contribution revealed is a new approach to quality testing methodology focused on mobile applications where it is possible to capture the interaction by being aware of the influence of external elements used to monitor the interaction. Firstly, the quality standards are studied in section 2. Secondly, the quality in use is explained through the section 3. In section 4, the context in use is studied and the context focused on mobile interactions is defined. Interaction capturing methods and existing systems are studied in section 5. The mobile interaction capturing system is presented in section 6. After defining it, the experiment and results are shown in section 7. Finally, the research is concluded and further work discussed in section 8.

2. Quality Standards

In order to perform a study focused on the quality and quantity in use, the first step is to analyse, compare and find the differences and connections between the standards, concepts and conclusions related to the quality field.

Since 1976, a wide array of quality models and frameworks have been published, recognized and taken into account even in the current standards, including the work made by Boehm [2-3] and Mc Call [4]. According to the conclusions obtained by the mentioned work, quality is a composed and multidimensional concept; it cannot be directly measured. In order to perform better management, the quality model for general purpose has a minimum amount of characteristics by which any kind of software can be evaluated.

In 1994, ISO DIS 8402 [5] has defined quality as all the characteristics of a product that have the capability to satisfy the explicit and implicit needs. Nigel Bevan, after defining usability as the ease of use and acceptability of a product for a particular class of users carrying out specific tasks in a specific environment, has made great progress inside the quality field with Motoei Azuma. In the work on quality in use in 1997 [6], they have described generic concepts of perceived quality (quality perceived by the users is studied as a subjective and inaccurate property) and quality in use. Furthermore, David Garvin [7] claims that these concepts can be as subjective as aesthetic evaluations; but he maintains that a quality of a product can be evaluated only in relation to the purpose for which it is created.

The ISO 9000 standard [8] defines quality as the level of compliance degree that a set of inherent characteristics of the product meets the requirements. As a consequence of this situation, the work on quality in use has merged with traditional approaches to quality bringing a wider and more important vision. This vision covers the quality perceived by users and the quality directly linked to the needs of users. For example, a big water resistant camera will show different quality. If its owner likes diving and he wants to make pictures under the water, it shows a wonderful quality. However, if the owner uses it to take photos during a tourist trip, the quality perceived is much worse than the first situation because the need is different.

Quality characteristics and the associated measures can be useful not only to evaluate a software product but also to define the quality requirements. Consequently, the last step is simple: the ISO/IEC 9126 [9] standard, which is the predecessor of SQuaRE [10], has mainly been replaced by two related multipart standards: ISO/IEC 9126 (Software product quality) and ISO/IEC 14598 (Software product evaluation).

It has to be stressed that the SQuaRE series of standards is only dedicated to software product quality. SQuaRE ISO/IEC 25000n—Quality Management Division addresses software product requirements specification, measurement and evaluation; and it is separated from the “Quality Management” of processes, which is defined in the ISO 9000 family of standards.

3. The Quality in Use

In order to study the best way to capture mobile interactions, the ISO/IEC 9126-4 standard is used as basis. According to ISO/IEC 9126, software quality is the totality of features and characteristics of a software product that bears in its ability to satisfy stated or implied needs.

In this definition, a quality represents a property of the software product defined in terms of a set of interdependent attributes (such as usability, security, reliability, performance, complexity, readability, reusability), expressed at different levels of detail and also take into account the particular context of software use. Finally, a number of methods of measurement (metric) should be defined in order to verify which of these attributes are nowadays important in software.

ISO/IEC 9126 defines a quality of software testing framework by three aspects: internal quality, external quality and quality in use.

Internal quality is the totality of characteristics of the software product from an internal view (e.g., resource utilisation, stability, analysability…). This kind of quality can be improved during code implementation, reviewing and testing.

External quality is the quality when software is running, which is measured and evaluated focusing on the software behaviour (e.g., number of wrong expected reactions). It is usually measured and evaluated for software testing in a simulated environment with simulated data and by using external metrics.

Quality in use is the quality of the software system that user can perceive when the software is used in an explicit context of use. It measures the extent to which users can complete their tasks in a particular environment. It is measured by four main capabilities of the software product in a specified context of use: effectiveness, satisfaction, productivity and safety.

The quality of a software product can be evaluated by measuring their internal attributes, external attributes or the attributes that make up the quality in use.

The main purpose is that software will help and have the desired interaction with the user in a particular context of use. As Fig. 1 exposes, the internal quality improvement of the product contributes in improving the external quality. In the same way, the external quality improves the quality in use. Consequently, the assessment of quality in use provides clear indicators of deficiencies.

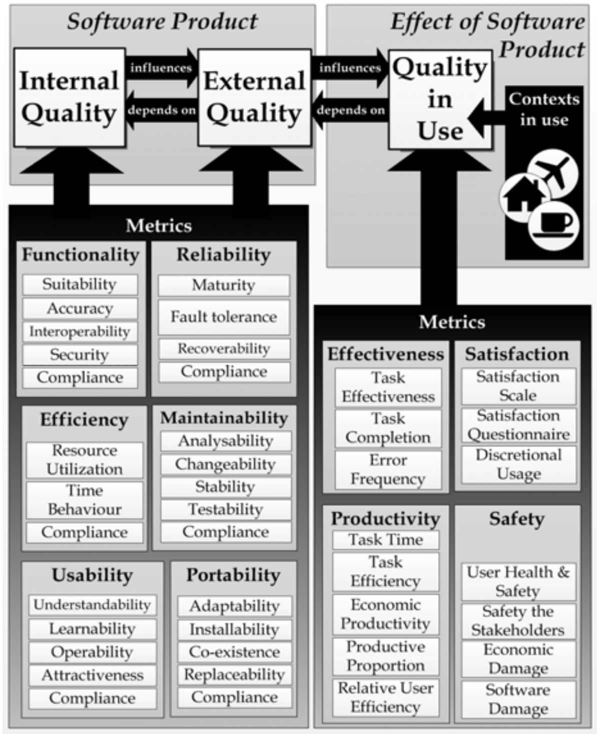

Fig. 1 The different quality aspects.

In order to calculate how the quality in use of the evaluated software is, its capabilities have to be measured. Focusing on mobile devices, every software capability has to be measured per task and also per user, who is surrounded by the context in which actions are needed to be tracked. Owing to the wide range of contexts, an explicit context in use definition focused on mobile interactions has to be taken into account.

4. Context in Use

So as to understand the role of context in the software evaluation, it is necessary to study what context is. Study and monitoring the context in which measurement takes place have been a routine for centuries. In fact, much of the early human factors work has been performed in the military sector to test equipment components in unstable, harsh and extreme environments to represent battlefield conditions [11]. The two main advantages to be aware of the context is that studying it ensures taken measurements reproducible and also reliable. But as explained previously, the context in which the product is used changes completely the result of the task performance.

The context has been taken into account since many decades ago and it is a term defined several times by a lot of researchers, experts and communities. According to ISO 9241-11 standard [12], context in use is defined as every user, task, equipment and also physical, and social environment that is affected by the interaction. In 2007, the National Institute of Standards and Technology [13] added the stakeholders to the context in use defined in the 9241 standard. Other context in use definition is specified by Kankainen [14], who defines context in use as the environment that involves the user and his community.

Focusing on the environment, Schilit [15] maintains that context should be defined answering where the user is, who the user is with, and what resources are nearby. Following the idea, Brown and Bovey [16] define the context as location, identities of the people around the user, season, the time, and the weather. Ryan, Pascoe and Morse during the implementation of their context aware archaeological assistant [17] add to the last definition not only the weather but also all the surrounding environmental context.

On one hand, several studies maintain that the context is just the physical location. In particular, the location is one of the most influencing aspects of the contexts but is not the unique. On the other hand, other studies add to the location more attributes such as weather to achieve a more accurate definition of the environment and the physical context. Furthermore, the stakeholders and the community of the user are taken into account in the context definition.

A great description of the context is the explained in the work of N. Savio and J. Braiterman [18]. They define the context by enumerating the following layers: culture, environment, activity, goals, attention, tasks, interface, device, connection and carrier.

In 2000, A.K. Dey and G.D. Abowd [19] define the context focusing on the user and the executed application. They define the context as any information that can be used to characterize the situation of an entity. And an entity as a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.

After seeing the different definitions of contexts and focusing it on the mobile topic, it is concluded that components (See Fig. 2) that form the context are: the environment (physical, ambient, technical and sociocultural), the user, his mobile device and the executed application.

Firstly, the environment is formed by four groups of attributes: physical, ambient, technical and sociocultural groups. Through the physical group the tangible environment (e.g., work area dimensions) is described. The aim of ambient group is to keep attributes that can describe meteorological conditions (such as the light level, the humidity of the environment or its temperature). The technical group defines every characteristic excluding the mobile device (i.e., connectivity attributes, hardware and software characteristics, etc.). The sociocultural attributes group defines the cultural and social agents that can determine the interaction (e.g., cultural habits, religion).

The user is defined by four main groups of attributes: personal information, knowledge, skills and attitudes. The personal attributes are name, age and sex. The knowledge is formed by the languages, the experience

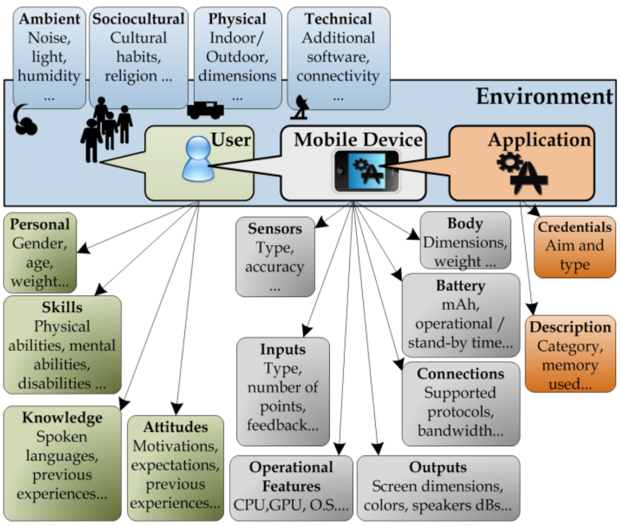

Fig. 2 Context in use components.

and eases with the tasks defined and similar products, the education level and culture. Physical abilities, mental abilities, disabilities and qualifications form the skills group. The attitudes group is formed by the personal motivations, and the past and expected experience performing the tasks.

The mobile device is formed by seven groups: connections, body, inputs, outputs, sensors, battery and operational features. The connections group defines every communication interface the mobile device can use such as the 3G connection and its bitrate. The body of the device defines the dimensions and the weight of the device. Two of the most important elements to define in this group are the output interfaces (e.g., the screen and speakers) as well as the input interfaces (e.g., multitouch screen, T9 keyboard …). The battery group describes the information about the battery of the device and the time the device can work without being charged. The sensors group describes the features of the sensors (e.g., accuracy, model …). Last but not least, the operational features group, is formed by attributes like the CPU (Central Processing Unit) information, memory, etc.

The application group defines the information about the tested application. This group is mainly formed by two groups of attributes: the description of the application and the required credentials (e.g., location access, contacts …). Inside the description group, it defines a brief explanation of the application, its aim, the memory used and the category of the app. Starting from the Google Play platform [20], the defined categories are books, news, business, communication, education, entertainment, finance, health, lifestyle, video, audio, medical, personalisation, photography, productivity, shopping, social, sports, tools, transport, travel, weather and widgets.

After defining the context, it has to be kept in mind that if the interaction is performed through mobile devices, the interaction will be extremely context-dependent. Therefore, context in use focused on mobile-human interaction has to be measured as frequently as possible. Another thing to keep in mind is that the factors which may affect the context such as the monitoring tools need to be carefully considered, because the result of the evaluations may vary in different settings.

5. Interaction Capturing Methods

Nowadays, several tools to capture interaction have been developed. Inside this wide range of tools, the authors can separate between tools developed in the research context and commercial tools.

Inside the commercial group, tools to enhance the analysis of the interaction such as Morae [21] and the Noldus portable usability Lab [22] can be seen. As one of the leading interaction capturer tool on the market, Morae platform offers a broad testing experience for its users. All the interaction data is captured digitally and indexed to one master timeline for instant retrieval and analysis. In particular, the mobile screen recording data is not available but it uses a log and an external camera to capture events such as the task start and end times, the buttons clicked and the menu items chosen, the errors made by users, comments and quotes, etc. The Noldus portable usability lab is other professional solution for on-site usability testing. This system has a portable set of tools to carry out interaction captures in real environments. The screen capture device has a high frame rate and provides high quality images of the screen at which the test participant is looking. The device is plugged into the test computer to record the image on the screen.

The most highlighted tool inside the research field is DRUM (Diagnostic Recorder for Usability Measurement) [23]. It is a software tool that allows the analysis of video. It is developed at NPL (National Physical Laboratory) for the project MUSiC [24] (Metrics for Usability Standards in Computing) to provide support for observational assessment of usability and reduce the time of analysis from 10 hours to only 3. This tool has been built since 1990, making a compilation of the extensive requirements of usability analysis, and a detailed study of existing tools to support the assessment, in close cooperation with Human Computer Interaction professionals and software engineers as well as industrial users to gather the needs identified in the usability testing. It has an event logger that can be run in real time. The tested user and the evaluator comments can be inputs to record at any time.

It can be perceived that the majority of the interaction capturer tools offer the choice to record the users during the tasks and logging the produced events. Furthermore, surveys are widely used to measure the user satisfaction. Centring on the evaluation of applications for mobile devices, these tools have also an external camera added to the mobile or on a helmet, allowing the user interactions capturing. These added elements influence the context (the ergonomics of mobile device and the comfort of the user). If a user whose phone has external capturing accessories such as the mentioned devices, he will feel uncomfortable and he will change his behaviour. Consequently, this interaction will be corrupted and it will show worse quality results than without the added devices.

On one hand, numerous tools are executed in a testing laboratory. All influencing factors can be controlled and data can be recorded with several capturing tools. However, the context, which is the most influential factor, is not taken into account or it can hardly be simulated. Furthermore, the tested users can have a tendency to show expected (but not real) results due to the being observed and being evaluated feelings (i.e., if the user detects cameras, he will try to show more positivity than the real).

The main weakness of this kind of tools is the added capturing elements like an external camera or the testing laboratory because they change the environment; and consequently, the retrieved information about the interaction. In order to solve the main weakness, the work exposed through this article aims to automate the capture, minimizing the subjectivity to the context generated by the capturing tools.

So as to automate the capture minimizing the alteration of the context, the capture must be done using the mobile device. The element of the context that is altered is the mobile device because the logging software can reduce its performance. It has a limited memory and limited processing; consequently, the amount of captured information will be reduced.

6. Mobile Interaction Capturing System

After seeing the main requirements to achieve a great interaction capture, the authors have defined an interaction capturer tool. The capturer is formed by two main applications: the capture tool and the analysis tool. The capture tool is the key application; it is installed on the mobile device and it captures the interaction. The analysis tool emulates the interaction to add all the metrics that allows the quality in use evaluation.

6.1 Capture Tool

The capture tool has to be executed by the user on the device in order to perform the assigned tasks. This software is developed to run in Symbian S60 systems and it is formed by two main modules: the GUI (Graphical User Interface) Controller and the Interaction Capturer.

The first module (See Fig. 3) is a GUI to guide the user during the experiments. This tiny application acts as a guide; it shows the tasks to complete and the generic context where user has to work. The user must select a task and notify the application that the asked task is going to be completed. This module knows the tasks to do through a configuration file. This file has the description of the tasks, the contexts in which the user has to perform the task and the questions to answer at the end of them. While the user is performing the task, the application records his activity. After doing the task, the user should notify the task ending through the mobile application and automatically this event is stored. Finally, the questionnaire is shown to capture the context information, which is filled at the end of the tasks to not disturb the user.

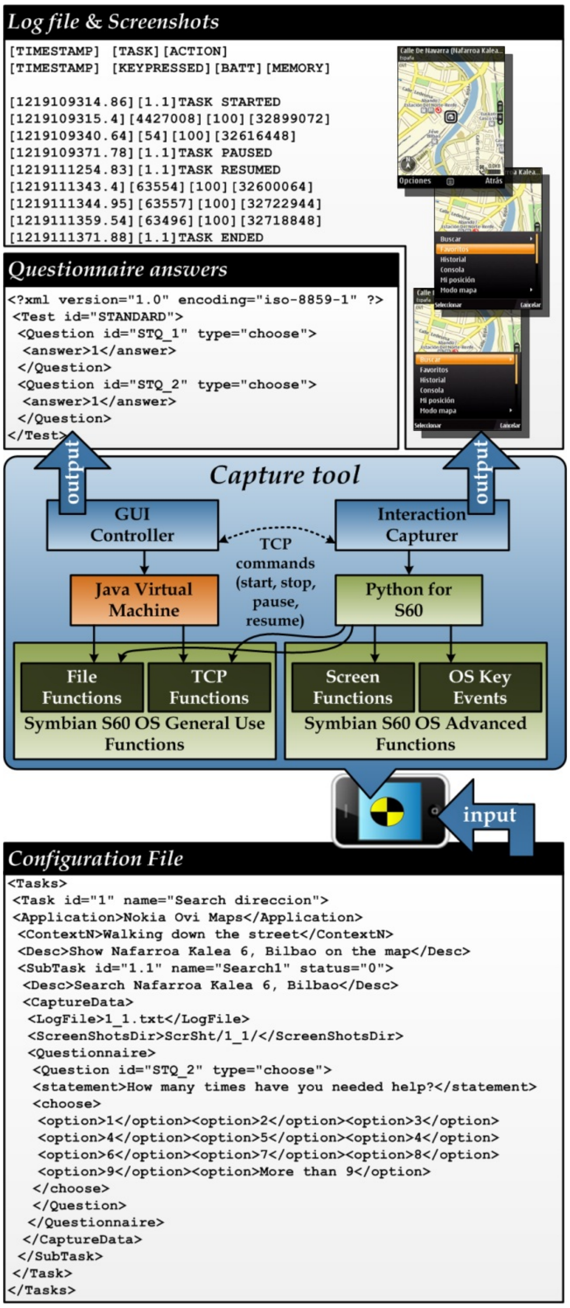

Fig. 3 Capture tool diagram and files description.

One problem the Java Virtual Machine (J2ME version) has is the access; through this technology, the screen and the key events cannot be accessed without having the focus on the application. To solve this problem, the Interaction Capturer module has been developed in PyS60 (Python for S60). PyS60 can access the screen and the key events without having the main focus on the application. Thanks to this module the interaction can be logged and the screenshots can be stored. The information generated by the mobile application is stored in three types of files: image files (png format), a test answers file (xml format) and a log file (text format). The image files store the screenshots obtained during the tasks performance and they are identified by their name. The name is composed by the task identifier and the timestamp of the screenshot. The log file stores the timestamp when a specific task is started, stopped, paused or resumed; the key pressed, the timestamp of the event, the battery level at this moment and the memory used. The answers file stores the answers of the questionnaire shown at the end of the task.

In order to enable the communication between the two components of the mobile capture tool, the authors have used the TCP (Transmission Control Protocol) protocol. The GUI Controller sends commands to the Interaction Capturer to notify the tasks are starting, stopping, resuming or pausing.

6.2 Analysis Tool

After the capture of the interaction, the data generated during the execution of the experiment is dumped in the PC (Personal Computer) provided with the interaction analysis tool (See Fig. 4). It is a desktop application composed by three modules: Simulator Module, Analysis Module and Reports Module.

Firstly, the Simulator Module takes the captured data (log, screenshots and test answers) to normalize and store it in the database of the system powered by MySQL. Unfortunately, some metrics needed to analyze the quality in use characteristics are not captured automatically. In order to capture them, the desktop application has a simulator to reproduce the interaction. The simulator takes the timestamp and the

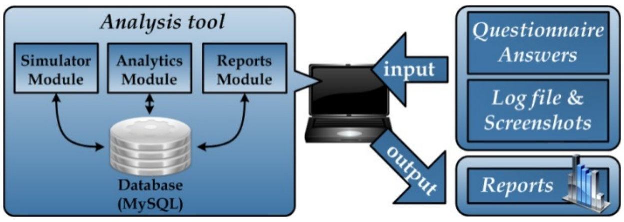

Fig. 4 Analysis tool diagram.

captured screenshots to simulate the interaction. During the analysis, the quality in use metrics not captured can be filled in order to calculate the final results.

After the simulation and the missing data addition, the four specified quality in use characteristics (effectiveness, satisfaction, productivity and safety) are calculated through the analysis module.

The quality in use metrics depend on a lot of information (See Tables 1-4). The types of data which compound this information are the proportional value of each missing or incorrect component in the task output (Ai); number of tasks completed (TC); number of tasks attempted (TA); number of errors made by the user (E); task time (T); total cost of the task (C); spent help time (H); spent error time (Et); search time (S); ordinary user's task efficiency (OU); expert user's task efficiency (EU); number of users reporting Repetitive Strain Injury as headaches or fatigue (RSI); total number of users (U); number of people put at hazard (PH); total number of people potentially affected by the system (PPA) ; the number of occurrences of economic damage (OED); number of occurrences of software corruption (OSC); total number of usage situations (US); questionnaire producing psychometric scales (PS); population (P); responses to a question(Qi); number of total responses (n); number of times that specific software functions/applications/systems are used (A) and also the number of times they are intended to be used (B).

Finally, the report builder can build reports which can include all information about the experiment.

6.3 Methodology

The methodology (See Fig. 5) to be used for the evaluation of the interaction is influenced by the design of the architecture and the data capture tool. It is divided in four main steps:

- In the beginning, the devices to be used by the users have to be configured. First of all, every task and its context have to be specified. The configuration consists in defining tasks and contexts in which the

Table 1 Effectiveness metrics.

Metric	Formula	Description
Task Effectiveness (TE)	|1-ΣAi|	What proportion of the goals is achieved correctly
Task Completion (TCM)	TC/TA	What proportion of the tasks is completed?
Error Frequency (EF)	E/T	What is the frequency of errors?

Table 2 Productivity metrics.

Metric	Formula	Description
Task Time (T)	T	How long does it take to complete a task?
Task Efficiency (TEF)	TE/T	How efficient are the users?
Economic Productivity (EP)	TE/C	How cost-effective is the user?
Productive Proportion (PP)	(T-H-Et-S)/T	What proportion of the time is the user performing productive action?
Relative User Efficiency (RUE)	OU/EU	How efficient is a user compared to an expert?

Table 3 Safety metrics.

Metric	Formula	Description
User Health and Safety (UHS)	1-RSI/U	What is the incidence of health problems among users of the product?
Safety of People Affected (SPA)	1-PH/PPA	What is the incidence of hazard to people affected by use of the system?
Economic Damage (ED)	1-OED/US	What is the incidence of economic damage?
Software Damage (SD)	1-OSC/US	What is the incidence of software corruption?

Table 4 Satisfaction metrics.

Metric	Formula	Description
Satisfaction Scale (SS)	PS/P	How satisfied is the user?
Satisfaction Questionnaire (SQ)	Σ(Qi)/n	How satisfied is the user with specific software features? Questions average.
Discretional Usage (DU)	A/B	What proportion of potential users chooses to use the system instead of others?

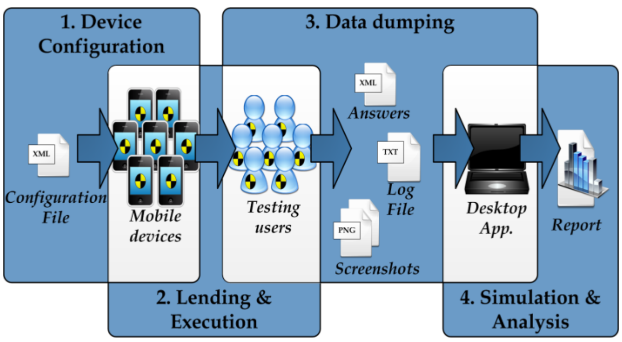

Fig. 5 Methodology.

users have to do the experiment. In order to do so, these contexts and tasks are defined by one xml file which is stored in the mobile device;

- Later, the users have to borrow the configured phones and do the specified tasks in the specified contexts. The task information is shown by the graphical user interface of the capturing system. When user chooses one task to do, he notifies the application he is going to start. After ending the task, the user notifies the application the task is ended. The system stops capturing;
- When the user ends every task, he has to go the lent device back. The interaction data is dumped from the device to the desktop application;
- After dumping information, the system experts has to introduce missing data (i.e., interaction errors, search times...) simulating the interaction by processing the recorded information. Finally, all necessary information to analyse the quality in use is stored. This quality in use information is generated with the analysis tool calculating metrics and finally making the final report.

7. Experiment

The implemented version of the system is validated by one preliminary evaluation. The authors have used four Nokia N96 phones used by four users (See Table 5), performing tasks (See Table 6) within two different contexts: at home and walking down the street.

According to the methodology exposed in this work, the devices are configured by creating and copying the configuration files according to three different tasks. Secondly, phones are lent and users have been performed the tasks. The experiment takes one week from the lending to the return of the devices. When the phones are returned and after dumping the data into the analysis tool, the expert evaluate the captured interaction. Finally, the report is generated.

As mentioned before, the quality in use has to be calculated measuring the four properties: effectiveness, productivity, safety and satisfaction. The averages (See Table 7) show that using the application walking down the street context is more unsafe than at home context. Although walking down the street users are more satisfied, they are more productive and efficient at home. Moreover, the proposed system shows that inexpert users are less productive than expert ones.

Table 5 Users description.

Id	Gender	Age (years)	Experience using maps 0 (low)-100 (high)
1111	Male	26	60
2222	Male	28	85
3333	Female	23	10
4444	Female	25	25

Table 6 Tasks description.

Task	Id	Description
Direction search (Search1)	1.1	Show NafarroaKalea 6, Bilbao on the map
Direction search (Show)	1.2	Show it on the satellite view
Services search (Location)	2.1	Locate the main pointer on Bilbao map
Services search (Search2)	2.2	Search nearby museums and go to Guggenheim
Configuration	3.1	Change route mode to on foot

Table 7 Averages of the calculated metrics.

Criteria	Effectiveness	Security	Productivity	Satisfaction
User 1111	0.683	1	1	1
User 2222	0.674	1	1	1
User 3333	0.675	1	1	0.989
User 4444	0.568	0.889	0.889	1
Home context	0.679	0.892	0.528	0.528
Walking context	0.621	0.881	0.438	0.656

The carrier furthermore, as Fig. 6 exposes, the authors have measured the required time to store the interaction metrics and the screenshots (Capturer Rate) and the time between the pressed keys (User Interaction Rate). The dataset retrieved is formed by 329 samples. According to this dataset the mobile interaction capturer needs an average of 0.35 seconds to store a 240×320 image in a jpeg format and write a line in the log file. This data shows the interaction of the user is not affected by the system if the user does not press keys faster than 0.35 seconds. Additionally, according to Fig. 7, the system needs an average of

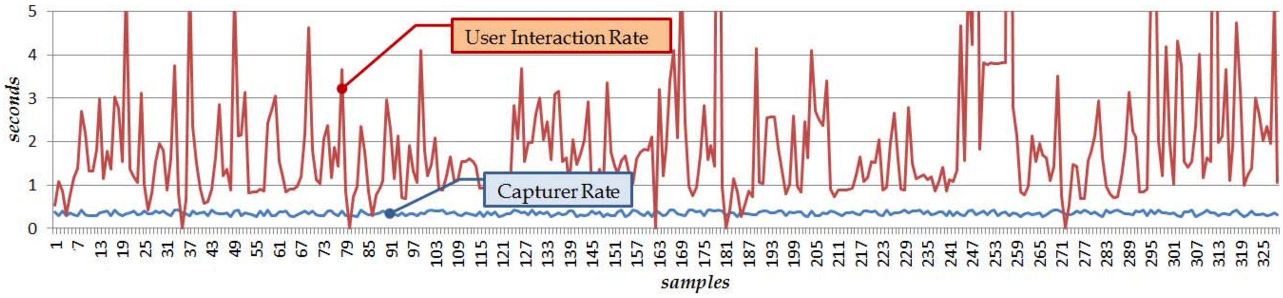

Fig. 6 Capturer and user interaction rates.

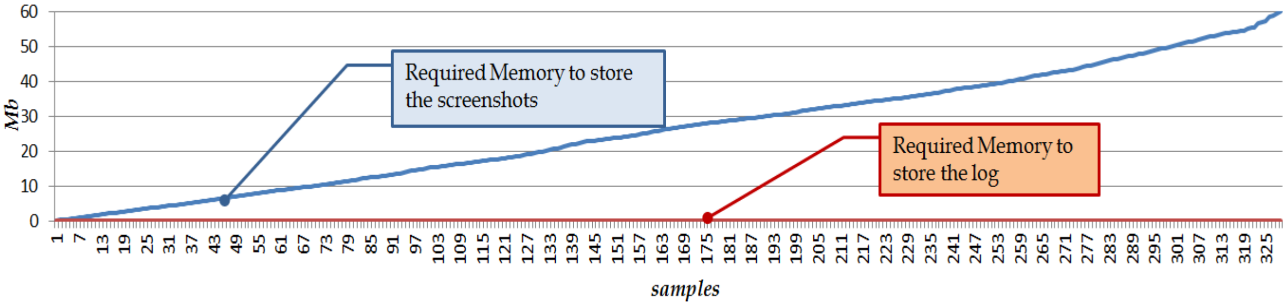

Fig. 7 Required memory to store the interaction.

16.99 Kbytes per key pressed to store the data. The required memory is moderately low because follows a lineal distribution. If there is 500 Mb reserved to store experiment interactions, 30120 pressed keys can be stored.

8. Conclusions

The implemented system can measure the quality-in-use in real environments without human observers. It is possible with a minimum change of the context, which is caused by the tiny application to capture the interaction stored in the mobile device if the user interacts faster than 2.85 pressed keys per second. However after the experiment, the authors interview the users and their main feedback is they do not like performing tasks when they are forced to do so. They feel okay and according to their answers they do not feel observed.

In this paper, the authors have studied a context model for mobile to calculate the quality in use of the mobile applications. A complete functional prototype to capture interaction in real environments with Symbian OS devices is developed and validated. According to the retrieved feedback, the next step is to study methods to test applications without forcing the user to perform tasks because the interaction has to be spontaneous. Because of that, the authors are going to study how to detect what action done by the user is interesting to capture.

This work reveals whether the quality in use testing is focused on mobile interaction, the context can be extremely easily influenced. But it is possible to capture the interaction without changing the context in use.

References

[1] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2011-2016, White Paper, Cisco, February 14, 2012.

[2] B.W. Boehm, J.R. Brown, M. Lipow, Quantitative evaluation of software quality, in: Proceedings of the 2nd International Conference on Software Engineering, 1976, pp. 592-605.

[3] B.W. Boehm, J.R. Brown, M. Lipow, J.L. MacLeod, M.J. Merrid, Characteristics of Software Quality, Elseiver North Holland, 1978.

[4] J.A. McCall, P.K. Richards, G.F. Walters, Factors in Software Quality. Vol. I, II, III, Final Technical Report,

RADC-TR-77-369, Rome Air Development Center, Air Force System Command, Griffith Air Force Base, New York, 1977.
[5] ISO DIS 8402, Quality Vocabulary, 1994.
[6] N. Bevan, M. Azuma, Quality in use: Incorporating human factors into the software engineering lifecycle, in: Proc. 3rd International Symposium and Forum on Software Engineering Standards, 1997, pp. 169-179.
[7] D. Garvin, What does product quality really mean?, Sloane Management Review (1984) 25-43.
[8] ISO 9000:2005, Quality Management Systems—Fundamentals and Vocabulary, 2005.
[9] ISO/IEC 9126:2001, Information Technology—Software Product Evaluation Quality Characteristics and Guidelines for Their Use, 2001.
[10] Software Engineering—Software Product Quality Requirements and Evaluation (SQuaRE)—Guide to SquaRE, 2005.
[11] D. Meister, The History of Human Factors and Ergonomics, CRC Press, 1999.
[12] ISO 9241-11:1998(E), Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs). Part 11: Guidance on Usability, 1998.
[13] NIST IR 7432, Information Access Division Information Technology Laboratory, Common Industry Specification for Usability Requirements, 2007.
[14] A. Kankainen, Thinking model and tools for understanding user experience related for information appliance product concepts, Dissertation of Anu Kankainen, 2002.
[15] B. Schilit, M. Theimer, Disseminating active map information to mobile hosts, IEEE Network 8 (5) (1994) 22-32.
[16] P.J. Brown, J.D. Bovey, X. Chen, Context-aware applications: From the laboratory to the marketplace, IEEE Personal Communications 4 (5) (1997) 58-64.
[17] N. Ryan, J. Pascoe, D. Morse, Enhanced reality fieldwork: The context-aware archaeological assistant, in: V. Gaffney, M.V. Leusen, S. Exxon (Eds.), Computer Applications in Archaeology 1997, Tempus Reparatum, 1998.
[18] N. Savio, J. Braiterman, Design Sketch: The Context of Mobile Interaction, in: Mobile HCI 2007, Singapore, 2007.
[19] A.K. Dey, G.D. Abowd, Towards a better understanding of context and context-awareness, in: CHI 2000 Workshop on What, Who, Where, When and How of Context-Awareness, New York, 2000.
[20] Google Play Homepage, http://play.google.com/.
[21] Techsmith Corporation, Hardware Usability Testing with Morae, 2009.
[22] Usability Lab Homepage, http://www.noldus.com/human-behavior-research/solutions/stationary-usability-lab.
[23] M. Macleod, R. Rengger, The development of DRUM: A software tool for video-assisted usability evaluation, in: Proc. of HCI'93, 1993, pp. 293-309.
[24] N. Bevan, M. Macleod, Usability measurement in context, Behaviour and Information Technology 13 (1994) 132-145.