

*Master Thesis*  
*Computer Science*  
*Thesis no: MCS-2003:01*  
*January 2003*



# **A GUI Designer's Usability Toolbox**

**Hanna Sjöberg**

Department of  
Software Engineering and Computer Science  
Blekinge Institute of Technology  
Box 520  
SE – 372 25 Ronneby  
Sweden

This thesis is submitted to the Department of Software Engineering and Computer Science at Blekinge Institute of Technology in partial fulfillment of the requirements for the degree of Master of Science in Computer Science. The thesis is equivalent to 20 weeks of full time studies.

### **Contact Information**

Author: Hanna Sjöberg  
E-mail: [hanna77@spray.se](mailto:hanna77@spray.se)

University advisor: Guohua Bai  
Department of Software Engineering and Computer Science

Department of  
Software Engineering and Computer Science  
Blekinge Institute of Technology  
Box 520  
SE – 372 25 Ronneby  
Sweden

## **ABSTRACT**

Usability expresses *how well* a computerised system supports human activity. The human-computer interaction occurs via the user interface, which is constituted by a Graphical User Interface (GUI) to a great extent. The issue is to integrate GUI design and usability engineering with traditional system development in order to slide on the man-machine scale.

The purpose of writing this thesis was to find out how to approach GUI design with respect to stakeholders' interests, technical issues, functional requirements and budget limitations. After having conducted four system development projects of varying nature, the outcome was a usability toolbox with techniques, tools and ideas for all situations. The usability toolbox offers inspiration and guidance for your interactive and creative moments and can support GUI design in any phase of a system's life cycle, in any project.

**Keywords:** GUI Design, Usability Engineering, Human-Computer Interaction.

# TABLE OF CONTENTS

Preface.....	i
Acknowledgment.....	ii
Research partners.....	iii
<b>1. INTRODUCTION.....</b>	<b>1</b>
1.1. BACKGROUND .....	1
1.2. PURPOSE .....	1
1.3. CONTEXT .....	1
1.4. GOAL .....	2
1.5. INPUT AND OUTPUT .....	2
1.6. THE AUDIENCE.....	3
1.7. DELIMITATION .....	3
<b>2. THE USABILITY TOOLBOX.....</b>	<b>4</b>
2.1. WHY A USABILITY TOOLBOX?.....	4
2.2. THE WATERFALL MODEL .....	4
2.3. MY MODEL .....	5
<b>3. “INSPIRATION” .....</b>	<b>6</b>
3.1. INFORMATION ECOLOGIES .....	7
3.2. THE INVISIBLE COMPUTER .....	8
<b>4. “INTERACTION” .....</b>	<b>10</b>
4.1. COMMUNICATION .....	10
4.2. LEARNING.....	11
<b>5. “CREATIVITY” .....</b>	<b>13</b>
5.1. REQUIREMENTS ANALYSIS .....	13
5.2. CONCEPTUAL MODEL DESIGN .....	13
5.3. DETAILED SCREEN DESIGN .....	14
5.4. HEURISTIC EVALUATION .....	14
5.5. USABILITY TESTING.....	14
<b>6. “GUIDANCE”.....</b>	<b>15</b>
6.1. THE ICEBERG ANALOGY OF USABILITY .....	15
6.1.1. Usability Factors.....	16
6.1.2. The User Model.....	17
6.2. GUI HEURISTICS .....	18
<b>7. PROJECT ONE: LOCAL ADMINISTRATION SYSTEM .....</b>	<b>19</b>
7.1. BACKGROUND.....	20
7.2. REQUIREMENTS ANALYSIS .....	20
7.3. USABILITY TESTING BEFORE REDESIGN.....	20
7.4. REDESIGN.....	22
7.4.1. Conceptual Model Design .....	22
7.4.2. Detailed Screen Design .....	23
7.5. USABILITY TESTING AFTER REDESIGN .....	23
7.6. REVISION OF DESIGN .....	25
7.7. CONCLUSION .....	27
<b>8. PROJECT TWO: WEB ADMINISTRATION SYSTEM.....</b>	<b>28</b>
8.1. REQUIREMENTS ANALYSIS .....	29
8.2. CONCEPTUAL MODEL DESIGN .....	29
8.3. DETAILED SCREEN DESIGN .....	30
8.4. USABILITY TESTING.....	30

8.5. IMPLEMENTATION.....	32
8.6. CONCLUSION.....	33
<b>9. PROJECT THREE: CUSTOMER SUPPORT SYSTEM.....</b>	<b>34</b>
9.1. REQUIREMENTS ANALYSIS.....	34
9.2. HEURISTIC EVALUATION.....	35
9.3. CONCEPTUAL MODEL DESIGN.....	35
9.3.1. <i>Information Architecture</i> .....	36
9.3.2. <i>Mock-ups</i> .....	36
9.4. DETAILED SCREEN DESIGN.....	37
9.5. CONCLUSION.....	38
<b>10. PROJECT FOUR: MOBILE APPLICATION.....</b>	<b>39</b>
10.1. MOBILE USABILITY.....	39
10.2. REQUIREMENTS ANALYSIS.....	40
10.3. CONCEPTUAL MODEL DESIGN.....	40
10.3.1. <i>Task Analysis</i> .....	40
10.3.2. <i>Information Architecture</i> .....	41
10.3.3. <i>Screen Map</i> .....	41
10.4. DETAILED SCREEN DESIGN.....	41
10.5. CONCLUSION.....	42
<b>11. EVALUATION OF THE USABILITY TOOLBOX.....</b>	<b>43</b>
11.1. "INSPIRATION".....	43
11.2. "INTERACTION".....	44
11.3. "CREATIVITY".....	44
11.3.1. <i>Requirements Analysis</i> .....	44
11.3.2. <i>Conceptual Model Design</i> .....	44
11.3.3. <i>Detailed Screen Design</i> .....	44
11.3.4. <i>Usability Testing</i> .....	45
11.3.5. <i>Heuristic Evaluation</i> .....	45
11.4. "GUIDANCE".....	45
11.4.1. <i>The Iceberg Analogy of Usability</i> .....	45
11.4.2. <i>GUI Heuristics</i> .....	46
11.5. TESTIMONIALS.....	46
11.5.1. <i>End-users</i> .....	46
11.5.2. <i>Customers</i> .....	46
11.5.3. <i>Colleagues</i> .....	47
<b>12. CONCLUSION.....</b>	<b>48</b>
12.1. DISCUSSION.....	48
12.2. FUTURE RESEARCH.....	49
<b>13. REFERENCES.....</b>	<b>50</b>
13.1. LITERATURE.....	50
13.2. ARTICLES.....	50
13.3. WEB SITES.....	50
13.4. OTHER.....	51

## PREFACE

I see my mother struggling with our videocassette recorder. She cannot figure out how to program it to record a television show.

I see my dentist blushing when he hears that I study computer science. He resignedly admits that he knows nothing about computers. He regrets that he asked me about what I am studying. Now he feels stupid. But in his next breath, he comes with suggestions like “*Why do you not develop good user interfaces for the disabled?*” and he presents all his propositions.

I see myself caught in a trap from which I cannot escape. I have troubles with buying a train ticket in a machine. I get confused when I am heating my food in the microwave oven. I cannot get the right temperature from the water tap. In these kinds of situations, I blame myself. I think that it is my fault that I cannot use this technology. But I do not want to feel this ashamed anymore. It is time for a more humane technology.

\* \* \*

I am writing this thesis because I want to work *with* people, *for* people.  
I want to look beyond technology.

Technology can be a good thing.  
Technology is essential for our existence. We cannot live without it.  
But the limit between love and hate is sometimes small. *I love you, then I hate you!*  
If technology is not adapted to you, you will dislike it or you will fear it.

Computers are great. They are so unbelievable.  
I have saved so much time and effort thanks to my computer.  
And I have had so much fun!  
But how many times have I not wished that it were dead?  
Sometimes I believe that my computer has a life of its own.  
I cannot figure out how to tame it, though I almost have a Master Degree in Computer Science...

We are not machines. Machines are not humans.  
You may say artificial intelligence, but I say no.  
We have life.

*The human race is filled with passion.  
And medicine, law, business, engineering are noble pursuits and necessary to sustain life.  
But poetry, beauty, romance, love, these are what we stay alive for<sup>1</sup>.*

How can technology be adapted to a world where human beings rule?

People use technology via the user interface. That is why I have chosen to focus on the design of user interfaces that bring two worlds together; that allow people to enjoy technology!

---

<sup>1</sup> Mr Keating in “Dead Poets Society”, Touchstone Pictures.

## ACKNOWLEDGMENT

It has been a tremendous pleasure to write this thesis, not only because I got the opportunity to dive straight down into an interesting subject, but mainly thanks to all the lovely people with whom I had the privilege to work:

### *Blekinge Institute of Technology:*

Thank you Guohua Bai, my supervisor, for your guidance and encouragement to do my very best. Thank you also for being the supervisor in “Advanced Topic in Computer Science”, which helped me broaden my view.

Thanks are also due to the International Office and the Department of Software Engineering and Computer Science for giving me the opportunity to go abroad and work with my thesis in Ireland.

### *Kreativum:*

Thank you Anders Strange, Marie Fransson, Sven Burreau, Daniel Holmström and Lotta Söderholm of Kreativum for giving me the chance to realise my projects and for all your help.

Thank you Monica Andersson of my former school, Korpadalsskolan, Åsa Nisbeth of Klockebackskolan and Rigmor Wickström of Hästaryd skola for your interest in my project and for taking part in usability testing.

### *Spokesoft:*

Thank you Brendan Lawlor and Stephen Lawlor for letting me into the “nerd world” and for giving me the opportunity to show you the world of usability.

### *Wireless in Motion:*

Thank you John Coleman, Ger Nunan, Richard Moore, Eoin Clayton, Hilary Stephens and Lyanne O’Sullivan for all your help and assistance, inspiration and cheerful moments.

### *National Software Centre / Corkbic:*

Thanks are due to the people in National Software Centre in Cork, Ireland. Especially thanks to Eileen Moloney of Corkbic.

\* \* \*

Without all of you, this work would not have been possible to finish! I hope you will enjoy reading the thesis because I owe it to you.

## RESEARCH PARTNERS



### **Blekinge Institute of Technology**

Applied information technology in focus, situated in “The Garden of Sweden” with three campus sites: Karlskrona (Telecom City), Ronneby (Soft Center) and Karlshamn (NetPort).

URL: <http://www.bth.se/>



### **Kreativum**

Science centre in Karlshamn, Sweden, inviting both young and old to explore the world in a totally new way. Here you can play and learn, understand and wonder.

URL: <http://www.kreativum.se/>



### **Spokesoft**

Customer Support Technology Specialists, based in Cork, Ireland. Spokesoft offers component-based design and development of Web Services on the Java 2 Enterprise Edition platform.

URL: <http://www.spokesoft.com/>



### **Wireless in Motion**

Specialized in customer-orientated solutions for the mobile data access market with offices in Cork and Belfast. Develops software for pocket pc:s, premium text messaging and other SMS services.

URL: <http://www.wireless-in-motion.com/>



### **Corkbic**

Cork BIC assists the creation and fostering of enterprise in the Cork and Kerry regions of Ireland and provides hands-on assistance to entrepreneurs and small businesses.

URL: <http://www.corkbic.com/>

## 1. INTRODUCTION

This chapter provides an introduction to the subject and background information on why this area was chosen. Additionally the research model is put forward.

### 1.1. BACKGROUND

This thesis is a continuation of my Bachelor Thesis from 2001. The investigation for the Bachelor Thesis consisted of redesigning and deliberately manipulating a Graphical User Interface (GUI) in three different versions. Usability testing showed that by just adding GUI Standards and applying known principles for GUI design, the level of usability was improved considerably in this particular case. However my colleague and myself wrote the following in the conclusion [26]:

*"Moreover, we would like to emphasize that effective GUI design is more than just following a set of rules. There is no cookbook approach that can rely on general principles and guidelines alone. Every GUI and its intended set of users are unique. Design guidelines must be tailored for and validated against the users' requirement, which can be accomplished through a user-centred attitude and design methodology".*

One could argue that there are at least three deficiencies with the way the investigation was conducted in 2001. Firstly, the redesign of the GUIs did not benefit from user feedback. All decisions were taken based on the designers' own experience and on GUI standards and guidelines. Secondly, the test users who took part in the usability testing were not potential end-users of the system. Finally, usability issues did only concern what was on the screen, when in fact the GUI is just a part of the user interface, which could be considered as the *entire user experience* of using a computer program.

### 1.2. PURPOSE

There are many structured methods for traditional system development, e.g. the waterfall model, and famous notation systems like UML, but there are few clear methodologies for GUI design. The purpose of this thesis was to try to establish a complement to traditional system development that is both time-efficient and easy to integrate with the creation of the rest of the system. Being a GUI designer involves a wider understanding of software projects than just visual design.

### 1.3. CONTEXT

This thesis treats GUI design from a usability point of view. Usability is about *how well* a system supports human activity and is a wide concept that depends on many factors. The GUI is with no doubt crucial for achieving usability since the *GUI is the system* for many users.

The GUI must be fitted into a context to be usable, meaning that the GUI actually is there for a good reason and helps the users do what they want to do. It does not matter how fancy or efficient a GUI is, if there is no request for it – it has to be *useful* [9].

Consequently, in the pursuit of usability, the whole context must be examined as well. Additional insights that are specific to the project in question are required, e.g. stakeholders' interests, technical issues, functional requirements and budget limitations. The question is thus how to integrate GUI design with traditional system development so that usability issues are not overlooked.

\* \* \*

In order to conduct an investigation I pursued four system development projects in cooperation with companies in Sweden and Ireland. The projects were in different phases of the system's life cycle and either aimed to evaluate and redesign an existing GUI or to develop a new product. The products concerned were web-, desktop and wireless applications.

## 1.4. GOAL

These were the distinct goals of this thesis:

- Define the “usability toolbox”, i.e. a collection of techniques, tools and ideas available for a GUI designer in order to ensure usability.
- Investigate *how* and *when* the components in the usability toolbox are applicable in various phases of traditional system development.
- A secondary goal would be to evaluate the level of usability if feasible.

The focus can be summarised in the following research questions:

- How to approach GUI design in a structured but yet flexible way?
- What are the resources available for a GUI designer to utilize in order to ensure usability?

## 1.5. INPUT AND OUTPUT

The input of this work was existing research, lectures and the curiosity of knowing the answers to the research question. This inspired me to formulate the usability toolbox, e.g. a set of resources for a GUI designer. By applying it in various projects, a relation could be established to the output of this thesis which was a revised usability toolbox (figure 1).

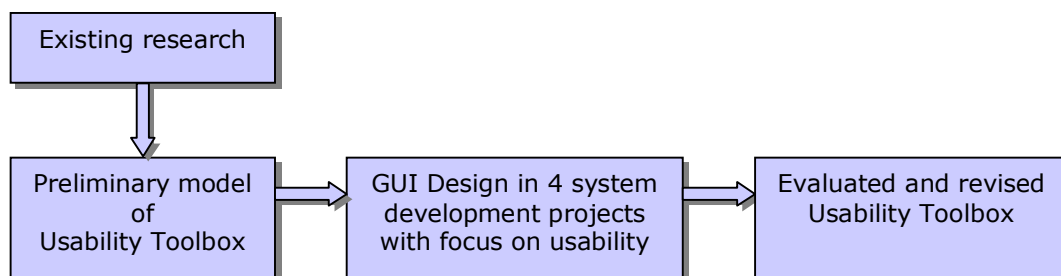


Figure 1. Input and output in the process of writing this thesis

## **1.6. THE AUDIENCE**

Since GUI design and usability issues have to trade off against a variety of requirements, this thesis should not only be of benefit to GUI designers, but to all members in the development team and organisation management.

## **1.7. DELIMITATION**

The scope of this thesis was to investigate the application of a usability toolbox in various system development projects with regard to GUI design and no other factors that might contribute to system usability.

## **2. THE USABILITY TOOLBOX**

A usability toolbox is according to my definition a collection of resources available for a GUI designer. Depending on the task, an appropriate “tool” can be chosen. Some tools may be used more often than others, and some may be of a rather abstract nature. During the system development process, these usability tools could be employed in the traditional activities to ensure an integrated work with the rest of the development team. Ideally a tool, e.g. the usability engineering principles, will move the whole development team towards a more usability-oriented attitude.

### **2.1. WHY A USABILITY TOOLBOX?**

There exist a number of approaches to achieving usability in system development, e.g. user-centred design, the LUCID framework and usability engineering. However, after having conducted the literature study for this thesis, my general impression was that none of these approaches were flexible enough. They offer many interesting ideas but one could argue that the techniques proposed seem to be far too time-consuming and not very realistic to bring onto a software project that has a limited budget. For instance, usability engineering as defined by Deborah Mayhew [7] includes sixteen steps during the entire development process. Additionally, existing methods are generally based on the assumption that the system is in its initial phase, when in fact most systems require maintenance as well. However, usability engineering suggests a great deal of techniques that might be worth adopting.

Introducing a usability toolbox could imply a more flexible approach to GUI design. There is a need for such a way of working since GUI design might occur in various phases of a system's lifecycle. The usability toolbox should be possible to be utilized in any phase of the system's lifecycle, e.g. either from scratch in the creation of a new product or in the maintenance phase to “clean up”.

### **2.2. THE WATERFALL MODEL**

The usability toolbox cannot be considered as a methodology, but rather as a complement to traditional system development. The waterfall model is a generic methodology for system development and was chosen as an example since it covers the entire life cycle of a system. It is often referred to as a system's development life cycle model. The waterfall model is quite linear where the typical steps are sequential in the sense that they feed each other with information. Consequently one phase cannot begin before the precedent one has been finished (figure 2). However, experience shows that the process can be quite iterative and that phases do overlap [16].

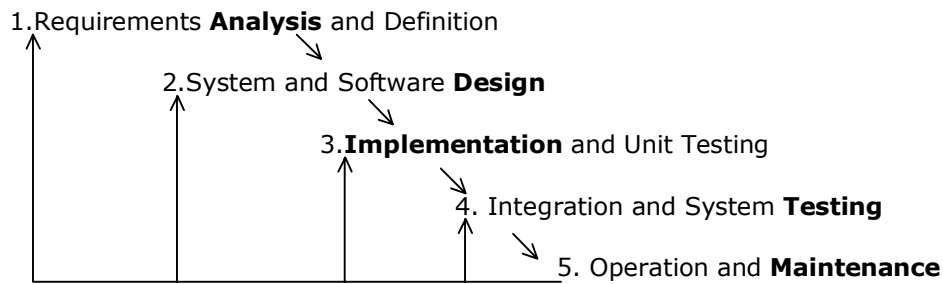


Figure 2. The Waterfall Model [16].

### 2.3. MY MODEL

The issue in this thesis is *how* and *when* the usability toolbox can complement traditional software development and contribute to system usability. I am aware of the fact that the metaphor of a toolbox may invoke the wrong association because the usability toolbox is an abstract expression for a set of tools, techniques and ideas for GUI design and should not be taken as a physical representation. The concept of “tool” can be defined as something you need to use to do your job, as a means of achieving a goal [21]. Hence, a usability tool in this thesis has a semantic meaning.

The usability toolbox was on basis of the literature study and related subjects at college. My model is divided in four main categories which each represent distinct applications (figure 3):

- **“Inspiration”** is a collective term for the rather philosophical ideas that the usability toolbox offers. This part can hopefully allow system development and GUI design in particular to slide on the man-machine scale, since my belief is that the focus throughout a system’s life cycle should alternate between humans and technology.
- **“Interaction”** is the part of the usability toolbox that provides support for various kinds of interaction in a system development project, e.g. the interaction between developers and stakeholders and the “Human-Computer Interaction”.
- **“Creativity”** is the most straightforward contribution of the usability toolbox since it contains techniques for the design process itself. This is the concrete work that you do!
- **“Guidance”** is the part of the usability toolbox that will guide your design so that it can benefit from your own and other’s experience and research.

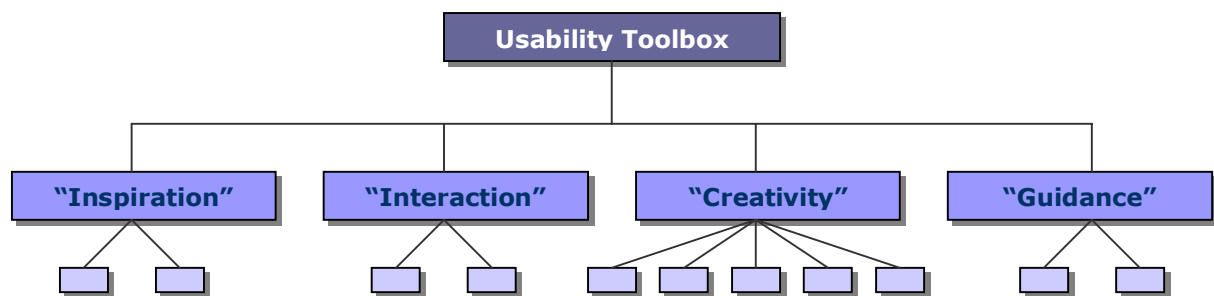


Figure 3. The usability toolbox with its four main categories and sub-components.

With this separation, the usability toolbox can offer support throughout the entire development cycle, in any phase, for any task. The four categories consist of components, that are described in the chapter of each category.

### 3. "INSPIRATION"

This part of the usability toolbox is thought to be a somewhat philosophical contribution to system development and GUI Design. It can hopefully inspire to a more humane technology. These usability tools are abstract but can possibly help to understand the environment in which a system is going to exist or the role an application is going to play. Such an understanding is vital to be able to tailor a GUI to its context.

\* \* \*

How we look at technology can definitely affect the way we design technical artifacts. Technology has become a significant part in our lives. Our homes are filled with technical devices and tools. We change along with our domestic environment as technology makes itself more prominent in our everyday life.

Technology is not something new. It is as old as the history of the human being. We are creative and curious, but also lazy. We are therefore keen on inventing new tools. But technology has evolved incredibly quickly over the last decades and has become extremely advanced in some areas. We keep being astonished of how applicable technology can be. The fact that television and telephones have been integrated into our lives may be natural to all of us now, but a more recent phenomenon like information technology has not yet been fully accepted [22]. We simply do not know where it will end? The technical possibilities seem so endless. What will happen in the future? Will our colleagues be robots? And as we keep on being more dependent on technology, we wonder if it is reliable. Many people are suspicious of the new information technology and use it with confusion, fear and resistance [22].

Where is the limit between man and computer? Is there any? Technology and man become more and more one. I guess several people have found a new identity on the Internet in chatting rooms. Their new digital friends are somewhere out there in cyber space. Shall we make technology more human or shall human beings become more technical?

Bioinformatic researchers claim that the human being resembles a computer more than we think. The DNA in the human body consists of information of how to build proteins. This information is arranged in sequences with a beginning and an end with the instructions in between, just like programming code<sup>2</sup>.

Yes, and you may say artificial intelligence, but here I say stop. We are not machines and vice versa. I do not promote avoiding technology. I just mean that we should use technology with heart – which is the subtitle of the book "Information Ecologies" by Bonnie Nardi and Vicki O'Day [8].

---

<sup>2</sup> Meidanis, J. & Setubal, J.C. *Introduction to Computational Molecular Biology*. Brooks/Cole Pub Co, 1997.

### 3.1. INFORMATION ECOLOGIES

An information ecology is a system of people, practises, values and technology in a particular local environment. The focus is not on technology, but on human activities that are served by technology [8].

The characteristics of an information technology are:

- **System:** has like a biological ecology strong interrelationships and dependencies among its different parts.
- **Diversity:** different kinds of people and different kinds of tools.
- **Co-evolution:** an ongoing evolution, which people ought to take part in, as newer, faster and different tools and services are offered by the technological development.
- **Keystone Species:** are most often represented by skilled people whose presence is necessary to support the effective use of technology.
- **Locality:** identity and place of technology that is influenced by the inhabitants in the ecology through unique knowledge about the local environment.

Information ecologies have *values* and goals depending on their activities [8]. Generally any place of work can be an information ecology, e.g. a library, a hospital intensive care unit (but not the *whole* hospital) or a copy shop. Humans help other humans use technology.

If services or tools do not correspond to the values of the information ecology that they are introduced into, they will remain meaningless since they do not fit into the context. They do not add extra value to the information ecology. In that case, they could never be *usable*.

There is an anecdote about two foreign men who visited a so-called "intelligent house" in Ronneby, Sweden. They did not dare to use the toilet because there was a big mysterious device near the water closet. They might for example have feared that it was a camera. We can be impressed by lights that turn off automatically and pens that ask you if you want coffee, but never forget that such devices should be adapted to a context and support activities by human beings.

As the values in an information ecology change, we might accept technical solutions due to circumstances that we would not otherwise. For example, if a new system is about to be introduced at my place, I do not want my home to look like an office. But if I get sick, I may accept turning my home into something more than just my home, namely a hospital.

The metaphor of information ecology can help us understand the environment in which the future users of our computer system work. The way we introduce technology into people's lives decides the design of the user interface. Designers ought to look on their task from a wider perspective than just considering *one* person working with *one* computer:

*Information technology is becoming an environment within which people operate, rather than a device they pick up and use[17, p40].*

### 3.2. THE INVISIBLE COMPUTER

The personal computer (PC) is perhaps the most frustrating technology ever. Faster, more powerful and "better" software releases constantly add to an already enormous complexity. But it is not too late to make a change. Technology is easy to change. The difficult aspects are social, organizational and cultural. We should no longer be trapped in a world created by technologists for technologists[11]:

*People are analogue, not digital; biological, not mechanical.  
It is time for a human-centred technology, a humane technology[11, p.viii].*

The computer industry of today is technology-driven and has a long way to go until the goal of a human-centred product development has been reached. Megabyte, gigabyte, terabyte. Kilobaud, megahertz, gigahertz. More capacity, more speed, more power. But how much of this do the customers understand and do they need it (figure 4)?



Figure 4. Three megabytes of ram? (Parisi, M.)

The term information appliance expresses well what should be the obvious aim for every designer, namely to make an artefact feel like a natural extension of the work and not dominating the activity. The vision of information appliances is to overcome complexity and to introduce simplicity as opposed to a personal computer, that indeed helps you to get your work done but at a cost of complexity and inconvenience. [11].

An information appliance is specialized in information (knowledge, facts, graphics, images, video, sound etc) and is designed to perform a specific activity and has the ability to share information among themselves [11].

However, information appliance does not necessarily mean that there should be separate devices for e-mail, calendar, address book, calculator, back-up of files, writing program, music player, movie player, software editor etc. The issue is that in some cases we do not need this "Swiss Army Knife", whose tools indeed are useful but not always ideal. The good

thing about information appliances is that you can have it all – when you want to. For example when you travel, it may be convenient to bring a laptop. But at your office or at home, you can chose the ideal and optimised tool for the task [11]. The current trend is that mobile phones get more and more features and will possible end up as the new PCs in an even tinier format.

Besides the PC there are plenty of computers around us already that we may not be aware of: microwave ovens, watches, coffee makers, dishwashing machines, telephones, cars etc. But when computers are embedded within such devices they do not require their users to know anything about their existence, they remain *invisible computers* [11].

## 4. "INTERACTION"

How do you communicate with your fellow development team mates and your customers? Do they get your message? Do you understand them? Communication is just as complex as human relations. Can we make things clearer by understanding how we learn and how the world around us affects our learning? How can theories on communication and learning make us better system developers? This chapter tries to find out.

### 4.1. COMMUNICATION

Good communication is vital in software projects. Just the way of working in projects suggests constantly meeting new people, thus communication with the users and other stakeholders on one hand and communication with one's colleagues on the other hand.

Communication is the base of all relations and the foundation of our personalities. It is a process of two or more people who send messages between each other, both deliberately and unconsciously. The social context in which the communication takes place determines the interpretation of the messages [10]. The codes may be ambiguous with cultural roots, which can render communication more difficult than it already is.

System developers are often challenged to make their customers express requirements of the computerised system to be. In such situations, a crucial issue is to make a real effort in respecting the opinions of one another and try to look from the other party's point of view. A good pedagogue speaks the customers' language in order to let everybody understand, even those who have no knowledge of computers.

Communication occurs both directly and indirectly between system developers and customers. Direct communication happens when they discuss during the development phase, which should be the case constantly during the entire process as advocated by human-centred designers [17]. When the product is finished, they communicate indirectly via the user interface when the user is using the system. The concept of *system image* is described in section 6.1.2.

\* \* \*

Even before we speak to new people, we might already have an opinion of them based on our own knowledge or on prejudices. This fact is important to take into consideration, because stereotypes and prejudices matter. They influence our communication. We cannot help it. Categorising people is a way of keeping order in the social reality [10].

During a project within the subject called "Pedagogy for System Developers" at Blekinge Institute of Technology, my colleagues, Jeanette Eriksson and Thomas Bladh, and I conducted the investigation "System Developer, who are you?" which aimed to clarify what people think about system developers<sup>3</sup>. We were curious of the view of us of business administration students and programmers, our possible future co-operators. It showed that it does not exist that many prejudices against us that we expected. On the contrary, the interviewees underlined important social characteristics like the ability to integrate with people. Yes indeed some mentioned an antisocial young man with greasy hair and thick glasses, but the general

---

<sup>3</sup> The extent of the investigation is not sufficient to classify the report as reference literature.

picture of a system developer was an open person with broad knowledge within computer science.

It is valuable to know what your future customers might think of you in order to improve communication and avoid misunderstandings [10]. We too need to sort our own categorisations out [4, p. 44]. E.g. it would perhaps be a mistake to assume that all female computer users like the colour pink and that all male users are "computer nerds".

## 4.2. LEARNING

The very first experience of using a computer program will probably affect the user's opinion of the program for a long time. You never get a second chance to make a first impression. The challenge in designing an artefact lies in building on users' existing knowledge in order to prevent a *breakdown*, i.e. bringing down the user's level of competence. By bringing users onto the development team, they can gradually get trained in using the new artefact [2]. During training with users, the system developer requires knowledge about the *process* as well as about the *content*. If you do not know the content, then how can you manage the process? And the other way around; you have to know how to spread your message.

There is a constant interaction between the psychological inner sphere of an individual and the social environment. Simultaneously as we struggle to take in or adapt new knowledge to the cognitive schemes in our brain, a social learning occurs as well. The social aspect is one dimension of learning and the other two are of a cognitive and a psychodynamic nature (figure 5).

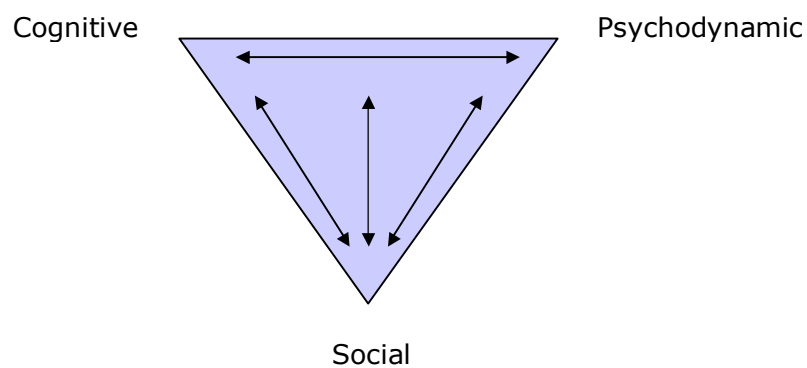


Figure 5. The three dimensions of learning [5].

All learning is *social learning*, regardless of what role you have in the learning process, since there is an interplay of the individual and the rest of the world [5]. We adapt to the prevailing norms in the society, which says how we can behave and how the world is perceived today. We also learn to learn from each other, how to handle conflicts and we get a deeper knowledge of ourselves.

The cognitive dimension can be represented by the concepts of *assimilation* and *accommodation* [5]. If I believe that there exist ten programming languages, but somebody tells me that there are eleven. Then my existing knowledge is expanded and the new fact is incorporated in me. This is *assimilation*. Contrary, if it appears that I have a total misunderstanding of object-oriented programming, then my whole world falls apart and I

need to restructure the existing schemes in my brain in order to acquire the correct knowledge. This is called *accommodation* and can be a very hard process, since it requires energy to overcome "the mountain".

Learning to use computers can be classified as assimilation as well as accommodation, depending on the existing knowledge and motivation of the individual. The slogan for beginners goes:

*Learning to use computers is like learning a new language (new vocabularies and skills). Learning about computers will probably change your life, at least little. Consider it an adventure [1]!*

Our thoughts always have an emotional dimension, meaning that we cannot separate them from our feelings. Consequently, our attitude towards the content will affect the learning. That is why aesthetical aspects matter in usability engineering [19]. The unconscious plays a significant role since it influences our motivation. Lacking in psychological energy can evoke a resistance towards the issue in question, which leads to misunderstandings and the intended learning fails. This psychodynamic factor is tightly connected to the cognitive and the social aspects and can only be separated theoretically.

\* \* \*

User interface designers should be aware of the fact that people learn differently and possibly provide alternatives for both beginners and expert users. But we must take into account common factors before we can deal with the differences among individual humans [14]. A user interface should promote advancing from one level of skill to another. It could be useful for a designer to know how users learn to perform new tasks and if they consult manuals or their colleagues when problems arise [4].

## 5. "CREATIVITY"

This part of the usability toolbox is intended for the concrete and creative design work. A collective term for the below described techniques could be Usability Engineering. There are however a variety of names (*User-centred Design, the LUCID Design Framework, Human-centred Design, User and Task Analysis, Interactive System Design.*) to describe almost the same thing: focus on users during design of software systems. I prefer to call it *usability engineering* like Jakob Nielsen and Deborah Mayhew, simply because usability is the heart of the problem around which we engineer.

Good design rarely happens by chance, so this is a suggestion of how to approach GUI design in a structured but yet flexible way. Depending on the phase of the system's life cycle, the appropriate technique can be selected. All techniques may not apply in every project, since tasks vary.

### 5.1. REQUIREMENTS ANALYSIS

The Requirements Analysis typically starts with defining the situation of concern, which is the objective of the design. The problem statement can be as short as one sentence long and includes the following parts:

- **Identifying the human activity that the proposed system will support**  
Making a description of tasks that together contributes to achieving the goal of performing the activity [6].
- **Identifying the users, who will perform the activity**  
Besides knowing human performance and behaviour in general, also consider individual characteristics to meet users' needs, for example work experience, educational level, age, previous computer experience, reading and language skills, which can be defined by questionnaires, interviews and/or observations[9].
- **Setting the system's usability**  
Deciding the levels of support that the system will provide. The usability goals are based on the user profile and can be both quantitative and qualitative [7].
- **Selecting the basic form of solution to the design problem**  
Specifying how the support is to be made available, including the user interface, the application software, the operating system, system resources and the hardware [6].

### 5.2. CONCEPTUAL MODEL DESIGN

The conceptual model establishes the overall architecture of the user interface and the way the users will interact with the software [24]. The purpose of conceptual model design is to make sure that the user's mental model of how to use the program corresponds to the designer's conceptual model and that the user interacts properly with the interface [6]. Another objective is to ensure consistency and simplicity in high-level user interface design. Mock-ups can be produced on paper or with aid of software [7]. Support for Conceptual Model Design are for instance *information architecture* [25], which is a hierarchical organisation of content and features of a system, and *task analysis*[4], which sequentially describes how users perform their tasks manually. Brainstorming sessions can be helpful to collect data, with or without stakeholders.

### **5.3. DETAILED SCREEN DESIGN**

The detailed user interface design brings it all together, so that the conceptual model and the visual screen design fit together as a whole and form a user interface that meets the usability goals. Evaluation can be conducted on the "real, implemented" product [7] or on a functional prototype.

### **5.4. HEURISTIC EVALUATION**

Heuristic evaluation is an inspection technique that can be useful when resources are not available for usability testing (see Section 5.5.). Guidelines and heuristics are well-known principles for interface design, which not only are useful in guiding design decisions, but they are also applied to evaluation [6].

### **5.5. USABILITY TESTING**

Usability testing is a structured method for evaluating the usability factors and measuring the usability goals. Testing can be conducted in formal labs, but most commonly in the users' real working environment. *Qualitative tests* provide a deeper understanding of users' behaviour and problems and can be performed early in the development cycle. Another advantage is that it is a cheap method, since only up to five test users are required. A disadvantage is however that the result is not measurable. On the contrary, *quantitative tests* enable validation against numeric goals. Progress can easily be shown and comparisons with other similar products can be made. The con is though that many test users are required to achieve a reliable statistical result [23].

Usability testing can provide valuable feedback on how users perceive a web site or any other kind of user interface. However, in spite of the potential savings, profits and revenues, most of Sweden's 50 major companies do not know the impact of their web sites according to a recent survey [18].

## 6. "GUIDANCE"

This part of the usability toolbox provides guidance for GUI design in terms of heuristics, i.e. known usability principles, and research such as theories on the user's mental model of a system.

### 6.1. THE ICEBERG ANALOGY OF USABILITY

Usability applies to all aspect of a system with which a human might interact [9]. Usability describes *how well* people perform activities with the aid of computerised systems [6]. A well-designed GUI can contribute to usability to a great extent, since the GUI *is* the program from a user's point of view.

There is more to GUI Design than screen design. Even though the screen is a critical aspect of any user interface, it is only the surface aspect of complex behaviours [3]. "The Iceberg Analogy of Usability" describes how visuals, interaction techniques and the so-called user model contribute to usability (*Figure 6*). The iceberg metaphor is powerful, because we all know what happened to Titanic. This metaphor can help us explain the different aspects of GUI design and also stress on the fact that usability comes from more than what we can see.

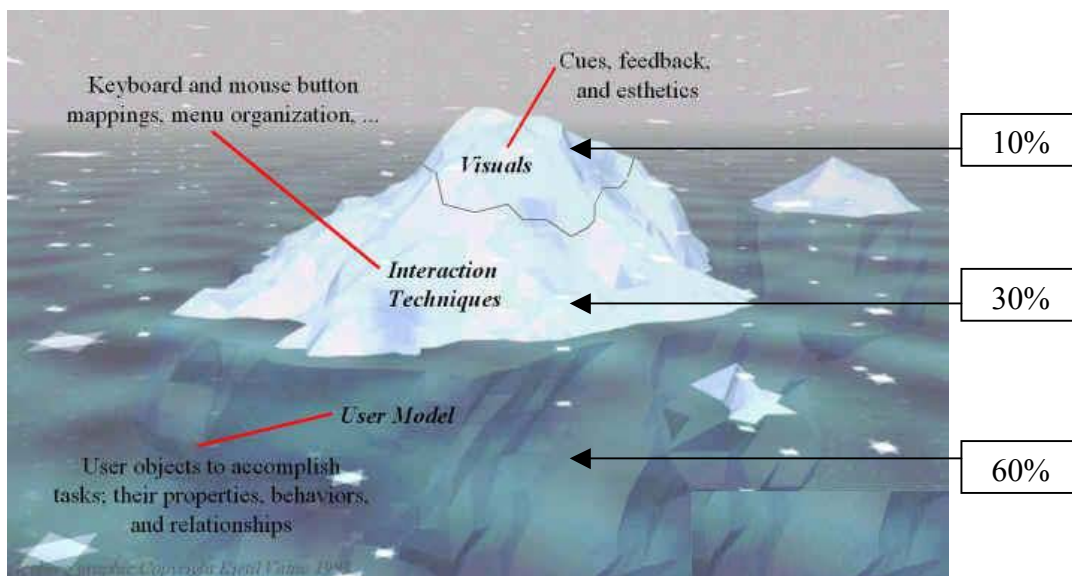


Figure 6. The Iceberg Analogy of Usability [20].

Visuals refer to visual cues, feedback and aesthetics and many lessons can be learned from the design of everyday things, e.g. a three-dimensional button has a high "visual affordance" if it invites to pushing and should provide appropriate feedback when an action is acted upon it [12]. Aesthetics are a subjective matter and includes the use of colour, line styles, typography etc. Surprisingly enough, the "look-aspect" is only estimated to contribute 10% to the overall usability [20].

Interaction techniques concern for example keyboard and mouse mappings, menu structures, shortcuts and navigation. 30% of the usability come from the "feel-aspect" of a user interface [20]. The "look and feel" depends most often on the platform's user interface toolkit and style guide.

The user model plays the major role in of achieving usability, namely 60%. This concept represents user objects to accomplish tasks, their properties, behaviours and relationships [20]. Since the user model is "below the surface", its construction and behaviour are not obvious to see. Normally, when the expression "below the surface" is used, it means that a phenomenon has occurred because of underlying factors. For example, if you want to understand why something is not working as expected, why people are unsatisfied etc, you must look behind formal rules and regulations. This part is often the most difficult one to deal with, since the problem may be of a social and psychological nature.

To achieve usability, the user interface has to correspond to the user model of how to use a program. If the GUI is consistent with the user model, then the GUI will work as the users expect it to do. The different parts of a user interface are highly dependent on one another, since the design propositions should have their origin in the *user model* (see section 6.1.2).

### 6.1.1. Usability Factors

Usability is a measure of *how well* a system supports human activity [6]. It is a multi-dimensional property and its definition varies with authors. Usability is defined accordingly in this thesis:

- **Learnability:** the users' ability to learn how to use the system.
- **Speed of performance:** the amount of time to perform a typical task.
- **Incidence of errors:** how many errors that occurred when performing a typical task.
- **Subjective satisfaction:** the users' opinion of using the system.

The above mentioned usability aspects together are not always the goal. One or a couple of the factors might be crucial, and the others of minor importance. Furthermore, the factors affects each other and special attention should to be paid to any conflicting aspects. It is common to associate usability with learnability and to claim that systems should be easy to learn. That opinion can be discussed, since it is a matter of context. It is evident that simple tasks should stay simple, but complex tasks may require complex user interfaces [14].

Usability put in a wider perspective reveals the endlessness of Human Computer Interaction (HCI), which is a multi-disciplinary area that includes computer science, design, philosophy, cognitive psychology etc [13]. Ideally all of them are required to achieve the ultimate goal of system development, namely system acceptability, whose attributes are shown in figure 7<sup>4</sup>.

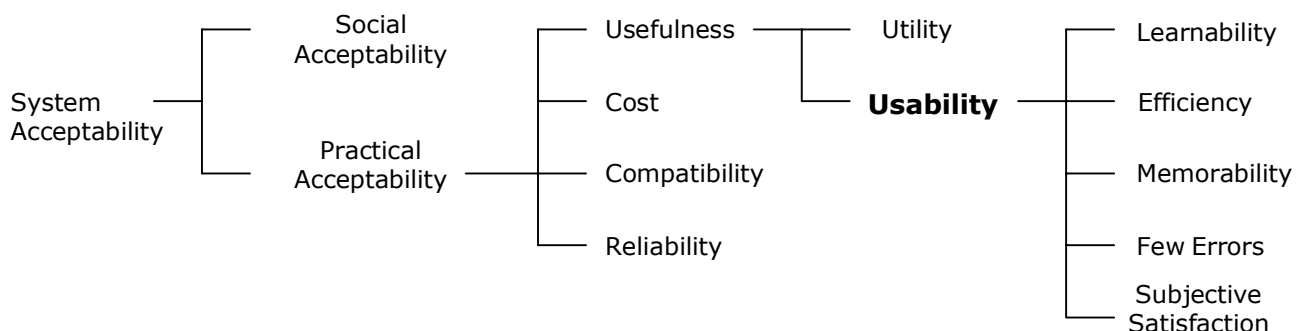


Figure 7. The Model of System Acceptability [9, p.25].

<sup>4</sup> The usability factors as defined by Jakob Nielsen

*Usefulness* concerns the degree to which a system enables a user to achieve his or her goals, and is an assessment of the user's motivation for using the product at all [15]. It can again be broken down into utility – whether the functionality of the system in principal can do what is needed - and usability – how well users can use that functionality [9].

### 6.1.2. The User Model

The iceberg analogy of usability is based on the belief that people have mental models of the world, which also apply to software. Users form their models on the basis of previous experience and the parts of the system they come into contact with, *the system image*. Designers construct a mental model as well and the relationships are shown in figure 8:

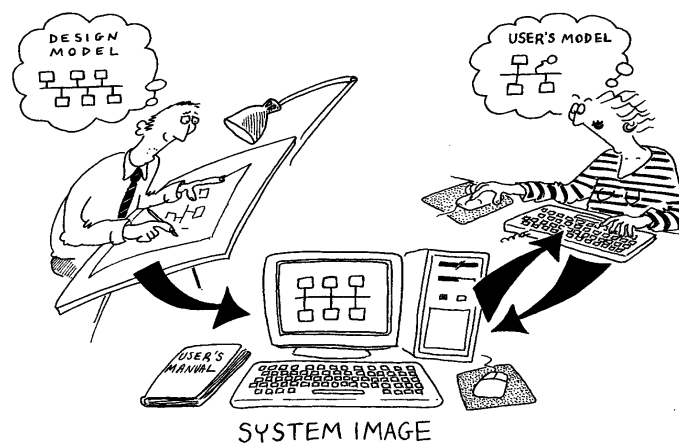


Figure 8. Three Models of a system [3].

The user model is based on a user's observations of the system's external behaviour, on her own or others' experience of using the system. When interacting with a user interface, the user can run the mental model and ask herself a set of questions to reconstruct the most recent events to evaluate the current situation and decide whether the desired effect was achieved [6]. The cycle of interaction, featuring the execution stages to the left and the evaluation stages to the right, is shown in figure 9.

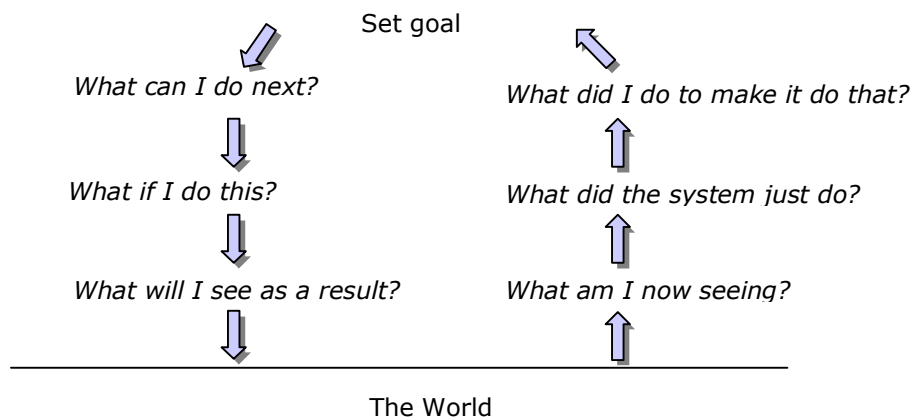


Figure 9. The Cycle of Interaction [3].

Misunderstandings can be avoided if the designer respects all stages in the interaction cycle and help the user build the adequate mental model [6]. One reason of why misunderstandings and errors might occur is that designers and users do not share the same conceptual model of the system and that designers speak their own language as shown in figure 10:

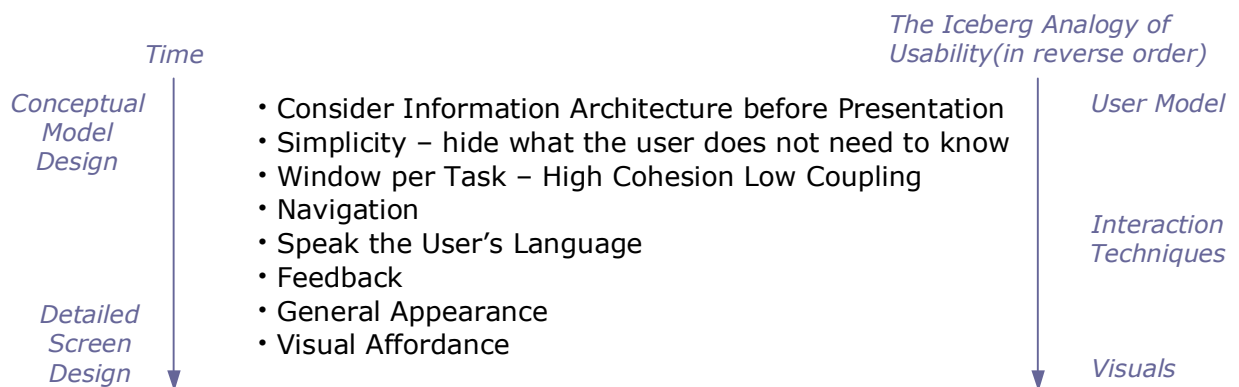


DILBERT reprinted by permission of United Feature Syndicate, Inc.

Figure 10. Saint Dogbert (United Feature Syndicate, Inc.)

## 6.2. GUI HEURISTICS

GUI Heuristics are based on known guidelines and standards for GUI Design. Whereas there exist hundreds of guidelines out there, heuristics are more likely to be applied since they are fewer and more general. They provide advice on usability characteristics of a GUI. They are derived from literature and experience and can be redefined depending on the type of user interface to be. The following eight heuristics mirror the "Iceberg Analogy of Usability" in reverse order and are here put in relation to the design activities as time goes by:



## 7. PROJECT ONE: LOCAL ADMINISTRATION SYSTEM

This project aimed to redesign the GUI of an existing administration system, which would handle categories and questions to a quiz game. Figure 11 presents how the usability toolbox was utilized. As the system was tailored to one specific customer and a limited number of users, the ideas of “information ecologies” helped to understand the environment in which the system was going to be used. The “Interaction”-part was particularly useful in the direct contact with the customer and end-users whereas the “Creativity”-tasks correspond to the concrete work carried out. Additionally, the usability toolbox provided guidance that was valuable when redesigning.

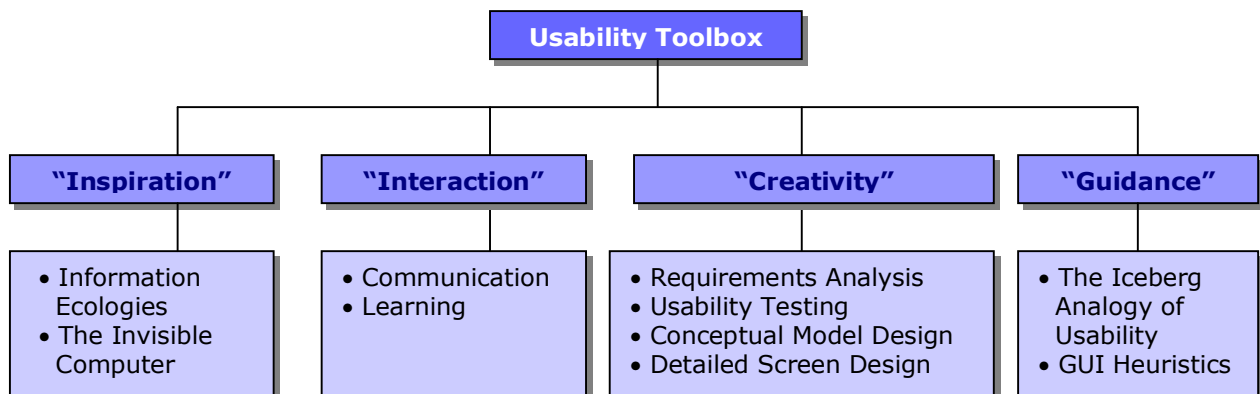


Figure 11: Parts of the Usability Toolbox utilised in this project.

The focus of this thesis is not only *how*, but also *when* the usability toolbox can be applicable in relation to traditional system development, e.g. the waterfall model. As illustrated in figure 12, this project took place in the maintenance phase and the design process was quite iterative.

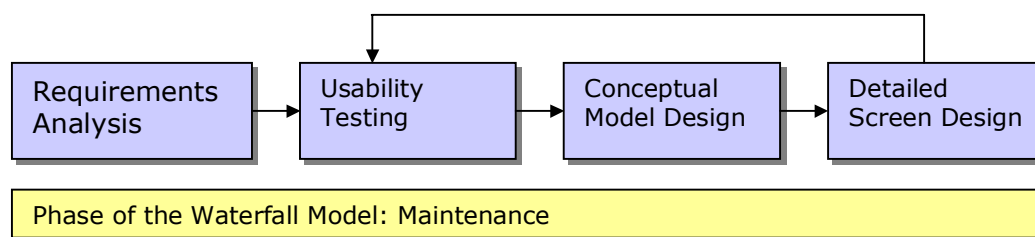


Figure 12: Design phases in relation to the waterfall model

## **7.1. BACKGROUND**

The system, whose GUI was going to be redesigned in this project, was developed in 2000 by a group of students including myself. Regarding GUI design, this was first made when the implementation started. All GUI design decisions were made by the developers without any deeper analysis or feedback from end-users. No structured approach to GUI design was adopted and not even considered due to a lack of knowledge of known usability principles.

The GUI was used as test material in my Bachelor Thesis and the usability testing conducted in 2001 showed that it had a number of deficiencies. The test users were in many cases not able to understand the designers' intentions. Misunderstandings and the inability to perform the tasks at all resulted in dislike of the program and frustration. However, a remarkable finding was that just supplying the GUI with a standard appearance enhanced the usability considerably, since it provided a sense of familiarity and consistency [26].

## **7.2. REQUIREMENTS ANALYSIS**

First of all, a requirements analysis was carried out and gave a thorough understanding of the product-specific context. The functionality provided by the local administration system is managing categories, questions and settings to a quiz game. The system is a Windows-application, developed in Borland Delphi, and locally installed in the customer's network.

In parallel to the requirements analysis, the user profile was set. After meetings and "brainstorming sessions" with future end-users, it could be concluded that a typical end-user of the system would be administrative staff or a guide in the company that owns the quiz game attraction. The user profile also revealed a person with a high level of computer experience, a so-called expert user, who is confident in using computers. The future users were active in the process of formulating the functionality of the system, so it is possible to state that the requirements were validated against users' needs. However some of their suggestions were not feasible within this project due to various constraints. The usability goal was set to "learnability", since it would be more important to be able to perform a task at all rather than doing it quickly.

## **7.3. USABILITY TESTING BEFORE REDESIGN**

The first round of usability testing aimed to evaluate the current GUI before starting with the redesign. The purpose was to find out where users encountered problems. Three of the future end-users (that fitted in the user profile) were selected as test users. The testing was made individually on three different occasions on customer-site.

Before the tests, the users were encouraged to think aloud and express their feelings. They were also informed that whatever happened was not their fault, so that they would not blame themselves if anything went wrong. Each user was asked to perform three representative tasks, during which the evaluator was observing. After each task, the user was interviewed to collect qualitative data. Figure 13 illustrates a sample screen of the original GUI. What would you do if you were asked to create a new category?

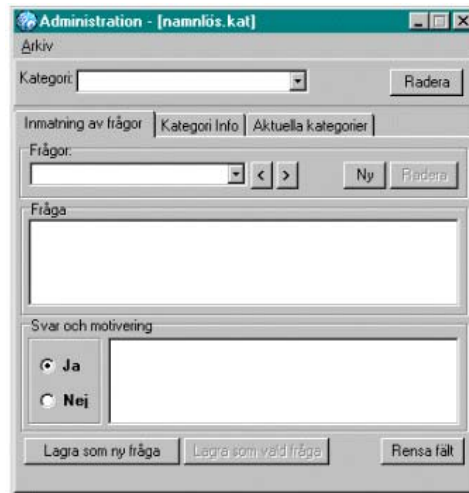


Figure 13. Sample screen of the original GUI

Here is a summary of observations made during the three testing sessions:

- In many cases, the tasks were not carried out as intended, but in a more time-consuming way.
- The test users were confused at several occasions and hesitated.
- All test users required help, otherwise they may not had been able to perform the task at all.
- The test users were looking for functions, but did not find them.

Typical user comments were:

- *I just pressed “New category” but nothing happened.*
- *Help, I am lost! I do not know what to do.*
- *I want feedback so that I know that I just did the right thing.*
- *I would need to go for a cup of tea before making a new effort in understanding how to do it.*

As the usability testing also was of a quantitative nature, the different usability factors was recorded accordingly:

- Learnability was measured by asking the users to rate how easy or difficult it was to figure out how to perform a task. The scale ranged from 1(difficult) to 5 (easy).
- Speed of performance was the amount of time required to perform a task.
- Incidence of errors was recorded through observations while the user tried to perform a task.
- Subjective satisfaction was the mark that the users put on the program after having performed all three tasks. The scale ranged from 1(not satisfied) to 5 (very satisfied).

Figure 14 presents the average results from the testing. *(The figures were derived by first calculating the average marks and numbers of the different tasks and the overall average was the mean value of the three tasks.)*

	<b>Learnability</b>	<b>Speed of Performance</b>	<b>Incidence of Errors</b>	<b>Subjective Satisfaction</b>
Task 1	4,3	5 min 8 sec	4,6	-
Task 2	2,7	4 min 56 sec	8,6	-
Task 3	4,8	2 min 20 sec	3	-
average	3,9	4 min 8 sec	5,4	3,3

Figure 14: Test results from usability testing before redesign.

One should interpret these results carefully and critically, as testing always is an artificial situation. Users may feel stressed and under pressure. The figures are not statistical evidence, but rather an indication of the current administration system's level of usability. Users themselves were not satisfied with their performances and claimed that their bad results were due to a lack of understanding of the task and that they should have understood the program better. They blamed themselves. One could also argue that the instructions that were given to the users before the testing were vague and not clear enough.

Surprisingly enough they gave good marks on the learnability in spite of their inability to perform the tasks correctly and with assistance on several occasions. They might not have been aware of the fact that they did not do the work properly (or at least not in the intended way).

The training that the previous tasks offered did not seem to contribute that much to the users' understanding of the program, since they encountered big problems in all three tasks.

One user commented that the program did not seem finished, as if it were under construction. This is an interesting comment, since this program indeed was finished and accepted by the customer.

## **7.4. REDESIGN**

The result of the usability testing clearly showed that users had difficulties in building the appropriate mental model in order to use the program properly. Obviously the designers had created a system image that did not correspond to the users' expectations. Apparently the GUI forced the users to work in a particular, strict order without indicating *how*.

### **7.4.1. Conceptual Model Design**

Hence, a vital part in redesigning the GUI was to make a clear information architecture, i.e. organizing objects, tasks and their relationships to avoid misunderstandings. For instance, the user should always be aware of what was being saved when using a "save"-function and what various "New"-options meant.

The major dilemma of the redesign was to establish a coherent visual layout that would allow the users to create the right mental model, which could minimise the risk of

misunderstandings and mistakes. Another highly important aspect was to let the GUI speak the users' language, i.e. adapt the terminology in the GUI to the users' natural set of words.

GUI Heuristics and principles were valuable in making new GUI design decisions and they provided the necessary guidance to produce any design solutions at all when the circumstances seemed too complicated. In addition, users' comments corresponded in several cases to the GUI principles, which may indicate that heuristics do derive from empirical studies.

#### 7.4.2. Detailed Screen Design

The redesigned GUI was added with familiar features in the Windows-environment like menus, toolbar and status bar. In general, the complexity was reduced due to a clearer division of objects and actions, e.g. logical grouping of components.

The application was developed in Borland Delphi, which enabled a quick and easy implementation of the new GUI design.

### 7.5. USABILITY TESTING AFTER REDESIGN

In order to assess the redesign, a second usability testing was conducted with the same three test users as during the testing *before* redesign. The goal was to decide if the redesign did make any difference to these three people when using the program.

The procedure was repeated with on-site testing sessions. This time too, there were three typical tasks to carry out. Figure 15 illustrates a sample screen of the redesigned GUI. Now, how would you create a new category?

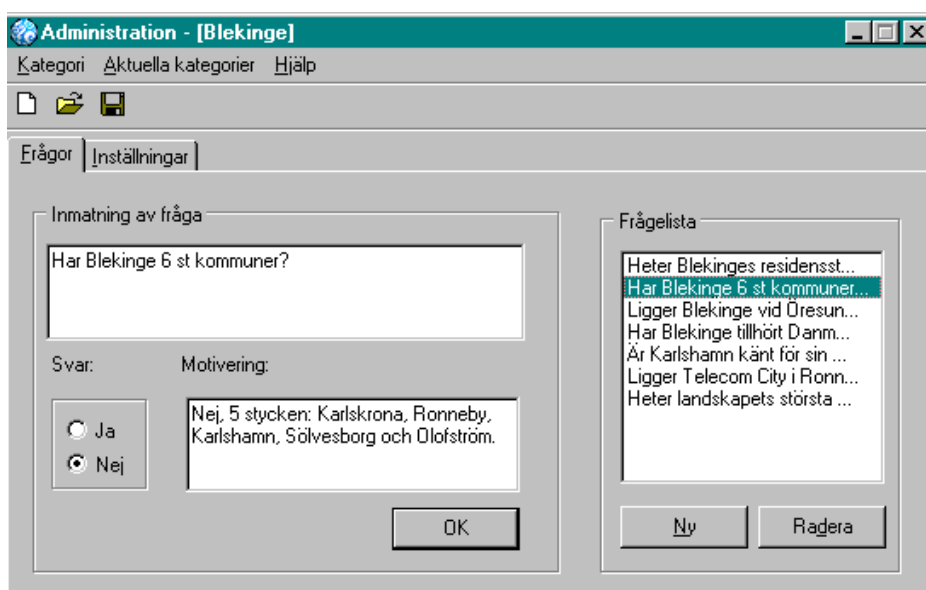


Figure 15: Sample screen of redesigned GUI

Here is a summary of observations made during the three testing sessions:

- One of the tasks involved catastrophic errors for all three test users.
- Users seemed to create a mental model, because they could predict consequences of their actions, but in some cases, their expectations did not correspond to the behaviour of the program.
- Users seemed confident and used short commands and keyboard navigation frequently.

Typical user comments were:

- *Is it as simple as it looks? I try... Yes, it was just like I wanted it to be.*
- *I want an "Undo"-button, because I understand that I did something wrong*
- *I did not understand some of the features intuitively, but once I had tried that feature it was clear.*

The different usability factors was measured accordingly:

- Learnability was measured by asking the users to rate how easy or difficult it was to figure out how to perform a task. The scale ranged from 1(difficult) to 5 (easy).
- Speed of performance was the amount of time required to perform a task.
- Incidence of errors was recorded through observations while the user tried to perform a task.
- Subjective satisfaction was the mark that the users put on the program after having performed all three tasks. The scale ranged from 1(not satisfied) to 5 (very satisfied).

Figure 16 presents the average results from the testing. *(The figures were derived by first calculating the average grades and numbers of the different tasks and the overall average was the mean value of the three tasks.)*

	<b>Learnability</b>	<b>Speed of Performance</b>	<b>Incidence of Errors</b>	<b>Subjective Satisfaction</b>
Task 1	5	3 min 18 sec	2	-
Task 2	5	1 min 05 sec	0,7	-
Task 3	5	0 min 45 sec	0,7	-
average	5	1min 43 sec	1,1	5

*Figure 16: Test results from Usability Testing after redesign.*

The figures presented above do not tell the whole truth and should therefore not be considered as evidence for a highly usable system. Looking at the learnability level, users seemed to find it extremely easy to figure out how to use the program. They did however commit catastrophic errors during one of the tasks. None of the users came close to performing that task correctly. They did however realize that something was wrong and tried to recover from their mistakes. One user in particular blamed himself and considered it carelessness from his side that he did not perform well. Possible explanations of why the other two users made mistakes are that they were tired at that time of the day (the tests were conducted late in the afternoon) and they rushed into the task without thinking too much.

Nevertheless, users do seem to create a mental model of how to approach the task. They typically spend some time inspecting the overall layout before starting to work. They also seem to go through the cycle of interaction as described in section 6.1.2 since they evaluate the system's response and form expectations for the next step.

Further, the figures of "speed of performance" and "incidence of errors" cannot be taken seriously, since the speed of performance was affected by the fact that some users were not

able to continue due to catastrophic mistakes. The error rate is fairly low indeed, but the nature of the errors was typically serious.

When comparing the two designs, users' preference was the second one, i.e. after redesign. They felt familiar with the GUI as it was consistent with the Microsoft Windows environment and resembled other programs. The new version had more space between components and a logical grouping which made it easier to grasp and allowed users to start working without first having to figure out the relationships of components. In summary, users' subjective satisfaction was higher and they enjoyed using the program.

The effects of a bad or good design should not be underestimated, in particular not the social one. When a user, who normally considers himself an expert user, gets "intimidated" by a computer program, he easily loses the joy of using that program and will in the worst case abandon the program, i.e. refuse to use it. The mood of one of the users after the first testing round expressed shame and resignation; *"I should have performed better"*. After the second testing, which turned out much better, the most satisfactory conclusion from his point of view seemed to be that he had performed better and thereby had strengthened his self confidence in terms of computer use. *"I am more satisfied with my own performance"*, was his first comment.

As the users themselves commented, they were influenced by their previous knowledge of the program from the first usability testing. Even though the GUI had changed, the concepts and the functionality were the same. This refers to the "carry over"-effect, which implies that the users were not novice users anymore in the sense that they already had experience of the program which affected their performance. Hence, it cannot be proven to what extent the redesign affected the system's usability. It can however be concluded that the system was perceived as more usable after redesign to the three testing subjects.

## **7.6. REVISION OF DESIGN**

The second usability testing revealed usability deficiencies and hence the design had to be revised further to span the gap between users' mental model and the design model. The major problem seemed to be the "Activated Categories"-panel. Once it was visible, it was not possible to return to the "Question"-panel without creating or opening a new category. During the testing, one user was busy entering new questions into a new category when accidentally or deliberately choosing the "wrong panel" and all the work with the questions was lost. In addition, users were confused over how to save the settings for the activated categories.

By splitting up unrelated tasks on different windows, e.g. a separate window for setting the activated categories (see figure 17), the complexity was reduced and the GUI did not only follow the principle of "high cohesion and low coupling" but was also more forgiving if users committed mistakes. This solution refers to the design pattern *Window per Task*.

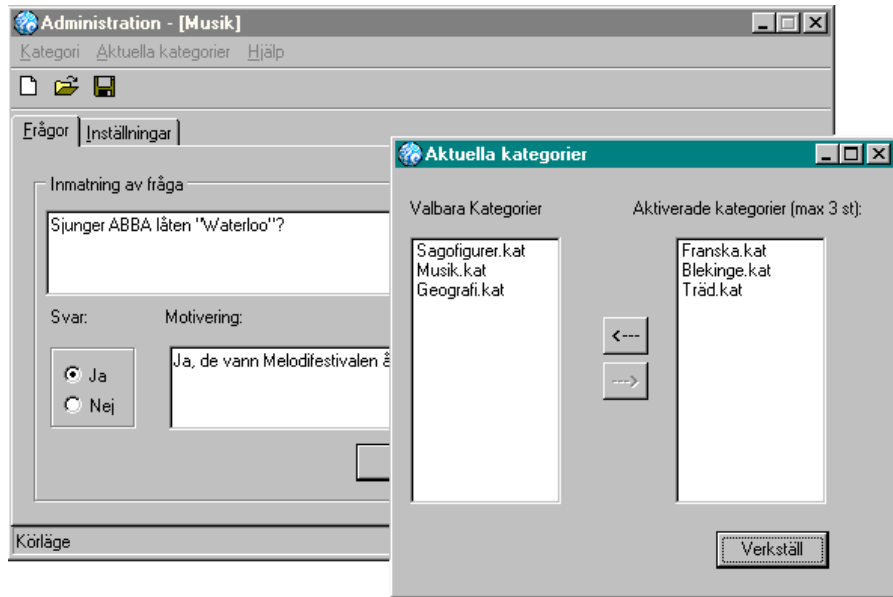


Figure 17: Window per Task

Concerning the problems that users encountered when creating new questions, in particular users' negligence of the "OK"-button to save a new question, the visual cues and logical grouping should be enough to guide the users. Certain components were added with a so-called hint, i.e. when the mouse cursor passed over the component a hint or tooltip was displayed (figure 18). Another property of the GUI is visual affordance, which is a term for describing visual clues of instinct operations (of technology or of anything) [12].

In this case, a button that should not or cannot be pressed at a given moment is disabled and the button that should be pressed is three-dimensional, raised and default, inviting the user to click it (figure 18).



Figure 18: visual clues and hints

The revised design seemed to be a satisfactory solution in terms of visual affordance, logical grouping, consistency with other program and feedback on users' operations. It is difficult to explain why the users committed such catastrophic errors during the performance of task no 1, but as much as they performed the other two tasks very well, it could be a coincidence that they started badly. The circumstances under which they operated may play a significant role here; users were under stress due to a heavy workload in the whole company and the testing sessions occurred late in the afternoon. Testing is also an artificial situation and the user might feel under pressure on being observed.

It is therefore possible to conclude that the design seemed to be on a satisfactory level of usability.

\* \* \*

The last step before finishing the project with the administration program was to revise the user manual and the technical documentation. The user manual is a part of the user interface and may be an important source of support during training or whenever questions occurred. In addition, a help text was added to the program for quicker guidance. The three test users claimed that they rarely used the user manual. In spite of this statement, it is always a good idea to provide the system with a user manual. In this case the manual mainly consisted of screenshots and step-by-step instructions.

## **7.7. CONCLUSION**

This project aimed to redesign a GUI since previous usability testing showed that test users did not perceive it as usable. The GUI design was made iteratively, where user testing provided the design work with valuable feedback. The usability testing performed were helpful from a qualitative point of view but cannot be considered as totally valid since there were only three test users and they were not novice users of the system on the second testing occasion. The testing did however show an improvement of usability after redesign. It was also obvious that a learning process indeed has three aspects, the social, cognitive and psychodynamic. Users' performance is affected by multiple factors, both environmental and psychological. Theories on communication were practised in both a developer-user and contractor-customer relationship.

## 8. PROJECT TWO: WEB ADMINISTRATION SYSTEM

The purpose of this project was to develop a new web-based administration system of a quiz game as a complement to the local administration system described in chapter 7. This product was requested since it would save time and effort for the administrators of the local system and allow external administrators to access the system via a web interface. This refers to the aspect of *usefulness*, which is necessary for usability to grow.

Figure 19 illustrates how the usability toolbox was utilized in this project. Future end-users were involved in the development and the insight in their “information ecologies” were of significant importance in order to understand how the end-users typically use information technology and which role it plays in their work. “The Invisible Computer” urged for simplicity in design. The “interaction”-part of the usability toolbox was helpful in the close contact with the customer’s management and future end-users, especially since developers and users do not always speak the same language. Developing a brand new product required a good deal of creativity and the usability toolbox suggested what tasks to carry out with the aid of guidelines and heuristics.

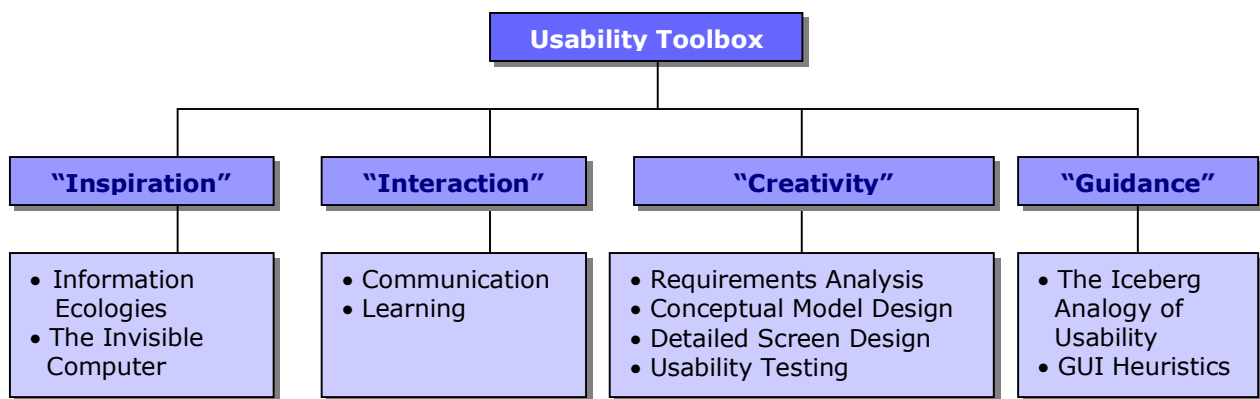


Figure 19. Parts of the usability toolbox utilised in this project

This project spanned over three phases of the waterfall model, from scratch to a finished product. Figure 20 shows the development phases (corresponds to the “creativity”-part of the usability toolbox) in relation to the waterfall model.

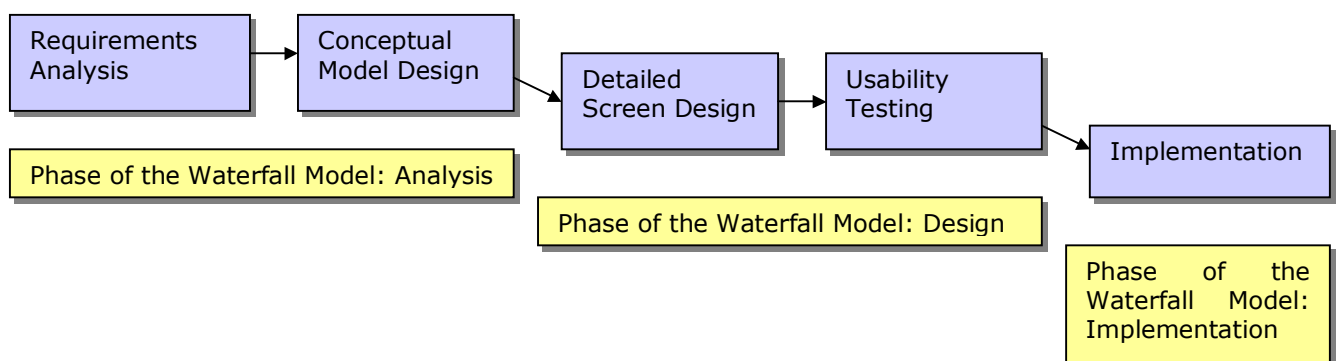


Figure 20: Development phases in relation to the waterfall model

## **8.1. REQUIREMENTS ANALYSIS**

Since this project was made totally from scratch, the requirements analysis served many purposes: requirements gathering, setting the user profile and the usability goals and choice of technical solution.

Agreeing on a solution that satisfied all stakeholders was a non-trivial task. Respect had to be taken to the end-users, the management, technical constraints and time. The system also needed to be integrated to the existing local administration system, a control system and an official web page. Consequently a great deal of the users' suggestions were not possible to implement due to various constraints, but were documented for future development nonetheless.

The functionality was restricted compared to the local administration system due to security but would enable the creating of new categories and questions, that could be sent automatically in an email as an attachment to be integrated with the customer's local network. The web application itself would be made available from a website and protected by a password. HTML and PHP were going to be used to implement the system.

The usability goal was set to a high learnability since most future users were so-called novice users with little or moderate experience of computers. The target population was teachers, who would use the quiz game to let their pupils train a specific subject. Three potential end-users, teachers in primary school, were interviewed on-site to get a deeper insight of their skills and habits. The user profile and the usability goals were crucial for making future design decisions.

## **8.2. CONCEPTUAL MODEL DESIGN**

The conceptual model design was presented to future end-users as a mock-up during a brainstorming session with teachers representing all schools in Karlshamn municipality. The mock-up was drawn by hand on a whiteboard and provided a step-by-step guidance of how to use the web application. All people present were discussing and suggesting new features. An important lesson learned was to speak the users' language and skip the technical terminology. Words like "browser" and "attachment" were unfamiliar to some of the users and they apologized for their lack of knowledge of computers. Apparently it was imperative to make the users feel confident in using the system and more important than ever to keep things simple. Any insecurity and resistance had to be eliminated by tailoring the system to users' needs. The users seemed however very enthusiastic in general.

\* \* \*

Since the system's functionality was fairly simple, the conceptual model design did not imply any problems. The discussion of the mock-up during the brainstorming session can actually be regarded as an informal, qualitative usability testing and the users did not seem confused. The proposed design was a wizard-like approach, where the GUI guides the user from page to page. GUI Heuristics and the iceberg analogy of usability were useful in this phase.

There was a certain overlapping of phases between the requirements analysis and the conceptual model design to leave space for the users to have an impact on the requirements

specification before signing the contract. The users had a great deal of good proposals, but unfortunately most of which were not realistic to implement within the scope of this project. Users' suggestions were however documented for future development.

### 8.3. DETAILED SCREEN DESIGN

A functional prototype was made in order to evaluate the second level of GUI design, detailed screen design. The prototype was designed on the basis of the mock-up, the conceptual model design. The software tool Borland Delphi was employed as it offers an easy and quick creation of GUIs. The purpose was to make the prototype look like a web page as much as possible, e.g. with blue underlined links and hand point cursors, as the system was going to be implemented for the web later on. GUI Heuristics were useful in this phase. The whole construction required approximately four hours, but during this time, ideas for the real implementation came up, e.g. logical sequences.

### 8.4. USABILITY TESTING

In order to assess the prototype of the web application, a usability testing was conducted with three future users, all primary school teachers. The testing sessions took place in the three schools in a computer room. Before each testing, the users were asked to verbalize their thoughts and not blame themselves if something went wrong. The users were given clear instructions of what to do. The users were asked to perform one task (since the system did not provide more functionality), during which they were being observed. Afterwards, they were interviewed for a more qualitative study. Figure 21 shows a sample screen of the prototype. How would you create a new question?

Skriv in frågor till kategorin Musik:

Fråga:

Svarsalternativ:

ja

nej

Motivering:

OK

Inlagda frågor:

Sjunger Marie Fredriksson i gruppen Roxette?

Visa Radera

När du har skrivit in alla frågor:

Gå vidare

Figure 21. Sample screen of the prototype

Here is a summary of observations during the three testing sessions:

- All users preferred using the mouse as navigation tool, not the keyboard. They never used short commands
- All users performed the task without any major problems
- The prototype was not well suited for the screen resolution, so unnecessary scrolling was required for all three users

Typical user comments were:

- *It looks exactly like the mock-up*
- *It is so easy that the children can use it*
- *There are not many alternatives, so I could not do anything wrong*

The different usability factors was measured accordingly:

- Learnability was measured by asking the users to rate how easy or difficult it was to figure out how to perform a task. The scale ranged from 1(difficult) to 5 (easy).
- Speed of performance was the amount of time required to perform a task.
- Incidence of errors was recorded through observations while the user tried to perform a task.
- Subjective satisfaction was the mark that the users put on the program after having performed the task. The scale ranged from 1(not satisfied) to 5 (very satisfied).

Figure 22 presents the average results from the testing. (*Only one task was carried out*).

	<b>Learnability</b>	<b>Speed of Performance</b>	<b>Incidence of Errors</b>	<b>Subjective Satisfaction</b>
User 1	5	5 min 45 sec	1	5
User 2	5	5 min 35 sec	1	5
User3	5	5 min 15 sec	0	5
Average	5	5 min 31 sec	0,7	5

Figure 22. Test results from Usability Testing of prototype.

The testing showed that the usability goals seem within reach. All three users together only committed two errors, of which one was catastrophic, since the question was updated, but the other was more a moment of hesitation rather than an error (clicked the “delete”-button on the keyboard instead of pushing the “delete”-button on the GUI). The users commented that this program is so easy that even the children can use it. The level of learnability is crucial in this program because everybody, even those who have very low experience of computers should be able to use it. The test users claimed that some of their colleagues use computers very rarely and with a combination of fear and resistance. According to the test users, this program is so simple that you cannot do anything wrong.

The speed of performance need not be commented further as it is not crucial in this program. User 1 took a long time to read the instructions, which the other two did not. Still it is fascinating to conclude that they worked almost in the same speed.

They made very few errors and did not require assistance at all. Consequently they were very satisfied with the program and gave it the highest score.

Only looking at the excellent figures, the result can be interpreted as though the GUI design was successful and that the web application's GUI can be designed accordingly.

However, the result was affected by a number of factors, which gives a different nuance to the picture. Since the users participated in the brainstorming sessions and were involved in the requirements analysis phase, they already had knowledge of the system. The users themselves commented that they still remembered how to pursue the different steps from the mock up presentation. They were thus trained during the development and this aspect has to be taken into consideration when analysing the result.

Furthermore, the result from only three test users is not satisfying as statistical evidence. It may be a coincidence that these three users did not commit that many mistakes. That these people performed well is no guarantee that everybody will. To be more convincing, at least 7 to 8 test users would have been required. However, if the purpose is to find usability deficiencies, then one single user is able to detect some of them. It is estimated that three to five test users find up to 75% of all usability problems [9]. To summarise, the usability testing gave no valid evidence in terms of being highly usable but rather an indication of the fact that the three test users in question did not encounter any major problems when using the program.

## **8.5. IMPLEMENTATION**

One main principle of this thesis is to slide on the man-machine scale. One important lesson learned from this investigation is that sooner or later the designer reaches a point where technical issues get more attention, i.e. when implementing the system. The system in question was implemented in PHP and HTML for web use. While doing so, many technical difficulties arose and occasionally the situation felt somewhat desperate. In such moments, it was important to stay on track and remain the users' advocate, i.e. do what is best for the users and not for the programmer. Implementation is highly dependent on the platform and the "look and feel" provided by the UI Toolkit. In this case, the operating system or platform constituted no constraint of the GUI design of the web application.

Already when preparing the mock-up at an early stage of the project, it was tempting to think in terms of "*is this difficult to program?*". Such thoughts had to be chased away in order to conduct an adequate requirements analysis. On the contrary, when users proposed several new features, technical aspects had to be considered to be able to decide whether that functionality could be added with respect to limited resources.

In this project, the design phase mostly concerned *designing the user interface* rather than planning the implementation of the system. Therefore, after having tested the prototype, the implementation followed immediately. The time spent on developing the prototype was approximately four hours, while the coding of the "real" application required about 50 hours. Hence it turned out to be a good idea to produce a quick prototype in order to conduct usability testing before "wasting" time on coding a program that users did not seem to perceive as usable.

The GUI was implemented exactly as the prototype's GUI with a couple of added features according to users' own proposals during the usability testing of the prototype. The grouping of elements was also improved.

## **8.6. CONCLUSION**

This project aimed to create a new web application and was thus conducted from scratch, throughout three phases of the system's life cycle. Mock-ups and prototypes in combination with brainstorming sessions with stakeholders and user testing supported the GUI design work. Usability testing with potential future users showed that the GUI scored high in all categories, but there is a validity problem with the testing as only three users were consulted. The test users were also affected by the carry-over effect as they took part in the brainstorming sessions prior to the usability testing. This example illustrates that it is awkward to measure usability, if it is possible at all.

## 9. PROJECT THREE: CUSTOMER SUPPORT SYSTEM

The goal of the project was to evaluate the GUI of a web application for customer support and suggest how to enhance the system's usability. No potential end-users were consulted since they were located far away. Additionally the product was generic and had no specific user group. Consequently no information ecology could be studied so the "invisible computer" was the inspiration source for getting a touch of humane technology. The "interaction"-part of the usability toolbox was useful from a learning point of view. The concrete work in this project consisted of evaluating the existing GUI and propose a new layout to improve usability. This was accomplished by carrying out the appropriate tasks. One valuable outcome of this project was that the GUI Heuristics were revised and crystallized as my personal set of guidelines based on my own and other's experience. Figure 23 shows an overview of how the usability toolbox was utilized in this project.

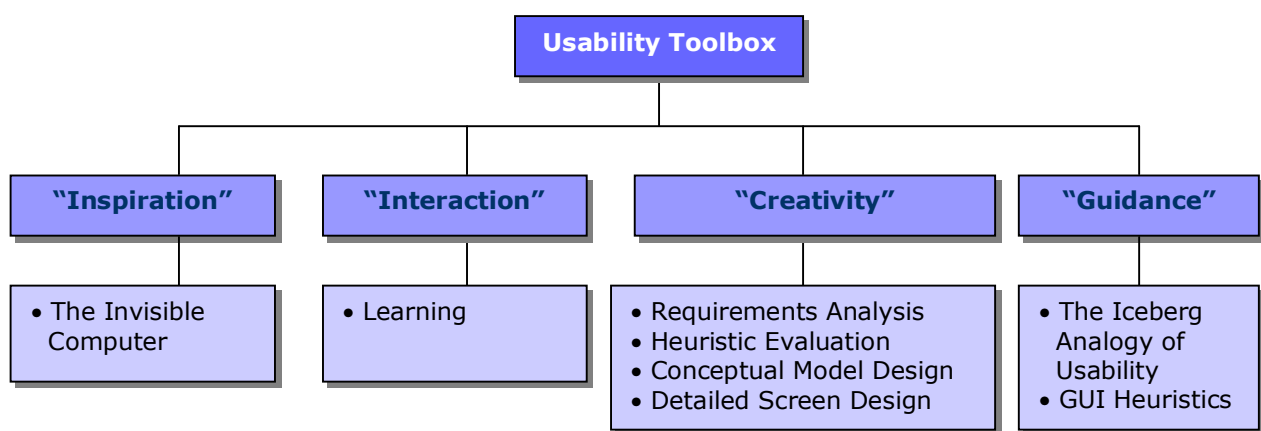


Figure 23. Parts of the usability toolbox utilised in this project

This project was in the maintenance phase of the waterfall model and consisted of four main steps as illustrated in figure 24.

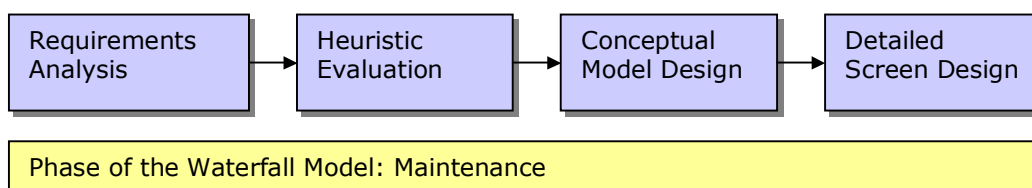


Figure 24. Design phases in relation to the waterfall model

### 9.1. REQUIREMENTS ANALYSIS

The first thing to do was to perform a requirements analysis, i.e. understanding the environment in which the system was running, the functionality of the system, and setting the usability goals.

The functionality provided is extensive and can be summarized as a customer support system. It enables companies and organizations to provide their clients with feedback and support. For instance, a client can ask a question and track the response from the company until a

resolution has been agreed upon. This allows the user to ascertain that his issue is being dealt with and to check its current status. All resolutions are stored in a knowledge base. The system can be used in any customer relationship. Consequently there is a great deal of various user groups on “both sides”. The service providers might for instance be a group of software developers or bank staff, whereas the customer side might consist of computer users or bank clients.

The user profile was not clear, since there is no specific target group. Due to the fact that the end-users were to be found on remote sites, it was early stated that user-centered techniques such as usability testing and brainstorming sessions with users could not be carried out. This was going to be solitary work for the GUI designer and a challenge to push the usability toolbox to its limits.

The usability priority was learnability. It was desirable to create a GUI that would suit both novice and expert users. Another keyword was customisability, which allows a user to chose which data to display and pretty much set up an own user profile. Furthermore, the GUI should be flexible enough to easily allow expansion and changes.

## **9.2. HEURISTIC EVALUATION**

The heuristic evaluation aimed to validate the current GUI against known usability principles. The evaluation resulted in a list of good and bad aspects. All comments had references to these principles for achieving usability in GUI design.

An accurate heuristic evaluation can only be carried out when the evaluator has a thorough insight of the system's context and requirements. It was therefore imperative to carry out a requirements analysis before starting to assess the GUI.

The most alarming finding was that the GUI seemed to provide a poor support for creating a mental model that would help using the system. This was mainly due to inconsistent and bad navigation facilities and an unstable visual appearance (grouping, alignment etc). Other flaws were for instance inappropriate use of colours and that unnecessary complexity was exposed to the users. In fact, the overall impression after having used the system was that the GUI did not provide enough visual cues and navigation mechanisms to create the appropriate mental model. Questions like “Where am I?”, “What can I do?” and “What did the system just do?” remained unanswered on too many occasions.

The GUI did however have many good characteristics as well. Information was presented as text and not as graphics (decreases download time). Error messages were mostly precise, polite and suggested how to recover. And most of all, the system was in possession of a brilliant functionality that had all the potential to support users in any kinds of situations, i.e. to be *useful*.

## **9.3. CONCEPTUAL MODEL DESIGN**

During this design phase, a conceptual model design was being worked out. The aim was to establish a high-level design to ensure a consistent navigation scheme and a basic page layout.

### *9.3.1. Information Architecture*

Before any paper mock-ups could be created, an information architecture had to be established to find a natural and logical organisation of content.

“Brainstorming” contributed to gather data about the system’s user groups, their objects and what actions could be acted upon these objects, e.g. a user can ask a question, to which notes can be added. The initial goal was just to clarify all possible concepts within the user domain without caring for internal relationships.

Once the various users, objects and actions had been stated, it was possible to establish an information architecture and group the concepts logically and in a hierarchical manner. The outcome was an information architecture which was no deeper than three levels.

The real challenge was to sort out how to establish an easy and logical presentation of related data and how to retrieve it. The concepts, interrelationships and functionality that the system provide are already complex. Without going into details, there are several kinds of staff, various ways of dealing with questions, multiple question statuses. Obviously one single approach would not be good enough, but many workflows. It is worth mentioning that the current GUI design was not made on the basis of any kind of known usability principles, but rather on the developers’ own experience and common sense.

If a question consists of several sub problems, it may be split up into many issues. The question was how much to show for the user. The only thing a user would care for is that he/she asked a question and would like a resolution for this problem. He or she does not care if other users are sharing the question or if the staff is divided into categories. There was obviously a need for hiding implementation issues for the user. This refers to the GUI Heuristic of “Simplicity – hide what the user does not need to know”. The heuristic of “Speaking the user’s language” was useful in terms of getting rid of technical terms.

### *9.3.2. Mock-ups*

The new page layout featured a main vertical navigation bar to the left and a local navigation bar on the top. This navigation scheme refers to the “inverse L”. By doing this, the problem of mixing links (where to go) with actions (what to do) was eliminated. The original navigation options consisted of both links (e.g. “home”) and actions (“expand all”) which made the actions invisible. This violated against the heuristic of “General Appearance/Grouping” as well. If related items are logically grouped, they will be more visible.

Furthermore, there was a need to split up the content into separate parts and not display everything on one screen. Conforming to the heuristic of “Window per task” reduces the risk of confusion, complexity and mistakes. A tab-system was added to distinguish various tasks and information, yet related to each other.

By doing so, the main menu to the left was also kept stable and available at all times, which refers to the heuristic of “Navigation”. Stable menus are forgiving if the user picks the wrong link and also provides a sense of stability since the menu options always are there.

The tab system is good from another point of view since it provides a page with a natural headline.

In summary, the conceptual model design consisted mainly of a revision of navigation structure and page layout according to the information architecture.

## 9.4. DETAILED SCREEN DESIGN

Once the basic page layout was established in the conceptual model design, it was time to go into details and create a coherent low-level design across the application. Typical issues were for instance use of colours, fonts and link appearance. If the conceptual model design aimed to clarify *what* should be on each screen, the detailed screen design rather aimed to say *how* this content should be displayed. Aesthetic issues do matter, since beauty impacts the subjective satisfaction [19]. An ugly product communicates the wrong values [27].

The heuristic of general appearance and visual affordance played a significant role since they mostly concern the “tip of the iceberg”, the look-aspect. The hypertext links were given a uniform appearance. Furthermore, links and buttons were distinguished since they typically represent navigational options (“view details”) and actions (“submit”). Moreover, the detailed screen design ensured consistency in terms of how feedback and error messages would be displayed.

The real challenge lay in establishing a layout whose elements suited most of the use cases, since some of them were more complex than others. A big problem was also all the interrelationships between information. It was vital to clarify the current location at all times, since some pieces of information linked to others. A way back was also provided. The navigation structure was further refined. The detailed screen design consisted of several pieces of paper, each one representing a screen which could be complemented and replaced as the interaction proceeded.

The cycle of interaction was a good support for the interaction design and helped to understand what steps a user would usually go through and what he or she would expect of the actions performed.

The heuristic of simplicity refers to hiding what the user does not need to see. For instance, do not display “previous” and “next” links if there are no more options or search results. By doing so, the questions of the cycle of interaction (*What did the system just do? What can I do next? What will happen if I do this?*) will be easier to answer. Simplicity also relates to hiding technical details, e.g. is it necessary to show FAQ id? One might argue that it would be more suitable to show question ID.

Ideally one should know what elements signify and lead to just by looking at them, that is to say, GUI elements should have visual affordance. This property was suggested to be implemented by enabling and disabling textareas and other input components. Another way of increasing visual affordance is to make sure that links and buttons invite to clicking. Visual affordance not only make things clearer but also reduce the user's mental workload.

If visual affordance is poor, users will look around and possibly make mistakes. It is therefore imperative that the GUI is forgiving by always providing a way back or to recover. This was implemented by having a stable main menu to the left and a local navigation bar in a tab

system on the top. Error messages were already clear and precise but were taken into consideration as well.

The general appearance is essential in terms of getting a balanced look. Alignment and grouping were improved, which also helped outline the relationships between elements in a clearer way. In practice, most of the heuristics relate to each other and can only be separated in theory. For instance, design solutions pop up during the conceptual model phase, but it would be a mistake to start making changes on the tip of the iceberg before caring for the mental model.

According to the iceberg analogy of usability, changes on the surface can only improve usability to a certain degree.

## **9.5. CONCLUSION**

This project aimed to evaluate the usability of the current GUI and propose suggestions of how to enhance the usability if found necessary. A valuable outcome of this evaluation was a set of heuristics that was vital to define in order to make the assessment. These GUI heuristics were inspired by literature but also by experience. The heuristics were selected with respect to the Iceberg analogy so that each part of the “iceberg” was represented by the heuristics. The idea was to keep the number of heuristics as few as possible but yet catch all the aspects to consider. There are hundreds of guidelines out there, but heuristics are more practical and easy to apply since they are high-level guidelines which can be more detailed if required.

It was earlier stated that this project was going to be conducted in lack of end-users. This was due to the fact that users were located far away but another factor was that the system is a general product to be sold to customers and not customised for a specific client. Consequently there were few real users out there since the product was not fully developed.

Due to the absence of users, it was more imperative than ever to be the user's advocate and make an attempt to provide a good support for creating the right mental model. User-centered techniques such as usability testing and brainstorming sessions with mock-ups were not possible to use. It was even quite impossible to set the user profile. When designing for the internet, there is not one homogeneous user category, but people with varying computer experience and skills. The primary usability goal was set to learnability in order for everyone to be able to learn how to use the system.

Since no users were involved and no user testing was performed, it cannot be proven that usability actually was improved. However in theory, it should have, since the GUI heuristics that were applied come from experience and research. Nonetheless, it cannot be concluded whether the level of usability was enhanced or not. One might argue that usability cannot be measured, since it is a multi-dimensional property in a large chain of aspects. Moreover, in order to be statistically valid, the testing demands resources and needs to be performed at least two occasions to be able to make a comparison (before and after redesign). This was an example of how a project that aims to evaluate and redesign an existing GUI can be conducted.

## 10. PROJECT FOUR: MOBILE APPLICATION

This project involved developing a new mobile application for handheld devices. Mobility introduces a number of new issues in GUI design and usability engineering so the rather philosophical part of the usability toolbox inspired to foster mobile usability with respect to both technical wonders and human needs. Since the development process was integrated with market research and frequent user feedback, the “interaction”-part of the usability toolbox was highly relevant. The concrete work corresponds to the selected techniques of the “creativity”-part. Considering “guidance”, the GUI heuristics were slightly redefined to suit mobile usability better and the iceberg analogy of usability was introduced to the rest of the development team. Figure 25 shows an overview of how this project carried out in relation to the usability toolbox.

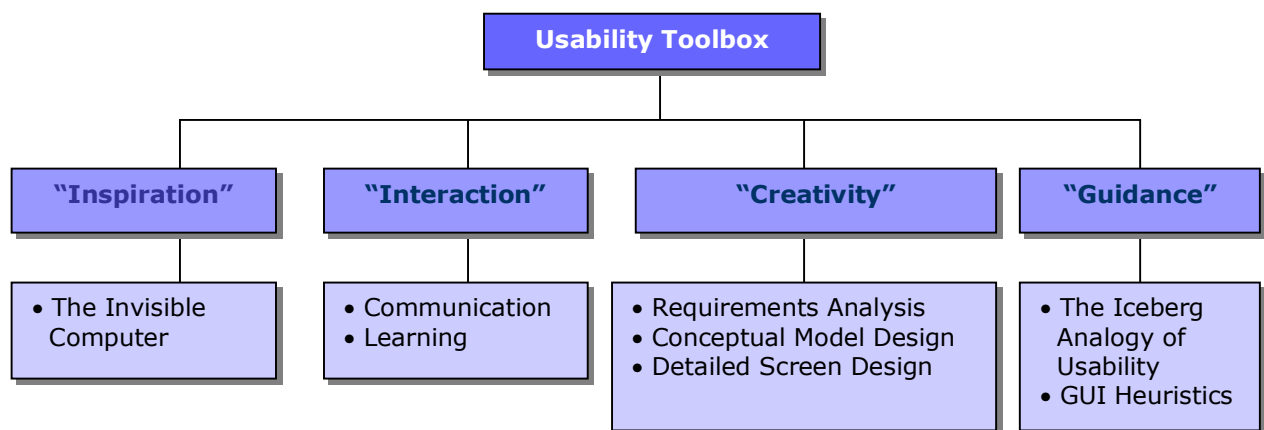


Figure 25. Parts of the usability toolbox utilised in this project

This project was made from scratch and spanned over the two first phases of the waterfall model as illustrated in figure 26:

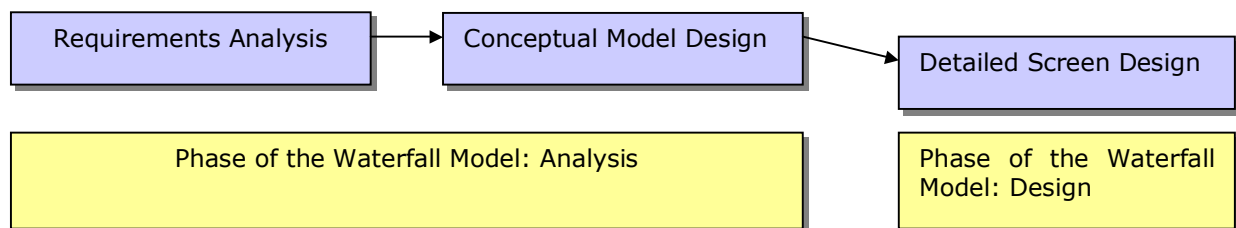


Figure 26. Design phases in relation to the waterfall model

### 10.1. MOBILE USABILITY

Mobile usability is an expression for usability issues on handheld devices such as Pocket PCs and is a new challenge since it implies a number of aspects that do not exist in traditional GUI design for desktop applications:

- Small screen
- Awkward input devices implies the need to minimise typing with virtual keyboard and to enable “poking” with the finger

- Little research in the world on mobile usability: few guidelines and standards
- Users' inexperience of (and possibly resistance to) using handheld computers

To be able to deal with these constraints, the GUI heuristics were revised and applied in the project that follows.

## 10.2. REQUIREMENTS ANALYSIS

The goal of this initial phase was to understand the system's functional requirements, set the usability goal, define the user group and get an insight in the technical solution being used. The requirements analysis was tightly integrated with market research in order to create a product that would be appealing to customers. The heart of the analysis was to find a mobile solution that actually would provide features that a desktop application could not. In order to maximise usability, make sure it is really useful. Furthermore, make sure the automatic processing is faster than the manual and intuitive to use. If it is awkward, people will not use it. Another aspect of mobility is that most people have little experience of use of Pocket PCs and it might be a big step to start to use a mobile device. The learning process required might be hard, a real *assimilation*.

The product being developed was a Pocket PC application for sales automation. It would replace paper forms and product catalogues and could save a great deal of manual work for sales representatives on the field. The functional requirements were gathered through frequent meetings with potential customers. The product was not tailored to one specific customer but rather generic to suit many companies. However, trade-offs were inevitable. Users sometimes wish too much, and some of which are not even possible to implement. In summary, the functionality provided included requisition and customer management.

The typical user would be a sales representative on the field, visiting customers who wish to place orders. Some users would be familiar with both laptops and mobile phones, whereas other users would be completely novice computer users.

The usability goal was set to *speed of performance*, sacrificing *learnability*, meaning that it may be a little bit painful in the beginning. The reason for this target was that there is a need for fast order processing.

## 10.3. CONCEPTUAL MODEL DESIGN

The conceptual model design aimed to create a high-level design and a basic page layout. Support for this kind of design were *task analysis*, *information architecture* and *screen map*.

### 10.3.1. Task Analysis

A thorough task analysis was conducted in order to understand the steps of manual processing, e.g. paper order form and product catalogue. Various workflows were analysed from different businesses in order to get a somewhat general picture of how sales processing works. When automating manual work, it is vital to understand how users perform their work manually to be able to create a system that can help these users to get their work done faster

and help them more than hinder them. The task analysis was carried out by interviewing several sales representatives.

### *10.3.2. Information Architecture*

The information architecture clarified the relationships of information and features provided and was the foundation of the menu structure. Due to restricted screen space, the number of menu options had to stay few. The information architecture outlined the main menu options and their sub-options. When establishing the information architecture, no considerations were taken to the actual screen design, since the goal was to logically organize the concepts in the user domain. The information architecture should be the natural and preferred way to present content and features, but the actual implementation does not always correspond to this ideal structure. One reason for overriding the information architecture might be the need for speed of performance, so that some steps in a sequence are made as one or totally eliminated.

### *10.3.3. Screen Map*

Once the information architecture was done, the basic screen layout was drawn on several paper mock-ups. These mock-ups were put together in a screen map that is a kind of a mirror of the information architecture but more practical and shows how the screens of the application relate to each other. This shows that it is not always possible to implement truly according to the information architecture and the screen map reflects this fact.

When struggling with the screen layout, other Pocket PC applications and competitors' similar products were reviewed to possibly find standard features and behaviours of Pocket PC GUIs. The challenge in designing a GUI for handheld computers is that there exist little research and that there is little experience to build on since a mobile application cannot be designed as a desktop application.

The conflicting usability properties of "speed of performance" and "learnability" raised a great deal of questions since it would be preferable to provide support for both novice and expert users. For instance, how would it be possible to embed accelerators? And how to provide several ways to the same function without being confusing? Well, the answer is to strive for the usability goal. Furthermore, difficult issues had to be sorted out such as how to present thousands of customers and products on a small monitor. Another aspect is to prepare for extensions of functionality in future releases, thus letting the GUI be flexible.

## **10.4. DETAILED SCREEN DESIGN**

The detailed screen design consisted of making a low-level design for a functional prototype. The goal was to accomplish a good design before starting the implementation. However, the screens could later on be used in the real product with logic added behind them. It was desirable to keep the number of screens low and reuse as much as possible. It is good from a usability point of view, since it may ensure consistency of forms and controls.

The prototype was developed in VB.net and it became obvious that platform constraints and UI toolkits do affect the design. Furthermore, it is somewhat vital to have a technical understanding when suggesting design solutions.

The main issues of concern were maximising the space for content on the screen and somehow minimise the space of menus and other navigation. Since there was no room for explaining labels, it was more crucial than ever to enhance visual affordance, grouping and other aspects of general appearance. Terminology of tabs, menu options and buttons was chosen carefully. Considering peoples' inexperience of using Pocket PC applications, one essential property of the GUI was to be forgiving if users made a mistake.

The big dilemma was to ensure a natural workflow within the application in spite of awkward input devices and the large amount of information to be displayed on a small screen. It was hard to realise that compromises and sacrifices had to be made and that some of the usability rules had to be broken. For instance, the heuristic of "Window per Task" was hard to follow since conforming to this rule would result in too many screens and possibly a slower processing.

Another question was whether a splash screen could add value. Additionally, it is hard to be consistent, even though it is desirable to always perform the same action in the same way to not confuse users. Eventually, it all fit together quite well as on the sample screen of the functional prototype in figure 27:

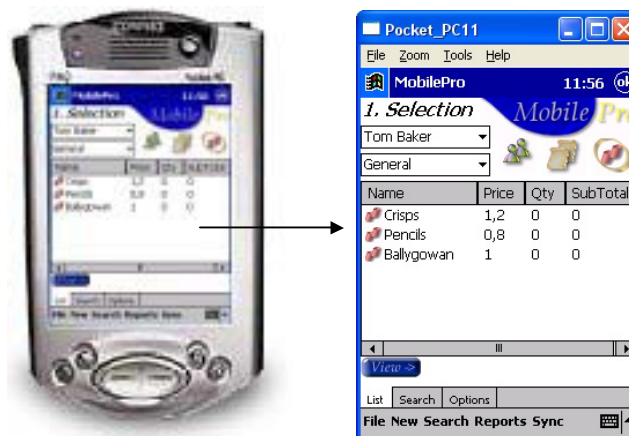


Figure 27. Sample screen of functional prototype.

## 10.5. CONCLUSION

The lessons learned from this project are firstly that mobility introduces new design challenges including small screen size where each keystroke and awkward input devices cost usability. Secondly, integration of marketing and requirements analysis for a generic product increases the chances for *usefulness* and early customer meetings with a functional prototype is a great support for requirements gathering and GUI design. Thirdly, technical insights and skills are useful when proposing GUI design solutions as the UI toolkit and platform constraints may affect the design. Last, it is very challenging to create a new design from scratch as opposed to redesigning an existing GUI. It may be somewhat frustrating to make everything fit together and look good on the screen. It can be quite painful to accept that sacrifices and trade-offs are necessary in pursuit of the usability goal.

## 11. EVALUATION OF THE USABILITY TOOLBOX

After having completed four distinct system development projects, the usability toolbox can be evaluated, i.e. which tools were employed, how and when? The first conclusion is that it is impossible to draw a generic model, since every project is unique. Nevertheless there seems to be a minimum approach, applicable in most contexts in any phase of the system's life cycle. Figure 28 illustrates the miniature usability toolbox and how other tools can provide extra support as resources allow so and depending on the context. The tasks that will apply at all times are requirements analysis, conceptual model design and detailed screen design, to which the iceberg analogy of usability and GUI heuristics can provide guidance.

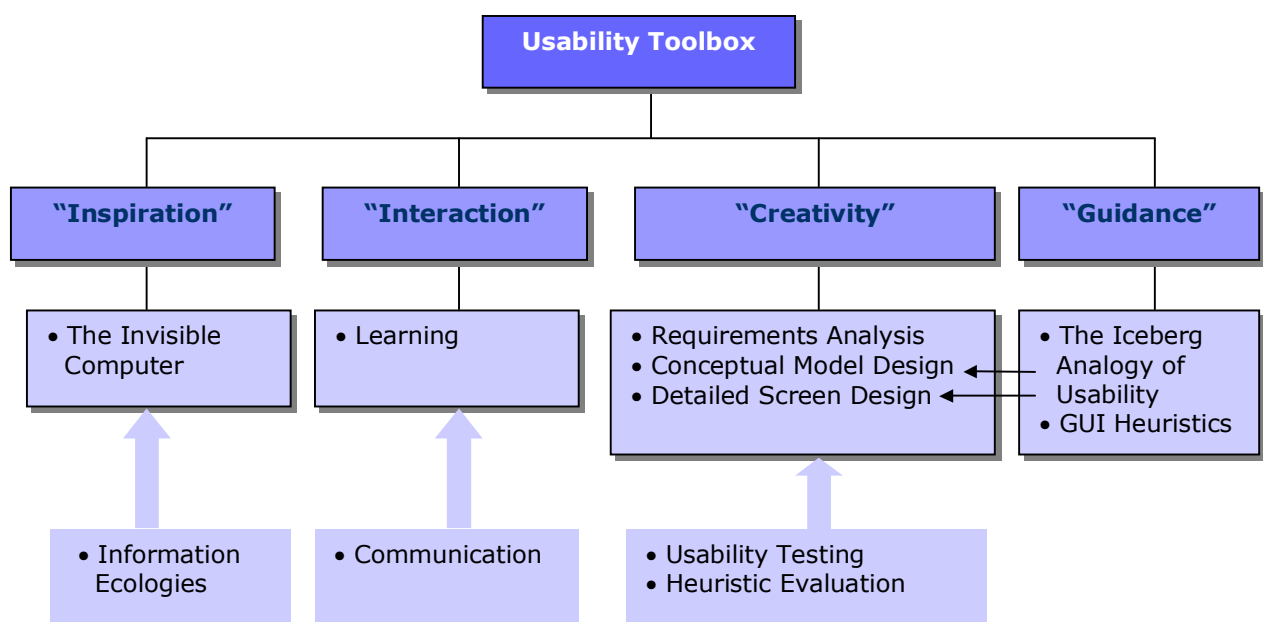


Figure 28. Miniature usability toolbox and additional parts.

### 11.1. "INSPIRATION"

Be inspired before starting to design, whenever you are stuck and remember the humane technology. This thesis is based on the belief that there has to be a balance between technology and human values in order for usability to grow. Therefore the theoretical, rather philosophical part of this thesis containing the metaphors of technology and information ecology is believed to be essential, even though their impact may not be immediately obvious. Nevertheless, they are vital. Instead of creating a high-tech, sometimes nerdy design, these theories urge us to look at the main issue: how technology can support human activity, how to overcome complexity if necessary.

As much as it is desirable to bring out the full potential of today's technology, it is essential to respect the product-specific context of each product. There is certainly a need for an understanding of both sides of the man-machine scale in order to carry out the task appropriately. For a designer to be a link between users, developers and other stakeholders, it is of significant importance to know what all parties represent.

In the pursuit of meeting deadlines and overcoming all kinds of obstacles, never forget why the product is being developed at all. Stop for a minute and remember the people who are going to use the system. Once again, usability is about a good support for human activity.

The philosophy of “the invisible computer” strives for simplicity in design and my motto when designing is “less is more”. My style is said to be typical Scandinavian, i.e. clean and clear design. Regarding information ecologies, this metaphor is especially useful when developing a product for a specific customer and a limit number of end-users, since it focuses on how humans use technology in a particular environment.

## **11.2. “INTERACTION”**

The theories on communication and learning were indeed helpful throughout all the projects as they focus on the interplay between humans and the human computer interaction. Three of the previously described projects that were conducted for this thesis involved a good deal of communication with stakeholders, whereas one project rather focused on the interaction between human and computer. Furthermore, most systems require training at some stage, be it informal or formal. Hence, it is always essential to have a certain understanding of learning to minimise the effort required for a user to be able to use a system.

## **11.3. “CREATIVITY”**

Usability Engineering involves a great deal of tasks and techniques, some of which are not feasible in all projects. However, requirements analysis, conceptual model design and detailed screen design should more or less be carried out at all times.

### *11.3.1. Requirements Analysis*

A requirements analysis should be carried out at all times regardless of the purpose of the project (a new product, redesign, evaluation etc) and the phase of the system's life cycle. GUI Design is best performed when understanding the product-specific context.

### *11.3.2. Conceptual Model Design*

The recommendation is that conceptual model design always should be created to a greater or lesser extent to prevent to rush into design without thinking. In fact, design means “planning” [27]. This kind of high-level design can be visualised on paper mock-ups. In case of an extensive functionality, it might be necessary to make a task analysis and an information architecture. Conceptual model design is crucial to get right since it affects the user's mental model of how to use the system.

### *11.3.3. Detailed Screen Design*

Visualise the conceptual model design on the screen and establish a detailed low-level design. When done properly, the look and grouping of elements will support the mental model and provide visual cues of how to do.

#### *11.3.4. Usability Testing*

Unfortunately there are not always resources to conduct usability testing, even though it can be conducted in a variety of ways permitting the designer to choose an appropriate form. In contrast to formal usability labs, usability tests can also be very informal and quick to conduct such as brainstorming sessions with stakeholders held to review a mock-up. Iterative testing on a prototype and on the finished product is the ideal case.

#### *11.3.5. Heuristic Evaluation*

Heuristic evaluation can be performed to review an existing design and in particular when usability testing with potential end-users cannot be carried out. It is quite a powerful method for achieving a sense of the system's current level of usability. Heuristic evaluation is time-efficient and need not require a large amount of time or resources. Bear the iceberg analogy of usability in mind and validate against heuristics.

### **11.4. "GUIDANCE"**

Trade-offs and sacrifices are inevitable in GUI Design and Usability Engineering. Hence, it is handy to have a number of guiding principles, both for the purpose of creating and motivating design proposals.

#### *11.4.1. The Iceberg Analogy of Usability*

The iceberg analogy of usability is especially useful when creating the conceptual model design in order to stay on the usability track and remember that the user's mental model of the system is essential. Another way of explaining the user's mental model is by introducing the metaphor of a food store. Imagine yourself walking into a food store for the first time. You will easily get lost if the products are not logically organised, e.g. where would you expect to find juice? In Sweden, you would probably start looking in the fridge, whereas in Ireland it can be placed anywhere from the cookies section to the tea shelf. Obviously, *cultural differences* play a significant role in forming users' and customers' expectations.

Category signs will definitely help you get around if it is not obvious enough. In case of a big food store, a clear categorisation and sign system is required to help the customers find what they are looking for. The problem though is that people categorise differently, e.g. is cacao a baking ingredient or a drink? Apparently, *terminology* matters a great deal. However, in a small shop, there is typically no need for a thorough categorisation and organisation of products since the shop is small enough to provide an overview of "everything" at one glance.

If your shopping goes smoothly, then you probably create the right mental model since you find what you are looking for without too much effort. The support for creating a good mental model is a clear organisation of content, which refers to information architecture in terms of GUI design. Note however that the food store metaphor fails to explain that an information architecture can describe actions and tasks and not only information structure.

The bottom line is: Do what you need! Do only produce those design documents that may be helpful for you. If the functionality provided by the system is extended, it might be useful to

make an information architecture and task analysis to reduce the complexity of the design and to understand what steps a user would take to carry out the task.

On the other hand, if the functionality is less extensive, it might not be necessary to conduct a thorough task analysis or even an information architecture. However, most enterprise software projects have extended functionality that requires an accurate analysis.

#### *11.4.2. GUI Heuristics*

There exist hundreds of guidelines for GUI Design and it is impossible to keep them all in mind. That is why the usability toolbox proposes GUI Heuristics, which are general principles that catch the most essential guidelines. The heuristics presented in section 6.2. were the ones that were used over and over again.

In projects where user-centered techniques such as usability testing and brainstorming sessions cannot be performed, GUI Heuristics in combination with the Iceberg analogy of usability can be helpful to a great extent.

### **11.5. TESTIMONIALS**

A crucial aspect of introducing the usability toolbox is the feedback from those who were affected by it, i.e. end-users, customers and colleagues in the development team. The following is a summary of comments on the usability toolbox of co-operators.

#### *11.5.1. End-users*

Future end-users were consulted to the greatest possible extent, in particular for the purpose of requirements gathering and usability testing. A general remark was that users were enthusiastic and active in proposing additional features. Brainstorming sessions were appreciated since they allowed users to express their opinion and feel involved in the product development. Concerning taking part in usability testing, users claimed enjoying it and being positively surprised that such scientific approaches existed.

#### *11.5.2. Customers*

End-users and customers are not always the same individuals, and this section refers to how the customers of software products reacted upon the usability toolbox, i.e. feedback from management and marketing staff. Below is the response from a customer on whose web site a heuristic evaluation was performed, and it shows that management staff appreciated the critics from a usability point of view:

- *“Overall, an extremely well researched paper. Comments and suggestions are valid from a web designer’s point of view - but more importantly they have the end-user in mind”.*
- *“The presentation of ideas is very good. The time taken to arrange screen shots has paid off. It is easy to read and assimilate information”.*
- *“Good business sense is evident”.*

- *“The time taken to suggest a new layout is good. The researcher has made many suggestions about the site and how she thinks it could work better and then she presents a suggested layout - nice touch. This makes her critique constructive. The researcher makes points related to both added value and the end-user”.*

The above testimonial emphasises the importance of designing both from an end-user's and a developer's point of view. Hence, the balance between human values and technology in system development. In summary, customers were greatly in favour of adding usability engineering to their product development. Customers generally requested usability evaluation and suggestions of enhancements. The values that the usability toolbox represents, spread via lectures and more informal meetings, challenged the customers to think in a brand new way.

### *11.5.3. Colleagues*

My colleagues were with no exception eager to achieve additional knowledge in the area of usability. They currently lacked in a formal process of adding usability to their system development, but followed their common sense and experience. The suggested approach of designing in two levels (conceptual model design and detailed screen design) opened a new world and were highly appreciated. Nevertheless, it takes time and experience to incorporate new aspects into an established work procedure.

## **12. CONCLUSION**

The purpose of this thesis was to define a usability toolbox and apply it in a number of software projects. The starting point was to define a set of tools, i.e. the usability toolbox, which could be used as a complement to the traditional system development. The idea was that the tools could be employed whenever suitable allowing GUI design work to be structured yet flexible.

In order to evaluate the usability toolbox, four system development projects of varying nature were conducted both in Sweden and in Ireland. Two of the projects aimed to design a new product whereas the purpose of the two others was to evaluate and redesign an existing user interface. Three of the projects brought potential end-users onto the development team and one project was conducted in a total absence of users. Various platforms were used. Consequently, the four projects provided an understanding of a variety of circumstances and constraints.

The conclusion is that the usability toolbox offers great support for GUI design activities in any phase of the system's life cycle. There is no cookbook approach, but this thesis puts forward a number of raw materials for the designer to make a fabulous cake.

It cannot be proven that usability was enhanced or ensured, since usability cannot be measured in all cases. Usability testing is powerful for qualitative evaluation, whereas quantitative testing may be useful for marketing and selling arguments. Another side of usability testing is that users are not always available. However, in lack of users, you can go a long way with heuristics solely.

Designing well is not easy. It takes a good deal of creativity and a structured approach to GUI design, insight in the product-specific context, understanding of human capacity and memory, trade-offs between conflicting aspects, knowledge in known design principles, but also the courage to break the rules when necessary. Most importantly, to become a good GUI designer, it takes plenty of experience.

### **12.1. DISCUSSION**

The usability toolbox is meant to be a means of improving system usability, which however has not been proven in this thesis. The investigation showed that the usability toolbox was applicable in the projects conducted, but no valid evidence was put forward that the level of usability actually was enhanced. One might argue that proofs of increased usability are a motivation for fully embracing the usability toolbox. However, for this to take effect, accurate usability testing would be necessary, which was not totally feasible within the scope of this thesis. One should also reflect on the reliability of usability testing, even when properly carried out. Is it possible to measure usability at all?

Furthermore, it can be stated that the usability toolbox applied to the four specific projects presented in this thesis, but that is no guarantee for a successful integration in all projects. Special attention should be paid to the attitude, routines and experience of the development team in question. For the purpose of maximizing system usability through for instance the usability toolbox, the developers and management involved ought to be "usability advocates". But the question is if the usability toolbox itself is useable for designers? Additionally, a

usability toolbox is a highly personal matter and reflects one's total amount of knowledge and experience within usability. It is not stable but will change over time. Ideally, each one of us will sooner or later have our own usability toolbox.

## **12.2. FUTURE RESEARCH**

- To which extent can UML support GUI Design?
- How can usability engineering interact with requirements engineering, two disciplines highly related to one another?
- Who is the user interface designer?
- Art versus Engineering in usability engineering?

## 13. REFERENCES

### 13.1. LITERATURE

- [1] Biow, L. *How To Use Your Computer*. Ziff Davis Pr, 1996.
- [2] Bødker, S. *Through the interface*. Lawrence Erlbaum Associates, 1991.
- [3] Draper, S. & Norman D. *User Centered System Design*. Lawrence Erlbaum Associates, 1986.
- [4] Hackos, J.T. & Redish, J.C. *User and Task Analysis for Interface Design*. John Wiley & Sons, 1998.
- [5] Illeris, K. *Lärandet i mötet mellan Piaget, Freud och Marx*. Studentlitteratur, 2001.
- [6] Lamming, M. & Newman, W. *Interactive system design*. Addison-Wesley, 1995.
- [7] Mayhew, D.J. *The Usability Engineering Lifecycle*. Morgan Kaufmann Publishers, 1999.
- [8] Nardi, B. & O'Day, V. *Information Ecologies – Using Technology with Heart*. MIT Press, 2000.
- [9] Nielsen, J. *Usability Engineering*. Morgan Kaufmann Publishers, 1994.
- [10] Nilsson, B. & Waldemarsson, A-K. *Kommunikation – Samspel mellan människor*. Studentlitteratur, 1994.
- [11] Norman, D. *The Invisible Computer*. MIT Press, 1998.
- [12] Norman, D. *The Psychology of Everyday Things*. Basic Books, 1988.
- [13] Preece, J. *Human-Computer Interaction*. Addison-Wesley, 1994.
- [14] Raskin, J. *The Humane Interface*. Addison-Wesley, 2000.
- [15] Rubin, J. *Handbook of usability testing*. John Wiley & Sons, 1994.
- [16] Sommerville, I. *Software Engineering*. Addison-Wesley Pub Co, 2000.
- [17] Winograd, T. *Bringing Design to Software*. Addison-Wesley, 1996.

### 13.2. ARTICLES

- [18] Larsson, R. *Få företag vet nyttan med sina webbplatser*, Dagens Industri: February 14 2002.
- [19] Norman, D. *Emotion and design: Attractive things work better*. Interactions Magazine, ix (4), 36-42: July 2002.

### 13.3. WEB SITES

- [20] Berry, D. *IBM's Ease of Use: The Iceberg Analogy of Usability*.  
[http://www-106.ibm.com/developerworks/library/w\\_berry/](http://www-106.ibm.com/developerworks/library/w_berry/), 2001-12-07.
- [21] Cambridge Dictionaries Online  
<http://dictionary.cambridge.org/>, 2003-01-27
- [22] Interactive Institute: *mänsklig teknik eller teknisk mänsklighet?*  
[http://smart.interactiveinstitute.se/smart/smart\\_sv/mansklig.html](http://smart.interactiveinstitute.se/smart/smart_sv/mansklig.html), 2002-03-01.
- [23] Jakob Nielsen's Alertbox: *Usability Metrics*.  
<http://www.useit.com>, 2003-01-06.

- [24] The LUCID (Logical User Centred Interaction Design) Design Framework  
<http://www.cognetics.com>, 2003-01-06.
- [25] The Argus Center for Information Architecture  
<http://www.argus-acia.com>, 2003-01-07.

#### **13.4. OTHER**

- [26] Sjöberg, H. & Ukus, D. *A GUI Standard's Impact on Usability – Bachelor Thesis in Computer Science*. Blekinge Institute of Technology, 2001.
- [27] Utbildningsradion, *Våra rum*. Sveriges Television, broadcast in November 2001 in SVT1.