IT Licentiate theses
2001-001

Design and Usability In Telemedicine

ERIK BORÄLV

UPPSALA UNIVERSITY
Department of Information Technology

UPPSALA UNIVERSITY

Design and Usability in Telemedicine

BY
ERIK BORÄLV

February 2001

DEPARTMENT OF HUMAN-COMPUTER INTERACTION
INFORMATION TECHNOLOGY
UPPSALA UNIVERSITY
UPPSALA
SWEDEN

Dissertation for the degree of Licentiate of Philosophy in Human-Computer Interaction
at Uppsala University 2001

Design and Usability in Telemedicine

Erik Borälv
Erik.Boralv@hci.uu.se

Department of Human-Computer Interaction
Information Technology
Uppsala University
Box 337
SE-751 05 Uppsala
Sweden

http://www.it.uu.se/

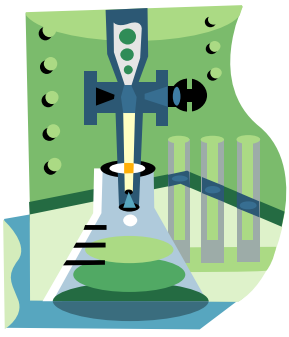
ISSN 1404-5117
Printed by the Department of Information Technology, Uppsala University, Sweden

Abstract

A design of computer systems, that effectively supports the user, is the major goal within human-computer interaction. To achieve this, we must understand and master several tasks. These tasks concern firstly what to develop and secondly how to develop the system.

The design and implementation of effective and efficient user interfaces is a prerequisite for the successful introduction of computer support in the medical domain. We base our work on a fundamental understanding of cognitive aspects of human-computer interaction, as well as on a detailed analysis of the specific needs and requirements of the end users, i.e. the medical professionals.

This thesis presents several approaches for development of systems for computer-supported work in health care. The solutions described concern vital problem areas: (1) the focus on the work tasks to be performed, (2) the cost of software and the way competition works in a networked world. Solutions to these problems can lead to more usable systems from a user's perspective but may also change the nature of computer applications.



Acknowledgements

First I would like to thank my supervisor Bengt Sandblad.

During the majority of the work I have been working on site with fellow researchers at the Medical and Biological Informatics department of the German Cancer Research Center in Heidelberg, Germany. There are many persons involved, but I have to say thanks to a few. Prof. Dr. H.P. Meinzer, for all of the invited evenings and the longstanding cooperation; Dr. Uwe Engelmann, for rewarding travels around Europe and the late night activities; Andre Schröter, for all the technical expertise; ex-member Dr. Athanasios M. Demiris, for all the squash games and for the idea to mix olives in the A-stuff. Much appreciated.

I would like to thank my colleagues at my department, that I have always enjoyed working with. I would especially like to mention Bengt Göransson, Eva Olsson and Mats Johnson.

I would also like to thank all other involved project members. In recent years especially Andreas Järund and Dr. Lars Edenbrandt.

This work was supported by the Swedish National Board for Industrial and Technical Development (NUTEK) and the Advanced Informatics in Medicine Program of the Commission of the European Communities.

Publications

This thesis is based on a number of research activities, parts of which have previously been published. This is a list of the relevant papers, and notes on my contribution to them.

[1] Domain Specific Style Guides - Design and Implementation. Olsson E, Göransson B, Borälv E, Sandblad B, Proceedings of the Motif & COSE International User Conference, Washington D.C. 1993, pp. 133-139.

[2] Usability and Efficiency - the Helios approach to development of user interfaces. Borälv E, Göransson B, Olsson E, Sandblad B, Computer methods and programs in biomedicine, supplement volume 45, December 1994, pp. 47-64.

[3] Teleradiology System MEDICUS. Engelmann U, Schröter A, Baur U, Schroeder A, Werner O, Wolsiffer K, Baur HJ, Göransson B, Borälv E, Meinzer HP. In: Lemke (Ed). CAR `96: Computer Assisted Radiology, 10th International Symposium and Exhibition, Paris. Amsterdam: Elsevier (1996) pp. 537-542.

[4] Experiences with the German teleradiology system MEDICUS. Engelmann U, Schröter A, Baur U, Werner O, Göransson B, Borälv E, Schwab M, Müller H, Bahner M, Meinzer HP. Computer Methods and Programs in Biomedicine 54 (1997) pp. 131-139.

[5] A Teleradiology System Design Case. Borälv E, Göransson B. Conference proceedings of Designing Interactive Systems 1997, ACM's Special Interest Group in Computer-Human Interaction (SIGCHI) in co-operation with the International Federation for Information Processing (IFIPWG 13.2), Amsterdam, 18-20 August 1997. ISBN 0-89791-863-0, pp. 27-30.

[6] Requirements for a new generation of Personal Digital Assistants intended for medical use. Borälv E, Engelmann U, Schröter A, Schwab M, Meinzer HP. Gell G, Holzinger A, Wiltgen M (eds). From PACS to Internet/Intranet, Information-Systems, Multimedia and Telemedicine - EuroPACS 2000. Wien: Österreichische Computergeesellschaft (2000) pp. 246-251.

[7] The WeAidU project – first experiences. To be submitted.

[8] The WeAidU Design Case. To be submitted.

Specification of my contribution

Paper 1

This paper describes how to develop a Style Guide for a specific context of use. It discusses what domain knowledge is and how that can help us to develop better systems. My contribution to this paper was to describe how to relate graphical components (widgets) to the domain. The main contribution to the project and this paper, was the design and implementation of domain specific components.

Paper 2

This paper is a continuation of paper 1 and describes in a larger setting how to use a Style Guide to implement a medical engineering environment. It defines a basic model of

development that will better use the domain specific aspects of the Style Guide. My role here was to define how the domain specific components fit into a development environment.

Paper 3

This papers outlines the basic functionality and rationale behind Teleradiology in general, and the MEDICUS system in particular. The core User Interface concepts and functions were designed by Bengt Göransson and me.

Paper 4

This paper introduces in more detail the ideas behind the Teleradiology system MEDICUS. My part here consisted of the core application scenarios.

Paper 5

In this paper the underlying design process is presented in the form of a method. This paper was co-written with Bengt Göransson. It introduces the concept of Design Patterns (or Design Criteria). It describes how criteria and requirements were transformed into a physical interface design.

It further shows how the concept "Work Task" (as defined in paper 1 and 2) can be used as the primary metaphor when developing a system.

Paper 6

This paper introduces the concept of a portable Teleradiology application and the requirements on such a system. My role in this was to develop the questionnaire and to do the analysis of the collected data. I also took part in defining the usage scenarios.

Paper 7

This paper introduces the notion of an on-line 24-hour diagnostic support and briefly describes how one can build such a support system. The project called WeAidU is presented.

My role in this project was the conceptual idea that decision support systems should be distributed for free, and that revenues should be collected by other means than software license fees. I also designed the User Interface and infrastructure for the WeAidU application.

Paper 8

This paper is close to paper 5 in its intendment. It describes the rationale behind the WeAidU application's look and feel. It presents the "Design Manifest" as introduced by key persons in the Human-Computer Interaction field.

It further describes a scenario for developing computer applications for computer-supported work. This scenario affects two vital parameters: (1) the focus on the work tasks to be performed, (2) the cost of software and the way competition works in a networked world.

Introduction

Human-Computer Interaction

Although the desktop computing revolution has greatly increased the range of possibilities, most users and developers would agree that getting a computer to do what they particularly want is as challenging as ever. Despite the successful introduction of personal computers to practically all professions, the daily experience of using computers is still one of emotional distress or tension. There simply is a world full of poorly designed programs that lack usability in the most elementary way. If usability aspects are not actively considered during development, then they are likely to not be regarded at all. Proper usability never happens by pure chance [15].

> Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them.
>
> Association for Computing Machinery's (ACM) definition of Human-Computer Interaction

Designing computer systems that effectively support the user during work, or decision-making as the case most often is, is the major goal within Human-Computer Interaction (HCI). To achieve this, we must know and master many tasks: we must know what to develop, but also how to develop the system. A working system alone is not enough, because the system is going to be used by a large number of different people with different needs and capabilities. The conditions of work are likely to change, and the surrounding world around the system and its users is definitely going to change. In addition, it is not a question of technical issues alone. Social aspects also play a vital role in any complex work situation.

Dividing the overall development task into smaller pieces begins with being able to understand how users think and perceive information and how people co-operate during work. First then the development process starts. Much of this initial work in HCI is treated by the underlying disciplines, for example cognitive psychology and software engineering. Those fields could be described as *objective* ones, where measurable goals and facts are in focus. Examples of such facts can be the human mind's capabilities in terms of memory capacity, the information flow before decision making, the error rates at a given speed of processing etc. Much of the knowledge in this area forms the basis and theory for the rest of the related HCI research areas.

What follows next is how to apply these basic facts and considerations in practice in order to be able to develop computer applications and systems with high usability [16]. Software design embraces many aspects: function, safety, human interface, ergonomics, graphics, algorithms, and data structure. Correspondingly, these various aspects of software design invariably have an impact on one another. The development is equally about what to develop and how to develop it. This *how* is addressed by paying attention to the way software is developed. The solution is to involve end-users and other stakeholders earlier and in more stages of the development process. The proposed solution combined with modern development strategies, such as prototype-driven development, produces a better final result [12].

Medical Informatics

The practice of medicine is to a great extent an information-management task. A physician's decision-making is based upon expert knowledge, information from the individual patient, and information from many previous patients, the latter known as experience [24]. Decision-making is often very difficult due to the fact that not only is the required expert knowledge in each individual medical field enormous, and growing daily, but, the information available for the individual patient is multi-disciplinary, imprecise and very often incomplete [4].

> Medical Informatics is both an Art and a Science.
>
> Science: where methods are conceived and experimentally validated by means of computer models and formalisms.
>
> Art: where computer processing systems are built and assessed.
>
> Handbook of Medical Informatics [3]

Medical Informatics is located at the intersection of information technology and the different disciplines of medicine and health care. The role of Medical Informatics is the common set of problems and solutions that both relates to medicine and computer science.

Telemedicine and telemedical applications

The introduction of telecommunication (Internet, Mobile Phones, and Wireless Communication) has already changed the way we communicate with each other—not only in emergency situations but also for everyday routine communication. This change will also affect the way we communicate in our professional life.

In medicine, and particularly within its highly specialized areas, the access to domain experts is limited for many practical reasons. It is also the case that access to medical data—such as electronic medical records, patient images, and laboratory results—is often limited by physical properties. In general, the main difficulty is that medical data is not available or accessible at some point: because it is not physically there, because the expert is not where the data is, or that data cannot be accessed without time-consuming manual procedures [23].

In order to address some of these issues, much effort has been invested in making medical data "portable" by storing it in computers instead of on film plates and paper. This initially resulted in a plethora of storage formats that still was as useless as paper was before. With an increased general computer maturity and international attempts at defining medical standards (for terminology, medical record data structures, protocols for data exchange, etc) things are improving and Telemedicine is rapidly showing gains over previously used procedures [1].

The user and the importance of utility and usability

In health care, as in many other work situations where computerized information systems are used, the purpose of the work performed by the involved professionals is never to

operate the computer. The computer is a tool that will be accepted and used only as long as it efficiently supports the efforts to provide good health care for the patient.

This means, among other things, that the user interface to the information system must be designed to optimize the health care work activities as such and not only to optimize the handling of the computer as a tool [22].

> Usability = The extent to which a product can be used by specified users to achieve goals with effectiveness, efficiency and satisfaction in a specified context of use.
>
> Effectiveness = The accuracy and completeness with which users achieve goals.
>
> Efficiency = The resources expended in relation to the accuracy and completeness with which users achieve goals.
>
> The definition of Usability according to ISO 9241

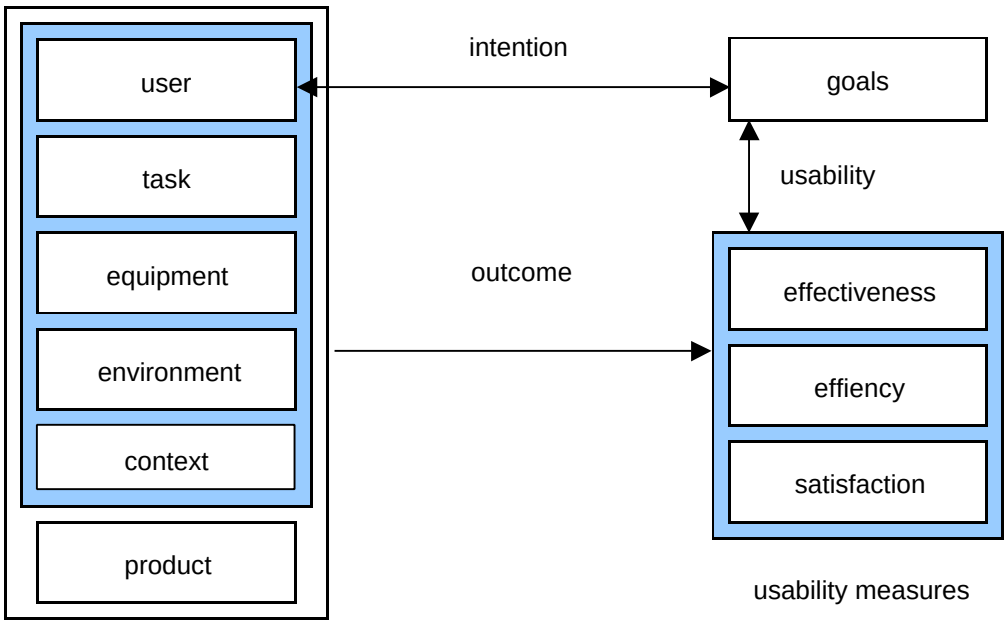


Figure I. The ISO 9241 usability framework.

In Figure I we see how the framework for measuring usability looks like.

Objectives

Central problems

The most central problem is that the medical world is still not computerized enough. This is primarily not because of lack of resources, even if this is an important factor too. Instead, it is the difficulty to know what to computerize and–perhaps even more difficult–how to computerize, that is the big issue.

Simply moving current tasks to the "digital" domain does not automatically solve any problems or make our work more efficient. In fact, some computerization has had the opposite effect when the selected solution, for example, has been inappropriate, or when not all aspects of the current situation has been taken into consideration.

The reason this happens is because the situation in the medical domain is often too complex to be reduced into a manageable set of constraints that are possible to implement in a computer system.

There are many reasons for this complexity:

- A hospital is full of professionals with very advanced skills. Their skills match the current situation and so reflect what they are required to be able to handle in the setting. Change the setting and the organization's entirety of skills will adapt to the change!

- A hospital is made up of numerous smaller and independent departments. Sometimes they extend over each other in terms of activities, sometime not. Each department have their own solutions and technology used to solve their tasks.

- The medical activity in itself is complex, both to understand and describe logically. Much of the internal knowledge is based on facts; some knowledge is based on experience and some on instinct. The health care process is very different from medical science. The former is more related to the art of medicine, whereas the latter is closely connected to the academic aspects and the biological disciplines of medicine.

This is the first of the central problems (knowing what the problem is) but the next step is equally difficult. Because, even if one knows the problem and how the problem arises, it is still difficult to find a working solution. The proposed solution also has to show real gain over the existing solution, because the existing solution will always be the first alternative in any competing situation. The focus of the medical organization is to treat patients, and if a solution works (but possibly unsatisfactorily in some sense) it is still likely to remain in use. The health care system is complex and making a transition from a known and working solution towards an only possibly better but definitely unfamiliar system is not a trivial step.

The problems we are trying to solve

A prerequisite for successful interface design is the understanding of the tasks the users need to perform. To manage this, a designer or developer has to be well acquainted with

the target domain. That includes having a fundamental knowledge of the task to be performed and familiarity with the work situation.

To accomplish an efficient interface, time and effort must be spent on an analysis of the task, the work contents and of the utilization of information. In short, the application only has to do two things: provide the right actions at the right moment, and to show a sufficient amount of information. If this can be accomplished we are in a good position to find a satisfactory solution [17].

But, it is always the case that the available screen space will be less than desired. We are also never sure of what the next action should be, or exactly what information to display. Therefore information has to be organized in a structured way so that we are reaching the best compromise between all the things that affect the system's overall usability.

Research methods

The main targets for our department's research has always been *real users*, in *real settings*. The focus is on the end user's situation (as opposed to the contractor's situation, the buyer's role, etc) and the daily work at hand. This is usually a task filled with many small and delicate problems. Problems that if solved would bring a major overall gain compared to today's situation. To summarize, we focus on today's work problems and how to solve the most immediate ones. This is in contrary to many other typical activities in the HCI area that look into the future and tries to find completely new directions of problem solving, or tries to introduce new forms of technology to solve the problems. For us, HCI is not about radical innovation but rather about evolutionary improvements.

Another tradition within our department is to take an active role in the projects where we are doing research. We are not merely providers of HCI expertise but more regular co-workers from an outside point of view. In the ideal situation we take part from the earliest phases, follow the initialization of the project, and then join the development teams as the project evolves. This has been the case in all my projects; I have been a member of the project groups from the first point were nobody in the group really knows what to develop, until the end when the product finally is launched and evaluated.

The strategies in the different projects have been slightly different, dependent of many things. In some projects there are many involved parties that control the freedom of choice. Time constraints also influence the possibilities to choose direction and methodology.

Research approaches

As a rule, in HCI, there are three predominant ways to conduct research: experimental studies, survey studies and observational studies [18].

Experimental design is used to control all variables and vary those that are being tested, and the strength of the experimental study is its ability to unambiguously localize an effect in a particular design. The method allows the study of isolated design factors, but requires a situation where variables are controlled, something that is not always feasible or possible.

Survey data are useful in describing systems, for detecting strong and weak points, and for suggesting improvements. Surveys, or questionnaires, provide a structured approach in which the user assesses factors related to the subject of our study. User assessments

are introspective and are subject to biases but it is possible to establish, through empirical verification, reliability and validity of user responses. Furthermore, when comparisons are made between different groups of subjects rather than with an absolute criterion, it may not matter that responses are biased.

Observational studies are easy to conduct but are open to interpretation. In an observational study, one or several systems may be selected and researchers observe users. Analysis may be at a purely verbal descriptive level or based on quantitative measures. Conclusions are tentative at best since researchers have little or no control over conditions interacting with the system [19].

For the conditions we work in (live development projects) it is often only practicable to perform surveys and observational studies. The major part of my work has been done as observational studies, complemented with surveys whenever possible. The experimental approach is not used because of the nature of the work. It would be impractical to repeat a development project with the same preconditions.

Testing different ways to develop medical applications with a focus on the HCI aspects in a laboratory setting is not doable. This makes it difficult to make traditional analysis of how well a method performs. For economical reasons it is also not possible to repeat a project with new methods in order to isolate the effect of the methods.

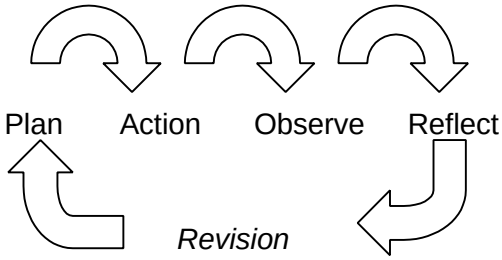
Action Research

The research done has an obvious qualitative approach, since all of our work is done within running projects. The first reason for doing qualitative research is that one wants to find deeper knowledge (the *why* and *how* questions), which is not practicable to measure in numbers. It is also a reflection of the fact that it is inside the development projects "where it's at", the source of knowledge. We often label the approach as *Action Research* (AR), but without defining or adhering to some particular instance or interpretation of AR (though *participatory action research* probably is close to our approach) [11].

Action Research has its academic roots in sociology, social psychology, psychology, organizational studies, and education. In AR literature one finds a varying degree of rejection of the classical notion of scientific research. This is a reminder of AR's origin where resolution of theoretical issues were of less importance, and the solution of social and organizational problems was regarded as more important. The main objective being that most things are done within groups and the insight that working in groups have a fundamental effect not only on the total outcome but also on the individual member of the group. Action-oriented research is said to involve generation of situation-specific knowledge, not mere application of some pre-existing knowledge.

According to Hopkins [8] action research can be described as an informal, qualitative, formative, subjective, interpretive, reflective and experiential model of inquiry in which all individuals involved in the study are knowing and contributing participants. Action research has the primary intent of providing a framework for qualitative investigations in complex working situations [13]. It consists of four stages, all repeated over the project duration: reconnaissance & plan, action, observation, and reflection & revision (see Figure II).

Figure II. Action Research Stages



During a project it is however common to fall back on quantitative methods, whenever possible, as they serve as a complement to the free form of observations that otherwise would be the only source of facts.

Results

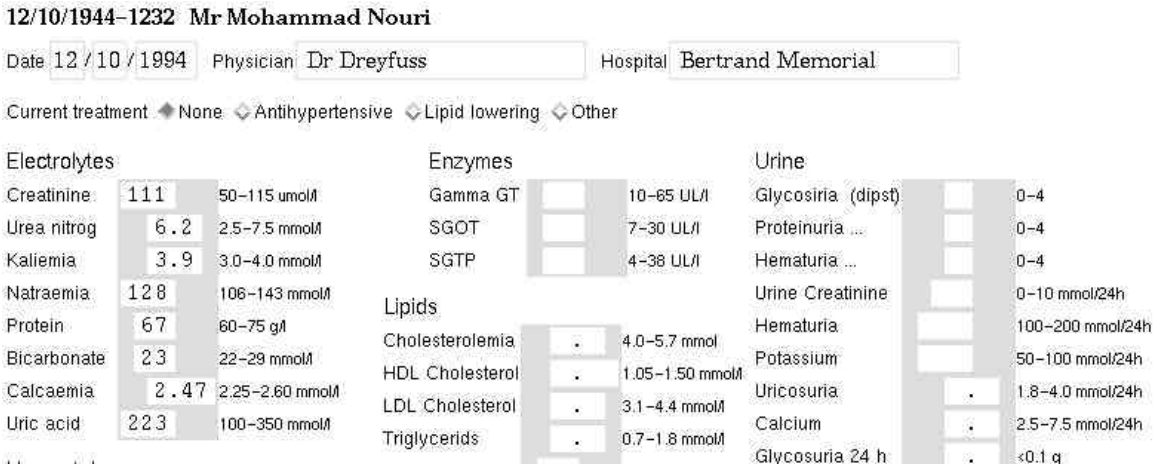
We have tried different approaches to reach the goal of a working system. Each of the projects described by the included papers has a different solution. The various approaches are mostly the result of our own reflection and increased experience. To some extent they show the varying needs of the projects.

Helios (paper 1 & 2)

Project description

In Helios the overall task was rather complex from a development point of view. The goal was to build a very advanced software tool that in turn was going to be used to construct a number of medical applications. The project contained a number of teams of developers and general stakeholders.

Figure III. Design example showing part of an interface element for a clinical test form



Solution

The solution suggested by our team was to produce *static knowledge* that could be agreed upon and finalized once and for all. The solution would finally form a collection of guidelines that would help a developer looking for design support. This knowledge base included basic cognitive facts, on how humans work in interaction with computer systems. It is quite possible to find vital and static knowledge here–we will always have a certain long-term memory capacity for example.

This static knowledge was combined with knowledge about the target domain (medical systems). The knowledge about the domain came from previous experience, field studies and trials. All of this was then combined into a *medical style guide*. The idea was that it would be possible to make something similar to a cookbook, which a future developer could read and use as a practical guide when developing a new medical system. Since some parts of the target domain are static we could even do design work that could be described as static. As an example, the medical domain will always be centered on patients. Therefore we are able to produce a number of designs and implementations that

will deal with frequent tasks that relate to patients. The patient card (with basic information about a patient: name, sex, age, etc) could be designed before any specific application development had started. In Figure III we see examples of static design that can serve as a basis for following development of additional forms.

This reasoning could be applied further, and small common components (called *widgets*) could be developed to better suit the target domain. Examples of such widgets are text entry fields in which the input can be verified before accepted as valid input, and navigation controls that could hold and control many objects of a similar type (patient cards for example).

MEDICUS/CHILI (paper 3, 4, 5 & 6)

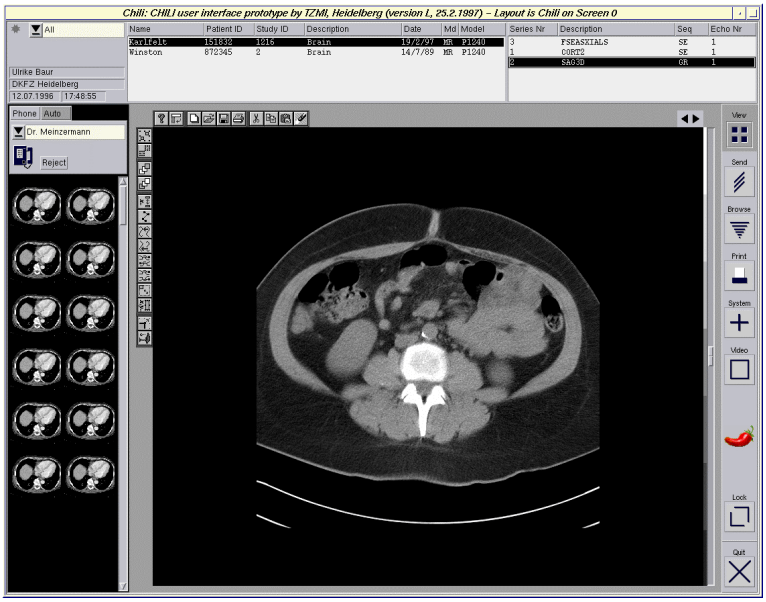
Project description

CHILI is the successor of MEDICUS. Both applications were developed for the same medical purposes, and virtually by the same group of people.

The context in the MEDICUS project was very different, compared to the sizable project Helios. MEDICUS and CHILI both had a small and well-acquainted group of predominantly software developers. The goal was again complex, a teleradiology station with possibilities to on-line communication and application sharing. It contained many conflicting requirements.

The main use of the application would be image viewing for interpretation and consultation purposes. Teleradiology is a means of electronically transmitting radiographic patient images and consultative text from one location to another. The final design proposal before application development started can be seen in Figure IV below.

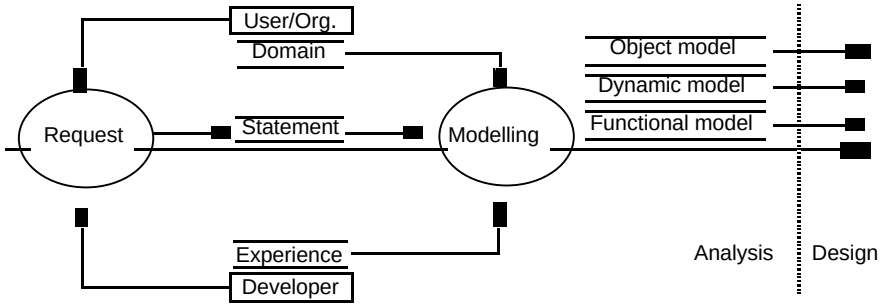
Figure IV. The original CHILI design



Solution

The approach here was to focus on things that were the most important according to some ranking. We call these *criteria*, and the whole process *criteria-based design*. The selected strategy was partly influenced by the current methods *a la mode*: Object Modeling Technique [20] and Design Patterns [7]. OMT was a predecessor to Rational Unified process [9].

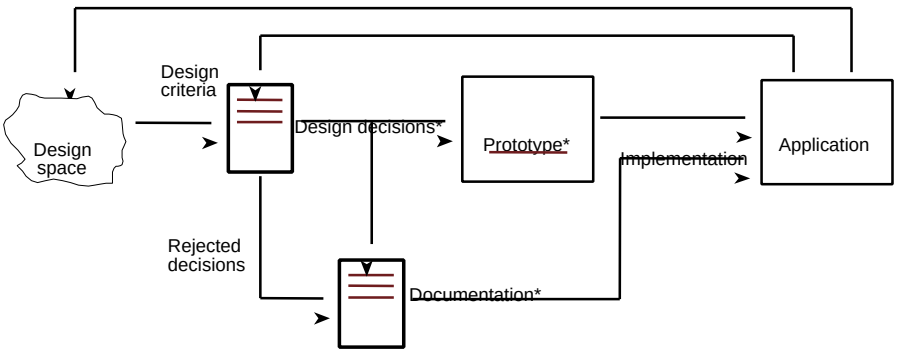
Figure V. The OMT model overview



Design patterns are tested, good, working solutions that are possible to generalize so that there is a well-defined rule. The following is an example or joke from the (dull?) world of typesetting: "Whenever in doubt, use Caslon." First there is a problem statement (we don't know what font to use), and then follows the general solution (Caslon apparently works in all those situations).

We then modified the OMT model (see Figure V) to better suit our purposes, which were not primarily to construct the system, but to specify and design how the system should work. The outcome ought to be a design proposal but as well contain the rationale for the proposed solution. The resulting model is seen in Figure VI.

Figure VI. Re-engineered OMT model



The model deals with how to manage conflicting options and how to preserve the knowledge about the selections that are made. The result is both a prototype and documentation. The resulting artifacts are marked with a '*'.

A short explanation of the method is as follows:

- The process is initiated with an initial set of design criteria. The whole team must agree on these criteria.
- During design, solutions are drawn from a very large set of possible solutions, the design space.
- If a proposed solution is consistent with all other chosen solutions and it is supported by the design criteria it is implemented in the prototype. The solution is transferred to the design documentation at the same time.
- If a solution is rejected, the proposed solution, plus the reasons why it was rejected are moved to the design documentation.
- The final prototype plus the design documentation serve as a basis for the development team as they start implementing the application.
- Formal and informal evaluations of the application both influence the original criteria and the design space.
- The process is repeated until all necessary functionality is incorporated in the design.

WeAidU (paper 7 & 8)

Project description

The aim in the WeAidU is to develop an Internet-based intelligent system providing near instantaneous professional aid in clinical decision making. The system involves artificial neural networks trained to interpret myocardial perfusion scintigrams. This makes it possible to enhance the effectiveness of clinical decision making when it comes to coronary artery disease. The system is designed to work on the Internet and uses Java technology to facilitate deployment and maintenance [6].

The following images show two generation of the same decision support. In Figure VII we see the first attempt which closely follows the previously outlined strategy with Work Tasks as the guiding metaphor. The second solution in Figure VIII shows a more traditional solution with pull-down menus. The visual change is largely dependent on a change of the application's functionality. The two solutions will enable a future comparison of the effect of the different styles of user interfaces.

Figure VII. The current production version of the WeAidU application

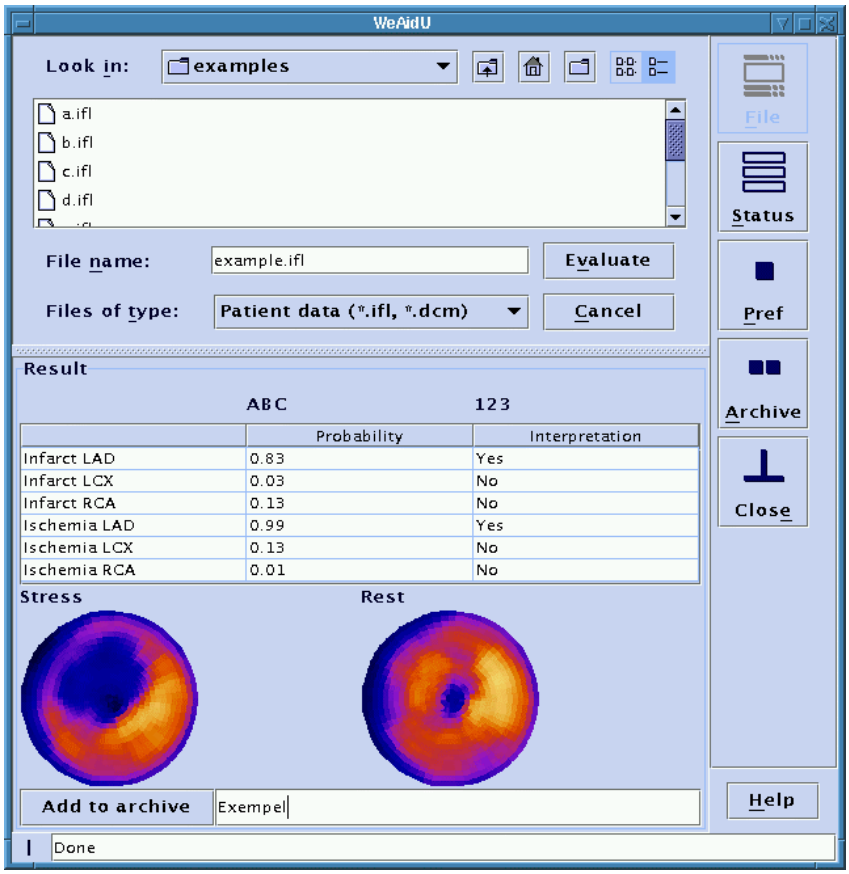
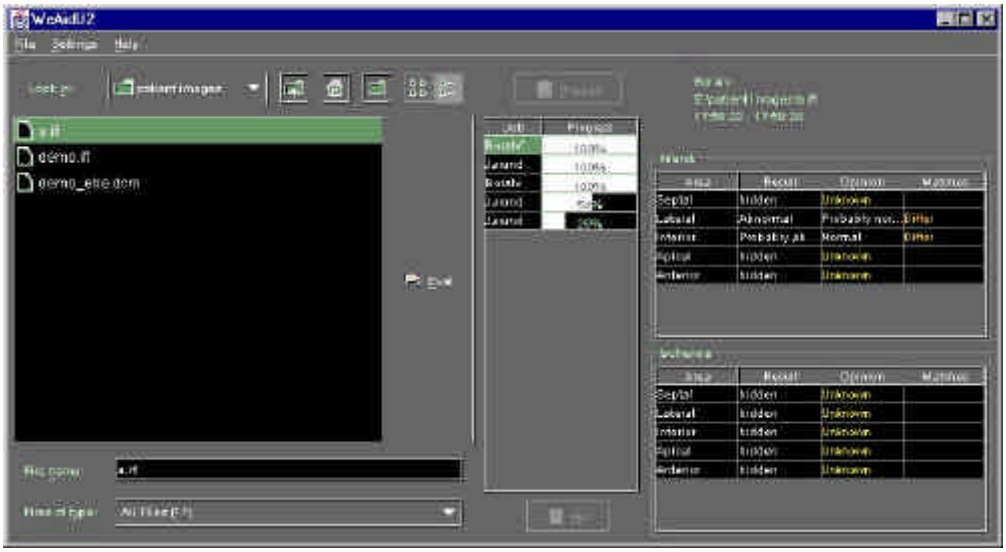


Figure VIII. The future version of the WeAidU application



Solution

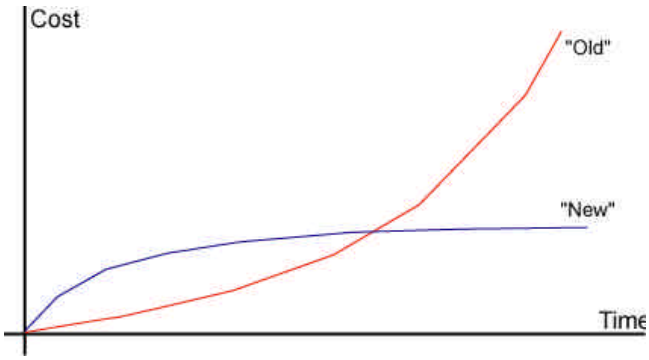
The WeAidU approach is an experimental strategy. It is based on the notion that we cannot initially know exactly what to develop and therefore we do care about that! The process has the characteristics of "finding out the requirements as you go". The reason for this is that we actually want a change, and a preliminary study of today's situation may not bring us to an appropriate solution [5]. It is also the case that the application is unique and therefore we have no fixed goal, just vague ideas to build on. The aim is to produce something that does not exist today.

The method is an interactive as well as an iterative process where the repeated cycle consists of the following steps:

- Abstraction
- Structure
- Representation
- Detail

Abstraction includes the study of the target domain and forming some kind of an elementary understanding. That rough understanding is then categorized into a structure that is possible to manipulate. A structure can include the relations between involved elements and users' interests. Then follows a transformation from the structure to a physical/visual representation that contains the most important properties of the desired solution. In an early stage of development the representation is often a prototype on some primal level of maturity. Afterwards details are added to the physical/visual representation [25].

Figure IX. Old and new way of estimating cost of changes vs. time



The technique described is similar in some aspects to the ideas found in *eXtreme Programming* (XP) [2]. One of the driving forces in XP is the insight that it is more efficient, both in the short and long run, to only solve the problems at hand, instead of solving all the possible problems that might arise. For a trained programmer/designer this can be very frustrating, as it is somewhat conflicting with the taught "golden standard" of developing the best possible solution (in which all similar problems are possible to solve with a single elegant solution). The reason why we can ignore a full study before starting development is because of the old "truth" about estimation of cost vs. time (see Figure IX): 'there is a small cost for early changes, and a large cost for late changes'. This statement does not hold for all forms of development anymore, not only because of new tools and more

powerful computers. If the whole development cycle is based upon change, and streamlined to handle changes (early or late ones) the result is a flattened cost curve.

Hardware advances and more suitable programming languages and tools help in this process but are not the most important issues. Instead it is the shift in methodology and attitude towards how software ought to be produced that can bring this effect.

Conclusion

Evaluation

In this thesis there are three described ways of developing Telemedicine systems with good usability. The first approach is based on formalized knowledge in the form of a domain specific style guide. The style guide contains a mix of objective HCI knowledge and heuristic guidelines. The second approach is based on the user interface designer's role: it presents a method of designing the user interface such that the knowledge and rationale behind the design is carried on through the whole development cycle. The third and latest effort is a heuristic method that redefines the whole approach to software development. Its standpoint is to utilize the economical and technical changes that software development has gone through the last decade. The possibilities are used to modify the development process so that late, and even continuous, changes in the development process are possible.

Guidelines

Style Guides and similar design tools (guidelines and standards) are very much requested by industry today. Some of the popularity is probably a result of the recent focus on quality assurance and the desire to develop systems according to pre-defined or well established methods. Furthermore, guidelines are high in applicability (as opposed to standards) and can be used from the very start of a project [14].

Even though guidelines primarily are based on accepted practice and provide a wide applicability, they are still the cause of some controversy. The question is whether guidelines really help in the process of developing usable systems. The main argument is that since guidelines are described on a general level (e.g. "strive for consistency") they still need some kind of interpretation to be of use in the current situation. But if an active interpretation is needed, then a developer without knowledge in HCI cannot make this interpretation correctly and the overall purpose of the guideline is not fulfilled.

Although there are limitations to the use and applicability of guidelines there are strong arguments for them. The question whether the result is improved by the use of guidelines is perhaps wrong: if the only source of HCI knowledge in a project is guidelines, then it is better than nothing at all.

Design methods

Improving usability measures by bringing forward an additional method is possibly contra-productive: part of the explanation why systems have poor usability is the lack of simple and clear development strategies. Adding yet another method to the plethora of existing rules may not bring us closer to the goal of more usable systems or efficient development.

The design method (the modified OMT model) described in this thesis is a conceptual one that tries to overcome the gap between the initial design and the final system. Having found out all the correct requirements and then achieving a good design is of little use if the final implementation does not regard these results. The major benefit from the proposed design method is to address this issue, so the purpose is not to introduce itself as another method as such. It is the preservation of both design and rationale that is important to sustain. Given the complexity of any user interface design much can be gained if underlying design knowledge can be maintained throughout the project. My work has made me convinced that if this kind of design decision can be made visible to all

members of the development group then the final system will be a lot closer to its original intended design and purpose.

Development methods

What if new computer languages and development tools delivered the long promised more efficient development cycles? What if it would cost the same to implement a feature on the first day as on the last day of a project? What if we were not restricted by hardware performance or software restrictions to only use the existing set of interface components? What if we are free to choose whatever solutions best suit our needs instead of the existing ones?

It is advantageous to make changes to the way software is developed but a lot in the software industry is based on tradition and proven practice. Possible changes are not always considered in a serious manner. We are not (yet) near solving all the questions listed above even if we are getting closer. The latest effort described in this thesis is an attempt to practice the proposals put forward by the eXtreme Programming movement.

To embrace change is not easy. It is not uncomplicated to ignore the uncomfortable feeling of not knowing what to develop and simply defer unresolved questions to a later moment when we might know more. A lot of effort is instead spent on developing functions answering questions like "wouldn't it be nice to be able to do this?" This is sometimes called "*creeping featuritis*"; the tendency for anything complicated to become even more complicated because of the fear of leaving things as simple as they can be. In this sense the attempted solution has perhaps not been a successful one, although much progress has been shown–mostly from the way software is meant to function in the future in the hospital setting.

In conclusion, bringing a change to the way software is developed and sold is a difficult task. The customary techniques have become so natural for us that it will take time to accomplish changes. It is not the scope of this thesis, but economical aspects play a vital part here. A mentality of sufficiency is good business, whereas a mentality of scarcity creates its own waste.

Future work

Problem statement

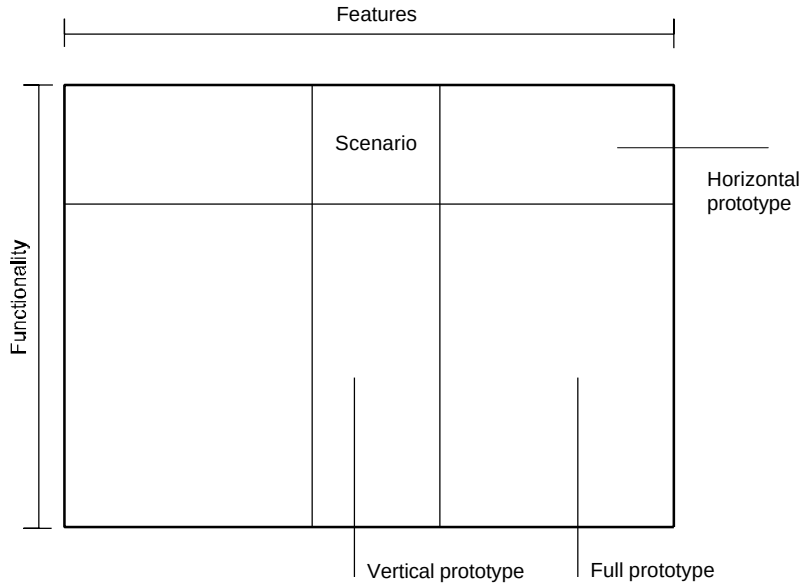
Embracing the whole development cycle in detail is truly a large task. Science can address well-formed problems but not, as Schön puts it, "messy, indeterminate situations" like those we often face in a real-world project [21]. The main problems with developing a usable system needs to be defined more accurately. What determines how the result is going to turn out is a key question. If this can be studied as a whole or if it needs to be divided into smaller parts needs to be addressed.

Still, a holistic approach would bring much more useful knowledge into the hands of those who need it. It is my sentiment that the whole is not always the sum of its parts. In this sense, a birds-eye view of the development process could bring about something more valuable and usable. From an industry point of view, methods and development techniques for usable systems are regarded as the most important issue within HCI research [10]. The focus of the work then has to be directed more at distributing knowledge instead of participating in the development process.

Strategy

The process of going from abstract requirements to a visual representation is a key target, as it involves much of what we believe affect usability of the final system. Exactly how to do this and what kind of methods to use is still not clear. It is very likely though that it must contain a prototype-oriented approach [25] that drives the process.

Figure X. Relation between different prototyping techniques



Using prototypes relieves one from the many intrinsic problems of software development. In Figure X we see some possible directions of using prototypes. For example, to only capture a work situation in a certain scenario, or to mimic the full functionality of some chosen feature in a vertical prototype. Apart from its traditional usage areas (i.e. testing and benchmarking), it can facilitate communication between stakeholders and serve as documentation of design decisions.

References

[1] Ball M.J, Douglas J, Garets D. Strategies and Technologies for Healthcare Information: Theory Into Practice. Springer Verlag. ISBN: 0387984429. 1999.

[2] Beck K. Extreme Programming Explained: Embrace Change. Addison-Wesley Pub Co. ISBN: 0201616416. 1999.

[3] Bemmel van J, Musen M. Handbook of Medical Informatics. Springer Verlag. ISBN: 3540633510. 1997.

[4] Berner S.E, Ball M.J. Clinical Decision Support Systems: Theory and Practice. Springer Verlag. ISBN: 0387985751. 1998.

[5] Cooper A. About Face: The Essentials of User Interface Design. IDG Books Worldwide Inc. ISBN: 1568843224. 1995.

[6] Eckel B. Thinking in Java. Prentice Hall Computer Books. ISBN: 0130273635. 2000.

[7] Gamma E, Helm R, Johnson R, Vlissides J, Design Patterns. Addison-Wesley Pub Co. ISBN: 0201633612. 1995.

[8] Hopkins, D. A teacher's guide to classroom research. Philadelphia: Open University Press. 1985

[9] Jacobson I. Object-Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley Pub Co. ISBN: 0201544350. 1994.

[10] Katzeff C, Svärd P.O. Användbarhet i praktiken. Stockholm, Sweden: Swedish Institute for Systems Development. 1995

[11] Kemmis, S. and McTaggart, R. The Action Research Reader. Third edition. Deakin University Press, Victoria, 1988.

[12] Laurel B. Art of Human-Computer Interface Design. Addison-Wesley Pub Co. ISBN: 0201517973. 1990.

[13] Lewin, K. "Frontiers in Group Dynamics: II. Channels of Group Life; Social Planning and Action Research," Human Relations (1:2), pp. 143-153.

[14] Olsson, E. Providing design knowledge to system developers by domain-specific style guides. Licentiate thesis, September 1999. IT/Human-Computer Interaction, Uppsala University.

[15] Nielsen J. Usability Engineering. Morgan Kaufmann Publishers. ISBN: 0125184069. 1994.

[16] Nielsen J, Mack R.L Usability Inspection Methods. John Wiley & Sons. ISBN: 0471018775. 1994.

[17] Norman D. A. The Design of Everyday Things. Currency/Doubleday. ISBN: 0385267746. 1990.

[18] Preece J, Rogers Y, Sharp H, Benyon D. Human-Computer Interaction. Addison-Wesley Pub Co. ISBN: 0201627698. 1994.

[19] Rapoport, R.N. "Three Dilemmas in Action Research," Human Relations, (23:4), 1970, pp. 499-513.

[20] Rumbaugh J, Blaha M, Premerlani W, Eddy F, Lorenson F. Object-Oriented Modeling and Design. Prentice Hall. ISBN: 0136298419. 1991.

[21] Schön D. A. The Reflective Practitioner: How Professionals Think in Action. Basic Books. ISBN: 0465068782. 1984.

[22] Shneiderman B. Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley Pub Co. ISBN: 0201694972. 1997.

[23] Shortliffe E.H, Wiederhold G, Perreault L, Fagan L. Medical Informatics: Computer Applications in Heath Care and Biomedicine. Springer Verlag. ISBN: 0387984720. 2000.

[24] Van Bemmel J., Gremy F, Zvarova J. Medical Decision Making. North-Holland. ISBN: 0444878408. 1985.

[25] Winograd T. Bringing Design to Software. Addison-Wesley Pub Co. ISBN: 0201854910. 1996.

Licentiate theses from the Department of Information Technology

2000-001 Katarina Boman: Low-Angle Estimation: Models, Methods and Bounds.

2000-002 Susanne Remle: Modeling and Parameter Estimation of the Diffusion Equation.

2000-003 Fredrik Larsson: Efficient Implementation of Model-Checkers for Networks of Timed Automata.

2000-004 Anders Wall: A Formal Approach to Analysis of Software Architectures for Real-Time Systems.

2000-005 Fredrik Edelvik: Finite Volume Solvers for the Maxwell Equations in Time Domain.

2000-006 Gustaf Naeser: A Flexible Framework for Detection of Feature Interactions in Telecommunication Systems.

2000-007 Magnus Larsson: Applying Configuration Management Techniques to Component-Based Systems.

2000-008 Marcus Nilsson: Regular Model Checking.

2000-009 Jan Nyström: A formalisation of the ITU-T Intelligent Network standard.

2000-010 Markus Lindgren: Measurement and Simulation Based Techniques for Real-Time Analysis.

2000-011 Bharath Bhikkaji: Model Reduction for Diffusion Systems.

2001-001 Erik Borälv: Design and Usability in Telemedicine.

UPPSALA
UNIVERSITY

Department of Information Technology, Uppsala University, Sweden