High usability Graphic User Interfaces (GUIs) for the continuous assessment of chronic diseases

Ignacio Peinado, Elena Villalba, Mª Teresa Arredondo
Life Supporting Technologies
ETSI Telecomunicación
Universidad Politécnica de Madrid
ipeinado@lst.tfo.upm.es

Abstract

This paper presents technical solutions for achieving high usability in user interfaces for the continuous assessment of chronic diseases. Our overall goal is to design usable systems, which support specific users in their specific context to reach their particular goals effectively, efficiently and with satisfaction. Usability is a quality attribute that assesses how easy user interfaces are to use, defining five quality components: learnability, efficiency, memorability, errors and satisfaction. Our approach is to keep interfaces as simple and clear as possible.

We have worked within the MyHeart project, which deals with patients with chronic cardiovascular diseases. Patients are equipped with an intelligent biomedical garment capable of acquiring, processing and evaluating physiological data such as ECVs. The results are sent via a wireless personal area network to a mobile phone or PDA and from there to a server farm. The mobile device will include an avatar that will perform the interaction with the user in a non-intrusive way and graphical elements with speech synthesis support. The best way to successfully apply usability criteria is putting the maximum effort before the design phase. In latter phases of the development, when designing the presentation model, we will consider the quality model previously defined, taking into account different heuristics, usability guidelines and accessibility criteria to improve the quality of the layouts.

1. Introduction

Among the current aging population, more and more health resources are dedicated to chronic diseases. Furthermore, most patients have more than one chronic disease, several medications and quality of life issues that loom larger than the narrow definition of medical care. The massive advances in technology have changed the paradigm from clinic and hospital-based reactive care to patients' self-management of their health status in cooperation with their relatives. This approach is possible by means of medical systems that monitor key physiological factors and convey the information to the health care professionals.

Two new concurrent trends in the Information and Technologies world have led to a new paradigm in the health care process, moving from e-Health to p-Health (personalized health). On one hand we have the pervasive diffusion of intelligence in the space around us through the development of wireless technologies, advanced mobile communications and intelligent sensors. On the other hand the increase of richness and completeness of communications through the development of multimedia technologies such as virtual reality and natural language and speech technologies. Within the p-health paradigm we are heading towards a shared immersive e-therapy in which the presence, simulation and experience components are key factors of the therapeutic process.

Despite all these changes, there are still some problems to solve. Graphical User Interfaces have evolved from the classic WIMP (Windows, Icons, Menus and Pointers) to new techniques such as gesture recognition, multimedia, 3-D, Virtual Reality, Computer Supported Cooperative Work, natural language and speech, ambient intelligence, etc. However through this evolution usability is always the main concert of every software system designer.

Usability has usually been related to ease of learning, which is not always easy to achieve and is only one part of the big picture. The whole picture is generally seen as encompassing at least five factors [1]:

1. Ease of learning
2. Ease of retention
3. Efficiency of use
4. Reliability of use
5. Goal satisfaction

The concept of usability is not a new idea, in fact, from the late 70's usability was considered as a software quality attribute. However, historically its importance has been relegated in relation to other quality factors such as efficiency, reliability, safety and even

aesthetics. For this reason, most software development methodologies do not include mechanisms to take into account this critical factor in the software development process. Nowadays, in a market saturated with brands that can develop systems with very similar technical characteristics usability has become a competitive advantage. Most importantly, in medical systems (e.g. for the assessment of chronic diseases) it has become a security problem, since that usability problems can induce errors that may have fatal consequences.

2. Methodologies

Landauer stated that 80% of maintenance costs – that represents the 80% of software development total costs – are due to user-system interaction problems rather than technical problems [1]. What's more, 64% of these costs are related to usability problems. It is not difficult to find in the literature examples of technically excellent systems that have failed due to their lack of usability. Usability engineering has become widely important as an evidence-based methodology that involves end users throughout the development process to produce information systems that are measurably easier to use, learn and remember.

One of the first approaches to the usability problem was proposed by Deborah Mayhew [2]. She defined the Usability Engineering Lifecycle, which she divided into three main stages: a) Requirements Analysis, b) Design/Testing/Developing and c) Installation. This development process follows the waterfall lifecycle. Its main flaw was its lack of real iterativeness: it starts with the requirement analysis, then goes to the Design/Testing/Development phase and ends in the installation stage, but it only returns to the requirements phase if a lack of functionality is detected in the latter stage.

Costabile [3] tried to fill the holes of the Usability Engineering Lifecycle by integrating user-centered methods into the software development process. Besides adding real iterativity, she stated three basic concepts:

- User and Task Analysis.
- Iterative system design and implementation by using prototypes of increasing complexities.
- Evaluation of prototypes with users.

The first one is similar to the requirements analysis of the Waterfall Lifecycle, but focusing on the users' needs and expectations and the tasks they need to accomplish. The second one is related to prototyping and tests. Although its importance is undeniable due to the fact that it is one of the first really iterative methodologies and tries to fulfill the user's satisfaction, using real prototypes has proven to be really inefficient. Besides the expense of money, usually test results are delivered in the latter stages, generally after the release of an alpha or beta version of the system, at a point where changes are quite improbable.

Usage-centered design (Costantine and Lockwood, 1999; 2002 [4], [5]) is a systematic, model-driven approach to visual and interaction design in software and Web-based applications. As the name suggests, usage-centered design differs from conventional user-centered approaches primarily in a shift of focus from users *per se* to usage. That is, to the tasks to be accomplished by users. This difference in emphasis is reflected in differing practices with a significant impact on the development life cycle and software integration and development. Most importantly, usage-centered design eschews the repetitive cycles of trial design and user testing common to traditional user-centered approaches. Thus, in the design process the final solutions may be derived more or less directly from robust and highly refined models reflecting genuine user needs. The final interface derives directly and systematically from a series of core models:

- Users' role: one kind of relationship users could have with a system (e.g. frequency, volume and direction of information exchange, etc.)
- Task model: set of task cases and a map of the interrelationships among those task cases.
- Content model: contents and organization of the user interface needed to support the identified tasks (abstract prototypes).

To sum up, the main differences between user-centered and usage-centered design are shown in the next table:

User-Centered Design	Usage-Centered Design
Focus on Users: User experience, user satisfaction	Focus on Usage: Improved tools supporting task accomplishment
Driven by user input	Driven by models
Substantial User involvement: User studies, participatory design, user feedback, user feedback, user testing	Selective User Involvement: Exploratory modelling, model validation, structured usability inspectors
Description of users, user characteristics	Models of user relationships with system
Design by iterative prototyping	Design by modelling
Varied, often informal processes	Systematical process
Evolution through trial-and-error	Derivation through engineering

Table 1. User-Centered vs. Usage-Centered

3. Results: a usability-driven approach to user interface design

As seen on the previous section, usability engineering has become fundamental in software development. Many approaches have been presented, each one with its advantages and its disadvantages. So, user-centred design guarantees users' satisfaction and provides real iterativity, while usage-centred design significantly simplifies all the development process. Our approach will try to benefit from the strengths of both models and to minimize their respective flaws.

The work presented in this paper is part of the MyHeart project (IST-2002-507816), an European research project whose aim is helping citizens to avoid or better manage cardio vascular diseases (CVDs).

Within MyHeart project, HF Management provides solutions for the continuous assessment of patients with cardiovascular diseases. When designing the application usage-centered design was the starting point, but some weaknesses were detected, most of them related with the user's experience and satisfaction.

Therefore, from user-centered design we will take the user and tasks model, meaning we will try to "know the users" and the tasks needed to accomplish their goals with satisfaction. Using the terminology of the Waterfall Lifecycle, we will state a **Requirements level** in which we will describe the possible users and the tasks to make during the utilization of the system. Three models will be needed in this phase: the **use cases** will describe user tasks accurately, describing the system in terms of functional requirements; the **task cases** will be an ordered set of activities and actions the user has to perform to achieve a concrete purpose or goal. In order to systematize this process, we will apply Norman's theory of action, which states a cyclic model of human-computer interaction. With this methodology we will split every tasks into a set of essential actions needed to achieve a concrete goal that will change the state of the system in a noticeable way. As we are dealing with chronic patients, it will also be necessary to know some specific characteristics of the potential users of the system (e.g. possible range of ages, social conditions, etc), what will lead us to define an **user model**, that will help us to refine the task model via a *cognitive walkthrough* (CW). The cognitive walkthrough is a usability inspection method in the form of a cognitive task analysis [6] which has been applied to the study of usability and learnability of several distinct medical information technologies. The purpose of a CW is to evaluate the cognitive processes of users performing a task, and it is used to determine whether the user's background knowledge and abilities are likely to be sufficient to produce the correct goal-action sequence required to perform a task. With all this information we can state a **quality model** that will define the guidelines to follow during all the development process.

The next step will be the **Analysis Level**, where the **Domain Model** is performed. The model consists of two diagrams: a **Sequence Diagram**, which will be an UML-like model that will represent the system behavior, and a **Roles Model** that will define the structure of the classes that will take part in the associated sequence diagram together with the relationships among these classes, specifying the role of each one of them.

At this point, there is still no graphical representation of the final user interface. In user-centered design this is the point where prototypes are manufactured. Instead of that, we will focus on the advantages of the usage-centered design paradigm, developing **abstract models** that will simulate the characteristics of the final interface, allowing us to go back to the first stage if errors are detected in a faster and

more systematical way. This phase will be called **Design Level**.

Finally, in the **Implementation Level** we will develop the Presentation model that will describe the concrete interaction objects composing the final GUI. At this point, we will take into account some aspects that have been ignored until now, such as the final implantation platform.

Figure 1 shows the different stages:

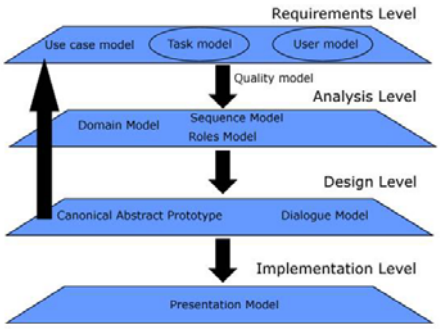

Figure 1. Model Stages

4. Conclusions

Usability has become a competitive advantage for most software development companies, and it is especially important in health care applications due to the particular characteristics of the potential users of the system. HF Management is a system for assessing patients with chronic heart diseases, usually elder people with other chronic illnesses and cognitive impairments. That is why usability is a must, it is extremely necessary to avoid any possible mistake due to usability problems, and that is why potential users have been involved since the early stages of the development process.

5. Acknowledgements

The "My HF Management" work is a complete integrated system part of the MyHeart project, "Fighting Cardiovascular Diseases by prevention and early diagnosis" (IST-2002-507816). MyHeart is a 6th Framework Project of the IST programme, partly funded by the European Commission.

6. References

[1] J. Nielsen. Usability Engineering. Morgan Kaufmann, 1993

[2] Mayhew, D.J. The Usability Engineering Lifecycle. Morgan Kaufmann, San Francisco, CA. 1999

[3] Costabile, M.F. Usability in the Software Life Cycle. Handbook of Software Engineering and Knowledge Engineering. World Scientific Publishing, pp. 179-192. Singapore, 2001.

[4] Constantine, L. L. and Lockwood, L. A. D. Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design. Reading, MA: Addison-Wesley. 1999

[5] Constantine, L. L. and Lockwood, L. A. D. Usage-centered engineering for Web applications. IEEE Software, 19 (2), March/April 2002

[6] Polson P, Lewis C, Rieman J, Wharton C. Cognitive walkthroughs: a method for theory-based evaluation of user interfaces. Int J Man-Machine Stud 1992

Address for correspondence

Ignacio Peinado Martínez
Life Supporting Technologies
ETSI Telecomunicación. B-303-1
Ciudad Universitaria s/n.
Madrid 28040.
e-mail: ipeinado@lst.tfo.upm.es