IT Licentiate theses
2001-006

Usability Design: A Framework for Designing Usable Interactive Systems in Practice

BENGT GÖRANSSON

UPPSALA UNIVERSITY
Department of Information Technology

UPPSALA UNIVERSITY

Usability Design: A Framework for Designing Usable Interactive Systems in Practice

BY
BENGT GÖRANSSON

June 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

Usability Design: A Framework for Designing Usable Interactive Systems in Practice

Bengt Göransson
Bengt.Goransson@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 1404-3203
Printed by the Department of Information Technology, Uppsala University, Sweden

Abstract

Today, many companies have started to become aware of the advantages of user-centred design. However, it is extremely rare that companies adopt a fully integrated user-centred design approach in one strategic shift. Rather, companies tend to adopt practices and methods in stages or adopt a particular method or practice only when a complex set of factors align to create readiness. There is a big market for companies vending usability methods, but poor usability of systems and products is still very common—the vendors often blaming it on factors outside their immediate influence. This, among other things, is a call for us to work for a user-centred design attitude as a major strategy for the system development process.

The content of this thesis is dedicated to the question of how to develop usable interactive systems in practice. Main focus is on how we can raise the awareness of usability; articulate the need for user-centred design within the industry and development organisations; and practice user-centred design. A framework for usability design as an unpretentious way of applying user-centred design is described and discussed.

For Ellinor and Carl, with love

Acknowledgements

Many thanks to my friends and supervisors, Bengt Sandblad and Jan Gulliksen, for helping me out with this thesis and guiding me in my research.

I would like to acknowledge and honour all my co-writers: Erik Borälv, Anders Frisk, Jan Gulliksen, Magnus Lif, Eva Olsson, Bengt Sandblad, Torsten Sandbäck and Vello Thomasson. Thank you for all your help in making this thesis possible, for rewarding discussions and lots of fun.

To all my colleagues and friends at the Department of Human-Computer Interaction at Uppsala University, I greatly appreciate your support and back-up. I owe Enea Redina, my friends and colleagues there, a deep and honest thank you for support, encouragement, as well as interesting and fruitful discussions. My friends at the Medical and Biological Informatics Division at the German Cancer Research Center (DKFZ) in Heidelberg, Germany: thank you for the HELIOS stuff and all the unforgettable evenings in Heidelberg.

I also wish to express my appreciation to my first tutor Mats Lind who introduced me to the subject of Human-Computer Interaction.

Thanks to all persons with whom I have been working at the Swedish National Tax Board (RSV), in particular: Kjellåke Henriksson, Esbjörn Franzén and Malin Pettersson. I also wish to mention a few others I have had the opportunity to work with: the development team at Pharma Point, especially Uno Liljedahl; the members of the Dalton project at Amersham Pharmacia Biotech; and Annika Hadenius at Enea Business Software.

I would like to express my humble admiration to all participants in the research and development projects I have been involved in during my professional career, and above all, to all users out there!

The Swedish Council for Work Life Research (RALF) has partly financed my research, thank you.

Finally, my love goes to my family Ellinor and Carl—you made it possible.

☺

Preface

This thesis has two segments: the first with a background and summary of my work; followed by one with papers representing the scientific result of my work. The papers are as follows:

i. **A User-Centered Approach to Object-Oriented User Interface Design.** Jan Gulliksen, Bengt Göransson and Magnus Lif (2001). Book chapter in: *Object Modeling and User Interface Design*, edited by Mark van Harmelen, Addison-Wesley, ISBN 0-201-65789-9.

ii. **Reengineering the Systems Development Process for User-Centred Design.** Jan Gulliksen and Bengt Göransson (2001). Paper accepted to be presented at the Eighth IFIP TC.13 conference on Human-Computer Interaction: INTERACT'01, Tokyo, Japan, July 9-13, 2001.

iii. **Usability designers improve the user-centred design process.** Bengt Göransson and Torsten Sandbäck (1999). In the proceedings of the IFIP TC.13 International Conference on Human-Computer Interaction: INTERACT'99, vol. II, pp. 163-167, Edinburgh, UK, August 30 – September 3, 1999.

iv. **The Design of a Smart Card-Based Home-Help System.** Anders Frisk, Bengt Göransson, Torsten Sandbäck and Vello Thomasson (2001). Paper accepted to be presented at the Eighth IFIP TC.13 conference on Human-Computer Interaction: INTERACT'01, Tokyo, Japan, July 9-13, 2001.

v. **A Teleradiology System Design Case.** Erik Borälv and Bengt Göransson (1997). In: Gerritt van der Veer, Austin Henderson, Susan Coles: *Designing Interactive Systems: Processes, Practices, Methods and Techniques*. DIS'97 conference proceedings of the ACM Special Interest Group in Computer-Human Interaction (SIG-

CHI) in co-operation with the International Federation for Information Processing (IFIP WG 13.2), pp. 27-30, Amsterdam, August 18-20, 1997.

vi. **Usability and efficiency. The Helios approach to development of user interfaces.** Erik Borälv, Bengt Göransson, Eva Olsson and Bengt Sandblad (1994). In: U. Engelmann, F. C. Jean and P. Degoulet (Eds.), *The HELIOS Software Engineering Environment*, Supplement to Computer Methods and Programs in Biomedicine, 45, pp. 63-76.

Reprints were made with permission from publishers where applicable.

Author's guide to the reader

I do not expect the reader to read this thesis from the first page to the last, unless you are particularly interested in the subject. The summary itself is worthwhile reading for those interested in usability and user-centred design in general. The first four chapters are more of a background, while in chapters 5 and 6 I formulate some new ideas and draw conclusions from my research results. Chapter 7 is a short description of the papers included, and in chapter 8 I give a few hints about my future research direction.

Regarding the papers in the second part of the thesis: there is an informal grouping of the papers into three groups. If your main concern is the development process, read: *i*, *ii* and *iii*. If you are more interested in detailed design issues and prototyping, read papers: *iv* and *v*. If you want to know more about the subject of Human-Computer Interaction and design principles read paper number *vi*.

About my co-writers

Erik Borälv. Post-graduate student at the Department of Human-Computer Interaction, Information Technology, Uppsala University.

Anders Frisk. Developer and consultant at Pharmasoft AB.

Jan Gulliksen. Associate professor at the Department of Human-Computer Interaction, Information Technology, Uppsala University.

Magnus Lif. Chief Human-Computer Interaction Officer at Icon Medialab London. Earlier at the Department of Human-Computer Interaction, Information Technology, Uppsala University.

Eva Olsson. Post-graduate student at the Department of Human-Computer Interaction, Information Technology, Uppsala University.

Bengt Sandblad. Associate professor and Head of the Department of Human-Computer Interaction, Information Technology, Uppsala University.

Torsten Sandbäck. Developer and consultant at Enea Redina AB.

Vello Thomasson. Developer and consultant at Jaybis AB.

TABLE OF CONTENTS

PAPERS i–vi

SUMMARY

– The high-tech industry is in denial of a simple fact that every person with a cell phone or a word processor can clearly see: Our computerized tools are too hard to use. The software engineers who create them have tried as hard as they can to make them easy to use and they have made some minor progress. They believe that their products are as easy to use as it is technically possible to make them. As engineers, their belief is in technology, and they have faith that only some new technology, like voice recognition or artificial intelligence, will improve the user's experience.

Alan Cooper in "The inmates are running the asylum: Why high-tech products drive us crazy and how to restore the sanity", 1999, p. 15.

1 Introduction

The areas of Human-Computer Interaction (HCI), User-Centred Design (UCD) and usability have been recognised and taken more seriously by the industry during the last couple of years. There are several reasons for this, but one important and tangible reason is—the rise and impact of the Internet and the World Wide Web (web). New companies focusing on the Internet (e.g. the "dot-com" companies) have, among other things, another mix of skills in their development organisations. Exposing services on the Internet, such as the web, to the general public has called for a new awareness of usability and has put focus on the interaction between users and the services provided via a device such as a computer. Companies and organisations that are now starting to incorporate HCI knowledge, user-centred design activities and usability competence into their system development processes do not do this for charity reasons, they see a commercial interest in developing systems that are usable: more users—more money; more satisfied users—more money; more productive users—more money etc.

There is really an interest from the industry to start to use methods and processes that have the potential to produce more usable interactive systems. This has been described in several independent reports over recent years. One example is the report by Katzeff & Svärd (1995). Their investigation told us that the Swedish industry in general was not very mature when it came to the usability aspects of a system. However, the companies studied were very interested in starting to learn more about this area. The development of methods and processes together with the integration of usability-related activities into their development process was highly ranked.

Through my research and applied work I have observed that companies and organisations approach the challenge of developing usable interactive systems in different ways:

- ❖ A majority do not see the need to dedicate themselves to usability. They see the usability of a system as something that can be taken care of with minor hands-on activities such as adding some graphics to the user interface.
- ❖ Others start to incorporate user-centred activities into their current development process on a more regular basis. They will still use whatever development process they are using, but add some activities.
- ❖ Very few companies and organisations are willing to go the whole way and adopt a truly UCD process. These companies and organisations realise that being committed to usability means that there must be a shift in the way that they develop interactive systems. They take a stance and commit themselves to a user-centred philosophy.

Dray and Siegel (1998, p.16) discuss this in an interesting article: "…companies tend to adopt UCD practices and methods in stages or adopt a particular method or practice only when a complex set of factors align to create readiness". When companies and organisations now explore the areas of HCI and usability, many of them will undoubtedly fall into traps already visited by earlier adopters of these areas. Such a trap is the very teasing attempt to just focus on the user interface of the system instead of taking the whole spectrum of usability into account. Another common approach is to continue to focus on, e.g. technology and not realise the need for UCD. Usability will always be suffering if the development is technology-centred. Alan Cooper describes this with a lot of insight and humour in his book: *The inmates are running the asylum* (Cooper, 1999). The development must be focussed on what is most accurate and usable for the user and not just what is technically possible to develop: "Programmers trade simplicity for control. They exchange success for understanding. They focus on what is possible to the exclusion of what is probable" (Cooper, 1999, p. 93).

There are a lot of other traps to fall into, many of them related to the fact that HCI, UCD and sometimes even usability are areas that still are relatively unknown to the industry, and not fully understood. One reason for this is that HCI traditionally is a rather academic field, adopted mostly by rather large in-house organisations, and focussed on research. But, as usable systems now are increasingly accepted as a competitive advantage, the industry will explore HCI, UCD and usability and make the community of HCI act in a more pragmatic and humble way. By the HCI community I mean all of us who are researchers or professional HCI practitioners.

There is still a long way to go before HCI knowledge; the concept of UCD and the activities ensuring usability are fully integrated in the industry's development processes. In the meantime I and my fellow researchers and developers try as hard as we can to make this happen.

My work, as described in this thesis, is dedicated to the task of providing knowledge and experiences on how to develop usable interactive systems in practice. It is my experience and belief that to achieve usable interactive systems, you have to apply HCI knowledge and use a user-centred design process. To be clear about what I mean by an interactive system, I use the definition of this term as defined in the international standard ISO[1] 13407 *Human-centred design processes for interactive systems* (1999). An interactive system is a: “combination of hardware and software components that receive input from and communicate output to a human user in order to support his or her performance of a task. NOTE The term ‘system’ is often used rather than ‘interactive system’”. I will use the terminology system and interactive system interchangeably throughout the summary of this thesis.

I apply my research and work in the broad context of system development, and user-centred design can be seen as an approach to interactive system development that focuses specifically on making systems usable. Although much of my work can be applied to products such as consumer electronics or appliances, my focus is on interactive systems used through a computer by professionals as a tool in their daily work.

In this summary I will describe and discuss the areas of: HCI, usability and UCD. Further on, I will introduce a framework for putting usability and UCD in practice that I call *Usability Design*. This framework builds on results and experiences from my research and practice.

2 Research objectives

> *– Action research combines theory and practice (and researchers and practitioners) through change and reflection in an immediate problematic situation within a mutually acceptable framework. Action research is an iterative process involving researchers and practitioners acting together on a particular cycle of activities, including problem diagnosis, action intervention, and reflective learning.*
>
> Avison, Lau, Myers and Nielsen, “Action Research”, in Communications of the ACM, January 1999, Vol. 42, No 1, p. 94.

A lot of users doing professional work and using interactive systems as tools or aid have from time to time difficulties in using these tools, in one

[1] ISO is the label used for standards coming from the International Organization for Standardization.

way or another. This can be related to a number of reasons, many of them obviously linked to poor usability of the systems. This has been reported in numerous reports, books etc. over the last decade (extensively reported by, e.g.: Jakob Nielsen on his web site http://www.useit.com/ and in his books, e.g. "Usability Engineering" (1993a) and lately "Designing Web Usability" (2000); Alan Cooper in "The inmates are running the asylum" (1999), etc.). Even if there is no total figure available stating the overall usability of all systems throughout the world, we can read about it in HCI literature, in papers, listen to it at conferences, hear indisputable gurus talk contemptuously about it, hear users talk about it with fatigue, experience it ourselves, etc., and today even see it on the TV news. Even further, maybe the worst thing is all those systems that we do not hear about, used within companies and organisations where the users do not have a choice. In many situations people are forced to use certain systems as a part of their daily work, no matter how poor the usability is. Here is just one small example of usability-related problems from "real life". In a recent article in a Swedish medical journal, physicians complained about the computerised medical record. One physician claimed that a disadvantage with the computerised medical record is that it does not give a sufficient overview of information. He was forced to go to different parts of the system to get enough information to make a decision. Even worse is the fact that the poor overview made him less confident in the decision-making. He estimated that the use of the computerised medical record prolonged his working day by one hour (Dagens Medicin, 04/01). This kinds of usability problems are by no means new. Our department at Uppsala University has for decades (example is: Allard, Lind, Sandblad and Schneider, 1984) done research on the topic of making the computerised medical record as usable as possible, but still systems as the one mentioned are developed.

I have asked myself the question—how can I as a researcher and practitioner improve this situation and supply the users with systems that are *usable*? My modest answer to this question is that I have to be present in the situations where systems are designed and built. And when I am in that position, influence and guide the development organisations and the individual projects to be more user-centred. I can do this by introducing methods, processes and practices, and have a *dialogue about usability*, so that all participants are aware of the importance of taking usability seriously. Further on, as a researcher I can reflect in practice and communicate my results to others for further utilisation, and hopefully make some change in the industry that is beneficial for the users.

In my research the practical applicability is very important, I would therefore like to clarify my view on the definition of practice: "**Practice.** A technical or management activity that contributes to the creation of the output (work products) of a process or enhances the capability of a process" (ISO/TR 18529:2000(E)). Practice is obviously connected to some sort of activity carried out, preferably by a skilled practitioner[2]. This relation is important since it is my belief that the role of the practitioner is of vital importance for both the outcome and the performance of a process.

2.1 Research scope and my aims

As my main aim and research problem is wide-ranging and kind of pretentious—how to develop usable systems in practice, I realise that I have faced a formidable task. This is, of course, more of a vision and can probably act as my main research objective, but has to be more detailed to be really useful. So, what I do is to focus on the problems and areas that I have found most urgent, based on my and others' experiences from several organisations, companies and projects.

A central, monolithic problem (as far as usable systems are concerned) is to efficiently integrate usability aspects and a user-centred perspective into system development processes.

[2] Practitioner is: "One person who practices an art or profession", Webster's student dictionary, 1997.

This problem can be broken down into a set of interesting research topics and areas that I use as guidance in my research:

- Make companies and organisations aware of the importance of developing usable systems.
- Put usability on the industries' agenda. Usability is still the underdog in development projects and in business strategies.
- Find out (develop) "easy to use" techniques: methods, activities, processes, project roles, etc. Many organisations have problems in applying existing methods and processes. Although people in organisations have heard about usability, they do not know how to achieve it.
- In the long run make the industry realise the potential in adopting a user-centred design approach. Starting with "usability in every project".
- See how there can be more usability skilled developers and practitioners in the system development industry. There are too few today. I think this is a logical conclusion to the fact that so many systems have a questionable usability.

The nature of these topics is that they are rather big issues, kind of imprecise and maybe not detailed enough. However, they indicate the breadth of my research. I am deliberately focussed on the breadth since I believe that this is the key to the question: *how to develop usable systems in practice.*

I have to make a remark about my standpoint on making usable systems a reality. I confront the problem from a system development perspective. This is, of course, not the only way to deal with the problem. But it is my natural position as a computer science graduate and also one of the main perspectives within HCI. Other researchers face the problem from their point of view. One example is to educate or inform the purchaser to be more aware of how to order or purchase usable systems. This is an interesting perspective, but not in my primary research focus. My key target group is the development organisation. However, I believe that much of my work can be of benefit to other stakeholders, such as people working with requirements.

RESEARCH CONTEXT. I work part time as a developer/consultant at Enea Redina and part time as a researcher at Uppsala University. This condition is of vital importance for the understanding of my work and my research approach. I have chosen to go both these ways in the total confidence that they benefit from each other, and that, by having this approach, I can make the most impact.

I have been involved in projects where I have had the possibility to carry out my research and practice at, e.g., the following organisations and companies over the last couple of years: The Swedish National Tax Board (RSV), The Swedish National Social Insurance Board (RFV), Swedish Medical Products Agency (Läkemedelsverket), Pharma Point, AffärsData, Dagens Nyheter, Amersham Pharmacia Biotech, Telia and German Cancer Research Center (DKFZ). They all represent organisations with quite different characteristics. They develop systems in different contexts. Some have in-house organisations for systems development, others are product companies and some buy development resources on a contract basis. They have all added values to my research and contributed to the breadth.

2.2 Research methods and work practice

Action research as a research method makes it possible for the researcher to apply his/her theories in practice, in a realistic work situation, take action and make a change in that situation. Action research is not like controlled experimental research with fixed parameters in a laboratory setting. Instead, action research projects are conducted in real life projects at the practitioner's work place[3].

My research approach falls into the category of action research. Action research is a methodology that has the dual aims of action and research (Dick, 1993):

- *Action* to bring about change in some community or organisation or program.
- *Research* to increase understanding on the part of the researcher or the client, or both (and often some wider community).

The mix of action and research can be tuned to the level that is accurate for the researcher's aims. One important aspect is that the researcher takes part in the studied situation, for instance a project, not just as an observer but also as a fully participating project member.

In the article *The History of Action Research*, Janet Masters (1995) describes the fundamentals of this research school. She summarises the history and background of action research into four basics themes: empowerment of participants; collaboration through participation; acquisition of knowledge; and social change. I try to apply the essence of these themes in my research. The empowering of the developers and the users in development projects is extremely important, as well as full participation on equal terms of all involved in our project teams. This is not an easy task since resources, time, etc. usually are very limited. To fully understand and take on a user-centred design view is usually, among other things, a major attitudinal and social change for a development organisation. This is not a change or revolution that happens fast or easily, on the contrary it is an evolutionary process that takes years. The acquisition of knowledge is

[3] More experimentally focused researchers take the position that results that have not been acquired in a laboratory are of no scientific value and therefore useless. Here I will not attempt a discussion about what can be valued as scientific or not, and simply point out that there are other research methods that are fully accepted, such as action research that contributes to the overall wisdom and insight in how to develop usable systems. Laboratory studies are very useful but, from my perspective it is equally important to go out in the field and study how people actually work, how systems are developed, etc.

part of the learning process that takes place within every organisation (or at least in most of them). For every project, knowledge is added to the organisation's knowledge base. It is the researcher's responsibility to reflect upon, analyse and generalise the results from a project and to put it forward as gained knowledge and communicate it to others for further utilisation.

The process that the researcher goes through to achieve these themes is a spiral of action research cycles consisting of four major phases: planning, acting, observing and reflecting (Masters, 1995). The figure below tries to put this cycle into a project context. As the researcher goes through all steps in the cycle, developers usually come in when the project starts, and move on when the project is finished. The researcher starts the research in advance by planning, then participates in the project and continues after the project to reflect and analyse (see Figure 1). This, among other things, distinguishes the research from pure consultant or development work, whereas the developer only participates in the execution of the project.

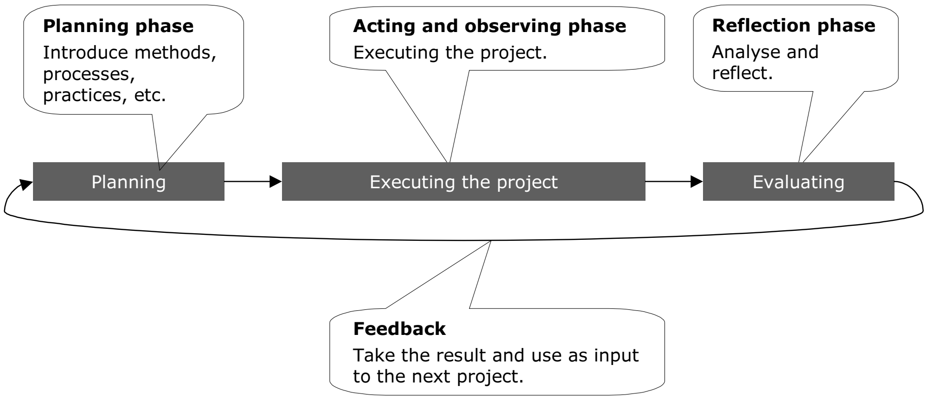

Planning phase
Introduce methods, processes, practices, etc.
Acting and observing phase
Executing the project.
Reflection phase
Analyse and reflect.
Planning
Executing the project
Evaluating
Feedback
Take the result and use as input to the next project.

Figure 1: A cyclic process of: planning, acting, observing and reflecting.

Reflection is a very important ingredient in action research. It is reflection as a dialogue between a number of researcher-participants; or between two researcher-participants; or even in its most extreme form the reflections of a single researcher-participant. Action research is either research by practitioners, or at least researchers collaborating with practitioners, engaged in a systematic attempt to understand and change practice by their own actions and reflection upon those actions—reflective practice (Webb, 1995).

The nature of action research is qualitative measurement. This in perspective to the more common quantitative measurement-based methods in HCI research—hard figures, like formal experimental research in psychology. Within our projects we use various methods to collect data such as questionnaires, field studies, observations, interviews, video studies, etc. The communicative outcome of my research is usually a paper describing a case, reflections, findings, etc. They are usually descriptive and aimed to describe why we did something; what we did; how we did it; and the lessons learnt.

More useful information on action research can be found in the on-line journal *Action Research International*, `http://www.scu.edu.au/schools/gcm/ar/ari/arihome.html`.

Practitioner-centred research

Lately I have been increasingly concerned with the question: How can we raise the impact of research in professional practice? Parts of this can be taken care of by conducting action research as discussed above, but I want to go further.

A school called Practitioner-Centred Research (PCR), related to action research has lately inspired both my research and myself. It has its background in the insight of the limited contribution of research to professional practice (Bourner & O'Hara, 1999). PCR comes from areas such as higher education, practice of law and medical practice. I can see that there are similarities in the way that I practice HCI in organisations and in the way, e.g. a lecturer teaches a class. We are both concerned about communicating knowledge in such a way that the apprentice can put it into practice.

The interesting thing about PCR is the mixture of professional practice and research. Bourner and O'Hara discuss this topic in terms of PCR located within the domain of the researching professional rather than the professional researcher—it is one thing to share new knowledge, but another thing entirely to share new practice. The sort of new knowledge that practitioners need if they are to adopt a new practice is both objective, an answer to the question "*can it work?*" and subjective, an answer to the question "*can it work for me?*".

I work both as a practitioner and researcher and have the opportunity to use existing methods, and partly developing new system development methods and processes, putting them into practice and using them as activities in the development process. In this way I get the chance to study, analyse and reflect upon the true value of the proposed course of actions or procedures. As a result of these studies I can further improve the methods and practices to formulate new theories and so on. This is an iterative process that has the potential to engage all parties involved in a system development project. The result of such a project is not only the developed system, but also knowledge and experiences about the process itself and the practice of it.

Reliability and validity of my research

Reliability and validity are important aspects of research in order for the reader to assess the results. Can we repeat the same "experiment" and get the same results? Can we be sure that we are "measuring" the right thing, without systematic errors? My answer is that we cannot possibly run two or more projects in parallel, or in sequence with exactly the same environment, context or presumptions. This is just not feasible as we conduct our research in clients' real projects. We have to plan, take action, observe and reflect in one project, taking into account that project's specific context. This will build up a knowledge base that can be utilised in the next project, etc. In this sense, we live as we learn: improvement by iteration (see the chapter *User-Centred Design*). It is hard for me to generalise the results from one project and say that it is true for all other projects. It is in the nature of action research that the reader of the results must bear in mind the presumptions for action research, the specific project and make her or his judgement based on that—*can it work* and *can it work for me*?

3 Human-Computer Interaction

> *– 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.*
>
> ACM Special Interest Group on Computer-Human Interaction (SIG-CHI) Curriculum Development Group, 1992, section 2.1.

To give a background to the area or subject of HCI I would like to take the definition above as a stance. This definition is, for instance, used and discussed in the two predominant course books in HCI classes in Sweden: Dix et al. *Human-Computer Interaction* (Dix, Finlay, Abowd & Beale, 1998) and Preece et al. *Human-Computer Interaction* (Preece, Rogers, Sharp, Benyon, Holland & Carey, 1994). The expression human-computer interaction was adopted during the mid-1980s, recognising HCI as a research and subject matter of its own. HCI had by then evolved from the more limited areas of user interface and man-machine interface, commonly used during the 1970s. By introducing the expression HCI, the area was accepted as being broader than just the design of the interface and was recognised to be concerned with all those aspects that relate to the interaction between users and computers. If we want to dig deeper in the origins of HCI and its ancestors, we find that the areas of ergonomics and human-factors can be considered to be the mothers of HCI. Ergonomics and human-factors are more or less synonyms. Ergonomics mostly used in Europe and human-factors used in North America (Dix et al., 1998, p. 2). They both concern studies of machines, systems, humans and the effects on the working environment and human performance. The Second World War boosted the interest in these fields as the war combatants sought after more effective weapon systems. As computers became more used and widespread, researchers started to specialise in the interaction between people and computers. Leading up to the current notion of HCI.

There is no single interpretation or understanding of the subject of HCI. But the perhaps most accepted definition is the one by the ACM SIGCHI

Curriculum Development Group cited in the ingress of this chapter. The diverse nature of HCI is explained as the curriculum group discusses the content of HCI: "Human-computer interaction is concerned with the joint performance of tasks by humans and machines; the structure of communication between human and machine; human capabilities to use machines (including the learnability of interfaces); algorithms and programming of the interface itself; engineering concerns that arise in designing and building interfaces; the process of specification, design, and implementation of interfaces; and design trade-offs. Human-computer interaction thus has science, engineering, and design aspects."

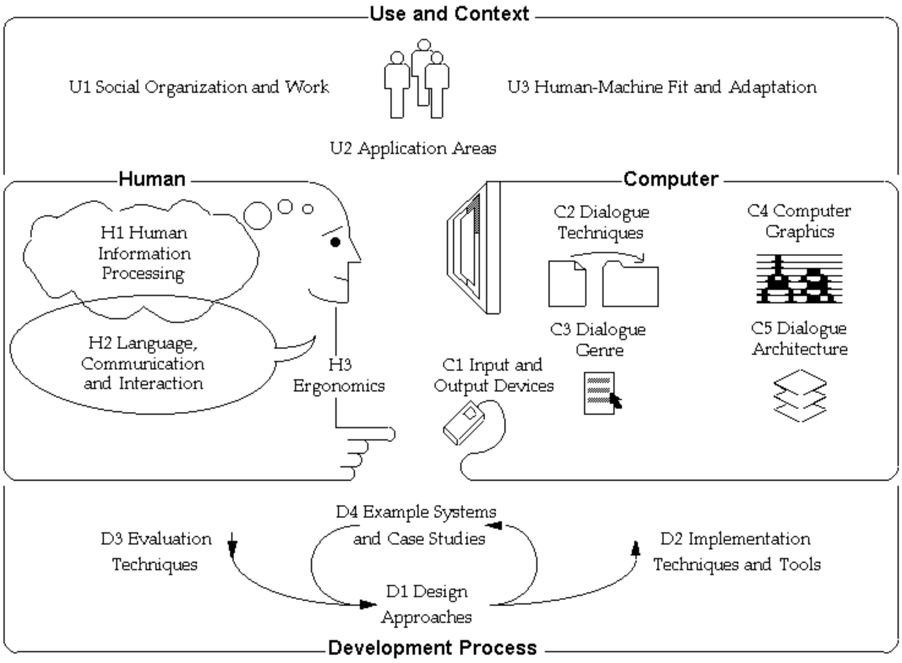

Use and Context
U1 Social Organization and Work
U3 Human-Machine Fit and Adaptation
U2 Application Areas
Human
H1 Human Information Processing
H2 Language, Communication and Interaction
H3 Ergonomics
Computer
C2 Dialogue Techniques
C4 Computer Graphics
C3 Dialogue Genre
C5 Dialogue Architecture
C1 Input and Output Devices
D4 Example Systems and Case Studies
D3 Evaluation Techniques
D2 Implementation Techniques and Tools
D1 Design Approaches
Development Process

Figure 2: The content of Human-Computer Interaction according to ACM SIGCHI Curriculum Development Group, 1992.

There are four major areas within HCI: Use and Context (U), Human characteristics (H), Computer (C) and Development Process (D). To successfully develop an interactive system we must get to know the intended users of the system: their background, capabilities and constraints, and also their working environment. Much can be known by understanding human behaviour, but we must also know about the specific users and their environment and tasks, etc. Further, we must know about the technical possibilities and limitations, development tools, etc., and we must have a process or a framework to guide our development. This sounds quite simple, but turns out to be a big challenge.

When it comes to the disciplines involved in HCI the ACM report states: "Because human-computer interaction studies a human and a machine in communication, it draws from supporting knowledge on both the machine and the human side. On the machine side, techniques in computer graphics, operating systems, programming languages, and development environments are relevant. On the human side, communication theory, graphic and industrial design disciplines, linguistics, social sciences, cognitive psychology, and human performance are relevant. And, of course, engineering and design methods are relevant." (ACM Special Interest Group on Computer-Human Interaction (SIGCHI) Curriculum Development Group, 1992, section 2.1).

3.1 Discussion about HCI

HCI is without doubt a multi-disciplinary subject. One can, however, argue how the different disciplines have influenced the subject. Both the ACM SIGCHI Curriculum Development Group and Dix et al. discuss HCI from a computer science perspective. They argue for HCI being taught in computer science classes, programmes, etc. From a computer science and systems design perspective, HCI can be seen as a subject that "…involves the design, implementation and evaluation of interactive systems in the context of the user's task and work." (Dix et al., 1998) Whilst others have more of behavioural science approach, e.g. Preece et al. It is undoubtedly so that computer science and behavioural sciences have had the most impact on the HCI community. There is nothing controversial about that, it is just interesting to reflect upon it, and maybe relate some of the problems for HCI to really make an impact in the industry, to the fact that HCI is a rather scientific subject buried deep in theories of computer systems and human behaviour, and not that easy to apply in an industrial environment. It is only recently that HCI has been more adopted by the industry and for that matter by a wider group of students. This can be observed, for instance, by the growing attendance, and number of companies represented, at the annual Computer-Human Interaction conference (CHI, organised by ACM) and the increasing number of students taking HCI classes at our department at Uppsala University. This positive trend is appreciated and should be taken as a signal that there is a real interest in the matters that HCI covers. Part of this is due to the impact of the web-related companies, now exploring the potential and limitations using the Internet. Companies trying to start business over the Internet have realised the need for HCI knowledge when setting up, e.g. their web sites. What we, the HCI community (researchers and practitioners), have to do is to make HCI more applicable for those willing to adopt the fundamentals of the field.

The approach within the HCI community is to be objective and precise. I think that this is a logical conclusion when looking at it from the perspective of computer science and behavioural sciences. However, can we still rely on the objective and precise knowledge provided by subjects as HCI when we are dealing increasingly with concepts such as user experiences, entertainment, games and fun? Even though HCI is a multidisciplinary field it lacks much of, e.g. UCD practice aspects in the education of the subject and the applied research. As HCI is rather scientific it has over the years been hard to fully gain an understanding within the subject about the importance of disciplines that are more of a creative and craft nature, such as graphic design, industrial design, etc. HCI is much closer to computer science and engineering than to crafting a design—the character of design and role of the designer is underestimated. This is, of course, noticed and discussed by others previous to me, such as, e.g. Winograd et al. (1996) in the book “Bringing Design to Software”.

HCI is traditionally a subject concerned with studies of how humans interact through computers or machines, and how this interaction can be improved by applying certain methods and techniques, being the results of academic research and focussed on computer science and behaviour science aspects. Although this is perfectly adequate it is also necessary to be a little bit more pragmatic and find methods, processes and practices that can be directly applied in commercial and industrial development projects. What I am trying to say here is that it is almost always impossible to use methods exactly as they are described in the literature. There are always certain circumstances for a specific project that makes it necessary to improvise or to have a slightly looser attitude. There is rarely any room for being fully scientific when working in industrial projects. You have to be prepared to “sacrifice” a bit of your methodical and maybe scientific ambitions on the altar of commercialism. Nevertheless, the research field of HCI is the very foundation of knowledge when dealing with the development of usable interactive systems.

4 Usability

–The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

ISO/IS 9241-11 Guidance on usability.

Is it important that an interactive system is usable? Most people would reply to that question with the rhetorical question: are not all systems usable? There is a contradiction in that one could say that presumably all developed systems are supposed to have a high degree of usability, but indeed they have not. The usability of a system is generally implicit and taken for granted. When a customer purchases or orders a system, he or she assumes that the developing organisation develops a usable system. This makes perfect sense from a customer perspective, but is unfortunately far from the truth. In many cases, the customer has to explicitly order a usable system by defining goals or requirements for the usability. This is something that is neither easy nor something that purchasers are acquainted with. The usability of an interactive system is extremely important when it comes to professionals using interactive systems as tools in their daily work. The systems have to be effective, efficient and satisfactory to use in order to support the users in their work activities. Focus must be on the professionals accomplishing their work, and not on the intrinsics of the system itself. In a real life work situation the professionals should be focused on their work activities and not be forced to struggle with a system that is not suitable for them.

Today, computers and computer systems are a natural part of almost every professional's work environment. Statistics from Statistics Sweden (SCB)[4] show us that in 1999, 65 % of all people employed in Sweden used a computerised tool (60 % of them used a personal computer). 32 % used computerised tools more than half of their working day. Just imagine if we could make some of the systems a little bit more usable. Let's say that, on an average, we could make the 65 % computer-users spend 10 minutes less per day struggling with their computers. Knowing that there were approximately 4 200 000 people employed in Sweden in December 2000 (employment figures from Statistics Sweden), we are talking about saving more than 10 million work days per year, not to mention the probable improvements in working environment and gains in better physical and mental health among all those people. The consultant company Cap Gemini made an investigation interviewing 975 persons. This investigation showed that a PC-user, on average, uses 2 hours and 22 minutes a week, trying to solve computer related problems (figures from the Swedish economic magazine *Veckans Affärer*, March 23, 1998). So, I think that my hypothetical example is not unrealistic.

[4] Statistics available from the web site: http://www.scb.se./, I have used table 1 of "Working environment for employed persons by sex and age. Per cent. 1999" (accessible as a pdf-document).

4.1 Defining usability

To be more explicit about how to achieve usable systems we need to define and understand what usability is. I prefer to use the term usability as defined in the ISO/IS 9241 standards on “Ergonomic requirements for office work with visual display terminals (VDTs), Part 11 – Guidance on usability”:

> *The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.*
>
> ISO/IS 9241–11:1998(E)

The different parts of the definition is additionally defined as:

Effectiveness: Accuracy and completeness with which users achieve specified goals.

Efficiency: Resources expended in relation to the accuracy and completeness with which users achieve goals.

Satisfaction: Freedom from discomfort, and positive attitudes towards the use of the product.

Context of use: Users, task, equipment (hardware, software and materials), and the physical and social environment in which the product is used.

I, and others, have found the ISO definition of usability valuable to use as it makes it possible to discuss usability with a common understanding on the matter. Further on, the definition stresses that usability is measurable. We can measure the usability and say that for a certain user in a certain context, e.g. system X, is more usable than system Y, or the usability for system Z has improved 50 %. The ISO definition covers more of the vital aspects of what is important for the users than what is commonly understood when discussing usability in general. I often hear people discuss usability in vague terms such as: “user friendly”, “easy to use” or as something that is only related to the user interface and graphics of an interactive system. But, that is only part of the scene. We must get to learn the potential users, their goals, their tasks, context of use, etc., in order to be able to develop a usable system.

Sometimes the term usability is referred to as the capability of a system or product to be used “easily”. This corresponds with the definition of usability as a software quality. An example of this is the ISO/IEC 9126 (concerned with quality in software) definition of the term usability: “A set of attributes of software which bear on the effort needed for use and on the individual assessment of such use by a stated or implied set of users”. However the attributes which a product requires for usability depend on the nature of the user, task and environment. *A product has no intrinsic usability*, only a capability to be used in a particular context. Usability cannot be assessed by studying a product in isolation. I find this declaration, taken from the ISO/IS 9241-11 standard on usability important.

4.2 Usability goals

It is fundamental to set up some kind of goals of usability, as argued for in ISO/IS 9241-11. By defining usability goals we have the possibility to both guide the design of the system and to measure the usability. The usability planned for a system can be defined, documented and verified. Among other things, this gives us the opportunity to use the definition of usability as a tool for focusing on usability in projects and use it to define methods, processes and practices to achieve usability.

ISO/IS 9241-11 states: “The usability of products can be improved by incorporating features and attributes known to benefit users in a particular context of use. In order to determine the level of usability achieved, it is necessary to measure the performance and satisfaction of users working with a product. Measurement of usability is particularly important in view of the complexity of interactions between the user, the goals, the task characteristics and the other elements of the context of use. A product can have significantly different levels of usability when used in different contexts.”

When specifying or measuring usability, the following information is needed (ISO/IS 9241-11):

- A description of the intended goals;
- A description of the components of the context of use including users, tasks, equipment and environments;
- Target or actual values of effectiveness, efficiency and satisfaction for the intended contexts.

Each project should preferably describe the goals of use for the system. Goals may be decomposed into sub-goals that specify components of an overall goal and the criteria, which would satisfy that goal. An example of an overall goal may be that a person at a biochemical laboratory wants to "Identify a protein". This overall goal might be decomposed into sub-goals such as: "Make a correct gel"; "Load a slide into Maldi". The level at which the overall goal is set is a function of the boundary of the system which is under consideration and which provides the context of use.

Usability measures of effectiveness, efficiency and satisfaction can be specified for overall goals or for narrower goals. Examples of measures of usability include (ISO/IS 9241-11, appendix B):

- Percentage of goals achieved. (*effectiveness*)
- Percentage of tasks completed successfully at first attempt. (*effectiveness*)
- Percentage of relevant functions used. (*effectiveness*)
- Time to complete a task. (*efficiency*)
- Number of persistent errors. (*efficiency*)
- Time spent on correcting errors. (*efficiency*)
- Rating scale for satisfaction. (*satisfaction*)
- Rating scale for error handling. (*satisfaction*)
- Frequency of reuse. (*satisfaction*)

Usability goals are necessary as they force us to think in ways that are more down to what the users need (and their intentions), instead of trying to make a presumption about requirements. First we need the goals, and then we can support the goals in terms of requirements and functions.

Whiteside, Bennet and Holtzblatt (1988) suggest a method for using usability goals to drive the design and for measuring the usability level:

- **Current level.** Measured either in the manual process or with the current or competitive product and used as a benchmark to set minimum acceptable levels for the planned product.
- **Minimum acceptable level.** Used during iterative evaluation and re-design to determine when to stop iterating[5].
- **Target level.** Used to drive and focus the design effort, actual expected level.
- **Optimal level.** Used as a target for the long term. What should be possible if time, money, etc. were not considered?

4.3 Discussion about usability

It takes an effort to define usability goals, to use them as acceptance metrics and as a driving force for design solutions. For many projects and organisations this effort is too much or just not applicable. This may be due to many reasons, e.g.:

- Usability metrics is not easy to translate into design solutions. It is one thing to establish the goals and measure them, another thing to design accordingly, i.e. to measure usability is sometimes questioned as the most efficient way of providing "good" design (as opposed to, e.g. an experienced designer doing the work).

[5] Note that this is a kind of "most favourable" setting for a project highly focused on usability. To stop iterating within a project is rarely related to only the achieved usability level. On the contrary, it is often so that to stop iterating is associated with other circumstances, such as time limits, money and the notion of "good enough".

- Not enough knowledge or experience in usability and metrics, or just not possible for different reasons to define usability goals.
- Often there are not enough resources in a project dedicated to usability to make, e.g. a thorough usability assessment.

The experienced problems mentioned in the list must never be an excuse for not defining some kind of goals of use for the system. Maybe it is not necessary, or possible, to have exact goals. The presence of just some major usability goals can sometimes be enough. Goals are essential, as argued for in the ISO standard. But every project must customise, find their appropriate level and tailor its own usability activities. If the resources are restricted and small: focus on what is most valuable for the end users within that context. If there are "enough" resources available, go the whole way and establish a full framework for usability. I am discussing more on this subject in the chapter *Usability Design*.

I have experienced the eagerness for more specific guidelines when it comes to usability and especially user interface design. Many development organisations ask (or even beg) for easy-to-use design guidelines. One example from the real world is: the most frequently asked question when my colleagues and I give a presentation on usability and user-centred design is what font to use in the user interface[6]. They seek for the easy solution to a design problem, instead of trying to understand the full range of usability and the process to achieve usable systems. There are such design guidelines or heuristics available, but they are often either to general or to detailed. Examples of general design guidelines are ISO/IS 9241-10 Dialogue Principles and the "usability guru" Jakob Nielsen's (Nielsen, 1994b). Here are just some examples:

- **Self-descriptiveness.** A dialogue is self-descriptive when each dialogue step is immediately comprehensible through feedback

[6] For the interested reader: it is preferable to use non-serif fonts on the computer screen for best readability. This is due to the poor screen resolution. The standard answer to the question is: Verdana.

from the system or is explained to the user on request. *(ISO/IS 9241-10)*

- **Conformity with user expectations.** A dialogue conforms with user expectations when it is consistent and corresponds to the user characteristics, such as task knowledge, education and experience, and to commonly accepted conventions. *(ISO/IS 9241-10)*
- **Match between system and the real world.** The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order. *(Nielsen, 1994b)*
- **User control and freedom.** Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo. *(Nielsen, 1994b)*

(These are just some examples...)

The solution on how to develop a usable system is never guidelines alone. They are part of the knowledge that usability professionals know by education, experience and use in their practice. You cannot simply apply a guideline without knowing how to transform the guideline into design. This is part of skills that has to be acquired. So, there is a paradox in that guidelines can only be applied by those who have the skills to apply them and consequently already know about the guidelines. More detailed guidelines such as: "use Verdana", are perfectly applicable and can easily be adopted, but are of little help when trying to design an interactive system for certain users, in a certain context, etc. They are also of the nature that a skilled professional already knows about them and how to apply them. They merely serve as a documentation of the current style of the user interface.

To obtain a usable system the development process is the key, and the notion of users, users' tasks and their context is inevitable. This can be one of the most frightening insights for a development organisation, since then they realise that they have to incorporate a user's view into the development process and cannot just rely on pure engineering and technical presumptions. Bringing the notion of users into the process will undoubtedly call for a more user-centred design approach.

The definition of usability deals mainly with *what* we as developers need to know to be able to develop usable systems, and how we can measure our progress. It does not concern *how to* actually develop usable systems. To do this we need to elaborate more on how to achieve efficient, effective and satisfactory systems through user-centred design. This topic will be further discussed in the next chapter.

There are different notions of the concept of usability. As I have my background in HCI I have adopted the concept of usability being scientifically defined and more or less objectively measurable. Other concepts, such the total *user experience*, have started to make an impact within the IT-industry. Designing the total user experience means that whenever the user is confronted with a product (i.e. an interactive system): The user shall experience a "positive" time together with that product. This means all the way from seeing, e.g. a software product on shelf at the store, through the installation of the product and the usage of that piece of software. Partly in contrast to the definition of usability—the total user experiences focus almost only on the user's attitude towards the product. It is what the user feels and thinks about the product that matters, not that much if the product is efficient and effective to use. The total user experience can be seen as a kind of answer or reaction from the industry to the more theoretical and sometimes "hard to apply" ISO definition of usability. Setting up usability goals, designing accordingly and evaluating against them, is not an easy task or something that the software industry in general is familiar with. The total user experience also has a much more commercial angle on the usability aspects: the customer is king. It is what the customer thinks and says that rules, not what we can measure in terms of usability goals, etc. I think it is sad that concepts like the total user experience takes precedence over usability as defined by the standard. It is from my point of view a step backward as usability again becomes unclear and hidden in some imprecise catch phrase. How can we measure concepts like the total user experience? How can we make sure that the end users get usable interactive systems if managers buy systems according to a fuzzy slogan like total user experience.

I will not in this context go into details about what composes a good user interface and interaction. Parts of that are discussed in some of the papers in this thesis: *The Design of a Smart Card-Based Home-Help System*, *A Teleradiology System Design Case* and *Usability and efficiency. The Helios approach to development of user interfaces.*

5 User-Centred Design

– Eighty percent of software life cycle costs occur after the product is released, in the maintenance phase. Of that work, 80% is due to unmet or unseen user requirements only 20% of this is due to bugs or reliability problems.

Karat, C. (1993), Usability Engineering in Dollars and Cents, IEEE Software, May 1993, pp 89.

– After the New York Stock Exchange upgraded its core trading systems using user-centred design techniques, productivity rose dramatically and users' error rates fell by a factor of 10 even though workloads more than doubled.

Cited in Gibbs, W W (1997). Taking Computers to Task. Scientific America, July 1997.

– Norwich Union, an insurance company in Australia, found that calls to its help desk reduced dramatically by two thirds after one of its core applications was improved using user-centred design techniques.

Norwich Rethinks Customer Service, Computer World, 24 November 1995.

If we cannot take usable systems for granted, how can we be sure, or at least make it more likely, that we develop usable systems? The definition of usability as discussed in the previous chapter is a great position to use as a stance in achieving usable systems. But, as we are talking about making usability a natural ingredient or a set of attributes of an interactive system in a specific context, we need to see usability in the light of the overall system development process. We also have to realise that usability is not always a major goal when developing a system. But, when we talk about interactive systems used by professionals in their daily work, usability is always one of the major factors contributing to the acceptance and success of the system. In order to achieve more usable systems, I have focussed on research and practice according to the philosophy of user-centred design.

5.1 Introduction

There is no sole and exact definition of user-centred design. The expression is rather vague and is interpreted slightly differently even within the HCI community. Yet there is an ISO standard on the subject—ISO/IS 13407 *Human-centred design processes for interactive systems*, but the standard is more of a description than a definition (more on this international standard can be found later on in this chapter). When searching for a definition I have found some examples that can serve as good illustrations. Preece et al. (1994, p. 722) define UCD as: "an approach which views knowledge about users and their involvement in the design process as a central concern". Donald Norman wrote back in 1986: "But user-centred design emphasizes that the purpose of the system is to serve the user, not to use a specific technology, not to be an elegant piece of programming. The needs of the users should dominate the design of the interface, and the needs of the interface should dominate the design of the rest of the system." (Norman, 1986). John Karat from IBM classifies UCD as: "For me, UCD is an iterative process whose goal is the development of usable systems, achieved through involvement of potential users of a system in system design." (Karat, 1996). He continues: "I suggest we consider UCD an adequate label under which to continue to gather our knowledge of how to develop usable systems. It captures a commitment the usability community supports—that you must involve users in system design—while leaving fairly open how this is accomplished." (Karat, 1997). Dennis Wixon from Digital Equipment, among others, states: "A user centred design process is one that sets users or data generated by users as the criteria by which a design is evaluated or as the generative source of design ideas." (cited in: Karat, Atwood, Dray, Rantzer & Wixon, 1996).

In the light of this, I would like to spend a few moments to explain what I prefer to read into user-centred design as far as the interpretation of the expression is concerned. *User*: by the user I mean the actual user of a system—end user. This category is almost always not the same as the one purchasing or ordering a system. This distinction is important, there is no substitute for end users such as people thinking that they *know* how end users work, react, etc. User is defined in ISO/IS 9241-11 as: “Person who interacts with the product”. *Centred*: to put something in the centre, or in the middle, does not necessarily mean that you involve them in your activities. But everything you do rotates around or focuses on them. From my perspective, however, in the context of UCD, the term centred has to include active involvement of end users. Note that this does not say that users should be in control of the design, but take active part and be the main source of information when designing the system and also the ones who ultimately benefit from the designed system. *Design*: design is a difficult concept used in many different ways and deserves a longer discussion. The term design used in the context of designing a computer system may have different meanings. “Design refers to both the process of developing a product, artefact or system and to the various representations (simulations or models) of the product that are produced during the design process.” (Preece et al., 1994, p. 352). There are some frustrating mismatches between different stakeholders’ interpretation of design in system development. System developers in general often see design as covering the entire development process, while someone designing the user interface looks at design as the shaping of the user interface. For the end user, the dialogues on the screen constitute the design. My experience is that it is crucial to be unambiguous about what we mean when we use the word design. Not that we can agree on a single definition, but to make it more likely that we know what we are talking about in a certain context. Why (user-centred) design process and not (user-centred) development process? I do not have an exact answer to that question. For me it is important to see user-centred design as a concept in the whole system life cycle. Lifecycle is defined as: “The stages and activities spanning the life of the system from the definition of its requirements to the termination of its use, covering its conception, development, operation, maintenance support and disposal.” (ISO/TR 18529, 2000). All those aspects are important in order to stay committed to the notion of usable systems, but the full breadth goes beyond the scope of my current work. My focus is on the development stage and the processes in that stage. An example of a commonly used development process is the Rational Unified Process (RUP). RUP defines what they call core workflows: business modelling, requirements specification, analysis & design, implementation, test and deployment (Rational Software Corporation, 2000). A typical iteration within the process covers the following workflows: requirements, analysis, design, implementation and test (Jacobsen, Booch, & Rumbaugh, 1999). Design in combination with user-centred (i.e. user-centred design) throughout the development process covers in the case of RUP the spectrum from business modelling to deployment. Note that I am not taking into account if RUP is user-centred or not, just using RUP as an example that people can relate to (more on the real topic of user-centred design and RUP later on in this chapter). The explicit design workflow in RUP is defined as: "A core workflow whose primary purpose is to formulate models that focus on non-functional requirements and the solution domain, and that prepares for the implementation, distribution, and performance" (Jabobson et al., 1999). For the purpose of developing usable systems, it is not enough to stick to this limited part of the process. To be able to develop usable systems as defined in ISO/IS 9241-11 we need to take the whole development process under consideration.

Two more things need to be sorted out: the use of user-centred as opposed to human-centred (used in, e.g. ISO/IS 13407), and the usage of a hyphen between user and centred or not. For sorting out these subjects I refer to an informal e-mail dialogue between one of the authors of ISO/TR 18529, Jonathan Earthy and Jan Gulliksen[7]. The conclusion from that dialogue is that the expression user-centred, with a hyphen, is favourable to human-centred. The rationales behind this is that human-centred was chosen by the group writing the ISO/IS 13407 standard as a broader term (not excluding, e.g. maintenance people, support people) than user-centred, and in an attempt to avoid an argument with others such as Norman & Draper (see next section *Background and history*) already us-

[7] Jan Gulliksen is an associate professor at the Department of Human-Computer Interaction at Uppsala University.

ing the user-centred expression. But now it seems that the people currently involved in the standardisation work prefer the term user-centred over the more conceptually difficult human-centred, and Norman & Draper do not appear to see themselves as the owners of the user-centred expression. The hyphen is preferred to show a link, but is not that dramatically significant.

So, to conclude this long explanation; user-centred design has a role in the overall system development process. Exactly why it is not called something like user-centred system development is fuzzy. From my point of view, user-centred design is a process built on a philosophy, attitude or similar with some key principles that call for the awareness of users and usability throughout the entire development process and even further throughout a system's life cycle.

As I have been involved in research and practice on the topic of user-centred design during more than a decade, I am increasingly convinced that UCD should be treated as Karat puts it: "*…consider UCD an adequate label under which to continue to gather our knowledge of how to develop usable systems…*". For me the key words are: *User* and *Centred*. This puts the focus on what is important in the process. If it is labelled a design process or a development process is not that important on this abstract level.

5.2 Background and history

The book *User Centred System Design* by Norman & Draper in 1986 is usually acknowledged as a pioneer work on the subject. The book, still worth reading, introduced a couple of new concepts that were fairly innovative (it had the sub-title: New Perspectives on Human-Computer Interaction). Donald Norman wrote one of the most cited book chapters in HCI: *Cognitive Engineering*. In this chapter he laid out some theories and premises that are fundamental, and today natural, when talking about user-centred design, e.g. how the theories of mental models (Johnson-Laird, 1983), affect system design. In short, a mental model is the user's understanding of how a system, problem, etc. is assembled. Our mental models are individually unique and we cannot see another person's mental model. But, we can use methods to try to understand some aspects of the users' mental models.

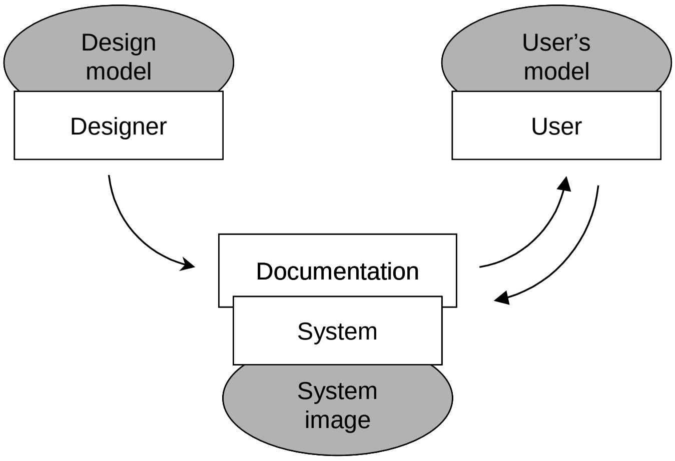


Figure 3: The designer should want the User's model to be compatible with the underlying conceptual model, the Design model (Norman, 1986).

The *Design model* is the conceptual model of the system to be built, held by the designer. The user develops a mental model of the system—the *User's model*. The user model is not formed from the design model. It results from the way the user interprets the *System image*. The system image is the physical image[8] (i.e. user interface) of the computerised work situation (including the documentation and instructions). The design challenge is to create a system so that the user can develop a mental model of that system consistent with the design model. The design model is supposed to be based on the user's task, requirements and capabilities. It must consider the user's background, experience and the powers and limitations of the user's information processing mechanisms.

Further on, Norman made his "prescriptions for design"[9]. One of those prescriptions was, do user-centred system design: "*Do user-centred system design: Start with the needs of the user*. From the point of view of

[8] Including whatever physical devices are available: knobs, dials, keyboards, mouse and displays.

[9] Norman talks about these prescriptions as: "The general ideas and global framework lead to a set of overriding design guidelines, not for guiding specific details of the design, but for structuring how the design process might proceed" (Norman, 1986, p. 59).

the user, the interface is the system. Concern for the nature of the interaction and for the user—these are the things that should force the design. Let the requirements for the interaction drive the design of the interface, let ideas about the interface drive the technology. The final design is a collaborative effort among many different disciplines, trading off the virtues and deficits of many different design approaches. But user-centred design emphasizes that the purpose of the system is to serve the user, not to use a specific technology, not to be an elegant piece of programming. The needs of the users should dominate the design of the interface, and the needs of the interface should dominate the design of the rest of the system." (Norman, 1986, p. 61).

Norman's theories are important as they help us to understand that all users are different and that we all interact with technology in different ways. We all establish our own mental model about the how the system works. To be able to understand what can cause usability problems, we have to use special methods to study users in action. Norman has attracted some criticism for treating users as passive objects to study and not inviting them into the design process (e.g. the Scandinavian school; Greenbaum & Kyng, 1991). But, his theories and discussions about user-centred design are very important.

Donald Norman was not the only one thinking in the direction of user-centred design. People at the IBM's Thomas J. Watson Research Center (e.g. John Gould) had for some time experimented with system design methodologies for developing usable systems. The most significant project was maybe the 1984 Olympic Message System[10] (Gould, Boies, Levy, Richards & Schoonard, 1987). This was a huge development effort with the goal of making the system as usable as possible, but also to test a user-centred design methodology. They were successful in both attempts. The system was really usable and the key principles that guided the user-

[10] The Olympic Message System was developed in order to provide a voice mail message service. Kiosks were placed around the Olympic village, allowing athletes to send and receive voice messages among themselves, but also to receive congratulations, etc. from around the world.

centred design proved to work. More on these principles in the section *User-centred design: the principles*. At Digital Equipment, e.g. Dennis Wixon and Karen Holtzblatt started to work on something they called Contextual Inquiry and Contextual Design (Wixon & Holtzblatt, 1990), later evolving to become Customer-Centred Design (Beyer & Holtzblatt, 1998). They were, e.g. influenced by ideas coming from Scandinavia and known as the Scandinavian school (or approach). This school was introduced to a broader audience by the book "Computers and Democracy: A Scandinavian Challenge" by Bjerknes, Ehn and Kyng (eds., 1987) and a doctoral thesis by Pelle Ehn: "Work-Oriented Design of Computer Artifacts" (Ehn, 1988). The Scandinavian school stresses the importance of users participating in the development on equal terms as the developers. It has a set of perspectives and practices for emphasising the role of users as active participants in the process through which computer artefacts are designed that have effect on the users' lives in and out of the workplace. It takes its stance in strong labour unions, acting as advocates for workers, and a history of socio-technical approaches which argued for the importance of the social dimension of work with technology. The legal and democratic rights of the users to influence and control the work situation are very important in the Scandinavian school. Users should be in control of the design process, not only involved. Greenbaum & Kyng coined the term *Cooperative design* as they summarised and elaborated on the Scandinavian approach (Greenbaum & Kyng, 1991). In North America the term *Participatory design* is commonly used for the theories based on the Scandinavian approach (e.g. Schuler & Namioka, 1993). Yet others such as Catterall et al. take a socio-technical standpoint: "User-centred design implies the active participation of users in the design process… [it] should be comprehensive and not simply part of an end-process evaluation procedure" (Catterall, Taylor & Galer, 1991, as cited in Preece et al., 1994 p. 376).

None of these very important landmarks of user-centred design approaches can be considered to be widely adopted and practiced development processes. Contextual design is maybe the most widespread and process-oriented one. But, it is noticeable that these approaches very much reflect a certain attitude or philosophy more than a development process. In parallel to this, a more pragmatic approach to developing a usable system has evolved: usability engineering. Usability engineering has been defined as "a process whereby the usability of a product is specified quantitatively, and in advance. Then, as the product is built, it can be demonstrated that it does or does not reach the required levels of usability." (Preece et al., 1994, p. 650). Usability engineering has adopted the general components of software engineering to provide an engineering-like process for the development of usable systems. The most read book on the subject is probably Jakob Nielsen's "Usability Engineering" (1993a). Usability engineering covers a broad range of techniques to analyse users, establish usability goals, evaluate design, etc., that are part of any user-centred design process. But usability engineering is not to be mistaken as being equal to user-centred design. While usability engineering is some hands-on activities for certain steps in a user-centred design process, the framework for UCD is broader than that. The best effort I have found to put forward a usability engineering approach in the light of user-centred design is: "The Usability Engineering Lifecycle" by Deborah Mayhew (1999). Mayhew really put the usability activities into practice by framing them into a process. This is essentially more than just applying some usability techniques.

Today huge software and hardware companies promote user-centred design as one of the most important success factors for their products. IBM continues on the UCD theme with their Ease of Use approach (see, http://www.ibm.com/easy/). Even Microsoft promotes the UCD approach (see, http://msdn.microsoft.com/ui/) in an active way, here is an example: "The term 'usability' in the context of creating software represents an approach that puts the user, rather than the system, at the center of the process. This philosophy called user-centred design, incorporates user concerns and advocacy from the beginning of the design process and dictates that the needs of the users should be foremost in any design decision.". That is rather interesting coming from a company infamous for their “function-centred” approach. Even further, they state that: “Let user-centred design be the arbitrator of design decisions, not user interface guidelines” (these quotes are taken from the MSDN online library at http://msdn.microsoft.com/library/techart/uidesign.htm). So, even Microsoft says that following their own user interface guidelines is not enough in order to develop usable systems.

As mentioned in the first section of this summary UCD has lately received growing awareness within several organisations. However, the successful deployment of a UCD approach is complicated. It requires a shift of focus in the entire development process, by all participants in the process, away from a general focus on, e.g. technology, to a focus on usability.

5.3 User-centred design: the principles

It would be easy if we could say that doing UCD is exactly doing like this or that. But, there is no such thing as *the* process or method to design a usable system. The truth is that designing a usable system requires a sequence of activities, each performed with an appropriate method, and the exact activities and methods may differ from organisation to organisation and even from project to project. In this perspective, UCD is *a process* and to be user-centred is an *attitude*. In my research and practice I have found two sources that set the scope for UCD and are widely accepted as fairly indisputable: the international standard ISO/IS 13407 *Human-centred design processes for interactive systems* (1999) and the key principles of design by Gould et al. (1997).

The standard ISO/IS 13407 describes user-centred design as: “An approach to interactive system development that focuses specifically on making systems usable. It is a multi-disciplinary activity, which incorporates human factors and ergonomics knowledge and techniques. The application of human factors and ergonomics to interactive systems design enhances effectiveness and efficiency, improves human working conditions, and counteracts possible adverse effects of use on human health, safety and performance. Applying ergonomics to the design of systems involves taking account of human capabilities, skills, limitations and needs.”

It goes on to say that “Human-centred systems empower users and motivate them to learn. The benefits can include increased productivity, enhanced quality of work, reductions in support and training costs and improved user health and safety. Although there is a substantial body of human factors, and ergonomics knowledge about how such design processes can be organised and used effectively, much of this information is only well known by specialists in those fields. This International Standard aims to help those responsible for managing hardware and software design processes to identify and plan effective and timely human-centred design activities. It complements existing design approaches and methods.”

The standard is a framework for those managing design processes and provides guidance on sources of information and standards relevant to the user-centred approach. It provides an overview of user-centred activities. It does not provide detailed coverage of all methods and techniques required for user-centred design.

The overall characteristics of the standard is the cyclic process:

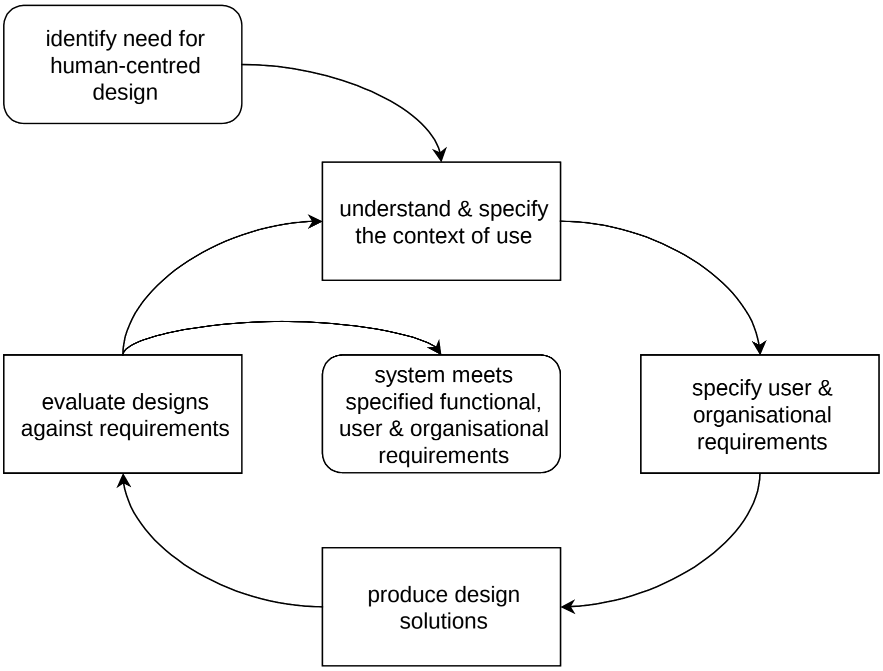

identify need for human-centred design
understand & specify the context of use
evaluate designs against requirements
system meets specified functional, user & organisational requirements
specify user & organisational requirements
produce design solutions

Figure 4: Activities in ISO/IS 13407 “Human-centred design processes for interactive systems”.

The four major activities (the squares in Figure 4 above) are described in ISO/IS 13407 as:

- **Understand and specify the context of use.** The characteristics of the users, tasks and the organisational and physical environment define the context in which the system is used. It is important to understand and identify the details of this context in order to guide early design decisions, and to provide a basis for evaluation: the characteristics of the intended users; the tasks the users are to perform (including the overall goals of use of the system) and the environment in which the users are to use the system.
- **Specify user and organisational requirements.** In most design processes, there is a major activity specifying the functional and other requirements for the system. For user-centred design, this activity should be extended to create an explicit statement of user and organisational requirements in relation to the context of use description.
- **Produce design solutions.** The next stage is to create potential design solutions by drawing on the established state of the art, the experience and knowledge of the participants and the results of the context of use analysis: making use of existing knowledge to develop proposed multi-disciplinary solutions; make the design solution more concrete using simulations, models, mock-ups etc.; show the design solutions to users and allow them to perform tasks (or simulated tasks); iterate this process until the user-centred design goals are met and manage the iteration of design solutions.
- **Evaluate designs against requirements.** Evaluation is an essential step in user-centred design and should take place at all stages in the system life cycle. Early in design the emphasis is on obtaining feedback that can be used to guide design, later when a more complete prototype is available it is possible to measure whether user and organisational objectives have been achieved:

provide feedback which can be used to improve design; assess whether user and organisational objectives have been achieved and monitoring long term use of the system.

The user-centred design process involves iterating these activities until the objectives are satisfied. The need for a user-centred design approach will have been identified from the operational objectives of the system, e.g. to satisfy requirements for usability.

The activities are further elaborated on in the ISO/TR18529 technical report: “This document is intended to assist those who wish to make their system development process and its associated support processes more human-centred, and to include knowledge from the human sciences in system design.” (ISO/TR 18529, 2000). The important thing with the standard and the technical report is that, on a high level, they point out certain user-centred activities and introduce the notion of a cyclic process where focus on the users is the critical part.

“What best defines user-centred design? I do not think that I have found anything better in print than chapters by Gould and by Whiteside, Bennett and Holtzblatt, though neither paper has UCD as a label”. (Karat, 1996. Authors note: Karat is referring to chapters in Helander, Landauer & Prabhu, 1997; Helander, 1988). As I mentioned in a previous section, researchers at IBM have defined some key principles (I refer to them as “Gould’s principles”) for how to develop usable systems. They are summarised in Gould (1997) but have evolved over the years as documented in, e.g. Gould et al. (1983, 1985 and1987). He discusses four design process principles:

- **Early – and continual – focus on users.** Designers must first understand who the users will be. This understanding is determined in part by directly studying their cognitive behavioural, anthropometric[11], and attitudinal characteristics, in part by studying the nature of the work expected to be accomplished,

[11] Anthropometrics concerns physical characteristics of people.

and in part by making users part of the design team through participative design or as consultants.

- **Empirical measurement.** Early in the development process, intended users' reactions to printed scenarios and user manuals should be observed and measured. Later on they should actually use simulations and prototypes to carry out real work, and their performance and reactions observed, recorded and analysed.

- **Iterative design.** When problems are found in user testing, as they will be, they must be fixed. This means that design must be iterative: there have to be a cycle of design, test and measure, followed by redesign etc., repeated as often as necessary. Empirical measurement and iterative design are necessary because designers, no matter how good they are, cannot get it right the first few times.

- **Integrated design – wherein all aspects of usability evolve together.** All usability factors must evolve together, and responsibility for all aspects of usability should be under one control. Integrated design requires a departure from fractionated development practices where various aspects of usability are developed in different loosely related departments, divisions, cities, companies. Integrated design assumes recognition at the very outset that usability is important, that it includes many factors, and that work must begin on it from the start.

The issue of user participation is extremely important. User participation should in all situations be preferred, i.e.:

- Users are experts in their work and therefore the only ones that can describe it.

- Users are the ones that are most suitable for testing and evaluating prototypes and systems that are developed for them.

But, on the other hand, user participation in a project is never, in itself, a guarantee for a usable system. Users must become more than just passive members of the project team. There must be activities, methods and skilled usability professionals in the project to ensure that users and user input really can make an impact in the development, and that user competence is utilised in the best possible way.

However, our experiences from several different development domains shows us that even such a simple model as the ISO/IS 13407 and the principles by Gould are not easy to follow or use. They can easily be misinterpreted and even misused. It is inevitably so that UCD is a difficult approach to make your property in a way that makes continuous user-centred iterations a natural and obvious approach in all your work. Neither ISO/IS 13407 nor Gould's principles seems to be enough for organisations to use as a process for developing usable systems.

The key question for a development organisation focusing on developing usable systems is how to adopt a user-centred design approach that fits into their context and their development environment. This topic is discussed in two of the papers in this thesis *A User-Centered Approach to Object-Oriented User Interface Design* and *Reengineering the Systems Development Process for User-Centred Design*. I am also further elaborating on the topic in the next sections and providing a list of heuristics for adopting a user-centred design approach.

5.4 Development processes in the industry

We have studied developing organisations carrying out several projects. Most of them have some kind of implicit or explicit development process that they follow. It's either an internally developed process, or a commercial product brought into the organisation. In my experience the system development process is central to the success of the final system. Various methods or techniques can always be applied to improve the final result of the system but it is only when viewing the entire process that we can tell where and when problems can occur. When organisations today try to improve their system development process, it is very common that they buy a commercial system development process, such as Rational Unified

Process (RUP) or Dynamic Systems Development Method (DSDM)[12]. When doing so they anticipate that every problem is to be solved through the use of the selected process. We have noticed that organisations, once convinced to use a certain process, are eager to fully start to use it and fail to reflect upon how user-centred it is. In general, these organisations are not fully aware of how to perform user-centred design and those aspects are not a concern when deciding to invest in a new development process. Several times I have asked people in those organisations: "Well, have you discussed usability and user-centred design before buying RUP?" and the answer is a long: "No…". So, we often find ourselves in a situation when we are asked to help them with usability and eventually user-centred design in such a context. When we do exactly that, we have to study in detail the process they are using and examine it from a user-centred standpoint. It takes a lot of effort for the stakeholders in the organisation to start to think in the direction of evaluating the newly introduced process from a user-centred perspective. But, it is a necessary exercise since it will illuminate the user-centredness of the organisations development work. An example of this is in the paper *Reengineering the Systems Development Process for User-Centred Design*.

The situation today for the organisations that I have been involved in is that they, in one way or another, use the Rational Unified Process, or parts of it. It is fascinating how successful RUP has become and how fast. The rationales behind using RUP vary but we have found some key factors: so many other organisations have started to use RUP that it has become a de facto standard; sharing a common process makes it is easier for organisations to hire external consultants when they are short of their own resources; the process is maintained and still develops; it is based on object oriented techniques, which are desirable for several reasons; there

[12] Details on Rational Unified Process can be found in Kruchten: *The Rational Unified Process—An Introduction* (1998) and Jacobson, Booch & Rumbaugh: *The Unified Software Development Process* (1999). Details on DSDM can be found in Stapleton: *DSDM – Dynamic Systems Development Method* (1997). I will not here describe the intrinsics of RUP or DSDM. However, some of it is covered in the paper *A User-Centered Approach to Object-Oriented User Interface Design*.

are supporting tools and education is available. There are of course other factors, but these are typical for the organisations we have met.

There are some competing processes such as DSDM, but RUP is totally dominating the market. RSV (The Swedish National Tax Board) had their internally developed process when we started to work together with them, but has now moved to RUP. RFV (The Swedish National Social Insurance Board) is in the process of starting to use RUP in all their projects, etc. Even organisations that do not commit themselves to follow every part in RUP use techniques that are essential parts of the process such as use cases (use cases are detailed and discussed later on). So, it is natural when we do our research in development projects to be confronted with RUP and other commercial processes and have to deal with how to be user-centred in that context.

It is therefore interesting to see how the essence of ISO/IS 13407 and the principles by Gould fit into the concept of the development processes used, e.g. RUP, in order to highlight the problems that we have faced when trying to practice user-centred design in that development environment. I will also devote a special section to discussing use cases, as they seem to be delicate to use when it comes to user-centred design.

Iterative design

Iterative design is one of the most significant characteristics of user-centred design, and most developing organisations today are aware that the traditional waterfall model for systems development is not the most favourable one.

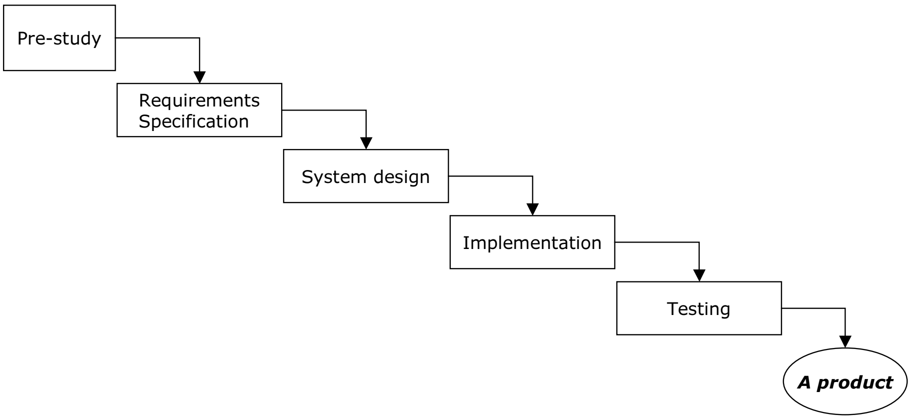


Figure 5: Example of a waterfall model from *Strukturerad Analys och Konstruktion av informationsbehandlingssystem* (Wigander, Svensson, Schoug, Rydin & Dahlgren, 1979).

The traditional waterfall model suits a principled approach to design: i.e., if we know what it is that we want to produce from the beginning, then we can structure our approach to design in order to reach that goal. This approach has its advantages when it comes to solving rather technical and already known, well-defined, engineering problems. But it doesn't take into account things such as (Budde, Kautz, Kuhlenkamp & Züllighoven, 1992); the impossibility of a complete and permanently correct description of the system; formal and non-executable specifications are largely unintelligible to users and developers; the need for user participation, etc. As pointed out in both ISO/IS 13407 and Gould's principles, designers do not find out all requirements for a system before they begin. All requirements for an interactive system cannot be determined from the start. The system must be iteratively built and the user's interaction with the system must be observed and evaluated in order to determine how to refine the system and make it more usable.

The waterfall model has in most cases been succeeded by the notion of iterative design. This is a good thing and most development processes today are described as iterative. However, it is our experience that the concept of iterative design is not easy to understand and to achieve in practice, and especially not when it comes to iterating with users. Most projects I have studied work somewhat iteratively with users in the beginning of a project to capture requirements, but after a while they stop to iterate and fall back into a waterfall of designing, implementing and testing. To highlight this crucial issue I would like to mention the notion of iterative design as explained in RUP. According to RUP an iteration is: “A distinct set of activities conducted according to a devoted (iteration) plan and evaluation criteria that results in a release, either internal or external” (Jacobson, Booch & Rumbaugh, 1999). This indicates that an iteration is a fully build (component) of a certain part of the system; and appears to coincide with the definition of incremental development wherein the system is built one section at a time. A more user-centred approach would be to see the iterative design as iterating possible solutions together with users (as in ISO/IS 13407 and Gould) and build the system by increments. If you follow RUP by the book, iterating with users is a process that requires too much time and resource planning to be really effective for a user-centred approach. It will most likely lead to a waterfall-like development process.

This is a problem that is not easily solved, but can be improved by having a constant and explicit focus on users throughout the project. It requires that the project, effectively, can handle changes in requirements, etc. and that the project team is open for these changes.

Techniques such as prototyping the user interface and interaction together with users, and iterating them, could be used very early in the development process in order to enhance the communication with users on requirements etc., but this is very seldom the case. Very often a waterfall development model is used anyhow, meaning that one activity must be finished before pursuing the next. This, in my opinion, is not necessary. A lot of work could be done using only parts of the result of the previous activities. To be able to do truly iterative design one must be prepared to frequently produce “deliverables” appropriate for user evaluation, according to some plan, etc. We have found that a major obstacle to this process is time. To produce deliverable results, all people involved in the process tend to require more time than allocated for analysis and design. Shortening the iterative cycle could prevent delays in this process. RUP recommends less than six months as a turnaround time for the iterative development process (Kruchten, 1998) (remember their notation of iterative design). The Swedish National Tax Board has decided on eight weeks for the iterative cycle. We believe that the development process can be significantly improved by forcing a shorter iterative cycle in which more usable systems can be developed. It could be reasonable to have a one-week iterative cycle; that is e.g. two days of analysis and design, two days of implementation and the last day for evaluation.

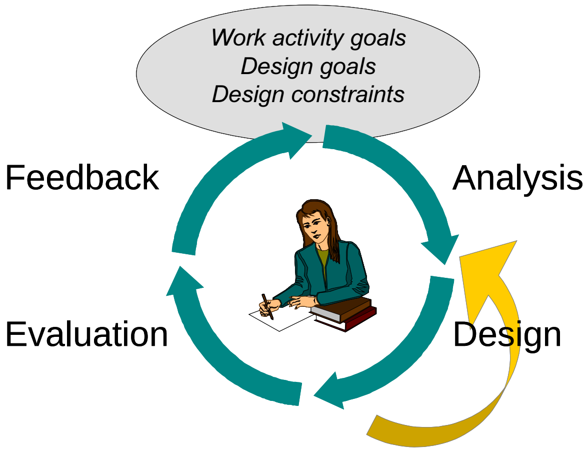


Figure 6: To teach or learn to adopt a fully iterative user-centred design activity has proven to meet several difficulties. The green (darker) arrows are the development phases. The yellow (lighter) arrow indicates the small informal design activities that are so important in the design phase.

In my research I spend a great deal of time teaching and promoting effective iterative user-centred design (see Figure 6). We have found that most developers regard iterating with users to be merely, e.g. the making of a phone call to a user that they knew, to get some specific information. This is not at all what is needed to make a proper iteration. Each iteration should contain: a proper analysis of the user's requirements and the context of use; a prototype design phase and a documented evaluation of the usability of the prototype that must result in suggestions for modifications in the following prototype design.

Focusing on users

The CHAOS-report on how to be successful in projects, states that the number one criterion for a successful project is active user involvement

(Standish Group, 1995)[13]. The user-centred design principles of *early and continual focus on users* and *understand users, their work tasks and their work context* imply user involvement and presumably active involvement. Focussing on users and understanding users, their work tasks, work context and user participation is crucial for any project aiming at UCD. Most projects do some of this but in a very limited way and do not find that much support in the development process for these kinds of activities.

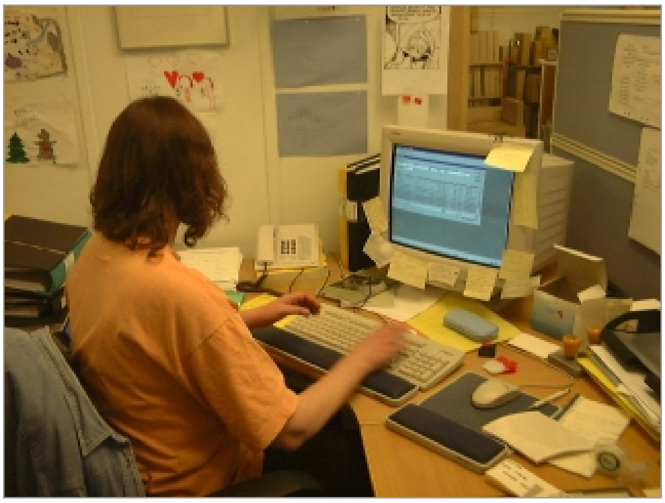


Figure 7: User at her work place. Illustrating user's work context.

The RUP approach to this is basically the use case technique, which is not appropriate enough for this purpose (see section *Use cases and use case modelling* later in this chapter).

Going out and study users in their own work context is something that is necessary in UCD, but does not have any natural part in most of the organisations and development processes that we have studied. In the figure above (Figure 7) there are several post-it notes on the screen. This is a very common situation and is a natural part of the user's work context. These notes are important for the user when performing her job, but are usually not considered when writing, e.g. use cases. A majority of the studied organisations do not even have a clear and common understanding of the user categories they are aiming at. If, e.g. a contextual task analysis

[13] Successful project defined as: completed on time, on budget, and with all features and functions originally specified.

(Mayhew, 1999; chapter 3) would have been a part of RUP and DSDM, would there be more usable systems around? I am sure of that. But, since no such explicit activity is a part of the development process, it is not performed and the competence to perform such an activity is not part of the project team.

The user involvement and participation is often defined as bringing users, or user representatives, into the project. This is important, but it is equally important to really make an effort to bring the developers closer to users' work environment/situation. It is critical that the developers responsible for the design of the user interface and interaction actually spend time at the users' work places to discover the nature of the users' work environment, work activities and especially the activities and procedures the users perform without being aware of it (tacit knowledge). This is not only true for the analysis phases in a project, but also for the design and evaluation phases.

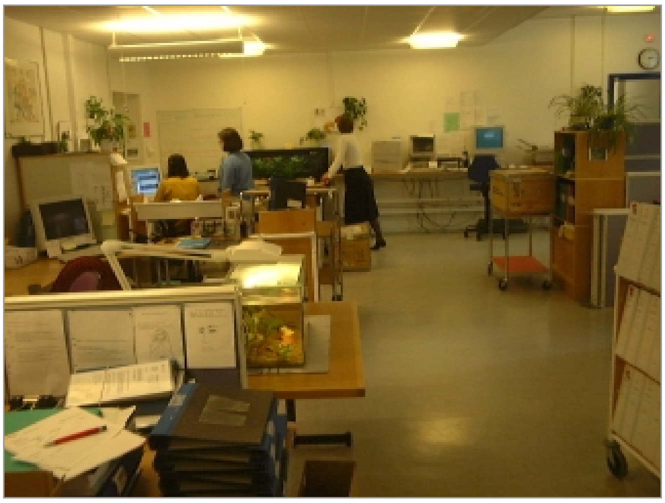


Figure 8: Users working environment where, e.g. contextual prototyping can take place.

The context of the users and their work situation is irreplaceable and must be experienced on site. We recommend setting up a project with "door-to-door" communication between the users and the developers. This means that parts of the developer organisation, to some extent, "move in" to the users' work place, and when possible run their part of the development project in that context. The approach will, of course, differ from situation to situation, much depending on, for instance, the nature of the development in terms of: in-house, contract or product. In an in-house development situation the users are known in advance and available, but for a product development team it will take some more effort to get in contact with users and start to work with them.

It is equally important to distinguish between developers, domain experts and real end users. The developers can never be considered to be users (unless they are developing a system for their own user community). They are always biased by their role in the development of the system. Domain experts, or users involved in the development, can be considered users in the sense of knowing the application domain. They are very important resources and should be used as main input regarding expertise knowledge of the target domain. But, there must be real end users as well. These are the ones who actually are going to use the final system. They are really the only persons who can inform us about the usability of the product. We must use them in all phases of the development—analysis, design and evaluation. This is a problem within almost all development organisations we have studied. Often they do not recognise the difference between a somewhat biased application expert and an end user. It also demands certain knowledge and experience to interview users and get the "right" answers. Often developers assume that users can tell exactly what they want, but this is simply not true. You have to use certain techniques to be able to elaborate on users' real needs.

Design solutions and empirical measurements

This takes us to the principles of *producing design solutions* and *empirical measurements* with sketches and prototypes. As I have stated in the discussion regarding iterative user-centred design, the measuring of usability is important. In order to measure we must define some usability goals or requirements. We have found that the concept of usability goals is difficult to adopt. We often see statements like "it should be easy to use…" in requirement specifications. However, this is not enough as guidance during the design, as discussed in section *4.2 Usability goals*. None of the projects that I have been involved in have managed to handle usability goals in an effective way. Maybe the concept of usability goals is too cumbersome and/or inefficient to use, but if so we need something else to use as driving force in the design. In the papers: *The Design of a Smart Card-Based Home-Help System* and *A Teleradiology System Design Case* we discuss the usage of design criteria and criteria-based design as a complement to the explicit usability goals. This technique has proven to be of great help when trying to find the best possible solutions to match the requirements. As we iterate through the analysis, design and evaluation we learn to a greater extent about what kind of design that will work and what will not work. Even though it is not possible in the first place to state that, e.g. "it must not take more than twenty seconds to submit a job to ProSpot…", we can learn that by applying certain design criteria (unique to the application domain and the context) we can achieve an efficient, effective and satisfactory design. Working with sketches and prototypes[14] as lightweight but concrete illustrations of the user interface and interaction for the system is essential (see Figure 9 below).

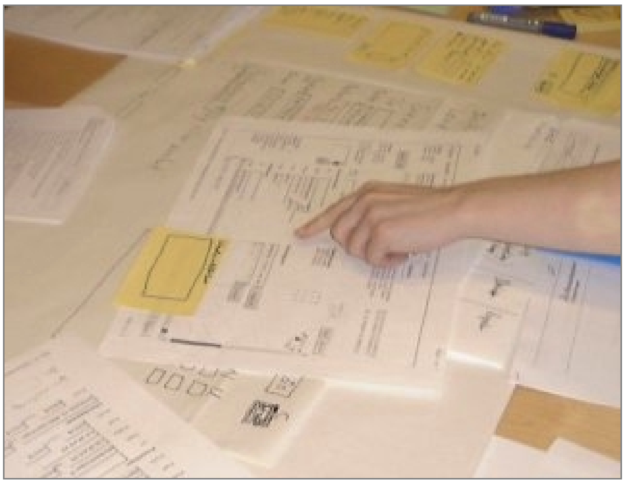


Figure 9: Paper sketches illustrating a proposed user interface.

This makes it possible for developers and users to discuss requirements, propose design solutions and evaluate without spending any effort on ac-

[14] In this case a prototype can be anything from a rough paper mock-up or sketch to an interactive computerised representation of the user interface and interaction.

tually implementing it. Paper and pencil is a good start before moving into computerised tools. Even if processes such as RUP encourage prototyping, it is my experience that projects spend too little time on investigating and evaluating design solutions. There seems to be a rush to start coding: "you can start coding while I go and see what the users want." By starting early with cheap paper prototypes or sketches, the project can actually save time and money by doing the right thing, instead of straight away heading for the first possible solution. If you get the chance, and it is feasible, work with several prototypes in parallel as proposed and discussed in the paper *The Design of a Smart Card-Based Home-Help System*. It is also my observation that there is a general lack of competence and experience in working with prototypes together with users. We often perform participatory design workshops in our projects (see Figure 10). These are facilitated workshops where users play a natural role together with usability and design people. These prototypes are used to drive the requirements as well as the design process.

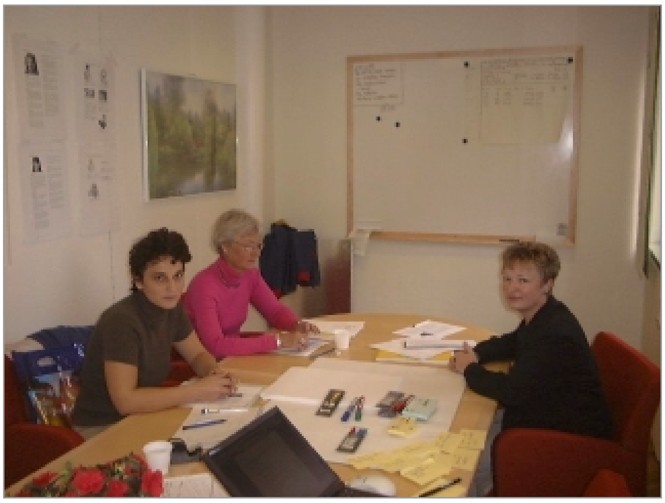


Figure 10: Participatory design workshop setting with users.

This kind of activity is something that I normally add to projects I take part in. Whether the project uses RUP or any other development process, these activities are normally not performed. The workshops or modelling activities in RUP and DSDM are usually rather formal and aim to deliver artefacts or products that represent user needs as well as user knowledge in some kind of formal textual language. This leads to almost endless modelling sessions where users are some kind of hostage. I have heard users saying things like: “Coming from six months of modelling sessions to a prototyping session was like coming out of a dark tunnel.” and “Now I can understand what we actually talked about during all those modelling sessions”. It is undoubtedly so that the developers need some kind of formal design description to be able implement the system, but it is not necessarily so that users need to be confronted with them. It is well known that it is easier for people to have opinions and input to something that is concrete such as a paper sketch or prototype rather than to a formal textual description.

Integrated design

Integrated design wherein all aspects of usability evolve together rarely happens in the projects and organisations that I have studied. If user-centred activities are a part of the project, these are often carried out in different parts of the project by different people and not fully synchronised. Most likely, vital user-centred activities are not performed at all. Activities such as developing on-line help and manuals are often seen as something that can be taken care of just before deploying the system and not at all developed in an iterative user-centred fashion. Gould stresses the importance of having one person or group in charge of all usability factors. Whereby someone in the project has the responsibility and authority to keep the project as user-centred as possible. As a direct response to this, the project role as a usability designer has been introduced and utilised in many of the projects and organisations I have been involved in, see section *6.3 The Usability Designer* and the paper *Usability designers improve the user-centred design process*.

User-centred attitude

The *attitude* to be user-centred is, as already pointed out, essential to the success of developing usable systems. However, I have found that many organisations become what I call “a slave under the development process”, where the selected process itself, e.g. RUP or DSDM, becomes the centre of activity. It seems that it is more important to follow e.g. RUP than to reflect upon how user-centred and effective the process is. In order to fully utilise from the process one is eager to fulfil everything that is stated in the process, and not really question what is best for the current project in a certain context. Often this leads to a "hyped" focus on producing, e.g. documents (artefacts in RUP and products in DSDM). Projects spend a lot of time writing documents, adding text to every heading, instead of focusing on the content of the project. To add user-centred activities is something that is usually necessary but something that most projects do not do. There are, though, good exceptions, like the work that is done at RSV in order to make their configuration of RUP more user-centred. This is work that we have been involved in and partly described in the paper *Reengineering the Systems Development Process for User-Centred Design*.

To be fully user-centred everybody in a project must be, if not dedicated to usability, at least aware of usability aspects and what constitutes user-centred design. I usually refer to this attitudinal insight as a paradigm shift for the entire development organisation. This is the ultimate, and maybe utopian, way to work with a user-centred design philosophy. The exact way to reach this paradigm shift is not clear. In my research, I have tried to influence all organisations and projects that I have had the opportunity to be involved in. Recently at Enea Redina, we have started a process for making the organisation work according to a user-centred design philosophy. It will take us some time to reach this goal and even if we do not reach it, we will hopefully learn something during the exploration. We have started with an approach to raise the "usability-awareness" among all developers and also to formulate a user-centred design process. Hopefully this can lead us to a greater understanding of what it takes for an organisation to commit itself to usability and UCD, and how it can be practiced in other organisations.

Use cases and use case modelling

In the paper *Reengineering the Systems Development Process for User-Centred Design* we discuss use cases and use case modelling. During our research we have found that use cases are so extensively used that we have to investigate the use of them and how they compare with a user-centred design philosophy. In this section I will only elaborate briefly on this topic as I feel it is crucial for any development project aiming for usable systems.

Use cases are defined in numerous ways, making them far from the standardised thing or item as they often are comprehended (Constantine & Lockwood, 2001). One definition used by Rational is: "a use case specifies a sequence of actions, including alternatives of the sequence, that the system can perform, interacting with actors of the system."(Jacobson, Booch & Rumbaugh, 1999, p. 135). As RUP is "an architecture-centric process" (Kruchten, 1998, section 5) and is use case driven (Jacobson, Booch & Rumbaugh, 1999, p. 5), use cases are seen as a kind of "silver bullet"—the solution to every potential problem.

Use case modelling is an abstract modelling technique that is facilitated by a use-case specifier[15]. Often these modellings are conducted in modelling seminars or workshops. It is our experience that this kind of seminar does not explicitly take into account or consider the users' characteristics, their real flow of work tasks or their context of work (working environment). Nor do they regard the human cognitive and perceptual constraints and abilities. The requirements on the system are described outgoing from the system and not from the users. If these use cases are used as a specification for the user interface and interaction, they will most certainly be insufficient for designing a usable system. Further, the concept of an actor is abstract and is not as rich, characteristic or precise as it ought to be in order to understand users, their context and make a user interface and interaction design. An actor in RUP is described as: "actors represent parties outside the system that collaborate with the system. Once we have identified all the actors of a system, we have identified the external environment of the system" (Jacobson, Booch & Rumbaugh, 1999, p. 134). They must

[15] In RUP the *system analyst* has the overall responsibility for the use cases, but it is the *use-case specifier* that makes the detailed description of them.

be complemented, or substituted by user profiles based on thorough user analysis, as for example described in Mayhew (1999, chapter 2 *User Profiles*). This relates, of course, to the user-centred design principle about understanding users and specifying their context of use.

As use case modelling is requirement engineering, users are only a part of the process during such an activity. When the project has found all potential use cases, they do not need to interact with the users anymore. So when the design starts, the project is quite confident that they know what to design and build. There is no natural continuation of user participation after the use cases are modelled.

The problem areas related to user-centred design that we have found, can briefly be divided into the following:

- ❖ **Modelling seminars.** Modelling seminars as forums for gathering information in a well-structured and formal way have drawbacks. Users feel uncomfortable away from their work context and have to learn and communicate through an abstract modelling language and notation. These seminars usually become a forum for "contracting" a design between developers and users rather than a co-operative forum for finding out requirements. The seminars are usually conducted without the necessary contextual environment and data information. The use cases are also often written by system analysts that are not specialists or trained in usability or in user interface and interaction design.
- ❖ **User participation and selection of users.** Representative users are often not participants in the seminars. Instead, the participants tend to be selected experts of the target domain, who already have a lot of presumptions about the forthcoming system. Real end-users are not recognised as being the valuable source of information that they are.
- ❖ **User centred activities (methods).** Use case modelling seminars are often used as the only source for understanding the users,

their context and work tasks. Other more user-centred activities such as user analysis and task analysis are seldom performed.

- **Iterations.** Analysis, design, re-design and evaluation: once you have decided on a use case model, it is too complex and complicated to change. The description of the model becomes a burden itself. Normally the possibilities of iterating the use case models are limited.
- **A shift of focus.** The focus for the use case seminars and the models are often the notation in itself. Once you start to describe the tasks in the domain with a formal notation, you shift focus from investigation to drawing or writing.
- **Use cases are fuzzy.** As pointed out by, e.g. Constantine & Lockwood (2001), writing use cases is not easy and often they end up being too much of a user interface specification and interaction design, instead of only capturing requirements. Examples of this come from a heuristic evaluation we conducted. We found more than 100 possible usability problems in a system designed through use cases. User studies revealed problems in the overall conceptual design pointing directly at the use cases. When we investigated how the user interface design and interaction was done, we got the answer that the use cases were regarded as the specification for the design.

The benefits of use cases are associated with more computer sciences aspects of the development, such as traceability of requirements for testing and a "common language" (Unified Modelling Language—UML) throughout the description of the system (system specification).

So, can use cases be used if you want to work in a truly user-centred design process? Yes, I believe so, but you have to be careful about how they are applied. Use cases must not be the only activity or technique for capturing user requirements or working with users. We have, for example, successfully conducted participatory prototyping workshops in parallel with use case seminars. Using essential use cases as proposed by Constantine & Lockwood (2001) or user interface modelling as described in the paper *A User-Centered Approach to Object-Oriented User Interface Design* are other examples. Use cases must not be regarded as *the* specification for the user interface and interaction design, but rather as one input to the actual design of the user interface and interaction. We can also regard use cases as documentation (or specification) for developers on how to design the internals of the system, and not necessarily something that users are confronted with.

Even though we have seen successful usage of use cases, it turns out that, in those cases, they usually are defined and written by persons with a strong background in usability and not strictly following the use case conventions.

5.5 Heuristics for UCD

I have compiled a list of heuristics to summarise our results from research and practice on integrating a user-centred design philosophy in numerous organisations and projects. This list is based on: ISO/IS 13407; Gould's principles; "best practices" from RUP and DSDM; and of course our own research and practice:

- **The organisation must define its own UCD process.** We think that to successfully adopt user-centred design with any of the commercial system development processes, it needs to be modified to meet the needs of the organisation or even individual projects. The organisation needs to specify its own user-centred design process, based on the commercial processes (or develop their own), and specify what complementary activities are needed. In doing this it may be advantageous to reuse the old methods or techniques previously established within the organisation.

- **Continuous active user participation throughout the whole development process.** Involve users directly in the project organisation as well as outside the project. Recognise the difference between work domain experts and real end users: use work domain experts continuously throughout the development project; end-users for more temporarily activities in analysis, design and primarily for evaluation of various design proposals. Strive for representative users. Specify where, when and how the users should participate in the development process. Stress the importance of approaching user's in their own work environment. Use a familiar concept terminology for the users. Identify appropriate phases for participation and describe their characteristics, e.g. for analysis, design, evaluation, and construction.

- **Iterative and incremental development.** Continuous iterations of design solutions together with users. One iteration must include: a proper analysis of the user's requirements and the context of use; a prototype design phase and a documented evaluation of the usability of the prototype. The evaluation must result in suggestions for modifications in the following prototype design. Incremental delivery of the system for actual use is valuable. This makes it possible to evaluate the real usage of the system and take action in further increments.
- **Use multidisciplinary and effective teams.** Different disciplines contribute to the whole, e.g. system architects, database gurus, programmers, information architects, usability designers, visual interaction designers, field study experts and so on. It is necessary to set up an effective and multidisciplinary team in order to cover all aspects of the development process.
- **Early and constant focus on users and users' tasks.** This starts at the very beginning of the development and continues throughout all iterations. Everyone in the project must understand the essence of the users, tasks, why and how they perform tasks, etc. This emphasises that you prioritise what is best for the users over what is technically possible. Activities such as user profiling, contextual inquires and task analysis must be a natural part of the development process. Constant focus on users and their

work tasks can be achieved by, e.g. “painting” or decorating the walls of your project room, or rooms, with descriptions of typical users, work tasks and scenarios (e.g. personas from Cooper, 1999).

- **Early and continuous prototyping.** Use paper sketches, mock-ups and prototypes for visualisations of ideas and solutions. The prototyping material is important so start with lightweight paper sketches before implementing anything. Work with prototypes together with users in their work environment—contextual prototyping. Start with the high level conceptual design and do not go into details too early. If it is feasible, work with several prototypes in parallel.
- **Dedicated design activities.** There must be dedicated and conscious design activities carried out by skilled designers. The design of the user interface and interaction is crucial for the success of any interactive system. Too often the design merely happens rather than being a structured and prioritised activity. Remember that to users the user interface is the system. More on this in the chapter *Usability Design*.
- **Evaluate ideas and solutions with end users.** Always establish usability goals and base the design on design criteria. Evaluate against these goals and criteria with users. Early in the development process, users’ reactions to paper sketches and mock-ups should be observed and measured. Later on, users should actually use simulations and prototypes to carry out real work, and their performance and reactions should be observed, recorded and analysed.
- **Take all usability aspects into the scope—integrated design.** Continuous development of all usability aspects in the system: user interface and interaction; on-line help; manuals; user training and education, etc. in the development work. Other parts of the context of use such as: hardware equipment, social and

physical environments, etc. must also be considered in the integrated design. This must be under the responsibility of one person or group. The responsible person (group) must be given the authority to act as an advocate for the users and the usability of the system.

- ❖ **Ensure a user-centred attitude.** A high “lowest level” of usability awareness among all members in the organisation/project/development team. See to that all project members have met real or potential users. Start-up the project with a visit to users’ workplaces. Make sure that all developers join. Baseline a common understanding of the users, their work tasks and work context, and needs. The important thing is that all persons involved in a project are conscious about usability and the users, but the degree will differ from project role to project role and over time.
- ❖ **Usability champion.** Have a usability champion, a.k.a. a usability designer, devoted to the project as an “engine” for the user-centred design process (more on this in the chapter *Usability Design*).

5.6 Discussion about UCD

User-centred design is a process based on an attitude. It demands the “users” of the process (the developers) to be user-centred. I believe that to be really effective, user-centred design must become the standard operating procedure for a developing organisation. Otherwise it will always be questioned and degraded. If user-centred design becomes *the way* to develop systems in an organisation there is no longer a need to speak out loudly about it or to question it. No matter if it is an in-house organisation or some other type of organisation, support from management is crucial to achieve this. Further on, there must be an understanding between the development organisation and the organisation buying the system to work according to a user-centred design philosophy. In our research and practice we see it as important to consider usability and user-centred design as a part of the development process, rather than something that is added on. But there is a long way to go for a development organisation to accomplish this. And in the meantime I think that user-centred design is like democracy: it has to be won everyday. Positive things are that large software producers such as Microsoft have started to talk about and promote *the need* for UCD.

Are development processes such as RUP usable in themselves? From different sources within Rational I have heard the estimation that it takes everything from two up to four years to be really good using the RUP process. Two of the inventors of the only Swedish (as far as I know of) commercially available usability-oriented method—the DELTA method[16], have described what they call the failure of a success story. With a remarkable insight they state the pros and cons of the method. “Methods need to be complex to be marketable, and simple to be implemented. They need a certain level of complexity to merit a name, a handbook, a logotype, and all the other manifestations that are marketable. But they need to be easy if you want to adapt them” (Carlshamre & Rantzer, 2001, p. 36). The principles and heuristics for user-centred design that I have discussed are easy to understand and comprehend, but maybe they are too simple to sell as a method or process. No matter what development process your organisation is using, it has to include the essence of the user-centred design principles if you want to work with a user-centred design approach.

Is a user-centred design philosophy the only alternative system development approach to develop usable systems? The answer to this question seems to be both yes and no. No, in the sense that there are other approaches with slightly different names around. Yes, in the sense that the core of user-centred design is shared among all of them. Processes such as: usage-centred design (Constantine & Lockwood, 1999), contextual design

[16] The method is available at: http://www.deltamethod.net/.

(customer-centred design) (Beyer & Holtzblatt, 1998) and scenario-based design (Carroll, 1995) are all variations on the user-centred design theme.

When it comes to actually getting the opportunity to perform user-centred design, there are a lot of obstacles to overcome. I will just briefly mention some of them:

- There are problems in understanding and recognising user-centred design. State of the art development processes do not honour usability and user-centred design, but organisations think that it can be added on without any cost.
- Lack of competence in usability and user-centred design. These topics are not sufficiently integrated in higher education and practices.
- Usability is often taken for granted and does not get any attention.
- If a client in the tender process (where a client orders a system from a developer organisation) does not specifically order a usable system, i.e. have usability requirements built into the requirements specification, the developer organisation is reluctant to spend any additional resources on making the system usable.

In my research and practice, I have focussed on the development process and tried to challenge some of these obstacles from that angle. Usability design and the usability designer project role, as described in the next chapter, are attempts to do this, but there are obviously also some other barriers to consider when trying to apply user-centred design. Hopefully, I can elaborate on these during my future research and practice.

6 Usability Design

> – *But user-centred design emphasizes that the purpose of the system is to serve the user, not to use a specific technology, not to be an elegant piece of programming. The needs of the users should dominate the design of the interface, and the needs of the interface should dominate the design of the rest of the system.*
>
> Donald Norman in "Cognitive Engineering", 1986, p. 61.

> – *If the designer is doing well, she is removing interface from a product.*
>
> – *…a better designer always creates lots less interface.*
>
> Alan Cooper in "The inmates are running the asylum: Why high-tech products drive us crazy and how to restore the sanity", 1999, pp. 200-201.

When I first started to think about how to make a real change in industrial projects, when it comes to usability and user-centred design, I thought that it would be a minor problem. I mean, it is so obvious that to develop usable systems you have to work with a user-centred design process and a user-centred attitude. Boy, was I wrong! I totally misinterpreted and failed to understand the impact that technology has and what really drives an organisation when it comes to designing system solutions. Even if many of the organisations expressed a willingness to understand and learn more about usability and user-centred design, many of the obstacles mentioned at the end of last chapter were evident. One of the most striking points was that a graphical user interface (GUI) was seen as equivalent to usability and did not really need any special focus in the development process. I found out that it is not possible just to introduce an organisation to UCD and tell them: if you do like this, everything will be perfect and your users will be happy. As discussed in previous chapters, starting to use a UCD process and, further on, to use it on a regular basis, is a great challenge. It surely has to be a step-wise adjustment to the "new" paradigm and for most organisations it will never be a total shift, rather an addition of some activities and methods to their present process and arsenal of methods.

The first result from my research on this matter was the concept and definition of a system development project role that we call *usability designer*. After my research and practice during the mid- and late-nineties I found it necessary to summarise the need for usability and user-centred design, as well as the competence to promote and perform it, into one "easy" concept. This concept, of course, does not cover all aspects of user-centred design, but it was necessary in order to make any change at all. My idea was to introduce the concept on a user-centred design and usability level rather than on, e.g. a user interface design level. The role became a kind of door-opener for talking about these issues in organisations and projects. The role is first described in the paper *Usability designers improve the user-centred design process* and further discussed in the papers *A User-Centered Approach to Object-Oriented User Interface Design* and *Reegineering the Systems Development Process for User-Centred Design*. It was not until I had practiced this role for a while, and introduced it into some organisations, that I found it necessary to formulate an explicit process to go along with the role. In this chapter I will briefly discuss the rough and lightweight framework for user-centred design and usability in practice, and how I have worked with it through my research and practice. I will also look at the role of the usability designer in greater detail.

6.1 What is usability design

Usability design is my attempt to "put a face" on UCD and try to get organisations and projects to start to adopt parts of the philosophy behind UCD. The main rationale behind the concept is that clients (buyers of software development) want the design solution. They are not particularly interested in all the fancy methods and theories that HCI and usability people talk about. I see usability design as an unpretentious and lightweight UCD process that can work in practice. Usability design is to some extent the marriage between usability engineering, and user interface and interaction design. For those familiar with the subject, it is supposed to be the best from two worlds: Jakob Nielsen (usability engineering) and Alan Cooper (interaction design). Usability design is also an expression that Gould et al. (1997) used when describing their principles. In an attempt to define usability design I like to say that usability design is: a user-centred design approach for developing usable interactive systems, combining usability engineering with interaction design, and emphasising extensive active user involvement throughout the iterative process. Usability design is one way to focus more on the solution, the design, than, e.g. usability engineering does, but still have one foot in usability and HCI. Is, then, usability design just another term for usability engineering? No, it is inspired by usability engineering but is focused more on design and integrated user-centred design. It is my experience that, in general, developing organisations have difficulty in understanding the full benefit and potential of "pure" usability engineering. Usability engineering as defined by Preece et al. (1994, p. 722): "an approach to system design in which levels of usability are specified and defined quantitatively in advance, and the system is engineered towards these measures, which are known as metrics.", focuses traditionally on metrics for measuring usability. Here we have too much analysing and evaluating, and not enough of more pragmatic design solutions. On the other hand, just user interface and interaction design is not enough, so usability design may be regarded as weaving them together. An example of how usability engineering is positioned: in recent books on the topic of usability engineering there are rarely any examples of design and how to shape a design, e.g. Usability Engineering by Xristine Faulkner (2000) has a couple of pictures in the whole book showing a user interface, but there is a considerable number of pages with questionnaires, heuristics, methods for analysing users and methods for evaluating. About ten pages out of approximately 230 are dedicated to design (Faulkner, 2000: the section *4.7 Strategies for representing design.*). About 80 pages give a good understanding of usability metrics, usability evaluation, design heuristics and expert evaluation. Another 60 pages tell the reader how to analyse users.

So what about interaction design? I see interaction design as a part of usability design. Interaction design usually covers the navigation and information structure parts of an interactive system and not the whole scope of user-centred design. Also the "usability" in usability design is important since it denotes the strong standpoint in the definition of usability and does not rely solely on the designer's design skills.

To highlight the different views on approaching the subject of user interface and interaction design, the following grouping developed by Wallace & Anderson (1993) is a good guide (see also the paper *A User-Centered Approach to Object-Oriented User Interface Design*):

- **Craft approach:** it focuses on the designer's need for talent, not for methodology. Each design project is viewed as unique, solutions evolve, under the guidance of a skilled human factors expert, to suit the circumstances.
- **Enhanced software engineering approach:** attempts to introduce human-computer interaction techniques into the repertoire of traditional systems engineering.
- **Cognitive engineering approach:** aims at applying theories from cognitive psychology to the problems facing designers and enable optimal design.
- **Technologist approach:** tries to solve the problems of interface design by providing appropriate tools.

My view as reflected in usability design is a kind of mixture of all these approaches, but with a favour for the craft approach and the enhanced software engineering approach. Usability design extends these approaches with the framework of user-centred design.

6.2 A process for usability design

The following figure (Figure 11) attempt to place the usability design process in the overall user-centred design life cycle (developed from ISO/TR 18529:2000(E)):

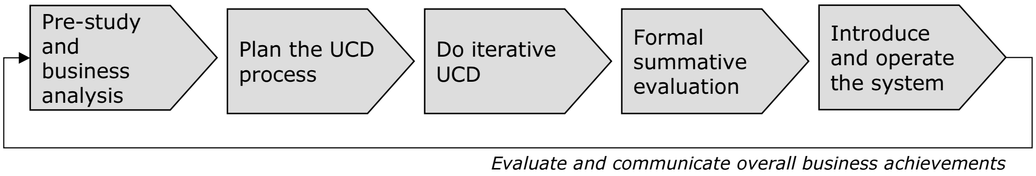

Pre-study and business analysis
Plan the UCD process
Do iterative UCD
Formal summative evaluation
Introduce and operate the system
Evaluate and communicate overall business achievements

Figure 11: User-centred design lifecycle.

Pre-study and business analysis can be anything from making a formal and thorough analysis of work procedures, business processes, etc., to just a statement or vision. Planning the user-centred design process is setting up the project with resources, activities, selecting methods, etc. Usability design matches approximately the box "Do iterative UCD". Summative evaluation is for the overall usability, as opposed to a formative evaluation, which is used within the usability design process to learn about what details in the design that are bad or good (Nielsen, 1993a, p. 170). Introduce and operate the system includes installation, manage changes, deliver user training, determine effect and so forth.

The usability design process is built upon the principles by Gould and ISO/IS 13407, and inherits some concepts from the Usability Engineering Lifecycle by Deborah Mayhew (1999), but has been somewhat simplified and adapted to the type of organisations and projects that I have been involved in. A more explicit focus on design has also been added. In her book, Mayhew (1999, p. 9 and p. 17) states: "**User interface design is key.** I present Usability Engineering tasks in the foreground, with less emphasis on the overall software engineering tasks, to communicate my belief that user requirements and user interface design should drive the overall development process, rather than be driven by it or be incidental to it. After all, the whole point of interactive products is to serve the users, and as far as users are concerned, the user interface *is* the product.". This makes perfectly sense, but, as I have explained earlier, I want to focus even more on the design. This is based on my experiences and knowledge gained from research and practice. Customers or clients, in general, want to have *solutions* (design). Often we try to argue for more analysis and more evaluations, they say fine but give me the solution: now. Therefore a successful user-centred design process must focus on providing the solution—*design*. How much time and resources that are allocated during the different process phases depends on the situation for the individual project. Some times we can do a lot of requirement analysis; sometimes we must just briefly touch that activity and go directly to design.

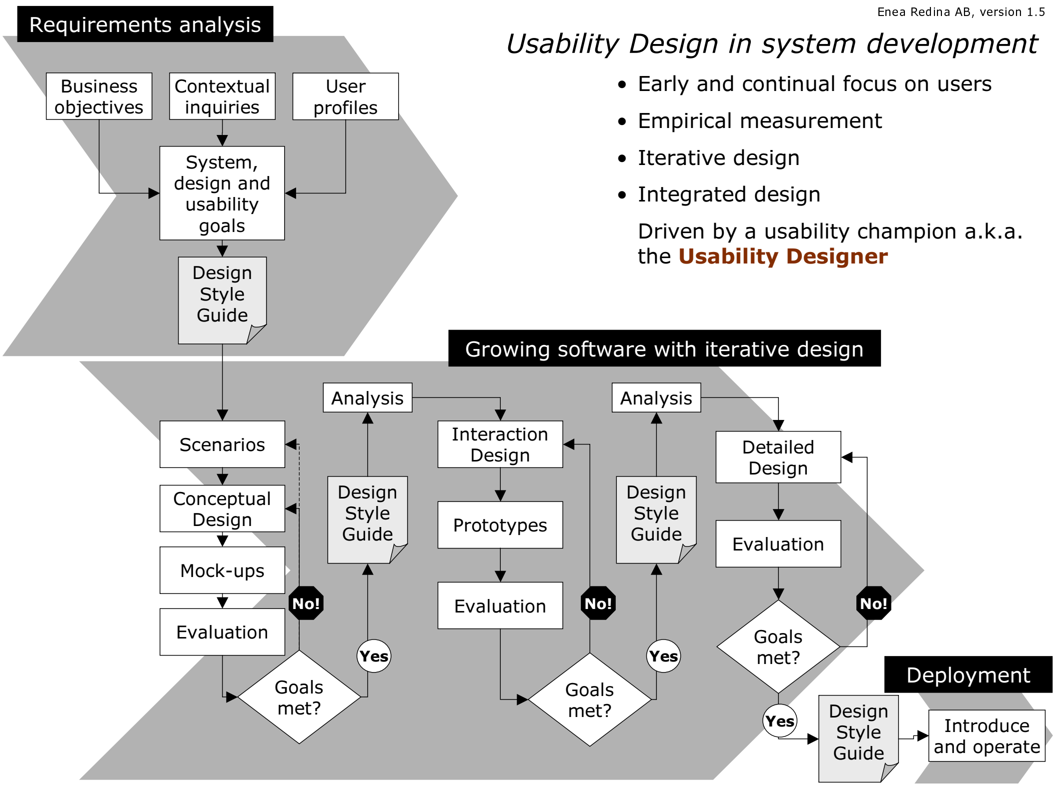
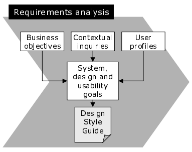


Figure 12: The usability design process.

The figure above (Figure 12) is a result from my research and practice, and describes roughly the steps in the development process that are focused on usability design aspects. The principles for user-centred design by Gould (see the upper right corner of the figure) are there to set the ground for the usability design process. The process should be used iteratively and can be used together with other development processes. The process contains activities that can be carried out with various methods. The exact selection of methods is done in an earlier stage (i.e. Planning the UCD process) and I will not here go into a discussion about which methods that can be applicable. However, for some activities I will give examples of methods. It is still a framework and is not fully developed or evaluated. It serves the dual purpose to be an understandable and communicative artefact, and a process that guides the user-centred design process. So far it has not been my purpose to put forward *the* complete user-centred design process, but as discussed in the section *8.1 Future work*, I have some plans to frame usability design into a more comprehensive user-centred design process.



Requirements analysis

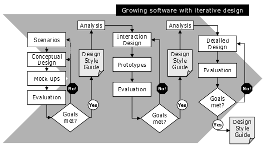
The *requirements phase* is in focus initially and will then, later on during the design, be “visited” when major changes according to functionality, user work tasks, user groups, etc. occur. This is an important statement since the somewhat simplified drawing of the process can be interpreted as: once you left the requirements phase, requirements are never changed (falling back to a waterfall process). This is not true; requirements will constantly evolve during the design process. As long as you baseline the fundamentals of the system, you will have enough guidance to continue the design. In this phase we focus on: understanding the business objectives; the targeted user groups, their work tasks and needs; and establishing system, design and usability goals.

The level and consciousness of business objectives differ radically from organisation to organisation. Some organisations have a complete set of business processes at hand and can precisely point out what kind of support they need to fulfil their business objectives. Others are very vague on their business objectives and might have more of a vision. In the latter case, some work needs to be done to elicit the objectives and make them visible.

There must be a clear understanding of the forthcoming users of the system, their work tasks and needs (as already pointed out in several places above). It is not enough just to realise that there are potential users of the system. The individual user groups have their own characteristics and their own needs. This must be taken into consideration when designing the system. There are several ways to learn about your users. You must at least analyse your users with respect to their background, skills, abilities, etc.—user profiles, and conduct some sort of field studies— contextual inquiries. The contextual inquiries (see Beyer & Holtzblatt, 1998; Mayhew, 1999) reveal what users do at work, their work tasks, goals, working environment, etc. and also to a large extent what they need. The contextual part is very important, as also pointed out numerous times in previous sections. We have also used the concepts of personas and goal-directed design (as described by Cooper, 1999) as complements to user profiles. A persona is "a precise description of our user and what he wishes to accomplish" (Cooper, 1999, p. 123). For a more detailed understanding of personas and goal-directed design, please consult the book by Cooper.

The goals for the system, design and usability are important in order to baseline the usability requirements and the design process. As said earlier, in the chapter *User-Centred Design*, usability goals are not easy to identify and use. We have found the idea of design criteria useful to drive the design process. Usability goals are more of requirements that can be used for acceptance testing or to see trends over time. Design criteria complement them as a tool for the designer. Usually we combine the usability goals with design criteria. They are high-level criteria that the project team agrees on. They are derived from the usability goals, but point more directly at a design decision. It is important that they are agreed on so they will be used and become a useful tool.

The result from this phase, as well as the other phases, can be documented in a Design Style Guide. This is a specific style guide document for *this system*. This Design Style Guide will state the usability design aspects of the system including: user categories, personas (perhaps), work task analyses, usability goals, scenarios, conceptual design and detailed design as well as design decisions, etc.



Growing software with iterative design

The *growing software phase* has three major loops: conceptual design, interaction design and detailed design. The loops in this phase are iterations with design, evaluation, analysis, re-design, evaluation and so on. The design and the system “grows” as we go through the loops. Over time the design will become increasingly detailed. We use usage scenarios to drive the exploration of design solutions. As always, users are participating in these activities. While a process cannot, in itself, guarantee that development will include users, it can encourage a focus on such issues, and provide opportunities for evaluation and user feedback. The conceptual design is a high level base lining of the overall structure of the user interface. Usually it is based on some sort of metaphor, e.g. the work task metaphor as described in the paper *A Teleradiology System Design Case*. The initial conceptual design is almost always done with paper and pencil and illustrated through mock-ups.

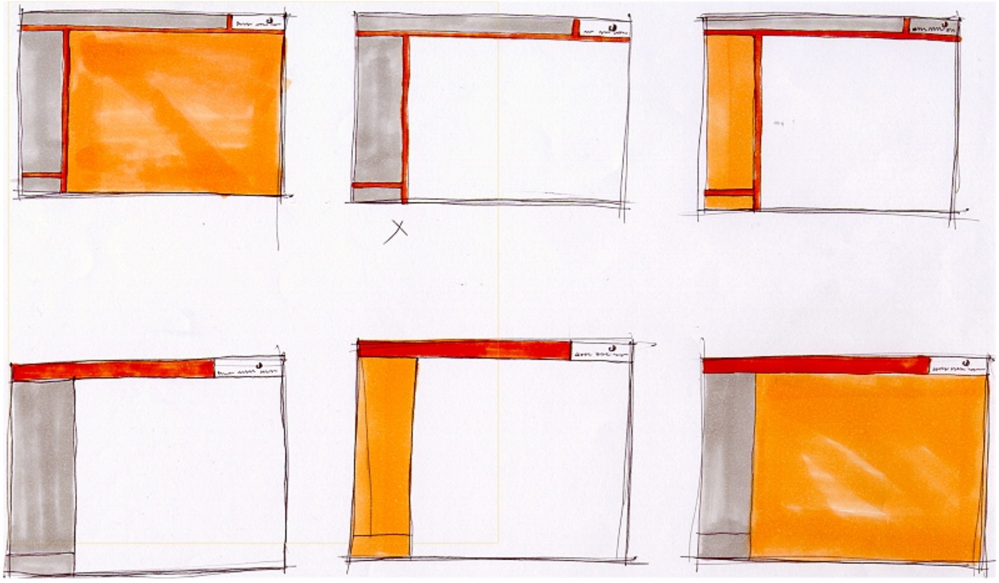


Figure 13: Examples of different, paper bound, conceptual design suggestions for a system.

Figure 13 shows examples of early paper mock-ups for a system. There are six variations on a theme. In Figure 14 (below) one conceptual design is selected and further outlined.

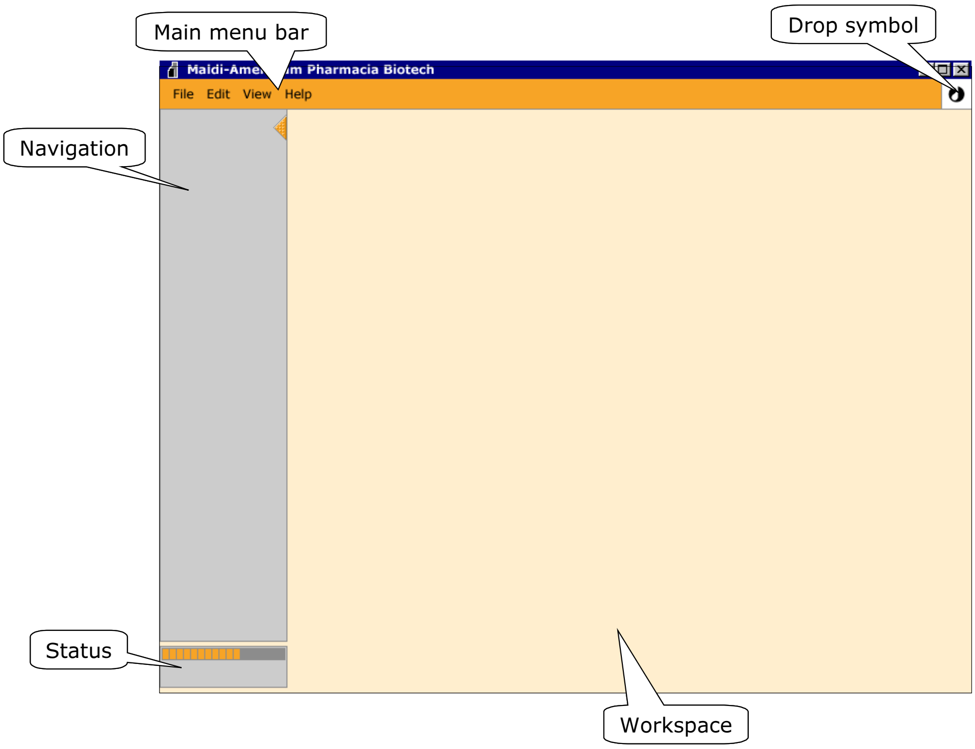


Figure 14: The conceptual design; annotated and outlined.

We give structure to the interaction and navigation as we move on to interaction design. Figure 15 illustrates a conceptual navigation structure for a system.

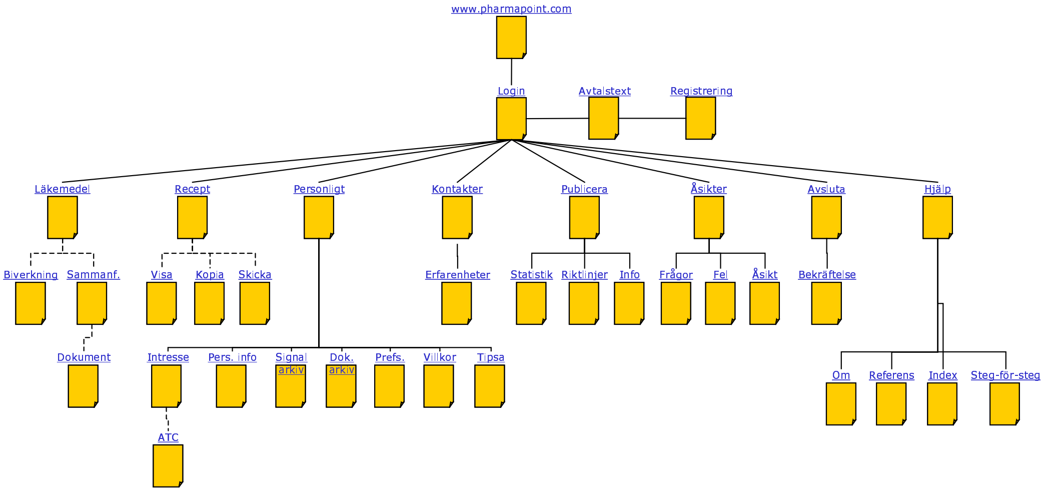
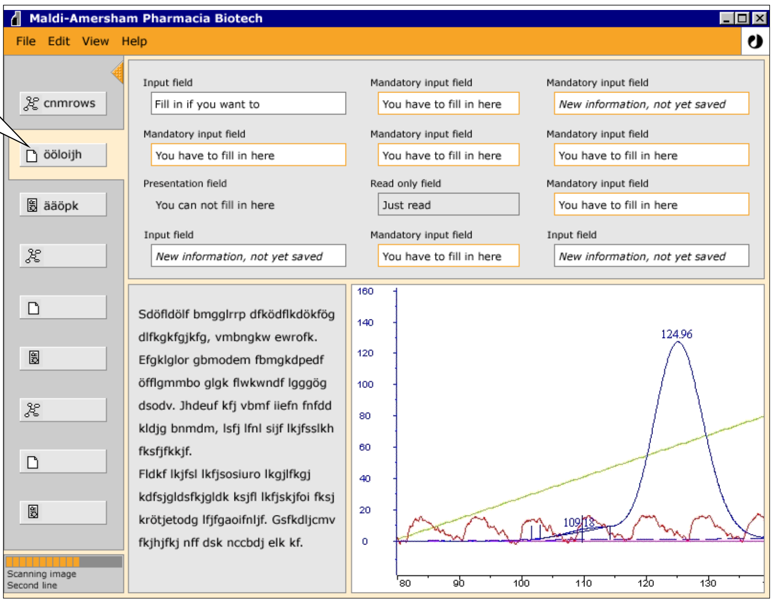


Figure 15: Example of visualisation of the navigation structure.

In Figure 16 chunks of information are identified and put into applicable parts of the user interface, as well as the functionality that comes with it. Navigation paths are identified and lined out.



Navigation with work task buttons

Input field
Fill in if you want to

Mandatory input field
You have to fill in here

Mandatory input field
New information, not yet saved

Mandatory input field
You have to fill in here

Mandatory input field
You have to fill in here

Mandatory input field
You have to fill in here

Presentation field
You can not fill in here

Read only field
Just read

Mandatory input field
You have to fill in here

Input field
New information, not yet saved

Mandatory input field
You have to fill in here

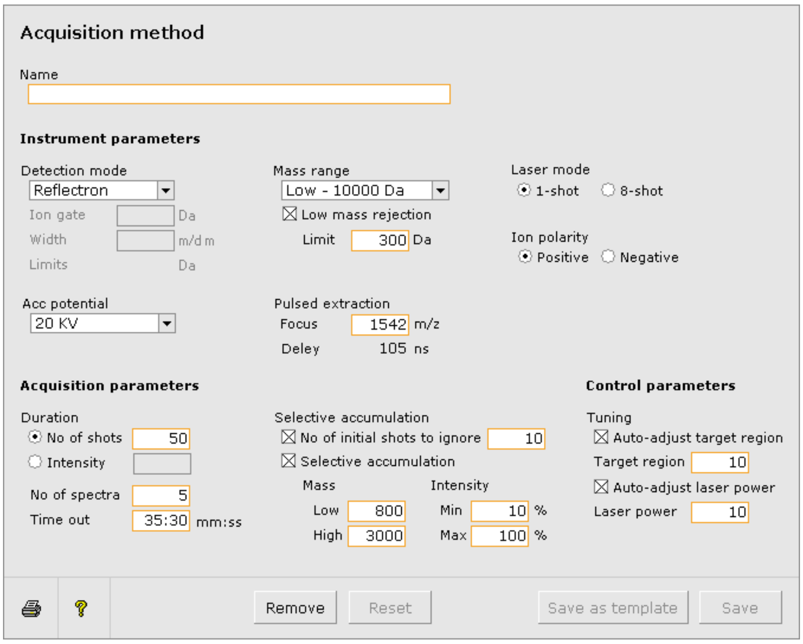
Input field
New information, not yet saved

Sdöfldölf bmgglrrp dfködflkdökfög dlfkgkfgjkfg, vmbngkw ewrofk. Efgklglor gbmodem fbmgkdpedf öfflgmmbo glgk flwkwndf lgggög dsodv. Jhdeuf kfj vbmf iiefn fnfdd kldjg bnmdm, lsfj lfnl sijf lkjfsslkh fksfjfkkjf.
Fldkf lkjfsl lkfjsosiuro lkgjlfkgj kdfsjgldsfkjgldk ksjfl lkfjskjfoi fksj krötjetodg lfjfgaoifnljf. Gsfkdljcmv fkjhjfkj nff dsk nccbdj elk kf.

Scanning image
Second line

Figure 16: Prototype showing concepts from the interaction design.

Detailed design takes us down to individual parts of the screen and to data fields, input fields, menus, buttons, etc. Figure 17 shows a detailed part of the user interface with all information and functions in place.



Acquisition method

Name

Instrument parameters

Detection mode
Reflectron
Ion gate Da
Width m/d m
Limits Da

Mass range
Low - 10000 Da
☒ Low mass rejection
Limit 300 Da

Laser mode
● 1-shot ○ 8-shot

Ion polarity
● Positive ○ Negative

Acc potential
20 KV

Pulsed extraction
Focus 1542 m/z
Deley 105 ns

Acquisition parameters

Duration
● No of shots 50
○ Intensity

No of spectra 5
Time out 35:30 mm:ss

Selective accumulation
☒ No of initial shots to ignore 10
☒ Selective accumulation

Mass
Low 800
High 3000

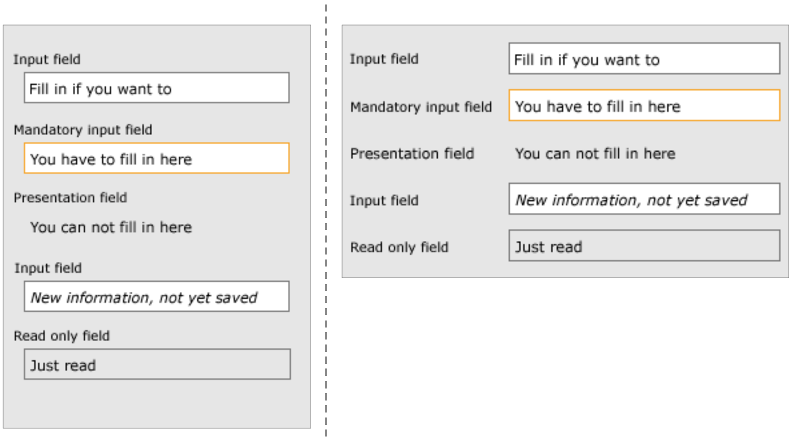
Intensity
Min 10 %
Max 100 %

Control parameters

Tuning
☒ Auto-adjust target region
Target region 10
☒ Auto-adjust laser power
Laser power 10

Figure 17: A detailed part of the system.

In Figure 18 we see examples of detailed data entry fields used throughout the system.



Input field

Fill in if you want to

Mandatory input field

You have to fill in here

Presentation field

You can not fill in here

Input field

New information, not yet saved

Read only field

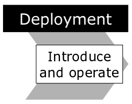
Just read

Input field	Fill in if you want to
Mandatory input field	You have to fill in here
Presentation field	You can not fill in here
Input field	*New information, not yet saved*
Read only field	Just read

Figure 18: Example of design details: making up the standard interaction elements for the system.

Note that there is no strict sequence between the three loops, but we recommend that you start with the conceptual design before going into more details. Hopefully the whole system is developed according to the philosophy of iterative design and growing software (incremental development). In that case the usability design will deliver appropriate parts of the user interface and interaction in increments, otherwise it will be more of a traditional specification for developers to implement.

Each loop contains an evaluation activity, as a natural part of the iterative approach. (Remember the UCD heuristic: continuous iterations of design solutions together with users. One iteration must include: a proper analysis of the user's requirements and the context of use; a prototype design phase and a *documented evaluation of the usability of the prototype* that must result in suggestions for modifications in the following prototype design.) Evaluation against usability goals is important and should lead to actions in the next iteration. It is, for several reasons, not always possible to continue to iterate until the exact goals are met. As mentioned in the section *4.2 Usability goals*, when iterating is stopped within a loop this is related not only to achieved usability goals, but also to other circumstances such as time limits, money and the notion of "good enough". You must not hesitate to do evaluation, but be cautious and sensible about what kind of evaluation method you use and put it in comparison with the overall resources available. It is always better to do a "quick and dirty" evaluation than not do one at all. We usually use rather informal evaluation methods as we start with the conceptual mock-ups. Informal evaluations have not always the purpose to exactly measure the usability against metrics, but rather to get insight into the evolving design. There is sometimes a conflict between the need for numbers and the need for insight. Numbers are fine to communicate the usability status and the need for improvements, the true purpose of usability is to set the design direction, not to generate numbers for reports and presentations: "Thus, the best usability methodology is the one least suited for generating detailed numbers" (Jakob Nielsen in his on-line Alertbox, February 18, 2001: http://www.useit.com/alertbox/20010218.html). I fully agree with Nielsen when he points out that the most effective usability tests are frequent small tests, with users asking them to think out loud while performing work tasks in the system. What we stress is to always evaluate with users, not only the ones already in the project but also with other, real end users. There are plentiful of evaluation methods to choose from, described in for example: Mayhew (1999), Faulkner (2000), Nielsen (1993a), Nielsen & Mack (1994).



Deployment

Deployment, as in introducing and operating the system, differ from project to project. It can be anything from actually packing everything into a commercial available product including: on-line help, user manuals, user training, etc., to "just" distributing an internal CD. My experience is that what happens after the actual development phase is critical for the success of the system. In too many cases a good system has been introduced in the wrong way, with unsuitable, or even no, on-line help, manuals or user training. The deployment will involve different people and include different activities depending on whether the organisation is, e.g. an in-house organisation, product developer or consultant. The most important factor is that deployment must never come as a surprise for the organisation. It must be planned up front in the project and not be seen as something less important that can be taken care of later. Later seems always to be too late. By delivering the system in increments, you will gain the possibility to introduce the system with less resources, but with greater options to smooth and fine-tune the process (and the system), increment by increment.

The process does not end with the deployment; it is rather the starting point for the next cycle in the system's overall life cycle. There will be updates, new versions, etc.

Ideally, the usability design process is carried out by a team led by the usability designer. There are other competences needed as well: visual interaction designers, graphic designers, developers for implementing prototypes, etc. My experience is that the size of this team differs from project to project. Sometimes there are only resources for a single usability designer to perform all activities, sometimes there is room for a larger team. This is also one of the purposes behind the process: it points out the most critical activities to carry out, not exactly how they are done or by whom.

The process is not at all a complete development process. It must most certainly be used in the context of a more comprehensive development process, but ultimately this is the process that underlies the user-centred philosophy driving the whole development project. And, as said in the heuristics for user-centred design, it must eventually be tailored for every organisation. The process does not explicitly express all parts of user-centred design, for example how users shall be participating, but has the underlying message that users are present and active throughout the whole process. Neither are such things as user manuals, on-line help and user training expressed in the process. There are, of course, activities and artefacts that must be present in a user-centred development project, but that would complicate the picture too much for my purposes with this process. There is a fine line between showing too much and showing too little in order to convince people that user-centred design is something that they should go ahead and strive for and actually practice.

Notes on design

Throughout this thesis I have made the point that design is very important, and I have even said that to successfully make an impact in the industry, we must deploy processes that have an explicit focus on the activities that lead to a design solution. In this section I will just make some notes on design without going into too much detail.

As discussed in section *5.1 Introduction*, the term design can refer to both the process of developing a product, artefact or system and to the various representations (simulations or models) of the product that are produced during the design process. When it comes to design in the context of the usability design process, I prefer to talk about the design of the user interface and the interaction (in this perspective I am not concerned about software design and the implementation of the design). In a broad sense, that is interaction design and graphic design.

When it comes to graphic design and interaction design, there appear to be two different worlds in the industry: one developing traditional business systems in proprietary environments such as Windows, with tools like Visual Basic and Java; and the other when doing something for the web. On the web, and particularly when it comes to promoting corporate web sites, graphic designers have had a great impact and we can see excesses in graphic expressions. On the other hand, in traditional business systems there is a striking absence of explicit interaction and graphic design, or conscious design at all. There are some logical rationales behind this. Much on the web is done by persons with a graphic design background in advertising and publishing, but there are fewer persons with skills in interaction and graphic design in more traditional system development environments, and very little attention is paid to graphics and aesthetics at all. I want to promote the usage of conscious and explicit interaction and graphic design through the concept of usability design. We shall always take the opportunity to use graphical representations to enhance the usability of interactive systems, but it must be in line with the overall usability goals. To make a travesty on the famous motto "form follows function"[17] I would say, "graphic follows usability". There are some examples of this in the papers *The Design of a Smart Card-Based Home-Help System* and *Usability and efficiency. The Helios approach to development of user interfaces*, where we used, e.g. non-standard user interface elements (widgets or components) to enhance user effectiveness (and to some extent user satisfaction). Such fundamental concepts from the psychology as the gestalt laws are essential when designing user interface. The gestalt laws formulate a number of principles of perceptual organisation and how certain perceptions are more likely to occur than others. To put it like Paul Mijksenaar (Professor at Delft University of Technology, Holland) in is book Visual Function (Mijksenaar, 1997):

– Design after all has the unique capacity to shape information by

- emphazising or understating
- comparing or ordering
- grouping or sorting
- selecting or omitting
- opting for immediate or delayed recognition
- presenting it in an entertaining fashion

Can a trained and experienced graphic and interaction designer substitute for, e.g. a usability engineering process? Usability engineering believes in the objective "truth" about a system being usable or not. And the only way to know whether it is like that or not is to measure the usability. So no matter if you hire the best and most well known designer to design your user interaction, etc., you still have to measure before you can decide if the designer made a good job. The fact is that a trained and experienced interaction designer does almost the same activities as defined in usability engineering, but she or he makes them in a more informal way. Crampton

[17] According to Paul Mijksenaar (1997) this motto originates back to 1851 from the sculptor Horatio Greenough, but is usually attributed to the architect Louis Sullivan from a lecture in 1896.

Smith and Tabor (1996) describe five activities that any interaction design goes through:

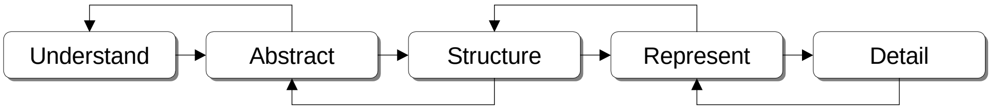


Figure 19: Understand – what is going on? Abstract – what are the main parts? Structure – how do the parts connect? Represent – how can the structure be represented? Detail – what attributes to use? (Crampton Smith and Tabor, 1996).

So one difference is the formalism. Another difference is that the designer usually does all the analysis and evaluation by him- or herself, whilst the usability engineer documents it and makes a point out of making it a part of the documentation of the system. When it comes to practice there are some problems in promoting usability engineering vs. design. Take this scenario: you are in a project and, after long discussions, have convinced the client to hire a designer. Fine, as the project goes along you start to talk to the client about evaluating the design. Of course, the client asks: so I hired the designer you recommended, and now you are telling me that he is no good? So, is it usability engineering or design? I have found that the combination of parts of the formalism in usability engineering and the power of design is one way of endorsing usability and user-centred design—*usability design*. Note the relative resemblance between the usability design process and the process by Crampton Smith and Tabor.

6.3 The Usability Designer

The usability designer development role as a usability champion, and an advocate for usability and user-centred design has, been defined and detailed in the papers *Usability designers improve the user-centred design process, Reengineering the Systems Development Process for User-Centred Design* and *A User-Centered Approach to Object-Oriented User Interface Design*. As mentioned in the beginning of this chapter, we defined this role as a kind of easy way, and sometimes the only way, into development organisations and projects. In this section I will only summarise and discuss the key principles behind this role.

The background to the role was the urgent need for a practical way to really be able to practice usability and user-centred design. During our time in different research as well as in industrial projects, we had noticed the difficulties to bring in—and to practice—user-centred design. We had realised the obstacles for practicing user-centred design mentioned in section *5.6 Discussion about* UCD:

- There are problems in understanding and recognising user-centred design. State of the art development processes do not honour usability and user-centred design, but organisations think that it can be added on without any cost.
- Lack of competence in usability and user-centred design.
- Usability is often taken for granted and does not get any attention.

Further on, when it came to practicing, we had experienced problems in the communication between the user and developer organisation, and to keep the usability in focus during development. Furthermore, we were usually engaged in projects where we could not fully follow a user-centred development process, or dictate the overall project framework. The reality was that we had to try to fit the user-centred activities into whatever system development process the development organisation was using, or believed they were using. Many of the organisations we worked with were not committed to, or even aware of, usability. So we defined the usability designer role as a kind of usability champion on a user-centred design level rather than on a user interface design level, with the explicit purpose not to introduce just another user interface designer, but someone with the capability to work for users' best and user-centred design throughout the whole organisation and in all projects. The role is not an easy one, but when successful, promises to make large impact in organisations and projects.

The concept of a kind of usability champion is clearly stated as a part of the key principle of integrated design by Gould: "In order for this to happen successfully, all aspects of usability should be under one focus or one person" (Gould, Boies & Ukelson, 1997, p. 239). And has been further been promoted in ISO/TR 18529:2000(E) (p. 8): "Act as advocate for end users and other stakeholders in the system development enterprise and the development team. NOTE 1 The stakeholder' advocate reminds the staff in the system development enterprise that the system is intended for use by real people and has to achieve quality in use. This role includes championing human-centred approaches, arranging for end-user involvement in conceptual studies, investigation and dissemination of context of use issues".

What activities does the usability designer perform? The role must be able to perform all activities in the usability design process. In some projects the usability designer does all of them, in other projects there are opportunities to set up teams with more specialised roles. In that case, the usability designer becomes more of a team leader, taking part in the activities, but not necessarily doing all the work. The characteristics of the role are:

- The usability designer is responsible for keeping the development process user-centred, focusing on usability aspects. Planning and performing activities related to the usability design process and making sure that the results of usability activities are further used in the development process, is very important for the usability designer. The role must also be given the authority to advocate for the users by management both in the development organisation and in the user organisation.

- It is crucial that the usability designer takes an active part in the design and development process, and does not only become another project manager. The usability designer can make a large impact by being present in most situations where the design of the system is discussed. By promoting a user's view in every situation, developers and others may always be forced to think twice before doing anything that would compromise usability.

- We emphasise the importance of a person participating in all the user-centred activities, to prevent valuable information from being lost in the transitions between the activities. This is in direct correlation to the principle of integrated design by Gould.
- The usability designer can to some extent be seen as a “discount” usability role, as it combines several skills in one role and in an efficient way copes with the user-centred design process.

One has to make an effort to successfully adopt a user-centred approach. We argue for having an official role that keeps the project’s focus on the users. To improve the development process, it is vital to have an explicit role that has an overall view and responsibility for the usability aspects. The explicity of the role is important since it puts usability on the agenda. Other experiences we have made are that the role boosts the awareness of usability in projects—usability becomes visible. As a result, the users’ status increases in the developer organisation. Coming from the view of practicing user-centred design: it is easier to sell in a combined role than arguing for a full set of specialists. Furthermore it is important that the role is flexible and that one finds the right level of ambition in projects.

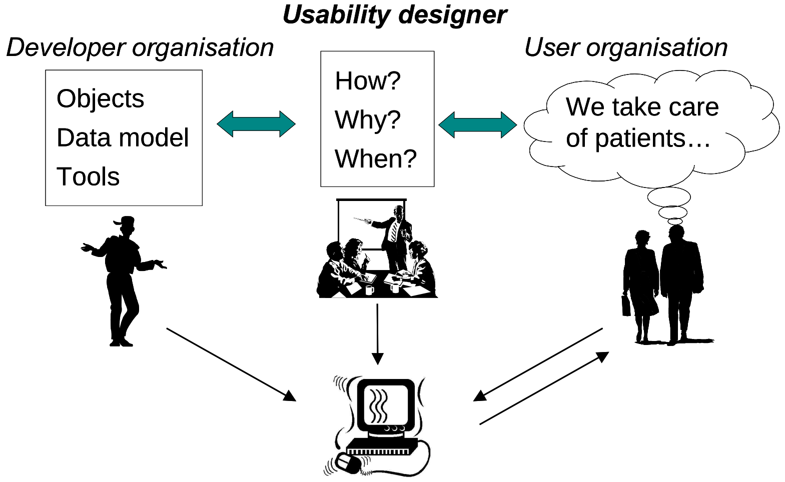


Figure 20: The usability designer as a development role to facilitate the user-centred design process.

The Figure 20 (above) illustrates how the usability designer has a position to ease the user-centred design process. Developers and users will of course also have a direct contact and the usability designer can enhance this contact. The users interact with the system that is designed and built by the usability designer and the developers.

If feasible, the usability designer can work as a team or in pairs with graphic designers, visual interaction designers or other similar roles, with skills in visual and interaction design. Depending on the requirements on the visualisation and the size of the project, the usability designer and the other designer will mix their activities. If the project is rather large the usability designer would most likely focus more on the process of making the system usable, and the other project roles will do more of the practical work such as user interface and interaction design. As an example we have successfully teamed up with visual interaction designers with an industrial designer background. Industrial designers taking part in interaction design is an interesting approach and is outlined in an article by Weed (1996) where he discusses the pros and cons of graphic designers vs visual interaction designers.

So much about the background and the characteristics of the role, but how has it been practiced? It started with us practicing as usability designers. We saw the "vacuum" in projects leaving much of usability aspects to ad-hoc or no attention at all. After a while we started to experiment with the role and try it out in numerous research and industrial projects.

This is a typical setting when carrying out a participatory design workshop with users and a usability designer:

Figure 21: A usability designer and users discussing a paper sketch during a participatory design workshop.

Our experiences so far are mixed but generally positive. There have been successes as well as failures. One problem or failure is that sometimes one usability designer is not enough to ensure a user-centred design approach. On the other hand, in many situations the usability designer covers enough of the user-centred design approach to at least make a significant effect on the system design. One example of a success comes from a project where a usability designer had the opportunity to introduce the notion of usability and user-centred design, although rather late after the system was specified and prototypes were at hand. At this stage, users complained about the usability of the prototypes and that they had too little impact on the design. By arranging participatory design workshops, led by the usability designer, resulting in screen design mock-ups, the development organisation realised that the usability could be significantly improved and they continued to work with the mock-ups that came out from those workshops. Even though the design could have been done as an isolated activity, the usability designer introduced the usability design process as a common ground, and stressed the process rather than an unconnected activity.

The role has been practiced in: in-house organisations such as RSV; product companies, e.g. Amersham Pharmacia Biotech and Pharma Point (service on the web); and consultant companies like Enea. It is very hard to compare the different organisations and how the usability designer has performed in these different contexts. But, there is no doubt that introducing a usability designer into a project puts more focus on the usability aspects. If we compare it to introducing other usability related roles such as user interface designers, we realise that by talking about usability design, rather than user interface design, we gain a greater understanding within the project for the importance of taking usability seriously. Usability becomes more than just user interface design, largely because the usability designer covers more than just user interface design. And I think that this has been the key to the effect and success in many of the situations that we have encountered. The big challenge is to get the usability designer into the organisation and into the individual projects. Here we have some good experiences in that the role seems to make sense to managers and project leaders, and is usually considered easy to grasp and benefit from.

Finally, something about the background and skills we think are essential for a usability designer. We anticipate that a usability designer has a background in computer science or behavioural sciences, the two most influential disciplines in HCI. A candidate for the role must certainly have more than three to five years experience in usability and user-centred design. Knowledge of system development processes and development tools in general is necessary to communicate with the developers and understand what is feasible to do and what is not. The candidate must have some skills in interaction as well as graphic design, not necessarily being a graphic specialist, but a certain feeling for aesthetics.

6.4 Discussion about usability design

The concept of usability design is the conclusion from theoretical studies, practical experiments within projects and our experiences from several development projects. It has been practiced, but is not yet fully evaluated. This is work in progress that I will continue in future research.

Our experience is that the usability designer role is a good introduction to the subjects of usability and user-centred design, and the first step into an organisation. The usability designer becomes in some sense a usability champion that advocates for usability in projects and in the organisation. He/she is also participating in projects carrying out activities according to the principles of the usability design process. I have seen this happen in several organisations. When that role has been established, the more pretentious task of making the whole organisation shift towards a user-centred design philosophy can start. A step in that direction is to have the development organisation adopt the usability design process as one part of their overall development process. This has been the case in at least two of the studied organisations. This is probably the largest impact from my research and practice.

How does this compare to design of web sites? The fundamentals are the same: use a user-centred design philosophy to design usable systems, whether they are on the web or not. Usability design and the usability designer fit into web projects as well. When it comes to corporate web sites, promoting a company, the design and the branding of the label is important, therefore graphic design has a prominent and decisive position, sometimes at the expense of usability as far as user efficiency and effectiveness are concerned. These sites are, thus, more like advertisement than business critical systems.

In a forthcoming book on interaction design, Preece, Rogers & Sharp (2002) elaborate on interaction design in much the same direction as I have portrayed usability design. They mention three characteristics that they state should form what they prefer to call the interaction design process: user focus, measurable usability criteria and iteration. This coincides well with my view on the user-centred design process. Further on, they discuss different jobs or work roles with respect to interaction design. One of the roles is "user experience designer", very much a role like the usability designer, which covers the whole width of user experience or usability. Remember the notion of user experience in parallel to usability mentioned in *4.3 Discussion about usability*. Hopefully their book can be used as, e.g. a course book to take the same standpoint that I have tried to put forward here, and complement more fundamental HCI books.

7 Papers in this thesis

I will in this chapter briefly describe the papers included in my thesis, their origin and my contribution.

i. A User-Centered Approach to Object-Oriented User Interface Design. Jan Gulliksen, Bengt Göransson and Magnus Lif.

> ***Description.*** This is the result of research and practice in many development projects, presented as a chapter in a book. It reviews what we have learned so far. Focus is on the user-centred design process.
>
> ***My contribution.*** My main contribution is the analysis of the development processes RUP and DSDM; the usability designer role and the experiences in promoting user-centred design at RSV. The part concerning user interface modelling (UIM) was essentially prepared by Magnus Lif.

ii. Reengineering the Systems Development Process for User-Centred Design. Jan Gulliksen and Bengt Göransson.

> ***Description.*** This is the result of modelling the system development process at RSV. It describes a method for the modelling and a case study from RSV.
>
> ***My contribution.*** We did the modelling together and with persons at RSV. Jan Gulliksen had the idea of doing this with the organisation's own techniques and I contributed with my knowledge in system development in general and user-centred design in particular.

iii. Usability designers improve the user-centred design process. Bengt Göransson and Torsten Sandbäck.

Description. This is the first of our papers where we tried to describe the usability designer role. It contains two case studies of the role in practice and an early definition of the work activities for the role.

My contribution. This is a total co-operation between Torsten Sandbäck and myself. We both contributed with one case study each and jointly made the definition of the usability designer role.

iv. The Design of a Smart Card-Based Home-Help System. Anders Frisk, Bengt Göransson, Torsten Sandbäck and Vello Thomasson.

Description. This paper is the result of a development project when the four authors still worked at the same company. The main part of the work took place during 1997, and the body of the paper dates back to early 1998, even though we just submitted it to a conference. It is more or less a description of how we conducted user-centred design within a project, with a major focus on visual design. Special attention is paid to the idea of multiple and parallel prototypes.

My contribution. I did most of the planning for the user-centred design process. Participated in all phases: analysis, design and evaluation. My project role was as principal usability expert and designer for the development team—usability designer. I introduced the notion of design criteria, as explained in the paper *A Teleradiology System Design Case.*

v. A Teleradiology System Design Case. Erik Borälv and Bengt Göransson.

Description. This paper describes the design case of the teleradiology application CHILI. It introduces the concept of design criteria as a method for design. Further it describes how criteria and requirements were transformed into user interface design. The concept of a work task metaphor is central as well as detailed user interface design.

My contribution. Erik Borälv and I produced the prototype of the teleradiology system CHILI together in Heidelberg, Germany. We put much of our common ideas and experiences into the prototype. We had no special separation of the different parts of the work. The others in the project referred to us as "the prototyping team" and "the usability experts".

vi. Usability and efficiency. The Helios approach to development of user interfaces. Erik Borälv, Bengt Göransson, Eva Olsson and Bengt Sandblad.

Description. This paper is included in my thesis mostly to give a background to early attempts to discuss UCD. It also contains some fundamentals regarding human-computer interaction and its implications on user interface design. It is a statement on user interface design. The context for this paper was our research group's involvement in a European research project— HELIOS. Our group developed a User Interface Style Guide and tools to support this Style Guide in the larger context of a system-engineering workbench for developing medical domain specific applications.

My contribution. My major contribution within HELIOS was the implementation of the Style Guide in a software tool. We defined a system architecture for supporting efficient development of user interfaces, this included reusable components to ensure a compliance with the Style Guide. This became a preliminary sum-up of my early work related to the more "hard core" parts of HCI—how to implement and construct user interfaces (programming).

8 Conclusions

At this point, the reader may have problems in determining what in this thesis is new undisputable knowledge (the traditional outcome of scientific research), and what is the author's subjective feeling or standpoint. As a practitioner you might ask yourself: what results can I use in my practice? As a fellow researcher you might ask: what is the objective knowledge in the results? My humble answer to these questions must be: *it depends*.

Clearly there are few results in this thesis that can be said to fully comply with the scientific experimental research definition setting up research with high standards of reliability and validity. But if we look at it from the angle of action research, and further from the practitioner-centred research, there is a lot that can be concluded. The usability design process and the usability designer role can be taken into practice by any development organisation. We have done that with the ambition of "usability in every project". As a direct result of this work, usability designers have been engaged in companies and organisations such as: RSV, Pharma Point, Amersham Pharmacia Biotech and Enea. Moreover, there are a number of other organisations now talking about user-centred design, usability and usability design in such a way that they are aware of their importance.

My ultimate goal is to not be "forced" to explicitly talk about usability and user-centred design, but to see it as a totally integrated part of the system development process and the daily routines—standard operating procedure, in the same manner as, e.g. object-orientation is a natural ingredient in system development. To achieve this, we need to make sure that all developers in projects have a user-centred attitude. Can this be done? Karat and Dayton (1995) argue for educating all developers to be skilled in usability. I think this is one way to go, but there must also be specialists such as usability designers. But we are not there yet. We need to take action now and extensive practice of user-centred design is essential to obtain the goal.

It is my hope that my work will inspire individuals and organisations to practice user-centred design and develop more usable systems, and that researchers will benefit from it and further elaborate the factors that lead to successful development processes and practices.

8.1 Future work

I have a list of to-do's and interesting research extensions on my agenda:

- Fully evaluate the effect of the usability designer role in various organisations and projects.
- Evaluate and validate the usability design process.

- Find out the possibilities to establish an education programme or similar for the usability designer role. There are too few persons out there with enough knowledge to become usability designers. My idea is that it should be an add-on to other programmes and have a mix of theory and practice, where practice has a prominent position.
- Put usability design into a framework of a complete user-centred development lifecycle. This is work in progress and I have a rough sketch of a process for this:

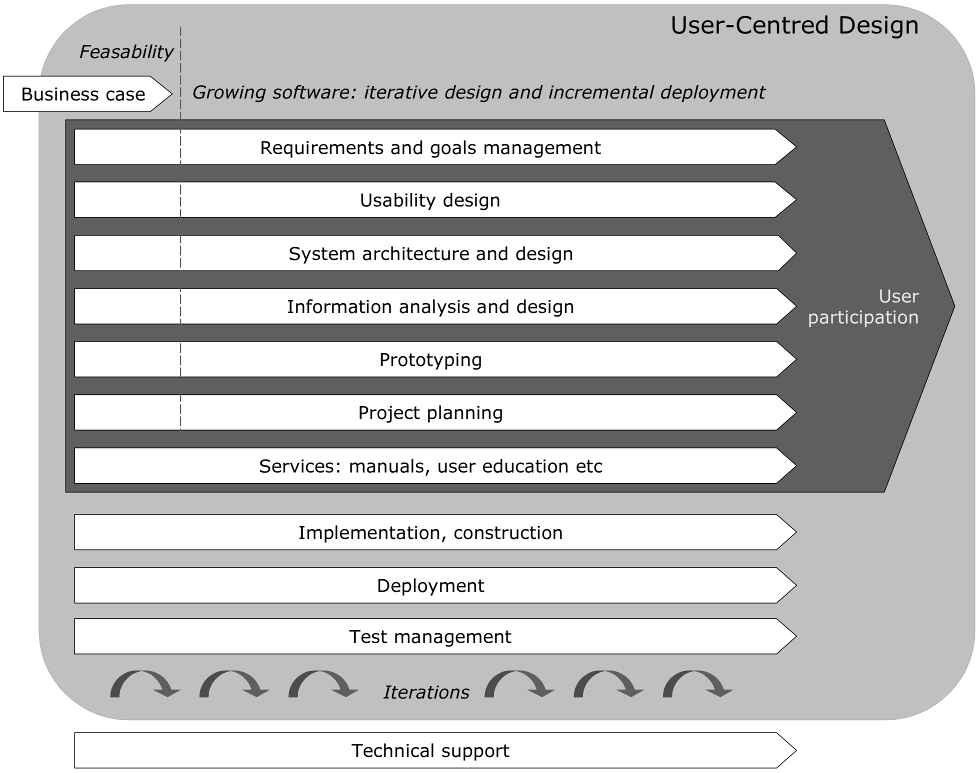

User-Centred Design
Feasability
Business case
Growing software: iterative design and incremental deployment
Requirements and goals management
Usability design
System architecture and design
Information analysis and design
Prototyping
Project planning
Services: manuals, user education etc
User participation
Implementation, construction
Deployment
Test management
Iterations
Technical support

Figure 22: Draft framework for a complete user-centred design process showing workflows from left to right. The process is both iterative and incremental.

This is an extension to the results that we obtained when modelling the system development process at RSV, see paper *Reengineering the Systems Development Process for User-Centred Design*. It is supposed to cover all parts of a system development process with a user-centred design philosophy. I will detail this process and experiment with it in future development projects.

- I will look more into requirements and requirements engineering: how to procure usable systems? The requirements engineering part of the development process is crucial for, e.g. the attitude to take usability seriously. This includes bringing usability aspects into requirements engineering as well as having development processes and work procedures that can facilitate from the requirements. This is pointed out by, e.g. Carlshamre & Rantzer (2001).

9 References

ACM SPECIAL INTEREST GROUP ON COMPUTER-HUMAN INTERACTION (SIGCHI) CURRICULUM DEVELOPMENT GROUP, (1992), Technical Report, ACM, New York.

ALLARD R., LIND M., SANDBLAD B. & SCHNEIDER W., (1984), *Sjukvårdsarbetet och datorn*, (only in Swedish), Arbetslivscentrum, forskningsrapport 45, ISSN 0348-7210 rapport.

BEYER H. & HOLTZBLATT K., (1998), *Contextual Design: Defining Customer-Centered Systems*, Morgan Kaufman, San Francisco.

BJERKNES G., EHN P. & KYNG M. (eds.), (1987), *Computers and Democracy: A Scandinavian Challenge*, Aldershot, UK, Avebury.

BOURNER T. & O'HARA S., (1999), *Practitioner Centred Research*, paper presented at the SEDA/SRHE conference: Research and Practice in Educational Development(s): exploring the links, 14-16 April 1999, available on-line at http//www.lmu.ac.uk/lss/staffsup/confreps/bourne~.htm, download date 2001-01-16. Bourner and O'Hara are at the University of Brighton, England.

BUDDE R., KAUTZ K., KUHLENKAMP K. & ZÜLLIGHOVEN H., (1992), *Prototyping - An Approach to Evolutionary System Development*, Springer-Verlag, Berlin Heidelberg.

CARLSHAMRE P. & RANTZER M., (2001), *Dissemination of Usability: Failure of a Success Story*, in the ACM/SIGCHI magazine interactions, january + februari 2001.

CARROLL J. M., (1995), *Scenario-Based Design: Envisioning Work and Technology in System Development*, New York, Wiley and Sons.

CARROLL J. M. & ROSSON M. B., (1985), *Usability specifications as a tool in iterative development*, in H. R. Hartson (ed.), Advances in human-computer interaction, Norwood, New Jersey.

CATTERALL B. J., TAYLOR B. C. & GALER M. D., (1991), *The HUFIT planning, analysis and specification toolset: Human factors as a normal part of the IT product design processing*, in Taking Software Design Seriously (Karat J., ed.), London, Academic Press.

CONSTANTINE L. L. & LOCKWOOD L. A. D., (1999), *Software for Use: A Practical Guide to the Models and Methods of Usage-Centred Design*, Boston, Addison-Wesley.

CONSTANTINE L. L. & LOCKWOOD L. A. D., (2001), *Structure and Style in Use Cases for User Interface Design*, in Mark van Harmelen (ed.), Object Modeling and User Interface Design, Addison-Wesley, ISBN 0-201-65789-9.

COOPER A., (1999), *The inmates are running the asylum: Why high-tech products drive us crazy and how to restore the sanity*, SAMS, Indianapolis, Indiana, ISBN-0-672-31649-8.

CRAMPTON SMITH G. & TABOR P., (1996), *The Role of the Artist-Designer*, in Winograd T., Bringing Design to Software, ACM Press, New York, New York.

DAGENS MEDICIN, (04/01), *Datorisering går i snigelfart*, (only in Swedish), available online at http://www.dagensmedicin.se/.

DICK B., (1993), *You want to do an action research thesis?*, available on-line at http://www.scu.edu.au/schools/gcm/ar/art/arthesis.html, download date 2001-01-16.

DIX A., FINLAY J., ABOWD G. & BEALE R., (1998), *Human-Computer Interaction*, second edition, Prentice Hall Europe.

DRAY S. & SIEGEL D., (1998), *User-Centred Design and the "Vision Thing"*, in the ACM/SIGCHI magazine, interactions march+april 1998.

EHN P., (1988), *Work-Oriented Design of Computer Artifacts*, Arbetslivscentrum, Stockholm.

FAULKNER X., (2000), *Usability Engineering*, Palgrave, New York, N.Y., ISBN 0–333–77321–7.

GOULD J.D. & LEWIS C., (1983), *Designing for usability*, in the proceedings of the CHI'83 conference Boston, Mass, Human factors in computing systems edited by Ann Janda, Amsterdam North-Holland.

GOULD J.D. & LEWIS C., (1985), *Designing for Usability: Key Principles and What Designers Think*. Communications of the ACM, Vol. 28, No. 3, pp. 300-311.

GOULD J. D., BOIES S. J., LEVY S., RICHARDS J. T. & SCHOONARD J., (1987), *The 1984 Olympic Message System: A test of behavioral principles of system design*, in Communications of the ACM, Vol. 30 No. 9, pp. 758-769.

GOULD J.D., BOIES S. J. & UKELSON J., (1997), *How to Design Usable Systems*, in Helander, Landauer & Prabhu (eds.), Handbook of Human-Computer Interaction, Elsevier Science B.V.

GREENBAUM J. & KYNG M. (eds.), (1991), *Design at Work: Cooperative Design of Computer Systems*, Hillsdale, NJ, Lawrence Erlbaum Associates.

HELANDER M. (ed.), (1988), *Handbook of Human-Computer Interaction*, North-Holland, Amsterdam.

HELANDER M., LANDAUER T. & PRABHU P. (eds.), (1997), *Handbook of Human-Computer Interaction*, Second completely revised edition, Elsevier Science B.V.

ISO/IEC 9126, *Information technology – Software product evaluation – Quality characteristics and guidelines for their use*, International Organization for Standardization, Geneve.

ISO/IS 9241-10, (1996), *Ergonomic requirements for office work with visual display terminals (VDTs) – Part 10: Dialogue principles,* first edition 1996-05-01, ref. number ISO 9241–10:1996(E), International Organization for Standardization, Geneve.

ISO/IS 9241-11, (1998), *Ergonomic requirments for office work with visual display terminals (VDTs) – Part 11: Guidance on usability*, first edition 1998-03-15, ref. number ISO 9241–11:1998(E), International Organization for Standardization, Geneve.

ISO/IS 13407, (1999), *Human-centred design processes for interactive systems*, International Organization for Standardization, Geneve.

ISO/TR 18529, (2000), *Ergonomics—Ergonomics of human-system interaction—Human-centred lifecycle process descirption*, first edition 2000-06-15, ref. number ISO/TR 18529:2000(E), International Organization for Standardization, Geneve.

JACOBSON I., BOOCH G. & RUMBAUGH J., (1999), *The Unified Software Development Process*, Addison Wesley Longman Inc., Reading, Mass., USA.

JOHNSON-LAIRD P. N., (1983), *Mental models*, Cambridge Univeristy Press, ISBN 0 521 24123 5.

KARAT J., (1996), *User Centred Design: Quality or Quackery?*, in the ACM/SIGCHI magazine, interactions july+august 1996.

KARAT J., (1997), *Evolving the Scope of User-Centred Design*, in Communications of the ACM, Vol. 40, No 7, pp. 33-38.

KARAT J., ATWOOD M., DRAY S., RANTZER M. & WIXON D., (1996), *User Centred Design: Quality or Quackery?*, panel session, in the proceedings of the CHI'96 conference.

KARAT J. & DAYTON T., (1995), *Practical Education for Improving Software Usability*, in the proceedings of the CHI'95 conference.

KATZEFF C. & SVÄRD P.O., (1995), *Användbarhet i Praktiken: En enkätstudie*. SISU rapport nr. 95:20, Svenska Institutet för Systemutveckling, available only in Swedish.

KRUCHTEN P., (1998), *The Rational Unified Process—An Introduction*, Addison Wesley Longman Inc., Reading, Mass., USA.

MASTERS J., (1995), *The history of Action Research*, in I. Hughes (ed) Action Research Electronic Reader, The University of Sydney, available on-line at
http://www.behs.cchs.usyd.edu.au/arow/Reader/rmasters.htm,
download date 2001-01-16.

MAYHEW D. J., (1999), *The usability engineering lifecycle, a practitioner's handbook for user interface design*, Morgan Kaufmann Publishers Inc., San Francisco, CA.

MIJKSENAAR P., (1997), *Visual Function*, 010 Publishers, Rotterdam, The Netherlands.

NIELSEN J., (1993a), *Usability Engineering*, Academic Press Inc., San Diego.

NIELSEN J., (1994b), *Heuristic Evaluation*, in Nielsen J. and Mack R.L, (eds.), Usability Inspection Methods, John Wiley & Sons, New York, NY.

NIELSEN J. & MACK R.L., (1994), *Usability Inspection Methods*, John Wiley & Sons Inc.

NIELSEN J., (2000), *Designing Web Usability: The Practice of Simplicity*, New Riders Publishing, Indianapolis.

NORMAN D.A., (1986), *Cognitive Engineering*, in D. A. Norman & S. W. Draper (eds.), User Centred System Design, Lawrence Erlbaum Associates Inc., Hillsdale, New Jersey.

PREECE J., ROGERS Y., SHARP H., BENYON D., HOLLAND S. & CAREY T., (1994), *Human-Computer Interaction*, Addison Wesley Publishing Company, Wokingham, England.

PREECE J., ROGERS Y. & SHARP H., (2002), *Interaction Design. Beyond Human-Computer Interaction*, rough draft picked-up at the Wiley and Sons booth at CHI'01, more information at: http://www.wiley.com/college/preece, Wiley & Sons Inc., ISBN 0-471-40249-4.

RATIONAL SOFTWARE CORPORATION, (2000), *Rational Unified Process*, version 2000 – 0011.

SCHULER D. & NAMIOKA A., (1993), *Participatory design: Principles and practices*, Erlbaum, Hillsdale, N.J.

STANDISH GROUP, (1995), *CHAOS report*, available at http://www.standishgroup.com/chaos.html.

STAPLETON J., (1997), *DSDM – Dynamic Systems Development Method*, Addison Wesley Longman Limited, Essex, England.

WALLACE M.D. & ANDERSON T.J., (1993), *Approaches to Interface Design*, Interacting with Computers, Vol. 5, No. 3, pp. 259-278.

WEBB J., (1995), *Extending the Theory-Practice Spiral: Action Research as Mechanism for crossing the Academic/Professional Divide*, first published in Web Journal of Current Legal Issues in association with Blackstone Press Ltd, available on-line at http://webjcli.ncl.ac.uk/articles2/webb2.html, download date 2001-01-16.

WEED B., (1996), *Visual Interaction Design: The Industrial Design of the Software Industry*, in the SIGCHI Bulletin, Vol. 28, No. 3, July 1996.

WIGANDER K-O., SVENSSON Å., SCHOUG L., RYDIN A. & DAHLGREN C., (1979), *Strukturerad Analys och Konstruktion av informationsbehandlingssystem*, Studentlitteratur, Lund.

WINOGRAD T., BENNETT J., DE YOUNG L. & HARTFIELD B. (eds), (1996), *Bringing Design to Software*, Addison-Wesley.

WHITESIDE J., BENNET J. & HOLTZBLATT K., (1988), *Usability engineering: our experience and evolution*, in Helander (ed.), Handbook of Human-Computer Interaction, North-Holland.

WIXON D. & HOLTZBLATT K., (1990), *Contextual Design: An Emergent View of System Design*, in the proceedings of the CHI'90 conference.

■ [end of summary]

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*.

2001-002 Johan Steensland: *Domain-based partitioning for parallel SAMR applications*.

2001-003 Erik K. Larsson: *On Identification of Continuous-Time Systems and Irregular Sampling*.

2001-004 Bengt Eliasson: *Numerical Simulation of Kinetic Effects in Ionospheric Plasma*.

2001-005 Per Carlsson: *Market and Resource Allocation Algorithms with Application to Energy Control*.

2001-006 Bengt Göransson: *Usability Design: A Framework for Designing Usable Interactive Systems in Practice*.

UPPSALA
UNIVERSITY